entropy

Approximate Bayesian Inference

Edited by

Pierre Alquier

Printed Edition of the Special Issue Published in *Entropy*

www.mdpi.com/journal/entropy

MDPI

Approximate Bayesian Inference

Approximate Bayesian Inference

Editor

Pierre Alquier

MDPI • Basel • Beijing • Wuhan • Barcelona • Belgrade • Manchester • Tokyo • Cluj • Tianjin

Editor
Pierre Alquier
RIKEN Center for Advanced
Intelligence Project (AIP)
Japan

Editorial Office
MDPI
St. Alban-Anlage 66
4052 Basel, Switzerland

This is a reprint of articles from the Special Issue published online in the open access journal *Entropy* (ISSN 1099-4300) (available at: https://www.mdpi.com/journal/entropy/special_issues/approx_Bayes_inference).

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

LastName, A.A.; LastName, B.B.; LastName, C.C. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-0365-3789-4 (Hbk)
ISBN 978-3-0365-3790-0 (PDF)

Cover image courtesy of Pierre Alquier

Contents

About the Editor

Pierre Alquier obtained his PhD in 2006 from Université Pierre et Marie Curie in Paris (today part of Sorbonne Universités). He worked as a research and teaching assistant at Université Paris Dauphine (2006–2007), as a Maître de Conférences (lecturer) at Université Paris Diderot (2007–2012), as a lecturer at UCD Dublin (2012–2014), and then as a professor of statistics at ENSAE Paris (2014–2019). There, he taught Bayesian statistics, stochastic processes, introductory machine learning, estimator aggregation, and online learning. He joined the RIKEN new center for Advanced Intelligence Projects (AIP) in Tokyo in 2019 as a research scientist on the Approximate Bayesian Inference team. His research interests are high-dimensional statistics; Bayesian inference and machine learning; variational inference; and PAC–Bayes bounds. He has served as an Area Chair for conferences in Machine Learning such as NeurIPS and AISTATS. He is currently a member of the Topical Advisory Panel of Entropy and an action editor for the Journal of Machine Learning Research as well as for Transactions in Machine Learning Research.

Editorial

Approximate Bayesian Inference

Pierre Alquier

Center for Advanced Intelligence Project (AIP), RIKEN, Tokyo 103-0027, Japan; pierrealain.alquier@riken.jp

Received: 28 October 2020; Accepted: 6 November 2020; Published: 10 November 2020

Abstract: This is the Editorial article summarizing the scope of the Special Issue: Approximate Bayesian Inference.

Keywords: Bayesian statistics; machine learning; variational approximations; PAC-Bayes; expectation-propagation; Markov chain Monte Carlo; Langevin Monte Carlo; sequential Monte Carlo; Laplace approximations; approximate Bayesian computation; Gibbs posterior

1. Introduction

Extremely popular for statistical inference, Bayesian methods are gaining importance in machine learning and artificial intelligence problems. Indeed, in many applications, it is important for any device not only to predict well, but also to provide a quantification of the uncertainty of the prediction.

The main problem when one is to apply Bayesian statistics is that the computation of the estimators is expensive and sometimes not feasible. Bayesian estimators are based on the posterior distribution on parameters θ given by:

$$\pi(\theta|x) = \frac{\mathcal{L}(\theta;x)\pi(\theta)}{\int \mathcal{L}(\theta;x)\pi(\mathrm{d}\theta)} \tag{1}$$

where π is the prior, x the observations, and $\mathcal{L}(\theta;x)$ the likelihood function. For example, the computation of the posterior mean $\int \theta\pi(\mathrm{d}\theta|x)$ requires a difficult evaluation of the integrals. Thanks to the development of computational power, Bayesian estimation became feasible in the 1980s and the 1990s through Markov Chain Monte Carlo (MCMC) methods, such as the Metropolis–Hastings algorithm [1] and the Gibbs sampler [2,3]. These algorithms target the exact posterior distribution. They proved to be useful in many contexts and are still an active area of research. The performances and applicability of MCMC were improved by variants such as the Hamiltonian MCMC [4,5], adaptive MCMC [6–8], etc. We refer the reader to the review [9], the books [10–12], and Part III in [13] for detailed introductions to MCMC. The surveys [14,15] provide an overview on more recent advances. The asymptotic theory of Markov chains, ensuring the consistency of these algorithms, was covered in the monographs [16,17]. A few non-asymptotic results are also available [18].

Sequential Monte Carlo emerged in the 1990s as a way to update sequentially (that is, for each new data) samples from the posterior in hidden state models. They allow thus the computation of a Bayesian version of filters (such as the Kalman filter [19]). For this reason, they are also referred to as "particle filters". We refer the reader to [20] for the state-of-the-art of the early years and to the recent books [21,22] for pedagogical introductions and an overview of the most recent progress.

However, many modern models in statistics are simply too complex to use such methodologies. In machine learning, the volume of the data used in practice makes MCMC too slow to be used: first, each iteration of the algorithm requires accessing all the data, then the number of iterations required to reach convergence explodes when the dimension is large. In these cases, it seems that targeting the exact posterior is no longer a realistic objective. This motivated the development of many new methodologies, where the target is no longer the exact posterior, but simply a part of the information contained in it, or an approximation.

Before a short overview of these approximations techniques, let us mention two important examples where approximations were an essential ingredient in the application of Bayesian methods. In 2006, Netflix released a dataset containing movie ratings by its users and challenged the machine learning community to improve on its own predictions for movies that were not rated [23]. Many algorithms were proposed, including methods based on matrix factorization. Bayesian matrix factorization is computationally intensive. The first success at scaling Bayesian methods to the Netflix dataset was based on a mean-field variational approximation of the posterior by [24]. Such approximations will be discussed below.

In computer vision problems, the best performances are reached by deep neural networks [25]. Bayesian neural networks became a popular research direction. A new field of Bayesian deep learning has emerged that relies on approximate Bayesian inference to provide uncertainty estimates for neural networks without increasing the computation cost too much [26–29]. In particular, References [28,29] scaled these algorithms to the size of benchmark datasets such as CIFAR-10 and ImageNet.

2. Approximation in the Modelization

In many practical situations, the statistician is not interested in building a complete model describing the data, but simply in learning some aspects of it. One can think for example of a classification problem where one does not want to learn the full distribution of the data, but only a good classifier. A natural idea is to replace $\pi(\theta|x)$ in (1) by:

$$\tilde{\pi}(\theta|x) = \frac{\exp\left[-\ell(x;\theta)\right]\pi(\theta)}{\int \exp\left[-\ell(x;\theta)\right]\pi(\mathrm{d}\theta)} \tag{2}$$

where $\ell(x;\theta)$ is a Taylor loss function—for example, the classification error. When $\ell(x;\theta) = -\log\mathcal{L}(\theta;x)$, we recover (1) as a special case. When $\ell(x;\theta) = -\alpha\log\mathcal{L}(\theta;x)$ for some $\alpha \neq 1$, we obtain tempered posteriors, which appeared for various computational and theoretical reasons in the statistical literature; see [30–34], respectively. The use of the general form (2) was advocated to the statistical community by [35].

It appears that this idea was already popular in the machine learning theory community, where distributions like $\tilde{\pi}(\theta|x)$ are often referred to as Gibbs posteriors or aggregation rules. The PAC-Bayesian theory was developed to provide upper bounds on the prediction risk of such distributions [36–38]. We refer the reader to nice tutorials on PAC-Bayes bounds [39,40]. References [41–43] emphasized the connection to information theory. Note that the dropout technique used in deep learning to improve the performances of neural networks [44] was studied with PAC-Bayes bounds in [40]; see also [26]. Many publications in the past few years indeed confirmed that PAC-Bayes bounds are very well suited to analyze the performances of deep learning [45–51]. See [52] for a recent survey on PAC-Bayes bounds.

Such distributions were also well known in game theory and in prediction with expert advice since the 1990s [53,54]. We refer to the book [55], the recent work [56], and to connected problems such as bandits [57,58].

Finally, many aggregation procedures studied in high-dimensional statistics can also be written under the form of (2); see [59–64] with various regression or classification losses. References [65] used a Gibbs posterior based on the quantile loss to estimate a VaR (Value at Risk, a measure of risk in finance).

3. Approximation in the Computations

Many works have been done in the past few years to compute estimators based on $\pi(\theta|x)$ or $\tilde{\pi}(\theta|x)$ in complex problems, or with very large datasets. Very often, this is at the cost of targeting an approximation rather than the exact posterior. It is then important to analyze the accuracy of the approximation.

The nature and accuracy of these approximations are extremely different from one algorithm to the other, and some of them are not well understood theoretically. Below, we group these algorithms into three groups. In Section 3.1, we present methods that still essentially rely on simulations. In Section 3.2, we present asymptotic approximations. Finally, in Section 3.3, we present optimization based methods (this grouping is for the ease of exposition and is of course a little crude; each subsection mentions methods that have little to do with each other).

3.1. Non-Exact Monte Carlo Methods

Monte Carlo methods based on Langevin diffusions were introduced in physics in the 1970s [66]. Let $(U_t)_{t\geq 0}$ be a diffusion process given by the stochastic differential equation:

$$\mathrm{d}U_t = \nabla \log \pi(U_t|x)\mathrm{d}t + \sqrt{2}\mathrm{d}W_t,$$

where $(W_t)_{t\geq 0}$ is a standard Brownian motion. It turns out that the invariant distribution of (U_t) is $\pi(\cdot|x)$. A discretization scheme with step $h > 0$ leads to the Markov chain $\tilde{U}_{n+1} = \tilde{U}_n + h\nabla \log \pi(U_n|x) + \sqrt{2h}\xi_n$, where the (ξ_n) are i.i.d standard Gaussian variables. However, it is important to note that (U_n) does not admit $\pi(\cdot|x)$ as an invariant distribution. Thus, the Langevin Monte Carlo method is not exact (it would become exact with $h \to 0$). Reference [67] proposed a correction of this method based on the Metropolis–Hastings algorithm, which leads to an exact algorithm, known as the MALA (the Monte Carlo Adjusted Langevin Algorithm). The Langevin Monte Carlo and MALA became popular in statistics and machine learning following [68]. This paper studies the asymptotic properties of both algorithms. Surprisingly, the exact method does not necessarily enjoy the best asymptotic guarantees. More recently, in the case where $\log \pi(U_n|x)$ is concave, non-asymptotic guarantees where proven for Langevin Monte Carlo with a running time that depends only polynomially on the dimension of the parameter θ; see [69–74]. Such results are usually not available for exact MCMC methods.

The implementation of the classical Metropolis–Hastings algorithm requires being able to compute the ratio $\mathcal{L}(\theta;x)/\mathcal{L}(\theta'|x)$ for any θ, θ'. In some models with complex likelihoods, or with intractable normalization constants, this is not possible. This led to a new direction, that is approximations of this likelihood ratio. A surprising and beautiful fact is that, if each likelihood is computed by an unbiased Monte Carlo estimator, the algorithm remains exact: this was studied under the name pseudo-marginal MCMC in [75]. Still, it sometimes requires much work to get unbiased estimates [76,77], when possible at all. Some authors proposed more general approximations of the likelihood ratio, leading to non-exact algorithms. References [78–81] proposed estimators based on subsampling when the data x are too large. Reference [82] proposed an estimator of the likelihood ratio when the likelihood has intractable constants, as in the exponential random graph model, and proved that, even if the resulting MCMC is inexact, it remains asymptotically close to the exact chain. A further theory was developed in [83–85]. More on MCMC for big data can be found in [86].

Finally, the ABC (Approximate Bayesian Computation) algorithm was proposed in population genetics for models where the likelihood is far too complex to be computed, but where it is relatively easy to sample from it [87,88]. It became extremely popular in some applications; we refer the reader to the survey [89], to Section 3 in [15], and more recently, to the book [90]. Some theoretical results were proven in [91]; we also refer the reader to [92–94] for some recent advances.

3.2. Asymptotic Approximations

Laplace's method provides a Gaussian approximation of the posterior centered on the Maximum Likelihood Estimator (MLE) and whose covariance matrix is the inverse of the Fisher information. This approximation can be theoretically justified in parametric models under appropriate regularity conditions thanks to the Bernstein–von Mises theorem. We refer the reader to Chapter 13 in [95] for a complete statement of this result. Integrated Nested Laplace Approximations (INLA) indeed became very popular in Gaussian latent models to compute approximations of the posterior marginals [96].

The extension of the Bernstein–von Mises theorem to nonparametric or semiparametric models is a quite technical and important research direction; see for example [97–101] and Chapter 10 in the monograph [102]. It is important to keep in mind that even in parametric models, when the assumptions of the theorem are not met, Laplace approximation can be wrong. The asymptotic of the posterior in such models was studied in detail in [103].

3.3. Approximations via Optimization

A huge number of methods are based on the idea of using optimization algorithms to find the best approximation of $\pi(\cdot|x)$, or $\tilde{\pi}(\cdot|x)$, in a set of probability distributions $\mathcal{Q}$ fixed by the statistician. The difference between the various methods is in the choice of the criterion used to define the "best" approximation. The set $\mathcal{Q}$ can be parametric (e.g., Gaussian distributions, inspired by Laplace's method) or not, the choice being prescribed by the feasibility of the optimization problem.

Variational approximations are based on the Kullback–Leibler divergence KL:

$$\hat{\pi}(\theta|x) = \operatorname*{argmin}_{q\in\mathcal{Q}} KL(q||\pi(\cdot|x)) \tag{3}$$

$$= \operatorname*{argmin}_{q\in\mathcal{Q}} \left\{\mathbb{E}_{\theta\sim q}[-\log \mathcal{L}(\theta; x)] + KL(q||\pi)\right\}, \tag{4}$$

where we remind that $KL(q||p) = \int \log(\mathrm{d}q/\mathrm{d}p)\mathrm{dp}$ when q is absolutely continuous with respect to p, and $KL(q||p) = +\infty$ otherwise. We refer the reader to the seminal papers [104,105], to the tutorial [106], and to the recent review of the huge literature on variational approximations [107]. Note that the approximation used in [108] in the early days of neural networks can also be interpreted as a variational approximation. Besides the aforementioned applications to recommender systems and to deep learning, variational inference was successfully used in network data analysis [109], economics and econometrics [110–113], finance [114], natural language processing [115], and video processing [116], among others. A huge range of optimization algorithm were used, from the coordinate-wise optimization in the original publications to message passing [117], the gradient and stochastic gradient algorithm [27,115,118], and the natural gradient [119]. The convexity and smoothness of the minimization problem were discussed in [120]. The scope of these methods was extended to models with intractable likelihood in [121]. Reference [122] pointed out a connection between (4) and PAC-Bayes bounds, which led to the first generalization error bounds for variational inference for some Gibbs posteriors, as in (2). The analysis was extended to various settings, including regular posteriors, as in (1), by [123–131]. In particular, Reference [132] proved that variational inference leads to the optimal estimation of some classes of functions with deep learning. Note that even when $\mathcal{Q}$ is the set of all Gaussian distributions on the parameter space, the approximation can be very different from the Laplace approximation. Indeed, Reference [129] contains an example of a mixture model where the MLE is not consistent, but Gaussian variational inference is.

The choice of the Kullback–Leibler divergence in (3) and (4) was initially motivated by the tractability of the computational program to which it leads. Recently, many authors questioned that choice and proposed extended definitions of variational inference using other divergences; for a presentation of the most popular divergences in statistics, see the introduction to information geometry [133]. Note that if we replace KL by another divergence, (3) and (4) are in general no longer equivalent, which leads to two possible ways to extend the definition. Reference [134] extended (3) by replacing the KL term by a Rényi divergence, and Reference [135] used the χ^2 divergence. However, Reference [136] discussed the computational difficulties induced by these changes, which might outweigh the benefits. Reference [137] discussed other criteria, including the Wasserstein distance, and provided some theoretical guarantees. On the other hand, References [138–141] proposed to use more general divergences in (3). This can be related to the generalized exponential family of [142] and the PAC-Bayes bounds in [143,144].

The very popular Expectation Propagation algorithm (EP) was introduced by [145]. EP can be interpreted as the minimization of the reverse KL, $KL(\pi(\cdot|x)||q))$, instead of (3). This was detailed in [146], where the author also proposed an extension with α-divergences called power EP. Algorithmic issues were discussed in [147] and by [148], who proposed stochastic optimization methods. A first theoretical analysis of EP was proposed in [149]. Let us mention that the textbook [150], which is a generalist introduction to machine learning, contains a full chapter entirely devoted to a pedagogical introduction to variational approximations and EP. The paper [151] focuses on the application of EP to hierarchical models, but also contains a very nice introduction to EP and the conditions ensuring its stability.

Finally, let us mention approximations by discrete distributions, of the form $q = \frac{1}{M}\sum_{i=1}^{M}\delta_{\theta_i}$ where δ_x is the Dirac mass at x. Note that this is typically the kind of approximation provided by the MCMC and sequential Monte Carlo methods, but in these methods, the θ_i are sampled. It is also possible to try to minimize a distance criterion between q and $\pi(\cdot|x)$. Unfortunately, when $\pi(\cdot|x)$ is continuous, both $KL(\pi(\cdot|x)||q)) = KL(q||\pi(\cdot|x))) = +\infty$, so it is not possible to use variational inference or EP in this case. An energy based criterion was proposed in [152]. Reference [153] proposed to use Stein divergences between q and $\pi(\cdot|x)$, and the technique became quite successful [154–156]. Another possible research direction is to use the Wasserstein distance [157].

4. Scope of This Special Issue

The objective of this Special Issue is to provide the latest advances in approximate Monte Carlo methods and in approximations of the posterior: the design of efficient algorithms, the study of the statistical properties of these algorithms, and challenging applications.

Funding: This research received no external funding.

Acknowledgments: The author gratefully thanks Emtiyaz Khan (RIKEN AIP) and Nicolas Chopin (ENSAE Paris) for useful comments.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ABC	Approximate Bayesian Computation
EP	Expectation Propagation
MALA	Monte Carlo Adjusted Langevin Algorithm
MCMC	Markov Chain Monte Carlo
MLE	Maximum Likelihood Estimator
PAC	Probably Approximately Correct
VaR	Value at Risk

References

1. Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller, A.H.; Teller, E. Equation of state calculations by fast computing machines. *J. Chem. Phys.* **1953**, *21*, 1087–1092. [CrossRef]
2. Geman, S.; Geman, D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.* **1984**, *6*, 721–741. [CrossRef] [PubMed]
3. Casella, G.; George, E.I. Explaining the Gibbs sampler. *Am. Stat.* **1992**, *46*, 167–174.
4. Duane, S.; Kennedy, A.D.; Pendleton, B.J.; Roweth, D. Hybrid Monte Carlo. *Phys. Lett. B* **1987**, *195*, 216–222. [CrossRef]
5. Neal, R. *Bayesian Learning for Neural Networks*; Springer Lecture Notes in Statistics; Springer: Berlin/Heidelberg, Germany, 1999; Volume 118.

6. Gilks, W.R.; Roberts, G.O.; Sahu, S.K. Adaptive Markov chain monte carlo through regeneration. *J. Am. Stat. Assoc.* **1998**, *93*, 1045–1054. [CrossRef]
7. Atchade, Y.; Fort, G.; Moulines, E.; Priouret, P. Adaptive Markov chain Monte Carlo: Theory and methods. In *Bayesian Time Series Models*; Cambridge University Press: Cambridge, UK, 2011; pp. 32–51.
8. Roberts, G.O.; Rosenthal, J.S. Examples of adaptive MCMC. *J. Comput. Graph. Stat.* **2009**, *18*, 349–367. [CrossRef]
9. Besag, J.; Green, P.; Higdon, D.; Mengersen, K. Bayesian Computation and Stochastic Systems. *Stat. Sci.* **1995**, *10*, 3–41. [CrossRef]
10. Andrieu, C.; De Freitas, N.; Doucet, A.; Jordan, M.I. An introduction to MCMC for machine learning. *Mach. Learn.* **2003**, *50*, 5–43. [CrossRef]
11. Brooks, S.; Gelman, A.; Jones, G.; Meng, X.L. (Eds.) *Handbook of Markov Chain Monte Carlo*; CRC Press: Boca Raton, FL, USA, 2011.
12. Robert, C.; Casella, G. *Monte Carlo Statistical Methods*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013.
13. Gelman, A.; Carlin, J.B.; Stern, H.S.; Dunson, D.B.; Vehtari, A.; Rubin, D.B. *Bayesian Data Analysis*, 3rd ed.; CRC Press: Boca Raton, FL, USA, 2013.
14. Chopin, N.; Gadat, S.; Guedj, B.; Guyader, A.; Vernet, E. On some recent advances on high dimensional Bayesian statistics. *ESAIM Proc. Surv.* **2015**, *51*, 293–319. [CrossRef]
15. Green, P.J.; Łatuszyński, K.; Pereyra, M.; Robert, C.P. Bayesian computation: A summary of the current state, and samples backwards and forwards. *Stat. Comput.* **2015**, *25*, 835–862. [CrossRef]
16. Meyn, S.P.; Tweedie, R.L. *Markov Chains and Stochastic Stability*; Springer: Berlin/Heidelberg, Germany, 2012.
17. Douc, R.; Moulines, E.; Priouret, P.; Soulier, P. *Markov Chains*; Springer: Berlin, Germany, 2018.
18. Joulin, A.; Ollivier, Y. Curvature, concentration and error estimates for Markov chain Monte Carlo. *Ann. Probab.* **2010**, *38*, 2418–2442. [CrossRef]
19. Kalman, R.E. A New Approach to Linear Filtering and Prediction Problems. *Trans. ASM J. Basic Eng.* **1960**, *82*, 35–45. [CrossRef]
20. Doucet, A.; De Freitas, N.; Gordon, N. (Eds.) *Sequential Monte Carlo Methods in Practice*; Springer: Berlin/Heidelberg, Germany, 2001.
21. Chopin, N.; Papaspiliopoulos, O. *An Introduction to Sequential Monte Carlo*; Springer: Berlin/Heidelberg, Germany, 2020.
22. Naesseth, C.A.; Lindsten, F.; Schön, T.B. Elements of Sequential Monte Carlo. *Found. Trends Mach. Learn.* **2019**, *12*, 307–392. [CrossRef]
23. Bennett, J.; Lanning, S. The Netflix prize. In Proceedings of the KDD Cup and Workshop, Los Gatos, CA, USA, 12 August 2005; pp. 35–38.
24. Lim, Y.J.; Teh, Y.W. Variational Bayesian approach to movie rating prediction. In Proceedings of the KDD Cup and Workshop, Jose, CA, USA, 12 August 2007; pp. 15–21.
25. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
26. Gal, Y.; Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1050–1059.
27. Mandt, S.; Hoffman, M.D.; Blei, D.M. Stochastic gradient descent as approximate Bayesian inference. *J. Mach. Learn. Res.* **2017**, *18*, 1–35.
28. Maddox, W.J.; Izmailov, P.; Garipov, T.; Vetrov, D.P.; Wilson, A.G. A simple baseline for Bayesian uncertainty in deep learning. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2019; pp. 13153–13164.
29. Osawa, K.; Swaroop, S.; Khan, M.E.; Jain, A.; Eschenhagen, R.; Turner, R.E.; Yokota, R. Practical deep learning with Bayesian principles. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 4287–4299.
30. Neal, R.M. Sampling from multimodal distributions using tempered transitions. *Stat. Comput.* **1996**, *6*, 353–366. [CrossRef]
31. Friel, N.; Pettitt, A.N. Marginal likelihood estimation via power posteriors. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2008**, *70*, 589–607. [CrossRef]

32. Walker, S.; Hjort, N.L. On Bayesian consistency. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2001**, *63*, 811–821. [CrossRef]
33. Grünwald, P.D.; Van Ommen, T. Inconsistency of Bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Anal.* **2017**, *12*, 1069–1103. [CrossRef]
34. Bhattacharya, A.; Pati, D.; Yang, Y. Bayesian fractional posteriors. *Ann. Stat.* **2019**, *47*, 39–66. [CrossRef]
35. Bissiri, P.G.; Holmes, C.C.; Walker, S.G. A general framework for updating belief distributions. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2016**, *78*, 1103–1130. [CrossRef] [PubMed]
36. Shawe-Taylor, J.; Williamson, R.C. A PAC analysis of a Bayesian estimator. In Proceedings of the Tenth Annual Conference on Computational Learning Theory, Nashville, TN, USA, 6–9 July 1997; pp. 2–9.
37. McAllester, D.A. Some PAC-Bayesian theorems. *Mach. Learn.* **1999**, *37*, 355–363. [CrossRef]
38. Catoni, O. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*; Monograph Series 56; IMS Lecture Notes: Beachwood, OH, USA, 2007.
39. Van Erven, T. PAC-Bayes mini-tutorial: A continuous union bound. *arXiv* **2014**, arXiv:1405.1580.
40. McAllester, D.A. A PAC-Bayesian tutorial with a dropout bound. *arXiv* **2013**, arXiv:1307.2118.
41. Catoni, O. *Statistical Learning Theory and Stochastic Optimization: Ecole d'Eté de Probabilités de Saint-Flour XXXI-2001*; Springer: Berlin/Heidelberg, Germany, 2004.
42. Zhang, T. From ϵ-entropy to KL-entropy: Analysis of minimum information complexity density estimation. *Ann. Stat.* **2006**, *34*, 2180–2210. [CrossRef]
43. Grünwald, P.D.; Mehta, N.A. A tight excess risk bound via a unified PAC-Bayesian–Rademacher–Shtarkov–MDL complexity. *Conf. Algorithmic Learn.* **2019**, *98*, 433–465.
44. Deng, L.; Hinton, G.; Kingsbury, B. New types of deep neural network learning for speech recognition and related applications: An overview. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 8599–8603.
45. Neyshabur, B.; Bhojanapalli, S.; McAllester, D.; Srebro, N. Exploring generalization in deep learning. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5947–5956.
46. Dziugaite, G.K.; Roy, D. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv* **2017**, arXiv:1703.11008.
47. Dziugaite, G.K.; Roy, D. Entropy-SGD optimizes the prior of a PAC-Bayes bound: Generalization properties of Entropy-SGD and data-dependent priors. In Proceedings of the 35th International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 1377–1386.
48. Amit, R.; Meir, R. Meta-learning by adjusting priors based on extended PAC-Bayes theory. In Proceedings of the 35th International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 205–214.
49. Nozawa, K.; Sato, I. PAC-Bayes Analysis of Sentence Representation. *arXiv* **2019**, arXiv:1902.04247.
50. Pitas, K. Better PAC-Bayes bounds for deep neural networks using the loss curvature. *arXiv* **2019**, arXiv:1909.03009.
51. Rivasplata, O.; Tankasali, V.M.; Szepesvari, C. PAC-Bayes with backprop. *arXiv* **2019**, arXiv:1908.07380 .
52. Guedj, B. A primer on PAC-Bayesian learning. In Proceedings of the Second Congress of the French Mathematical Society, Lille, France, 4–8 June 2018.
53. Vovk, V.G. Aggregating strategies. In Proceedings of the Third Annual Workshop on Computational Learning Theory, Rochester, NY, USA, 6–8 August 1990.
54. Littlestone, N.; Warmuth, M.K. The weighted majority algorithm. *Inf. Comput.* **1994**, *108*, 212–261. [CrossRef]
55. Cesa-Bianchi, N.; Lugosi, G. *Prediction, Learning, and Games*; Cambridge University Press: Cambridge, UK, 2006.
56. Besson, R.; Le Pennec, E.; Allassonnière, S. Learning from both experts and data. *Entropy* **2019**, *21*, 1208. [CrossRef]
57. Seldin, Y.; Auer, P.; Shawe-Taylor, J.S.; Ortner, R.; Laviolette, F. PAC-Bayesian analysis of contextual bandits. In Proceedings of the Advances in Neural Information Processing Systems, Granada, Spain, 12–14 December 2011; pp. 1683–1691.
58. Bubeck, S.; Cesa-Bianchi, N. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Found. Trends Mach. Learn.* **2012**, *5*, 1–122. [CrossRef]

59. Leung, G.; Barron, A.R. Information theory and mixing least-squares regressions. *IEEE Trans. Inf. Theory* **2006**, *52*, 3396–3410. [CrossRef]
60. Jiang, W.; Tanner, M.A. Gibbs posterior for variable selection in high-dimensional classification and data mining. *Ann. Stat.* **2008**, *36*, 2207–2231. [CrossRef]
61. Dalalyan, A.S.; Tsybakov, A.B. Sparse regression learning by aggregation and Langevin Monte-Carlo. *J. Comput. Syst. Sci.* **2012**, *78*, 1423–1443. [CrossRef]
62. Suzuki, T. PAC-Bayesian bound for Gaussian process regression and multiple kernel additive model. In Proceedings of the 25th Annual Conference on Learning Theory, Edinburgh, Scotland, 25–27 June 2012; pp. 8.1–8.20.
63. Dalalyan, A.S.; Salmon, J. Sharp oracle inequalities for aggregation of affine estimators. *Ann. Stat.* **2012**, *40*, 2327–2355. [CrossRef]
64. Dalalyan, A.S.; Grappin, E.; Paris, Q. On the exponentially weighted aggregate with the Laplace prior. *Ann. Stat.* **2018**, *46*, 2452–2478. [CrossRef]
65. Syring, N.; Hong, L.; Martin, R. Gibbs posterior inference on Value-At-Risk. *Scand. Actuar. J.* **2019**, *7*, 548–557. [CrossRef]
66. Ermak, D.L. A computer simulation of charged particles in solution. I. Technique and equilibrium properties. *J. Chem. Phys.* **1975**, *62*, 4189–4196. [CrossRef]
67. Rossky, P.J.; Doll, J.D.; Friedman, H.L. Brownian dynamics as smart Monte Carlo simulation. *J. Chem. Phys.* **1978**, *69*, 4628–4633. [CrossRef]
68. Roberts, G.O.; Tweedie, R.L. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli* **1996**, *2*, 341–363. [CrossRef]
69. Dalalyan, A.S. Further and stronger analogy between sampling and optimization: Langevin Monte Carlo and gradient descent. In Proceedings of the 2017 Conference on Learning Theory, PMLR, Amsterdam, The Netherlands, 7–10 July 2017; pp. 678–689.
70. Raginsky, M.; Rakhlin, A.; Telgarsky, M. Non-convex learning via Stochastic Gradient Langevin Dynamics: A nonasymptotic analysis. In Proceedings of the 2017 Conference on Learning Theory, PMLR, Amsterdam, The Netherlands, 7–10 July 2017; pp. 1674–1703.
71. Cheng, X.; Chatterji, N.S.; Bartlett, P.L.; Jordan, M.I. Underdamped Langevin MCMC: A non-asymptotic analysis. In Proceedings of the 31st Conference on Learning Theory, PMLR, Stockholm, Sweden, 6–9 July 2018; pp. 300–323.
72. Dalalyan, A.S.; Riou-Durand, L.; Karagulyan, A. Bounding the error of discretized Langevin algorithms for non-strongly log-concave targets. *arXiv* **2019**, arXiv:1906.08530.
73. Durmus, A.; Moulines, E. High-dimensional Bayesian inference via the unadjusted Langevin algorithm. *Bernoulli* **2019**, *25*, 2854–2882. [CrossRef]
74. Mou, W.; Flammarion, N.; Wainwright, M.J.; Bartlett, P.L. Improved bounds for discretization of Langevin diffusions: Near-optimal rates without convexity. *arXiv* **2019**, arXiv:1907.11331.
75. Andrieu, C.; Roberts, G.O. The pseudo-marginal approach for efficient Monte Carlo computations. *Ann. Stat.* **2009**, *37*, 697–725. [CrossRef]
76. Lyne, A.M.; Girolami, M.; Atchadé, Y.; Strathmann, H.; Simpson, D. On Russian roulette estimates for Bayesian inference with doubly-intractable likelihoods. *Stat. Sci.* **2015**, *30*, 443–467. [CrossRef]
77. Vats, D.; Gonçalves, F.; Łatuszyński, K.; Roberts, G.O. Efficient Bernoulli factory MCMC for intractable likelihoods. *arXiv* **2020**, arXiv:2004.07471.
78. Korattikara, A.; Chen, Y.; Welling, M. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In Proceedings of the International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 181–189.
79. Huggins, J.; Campbell, T.; Broderick, T. Coresets for Scalable Bayesian Logistic Regression. In Proceedings of the Advances in Neural Information Processing Systems 29, Barcelona, Spain, 5–10 December 2016; pp. 4080–4088.
80. Quiroz, M.; Kohn, R.; Villani, M.; Tran, M.N. Speeding up MCMC by efficient data subsampling. *J. Am. Stat. Assoc.* **2018**, *114*, 831–843. [CrossRef]
81. Maire, F.; Friel, N.; Alquier, P. Informed sub-sampling MCMC: Approximate Bayesian inference for large datasets. *Stat. Comput.* **2019**, *29*, 449–482. [CrossRef]

82. Alquier, P.; Friel, N.; Everitt, R.; Boland, A. Noisy Monte Carlo: Convergence of Markov chains with approximate transition kernels. *Stat. Comput.* **2016**, *26*, 29–47. [CrossRef]
83. Medina-Aguayo, F.J.; Lee, A.; Roberts, G.O. Stability of noisy metropolis–hastings. *Stat. Comput.* **2016**, *26*, 1187–1211. [CrossRef] [PubMed]
84. Rudolf, D.; Schweizer, N. Perturbation theory for Markov chains via Wasserstein distance. *Bernoulli* **2018**, *24*, 2610–2639. [CrossRef]
85. Stoehr, J.; Benson, A.; Friel, N. Noisy Hamiltonian Monte Carlo for doubly intractable distributions. *J. Comput. Graph. Stat.* **2019**, *28*, 220–232. [CrossRef]
86. Bardenet, R.; Doucet, A.; Holmes, C. On Markov chain Monte Carlo methods for tall data. *J. Mach. Learn. Res.* **2017**, *18*, 1515–1557.
87. Tavaré, S.; Balding, D.; Griffith, R.; Donnelly, P. Inferring coalescence times from DNA sequence data. *Genetics* **1997**, *145*, 505–518.
88. Beaumont, M.A.; Zhang, W.; Balding, D.J. Approximate Bayesian computation in population genetics. *Genetics* **2002**, *162*, 2025–2035.
89. Marin, J.-M.; Pudlo, P.; Robert, C.P.; Ryder, R.J. Approximate Bayesian computational methods. *Stat. Comput.* **2012**, *22*, 1167–1180. [CrossRef]
90. Sisson, S.A.; Fan, Y.; Beaumont, M. (Eds.) *Handbook of Approximate Bayesian Computation*; CRC Press: Boca Raton, FL, USA, 2018.
91. Biau, G.; Cérou, F.; Guyader, A. New insights into approximate Bayesian computation. *Ann. De L'IHP Probab. Stat.* **2015**, *51*, 376–403. [CrossRef]
92. Bernton, E.; Jacob, P.E.; Gerber, M.; Robert, C.P. Approximate Bayesian computation with the Wasserstein distance. *J. R. Stat. Soc. Ser. B* **2019**, *81*, 235–269. [CrossRef]
93. Buchholz, A.; Chopin, N. Improving approximate Bayesian computation via quasi-Monte Carlo. *J. Comput. Graph. Stat.* **2019**, *28*, 205–219. [CrossRef]
94. Nguyen, H.D.; Arbel, J.; Lü, H.; Forbes, F. Approximate Bayesian computation via the energy statistic. *IEEE Access* **2020**, *8*, 131683–131698. [CrossRef]
95. Van der Vaart, A.W. *Asymptotic Statistics*; Cambridge University Press: Cambridge, UK, 2000.
96. Rue, H.; Martino, S.; Chopin, N. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2009**, *71*, 319–392. [CrossRef]
97. Freedman, D. Wald Lecture: On the Bernstein-von Mises theorem with infinite-dimensional parameters. *Ann. Stat.* **1999**, *em 27*, 1119–1141. [CrossRef]
98. Boucheron, S.; Gassiat, E. A Bernstein-von Mises theorem for discrete probability distributions. *Electron. J. Stat.* **2009**, *3*, 114–148. [CrossRef]
99. Bickel, P.J.; Kleijn, B.J. The semiparametric Bernstein–von Mises theorem. *Ann. Stat.* **2012**, *40*, 206–237. [CrossRef]
100. Rivoirard, V.; Rousseau, J. Bernstein–von Mises theorem for linear functionals of the density. *Ann. Stat.* **2012**, *40*, 1489–1523. [CrossRef]
101. Castillo, I.; Nickl, R. On the Bernstein–von Mises phenomenon for nonparametric Bayes procedures. *Ann. Stat.* **2014**, *42*, 1941–1969. [CrossRef]
102. Ghosal, S.; Van der Vaart, A. *Fundamentals of Nonparametric Bayesian Inference*; Cambridge University Press: Cambridge, UK, 2017.
103. Watanabe, S. *Mathematical Theory of Bayesian Statistics*; CRC Press: Boca Raton, FL, USA, 2018.
104. Attias, H. Inferring parameters and structure of latent variable models byvariational Bayes. In Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, Stockholm, Sweden, 30 July–1 August 1999; pp. 21–30.
105. Jordan, M.I.; Ghahramani, Z.; Jaakkola, T.S.; Saul, L.K. An introduction to variational methods for graphical models. *Mach. Learn.* **1999**, *37*, 183–233. [CrossRef]
106. Wainwright, M.J.; Jordan, M.I. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.* **2008**, *1*, 1–305. [CrossRef]
107. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877. [CrossRef]

108. Hinton, G.E.; Van Camp, D. Keeping the neural networks simple by minimizing the description length of the weights. In Proceedings of the Sixth Annual Conference on Computational Learning Theory, Santa Cruz, CA, USA, 26–28 July 1993; pp. 5–13.
109. Salter-Townshend, M.; Murphy, T.B. Variational Bayesian inference for the latent position cluster model for network data. *Comput. Stat. Data Anal.* **2013**, *57*, 661–671. [CrossRef]
110. Braun, M.; McAuliffe, J. Variational inference for large-scale models of discrete choice. *J. Am. Stat. Assoc.* **2010**, *105*, 324–335. [CrossRef]
111. Wu, G. Fast and scalable variational Bayes estimation of spatial econometric models for Gaussian data. *Spat. Stat.* **2018**, *24*, 32–53. [CrossRef]
112. Baltagi, B.H.; Bresson, G.; Etienne, J.M. Carbon dioxide emissions and economic activities: A mean field variational Bayes semiparametric panel data model with random coefficients. *Ann. Econ. Stat.* **2019**, *134*, 43–77. [CrossRef]
113. Gefang, D.; Koop, G.; Poon, A. Computationally efficient inference in large Bayesian mixed frequency VARs. *Econ. Lett.* **2020**, *191*, 109120. [CrossRef]
114. Gunawan, D.; Kohn, R.; Nott, D. Variational Approximation of Factor Stochastic Volatility Models. *arXiv* **2020**, arXiv:2010.06738.
115. Hoffman, M.D.; Blei, D.M.; Wang, C.; Paisley, J. Stochastic variational inference. *J. Mach. Learn. Res.* **2013**, *14*, 1303–1347.
116. Li, X.; Zheng, Y. Patch-based video processing: A variational Bayesian approach. *IEEE Trans. Circuits Syst. Video Technol.* **2009**, *19*, 27–40.
117. Winn, J.; Bishop, C.M. Variational Message Passing. *J. Mach. Learn. Res.* **2005**, *6*, 661–694.
118. Broderick, T.; Boyd, N.; Wibisono, A.; Wilson, A.C.; Jordan, M.I. Streaming Variational Bayes. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 1727–1735.
119. Khan, M.E.; Lin, W. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Lauderdale, FL, USA, 20 April 2017; pp. 878–887.
120. Domke, J. Provable smoothness guarantees for black-box variational inference. *arXiv* **2019**, arXiv:1901.08431.
121. Tran, M.N.; Nott, D.J.; Kohn, R. Variational Bayes with intractable likelihood. *J. Comput. Graph. Stat.* **2017**, *26*, 873–882. [CrossRef]
122. Alquier, P.; Ridgway, J.; Chopin, N. On the properties of variational approximations of Gibbs posteriors. *J. Mach. Learn. Res.* **2016**, *17*, 8374–8414.
123. Sheth, R.; Khardon, R. Excess risk bounds for the Bayes risk using variational inference in latent Gaussian models. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020; pp. 5151–5161.
124. Cottet, V.; Alquier, P. 1-Bit matrix completion: PAC-Bayesian analysis of a variational approximation. *Mach. Learn.* **2018**, *107*, 579–603. [CrossRef]
125. Wang, Y.; Blei, D.M. Frequentist consistency of variational Bayes. *J. Am. Stat. Assoc.* **2019**, *114*, 1147–1161. [CrossRef]
126. Chérief-Abdellatif, B.-E. Consistency of ELBO maximization for model selection. In Proceedings of the 1st Symposium on Advances in Approximate Bayesian Inference, PMLR, Montreal, QC, Canada, 2 December 2018; pp. 11–31.
127. Guha, B.S.; Bhattacharya, A.; Pati, D. Statistical Guarantees and Algorithmic Convergence Issues of Variational Boosting. *arXiv* **2020**, arXiv:2010.09540.
128. Chérief-Abdellatif, B.-E.; Alquier, P.; Khan, M.E. A Generalization Bound for Online Variational Inference. *arXiv* **2019**, arXiv:1904.03920.
129. Alquier, P.; Ridgway, J. Concentration of tempered posteriors and of their variational approximations. *Ann. Stat.* **2020**, *48*, 1475–1497. [CrossRef]
130. Yang, Y.; Pati, D.; Bhattacharya, A. α-variational inference with statistical guarantees. *Ann. Stat.* **2020**, *48*, 886–905. [CrossRef]
131. Zhang, F.; Gao, C. Convergence rates of variational posterior distributions. *Ann. Stat.* **2020**, *48*, 2180–2207. [CrossRef]
132. Chérief-Abdellatif, B.E. Convergence Rates of Variational Inference in Sparse Deep Learning. *arXiv* **2019**, arXiv:1908.04847.

133. Nielsen, F. An elementary introduction to information geometry. *Entropy* **2020**, *22*, 1110. [CrossRef]
134. Li, Y.; Turner, R.E. Rényi divergence variational inference. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 1073–1081.
135. Dieng, A.B.; Tran, D.; Ranganath, R.; Paisley, J.; Blei, D. Variational inference via χ-upper bound minimization. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 2732–2741.
136. Geffner, T.; Domke, J. On the Difficulty of Unbiased Alpha Divergence Minimization. *arXiv* **2019**, arXiv:2010.09541.
137. Huggins, J.; Kasprzak, M.; Campbell, T.; Broderick, T. Validated Variational Inference via Practical Posterior Error Bounds. In Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, Sicily, Italy, 3 June 2020; pp. 1792–180.
138. Reid, M.D.; Frongillo, R.M.; Williamson, R.C.; Mehta, N. Generalized mixability via entropic duality. In Proceedings of the 28th Conference on Learning Theory, Paris, France, 3–6 July 2015; pp. 1501–1522.
139. Knoblauch, J.; Jewson, J.; Damoulas, T. Generalized variational inference: Three arguments for deriving new posteriors. *arXiv* **2019**, arXiv:1904.02063.
140. Alemi, A.A. Variational Predictive Information Bottleneck. In Proceedings of the 2nd Symposium Advances Approximate Bayesian Inference, PMLR, Vancouver, BC, Canada, 8 December 2019; pp. 1–6.
141. Alquier, P. Non-exponentially weighted aggregation: Regret bounds for unbounded loss functions. *arXiv* **2020**, arXiv:2009.03017.
142. Grunwald, P.D.; Dawid, A.P. Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *Ann. Stat.* **2004**, *32*, 1367–1433. [CrossRef]
143. Bégin, L.; Germain, P.; Laviolette, F.; Roy, J.-F. PAC-Bayesian bounds based on the Rényi divergence. In Proceedings of the 19th International Conference Artificial Intelligence and Statistics PMLR, Cadiz, Spain, 9–11 May 2016; pp. 435–444.
144. Alquier, P.; Guedj, B. Simpler PAC-Bayesian bounds for hostile data. *Mach. Learn.* **2018**, *107*, 887–902. [CrossRef]
145. Minka, T.P. Expectation propagation for approximate Bayesian inference. In Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence, Seattle, WA, USA, 2–5 August 2001; pp. 362–369.
146. Minka, T. *Divergence Measures and Message Passing*; Technical Report; Microsoft Research: Redmond, DC, USA, 2005.
147. Seeger, M.; Nickisch, H. Fast convergent algorithms for expectation propagation approximate Bayesian inference. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 652–660.
148. Li, Y.; Hernández-Lobato, J.M.; Turner, R.E. Stochastic expectation propagation. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 2323–2331.
149. Dehaene, G.P.; Barthelmé, S. Bounding errors of expectation-propagation. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 244–252.
150. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2006.
151. Vehtari, A.; Gelman, A.; Sivula, T.; Jylänki, P.; Tran, D.; Sahai, S.; Blomstedt, P.; Cunningham, J.P.; Schiminovich, D.; Robert, C.P. Expectation Propagation as a Way of Life: A Framework for Bayesian Inference on Partitioned Data. *J. Mach. Learn. Res.* **2020**, *21*, 1–53.
152. Joseph, V.R.; Dasgupta, T.; Tuo, R.; Wu, C. Sequential exploration of complex surfaces using minimum energy designs. *Technometrics* **2015**, *57*, 64–74. [CrossRef]
153. Liu, Q.; Wang, D. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 2378–2386.
154. Chen, W.Y.; Mackey, L.; Gorham, J.; Briol, F.-X.; Oates, C.J. Stein points. In Proceedings of the 35th International Conference on Machine Learningc PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 843–852.
155. Chen, W.Y.; Barp, A.; Briol, F.-X.; Gorham, J.; Girolami, M.; Mackey, L.; Oates, C. Stein Point Markov Chain Monte Carlo. In Proceedings of the 36th International Conference on Machine Learningc PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 1011–1021.

156. Kassab, R.; Simeone, O. Federated Generalized Bayesian Learning via Distributed Stein Variational Gradient Descent. *arXiv* **2020**, arXiv:2009.06419.
157. Nitanda, A.; Suzuki, T. Stochastic Particle Gradient Descent for Infinite Ensembles. *arXiv* **2017**, arXiv:1712.05438.

Article

Coupled VAE: Improved Accuracy and Robustness of a Variational Autoencoder

Shichen Cao [1], Jingjing Li [2], Kenric P. Nelson [3,*] and Mark A. Kon [2]

[1] Worcester Polytechnic Institute, Worcester, MA 01609, USA; cao.schen@gmail.com
[2] Mathematics & Statistics Department, Boston University, Boston, MA 02215, USA; jli0203@bu.edu (J.L.); mkon@bu.edu (M.A.K.)
[3] Photrek, Watertown, MA 02472, USA
* Correspondence: kenric.nelson@gmail.com

Abstract: We present a coupled variational autoencoder (VAE) method, which improves the accuracy and robustness of the model representation of handwritten numeral images. The improvement is measured in both increasing the likelihood of the reconstructed images and in reducing divergence between the posterior and a prior latent distribution. The new method weighs outlier samples with a higher penalty by generalizing the original evidence lower bound function using a coupled entropy function based on the principles of nonlinear statistical coupling. We evaluated the performance of the coupled VAE model using the Modified National Institute of Standards and Technology (MNIST) dataset and its corrupted modification C-MNIST. Histograms of the likelihood that the reconstruction matches the original image show that the coupled VAE improves the reconstruction and this improvement is more substantial when seeded with corrupted images. All five corruptions evaluated showed improvement. For instance, with the Gaussian corruption seed the accuracy improves by 10^{14} (from $10^{-57.2}$ to $10^{-42.9}$) and robustness improves by 10^{22} (from $10^{-109.2}$ to $10^{-87.0}$). Furthermore, the divergence between the posterior and prior distribution of the latent distribution is reduced. Thus, in contrast to the β-VAE design, the coupled VAE algorithm improves model representation, rather than trading off the performance of the reconstruction and latent distribution divergence.

Keywords: machine learning; entropy; robustness; statistical mechanics; complex systems

Citation: Cao, S.; Li, J.; Nelson, K.P.; Kon, M.A. Coupled VAE: Improved Accuracy and Robustness of a Variational Autoencoder. *Entropy* **2022**, *24*, 423. https://doi.org/10.3390/e24030423

Academic Editor: Pierre Alquier

Received: 15 February 2022
Accepted: 13 March 2022
Published: 18 March 2022

1. Introduction

An overarching challenge in machine learning is the development of methodologies that ensure the accuracy and robustness of models given limited training data. By accuracy, we refer to the metrics of information theory, such as minimizing the cross-entropy or divergence of an algorithm. In this paper, we define a measure of robustness based on a generalization of information theory. The variational autoencoder (VAE) contributes to improved learning of models by utilizing approximate variational inference [1,2]. By storing a statistical model rather than a deterministic model at the latent layer, the algorithm has increased flexibility in its use for reconstruction and other applications. The variational inference is optimized by minimization of a loss function, the so-called negative evidence lower bound, which has two components. The first component is a cross-entropy between the generated and the source data, also known as the expected negative log-likelihood, while the second is a divergence between the prior and the posterior distributions of the latent layer.

Our goal in this research is to provide an evaluation as to whether a generalization of information theory can be applied to improving the robustness of machine learning algorithms. Robustness of autoencoders to outliers is critical for generating a reliable representation of particular data types in the encoded space when using corrupted training data [3]. In this paper, a generalized entropy function is used to modify the negative evidence lower bound loss function of a variational autoencoder. With the MNIST handwritten numerals dataset, we are able to measure the improvement in the robustness of the reconstruction, using a metric also derived from the generalization of information theory. In addition, we find that the accuracy of the reconstruction, as measured by Shannon information theory, is also improved. Furthermore, the divergence between the latent distribution posterior and prior is also reduced. This is important to ensure that the reconstruction improvement is not a result of degrading the latent layer.

Our study builds from the work of Kingma and Welling [4] on variational autoencoders and Tran et al. [5] on deep probabilistic programming. Variational autoencoders are an unsupervised learning method for training encoder and decoder neural networks. Between the encoder and decoder, the parameters of a multidimensional distribution are learned to form a compressed latent representation of the training data [6]. It is an effective method for generating complex datasets such as images and speech. Zalger [7] implemented the application of VAE for aircraft turbomachinery design and Xu et al. [8] used VAEs to achieve unsupervised anomaly detection for seasonal key performance indicators (KPIs) in web applications. VAEs have been used to construct probabilistic models of complex physical phenomena [9]. Autoencoders can use a variety of latent variable models, but restricting the models can enhance performance. Sparse autoencoders add a penalty for the number of active hidden layer nodes used in the model. Variational autoencoders further restrict the model to a probability distribution $q_\phi(\mathbf{z}|\mathbf{x})$ specified by a set of encoder parameters ϕ which approximates the actual conditional probability $p(\mathbf{z}|\mathbf{x})$. Variational inference, as reviewed by Blei et al. [10], is used to learn this approximation by minimizing an objective function such as the Kullback–Liebler divergence. The decoder learns a set of parameters θ for a generative distribution $q_\theta(\mathbf{x}'|\mathbf{z})$, where $\mathbf{z}$ is the latent variable, and $\mathbf{x}'$ is the output generated data. The complexity of the data distribution $p(\mathbf{x})$ makes direct computation of the divergence between the approximate and exact latent conditional probabilities intractable; however, a variational or evidence lower bound (ELBO) is computable and consists of two components, the expected reconstruction log-likelihood of the generated data (cross-entropy) and the negative of the divergence between the latent posterior conditional probability $q_\phi(\mathbf{z}|\mathbf{x})$ and a latent prior distribution $p(\mathbf{z})$, which is typically a standard normal distribution but can be more sophisticated for particular model requirements.

Recently, Higgins et al. [11] proposed a β-VAE framework, which can provide a more disentangled latent representation $\mathbf{z}$ [12] by increasing the weight of the KL-divergence term of the ELBO. Since the KL-divergence is a regularization that constrains the capacity of the latent information channel $\mathbf{z}$, increasing the weight of the regularization with $\beta > 1$ puts pressure on the learnt posterior so it is more tightly packed. The effect seems to be an encouragement of each dimension to store distinct information and excess dimensions as highly packed noise. However, this improvement is a trade-off between the divergence and reconstruction components of the ELBO metric. We will show that the coupled VAE algorithm improves both components of the ELBO.

The next section provides an introduction to the design of the variational autoencoder. A comparison with other generative algorithms is included. Section 3 introduces nonlinear statistical coupling and its application to defining metrics for the robustness, accuracy, and decisiveness of decision algorithms. In this paper, use of the uppercase letter for the terms 'Robustness', 'Accuracy', and 'Decisiveness' refers to the specific metrics, which will be introduced in Section 3.1. Lowercase letters for these terms will be used when referring to the general properties. Following the definition of the reconstruction assessment metrics, the generalization of the negative ELBO is defined. This coupled negative ELBO provides control over the weighting of rare versus common samples in the distribution of the training set. Additional details of the derivation of the generalized negative ELBO function and metrics are provided in Appendices A.1 and A.2, respectively. In Section 4, the improved autoencoder is evaluated using the MNIST handwritten numeral test set. Measurements of the reconstruction and the characteristics of the posterior latent variables are analyzed. Section 5 provides a visualization of the changes in the latent distribution using a 2-dimensional distribution. Section 6 demonstrates that the coupled VAE algorithm provides significantly improved stability in the model performance when the input image is corrupted from the training set. This provides evidence of the improved robustness of the algorithm. Section 7 provides a discussion, conclusion, and suggestions for future research.

2. The Variational Autoencoder

A variational autoencoder consists of an encoder, a decoder, and a loss function. Figure 1 represents the basic structure of an autoencoder. The encoder Q is a neural network that converts high-dimensional information from the input data into a low-dimensional hidden, latent representation $\mathbf{z}$. Some information is lost during this data compression because the dimension is reduced. The decoder P decompresses from latent space $\mathbf{z}$ to reconstruct the data. While, in general, autoencoders can learn a variety of representations, VAEs especially learn the parameters of a probability distribution. The model used here learns the means and standard deviations θ of a collection of multivariate Gaussian distributions and stores this information in a two-layered space. The training loss function, which is the negative evidence lower bound, is optimized by using stochastic gradient descent.

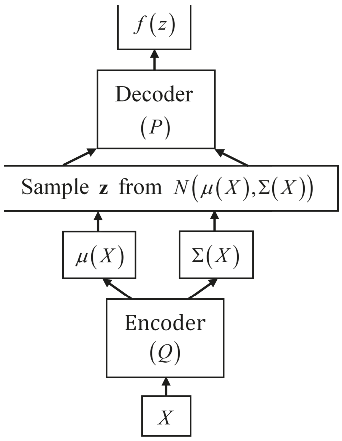


Figure 1. The variational autoencoder consists of an encoder, a probability model, and a decoder.

2.1. Vae Loss Function

The encoder reads the input data and compresses and transforms it into a fixed-shape latent representation $\mathbf{z}$, while the decoder decompresses and reconstructs the information from this latent representation, outputting specific distribution parameters to generate a new reconstruction $\mathbf{x}'$. The true posterior distribution $p(\mathbf{z}|\mathbf{x}^{(i)})$ of $\mathbf{z}$ given i^{th} datapoint $\mathbf{x}^{(i)}$ is unknown, but we use the Gaussian approximation $q(\mathbf{z}|\mathbf{x}^{(i)})$ with mean vector $\mu^{(i)}$ and covariance matrix $\mathrm{diag}(\sigma_1^2, \cdots, \sigma_d^2)^{(i)}$ instead. The goal of the algorithm is to maximize the variational or evidence lower bound (ELBO) on the marginal density of individual datapoints.

For a dataset $\mathbf{X} = \left\{\mathbf{x}^{(i)}\right\}_{i=1}^{N}$ consisting of N independent and identically distributed samples, the variational lower bound for the i^{th} datapoint or image $\mathbf{x}^{(i)}$ in the original VAE algorithm [4] is

$$ELBO\left(\mathbf{x}^{(i)}\right) = -D_{KL}\left(q\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \parallel p(\mathbf{z})\right) + \mathbb{E}_{q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log p\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\right]. \tag{1}$$

The first term on the right-hand side is the negative Kullback–Leibler divergence between the posterior variational approximation $q(\mathbf{z}|\mathbf{x})$ and a prior distribution $\mathbf{z}$ which is selected to be a standard Gaussian distribution. The second term on the right-hand side is denoted as the expected reconstruction log-likelihood, and is referred to as the cross-entropy. Let n_z be the dimensionality of $\mathbf{z}$; then, the Kullback–Leibler divergence simplifies to

$$-D_{KL}\left(q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)||p(\mathbf{z})\right) = \int q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)\left(\log p(\mathbf{z}) - \log q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)\right)d\mathbf{z} \tag{2}$$

$$= \frac{1}{2}\sum_{j=1}^{n_z}\left(1 + \log\left((\sigma_j)^2\right) - (\mu_j)^2 - (\sigma_j)^2\right). \tag{3}$$

The expected reconstruction log-likelihood (cross-entropy) $E_{q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log p\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\right]$ can be estimated by sampling, i.e.,

$$\mathbb{E}_{q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log p\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\right] = \frac{1}{L}\sum_{l=1}^{L}\left(\log p\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right)\right), \tag{4}$$

where L denotes the number of samples for each datapoint and we set $L = 1$ in our study. Supposing data $\mathbf{x}$ given $\mathbf{z}$ has the following probability density,

$$\log p(\mathbf{x}|\mathbf{z}) = \sum_{i=1}^{n_x}(x_i \log y_i + (1 - x_i)\log(1 - y_i)), \tag{5}$$

where $\mathbf{y}$ is the output of the decoder. Therefore, the loss function can be calculated by

$$\mathcal{L}\left(\mathbf{x}^{(i)}\right) = -ELBO\left(\mathbf{x}^{(i)}\right) = D_{KL}\left(q\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \parallel p(\mathbf{z})\right) - \frac{1}{L}\sum_{l=1}^{L}\left(\log p\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right)\right). \tag{6}$$

For our work, the loss function is modified to improve the robustness of the variational autoencoder, something that will be discussed in Section 4.

2.2. Comparison with Other Generative Machine Learning Methods

The paradigm of generative adversarial networks (GANs) is a recent advance in generative machine learning methods. The basic idea of GANs was published in a 2010 blog post by Niemitalo [13], and the name 'GAN' was introduced by Goodfellow et al. [14]. In comparison with variational autoencoders, generative adversarial networks are used for optimizing generative tasks specifically. GANs can produce models with true latent spaces, as is the case of bidirectional GAN (BiGAN) and adversarially learned inference (ALI) [15,16], which are designed to improve the performance of GANs. However, GANs cannot generate reasonable results when data are high-dimensional [17]. By contrast, as a probabilistic model, the specific goal of a variational autoencoder is to marginalize out noninformative variables during the training process. The ability to use complex priors in the latent space enables existing expert knowledge to be incorporated.

Bayesian networks form another generative model. Pearl [18] proposed the Bayesian network paradigm in 1985. Bayesian networks have a strong ability to capture the symbolic figures of input information and combine objective probabilities with subjective estimates for both qualitative and quantitative modeling. The basic concept of Bayesian networks is built on Bayes's theorem. Another effective way to solve for the posterior of the distribution derived from neural networks is to train and predict using variational inference techniques [19]. Compared with the original Bayesian network, the basic building blocks of deep networks provide multiple loss functions for making multitarget predictions, for transfer learning, and for varying outputs depending on the situation. The improvement of the deeper architectures, using VAE specifically, continues to occur.

Other generative models are now commonly combined with a variational autoencoder to improve performance. Ebbers et al. [20] developed a VAE with a hidden Markov model (HMM) as the latent model for discovering acoustic units. Dilokthanakul et al. [2] studied the use of Gaussian mixture models as the prior distribution of the VAE to perform unsupervised clustering through deep generative models. They showed a heuristic algorithm called 'minimum information constraint' and it is capable of improving the unsupervised clustering performance with this model. Srivastava and Sutton [1] presented the effective autoencoding variational Bayes-based inference method for latent Dirichlet allocation (LDA). This model solves the problems caused by autoencoding variational Bayes by the Dirichlet prior and by component collapsing. Additionally, this model matches traditional methods' inaccuracy with much better inference time.

3. Accounting for Risk with Coupled Entropy

Machine learning algorithms, including the VAE, have achieved efficient learning and inference for many image processing applications. Nevertheless, assuring accurate forecasts of the uncertainty is still a challenge. Problems such as outliers and overfitting impact the robustness of scientific prediction and engineering systems. This paper concentrates on assessing and improving the robustness of the VAE algorithm.

In this study, we draw upon the principles of nonlinear statistical coupling (NSC) [21,22] to define a generalization to information theory and apply the resulting entropic functions to the definition of the negative ELBO loss function for the training of the variational autoencoder [23]. NSC is derived from nonextensive statistical mechanics [24], which generalizes the variational calculus of maximum entropy to include constraints related to the nonlinear dynamics of complex systems and in turn to the nonexponential decay of the maximizing distributions. The NSC frame focuses this theory on the role of nonlinear coupling κ in generalizing entropy and its related functions. The approach defines a family of heavy-tailed (positive coupling) and compactly supported (negative coupling) distributions which maximize a generalized entropy function referred to as coupled entropy. The variational methods underlying NSC can be applied to a variety of problems in mathematical physics [25,26]. Here, we examine how NSC can broaden the role of approximate variational inference in machine learning to include sensitivity to the risks of outlier events occurring in the tail of the distribution of the phenomena being learned.

3.1. Assessing Probabilistic Forecasts with the Generalized Mean

First, proper metrics are needed to evaluate the accuracy and robustness of machine learning algorithms, such as VAE. The arithmetic mean and the standard deviation are widely used to measure the central tendency and fluctuation, respectively, of a random variable. Nevertheless, these are inappropriate for probabilities, which are formed by ratios. A random variable formed by the ratio of two independent random variables has a central tendency determined by the geometric mean, as described by McAlister [27]. Information theory addresses this issue by taking the logarithm of the probabilities, then the arithmetic mean; however, we will show that the generalizations of information theory are easier to report and visualize in the probability domain.

In [28], a risk profile was introduced, which is the spectrum of the generalized means of probabilities and provides an assessment of the the central tendency and fluctuations of probabilistic inferences. The generalized mean $(\frac{1}{N}\sum_{i=1}^{N} p_i^r)^{\frac{1}{r}}$ is a translation of generalized information-theoretic metrics back to the probability domain, and is derived in the next section. Its use as a metric for evaluating and training inference algorithms is related to the Wasserstein distance [29], which incorporates the generalized mean. The accuracy of the likelihoods is measured with robust, neutral, and decisive risk bias using the $r = -\frac{2}{3}$, $r = 0$ (geometric) and $r = 1$ (arithmetic) means, respectively. With no risk bias ($r = 0$), the geometric mean is equivalent to transforming the cross-entropy between the forecast p_i and the distribution of the test samples to the probability domain. The arithmetic mean ($r = 1$) is a simple measure of the Decisiveness (i.e., were the class probabilities in the right order so that a correct decision can be made?). This measure de-weights probabilities near zero since increasing r reduces the influence of small probabilities on the average. To complement the arithmetic mean, we choose a negative conjugate value. The conjugate is not the harmonic mean ($r = -1$) because this turns out to be too severe a test. Instead, $r = -\frac{2}{3}$ is chosen based on a dual transformation between heavy-tail (positive κ) and compact-support (negative κ) domains of the coupled Gaussian distribution. The risk sensitivity r can be decomposed into the nonlinear coupling and the power and dimension of the variable $r(\kappa, \alpha, d) = \frac{-\alpha\kappa}{1+d\kappa}$. The dual transformation between the positive/negative domains of the coupled Gaussians has the following relationship: $\hat{\kappa} \Leftrightarrow \frac{-\kappa}{1+d\kappa}$. Taking $\alpha = 2$ and $d = 1$, the coupling for a risk bias of one is $1 = \frac{-2\kappa}{1+\kappa} \Rightarrow \kappa = -\frac{1}{3}$ and the conjugate values are $\hat{\kappa} = \frac{\frac{1}{3}}{1-\frac{1}{3}} = \frac{1}{2}$ and $\hat{r} = \frac{-2\cdot\frac{1}{2}}{1+\frac{1}{2}} = -\frac{2}{3}$ [23]. The Robustness metric increases the weight of probabilities near zero since negative powers invert the probabilities prior to the average.

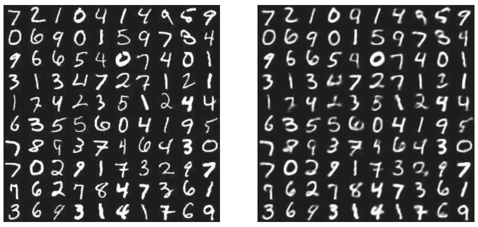
For simplicity, we refer to these three metrics as the Robustness, Accuracy, and Decisiveness. The label 'accuracy' is used for the neutral accuracy, since 'neutralness' is not appropriate and 'neutral' does not express that this metric is the central tendency of the accuracy. Summarizing:

$$Decisiveness\ (\text{arithmetic mean}) : \frac{1}{N}\sum_{i=1}^{N} p_i. \tag{7}$$

$$Accuracy\ (\text{geometric mean}) : \prod_{i=1}^{N} p_i^{\frac{1}{N}}. \tag{8}$$

$$Robustness\ (-2/3\,\text{mean}) : \left(\frac{1}{N}\sum_{i=1}^{N} p_i^{-\frac{2}{3}}\right)^{-\frac{3}{2}}. \tag{9}$$

Similar to the standard deviation, the arithmetic mean and $-2/3$ mean play roles as measures of the fluctuation. Figure 2 shows an example of input images from the MNIST dataset and the generated output images produced by the VAE. Despite the blur in some output images, the VAE succeeds in generating very similar images to the input. However, the histogram in Figure 3, which plots the frequency of the likelihoods over a log scale, shows that the probabilities of ground truth range over a large scale. The geometric mean or Accuracy captures the central tendency of the distribution at 10^{-37} . The Robustness and the Decisiveness capture the span of the fluctuation in the distribution. The $-2/3$ mean or Robustness is 10^{-77} and the arithmetic mean or Decisiveness is 10^{-15}. The minimal value of the $-2/3$ mean metric is an indicator of the poor robustness of the VAE model, which can be improved. We measure and display the performance in the probability space in order to simplify the comparison between the three metrics. In the next subsection, we will show their relationship with a generalization of the log-likelihood. If, however, we were to plot histograms in the log-space, separate histograms would be required for each metric. By using the probability space, we can display one histogram overlaid with three different means. Appendix A.2 describes the origin of the Robustness–Accuracy–Decisiveness metrics.



(**a**) (**b**)

Figure 2. Example set of (**a**) MNIST input images and (**b**) VAE-generated output images.

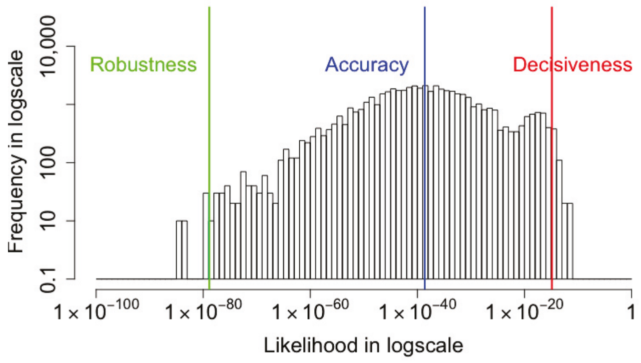


Figure 3. A histogram of the likelihoods that the VAE-reconstructed images match the input images. The objective of the coupled VAE research is to demonstrate that the Robustness, which is the $-2/3$ generalized mean, can be increased by penalizing the cost of producing outlier reconstructions. The Accuracy is the exponential of the average log-likelihood and the Decisiveness is the arithmetic mean.

In order to improve performance against the robust metric, the training of the variational autoencoder needs to incorporate this generalized metric. To do so, we derive a coupled loss function in the next subsection.

3.2. Definition of Negative Coupled ELBO

As we discussed in Section 2, the goal of a VAE algorithm is to optimize a low-dimensional model of a high-dimensional input dataset. This is accomplished using approximate variational inference by maximizing an evidence lower bound (ELBO). Equivalently, the negative ELBO defines a loss function which can be minimized, $\mathcal{L}(x^{(i)}) = -ELBO(x^{(i)})$. In this paper, we provide initial evidence that the accuracy and robustness of the variational inference can be improved by generalizing the negative ELBO to account for the risk of outlier events. Here, we provide a definition of the generalization and in Appendix A.1 a derivation is provided.

The generalized loss function in the coupled variational autoencoder (VAE) method is defined as follows.

Definition 1. *(Negative Coupled ELBO). Given the i^{th} datapoint $\mathbf{x}^{(i)}$, the corresponding latent variable value $\mathbf{z}$, and the output value $\mathbf{y}$ of the decoder using the Bernoulli distribution, then the loss function for the coupled VAE algorithm is given by*

$$\mathcal{L}_\kappa\left(\mathbf{x}^{(i)}\right) = D_\kappa\left(q\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \parallel p(\mathbf{z})\right) + H_\kappa(\mathbf{x}, \mathbf{y}), \tag{10}$$

where

$$
\begin{aligned}
&D_\kappa(q(\mathbf{z}|\mathbf{x}^{(i)}) \parallel p(\mathbf{z})) \\
&\equiv \prod_{j=1}^{n_z} \int \frac{q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}}}{\int q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}} dz_j} \frac{1}{2} (\ln_\kappa(q(z_j|\mathbf{x}^{(i)})^{-\frac{2}{1+\kappa}}) - \ln_\kappa(p(z_j)^{-\frac{2}{1+\kappa}}))dz_j \\
&= \prod_{j=1}^{n_z} \frac{1}{2\kappa} \int \frac{\left(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z_j-\mu_i)^2}{2\sigma^2}}\right)^{1+\frac{2\kappa}{1+\kappa}}}{\int \left(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z_j-\mu_i)^2}{2\sigma^2}}\right)^{1+\frac{2\kappa}{1+\kappa}} dz_j} \cdot \left(\left(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z_j-\mu_i)^2}{2\sigma^2}}\right)^{-\frac{2\kappa}{1+\kappa}} - \left(\frac{1}{\sqrt{2\pi}} e^{-\frac{z_j{}^2}{2}}\right)^{-\frac{2\kappa}{1+\kappa}}\right) dz_j
\end{aligned}
\tag{11}
$$

is the generalized (coupled) KL-divergence in the original loss function in Equation (6), and

$$
H_\kappa(\mathbf{x}, \mathbf{y}) \equiv -\frac{1}{2L} \sum_{l=1}^{L} \sum_{i=1}^{n_x} \left(x_i \ln_\kappa \left((y_i)^{\frac{2}{1+\kappa}} \right) + (1 - x_i) \ln_\kappa \left((1 - y_i)^{\frac{2}{1+\kappa}} \right) \right) \tag{12}
$$

is the generalized reconstruction loss (coupled cross-entropy) in the original loss function in Equation (6).

In the next section, we show preliminary experimental evidence that the negative coupled ELBO can be used to improve the robustness and accuracy of the variational inference. We show that increasing the coupling parameter of the loss function has the effect of increasing the Accuracy (8) and Robustness (9) metrics of the generated data. Additionally, we show that the improvement in the generation process is not at the expense of the divergence between the posterior and the prior latent distributions. Thus, the overall ELBO is improved, indicating an improvement in the approximate variational inference. Furthermore, in Section 6, we show that improvements are more substantial when the algorithm is seeded by images from the corrupted MNIST database. While the experimental results of this report focus on a two-layer dense neural network and the (corrupted)-MNIST datasets, the generalization of information-theoretic cost functions for machine learning training is applicable to a broader range of architectures and datasets. For instance, the CIFAR-10 reconstruction is typically processed with a deep neural network [30] and is planned for future research.

4. Results Using the MNIST Handwritten Numerals

The MNIST handwritten digit database is a large database of handwritten digits consisting of a training set of 60,000 images and a test set of 10,000 images widely used for evaluating machine learning and pattern recognition methods. The digits have been size-normalized and centered in fixed-size images. Each image in the database contains 28 by 28 grayscale pixels. Pixel values vary from 0 to 255. Zero means the pixel is white, or background, while 255 means the pixel is black, or foreground [31]. In this and the next section, we examine the performance of the coupled VAE algorithm in reconstructing images of the MNIST database. In Section 6, we show the stability of the coupled VAE when reconstruction is distorted by samples from the corrupted MNIST database.

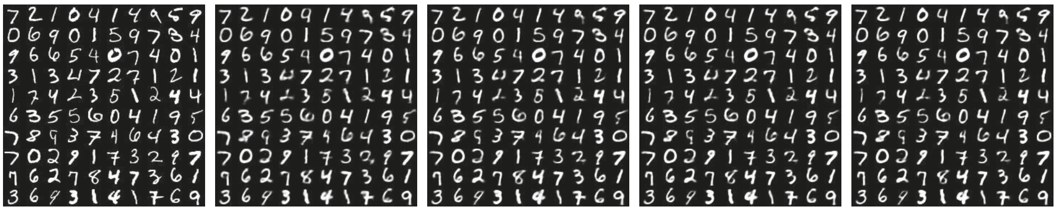
For this research, we used the MNIST database as the input since it was used in the traditional VAE. Specifically, input $\mathbf{x}$ is a batch of 28 by 28 pixel photos of handwritten numbers. The encoder encodes the data, which are 784-dimensional for each image in a batch into the latent layer space. For our experiment, the dimension of the latent variable $\mathbf{z}$ can be from 2 to 20. Taking the latent layers $\mathbf{z}$ as the input, the probability distribution of each pixel is computed using a Bernoulli or Gaussian distribution by the decoder. The decoder outputs the corresponding 784 parameters to reconstruct an image. We used specific numbers of images from the training set as the batch size and a fixed number of epochs. Additionally, for the learned MNIST manifold, visualizations of learned data and reproduced results were plotted. The algorithm and experiments were developed with Python and the TensorFlow library. Our Python code can be found in the Data Availability Statement.

The input images and output images for different values of coupling κ are shown in Figure 4. $\kappa = 0$ represents the original VAE model. Compared with the original algorithm, output images generated by the modified coupled VAE model show small improvements in detail and clarity. For instance, the fifth digit in the first row of the input images is '4', but the output image in the original VAE is more like '9' rather than '4', while the coupled VAE method generates '4' correctly. For the seventh digit '4' in the first row, the generated image in the coupled VAE has an improved clarity compared to the traditional VAE.

Figure 5 shows the likelihood histograms for 5000 input images with coupling values of $\kappa = 0, 0.025, 0.05, 0.1$. The red, blue, and green lines represent the arithmetic mean (decisiveness), geometric mean (central tendency), and $-2/3$ mean (robustness), respectively. When $\kappa = 0$, the minimal value of the Robustness metric indicates that the original VAE suffers from poor robustness. As κ becomes large, the geometric mean and the $-2/3$ mean metrics start to increase while the arithmetic mean metric mostly stays the same. Since the probability of producing a correct image by a uniform random sampling is $\frac{1}{2^{28\times28}} = 9.8 \times 10^{-237}$, the accuracy achieved by the VAE algorithm is significantly improved, even though the absolute value of the Accuracy metric seems small. As the coupling κ increases, the coupled loss function approaches infinity faster. This eventually causes computational errors. For instance, when $\kappa = 0.2$, the loss function has a computational error at the 53^{rd} epoch; when $\kappa = 0.5$, the loss function has a computational error at the 8^{th} epoch. Further investigations of the computational bounds of the algorithm are planned. The specific relationship between coupling κ and probabilities for input images is shown in Table 1. The increased Robustness metric shows that the modified loss does improve the robustness of the the reconstructed image. In the next section, we also examine the performance of the divergence between the posterior and prior distributions of the latent layer.

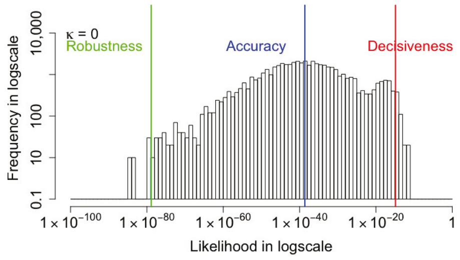
Furthermore, compared with the original VAE model, the geometric mean, which measures the accuracy of the input image likelihood, is larger for the coupled algorithm. The improvement of this metric means that the input images (truth) are assigned to higher likelihoods on average by the coupled VAE model.

The standard deviation σ of latent variables $\mathbf{z}$ is shown in rose plots in Figure 6. The angular position of a bar represents the value of σ, clockwise from 0 to 1. The radius of the bar measures the frequency of different σ values from 0 to 100. As the coupling κ increases, the range and the average value of these standard deviations decrease. To be specific, when $\kappa = 0$, σ of all dimensions in all 5000 batches ranges from 0.09 to 0.72; when $\kappa = 0.025$, σ ranges from 0.02 to 0.3; when $\kappa = 0.05$, σ ranges from 0.001 to 0.09; when $\kappa = 0.1$, σ ranges from 0.00007 to 0.06.

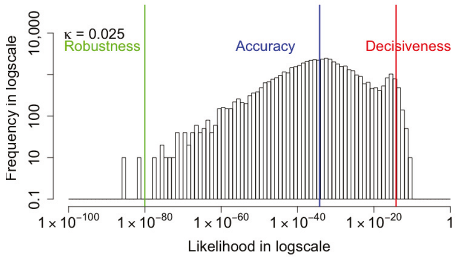


(**a**) Input image (**b**) $\kappa = 0$ (**c**) $\kappa = 0.025$ (**d**) $\kappa = 0.05$ (**e**) $\kappa = 0.1$

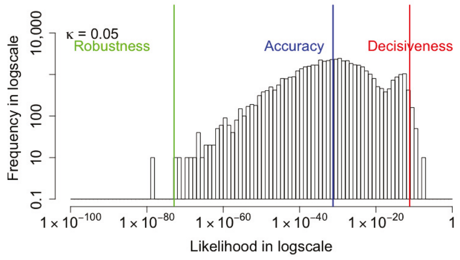
Figure 4. (**a**) The MNIST input images and (**b**) the output images generated by the original VAE. (**c**–**e**) The output images generated by the modified coupled VAE model show small improvements in detail and clarity. For instance, the fifth digit in the first row of the input images is '4', but the output image in the original VAE is more like '9' rather than '4', while the coupled VAE method produced '4' correctly.



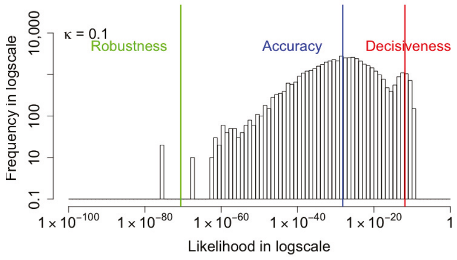
(**a**) $\kappa = 0$



(**b**) $\kappa = 0.025$



(**c**) $\kappa = 0.05$

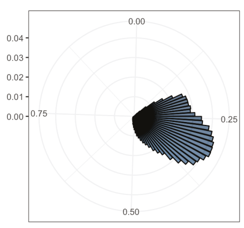


(**d**) $\kappa = 0.1$

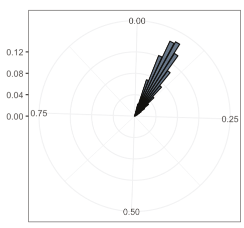
Figure 5. The histograms of likelihood for the reconstruction of the input images with various coupling κ values. The red, blue, and green lines represent the arithmetic mean (Decisiveness), geometric mean (Accuracy), and $-2/3$ mean (Robustness), respectively. The minimal value of the Robustness metric indicates that the original VAE suffers from poor robustness. As κ increases, the Robustness and Accuracy improve while the Decisiveness is mostly unchanged.

Table 1. The Decisiveness, Accuracy, and Robustness of the reconstruction likelihood as a function of the coupling κ.

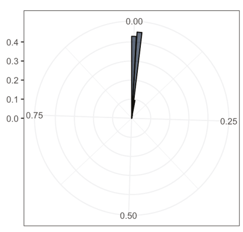
Coupling κ	**Decisiveness**	**Accuracy**	**Robustness**
0	1.31×10^{-15}	2.41×10^{-39}	1.40×10^{-79}
0.025	6.61×10^{-15}	5.89×10^{-35}	9.91×10^{-81}
0.05	7.18×10^{-12}	5.80×10^{-32}	1.31×10^{-73}
0.1	1.34×10^{-12}	7.09×10^{-29}	2.57×10^{-71}



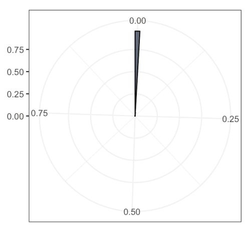
(**a**) $\kappa = 0$



(**b**) $\kappa = 0.025$



(**c**) $\kappa = 0.05$

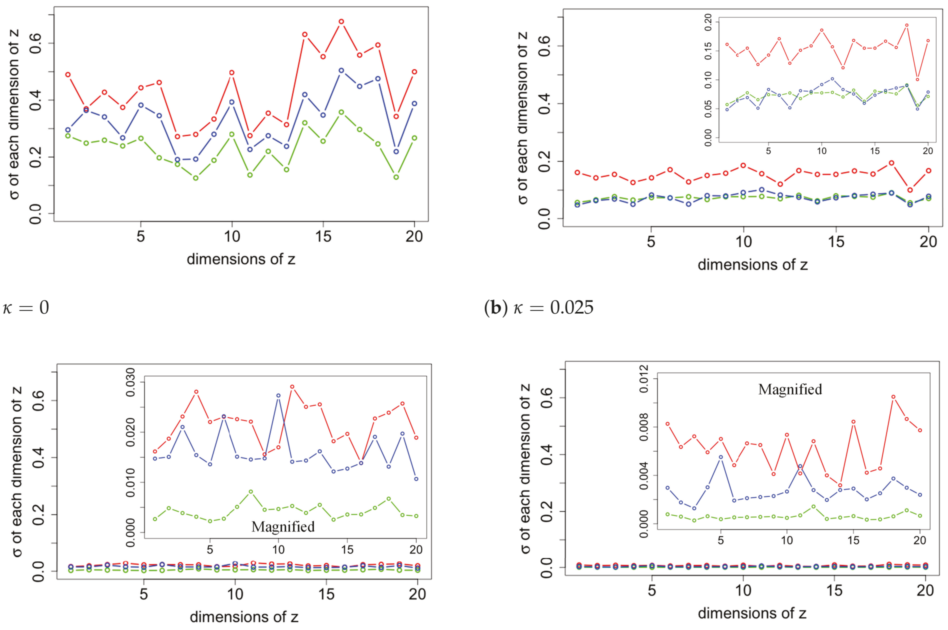


(**d**) $\kappa = 0.1$

Figure 6. The rose plots of the various standard deviation values in 20 dimensions. The range and average values of these standard deviations are reduced as coupling increases.

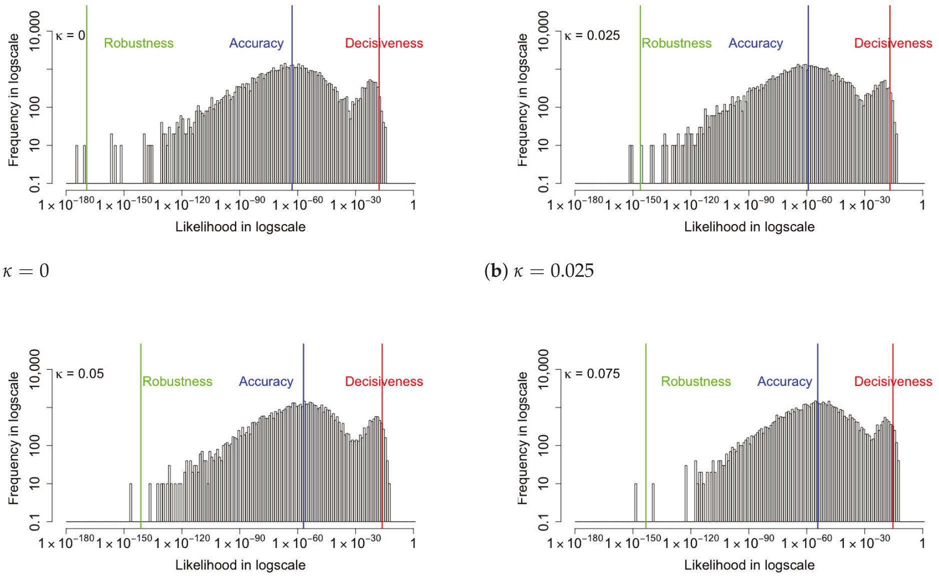
We note that as coupling parameter κ increases, the variability of the latent space diminishes. One possible method to address this problem is to use heavy-tail distribution in the latent layer. Chen et al. [32] and Nelson [23] used the Student's t as the distribution [33] of the latent layer to incorporate heavy-tail decay.

We choose samples in which the likelihoods of input images are close to the three metrics and plot the standard deviation σ of each dimension of the latent variable $\mathbf{z}$ of these samples in Figure 7. The red, blue, and green lines represent samples near the decisiveness, accuracy, and robustness, respectively. It shows that when $\kappa = 0$, the standard deviations of $\mathbf{z}$ range from 0.1 to 0.7. However, as κ increases, values of σ fluctuate less and decrease toward 0. Magnified plots are shown to visualize the results further. The general trend for σ is to be more significant for samples near decisiveness, intermediate near the accuracy, and smaller for samples near robustness. An exception is $\kappa = 0.025$, where σ overlaps for samples near the robustness and accuracy. The histogram likelihood plots with a two-dimensional latent variable are shown in Figure 8. The increased values of the arithmetic mean metric and $-2/3$ mean metric show that the accuracy and robustness of the output MNIST images in the VAE model have been improved, consistent with the result in the 20-D model. While the performance improvements are modest, we will show in Section 6 that the performance improvements when the algorithm is seeded with corrupted images is much more substantial. First, we provide a visualization of the changes in the latent distribution using two dimensions.



(**a**) $\kappa = 0$ (**b**) $\kappa = 0.025$ (**c**) $\kappa = 0.05$ (**d**) $\kappa = 0.1$

Figure 7. The standard deviation of latent variable samples near the three generalized mean metrics. The red, blue, and green lines represent samples near the Decisiveness, Accuracy, and Robustness, respectively. As κ increases, values of σ fluctuate less and decrease toward 0. Magnified plots are shown to visualize the results further.



(**a**) $\kappa = 0$ (**b**) $\kappa = 0.025$

(**c**) $\kappa = 0.05$ (**d**) $\kappa = 0.1$

Figure 8. The histogram likelihood plots with a two-dimensional latent variable. Like the 20-D model, the increased values of the arithmetic mean metric and $-2/3$ mean metric show that the accuracy and robustness of the VAE model have been improved.

5. Visualization of Latent Distribution

In order to understand the relationship between increasing coupling of the loss function with the means and the standard deviations of the Gaussian model, we examine a two-dimensional model which can be visualized. Compared with the high-dimensional model, the probability likelihoods for the two-dimensional model are lower, indicating that the higher dimensions do improve the model. Nevertheless, like the 20-dimensional model, the distribution of likelihood is compressed toward higher values as the coupling increases and, therefore, can be used to analyze the results further. Larger likelihood of input images along with both means closer to the origin and smaller standard deviations of latent variables are the primary characteristics as the coupling parameter of the loss function is increased. As a result, both the robustness and accuracy of likelihoods increase. To be specific, when κ increases from 0 to 0.075, the geometric mean metric increases from 1.20×10^{-63} to 4.67×10^{-55}, and the $-2/3$ mean metric increases from 5.03×10^{-170} to 5.17×10^{-144}, while the arithmetic metric does not change very much. In this case, the reconstructed images have a higher probability of replicating the input image using the coupled VAE method.

The rose plots in Figure 9 show that the range and variability of the mean values of latent variables decrease as the coupling κ increases. From the view of means, the posterior distribution of the latent space is closer to the prior, the standard Gaussian distribution. From the view of standard deviations, the posterior distribution of the latent space is further from the prior.

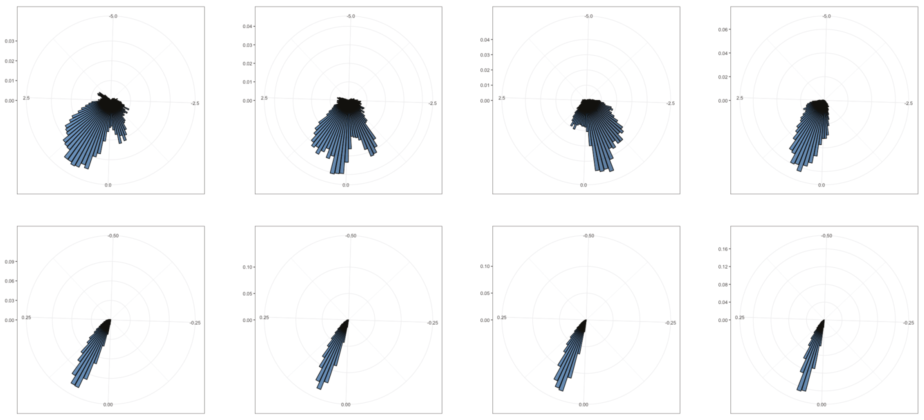

(**a**) $\kappa = 0$ (**b**) $\kappa = 0.025$ (**c**) $\kappa = 0.05$ (**d**) $\kappa = 0.075$

Figure 9. The rose plots of the various mean (above four figures) and standard deviation (below four figures) values in 2 dimensions. The range of means is reduced and mean values become closer to 0 as coupling increases.

The latent space plots shown in Figure 10 are the visualizations of images of the numerals from 0 to 9. Images are embedded in a 2D map where the axis is the values of the 2D latent variable. The same color represents images that belong to the same numeral, and they cluster together since they have higher similarity to each other. The distances between spots represent the similarities of images. The latent space plots show that the different clusters shrink together more tightly when coupling becomes larger. The plots shown in Figure 11 are the visualizations of the learned data manifold generated by the decoder network of the coupled VAE model. A grid of values from a two-dimensional Gaussian distribution is sampled. The distinct digits each exist in different regions of the latent space and smoothly transform from one digit to another. This smooth transformation can be quite useful when the interpolation between two observations is needed. Additionally, the distribution of distinct digits in the plot becomes more even, and the sharpness of the digits increases when κ increases.

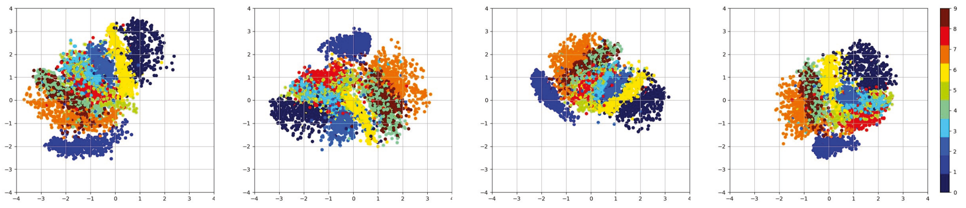

(**a**) $\kappa = 0$ (**b**) $\kappa = 0.025$ (**c**) $\kappa = 0.05$ (**d**) $\kappa = 0.075$

Figure 10. The plot of the latent space of VAE trained for 200 epochs on MNIST with various κ values. Different numerals cluster together more tightly as coupling κ increases.

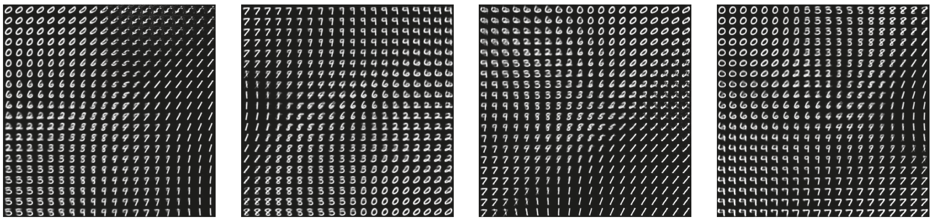

(**a**) $\kappa = 0$ (**b**) $\kappa = 0.025$ (**c**) $\kappa = 0.05$ (**d**) $\kappa = 0.075$

Figure 11. The plot of visualization of learned data manifold for generative models with the axes as the values of each dimension of latent variables. The distinct digits each exist in different regions of the latent space and smoothly transform from one digit to another.

As shown in Table 2, as the coupling increases from 0 to 0.075, the negative ELBO (the loss) decreases from 172.3 to 146.7, the coupled KL-divergence decreases from 5.8 to 5.6, and the coupled reconstruction loss decreases from 166.5 to 141.1. It shows that the reconstruction loss plays a dominant role (with proportion over 96%), while the divergence term has a much lower effect (with proportion under 4%) in the loss function. The overall improvement of coupled loss is based on both the smaller coupled KL-divergence and the smaller coupled reconstruction error, instead of a trade-off between them. There is a high degree of variability in this improvement, so there are reasons to be cautious about the degree of improvement. In addition, since the coupled loss function is adjusting the metric, the property being measured is also adjusting. Part of our future research plan is to explore how the relative performance between the reconstruction and the latent space can be compared.

Table 2. Components of coupled ELBO with a 2-dimensional latent layer under different values of coupling. The improvement in the coupled KL-divergence is very slight, while it is larger for the coupled reconstruction loss.

Coupling κ	Coupled KL-Divergence	Coupled RE Loss	Coupled ELBO	KL Proportion	RE Proportion
0	5.8 +/− 1.7	166.5 +/− 52.2	172.3	3.38%	96.62%
0.025	5.7 +/− 1.6	156.4 +/− 49.8	162.1	3.53%	96.47%
0.05	5.6 +/− 1.6	149.2 +/− 46.6	154.8	3.61%	96.39%
0.075	5.6 +/− 1.7	141.1 +/− 44.6	146.7	3.82%	96.18%

6. Performance with Corrupted Images

We also evaluate the performance of the coupled VAE algorithm when keyed by images from the corrupted MNIST (C-MNIST) dataset [34]. The reconstructed images under 5 different corruptions: Gaussian corruption, glass blur corruption, impulse noise corruption, shot noise corruption and shear corruption, with two coupling values $\kappa = 0.0$ and $\kappa = 0.1$ are shown in the Figure 12. Based on the visualization of the generated images, the qualitative visual improvement in clarity using the coupling is modest.

We also conduct the further analyses for the performance of the coupled VAE with each corruption. For the MNIST images with Gaussian corruption, as shown in the Figure 13, when the coupling parameter κ increases, all the three metrics—robustness, central tendency, and decisiveness—increase. The robustness improves the most, central tendency is the next, and decisiveness has the least improvement. Furthermore, we confirm that the reconstruction improvement is not a trade-off with latent distribution divergence, as shown in Table 3. This is in contrast to the β-VAE [11] method which merely alters the weight between the reconstruction and divergence components of the negative ELBO cost function.

In the Table 3, analyses of the components of the coupled ELBO are provided. Comparisons as the coupling changes are somewhat confusing because the metric itself is changing. Therefore, as the coupling increases the measure of performance is more difficult. Nevertheless, there is still an overall tendency towards improved performance, even with this caveat. The second column shows that the coupled KL-divergence initially increases when moving away from the standard VAE design with $\kappa = 0$, however, it then steadily decreases with increasing κ. This may be due to the distinct difference between the logarithm and even a slight deviation from the logarithm. The coupled reconstruction loss (column three) shows steady improvement. The overall negative coupled ELBO shows consistent improvement as the coupling increases. The relative importance of the divergence and reconstruction varies as the coupling increases but in each case it is approximately a 15% to 85% relative weighting.

The improvement of the three metrics with glass blur corruption, impulse noise corruption, shot noise corruption and shear corruption is also observed and shown in Figures 14–17, respectively. Similar to the Gaussian corruption, all the three metrics gradually increase as the coupling parameter κ increases from 0 to 0.1. The respective analyses of the components of the coupled ELBO with glass blur corruption, impulse noise corruption, shot noise corruption and shear corruption are provided in Tables 4–7. The four corruptions share the consistent results, the coupled KL-divergence initially increases when moving away from the standard VAE design with $\kappa \leq 0.025$, but it then steadily decreases with increasing κ. The overall negative coupled ELBO shows consistent improvement as κ increases. It means that if the coupling parameter is relatively large (> 0.025), both the KL-divergence and the reconstruction loss will be improved, thus the overall improvement of the algorithm is not a trade-off between the reconstruction accuracy and the latent distribution divergence.

Table 3. The components of the coupled ELBO under **Gaussian** corruptions are provided in the table. The coupled KL-divergence initially increases when moving away from the standard VAE design with $\kappa = 0$ to $\kappa = 0.025$, however, it then steadily decreases with increasing κ. The coupled reconstruction loss (column three) shows steady improvement. The overall negative coupled ELBO shows consistent improvement as the coupling increases. The relative importance of the divergence and reconstruction varies as the coupling increases but in each case it is approximately a 15% to 85% relative weighting.

Coupling κ	Coupled KL-Divergence	Coupled RE Loss	Coupled ELBO	KL Proportion	RE Proportion
$\kappa = 0$	23.9 +/− 3.8	131.6 +/− 40.7	155.5	15.34%	84.66%
$\kappa = 0.025$	29.6 +/− 2.3	119.9 +/− 38.5	149.5	19.80%	80.20%
$\kappa = 0.05$	26.0 +/− 0.9	111.1 +/− 36.5	137.1	18.94%	80.06%
$\kappa = 0.075$	21.4 +/− 0.5	104.4 +/− 34.3	125.8	16.98%	83.02%
$\kappa = 0.1$	18.4 +/− 0.6	98.9 +/− 32.7	117.3	15.71%	84.28%

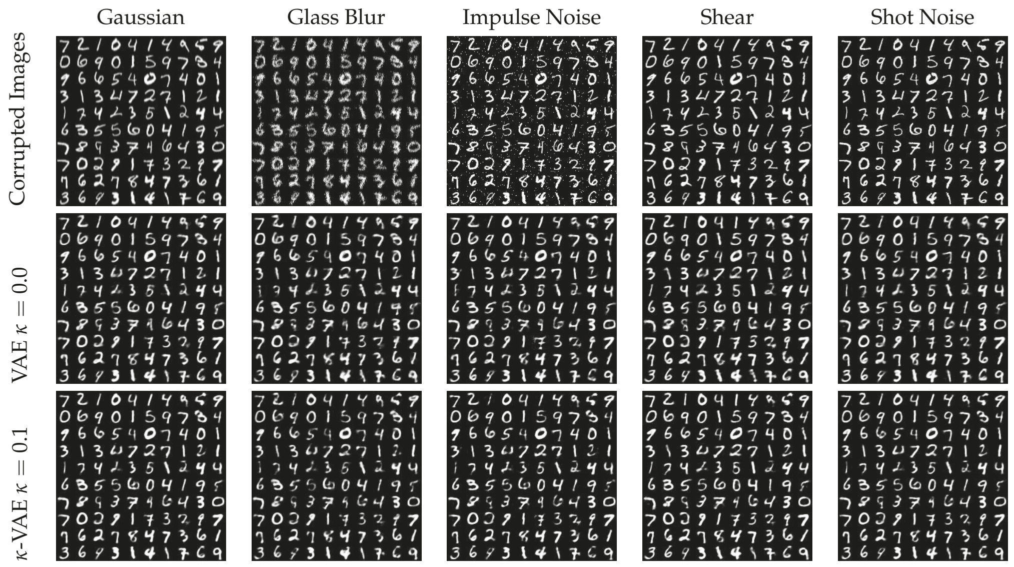


Figure 12. The images with 5 different corruptions are shown in the first row. The reconstructed images when $\kappa = 0.0$ and $\kappa = 0.1$ are shown in the second and third rows, respectively. The qualitative visual improvement in clarity using the coupling is modest.

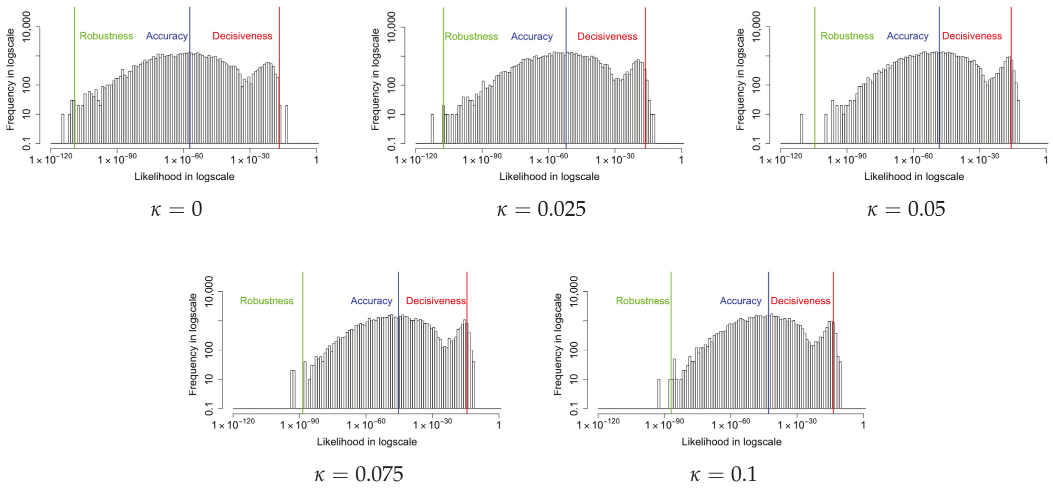


Figure 13. The histograms of marginal likelihood for the MNIST images with **Gaussian** corruption shown. All three metrics increase as the coupling parameter κ increases. The robustness improves the most, central tendency is the next, and decisiveness has the least improvement. From $\kappa = 0.0$ to $\kappa = 0.1$, the Robustness improves from $10^{-109.2}$ to $10^{-87.0}$, the Accuracy improves from $10^{-57.2}$ to $10^{-42.9}$, and the Decisiveness improves from $10^{-16.8}$ to $10^{-13.6}$.

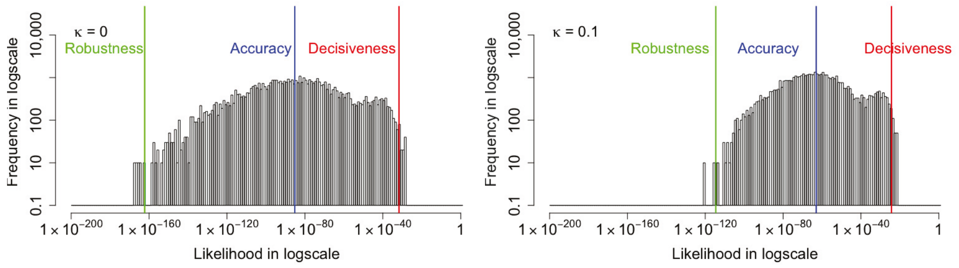


Figure 14. The histograms of marginal likelihood for the MNIST images with **glass blur** corruption are shown. All the three metrics increase as the coupling parameter κ increases from 0 to 0.1.

Table 4. The components of the coupled ELBO under **glass blur** corruptions are provided in the table. The coupled KL-divergence initially increases when moving away from the standard VAE design with $\kappa \le 0.025$, but it then steadily decreases with increasing κ. The coupled reconstruction loss shows steady improvement. The overall negative coupled ELBO shows consistent improvement as κ increases.

Coupling κ	**Coupled KL-Divergence**	**Coupled RE Loss**	**Coupled ELBO**	**KL Proportion**	**RE Proportion**
$\kappa = 0$	22.3 +/− 3.5	196.1 +/− 55.3	218.4	10.19%	89.81%
$\kappa = 0.025$	29.4 +/− 2.0	178.8 +/− 50.1	208.2	14.12%	85.88%
$\kappa = 0.05$	25.5 +/− 0.7	164.1 +/− 45.7	189.6	13.44%	86.56%
$\kappa = 0.075$	20.9 +/− 0.4	154.0 +/− 43.0	174.9	11.96%	88.04%
$\kappa = 0.1$	18.0 +/− 0.4	145.1 +/− 40.0	163.1	11.05%	88.95%

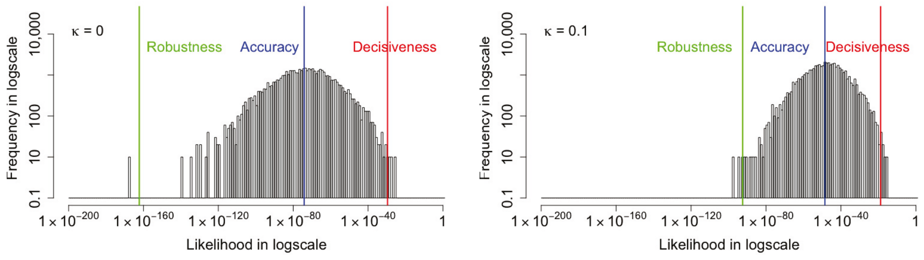


Figure 15. The histograms of marginal likelihood for the MNIST images with **impulse noise** corruption are shown. All the three metrics increase as the coupling κ increases from 0 to 0.1.

Table 5. The components of the coupled ELBO under **impulse noise** corruptions are provided in the table. The coupled KL-divergence initially increases when moving away from the standard VAE design with $\kappa \le 0.025$, but it then steadily decreases with increasing κ. The overall negative coupled ELBO shows consistent improvement as κ increases.

Coupling κ	**Coupled KL-Divergence**	**Coupled RE Loss**	**Coupled ELBO**	**KL Proportion**	**RE Proportion**
$\kappa = 0$	24.2 +/− 3.8	170.7 +/− 34.7	195.0	12.43%	87.57%
$\kappa = 0.025$	29.9 +/− 2.2	148.0 +/− 31.0	177.9	16.81%	83.19%
$\kappa = 0.05$	26.0 +/− 0.8	131.6 +/− 28.5	157.7	16.52%	83.48%
$\kappa = 0.075$	21.4 +/− 0.6	120.9 +/− 26.7	142.3	15.05%	84.95%
$\kappa = 0.1$	18.5 +/− 0.6	111.8 +/− 25.2	130.3	14.21%	85.79%

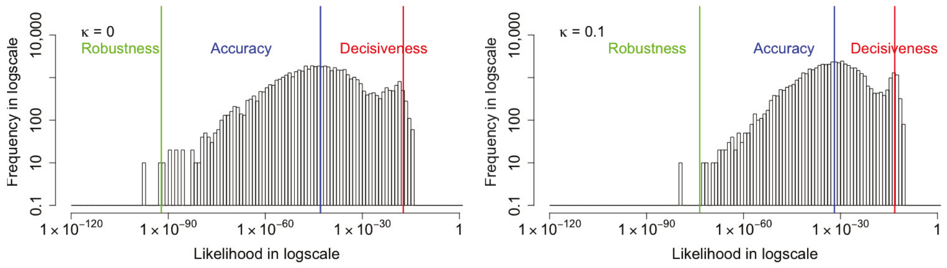


Figure 16. The histograms of marginal likelihood for the MNIST images with **shot noise** corruption are shown. All the three metrics increase as the coupling parameter κ increases from 0 to 0.1.

Table 6. The components of the coupled ELBO under **shot noise** corruptions are provided in the table. The coupled KL-divergence increases when moving away from the standard VAE design with $\kappa \leq 0.025$, but it then steadily decreases with increasing κ. The coupled reconstruction loss shows steady improvement. The overall negative coupled ELBO shows consistent improvement as κ increases.

Coupling κ	Coupled KL-Divergence	Coupled RE Loss	Coupled ELBO	KL Proportion	RE Proportion
$\kappa = 0$	23.9 +/− 3.8	98.9 +/− 28.3	122.8	19.45%	80.55%
$\kappa = 0.025$	29.9 +/− 2.4	88.9 +/− 26.2	118.8	25.14%	74.86%
$\kappa = 0.05$	26.1 +/− 1.0	81.8 +/− 25.0	108.0	24.21%	75.80%
$\kappa = 0.075$	21.6 +/− 0.7	77.6 +/− 23.9	99.2	21.80%	78.20%
$\kappa = 0.1$	18.6 +/− 0.6	73.4 +/− 22.8	92.0	20.17%	79.83%

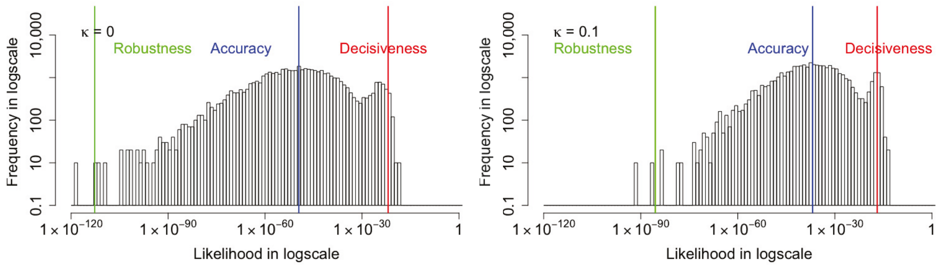


Figure 17. The histograms of marginal likelihood for the MNIST images with **shear** corruption are shown. All the three metrics increase as the coupling parameter κ increases from 0 to 0.1.

Table 7. The components of the coupled ELBO under **shear** corruptions are provided. The coupled KL-divergence increases when moving away from the standard VAE design with $\kappa \leq 0.025$, but it then steadily decreases with increasing κ. The coupled reconstruction loss shows steady improvement. The overall negative coupled ELBO shows consistent improvement as κ increases.

Coupling κ	Coupled KL-Divergence	Coupled RE Loss	Coupled ELBO	KL Proportion	RE Proportion
$\kappa = 0$	24.8 +/− 4.0	114.1 +/− 31.7	138.9	17.85%	82.15%
$\kappa = 0.025$	30.4 +/− 2.4	102.3 +/− 29.0	132.7	22.92%	77.08%
$\kappa = 0.05$	26.1 +/− 0.9	94.7 +/− 27.5	120.8	21.61%	78.39%
$\kappa = 0.075$	21.8 +/− 0.7	89.5 +/− 26.3	111.3	19.61%	80.39%
$\kappa = 0.1$	18.6 +/− 0.6	84.9 +/− 24.9	103.5	17.97%	82.03%

7. Discussion and Conclusions

This investigation sought to determine whether the accuracy and robustness of variational autoencoders can be improved using certain statistical methods developed within the area of complex systems theory. Our investigation provides evidence that the tail shape of the negative evidence lower bound can be controlled in such a way that the cost of outlier events is adjustable. We refer to this method as a coupled VAE, since the control parameter models the nonlinear deviation from the exponential and logarithmic functions of linear analysis. A positive coupling parameter increases the cost of these tail events and thereby trains the algorithm to be robust against such outliers. Additionally, this improves both the accuracy of reconstructed images and reduces the divergence of the posterior latent distribution from the prior. We have been able to document this improvement using the histogram of the reconstructed marginal likelihoods. Metrics of the histogram are formed from the arithmetic mean, geometric mean, and $-2/3$ mean, which represent Decisiveness, Accuracy, and Robustness, respectively. Both the accuracy and the robustness are improved by increasing the coupling of the loss function. There is a limit to such increases in the coupling beyond which the training process no longer converges.

These performance improvements have been evaluated for the MNIST handwritten numeral dataset and its corrupted modification C-MNIST. We used a two-layer dense neural network for the encoder/decoder. The latent layer is a 20-dimensional Gaussian distribution and for visualization a 2-dimensional distribution was also examined. Without the corruption, we observed improvements in both components of the negative coupled ELBO loss function, namely the image reconstruction loss (marginal likelihood) and the latent distribution (divergence between the prior and posterior). Thus, the coupled VAE is able to improve the model representation, rather than just trading off reconstruction and divergence performance, as does the highly cited β-VAE design. The likelihood of the reconstructed image matching the original improves in Accuracy by 10^{10} and in Robustness by 10^{8} when the coupling parameter was increased from $\kappa = 0$ (the standard VAE) to $\kappa = 0.1$ (the largest value of the coupled VAE reported). The Decisiveness did not change significantly, though there is potential that negative values of the coupling could influence this metric. The performance improvements when the algorithm is seeded by the C-MNIST dataset are far more significant, demonstrating the improved stability of the algorithm. All five corruptions examined (Gaussian, glass blur, impulse noise, shot noise, and shear) show significant improvement in Robustness and Accuracy and some improvement in the Decisiveness. For example, under the Gaussian corruption, the improvements in the reconstruction likelihood for Accuracy are 10^{14} and those for the Robustness are 10^{20} when the coupling parameter is increased from $\kappa = 0$ (the standard VAE) to $\kappa = 0.1$. The significant improvement in Robustness using the corrupted MNIST dataset demonstrates that the coupled negative ELBO cost function reduces the risk of overfitting by forcing the network to learn general solutions that are less likely to create outliers.

The modifications of the latent posterior distributions have been further examined using a two-dimensional representation. We show that the latent variables have both a tighter distribution of the mean about its prior value of zero, and a movement of standard deviations towards zero, away from the prior of one, as coupling κ increases. Overall, the coupled KL-divergence does indeed decrease as the coupling is increased, indicating improvement in the latent representation. Thus, improvements in the reconstruction evident from both visual clarity of images and increased accuracy in measured likelihoods are not due to a trade-off with the latent representation. Rather, the negative coupled ELBO metric shows improvement in both latent layer divergence and output image reconstruction. This improvement in the two components of the evidence lower bound provides evidence that the coupled VAE improves the approximate variational inference of the model.

In future research, we plan to study the coupled Gaussian distribution as the prior and posterior distribution of the latent layer. This may be helpful for achieving greater separation between the images into distinct clusters similar to what has been achieved with t-stochastic neighborhood embedding methods [35]. If so, it may be possible to improve the decisiveness of the likelihoods in addition to further improvements in the accuracy and robustness. Since our approach generalizes the training of the decoder and encoder networks, it is expected to be seamlessly applicable to other datasets and neural network architectures. We are conducting research to apply our method to a convolutional neural network design that can process more complex datasets such as CIFAR-10. This first demonstration of the coupled ELBO cost function has provided experimental results applied to a shallow neural network but the approach is also applicable to the training of deep neural networks.

Author Contributions: S.C. implemented the coupled VAE algorithm programming structure, generated essential output data for analysis, and drafted and modified the paper. J.L. conducted statistical analysis of the results and derived the coupled ELBO based on the concept of nonlinear statistical coupling. K.P.N. originated the concept of nonlinear statistical coupling and mentored the team in applying the methods to the design of a variational autoencoder. M.A.K. provided oversight of the statistical analysis and the writing of the results, discussion, and conclusions. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The algorithm and data can be accessed at https://github.com/Photrek/Coupled-VAE-Improved-Robustness-and-Accuracy-of-a-Variational-Autoencoder, accessed on 5 February 2022.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Appendix A.1. Derivation of Negative Coupled ELBO

Generalizing the negative ELBO is accomplished using the principles of nonlinear statistical coupling (NSC) to generalize information theory. As described in Section 2.1, the negative ELBO consists of two components, the KL-divergence between the prior and posterior latent distribution, and the cross-entropy or negative log-likelihood of the reconstructed image in relation to the original image. NSC is an approach to modeling the statistics of complex systems that unifies heavy-tailed distributions, generalized information metrics, and fusion of information. Its application to the cost functions of a VAE provides control over the trade-off between decisive and robust generative models. Decisive refers to the characteristic of confident probabilities and robust refers to the characteristic of dampening extremes in the probabilities.

In the VAE algorithm, the loss function consists of the KL-divergence between the posterior approximation $q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)$ and a prior $p(\mathbf{z})$ and the cross-entropy between the reported probabilities and the training sample distribution.

$$\mathcal{L}\left(\mathbf{x}^{(i)}\right) = D_{KL}\left(q\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \parallel p(\mathbf{z})\right) - \frac{1}{L}\sum_{l=1}^{L}\left(\log p\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right)\right), \tag{A1}$$

where L is the number of reconstructions per test sample, and the KL-divergence is given by

$$D_{KL}\left(q\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \parallel p(\mathbf{z})\right) = \int q\left(\mathbf{z}|\mathbf{x}^{(i)}\right)\left(\log q\left(\mathbf{z}|\mathbf{x}^{(i)}\right) - \log p(\mathbf{z})\right)d\mathbf{z}. \tag{A2}$$

Even though $\mathbf{x}^{(\mathbf{i})}$ given $\mathbf{z}$ is a grayscaled value, which is not Bernoulli distributed, we can still use the probability mass function of Bernoulli distribution, then the cross entropy term is given by

$$-\frac{1}{L}\sum_{l=1}^{L}\left(\log p\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right)\right) = -\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_x}\left[x_i \log y_i + (1-x_i)\log(1-y_i)\right], \tag{A3}$$

where $\mathbf{y} = \text{Sigmod}(\mathbf{f_2}(\tanh(\mathbf{f_1}(\mathbf{z}))))$ while f_1 and f_2 are linear models and n_x is the dimensionality of $\mathbf{x}$.

The negative ELBO loss function is modified by coupled generalizations of the KL-divergence and cross-entropy. The purpose is to increase the weighting of rare events in the training dataset and thereby improve the robustness of the VAE model. The connection with the assessment metrics defined in Section 3.1 is that the power of the generalized mean can be decomposed into functions of the coupling and second parameter α, related to the power in the distribution of the random variable. For Gaussians and their generalizations, known as coupled Gaussians, $\alpha = 2$. Making use of $r(\kappa,\alpha,d) = \frac{-\alpha\kappa}{1+d\kappa}$ with $\alpha = 2$, the generalized mean is $\left(\sum p_i^{1+r}\right)^{\frac{1}{r}} = \left(\sum p_i^{1-\frac{2\kappa}{1+\kappa}}\right)^{-\frac{1+\kappa}{2\kappa}}$. When the coupling $\kappa \to 0$, the generalized mean is asymptotically equal to the geometric mean.

The coupled entropy function takes the form of a generalized logarithmic function applied to the generalized mean [22].

$$H_\kappa(\mathbf{p}) \equiv \frac{1}{2}\ln_\kappa\left(\left(\sum p_i^{1+\frac{2\kappa}{1+\kappa}}\right)^{\frac{-1}{\kappa}}\right) \equiv \frac{p_i^{1+\frac{2\kappa}{1+\kappa}}}{2\sum p_i^{\frac{1+2\kappa}{1+\kappa}}}\ln_\kappa p_i^{-\frac{2}{1+\kappa}} \equiv \frac{1}{2\kappa}\left(\left(\sum p_i^{\frac{1+3\kappa}{1+\kappa}}\right)^{-1} - 1\right), \tag{A4}$$

where $\ln_\kappa(x)$ is the generalization of the logarithm function in Equation (A15).

Similar to the generalization of coupled entropy function, the generalized logarithmic is applied to the KL-divergence. The first term in KL-divergence becomes

$$-\int q(\mathbf{z}|\mathbf{x}^{(i)})\log q(\mathbf{z}|\mathbf{x}^{(i)})d\mathbf{z} \Rightarrow \frac{1}{2}\prod_{j=1}^{n_z}\int\frac{q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}}}{\int q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}}dz_j}\ln_\kappa(q(z_j|\mathbf{x}^{(i)})^{-\frac{2}{1+\kappa}})dz_j, \tag{A5}$$

and the second term in KL-divergence becomes

$$-\int q(\mathbf{z}|\mathbf{x}^{(i)})\log p(\mathbf{z})d\mathbf{z} \Rightarrow \frac{1}{2}\prod_{j=1}^{n_z}\int\frac{q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}}}{\int q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}}dz_j}\ln_\kappa(p(z_j)^{-\frac{2}{1+\kappa}})dz_j, \tag{A6}$$

Therefore, the coupled divergence with n_z as the dimensionality of $\mathbf{z}$ can be written as

$$\begin{aligned}
&D_\kappa(q(\mathbf{z}|\mathbf{x}^{(i)}) \parallel p(\mathbf{z})) \\
&\equiv \prod_{j=1}^{n_z} \int \frac{q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}}}{\int q(z_j|\mathbf{x}^{(i)})^{1+\frac{2\kappa}{1+\kappa}} dz_j} \frac{1}{2} (\ln_\kappa(q(z_j|\mathbf{x}^{(i)})^{-\frac{2}{1+\kappa}}) - \ln_\kappa(p(z_j)^{-\frac{2}{1+\kappa}}))dz_j \\
&= \prod_{j=1}^{n_z} \frac{1}{2\kappa} \int \frac{(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z_j-\mu_i)^2}{2\sigma^2}})^{1+\frac{2\kappa}{1+\kappa}}}{\int (\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z_j-\mu_i)^2}{2\sigma^2}})^{1+\frac{2\kappa}{1+\kappa}} dz_j} \cdot ((\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z_j-\mu_i)^2}{2\sigma^2}})^{-\frac{2\kappa}{1+\kappa}} - (\frac{1}{\sqrt{2\pi}} e^{-\frac{z_j^2}{2}})^{-\frac{2\kappa}{1+\kappa}})dz_j
\end{aligned} \quad . \tag{A7}$$

The original cross-entropy can also be modified in a similar way. Applying the generalization of the logarithmic function, the terms $\log(y_i)$ and $\log(1-y_i)$ are modified to $\frac{1}{2}\ln_\kappa\left((y_i)^{\frac{2}{1+\kappa}}\right)$ and $\frac{1}{2}\ln_\kappa\left((1-y_i)^{\frac{2}{1+\kappa}}\right)$, thus

$$\log p\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right) \Rightarrow \sum_{i=1}^{n_x} \left(x_i \frac{1}{2}\ln_\kappa\left((y_i)^{\frac{2}{1+\kappa}}\right) + (1-x_i)\frac{1}{2}\ln_\kappa\left((1-y_i)^{\frac{2}{1+\kappa}}\right)\right). \tag{A8}$$

Therefore, the coupled cross-entropy is the generalization of the cross-entropy term in Equation (A14), which is defined as

$$H_\kappa(x_i, y_i) \equiv -\frac{1}{2L} \sum_{l=1}^{L} \sum_{i=1}^{n_x} \left(x_i \ln_\kappa\left((y_i)^{\frac{2}{1+\kappa}}\right) + (1-x_i)\ln_\kappa\left((1-y_i)^{\frac{2}{1+\kappa}}\right)\right). \tag{A9}$$

Adding Equations (A7) and (A9) gives the negative coupled ELBO,

$$\mathcal{L}_\kappa\left(\mathbf{x}^{(i)}\right) = D_\kappa\left(q\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \parallel p(\mathbf{z})\right) + H_\kappa(\mathbf{x}, \mathbf{y}), \tag{A10}$$

as defined in Equations (10)–(12).

Appendix A.2. Origin of the Generalized Probability Metrics

The generalized probability metrics derive from a translation of a generalized entropy function back to the probability domain. Use of the geometric mean for Accuracy derives from the Boltzmann–Gibbs–Shannon entropy, which measures the average uncertainty of a system and is equal to the arithmetic average of the negative logarithm of the probability distribution,

$$H(\mathbf{P}) \equiv -\sum_{i=1}^{N} p_i \ln p_i = -\ln\left(\prod_{i=1}^{N} p_i^{p_i}\right). \tag{A11}$$

Translating the entropy back to the probability domain via the inverse of the negative logarithm, which is the exponential of the negative, results in the weighted geometric mean of the probabilities

$$P_{avg} \equiv \exp(-H(\mathbf{P})) = \exp\left(\ln\left(\prod_{i=1}^{N} p_i^{p_i}\right)\right) = \prod_{i=1}^{N} p_i^{p_i}. \tag{A12}$$

The role of this function in defining the central tendency of the y-axis of a density is illustrated with the Gaussian distribution. Utilizing the continuous definition of entropy for a density $f(x)$ for a random variable x, the neutral accuracy or central tendency of the density is

$$f_{avg} \equiv \exp(-H(f(x))) = \exp\left(\int_X f(x) \ln f(x) dx \right). \tag{A13}$$

For the Gaussian, the average density is equal to the density at the mean plus the standard deviation $f(\mu \pm \sigma)$.

The use of the geometric mean as a metric for the neutral accuracy in the previous section is related to the cross-entropy between the reported probability of the algorithm and the probability distribution of the test set. The cross-entropy between a 'quoted' or predicted probability distribution $\mathbf{q}$ and the distribution of the test set $\mathbf{p}$ is

$$H(\mathbf{p}, \mathbf{q}) \equiv -\sum_i p_i \ln q_i. \tag{A14}$$

In evaluating an algorithm, the actual distribution is defined by the test samples which, for equally probable independent samples, each have a probability of $p_i = \frac{1}{N}$. Translated to the probability domain, the cross-entropy becomes the geometric mean of the reported probabilities (8), thus showing that use of the geometric mean of the probabilities as a measure of Accuracy for reported probabilities is equivalent to the use of cross-entropy as a metric of forecasting performance.

Likewise, the use of the generalized mean as a metric for Robustness and Decisiveness derives from a generalization of the cross-entropy. While there are a variety of proposed generalizations to information theory, in [22,36–38], the Renyi and Tsallis entropies were both shown to translate to a generalized mean upon transformation to the probability domain. Here, we show that the derivation of this transformation uses the coupled entropy, which derives from the Tsallis entropy, but utilizes a modified normalization. The nonlinear statistical coupling (or simply the coupling) has been shown to (a) quantify the relative variance of a superstatistics model in which the variance of exponential distribution fluctuates according to a gamma distribution, and (b) be equal to the inverse of the degree of freedom of the Student's t distribution. The coupling is related to the risk bias by the expression $r = \frac{-2\kappa}{1+\kappa}$, where the numeral 2 is associated with the power 2 of the Student's t distribution, and the ratio $r = \frac{-2\kappa}{1+\kappa}$ is associated with a duality between the positive and negative domains of the coupling. The coupled entropy uses a generalization of the logarithmic function,

$$\ln_\kappa(x) \equiv \frac{1}{\kappa}(x^\kappa - 1), \;\; x > 0, \tag{A15}$$

which provides a continuous set of functions with power. The coupled entropy aggregates the probabilities of a distribution using the generalized mean and translates this to the entropy domain using the generalized logarithm. Using the equiprobable for the sample probabilities, $p_i = \frac{1}{N}$, the coupled cross-entropy 'score' for the forecasted probabilities $\mathbf{q}$ for the event labels $\mathbf{e}$ is

$$S_\kappa(\mathbf{e}, \mathbf{q}) \equiv \frac{-2}{1+\kappa} \ln_{\left(\frac{-2\kappa}{1+\kappa}\right)} \left(\left(\frac{1}{N} \sum_{i=1}^{N} q_i^{\frac{-2\kappa}{1+\kappa}} \right)^{\frac{-1-\kappa}{2\kappa}} \right) \equiv \frac{1}{\kappa} \left(\left(\frac{1}{N} \sum_{i=1}^{N} q_i^{\frac{-2\kappa}{1+\kappa}} \right) - 1 \right), \tag{A16}$$

where q_i is the probability of event e_i which occurred. Thus, the coupled cross-entropy is a local scoring rule dependent only on the probabilities of the actual events.

References

1. Srivastava, A.; Sutton, C. Autoencoding Variational Inference for Topic Models. *arXiv* **2017**, arXiv:1703.01488
2. Dilokthanakul, N.; Mediano, P.A.M.; Garnelo, M.; Lee, M.C.H.; Salimbeni, H.; Arulkumaran, K.; Shanahan, M. Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders. *arXiv* **2016**, arXiv:1611.02648.
3. Akrami, H.; Joshi, A.A.; Li, J.; Aydore, S.; Leahy, R.M. Robust variational autoencoder. *arXiv* **2019**, arXiv:1905.09961.

4. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. In Proceedings of the International Conference on Learning Representations (ICLR), Banff, AB, Canada, 14–16 April 2014 .
5. Tran, D.; Hoffman, M.D.; Saurous, R.A.; Brevdo, E.; Murphy, K.; Blei, D.M. Deep probabilistic programming. In Proceedings of the Fifth International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
6. Bowman, S.R.; Vilnis, L.; Vinyals, O.; Dai, A.M.; Jozefowicz, R.; Bengio, S. Generating Sentences from a Continuous Space. In Proceedings of the Twentieth Conference on Computational Natural Language Learning (CoNLL), Beijing, China, 26–31 July 2015.
7. Zalger, J. *Application of Variational Autoencoders for Aircraft Turbomachinery Design*; Technical report; Stanford University: Stanford, CA, USA, 2017.
8. Xu, H.; Feng, Y.; Chen, J.; Wang, Z.; Qiao, H.; Chen, W.; Zhao, N.; Li, Z.; Bu, J.; Li, Z.; et al. Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications. In Proceedings of the 2018 World Wide Web Conference on World Wide Web, Lyon, France, 23–27 April 2018; ACM Press: New York, New York, USA, 2018; pp. 187–196.
9. Luchnikov, I.A.; Ryzhov, A.; Stas, P.J.; Filippov, S.N.; Ouerdane, H. Variational autoencoder reconstruction of complex many-body physics. *Entropy* **2019**, *21*, 1091. [CrossRef]
10. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D. Variational Inference: A Review for Statisticians. *J. Am. Stat. Assoc.* **2016**, *112*, 859–877. [CrossRef]
11. Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. In Proceedings of the ICLR, Toulon, France, 24–26 April 2017.
12. Burgess, C.P.; Higgins, I.; Pal, A.; Matthey, L.; Watters, N.; Desjardins, G.; Lerchner, A. Understanding disentangling in *beta*-VAE. *arXiv* **2018**, arXiv:1804.03599.
13. Niemitalo, O. A Method for Training Artificial Neural Networks to Generate Missing Data within a Variable Context. 2010. Internet Archive (Wayback Machine). Available online: https://web.archive.org/web/20120312111546/http://yehar.com/blog/?p=167 (accessed on 5 February 2022).
14. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Dutchess County, NY, USA, 2014; pp. 2672–2680.
15. Donahue, J.; Darrell, T.; Krähenbühl, P. Adversarial feature learning. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017—Conference Track Proceedings, International Conference on Learning Representations, ICLR, Toulon, France, 24–26 April 2017.
16. Dumoulin, V.; Belghazi, I.; Poole, B.; Mastropietro, O.; Lamb, A.; Arjovsky, M.; Courville, A. Adversarially learned inference. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017—Conference Track Proceedings, International Conference on Learning Representations, ICLR, Toulon, France, 24–26 April 2017.
17. Neyshabur, B.; Bhojanapalli, S.; Chakrabarti, A. Stabilizing GAN training with multiple random projections. *arXiv* **2017**, arXiv:1705.07831.
18. Pearl, J. *Bayesian Netwcrks: A Model cf Self-Activated Memory for Evidential Reasoning*; Technical Report; University of California: Oakland, CA, USA, 1985.
19. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
20. Ebbers, J.; Heymann, J.; Drude, L.; Glarner, T.; Haeb-Umbach, R.; Raj, B. Hidden Markov Model Variational Autoencoder for Acoustic Unit Discovery. In Proceedings of the INTERSPEECH 2017, Stockholm, Sweden, 20–24 August 2017; pp. 488–492.
21. Nelson, K.P.; Umarov, S. Nonlinear statistical coupling. *Phys. Stat. Mech. Its Appl.* **2010**, *389*, 2157–2163. [CrossRef]
22. Nelson, K.P.; Umarov, S.R.; Kon, M.A. On the average uncertainty for systems with nonlinear coupling. *Phys. Stat. Mech. Its Appl.* **2017**, *468*, 30–43. [CrossRef]
23. Nelson, K.P. Reduced Perplexity: A simplified perspective on assessing probabilistic forecasts. In *Info-Metrics Volume*; Chen, M.; Dunn, J.M.; Golan, A.; Ullah, A., Eds.; Oxford University Press: Oxford, UK, 2020.
24. Tsallis, C. *Introduction to Nonextensive Statistical Mechanics: Approaching a Complex World*; Springer: New York, NY, USA, 2009; pp. 1–382.
25. Weberszpil, J.; Helayël-Neto, J.A. Variational approach and deformed derivatives. *Phys. Stat. Mech. Its Appl.* **2016**, *450*, 217–227. [CrossRef]
26. Venkatesan, R.; Plastino, A. Generalized statistics variational perturbation approximation using q-deformed calculus. *Phys. Stat. Mech. Its Appl.* **2010**, *389*, 1159–1172. [CrossRef]
27. McAlister, D. XIII. The law of the geometric mean. *Proc. R. Soc.* **1879**, *29*, 367–376.
28. Nelson, K.P.; Scannell, B.J.; Landau, H. A risk profile for information fusion algorithms. *Entropy* **2011**, *13*, 1518–1532. [CrossRef]
29. Frogner, C.; Zhang, C.; Mobahi, H.; Araya, M.; Poggio, T.A. Learning with a Wasserstein Loss. In *Advances in Neural Information Processing Systems 28*; Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2015; pp. 2053–2061.
30. Vahdat, A.; Kautz, J. Nvae: A deep hierarchical variational autoencoder. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 19667–19679.
31. LeCun, Y.; Cortes, C.; Burges, C.J. The MNIST Database of Handwritten Digits. 1998. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 5 February 2022).
32. Chen, K.R.; Svoboda, D.; Nelson, K.P. Use of Student's t-Distribution for the Latent Layer in a Coupled Variational Autoencoder. *arXiv* **2020**, arXiv:2011.10879.

33. Takahashi, H.; Iwata, T.; Yamanaka, Y.; Yamada, M.; Yagi, S. Student-t Variational Autoencoder for Robust Density Estimation. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 2696–2702.
34. Mu, N.; Gilmer, J. Mnist-c: A robustness benchmark for computer vision. *arXiv* **2019**, arXiv:1906.02337.
35. Van Der Maaten, L.; Hinton, G. Visualizing Data using T-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
36. Thurner, S.; Corominas-Murtra, B.; Hanel, R. Three faces of entropy for complex systems: Information, thermodynamics, and the maximum entropy principle. *Phys. Rev. E* **2017**, *96*, 032124. [CrossRef] [PubMed]
37. Abe, S. Stability of Tsallis entropy and instabilities of Rényi and normalized Tsallis entropies: A basis for q-exponential distributions. *Phys. Rev. E* **2002**, *66*, 046134. [CrossRef] [PubMed]
38. Rényi, A. On the foundations of information theory. *Rev. L'Inst. Int. Stat.* **1965**, 33, 1–14. [CrossRef]

entropy

Article

Sequential Learning of Principal Curves: Summarizing Data Streams on the Fly

Le Li [1] and Benjamin Guedj [2,*]

1 Department of Statistics, Central China Normal University, Wuhan 430079, China; leli@mail.ccnu.edu.cn
2 Inria, Lille-Nord Europe Research Centre and Inria London, France and Centre for Artificial Intelligence, Department of Computer Science, University College London, London WC1V 6LJ, UK
* Correspondence: b.guedj@ucl.ac.uk

Abstract: When confronted with massive data streams, summarizing data with dimension reduction methods such as PCA raises theoretical and algorithmic pitfalls. A principal curve acts as a nonlinear generalization of PCA, and the present paper proposes a novel algorithm to automatically and sequentially learn principal curves from data streams. We show that our procedure is supported by regret bounds with optimal sublinear remainder terms. A greedy local search implementation (called `slpc`, for sequential learning principal curves) that incorporates both sleeping experts and multi-armed bandit ingredients is presented, along with its regret computation and performance on synthetic and real-life data.

Keywords: sequential learning; principal curves; data streams; regret bounds; greedy algorithm; sleeping experts

Citation: Li, L.; Guedj, B. Sequential Learning of Principal Curves: Summarizing Data Streams on the Fly. *Entropy* **2021**, *23*, 1534. https://doi.org/10.3390/e23111534

Academic Editor: Mohamed Medhat Gaber

Received: 22 August 2021
Accepted: 1 November 2021
Published: 18 November 2021

1. Introduction

Numerous methods have been proposed in the statistics and machine learning literature to sum up information and represent data by condensed and simpler-to-understand quantities. Among those methods, principal component analysis (PCA) aims at identifying the maximal variance axes of data. This serves as a way to represent data in a more compact fashion and hopefully reveal as well as possible their variability. PCA was introduced by [1,2] and further developed by [3]. This is one of the most widely used procedures in multivariate exploratory analysis targeting dimension reduction or feature extraction. Nonetheless, PCA is a linear procedure and the need for more sophisticated nonlinear techniques has led to the notion of principal curve. Principal curves may be seen as a nonlinear generalization of the first principal component. The goal is to obtain a curve which passes "in the middle" of data, as illustrated by Figure 1. This notion of skeletonization of data clouds has been at the heart of numerous applications in many different domains, such as physics [4,5], character and speech recognition [6,7], mapping and geology [5,8,9], to name but a few.

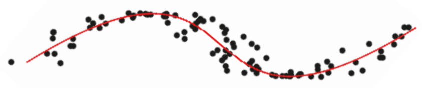


Figure 1. A principal curve.

1.1. Earlier Works on Principal Curves

The original definition of principal curve dates back to [10]. A principal curve is a smooth (C^∞) parameterized curve $\mathbf{f}(s) = (f_1(s), \ldots, f_d(s))$ in $\mathbb{R}^d$ which does not intersect itself, has finite length inside any bounded subset of $\mathbb{R}^d$ and is self-consistent. This last requirement means that $\mathbf{f}(s) = \mathbb{E}[X|s_{\mathbf{f}}(X) = s]$, where $X \in \mathbb{R}^d$ is a random vector and the so-called projection index $s_{\mathbf{f}}(x)$ is the largest real number s minimizing the squared Euclidean distance between $\mathbf{f}(s)$ and x, defined by

$$s_{\mathbf{f}}(x) = \sup\left\{s : \|x - \mathbf{f}(s)\|_2^2 = \inf_{\tau} \|x - \mathbf{f}(\tau)\|_2^2\right\}.$$

Self-consistency means that each point of $\mathbf{f}$ is the average (under the distribution of X) of all data points projected on $\mathbf{f}$, as illustrated by Figure 2.

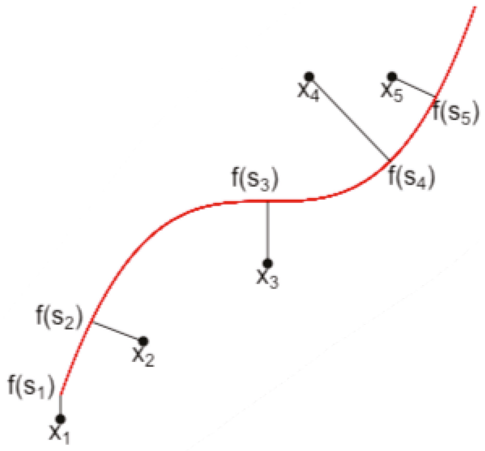


Figure 2. A principal curve and projections of data onto it.

However, an unfortunate consequence of this definition is that the existence is not guaranteed in general for a particular distribution, let alone for an online sequence for which no probabilistic assumption is made. In order to handle complex data structures, Ref. [11] proposed principal curves (PCOP) of principal oriented points (POPs) which are defined as the fixed points of an expectation function of points projected to a hyperplane minimising the total variance. To obtain POPs, a cluster analysis is performed on the hyperplane and only data in the local cluster are considered. Ref. [12] introduced the local principal curve (LPC), whose concept is similar to that of [11], but accelerates the computation of POPs by calculating local centers of mass instead of performing cluster analysis, and local principal component instead of principal direction. Later, Ref. [13] also considered LPC in data compression and regression to reduce the dimension of predictors space to low-dimension manifold. Ref. [14] extended the idea of localization to independent component analysis (ICA) by proposing a local-to-global non-linear ICA framework for visual and auditory signal. Ref. [15] considered principal curves from a different perspective: as the ridge of a smooth probability density function (PDF) generating dataset, where the ridges are collections of all points; the local gradient of a PDF is an eigenvector of the local Hessian, and the eigenvalues corresponding to the remaining eigenvectors are negative. To estimate principal curves based on this definition, the subspace constrained mean shift (SCMS) algorithm was proposed. All the local methods above require strong assumptions on the PDF, such as twice continuous differentiability, which may prove challenging to be satisfied in the settings of online sequential data. Ref. [16] proposed a new concept of principal curves which ensures its existence for a large class of distributions. Principal curves $\mathbf{f}^\star$ are defined as the curves minimizing the expected squared distance over a class $\mathcal{F}_L$ of curves whose length is smaller than $L > 0$; namely,

$$\mathbf{f}^\star \in \arg\inf_{\mathbf{f}\in\mathcal{F}_L} \Delta(\mathbf{f}),$$

where

$$\Delta(\mathbf{f}) = \mathbb{E}[\Delta(\mathbf{f}, X)] = \mathbb{E}\Big[\inf_s \|\mathbf{f}(s) - X\|_2^2\Big].$$

If $\mathbb{E}\|X\|_2^2 < \infty$, $\mathbf{f}^\star$ always exists but may not be unique. In practical situations where only i.i.d. copies $X_1, \ldots, X_n$ of X are observed, the method of [16] considers classes $\mathcal{F}_{k,L}$ of all polygonal lines with k segments and length not exceeding L, and chooses an estimator $\hat{\mathbf{f}}_{k,n}$ of $\mathbf{f}^\star$ as the one within $\mathcal{F}_{k,L}$, which minimizes the empirical counterpart

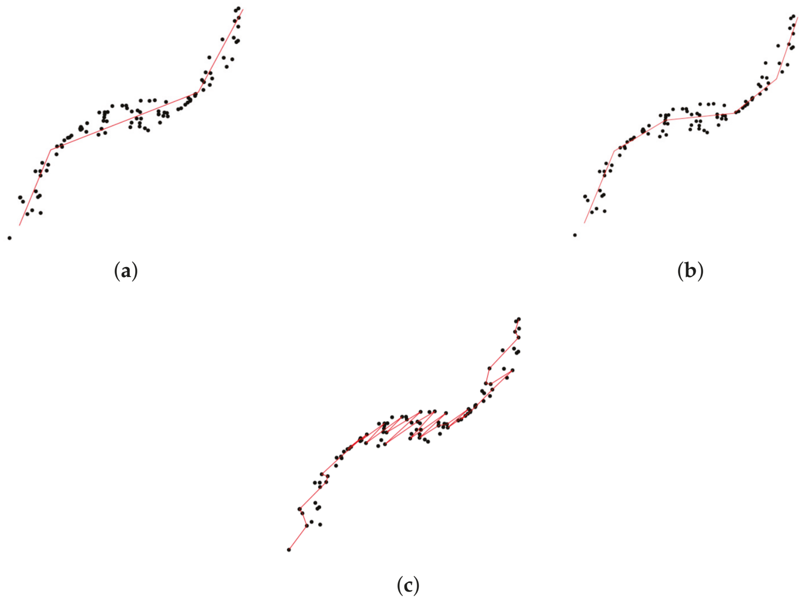
$$\Delta_n(\mathbf{f}) = \frac{1}{n}\sum_{i=1}^{n} \Delta(\mathbf{f}, X_i)$$

of $\Delta(\mathbf{f})$. It is proved in [17] that if X is almost surely bounded and $k \propto n^{1/3}$, then

$$\Delta\Big(\hat{\mathbf{f}}_{k,n}\Big) - \Delta(\mathbf{f}^\star) = \mathcal{O}\Big(n^{-1/3}\Big).$$

As the task of finding a polygonal line with k segments and length of at most L that minimizes $\Delta_n(\mathbf{f})$ is computationally costly, Ref. [17] proposed a polygonal line algorithm. This iterative algorithm proceeds by fitting a polygonal line with k segments and considerably speeds up the exploration part by resorting to gradient descent. The two steps (projection and optimization) are similar to what is done by the k-means algorithm. However, the polygonal line algorithm is not supported by theoretical bounds and leads to variable performance depending on the distribution of the observations.

As the number of segments, k, plays a crucial role (a too small a k value leads to a poor summary of data, whereas a too-large k yields overfitting; see Figure 3), Ref. [18] aimed to fill the gap by selecting an optimal k from both theoretical and practical perspectives.



(**a**)

(**b**)

(**c**)

Figure 3. Principal curves with different numbers (k) of segments. (**a**) A too small k. (**b**) Right k. (**c**) A too large k.

Their approach relies strongly on the theory of model selection by penalization introduced by [19] and further developed by [20]. By considering countable classes $\{\mathcal{F}_{k,\ell}\}_{k,\ell}$ of polygonal lines with k segments and total length $\ell \leq L$, and whose vertices are on a lattice, the optimal $(\hat{k}, \hat{\ell})$ is obtained as the minimizer of the criterion

$$\operatorname{crit}(k, \ell) = \Delta_n\left(\hat{\mathbf{f}}_{k,\ell}\right) + \operatorname{pen}(k, \ell),$$

where

$$\operatorname{pen}(k, \ell) = c_0\sqrt{\frac{k}{n}} + c_1\frac{\ell}{n} + c_2\frac{1}{\sqrt{n}} + \delta^2\sqrt{\frac{w_{k,\ell}}{2n}}$$

is a penalty function where δ stands for the diameter of observations and $w_{k,\ell}$ denotes the weight attached to class $\mathcal{F}_{k,\ell}$; and it has constants c_0, c_1, c_2 depending on δ, maximum length L and a certain number of dimensions of observations. Ref. [18] then proved that

$$\mathbb{E}\left[\Delta(\hat{\mathbf{f}}_{\hat{k},\hat{\ell}}) - \Delta(\mathbf{f}^\star)\right] \leq \inf_{k,\ell}\left\{\mathbb{E}\left[\Delta(\hat{\mathbf{f}}_{k,\ell}) - \Delta(\mathbf{f}^\star)\right] + \operatorname{pen}(k, \ell)\right\} + \frac{\delta^2\Sigma}{2^{3/2}}\sqrt{\frac{\pi}{n}}, \tag{1}$$

where Σ is a numerical constant. The expected loss of the final polygonal line $\hat{\mathbf{f}}_{\hat{k},\hat{\ell}}$ is close to the minimal loss achievable over $\mathcal{F}_{k,\ell}$ up to a remainder term decaying as $1/\sqrt{n}$.

1.2. Motivation

The big data paradigm—where collecting, storing and analyzing massive amounts of large and complex data becomes the new standard—commands one to revisit some of the classical statistical and machine learning techniques. The tremendous improvements of data acquisition infrastructures generates new continuous streams of data, rather than batch datasets. This has drawn great interest to sequential learning. Extending the notion of principal curves to the sequential settings opens up immediate practical application possibilities. As an example, path planning for passengers' locations can help taxi companies to better optimize their fleet. Online algorithms that could yield instantaneous path summarization would be adapted to the sequential nature of geolocalized data. Existing theoretical works and practical implementations of principal curves are designed for the batch setting [7,16–18,21] and their adaptation to the sequential setting is not a smooth process. As an example, consider the algorithm in [18]. It is assumed that vertices of principal curves are located on a lattice, and its computational complexity is of order $\mathcal{O}(nN^p)$ where n is the number of observations, N the number of points on the lattice and p the maximum number of vertices. When p is large, running this algorithm at each epoch yields a monumental computational cost. In general, if data are not identically distributed or even adversary, algorithms that originally worked well in the batch setting may not be ideal when cast onto the online setting (see [22], Chapter 4). To the best of our knowledge, little effort has been put so far into extending principal curves algorithms to the sequential context.

Ref. [23] provided an incremental version of the SCMS algorithm [15] which is based on a definition of a principal curve as the ridge of a smooth probability density function generating observations. They applied the SCMS algorithm to the input points that are associated with the output points which are close to the new incoming sample and leave the remaining outputs unchanged. Hence, this algorithm can be used to deal with sequential data. As presented in the next section, our algorithm for sequentially updating principal curve vertices that are close to new data is similar in spirit to that of incremental SCMS. However, a difference is that our algorithm outputs polygonal lines. In addition, the computation complexity of our method is $\mathcal{O}(n^2)$, and incremental SCMS has $\mathcal{O}(n^3)$ complexity, where n is the number of observations. Ref. [24] considered sequential principal curves analysis in a fairly different setting in which the goal was to derive in an adaptive fashion a set of nonlinear sensors by using a set of preliminary principal curves. Unfolding sequentially principal curves and a sequential path for Jacobian integration were considered. The "sequential" in this setting represented the generalization of principal curves to principal surfaces or even a principal manifold of higher dimensions. This way of sequentially exploiting principal curves was firstly proposed by [11] and later extended by [14,25,26] to give curvilinear representations using sequence of local-to-global curves. In addition, Refs. [15,27,28] presented, respectively, principal polynomial and non-parametric regressions to capture the nonlinear nature of data. However, these methods are not originally designed for treating sequential data. The present paper aims at filling this gap: our goal was to propose an online perspective to principal curves by automatically and sequentially learning the best principal curve summarizing a data stream. Sequential learning takes advantage of the latest collected (set of) observations and therefore suffers a much smaller computational cost.

Sequential learning operates as follows: a blackbox reveals at each time t some deterministic value $x_t, t = 1, 2, \ldots$, and a forecaster attempts to predict sequentially the next value based on past observations (and possibly other available information). The performance of the forecaster is no longer evaluated by its generalization error (as in the batch setting) but rather by a regret bound which quantifies the cumulative loss of a forecaster in the first T rounds with respect to some reference minimal loss. In sequential learning, the velocity of algorithms may be favored over statistical precision. An immediate use of aforecited techniques [17,18,21] at each time round t (treating data collected until t as a batch dataset) would result in a monumental algorithmic cost. Rather, we propose a novel algorithm which adapts to the sequential nature of data, i.e., which takes advantage of previous computations.

The contributions of the present paper are twofold. We first propose a sequential principal curve algorithm, for which we derived regret bounds. We then present an implementation, illustrated on a toy dataset and a real-life dataset (seismic data). The sketch of our algorithm's procedure is as follows. At each time round t, the number of segments of k_t is chosen automatically and the number of segments k_{t+1} in the next round is obtained by only using information about k_t and a small number of past observations. The core of our procedure relies on computing a quantity which is linked to the mode of the so-called Gibbs quasi-posterior and is inspired by quasi-Bayesian learning. The use of quasi-Bayesian estimators is especially advocated by the PAC-Bayesian theory, which originated in the machine learning community in the late 1990s, in the seminal works of [29] and McAllester [30,31]. The PAC-Bayesian theory has been successfully adapted to sequential learning problems; see, for example, Ref. [32] for online clustering. We refer to [33,34] for a recent overview of the field.

The paper is organized as follows. Section 2 presents our notation and our online principal curve algorithm, for which we provide regret bounds with sublinear remainder terms in Section 3. A practical implementation was proposed in Section 4, and we illustrate its performance on synthetic and real-life datasets in Section 5. Proofs of all original results claimed in the paper are collected in Section 6.

2. Notation

A parameterized curve in $\mathbb{R}^d$ is a continuous function $\mathbf{f} : I \longrightarrow \mathbb{R}^d$ where $I = [a, b]$ is a closed interval of the real line. The length of $\mathbf{f}$ is given by

$$\mathcal{L}(\mathbf{f}) = \lim_{M \to \infty} \left\{ \sup_{a=s_0<s_1<\cdots<s_M=b} \sum_{i=1}^{M} \|\mathbf{f}(s_i) - \mathbf{f}(s_{i-1})\|_2 \right\}.$$

Let $x_1, x_2, \ldots, x_T \in B(0, \sqrt{d}R) \subset \mathbb{R}^d$ be a sequence of data, where $B(\mathbf{c}, R)$ stands for the ℓ_2-ball centered in $\mathbf{c} \in \mathbb{R}^d$ with radius $R > 0$. Let $\mathcal{Q}_\delta$ be a grid over $B(0, \sqrt{d}R)$, i.e., $\mathcal{Q}_\delta = B(0, \sqrt{d}R) \cap \Gamma_\delta$ where Γ_δ is a lattice in $\mathbb{R}^d$ with spacing $\delta > 0$. Let $L > 0$ and define for each $k \in [\![1, p]\!]$ the collection $\mathcal{F}_{k,L}$ of polygonal lines $\mathbf{f}$ with k segments whose vertices are in $\mathcal{Q}_\delta$ and such that $\mathcal{L}(\mathbf{f}) \leq L$. Denote by $\mathcal{F}_p = \cup_{k=1}^{p} \mathcal{F}_{k,L}$ all polygonal lines with a number of segments $\leq p$, whose vertices are in $\mathcal{Q}_\delta$ and whose length is at most L. Finally, let $\mathcal{K}(\mathbf{f})$ denote the number of segments of $\mathbf{f} \in \mathcal{F}_p$. This strategy is illustrated by Figure 4.

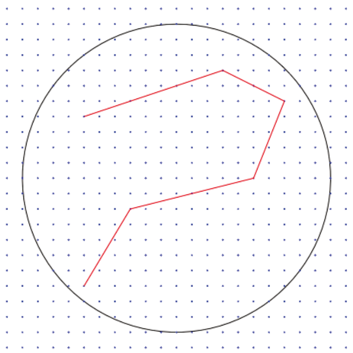


Figure 4. An example of a lattice Γ_δ in $\mathbb{R}^2$ with $\delta = 1$ (spacing between blue points) and $B(0, 10)$ (black circle). The red polygonal line is composed of vertices in $\mathcal{Q}_\delta = B(0, 10) \cap \Gamma_\delta$.

Our goal is to learn a time-dependent polygonal line which passes through the "middle" of data and gives a summary of all available observations $x_1, \dots, x_{t-1}$ (denoted by $(x_s)_{1:(t-1)}$ hereafter) before time t. Our output at time t is a polygonal line $\hat{\mathbf{f}}_t \in \mathcal{F}_p$ depending on past information $(x_s)_{1:(t-1)}$ and past predictions $(\hat{\mathbf{f}}_s)_{1:(t-1)}$. When x_t is revealed, the instantaneous loss at time t is computed as

$$\Delta\left(\hat{\mathbf{f}}_t, x_t\right) = \inf_{s \in I} \|\hat{\mathbf{f}}_t(s) - x_t\|_2^2. \tag{2}$$

In what follows, we investigate regret bounds for the cumulative loss based on (2). Given a measurable space Θ (embedded with its Borel σ-algebra), we let $\mathcal{P}(\Theta)$ denote the set of probability distributions on Θ, and for some reference measure π, we let $\mathcal{P}_\pi(\Theta)$ be the set of probability distributions absolutely continuous with respect to π.

For any $k \in [\![1, p]\!]$, let π_k denote a probability distribution on $\mathcal{F}_{k,L}$. We define the *prior* π on $\mathcal{F}_p = \cup_{k=1}^p \mathcal{F}_{k,L}$ as

$$\pi(\mathbf{f}) = \sum_{k \in [\![1,p]\!]} w_k \pi_k(\mathbf{f}) \mathbb{1}_{\{\mathbf{f} \in \mathcal{F}_{k,L}\}}, \quad \mathbf{f} \in \mathcal{F}_p,$$

where $w_1, \dots, w_p \geq 0$ and $\sum_{k \in [\![1,p]\!]} w_k = 1$.

We adopt a quasi-Bayesian-flavored procedure: consider the Gibbs quasi-posterior (note that this is not a proper posterior in all generality, hence the term "quasi"):

$$\hat{\rho}_t(\cdot) \propto \exp(-\lambda S_t(\cdot))\pi(\cdot),$$

where

$$S_t(\mathbf{f}) = S_{t-1}(\mathbf{f}) + \Delta(\mathbf{f}, x_t) + \frac{\lambda}{2}\left(\Delta(\mathbf{f}, x_t) - \Delta(\hat{\mathbf{f}}_t, x_t)\right)^2,$$

as advocated by [32,35] who then considered realizations from this quasi-posterior. In the present paper, we will rather focus on a quantity linked to the mode of this quasi-posterior. Indeed, the mode of the quasi-posterior $\hat{\rho}_{t+1}$ is

$$\arg\min_{\mathbf{f} \in \mathcal{F}_p} \left\{ \underbrace{\sum_{s=1}^t \Delta(\mathbf{f}, x_s)}_{(i)} + \underbrace{\frac{\lambda}{2} \sum_{s=1}^t \left(\Delta(\mathbf{f}, x_t) - \Delta(\hat{\mathbf{f}}_t, x_t)\right)^2}_{(ii)} + \underbrace{\frac{\ln \pi(\mathbf{f})}{\lambda}}_{(iii)} \right\},$$

where *(i)* is a cumulative loss term, *(ii)* is a term controlling the variance of the prediction $\mathbf{f}$ to past predictions $\hat{\mathbf{f}}_s, s \leq t$, and *(iii)* can be regarded as a penalty function on the complexity of $\mathbf{f}$ if π is well chosen. This mode hence has a similar flavor to follow the best expert or follow the perturbed leader in the setting of prediction with experts (see [22,36], Chapters 3 and 4) if we consider each $\mathbf{f} \in \mathcal{F}_p$ as an expert which always delivers constant advice. These remarks yield Algorithm 1.

Algorithm 1 Sequentially learning principal curves.

```
1: Input parameters: $p > 0, \eta > 0, \pi(z) = \mathrm{e}^{-z}\mathbb{1}_{\{z>0\}}$ and penalty function $h : \mathcal{F}_p \to \mathbb{R}^+$
2: Initialization: For each $\mathbf{f} \in \mathcal{F}_p$, draw $z_{\mathbf{f}} \sim \pi$ and $\Delta_{\mathbf{f},0} = \frac{1}{\eta}(h(\mathbf{f}) - z_{\mathbf{f}})$
3: For $t = 1, \ldots, T$
4:     Get the data $x_t$
5:     Obtain
           $$\hat{\mathbf{f}}_t = \arg\inf_{\mathbf{f}\in\mathcal{F}_p} \left\{ \sum_{s=0}^{t-1} \Delta_{\mathbf{f},s} \right\},$$
       where $\Delta_{\mathbf{f},s} = \Delta(\mathbf{f}, x_s)$, $s \geq 1$.
6: End for
```

3. Regret Bounds for Sequential Learning of Principal Curves

We now present our main theoretical results.

Theorem 1. *For any sequence $(x_t)_{1:T} \in B(0, \sqrt{d}R)$, $R \geq 0$ and any penalty function $h : \mathcal{F}_p \to \mathbb{R}^+$, let $\pi(z) = \mathrm{e}^{-z}\mathbb{1}_{\{z>0\}}$. Let $0 < \eta \leq \frac{1}{d(2R+\delta)^2}$; then the procedure described in Algorithm 1 satisfies*

$$\sum_{t=1}^{T} \mathbb{E}_\pi\left[\Delta(\hat{\mathbf{f}}_t, x_t)\right] \leq (1 + c_0(\mathrm{e}-1)\eta) S_{T,h,\eta} + \frac{1}{\eta}\left(1 + \ln \sum_{\mathbf{f}\in\mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})}\right),$$

where $c_0 = d(2R+\delta)^2$ and

$$S_{T,h,\eta} = \inf_{k\in[\![1,p]\!]} \left\{ \inf_{\substack{\mathbf{f}\in\mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{h(\mathbf{f})}{\eta} \right\} \right\}.$$

The expectation of the cumulative loss of polygonal lines $\hat{\mathbf{f}}_1, \ldots, \hat{\mathbf{f}}_T$ is upper-bounded by the smallest penalized cumulative loss over all $k \in \{1, \ldots, p\}$ up to a multiplicative term $(1 + c_0(\mathrm{e}-1)\eta)$, which can be made arbitrarily close to 1 by choosing a small enough η. However, this will lead to both a large $h(\mathbf{f})/\eta$ in $S_{T,h,\eta}$ and a large $\frac{1}{\eta}(1 + \ln \sum_{\mathbf{f}\in\mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})})$. In addition, another important issue is the choice of the penalty function h. For each $\mathbf{f} \in \mathcal{F}_p$, $h(\mathbf{f})$ should be large enough to ensure a small $\sum_{\mathbf{f}\in\mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})}$, but not too large to avoid overpenalization and a larger value for $S_{T,h,\eta}$. We therefore set

$$h(\mathbf{f}) \geq \ln(p\mathrm{e}) + \ln\left|\{\mathbf{f} \in \mathcal{F}_p, \mathcal{K}(\mathbf{f}) = k\}\right| \tag{3}$$

for each $\mathbf{f}$ with k segments (where $|M|$ denotes the cardinality of a set M) since it leads to

$$\sum_{\mathbf{f}\in\mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})}) = \sum_{k\in[\![1,p]\!]} \sum_{\substack{\mathbf{f}\in\mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}} \mathrm{e}^{-h(\mathbf{f})} \leq \sum_{k\in[\![1,p]\!]} \frac{1}{p\mathrm{e}} \leq \frac{1}{\mathrm{e}}.$$

The penalty function $h(\mathbf{f}) = c_1\mathcal{K}(\mathbf{f}) + c_2L + c_3$ satisfies (3), where c_1, c_2, c_3 are constants depending on R, d, δ, p (this is proven in Lemma 3, in Section 6). We therefore obtain the following corollary.

Corollary 1. *Under the assumptions of Theorem 1, let*

$$\eta = \min\left\{\frac{1}{d(2R+\delta)^2}, \sqrt{\frac{c_1p + c_2L + c_3}{c_0(\mathrm{e}-1)\inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^T \Delta(\mathbf{f}, x_t)}}\right\}.$$

Then

$$\sum_{t=1}^T \mathbb{E}\left[\Delta(\hat{\mathbf{f}}_t, x_t)\right] \leq \inf_{k\in[\![1,p]\!]}\left\{\inf_{\substack{\mathbf{f}\in\mathcal{F}_p\\ \mathcal{K}(\mathbf{f})=k}}\left\{\sum_{t=1}^T \Delta(\mathbf{f}, x_t) + \sqrt{c_0(\mathrm{e}-1)r_{T,k,L}}\right\}\right\}$$
$$+ \sqrt{c_0(\mathrm{e}-1)r_{T,p,L}} + c_0(\mathrm{e}-1)(c_1p + c_2L + c_3),$$

where $r_{T,k,L} = \inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^T \Delta(\mathbf{f}, x_t)(c_1k + c_2L + c_3)$.

Proof. Note that

$$\sum_{t=1}^T \mathbb{E}\left[\Delta(\hat{\mathbf{f}}_t, x_t)\right] \leq S_{T,h,\eta} + \eta c_0(\mathrm{e}-1)\inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^T \Delta(\mathbf{f}, x_t) + c_0(\mathrm{e}-1)(c_0p + c_2L + c_3),$$

and we conclude by setting

$$\eta = \sqrt{\frac{c_1p + c_2L + c_3}{c_0(\mathrm{e}-1)\inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^T \Delta(\mathbf{f}, x_t)}}.$$

□

Sadly, Corollary 1 is not of much practical use since the optimal value for η depends on $\inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^T \Delta(\mathbf{f}, x_t)$ which is obviously unknown, even more so at time $t = 0$. We therefore provide an adaptive refinement of Algorithm 1 in the following Algorithm 2.

Algorithm 2 Sequentially and adaptively learning principal curves.

```
1: Input parameters: $p > 0$, $L > 0$, $\pi$, $h$ and $\eta_0 = \frac{\sqrt{c_1p+c_2L+c_3}}{c_0\sqrt{\mathrm{e}-1}}$
2: Initialization: For each $\mathbf{f} \in \mathcal{F}_p$, draw $z_{\mathbf{f}} \sim \pi$, $\Delta_{\mathbf{f},0} = \frac{1}{\eta_0}(h(\mathbf{f}) - z_{\mathbf{f}})$ and $\hat{\mathbf{f}}_0 = \arg\inf_{\mathbf{f}\in\mathcal{F}_p} \Delta_{\mathbf{f},0}$
3: For $t = 1, \ldots, T$
4:     Compute $\eta_t = \frac{\sqrt{c_1p+c_2L+c_3}}{c_0\sqrt{(\mathrm{e}-1)t}}$
5:     Get data $x_t$ and compute $\Delta_{\mathbf{f},t} = \Delta(\mathbf{f}, x_t) + \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)(h(\mathbf{f}) - z_{\mathbf{f}})$
6:     Obtain
           $\hat{\mathbf{f}}_t = \arg\inf_{\mathbf{f}\in\mathcal{F}_p}\left\{\sum_{s=0}^{t-1}\Delta_{\mathbf{f},s}\right\}.$     (4)
7: End for
```

Theorem 2. *For any sequence* $(x_t)_{1:T} \in B(0, \sqrt{d}R)$, $R \geq 0$, *let* $h(\mathbf{f}) = c_1\mathcal{K}(\mathbf{f}) + c_2L + c_3$ *where* c_1, c_2, c_3 *are constants depending on* $R, d, \delta, \ln p$. *Let* $\pi(z) = \mathrm{e}^{-z}\mathbb{1}_{\{z>0\}}$ *and*

$$\eta_0 = \frac{\sqrt{c_1p + c_2L + c_3}}{c_0\sqrt{\mathrm{e}-1}}, \quad \eta_t = \frac{\sqrt{c_1p + c_2L + c_3}}{c_0\sqrt{(\mathrm{e}-1)t}},$$

where $t \geq 1$ and $c_0 = d(2R+\delta)^2$. Then the procedure described in Algorithm 2 satisfies

$$\sum_{t=1}^{T} \mathbb{E}\Big[\Delta(\hat{\mathbf{f}}_t, x_t)\Big] \leq \inf_{k \in [\![1,p]\!]} \left\{ \inf_{\substack{\mathbf{f} \in \mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + c_0 \sqrt{(\mathrm{e}-1)T(c_1 k + c_2 L + c_3)} \right\} \right\}$$
$$+ 2c_0 \sqrt{(\mathrm{e}-1)T(c_1 p + c_2 L + c_3)}.$$

The message of this regret bound is that the expected cumulative loss of polygonal lines $\hat{\mathbf{f}}_1, \dots, \hat{\mathbf{f}}_T$ is upper-bounded by the minimal cumulative loss over all $k \in \{1, \dots, p\}$, up to an additive term which is sublinear in T. The actual magnitude of this remainder term is $\sqrt{kT}$. When L is fixed, the number k of segments is a measure of complexity of the retained polygonal line. This bound therefore yields the same magnitude as (1), which is the most refined bound in the literature so far ([18] where the optimal values for k and L were obtained in a model selection fashion).

4. Implementation

The argument of the infimum in Algorithm 2 is taken over $\mathcal{F}_p = \cup_{k=1}^{p} \mathcal{F}_{k,L}$ which has a cardinality of order $|\mathcal{Q}_\delta|^p$, making any greedy search largely time-consuming. We instead turn to the following strategy: Given a polygonal line $\hat{\mathbf{f}}_t \in \mathcal{F}_{k_t,L}$ with k_t segments, we consider, with a certain proportion, the availability of $\hat{\mathbf{f}}_{t+1}$ within a neighborhood $\mathcal{U}(\hat{\mathbf{f}}_t)$ (see the formal definition below) of $\hat{\mathbf{f}}_t$. This consideration is well suited for the principal curves setting, since if observation x_t is close to $\hat{\mathbf{f}}_t$, one can expect that the polygonal line which well fits observations $x_s, s = 1, \dots, t$ lies in a neighborhood of $\hat{\mathbf{f}}_t$. In addition, if each polygonal line $\mathbf{f}$ is regarded as an action, we no longer assume that all actions are available at all times, and allow the set of available actions to vary at each time. This is a model known as "sleeping experts (or actions)" in prior work [37,38]. In this setting, defining the regret with respect to the best action in the whole set of actions in hindsight remains difficult, since that action might sometimes be unavailable. Hence, it is natural to define the regret with respect to the best ranking of all actions in the hindsight according to their losses or rewards, and at each round one chooses among the available actions by selecting the one which ranks the highest. Ref. [38] introduced this notion of regret and studied both the full-information (best action) and partial-information (multi-armed bandit) settings with stochastic and adversarial rewards and adversarial action availability. They pointed out that the **EXP4** algorithm [37] attains the optimal regret in the adversarial rewards case but has a runtime exponential in the number of all actions. Ref. [39] considered full and partial information with stochastic action availability and proposed an algorithm that runs in polynomial time. In what follows, we materialize our implementation by resorting to "sleeping experts", i.e., a special set of available actions that adapts to the setting of principal curves.

Let σ denote an ordering of $|\mathcal{F}_p|$ actions, and $\mathcal{A}_t$ a subset of the available actions at round t. We let $\sigma(\mathcal{A}_t)$ denote the highest ranked action in $\mathcal{A}_t$. In addition, for any action $\mathbf{f} \in \mathcal{F}_p$ we define the reward $r_{\mathbf{f},t}$ of $\mathbf{f}$ at round $t, t \geq 0$ by

$$r_{\mathbf{f},t} = c_0 - \Delta(\mathbf{f}, x_t).$$

It is clear that $r_{\mathbf{f},t} \in (0, c_0)$. The convention from losses to gains is done in order to facilitate the subsequent performance analysis. The reward of an ordering σ is the cumulative reward of the selected action at each time:

$$\sum_{t=1}^{T} r_{\sigma(\mathcal{A}_t),t},$$

and the reward of the best ordering is $\max_\sigma \sum_{t=0}^{T} r_{\sigma(\mathcal{A}_t),t}$ (respectively, $\mathbb{E}\Big[\max_\sigma \sum_{t=1}^{T} r_{\sigma(\mathcal{A}_t),t}\Big]$ when $\mathcal{A}_t$ is stochastic).

Our procedure starts with a **partition** step which aims at identifying the "relevant" neighborhood of an observation $x \in \mathbb{R}^d$ with respect to a given polygonal line, and then proceeds with the definition of the **neighborhood** of an action $\mathbf{f}$. We then provide the full implementation and prove a regret bound.

Partition. For any polygonal line $\mathbf{f}$ with k segments, we denote by $\overrightarrow{\mathbf{V}} = (v_1, \ldots, v_{k+1})$ its vertices and by $s_i, i = 1, \ldots, k$ the line segments connecting v_i and v_{i+1}. In the sequel, we use $\mathbf{f}(\overrightarrow{\mathbf{V}})$ to represent the polygonal line formed by connecting consecutive vertices in $\overrightarrow{\mathbf{V}}$ if no confusion arises. Let $V_i, i = 1, \ldots, k+1$ and $S_i, i = 1, \ldots, k$ be the Voronoi partitions of $\mathbb{R}^d$ with respect to $\mathbf{f}$, i.e., regions consisting of all points closer to vertex v_i or segment s_i. Figure 5 shows an example of Voronoi partition with respect to $\mathbf{f}$ with three segments.

Neighborhood. For any $x \in \mathbb{R}^d$, we define the neighborhood $\mathcal{N}(x)$ with respect to $\mathbf{f}$ as the union of all Voronoi partitions whose closure intersects with two vertices connecting the projection $\mathbf{f}(s_{\mathbf{f}}(x))$ of x to $\mathbf{f}$. For example, for the point x in Figure 5, its neighborhood $\mathcal{N}(x)$ is the union of S_2, V_3, S_3 and V_4. In addition, let $\mathcal{N}_t(x) = \{x_s \in \mathcal{N}(x), s = 1, \ldots, t.\}$ be the set of observations $x_{1:t}$ belonging to $\mathcal{N}(x)$ and $\bar{\mathcal{N}}_t(x)$ be its average. Let $\mathcal{D}(M) = \sup_{x,y \in M} ||x - y||_2$ denote the diameter of set $M \subset \mathbb{R}^d$. We finally define the local grid $\mathcal{Q}_{\delta,t}(x)$ of $x \in \mathbb{R}^d$ at time t as

$$\mathcal{Q}_{\delta,t}(x) = B(\bar{\mathcal{N}}_t(x), \mathcal{D}(\mathcal{N}_t(x)) \cap \mathcal{Q}_\delta.$$

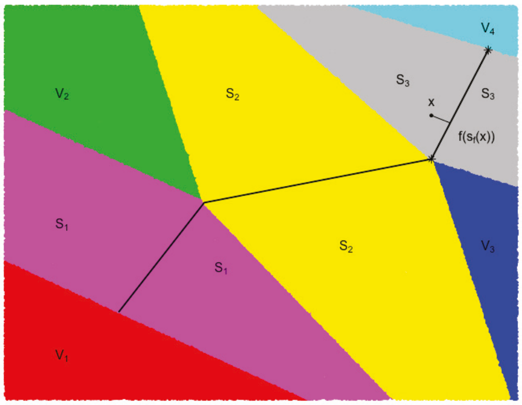


Figure 5. An example of a Voronoi partition.

We can finally proceed to the definition of the neighborhood $\mathcal{U}(\hat{\mathbf{f}}_t)$ of $\hat{\mathbf{f}}_t$. Assume $\hat{\mathbf{f}}_t$ has $k_t + 1$ vertices $\overrightarrow{\mathbf{V}} = (\underbrace{v_{1:i_t-1}}_{(i)}, \underbrace{v_{i_t:j_t-1}}_{(ii)}, \underbrace{v_{j_t:k_t+1}}_{(iii)})$, where vertices of (ii) belong to $\mathcal{Q}_{\delta,t}(x_t)$ while those of (i) and (iii) do not. The neighborhood $\mathcal{U}(\hat{\mathbf{f}}_t)$ consists of $\mathbf{f}$ sharing vertices (i) and (iii) with $\hat{\mathbf{f}}_t$, but can be equipped with different vertices (ii) in $\mathcal{Q}_{\delta,t}(x_t)$; i.e.,

$$\mathcal{U}(\hat{\mathbf{f}}_t) = \left\{ \mathbf{f}(\overrightarrow{\mathbf{V}}), \quad \overrightarrow{\mathbf{V}} = \left(v_{1:i_t-1}, v_{1:m}, v_{j_t:k_t+1} \right) \right\},$$

where $v_{1:m} \in \mathcal{Q}_{\delta,t}(x_t)$ and m is given by

$$m = \begin{cases} j_t - i_t - 1 & \text{reduce segments by 1 unit,} \\ j_t - i_t & \text{same number of segments,} \\ j_t - i_t + 1 & \text{increase segments by 1 unit.} \end{cases}$$

In Algorithm 3, we initiate the principal curve $\hat{\mathbf{f}}_1$ as the first component line segment whose vertices are the two farthest projections of data $x_{1:t_0}$ (t_0 can be set to 20 in practice) on the first component line. The reward of $\mathbf{f}$ at round t in this setting is therefore $r_{\mathbf{f},t} = c_0 - \Delta(\mathbf{f}, x_{t_0+t})$. Algorithm 3 has an exploration phase (when $I_t = 1$) and an exploitation phase ($I_t = 0$). In the exploration phase, it is allowed to observe rewards of all actions and to choose an optimal perturbed action from the set $\mathcal{F}_p$ of all actions. In the exploitation phase, only rewards of a part of actions can be accessed and rewards of others are estimated by a constant, and we update our action from the neighborhood $\mathcal{U}\left(\hat{\mathbf{f}}_{t-1}\right)$ of the previous action $\hat{\mathbf{f}}_{t-1}$. This local update (or search) greatly reduces computation complexity since $|\mathcal{U}(\hat{\mathbf{f}}_{t-1})| \ll |\mathcal{F}_p|$ when p is large. In addition, this local search will be enough to account for the case when x_t locates in $\mathcal{U}\left(\hat{\mathbf{f}}_{t-1}\right)$. The parameter β needs to be carefully calibrated since it should not be too large to ensure that the condition $cond(t)$ is non-empty; otherwise, all rewards are estimated by the same constant and thus lead to the same descending ordering of tuples for both $\left(\sum_{s=1}^{t-1} \hat{r}_{\mathbf{f},s}, \mathbf{f} \in \mathcal{F}_p\right)$ and $\left(\sum_{s=1}^{t} \hat{r}_{\mathbf{f},s}, \mathbf{f} \in \mathcal{F}_p\right)$. Therefore, we may face the risk of having $\hat{\mathbf{f}}_{t+1}$ in the neighborhood of $\hat{\mathbf{f}}_t$ even if we are in the exploration phase at time $t+1$. Conversely, very small β could result in large bias for the estimation $\frac{r_{\mathbf{f},t}}{\mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f} | \mathcal{H}_t\right)}$ of $r_{\mathbf{f},t}$. Note that the exploitation phase is close yet different to the label efficient prediction ([40], Remark 1.1) since we allow an action at time t to be different from the previous one. Ref. [41] proposed the *geometric resampling* method to estimate the conditional probability $\mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f} | \mathcal{H}_t\right)$ since this quantity often does not have an explicit form. However, due to the simple exponential distribution of $z_{\mathbf{f}}$ chosen in our case, an explicit form of $\mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f} | \mathcal{H}_t\right)$ is straightforward.

Algorithm 3 A locally greedy algorithm for sequentially learning principal curves.

```
1:  Input parameters: $p > 0$, $R > 0$, $L > 0$, $\epsilon > 0$, $\alpha > 0$, $1 > \beta > 0$ and any penalty function $h$
2:  Initialization: Given $(x_t)_{1:t_0}$, obtain $\hat{\mathbf{f}}_1$ as the first principal component
3:  For $t = 2, \dots, T$
4:      Draw $I_t \sim Bernoulli(\epsilon)$ and $z_{\mathbf{f}} \sim \pi$.
5:      Let
            $$\hat{\sigma}_t = \text{sort}\left(\mathbf{f}, \quad \sum_{s=1}^{t-1} \hat{r}_{\mathbf{f},s} - \frac{1}{\eta_{t-1}} h(\mathbf{f}) + \frac{1}{\eta_{t-1}} z_{\mathbf{f}}\right),$$
        i.e., sorting all $\mathbf{f} \in \mathcal{F}_p$ in descending order according to their perturbed cumulative reward till $t-1$.
6:      If $I_t = 1$, set $\mathcal{A}_t = \mathcal{F}_p$ and $\hat{\mathbf{f}}_t = \hat{\sigma}^t(\mathcal{A}_t)$ and observe $r_{\hat{\mathbf{f}}_t,t}$
7:
            $$\hat{r}_{\mathbf{f},t} = r_{\mathbf{f},t} \quad \text{for} \quad \mathbf{f} \in \mathcal{F}_p.$$
8:      If $I_t = 0$, set $\mathcal{A}_t = \mathcal{U}(\hat{\mathbf{f}}_{t-1})$, $\hat{\mathbf{f}}_t = \hat{\sigma}^t(\mathcal{A}_t)$ and observe $r_{\hat{\mathbf{f}}_t,t}$
9:
            $$\hat{r}_{\mathbf{f},t} = \begin{cases} \frac{r_{\mathbf{f},t}}{\mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f} | \mathcal{H}_t\right)} & \text{if } \mathbf{f} \in \mathcal{U}(\hat{\mathbf{f}}_{t-1}) \cap cond(t) \text{ and } \hat{\mathbf{f}}_t = \mathbf{f}, \\ \alpha & \text{otherwise,} \end{cases}$$
        where $\mathcal{H}_t$ denotes all the randomness before time $t$ and $cond(t) = \left\{\mathbf{f} \in \mathcal{F}_p : \mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f} | \mathcal{H}_t\right) > \beta\right\}$. In particular, when $t = 1$, we set $\hat{r}_{\mathbf{f},1} = r_{\mathbf{f},1}$ for all $\mathbf{f} \in \mathcal{F}_p$, $\mathcal{U}\left(\hat{\mathbf{f}}_0\right) = \emptyset$ and $\hat{r}_{\hat{\sigma}^1(\mathcal{U}(\hat{\mathbf{f}}_0)),1} \equiv 0$.
10: End for
```

Theorem 3. *Assume that $p > 6$, $T \geq 2|\mathcal{F}_p|^2$ and let $\beta = |\mathcal{F}_p|^{-\frac{1}{2}} T^{-\frac{1}{4}}$, $\alpha = \frac{c_0}{\beta}$, $\hat{c}_0 = \frac{2c_0}{\beta}$, $\epsilon = 1 - |\mathcal{F}_p|^{\frac{1}{2} - \frac{3}{p}} T^{-\frac{1}{4}}$ and*

$$\eta_1 = \eta_2 = \cdots = \eta_T = \frac{\sqrt{c_1 p + c_2 L + c_3}}{\sqrt{T(e-1)\hat{c}_0}}.$$

Then the procedure described in Algorithm 3 satisfies the regret bound

$$\sum_{t=1}^{T} \mathbb{E}\left[\Delta\left(\hat{\mathbf{f}}_t, x_t\right)\right] \leq \inf_{\mathbf{f} \in \mathcal{F}_p} \mathbb{E}\left[\sum_{t=1}^{T} \Delta(\mathbf{f}, x_t)\right] + \mathcal{O}(T^{\frac{3}{4}}).$$

The proof of Theorem 3 is presented in Section 6. The regret is upper bounded by a term of order $\left(|\mathcal{F}_p|^{\frac{1}{2}} T^{\frac{3}{4}}\right)$, sublinear in T. The term $(1-\epsilon)c_0 T = c_0 |\mathcal{F}_p|^{\frac{1}{2}} T^{\frac{3}{4}}$ is the price to pay for the local search (with a proportion $1-\epsilon$) of polygonal line $\hat{\mathbf{f}}_t$ in the neighborhood of the previous $\hat{\mathbf{f}}_{t-1}$. If $\epsilon = 1$, we would have that $\hat{c}_0 = c_0$, and the last two terms in the first inequality of Theorem 3 would vanish; hence, the upper bound reduces to Theorem 2. In addition, our algorithm achieves an order that is smaller (from the perspective of both the number $|\mathcal{F}_p|$ of all actions and the total rounds T) than [39] since at each time, the availability of actions for our algorithm can be either the whole action set or a neighborhood of the previous action while [39] consider at each time only partial and independent stochastic available set of actions generated from a predefined distribution.

5. Numerical Experiments

We illustrate the performance of Algorithm 3 on synthetic and real-life data. Our implementation (hereafter denoted by `slpc`—Sequential Learning of Principal Curves) is conducted with the R language and thus our most natural competitors are the R package `princurve`, which is the algorithm from [10], and `incremental, which is the algorithm from SCMS` [23]. We let $p = 50$, $R = \max_{t=1,\ldots,T} ||x||_2/\sqrt{d}$, $L = 0.1p\sqrt{d}R$. The spacing δ of the lattice is adjusted with respect to data scale.

Synthetic data We generate a dataset $\{x_t \in \mathbb{R}^2, t = 1, \ldots, 500\}$ uniformly along the curve $y = 0.05 \times (x-5)^3$, $x \in [0, 10]$. Table 1 shows the regret (first row) for

- the ground truth (sum of squared distances of all points to the true curve),
- `princurve` and `incremental SCMS` (sum of squared distances between observation x_{t+1} and fitted `princurve` on observations $x_{1:t}$),
- `slpc` (regret being equal to $\sum_{t=0}^{T-1} \mathbb{E}[\Delta(\hat{\mathbf{f}}_{t+1}, x_{t+1})]$ in both cases).

The mean computation time with different values for the time horizons T are also reported.

Table 1. The first line is the regret (cumulative loss) on synthetic data (average over 10 trials, with standard deviation in brackets). Second and third lines are the average computation time for two values of the time horizon T. `princurve` and `incremental SCMS` are deterministic, hence the zero standard deviation for regret.

Ground Truth	`Princurve`	`Incremental SCMS`	`slpc`
2.48 (0)	26.02 (0)	19.09 (0)	20.83 (3.23)
T = 500	0.029 s (0.0001 s)	18.79 s (0.007 s)	1.44 s (0.030 s)
T = 5000	0.35 s (0.006 s)	>60 s (NA)	4.13 s (0.807 s)

Table 1 demonstrates the advantages of our method `slpc`, as it achieved the optimal tradeoff between performance (in terms of regret) and runtime. Although `princurve` outperformed the other two algorithms in terms of computation time, it yielded the largest regret, since it outputs a curve which does not pass in "the middle of data" but rather bends towards the curvature of the data cloud, as shown in Figure 6 where the predicted principal curves $\hat{\mathbf{f}}_{t+1}$ for `princurve`, `incremental SCMS` and `slpc` are presented. `incremental SCMS` and `slpc` both yielded satisfactory results, although the mean computation time of `splc` was significantly smaller than that of `incremental SCMS` (the reason being that eigenvectors of the Hessian of PDF need to be computed in `incremental SCMS`). Figure 7 showed, respectively, the estimation of the regret of `slpc` and its per-round value (i.e., the cumulative loss divided by the number of rounds) both with respect to the round t. The jumps in the per-round curve occurred at the beginning, due to the initialization from a first principal component and to the collection of new data. When data accumulates, the vanishing pattern of the per-round curve illustrates that the regret is sublinear in t, which matches our aforementioned theoretical results.

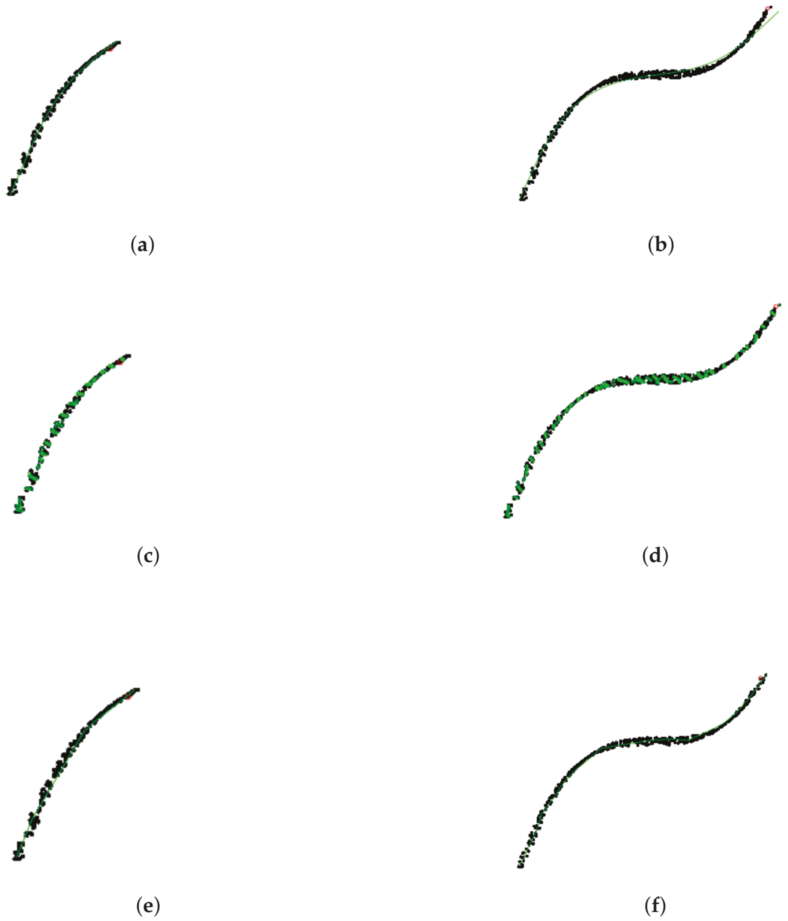
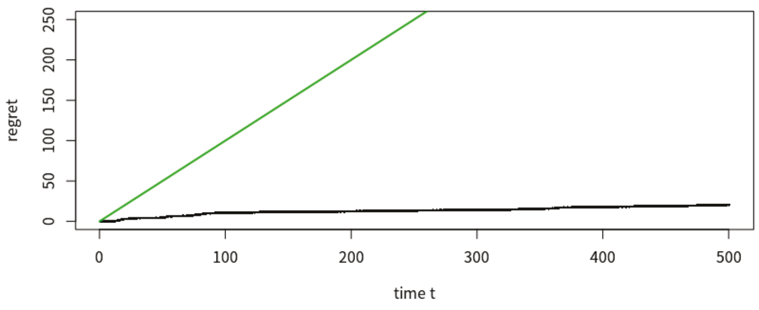
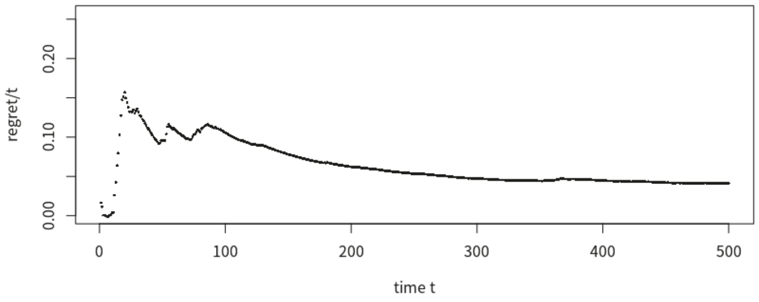


Figure 6. Synthetic data. Black dots represent data $x_{1:t}$. The red point is the new observation x_{t+1}. `princurve` (solid red) and `slpc` (solid green). (**a**) $t = 150$, `princurve`. (**b**) $t = 450$, `princurve`. (**c**) $t = 150$, `incremental SCMS`. (**d**) $t = 450$, `incremental SCMS`. (**e**) $t = 150$, `slpc`. (**f**) $t = 450$, `slpc`.

In addition, to better illustrate the way `slpc` works between two epochs, Figure 8 focuses on the impact of collecting a new data point on the principal curve. We see that only a local vertex is impacted, whereas the rest of the principal curve remains unaltered. This cutdown in algorithmic complexity is one the key assets of `slpc`.

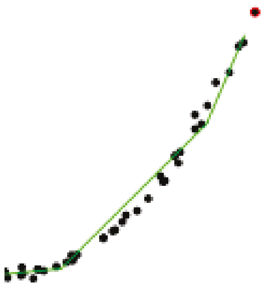


(**a**)

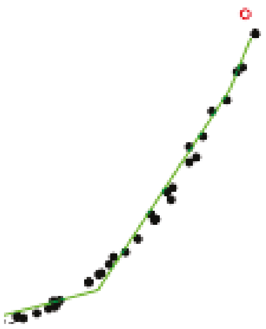


(**b**)

Figure 7. Mean estimation of regret and per-round regret of `slpc` with respect to time round t, for the horizon $T = 500$. (**a**) Mean estimation of the regret of `slpc` over 20 trials (black line) and a bisection line (green) with respect to time round t. (**b**) Per-round of estimated regret of `slpc` with respect to t.



(**a**)



(**b**)

Figure 8. Synthetic data. Zooming in: how a new data point impacts the principal curve only locally. (**a**) At time $t = 97$. (**b**) And at time $t = 98$.

Synthetic data in high dimension. We also apply our algorithm on a dataset $\{x_t \in \mathbb{R}^6, t = 1, 2, \dots, 200\}$ in higher dimension. It is generated uniformly along a parametric curve whose coordinates are

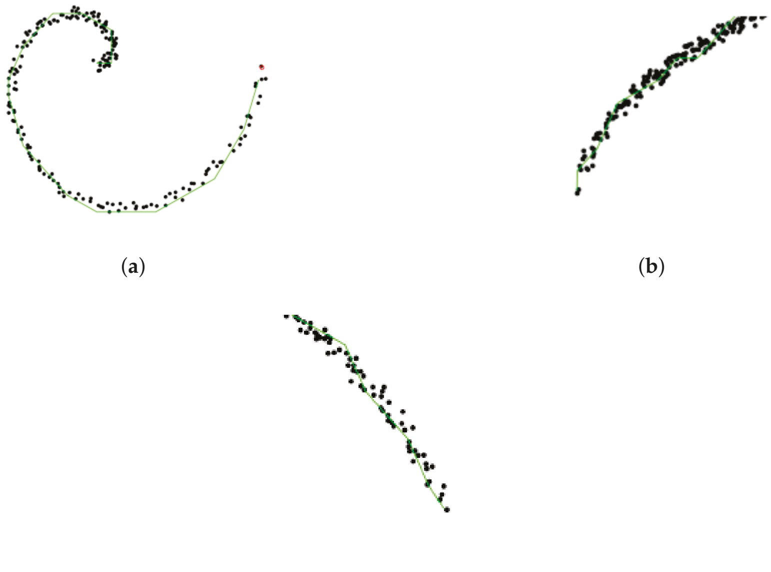
$$\begin{pmatrix} 0.5t\cos(t) \\ 0.5t\sin(t) \\ 0.5t \\ -t \\ \sqrt{t} \\ 2\ln(t+1) \end{pmatrix}$$

where t takes 100 equidistant values in $[0, 2\pi]$. To the best of our knowledge, [10,16,18] only tested their algorithm on 2-dimensional data. This example aims at illustrating that our algorithm also works on higher dimensional data. Table 2 shows the regret for the ground truth, `princurve` and `slpc`.

Table 2. Regret (cumulative loss) on synthetic high dimensional data in (average over 10 trials, with standard deviation in brackets). `princurve` and `incremental SCMS` are deterministic, hence the zero standard deviation.

Ground Truth	`Princurve`	`Incremental SCMS`	`slpc`
3.290 (0)	14.204 (0)	5.38 (0)	6.797 (0.409)

In addition, Figure 9 shows the behaviour of `slpc` (green) on each dimension.



(**a**) (**b**) (**c**)

Figure 9. `slpc` (green line) on synthetic high dimensional data from different perspectives. Black dots represent recordings $x_{1:99}$; the red dot is the new recording x_{200}. (**a**) `slpc`, $t = 199$, 1st and 2nd coordinates. (**b**) `slpc`, $t = 199$, 3th and 5th coordinates. (**c**) `slpc`, $t = 199$, 4th and 6th coordinates.

Seismic data. Seismic data spanning long periods of time are essential for a thorough understanding of earthquakes. The "Centennial Earthquake Catalog" [42] aims at providing a realistic picture of the seismicity distribution on Earth. It consists in a global catalog of locations and magnitudes of instrumentally recorded earthquakes from 1900 to 2008. We focus on a particularly representative seismic active zone (a lithospheric border close to Australia) whose longitude is between E130° to E180° and latitude between S70° to N30°, with $T = 218$ seismic recordings. As shown in Figure 10, `slpc` recovers nicely the tectonic plate boundary, but both `princurve` and `incremental SCMS` with well-calibrated bandwidth fail to do so.

Lastly, since no ground truth is available, we used the R^2 coefficient to assess the performance (residuals are replaced by the squared distance between data points and their projections onto the principal curve). The average over 10 trials was 0.990.

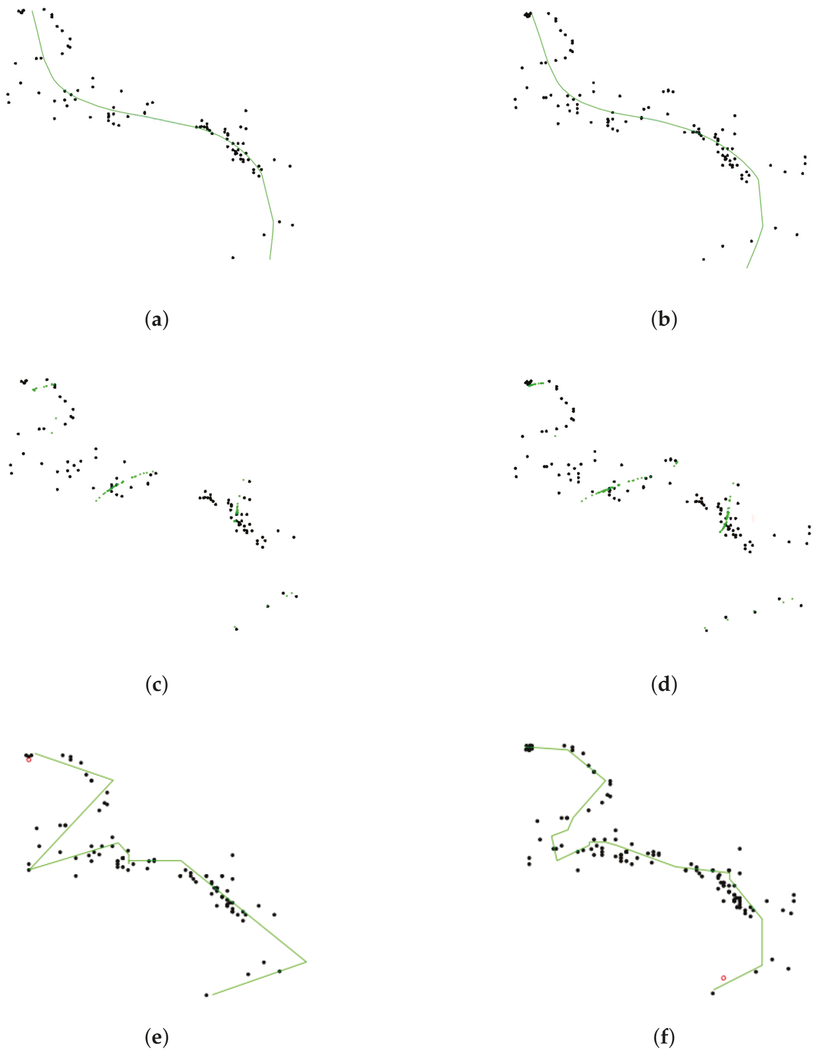

(**a**) (**b**) (**c**) (**d**) (**e**) (**f**)

Figure 10. Seismic data. Black dots represent seismic recordings $x_{1:t}$; the red dot is the new recording x_{t+1}. (**a**) `princurve`, $t = 100$. (**b**) `princurve`, $t = 125$. (**c**) `incremental SCMS`, $t = 100$. (**d**) `incremental SCMS`, $t = 125$. (**e**) `slpc`, $t = 100$. (**f**) `slpc`, $t = 125$.

Back to Seismic Data. Figure 11 was taken from the USGS website (https://earthquake.usgs.gov/data/centennial/) and gives the global locations of earthquakes for the period 1900–1999. The seismic data (latitude, longitude, magnitude of earthquakes, etc.) used in the present paper may be downloaded from this website.

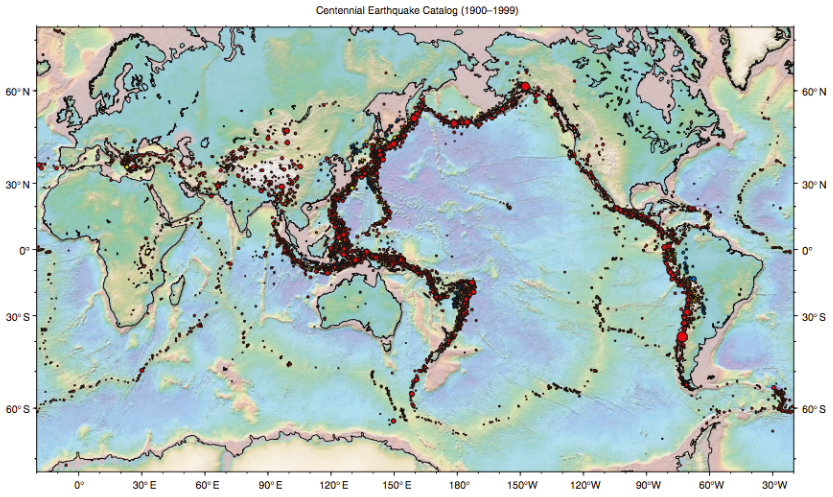
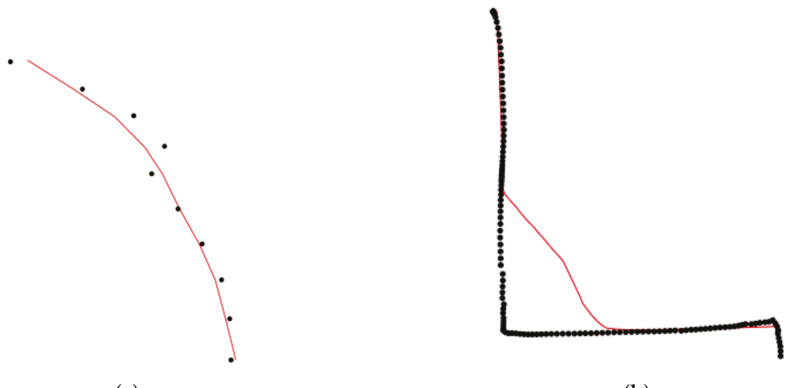
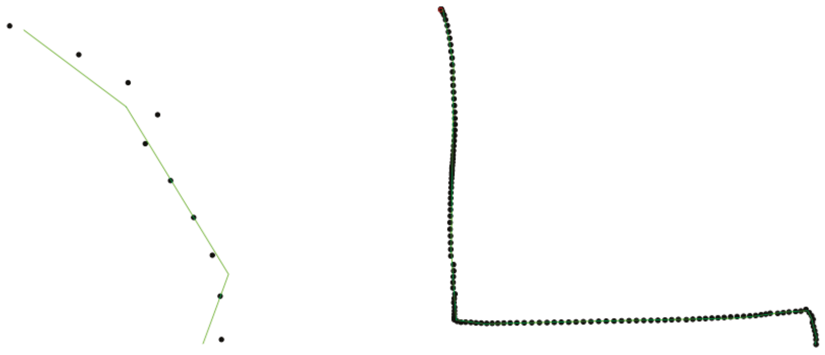


Figure 11. Seismic data from https://earthquake.usgs.gov/data/centennial/.

Daily Commute Data. The identification of segments of personal daily commuting trajectories can help taxi or bus companies to optimize their fleets and increase frequencies on segments with high commuting activity. Sequential principal curves appear to be an ideal tool to address this learning problem: we tested our algorithm on trajectory data from the University of Illinois at Chicago (https://www.cs.uic.edu/~boxu/mp2p/gps_data.html). The data were obtained from the GPS reading systems carried by two of the laboratory members during their daily commute for 6 months in the Cook county and the Dupage county of Illinois. Figure 12 presents the learning curves yielded by `princurve` and `slpc` on geolocalization data for the first person, on May 30. A particularly remarkable asset of `slpc` is that abrupt curvature in the data sequence was perfectly captured, whereas `princurve` does not enjoy the same flexibility. Again, we used the R^2 coefficient to assess the performance (where residuals are replaced by the squared distances between data points and their projections onto the principal curve). The average over 10 trials was 0.998.



(a) (b)



(c) (d)

Figure 12. Daily commute data. Black dots represent collected locations $x_{1:t}$. The red point is the new observation x_{t+1}. `princurve` (solid red) and `slpc` (solid green). (**a**) $t = 10$, `princurve`. (**b**) $t = 127$, `princurve`. (**c**) $t = 10$, `slpc`. (**d**) $t = 127$, `slpc`.

6. Proofs

This section contains the proof of Theorem 2 (note that Theorem 1 is a straightforward consequence, with $\eta_t = \eta$, $t = 0, \dots, T$) and the proof of Theorem 3 (which involves intermediary lemmas). Let us first define for each $t = 0, \dots, T$ the following forecaster sequence $(\hat{\mathbf{f}}_t^\star)_t$

$$\hat{\mathbf{f}}_0^\star = \arg\inf_{\mathbf{f}\in\mathcal{F}_p} \{\Delta_{\mathbf{f},0}\} = \arg\inf_{\mathbf{f}\in\mathcal{F}_p} \left\{ \frac{1}{\eta_0} h(\mathbf{f}) - \frac{1}{\eta_0} z_{\mathbf{f}} \right\},$$

$$\hat{\mathbf{f}}_t^\star = \arg\inf_{\mathbf{f}\in\mathcal{F}_p} \left\{ \sum_{s=0}^{t} \Delta_{\mathbf{f},s} \right\} = \arg\inf_{\mathbf{f}\in\mathcal{F}_p} \left\{ \sum_{s=1}^{t} \Delta(\mathbf{f}, x_s) + \frac{1}{\eta_{t-1}} h(\mathbf{f}) - \frac{1}{\eta_{t-1}} z_{\mathbf{f}} \right\}, \quad t \geq 1.$$

Note that $\hat{\mathbf{f}}_t^\star$ is an "illegal" forecaster since it peeks into the future. In addition, denote by

$$\mathbf{f}^\star = \arg\inf_{\mathbf{f}\in\mathcal{F}_p} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{1}{\eta_T} h(\mathbf{f}) \right\}$$

the polygonal line in $\mathcal{F}_p$ which minimizes the cumulative loss in the first T rounds plus a penalty term. $\mathbf{f}^\star$ is deterministic, and $\hat{\mathbf{f}}_t^\star$ is a random quantity (since it depends on $z_{\mathbf{f}}$,

$\mathbf{f} \in \mathcal{F}_p$ drawn from π). If several $\mathbf{f}$ attain the infimum, we chose $\mathbf{f}_T^\star$ as the one having the smallest complexity. We now enunciate the first (out of three) intermediary technical result.

Lemma 1. *For any sequence $x_1, \dots, x_T$ in $B(0, \sqrt{d}R)$,*

$$\sum_{t=0}^{T} \Delta_{\hat{\mathbf{f}}_t^\star, t} \leq \sum_{t=0}^{T} \Delta_{\hat{\mathbf{f}}_T^\star, t}, \qquad \pi\text{-almost surely.} \tag{5}$$

Proof. Proof by induction on T. Clearly (5) holds for $T = 0$. Assume that (5) holds for $T - 1$:

$$\sum_{t=0}^{T-1} \Delta_{\hat{\mathbf{f}}_t^\star, t} \leq \sum_{t=0}^{T-1} \Delta_{\hat{\mathbf{f}}_{T-1}^\star, t}.$$

Adding $\Delta_{\hat{\mathbf{f}}_T^\star, T}$ to both sides of the above inequality concludes the proof. □

By (5) and the definition of $\hat{\mathbf{f}}_T^\star$, for $k \geq 1$, we have π-almost surely that

$$\begin{aligned}
\sum_{t=1}^{T} \Delta(\hat{\mathbf{f}}_t^\star, x_t) &\leq \sum_{t=1}^{T} \Delta(\hat{\mathbf{f}}_T^\star, x_t) + \frac{1}{\eta_T} h(\hat{\mathbf{f}}_T^\star) - \frac{1}{\eta_T} Z_{\hat{\mathbf{f}}_T^\star} + \sum_{t=0}^{T} \left(\frac{1}{\eta_{t-1}} - \frac{1}{\eta_t} \right) \left(h(\hat{\mathbf{f}}_t^\star) - Z_{\hat{\mathbf{f}}_t^\star} \right) \\
&\leq \sum_{t=1}^{T} \Delta(\mathbf{f}^\star, x_t) + \frac{1}{\eta_T} h(\mathbf{f}^\star) - \frac{1}{\eta_T} Z_{\mathbf{f}^\star} + \sum_{t=0}^{T} \left(\frac{1}{\eta_{t-1}} - \frac{1}{\eta_t} \right) \left(h(\hat{\mathbf{f}}_t^\star) - Z_{\hat{\mathbf{f}}_t^\star} \right) \\
&= \inf_{\mathbf{f} \in \mathcal{F}_p} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{1}{\eta_T} h(\mathbf{f}) \right\} - \frac{1}{\eta_T} Z_{\mathbf{f}^\star} + \sum_{t=0}^{T} \left(\frac{1}{\eta_{t-1}} - \frac{1}{\eta_t} \right) \left(h(\hat{\mathbf{f}}_t^\star) - Z_{\hat{\mathbf{f}}_t^\star} \right),
\end{aligned}$$

where $1/\eta_{-1} = 0$ by convention. The second and third inequality is due to respectively the definition of $\hat{\mathbf{f}}_T^\star$ and $\mathbf{f}_T^\star$. Hence

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T} \Delta\left(\hat{\mathbf{f}}_t^\star, x_t \right) \right] &\leq \inf_{\mathbf{f} \in \mathcal{F}_p} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{1}{\eta_T} h(\mathbf{f}) \right\} - \frac{1}{\eta_T} \mathbb{E}[Z_{\mathbf{f}_T^\star}] \\
&\quad + \sum_{t=0}^{T} \mathbb{E}\left[\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) \left(-h(\hat{\mathbf{f}}_t^\star) + Z_{\hat{\mathbf{f}}_t^\star} \right) \right] \\
&\leq \inf_{\mathbf{f} \in \mathcal{F}_p} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{1}{\eta_T} h(\mathbf{f}) \right\} + \sum_{t=1}^{T} \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) \mathbb{E}\left[\sup_{\mathbf{f} \in \mathcal{F}_p} (-h(\mathbf{f}) + Z_{\mathbf{f}}) \right] \\
&= \inf_{\mathbf{f} \in \mathcal{F}_p} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{1}{\eta_T} h(\mathbf{f}) \right\} + \frac{1}{\eta_T} \mathbb{E}\left[\sup_{\mathbf{f} \in \mathcal{F}_p} (-h(\mathbf{f}) + Z_{\mathbf{f}}) \right],
\end{aligned}$$

where the second inequality is due to $\mathbb{E}[Z_{\mathbf{f}_T^\star}] = 0$ and $\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) > 0$ for $t = 0, 1, \dots, T$ since η_t is decreasing in t in Theorem 2. In addition, for $y \geq 0$, one has

$$\mathbb{P}(-h(\mathbf{f}) + Z_{\mathbf{f}} > y) = \mathrm{e}^{-h(\mathbf{f}) - y}.$$

Hence, for any $y \geq 0$

$$\mathbb{P}\left(\sup_{\mathbf{f} \in \mathcal{F}_p} (-h(\mathbf{f}) + Z_{\mathbf{f}}) > y \right) \leq \sum_{\mathbf{f} \in \mathcal{F}_p} \mathbb{P}(Z_{\mathbf{f}} \geq h(\mathbf{f}) + y) = \sum_{\mathbf{f} \in \mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})} \mathrm{e}^{-y} = u\mathrm{e}^{-y},$$

where $u = \sum_{\mathbf{f} \in \mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})}$. Therefore, we have

$$\begin{aligned}
\mathbb{E}\left[\sup_{\mathbf{f} \in \mathcal{F}_p} (-h(\mathbf{f}) + Z_{\mathbf{f}}) - \ln u\right] &\leq \mathbb{E}\left[\max\left(0, \sup_{\mathbf{f} \in \mathcal{F}_p} (-h(\mathbf{f}) + Z_{\mathbf{f}} - \ln u)\right)\right] \\
&\leq \int_0^{\infty} \mathbb{P}\left(\max\left(0, \sup_{\mathbf{f} \in \mathcal{F}_p} (-h(\mathbf{f}) + Z_{\mathbf{f}} - \ln u)\right) > y\right) \mathrm{d}y \\
&\leq \int_0^{\infty} \mathbb{P}\left(\sup_{\mathbf{f} \in \mathcal{F}_p} (-h(\mathbf{f}) + Z_{\mathbf{f}}) > y + \ln u\right) \mathrm{d}y \\
&\leq \int_0^{\infty} u \mathrm{e}^{-(y + \ln u)} \mathrm{d}y = 1.
\end{aligned}$$

We thus obtain

$$\mathbb{E}\left[\sum_{t=1}^{T} \Delta\left(\hat{\mathbf{f}}_t^{\star}, x_t\right)\right] \leq \inf_{\mathbf{f} \in \mathcal{F}_p} \left\{\sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{1}{\eta_T} h(\mathbf{f})\right\} + \frac{1}{\eta_T}\left(1 + \ln \sum_{\mathbf{f} \in \mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})}\right). \tag{6}$$

Next, we control the regret of Algorithm 2.

Lemma 2. *Assume that $z_{\mathbf{f}}$ is sampled from the symmetric exponential distribution in $\mathbb{R}$, i.e., $\pi(z) = \mathrm{e}^{-z} \mathbb{1}_{\{z>0\}}$. Assume that $\sup_{t=1,\ldots,T} \eta_{t-1} \leq \frac{1}{d(2R+\delta)^2}$, and define $c_0 = d(2R+\delta)^2$. Then for any sequence $(x_t) \in B(0, \sqrt{d}R)$, $t = 1, \ldots, T$,*

$$\sum_{t=1}^{T} \mathbb{E}\left[\Delta\left(\hat{\mathbf{f}}_t, x_t\right)\right] \leq \sum_{t=1}^{T} (1 + \eta_{t-1} c_0 (\mathrm{e} - 1)) \mathbb{E}\left[\Delta\left(\hat{\mathbf{f}}_t^{\star}, x_t\right)\right]. \tag{7}$$

Proof. Let us denote by

$$F_t(Z_{\mathbf{f}}) = \Delta\left(\hat{\mathbf{f}}_t, x_t\right) = \Delta\left(\operatorname*{arg\,inf}_{\mathbf{f} \in \mathcal{F}} \left(\sum_{s=1}^{t-1} \Delta(\mathbf{f}, x_s) + \frac{1}{\eta_{t-1}} h(\mathbf{f}) - \frac{1}{\eta_{t-1}} Z_{\mathbf{f}}\right), x_t\right)$$

the instantaneous loss suffered by the polygonal line $\hat{\mathbf{f}}_t$ when x_t is obtained. We have

$$\begin{aligned}
\mathbb{E}[\Delta\left(\hat{\mathbf{f}}_t^{\star}, x_t\right)] &= \int F_t(z - \eta_{t-1} \Delta(\mathbf{f}, x_t)) \pi(z) \mathrm{d}z \\
&= \int F_t(z) \pi(z + \eta_{t-1} \Delta(\mathbf{f}, x_t)) \mathrm{d}z \\
&= \int F_t(z) \mathrm{e}^{-(z + \eta_{t-1} \Delta(\mathbf{f}, x_t))} \mathrm{d}z \\
&\geq \mathrm{e}^{-\eta_{t-1} d(2R+\delta)^2} \int F_t(z) \mathrm{e}^{-z} \mathrm{d}z \\
&= \mathrm{e}^{-\eta_{t-1} d(2R+\delta)^2} \mathbb{E}[\Delta\left(\hat{\mathbf{f}}_t, x_t\right)],
\end{aligned}$$

where the inequality is due to the fact that $\Delta(\mathbf{f}, x) \leq d(2R+\delta)^2$ holds uniformly for any $\mathbf{f} \in \mathcal{F}_p$ and $x \in B(0, \sqrt{d}R)$. Finally, summing on t on both sides and using the elementary inequality $\mathrm{e}^x \leq 1 + (\mathrm{e} - 1)x$ if $x \in (0, 1)$ concludes the proof. □

Lemma 3. *For $k \in [\![1,p]\!]$, we control the cardinality of set $\{\mathbf{f} \in \mathcal{F}_p, \mathcal{K}(\mathbf{f}) = k\}$ as*

$$\ln\left|\{\mathbf{f} \in \mathcal{F}_p, \mathcal{K}(\mathbf{f}) = k\}\right| \leq \left(\ln(8pe V_d) + 3d^{\frac{3}{2}} - d\right)k + \left(\frac{\ln 2}{\delta\sqrt{d}} + \frac{d}{\delta}\right)L + d\ln\left(\frac{\sqrt{d}(2R+\delta)}{\delta}\right)$$
$$\overset{\Delta}{=} c_1 k + c_2 L + c_3,$$

where V_d denotes the volume of the unit ball in $\mathbb{R}^d$.

Proof. First, let $N_{k,\delta}$ denote the set of polygonal lines with k segments and whose vertices are in $\mathcal{Q}_\delta$. Notice that $N_{k,\delta}$ is different from $\{\mathbf{f} \in \mathcal{F}_p, \mathcal{K}(\mathbf{f}) = k\}$ and that

$$\left|\{\mathbf{f} \in \mathcal{F}_p, \mathcal{K}(\mathbf{f}) = k\}\right| \leq \binom{p}{k}\left|N_{k,\delta}\right|.$$

Hence

$$\begin{aligned}\ln\left|\{\mathbf{f} \in \mathcal{F}_p, \mathcal{K}(\mathbf{f}) = k\}\right| &\leq \ln\binom{p}{k} + \ln\left|N_{k,\delta}\right| \\ &\leq k\ln\frac{pe}{k} + k\left(\ln 8V_d + 3d^{\frac{3}{2}} - d\right) + \left(\frac{\ln 2}{\sqrt{d}\delta} + \frac{d}{\delta}\right)L + d\ln\left(\frac{\sqrt{d}(2R+\delta)}{\delta}\right) \\ &\leq k\ln(pe) + k\left(\ln 8V_d + 3d^{\frac{3}{2}} - d\right) + \left(\frac{\ln 2}{\sqrt{d}\delta} + \frac{d}{\delta}\right)L + d\ln\left(\frac{\sqrt{d}(2R+\delta)}{\delta}\right),\end{aligned}$$

where the second inequality is a consequence to the elementary inequality $\binom{p}{k} \leq \left(\frac{pe}{k}\right)^k$ combined with Lemma 2 in [16]. □

We now have all the ingredients to prove Theorem 1 and Theorem 2.

First, combining (6) and (7) yields that

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\Delta(\hat{\mathbf{f}}_t, x_t)\right] &\leq \inf_{\mathbf{f}\in\mathcal{F}_p}\left\{\sum_{t=1}^{T}\Delta(\mathbf{f}, x_t) + \frac{1}{\eta_T}h(\mathbf{f})\right\} + \frac{1}{\eta_T}\left(\frac{1}{2} + \ln\sum_{\mathbf{f}\in\mathcal{F}_p} e^{-h(\mathbf{f})}\right) \\ &\quad + c_0(e-1)\sum_{t=1}^{T}\eta_{t-1}\mathbb{E}\left[\Delta(\hat{\mathbf{f}}_t^\star, x_t)\right] \\ &\leq \inf_{k\in[\![1,p]\!]}\left\{\inf_{\substack{\mathbf{f}\in\mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}}\left\{\sum_{t=1}^{T}\Delta(\mathbf{f}, x_t) + \frac{h(\mathbf{f})}{\eta_T}\right\}\right\} + \frac{1}{\eta_T}\left(\frac{1}{2} + \ln\sum_{\mathbf{f}\in\mathcal{F}_p} e^{-h(\mathbf{f})}\right) \\ &\quad + c_0(e-1)\sum_{t=1}^{T}\eta_{t-1}\mathbb{E}\left[\Delta(\hat{\mathbf{f}}_t^\star, x_t)\right].\end{aligned}$$

Assume that $\eta_t = \eta$, $t = 0, \dots, T$ and $h(\mathbf{f}) = c_1\mathcal{K}(\mathbf{f}) + c_2L + c_3$ for $\mathbf{f} \in \mathcal{F}_p$, then $(\frac{1}{2} + \sum_{\mathbf{f}\in\mathcal{F}_p} e^{-h(\mathbf{f})}) \leq 0$ and moreover

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\Delta(\hat{\mathbf{f}}_t, x_t)\right] &\leq S_{T,h,\eta} + \frac{1}{\eta}\left(\frac{1}{2} + \ln\sum_{\mathbf{f}\in\mathcal{F}_p} e^{-h(\mathbf{f})}\right) + c_0(e-1)\eta\sum_{t=1}^{T}\mathbb{E}\left[\Delta(\hat{\mathbf{f}}_t^\star, x_t)\right] \\ &\leq S_{T,h,\eta} + c_0(e-1)\eta S_{T,h,\eta} \\ &\leq S_{T,h,\eta} + \eta c_0(e-1)\inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^{T}\Delta(\mathbf{f}, x_t) + c_0(e-1)(c_1p + c_2L + c_3),\end{aligned}$$

where

$$S_{T,h,\eta} = \inf_{k\in[\![1,p]\!]} \left\{ \inf_{\substack{\mathbf{f}\in\mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{h(\mathbf{f})}{\eta} \right\} \right\}$$

and the second inequality is obtained with Lemma 1. By setting

$$\eta = \sqrt{\frac{c_1 p + c_2 L + c_3}{c_0(\mathrm{e}-1)\inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^{T}\Delta(\mathbf{f}, x_t)}}$$

we obtain

$$\sum_{t=1}^{T} \mathbb{E}\Big[\Delta(\hat{\mathbf{f}}_t, x_t)\Big] \le \inf_{k\in[\![1,p]\!]} \left\{ \inf_{\substack{\mathbf{f}\in\mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \sqrt{c_0(\mathrm{e}-1) r_{T,k,L}} \right\} \right\} + \sqrt{c_0(\mathrm{e}-1)L_{T,p,L}} + c_0(\mathrm{e}-1)c_1 p + c_2 L + c_3,$$

where $r_{T,k,L} = \inf_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^{T}\Delta(\mathbf{f}, x_t)(c_1 k + c_2 L + c_3)$. This proves Theorem 1.

Finally, assume that

$$\eta_0 = \frac{\sqrt{c_1 p + c_2 L + c_3}}{c_0\sqrt{(\mathrm{e}-1)}} \quad \text{and} \quad \eta_t = \frac{\sqrt{c_1 p + c_2 L + c_3}}{c_0\sqrt{(\mathrm{e}-1)t}}, \qquad t = 1, \dots, T.$$

Since $\mathbb{E}\Big[\Delta(\hat{\mathbf{f}}_t^\star, x_t)\Big] \le c_0$ for any $t = 1, \dots, T$, we have

$$\begin{aligned}
\sum_{t=1}^{T} \mathbb{E}\Big[\Delta(\hat{\mathbf{f}}_t, x_t)\Big] &\le \inf_{k\in[\![1,p]\!]} \left\{ \inf_{\substack{\mathbf{f}\in\mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + \frac{h(\mathbf{f})}{\eta_T} \right\} \right\} + \frac{1}{\eta_T}\left(1 + \ln \sum_{\mathbf{f}\in\mathcal{F}_p} \mathrm{e}^{-h(\mathbf{f})}\right) \\
&\quad + c_0^2(\mathrm{e}-1)\sum_{t=1}^{T}\eta_{t-1} \\
&\le \inf_{k\in[\![1,p]\!]} \left\{ \inf_{\substack{\mathbf{f}\in\mathcal{F}_p \\ \mathcal{K}(\mathbf{f})=k}} \left\{ \sum_{t=1}^{T} \Delta(\mathbf{f}, x_t) + c_0\sqrt{(\mathrm{e}-1)T(c_0 k + c_2 L + c_3)} \right\} \right\} \\
&\quad + 2c_0\sqrt{(\mathrm{e}-1)T(c_0 p + c_2 L + c_3)},
\end{aligned}$$

which concludes the proof of Theorem 2.

Lemma 4. *Using Algorithm 3, if* $0 < \epsilon \le 1$, $0 < \beta < 1$, $\alpha \ge \frac{(1-\beta)c_0}{\beta}$ *and* $\Big|\mathcal{U}\Big(\hat{\mathbf{f}}_{t-1}\Big)\Big| \ge 2$ *for all* $t \ge 2$, *where* $\Big|\mathcal{U}\Big(\hat{\mathbf{f}}_{t-1}\Big)\Big|$ *is the cardinality of* $\mathcal{U}\Big(\hat{\mathbf{f}}_{t-1}\Big)$, *then we have*

$$\sum_{t=1}^{T} \mathbb{E}\Big[r_{\hat{\mathbf{f}}_t,t}\Big] \ge \sum_{t=1}^{T} \mathbb{E}\Big[\hat{r}_{\hat{\sigma}^t(\mathcal{A}_t),t}\Big] - 2(1-\epsilon)\alpha\beta\sum_{t=1}^{T}\Big|\mathcal{U}\Big(\hat{\mathbf{f}}_{t-1}\Big)\Big|.$$

Proof. First notice that $\mathcal{A}_t = \mathcal{U}\Big(\hat{\mathbf{f}}_{t-1}\Big)$ if $I_t = 0$, and that for $t \ge 2$

$$
\begin{aligned}
\mathbb{E}\left[r_{\hat{\mathbf{f}}_t,t}\middle|\mathcal{H}_t, I_t = 0\right] =& \mathbb{E}\left[r_{\hat{\sigma}^t(\mathcal{A}_t),t}\middle|\mathcal{H}_t, I_t = 0\right] \\
=& \sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)} r_{\mathbf{f},t}\mathbb{P}\left(\hat{\sigma}^t(\mathcal{A}_t) = \mathbf{f}\middle|\mathcal{H}_t\right) + \sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)^c} r_{\mathbf{f},t}\mathbb{P}\left(\hat{\sigma}^t(\mathcal{A}_t) = \mathbf{f}\middle|\mathcal{H}_t\right) \\
\geq& \sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)} r_{\mathbf{f},t} + \sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)^c} \alpha\mathbb{P}\left(\hat{\sigma}^t(\mathcal{A}_t) = \mathbf{f}\middle|\mathcal{H}_t\right) \\
&- (1-\beta)\sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)} r_{\mathbf{f},t} - \sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)^c} (\alpha - r_{\mathbf{f},t})\mathbb{P}\left(\hat{\sigma}^t(\mathcal{A}_t) = \mathbf{f}\middle|\mathcal{H}_t\right) \\
=& \mathbb{E}\left[\hat{r}_{\hat{\sigma}^t(\mathcal{A}_t),t}\middle|\mathcal{H}_t, I_t = 0\right] - (1-\beta)\sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)} r_{\mathbf{f},t} \\
&- \sum_{\mathbf{f}\in\mathcal{A}_t\cap cond(t)^c} (\alpha - r_{\mathbf{f},t})\mathbb{P}\left(\hat{\sigma}^t(\mathcal{A}_t) = \mathbf{f}\middle|\mathcal{H}_t\right) \\
\geq& \mathbb{E}\left[\hat{r}_{\hat{\sigma}^t(\mathcal{A}_t),t}\middle|\mathcal{H}_t, I_t = 0\right] - (1-\beta)c_0|\mathcal{A}_t| - \alpha\beta|\mathcal{A}_t| \\
\geq& \mathbb{E}\left[\hat{r}_{\hat{\sigma}^t(\mathcal{A}_t),t}\middle|\mathcal{H}_t, I_t = 0\right] - 2\alpha\beta|\mathcal{A}_t|,
\end{aligned}
$$

where $cond(t)^c$ denotes the complement of set $cond(t)$. The first inequality above is due to the assumption that for all $\mathbf{f} \in \mathcal{A}_t \cap cond(t)$, we have $\mathbb{P}\left(\hat{\sigma}^t(\mathcal{A}_t) = \mathbf{f}\middle|\mathcal{H}_t\right) \geq \beta$. For $t = 1$, the above inequality is trivial since $\hat{r}_{\hat{\sigma}^1(\mathcal{U}(\hat{\mathbf{f}}_0)),1} \equiv 0$ by its definition. Hence, for $t \geq 1$, one has

$$
\begin{aligned}
\mathbb{E}\left[r_{\hat{\mathbf{f}}_t,t}\middle|\mathcal{H}_t\right] &= \epsilon\mathbb{E}\left[r_{\hat{\sigma}^t(\mathcal{F}_p),t}\middle|\mathcal{H}_t, I_t = 1\right] + (1-\epsilon)\mathbb{E}\left[r_{\hat{\sigma}^t(\mathcal{A}_t),t}\middle|\mathcal{H}_t, I_t = 0\right] \\
&\geq \mathbb{E}\left[\hat{r}_{\hat{\mathbf{f}}_t,t}\middle|\mathcal{H}_t\right] - 2\alpha\beta|\mathcal{A}_t|. \qquad (8)
\end{aligned}
$$

Summing on both sides of inequality (8) over t terminates the proof of Lemma 4. □

Lemma 5. *Let $\hat{c}_0 = \frac{c_0}{\beta} + \alpha$. If $0 < \eta_1 = \eta_2 = \cdots = \eta_T = \eta < \frac{1}{\hat{c}_0}$, then we have*

$$
\mathbb{E}\left[\max_{\hat{\sigma}}\left\{\sum_{t=1}^{T}\hat{r}_{\hat{\sigma}(\mathcal{A}_t),t} - \frac{1}{\eta}h(\hat{\sigma}(\mathcal{A}_t))\right\}\right] - \sum_{t=1}^{T}\mathbb{E}\left[\hat{r}_{\hat{\sigma}^t(\mathcal{A}_t),t}\right] \leq \hat{c}_0^2(e-1)\eta T + \hat{c}_0(e-1)(c_1 p + c_2 L + c_3).
$$

Proof. By the definition of $\hat{r}_{\mathbf{f},t}$ in Algorithm 3, for any $\mathbf{f} \in \mathcal{F}_p$ and $t \geq 1$, we have

$$
\hat{r}_{\mathbf{f},t} \leq \max\left\{\frac{r_{\mathbf{f},t}}{\mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f}\middle|\mathcal{H}_t\right)}, \alpha, r_{\mathbf{f},t}\right\} \leq \max\left\{\frac{c_0}{\beta}, \alpha\right\} \leq \hat{c}_0,
$$

where in the second inequality we use that $r_{\mathbf{f},t} \leq c_0$ for all $\mathbf{f}$ and t, and that $\mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f}\middle|\mathcal{H}_t\right) \geq \beta$ when $\mathbf{f} \in \mathcal{U}\left(\hat{\mathbf{f}}_{t-1}\right) \cap cond(t)$. The rest of the proof is similar to those of Lemmas 1 and 2. In fact, if we define by $\hat{\Delta}(\mathbf{f}, x_t) = \hat{c}_0 - \hat{r}_{\mathbf{f},t}$, then one can easily observe the following relation when $I_t = 1$ (similar relation in the case that $I_t = 0$)

$$\hat{\mathbf{f}}_t = \hat{\sigma}^t(\mathcal{F}_p) = \arg\max_{\mathbf{f}\in\mathcal{F}_p}\left\{\sum_{s=1}^{t-1}\hat{r}_{\mathbf{f},s} + \frac{1}{\eta}(z_{\mathbf{f}} - h(\mathbf{f}))\right\}$$
$$= \arg\min_{\mathbf{f}\in\mathcal{F}_p}\left\{\sum_{s=1}^{t-1}\hat{\Delta}(\mathbf{f}, x_s) + \frac{1}{\eta}(h(\mathbf{f}) - z_{\mathbf{f}})\right\}.$$

Then applying Lemmas 1 and 2 on this newly defined sequence $\hat{\Delta}\left(\hat{\mathbf{f}}_t, x_t\right), t = 1, \dots T$ leads to the result of Lemma 5. □

The proof of the upcoming Lemma 6 requires the following submartingale inequality: let $Y_0, \dots Y_T$ be a sequence of random variable adapted to random events $\mathcal{H}_0, \dots, \mathcal{H}_T$ such that for $1 \le t \le T$, the following three conditions hold:

$$\mathbb{E}[Y_t|H_t] \le 0, \quad \mathrm{Var}(Y_t|H_t) \le a^2, \quad Y_t - \mathbb{E}[Y_t|H_t] \le b.$$

Then for any $\lambda > 0$,

$$\mathbb{P}\left(\sum_{t=1}^{T} Y_t > Y_0 + \lambda\right) \le \exp\left(-\frac{\lambda^2}{2T(a^2 + b^2)}\right).$$

The proof can be found in Chung and Lu [43] (Theorem 7.3).

Lemma 6. *Assume that $0 < \beta < \frac{1}{|\mathcal{F}_p|}$, $\alpha \ge \frac{c_0}{\beta}$ and $\eta > 0$, then we have*

$$\mathbb{E}\left[\max_{\sigma}\left\{\sum_{t=1}^{T} r_{\sigma(\mathcal{A}_t),t} - \frac{1}{\eta}h(\sigma(\mathcal{A}_t))\right\}\right] - \mathbb{E}\left[\max_{\hat{\sigma}}\left\{\sum_{t=1}^{T}\hat{r}_{\hat{\sigma}(\mathcal{A}_t),t} - \frac{1}{\eta}h(\hat{\sigma}(\mathcal{A}_t))\right\}\right]$$
$$\le (1 - |\mathcal{F}_p|\beta)\sqrt{2T\left[\frac{c_0^2}{\beta} + \alpha^2(1-\beta) + (c_0 + 2\alpha)^2\right]\ln\left(\frac{1}{\beta}\right)} + |\mathcal{F}_p|\beta c_0 T.$$

Proof. First, we have almost surely that

$$\max_{\sigma}\left\{\sum_{t=1}^{T} r_{\sigma(\mathcal{A}_t),t} - \frac{1}{\eta}h(\sigma(\mathcal{A}_t))\right\} - \max_{\hat{\sigma}}\left\{\sum_{t=1}^{T}\hat{r}_{\hat{\sigma}(\mathcal{A}_t),t} - \frac{1}{\eta}h(\hat{\sigma}(\mathcal{A}_t))\right\} \le \max_{\mathbf{f}\in\mathcal{F}_p}\sum_{t=1}^{T}(r_{\mathbf{f},t} - \hat{r}_{\mathbf{f},t}).$$

Denote by $Y_{\mathbf{f},t} = r_{\mathbf{f},t} - \hat{r}_{\mathbf{f},t}$. Since

$$\mathbb{E}\left[\hat{r}_{\mathbf{f},t}\middle|\mathcal{H}_t\right] = \begin{cases} r_{\mathbf{f},t} + (1-\epsilon)\alpha\left(1 - \mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f}|\mathcal{H}_t\right)\right) & \text{if } \mathbf{f} \in \mathcal{U}(\hat{\mathbf{f}}_{t-1}) \cap cond(t), \\ \epsilon r_{\mathbf{f},t} + (1-\epsilon)\alpha & \text{otherwise,} \end{cases}$$

and $\alpha > c_0 \ge r_{\mathbf{f},t}$ uniformly for any $\mathbf{f}$ and t, we have uniformly that $\mathbb{E}[Y_t|\mathcal{H}_t] \le 0$, satisfying the first condition.

For the second condition, if $\mathbf{f} \in \mathcal{U}\left(\hat{\mathbf{f}}_{t-1}\right) \cap cond(t)$, then

$$\begin{aligned}\operatorname{Var}(Y_t|\mathcal{H}_t) =& \mathbb{E}\left[\hat{r}_{\mathbf{f},t}^2|\mathcal{H}_t\right] - (\mathbb{E}[\hat{r}_{\mathbf{f},t}|\mathcal{H}_t])^2 \\ \leq& \epsilon r_{\mathbf{f},t}^2 + (1-\epsilon)\left[\frac{r_{\mathbf{f},t}^2}{\mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f}|\mathcal{H}_t\right)} + \alpha\left(1 - \mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f}|\mathcal{H}_t\right)\right)\right] \\ &- \left[r_{\mathbf{f},t} + (1-\epsilon)\alpha\left(1 - \mathbb{P}\left(\hat{\mathbf{f}}_t = \mathbf{f}|\mathcal{H}_t\right)\right)\right]^2 \\ \leq& \frac{r_{\mathbf{f},t}^2}{\beta} + \alpha^2(1-\beta) \leq \frac{c_0^2}{\beta} + \alpha^2(1-\beta).\end{aligned}$$

Similarly, for $\mathbf{f} \notin \mathcal{U}\left(\hat{\mathbf{f}}_{t-1}\right) \cap cond(t)$, one can have $\operatorname{Var}(Y_t|\mathcal{H}_t) \leq \alpha^2$. Moreover, for the third condition, since

$$\mathbb{E}[Y_{\mathbf{f},t}|\mathcal{H}_t] \geq -2\alpha,$$

then

$$Y_{\mathbf{f},t} - \mathbb{E}[Y_{\mathbf{f},t}|\mathcal{H}_t] \leq r_{\mathbf{f},t} + 2\alpha \leq c_0 + 2\alpha.$$

Setting $\lambda = \sqrt{2T\left[\frac{c_0^2}{\beta} + \alpha^2(1-\beta) + (c_0+2\alpha)^2\right]\ln\left(\frac{1}{\beta}\right)}$ leads to

$$\mathbb{P}\left(\sum_{t=1}^{T} Y_{\mathbf{f},t} \geq \lambda\right) \leq \beta.$$

Hence the following inequality holds with probability $1 - \left|\mathcal{F}_p\right|\beta$

$$\max_{\mathbf{f}\in\mathcal{F}_p} \sum_{t=1}^{T} (r_{\mathbf{f},t} - \hat{r}_{\mathbf{f},t}) \leq \sqrt{2T\left[\frac{c_0^2}{\beta} + \alpha^2(1-\beta) + (c_0+2\alpha)^2\right]\ln\left(\frac{1}{\beta}\right)}.$$

Finally, noticing that $\max_{\mathbf{f}\in\mathcal{F}_p} \sum_{t=1}^{T}(r_{\mathbf{f},t} - \hat{r}_{\mathbf{f},t}) \leq c_0 T$ almost surely, we terminate the proof of Lemma 6. □

Proof of Theorem 3. Assume that $p > 6$, $T \geq 2|\mathcal{F}_p|^2$ and let

$$\beta = \left|\mathcal{F}_p\right|^{-\frac{1}{2}} T^{-\frac{1}{4}}, \qquad \alpha = \frac{c_0}{\beta}, \qquad \hat{c}_0 = \frac{2c_0}{\beta},$$
$$\eta_1 = \eta_2 = \cdots = \eta_T = \frac{\sqrt{c_1 p + c_2 L + c_3}}{\sqrt{T(e-1)\hat{c}_0}}, \qquad \epsilon = 1 - \left|\mathcal{F}_p\right|^{\frac{1}{2}-\frac{3}{p}} T^{-\frac{1}{4}}.$$

With those values, the assumptions of Lemmas 4, 5 and 6 are satisfied. Combining their results lead to the following

$$\begin{aligned}
\sum_{t=1}^{T}\mathbb{E}\left[r_{\hat{\mathbf{f}}_t,t}\right] \geq & \,\mathbb{E}\left[\max_{\sigma}\left\{\sum_{t=1}^{T} r_{\sigma(\mathcal{A}_t),t} - \frac{1}{\eta}h(\sigma(\mathcal{A}_t))\right\}\right] - 2\alpha\beta(1-\epsilon)\sum_{t=1}^{T}\left|\mathcal{U}\left(\hat{\mathbf{f}}_{t-1}\right)\right| \\
& - \hat{c}_0^2(e-1)\eta T - \hat{c}_0(e-1)(c_1 p + c_2 L + c_3) \\
& - (1-|\mathcal{F}_p|\beta)\sqrt{2T\left[\frac{c_0^2}{\beta} + \alpha^2(1-\beta) + (c_0+2\alpha)^2\right]\ln\left(\frac{1}{\beta}\right)} - |\mathcal{F}_p|\beta c_0 T \\
\geq & \,\mathbb{E}\left[\max_{\sigma}\left\{\sum_{t=1}^{T} r_{\sigma(\mathcal{A}_t),t} - \frac{1}{\eta}h(\sigma(\mathcal{A}_t))\right\}\right] - (1-\epsilon)|\mathcal{F}_p|^{\frac{3}{p}} c_0 T \\
& - \hat{c}_0^2(e-1)\eta T - \hat{c}_0(e-1)(c_1 p + c_2 L + c_3) \\
& - (1-|\mathcal{F}_p|\beta)\sqrt{2T\left[\frac{c_0^2}{\beta} + \alpha^2(1-\beta) + (c_0+2\alpha)^2\right]\ln\left(\frac{1}{\beta}\right)} - |\mathcal{F}_p|\beta c_0 T \\
\geq & \,\mathbb{E}\left[\max_{\sigma}\left\{\sum_{t=1}^{T} r_{\sigma(\mathcal{A}_t),t} - \frac{1}{\eta}h(\sigma(\mathcal{A}_t))\right\}\right] - \mathcal{O}\left(|\mathcal{F}_p|^{\frac{1}{2}} T^{\frac{3}{4}}\right),
\end{aligned}$$

where the second inequality is due to the fact that the cardinality $\left|\mathcal{U}\left(\hat{\mathbf{f}}_{t-1}\right)\right|$ is upper bounded by $|\mathcal{F}_p|^{\frac{3}{p}}$ for $t \geq 1$. In addition, using the definition of $r_{\mathbf{f},t}$ that $r_{\mathbf{f},t} = c_0 - \Delta(\mathbf{f}, x_t)$ terminates the proof of Theorem 3. □

Author Contributions: Conceptualization, L.L. and B.G.; Formal analysis, L.L. and B.G.; Methodology, B.G.; Project administration, B.G.; Software, L.L.; Supervision, B.G.; Writing—original draft, L.L. and B.G.; Writing—review and editing, L.L. and B.G. All authors have read and agreed to the published version of the manuscript.

Funding: LL is funded and supported by the Fundamental Research Funds for the Central Universities (Grand No. 30106210158) and National Natural Science Foundation of China (Grant No. 61877023), the Fundamental Research Funds for the Central Universities (CCNU19TD009). BG is supported in part by the U.S. Army Research Laboratory and the U. S. Army Research Office, and by the U.K. Ministry of Defence and the U.K. Engineering and Physical Sciences Research Council (EPSRC) under grant number EP/R013616/1. BG acknowledges partial support from the French National Agency for Research, grants ANR-18-CE40-0016-01 and ANR-18-CE23- 0015-02.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pearson, K. On lines and planes of closest fit to systems of point in space. *Philos. Mag.* **1901**, *2*, 559–572. [CrossRef]
2. Spearman, C. "General Intelligence", Objectively Determined and Measured. *Am. J. Psychol.* **1904**, *15*, 201–292. [CrossRef]
3. Hotelling, H. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.* **1933**, *24*, 417–441. [CrossRef]
4. Friedsam, H.; Oren, W.A. The application of the principal curve analysis technique to smooth beamlines. In Proceedings of the 1st International Workshop on Accelerator Alignment, Stanford , CA, USA, 31 July–2 August 1989.
5. Brunsdon, C. Path estimation from GPS tracks. In Proceedings of the 9th International Conference on GeoComputation, Maynoorth, Ireland, 3–5 September 2007.
6. Reinhard, K.; Niranjan, M. Parametric Subspace Modeling Of Speech Transitions. *Speech Commun.* **1999**, *27*, 19–42. [CrossRef]
7. Kégl, B.; Krzyżak, A. Piecewise linear skeletonization using principal curves. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 59–74. [CrossRef]
8. Banfield, J.D.; Raftery, A.E. Ice floe identification in satellite images using mathematical morphology and clustering about principal curves. *J. Am. Stat. Assoc.* **1992**, *87*, 7–16. [CrossRef]

9. Stanford, D.C.; Raftery, A.E. Finding curvilinear features in spatial point patterns: Principal curve clustering with noise. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 601–609. [CrossRef]
10. Hastie, T.; Stuetzle, W. Principal curves. *J. Am. Stat. Assoc.* **1989**, *84*, 502–516. [CrossRef]
11. Delicado, P. Another Look at Principal Curves and Surfaces. *J. Multivar. Anal.* **2001**, *77*, 84–116. [CrossRef]
12. Einbeck, J.; Tutz, G.; Evers, L. Local principal curves. *Stat. Comput.* **2005**, *15*, 301–313. [CrossRef]
13. Einbeck, J.; Tutz, G.; Evers, L. Data Compression and Regression through Local Principal Curves and Surfaces. *Int. J. Neural Syst.* **2010**, *20*, 177–192. [CrossRef]
14. Malo, J.; Gutiérrez, J. V1 non-linear properties emerge from local-to-global non-linear ICA. *Netw. Comput. Neural Syst.* **2006**, *17*, 85–102. [CrossRef]
15. Ozertem, U.; Erdogmus, D. Locally Defined Principal Curves and Surfaces. *J. Mach. Learn. Res.* **2011**, *12*, 1249–1286.
16. Kégl, B. Principal Curves: Learning, Design, and Applications. Ph.D. Thesis, Concordia University, Montreal, QC, Canada, 1999.
17. Kégl, B.; Krzyżak, A.; Linder, T.; Zeger, K. Learning and design of principal curves. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 281–297. [CrossRef]
18. Biau, G.; Fischer, A. Parameter selection for principal curves. *IEEE Trans. Inf. Theory* **2012**, *58*, 1924–1939. [CrossRef]
19. Barron, A.; Birgé, L.; Massart, P. Risk bounds for model selection via penalization. *Probab. Theory Relat. Fields* **1999**, *113*, 301–413. [CrossRef]
20. Birgé, L.; Massart, P. Minimal penalties for Gaussian model selection. *Probab. Theory Relat. Fields* **2007**, *183*, 33–73. [CrossRef]
21. Sandilya, S.; Kulkarni, S.R. Principal curves with bounded turn. *IEEE Trans. Inf. Theory* **2002**, *48*, 2789–2793. [CrossRef]
22. Cesa-Bianchi, N.; Lugosi, G. *Prediction, Learning and Games*; Cambridge University Press: New York, NY, USA, 2006.
23. Rudzicz, F.; Ghassabeh, Y.A. Incremental algorithm for finding principal curves. *IET Signal Process.* **2015**, *9*, 521–528.
24. Laparra, V.; Malo, J. Sequential Principal Curves Analysis. *arXiv* **2016**, arXiv:1606.00856.
25. Laparra, V.; Jiménez, S.; Camps-Valls, G.; Malo, J. Nonlinearities and Adaptation of Color Vision from Sequential Principal Curves Analysis. *Neural Comput.* **2012**, *24*, 2751–2788. [CrossRef]
26. Laparra, V.; Malo, J. Visual Aftereffects and Sensory Nonlinearities from a single Statistical Framework. *Front. Hum. Neurosci.* **2015**, *9*. [CrossRef]
27. Laparra, V.; Jiménez, S.; Tuia, D.; Camps-Valls, G.; Malo, J. Principal Polynomial Analysis. *Int. J. Neural Syst.* **2014**, *24*, 1440007. [CrossRef]
28. Laparra, V.; Malo, J.; Camps-Valls, G. Dimensionality Reduction via Regression in Hyperspectral Imagery. *IEEE J. Sel. Top. Signal Process.* **2015**, *9*, 1026–1036. [CrossRef]
29. Shawe-Taylor, J.; Williamson, R.C. A PAC analysis of a Bayes estimator. In Proceedings of the 10th annual conference on Computational Learning Theory, Nashville, TN, USA, 6–9 July 1997; pp. 2–9. [CrossRef]
30. McAllester, D.A. Some PAC-Bayesian Theorems. *Mach. Learn.* **1999**, *37*, 355–363. [CrossRef]
31. McAllester, D.A. PAC-Bayesian Model Averaging. In Proceedings of the 12th Annual Conference on Computational Learning Theory, Santa Cruz, CA, USA, 7–9 July 1999; pp. 164–170.
32. Li, L.; Guedj, B.; Loustau, S. A quasi-Bayesian perspective to online clustering. *Electron. J. Stat.* **2018**, *12*, 3071–3113. [CrossRef]
33. Guedj, B. A Primer on PAC-Bayesian Learning. In Proceedings of the Second Congress of the French Mathematical Society, Long Beach, CA, USA, 10 June 2019; pp. 391–414.
34. Alquier, P. User-friendly introduction to PAC-Bayes bounds. *arXiv* **2021**, arXiv:2110.11216.
35. Audibert, J.Y. Fast Learning Rates in Statistical Inference through Aggregation. *Ann. Stat.* **2009**, *37*, 1591–1646. [CrossRef]
36. Hutter, M.; Poland, J. Adaptive Online Prediction by Following the Perturbed Leader. *J. Mach. Learn. Res.* **2005**, *6*, 639–660.
37. Auer, P.; Cesa-Bianchi, N.; Freund, Y.; Schapire, R.E. The Nonstochastic multiarmed Bandit problem. *SIAM J. Comput.* **2003**, *32*, 48–77. [CrossRef]
38. Kleinberg, R.D.; Niculescu-Mizil, A.; Sharma, Y. Regret Bounds for Sleeping Experts and Bandits. In *COLT*; Springer: Berlin/Heidelberg, Germany, 2008.
39. Kanade, V.; McMahan, B.; Bryan, B. Sleeping Experts and Bandits with Stochastic Action Availability and Adversarial Rewards. *Artif. Intell. Stat.* **2009**, *3*, 1137–1155.
40. Cesa-Bianchi, N.; Lugosi, G.; Stoltz, G. Minimizing regret with label-efficient prediction. *IEEE Trans. Inf. Theory* **2005**, *51*, 2152–2162. [CrossRef]
41. Neu, G.; Bartók, G. *An Efficient Algorithm for Learning with Semi-Bandit Feedback*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8139, pp. 234–248.
42. Engdahl, E.R.; Villaseñor, A. 41 Global seismicity: 1900–1999. *Int. Geophys.* **2002**, *81*, 665–690.
43. Chung, F.; Lu, L. Concentration Inequalities and Martingale Inequalities: A Survey. *Internet Math.* **2006**, *3*, 79–127. [CrossRef]

Article

Still No Free Lunches: The Price to Pay for Tighter PAC-Bayes Bounds

Benjamin Guedj [1,2,*] **and Louis Pujol** [3,*]

1 Centre for Artificial Intelligence, Department of Computer Science, University College London, London WC1V 6LJ, UK
2 Inria Lille—Nord Europe Research Centre and Inria London, 59800 Lille, France
3 Laboratoire de Mathématiques d'Orsay, Université Paris-Saclay, CNRS, 91405 Orsay, France
* Correspondence: b.guedj@ucl.ac.uk (B.G.); louis.pujol@universite-paris-saclay.fr (L.P.)

Abstract: "No free lunch" results state the impossibility of obtaining meaningful bounds on the error of a learning algorithm without prior assumptions and modelling, which is more or less realistic for a given problem. Some models are "expensive" (strong assumptions, such as sub-Gaussian tails), others are "cheap" (simply finite variance). As it is well known, the more you pay, the more you get: in other words, the most expensive models yield the more interesting bounds. Recent advances in robust statistics have investigated procedures to obtain tight bounds while keeping the cost of assumptions minimal. The present paper explores and exhibits what the limits are for obtaining tight probably approximately correct (PAC)-Bayes bounds in a robust setting for cheap models.

Keywords: statistical learning theory; PAC-Bayes theory; no free lunch theorems

Citation: Guedj, B.; Pujol, L. Still No Free Lunches: The Price to Pay for Tighter PAC-Bayes Bounds. *Entropy* **2021**, *23*, 1529. https://doi.org/10.3390/e23111529

Academic Editor: Wray Buntine

Received: 29 August 2021
Accepted: 3 November 2021
Published: 18 November 2021

1. Introduction

For the sake of clarity, we focus on the supervised learning problem. We collect a sequence of input–output pairs $(X_i, Y_i)_{i=1}^N \in (\mathcal{X} \times \mathcal{Y})^N$, which we assume to be N independent realisations of a random variable drawn from a distribution P on $\mathcal{X} \times \mathcal{Y}$. The overarching goal in statistics and machine learning is to select a hypothesis f over a space $\mathcal{F}$ which, given a new input x in $\mathcal{X}$, delivers an output $f(x)$ in $\mathcal{Y}$, hopefully close (in a certain sense) to the unknown true output y. The quality of f is assessed through a loss function ℓ which characterises the discrepancy between the true output y and its prediction $f(x)$, and we define a global notion of risk as

$$R(f) = \mathbb{E}_{(X,Y)\sim \mathrm{P}}[\ell(f(X), Y)].$$

The aim of machine learning is to find a good (in the sense of a low risk) hypothesis $f \in \mathcal{F}$. In the generalised Bayes setting, the learning algorithm does not output a single hypothesis but rather a *distribution* ρ over the hypotheses space $\mathcal{F}$ and the associated bounds are called PAC-Bayesian bounds (see [1] for a survey of the topic).

As many probabilistic bounds stated in the statistics and machine learning literature, PAC-Bayesian bounds (where PAC stands for probably approximately correct—see [2]) commonly requires strong assumptions to hold, such as sub-Gaussian behaviour of some random variables. These assumptions can be misleading when dealing with true data as they do not take into account some practical situations, such as outlier contamination. Many efforts have been made recently to keep tight generalisation bounds valid with a few set of assumptions about the underlying distribution: this is known as robust learning [see [3] for a survey of the topic].

In this work we explore the possibility to establish a connection between recent techniques introduced by robust machine learning and PAC-Bayesian generalisation bounds. The result of our work is negative as we were not able to prove a PAC-Bayes bound in a robust statistics setting. However, we found it useful to write down our findings in order to give the interested reader a review of material involved in both robust statistics and PAC-Bayes theory and present the fundamental issues we faced as we believe it to be useful to the community.

Organisation of the paper. We introduce an elementary example and set a basic notation to illustrate the problem of robustness in Section 2, before providing an overview of recent advances in robust statistics in Section 3, and briefly introduce the field of PAC-Bayes learning in Section 4. We then propose in Section 5 a detailed study of the structural limits which do not allow for PAC-Bayes bounds which are simultaneously tight without requiring strong assumptions. The paper closes with a discussion in Section 6.

2. About the "No Free Lunch" Results

A class of results in statistics is known as "no free lunch" statements [see [4], Chapter 7]. The "no free lunch" results typically state that if one does not consider the restrictions on the modelling of the data-generating process, one cannot obtain meaningful deviation bounds in a non-asymptotic regime. The well-known trade-off is that the more restrictive the assumptions, the tighter the bounds. Let us illustrate this classical phenomenon by a simple example.

Assume that we have a dataset consisting in N real observations $x_1, \dots, x_N \in \mathbb{R}$ and consider they are independent, identically distributed (iid) realisations of a random variable X. Our goal is to estimate the mean of X and build a confidence interval for this estimate. As a start, let us focus on the empirical mean, denoted by $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$. As "no free lunch" results state, we have to consider a class of distributions to which the data-generating distribution P belongs.

2.1. Expensive and Cheap Models

If there is always a price to pay in order to derive insightful result, there is a variety of degrees of restrictions. In the remainder of the paper, we will focus on two classical models corresponding to a different level of demand on the random variables.

A first type of restriction we can make is an "expensive modelling". For $\sigma > 0$, let $\mathcal{P}^{\sigma}_{\text{expensive}}$ be the set of all real-valued random variables X satisfying:

$$\log(\mathbb{E}[\exp\{\lambda(X - \mathbb{E}[X])\}]) \leq \frac{\lambda^2\sigma^2}{2}.$$

This $\mathcal{P}^{\sigma}_{\text{expensive}}$ is the class of sub-Gaussian random variables with variance factor σ^2 [see [5] for a complete coverage of the topic]. We call this model "expensive" as this restriction is often considered unrealistic for real-life datasets and is hard or impossible to check in practice.

An alternative type of restriction is a "cheap modelling". For $\sigma > 0$, let $\mathcal{P}^{\sigma}_{\text{cheap}}$ be the set of real-valued random variables with a finite variance, upper bounded by σ^2. We call this model "cheap" as this is considerably less restrictive than the expensive one and is much more likely to hold in practice.

2.2. Confidence Interval for the Empirical Mean

Proposition 1 (Confidence intervals). *If we assume that $X \in \mathcal{P}^{\sigma}_{expensive}$, then for all $\delta \in (0, 1/2)$, the following random interval is a confidence interval for the mean of X at level $1 - \delta$:*

$$\left[\bar{x} \pm \frac{\sigma}{\sqrt{N}}\sqrt{2} \times \sqrt{2\log\left(\frac{1}{\delta}\right)}\right]. \tag{1}$$

If we assume that $X \in \mathcal{P}^{\sigma}_{cheap}$, then for all $\delta \in (0, 1)$, the following random interval is a confidence interval for the mean of X at level $1 - \delta$:

$$\left[\bar{x} \pm \frac{\sigma}{\sqrt{N}}\sqrt{\frac{1}{\delta}}\right]. \tag{2}$$

In the case of a cheap model, there is no hope to obtain a significantly tighter confidence interval with respect to δ if one uses the empirical mean [as proved in [6], Proposition 6.2].

Proof. To establish the first confidence interval (1), we first remark that if $X \in \mathcal{P}^{\sigma}_{\text{expensive}}$, then $\bar{x} \in \mathcal{P}^{\sigma/\sqrt{N}}_{\text{expensive}}$ and $\mathbb{E}[\bar{x}] = \mathbb{E}[X]$. So, applying Theorem 2.1 of [5] to $\bar{x} - \mathbb{E}[X]$ we obtain, for all $a > 0$:

$$\begin{aligned}
\mathbb{P}(|\bar{x} - \mathbb{E}[X]| > a) &= \mathbb{P}(\bar{x} - \mathbb{E}[X] > a) + \mathbb{P}(\bar{x} - \mathbb{E}[X] < -a) \\
&\leq 2\max(\mathbb{P}(\bar{x} - \mathbb{E}[X] > a), \mathbb{P}(\bar{x} - \mathbb{E}[X] < -a)) \\
&\leq 2\exp\left(-\frac{Na^2}{2\sigma^2}\right).
\end{aligned}$$

Setting $\delta = \exp\left(-\frac{Na^2}{2\sigma^2}\right)$ leads to the expected result. The second confidence interval (2) is obtained through Chebychev's inequality. $\mathbb{E}[\bar{x}] = \mathbb{E}[X]$ and as $X \in \mathcal{P}^{\sigma}_{\text{cheap}}$, $\text{Var}(\bar{x}) = \frac{\text{Var}(X)}{N} \leq \frac{\sigma^2}{N}$. So for all $a > 0$

$$\mathbb{P}(|\bar{x} - \mathbb{E}[X]| > a) \leq \frac{\sigma^2}{Na^2}.$$

Now, setting $\delta = \frac{\sigma^2}{Na^2}$ we get

$$\mathbb{P}\left(|\bar{x} - \mathbb{E}[X]| > \frac{\sigma}{\sqrt{N}}\sqrt{\frac{1}{\delta}}\right) \leq \delta.$$

□

Note that the dependence in δ is fairly different in both confidence intervals defined in (1) and (2): for fixed σ^2 and N, the $\sqrt{2} \times \sqrt{2\log(1/\delta)}$ regime (following the lunch metaphor, the "good lunch") is much more favourable than the $1/\sqrt{\delta}$ regime (the "bad lunch"). We illustrate this in Figure 1, where we plot $\sqrt{2} \times \sqrt{2\log(1/\delta)}$ and $1/\sqrt{\delta}$ as a function of $\delta \in (0, 1/2)$. We remark that for small values of δ, corresponding to a higher confidence level, the interval (1) will be much tighter than (2).

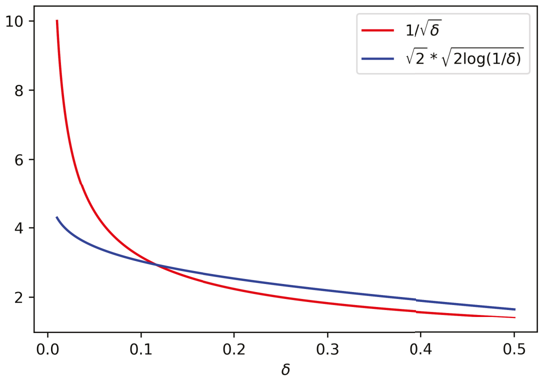


Figure 1. $\sqrt{2} \times \sqrt{2\log(1/\delta)}$ and $1/\sqrt{\delta}$ with respect to δ.

So, while it is clear that the best confidence interval requires more stringent assumptions, there have been attempts at relaxing those assumptions—or in other words, keeping equally good lunches at a cheaper cost.

3. Robust Statistics

Robust statistics address the following question: can we obtain tight bounds with minimal assumptions—or in other words, can we get a good cheap lunch? In the mean estimation case hinted in Section 2, the question becomes the following: if $\mathrm{P} \in \mathcal{P}^{\sigma}_{\text{cheap}}$, can we build a confidence interval at level $1-\delta$ with a size proportional to $\frac{\sigma}{\sqrt{N}}\sqrt{2\log(1/\delta)}$?

As mentioned above, there is no hope to achieve this goal with the empirical mean. Different alternative estimators have thus been considered in robust statistics, such as M-estimators [6] or median-of-means (MoM) estimators [see [7] for a recent survey, and references therein].

The key idea of MoM estimators is to achieve a compromise between the unbiased but non-robust empirical mean and the biased but robust median. As before, let us consider a sample of N real numbers $x_1, \ldots, x_N$, assumed to be an iid sequence drawn from a distribution P. Let $K \leq N$ be a positive integer and assume for simplicity that K is a divisor of N. To compute the MoM estimator, the first step consists of dividing the sample $(x_1, \ldots, x_N)$ into K non-overlapping blocks $B_1, \ldots, B_K$, each of length N/K. For each block, we then compute the empirical mean

$$\bar{x}_{B_i} = \frac{K}{N} \sum_{j \in B_i} x_j.$$

The MoM estimator is defined as the median of those means:

$$\mathrm{MoM}_K(x_1 \ldots, x_N) = \mathrm{median}\{\bar{x}_{B_1}, \ldots, \bar{x}_{B_K}\}.$$

This estimator has the following nice property.

Proposition 2 ([7], Proposition 12)**.** *Assume* $\mathrm{P} \in \mathcal{P}^{\sigma}_{cheap}$*, for* $\delta = \exp\left(-\frac{K}{8}\right)$,

$$\left[\mathrm{MOM}_K \pm \frac{\sigma}{\sqrt{N}} \times 4\sqrt{2\log\left(\frac{1}{\delta}\right)}\right] \tag{3}$$

is a confidence interval for the mean of X at the level $1-\delta$.

This property is quite encouraging, as for a cheap model we obtain a confidence interval similar, up to a numerical constant, to the best one (1) in Section 2. However, we also spot here an important limitation. The confidence interval (3) for MoM is only valid for the particular error threshold $\delta = \exp(-K/8)$, which depends on the number of blocks K (a parameter for the estimator MoM_K). The estimator must be changed each time we want to evaluate a different confidence level.

An ever more limiting feature is that the error threshold δ is constrained and cannot be set arbitrarily small, as in (1) or (2). Obviously, the number of blocks cannot exceed the sample size N, and the error threshold reaches its lowest tolerable value $\exp(-N/8)$. In other words, the interval defined in (3) can have confidence at most $1-\exp(-N/8)$.

Is this strong limitation specific to MoM estimators? No, say [8], [Theorem 3.2 and following remark]. This limitation is universal; over the class $\mathcal{P}^{\sigma}_{\text{cheap}}$, there is no estimator $\hat{x}$ of the mean such that there exists a constant $L > 1$ such that

$$\left[\hat{x} \pm \frac{\sigma}{\sqrt{N}} \times L\sqrt{2\log\left(\frac{1}{\delta}\right)}\right]$$

is a confidence interval at level $1 - \delta$ for δ lower than $e^{-\mathcal{O}(N)}$.

To sum up, a good and cheap lunch is possible, with the limitation that the bound is no longer valid for all confidence levels.

4. PAC-Bayes

We now briefly introduce the generalised Bayesian setting in machine learning, and the resulting generalisation bounds, the PAC-Bayesian bounds. PAC-Bayes is a sophisticated framework to derive new learning algorithms and obtain (often state-of-the-art) generalisation bounds, while maintaining probability distributions over hypotheses; as such, we are interested in studying how PAC-Bayes is compatible with good and cheap lunches. We refer the reader to [1,9] and the many references therein for recent surveys on PAC-Bayes including historical notes and main bounds. We focus on classical bounds from the PAC-Bayes literature, based on the empirical risk as a risk estimator—and we instantiate those bounds in two regimes matching the "expensive" and "cheap" models introduced in Section 2.

4.1. Notation

For any $f \in \mathcal{F}$, we define the empirical risk $R_N(f)$ as:

$$R_N(f) = \frac{1}{N}\sum_{i=1}^{N} \ell(f(X_i), Y_i).$$

In the following, we consider integrals over the hypotheses space $\mathcal{F}$. To keep the notation as compact as possible, we will write $\mu[g] = \int g \mathrm{d}\mu$ if μ is a measure over $\mathcal{F}$ and $g \in \mathcal{F}$ a μ-integrable function.

4.2. Generalised Bayes and PAC Bounds

The main advantage of PAC-Bayes over deterministic approaches which output single hypotheses (through optimisation of a particular criterion such as in model selection, etc.) is that the distributions allow us to capture uncertainty on hypotheses, and take into account correlations among possible hypotheses.

Denoting by ρ the posterior distribution, the quantity to control is:

$$\rho[R] = \int_{\mathcal{F}} R(f)\mathrm{d}\rho(f)$$

which is an aggregated risk over the class $\mathcal{F}$ and represents the expected risk if the predictor f is drawn from ρ for each new prediction. The distribution ρ is usually data-dependent and is referred to as a "posterior" distribution (by analogy with Bayesian statistics). We also fix a reference measure π over $\mathcal{F}$, called the "prior" (for similar reasons). We refer to [1,10] for in-depth discussions on the choice of the prior: a recent streamline of work has further investigated the choice of data-dependent priors [11–14].

The generalisation bounds associated to this setting are known as "PAC-Bayesian" bounds, where PAC stands for probably approximately correct. One important feature of PAC-Bayes bounds is that they hold true for any prior π and posterior ρ. In practice, bounds are optimised with respect to ρ and possibly π. In the following, we focus on establishing bounds for any choice of π and ρ and do not mean to optimise.

4.3. Notion of Divergence

An important notion used in PAC-Bayesian theory is the divergence between two probability distributions [see [15], for example, for a survey on divergences]. Let $\mathcal{E}$ be a measurable space and μ and ν two probability distributions on $\mathcal{E}$. Let f be a non-negative convex function defined on $\mathbb{R}_+$ such that $f(1) = 0$, we define the f-divergence between μ and ν by

$$\mathcal{D}_f(\mu, \nu) = \begin{cases} \int f\left(\frac{\mathrm{d}\mu}{\mathrm{d}\nu}\right)\mathrm{d}\nu & \text{if } \mu \ll \nu, \\ +\infty & \text{otherwise.} \end{cases}$$

Note that we also use the notation f to denote hypotheses elsewhere in the paper, but we believe the context to always be clear enough to avoid ambiguity.

Applying Jensen inequality, we have that $\mathcal{D}_f(\mu, \nu)$ is always non-negative and equal to zero if and only if $\mu = \nu$. The class of f-divergences includes many celebrated divergences, such as the Kullback–Leibler (KL) divergence, the reversed KL, the Hellinger distance, the total variation distance, χ^2-divergences, α-divergences, etc. Most PAC-Bayesian generalisation bounds involve the KL divergence.

A divergence can be thought of as a transport cost between two probability distributions. This interpretation will be useful for explaining PAC-Bayesian inequalities, where the divergence plays the role of a complexity term. In the following, we will just use two types of divergence. The first is the Kullback–Leibler divergence and corresponds to the choice $f(x) = x \log x$, which we denote it by

$$\mathrm{KL}(\mu, \nu) = \begin{cases} \int \log\left(\frac{\mathrm{d}\mu}{\mathrm{d}\nu}\right)\mathrm{d}\mu & \text{if } \mu \ll \nu, \\ +\infty & \text{otherwise.} \end{cases}$$

The second is linked to Pearson's χ^2-divergence and corresponds to the choice $f(x) = x^2 - 1$. It is referred to as $\mathcal{D}_2$:

$$\mathcal{D}_2(\mu, \nu) = \begin{cases} \int \left(\frac{\mathrm{d}\mu}{\mathrm{d}\nu}\right)^2 \mathrm{d}\nu \ - 1 & \text{if } \mu \ll \nu, \\ +\infty & \text{otherwise.} \end{cases}$$

To illustrate the behaviour of these two divergences, consider the case where μ and ν are normal distributions on $\mathbb{R}^d$.

Proposition 3. *If $\mathcal{E} = \mathbb{R}^d$, $\mu = \mathcal{N}(a, I)$, and $\nu = \mathcal{N}(0, I)$ (where I stands for the $d \times d$ identity matrix), we have*

$$\begin{cases} \mathcal{D}_2(\mu, \nu) = e^{\|a\|^2} - 1, \\ \mathrm{KL}(\mu, \nu) = \frac{1}{2}\|a\|^2. \end{cases}$$

Proof. We have:

$$\begin{cases} \mathrm{d}\mu(x) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}(x-a)^{\mathrm{T}}(x-a)\right)\mathrm{d}x, \\ \mathrm{d}\nu(x) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}x^{\mathrm{T}}x\right)\mathrm{d}x, \\ \frac{\mathrm{d}\mu}{\mathrm{d}\nu}(x) = \exp\left(-\frac{1}{2}\left[-2x^{\mathrm{T}}a + a^{\mathrm{T}}a\right]\right) = \exp(-\|a\|^2/2)\exp(x^{\mathrm{T}}a). \end{cases}$$

Then:

$$\begin{aligned} \mathcal{D}_2(\mu, \nu) &= \exp\left(-\|a\|^2\right) \int \exp\left(2x^{\mathrm{T}}a\right) \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}x^{\mathrm{T}}x\right)\mathrm{d}x - 1 \\ &= \exp\left(-\|a\|^2\right) \int \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}x^{\mathrm{T}}x + 2x^{\mathrm{T}}a\right)\mathrm{d}x - 1 \\ &= \exp\left(-\|a\|^2\right) \exp\left(2\|a\|^2\right) \int \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}(x-2a)^{\mathrm{T}}(x-2a)\right)\mathrm{d}x - 1 \\ &= e^{\|a\|^2} - 1. \end{aligned}$$

And finally:

$$\begin{aligned}\mathrm{KL}(\mu,\nu) &= \int \left(-\frac{\|a\|^2}{2} + x^{\mathrm{T}} a\right) \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}(x-a)^{\mathrm{T}}(x-a)\right) \mathrm{d}x \\ &= -\frac{\|a\|^2}{2} + \int x^{\mathrm{T}} a \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}(x-a)^{\mathrm{T}}(x-a)\right) \mathrm{d}x \\ &= -\frac{\|a\|^2}{2} + \|a\|^2 = \frac{\|a\|^2}{2}.\end{aligned}$$

□

We therefore see that the divergence $\mathcal{D}_2$ penalises much more strongly the gap between the means of both distributions than the Kullback–Leibler divergence.

The following technical lemma involving the Kullback–Leibler divergence and a change of measure from posterior to prior distribution is pivotal in the PAC-Bayes literature:

Lemma 1 ([5–16], Corollary 4.15)**.** *Let g be a measurable function $g : \mathcal{F} \mapsto \mathbb{R}$ such that $\pi[e^g]$ is finite. Let π and ρ be respectively prior and posterior measures as defined in Section 4.1. The following inequality holds:*

$$\rho[g] \leq \log \pi[e^g] + \mathrm{KL}(\rho, \pi).$$

4.4. Expensive PAC-Bayesian Bound

The first PAC-Bayesian bound we present is called "expensive PAC-Bayesian bound" in the spirit of Section 2: it is obtained under a sub-Gaussian tails assumption. More precisely, we suppose here that for any $f \in \mathcal{F}$, the distribution of the random variable $\ell(f(X), Y)$ belongs to $\mathcal{P}^{\sigma}_{\text{expensive}}$, which means

$$\log \mathbb{E}[\exp\{\lambda(\ell(f(X), Y) - R(f))\}] \leq \frac{\lambda^2\sigma^2}{2}, \quad \forall \lambda \in \mathbb{R}.$$

In this setting, we have the following bound, close to the ones obtained by [10].

Proposition 4. *Assume that for any $f \in \mathcal{F}$, $\ell(f(X), Y) \in \mathcal{P}^{\sigma}_{expensive}$. For any prior π, posterior ρ, and any $\delta \in (0,1)$, the following inequality holds true with a probability greater than $1-\delta$:*

$$\rho[R] \leq \rho[R_N] + \frac{\sigma}{\sqrt{N}} \sqrt{2\left(\log\left(\frac{1}{\delta}\right) + \mathrm{KL}(\rho, \pi)\right)}.$$

Proof. The proof is decomposed in two steps. The first leverages Lemma 1. Let λ be a positive number and apply Lemma 1 to the function $\lambda(R - R_N)$:

$$\rho[R] \leq \rho[R_N] + \frac{1}{\lambda}\left(\log \pi\left[e^{\lambda(R-R_N)}\right] + \mathrm{KL}(\rho, \pi)\right).$$

The second step is to control the deviations of $\log \pi\left[e^{\lambda(R-R_N)}\right]$. With a probability $1-\delta$, we have, by Markov's inequality

$$\pi\left[e^{\lambda(R-R_N)}\right] \leq \frac{\mathbb{E}\left[\pi\left[e^{\lambda(R-R_N)}\right]\right]}{\delta}.$$

By Fubini's theorem, we can exchange the symbols $\mathbb{E}$ and π. Using the assumption $\mathcal{P}^{\sigma}_{\text{expensive}}$, we obtain with a probability greater than $1-\delta$

$$\pi\left[e^{\lambda(R-R_N)}\right] \leq \frac{\exp\{\lambda^2\sigma^2/2N\}}{\delta}.$$

Now, putting these results together and setting

$$\lambda = \frac{\sqrt{2N\left(\log\left(\frac{1}{\delta}\right) + \mathrm{KL}(\rho,\pi)\right)}}{\sigma}$$

we obtain the desired bound. □

A PAC-Bayesian inequality is a bound which treats the complexity in the following manner:

- At first, a global complexity measure is introduced with the change of measure and is characterised by the divergence term, measuring the price to switch from π (the reference distribution) to ρ (the posterior distribution on which all inference and prediction is based);
- Next, the stochastic assumption on the data-generating distribution is used to control $\pi\left[e^{\lambda(R-R_N)}\right]$ with high probability.

4.5. Cheap PAC-Bayesian Bounds

4.5.1. Using χ^2 Divergence

The vast majority of works in the PAC-Bayesian literature focuses on an expensive model. The main reason is that it includes the situation where the loss ℓ is bounded, a common (yet debatable) assumption in machine learning. The case where $\ell(f(X,Y)$ belongs to a cheap model has attracted far less attention; recently, ref. [17] have obtained the following bound.

Proposition 5 ([17], Theorem 1)**.** *Assume that for any $f \in \mathcal{F}$, $\ell(f(X),Y) \in \mathcal{P}^{\sigma}_{cheap}$. For any prior π, posterior ρ, and any $\delta \in (0,1)$, the following inequality holds true with a probability greater than $1-\delta$*

$$\rho[R] \le \rho[R_N] + \frac{\sigma}{\sqrt{N}}\sqrt{\frac{\mathcal{D}_2(\rho,\pi)+1}{\delta}}.$$

The proof (see [17]) uses the same elementary ingredients as in the expensive case, replacing the Kullback–Leibler divergence by $\mathcal{D}_2$ and the dependence in δ moves from $\sqrt{2\log(1/\delta)}$ to $\frac{1}{\sqrt{\delta}}$. Note the correspondence between these two bounds and the confidence intervals introduced in Section 2.

4.5.2. Using Huber-Type Losses

With a different approach, ref. [18] obtained asymptotic PAC-Bayesian bounds for δ-dependent risk estimators based on the empirical mean of Huber-type influence functions. The author of [18] studied in a slightly more restrictive model than $\mathcal{P}_{\text{cheap}}$, assuming in addition that the order 3 moment of $\ell(f(X),Y)$ is bounded for $f \in \mathcal{H}$. We rephrase here Theorem 9 of [18]: with a probability greater than $1-\delta$,

$$\rho[R] \le \rho[\hat{R}_{\delta,N}] + \frac{1}{\sqrt{N}}\left(\mathrm{KL}(\rho,\pi) + \frac{\log(8\pi\sigma\delta^{-2})}{2} + \sigma + \pi^*_N(\mathcal{F}) - 1\right) + o\left(\frac{1}{N}\right),$$

where $\pi^*_N(\mathcal{F})$ is a term depending on the quality of the prior. In Remark 10, the author notes that assuming only finite moments for $\ell(f(X),Y)$, it is impossible in practice to choose a prior such that $\frac{\pi^*_N(\mathcal{F})}{\sqrt{N}}$ decreases at rate $1/\sqrt{N}$ or faster. Then, the dominant term necessarily converges at a slower rate than that of Proposition 4. However, this bounds leads to the definition of a robust PAC-Bayes estimator which proves efficient on simulated data (see Section 5 of [18]).

5. A Good Cheap Lunch: Towards a Robust PAC-Bayesian Bound?

If we take a closer look at the aforementioned PAC-Bayesian bounds from a robust statistics perspective, the following question arises: **can we obtain a PAC-Bayesian bound with a $\sqrt{\log(1/\delta)}$ dependence (possibly up to a numerical constant) in the confidence level with the cheap model?** In this section, we shed light on some structural issues. In the following, we assume the existence of $\sigma > 0$ such that for any $f \in \mathcal{F}$, $\ell(f(X), Y) \in \mathcal{P}^{\sigma}_{\text{cheap}}$.

5.1. A Necessary Condition

Let $\widehat{R}$ be an estimator of the risk (not necessarily the classical empirical risk). Here is a prototype of the inequality we are looking for: for any $\delta \in (0,1)$, with probability $1-\delta$

$$\rho[R] \leq \rho\Big[\widehat{R}\Big] + \frac{\sigma}{\sqrt{N}}\mathrm{A}(\rho, \pi, \delta),$$

where

$$\mathrm{A}(\rho, \pi, \delta) \underset{\delta \to 0}{=} \mathcal{O}\left(\sqrt{\log(1/\delta)}\right).$$

If we choose $\rho = \pi = \delta_{\{f\}}$ (Dirac mass in the single hypothesis f), the existence of such a PAC-Bayesian bound valid for all δ implies that

$$\left[\widehat{R}(f) \pm \frac{\sigma}{\sqrt{N}} \times c\sqrt{\log(1/\delta)}\right]$$

is a confidence interval for the risk $R(f)$ for any level $1-\delta$, where c is a constant.

Thus, a necessary condition for a PAC-Bayesian bound to be valid for all of the risk level δ is to have tight confidence intervals for any $f \in \mathcal{F}$.

However, as covered in Section 3, such estimators do not exist over the class $\mathcal{P}^{\sigma}_{\text{cheap}}$, and the possibility to derive a tight confidence interval is limited by the fact that the level δ must be greater that a positive constant of the form $e^{-\mathcal{O}(N)}$.

5.2. A δ-Dependent PAC-Bayesian Bound?

As a consequence, there is simply no hope for a robust PAC-Bayesian bound valid for any error threshold δ, for essentially the same reason which prevents it in the mean estimation case. The question we address now is the possibility of obtaining a robust PAC-Bayesian bound, with a dependence of magnitude $\sqrt{2\log(1/\delta)}$ (possibly up to a constant), with a possible limitation on the error threshold δ. In the following, we assume to have an estimator of the risk $\widehat{R}$ and an error threshold $\delta > 0$ such that there exists a constant $C > 0$ such that for any $f \in \mathcal{F}$,

$$\left[\widehat{R}(f) \pm \frac{\sigma}{\sqrt{N}} \times C\sqrt{\log(1/\delta)}\right]$$

is a confidence interval for $R(f)$ at level $1-\delta$. MoM is an example of such estimator. Let us stress that δ is fixed and cannot be used as a free parameter.

As seen above, a PAC-Bayesian bound proof proceeds in two steps:

- First, we use a convexity argument to control the target quantity $\rho[R - \widehat{R}]$ by an upper-bound involving a divergence term and a term of the form $g^{-1}\left(\pi\left[g(R-\widehat{R})\right]\right)$ where g is a non-negative, increasing, and convex function;
- Second, we control the term $\pi\left[g(R-\widehat{R})\right]$ in high probability, using Markov's inequality.

The first step does not require any use of a stochastic model on the data, and is always valid, regardless of whether we have a cheap or an expensive model. The second step uses the model and introduce the dependence in the error rate δ on the right-term of the bound: $g^{-1}(1/\delta)$. In the case of the "expensive bound", we had $g = \exp$, and the dependence was $\log(1/\delta)$, the final rate $\sqrt{\log(1/\delta)}$ was obtained by choosing a relevant value for λ.

Let us follow this scheme to obtain a robust PAC-Bayesian bound. The first step gives

$$\rho[R] \leq \rho[\widehat{R}] + \frac{1}{\lambda}\Big(\log \pi\Big[e^{\lambda(R-\widehat{R})}\Big] + \mathrm{KL}(\rho,\pi)\Big).$$

Our goal is now to control $\pi\Big[e^{\lambda(R-\widehat{R})}\Big]$ in high probability.

5.2.1. The Case $\pi = \delta_{\{f\}}$

Let us start with a very special case, where the prior is a Dirac mass on some hypothesis $f \in \mathcal{F}$. Then

$$\frac{1}{\lambda}\log \pi\Big[e^{\lambda(R-\widehat{R})}\Big] = R(f) - \widehat{R}(f).$$

Using how $\widehat{R}$ is defined, we can bound this quantity in the following way: with probability $1-\delta$,

$$R(f) - \widehat{R}(f) \leq \frac{\sigma}{\sqrt{N}} \times C\sqrt{\log(1/\delta)}.$$

Another way to formulate this result is to say that there exists an event $\mathcal{A}_f$ with a probability greater than $1-\delta$ such that for all $\omega \in \mathcal{A}_f$, the following holds true:

$$(R(f) - \widehat{R}(f,\omega)) \leq \frac{\sigma}{\sqrt{N}} \times C\sqrt{2\log(1/\delta)}.$$

In this example, we can control $\log \pi\Big[e^{\lambda(R-\widehat{R})}\Big]$ at the price of a maximal constraint on the choice of the posterior. Indeed, the only possible choice for ρ for the Kullback–Leibler $\mathrm{KL}(\rho,\pi)$ to make sense is $\rho = \pi = \delta_{\{f\}}$.

5.2.2. The Case $\pi = \alpha\delta_{\{f_1\}} + (1-\alpha)\delta_{\{f_2\}}$

Consider now a somewhat more sophisticated choice of prior which is a mixture of two Dirac masses in two distinct hypotheses. We do not fix the mixing proportion α and allow it to move freely between 0 and 1. The goal is to control the quantity

$$\pi\Big[e^{\lambda(R-\widehat{R})}\Big] = \alpha e^{\lambda(R(f_1)-\widehat{R}(f_1))} + (1-\alpha)e^{\lambda(R(f_2)-\widehat{R}(f_2))}.$$

More precisely, for all $\alpha \in (0,1)$, we want to find an event $\mathcal{A}_\alpha$ on which this quantity is under control. In view of the prior's structure, the only way to ensure such a control is to have $\mathcal{A}_\alpha \subset \mathcal{A}_{f_2} \cap \mathcal{A}_{f_2}$, where $\mathcal{A}_{f_1}$ (resp. $\mathcal{A}_{f_2}$) is the favourable event for the concentration of $\widehat{f_1}$ (resp. $\widehat{f_2}$) around its mean.

By the union bound, we have that with a probability greater than $1-2\delta$

$$\frac{1}{\lambda}\log \pi\Big[e^{\lambda(R-\widehat{R})}\Big] \leq \frac{\sigma}{\sqrt{N}} \times C\sqrt{\log(1/\delta)}.$$

We face a double problem here. As above, if we want the final bound to be non-vacuous, we have to ensure that $\mathrm{KL}(\rho,\pi)$ is finite, which restricts the support for the posterior to be included in the set $\{f_1, f_2\}$. In addition, the PAC-Bayesian bound holds with a probability greater than $1-2\delta$…

5.2.3. Limitation

… which hints at the fact that this will become $1-K\delta$ if the support for the prior contains K distinct hypotheses. If $K \geq 1/\delta$, the bound becomes vacuous. In particular, we cannot obtain a relevant bound using this approach in the situation where the cardinal of $\mathcal{F}$ is infinite (which is commonly the case in most PAC-Bayes works).

This limiting fact highlights that to derive PAC-Bayesian bounds, we cannot rely on the construction of confidence interval for all $R(f)$ for a fixed error threshold δ. The issue is that when we want to transfer this local property into a global one (valid for any mixture of hypotheses by the prior π), we cannot avoid a worst-case reasoning by the use of the union bound.

The established bounds in the PAC-Bayesian literature, both in cheap and expensive models, repeatedly use the fact that when we assume that for any $f \in \mathcal{F}$,

$$\log \mathbb{E}\left[e^{\lambda(R(f)-\ell(f(X),Y))}\right] \leq \frac{\lambda^2\sigma^2}{2}, \ \forall \lambda \in \mathbb{R}$$

or

$$\operatorname{var}(\ell(f(X),Y)) \leq \sigma^2,$$

we make an implicit assumption on the integrability of the tail of the distribution of $\ell(f(X),Y)$. This argument is crucial for the second step of the PAC-Bayesian proof because, by Fubini's theorem, it allows us to convert a local property (the tail distribution of each $\ell(f(X),Y)$) into a global one (the control of $\pi\left[e^{\lambda(R-R_N)}\right]$ or $\pi\left[(R-R_N))^2\right]$ in high probability).

5.3. Is That the End of the Story?

We have identified a structural limitation to derive a tight PAC-Bayesian bound in a cheap model. We make the case that we cannot replicate the PAC-Bayesian proof presented in Section 4. To conclude this section, we want to highlight the fact that, up to our knowledge, no proof of PAC-Bayesian bounds avoids these two steps (see, for example, the general presentation in [19]).

What if we try to avoid the change of the measure step and try to control directly $\rho[R]-\rho[\widehat{R}]$ in high probability? We remark that ρ can only be chosen with the information given by the observation of $\widehat{R}(f)$, where $f \in \mathcal{F}$. In particular, we cannot obtain any information of the concentration of each $\widehat{R}(f)$ around $R(f)$ as such knowledge requires to know the true risk. So, it seems that a direct control cannot avoid starting as a "worst-case" bound:

$$\rho[R]-\rho[\widehat{R}] \leq \sup_{f\in\mathcal{F}}\left\{R(f)-\widehat{R}(f)\right\}.$$

Then, we have to control $\sup_{f\in\mathcal{F}}\left\{R(f)-\widehat{R}(f)\right\}$ in high probability (see [20] for a general presentation on such controls, and [7] for the recent results in the special case where $\widehat{R}$ is a MoM estimator). However, the obtained bound will take the following prototypic form:

$$\rho[R] \leq \rho[\widehat{R}] + \text{complexity term},$$

where the complexity term does not depend on the distribution ρ. Thus, the optimisation of the right term leads to choosing ρ as the Dirac mass in $\operatorname*{arg\,min}_{f\in\mathcal{F}} \widehat{R}(f)$.

So, the overall procedure amounts to a slightly modified empirical risk minimisation (where the empirical mean is replaced with any estimator of the risk), and will not fall into the category of generalised Bayesian approaches which take into account the uncertainty on hypotheses. Pretty much all the strengths of PAC-Bayes would then be lost.

6. Conclusions

The present paper contributes a better understanding of the profound structural reasons why good cheap lunches (tight bounds under minimal assumptions) are not possible with PAC-Bayes by walking gently through elementary examples.

From a theoretical perspective, PAC-Bayesian bounds requires too strong assumptions to adapt robust statistics results (where almost good lunches can be obtained for cheap models—with the limitation that the confidence level is constrained). The second step of the proof we have shown requires us to transform a local hypothesis, a control of some moments of $\ell(f(X),Y)$, into a global one, valid for all mixture of hypotheses by the prior π. As covered above, this transformation seems impossible.

To close on a more positive note after this negative result, let us stress that even if the conciliation of PAC-Bayes and robust statistics appears challenging, we believe that the recent ideas from robust statistics could be used in practical algorithms inspired by PAC-Bayes. In particular, we leave as an avenue for future work the empirical study of PAC-Bayesian posteriors (such as the Gibbs measure defined as $\rho \propto \exp(-\gamma\widehat{R})\pi$ for any inverse temperature $\gamma > 0$) where the risk estimator is not the empirical mean (as in most PAC-Bayes works) but rather a robust estimator, such as MoM.

Author Contributions: Conceptualization, B.G. and L.P.; Formal analysis, B.G. and L.P.; Supervision, B.G.; Writing—original draft, L.P.; Writing—review & editing, B.G. and L.P. All authors have read and agreed to the published version of the manuscript.

Funding: B.G. is supported in part by the U.S. Army Research Laboratory and the U.S. Army Research Office, and by the U.K. Ministry of Defence and the U.K. Engineering and Physical Sciences Research Council (EPSRC) under grant number EP/R013616/1. B.G. acknowledges partial support from the French National Agency for Research, grants ANR-18-CE40-0016-01 and ANR-18-CE23-0015-02.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Guedj, B. A primer on PAC-Bayesian learning. *arXiv* **2019**, arXiv:1901.05353.
2. Valiant, L.G. A Theory of the Learnable. *Commun. ACM* **1984**, *27*, 1134–1142. [CrossRef]
3. Lecué, G.; Lerasle, M. Robust machine learning by median-of-means: Theory and practice. *Ann. Stat.* **2020**, *48*, 906–931. [CrossRef]
4. Devroye, L.; Györfi, L.; Lugosi, G. *A Probabilistic Theory of Pattern Recognition*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 1996; Volume 31.
5. Boucheron, S.; Lugosi, G.; Massart, P. *Concentration Inequalities: A Nonasymptotic Theory of Independence*; Oxford University Press: Oxford, UK, 2013.
6. Catoni, O. Challenging the empirical mean and empirical variance: A deviation study. *Ann. l'IHP Probabilités Stat.* **2012**, *48*, 1148–1185. [CrossRef]
7. Lerasle, M. Lecture Notes: Selected topics on robust statistical learning theory. *arXiv* **2019**, arXiv:1908.10761.
8. Devroye, L.; Lerasle, M.; Lugosi, G.; Oliveira, R.I. Sub-Gaussian mean estimators. *Ann. Stat.* **2016**, *44*, 2695–2725. [CrossRef]
9. Alquier, P. User-friendly introduction to PAC-Bayes bounds. *arXiv* **2021**, arXiv:2110.11216.
10. Catoni, O. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*; Lecture Notes-Monograph Series; IMS: Danbury, SC, USA, 2007.
11. Dziugaite, G.K.; Roy, D.M. Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data. In Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, 11–15 August 2017; Elidan, G., Kersting, K., Ihler, A.T., Eds.; AUAI Press: Montreal, QC, Canada, 2017.
12. Pérez-Ortiz, M.; Rivasplata, O.; Guedj, B.; Gleeson, M.; Zhang, J.; Shawe-Taylor, J.; Bober, M.; Kittler, J. Learning PAC-Bayes Priors for Probabilistic Neural Networks. *arXiv* **2021**, arXiv:2109.10304.
13. Pérez-Ortiz, M.; Rivasplata, O.; Shawe-Taylor, J.; Szepesvári, C. Tighter risk certificates for neural networks. *arXiv* **2020**, arXiv:2007.12911.
14. Dziugaite, G.K.; Hsu, K.; Gharbieh, W.; Arpino, G.; Roy, D. On the role of data in PAC-Bayes. In Proceedings of the 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, Virtual Event, 13–15 April 2021; Banerjee, A., Fukumizu, K., Eds.; PMLR: New York, NY, USA, 2021; Volume 130, pp. 604–612.
15. Csiszár, I.; Shields, P.C. Information theory and statistics: A tutorial. In *Foundations and Trends® in Communications and Information Theory*; Now Publishers Inc.: Norwell, MA, USA, 2004; Volume 1, pp. 417–528.
16. Csiszár, I. I-divergence geometry of probability distributions and minimization problems. *Ann. Probab.* **1975**, *3*, 146–158. [CrossRef]
17. Alquier, P.; Guedj, B. Simpler PAC-Bayesian bounds for hostile data. *Mach. Learn.* **2018**, *107*, 887–902. [CrossRef]

18. Holland, M.J. PAC-Bayes under potentially heavy tails. In *Advances in Neural Information Processing Systems 32, Proceedings of the Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019*; Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R., Eds.; Neural Information Processing Systems Foundation, Inc.: Montreal, QC, Canada, 2019; pp. 2711–2720.
19. Bégin, L.; Germain, P.; Laviolette, F.; Roy, J.F. PAC-Bayesian bounds based on the Rényi divergence. In *Artificial Intelligence and Statistics*; PMLR: New York, NY, USA, 2016; pp. 435–444.
20. Van der Vaart, A.W.; Wellner, J.A. Weak convergence. In *Weak Convergence and Empirical Processes*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 16–28.

entropy

MDPI

Article

A Scalable Bayesian Sampling Method Based on Stochastic Gradient Descent Isotropization

Giulio Franzese *, Dimitrios Milios, Maurizio Filippone and Pietro Michiardi

Data Science Department, Eurecom, 06410 Biot, France; dimitrios.milios@eurecom.fr (D.M.); maurizio.filippone@eurecom.fr (M.F.); pietro.michiardi@eurecom.fr (P.M.)
* Correspondence: giulio.franzese@eurecom.fr

Abstract: Stochastic gradient SG-based algorithms for Markov chain Monte Carlo sampling (SGMCMC) tackle large-scale Bayesian modeling problems by operating on mini-batches and injecting noise on SGsteps. The sampling properties of these algorithms are determined by user choices, such as the covariance of the injected noise and the learning rate, and by problem-specific factors, such as assumptions on the loss landscape and the covariance of SG noise. However, current SGMCMC algorithms applied to popular complex models such as Deep Nets cannot simultaneously satisfy the assumptions on loss landscapes and on the behavior of the covariance of the SG noise, while operating with the practical requirement of non-vanishing learning rates. In this work we propose a novel practical method, which makes the SG noise isotropic, using a fixed learning rate that we determine analytically. Extensive experimental validations indicate that our proposal is competitive with the state of the art on SGMCMC.

Keywords: Bayesian sampling; stochastic gradients; Monte Carlo integration

Citation: Franzese, G.; Milios, D.; Filippone, M.; Michiardi, P. A Scalable Bayesian Sampling Method Based on Stochastic Gradient Descent Isotropization. *Entropy* **2021**, *23*, 1426. https://doi.org/10.3390/e23111426

Academic Editor: Pierre Alquier

Received: 21 September 2021
Accepted: 26 October 2021
Published: 28 October 2021

1. Introduction

Stochastic gradient (SG) methods have been extensively studied as a means for MCMC-based Bayesian posterior sampling algorithms to scale to large data regimes. Variants of SG-MCMC algorithms have been studied through the lens of first [1–3] or second-order [4,5] Langevin Dynamics, which are mathematically convenient continuous-time processes that correspond to discrete-time gradient methods with and without momentum, respectively. The common traits underlying many methods from the literature can be summarized as follows: they address large data requirements using SG and mini-batching, they inject Gaussian noise throughout the algorithm execution, and they avoid the expensive Metropolis-Hasting accept/reject tests that use the whole data [1,2,4].

Despite mathematical elegance and some promising results restricted to simple models, current approaches fall short in dealing with the complexity of the loss landscape typical of popular modern machine learning models, e.g., neural networks [6,7], for which stochastic optimization poses some serious challenges [8,9].

In general, SG-MCMC algorithms inject random noise to SG descent algorithms: the covariance of such noise and the learning rate, or step-size in the stochastic differential equation simulation community, are tightly related to the assumptions on the loss landscape, which together with the SG noise, determine the sampling properties of these methods [5]. However, current SG-MCMC algorithms applied to popular complex models such as Deep Nets, cannot simultaneously satisfy the assumptions on posterior distribution geometry and on the behavior of the covariance of the SG noise, while operating with the practical requirement of non-vanishing learning rates. In this paper, in accordance with most of the Neural Network related literature, we refer to the posterior distribution geometry as loss landscape. Some recent work [10], instead, argue for fixed step sizes, but settle for variational approximations of quadratic losses. Although we are not the first to highlight these issues, including the lack of a unified notation [5], we believe that studying the role of noise in SG-MCMC algorithms has not received enough attention, and a deeper understanding is truly desirable, as it can clarify how various methods compare. Most importantly, this endeavor can suggest novel and more practical algorithms relying on fewer parameters and less restrictive assumptions.

In this work we chose a mathematical notation that emphasizes the role of noise covariances and learning rate on the behavior of SG-MCMC algorithms (Section 2). As a result, the equivalence between learning rate annealing and extremely large injected noise covariance becomes apparent, and this allows us to propose a novel practical SG-MCMC algorithm (Section 3). We derive our proposal, by first analyzing the case where we inject the smallest complementary noise such that its combined effects with the SG noise result in an isotropic noise. Thanks to this isotropic property of the noise, it is possible to deal with intricate loss surfaces typical of deep models, and produce samples from the true posterior without learning rate annealing. This, however, comes at the expense of cubic complexity matrix operations. We address such issues through a practical variant of our scheme, which employs well-known approximations to the SG noise covariance (see, e.g., [11]). The result is an algorithm that produces approximate posterior samples with a fixed, theoretically derived, learning rate. Please note that in generic Bayesian deep learning setting, none of the existing implementations of SG-MCMC methods converge to the true posterior without learning rate annealing. In contrast, our method automatically determines an appropriate learning rate through a simple estimation procedure. Furthermore, our approach can be readily applied to pre-trained models: after a "warmup" phase to compute SG noise estimates, it can efficiently perform Bayesian posterior sampling.

We evaluate SG-MCMC algorithms (Section 4) through an extensive experimental campaign, where we compare our approach to several alternatives, including Monte Carlo Dropout (MCD) [12] and Stochastic Weighted Averaging Gaussians (SWAG, [9]), which have been successfully applied to the Bayesian deep learning setting. Our results indicate that our approach offers performance that are competitive to the state of the art, according to metrics that aim at assessing the predictive accuracy and uncertainty.

2. Preliminaries and Related Work

Consider a dataset of $m-$dimensional observations $\mathcal{D} = \{\boldsymbol{U}_i\}_{i=1}^N$. Given prior $p(\boldsymbol{\theta})$ for a d-dimensional set of parameters, and a likelihood model $p(\mathcal{D}|\boldsymbol{\theta})$, the posterior is obtained by means of Bayes theorem as follows:

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathcal{D})} \tag{1}$$

where $p(\mathcal{D})$ is also known as the model evidence, defined as the integral $p(\mathcal{D}) = \int p(\mathcal{D}|\boldsymbol{\theta})\, p(\boldsymbol{\theta}) d\boldsymbol{\theta}$. Except when the prior and the likelihood function are conjugate, Equation (1) is analytically intractable [13]. However, the joint likelihood term in the numerator is typically not hard to compute; this is a key element of many MCMC algorithms, since the normalization constant $p(\mathcal{D})$ does not affect the shape of the distribution in any way other than scaling. The posterior distribution is necessary to obtain predictive distributions for new test observations $\boldsymbol{U}_*$, as:

$$p(\boldsymbol{U}_*|\mathcal{D}) = \int p(\boldsymbol{U}_*|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \tag{2}$$

We focus in particular on Monte Carlo methods to obtain an estimate of this predictive distribution, by averaging over N_{MC} samples obtained from the posterior over $\boldsymbol{\theta}$, i.e., $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta}|\mathcal{D})$

$$p(\boldsymbol{U}_*|\mathcal{D}) \approx \frac{1}{N_{\mathrm{MC}}} \sum_{i=1}^{N_{\mathrm{MC}}} p(\boldsymbol{U}_*|\boldsymbol{\theta}^{(i)}) \tag{3}$$

We develop our work by working with an unnormalized version of the logarithm of the posterior density, by expressing the negative logarithm of the joint distribution of the dataset $\mathcal{D}$ and parameters $\boldsymbol{\theta}$ as:

$$-f(\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(\boldsymbol{U}_i|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}). \tag{4}$$

For computational efficiency, we use a minibatch stochastic gradient $\boldsymbol{g}(\boldsymbol{\theta})$, which guarantees that the estimated gradient is an unbiased estimate of the true gradient $\nabla f(\boldsymbol{\theta})$, and we assume that the randomness due to the minibatch introduces a Gaussian noise:

$$\boldsymbol{g}(\boldsymbol{\theta}) \sim N(\nabla f(\boldsymbol{\theta}), 2\boldsymbol{B}(\boldsymbol{\theta})), \tag{5}$$

where the matrix $\boldsymbol{B}(\boldsymbol{\theta})$ denotes the SG noise covariance, which depends on the parametric model, the data distribution and the minibatch size.

A survey of algorithms to sample from the posterior using SG methods can be found in Ma et al. [5]. In Appendix A we report some well-known facts which are relevant for the derivations in our paper. As shown in the literature [10,14], there are structural similarities between SG-MCMC algorithms and stochastic optimization methods, and both can be used to draw samples from posterior distributions. Notice that the original goal of stochastic optimization is to find the minimum of a given cost function, and the stochasticity is introduced by sub-sampling the dataset to scale. SG-MCMC methods instead aim at sampling from a given distribution, i.e., collecting multiple values, and the stochasticity is necessary explore the whole landscape. In what follows, we use a unified notation to compare many existing algorithms in light of the role played by their noise components.

It is well-known [15–17] that stochastic gradient descent (SGD), with and without momentum, can be studied through the following stochastic differential equation (SDE), when the learning rate η is small enough (In this work we do not consider discretization errors. The reader can refer to classical SDE texts such as [18] to investigate the topic in greater depth.):

$$d\boldsymbol{z}_t = \boldsymbol{s}(\boldsymbol{z}_t)dt + \sqrt{2\eta\boldsymbol{D}(\boldsymbol{z}_t)}d\boldsymbol{W}_t. \tag{6}$$

where $\boldsymbol{s}$ is usually referred to as driving force and $\boldsymbol{D}$ as diffusion matrix We use a generic form of the SDE, with variable $\boldsymbol{z}$ instead of $\boldsymbol{\theta}$, which accommodates SGD variants, with and without momentum. By doing this, we will be able to easily cast the expression for the two cases in what follows (The operator $\nabla^\top$ applied to matrix $\boldsymbol{D}(\boldsymbol{z})$ produces a row vector whose elements are the divergences of the $\boldsymbol{D}(\boldsymbol{z})$ columns. Our notation is aligned with Chen et al. [4]).

Definition 1. *A distribution $\rho(\boldsymbol{z}) \propto \exp(-\phi(\boldsymbol{z}))$ is said to be a **stationary** distribution for the SDE of the form* (6)*, if and only if it satisfies the following Fokker-Planck equation (FPE):*

$$0 = \mathrm{Tr}\left\{\nabla\left[-\boldsymbol{s}(\boldsymbol{z})^\top\rho(\boldsymbol{z}) + \nabla^\top(\boldsymbol{D}(\boldsymbol{z})\rho(\boldsymbol{z}))\right]\right\}. \tag{7}$$

Please note that in general, the stationary distribution does not converge to the desired posterior distribution, i.e., $\phi(\boldsymbol{z}) \neq f(\boldsymbol{z})$, as shown by Chaudhari and Soatto [8]. Additionally, given an initial condition for $\boldsymbol{z}_t$, its distribution is going to converge to $\rho(\boldsymbol{z})$ only for $t \to \infty$. In practice, we observe the SDE dynamics for a finite amount of time: then, we declare that the process is approximately in the stationary regime once the potential has reached low and stable values.

Next, we briefly overview known approaches to Bayesian posterior sampling, and interpret them as variants of an SGD process, using the FPE formalism.

2.1. Gradient Methods without Momentum

The generalized updated rule of SGD, described as a discrete-time stochastic process, writes as:

$$\delta\boldsymbol{\theta}_n = -\eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})(\boldsymbol{g}(\boldsymbol{\theta}_{n-1}) + \boldsymbol{w}_n), \tag{8}$$

where $\boldsymbol{P}(\boldsymbol{\theta}_{n-1})$ is a user-defined preconditioning matrix, and $\boldsymbol{w}_n$ is a noise term, distributed as $\boldsymbol{w}_n \sim N(\mathbf{0}, 2\boldsymbol{C}(\boldsymbol{\theta}_n))$, with a user-defined covariance matrix $\boldsymbol{C}(\boldsymbol{\theta}_n)$. Then, the corresponding continuous-time SDE is [15]:

$$d\boldsymbol{\theta}_t = -\boldsymbol{P}(\boldsymbol{\theta}_t)\nabla f(\boldsymbol{\theta}_t)dt + \sqrt{2\eta \boldsymbol{P}(\boldsymbol{\theta}_t)^2 \boldsymbol{\Sigma}(\boldsymbol{\theta}_t)}d\boldsymbol{W}_t. \tag{9}$$

In this paper we use the symbol n to indicate discrete time, while t for continuous time. We denote by $\boldsymbol{C}(\boldsymbol{\theta})$ the covariance of the *injected noise* and $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ the *composite noise* covariance. Please note that $\boldsymbol{\Sigma}(\boldsymbol{\theta}_t) = \boldsymbol{B}(\boldsymbol{\theta}_t) + \boldsymbol{C}(\boldsymbol{\theta}_t)$ combines the SG and the injected noise. Notice that our choice of notation is different from the standard one, in which the starting discrete-time process is in the form $\delta\boldsymbol{\theta}_n = -\eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})(\boldsymbol{g}(\boldsymbol{\theta}_{n-1})) + \boldsymbol{w}_n$. By directly grouping the injected noise with the stochastic gradient we can better appreciate the relationship between annealing the learning rate and extremely large injected noise. Moreover, as will be explained in Section 3, this allows derivation of a new sampling algorithm.

We define the stationary distribution of the SDE in Equation (9) as $\rho(\boldsymbol{\theta}) \propto \exp(-\phi(\boldsymbol{\theta}))$. Please note that when $\boldsymbol{C} = \mathbf{0}$, the potential $\phi(\boldsymbol{\theta})$ differs from the desired posterior $f(\boldsymbol{\theta})$ [8]. The following theorem, which is an adaptation of known results in light of our formalism, states the conditions for which the *noisy* SGD converges to the true posterior distribution (proof in Appendix A).

Theorem 1. *Consider dynamics of the form* (9) *and define the stationary distribution* $\rho(\boldsymbol{\theta}) \propto \exp(-\phi(\boldsymbol{\theta}))$. *If*

$$\nabla^\top\left(\boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}\right) = \mathbf{0}^\top \quad \text{and} \quad \eta \boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}, \tag{10}$$

then $\phi(\boldsymbol{\theta}) = f(\boldsymbol{\theta})$.

Stochastic Gradient Langevin Dynamics (SGLD) [1] is a simple approach to satisfy Equation (10); it uses no preconditioning, $\boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{I}$, and sets the injected noise covariance to $\boldsymbol{C}(\boldsymbol{\theta}) = \eta^{-1}\boldsymbol{I}$. In the limit for $\eta \to 0$, it holds that $\boldsymbol{\Sigma}(\boldsymbol{\theta}) = \boldsymbol{B}(\boldsymbol{\theta}) + \eta^{-1}\boldsymbol{I} \simeq \eta^{-1}\boldsymbol{I}$. Then, $\nabla^\top\left(\boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}\right) = \eta\nabla^\top \boldsymbol{I} = \mathbf{0}^\top$, and $\eta\boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}$. Although SGLD succeeds in (asymptotically) generating samples from the true posterior, its mixing rate is unnecessarily slow, due to the extremely small learning rate [2].

An extension to SGLD is Stochastic Gradient Fisher Scoring (SGFS) [2], which can be tuned to switch between sampling from an approximate posterior, using a non-vanishing learning rate, and the true posterior, by annealing the learning rate to zero. SGFS uses preconditioning, $\boldsymbol{P}(\boldsymbol{\theta}) \propto \boldsymbol{B}(\boldsymbol{\theta})^{-1}$. In practice, however, $\boldsymbol{B}(\boldsymbol{\theta})$ is ill conditioned for complex models such as deep neural networks. Then, many of its eigenvalues are almost zero [8], and computing $\boldsymbol{B}(\boldsymbol{\theta})^{-1}$ is problematic. An in-depth analysis of SGFS reveals that conditions (10) would be met with a non-vanishing learning rate only if, at convergence, $\nabla^\top(\boldsymbol{B}(\boldsymbol{\theta})^{-1}) = \mathbf{0}^\top$, which would be trivially true if $\boldsymbol{B}(\boldsymbol{\theta})$ was constant. However, recent work [6,7] suggest that this condition is difficult to justify for deep neural networks.

The Stochastic Gradient Riemannian Langevin Dynamics (SGRLD) algorithm [3] extends SGFS to the setting in which $\nabla^\top(\boldsymbol{B}(\boldsymbol{\theta})^{-1}) \neq \mathbf{0}^\top$. The process dynamic is adjusted by adding the term $\nabla^\top(\boldsymbol{B}(\boldsymbol{\theta})^{-1})$. However, the term $\nabla^\top(\boldsymbol{B}(\boldsymbol{\theta})^{-1})$ has not a clear estimation procedure, restricting SGRLD to cases where it can be computed analytically.

The work by [10] investigates constant-rate SGD (with no injected noise), and determines analytically the learning rate and preconditioning that minimize the Kullback–Leibler (KL) divergence between an approximation and the true posterior. Moreover, it shows that the preconditioning used in SGFS is optimal, in the sense that it converges to the true posterior, when $\boldsymbol{B}(\boldsymbol{\theta})$ is constant and the true posterior has a quadratic form.

In summary, to claim convergence to the true posterior distribution, existing approaches require either vanishing learning rates or assumptions on the SG noise covariance that are difficult to verify in practice, especially when considering deep models. We instead propose a novel practical method that induces isotropic SG noise and thus satisfies Theorem 1. We determine analytically a fixed learning rate, and we require weaker assumptions on the loss shape.

2.2. Gradient Methods with Momentum

Momentum-corrected methods emerge as a natural extension to SGD approaches. The general set of update equations for (discrete-time) momentum-based algorithms is:

$$\begin{cases} \delta\boldsymbol{\theta}_n = \eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})\boldsymbol{M}^{-1}\boldsymbol{r}_{n-1} \\ \delta\boldsymbol{r}_n = -\eta \boldsymbol{A}(\boldsymbol{\theta}_{n-1})\boldsymbol{M}^{-1}\boldsymbol{r}_{n-1} - \eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})(\boldsymbol{g}(\boldsymbol{\theta}_{n-1}) + \boldsymbol{w}_n), \end{cases}$$

where $\boldsymbol{P}(\boldsymbol{\theta}_{n-1})$ is a preconditioning matrix, M is the mass matrix and $\boldsymbol{A}(\boldsymbol{\theta}_{n-1})$ is the friction matrix, as shown by [4,19]. As with the first order counterpart, the noise term is distributed as $\boldsymbol{w}_n \sim N(\boldsymbol{0}, 2\boldsymbol{C}(\boldsymbol{\theta}_n)))$. Then, the SDE to describe continuous-time system dynamics is:

$$\begin{cases} d\boldsymbol{\theta}_t = \boldsymbol{P}(\boldsymbol{\theta}_t)\boldsymbol{M}^{-1}\boldsymbol{r}_t dt \\ d\boldsymbol{r}_t = -(A(\boldsymbol{\theta}_t)\boldsymbol{M}^{-1}r_t + \boldsymbol{P}(\boldsymbol{\theta}_t)\nabla f(\boldsymbol{\theta}_t))dt + \sqrt{2\eta \boldsymbol{P}(\boldsymbol{\theta}_t)^2\boldsymbol{\Sigma}(\boldsymbol{\theta}_t)}d\boldsymbol{W}_t. \end{cases} \tag{11}$$

where $\boldsymbol{P}(\boldsymbol{\theta}_t)^2 = \boldsymbol{P}(\boldsymbol{\theta}_t)\boldsymbol{P}(\boldsymbol{\theta}_t)$ and we assume $\boldsymbol{P}(\boldsymbol{\theta}_t)$ to be symmetric. The theorem hereafter describes the conditions for which noisy SGD with momentum converges to the true posterior distribution (Appendix A).

Theorem 2. *Consider dynamics of the form* (11) *and define the stationary distribution for $\boldsymbol{\theta}_t$ as $\rho(\boldsymbol{\theta}) \propto \exp(-\phi(\boldsymbol{\theta}))$. If*

$$\nabla^\top \boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{0}^\top \quad \textit{and} \quad \boldsymbol{A}(\boldsymbol{\theta}) = \eta \boldsymbol{P}(\boldsymbol{\theta})^2\boldsymbol{\Sigma}(\boldsymbol{\theta}), \tag{12}$$

then $\phi(\boldsymbol{\theta}) = f(\boldsymbol{\theta})$.

In the naive case, where $\boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{I}, \boldsymbol{A}(\boldsymbol{\theta}) = \boldsymbol{0}, \boldsymbol{C}(\boldsymbol{\theta}) = \boldsymbol{0}$, Equation (12) are not satisfied and the stationary distribution does not correspond to the true posterior [4]. To generate samples from the true posterior it is sufficient to set $\boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{I}, \boldsymbol{A}(\boldsymbol{\theta}) = \eta\boldsymbol{B}(\boldsymbol{\theta}), \boldsymbol{C}(\boldsymbol{\theta}) = \boldsymbol{0}$ (as in Equation (9) in [4]).

Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) [4] suggests that estimating $\boldsymbol{B}(\boldsymbol{\theta})$ can be costly. Hence, the injected noise $\boldsymbol{C}(\boldsymbol{\theta})$ is chosen such that $\boldsymbol{C}(\boldsymbol{\theta}) = \eta^{-1}\boldsymbol{A}(\boldsymbol{\theta})$, where $\boldsymbol{A}(\boldsymbol{\theta})$ is user-defined. When $\eta \to 0$, the following approximation holds: $\boldsymbol{\Sigma}(\boldsymbol{\theta}) \simeq \boldsymbol{C}(\boldsymbol{\theta})$. It is then trivial to check that conditions (12) hold without the need for explicitly estimating $\boldsymbol{B}(\boldsymbol{\theta})$. A further practical reason to avoid setting $\boldsymbol{A}(\boldsymbol{\theta}) = \eta\boldsymbol{B}(\boldsymbol{\theta})$ is that the computational cost for the operation $\boldsymbol{A}(\boldsymbol{\theta}_{n-1})\boldsymbol{M}^{-1}\boldsymbol{r}_{n-1}$ has $\mathcal{O}(D^2)$ complexity, whereas if $\boldsymbol{C}(\boldsymbol{\theta})$ is diagonal, this is reduced to $\mathcal{O}(D)$. This, however, severely slows down the sampling process.

Stochastic Gradient Riemannian Hamiltonian Monte Carlo (SGRHMC) is an extension to SGHMC [5]), which considers a generic, space-varying preconditioning matrix $\boldsymbol{P}(\boldsymbol{\theta})$ derived from information geometric arguments [20]. SGRHMC suggests setting $\boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{G}(\boldsymbol{\theta})^{-\frac{1}{2}}$, where $\boldsymbol{G}(\boldsymbol{\theta})$ is the Fisher Information matrix. To meet the requirement $\nabla^\top \boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{0}^\top$, it includes a correction term, $-\nabla^\top \boldsymbol{P}(\boldsymbol{\theta})$. The injected noise is set to $\boldsymbol{C}(\boldsymbol{\theta}) = \eta^{-1}\boldsymbol{I} - \boldsymbol{B}(\boldsymbol{\theta})$, consequently $\boldsymbol{\Sigma} = \eta^{-1}\boldsymbol{I}$, and the friction matrix is set to $\boldsymbol{A}(\boldsymbol{\theta}) = \boldsymbol{P}(\boldsymbol{\theta})^2$. With all these choices, Theorem 2 is satisfied. Although appealing, the main drawbacks of this method are the need for an analytical expression of $\nabla^\top \boldsymbol{P}(\boldsymbol{\theta})$, and the assumption for $\boldsymbol{B}(\boldsymbol{\theta})$ to be known.

From a practical standpoint, momentum-based methods suffer from the requirement to tune many hyperparameters, including the learning rate, and the parameters that govern the simulation of a second-order Langevin dynamics.

The method we propose in this work can be applied to momentum-based algorithms; in this case, it could be viewed as an extension of the work in [11], albeit addressing the complex loss landscapes typical of deep neural networks. However, we leave this avenue of research for future work.

3. Sampling by Layer-Wise Isotropization

We present a simple and practical approach to inject noise to SGD iterates to perform Bayesian posterior sampling. Our goal is to sample from the true posterior distribution (or approximations thereof) using a *constant* learning rate, and to rely on more lenient assumptions about the shape of the loss landscape that characterize deep models, compared to previous works. In general, in modern machine learning applications, we deal with multi-layer neural networks [21]. We exploit the natural subdivision of the parameters of these architecture into different layers to propose a practical sampling scheme

Careful inspection of Theorem 1 reveals that the matrices $\boldsymbol{P}(\boldsymbol{\theta}), \boldsymbol{\Sigma}(\boldsymbol{\theta})$ are instrumental in determining the convergence properties of SG methods to the true posterior. Therefore, we consider the constructive approach of *designing* $\eta \boldsymbol{P}(\boldsymbol{\theta})$ to obtain a sampling scheme that meets our goals; we set $\eta \boldsymbol{P}(\boldsymbol{\theta})$ to be a constant, diagonal matrix which we constrain to be layer-wise uniform:

$$\eta \boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{\Lambda}^{-1} = \text{diag}([\underbrace{\lambda^{(1)}, \ldots, \lambda^{(1)}}_{\text{layer } 1}, \ldots, \underbrace{\lambda^{(N_l)}, \ldots \lambda^{(N_l)}}_{\text{layer } N_l}])^{-1}. \tag{13}$$

By properly selecting the set of parameters $\{\lambda^i\}$ we can achieve the simultaneous result of non-vanishing learning rate and well-conditioned preconditioning matrix. This implies a layer-wise learning rate $\eta^{(p)} = \frac{1}{\lambda^{(p)}}$ for the p-th layer, without further preconditioning.

We can now prove (see Appendix B), as a corollary to Theorem 1, that our design choices can guarantee convergence to the true posterior distribution.

Corollary 1. *(Theorem 1) Consider dynamics of the form (9) and define the stationary distribution $\rho(\boldsymbol{\theta}) \propto \exp(-\phi(\boldsymbol{\theta}))$. If $\eta \boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{\Lambda}^{-1}$ as in (13), $\boldsymbol{C}(\boldsymbol{\theta}) = \boldsymbol{\Lambda} - \boldsymbol{B}(\boldsymbol{\theta})$ and $\boldsymbol{C}(\boldsymbol{\theta}) \succ 0 \quad \forall \boldsymbol{\theta}$, then $\phi(\boldsymbol{\theta}) = f(\boldsymbol{\theta})$.*

If aforementioned conditions are satisfied, it is in fact simple to show that the relevant matrices satisfy the conditions in Equation (10). The covariance matrix of the composite noise is said to be *isotropic* within the layers of (deep) models. In fact, $\boldsymbol{\Sigma}(\boldsymbol{\theta}) = \boldsymbol{C}(\boldsymbol{\theta}) + \boldsymbol{B}(\boldsymbol{\theta}) = \text{diag}\left(\left[\lambda^{(1)}, \ldots, \lambda^{(1)}, \ldots, \lambda^{(N_l)}, \ldots \lambda^{(N_l)}\right]\right)$. From a practical point of view, we choose $\boldsymbol{\Lambda}$ to be, among all valid matrices satisfying $\boldsymbol{\Lambda} - \boldsymbol{B}(\boldsymbol{\theta}) \succ 0$, the smallest (the one with the smallest λ's). Indeed, larger $\boldsymbol{\Lambda}$ induce a smaller learning rate, thus unnecessarily reducing sampling speed.

Now, let us consider an ideal case, in which we assume the SG noise covariance $\boldsymbol{B}(\boldsymbol{\theta})$ and $\boldsymbol{\Lambda}$ to be known in advance. The procedure described in Algorithm 1 illustrates a naive SG method that uses the *injected noise* covariance $\boldsymbol{C}(\boldsymbol{\theta})$ to sample from the true posterior.

Algorithm 1 Idealized posterior sampling

```
{Initialization: $\boldsymbol{\theta}_0$}
SAMPLE $(\boldsymbol{\theta}_0, \boldsymbol{B}(\boldsymbol{\theta}), \boldsymbol{\Lambda})$:
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_0$
loop
  $\boldsymbol{g} = \nabla \tilde{f}(\boldsymbol{\theta})$
  $\boldsymbol{n} \sim N(0, \boldsymbol{I})$
  $\boldsymbol{C}(\boldsymbol{\theta})^{1/2} \leftarrow (\boldsymbol{\Sigma} - \boldsymbol{B}(\boldsymbol{\theta}))^{1/2}$
  $\boldsymbol{g} \leftarrow \boldsymbol{\Sigma}^{-1}(\boldsymbol{g} + \sqrt{2}\boldsymbol{C}(\boldsymbol{\theta})^{1/2}\boldsymbol{n})$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \boldsymbol{g}$
end loop
```

This deceivingly simple procedure generate samples from the true posterior, with a non-vanishing learning rate, as shown earlier. However, it cannot be used in practice as $\boldsymbol{B}(\boldsymbol{\theta})$ and $\boldsymbol{\Lambda}$ are unknown. Furthermore, the algorithm requires computationally expensive operations, i.e., to compute $(\boldsymbol{\Sigma} - \boldsymbol{B}(\boldsymbol{\theta}))^{\frac{1}{2}}$, which requires $\mathcal{O}(d^3)$ operations, and $\boldsymbol{C}(\boldsymbol{\theta})^{\frac{1}{2}}$, which costs $\mathcal{O}(d^2)$ multiplications.

Next, we describe a practical variant of our approach, where we use approximations at the expense of generating samples from the true posterior distribution. We note that [10] suggest exploring a related preconditioning, but do not develop this path in their work. Moreover, the proposed method shares similarities with a scheme proposed in [22] although the analysis we perform here is different.

3.1. A Practical Method: Isotropic SGD

To render the idealized sampling method practical, it is necessary to consider some additional assumptions. As we explain at the end of this section, the assumptions that follow are less strict than other approaches in the literature.

Assumption 1. *The* SG *noise covariance $\boldsymbol{B}(\boldsymbol{\theta})$ can be approximated with a diagonal matrix, i.e., $\boldsymbol{B}(\boldsymbol{\theta}) = \operatorname{diag}(\boldsymbol{b}(\boldsymbol{\theta}))$.*

Assumption 2. *The signal-to-noise ratio (SNR) of a gradient is small enough such that in the stationary regime, the second-order moment of the gradient is a good estimate of the true variance. Hence, combining with Assumption 1, $\boldsymbol{b}(\boldsymbol{\theta}) \simeq \frac{\mathrm{E}[\boldsymbol{g}(\boldsymbol{\theta}) \odot \boldsymbol{g}(\boldsymbol{\theta})]}{2}$, where $\odot$ indicates the element-wise product.*

Assumption 3. *The sum of the variances of noise components, layer by layer, can be assumed to constant in the stationary regime. Then, $\beta^{(p)} = \sum_{j \in I_p} b_j(\boldsymbol{\theta})$, where I_p is the set of indices of parameters belonging to p_{th} layer.*

The diagonal covariance assumption (i.e., Assumption 1) is common in other works, such as [2,11]. The small signal-to-noise ratio as stated in Assumption 2 is in line with recent studies, such as [11,23]. Assumption 3 is similar to those appeared in earlier work, such as [24]. Please note that Assumptions 2 and 3 must hold in the stationary regime when the process reaches the bottom valley of the loss landscape. The matrix $(b(\boldsymbol{\theta}))$ has been associated in the literature with the *empirical* Fisher information matrix [2,25]. As we do not consider this matrix for preconditioning purposes, we do not further investigate this connection.

Given our assumptions, and our design choices, it is then possible to show (see Appendix B) that the optimal (i.e., the smallest possible) $\boldsymbol{\Lambda} = \left[\lambda^{(1)}, \ldots, \lambda^{(1)}, \ldots, \lambda^{(N_l)}, \ldots \lambda^{(N_l)}\right]$ satisfying Corollary 1 can be obtained as $\lambda^{(p)} = \beta^{(p)}$. Please note that we do not assume $\boldsymbol{B}(\boldsymbol{\theta})$ to be known, but use a simple procedure to estimate its components by computing: $\lambda^{(p)} = \sum_{j \in I_p} b_j(\boldsymbol{\theta}) = \frac{||\boldsymbol{g}^{(p)}(\boldsymbol{\theta})||^2}{2}$, where $\boldsymbol{g}^{(p)}(\boldsymbol{\theta})$ is the portion of stochastic gradient corresponding to the p-th layer. Then, the composite noise matrix $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}$ is a layer-wise isotropic covariance matrix, which inspires the name of our proposed method as *Isotropic* SGD (I-SGD).

The practical implementation of I-SGD is shown in Algorithm 2. The advantage of I-SGD is that it can either be used to obtain posterior samples starting from a pre-trained model, or do so by training a model from scratch. In either case, the estimates of $\boldsymbol{B}(\boldsymbol{\theta})$ are used to compute $\boldsymbol{\Lambda}$, as discussed above. An important consideration is that once all $\lambda^{(i)}$ have been estimated, the learning rate, layer by layer, is determined *automatically*. In fact, for the p-th layer, the learning rate is: $\eta^{(p)} = {\lambda^{(p)}}^{-1}$. A simpler approach is to use a unique learning rate for all layers, where the equivalent λ is the sum of all $\lambda^{(p)}$.

Algorithm 2 I-SGD: practical posterior sampling

```
SAMPLE ($\boldsymbol{\theta}_0$):
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_0$
loop
  $\boldsymbol{g} = \nabla \tilde{f}(\boldsymbol{\theta})$
  for $p \leftarrow 1$ to $N_l$ do
    $\boldsymbol{n} \sim N(\boldsymbol{0}, \boldsymbol{I})$
    $\boldsymbol{C}(\boldsymbol{\theta})^{1/2} \leftarrow \left(\lambda^{(p)} - (1/2)\left(\boldsymbol{g}^{(p)} \odot \boldsymbol{g}^{(p)}\right)\right)$
    $\boldsymbol{g}^{(p)} \leftarrow 1/\lambda^{(p)}\left(\boldsymbol{g}^{(p)} + \sqrt{2}\boldsymbol{C}(\boldsymbol{\theta})^{1/2}\boldsymbol{n}\right)$
  end for
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \boldsymbol{g}$
end loop
```

A Remark on Convergence

In summary, I-SGD is a practical method to perform approximate Bayesian posterior sampling, backed up by solid theoretical foundations. Our assumptions, which are at the origin of the approximate nature of I-SGD, are less strict than those used in the literature of SG-MCMC methods. More precisely, the theory behind I-SGD can explain convergence to the true posterior with a non-vanishing learning rate in the particular case when Assumption 1 holds and the estimation of $\boldsymbol{B}(\boldsymbol{\theta})$ is perfect. Even with perfect estimates, this is not the case for SGFS, which requires the correction term $\nabla^\top \boldsymbol{B}(\boldsymbol{\theta})^{-1} = 0$. Additionally, both SGRLD and SGRHMC are more demanding than I-SGD because they require computing $\nabla^\top \boldsymbol{B}(\boldsymbol{\theta})^{-1}$, for which an estimation procedure is elusive. Finally, the method by Springenberg et al. [11] needs a *constant*, diagonal $\boldsymbol{B}(\boldsymbol{\theta})$, a condition that does not necessarily hold for deep models.

3.2. Computational Cost

The computational cost of I-SGD is as follows. As with [4], we define the cost of computing a gradient minibatch as $C_g(N_b, d)$. Thanks to Assumptions 1 and 2, the computational cost for estimating the noise covariance scales as $\mathcal{O}(d)$ multiplications. The computational cost of generating random samples with the desired covariance scales as $\mathcal{O}(d)$ square roots and $\mathcal{O}(d)$ multiplications (without considering the cost of generating random numbers). The overall cost of our method is the sum of the above terms. Notice that the cost of estimating the noise covariance does not depend on the minibatch size N_b. We would like to stress that in many modern models, the real computational bottleneck is the backward propagation for the computation of the gradients. As all the SG-MCMC methods considered in this work require one gradient evaluation per step, the different methods have in practice the same complexity.

The space complexity of I-SGD is the same as SGHMC,SGFS and variants: it scales as $\mathcal{O}(N_{\text{sam}}d)$, where N_{sam} is the number of posterior samples.

4. Experiments

The empirical analysis of our method, and its comparison to alternative approaches from the literature, is organized as follows. First, we proceed with a validation of I-SGD using the standard UCI datasets [26] and a shallow neural network. Then we move to the case of deeper models: we begin with a simple CNN used on the MNIST [27] dataset, then move to the standard RESNET-18 [28] deep network using the CIFAR-10 [29] dataset.

We compare I-SGD to other Bayesian sampling methods such as SGHMC [4], SGLD [2], and to alternative approaches to approximate Bayesian inference, including MCD [12], SWAG [9] and VSGD [10]. In general, our result indicates that: (1) I-SGD achieves similar or superior performance regarding competitors, when measuring uncertainty quantification, even with simple datasets and models; (2) I-SGD is simple to tune, when compared to alternatives; (3) I-SGD is competitive when used for deep Bayesian modeling, even when compared to standard methods used in the literature. In particular, the proposed method shares some of the strengths of VSGD, such as learning rates determined automatically and the simplicity of SGLD. Appendix B includes additional implementation details on I-SGD. Appendix C presents detailed configurations of all methods we compare, and additional experimental results.

4.1. A Disclaimer on Performance Characterization

It is important to stress a detail on the analysis of the experimental campaign. The discussion is usually focused on the goodness of the various methods for representing the true posterior distribution. Different methods can or cannot claim convergence to the true posterior according to certain assumptions and the nature of the hyperparameters. In the experimental validation of the results, however, we do not have access to the form of the true posterior as it is exactly the problem we are trying to solve. The practical solution adopted is to compare the different methods in terms of *proxy* metrics evaluated on the test sets, such as the accuracy and uncertainty metrics. Being better in terms of these performance metrics does not imply that the sampling method is better at approximating the posterior distribution, and outperforming competitors in terms of these metric do not provide sufficient information about the intrinsic quality of the sampling scheme.

4.2. Regression Tasks, with Simple Models

We consider several regression tasks defined on the UCI datasets. We use a simple neural network configuration with two fully connected layers and a ReLU activation function; the hidden layer includes 50 units. In this set of experiments, we use the following metrics: the root mean square error (RMSE) to judge the model predictive performance and the mean negative log-likelihood (MNLL) as a proxy for uncertainty quantification. We note that the task of tuning our competitors was far from trivial. We used our own version of SGHMC, based on [11], to ensure a proper understanding of the implementation internals, and we proceeded with a tuning process to find appropriate values for the numerous hyperparameters. In this set of experiments, we omit results for SWAG, which we keep for more involved scenarios.

Tables 1 and 2 report a complete overview of our results, for a selection of UCI datasets. For each method and each dataset, we also included how many out of the 10 splits considered failed to converge, indicated as $F = \ldots$. As explained in Appendix C we implemented a temperature scaled version of VSGD. A clear picture emerges from this first set of experiments: while for the RMSE the performance is similar for different methods, for the MNLL averaging over multiple samples clearly improves the uncertainty quantification capabilities. SGHMC is in many cases better than alternatives, considering however the standard deviation of the results it is difficult to claim clear superiority of one method over the others.

Table 1. RMSE results for regression on UCI datasets.

Method	WINE	PROTEIN	NAVAL	KIN8NM	POWER	BOSTON
SGLD	0.759 ± 0.07	5.687 ± 0.05	0.007 ± 0.00 (F = 6.000)	0.171 ± 0.07 (F = 3.000)	11.753 ± 3.25	9.602 ± 2.06
I-SGD	0.635 ± 0.05	4.699 ± 0.03	0.001 ± 0.00	0.079 ± 0.00	4.320 ± 0.13	3.703 ± 1.19
Baseline	0.641 ± 0.05	4.733 ± 0.05	0.001 ± 0.00	0.080 ± 0.00	4.354 ± 0.12	3.705 ± 1.19
VSGD	0.635 ± 0.05	4.699 ± 0.03	0.001 ± 0.00	0.079 ± 0.00	4.325 ± 0.13	3.588 ± 1.06 (F = 1.000)
SGHMC	0.628 ± 0.04	4.712 ± 0.03	0.000 ± 0.00 (F = 2.000)	0.076 ± 0.00 (F = 1.000)	4.310 ± 0.14	3.659 ± 1.24
SGLD T	0.752 ± 0.07	5.673 ± 0.04	0.007 ± 0.00 (F = 6.000)	0.169 ± 0.07 (F = 3.000)	11.351 ± 3.02	9.417 ± 2.07
DROP	0.637 ± 0.04	4.968 ± 0.05	0.003 ± 0.00	0.139 ± 0.01	4.531 ± 0.16	3.803 ± 1.26
SGHMC T	0.628 ± 0.04	4.684 ± 0.03	0.000 ± 0.00 (F = 6.000)	0.076 ± 0.00	4.326 ± 0.13	3.692 ± 1.19

Table 2. MNLL results for regression on UCI datasets.

Method	WINE	PROTEIN	NAVAL	KIN8NM	POWER	BOSTON
SGLD	1.546 ± 0.25	5.604 ± 0.08	-1.751 ± 0.28 (F = 6.000)	5.140 ± 7.05 (F=3.000)	8.429 ± 3.14	30.386 ± 15.77
I-SGD	1.129 ± 0.15	4.371 ± 0.03	-2.466 ± 1.12	-0.460 ± 0.65	3.122 ± 0.07	9.799 ± 5.69
Baseline	1.182 ± 0.03	3.964 ± 0.04	0.920 ± 0.00	0.924 ± 0.00	3.071 ± 0.06	5.421 ± 2.73
VSGD	1.128 ± 0.15	4.371 ± 0.03	-2.466 ± 1.12	-0.480 ± 0.65	3.088 ± 0.06	8.413 ± 5.89 (F = 1.000)
SGHMC	1.041 ± 0.12	4.142 ± 0.02	-2.763 ± 1.33 (F = 2.000)	-0.798 ± 0.39 (F = 1.000)	2.924 ± 0.04	3.097 ± 0.83
SGLD T	1.526 ± 0.24	5.591 ± 0.07	-1.752 ± 0.28 (F = 6.000)	5.118 ± 7.06 (F = 3.000)	8.288 ± 3.04	33.212 ± 19.69
DROP	1.065 ± 0.12	4.218 ± 0.06	-2.322 ± 0.75	-0.086 ± 0.41	2.941 ± 0.04	3.989 ± 1.23
SGHMC T	1.104 ± 0.14	4.191 ± 0.02	-2.966 ± 1.89 (F = 6.000)	-0.756 ± 0.42	3.116 ± 0.07	9.826 ± 5.72

4.3. Classification Tasks, with Deeper Models

Next, we compare I-SGD against competitors on image classification tasks. First, we use the MNIST dataset, and a simple LeNet-5 CNN [30]. All methods are compared based on the test accuracy ACC,MNLL and the expected calibration error (ECE, [31]). Additionally, at test time, we carry out predictions on both MNIST and NOT-MNIST; the latter is a dataset equivalent to MNIST , but it represents letters rather than numbers. (http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html, accessed on 24 October 2021) This experimental setup is often used to check whether the entropy of the predictions on NOT-MNIST is higher than the entropy of the predictions on MNIST (the entropy of the output of an N_{cl} classes classifier, represented by the vector $\mathbf{p}$, is defined as $-\sum_{i=1}^{N_{cl}} p_i \log p_i$).

Table 3 indicates that all methods are essentially equivalent in terms of accuracy and MNLL. We consider, together with the classical in and out of distribution entropies the regions of convergence (ROCS) diagrams comparing detection of out of distribution samples and false alarms when using as test statistic the entropy. Results, reported in Figure 1, clearly shows that: (1) collecting multiple samples improve the uncertainty quantification capabilities (2) I-SGD is competitive (but not the best scheme) and importantly outperform the closest approach to ours, i.e., VSGD. The experimental results show that I-SGD improves the quality of the BASELINE model with respect to all metrics. To test whether the improvements are due just to "*additional training*" or are intrinsically due to the Bayesian averaging properties, we do consider alternative deterministic baselines (details in Appendix C). For this set of experiments the best performing is BASELINE R. As can be appreciated by comparing Table 3 and Figure 1, while it is possible to increase the classical metrics, I-SGD (and other methods) still outperform by a large margin the baselines in terms of detection of out of distribution samples.

Table 3. Results for classification on MNIST dataset.

Method	ACC	MNLL	Mean H_0	ECE	Mean H_1	Failed
I-SGD	9916.3333 ± 2.8674	263.5311 ± 16.3600	0.0368 ± 0.0019	0.0491 ± 0.0003	0.4558 ± 0.0591	0.0000
SGHMC	9930.6667 ± 2.4944	268.2559 ± 6.8172	0.0593 ± 0.0018	0.0531 ± 0.0003	1.0369 ± 0.0346	0.0000
DROP	9912.6667 ± 6.0185	362.8973 ± 24.8881	0.0960 ± 0.0090	0.0541 ± 0.0011	0.5507 ± 0.0577	0.0000
BASELINE	9886.6667 ± 11.0252	352.6640 ± 20.8622	0.0353 ± 0.0058	0.0468 ± 0.0001	0.0019 ± 0.0003	0.0000
BASELINE r	9919.0000 ± 9.4163	242.7644 ± 17.0736	0.0303 ± 0.0001	0.0482 ± 0.0006	0.0021 ± 0.0002	0.0000
SWAG	9917.0000 ± 2.8284	308.8182 ± 20.0979	0.0675 ± 0.0108	0.0524 ± 0.0011	0.3953 ± 0.0442	0.0000
SGLD	9927.0000 ± 1.0000	279.7685 ± 16.6563	0.0556 ± 0.0034	0.0531 ± 0.0004	1.3032 ± 0.1942	1.0000
VSGD	9927.3333 ± 6.7987	225.3725 ± 16.3739	0.0274 ± 0.0008	0.0481 ± 0.0005	0.0414 ± 0.0070	0.0000
I-SGD T	9915.6667 ± 0.9428	255.9641 ± 12.8051	0.0289 ± 0.0014	0.0478 ± 0.0002	0.0284 ± 0.0122	0.0000
SGHMC T	9937.0000 ± 0.0000	231.5332 ± 0.0000	0.0434 ± 0.0000	0.0518 ± 0.0000	0.4623 ± 0.0000	2.0000

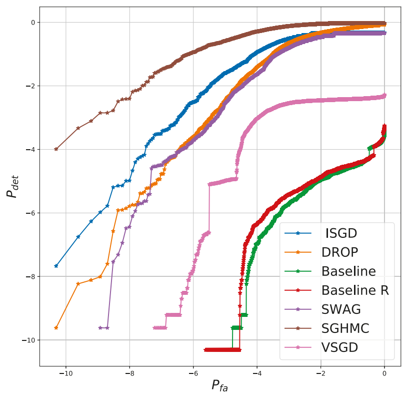


Figure 1. Detection/False alarm diagrams for different methods.

We now move on to a classical image classification problem with deep convolutional networks, whereby we use the CIFAR10 dataset, and the RESNET-18 network architecture. For this set of experiments, we compare I-SGD, SGHMC, SWAG, and VSGD using again test accuracy and MNLL, which we report in Table 4. As usual, we compare the results against the baseline of the individual network resulting from the pre-training phase. Results are obtained averaging over three independent seeds. Notice, as expanded in Appendix C that for SWAG we do consider two variants: the Bayesian correct one (SWAG) and a second variant that has better performance (SWAG wd). We stress again, as highlighted in Section 4.1

that not always goodness of approximation of the posterior and performance correlate positively. Additionally in this case, we found I-SGD to be competitive with other methods and superior to the baseline. Among the competitors, we found I-SGD to the easiest to tune, given the feature of a fixed learning rate informed by theoretical considerations; we believe that this is an important aspect to consider for a wide adoption of our proposal by practitioners.

Table 4. Results for classification on CIFAR10 10 dataset.

Method	ACC	MNLL	mean H_0	ECE
I-SGD	8591.3333 ± 17.4611	4393.3557 ± 107.0878	0.6107 ± 0.0337	0.0731 ± 0.0075
SGHMC	8634.6667 ± 5.1854	4357.8998 ± 11.2722	0.6300 ± 0.0023	0.0819 ± 0.0017
SWAG wd	8740.6667 ± 35.5653	3931.9900 ± 45.6605	0.4130 ± 0.0066	0.0275 ± 0.0015
SWAG	8061.0000 ± 11.4310	5903.2605 ± 62.8167	0.5308 ± 0.0135	0.0163 ± 0.0019
BASELINE	8273.3333 ± 26.7872	8050.4467 ± 109.9864	0.2250 ± 0.0005	0.0809 ± 0.0020
VSGD	8255.6667 ± 24.1155	8919.8062 ± 106.3571	0.1761 ± 0.0078	0.0905 ± 0.0020

5. Conclusions

SG methods allowed Bayesian posterior sampling algorithms, such as MCMC, to regain relevance in an age when datasets have reached extremely large sizes. However, despite mathematical elegance and promising results, current approaches from the literature are restricted to simple models. Indeed, the sampling properties of these algorithms are determined by simplifying assumptions on the loss landscape, which do not hold for the kind of complex models which are popular these days, such as deep models. Meanwhile, SG-MCMC algorithms require vanishing learning rates, which force practitioners to develop creative annealing schedules that are often model specific and difficult to justify.

We have attempted to target these weaknesses by suggesting a simpler algorithm that relies on fewer parameters and less strict assumptions compared to the literature on SG-MCMC. We used a unified mathematical notation to deepen our understanding of the role of the covariance of the noise of stochastic gradients and learning rate on the behavior of SG-MCMC algorithms. We then presented a practical variant of the SGD algorithm, which uses a constant learning rate, and an additional noise to perform Bayesian posterior sampling. Our proposal is derived from the ideal method, in which it is guaranteed that samples are generated from the true posterior. When the learning rate and noise terms are empirically estimated, with no user intervention, our method offers a very good approximation to the posterior, as demonstrated by the extensive experimental campaign.

We verified empirically the quality of our approach, and compared its performance to state-of-the-art SG-MCMC and alternative methods. Results, which span a variety of settings, indicated that our method is competitive to the alternatives from the state-of-the-art, while being much simpler to use.

Author Contributions: Formal analysis, G.F., D.M., M.F. and P.M.; Methodology, G.F., D.M., M.F. and P.M.; Software, G.F., D.M., M.F. and P.M.; Writing—original draft, G.F., D.M., M.F. and P.M.; Writing—review & editing, G.F., D.M., M.F. and P.M. All authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: MF gratefully acknowledges support from the AXA Research Fund and the Agence Nationale de la Recherche (grant ANR-18-CE46-0002 and ANR-19-P3IA-0002).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Background and Related Material

Appendix A.1. The Minibatch Gradient Approximation

Starting from the gradient of the logarithm of the posterior density:

$$-\nabla f(\boldsymbol{\theta}) = \sum_{i=1}^{N} \nabla \log p(\boldsymbol{U}_i|\boldsymbol{\theta}) + \nabla \log p(\boldsymbol{\theta}),$$

it is possible to define its *minibatch* version by computing the gradient on a random subset $\mathcal{I}_{N_b}$ with cardinality N_b of all the indices. The minibatch gradient $\boldsymbol{g}(\boldsymbol{\theta})$ is computed as

$$-\boldsymbol{g}(\boldsymbol{\theta}) = \frac{N}{N_b} \sum_{i=1}^{N_b} \nabla \log p(\boldsymbol{U}_i|\boldsymbol{\theta}) + \nabla \log p(\boldsymbol{\theta}),$$

By simple calculations it is possible to show that the estimation is unbiased ($E(\boldsymbol{g}(\boldsymbol{\theta})) = \nabla f(\boldsymbol{\theta})$). The estimation error covariance is defined to be $E\left[(\boldsymbol{g}(\boldsymbol{\theta}) - \nabla f(\boldsymbol{\theta}))(\boldsymbol{g}(\boldsymbol{\theta}) - \nabla f(\boldsymbol{\theta}))^\top\right] = 2\boldsymbol{B}(\boldsymbol{\theta})$.

If the minibatch size is large enough, invoking the central limit theorem, we can state that the minibatch gradient is normally distributed:

$$\boldsymbol{g}(\boldsymbol{\theta}) \sim N(\nabla f(\boldsymbol{\theta}), 2\boldsymbol{B}(\boldsymbol{\theta})).$$

Appendix A.2. Gradient Methods without Momentum

Appendix A.2.1. The SDE from Discrete Time

We start from the generalized updated rule of SGD:

$$\delta\boldsymbol{\theta}_n = -\eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})(\boldsymbol{g}(\boldsymbol{\theta}_{n-1}) + \boldsymbol{w}_n).$$

Since $\boldsymbol{g}(\boldsymbol{\theta}_{n-1}) \sim N(\nabla f(\boldsymbol{\theta}_{n-1}), 2\boldsymbol{B}(\boldsymbol{\theta}_{n-1}))$ we can rewrite the above equation as:

$$\delta\boldsymbol{\theta}_n = -\eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})(\nabla f(\boldsymbol{\theta}_{n-1}) + \boldsymbol{w}_n^{'}),$$

where $\boldsymbol{w}_n^{'} \sim N(0, 2\boldsymbol{\Sigma}(\boldsymbol{\theta}_{n-1}))$. If we separate deterministic and random component we can equivalently write:

$$\delta\boldsymbol{\theta}_n = -\eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})\nabla f(\boldsymbol{\theta}_{n-1}) + \eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})\boldsymbol{w}_n^{'} = -\eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})\nabla f(\boldsymbol{\theta}_{n-1}) + \sqrt{2\eta \boldsymbol{P}^2(\boldsymbol{\theta}_{n-1})\boldsymbol{\Sigma}(\boldsymbol{\theta}_{n-1})}\boldsymbol{v}_n$$

where $\boldsymbol{v}_n \sim N(0, \sqrt{\eta}\boldsymbol{I})$. When η is small enough ($\eta \to dt$) we can interpret the above equation as the discrete-time simulation of the following SDE [15]:

$$d\boldsymbol{\theta}_t = -\boldsymbol{P}(\boldsymbol{\theta}_t)\nabla f(\boldsymbol{\theta}_t)dt + \sqrt{2\eta \boldsymbol{P}(\boldsymbol{\theta}_t)^2 \boldsymbol{\Sigma}(\boldsymbol{\theta}_t)}d\boldsymbol{W}_t,$$

where $d\boldsymbol{W}_t$ is a $d-$dimensional Brownian motion.

Appendix A.2.2. Proof of Theorem 1

The stationary distribution of the above SDE, $\rho(\boldsymbol{\theta}) \propto \exp(-\phi(\boldsymbol{\theta}))$, satisfies the following FPE

$$0 = \mathrm{Tr}\left\{\nabla\left[\nabla^\top(f(\boldsymbol{\theta}))\boldsymbol{P}(\boldsymbol{\theta})\rho(\boldsymbol{\theta}) + \eta\nabla^\top(\boldsymbol{P}(\boldsymbol{\theta})^2\boldsymbol{\Sigma}(\boldsymbol{\theta})\rho(\boldsymbol{\theta}))\right]\right\},$$

that we rewrite as

$$0 = \mathrm{Tr}\{\nabla[\nabla^\top(f(\boldsymbol{\theta}))\boldsymbol{P}(\boldsymbol{\theta})\rho(\boldsymbol{\theta}) - \eta\nabla^\top(\phi(\boldsymbol{\theta}))\boldsymbol{P}(\boldsymbol{\theta})^2\boldsymbol{\Sigma}(\boldsymbol{\theta})\rho(\boldsymbol{\theta}) + \eta\nabla^\top(\boldsymbol{P}(\boldsymbol{\theta})^2\boldsymbol{\Sigma}(\boldsymbol{\theta}))\rho(\boldsymbol{\theta})]\}.$$

The above equation is verified with $\nabla f(\boldsymbol{\theta}) = \nabla \phi(\boldsymbol{\theta})$ if

$$\begin{cases} \nabla^\top (\boldsymbol{P}(\boldsymbol{\theta})^2 \boldsymbol{\Sigma}(\boldsymbol{\theta})) = \mathbf{0} \\ \eta \boldsymbol{P}(\boldsymbol{\theta})^2 \boldsymbol{\Sigma}(\boldsymbol{\theta}) = \boldsymbol{P}(\boldsymbol{\theta}) \rightarrow \eta \boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1} \end{cases}$$

that proves Theorem 1.

Appendix A.3. Gradient Methods with Momentum

Appendix A.3.1. The SDE from Discrete Time

The general set of update equations for (discrete-time) momentum-based algorithms is:

$$\begin{cases} \delta\boldsymbol{\theta}_n = \eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1}) \boldsymbol{M}^{-1} \boldsymbol{r}_{n-1} \\ \delta \boldsymbol{r}_n = -\eta \boldsymbol{A}(\boldsymbol{\theta}_{n-1}) \boldsymbol{M}^{-1} \boldsymbol{r}_{n-1} - \eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})(\boldsymbol{g}(\boldsymbol{\theta}_{n-1}) + \boldsymbol{w}_n). \end{cases}$$

Similarly to the case without momentum, we rewrite the second equation of the system as

$$\begin{aligned} \delta \boldsymbol{r}_n &= -\eta \boldsymbol{A}(\boldsymbol{\theta}_{n-1}) \boldsymbol{M}^{-1} \boldsymbol{r}_{n-1} - \eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1})(\boldsymbol{g}(\boldsymbol{\theta}_{n-1}) + \boldsymbol{w}_n) = \\ &- \eta \boldsymbol{A}(\boldsymbol{\theta}_{n-1}) \boldsymbol{M}^{-1} \boldsymbol{r}_{n-1} - \eta \boldsymbol{P}(\boldsymbol{\theta}_{n-1}) \nabla f(\boldsymbol{\theta}_{n-1}) + \sqrt{2\eta \boldsymbol{P}^2(\boldsymbol{\theta}_{n-1}) \boldsymbol{\Sigma}(\boldsymbol{\theta}_{n-1})} \boldsymbol{v}_n \end{aligned}$$

where again $\boldsymbol{v}_n \sim N(0, \sqrt{\eta} \boldsymbol{I})$. If we define the supervariable $\boldsymbol{z} = [\boldsymbol{\theta}, \boldsymbol{r}]^\top$ we can rewrite the system as

$$\delta \boldsymbol{z}_n = -\eta \begin{bmatrix} \mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta}_{n-1}) \\ \boldsymbol{P}(\boldsymbol{\theta}_{n-1}) & \boldsymbol{A}(\boldsymbol{\theta}_{n-1}) \end{bmatrix} \boldsymbol{s}(\boldsymbol{z}_{n-1}) + \sqrt{2\eta \boldsymbol{D}(\boldsymbol{z}_{n-1})} \boldsymbol{\nu}_n$$

where $\boldsymbol{s}(\boldsymbol{z}) = \begin{bmatrix} \nabla f(\boldsymbol{\theta}) \\ \boldsymbol{M}^{-1} \boldsymbol{r} \end{bmatrix}$, $\boldsymbol{D}(\boldsymbol{z}) = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{P}(\boldsymbol{\theta})^2 \boldsymbol{\Sigma}(\boldsymbol{\theta}) \end{bmatrix}$ and $\boldsymbol{\nu}_n \sim N(0, \sqrt{\eta} \boldsymbol{I})$.

As the learning rate goes to zero ($\eta \rightarrow dt$), similarly to the previous case, we can interpret the above difference equation as a discretization of the following FPE

$$d\boldsymbol{z}_t = -\begin{bmatrix} \mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta}_t) \\ \boldsymbol{P}(\boldsymbol{\theta}_t) & \boldsymbol{A}(\boldsymbol{\theta}_t) \end{bmatrix} \boldsymbol{s}(\boldsymbol{z}_t) + \sqrt{2\eta \boldsymbol{D}(\boldsymbol{z}_t)} d\boldsymbol{W}_t$$

Appendix A.3.2. Proof of Theorem 2

As before we assume that the stationary distribution has form $\rho(\boldsymbol{z}) \propto \exp(-\phi(\boldsymbol{z}))$. The corresponding FPE is

$$0 = \operatorname{Tr}\left(\nabla \left(\boldsymbol{s}(\boldsymbol{z})^\top \begin{bmatrix} \mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta}) \\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta}) \end{bmatrix} \rho(\boldsymbol{z}) + \eta \left(\nabla^\top (\boldsymbol{D}(\boldsymbol{z}) \rho(\boldsymbol{z})) \right) \right) \right).$$

Notice that since $\nabla^\top \boldsymbol{D}(\boldsymbol{z}) = 0$ we can rewrite

$$\begin{aligned} 0 &= \operatorname{Tr}\left(\nabla \left(\boldsymbol{s}(\boldsymbol{z})^\top \begin{bmatrix} \mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta}) \\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta}) \end{bmatrix} \rho(\boldsymbol{z}) + \eta \nabla^\top (\rho(\boldsymbol{z})) \boldsymbol{D}(\boldsymbol{z}) \right) \right) \\ &= \operatorname{Tr}\left(\nabla \left(\boldsymbol{s}(\boldsymbol{z})^\top \begin{bmatrix} \mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta}) \\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta}) \end{bmatrix} \rho(\boldsymbol{z}) - \eta \nabla^\top (\phi(\boldsymbol{z})) \boldsymbol{D}(\boldsymbol{z}) \rho(\boldsymbol{z}) \right) \right) \\ &= \operatorname{Tr}\left(\nabla \left(\boldsymbol{s}(\boldsymbol{z})^\top \begin{bmatrix} \mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta}) \\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta}) \end{bmatrix} \rho(\boldsymbol{z}) - \eta \nabla^\top (\phi(\boldsymbol{z})) \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{P}(\boldsymbol{\theta})^2 \boldsymbol{\Sigma}(\boldsymbol{\theta}) \end{bmatrix} \rho(\boldsymbol{z}) \right) \right) \end{aligned}$$

that is verified with $\nabla \phi(\boldsymbol{z}) = \boldsymbol{s}(\boldsymbol{z})$ if

$$\begin{cases} \nabla^\top \boldsymbol{P}(\boldsymbol{\theta}) = \mathbf{0} \\ \boldsymbol{A}(\boldsymbol{\theta}) = \eta \boldsymbol{P}(\boldsymbol{\theta})^2 \boldsymbol{\Sigma}(\boldsymbol{\theta}). \end{cases}$$

If $\nabla^\top \boldsymbol{P}(\boldsymbol{\theta}) = \mathbf{0}$ in fact

$$\mathrm{Tr}\left(\nabla\left(\nabla^\top(\phi(z))\rho(z)\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \mathbf{0}\end{bmatrix}\right)\right) = \nabla^\top\left(\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \mathbf{0}\end{bmatrix}\nabla(\phi(z))\rho(z)\right) =$$
$$\nabla^\top\left(\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \mathbf{0}\end{bmatrix}\right)\nabla(\phi(z))\rho(z) + \mathrm{Tr}\left(\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \mathbf{0}\end{bmatrix}\nabla\left(\nabla^\top(\phi(z))\rho(z)\right)\right) = 0,$$

since $\nabla^\top\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \mathbf{0}\end{bmatrix} = \mathbf{0}$ and the second term is zero due to the fact that $\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \mathbf{0}\end{bmatrix}$ is anti-symmetric while $\nabla\left(\nabla^\top(\phi(z))\rho(z)\right)$ is symmetric.

Thus, we can rewrite

$$\mathrm{Tr}\left(\nabla\left(\boldsymbol{s}(z)^\top\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta})\end{bmatrix}\rho(z) - \eta\nabla^\top(\phi(z))\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{P}(\boldsymbol{\theta})^2\boldsymbol{\Sigma}(\boldsymbol{\theta})\end{bmatrix}\rho(z)\right)\right) =$$
$$\mathrm{Tr}\left(\nabla\left(\boldsymbol{s}(z)^\top\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta})\end{bmatrix}\rho(z) - \nabla^\top(\phi(z))\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \eta\boldsymbol{P}(\boldsymbol{\theta})^2\boldsymbol{\Sigma}(\boldsymbol{\theta})\end{bmatrix}\rho(z)\right)\right) =$$
$$\mathrm{Tr}\left(\nabla\left(\boldsymbol{s}(z)^\top\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta})\end{bmatrix}\rho(z) - \nabla^\top(\phi(z))\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{A}(\boldsymbol{\theta})\end{bmatrix}\rho(z)\right)\right) =$$
$$\mathrm{Tr}\left(\nabla\left(\left(\boldsymbol{s}(z)^\top - \nabla^\top(\phi(z))\right)\begin{bmatrix}\mathbf{0} & -\boldsymbol{P}(\boldsymbol{\theta})\\ \boldsymbol{P}(\boldsymbol{\theta}) & \boldsymbol{A}(\boldsymbol{\theta})\end{bmatrix}\rho(z)\right)\right) = 0$$

and then $\nabla\phi(z) = \boldsymbol{s}(z)$ proving Theorem 2.

Appendix B. I-SGD Method Proofs and Details

Appendix B.1. Proof of Corollary 1

The requirement $\boldsymbol{C}(\boldsymbol{\theta}) \succeq 0 \quad \forall\boldsymbol{\theta}$, ensures that the injected noise covariance is valid. The composite noise matrix is equal to $\boldsymbol{\Sigma}(\boldsymbol{\theta}) = \boldsymbol{\Lambda}$. Since $\nabla^\top\boldsymbol{\Sigma}(\boldsymbol{\theta}) = \nabla^\top\boldsymbol{\Lambda} = \mathbf{0}$ and $\eta\boldsymbol{P}(\boldsymbol{\theta}) = \boldsymbol{\Lambda}^{-1}$ by construction, then Theorem 1 is satisfied.

Appendix B.2. Proof of Optimality of $\boldsymbol{\Lambda}$

Our design choice is to select $\lambda^{(p)} = \beta^{(p)}$. By the assumptions, the matrix $\boldsymbol{B}(\boldsymbol{\theta})$ is diagonal, and consequently $\boldsymbol{C}(\boldsymbol{\theta}) = \boldsymbol{\Lambda} - \boldsymbol{B}(\boldsymbol{\theta})$ is diagonal as well. The preconditioner $\boldsymbol{\Lambda}$ must be chosen to satisfy the positive semidefinite constraint, i.e., $\boldsymbol{C}(\boldsymbol{\theta})_{ii} \geq 0 \quad \forall i, \forall\boldsymbol{\theta}$. Equivalently, we must satisfy $\lambda^{(p)} - b_j(\boldsymbol{\theta}) \geq 0 \quad \forall j \in I_p, \forall p, \forall\boldsymbol{\theta}$, where I_p is the set of indices of parameters belonging to p_{th} layer. By assumption 3, i.e., $\beta^{(p)} = \sum_{k\in I_p} b_k(\boldsymbol{\theta})$, it is easy to show that $b_j(\boldsymbol{\theta}), j \in I_p$, is upper bounded as $b_j(\boldsymbol{\theta}) \leq \beta^{(p)}$. To satisfy the positive semidefinite requirement in all cases the minimum valid set of $\lambda^{(p)}$ is then determined as $\lambda^{(p)} = \beta^{(p)}$.

Appendix B.3. Algorithmic Details

In this section, we provide further details about the practical implementation of the proposed scheme. At any (discrete) time instant a minibatch version of the gradient is computed that is distributed, according to the hypotheses of the main paper, as $\boldsymbol{g}(\boldsymbol{\theta}) \sim N(\nabla f(\boldsymbol{\theta}), 2b(\boldsymbol{\theta}))$. Since we assumed that the second-order moment is a good approximation of the variance, we can estimate $b(\boldsymbol{\theta})$ as $\frac{1}{2}(\boldsymbol{g}(\boldsymbol{\theta}) \odot \boldsymbol{g}(\boldsymbol{\theta}))$. In practice, we found that the following running average estimation procedure to be the most robust

$$b(\boldsymbol{\theta}) \leftarrow \mu b(\boldsymbol{\theta}) + (1-\mu)\frac{1}{2}(\boldsymbol{g}(\boldsymbol{\theta}) \odot \boldsymbol{g}(\boldsymbol{\theta})) \tag{A1}$$

where $\mu \in (0,1]$. In all experiments we considered $\mu = 0.5$

After a warmup period, the various $\lambda^{(p)}$, layer per layer, are estimated as $\lambda^{(p)} = \sum_{k\in I_p} b_k(\boldsymbol{\theta})$ and kept constant until the end. The estimation procedure continues during sampling phase, as the quantity $\lambda^{(p)} - b(\boldsymbol{\theta})$ is necessary at every step. As the learning rate is derived as $\frac{2}{\lambda^{(p)}}$, we found that the usage of second-order moments instead of variances, and in certain cases temperature scaling, kept the simulated trajectories more stable.

Appendix C. Methodology

We hereafter present additional implementation details.

Appendix C.1. Regression Tasks, with Simple Models

For this set of experiments we considered , the BASELINE is obtained by running the ADAM optimizer for 20,000 steps with learning rate 0.01 and default parameters. At test time we use 100 samples to estimate the predictive posterior distribution, using Equation (3), for the *sampling* methods (I-SGD,SGLD,SGHMC,VSGD), with a keep-every value equal to 1000. The I-SGD and VSGD sampling methods are started from the BASELINE. For I-SGD we selected temperature 0.01, while for SGHMC and SGLD we do performed experiments for temperatures 1 and 0.01. We modified the implementation of VSGD as the original implementation produced unstable learning rates (as noticed also in [9]). A simple and effective solution we implement that we kept throughout the experimental campaigns is to divide the learning rate by the number of parameters (thus performing variational inference on a tempered version of the posterior). For SGLD the learning rate decay is the one suggested in [2], with initial and finial learning rate equal to 10^{-6} and 10^{-8} respectively. For MCD we collected 1000 samples with standard dropout rate of 0.5. All our experiments use 10-splits. The considered batch size is 64 for all methods.

Appendix C.2. Classification Task, CONVNET

For the LENET-5 on MNIST experiment, we do consider also the SWAG algorithm. At test time we use 30 samples for all methods. Baselines are again trained using ADAM optimizer for 20,000 steps with learning rate 0.01 and default parameters. For I-SGD and SGHMC we collected samples for the different temperatures of 1 and 0.01. SGLD has initial and final learning rates of 10^{-3} and 10^{-5}. For all the sampling methods we do collect 100 samples with a keep-every of 10,000 steps. SWAG results are obtained by collecting the statistics over 300 epochs using ADAM optimizer and decreasing the learning rate every epoch in accordance with the original paper schedule [9]. DROP results are obtained by training the networks with SGD, with learning rate 0.005 and momentum 0.5. The number of collected samples for this method is 1000. The batch size for all the methods is 128.

As explained in the main text, we performed an ablation study on the considered baselines. In Table A1 we do report the results for the additional variants obtained by early stopping (10,000 iterations instead of 20,000) BASELINE S, to ablate overfitting, and BASELINE L, by training for 30,000 iterations. Finally, we include the best performing BASELINE R, obtained starting from BASELINE, reducing the learning rate by a factor of 10 and training for 10,000 more iterations.

Table A1. Baselines comparison for classification on MNIST dataset.

Method	ACC	MNLL	**Mean H_0**	ECE	**Mean H_1**	**Failed**
BASELINE	9886.6667 ± 11.0252	352.6640 ± 20.8622	0.0353 ± 0.0058	0.0468 ± 0.0001	0.0019 ± 0.0003	0.0000
BASELINE l	9871.6667 ± 20.7579	389.7142 ± 79.0354	0.0378 ± 0.0051	0.0468 ± 0.0008	0.0025 ± 0.0006	0.0000
BASELINE s	9893.0000 ± 4.8990	339.8170 ± 7.9855	0.0392 ± 0.0042	0.0477 ± 0.0008	0.0024 ± 0.0001	0.0000
BASELINE r	9919.0000 ± 9.4163	242.7644 ± 17.0736	0.0303 ± 0.0001	0.0482 ± 0.0006	0.0021 ± 0.0002	0.0000

Appendix C.3. Classification Task, Deeper Models

We here report details for the RESNET-18 on CIFAR10 experiments. The BASELINE is obtained with ADAM optimizer with learning rate 0.01 decreased by a factor of 10 every 50 epochs for a total of 200 epochs and weight decay of 0.05. For this set of experiments no temperature scaling was required. We could not find good hyperparameters for the SGLD scheme. Concerning I-SGD, SGHMC and VSGD the keep-every value is chosen as 10,000 and the number of collected samples is 30. For SWAG we used the default parameters described in [9]. Notice that for SWAG we performed the following ablation study: we trained the networks considering as loss function the joint log-likelihood and included or not the suggested weight decay of the original work [9]. From a purely Bayesian perspective no weight decay should be considered to be the information is implicit in the prior; however, we found that without the extra decay SWAG was not able to obtain competitive results. As underlined in Section 4.1, not necessarily a better posterior approximation translates into better empirical results.

Appendix C.4. Definition of the Metrics

For regression datasets, we consider RMSE and MNLL. Consider a single datapoint $\boldsymbol{U}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$, with $\boldsymbol{x}_i$ the input of the model and $\boldsymbol{y}_i$ the true corresponding output. The output of the model, for a single sample of parameters $\boldsymbol{\theta}_j$, is $\hat{\boldsymbol{y}}_{\boldsymbol{\theta}_j}(\boldsymbol{x}_i)$. RMSE is defined as $\frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{y}_i - \mu(\boldsymbol{x}_i)||^2$, where $\mu(\boldsymbol{x}_i)$ is the empirical mean $\frac{1}{N_{MC}}\sum_{j=1}^{N_{MC}} \hat{\boldsymbol{y}}_{\boldsymbol{\theta}_j}(\boldsymbol{x}_i)$. MNLL is defined instead as $(\frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{2}\log(2\pi\sigma_i^2) + \frac{1}{2}\frac{||\boldsymbol{y}_i - \mu(\boldsymbol{x}_i)||^2}{\sigma_i^2}\right)$, where σ_i^2 is the empirical variance.

For classification datasets, we consider ACC,MNLL and entropy. Consider a single datapoint $\boldsymbol{U}_i = (\boldsymbol{x}_i, y_i)$, with $\boldsymbol{x}_i$ the input of the model and y_i the true corresponding label. The output of the model, for a single sample of parameters $\boldsymbol{\theta}_j$, is the N_{cl} vector $\mathbf{p}_{\boldsymbol{\theta}_j}(\boldsymbol{x}_i)$. The averaged probability vector for a single sample is $\mathbf{p}(\boldsymbol{x}_i) = \frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}} \mathbf{p}_{\boldsymbol{\theta}_j}(\boldsymbol{x}_i)$.ACC is defined as $\frac{1}{N}\sum_{i=1}^{N} 1(\arg\max \mathbf{p}(\boldsymbol{x}_i) = y_i)$. MNLL is computed as $\frac{1}{N}\sum_{i=1}^{N}\log(\mathbf{p}_{y_i}(\boldsymbol{x}_i))$. Entropy, as stated in the main text, is instead computed according to $\frac{1}{N}\sum_{i=1}^{N}\left(\sum_{k=1}^{N_{cl}} \mathbf{p}_k(\boldsymbol{x}_i)\log(\mathbf{p}_k(\boldsymbol{x}_i))\right)$.

References

1. Welling, M.; Teh, Y.W. Bayesian learning via stochastic gradient Langevin dynamics. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June–2 July 2011; pp. 681–688.
2. Ahn, S.; Korattikara, A.; Welling, M. Bayesian posterior sampling via stochastic gradient Fisher scoring. *arXiv* **2012**, arXiv:1206.6380.
3. Patterson, S.; Teh, Y.W. Stochastic Gradient Riemannian Langevin Dynamics on the Probability Simplex. In *Advances in Neural Information Processing Systems 26*; Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2013; pp. 3102–3110.
4. Chen, T.; Fox, E.; Guestrin, C. Stochastic gradient hamiltonian monte carlo. In Proceedings of the International Conference on Machine Learning, Bejing, China, 22–24 June 2014; pp. 1683–1691.
5. Ma, Y.A.; Chen, T.; Fox, E. A complete recipe for stochastic gradient MCMC. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 2917–2925.
6. Draxler, F.; Veschgini, K.; Salmhofer, M.; Hamprecht, F.A. Essentially no barriers in neural network energy landscape. *arXiv* **2018**, arXiv:1803.00885.
7. Garipov, T.; Izmailov, P.; Podoprikhin, D.; Vetrov, D.; Wilson, A.G. Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018.
8. Chaudhari, P.; Soatto, S. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In Proceedings of the 2018 Information Theory and Applications Workshop (ITA), San Diego, CA, USA, 11–16 February 2018; pp. 1–10.
9. Maddox, W.J.; Izmailov, P.; Garipov, T.; Vetrov, D.P.; Wilson, A.G. A simple baseline for bayesian uncertainty in deep learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 13132–13143.
10. Mandt, S.; Hoffman, M.D.; Blei, D.M. Stochastic gradient descent as approximate bayesian inference. *J. Mach. Learn. Res.* **2017**, *18*, 4873–4907.
11. Springenberg, J.T.; Klein, A.; Falkner, S.; Hutter, F. Bayesian optimization with robust Bayesian neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 4134–4142.

12. Gal, Y.; Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the International Conference on Machine Learning, ICML, New York, NY, USA, 19–24 June 2016; pp. 1050–1059.
13. Bishop, C.M. *Pattern Recognition and Machine Learning*, 1st ed.; 2006. corr. 2nd printing 2011 ed.; Springer: Berlin/Heidelberg, Germany, 2006.
14. Chen, C.; Carlson, D.; Gan, Z.; Li, C.; Carin, L. Bridging the gap between stochastic gradient MCMC and stochastic optimization. In Proceedings of the Artificial Intelligence and Statistics, Cadiz, Spain, 9–11 May 2016; pp. 1051–1060.
15. Gardiner, C.W. *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences*, 3rd ed.; Springer Series in Synergetics; Springer: Berlin/Heidelberg, Germany, 2004; Volume 13.
16. Kushner, H.; Yin, G. *Stochastic Approximation and Recursive Algorithms and Applications*; Stochastic Modelling and Applied Probability; Springer: New York, NY, USA, 2003.
17. Ljung, L.; Pflug, G.; Walk, H. *Stochastic Approximation and Optimization of Random Systems*; Birkhauser Verlag: Basel, Switzerland, 1992.
18. Kloeden, P.E.; Platen, E. *Numerical Solution of Stochastic Differential Equations*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013; Volume 23.
19. Neal, R.M. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*; CRC Press: Boca Raton, FL, USA, 2011; Volume 2, p. 2.
20. Girolami, M.; Calderhead, B. Riemann manifold langevin and hamiltonian monte carlo methods. *J. R. Stat. Soc. Ser. B (Stat. Methodol.)* **2011**, *73*, 123–214. [CrossRef]
21. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
22. Vollmer, S.J.; Zygalakis, K.C.; Teh, Y.W. Exploration of the (non-) asymptotic bias and variance of stochastic gradient Langevin dynamics. *J. Mach. Learn. Res.* **2016**, *17*, 5504–5548.
23. Saxe, A.M.; Bansal, Y.; Dapello, J.; Advani, M.; Kolchinsky, A.; Tracey, B.D.; Cox, D.D. On the information bottleneck theory of deep learning. *J. Stat. Mech. Theory Exp.* **2019**, *2019*, 124020. [CrossRef]
24. Zhu, Z.; Wu, J.; Yu, B.; Wu, L.; Ma, J. The anisotropic noise in stochastic gradient descent: Its behavior of escaping from minima and regularization effects. *arXiv* **2018**, arXiv:1803.00195.
25. Scott, W.A. Maximum likelihood estimation using the empirical fisher information matrix. *J. Stat. Comput. Simul.* **2002**, *72*, 599–611. [CrossRef]
26. Dua, D.; Graff, C. UCI Machine Learning Repository. 2017. Available online: https://archive.ics.uci.edu/ml/index.php (accessed on 25 October 2021)
27. LeCun, Y.; Cortes, C. MNIST Handwritten Digit Database. 2010. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 25 October 2021)
28. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
29. Krizhevsky, A.; Nair, V.; Hinton, G. CIFAR-10 (Canadian Institute for Advanced Research). Available online: http://www.cs.toronto.edu/~kriz/cifar.html (accessed on 25 October 2021)
30. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
31. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On calibration of modern neural networks. *arXiv* **2017**, arXiv:1706.04599.

entropy

MDPI

Article

PAC-Bayes Unleashed: Generalisation Bounds with Unbounded Losses

Maxime Haddouche [1], Benjamin Guedj [2,3,*], Omar Rivasplata [2] and John Shawe-Taylor [2]

[1] ENS Paris-Saclay, 91190 Gif-sur-Yvette, France; maxime.haddouche@ens-paris-saclay.fr
[2] Centre for Artificial Intelligence, Department of Computer Science, University College London, London WC1V 6LJ, UK; o.rivasplata@cs.ucl.ac.uk (O.R.); j.shawe-taylor@ucl.ac.uk (J.S.-T.)
[3] Inria, Lille–Nord Europe Research Centre and Inria London Programme, 59800 Lille, France
* Correspondence: b.guedj@ucl.ac.uk

Abstract: We present new PAC-Bayesian generalisation bounds for learning problems with unbounded loss functions. This extends the relevance and applicability of the PAC-Bayes learning framework, where most of the existing literature focuses on supervised learning problems with a bounded loss function (typically assumed to take values in the interval [0;1]). In order to relax this classical assumption, we propose to allow the range of the loss to depend on each predictor. This relaxation is captured by our new notion of *HYPothesis-dependent rangE* (HYPE). Based on this, we derive a novel PAC-Bayesian generalisation bound for unbounded loss functions, and we instantiate it on a linear regression problem. To make our theory usable by the largest audience possible, we include discussions on actual computation, practicality and limitations of our assumptions.

Keywords: statistical learning theory; PAC-Bayes; generalisation bounds

Citation: Haddouche, M.; Guedj, B.; Rivasplata, O.; Shawe-Taylor, J. PAC-Bayes Unleashed: Generalisation Bounds with Unbounded Losses. *Entropy* **2021**, *23*, 1330. https://doi.org/10.3390/e23101330

Academic Editor: Boris Ryabko

Received: 22 August 2021
Accepted: 25 September 2021
Published: 12 October 2021

1. Introduction

Since its emergence in the late 1990s, the PAC-Bayes theory (see the seminal works of [1–3], the recent survey by [4] and work by [5]) has been a powerful tool to obtain generalisation bounds and to derive efficient learning algorithms. Generalisation bounds are helpful for understanding how a learning algorithm may perform on future similar batches of data. While the classical generalization bounds typically address the performance of individual predictors from a given hypothesis class, PAC-Bayes bounds typically address a randomized predictor defined by a distribution over the hypothesis class.

PAC-Bayes bounds were originally meant for binary classification problems [6–8], but the literature now includes many contributions involving any bounded loss function (without loss of generality, with values in $[0;1]$), not just the binary loss. Our goal is to provide new PAC-Bayes bounds that are valid for unbounded loss functions, and thus extend the usability of PAC-Bayes to a much larger class of learning problems. To do so, we reformulate the general PAC-Bayes theorem of [9] and use it as basic building block to derive our new PAC-Bayes bound.

Some ways to circumvent the bounded range assumption on the losses have been explored in the recent literature. For instance, one approach consists of assuming a tail decay rate on the loss, such as sub-gaussian or sub-exponential tails [10,11]; however, this approach requires the knowledge of additional parameters. Some other works have also looked into the analysis for heavy-tailed losses, e.g., ref. [12] proposed a polynomial moment-dependent bound with f-divergences, while [13] devised an exponential bound that assumes the second (uncentered) moment of the loss is bounded by a constant (with a truncated risk estimator, as recalled in Section 4 below). A somewhat related approach was explored by [14], who do not assume boundedness of the loss, but instead control higher-order moments of the generalization gap through the Efron-Stein variance proxy. See also [5].

We investigate a different route here. We introduce the *HYPothesis-dependent rangE* (HYPE) condition, which means that the loss is upper-bounded by a term that depends on the chosen predictor (but does not depend on the data). Thus, effectively, the loss may have an arbitrarily large range. The HYPE condition allows us to derive an upper bound on the exponential moment of a suitably chosen functional, which, combined with the general PAC-Bayes theorem, leads to our new PAC-Bayes bound. To illustrate it, we instantiate the new bound on a linear regression problem, which additionally serves the purpose of illustrating that our HYPE condition is easy to verify in practice, given an explicit formulation of the loss function. In particular, we shall see in the linear regression setting that a mere use of the triangle inequality is enough to check the HYPE condition. The technical assumptions on which our results are based are comparable to those of the classical PAC-Bayes bounds; we state them in full detail, with discussions, for the sake of clarity and to make our work accessible.

Our contributions are twofold. (i) We propose PAC-Bayesian bounds holding with unbounded loss functions, therefore overcoming a limitation of the mainstream PAC-Bayesian literature for which a bounded loss is usually assumed. (ii) We analyse the bound, its implications, limitations of our assumptions, and their usability by practitioners. We hope this will extend the PAC-Bayes framework into a widely usable tool for a significantly wider range of problems, such as unbounded regression or reinforcement learning problems with unbounded rewards.

Outline. Section 2 introduces our notation and definition of the HYPE condition and provides a general PAC-Bayesian bound, which is valid for any learning problem complying with a mild assumption. For the sake of completeness, we present how our approach (designed for the unbounded case) behaves in the bounded case (Section 3). This section is not the core of our work, but rather serves as a safety check and particularises our bound to more classical PAC-Bayesian assumptions. We also provide numerical experiments. Section 4 introduces the notion of *softening functions* and particularises Section 2's PAC-Bayesian bound. In particular, we make explicit all terms in the right-hand side. Section 5.1 extends our results to linear regression (which has been studied from the perspective of PAC-Bayes in the literature, most recently by [15]). We also experimentally illustrate the behaviour of our bound. Finally, Section 6 presents, in detail, related works and Section 7 contains all proofs of the original claims we make in the paper.

2. Framework and Preliminary Results

The learning problem is specified by three variables $(\mathcal{H}, \mathcal{Z}, \ell)$ consisting of a set $\mathcal{H}$ of predictors, the data space $\mathcal{Z}$, and a loss function $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}^+$.

For a given positive integer m, we consider size-m datasets. The space of all possible datasets of this fixed size is $\mathcal{S} = \mathcal{Z}^m$; an arbitrary element of this space is $s = (z_1, \dots, z_m)$. We denote S as a random dataset: $S = (Z_1, \dots, Z_m)$ where the random data points Z_i are independent and sampled from the same distribution μ over $\mathcal{Z}$. We call μ the data-generating distribution. The assumption that the Z_i's are *independent and identically distributed* is typically called the i.i.d. data assumption. It means that the random sample S (of size m) has distribution $\mu^{\otimes m}$ which is the product of m copies of μ.

For any predictor $h \in \mathcal{H}$, we define the *empirical risk* of h over a sample s, denoted $R_s(h)$, and the *theoretical risk* of h, denoted $R(h)$, as:

$$R_s(h) = \frac{1}{m} \sum_{i=1}^{m} \ell(h, z_i) \qquad \text{and} \qquad R(h) = \mathbb{E}_\mu[\ell(h, Z)]$$

respectively, where $\mathbb{E}_\mu[\ell(h, Z)]$ denotes the expectation with respect to $Z \sim \mu$. Finally, we define the *risk gap* $\Delta_s(h) = R(h) - R_s(h)$ for any $h \in \mathcal{H}$ and $s \in \mathcal{S}$. Often, $\Delta_s(h)$ is referred to as the generalisation gap.

Notice that for a random dataset S, the empirical risk $R_S(h)$ is random, with expected value $\mathbb{E}_{\mu^{\otimes m}}[R_S(h)] = R(h)$, where $\mathbb{E}_{\mu^{\otimes m}}$ the expectation under the distribution of the random sample S.

In general, $\mathbb{E}_\mu[\cdot]$ denotes an expectation under the distribution μ. When we want to emphasize the role of the random variable $Z \sim \mu$ we write $\mathbb{E}_Z[\cdot]$ or $\mathbb{E}_{Z\sim\mu}[\cdot]$ instead of $\mathbb{E}_\mu[\cdot]$. We use a similar convention for expectations related to any other distributions and random quantities. We now introduce the key concept to our analysis.

Definition 1. (HYPE). *A loss function $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}^+$ is said to satisfy the **hypothesis-dependent range** (`HYPE`) condition if there exists a function $K : \mathcal{H} \to \mathbb{R}^+ \backslash \{0\}$ such that $\sup_{z\in\mathcal{Z}} \ell(h,z) \leq K(h)$ for every predictor h. We then say that ℓ is `HYPE`(K) compliant.*

Let $\mathcal{M}_1^+(\mathcal{H})$ be the set of probability distributions on $\mathcal{H}$. We assume that all considered probability measures on $\mathcal{H}$ are defined on a fixed σ-algebra over $\mathcal{H}$, while the notation $\mathcal{M}_1^+(\mathcal{H})$ hides the σ-algebra, for simplicity. For $P, P' \in \mathcal{M}_1^+(\mathcal{H})$, the notation $P' \ll P$ indicates that P' is absolutely continuous with respect to P (i.e., $P'(A) = 0$ if $P(A) = 0$ for measurable $A \subset \mathcal{H}$). We write $P' \sim P$ to indicate that $P' \ll P$ and $P \ll P'$, i.e., these two distributions are absolutely continuous with respect to each other.

We now recall a result from Germain et al. [9]. Note that while implicit in many PAC-Bayes works (including theirs), we make it explicit that both the prior P and the posterior Q must be absolutely continuous with respect to each other. We discuss this restriction below.

Theorem 1. (Adapted from [9], Theorem 2.1.) *For any $P \in \mathcal{M}_1^+(\mathcal{H})$ with no dependency on data, for any function $F : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}$, define the exponential moment:*

$$\chi := \mathbb{E}_S \mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right].$$

If F is convex, then for any $\delta \in [0;1]$, with probability of at least $1-\delta$ over random samples S, simultaneously for all $Q \in \mathcal{M}_1^+(\mathcal{H})$ such that $Q \sim P$ we have:

$$F\left(\mathbb{E}_{h\sim Q}[R_S(h)], \mathbb{E}_{h\sim Q}[R(h)]\right) \leq \mathrm{KL}(Q||P) + \log\left(\frac{\chi}{\delta}\right).$$

The proof is deferred to Section 7.1. Note that the proof in [9] requires that $P \ll Q$, although it is not explicitly stated; we highlight this in our own proof. While $Q \ll P$ is classical and necessary for the $\mathrm{KL}(Q||P)$ to be meaningful, $P \ll Q$ appears to be more restrictive. In particular, we have to choose Q such that it has the exact same support as P (e.g., choosing a Gaussian and a truncated Gaussian is not possible). However, we can still apply our theorem when P and Q belong to the same parametric family of distributions, e.g., both 'full-support' Gaussian or Laplace distributions, but these are just two examples and there are many others.

Note that Alquier et al. [10] (Theorem 4.1) adapted a result from Catoni [8], which only requires $Q \ll P$. This comes at the expense of what Alquier et al. [10] (Definition 2.3) called a *Hoeffding's assumption*, which means that the exponential moment χ is assumed to be bounded by a function depending only on the hyperparameters (such as the dataset size m or parameters given by Hoeffding's assumption). Our analysis does not require this assumption, which might prove restrictive in practice.

Theorem 1 may be seen as a basis to recover many classical PAC-Bayesian bounds. For instance, $F(x,y) = 2m(x-y)^2$, recovers McAllester's bound as recalled in [4] (Theorem 1). To get a usable bound, the outstanding task is to bound the exponential moment χ. Note that a previous attempt has been made in [11], as described in Section 6.1 below. Furthermore, under the assumption that the distribution P has no dependency on the data, we may swap the order of integration in the exponential moment thanks to Fubini-Tonelli's theorem and the positiveness of the exponential:

$$\chi = \mathbb{E}_{h\sim P}\mathbb{E}_S\left[e^{F(R_S(h),R(h))}\right].$$

This is the starting point for the way that the exponential moment was handled in several works in the PAC-Bayes literature. Essentially, for a fixed h, one may upper-bound the innermost expectation (with respect to S) using standard exponential moment inequalities.

In this work, we will use Theorem 1 with $F(x,y) = m^\alpha D(x,y)$, where $\alpha > 0$, and $D : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}$ is a convex function. In this case, the high-probability inequality of the theorem takes the form:

$$D\big(\mathbb{E}_{h\sim Q}[R_S(h)], \mathbb{E}_{h\sim Q}[R(h)]\big) \leq \frac{1}{m^\alpha}\left(\mathrm{KL}(Q||P) + \log\left(\frac{1}{\delta}\mathbb{E}_{h\sim P}\mathbb{E}_S\, e^{m^\alpha D(R_S(h),R(h))}\right)\right). \quad (1)$$

Our goal is to control $\mathbb{E}_S\, e^{m^\alpha D(R_S(h),R(h))}$ for a fixed h, when $D(x,y) = y - x$. This will readily give us control on the exponential moment χ. To do so, we propose the following theorem:

Theorem 2. *Let $h \in \mathcal{H}$ be a fixed predictor and $\alpha \in \mathbb{R}$. If the loss function ℓ is $\texttt{HYPE}(K)$ compliant, then for $\Delta_S(h) = R(h) - R_S(h)$ we have:*

$$\mathbb{E}_S\left[e^{m^\alpha \Delta_S(h)}\right] \leq \exp\left(\frac{K(h)^2}{2m^{1-2\alpha}}\right).$$

Proof. Let $h \in \mathcal{H}$. Then:

$$\begin{aligned}\mathbb{E}_S\left[e^{m^\alpha \Delta_S(h)}\right] &= \mathbb{E}\left[\exp\left(m^{\alpha-1}\sum_{i=1}^m (l(h,Z_i) - R(h))\right)\right] \\ &= \mathbb{E}\left[\prod_{i=1}^m \exp\Big(m^{\alpha-1}(\ell(h,Z_i) - R(h))\Big)\right] \\ &= \prod_{i=1}^m \mathbb{E}\Big[\exp\Big(m^{\alpha-1}(\ell(h,Z_i) - R(h))\Big)\Big].\end{aligned}$$

We now apply Hoeffding's lemma, for any $i \in \{1..m\}$, the random (in Z_i) variable $\ell(h,Z_i) - R(h)$ is centered, taking values in $[-K(h);K(h)]$, so that:

$$\mathbb{E}\Big[\exp\Big(m^{\alpha-1}(\ell(h,Z_i) - R(h))\Big)\Big] \leq \exp\left(m^{2\alpha-2}\frac{4K(h)^2}{8}\right)$$

and finally:

$$\mathbb{E}_S\left[e^{m^\alpha \Delta_S(h)}\right] \leq \prod_{i=1}^m \exp\left(m^{2\alpha-2}\frac{4K(h)^2}{8}\right) = \exp\left(\frac{K(h)^2}{2m^{1-2\alpha}}\right).$$

□

The strength of this result lies in the fact that $\frac{K(h)^2}{m^{1-2\alpha}}$, is a decreasing factor in m, when $\alpha \leq 1/2$, and more generally, one can control how fast the exponential moment will explode when m grows by the choice of the hyperparameter α.

For convenient cross-referencing, we state the following rewriting of Theorem 1.

Theorem 3. *Let the loss ℓ be $\texttt{HYPE}(K)$ compliant. For any $P \in \mathcal{M}_1^+(\mathcal{H})$ with no data dependency, for any $\alpha \in \mathbb{R}$ and for any $\delta \in [0;1]$, with probability of at least $1-\delta$ over size-m random samples S, simultaneously for all Q such that $Q \sim P$ we have:*

$$\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}[R_S(h)] + \frac{1}{m^\alpha}\left(\mathrm{KL}(Q||P) + \log\frac{\mathbb{E}_{h\sim P}\left[\exp\left(\frac{K(h)^2}{2m^{1-2\alpha}}\right)\right]}{\delta}\right).$$

Proof. We first apply Theorem 1 with $F(x,y) = m^\alpha(y-x)$. More precisely, we use Equation (1) with $D(x,y) = y - x$. We then conclude with Theorem 2. □

3. Safety Check: The Bounded Loss Case

3.1. Theoretical Results

At this stage, the reader might wonder whether this new approach allows for the recovery of known results in the bounded case: the answer is yes.

In this section, we study the case where ℓ is bounded by some constant $C \in \mathbb{R}^+ \setminus \{0\}$. In other words, we consider the case that $\sup_h \sup_z \ell(h,z) \le C$. We provide a bound, valid for any choice of "priors" P and "posteriors" Q such that $P \sim Q$, which is an immediate corollary of Theorem 3.

Proposition 1. *Let ℓ be $\texttt{HYPE}(K)$ compliant, with $K(h) = C$ constant, and let $\alpha \in \mathbb{R}$. Let $P \in \mathcal{M}_1^+(\mathcal{H})$ be a distribution with no data dependency. Then, for any $\delta \in [0;1]$, with probability of at least $1-\delta$ over random m-samples S, simultaneously for all $Q \in \mathcal{M}_1^+(\mathcal{H})$ such that $Q \sim P$ we have:*

$$\mathbb{E}_{h\sim Q}[R(h)] \le \mathbb{E}_{h\sim Q}[R_S(h)] + \frac{\mathrm{KL}(Q||P) + \log(1/\delta)}{m^\alpha} + \frac{C^2}{2m^{1-\alpha}}.$$

Remark 1. *We provide Proposition 1 to evaluate the robustness of our approach. For instance, by comparing it with the PAC-Bayesian bound found in Germain et al. [11]. This discussion can be found in Section 6.1, where the bound from Germain et al. [11] is presented in detail.*

Remark 2. *At first glance, a naive remark: in order to control the rate of convergence of all the terms of the bound in Proposition 1 (as is often the case in classical PAC-Bayesian bounds), then the only case of interest is in fact $\alpha = \frac{1}{2}$. However, one could notice that the factor C^2 is not optimisable, while the KL is. In this way, if it appears that C^2 is too big, in practice, one wants to have the ability to attenuate its influence as much as possible and this may lead us to consider $\alpha < 1/2$. The following lemma answers this question.*

Lemma 1. *For any given $K_1 > 0$, the function $f_{K_1}(\alpha) := \frac{K_1}{m^\alpha} + \frac{C^2}{m^{1-\alpha}}$ reaches its minimum at*

$$\alpha_0 = \frac{1}{2} + \frac{1}{2\log(m)} \log\left(\frac{2K_1}{C^2}\right).$$

Proof. The explicit calculus of the f'_{K_1} and the resolution of $f'_{K_1}(\alpha) = 0$ provides the result. □

Remark 3. *Lemma 1 indicates that with a fixed "prior" P and "posterior" Q, taking $K_1 = \mathrm{KL}(Q||P) + \log(1/\delta)$, gives the optimised value of the bound in Proposition 1. We numerically show in Section 3.2 (first experiment there) that optimising α leads to significantly better results.*

Now the only remaining question is how to optimise the KL divergence. To do so, we may need to fix an "informed prior" to minimise the KL divergence with an interesting posterior. This idea has been studied by [16,17] and, more recently, by Mhammedi et al. [18], Rivasplata et al. [5], among others. We will adapt it to our problem in the simplest way.

We now introduce some additional notation. For a sample $s = (z_1, \dots, z_m)$ and $k \in \{1..m\}$, we define $s_{\le k} := \{z_1, \dots, z_k\}$ and $s_{>k} := \{z_{k+1}, \dots, z_m\}$. Then, similarly, for a random sample S, we have the splits $S_{\le k}$ and $S_{>k}$.

Proposition 2. *Let ℓ be $\texttt{HYPE}(K)$ compliant, with constant $K(h) = C$, and $\alpha_1, \alpha_2 \in \mathbb{R}$. Consider any "priors" $P_1 \in \mathcal{M}_1^+(\mathcal{H})$ (possibly dependent on $S_{>m/2}$) and $P_2 \in \mathcal{M}_1^+(\mathcal{H})$ (possibly dependent* *on $S_{\leq m/2}$). Then, for any $\delta \in [0;1]$, with probability of at least $1-\delta$ over random size-m samples S, simultaneously for all $Q \in \mathcal{M}_1^+(\mathcal{H})$ such that $Q \sim P_1$ and $Q \sim P_2$ we have:*

$$\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}[R_S(h)] + \frac{1}{2}\left(\frac{\mathrm{KL}(Q||P_1)+\log(2/\delta)}{(m/2)^{\alpha_1}} + \frac{C^2}{2(m/2)^{1-\alpha_1}}\right) + \frac{1}{2}\left(\frac{\mathrm{KL}(Q||P_2)+\log(2/\delta)}{(m/2)^{\alpha_2}} + \frac{C^2}{2(m/2)^{1-\alpha_2}}\right).$$

Proof. Let P_1, P_2, Q be as stated in Proposition 2. We first notice that by using Proposition 1 on the two halves of the sample, we obtain, with a probability of at least $1-\delta/2$:

$$\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}\left[\frac{1}{m/2}\sum_{i=1}^{m/2}\ell(h,Z_i)\right] + \frac{\mathrm{KL}(Q||P_1)+\log(2/\delta)}{(m/2)^{\alpha_1}} + \frac{C^2}{2(m/2)^{1-\alpha_1}}$$

and also with probability at least $1-\delta/2$:

$$\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}\left[\frac{1}{m/2}\sum_{i=1}^{m/2}\ell(h,Z_{m/2+i})\right] + \frac{\mathrm{KL}(Q||P_2)+\log(2/\delta)}{(m/2)^{\alpha_2}} + \frac{C^2}{2(m/2)^{1-\alpha_2}}.$$

Hence, with a probability of at least $1-\delta$, both inequalities hold, and the result follows by adding them and dividing by 2. □

Remark 4. *One can notice that the main difference between Proposition 2 and Proposition 1 lies in the implicit PAC-Bayesian paradigm that our priors must not depend on the data. With this last proposition, we implicitly allow P_1 to depend on $S_{>m/2}$ and P_2 on $S_{\leq m/2}$, which can in practice lead to far more accurate priors. We numerically show this fact in Section 3.2's second experiment. Note that this idea is not new and has been studied, for instance, in [19] for the specific case of SVMs.*

3.2. Numerical Experiments

Our experimental framework has been inspired by the work of [18].

Settings. We generate synthetic data for classification, and we are using the 0–1 loss. The data space is $\mathcal{Z} = \mathcal{X} \times \mathcal{Y} = \mathbb{R}^d \times \{0,1\}$ with $d \in \mathbb{N}$. The set of predictors $\mathcal{H}$ is parameterised with d-dimensional 'weight' vectors: $\mathcal{H} = \{h_w : \mathcal{X} \to \mathcal{Y} \mid w \in \mathbb{R}^d\}$. For simplicity, we identify h_w with w and we also identify the space $\mathcal{H}$, with the weight space $\mathcal{W} = \mathbb{R}^d$. For $z = (x,y) \in \mathcal{Z}$ and $w \in \mathcal{W}$, we define the loss as $\ell(w,z) := |\mathbb{1}\{\phi(w^\top x) > 1/2\} - y|$, where $\phi(r) = \frac{1}{1+e^{-r}}$. We want to learn an optimised predictor given a dataset $S = (Z_i)_{i=1..m}$ where $Z_i = (X_i, Y_i)$. To do so, we use *regularised logistic regression* and compute:

$$\hat{w}(S) := \arg\min_{w\in\mathcal{W}} \lambda\frac{||w||^2}{2} - \frac{1}{m}\sum_{i=1}^{m} y_i \log\Big(\phi(w^\top x_i)\Big) + (1-y_i)\log\Big(1-\phi(w^\top x_i)\Big) \quad (2)$$

where λ is a fixed regularisation parameter.

We also restrict the probability distributions (over $\mathcal{W} = \mathbb{R}^d$), considered for this learning problem. We consider the Gaussian distribution $\mathcal{N}(w, \sigma^2 I_d)$ with centre $w \in \mathbb{R}^d$ and diagonal covariance $\sigma^2 I_d \in \mathbb{R}^{d\times d}$ with $\sigma^2 > 0$.

Parameters. We set $\delta = 0.05, \lambda = 0.01$. We approximately solve Equation (2) by using the `minimize` function of the optimisation module in Python, with the Powell method. To approximate gaussian expectations, we use Monte-Carlo sampling.

Synthetic data. We generate synthetic data for $d = 10$ according to the following process: for a fixed sample size m, we draw $X_1, ..., X_m$ under the multivariate Gaussian distribution $\mathcal{N}(0, I_d)$ and for each i we compute the label if X_i as: $Y_i = \mathbb{1}\{\phi(w^{*\top} x_i) > 1/2\}$ where w^* is the vector formed by the d first digits of the number π.

Normalisation trick. Given the predictors shape, we notice that for any $w \in \mathcal{W}$:

$$\mathbb{1}\{\phi(w^\top x) > 1/2\} = 1 \quad \Leftrightarrow \quad \frac{1}{1+\exp(-w^\top x)} > \frac{1}{2} \quad \Leftrightarrow \quad w^\top x < 0.$$

Thus, the value of the prediction is exclusively determined by the sign of the inner product, and this quantity is definitely not influenced by the norm of the vector. Then, for any sample S, we call the **normalisation trick** the fact of considering $\hat{w}(S)/||\hat{w}(S)||$ instead of $\hat{w}(S)$ in our calculations. This process will not deteriorate the quality of the prediction and will considerably enhance the value of the KL divergence.

3.2.1. First experiment

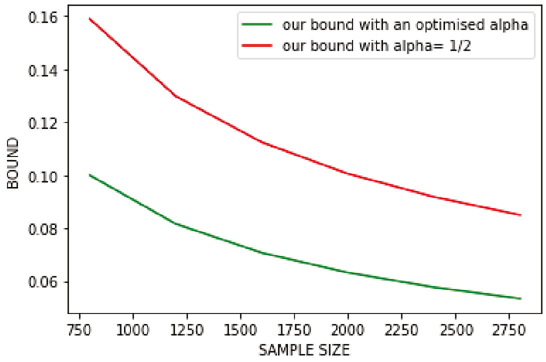
Our goal here is to highlight the point discussed in Remark 2, e.g., the influence of the parameter α in Proposition 1. We arbitrarily fix $\sigma_0^2 = 1/2$, and define our *naive prior* as $P_0 = \mathcal{N}(0, \sigma_0^2 I_d)$. For a fixed dataset S, we define our posterior as $P(S) := \mathcal{N}(\hat{h}(S), \sigma^2 I_d)$, with $\sigma^2 \in \{1/2, \dots, 1/2^J\}$ (for $J = \log_2(m)$) such that it is minimising the bound among candidates. We computed two curves: first, Proposition 1 with $\alpha = 1/2$ second, Proposition 1 again with α equals to the value proposed in Lemma 1. Notice that to compute this last bound, we first optimised our choice of posterior with $\alpha = 1/2$ and then optimised α, to be consistent with Lemma 1. Indeed, we proved this lemma by assuming that the KL divergence was already fixed, hence our optimisation process is in two steps. Note that we chose to apply the normalisation trick here, we then obtained the left curve of Figure 1.

Discussion. From this curve, we formulate several remarks. First, we remark on this specific case, our theorem provides a tight result in practice (with an error rate lesser than 10% for the bound with optimised alpha). Second, we can now confirm that choosing an optimised α leads to a tighter bound. In further studies, it will be relevant to adjust α with regards to the different terms of our bound instead of looking for an identical convergence rate for all terms.

3.2.2. Second Experiment

We now study Proposition 2 to see if an informed prior effectively provides a tighter bound than a naive one. We will use the notations introduced in Proposition 2. For a dataset S, we define $w_1(S) = w(S_{>m/2})$ as the vector resulting from the optimisation of Equation (2) on $S_{>m/2}$. Similarly, we define $w_2(S) := w(S_{\leq m/2})$. We arbitrarily fix $\sigma_0^2 = 1/2$, and define our *informed priors* as: $P_1 = \mathcal{N}(w_1(S), \sigma_0^2 I_d)$ and $P_2 = \mathcal{N}(w_2(S), \sigma_0^2 I_d)$. Finally, we define our posterior as $P(S) := \mathcal{N}(\hat{w}(S), \sigma^2 I_d)$, with $\sigma^2 \in \{1/2, ..., 1/2^J\}$ (for $J = \log_2(m)$) with σ^2 optimising the bound among the same candidate than the first experiment. We computed two curves: first, Proposition 1 with α optimised accordingly to Lemma 1 secondly, Proposition 2 with α_1, α_2 optimised as well, and informed priors as defined above. We chose to not apply the normalisation trick here, we then obtained the right curve of Figure 1.

Discussion. It is clear, that with this framework, having an informed prior is a powerful tool to enhance the quality of our bound. Notice that we voluntarily chose to not apply the normalisation trick here. The reason is that this trick appears to be too powerful in practice, and applying it leads to counterproductive results; to highlight our point: the bound without informed prior would be tighter than the one with informed prior. Furthermore, this trick is linked to the specific structure of our problem and is not valid for any classification problem. Thus, the idea of providing informed priors remains an interesting tool for most cases.



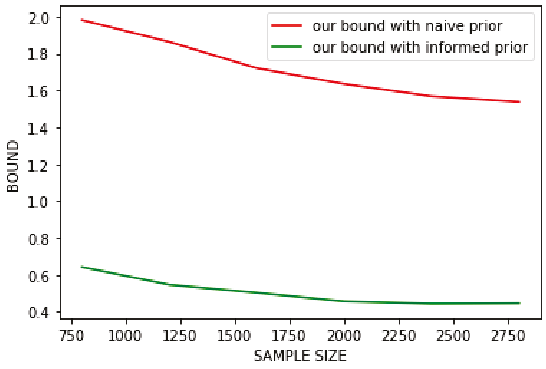


Figure 1. Above, result of the first experiment which highlight the importance of optimising α. Below, result of the second experiment which show how effective an informed prior is.

4. PAC Bayesian Bounds with Smoothed Estimator

We now move on to control the right-hand side term in Theorem 3 when K is not constant. A first step is to consider a transformed estimate of the risk, inspired by the truncated estimator from [20], also used in [21], and more recently in [13]. The following is inspired by the results of [13], which we summarise in Section 6.

The idea is to modify the estimator $R_S(h)$ for any h by introducing a threshold t and a function ψ which will attenuate the influence of the empirical losses $(\ell(h, Z_i))_{i=1..m}$ that exceed t.

Definition 2. ψ-risks. *For every $t > 0$, $\psi : \mathbb{R}^+ \to \mathbb{R}^+$, for any $h \in \mathcal{H}$, we define the empirical ψ-risk $R_{S,\psi,t}$ and the theoretical ψ-risk $R_{\psi,t}$ as follows:*

$$R_{S,\psi,t}(h) := \frac{t}{m}\sum_{i=1}^{m} \psi\left(\frac{\ell(h, Z_i)}{t}\right) \quad \text{and} \quad R_{\psi,t}(h) = \mathbb{E}_\mu\left[t\,\psi\left(\frac{\ell(h, Z)}{t}\right)\right]$$

where $Z \sim \mu$. Notice that $\mathbb{E}_S\left[R_{S,\psi,t}(h)\right] = R_{\psi,t}(h)$.

We now focus on what we call *softening functions*, i.e., functions that will temper high values of the loss function ℓ.

Definition 3. (Softening function). *We say that $\psi : \mathbb{R}^+ \to \mathbb{R}^+$ is a softening function if:*

- $\forall x \in [0; 1], \psi(x) = x$,
- *ψ is non-decreasing,*
- $\forall x \geq 1, \psi(x) \leq x$.

We let $\mathcal{F}$ denote the set of all softening functions.

Remark 5. *Notice that those three assumptions ensure that ψ is continuous at 1. For instance, the functions $f : x \mapsto x\mathbb{1}\{x \leq 1\} + \mathbb{1}\{x > 1\}$ and $g : x \mapsto x\mathbb{1}\{x \leq 1\} + (2\sqrt{x} - 1)\mathbb{1}\{x > 1\}$ are in $\mathcal{F}$. In Section 6 we compare these softening functions and those used by Holland [13].*

Using $\psi \in \mathcal{F}$, for a fixed threshold $t > 0$, the softened loss function $t\psi\left(\frac{\ell(h,z)}{t}\right)$ verifies for any $h \in \mathcal{H}, z \in \mathcal{Z}$:

$$t\,\psi\left(\frac{\ell(h,z)}{t}\right) \leq t\,\psi\left(\frac{K(h)}{t}\right)$$

because ψ is non-decreasing. In this way, the exponential moment in Theorem 3 can be far more controllable. The trade-off lies in the fact that softening ℓ (instead of taking directly ℓ) will deteriorate our ability to distinguish between two bad predictions when both of them are greater than t. For instance, if we choose $\psi \in \mathcal{F}$ such as $\psi = 1$ on $[1; +\infty)$ and $t > 0$, if $\psi(\ell(h,z)/t) = 1$ for a certain pair (h,z), then we cannot tell how far $\ell(h,z)$ is from t and we only can affirm that $\ell(h,z) \geq t$.

We now move on to the following lemma, which controls the shortfall between $\mathbb{E}_{h\sim Q}[R(h)]$ and $\mathbb{E}_{h\sim Q}[R_{\psi,t}(h)]$ for all $Q \in \mathcal{M}_1^+(\mathcal{H})$, for a given ψ and $t > 0$. To do that, we assume that K admits a finite moment under any posterior distribution:

$$\forall Q \in \mathcal{M}_1^+(\mathcal{H}),\ \ \mathbb{E}_{h\sim Q}[K(h)] < +\infty. \tag{3}$$

For instance, in the case of $\mathcal{H}$ identified with a weight space $\mathcal{W} = \mathbb{R}^N$, and if K is polynomial in $||w||$ (where $||.||$ denotes the Euclidean norm), then this assumption holds if we consider Gaussian priors and posteriors.

Lemma 2. *Assume that Equation (3) holds, and let $\psi \in \mathcal{F}, Q \in \mathcal{M}_1^+(\mathcal{H}), t > 0$. We have:*

$$\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}[R_{\psi,t}(h)] + \mathbb{E}_{h\sim Q}[K(h)\mathbb{1}\{K(h) \geq t\}].$$

Proof. Let $\psi \in \mathcal{F}, Q \in \mathcal{M}_1^+(\mathcal{H}), t > 0$. We have, for $h \in \mathcal{H}$:

$$\begin{aligned} &R(h) - R_{\psi,t}(h) \\ &\quad = \mathbb{E}_{Z\sim\mu}\left[\ell(h,Z) - t\psi\left(\frac{\ell(h,Z)}{t}\right)\right] \end{aligned}$$

and using that $\forall x \in [0,1], \psi(x) = x$,

$$= \mathbb{E}_{Z\sim\mu}\left[\left(\ell(h,Z) - t\psi\left(\frac{\ell(h,Z)}{t}\right)\right)\mathbb{1}\{\ell(h,Z) \geq t\}\right]$$

while using that $\ell(h,z) \leq K(h)$,

$$= \mathbb{E}_{Z\sim\mu}\left[\left(\ell(h,Z) - t\psi\left(\frac{\ell(h,Z)}{t}\right)\right)\mathbb{1}\{\ell(h,Z) \geq t\}\mathbb{1}\{K(h) \geq t\}\right]$$

and continuing:

$$\begin{aligned} &\leq \mathbb{E}_{Z\sim\mu}[\ell(h,Z)\mathbb{1}\{\ell(h,Z) \geq t\}]\mathbb{1}\{K(h) \geq t\} && (\psi \geq 0) \\ &\leq K(h)\mathbb{P}_{Z\sim\mu}\{\ell(h,Z) \geq t\}\mathbb{1}\{K(h) \geq t\} && (\ell(h,Z) \leq K(h)) \end{aligned}$$

Finally, by crudely bounding the probability by 1, we get:

$$R(h) \leq R_{\psi,t}(h) + K(h)\mathbb{1}\{K(h) \geq t\}.$$

Hence the result by integrating over $\mathcal{H}$ with respect to Q. $\square$

Finally we present the following theorem, which provides a PAC-Bayesian inequality bounding the theoretical risk by the empirical ψ-risk for $\psi \in \mathcal{F}$.

Theorem 4. *Let ℓ be $\texttt{HYPE}(K)$ compliant, and assume K satisfies Equation (3). Then for any $P \in \mathcal{M}_1^+(\mathcal{H})$ with no data dependency, for any $\alpha \in \mathbb{R}$, for any $\psi \in \mathcal{F}$ and for any $\delta \in [0;1]$, with probability of at least $1-\delta$ over size-m random samples S, simultaneously for all Q such that $Q \sim P$ we have:*

$$\begin{aligned}\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}\left[R_{S,\psi,t}(h)\right] &+ \mathbb{E}_{h\sim Q}[K(h)\mathbb{1}\{K(h)\geq t\}] \\ &+ \frac{\mathrm{KL}(Q||P) + \log\left(\frac{1}{\delta}\right)}{m^\alpha} \\ &+ \frac{1}{m^\alpha}\log\left(\mathbb{E}_{h\sim P}\left[\exp\left(\frac{t^2}{2m^{1-2\alpha}}\psi\left(\frac{K(h)}{t}\right)^2\right)\right]\right).\end{aligned}$$

Proof. Let $\psi \in \mathcal{F}$, we define the ψ-loss:

$$\ell_2(h,z) = t\psi\left(\frac{\ell(h,z)}{t}\right).$$

Since ψ is non decreasing, we have for all $(h,z) \in \mathcal{H} \times \mathcal{Z}$:

$$\ell_2(h,z) \leq t\psi\left(\frac{K(h)}{t}\right) := K_2(h).$$

Thus, we apply Theorem 3 to the learning problem defined with ℓ_2: for any α and $\delta \in (0,1)$, with probability at least $1-\delta$ over size-m random samples S, simultaneously for all Q such that $Q \sim P$ we have:

$$\begin{aligned}\mathbb{E}_{h\sim Q}\left[R_{\psi,t}(h)\right] \leq \mathbb{E}_{h\sim Q}\left[R_{S,\psi,t}(h)\right] &+ \frac{\mathrm{KL}(Q||P) + \log\left(\frac{1}{\delta}\right)}{m^\alpha} \\ &+ \frac{1}{m^\alpha}\log\left(\mathbb{E}_{h\sim P}\left[\exp\left(\frac{K_2(h)^2}{2m^{1-2\alpha}}\right)\right]\right).\end{aligned}$$

We then add $\mathbb{E}_{h\sim Q}[K(h)\mathbb{1}\{K(h)\geq t\}]$ on both sides of the latter inequality and apply Lemma 2. $\square$

Remark 6. *Notice that the function $\psi : x \mapsto x\mathbb{1}\{x \leq 1\} + \mathbb{1}\{x > 1\}$ is such that for any given prior P we have $\mathbb{E}_{h\sim P}\left[\exp\left(\frac{t^2}{2m^{1-2\alpha}}\psi\left(\frac{K(h)}{t}\right)^2\right)\right] < +\infty$. So the exponential moment can be controlled with a good choice of ψ. Thus the strength of Theorem 4 is to provide a PAC-Bayesian bound valid for any set of posterior measures verifying Equation (3). The choice of ψ minimising the bound is still an open problem.*

5. The Linear Regression Problem

5.1. Theoretical Result

We now focus on the celebrated linear regression problem and see how our theory translates to that particular learning problem. We assume that the data is a size-m random sample $S = (Z_i)_{i=1..m}$ where the Z_i are i.i.d. drawn from the distribution μ, and $Z_i = (X_i, Y_i)$ with $X_i \in \mathbb{R}^N$, $Y_i \in \mathbb{R}$.

Our goal here is to find the most accurate predictor h_w (with $w \in \mathbb{R}^N$), with respect to the loss function $\ell(h_w, z) = |\langle w, x\rangle - y|$, where $z = (x, y)$. We will make the following mild assumption: there exists $B, C \in \mathbb{R}+\backslash\{0\}$ such that for all $z = (x, y)$ drawn under μ:

$$||x|| \leq B \text{ and } |y| \leq C$$

where $||.||$ is the norm associated to the classical inner product of $\mathbb{R}^N$. Under this assumption we note that for all $z = (x, y)$ drawn according to μ, we have:

$$\ell(h_w, z) = |\langle w, x\rangle - y| \leq |\langle w, x\rangle| + |y| \leq ||w||.||x|| + |y| \leq B||w|| + C.$$

Thus we define $K(h_w) = B||w|| + C$ for $w \in \mathbb{R}^N$. If we first restrict ourselves to the framework of Section 2, we want to use Theorem 3 and doing so, our goal is to bound $\xi := \mathbb{E}_{w\sim P}\left[\exp\left(\frac{K(w)^2}{2m^{1-2\alpha}}\right)\right]$. The shape of K invites us to consider a Gaussian prior. Indeed, we notice that if $P = \mathcal{N}(0, \sigma^2 \mathbf{I}_N)$ with $0 < \sigma^2 < \frac{m^{1-2\alpha}}{B^2}$, then $\xi < +\infty$. Notice that we cannot take just any Gaussian prior, however with a small α, the condition $0 < \sigma^2 < \frac{m^{1-2\alpha}}{B^2}$ may become quite loose. Thus, we have the following:

Theorem 5. *Let $\alpha \in \mathbb{R}$ and $N \geq 6$. Assume that the loss ℓ is $\texttt{HYPE}(K)$ compliant with $K(h) = B||h|| + C$, with $B > 0, C \geq 0$. For a prior distribution, consider any Gaussian $P = \mathcal{N}(0, \sigma^2 \mathbf{I}_N)$ with $\sigma^2 = t\frac{m^{1-2\alpha}}{B^2}$, $0 < t < 1$. Then, for any $\delta \in [0;1]$, with probability of at least $1-\delta$ over size-m random samples S, simultaneously for all $Q \in \mathcal{M}_1^+(\mathcal{H})$ such that $P \sim Q$ we have:*

$$\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}[R_S(h)] + \frac{\mathrm{KL}(Q||P) + \log(2/\delta)}{m^\alpha} + \frac{C^2}{2m^{1-\alpha}}\left(1 + f(t)^{-1}\right)$$
$$+ \frac{N}{m^\alpha}\left(\log\left(1 + \left(\frac{C}{\sqrt{2f(t)m^{1-2\alpha}}}\right)\right) + \log\left(\frac{1}{\sqrt{1-t}}\right)\right)$$

where $f(t) = \frac{1-t}{t}$.

The proof is deferred to Section 7.2. To compare our result with those found in the literature, we can fix $\alpha = 1/2$. Doing so, we lose the dependency in m for the choice of the variance of the prior (which now only depends on B), but we recover the classic decreasing factor $1/\sqrt{m}$.

Remark 7. *Notice that for now we did not use Section 4, even if we could (because K is polynomial in $||w||$ and we consider Gaussian priors and posteriors, so Equation (3) is satisfied). Doing so, we obtained a bound which appears to depend linearly on the dimension N. In practice, N may be too big, and in this case, introducing an adapted softening function ψ (one can think for instance of $\psi(x) = x\mathbb{1}\{x \leq 1\} + \mathbb{1}\{x > 1\}$) is a powerful tool to attenuate the weight of the exponential moment. This also extends the class of authorised Gaussian priors by avoidance, to stick with a variance $\sigma^2 = t\frac{m^{1-2\alpha}}{B^2}$, $0 < t < 1$.*

5.2. Numerical Experiment

5.2.1. Setting

In this section we apply Theorem 5 on a concrete linear regression problem. The situation is as follows: we want to approximate the function $f(x) = \sqrt{\langle w^*, x\rangle}$, where $w^* \in \mathbb{R}^d$. We assume that $\mathcal{W} = [-c, c]^d$ so that w^* lies in an hypercube centred at 0 of half-side $c > 0$, i.e., the set $\{(w_i)_{i=1,\dots,d} \mid \forall i, |w_i| \leq c\}$. Doing so we have $||w^*|| \leq c\sqrt{d}$.

Furthermore, we assume that input data are drawn inside a hypercube of half-side $e > 0$, i.e., $\mathcal{X} = [-e, e]^d$. Doing so we have for any data x, $||x|| \leq e\sqrt{d}$.

For any data $x \in \mathbb{R}^d$, we define $y = f(x)$. As before, we identify the hypothesis set $\mathcal{H}$ with the weight space $\mathcal{W} = \mathbb{R}^d$. As described in Section 5.1, we set $\ell(h_w, x, y) = |\langle w, x\rangle - y|$. We then remark that for any (w, x, y):

$$\begin{aligned}\ell(h_w, x, y) &\leq |\langle w, x\rangle| + |y| \leq ||w||||x|| + |\sqrt{\langle w^*, x\rangle}| \\ &\leq e\sqrt{d}||w|| + \sqrt{||w^*||.||x||} \leq e\sqrt{d}||w|| + \sqrt{c\sqrt{d}.e\sqrt{d}} \\ &\leq e\sqrt{d}||w|| + \sqrt{cde}.\end{aligned}$$

Then we can define $B = e\sqrt{d}$ and $C = \sqrt{cde}$ to apply Theorem 5. We restrict (as before) the class of distributions over $\mathcal{W}$ to be d-dimensional Gaussians:

$$\left\{\mathcal{N}(w, \sigma^2 I_d) \mid w \in \mathcal{H}, \sigma^2 \in \mathbb{R}^+\right\},$$

which is the set of candidate distributions for this learning problem. Recall that in practice, given a fixed $\alpha \in \mathbb{R}$, we are only allowed to consider priors such that their variance $\sigma^2 \in \left]0; \frac{m^{1-2\alpha}}{B^2}\right[$. We want to learn an optimised predictor (posterior) given a random dataset $S = ((X_i, Y_i))_{i=1,\ldots,m}$. To do so, we consider synthetic data.

5.2.2. Synthetic Data

We draw w^* under a Gaussian (with mean 0 and standard deviation equal to 5) truncated to the hypercube centered at 0 of the half-side $c > 0$. We generate synthetic data according to the following process: for a fixed sample size m, we draw $X_1, \ldots, X_m$ under a Gaussian (with mean 0 and standard deviation equal to 5) truncated to the hypercube centered at 0 of the half-side $e > 0$.

5.2.3. Experiment

First, we fix $c = e = 10$. Our goal here is to obtain a generalisation bound on our problem. We fix arbitrarily, for a fixed $\alpha \in \mathbb{R}$, $t_0 = 1/2$ and $\sigma_0^2 = t_0 \frac{m^{1-2\alpha}}{B^2}$ and we define our *naive prior* as $P_0 = \mathcal{N}(0, \sigma_0^2 I_d)$. For a given dataset S, we define our posterior as $Q(S) := \mathcal{N}(\hat{w}(S), \sigma^2 I_d)$, with $\sigma^2 \in \{\sigma_0^2/2, \ldots, \sigma_0^2/2^J\}$ ($J = \log_2(m)$), such that it is minimising the bound among candidates. Note that all the previously defined parameters are dependent on α, which is why we choose $\alpha \in \{i/\texttt{step} \mid 0 \leq i \leq \texttt{step}\}$ for `step` a fixed integer (in practice `step` = 8 or 16) and we take the value of α minimising the bound among the candidates as well. Figure 2 contains two figures, one with $d = 10$, the other with $d = 50$. On each figure are computed the right-hand side term in Theorem 5 with an optimised α for each step.

5.2.4. Discussion

To the the best of our knowledge, this is the first attempt to numerically compute PAC-Bayes bounds for unbounded problems, making it impossible to compare to other results. We stress, however, that obtaining numerical values for the bound without assuming a bounded loss is a significant first step. Furthermore, we consider a rather hard problem: f is not linear, so we cannot rely on a linear approximation fitting perfectly data, and the larger the dimension, the larger the error, as illustrated by Figure 2. Thus, for any posterior Q, the quantity $\mathbb{E}_{h\sim Q}[R(h)]$ is potentially large in practice and our bound might not be tight. Finally, notice that optimising α (instead of taking $\alpha = 1/2$ to recover a classic convergence rate) leads to a significantly better bound. A numerical example of this assertion is presented in Section 3.2. We aim to conduct further studies to consider the convergence rate as an hyperparameter to optimise, rather than selecting the same rate for all terms in the bound.

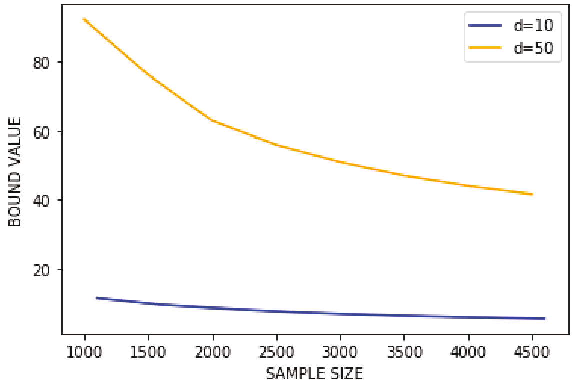


Figure 2. Evaluation of the right hand side in Theorem 5 with $d = 10$ and $d = 50$.

6. Existing Work

6.1. Germain et al., 2016

In Germain et al. [11] (Section 4), a PAC-Bayesian bound has been provided for all *sub-gamma* losses with a variance t^2 and scale parameter $c > 0$, under a data distribution μ and a prior P, i.e. losses such that for every $\lambda \in \left(0, \frac{1}{c}\right)$ the following is satisfied:

$$\log\left(\frac{1}{\delta}\mathbb{E}_{h\sim P}\mathbb{E}_S\, e^{\lambda(R(h)-R_S(h))}\right) \leq \frac{t^2}{c^2}(-\log(1-c\lambda)-\lambda c) \leq \frac{\lambda^2 t^2}{2(1-c\lambda)}.$$

Note that a sub-gamma loss (with regards to μ and P) is potentially unbounded. Germain et al. then propose the following PAC-Bayesian bound:

Theorem 6. *Ref. [11]. If the loss ℓ is sub-gamma with a variance t^2 and scale parameter c, under the data distribution μ and a fixed prior $P \in \mathcal{H}$, then for any $\delta \in [0;1]$, with probability $1-\delta$ over size-m random samples, simultaneously for all $Q \ll P$ we have:*

$$\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}[R_S(h)] + \frac{\mathrm{KL}(Q||P) + \log(1/\delta)}{m} + \frac{t^2}{2(1-c)}.$$

Theorem 6 will be quoted several times in this paper given that it is a concrete PAC Bayesian bound provided with the will to overcome the constraint of a bounded loss. It is also one of the only one found in the literature.

Can we apply this theorem to the bounded case? The answer is yes: we remark that thanks to Hoeffding's lemma, if ℓ is bounded by $C > 0$, then for any $h \in \mathcal{H}$ it holds that $R_S(h) - R(h) \in [-C, C]$ almost surely. So, $\forall \lambda \in \mathbb{R}$, $\log \mathbb{E}_{z\sim\mu}\left[e^{\lambda(R(h)-R_S(h))}\right] \leq \frac{\lambda^2 C^2}{2}$. Therefore, for any prior P, we have:

$$\log \mathbb{E}_{h\sim P}\mathbb{E}_{z\sim\mu}\left[e^{\lambda(R(h)-R_S(h))}\right] \leq \frac{\lambda^2 C^2}{2}.$$

Thus, ℓ is sub-gamma with variance C^2 and scale parameter 0. Then, Theorem 6 can be applied with $t^2 = C^2$, $c = 0$.

Comparison with Proposition 1. We remark that by taking $K = C$ and $\alpha = 1$ in Proposition 1, we are recovering Theorem 6. However, our approach allows us to say that if we can obtain a more precise form of K such that $\forall h \in \mathcal{H}$, $K(h) \leq C$ and K is non-constant, Theorem 3, will ensure that:

$$\frac{1}{m^\alpha} \log\left(\mathbb{E}_{h\sim P}\left[\exp\left(\frac{K(h)^2}{2m^{1-2\alpha}}\right)\right]\right) \leq \frac{C^2}{2m^{1-\alpha}}.$$

Thus, having precise information on the behavior of the loss function ℓ, with regards to the predictor h, allows us to obtain a tighter control of the exponential moment, and hence a tighter bound.

Remark 8. *We can see that Theorem 6 cannot control the factor $C^2/2$. However, Ref. [11] remarked on this apparent weakness and partially corrected this issue [11] (Section 4, Equations (13) and (14)). Indeed, they proposed to balance the influence of m between the different terms of the PAC-Bayes bound by providing the same convergence rate in $1/\sqrt{m}$ to all terms.*

We can then see Proposition 1 as a proper generalisation of Germain et al. [11] (Section 4, Equations (13) and (14)). Indeed, our bound exhibits properly the influence of the parameter α. Thus, we understand (and Lemma 1 proves it) that the choice of α deserves a study in itself in the way it is now a parameter of our optimisation problem. This fact has already been highlighted in Alquier et al. [10] (Theorem 4.1) (where $\lambda := m^\alpha$).

6.2. Holland, 2019

In [13], Holland proposed a PAC Bayesian inequality with unbounded loss. For that, he introduced a function ψ verifying a few specific conditions, different to those used in Section 4 to define our set of softening functions. Indeed, he considered a function ψ such that:

- ψ is bounded,
- ψ is non decreasing,
- it exists $b > 0$ such that for all $u \in \mathbb{R}$:

$$-\log\left(1 - u + \frac{u^2}{b}\right) \leq \psi(u) \leq \log\left(1 + u + \frac{u^2}{b}\right). \tag{4}$$

We remark that, as Holland did, we supposed that our softening functions are non-decreasing. We chose softening functions to be equal to the identity function ($x \mapsto x$) on $[0,1]$, which is quite restrictive. However, we are imposing softening functions to be lesser than the identity on $[1,+\infty)$; whereas, Holland supposed ψ to be bounded and satisfy Equation (4). A concrete example of such a function ψ, lies in the piecewise polynomial function of Catoni and Giulini [21], defined by:

$$\psi(u) = \begin{cases} -2\sqrt{2}/3 & \text{if } u \leq -\sqrt{2} \\ u - u^3/6 & \text{if } u \in [-2\sqrt{2}/3, 2\sqrt{2}/3] \\ 2\sqrt{2}/3 & \text{otherwise.} \end{cases}$$

As in Section 4, we are considering the ψ-empirical risk $R_{S,\psi,t}$ for any $t > 0$. Holland provided his theorem given the fact the following assumptions are realised:

- Bounds on lower-order moments. For all $h \in \mathcal{H}$, we have $\mathbb{E}_{Z\sim\mu}[\ell(h,Z)^2] \leq M_2 < +\infty$ and $\mathbb{E}_{Z\sim\mu}[\ell(h,Z)^3] \leq M_3 < +\infty$.
- Bounds on the risk. For all $h \in \mathcal{H}$, we suppose $R(h) \leq \sqrt{mM_2/(4\log(\delta^{-1})}$.
- Large enough confidence, we require $\delta \leq e^{-1/9}$.

Now we can state Holland's theorem.

Theorem 7. *Ref. [13]. Let P be a prior distribution on model $\mathcal{H}$. Let the three assumptions listed above hold. Setting $t^2 = mM_2/(2\log(\delta^{-1}))$, then for any $\delta \in [0;1]$, with probability of at least $1-\delta$ over the random draw of the size-m sample S, simultaneously for all Q it holds that:*

$$\begin{aligned}\mathbb{E}_{h\sim Q}[R(h)] \leq \mathbb{E}_{h\sim Q}\left[R_{S,\psi,t}(h)\right] + \frac{1}{\sqrt{m}}\left(\mathrm{KL}(Q||P) + \frac{1}{2}\log\left(\frac{8\pi M_2}{\delta^2}\right) - 1\right) \\ + \frac{1}{\sqrt{m}}\nu^*(\mathcal{H}) + O\left(\frac{1}{m}\right)\end{aligned}$$

where:

$$\nu^*(\mathcal{H}) := \frac{\mathbb{E}_{h\sim P}\left[\exp\left(\sqrt{m}(R(h) - R_{S,\psi,t}(h))\right)\right]}{\mathbb{E}_{h\sim P}\left[\exp\left(R(h) - R_{S,\psi,t}(h)\right)\right]}.$$

7. Proofs

7.1. Proof of Theorem 1

Proof. Let $F : \mathbb{R}^+ \times \mathbb{R}^+ \mapsto \mathbb{R}$ be a convex function, P a fixed prior, and $\delta \in [0,1]$. Since $\mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right]$ is a nonnegative random variable, we know that, by Markov's inequality, for any $h \in \mathcal{H}$:

$$\mathbb{P}\left(\mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right] > \frac{1}{\delta}\mathbb{E}_S\, \mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right]\right) \leq \delta.$$

So with probability of at least $1-\delta$, we have:

$$\mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right] \leq \frac{1}{\delta}\mathbb{E}_S\, \mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right] = \frac{\mathcal{X}}{\delta}.$$

Applying the log function on each side of this inequality gives us with probability of at least $1-\delta$ over samples S:

$$\log\left(\mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right]\right) \leq \log\left(\frac{\mathcal{X}}{\delta}\right).$$

We now rename $A := \log\left(\mathbb{E}_{h\sim P}\left[e^{F(R_S(h),R(h))}\right]\right)$.

Furthermore, if we denote by $\frac{dQ}{dP}$ the Radon-Nikodym derivative of Q with respect to P when $Q \ll P$, we then have, for all Q such that $Q \sim P$:

$$\begin{aligned}A &= \log\left(\mathbb{E}_{h\sim Q}\left[\frac{dP}{dQ}e^{F(R_S(h),R(h))}\right]\right) \\ &= \log\left(\mathbb{E}_{h\sim Q}\left[\left(\frac{dQ}{dP}\right)^{-1}e^{F(R_S(h),R(h))}\right]\right) \qquad \left(\tfrac{dP}{dQ} = \left(\tfrac{dQ}{dP}\right)^{-1}\right)\end{aligned}$$

and by concavity of log and Jensen's inequality,

$$\begin{aligned}&\geq -\mathbb{E}_{h\sim Q}\left[\log\left(\frac{dQ}{dP}\right)\right] + \mathbb{E}_{h\sim Q}[F(R_S(h), R(h))] \\ &= -\mathrm{KL}(Q||P) + \mathbb{E}_{h\sim Q}[F(R_S(h), R(h))]\end{aligned}$$

while by convexity of F with Jensen's inequality,

$$\geq -\mathrm{KL}(Q||P) + F\left(\mathbb{E}_{h\sim Q}[R_S(h)], \mathbb{E}_{h\sim Q}[R(h)]\right).$$

Hence, for Q such that $Q \sim P$,

$$F\big(\mathbb{E}_{h\sim Q}[R_S(h)], \mathbb{E}_{h\sim Q}[R(h)]\big) \leq \mathrm{KL}(Q||P) + A.$$

So with probability $1-\delta$, for Q such that $Q \sim P$,

$$F\big(\mathbb{E}_{h\sim Q}[R_S(h)], \mathbb{E}_{h\sim Q}[R(h)]\big) \leq \mathrm{KL}(Q||P) + \log\Big(\frac{\chi}{\delta}\Big).$$

This completes the proof of Theorem 1. $\square$

7.2. Proof of Theorem 5

We first provide a technical property. Recall that:

$$\xi = \mathbb{E}_{h\sim P}\left[\exp\left(\frac{K(h)^2}{2m^{1-2\alpha}}\right)\right].$$

Proposition 3. *Let $\alpha \in \mathbb{R}$. Suppose the loss ℓ is* HYPE(K) *compliant with $K(h) = B||h|| + C$, with $B > 0$, $C \geq 0$. Then, for any Gaussian prior $P = \mathcal{N}(0, \sigma^2 \mathbf{I}_N)$ with $\sigma^2 = t\frac{m^{1-2\alpha}}{B^2}$, $0 < t < 1$ and $N \geq 6$ we have:*

$$\xi \leq 2\exp\left(\frac{C^2}{2m^{1-2\alpha}f(t)}(1+f(t))\right)\frac{1}{\left(\sqrt{1-t}\right)^N}\left(1+\left(\frac{C}{\sqrt{2f(t)m^{1-2\alpha}}}\right)\right)^{N-1}$$

with $f(t) = \frac{1-t}{t}$.

Proof. We recall that $\sigma^2 = t\frac{m^{1-2\alpha}}{B^2}$. By expliciting the expectation and $K(h)$ we thus obtain:

$$\begin{aligned}
\xi &= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^N \int_{h\in\mathbb{R}^N} \exp\left(\frac{(B||h||+C)^2}{2m^{1-2\alpha}} - \frac{||h||^2B^2}{2tm^{1-2\alpha}}\right)dh \\
&= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^N \int_{h\in\mathbb{R}^N} \exp\left(-\frac{1}{2m^{1-2\alpha}}\left(f(t)B^2||h||^2 - 2BC||h|| - C^2\right)\right)dh \\
&= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^N \int_{h\in\mathbb{R}^N} \exp\left(-\frac{B^2f(t)}{2m^{1-2\alpha}}\left(||h||^2 - \frac{2C||h||}{Bf(t)} - \frac{C^2}{B^2f(t)}\right)\right)dh \\
&= \exp\left(\frac{C^2}{2m^{1-2\alpha}f(t)}(1+f(t))\right)\frac{1}{(\sqrt{2\pi\sigma^2})^N}\int_{h\in\mathbb{R}^N}\exp\left(-\frac{B^2f(t)}{2m^{1-2\alpha}}\left(||h|| - \frac{C}{Bf(t)}\right)^2\right)dh.
\end{aligned}$$

We will use the spherical coordinates in N-dimensional Euclidean space given in [22]:

$$\varphi : (h_1, ..., h_N) \to (r, \varphi_1, ..., \varphi_{N-1})$$

where especially $r = ||h||$ and also the Jacobian of ϕ is given by:

$$d^N V = r^{N-1}\prod_{k=1}^{N-2}\sin^k(\varphi_{N-1-k}) = r^{N-1}d_{S^{N-1}}V.$$

Let us also precise that as given in Blumenson [22] (page 66), we have that the surface of the sphere of radius 1 in N-dimensional space is:

$$\int_{\varphi_1,\ldots,\varphi_{N-1}} d_{S^{N-1}}V\, d\varphi_1 \ldots d\varphi_{N-1} = \frac{2\sqrt{\pi}^N}{\Gamma\left(\frac{N}{2}\right)}$$

where Γ is the Gamma function defined as:

$$\Gamma(x) = \int_0^{+\infty} t^{x-1}e^{-t}dt \quad \text{for } x > -1.$$

Then, if we set:

$$A := \int_{h\in\mathbb{R}^N} \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}\left(||h|| - \frac{C}{Bf(t)}\right)^2\right)dh$$

we obtain by a change of variable:

$$\begin{aligned}
A &= \int_{r,\varphi_1,\dots,\varphi_{N-1}} \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}\left(r - \frac{C}{Bf(t)}\right)^2\right)d^N V\, dr d\varphi_1 \dots d\varphi_{N-1} \\
&= \left(\frac{2\sqrt{\pi}^N}{\Gamma\left(\frac{N}{2}\right)}\right)\int_{r=0}^{+\infty} \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}\left(r - \frac{C}{Bf(t)}\right)^2\right)r^{N-1}dr \\
&= \left(\frac{2\sqrt{\pi}^N}{\Gamma\left(\frac{N}{2}\right)}\right)\int_{r=-\frac{C}{Bf(t)}}^{+\infty} \left(r + \frac{C}{Bf(t)}\right)^{N-1} \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}r^2\right)dr \\
&= \left(\frac{2\sqrt{\pi}^N}{\Gamma\left(\frac{N}{2}\right)}\right)\sum_{k=0}^{N-1}\binom{N-1}{k}\left(\frac{C}{Bf(t)}\right)^{N-k-1}\int_{r=-\frac{C}{Bf(t)}}^{+\infty} r^k \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}r^2\right)dr.
\end{aligned}$$

We fix a random variable X such that:

$$X \sim \mathcal{N}\left(0, \frac{m^{1-2\alpha}}{B^2(f(t))}\right).$$

We then have for any k positive integer, if k is even:

$$\begin{aligned}
\int_{r=-\frac{C}{Bf(t)}}^{+\infty} r^k \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}r^2\right)dr &\le \int_{r=-\infty}^{+\infty} r^k \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}r^2\right)dr \\
&\le \sqrt{2\pi\frac{m^{1-2\alpha}}{B^2 f(t)}}\mathbb{E}[|X|^k].
\end{aligned}$$

And if k is odd:

$$\begin{aligned}
\int_{r=-\frac{C}{Bf(t)}}^{+\infty} r^k \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}r^2\right)dr &\le \int_{r=0}^{+\infty} r^k \exp\left(-\frac{B^2 f(t)}{2m^{1-2\alpha}}r^2\right)dr \\
&\le \sqrt{2\pi\frac{m^{1-2\alpha}}{B^2 f(t)}}\mathbb{E}[|X|^k \mathbb{1}(X \ge 0)] \\
&\le \sqrt{2\pi\frac{m^{1-2\alpha}}{B^2 f(t)}}\mathbb{E}[|X|^k].
\end{aligned}$$

So we have:

$$A \le \left(\frac{2\sqrt{\pi}^N}{\Gamma\left(\frac{N}{2}\right)}\right)\sum_{k=0}^{N-1}\binom{N-1}{k}\left(\frac{C}{Bf(t)}\right)^{N-k-1}\sqrt{2\pi\frac{m^{1-2\alpha}}{B^2 f(t)}}\mathbb{E}[|X|^k].$$

As precised in [23], we have for any k:

$$\mathbb{E}[|X|^k] = \left(\sqrt{\frac{m^{1-2\alpha}}{B^2 f(t)}}\right)^k 2^{k/2}\frac{\Gamma\left(\frac{k+1}{2}\right)}{\sqrt{\pi}}.$$

So finally:

$$A \leq 2\sqrt{\pi}^N \sum_{k=0}^{N-1} \binom{N-1}{k} \left(\frac{C}{Bf(t)}\right)^{N-k-1} \left(\sqrt{\frac{2m^{1-2\alpha}}{B^2 f(t)}}\right)^{k+1} \frac{\Gamma\left(\frac{k+1}{2}\right)}{\Gamma\left(\frac{N}{2}\right)}.$$

Lemma 3. *If $N \geq 6$, then:*

$$\max_{k=0..N-1} \frac{\Gamma\left(\frac{k+1}{2}\right)}{\Gamma\left(\frac{N}{2}\right)} = 1.$$

Proof. As precised in the introduction of Srinivasan and Zvengrowski [24], Gauss [25] (page 147) proved that on the interval $[x_0, +\infty)$ where $x_0 \in [1.46, 1.47]$, Γ is a monotonic increasing function. So, for $N - 1 \geq k \geq 2, \Gamma(\frac{k+1}{2}) \leq \Gamma(\frac{N}{2})$. And because $\Gamma(1/2) = \sqrt{\pi}, \Gamma(1) = 1$, we have:

$$\max_{k=0..N-1} \frac{\Gamma\left(\frac{k+1}{2}\right)}{\Gamma\left(\frac{N}{2}\right)} = \max\left(\frac{\sqrt{\pi}}{\Gamma\left(\frac{N}{2}\right)}, \frac{\Gamma\left(\frac{N-1+1}{2}\right)}{\Gamma\left(\frac{N}{2}\right)}\right) = \max\left(\frac{\sqrt{\pi}}{\Gamma\left(\frac{N}{2}\right)}, 1\right)$$

Because $N \geq 6$, and Γ is monotone and increasing on $[3; +\infty]$, we have $\Gamma(N/2) \geq \Gamma(3) \geq \sqrt{\pi}$. Hence the result. □

Using Lemma 3 allows us to write:

$$A \leq 2\sqrt{\pi}^N \sum_{k=0}^{N-1} \binom{N-1}{k} \left(\frac{C}{Bf(t)}\right)^{N-k-1} \left(\sqrt{\frac{2m^{1-2\alpha}}{B^2 f(t)}}\right)^{k+1}.$$

We recall that $\sigma^2 = t\frac{m^{1-2\alpha}}{B^2}$ and $f(t) = \frac{1-t}{t}$. Then we can write:

$$A \leq 2\sqrt{\pi}^N \sum_{k=0}^{N-1} \binom{N-1}{k} \left(\frac{C}{Bf(t)}\right)^{N-k-1} \left(\sqrt{\frac{2\sigma^2}{1-t}}\right)^{k+1}.$$

We now conclude with the final bound on ξ:

$$\begin{aligned}
\xi &\leq \exp\left(\tfrac{C^2}{2m^{1-2\alpha} f(t)}(1+f(t))\right)\tfrac{1}{(\sqrt{2\pi\sigma^2})^N}\ A \\
&\leq \exp\left(\tfrac{C^2}{2m^{1-2\alpha} f(t)}(1+f(t))\right)\tfrac{1}{(\sqrt{2\pi\sigma^2})^N} 2\sqrt{\pi}^N\ \textstyle\sum_{k=0}^{N-1}\binom{N-1}{k}\left(\tfrac{C}{Bf(t)}\right)^{N-k-1}\left(\sqrt{\tfrac{2\sigma^2}{1-t}}\right)^{k+1} \\
&\leq 2\exp\left(\tfrac{C^2}{2m^{1-2\alpha} f(t)}(1+f(t))\right)\textstyle\sum_{k=0}^{N-1}\binom{N-1}{k}\left(\tfrac{C}{Bf(t)}\right)^{N-k-1}\left(\sqrt{\tfrac{1}{1-t}}\right)^{k+1}\left(\sqrt{\tfrac{B^2}{2tm^{1-2\alpha}}}\right)^{N-k-1} \\
&\leq 2\exp\left(\tfrac{C^2}{2m^{1-2\alpha} f(t)}(1+f(t))\right)\textstyle\sum_{k=0}^{N-1}\binom{N-1}{k}\left(\tfrac{C\sqrt{t}}{(1-t)\sqrt{2m^{1-2\alpha}}}\right)^{N-k-1}\left(\sqrt{\tfrac{1}{1-t}}\right)^{k+1} \\
&\leq 2\frac{\exp\left(\frac{C^2}{2m^{1-2\alpha} f(t)}(1+f(t))\right)}{\left(\sqrt{1-t}\right)^N}\textstyle\sum_{k=0}^{N-1}\binom{N-1}{k}\left(\tfrac{C}{\sqrt{2f(t)m^{1-2\alpha}}}\right)^{N-k-1} \\
&\leq 2\frac{\exp\left(\frac{C^2}{2m^{1-2\alpha} f(t)}(1+f(t))\right)}{\left(\sqrt{1-t}\right)^N}\left(1+\left(\tfrac{C}{\sqrt{2f(t)m^{1-2\alpha}}}\right)\right)^{N-1}.
\end{aligned}$$

This completes the proof of Proposition 3. □

Proof of Theorem 5. We combine Theorem 3 with Proposition 3. We also upper-bound $N-1$ by N. □

Author Contributions: Conceptualization, M.H., B.G. and J.S.-T.; Formal analysis, M.H., B.G. and O.R.; Project administration, B.G.; Supervision, B.G.; Writing—original draft, M.H., B.G. and O.R.; Writing—review and editing, M.H., B.G., O.R. and J.S.-T. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by the U.S. Army Research Laboratory and the U. S. Army Research Office, and by the U.K. Ministry of Defence and the U.K. Engineering and Physical Sciences Research Council (EPSRC) under grant number EP/R013616/1. BG acknowledges partial support from the French National Agency for Research, grants ANR-18-CE40-0016-01 and ANR-18-CE23-0015-02.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shawe-Taylor, J.; Williamson, R.C. A PAC analysis of a Bayes estimator. In Proceedings of the 10th Annual Conference on Computational Learning Theory, Nashville, TN, USA, 6–9 July 1997; ACM: New York, NY, USA, 1997; pp. 2–9.
2. McAllester, D.A. Some PAC-Bayesian theorems. In Proceedings of the Eleventh Annual Conference on Computational Learning Theory, Madison, WI, USA, 24–26 July 1998; ACM: New York, NY, USA, 1998; pp. 230–234.
3. McAllester, D.A. PAC-Bayesian model averaging. In Proceedings of the Twelfth Annual Conference on Computational Learning Theory, Santa Cruz, CA, USA, 7–9 July 1999; ACM: New York, NY, USA, 1999; pp. 164–170.
4. Guedj, B. A Primer on PAC-Bayesian Learning. *arXiv* **2019**, arXiv:stat.ML/1901.05353.
5. Rivasplata, O.; Kuzborskij, I.; Szepesvári, C.; Shawe-Taylor, J. PAC-Bayes Analysis Beyond the Usual Bounds. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, Online, 6–12 December 2020.
6. Seeger, M. PAC-Bayesian Generalization Error Bounds for Gaussian Process Classification. *J. Mach. Learn. Res.* **2002**, *3*, 233–269.
7. Langford, J. Tutorial on practical prediction theory for classification. *J. Mach. Learn. Res.* **2005**, *6*, 273–306.
8. Catoni, O. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*; Institute of Mathematical Statistics: Waite Hill, OH, USA, 2007.
9. Germain, P.; Lacasse, A.; Laviolette, F.; Marchand, M. PAC-Bayesian Learning of Linear Classifiers. In Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, QC, Canada, 14–18 June 2009; Association for Computing Machinery: New York, NY, USA, 2009; pp. 353–360.
10. Alquier, P.; Ridgway, J.; Chopin, N. On the properties of variational approximations of Gibbs posteriors. *J. Mach. Learn. Res.* **2016**, *17*, 1–41.
11. Germain, P.; Bach, F.; Lacoste, A.; Lacoste-Julien, S. PAC-Bayesian Theory Meets Bayesian Inference. In *Advances in Neural Information Processing Systems 29*; Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2016; pp. 1884–1892.
12. Alquier, P.; Guedj, B. Simpler PAC-Bayesian bounds for hostile data. *Mach. Learn.* **2018**, *107*, 887–902. [CrossRef]
13. Holland, M. PAC-Bayes under potentially heavy tails. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H., Beygelzimer, A., d Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2019; pp. 2715–2724.
14. Kuzborskij, I.; Szepesvári, C. Efron-Stein PAC-Bayesian Inequalities. *arXiv* **2019**, arXiv:1909.01931.
15. Shalaeva, V.; Fakhrizadeh Esfahani, A.; Germain, P.; Petreczky, M. Improved PAC-Bayesian Bounds for Linear Regression. In Proceedings of the AAAI 2020—Thirty-Fourth AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020.
16. Lever, G.; Laviolette, F.; Shawe-Taylor, J. Distribution-Dependent PAC-Bayes Priors. In *Algorithmic Learning Theory*; Hutter, M., Stephan, F., Vovk, V., Zeugmann, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 119–133.
17. Lever, G.; Laviolette, F.; Shawe-Taylor, J. Tighter PAC-Bayes Bounds through Distribution-Dependent Priors. *Theor. Comput. Sci.* **2013**, *473*, 4–28. [CrossRef]
18. Mhammedi, Z.; Grünwald, P.; Guedj, B. PAC-Bayes Un-Expected Bernstein Inequality. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H., Beygelzimer, A., d Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2019; pp. 12202–12213.
19. Parrado-Hernández, E.; Ambroladze, A.; Shawe-Taylor, J.; Sun, S. PAC-Bayes bounds with data dependent priors. *J. Mach. Learn. Res.* **2012**, *13*, 3507–3531.
20. Catoni, O. Challenging the empirical mean and empirical variance: A deviation study. *Ann. Inst. H. Poincaré Probab. Statist.* **2012**, *48*, 1148–1185. [CrossRef]

21. Catoni, O.; Giulini, I. Dimension-free PAC-Bayesian bounds for matrices, vectors, and linear least squares regression. *arXiv* **2017**, arXiv:math.ST/1712.02747.
22. Blumenson, L.E. A Derivation of n-Dimensional Spherical Coordinates. *Am. Math. Mon.* **1960**, *67*, 63–66. [CrossRef]
23. Winkelbauer, A. Moments and Absolute Moments of the Normal Distribution. *arXiv* **2012**, arXiv:math.ST/1209.4340.
24. Srinivasan, G.K.; Zvengrowski, P. On the Horizontal Monotonicity of $|\Gamma(s)|$. *Can. Math. Bull.* **2011**, *54*, 538–543. [CrossRef]
25. Gauss, C.F. Disquisitiones Generales Circa Seriem Infinitam (reprint). In *Werke*; Cambridge University Press: Cambridge, UK, 2011; Volume 3.

Article

Differentiable PAC–Bayes Objectives with Partially Aggregated Neural Networks

Felix Biggs [1] and Benjamin Guedj [1,2,*]

[1] Centre for Artificial Intelligence, Department of Computer Science, University College London, London WC1V 6LJ, UK; fbiggs@cs.ucl.ac.uk
[2] Inria Lille—Nord Europe Research Centre and Inria London, 59800 Lille, France
* Correspondence: benjamin.guedj@inria.fr

Abstract: We make two related contributions motivated by the challenge of training stochastic neural networks, particularly in a PAC–Bayesian setting: (1) we show how averaging over an ensemble of stochastic neural networks enables a new class of partially-aggregated estimators, proving that these lead to unbiased lower-variance output and gradient estimators; (2) we reformulate a PAC–Bayesian bound for signed-*output* networks to derive in combination with the above a directly optimisable, differentiable objective and a generalisation guarantee, without using a surrogate loss or loosening the bound. We show empirically that this leads to competitive generalisation guarantees and compares favourably to other methods for training such networks. Finally, we note that the above leads to a simpler PAC–Bayesian training scheme for sign-*activation* networks than previous work.

Keywords: statistical learning theory; PAC–Bayes theory; deep learning

Citation: Biggs, F.; Guedj, B. Differentiable PAC–Bayes Objectives with Partially Aggregated Neural Networks. *Entropy* **2021**, *23*, 1280. https://doi.org/10.3390/e23101280

Academic Editor: Udo Von Toussaint

Received: 22 August 2021
Accepted: 27 September 2021
Published: 29 September 2021

1. Introduction

The use of stochastic neural networks has become widespread in the PAC–Bayesian and Bayesian deep learning [1] literature as a way to quantify predictive uncertainty and obtain generalisation bounds. PAC–Bayesian theorems generally bound the expected loss of *randomised* estimators, so it has proven easier to obtain non-vacuous numerical guarantees on generalisation in such networks.

However, we observe that when training these in the PAC–Bayesian setting, the objective used is generally somewhat divorced from the bound on misclassification loss itself, often because non-differentiability leads to difficulties with direct optimisation. For example, Langford and Caruana [2], Zhou et al. [3], and Dziugaite and Roy [4] all initially train non-stochastic networks before using them as the mode of a distribution, with variance chosen, respectively, through a computationally-expensive sensitivity analysis, as a proportion of weight norms, or by optimising an objective with both a surrogate loss function and a different dependence on the Kullback–Leibler (KL) divergence from their bound.

In exploring methods to circumvent this gap, we also note that PAC–Bayesian bounds can often be straightforwardly adapted to aggregates or averages of estimators, leading directly to analytic and differentiable objective functions (for example, [5]). Unfortunately, averages over deep stochastic networks are usually intractable or, if possible, very costly (as found by [6]).

Motivated by these observations, our main contribution is to obtain a compromise by defining new and general "*partially-aggregated*" Monte Carlo estimators for the average output and gradients of deep stochastic networks (Section 3), with the direct optimisation of PAC–Bayesian bounds in mind. Although our main focus here is on the use of this estimator in a PAC–Bayesian application, we emphasise that the technique applies generally to stochastic networks and thus has links to other variance-reduction techniques for training them, such as the pathwise estimator used in the context of neural networks by [7] amongst many others or Flipout [8]; indeed, it can be used in combination with these techniques. We provide proofs (Section 4) that this application leads to lower variances than a Monte Carlo forward pass and lower variance final-layer gradients than REINFORCE [9].

A further contribution of ours is a first application of this general estimator to non-differentiable "signed-output" networks (with a final output $\in \{-1, +1\}$ and arbitrarily complex other structure, see Section 4). As well as reducing variances as stated above, a small amount of additional structure in combination with partial-aggregation enables us to extend the pathwise estimator to the other layers, which usually requires a fully differentiable network and eases training by reducing the variance of gradient estimates.

We adapt a binary classification bound (Section 5) from Catoni [10] to these networks, yielding straightforward and directly differentiable objectives when used in combination with aggregation. Closing this gap between objectives and bounds leads to improved theoretical properties.

Further, since most of the existing PAC–Bayes bounds for neural networks have a heavy dependency on the distance from initialisation of the obtained solution, we would intuitively expect these lower variances to lead to faster convergence and tighter bounds (from finding low-error solutions nearer to the initialisation). We indeed observe this experimentally, showing that training PAC–Bayesian objectives in combination with partial aggregation leads to competitive experimental generalisation guarantees (Section 6), and improves upon naive Monte Carlo and REINFORCE.

As a useful corollary, this application also leads us to a similar but simpler PAC–Bayesian training method for sign-activation neural networks than Letarte et al. [6], which successfully aggregated networks with all sign activation functions $\in \{+1, -1\}$ and a non-standard tree structure, but incurred an exponential KL divergence penalty and a heavy computational cost (so that in practice they often resorted to a Monte Carlo estimate). Further, the lower variance of our obtained estimator predictions enables us to use the Gibbs estimator directly (where we draw a single sample function for every new example), leading to a modified bound on the misclassification loss which is a factor of two tighter without a significant performance penalty.

We discuss further and outline future work in Section 7.

2. Background

We begin here by setting out our notation and the requisite background.

Generally, we consider parameterised functions, $\{f_\theta : \mathcal{X} \to \mathcal{Y} | \theta \in \Theta \subset \mathbb{R}^N\}$, in a specific form, choosing $\mathcal{X} \subset \mathbb{R}^{d_0}$ and an arbitrary output space $\mathcal{Y}$ which could be for example $\{-1, +1\}$ or $\mathbb{R}$. We wish to find functions minimizing the out-of-sample risk, $R(f) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\ell(f(\boldsymbol{x}), y)$, for some loss function ℓ, for example the 0-1 misclassification loss for classification, $\ell_{0-1}(y, y') = \mathbf{1}\{y \neq y'\}$, or the binary linear loss, $\ell_{\text{lin}}(y, y') = \frac{1}{2}(1 - yy')$, with $\mathcal{Y} = \{+1, -1\}$. These must be chosen based on an i.i.d. sample $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m \sim \mathcal{D}^m$ from the data distribution $\mathcal{D}$, using the surrogate of in-sample empirical risk, $R_S(f) = \frac{1}{m}\sum_{i=1}^m \ell(f(\boldsymbol{x}_i), y_i)$. We denote the expected and empirical risks under the misclassification and linear losses, respectively R^{0-1}, R^{lin}, R_S^{0-1} and R_S^{lin}.

In this paper, we consider learning a distribution (PAC–Bayesian posterior), Q, over the parameters θ. PAC–Bayesian theorems then provide bounds on the expected generalization risk of randomised classifiers, where every prediction is made using a newly sampled function from our posterior, f_θ, $\theta \sim Q$.

We also consider averaging the above to obtain Q-aggregated prediction functions,

$$F_Q(\boldsymbol{x}) := \mathbb{E}_{\theta\sim Q} f_\theta(\boldsymbol{x}). \tag{1}$$

In the case of a convex loss function, Jensen's inequality lower bounds the risk of the randomised function by its Q-aggregate: $\ell(F_Q(x), y) \leq \mathbb{E}_{f\sim Q}\ell(f(x), y)$. The equality is achieved by the linear loss, a fact we will exploit to obtain an easier PAC–Bayesian optimisation objective in Section 5.

2.1. Analytic Q-Aggregates for Signed Linear Functions

Q-aggregate predictors are analytically tractable for "signed-output" functions (here the sign function and "signed" functions have outputs $\in \{+1, -1\}$, as the terminology "binary", used sometimes in the literature, suggests to us too strongly an output $\in \{0, 1\}$) of the form $f_w(\boldsymbol{x}) = \operatorname{sign}(\boldsymbol{w} \cdot \boldsymbol{x})$ under a normal distribution, $Q(w) = N(\boldsymbol{\mu}, \mathbb{I})$, as specifically considered in a PAC–Bayesian context for binary classification by [5], obtaining an differentiable objective similar to the SVM. Provided $\boldsymbol{x} \neq \boldsymbol{0}$:

$$F_Q(\boldsymbol{x}) := \mathbb{E}_{\boldsymbol{w} \sim N(\boldsymbol{\mu}, \mathbb{I})} \operatorname{sign}(\boldsymbol{w} \cdot \boldsymbol{x}) = \operatorname{erf}\left(\frac{\boldsymbol{\mu} \cdot \boldsymbol{x}}{\sqrt{2} \|\boldsymbol{x}\|} \right). \tag{2}$$

In Section 4, we will consider aggregating signed output ($f(x) \in \{+1, -1\}$) functions of a more general form.

2.2. Monte Carlo Estimators for More Complex Q-Aggregates

The framework of Q-aggregates can be extended to less tractable cases (for example, with f_θ a randomised or a "Bayesian" neural network, see, e.g., [1]) through a simple and unbiased Monte Carlo approximation:

$$F_Q(\boldsymbol{x}) = \mathbb{E}_{\theta \sim Q} f_\theta(\boldsymbol{x}) \approx \frac{1}{T} \sum_{t=1}^{T} f_{\theta^t}(\boldsymbol{x}) := \hat{F}_Q(\boldsymbol{x}). \tag{3}$$

If we go on to parameterize our posterior Q by $\phi \in \Phi \subset \mathbb{R}^N$ as Q_ϕ and wish to obtain gradients without a closed form for $F_{Q_\phi}(\boldsymbol{x}) = \mathbb{E}_{\theta \sim Q_\phi} f_\theta(\boldsymbol{x})$, there are two possibilities. One is REINFORCE [9], which requires only a differentiable density function, $q_\phi(\theta)$ and makes a Monte Carlo approximation to the left hand side of the identity $\nabla_\phi \mathbb{E}_{\theta \sim q_\phi} f_\theta(\boldsymbol{x}) = \mathbb{E}_{\theta \sim q_\phi}[f_\theta(\boldsymbol{x}) \nabla_\phi \log q_\phi(\theta)]$.

The other is the pathwise estimator, which additionally requires that $f_\theta(\boldsymbol{x})$ be differentiable w.r.t. θ, and that the probability distribution chosen has a standardization function, S_ϕ, which removes the ϕ dependence, turning a parameterised q_ϕ into a non-parameterised distribution p: for example, $S_{\mu,\sigma}(X) = (X - \mu)/\sigma$ to transform a general normal distribution into a standard normal. If this exists, the right hand side of $\nabla_\phi \mathbb{E}_{\theta \sim q_\phi} f_\theta(\boldsymbol{x}) = \mathbb{E}_{\epsilon \sim p} \nabla_\phi f_{S_\phi^{-1}(\epsilon)}(\boldsymbol{x})$ generally yields lower-variance estimates than REINFORCE (see for a modern survey [11]).

The variance introduced by REINFORCE can make it difficult to train neural networks when the pathwise estimator is not available, for example when non-differentiable activation functions are used. Below we find a compromise between the analytically closed form of (2) and the above estimator that enables us to make differentiable certain classes of network and extend the pathwise estimator where otherwise it could not be used. Through this we are able to stably train a new class of network.

2.3. PAC–Bayesian Approach

We use PAC–Bayes in this paper to obtain generalisation guarantees and theoretically-motivated training methods. The primary bound utilised is based on the following theorem, valid for a loss taking values in $[0, 1]$:

Theorem 1 ([10], Theorem 1.2.6)**.** *Given probability measure P on hypothesis space $\mathcal{F}$ and $\alpha > 1$, for all Q on $\mathcal{F}$ with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$,*

$$\mathbb{E}_{f \sim Q} R(f) \leq \inf_{\lambda > 1} \Phi^{-1}_{\lambda/m} \left[\mathbb{E}_{f \sim Q} R_S(f) + \frac{\alpha}{\lambda} \Delta \right]$$

with $\Phi^{-1}_\gamma(t) = \frac{1 - \exp(-\gamma t)}{1 - \exp(-\gamma)}$ *and* $\Delta = \mathrm{KL}(Q|P) - \log \delta + 2 \log\left(\frac{\log \alpha^2 \lambda}{\log \alpha} \right)$.

This slightly opaque formulation (used previously by [3]) gives essentially identical results when KL/m is large to the better-known "small-kl" PAC–Bayes bounds originated by Langford and Seeger [12], Seeger et al. [13]. It is chosen because it leads to objectives that are *linear* in the empirical loss and KL divergence, like

$$\mathbb{E}_{f\sim Q}R_S(f) + \frac{\mathrm{KL}(Q|P)}{\lambda}. \tag{4}$$

This objective is minimised by a Gibbs distribution and is closely related to the evidence lower bound (ELBO) usually optimised by Bayesian Neural Networks [1]. Such a connection has been noted throughout the PAC–Bayesian literature; we refer the reader to [14] or [15] for a formalised treatment.

3. The Partial Aggregation Estimator

Here we outline our main contribution: a reformulation of Q-aggregation for neural networks leading to different, lower-variance, Monte Carlo estimators for their outputs and gradients. These estimators apply to networks with a dense final layer, and arbitrary stochastic other structure (for example convolutions, residual layers or a non-feedforward structure). Specifically, they take the form

$$f_\theta(\boldsymbol{x}) = A(\boldsymbol{w}\cdot\boldsymbol{\eta}_{\theta^{\neg w}}(\boldsymbol{x})) \tag{5}$$

with $\theta = \mathrm{vec}(\boldsymbol{w}, \theta^{\neg w}) \in \Theta \subset \mathbb{R}^D$, $\boldsymbol{w} \in \mathbb{R}^d$, and $\theta^{\neg w} \in \Theta^{\neg w} \subset \mathbb{R}^{D-d}$ the parameter set excluding $\boldsymbol{w}$, for the non-final layers of the network. These non-final layers are included in $\boldsymbol{\eta}_{\theta^{\neg w}} : \mathcal{X} \to \mathcal{A}^d \subseteq \mathbb{R}^d$ and the final activation is $A : \mathbb{R} \to \mathcal{Y}$. For simplicity we have used a one-dimensional output but we note that the formulation and below derivations trivially extend to a vector-valued function with elementwise activations. We require the distribution over parameters to factorise like $Q(\theta) = Q^w(\boldsymbol{w})Q^{\neg w}(\theta^{\neg w})$, which is consistent with the literature.

We recover a similar functional form to that considered in Section 2.1 by rewriting the function as $A(\boldsymbol{w}\cdot\boldsymbol{a})$ with $\boldsymbol{a} \in \mathcal{A}^d$ the randomised hidden-layer activations. The "aggregated" activation function on the final layer, which we define as $I(\boldsymbol{a}) := \int A(\boldsymbol{w}\cdot\boldsymbol{a})\,\mathrm{d}Q^w(\boldsymbol{w})$, may then be analytically tractable. For example, with $\boldsymbol{w} \sim N(\boldsymbol{\mu}, \mathbb{I})$ and a sign final activation, we recall (2) where $I(\boldsymbol{a}) = \mathrm{erf}\left(\frac{\boldsymbol{\mu}\cdot\boldsymbol{a}}{\sqrt{2}\|\boldsymbol{a}\|}\right)$.

Using these definitions we can write the Q-aggregate in terms of the conditional distribution on the activations, $\boldsymbol{a}$, which takes the form $\tilde{Q}^{\neg w}(\boldsymbol{a}|\boldsymbol{x}) := (\boldsymbol{\eta}_{(\cdot)}(\boldsymbol{x})) \circ Q^{\neg w}$, (i.e., the distribution of $\boldsymbol{\eta}_{\theta^{\neg w}}(\boldsymbol{x})|\boldsymbol{x}$, with $\theta^{\neg w} \sim Q^{\neg w}$). The Q-aggregate can then be stated as

$$\begin{aligned}
F_Q(\boldsymbol{x}) &:= \mathbb{E}_{\theta\sim Q}[f_\theta(\boldsymbol{x})] \\
&= \int_{\theta^{\neg w}}\left[\int_{\mathbb{R}^d} A(\boldsymbol{w}\cdot\boldsymbol{\eta}_{\theta^{\neg w}}(\boldsymbol{x}))\,\mathrm{d}Q^w(\boldsymbol{w})\right]\mathrm{d}Q^{\neg w}(\theta^{\neg w}) \\
&= \int_{\theta^{\neg w}} I(\boldsymbol{\eta}_{\theta^{\neg w}}(\boldsymbol{x}))\,\mathrm{d}Q^{\neg w}(\theta^{\neg w}) \\
&= \int_{\mathcal{A}^d} I(\boldsymbol{a})\,\mathrm{d}\{(\boldsymbol{\eta}_{(\cdot)}(\boldsymbol{x}))\circ Q^{\neg w}\}(\boldsymbol{a}) \\
&=: \int_{\mathcal{A}^d} I(\boldsymbol{a})\,\mathrm{d}\tilde{Q}^{\neg w}(\boldsymbol{a}|\boldsymbol{x}).
\end{aligned}$$

In most cases, the final integral cannot be calculated exactly or involves a large summation, so we resort to a Monte Carlo estimate, for each $\boldsymbol{x}$ drawing T samples of the randomised activations, $\{\boldsymbol{a}^t\}_{t=1}^T \sim \tilde{Q}^{\neg w}(\boldsymbol{a}|\boldsymbol{x})$ to obtain the "partially-aggregated" estimator

$$F_Q(\boldsymbol{x}) = \int_{\mathcal{A}^d} I(\boldsymbol{a})\,\mathrm{d}\tilde{Q}^{\neg w}(\boldsymbol{a}|\boldsymbol{x}) \approx \frac{1}{T}\sum_{t=1}^{T} I(\boldsymbol{a}^t) = \hat{F}_Q^*(\boldsymbol{x}). \tag{6}$$

This is quite similar to the original estimator from (3), but in fact the aggregation of the final layer may significantly reduce the variance of the outputs and also make better gradient estimates possible, as we will show below.

3.1. Reduced Variance Estimates

Proposition 1. Lower variance outputs: *For a neural network as defined by Equation* (5) *and the unbiased Q-aggregation estimators defined by Equations* (3) *and* (6)*,*

$$\mathbb{V}_Q[\hat{F}^*_Q(\boldsymbol{x})] \leq \mathbb{V}_Q[\hat{F}_Q(\boldsymbol{x})].$$

Proof. Treating $\boldsymbol{a}$ as a random variable, always conditioned on $\boldsymbol{x}$, we have

$$\begin{aligned}&\mathbb{V}_Q[\hat{F}_Q(\boldsymbol{x})] - \mathbb{V}_Q[\hat{F}^\star_Q(\boldsymbol{x})] = \mathbb{E}_Q|\hat{F}_Q(\boldsymbol{x})|^2 - \mathbb{E}_Q|\hat{F}^\star_Q(\boldsymbol{x})|^2\\ &= \frac{1}{T}\mathbb{E}_{\boldsymbol{a}|\boldsymbol{x}}\left[\mathbb{E}_{\boldsymbol{w}}|A(\boldsymbol{w}\cdot\boldsymbol{a})|^2 - |\mathbb{E}_{\boldsymbol{w}}A(\boldsymbol{w}\cdot\boldsymbol{a})|^2\right]\\ &= \frac{1}{T}\mathbb{E}_{\boldsymbol{a}|\boldsymbol{x}}[\mathbb{V}_{\boldsymbol{w}}[A(\boldsymbol{w}\cdot\boldsymbol{a})]] \geq 0.\end{aligned}$$

□

From the above we see that the aggregate outputs estimated through partial-aggregation have lower variances. Next, we consider the two unbiased gradient estimators for the distribution over final-layer weights, $\boldsymbol{w}$, arising from partial-aggregation or REINFORCE (as would be used, for example, where the final layer is non-differentiable). Assuming $Q^{\boldsymbol{w}}$ has a density, $q_\phi(\theta^{\boldsymbol{w}})$, parameterised by ϕ, these use forward samples of $\{\boldsymbol{w}^t, \theta^{\neg\boldsymbol{w},(t)}\}_{t=1}^T$ as:

$$\begin{aligned}\hat{\boldsymbol{G}}(\boldsymbol{x}) &:= \frac{1}{T}\sum_{t=1}^T A(\boldsymbol{w}^t\cdot\boldsymbol{\eta}^t)\nabla_\phi \log q_\phi(\boldsymbol{w}^t)\\ \hat{\boldsymbol{G}}^*(\boldsymbol{x}) &:= \frac{1}{T}\sum_{t=1}^T \nabla_\phi I_{q_\phi}(\boldsymbol{\eta}^t).\end{aligned}$$

Proposition 2. Lower variance gradients: *Under the conditions of Proposition 1 and the above definitions,*

$$\mathrm{Cov}_Q[\hat{\boldsymbol{G}}^*(\boldsymbol{x})] \preceq \mathrm{Cov}_Q[\hat{\boldsymbol{G}}(\boldsymbol{x})]$$

where $A \preceq B \iff B - A$ *is positive semi-definite. Thus, for all* $\boldsymbol{u} \neq \boldsymbol{0}$, $\mathbb{V}[\hat{\boldsymbol{G}}^*(\boldsymbol{x})\cdot\boldsymbol{u}] \leq \mathbb{V}[\hat{\boldsymbol{G}}(\boldsymbol{x})\cdot\boldsymbol{u}]$.

Proof. Writing $\boldsymbol{v} := \nabla_\phi \log q_\phi(\boldsymbol{w})$ and using the unbiasedness of the estimators,

$$\begin{aligned}&\mathrm{Cov}_Q[\hat{\boldsymbol{G}}_Q(\boldsymbol{x})] - \mathrm{Cov}_Q[\hat{\boldsymbol{G}}^\star_Q(\boldsymbol{x})]\\ &= \mathbb{E}_Q[\hat{\boldsymbol{G}}_Q(\boldsymbol{x})\hat{\boldsymbol{G}}_Q(\boldsymbol{x})^T] - \mathbb{E}_Q[\hat{\boldsymbol{G}}^\star_Q(\boldsymbol{x})\hat{\boldsymbol{G}}^\star_Q(\boldsymbol{x})^T]\\ &= \frac{1}{T}\mathbb{E}_{\boldsymbol{a}|\boldsymbol{x}}\left[\mathbb{E}_{\boldsymbol{w}}[A(\boldsymbol{w}\cdot\boldsymbol{a})^2\boldsymbol{v}\boldsymbol{v}^T] - \nabla_\phi I_{q_\phi}(\boldsymbol{\eta}^t)\left(\nabla_\phi I_{q_\phi}(\boldsymbol{\eta}^t)^T\right)\right]\\ &= \frac{1}{T}\mathbb{E}_{\boldsymbol{a}|\boldsymbol{x}}[\mathrm{Cov}_{\boldsymbol{w}}[A(\boldsymbol{w}\cdot\boldsymbol{a})\nabla_\phi \log q_\phi(\boldsymbol{w})]] \succeq 0\end{aligned}$$

where in the final line we have used that $\nabla_\phi I_{q_\phi}(\boldsymbol{\eta}^t) = \nabla_\phi \mathbb{E}_{\boldsymbol{w}}[A(\boldsymbol{w}\cdot\boldsymbol{\eta}^t)] = \mathbb{E}_{\boldsymbol{w}}[A(\boldsymbol{w}\cdot\boldsymbol{\eta}^t)\boldsymbol{v}]$. □

3.2. Single Hidden Layer

For clarity (and to introduce notation to be used in Section 4.2) we will briefly consider the case of a neural network with one hidden layer, $f_\theta(\boldsymbol{x}) = A_2(\boldsymbol{w}_2\cdot\boldsymbol{A}_1(W_1\boldsymbol{x}))$. The randomised parameters are $\theta = \mathrm{vec}(\boldsymbol{w}_2, W_1)$, $W_1 \in \mathbb{R}^{d_1\times d_0}$, $\boldsymbol{w}_2 \in \mathbb{R}^{d_1}$ and the elementwise activations are $\boldsymbol{A}_1 : \mathbb{R}^{d_1} \to \mathcal{A}_1^{d_1} \subseteq \mathbb{R}^{d_1}$ and $A_2 : \mathbb{R} \to \mathcal{Y}$. We choose the distribution $Q(\theta) =: Q_2(\boldsymbol{w}_2)Q_1(W_1)$ to factorise over the layers. This is identical to the above and sets $\boldsymbol{\eta}_{W_1}(\boldsymbol{x}) = \boldsymbol{A}_1(W_1\boldsymbol{x})$.

Sampling $\boldsymbol{a}$ is straightforward if sampling W_1 is. Further, if the final layer aggregate is differentiable, and so is the hidden layer activation $\boldsymbol{A}_1$, we may be able to use the lower-variance pathwise gradient estimator for gradients with respect to Q_1. We note that this may be possible even if the activation A_2 is not differentiable, as in Section 4, where we extend the pathwise estimator where we could not otherwise use it.

Computationally, we may implement the above by analytically finding the distribution on the "pre-activations" $W_1\boldsymbol{x}$ (trivial for the normal distribution) before sampling this and passing through the activation. With the pathwise estimator this is known as the "local reparameterization trick" [16], which can lead to considerable computational savings on parallel minibatches compared to direct hierarchical sampling, $\boldsymbol{a} = A_1(W_1\boldsymbol{x})$ with $W_1 \sim Q_1$. We will utilise this in all our reparameterizable dense networks, and a variation on it to save computation when using REINFORCE in Sections 4.2 and 6.

4. Aggregating Signed-Output Networks

Here we consider a first practical application of the aggregation estimator to stochastic neural networks with a final dense sign-activated layer. We have seen above that this partial aggregation leads to better-behaved training objectives and lower-variance gradient estimates across arbitrary other network structure, It may also allow use of pathwise gradients for the other layers, which would not be possible otherwise due to the non-differentiability of the final layer.

Specifically, these networks take the form of Equation (5) with the final layer activation a sign function and weights drawn from a unit variance normal distribution, $Q^w(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}, \mathbb{I})$. The aggregate $I(\boldsymbol{a})$ is given by Equation (2). Normally-distributed weights are chosen because of the simple analytic forms for the aggregate (reminiscent of the tanh activation occasionally used for neural networks) and KL divergence (effectively an L^2 regularisation penalty); we note however that closed forms are available for other commonly-used distributions such as the Laplace.

Using Equations (3) and (6) with independent samples $\{(\boldsymbol{w}^t, \theta^{\neg \boldsymbol{w},(t)})\}_{t=1}^T \sim Q$ and $\boldsymbol{\eta}^t := \boldsymbol{\eta}_{\theta^{\neg \boldsymbol{w},(t)}}(\boldsymbol{x})$ leads to the two unbiased estimators for the output (henceforth assuming the technical condition $\mathbb{P}_{\boldsymbol{\eta}|\boldsymbol{x}}\{\boldsymbol{\eta} = \boldsymbol{0}\} = 0$ that allows aggregation to be well-defined).

$$\hat{F}_Q(\boldsymbol{x}) := \frac{1}{T}\sum_{t=1}^{T} \operatorname{sign}(\boldsymbol{w}^t \cdot \boldsymbol{\eta}^t) \tag{7}$$

$$\hat{F}_Q^*(\boldsymbol{x}) := \frac{1}{T}\sum_{t=1}^{T} \operatorname{erf}\left(\frac{\boldsymbol{\mu} \cdot \boldsymbol{\eta}^t}{\sqrt{2}\|\boldsymbol{\eta}^t\|}\right). \tag{8}$$

It follows immediately from Propositions 1 and 2 that the latter and the associated gradient estimators have lower variances than the former or the REINFORCE gradient estimates (which we would otherwise have to use due to the non-differentiability of the final layer).

4.1. Lower Variance Estimates of Aggregated Sign-Output Networks

We clarify the situation with the lower variance estimates further below. In particular, we find that the reduction in variance from using the partial-aggregation estimator is controlled by the norm $\|\boldsymbol{\mu}\|$, so that for small $\|\boldsymbol{\mu}\|$ (as could be expected early in training) the difference can be large, while as $\|\boldsymbol{\mu}\|$ grows, the difference in variance is controlled and we could reasonably revert to the Monte Carlo (or Gibbs) estimator. Note also that as $F_Q(\boldsymbol{x}) \to \pm 1$ (as expected after training), both variances disappear.

We also show that a stricter condition than Proposition 2 holds on the variances of the aggregated gradients here, and thus that the non-aggregated gradients are noisier in all cases than the aggregate.

Proposition 3. *With the definitions given by Equation ([7]), for all $\boldsymbol{x} \in \mathbb{R}^{d_0}$, $T \in \mathbb{N}$, and Q with normally-distributed final layer,*

$$0 \leq \mathbb{V}_Q[\hat{F}_Q(\boldsymbol{x})] - \mathbb{V}_Q[\hat{F}_Q^*(\boldsymbol{x})] \leq \frac{1}{T}\left(1 - \left|\operatorname{erf}\left(\frac{\|\boldsymbol{\mu}\|}{\sqrt{2}}\right)\right|^2\right).$$

Proof. The left identity follows directly from Proposition [1]. We also have

$$\begin{aligned}
\mathbb{V}_Q[\hat{F}_Q(\boldsymbol{x})] - \mathbb{V}_Q[\hat{F}_Q^\star(\boldsymbol{x})] &= \frac{1}{T}\mathbb{E}_{\boldsymbol{a}|\boldsymbol{x}}[\mathbb{V}_{\boldsymbol{w}}[\operatorname{sign}(\boldsymbol{w}\cdot\boldsymbol{a})]] \\
&= \frac{1}{T}\left(1 - \mathbb{E}_{\boldsymbol{a}|\boldsymbol{x}}\left|\operatorname{erf}\left(\frac{\boldsymbol{\mu}\cdot\boldsymbol{\eta}}{\sqrt{2}\|\boldsymbol{\eta}\|}\right)\right|^2\right) \\
&\leq \frac{1}{T}\left(1 - \left|\operatorname{erf}\left(\frac{\|\boldsymbol{\mu}\|}{\sqrt{2}}\right)\right|^2\right).
\end{aligned}$$

□

Proposition 4. *Under the conditions of Proposition [3],*

$$\operatorname{Cov}[\hat{\boldsymbol{G}}^*(\boldsymbol{x})] \preceq \operatorname{Cov}[\hat{\boldsymbol{G}}(\boldsymbol{x})] + \frac{1-2/\pi}{T}\mathbb{I}.$$

Thus, for all $\boldsymbol{u}$ with $\|\boldsymbol{u}\| = 1$,

$$\mathbb{V}[\hat{\boldsymbol{G}}^*(\boldsymbol{x})\cdot\boldsymbol{u}] \leq \mathbb{V}[\hat{\boldsymbol{G}}(\boldsymbol{x})\cdot\boldsymbol{u}] + \frac{1-2/\pi}{T}.$$

Proof. It is straightforward to show that

$$\begin{aligned}
\hat{\boldsymbol{G}}(\boldsymbol{x}) &:= \frac{1}{T}\sum_{t=1}^{T}\operatorname{sign}(\boldsymbol{w}^t\cdot\boldsymbol{\eta}^t)(\boldsymbol{\mu}-\boldsymbol{w}^t) \\
\hat{\boldsymbol{G}}^*(\boldsymbol{x}) &:= \frac{1}{T}\sum_{t=1}^{T}\frac{\boldsymbol{\eta}^t}{\|\boldsymbol{\eta}^t\|}\sqrt{\frac{2}{\pi}}\exp\left[-\frac{1}{2}\left(\frac{\boldsymbol{\mu}\cdot\boldsymbol{\eta}^t}{\|\boldsymbol{\eta}^t\|}\right)^2\right] \\
\operatorname{Cov}[\hat{\boldsymbol{G}}(\boldsymbol{x})] &= \frac{1}{T}\left(\mathbb{I} - \boldsymbol{G}\boldsymbol{G}^T\right) \\
\operatorname{Cov}[\hat{\boldsymbol{G}}^*(\boldsymbol{x})] &= \frac{1}{T}\left(\mathbb{E}\left[\frac{\boldsymbol{\eta}\boldsymbol{\eta}^T}{\|\boldsymbol{\eta}\|^2}\frac{2}{\pi}e^{-\left(\frac{\boldsymbol{\mu}\cdot\boldsymbol{\eta}}{\|\boldsymbol{\eta}\|}\right)^2}\right] - \boldsymbol{G}\boldsymbol{G}^T\right)
\end{aligned}$$

so for $\boldsymbol{u} \neq \boldsymbol{0}$,

$$\begin{aligned}
&T\boldsymbol{u}^T\left(\operatorname{Cov}[\hat{\boldsymbol{G}}(\boldsymbol{x})] - \operatorname{Cov}[\hat{\boldsymbol{G}}^*(\boldsymbol{x})]\right)\boldsymbol{u} \\
&= \|\boldsymbol{u}\|^2 - \frac{2}{\pi}\mathbb{E}\left[\frac{|\boldsymbol{u}\cdot\boldsymbol{\eta}|^2}{\|\boldsymbol{\eta}\|^2}e^{-\left(\frac{\boldsymbol{\mu}\cdot\boldsymbol{\eta}}{\|\boldsymbol{\eta}\|}\right)^2}\right] \geq \|\boldsymbol{u}\|^2\left(1 - \frac{2}{\pi}\right) > 0.
\end{aligned}$$

Above we have brought $\boldsymbol{u}$ inside the term with an expectation, which is then bounded using Cauchy–Schwarz on $|\boldsymbol{u}\cdot\boldsymbol{\eta}|/\|\boldsymbol{\eta}\| \leq \|u\|$, and $e^{-|t|} \leq 1$ for all $t \in \mathbb{R}$. □

4.2. All Sign Activations

Here we examine an important special case previously examined by Letarte et al. [6]: a feed-forward network with all sign activations and normal weights. This takes the form

$$f_\theta(\boldsymbol{x}) = \operatorname{sign}(\boldsymbol{w}_L\cdot\operatorname{sign}(W_{L-1}\dots\operatorname{sign}(W_1\boldsymbol{x})\dots))$$

with $\theta := \mathrm{vec}(\boldsymbol{w}_L, \dots, W_1)$ and $W_l := [\boldsymbol{w}_{l,1} \dots \boldsymbol{w}_{l,d_l}]^T$; $l \in \{1, ..., L\}$ are the layer indices. We choose unit-variance normal distributions on the weights, which factorise into $Q_l(W_l) = \prod_{i=1}^{d_l} q_{l,i}(\boldsymbol{w}_{l,i})$ with $q_{l,i} = \mathcal{N}(\boldsymbol{\mu}_{l,i}, \mathbb{I}_{d_{l-1}})$. In the notation of Section 3, $\boldsymbol{\eta}_{\theta\neg w}(\boldsymbol{x}) = \mathrm{sign}(W_{L-1} \dots \mathrm{sign}(W_1\boldsymbol{x}) \dots)$ is the final layer activation, which could easily be sampled by mapping $\boldsymbol{x}$ through the first $L-1$ layers with draws from the weight distribution.

Instead, we go on to make an iterative replacement of the weight distributions on each layer by conditionals on the layer activations to obtain the summation

$$\begin{aligned} F_Q(\boldsymbol{x}) = \sum_{\boldsymbol{a}_1 \in \{+1,-1\}^{d_1}} \tilde{Q}_1(\boldsymbol{a}_1|\boldsymbol{x}) \quad \times \quad \dots \\ \dots \quad \times \sum_{\boldsymbol{a}_{L-1} \in \{+1,-1\}^{d_{L-1}}} \tilde{Q}_{L-1}(\boldsymbol{a}_{L-1}|\boldsymbol{a}_{L-2}) \; \mathrm{erf}\left(\frac{\boldsymbol{\mu}_L \cdot \boldsymbol{a}_{L-1}}{\sqrt{2}\|\boldsymbol{a}_{L-1}\|}\right). \end{aligned} \tag{9}$$

The number of terms is exponential in the depth so we instead hierarchically sample the $\boldsymbol{a}_l$. Like local reparameterisation, this leads to a considerable computational saving over sampling a separate weight matrix for every input. The conditionals can be found in closed form: we can factorise individual hidden units $\tilde{Q}_l(\boldsymbol{a}_l|\boldsymbol{a}_{l-1}) := \prod_{i=1}^{d_l} \tilde{q}_{l,i}(a_{l,i}|\boldsymbol{a}_{l-1})$, and find their activation distributions (with $\boldsymbol{a}_0 := \boldsymbol{x}$ and z a dummy variable):

$$\begin{aligned} \tilde{q}_{l,i}(a_{l,i} = \pm 1 \,|\boldsymbol{a}_{l-1}) &= \int_0^\infty \mathcal{N}(z; \pm\boldsymbol{\mu}_{l,i} \cdot \boldsymbol{a}_{l-1}, \|\boldsymbol{a}_{l-1}\|^2) \, \mathrm{d}z \\ &= \frac{1}{2}\left[1 \pm \mathrm{erf}\left(\frac{\boldsymbol{\mu}_{l,i} \cdot \boldsymbol{a}_{l-1}}{\sqrt{2}\|\boldsymbol{a}_{l-1}\|}\right)\right]. \end{aligned}$$

A marginalised REINFORCE-style gradient estimator for *conditional* distributions can then be used; this does not necessarily have better statistical properties but in combination with the above is much more computationally efficient. This idea of "conditional sampling" is inspired by the local reinforce trick. Using samples $\{(\boldsymbol{a}_1^t \dots \boldsymbol{a}_{L-1}^t)\}_{t=1}^T \sim \tilde{Q}$,

$$\frac{\partial F_Q(\boldsymbol{x})}{\partial \boldsymbol{\mu}_{l,i}} \approx \frac{1}{T}\sum_{t=1}^{T} \mathrm{erf}\left(\frac{\boldsymbol{\mu}_L \cdot \boldsymbol{a}_{L-1}^t}{\sqrt{2}\|\boldsymbol{a}_{L-1}^t\|}\right) \frac{\partial}{\partial \boldsymbol{\mu}_{l,i}} \log \tilde{q}_{l,i}(a_{l,i}^t|\boldsymbol{a}_{l-1}^t). \tag{10}$$

This formulation along with Equation (9) resembles the PBGNet model of [6], but derived in a very different way. Indeed both are equivalent in the single-hidden-layer case, but with more layers PBGNet uses an unusual tree-structured network to make the individual activations independent and avoid an exponential computational dependency on the depth in Equation (9). This makes the above summation exactly calculable but is also still not efficient enough in practice, so they resort further to a Monte Carlo approximation: informally, this draws new samples for every layer l based on an average of those from the previous layer, $\boldsymbol{a}_l|\{\boldsymbol{a}_{l-1}^{(t)}\}_{t=1}^T \sim \frac{1}{T}\sum_{t=1}^T \tilde{Q}(\boldsymbol{a}_l|\boldsymbol{a}_{l-1}^{(t)})$.

This is all justified within the tree-structured framework but leads to an exponential KL penalty which—as hinted by Letarte et al. [6] and shown empirically in Section 6—makes PAC–Bayes bound optimisation strongly favour shallower such networks. Our formulation avoids this, is more general—applying to alternative network structures—and we believe it is significantly easier to understand and use in practice.

5. PAC–Bayesian Objectives with Signed-Outputs

We now move to obtain binary classifiers with guarantees for the expected misclassification error, R^{0-1}, which we do by optimizing PAC–Bayesian bounds. Such bounds (as in Theorem 1) will usually involve the non-differentiable and non-convex misclassification loss ℓ_{0-1}. However, to train a neural network we need to replace this by a differentiable surrogate, as discussed in the introduction.

Here we adopt a different approach by using our signed-output networks, where since $f(\boldsymbol{x}) \in \{+1, -1\}$, there is an exact equivalence between the linear and misclassification losses, $\ell_{0-1}(f(x), y) = \ell_{\text{lin}}(f(x), y)$, avoiding an extra factor of two from the inequality $\ell_{0-1} \leq 2\ell_{\text{lin}}$.

Although we have only moved the non-differentiability into f, the form of a PAC–Bayesian bound and the linearity of the loss and expectation allow us to go further and aggregate,

$$\mathbb{E}_{f\sim Q}\ell_{0-1}(f(\boldsymbol{x}), y) = \mathbb{E}_{f\sim Q}\ell_{\text{lin}}(f(\boldsymbol{x}), y) = \ell_{\text{lin}}(F_Q(\boldsymbol{x}), y) \tag{11}$$

which allows us to use the tools discussed in Section 4 to obtain lower-variance estimates and gradients. Below we prove a small result to show the utility of this:

Proposition 5. *Under the conditions of Proposition 3 and $y \in \{+1, -1\}$,*

$$\begin{aligned}\mathbb{V}_Q[\ell_{\text{lin}}(\hat{F}^*_Q(\boldsymbol{x}), y)] &\leq \mathbb{V}_Q[\ell_{\text{lin}}(\hat{F}_Q(\boldsymbol{x}), y)] \\ &\leq \mathbb{V}_{f\sim Q}[\ell_{0-1}(f(\boldsymbol{x}), y)] = \frac{1}{4}(1 - |F_Q(\boldsymbol{x})|^2).\end{aligned}$$

Proof.

$$\mathbb{V}_Q[\ell_{\text{lin}}(\hat{F}_Q(\boldsymbol{x}), y)] = \mathbb{E}_Q\left|\frac{1}{2}(yF_Q(\boldsymbol{x}) - y\hat{F}_Q(\boldsymbol{x}))\right|^2 = \frac{1}{4}\mathbb{V}_Q[\hat{F}_Q(\boldsymbol{x})]$$

and a similar result for $\hat{F}^*_Q$. $f = \hat{F}_Q$ if $T = 1$ and $\ell_{\text{lin}}(f(\boldsymbol{x}), y) = \ell_{0-1}(f(\boldsymbol{x}), y)$. The result then follows from this and Proposition 3. □

Combining (11) with Theorem 1, we obtain a directly optimizable, differentiable bound on the misclassification loss without introducing the above-mentioned factor of 2.

Theorem 2. *Given P on θ and $\alpha > 1$, for all Q on θ and $\lambda > 1$ simultaneously with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$,*

$$\mathbb{E}_{\theta\sim Q}R^{0-1}(f_\theta) \leq \Phi^{-1}_{\lambda/m}\left[R^{\text{lin}}_S(F_Q) + \frac{\alpha}{\lambda}\Delta\right]$$

with $\Phi^{-1}_\gamma(t) = \frac{1-\exp(-\gamma t)}{1-\exp(-\gamma)}$, $f_\theta : \mathbb{R}^d \to \{+1, -1\}$, $\theta \in \Theta$, and $\Delta = \text{KL}(Q|P) - \log\delta + 2\log\left(\frac{\log\alpha^2\lambda}{\log\alpha}\right)$.

Thus, for each λ, which can be held fixed ("**fix-λ**") or simultaneously optimized throughout training for automatic regularisation tuning ("**optim-λ**"), we obtain a gradient descent objective:

$$R^{\text{lin}}_S(\hat{F}^*_Q) + \frac{\text{KL}(Q|P)}{\lambda}. \tag{12}$$

6. Experiments

All experiments (Table 1) run on "binary"-MNIST, dividing MNIST into two classes, of digits 0–4 and 5–9. Neural networks had three hidden layers with 100 units per layer and **sign**, sigmoid (**sgmd**) or **relu** activations, before a single-unit final layer with sign activation. Q was chosen as an isotropic, unit-variance normal distribution with initial means drawn from a truncated normal distribution of variance 0.05. The data-free prior P was fixed equal to the initial Q, as motivated by Dziugaite and Roy [4] (Section 5 and Appendix B).

Table 1. Average (from ten runs) binary-MNIST losses and bounds ($\delta = 0.05$) for the best epoch and optimal hyperparameter settings of various algorithms. Hyperparameters and epochs were chosen by bound if available and non-vacuous, otherwise by training linear loss. Bold numbers indicate the best values and standard deviation is reported in italics.

	mlp	**pbg**	**Reinforce**		**Fix-λ**			**Optim-λ**		
			sign	**relu**	**sign**	**sgmd**	**relu**	**sign**	**sgmd**	**relu**
Train Linear	**0.78**	8.72	26.0	18.6	8.77	7.60	6.35	6.71	6.47	5.41
error, 1σ	*0.08*	*0.08*	*0.8*	*1.4*	*0.04*	*0.19*	*0.10*	*0.11*	*0.18*	*0.16*
Test 0–1	**1.82**	5.26	25.4	17.9	8.73	7.88	6.51	6.85	6.84	5.61
error, 1σ	*0.16*	*0.18*	*1.0*	*1.5*	*0.23*	*0.30*	*0.19*	*0.27*	*0.21*	*0.20*
Bound 0–1	-	40.8	100	100	21.7	18.8	**15.5**	22.6	19.3	16.0
error, 1σ	-	*0.2*	*0.0*	*0.0*	*0.04*	*0.17*	*0.04*	*0.03*	*0.31*	*0.05*

The objectives **fix-λ** and **optim-λ** from Section 5 were used for batch-size 256 gradient descent with Adam [17] for 200 epochs. Every five epochs, the bound (for a minimising λ) was evaluated using the entire training set; the learning rate was then halved if the bound was unimproved from the previous two evaluations. The best hyperparameters were selected using the best bound achieved in these evaluations through a grid search of initial learning rates $\in \{0.1, 0.01, 0.001\}$, sample sizes $T \in \{1, 10, 50, 100\}$. Once these were selected training was repeated 10 times to obtain the values in Table 1.

λ in **optim-**λ was optimised through Theorem 2 on alternate mini-batches with SGD and a fixed learning rate of 10^{-4} (whilst still using the objective (12) to avoid effectively scaling the learning rate with respect to empirical loss by the varying λ). After preliminary experiments in **fix-**λ, we set $\lambda = m =$ 60,000, the training set size, as is common in Bayesian deep learning.

We also report the values of three baselines: **reinforce**, which uses the fix-λ objective without partial-aggregation, forcing the use of REINFORCE gradients everywhere; **mlp**, an unregularised non-stochastic relu neural network with tanh output activation; and the PBGNet model (**pbg**) from Letarte et al. [6]. For the latter, a misclassification error bound obtained through $\ell_{0-1} \leq 2\ell_{\text{lin}}$ must be used as their test predictions were made through the sign of a prediction function $\in [-1, +1]$, not $\in \{+1, -1\}$. Further, despite significant additional hyperparameter exploration, we were unable to train a three layer network through the PBGNet algorithm directly comparable to our method, likely because of the exponential KL penalty (in their Equation 17) within that framework; to enable comparison, we therefore allowed the number of hidden layers in this scenario to vary $\in \{1, 2, 3\}$. Other baseline tuning and setup was similar to the above, see the Appendix A for more details.

During evaluation **reinforce** draws a new set of weights for every test example, equivalent to the evaluation of the other models; but doing so during training, with multiple parallel samples, is prohibitively expensive. Two different approaches to straightforward, not partially-aggregated, gradient estimation for this case suggest themselves, arising from different approximations to the Q-expected loss of the minibatch, $B \subseteq S$ (with data indices $\mathcal{B}$). From the identities

$$\begin{aligned}\nabla_\phi \mathbb{E}_{\theta\sim q_\phi} R_B(f_\theta) &= \mathbb{E}_{\theta\sim q_\phi} \frac{1}{|B|} \sum_{i\in\mathcal{B}} \ell(f_\theta(\boldsymbol{x}_i), y_i) \nabla_\phi \log q_\phi(\theta) \\ &= \frac{1}{|B|} \sum_{i\in\mathcal{B}} \mathbb{E}_{\theta\sim q_\phi} \ell(f_\theta(\boldsymbol{x}_i), y_i) \nabla_\phi \log q_\phi(\theta)\end{aligned}$$

we obtain two slightly different estimators for $\nabla_\phi \mathbb{E}_{\theta\sim q_\phi} R_B(f_\theta)$:

$$\frac{1}{T|B|} \sum_{t=1}^{T} \sum_{i\in\mathcal{B}} \ell(f_{\theta^{(t,i)}}(\boldsymbol{x}_i), y_i) \nabla_\phi \log q_\phi(\theta^{(t,i)})$$

$$\frac{1}{T|B|}\sum_{i\in\mathcal{B}}\sum_{t=1}^{T}\ell(f_{\theta^t}(\boldsymbol{x}_i),y_i)\nabla_\phi \log q_\phi(\theta^t).$$

The first draws many more samples and has lower variance but is much slower computationally; even aside from the $O(|B|)$ increase in computation, there is a slowdown as the optimised BLAS matrix routines cannot be used, and the very large matrices involved may not fit in memory (see for more information [16]).

Therefore, as is standard in the Bayesian Neural Network literature with the pathwise estimator, we use the latter formulation, which has a similar computational complexity to local-reparameterisation and our marginalised REINFORCE estimator (10). We should note though that in preliminary experiments, the alternate estimator did not appear to lead to improved results. This clarifies the advantages of marginalised sampling, which can lead to lower variance with a similar computational cost.

7. Discussion

The experiments demonstrate that partial-aggregation enables training of multi-layer non-differentiable neural networks in a PAC–Bayesian context, which is not possible with REINFORCE gradients and a multiple-hidden-layer PBGNet [6]. These obtained only vacuous bounds, and our misclassification bounds also improve those of a single-hidden-layer PBGNet.

Our experiments raise a couple of questions: firstly, why is it that lower variance estimates empirically lead to tighter bounds? We speculate that the faster convergence of SGD in this case takes us to a more "local" minimum of the objective, closer to our initialisation. Since most existing PAC–Bayes bounds for neural networks have a very strong dependence on this distance from initialisation through the KL term, this leads to tighter bounds. This distance could also be reduced through other methods we consider out-of-scope, such as the data-dependent bounds employed by Dziugaite and Roy [18] and Letarte et al. [6].

A second and harder question is asking why the non-stochastic mlp model obtains a lower overall error. The bound optimisation is empirically quite conservative, but does not necessarily lead to better generalisation; understanding this gap is a key question in the theory of deep learning.

In future work we will develop significant new tools to extend partial-aggregation to multi-class classification, and to improve test prediction bounds for $\text{sign}(\hat{F}_Q(\boldsymbol{x}))$ with $T > 1$, as in PBGNet, which gave slightly improved predictive performance despite the inferior theoretical guarantees.

Author Contributions: Conceptualization, F.B. and B.G.; Formal analysis, F.B. and B.G.; Methodology, F.B. and B.G.; Project administration, B.G.; Software, F.B.; Writing—original draft, F.B.; Writing—review and editing, F.B. and B.G. Both authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by the U.S. Army Research Laboratory and the U. S. Army Research Office, and by the U.K. Ministry of Defence and the U.K. Engineering and Physical Sciences Research Council (EPSRC) under grant number EP/R013616/1. BG acknowledges partial support from the French National Agency for Research, grants ANR-18-CE40-0016-01 and ANR-18-CE23-0015-02. FB acknowledges the support from the Foundational Artificial Intelligence Centre for Doctoral Training at University College London.

Data Availability Statement: Not applicable

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Further Experimental Details

Appendix A.1. Aggregating Biases with the Sign Function

We used a bias term in our network layers, leading to a simple extension of the above formulation, omitted in the main text for conciseness:

$$\mathbb{E}_{\boldsymbol{w}\sim\mathcal{N}(\boldsymbol{\mu},\Sigma),b\sim\mathcal{N}(\beta,\sigma^2)}\ \mathrm{sign}(\boldsymbol{w}\cdot\boldsymbol{x}+b)=\mathrm{erf}\left(\frac{\boldsymbol{\mu}\cdot\boldsymbol{x}+\beta}{\sqrt{2(\boldsymbol{x}^T\Sigma\boldsymbol{x}+\sigma^2)}}\right)$$

since $\boldsymbol{w}\cdot\boldsymbol{x}+b\sim\mathcal{N}(\boldsymbol{\mu}\cdot\boldsymbol{x}+\beta,\boldsymbol{x}^T\Sigma\boldsymbol{x}+\sigma^2)$ and

$$\begin{aligned}\mathbb{E}_{z\sim\mathcal{N}(\alpha,\beta^2)}\,\mathrm{sign}\,z&=P(z\geq 0)-P(z<0)\\&=[1-\Phi(-\alpha/\beta)]-\Phi(-\alpha/\beta)\\&=2\Phi(\alpha/\beta)-1=\mathrm{erf}(\alpha/\sqrt{2}\beta).\end{aligned}$$

The bias and weight co-variances were chosen to be diagonal with a scale of 1, which leads to some simplification in the above.

Appendix A.2. Dataset Details

We used the MNIST dataset version 3.0.1, available online at http://yann.lecun.com/exdb/mnist/ (accessed on 4 June 2021), which contains 60,000 training examples and 10,000 test examples, which were used without any further split, and rescaled to lie in the range $[0,1]$. For the "binary"-MINST task, the labels $+1$ and -1 were assigned to digits in $\{5,6,7,8,9\}$ and $\{0,1,2,3,4\}$, respectively, and images were scaled into the interval $[0,1]$.

Appendix A.3. Hyperparameter Search for Baselines

The baseline comparison values offered with our experiments were optimized similarly to the above, for completeness we report everything here.

The MLP model had three hidden ReLu layers of size 100 each trained with Adam, a learning rate $\in\{0.1,0.01,0.001\}$ and a batch size of 256 for 100 epochs. Complete test and train evaluation was performed after every epoch, and in the absence of a bound, the model and epoch with lowest train linear loss was selected.

For PBGNet we choose the values of hyperparameters from within these values giving the least bound value. Note that, unlike in [6], we do not allow the hidden size to vary $\{\in 10,50,100\}$, and we use the entire MNIST training set as we do not need a validation set. While attempting to train a three hidden layer network, we also searched through the hyperparameter settings with a batch size of 64 as in the original, but after this failed, we returned to the original batch size of 256 with Adam. All experiments were performed using the code from the original paper, available at https://github.com/gletarte/dichotomize-and-generalize (accessed on 4 June 2021).

Since we were unable to train a multiple-hidden-layer network through the PBGNet algorithm, for this model only we explored different numbers of hidden layers $\in\{1,2,3\}$.

Appendix A.4. Final Hyperparameter Settings

In Table A1 we report the hyperparameter settings used for the experiments in Table 1 after exploration. To save computation, hyperparameter settings that were not learning (defined as having a whole-train-set linear loss of >0.45 after ten epochs) were terminated early. This was also done on the later evaluation runs, where in a few instances the fix-λ sigmoid network failed to train after ten epochs; to handle this we reset the network to obtain the main experimental results.

Table A1. Chosen hyperparameter settings and additional details for results in Table 1. Best hyperparameters were chosen by bound if available and non-vacuous, otherwise by best training linear loss through a grid search as described in Section 6 and Appendix A.3. Run times are rounded to nearest 5 min.

	mlp	**pbg**	**Reinforce**		**Fix-λ**			**Optim-λ**		
			sign	**relu**	**sign**	**relu**	**sgmd**	**sign**	**relu**	**sgmd**
Init. LR	0.001	0.01	0.1	0.1	0.01	0.1	0.1	0.01	0.1	0.1
Samples, T	-	100	100	100	100	50	10	100	100	10
Hid. Layers	3	1	3	3	3	3	3	3	3	3
Hid. Size	100	100	100	100	100	100	100	100	100	100
Mean KL	-	2658	15,020	13,613	2363	3571	3011	5561	3204	4000
Runtime/min	10	5	40	40	35	30	25	35	30	25

For clarity we repeat here the hyperparameter settings and search space:

- Initial Learning Rate $\in \{0.1, 0.01, 0.001\}$.
- Training Samples $\in \{1, 10, 50, 100\}$.
- Hidden Size $= 100$.
- Batch Size $= 256$.
- Fix-λ, $\lambda = m =$ 60,000.
- Number of Hidden Layers $= 3$ for all models, except PBGNet $\in \{1, 2, 3\}$.

Appendix A.5. Implementation and Runtime

Experiments were implemented using Python and the TensorFlow library [19]. Reported approximate runtimes are for execution on a NVIDIA GeForce RTX 2080 Ti GPU.

References

1. Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; Wierstra, D. Weight Uncertainty in Neural Network. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015; Volume 37, pp. 1613–1622.
2. Langford, J.; Caruana, R. (Not) Bounding the True Error. In *Advances in Neural Information Processing Systems 14*; Dietterich, T.G., Becker, S., Ghahramani, Z., Eds.; MIT Press: Cambridge, MA, USA, 2002; pp. 809–816.
3. Zhou, W.; Veitch, V.; Austern, M.; Adams, R.P.; Orbanz, P. Non-Vacuous Generalization Bounds at the ImageNet Scale: A PAC-Bayesian Compression Approach. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
4. Dziugaite, G.K.; Roy, D.M. Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data. In Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2016, Sydney, NSW, Australia, 11–15 August 2017.
5. Germain, P.; Lacasse, A.; Laviolette, F.; Marchand, M. PAC-Bayesian Learning of Linear Classifiers. In Proceedings of the 26th Annual International Conference on Machine Learning—ICML'09, Montreal, QC, Canada, 14–18 June 2009; pp. 1–8. [CrossRef]
6. Letarte, G.; Germain, P.; Guedj, B.; Laviolette, F. Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H., Beygelzimer, A., dAlché Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2019; pp. 6872–6882.
7. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. *arXiv* **2013**, arXiv:1312.6114.
8. Wen, Y.; Vicol, P.; Ba, J.; Tran, D.; Grosse, R. Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
9. Williams, R.J. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.* **1992**, *8*, 229–256. [CrossRef]
10. Catoni, O. PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning. *IMS Lect. Notes Monogr. Ser.* **2007**, *56*, 1–163. [CrossRef]
11. Mohamed, S.; Rosca, M.; Figurnov, M.; Mnih, A. Monte Carlo Gradient Estimation in Machine Learning. *arXiv* **2019**, arXiv:1906.10652.
12. Langford, J.; Seeger, M. Bounds for Averaging Classifiers. 2001. Available online: https://www.cs.cmu.edu/~jcl/papers/averaging/averaging_tech.pdf (accessed on 4 June 2021)
13. Seeger, M.; Langford, J.; Megiddo, N. An improved predictive accuracy bound for averaging classifiers. In Proceedings of the 18th International Conference on Machine Learning, Williamstown, MA, USA, 28 June–1 July 2001; pp. 290–297.

14. Germain, P.; Bach, F.; Lacoste, A.; Lacoste-Julien, S. PAC-Bayesian Theory Meets Bayesian Inference. In *Advances in Neural Information Processing Systems 29*; Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2016; pp. 1884–1892.
15. Knoblauch, J.; Jewson, J.; Damoulas, T. Generalized Variational Inference: Three Arguments for Deriving New Posteriors. *arXiv* **2019**, arXiv:1904.02063.
16. Kingma, D.P.; Salimans, T.; Welling, M. Variational Dropout and the Local Reparameterization Trick. In *Advances in Neural Information Processing Systems 28*; Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2015; pp. 2575–2583.
17. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
18. Dziugaite, G.K.; Roy, D.M. Data-dependent PAC–Bayes priors via differential privacy. In *Advances in Neural Information Processing Systems 31*; Curran Associates, Inc.: Red Hook, NY, USA, 2018; pp. 8430–8441.
19. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. *arXiv* **2015**, arXiv:1603.04467.

entropy

Article

Meta-Strategy for Learning Tuning Parameters with Guarantees

Dimitri Meunier [1] and Pierre Alquier [2,*]

[1] Istituto Italiano di Tecnologia, 16163 Genoa, Italy; dimitri.meunier.21@ucl.ac.uk
[2] RIKEN Center for Advanced Intelligence Project, Tokyo 103-0027, Japan
* Correspondence: pierrealain.alquier@riken.jp

Abstract: Online learning methods, similar to the online gradient algorithm (OGA) and exponentially weighted aggregation (EWA), often depend on tuning parameters that are difficult to set in practice. We consider an online meta-learning scenario, and we propose a meta-strategy to learn these parameters from past tasks. Our strategy is based on the minimization of a regret bound. It allows us to learn the initialization and the step size in OGA with guarantees. It also allows us to learn the prior or the learning rate in EWA. We provide a regret analysis of the strategy. It allows to identify settings where meta-learning indeed improves on learning each task in isolation.

Keywords: meta-learning; hyperparameters; priors; online learning; Bayesian inference; online optimization; gradient descent

Citation: Meunier, D.; Alquier, P. Meta-Strategy for Learning Tuning Parameters with Guarantees. *Entropy* **2021**, *23*, 1257. https://doi.org/10.3390/e23101257

Academic Editor: Gholamreza Anbarjafari

Received: 9 August 2021
Accepted: 23 September 2021
Published: 27 September 2021

1. Introduction

In many applications of modern supervised learning, such as medical imaging or robotics, a large number of tasks is available but many of them are associated with a small amount of data. With few datapoints per task, learning them in isolation would give poor results. In this paper, we consider the problem of learning from a (large) sequence of regression or classification tasks with small sample size. By exploiting their similarities we seek to design algorithms that can utilize previous experience to rapidly learn new skills or adapt to new environments.

Inspired by human ingenuity in solving new problems by leveraging prior experience, *meta-learning* is a subfield of machine learning whose goal is to automatically adapt a learning mechanism from past experiences to rapidly learn new tasks with little available data. Since it "learns the learning mechanism" it is also referred to as *learning-to-learn* [1]. It is seen as a critical problem for the future of machine learning [2]. Numerous formulations exist for meta-learning and we focus on the problem of *online meta-learning* where the tasks arrive one at a time and the goal is to efficiently transfer information from the previous tasks to the new ones such that we learn the new tasks as efficiently as possible (this has also been refered to as *lifelong learning*). Each task is in turn processed *online*. To sum up, we have a stream of tasks and for each task a stream of observations.

In order to solve online tasks, diverse well-established strategies exist: perceptron, online gradient algorithm (OGA), online mirror descent, follow-the-regularized-leader, exponentially weighted aggregation (EWA, also refered to as *generalized Bayes* etc.) We refer the reader to [3–6] for introductions to these algorithms and to so-called regret bounds, that control their generalization errors. We refer to these algorithms as the *within-task* strategies. The big challenge is to design a meta-strategy that uses past experiences to adapt a within-task strategy to perform better on the next tasks.

In this paper, we propose a new meta-learning strategy. The main idea to learn the tuning parameters is to minimize its regret bound. We provide a meta-regret analysis for our strategy. We illustrate our results in the case where the within-task strategy is the online gradient algorithm, and exponentially weighted aggregation. In the case of OGA, the tuning parameters considered are the initialization and the gradient steps. For EWA, we consider either the learning rate, or the prior. In each case, we compare the regret incurred when learning the tasks in isolation to our meta-regret bound. This allows us to identify settings where meta-learning indeed improves on learning in isolation.

1.1. Related Works

Meta-learning is similar to multitask learning [7–9] in the sense that the learner faces many tasks to solve. However, in multitask learning, the learner is given a fixed number of tasks, and can learn the connections between these tasks. In meta-learning, the learner must prepare to face future tasks that are not given yet.

Meta-learning is often referred to as learning-to-learn or lifelong learning. The authors of [10] proposed the following distinction: "learning-to-learn" for situations where the tasks are presented simultaneously, and "lifelong learning" for situations where they are presented sequentially. Following this terminology, learning-to-learn algorithms were proposed very early in the literature, with generalization guarantees [11–16].

On the other hand, in the lifelong learning scenario, until recently, algorithms were proposed without generalization guarantees [17,18]. A theoretical study was proposed by [10], but the strategies in that paper are not feasible in practice. This problem was recently improved [19–26]. In a similar context, in [27], the authors propose an efficient strategy to learn the starting point of OGA. However, an application of this strategy to learning the step size do not show any improvement over learning in isolation [28]. The closest work to this paper is [29] in which they also suggest a regret bound minimization strategy. This paper indeed provides a meta-regret bound for learning both the initialization and the gradient step. Note, however, that this paper remains specific to OGA, while our work can be potentially applied to any online learning algorithm. Indeed, we provide another example: the generalized Bayesian algorithm EWA, for which we learn the prior, or the learning rate. To learn the prior is new in the online setting, to our knowledge. It can be related to works in the batch setting [11,13,15,16], but the improvement with respect to learning in isolation is not quantified in these works.

Finally, it is important to note that we focus on the case where the number of tasks T is large, while the sample size n and algorithmic complexity of each task is moderately small. When each task is extremely complex, for example training a deep neural network on a huge dataset, our procedure (as well as those discussed above) will become too expansive. Alternative approaches were proposed, based on optimization via multi-armed bandits [30,31].

1.2. Organization of the Paper

In Section 2, we introduce the formalism of meta-learning and the notations that will be used throughout the paper. In Section 3, we introduce our meta-learning strategy, and its theoretical analysis. In Section 4, we provide the details of our method in the case of meta-learning the initialization and the step size in the online gradient algorithm. Based on our theoretical results, there are also explicit situations where meta-learning indeed improves on learning the tasks independently. This is confirmed by experiments reported in this section. In Section 5, we provide the details of our methodology when the algorithm used within tasks is a generalized Bayesian algorithm: EWA. We show how our meta-strategy can be used to tune the learning rate; we also discuss how it can be used to learn priors. The proofs of the main results are given in Section 6.

2. Notations and Preliminaries

By convention, vectors $v \in \mathbb{R}^d$ are seen as $d \times 1$ matrices (columns). Let $\|v\|$ denote the Euclidean norm of v. Let A^T denote the transposition of any $d \times k$ matrix A, and I_d the $d \times d$ identity matrix. For two real numbers a and b, let $a \vee b = \max(a, b)$ and $a \wedge b = \min(a, b)$. For $z \in \mathbb{R}$, z_+ is its positive part $z_+ = z \vee 0$. Given a finite set S, we let $\mathrm{card}(S)$ denote the cardinality of S.

The learner has to solve tasks $t = 1, \ldots, T$ sequentially. Each task t consists in n rounds $i = 1, \ldots, n$. At each round i of task t, the learner has to take a decision $\theta_{t,i}$ in a decision space $\Theta \subseteq \mathbb{R}^d$ for some $d > 0$. Then, a convex loss function $\ell_{t,i} : \Theta \to \mathbb{R}$ is revealed to the learner, who incurs the loss $\ell_{t,i}(\theta_{t,i})$. Classical examples with $\Theta \subset \mathbb{R}^d$ include regression tasks, where $\ell_{t,i}(\theta) = (y_{t,i} - x_{t,i}^T\theta)^2$ for some $x_{t,i} \in \mathbb{R}^d$ and $y_{t,i} \in \mathbb{R}$. For classification tasks, $\ell_{t,i}(\theta) = (1 - y_{t,i}x_{t,i}^T\theta)_+$ for some $x_{t,i} \in \mathbb{R}^d$, $y_{t,i} \in \{-1, +1\}$.

Throughout the paper, we will assume that the learner uses, for each task, an online decision strategy called *within-task strategy*, parametrized by a tuning parameter $\lambda \in \Lambda$ where Λ is a closed, convex subset of $\mathbb{R}^p$ for some $p > 0$. Example of such strategies include the online gradient algorithm, given by $\theta_{t,i} = \theta_{t,i-1} - \gamma\nabla\ell_{t,i}(\theta_{t,i-1})$. In this case, the tuning parameters are the initialization, or starting point, $\theta_{t,1} = \vartheta$ and the learning rate, or step size, γ. That is, $\lambda = (\vartheta, \gamma)$, so $p = d + 1$. The parameter λ is kept fixed during the whole task. It is of course possible to use the same parameter λ in *all* the tasks. However, we will be interested here in defining *meta-strategies* that will allow us to improve λ task after task, based on the information available so far. In Section 3, we will define such strategies. For now, let λ_t denote the tuning parameter used by the learner all along task t. Figure 1 provides a recap of all the notations.

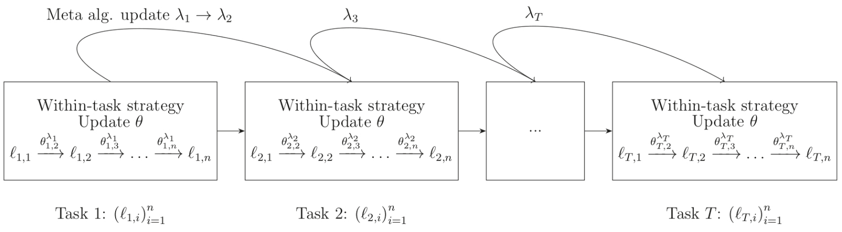

Meta alg. update $\lambda_1 \to \lambda_2$
λ_3
λ_T
Within-task strategy Update θ
$\ell_{1,1} \xrightarrow{\theta_{1,2}^{\lambda_1}} \ell_{1,2} \xrightarrow{\theta_{1,3}^{\lambda_1}} \ldots \xrightarrow{\theta_{1,n}^{\lambda_1}} \ell_{1,n}$
Within-task strategy Update θ
$\ell_{2,1} \xrightarrow{\theta_{2,2}^{\lambda_2}} \ell_{2,2} \xrightarrow{\theta_{2,3}^{\lambda_2}} \ldots \xrightarrow{\theta_{2,n}^{\lambda_2}} \ell_{2,n}$
...
Within-task strategy Update θ
$\ell_{T,1} \xrightarrow{\theta_{T,2}^{\lambda_T}} \ell_{T,2} \xrightarrow{\theta_{T,3}^{\lambda_T}} \ldots \xrightarrow{\theta_{T,n}^{\lambda_T}} \ell_{T,n}$
Task 1: $(\ell_{1,i})_{i=1}^n$
Task 2: $(\ell_{2,i})_{i=1}^n$
Task T: $(\ell_{T,i})_{i=1}^n$

Figure 1. The dynamics of meta-learning.

Let $\theta_{t,i}^\lambda$ denote the decision at round i of task t when the online strategy is used with parameter λ. We will assume that a regret bound is available for the within-task strategy. By this, we mean that there is a set $\Theta_0 \subset \Theta$ of parameters of interest, and that the learner knows a function $\mathcal{B}_n : \Theta \times \Lambda \to \mathbb{R}$ such that, for any task t, for any $\lambda \in \Lambda$,

$$\sum_{i=1}^n \ell_{t,i}(\theta_{t,i}^\lambda) \le \underbrace{\inf_{\theta\in\Theta_0}\left\{\sum_{i=1}^n \ell_{t,i}(\theta) + \mathcal{B}_n(\theta,\lambda)\right\}}_{=:\mathcal{L}_t(\lambda)}. \tag{1}$$

For OGA, regret bounds can be found, for example, in [4,6] (in this case, $\Theta_0 = \Theta$). Other examples include exponentially weighted aggregation (bounds in [3], here Θ_0 is a finite set of predictors while decisions Θ are probability distributions on Θ_0). More examples will be discussed in the paper. For a fixed parameter θ, the quantity $\sum_{i=1}^n \ell_{t,i}(\theta_{t,i}^\lambda) - \sum_{i=1}^n \ell_{t,i}(\theta)$ measures the difference between the total loss suffered during task t, and the loss what one would have suffered using the parameter θ. It is thus called "the regret with respect to parameter θ", and $\mathcal{B}_n(\theta,\lambda)$ is usually referred to as a "regret bound". We will call $\mathcal{L}_t(\lambda)$ the "meta-loss". In [29], the authors study a meta-strategy that minimizes the meta-loss of OGA. Indeed, if (1) is tight, to minimize the right-hand side is a good way to ensure that the left-hand side, that is, the cumulated loss, is small. In this work, we will focus on meta-strategies minimizing the meta-loss in a more general context.

The simplest meta-strategy is learning in isolation. That is, we keep $\lambda_t = \lambda_0 \in \Lambda$ for all tasks. The total loss after task T is then given by:

$$\sum_{t=1}^T\sum_{i=1}^n \ell_{t,i}(\theta_{t,i}^{\lambda_0}) \le \sum_{t=1}^T \mathcal{L}_t(\lambda_0). \tag{2}$$

However, when the learner uses a meta-strategy to improve the tuning parameter at the end of each task, the total loss is given by $\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\lambda_t})$. We will, in this paper, investigate strategies with meta-regret bounds; that is, bounds of the form

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\lambda_t}) \leq \inf_{\lambda\in\Lambda}\left\{\sum_{t=1}^{T}\mathcal{L}_t(\lambda)+\mathcal{C}_T(\lambda)\right\}. \tag{3}$$

Of course, such bounds will be relevant only if the right-hand side of (3) is not larger than the right-hand side of (2), and is significantly smaller in some favourable settings. We show when this is the case in Section 4.

3. Meta-Learning Algorithms

In this section, we provide two meta-strategies to update λ at the end of each task. The first one is a direct application of OGA to meta-learning. It is computationally simpler, but feasible only in the special case where we have an explicit formula for the (sub-)gradient of each $\mathcal{L}_t(\lambda)$. The second one is an application of implicit online learning to our setting. In Section 4, we provide an example where this is the case. The second meta-strategy can be used without this assumption. In both cases, we provide a regret bound as (3), under the following condition.

Assumption 1. *For any $t\in\{1,\ldots,T\}$, the function $\lambda\mapsto\mathcal{L}_t(\lambda)$ is L-Lipschitz and convex.*

3.1. Special Case: The Gradient of the Meta-Loss Is Available in Closed Form

As each $\mathcal{L}_t$ is convex, its subdifferential at each point of Λ is non-empty. For the sake of simplicity, we will use the notation $\lambda\mapsto\nabla\mathcal{L}_t(\lambda)$ in the following formulas to denote *any* element of its subdifferential at λ. We define the online gradient meta-strategy (OGMS) with step $\alpha>0$ and starting point $\lambda_1\in\Lambda$: for any $t>1$,

$$\lambda_t=\Pi_\Lambda[\lambda_{t-1}-\alpha\nabla\mathcal{L}_{t-1}(\lambda_{t-1})] \tag{4}$$

where Π_Λ denotes the orthogonal projection on Λ.

3.2. The General Case

We now cover the general case, where a formula for the gradient of $\mathcal{L}_t(\lambda)$ might not be available. We propose to apply a strategy that was first defined in [32] for online learning, and studied under the name "implicit online learning" (we refer the reader to [33] and the references therein). In the meta-learning context, this gives the online proximal meta-strategy (OPMS) with step $\alpha>0$ and starting point $\lambda_1\in\Lambda$, defined by:

$$\lambda_t=\operatorname*{argmin}_{\lambda\in\Lambda}\left\{\mathcal{L}_{t-1}(\lambda)+\frac{\|\lambda-\lambda_{t-1}\|^2}{2\alpha}\right\}. \tag{5}$$

Using classical notations, e.g., [34], we can rewrite this definition with the proximal operator (hence the name of the method). Indeed $\lambda_t=\operatorname{prox}_{\alpha\mathcal{L}_{t-1}}(\lambda_{t-1})$ where prox is the proximal operator given for any $x\in\Lambda$ and any convex function $f:\Lambda\to\mathbb{R}$,

$$\operatorname{prox}_f(x)=\operatorname*{argmin}_{\lambda\in\Lambda}\left\{f(\lambda)+\frac{\|x-\lambda\|^2}{2}\right\}. \tag{6}$$

This strategy is feasible in practice in the regime we are interested in; that is, when n is small or moderately large, and $T\to\infty$. The learner has to store all the losses of the current task $\ell_{t-1,1}, \dots, \ell_{t-1,n}$. At the end of the task, the learner can use any convex optimization algorithm to minimize, with respect to $(\theta, \lambda) \in \Theta \times \Lambda$, the function

$$F_t(\theta, \lambda) = \sum_{i=1}^{n} \ell_{t,i}(\theta) + \mathcal{B}_n(\theta, \lambda) + \frac{\|\lambda - \lambda_{t-1}\|^2}{2\alpha}. \tag{7}$$

We can use a (projected) gradient descent on F_t or its accelerated variants [35].

3.3. Regret Analysis

A direct application of known results to the setting of this paper leads to the following proposition. For the sake of completeness, we still provide the proofs in Section 6.

Proposition 1. *Under Assumption 1, using either OGMS or OPMS with step $\alpha > 0$ and starting point $\lambda_1 \in \Lambda$ leads to*

$$\sum_{t=1}^{T} \sum_{i=1}^{n} \ell_{t,i}(\theta_{t,i}^{\lambda_t}) \leq \inf_{\lambda \in \Lambda} \left\{ \sum_{t=1}^{T} \mathcal{L}_t(\lambda) + \frac{\alpha T L^2}{2} + \frac{\|\lambda - \lambda_1\|^2}{2\alpha} \right\}. \tag{8}$$

The proof can be found in Section 6.

4. Example: Learning the Tuning Parameters of Online Gradient Descent

In all this section, we work under the following condition.

Assumption 2. *For any $(t, i) \in \{1, \dots, T\} \times \{1, \dots, n\}$, the function $\ell_{t,i}$ is Γ-Lipschitz and convex.*

4.1. Explicit Meta-Regret Bound

We study the situation where the learner uses (projected) OGA as a within-task strategy; that is, $\Theta = \{\theta \in \mathbb{R}^d : \|\theta\| \leq C\}$ and, for any $i > 1$,

$$\theta_{t,i} = \Pi_{\Theta}[\theta_{t,i-1} - \gamma \nabla \ell_{t,i}(\theta_{t,i-1})]. \tag{9}$$

With such a strategy, we already mentioned that $\lambda = (\vartheta, \gamma) \in \Lambda \subset \Theta \times \mathbb{R}_+$ contains an initialization and a step size. An application of the results in Chapter 11 in [3] gives $\mathcal{B}_n(\theta, \lambda) = \mathcal{B}_n(\theta, (\vartheta, \gamma)) = \gamma \Gamma^2 n/2 + \|\theta - \vartheta\|^2/(2\gamma)$. So

$$\mathcal{L}_t((\vartheta, \gamma)) = \inf_{\|\theta\| \leq C} \left\{ \sum_{i=1}^{n} \ell_{t,i}(\theta) + \frac{\gamma \Gamma^2 n}{2} + \frac{\|\theta - \vartheta\|^2}{2\gamma} \right\}. \tag{10}$$

It is quite direct to check Assumption 1. We summarize this in the following proposition.

Proposition 2. *Under Assumption 2, assume that the learner uses OGA as an inner algorithm. Assume $\Lambda = \{\vartheta \in \mathbb{R}^d : \|\vartheta\| \leq C\} \times [\underline{\gamma}, \bar{\gamma}]$ for some $C > 0$ and $0 < \underline{\gamma} < \bar{\gamma} < \infty$. Then Assumption 1 is satisfied with*

$$L := \sqrt{\frac{n^2 \Gamma^4}{4} + \frac{4C^2}{\underline{\gamma}^2} + \frac{4C^4}{\underline{\gamma}^4}}. \tag{11}$$

So, when the learner uses one of the meta-strategies OGMS or OPMS, we can apply Proposition 1 respectively. This leads to the following theorem.

Theorem 1. *Under the assumptions of Proposition 2, with $\underline{\gamma} = 1/n^\beta$ for some $\beta > 0$ and $\bar{\gamma} = C^2$, when the learner uses either OGMS or OPMS with*

$$\alpha = \frac{C}{L}\sqrt{\frac{4+C^2}{T}} \tag{12}$$

(where L is given by (11)), we have:

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\lambda_t}) \leq \inf_{\theta_1,\ldots,\theta_T\in\Theta}\left\{\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_t) + \mathcal{C}(\Gamma,C)\left[n^{1\vee 2\beta}\sqrt{T} + \left(n^{1-\beta}+\sigma(\theta_1^T)\sqrt{n}\right)T\right]\right\} \tag{13}$$

where $\mathcal{C}(\Gamma,C) > 0$ depends only on (Γ,C) and where:

$$\sigma(\theta_1^T) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left\|\theta_t - \frac{1}{T}\sum_{s=1}^{T}\theta_s\right\|^2}. \tag{14}$$

Let us compare this result with learning in isolation, as defined in (2); that is, solving the sequence of tasks with a constant hyperparameter $\lambda = (\vartheta,\gamma)$. For the usual choice $\vartheta = 0$ and $\gamma = c/\sqrt{n}$ where c is a constant that does not depend on n nor T, OGA leads to a regret in $\mathcal{O}(\sqrt{n})$. After T tasks, learning in isolation thus leads to a regret in $T\sqrt{n}$. Our strategies with $\beta = 1$ lead to a regret in

$$n^2\sqrt{T} + \left(1+\sigma(\theta_1^T)\sqrt{n}\right)T. \tag{15}$$

The term $n^2\sqrt{T}$ is the price to pay for meta-learning. In the regime we are interested in (small n, large T), which is smaller than $T\sqrt{n}$. Consider the leading term. In the worst case scenario, this is also $T\sqrt{n}$. However, there are good predictors $\theta_1,\ldots,\theta_T$ for tasks $1,\ldots,T$, respectively, such that $\sigma(\theta_1^T)$ is small, and in this case we see the improvement with respect to learning in isolation. The extreme case is when there is a good predictor θ^* that predicts well for all tasks. In this case, regret with respect to $\theta_1 = \cdots = \theta_T = \theta^*$ is in $n^2\sqrt{T}+T$, which improves significantly on learning in isolation. Note however that, using a different meta-strategy, specifically designed for OGA, Ref. [29] obtain a better dependence on T when $\sigma(\theta_1^T) = 0$.

Let us now discuss the implementation of our meta-stategy. We first remark that under the quadratic loss, it is possible to derive a formula for $\mathcal{L}_t$, which allows to use OGMS. We then discuss OPMS for the general case.

4.2. Special Case: Quadratic Loss

First, consider $\ell_{t,i} = (y_{t,i} - x_{t,i}^T\theta)^2$ for some $y_{t,i}\in\mathbb{R}$ and $x_{t,i}\in\mathbb{R}^d$. Assumption 2 is satisfied if we assume, moreover that all $|y_{t,i}|\leq c$ and $\|x_{t,i}\|\leq b$, with $\Gamma = 2bc+2b^2C$. In this case,

$$\mathcal{L}_t((\vartheta,\gamma)) = \inf_{\|\theta\|\leq C}\left\{\sum_{i=1}^{n}(y_{t,i}-x_{t,i}^T\theta)^2 + \frac{\gamma\Gamma^2 n}{2} + \frac{\|\theta-\vartheta\|^2}{2\gamma}\right\}. \tag{16}$$

Define $Y_t = (y_{t,1},\ldots,y_{t,n})^T$ and $X_t = (x_{t,1}|\ldots|x_{t,n})^T$. The minimizer of $\sum_{i=1}^n(y_{t,i} - x_{t,i}^T\theta)^2 + \|\theta-\vartheta\|^2/(2\gamma)$ with respect to θ is known as the ridge regression estimator:

$$\hat{\theta}_t = \left(X_t^T X_t + \frac{I_d}{2\gamma}\right)^{-1}\left(X_t^T Y_t + \frac{\vartheta}{2\gamma}\right). \tag{17}$$

This also coincides with the minimizer in the right-hand side of (16) on the condition that $\|\hat{\theta}_t\|\leq C$. In this case, by plugging $\hat{\theta}_t$ in (16), we have a close form formula for $\mathcal{L}_t((\vartheta,\gamma))$, and an explicit (but cumbersome) formula for its gradient. It is thus possible to use the OGMS strategy to update $\lambda = (\vartheta,\gamma)$.

4.3. The General Case

In the general case, denote $\lambda_{t-1} = (\vartheta_{t-1}, \gamma_{t-1})$, then $\lambda_t = (\vartheta_t, \gamma_t)$ is obtained by minimizing

$$F_t(\theta, (\vartheta, \gamma)) = \sum_{i=1}^{n} \ell_{t,i}(\theta) + \frac{\gamma\Gamma^2 n}{2} + \frac{\|\theta - \vartheta\|^2}{2\gamma} + \frac{\|\vartheta - \vartheta_{t-1}\|^2 + (\gamma - \gamma_{t-1})^2}{2\alpha} \tag{18}$$

with respect to $\theta, \vartheta, \gamma$. Any efficient minimization procedure can be used. In our experiments, we used a projected gradient descent, the gradient being given by:

$$\frac{\partial F_t}{\partial \theta} = \sum_{i=1}^{n} \nabla \ell_{t,i}(\theta) + \frac{\theta - \vartheta}{\gamma}, \tag{19}$$

$$\frac{\partial F_t}{\partial \vartheta} = \frac{\vartheta - \theta}{\gamma} + \frac{\vartheta - \vartheta_{t-1}}{\alpha}, \tag{20}$$

$$\frac{\partial F_t}{\partial \gamma} = \frac{\Gamma^2 n}{2} - \frac{\|\theta - \vartheta\|^2}{2\gamma^2} + \frac{\gamma - \gamma_{t-1}}{\alpha}. \tag{21}$$

Note that even though we do not *stricto sensu* obtain the minimizer of F_t, we can get arbitrarily close to it by taking a large enough number of steps. The main difference between this algorithm and the strategy suggested in [29] is that it is obtained by applying the general proximal update introduced in Equation (7), while they decoupled the update for the initialization step and the learning rate.

4.4. Experimental Study

In this section we compare simulated data for the numerical performance of OPMS w.r.t learning the task in isolation with online gradient descent (I-OGA). To measure the impact of learning the gradient step γ, we also introduce mean-OPMS that uses the same strategy as OPMS but only learns the starting point ϑ (it is thus close to [27]). We present the results for regression tasks with the mean-squared-error loss, and then for classification with the hinge loss. The notebooks of the experiments can be found online: https://dimitri-meunier.github.io/ (accessed on 26 September 2021).

4.4.1. Synthetic Regression

At each round $t = 1, \ldots, T$, the meta learner sequentially receives a regression task that corresponds to a dataset $(x_{t,i}, y_{t,i})_{i=1,\ldots,n}$ generated as $y_{t,i} = x_{t,i}^T \theta_t + \epsilon_{t,i}$, $x_{t,i} \in \mathbb{R}^d$. The noise is $\epsilon_{t,i} \sim \mathcal{U}([-\sigma^2, \sigma^2])$ and the $\epsilon_{t,i}$ are all independent, the inputs are uniformly sampled on the $(d-1)$-unit sphere $\mathcal{S}^{d-1}$ and $\theta_t = ru + \theta_0$, $u \sim \mathcal{U}\left(\mathcal{S}^{d-1}\right)$, $\theta_0 \in \mathbb{R}^d$, $r \in \mathbb{R}_+$. We take $d = 20$, $n = 30$, $T = 200$, $\sigma^2 = 0.5$ and θ_0 with all components equal to 5. In this setting, θ_0 is a common bias between the tasks, σ^2 is the inter-task variance and r characterizes the tasks similarity. We experiment with different values of $r \in \{0, 5, 10, 30\}$ to observe the impact of task similarity on the meta-learning process. The smaller r, the closer are the tasks and for the extreme case of $r = 0$ the tasks are identical, in the sense that the parameters θ_t of the tasks are all the same. We draw attention to the fact that a cross-validation procedure to select α (the parameter of OGMS or OPMS, see Equation (5)) or γ is not valid in the online settings, as it would require having knowledge of several tasks in advance for the former and several datapoints in advance for each task for the latter. Moreover, the theoretical values are based on worst-case analysis and lead in practice to slow learning. In practice, to set these values to the correct order of magnitude without adjusting the constants led to better results. So, for mean-OPMS and OPMS we set $\alpha = 1/\sqrt{T}$, for OPMS and I-OGA we set $\gamma = 1/\sqrt{n}$. Instead of cross-validation, one can launch several online learners in parallel with different parameter values to pick the best one (or aggregate them). That is the strategy we use to select Γ for OPMS. Note that the exact value of Γ is usually unkown in practice; its automatic calibration is an important open question. To solve (18), after each task we use the exact solution for mean-OPMS and projected Newton descent with 10 steps for OPMS. We observed that not reaching the exact solution of (18) does not harm the performance of the algorithm and 10 steps are sufficient to reach convergence. The results are displayed in Table 1 and Figure 2. On Figure 2, for each task $t = 1, \dots, T$, we report the average end-of-task loss $MSE_t = \sum_{i=1}^{n} \ell_{t,i}(\theta_{t,n})/n$ averaged over 50 independent runs (with their confidence intervals). Table 1 reports MSE_t averaged over the 100 most recent tasks. The results confirm our theoretical findings: learning γ can bring a substantial benefit over just learning the starting point, which in turn brings a considerable benefit with respect to learning the tasks in isolation. Learning the gradient step makes the meta-learner more robust to task dissimilarities (i.e. when r increases) as shown in Figure 2. In the regime where r is low, learning the gradient step does not help the meta-learner as it takes more steps to reach convergence. Overall both meta learners are consistently better than learning the task in isolation since the number of observation per task is low.

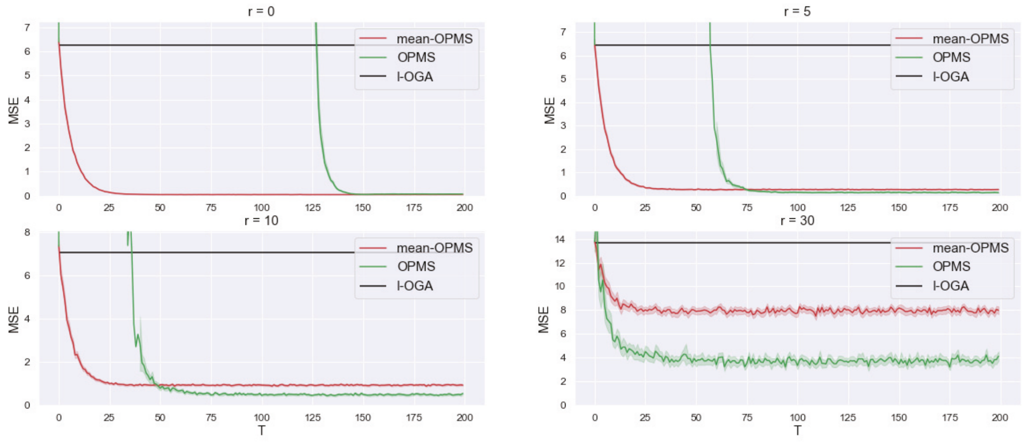


Figure 2. Performance of learning in isolation with OGA (**I-OGA**), OPMS to learn initialization (**mean-OPMS**) and OPMS to learn initialization and step size (**OPMS**). We report the average end-of-task MSE losses at the end of each task, for different values of the task-similarity index $r \in \{0, 5, 10, 30\}$. The results are averaged over 50 independent runs to get confidence intervals.

Table 1. Average end-of-task MSE of the 100 last tasks (averaged over 50 independent runs).

	r = 0	**r = 5**	**r = 10**	**r = 30**
I-OGA	6.24	6.44	7.06	13.60
mean OPMS	0.05	0.27	0.93	7.93
OPMS	0.07	0.15	0.49	3.72

4.4.2. Synthetic Classification

At each round $t = 1, \dots, T$, the meta learner sequentially receives a binary classification task with the Hinge loss that corresponds to a dataset $(x_{t,i}, y_{t,i})_{i=1,\dots,n}$. The binary labels $\{-1, 1\}$ are generated as a logistic model $\mathbb{P}(y = 1) = (1 + \exp(-x^t \theta_t))^{-1}$. The task parameters θ_t and the inputs are generated as in the regression setting. To add some noise, we shuffle 10% of the labels. We take $d = 10$, $n = 100$, $T = 500$, $r = 2$. For mean-OPMS and OPMS we set $\alpha = 1/\sqrt{T}$, for OPMS and I-OGA we set $\gamma = 1/\sqrt{n}$. For the optimisation of F_t (18) with both OPMS and mean-OPMS we use a projected gradient descent with 50 steps.

On Figure 3, for each task $t = 1, \dots, T$, we report the regret on the end-of-task losses: $R(t) = \frac{1}{nt} \sum_{k=1}^{t} \sum_{i=1}^{n} \ell_{k,i}(\theta_{k,n})$, averaged over 10 independent runs (with their confidence intervals). As the for regression setting, the results confirm our theoretical findings: by learning γ (OPMS), we reach a better overall performance than just learning the initialization (mean-OPMS) and a substantially stronger than independent task learning (I-OGA). Note that, in the classification regime, there is no known closed formed expression for the meta-gradient; therefore, OGMS cannot be used.

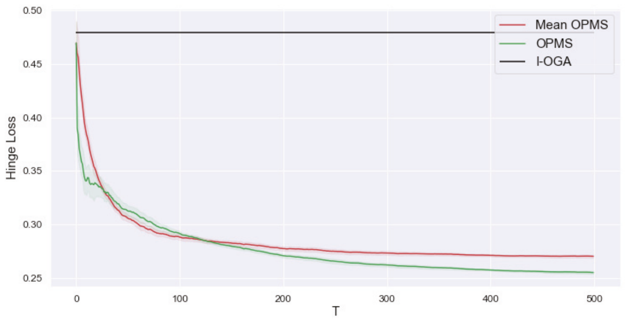


Figure 3. Performance of learning in isolation with OGA (**I-OGA**), OPMS to learn the initialization (**mean-OPMS**) and OPMS to learn the initialization and step size (**OPMS**) on a sequence of classification tasks with the Hinge loss. We report the meta-regret of the Hinge loss. The results are averaged over 10 independent runs (dataset generation) to get confidence intervals.

5. Second Example: Learning the Prior or the Learning Rate in Exponentially Weighted Aggregation

In this section, we will study a generalized Bayesian method, exponentially weighted aggregation. Consider a *finite* set $\Theta_0 = \{\theta_1, \dots, \theta_M\} \subset \mathbb{R}^d$. EWA depends on a prior distribution π on Θ_0, and on a learning rate $\eta > 0$, and returns a decision in $\Theta = \mathrm{conv}(\theta_1, \dots, \theta_M)$ the convex envelope of Θ_0. In this section, we work under the following condition.

Assumption 3. *There is a $B \in \mathbb{R}_+^*$, such that for any $(t,i) \in \{1, \dots, T\} \times \{1, \dots, n\}$, the function $\ell_{t,i}$ is $\Theta \to [0, B]$ and convex.*

We will sometimes use a stronger assumption.

Assumption 4. *There is a $C \in \mathbb{R}_+^*$, such that for any $(t,i) \in \{1, \dots, T\} \times \{1, \dots, n\}$, the function $\theta \mapsto \exp(-\ell_{t,i}(\theta)/C)$ is concave.*

Examples of a situation in which Assumption 4 is satisfied are provided in [3]. Note that Assumption 4 implies Assumption 3.

5.1. Reminder on EWA

The update in EWA is given by:

$$\theta_{t,i} = \sum_{\theta \in \Theta_0} p_{t,i}(\theta)\theta \tag{22}$$

where $p_{t,i}$ are weights defined by

$$p_{t,i}(\theta) = \frac{\exp\left[-\eta \sum_{j=1}^{i-1} \ell_{t,j}(\theta)\right] \pi(\theta)}{\sum_{\vartheta \in \Theta_0} \exp\left[-\eta \sum_{j=1}^{i-1} \ell_{t,j}(\vartheta)\right] \pi(\vartheta)}. \tag{23}$$

The strategy is studied in detail in [3]. We refer the reader to [36] and the references therein for connections to Bayesian inference. We recall the following regret bounds from [3]. First, under Assumption 3,

$$\sum_{i=1}^{n} \ell_{t,i}(\theta_{t,i}) \leq \min_{\theta\in\Theta_0}\left[\sum_{i=1}^{n} \ell_{t,i}(\theta) + \frac{\eta n B^2}{8} + \frac{\log \frac{1}{\pi(\theta)}}{\eta}\right]. \tag{24}$$

Moreover, under the stronger Assumption 4,

$$\sum_{i=1}^{n} \ell_{t,i}(\theta_{t,i}) \leq \min_{\theta\in\Theta_0}\left[\sum_{i=1}^{n} \ell_{t,i}(\theta) + C\log\frac{1}{\pi(\theta)}\right]. \tag{25}$$

In Section 5.2, we work in the general setting (Assumption 3), and we use our meta-strategy OPMS or OGMS to learn η. In Section 5.3, we use OPMS or OGMS to learn π under Assumption 4.

5.2. Learning the Rate η

Consider the uniform prior $\pi(\theta) = 1/M$ for any $\theta \in \Theta_0$. Then, the regret bound (24) becomes:

$$\sum_{i=1}^{n} \ell_{t,i}(\theta_{t,i}) \leq \min_{\theta\in\Theta_0}\sum_{i=1}^{n} \ell_{t,i}(\theta) + \frac{\eta n B^2}{8} + \frac{\log M}{\eta} \tag{26}$$

and it is then possible to optimize it explicitly with respect to η. The value minimizing the bound is $\eta = (2/B)\sqrt{2\log(M)/n}$ and the regret bound becomes:

$$\sum_{i=1}^{n} \ell_{t,i}(\theta_{t,i}) \leq \min_{\theta\in\Theta_0}\sum_{i=1}^{n} \ell_{t,i}(\theta) + B\sqrt{\frac{n\log M}{2}}. \tag{27}$$

In practice, however, while it is often reasonable to assume that the loss function is bounded (as in Assumption 3), very often one does not know a tight upper bound. Thus, one may use a constant B that satisfies Assumption 3, but that is far too large. Even though one does not know a better upper bound than B, one would like a regret bound that depends on the tightest possible upper bound.

In the meta-learning framework, define:

$$\mathcal{L}_t(\eta) = \min_{\theta\in\Theta_0}\sum_{i=1}^{n} \ell_{t,i}(\theta) + \frac{\eta n\left[\max_{\vartheta\in\Theta_0, 1\leq i\leq n} \ell_{t,i}(\vartheta)\right]^2}{8} + \frac{\log M}{\eta} \tag{28}$$

for $\eta \in \Lambda = [1/n, 1]$. It is immediately necessary to prove that this function is convex and L-Lipschitz with $L = n^2\log(M) + nB^2/8$. So, Assumption 1 is satisfied, allowing for the use of the OPMS or OGMS strategy without needed a tight upper bound on the losses. Note that, in this context, the OGMS strategy is given by:

$$\eta_t = \frac{1}{n} \vee \left[\eta_{t-1} - \alpha\left(\frac{n\left[\max_{\theta\in\Theta_0, 1\leq i\leq n} \ell_{t,i}(\theta)\right]^2}{8} - \frac{\log M}{\eta_{t-1}^2}\right)\right] \wedge 1.$$

Theorem 2. *Under Assumption 3, using OGMS or OPMS on $\mathcal{L}_t(\eta)$, as in (28) with $\eta_1 = 1$, $L = n^2\log(M) + nB^2/8$ and*

$$\alpha = \frac{1}{L}\sqrt{\frac{2}{T}} \tag{29}$$

we have

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\eta_t}) \leq \sum_{t=1}^{T}\min_{\theta\in\Theta_0}\sum_{i=1}^{n}\ell_{t,i}(\theta) + bT\sqrt{\frac{n\log(M)}{2}} + T\log(M) + \frac{b^2T}{8} + \left(n^2\log M + \frac{nB^2}{8}\right)\sqrt{2T} \quad (30)$$

where

$$b = \max_{\theta\in\Theta_0, 1\leq t\leq T, 1\leq i\leq n}|\ell_{t,i}(\theta)|. \quad (31)$$

Let us compare learning in isolation with meta-learning in this context. When learning in isolation, the hyperparameter η is fixed (as in (2)). If we fix it to the value $\eta_0 = (2/B)\sqrt{2\log(M)/n}$ as in (27), the meta-regret is in $BT\sqrt{n\log(M)/2}$. On the other hand, meta-learning leads to a meta-regret in $bT\sqrt{n\log(M)/2} + n^2\log M\sqrt{2T} + \mathcal{O}(nB^2\sqrt{T}+T)$. In other words, we replace the potentially loose upper bound B by the tightest possible bound b, at the cost of an additional $n^2\log M\sqrt{2T} + \mathcal{O}(nB^2\sqrt{T}+T)$ term. Here again, when T is large enough with respect to n, this term is negligible.

5.3. Learning the Prior π

Under Assumption 4, we have the regret bound in (25). Without any information on Θ_0, it seems natural to use the uniform prior π on $\Theta_0 = \{\theta_1,\ldots,\theta_M\}$, which leads to

$$\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}) \leq \min_{\theta\in\Theta_0}\sum_{i=1}^{n}\ell_{t,i}(\theta) + C\log M. \quad (32)$$

If some additional information was available, such as, for example: "the best θ is always either θ_1 or θ_2", one would rather chose the uniform prior on $\{\theta_1,\theta_2\}$, and obtain the bound:

$$\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}) \leq \min_{\theta\in\Theta_0}\sum_{i=1}^{n}\ell_{t,i}(\theta) + C\log 2. \quad (33)$$

Unfortunately, such information is generally not available. However, in the context of meta-learning, we can take advantage of the previous tasks to learn such information.

Thus, let us define, for any task t,

$$\theta_t^* = \operatorname*{argmin}_{\theta\in\Theta_0}\sum_{i=1}^{n}\ell_{t,i}(\theta) \quad (34)$$

and

$$\mathcal{L}_t(\pi) = \sum_{i=1}^{n}\ell_{t,i}(\theta_t^*) + C\log\frac{1}{\pi(\theta_t^*)} \quad (35)$$

for $\pi = (\pi(\theta_1),\ldots,\pi(\theta_M)) \in \Lambda$ with

$$\Lambda = \left\{x\in(\mathbb{R}_+)^M : \sum_{h=1}^{M}x_h = 1 \text{ and } x_h \geq \frac{1}{2M}\right\}. \quad (36)$$

It is important to check that $\mathcal{L}_t$ is convex and L-Lipschitz with $L = 2CM$ on Λ; this allows us to use OPMS (or OGMS).

Theorem 3. *Under Assumption 4, using OPMS on $\mathcal{L}_t(\pi)$ as in (35) with $\pi_1 = (1/M,\ldots,1/M)$, $L = 2CM$ and*

$$\alpha = \frac{1}{2CM\sqrt{T}}, \quad (37)$$

define $I^* = \{\theta_1^*, \ldots, \theta_T^*\}$ *where each* θ_t^* *is as in* (34) *and* $m^* = \mathrm{card}(I^*)$. *We have*

$$\sum_{t=1}^{T} \sum_{i=1}^{n} \ell_{t,i}(\theta_{t,i}^{\pi_t}) \leq \sum_{t=1}^{T} \sum_{i=1}^{n} \ell_{t,i}(\theta_t^*) + CT \log(2m^*) + 2CM\sqrt{T}. \tag{38}$$

When learning in isolation with a uniform prior, the meta-regret is $TC \log(M)$. On the other hand, if m^* is small (that is, many of the θ_i^*s are similar), meta-learning leads to a meta-regret in $CT \log(2m^*) + 2CM\sqrt{T}$. For a T that is large enough, this is an important improvement.

5.4. Discussion on the Continuous Case

Let us now discuss the possibility of meta-learning for generalized Bayesian methods when Θ_0 is no longer a finite set. There is a general formula for EWA, given by

$$\rho_{t,i}(\mathrm{d}\theta) = \underset{\rho}{\mathrm{argmin}} \left\{ \mathbb{E}_{\theta\sim\rho}\left[\sum_{j=1}^{i-1} \ell_{t,j}(\theta)\right] + \frac{\mathcal{K}(\rho, \pi)}{\eta} \right\} \tag{39}$$

where the minimum is taken over for all probability distributions that are absolutely continuous with π, and where π is a prior distribution, $\eta > 0$ a learning rate and $\mathcal{K}$ is the Kullback–Leibler divergence (KL). Meta-learning for such an update rule is proven in [10,37] but usually does not lead to feasible strategies. Online variational inference [38,39] consists in replacing the minimization on the set of all probability distributions by minimization in a smaller set in order to define a feasible approximation of $\rho_{t,i}$. For example, let $(q_\mu)_{\mu\in M}$ be a parametric family of probability distributions, Thus, we define:

$$\mu_{t,i} = \underset{\mu\in M}{\mathrm{argmin}} \left\{ \mathbb{E}_{\theta\sim q_\mu}\left[\sum_{j=1}^{i-1} \ell_{t,j}(\theta)\right] + \frac{\mathcal{K}(q_\mu, \pi)}{\eta} \right\}. \tag{40}$$

It is discussed in [40] that, generally, when μ is a location-scale parameter and $\ell_{t,j}$ is Γ-Lipschitz and convex, then $\bar{\ell}_{t,i}(\mu) := \mathbb{E}_{\theta\sim q_\mu}[\ell_{t,j}(\theta)]$ is 2Γ-Lipschitz and convex. In this case, under the assumption that $\mathcal{K}(q_\mu, \pi)$ is α-strongly convex in μ, a regret bound for such strategies was derived in [39]:

$$\sum_{i=1}^{n} \mathbb{E}_{\theta\sim q_{\mu_{t,i}}}[\ell_{t,i}(\theta)] \leq \inf_{\mu\in\mathcal{M}} \left\{ \mathbb{E}_{\theta\sim q_\mu}\left[\sum_{i=1}^{n} \ell_{t,i}(\theta)\right] + \frac{\eta 4\Gamma^2 n}{\alpha} + \frac{\mathcal{K}(q_\mu, \pi)}{\eta} \right\}. \tag{41}$$

A complete study of meta-learning of the rate $\eta > 0$ and of the prior π in this context is an important objective (possibly, with a restriction that $\pi \in \{q_\mu, \mu \in M\}$). However, this raises many problems. For example, the KL divergence $\mathcal{K}(q_\mu, q_{\mu'})$ is not always convex with respect to the parameter μ'. In this case, it might help to replace it by a convex relaxation that would allow for the use of OGMS or OPMS. This relates to [41,42], who advocate going beyond the KL divergence in (39); see also [36] and the references therein. This will be the object of future works.

6. Proofs

We start with a preliminary lemma that will be used in the proof of Proposition 1.

Lemma 1. *Let* a, b, c *be three vectors in* $\mathbb{R}^p$. *Then:*

$$(a-b)^T(b-c) = \frac{\|a-c\|^2 - \|a-b\|^2 - \|b-c\|^2}{2}. \tag{42}$$

Proof. expand $\|a-c\|^2 = \|a\|^2 + \|c\|^2 - 2a^T c$ in the r.h.s, as well as $\|a-b\|^2$ and $\|b-c\|^2$. Then simplify. □

We now prove Proposition 1 separately for the general OGMS strategy, and then for OGMS.

Proof of Proposition 1 for OPMS. As mentioned earlier, this strategy is an application to the meta-learning setting of implicit online learning [32,33]. We follow here a proof from Chapter 11 in [3]. We refer the reader to [43] and the references therein for tighter bounds under stronger assumptions.

First, λ_t is defined as the minimizer of a convex function in (5). So, the subdifferential of this function at λ_t contains 0. In other words, there is a $z_t \in \partial\mathcal{L}_{t-1}(\lambda_t)$, such that

$$z_t = \frac{\lambda_{t-1} - \lambda_t}{\alpha}. \tag{43}$$

By convexity, for any λ, for any $z \in \partial\mathcal{L}_{t-1}(\lambda_t)$,

$$\mathcal{L}_{t-1}(\lambda) \geq \mathcal{L}_{t-1}(\lambda_t) + (\lambda - \lambda_t)^T z. \tag{44}$$

The choice $z = z_t$ gives:

$$\mathcal{L}_{t-1}(\lambda) \geq \mathcal{L}_{t-1}(\lambda_t) + \frac{(\lambda - \lambda_t)^T(\lambda_{t-1} - \lambda_t)}{\alpha}, \tag{45}$$

that is,

$$\begin{aligned}
\mathcal{L}_{t-1}(\lambda_t) &\leq \mathcal{L}_{t-1}(\lambda) + \frac{(\lambda - \lambda_t)^T(\lambda_t - \lambda_{t-1})}{\alpha} \\
&= \mathcal{L}_{t-1}(\lambda) + \frac{\|\lambda - \lambda_{t-1}\|^2 - \|\lambda - \lambda_t\|^2}{2\alpha} - \frac{\|\lambda_t - \lambda_{t-1}\|^2}{2\alpha} \\
&= \mathcal{L}_{t-1}(\lambda) + \frac{\|\lambda - \lambda_{t-1}\|^2 - \|\lambda - \lambda_t\|^2}{2\alpha} - \alpha\frac{\|z_t\|^2}{2}
\end{aligned} \tag{46}$$

where we used Lemma 1. Then, note that

$$\begin{aligned}
\mathcal{L}_{t-1}(\lambda_{t-1}) &= \mathcal{L}_{t-1}(\lambda_t) + [\mathcal{L}_{t-1}(\lambda_{t-1}) - \mathcal{L}_{t-1}(\lambda_t)] \\
&\leq \mathcal{L}_{t-1}(\lambda_t) + \|\lambda_{t-1} - \lambda_t\| L \\
&\leq \mathcal{L}_{t-1}(\lambda_t) + \alpha\|z_t\| L.
\end{aligned} \tag{47}$$

Combining this inequality with (46) gives

$$\mathcal{L}_{t-1}(\lambda_{t-1}) \leq \mathcal{L}_{t-1}(\lambda) + \frac{\|\lambda - \lambda_{t-1}\|^2 - \|\lambda - \lambda_t\|^2}{2\alpha} + \alpha\left(\|z_t\| L - \frac{\|z_t\|^2}{2}\right). \tag{48}$$

Now, for any $x \in \mathbb{R}$, $-x^2/2 + xL - L^2/2 \leq 0$. In particular, $\|z_t\|L - \|z_t\|^2/2 \leq L^2/2$ and so the above can be rewritten:

$$\mathcal{L}_{t-1}(\lambda_{t-1}) \leq \mathcal{L}_{t-1}(\lambda) + \frac{\|\lambda - \lambda_{t-1}\|^2 - \|\lambda - \lambda_t\|^2}{2\alpha} + \frac{\alpha L^2}{2}. \tag{49}$$

Summing the inequality for $t = 2$ to $T+1$ leads to:

$$\sum_{t=1}^{T} \mathcal{L}_t(\lambda_t) \leq \sum_{t=1}^{T} \mathcal{L}_t(\lambda) + \frac{\|\lambda - \lambda_1\|^2 - \|\lambda - \lambda_{T+1}\|^2}{2\alpha} + \frac{\alpha T L^2}{2}. \tag{50}$$

This ends the proof. □

Proof of Proposition 1 for OGMS. The beginning of the proof follows the proof of Theorem 11.1 in [3].

Note that we can rewrite (4) as

$$\begin{cases} \tilde{\lambda}_t = \lambda_{t-1} - \alpha \nabla \mathcal{L}_{t-1}(\lambda_{t-1}) \\ \lambda_t = \Pi_\Lambda(\tilde{\lambda}_t) \end{cases}$$

rearranging the first line, we obtain:

$$\nabla \mathcal{L}_{t-1}(\lambda_{t-1}) = \frac{\lambda_{t-1} - \tilde{\lambda}_t}{\alpha}. \tag{51}$$

By convexity, for any λ,

$$\begin{aligned} \mathcal{L}_{t-1}(\lambda) &\geq \mathcal{L}_{t-1}(\lambda_{t-1}) + (\lambda - \lambda_{t-1})^T \nabla \mathcal{L}_{t-1}(\lambda_{t-1}) && (52) \\ &= \mathcal{L}_{t-1}(\lambda_{t-1}) + \frac{(\lambda - \lambda_{t-1})^T (\lambda_{t-1} - \tilde{\lambda}_t)}{\alpha}, && (53) \end{aligned}$$

that is,

$$\mathcal{L}_{t-1}(\lambda_{t-1}) \leq \mathcal{L}_{t-1}(\lambda) - \frac{(\lambda - \lambda_{t-1})^T (\lambda_{t-1} - \tilde{\lambda}_t)}{\alpha}. \tag{54}$$

Lemma 1 gives:

$$\begin{aligned} (\lambda - \lambda_{t-1})^T (\lambda_{t-1} - \tilde{\lambda}_t) &= \frac{\|\lambda - \tilde{\lambda}_t\|^2 - \|\lambda - \lambda_{t-1}\|^2 - \|\lambda_{t-1} - \tilde{\lambda}_t\|^2}{2} \\ &= \frac{\|\lambda - \tilde{\lambda}_t\|^2 - \|\lambda - \lambda_{t-1}\|^2 - \alpha^2 \|\nabla \mathcal{L}_{t-1}(\lambda_{t-1})\|^2}{2} && (55) \\ &\geq \frac{\|\lambda - \lambda_t\|^2 - \|\lambda - \lambda_{t-1}\|^2 - \alpha^2 \|\nabla \mathcal{L}_{t-1}(\lambda_{t-1})\|^2}{2}, && (56) \end{aligned}$$

the last step being justified by:

$$\|\lambda - \tilde{\lambda}_t\|^2 \geq \|\lambda - \Pi_\Lambda(\tilde{\lambda}_t)\|^2 = \|\lambda - \lambda_t\|^2 \tag{57}$$

for any $\lambda \in \Lambda$. Plug (56) in (54) to get:

$$\mathcal{L}_{t-1}(\lambda_{t-1}) \leq \mathcal{L}_{t-1}(\lambda) + \frac{\|\lambda - \lambda_{t-1}\|^2 - \|\lambda - \lambda_t\|^2}{2\alpha} + \frac{\alpha \|\nabla \mathcal{L}_{t-1}(\lambda_{t-1})\|^2}{2} \tag{58}$$

and the Lipschitz assumption gives:

$$\mathcal{L}_{t-1}(\lambda_{t-1}) \leq \mathcal{L}_{t-1}(\lambda) + \frac{\|\lambda - \lambda_{t-1}\|^2 - \|\lambda - \lambda_t\|^2}{2\alpha} + \frac{\alpha L^2}{2} \tag{59}$$

sum the inequality for $t = 2$ to $T + 1$ to get:

$$\sum_{t=1}^{T} \mathcal{L}_t(\lambda_t) \leq \sum_{t=1}^{T} \mathcal{L}_t(\lambda) + \frac{\|\lambda - \lambda_1\|^2 - \|\lambda - \lambda_{T+1}\|^2}{2\alpha} + \frac{\alpha T L^2}{2}. \tag{60}$$

This ends the proof of the statement for OGMS. $\square$

We now provide a lemma that will be useful for the proof of Proposition 2.

Lemma 2. *Let $G(u, v)$ be a convex function of $(u, v) \in U \times V$. Define $g(u) = \inf_{v \in V} G(u, v)$. Then g is convex.*

Proof. indeed, let $\lambda \in [0,1]$ and $(x,y) \in U^2$,

$$\begin{aligned} g(\lambda x + (1-\lambda)y) &= \inf_{v \in V} G(\lambda x + (1-\lambda)y, v) && (61) \\ &\leq G(\lambda x + (1-\lambda)y, \lambda x' + (1-\lambda)y') && (62) \\ &\leq \lambda G(x,x') + (1-\lambda)G(y,y') && (63) \end{aligned}$$

where the last two inequalities hold for any $(x',y') \in V^2$. Let us now take the infimum with respect to $(x',y') \in V^2$ in both sides, this gives:

$$\begin{aligned} g(\lambda x + (1-\lambda)y) &\leq \inf_{x' \in V} \lambda G(x,x') + \inf_{y' \in V} (1-\lambda)G(y,y') && (64) \\ &= \lambda g(x) + (1-\lambda)g(y), && (65) \end{aligned}$$

that is, g is convex. □

Proof of Proposition 2. Apply Lemma 2 to $u = (\vartheta, \gamma)$, $v = \theta$, $U = \Lambda$, $V = \Theta$ and

$$G(u,v) = \sum_{i=1}^{n} \ell_{i,t}(\theta) + \frac{\gamma \Gamma^2 n}{2} + \frac{\|\vartheta - \theta\|^2}{2\gamma}. \tag{66}$$

This shows $g(u) = \mathcal{L}_t((\vartheta,\gamma))$ is convex with respect (ϑ,γ). Additionally, G is differentiable w.r.t $u = (\vartheta,\gamma)$, so

$$\frac{\partial G}{\partial \vartheta} = \frac{\vartheta - \theta}{\gamma}, \text{ and } \frac{\partial G}{\partial \gamma} = \frac{n\Gamma^2}{2} - \frac{\|\vartheta - \theta\|^2}{2\gamma^2}. \tag{67}$$

As a consequence, for $(\theta,\vartheta) \in \Theta^2$ and $\underline{\gamma} \leq \gamma \leq \overline{\gamma}$,

$$\left\| \frac{\partial G}{\partial \vartheta} \right\|^2 \leq \frac{4C^2}{\underline{\gamma}^2}, \text{ and } \left| \frac{\partial G}{\partial \gamma} \right|^2 \leq \frac{n^2\Gamma^4}{4} + \frac{4C^4}{\underline{\gamma}^4}. \tag{68}$$

This leads to

$$\begin{aligned} \|\nabla_u G(u,v)\| &= \sqrt{\left\| \frac{\partial G}{\partial \vartheta} \right\|^2 + \left| \frac{\partial G}{\partial \gamma} \right|^2} && (69) \\ &\leq \sqrt{\frac{n^2\Gamma^4}{4} + \frac{4C^2}{\underline{\gamma}^2} + \frac{4C^4}{\underline{\gamma}^4}} =: L, && (70) \end{aligned}$$

that is, for each v, $G(u,v)$ is L-Lipschitz in u. So, $g(u) = \inf_{v \in V} G(u,v)$ is L-Lipschitz in u. □

Proof of Theorem 1. Thanks to the Assumption 2, we can apply Proposition 2. That is, Assumption (1) is satisfied, and we can apply Proposition 1. This gives:

$$\begin{aligned} \sum_{t=1}^{T} \sum_{i=1}^{n} \ell_{t,i}(\theta_{t,i}^{\lambda_t}) \leq \inf_{\theta_1,\ldots,\theta_T \in \Theta} \inf_{(\vartheta,\gamma) \in \Lambda} \Bigg\{ \sum_{t=1}^{T} \Bigg[\sum_{i=1}^{n} \ell_{t,i}(\theta_t) \\ + \frac{\gamma \Gamma^2 n}{2} + \frac{\|\theta_t - \vartheta\|^2}{2\gamma} \Bigg] + \frac{\alpha T L^2}{2} + \frac{\|\vartheta - \vartheta_1\|^2 + |\gamma - \gamma_1|^2}{2\alpha} \Bigg\}. \end{aligned} \tag{71}$$

We use direct bounds for the last two terms: $\|\vartheta - \vartheta_1\|^2 \leq 4C^2$ and $|\gamma - \gamma_1|^2 \leq |\overline{\gamma} - \underline{\gamma}|^2 \leq \overline{\gamma}^2 = C^4$. Then note that

$$\sum_{t=1}^{T} \|\theta_t - \vartheta\|^2 = T\left\|\vartheta - \frac{1}{T}\sum_{s=1}^{T}\theta_s\right\|^2 + \sum_{t=1}^{T}\left\|\theta_t - \frac{1}{T}\sum_{s=1}^{T}\theta_s\right\|^2 \tag{72}$$

$$= T\left\|\vartheta - \frac{1}{T}\sum_{s=1}^{T}\theta_s\right\|^2 + T\sigma^2(\theta_1^T). \tag{73}$$

Upper bounding the infimum on ϑ in (71) by $\vartheta = \frac{1}{T}\sum_{s=1}^{T}\theta_s$ leads to

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\lambda_t}) \leq \inf_{\theta_1,\ldots,\theta_T\in\Theta}\inf_{\gamma\in[\underline{\gamma},\overline{\gamma}]}\left\{\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_t) + \frac{\gamma\Gamma^2 nT}{2} + \frac{T\sigma^2(\theta_1^T)}{2\gamma} + \frac{\alpha T L^2}{2} + \frac{C^2(4+C^2)}{2\alpha}\right\}. \tag{74}$$

The right-hand side of (74) is minimized with respect to α if $\alpha = \frac{C}{L}\sqrt{\frac{4+C^2}{T}}$, which is the value proposed in the theorem, and we obtain:

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\lambda_t}) \leq \inf_{\theta_1,\ldots,\theta_T\in\Theta}\inf_{\gamma\in[\underline{\gamma},\overline{\gamma}]}\left\{\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_t) + \frac{\gamma\Gamma^2 nT}{2} + \frac{T\sigma^2(\theta_1^T)}{2\gamma} + CL\sqrt{(4+C^2)T}\right\}. \tag{75}$$

The infimum with respect to γ in the r.h.s is reached for

$$\gamma^* = \left(\underline{\gamma} \vee \frac{\sigma(\theta_1^T)}{\Gamma\sqrt{n}}\right) \wedge \overline{\gamma}. \tag{76}$$

First, note that

$$\frac{\gamma^*\Gamma^2 nT}{2} \leq \left(\underline{\gamma} \vee \frac{\sigma(\theta_1^T)}{\Gamma\sqrt{n}}\right)\frac{\Gamma^2 nT}{2} \tag{77}$$

$$\leq \left(\underline{\gamma} + \frac{\sigma(\theta_1^T)}{\Gamma\sqrt{n}}\right)\frac{\Gamma^2 nT}{2} \tag{78}$$

$$= \frac{\Gamma^2 T n^{1-\beta}}{2} + \frac{\sigma(\theta_1^T)\Gamma T\sqrt{n}}{2}, \tag{79}$$

using $\underline{\gamma} = n^{-\beta}$. Then,

$$\frac{T\sigma^2(\theta_1^T)}{2\gamma^*} \leq \frac{T\sigma^2(\theta_1^T)}{2}\left(\frac{1}{\overline{\gamma}} \vee \frac{\Gamma\sqrt{n}}{\sigma(\theta_1^T)}\right) \tag{80}$$

$$\leq \frac{T\sigma^2(\theta_1^T)}{2}\left(\frac{1}{\overline{\gamma}} + \frac{\Gamma\sqrt{n}}{\sigma(\theta_1^T)}\right) \tag{81}$$

$$= \frac{T\sigma^2(\theta_1^T)}{2C^2} + \frac{\sigma(\theta_1^T)\Gamma T\sqrt{n}}{2} \tag{82}$$

$$\leq \frac{T\sigma(\theta_1^T)}{C} + \frac{\sigma(\theta_1^T)\Gamma T\sqrt{n}}{2}, \tag{83}$$

using $\overline{\gamma} = C^2$ and $\sigma(\theta_1^T) \leq 2C$. Plugging (77), (80) and the definition of L into (75) gives

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\lambda_t}) \leq \inf_{\theta_1,\dots,\theta_T\in\Theta}\Bigg\{\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_t) + C\sqrt{\left(\frac{n^2\Gamma^4}{4} + 4C^2n^{2\beta} + 4C^4n^{4\beta}\right)(4+C^2)T} \tag{84}$$

$$+ \frac{\Gamma^2 T n^{1-\beta}}{2} + \sigma(\theta_1^T)T\left(\Gamma\sqrt{n} + \frac{1}{C}\right)\Bigg\} \tag{85}$$

$$= \inf_{\theta_1,\dots,\theta_T\in\Theta}\Bigg\{\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_t) + C\sqrt{(4+C^2)\left(\frac{n^2\Gamma^4}{4n^{2\vee 4\beta}} + \frac{4C^2n^{2\beta}}{n^{2\vee 4\beta}} + \frac{4C^4n^{4\beta}}{n^{2\vee 4\beta}}\right)}n^{1\vee 2\beta}\sqrt{T} \tag{86}$$

$$+\left[\frac{\Gamma^2}{2}n^{1-\beta} + \left(\Gamma + \frac{1}{nC}\right)\sigma(\theta_1^T)\sqrt{n}\right]T\Bigg\} \tag{87}$$

$$\leq \inf_{\theta_1,\dots,\theta_T\in\Theta}\Bigg\{\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_t) + C\sqrt{(4+C^2)\left(\frac{\Gamma^2}{4} + 4C^2 + 4C^4\right)}n^{1\vee 2\beta}\sqrt{T} \tag{88}$$

$$+\left[\frac{\Gamma^2}{2}n^{1-\beta} + \left(\Gamma + \frac{1}{C}\right)\sigma(\theta_1^T)\sqrt{n}\right]T\Bigg\} \tag{89}$$

$$\leq \inf_{\theta_1,\dots,\theta_T\in\Theta}\Bigg\{\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_t) + \mathcal{C}(\Gamma,C)\left[n^{1\vee 2\beta}\sqrt{T} + \left(n^{1-\beta} + \sigma(\theta_1^T)\sqrt{n}\right)T\right] \tag{90}$$

where we took

$$\mathcal{C}(\Gamma,C) = \max\left(C\sqrt{(4+C^2)\left(\frac{\Gamma^2}{4} + 4C^2 + 4C^4\right)}, \frac{\Gamma^2}{2}, \Gamma + \frac{1}{C}\right). \tag{91}$$

This ends the proof. □

Proof of Theorem 2. A direct application of Proposition 1 gives

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\eta_t}) \leq \inf_{\eta\geq\frac{1}{n}}\Bigg\{\sum_{t=1}^{T}\min_{\theta\in\Theta_0}\Bigg[\sum_{i=1}^{n}\ell_{t,i}(\theta) + \frac{\eta n\left[\max_{\vartheta\in\Theta_0,1\leq i\leq n}\ell_{t,i}(\vartheta)\right]^2}{8} + \frac{\log M}{\eta}\Bigg] + \frac{\alpha T L^2}{2} + \frac{(\eta-1)^2}{2\alpha}\Bigg\}. \tag{92}$$

Thus, we have

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\eta_t}) \leq \inf_{\eta\geq\frac{1}{n}}\left\{\sum_{t=1}^{T}\min_{\theta\in\Theta_0}\left[\sum_{i=1}^{n}\ell_{t,i}(\theta) + \frac{\eta n b^2}{8} + \frac{\log M}{\eta}\right] + \frac{\alpha T L^2}{2} + \frac{(\eta-1)^2}{2\alpha}\right\}. \tag{93}$$

Now, plugging in the right-hand side

$$\eta = \frac{1}{n}\vee\left(\frac{2}{b}\sqrt{\frac{2\log M}{n}}\right)\wedge 1, \tag{94}$$

we obtain:

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\eta_t}) \leq \sum_{t=1}^{T}\min_{\theta\in\Theta_0}\left[\sum_{i=1}^{n}\ell_{t,i}(\theta) + \frac{b^2}{8} + b\sqrt{\frac{n\log(M)}{2}} + \log(M)\right] + \frac{\alpha T L^2}{2} + \frac{1}{2\alpha}. \tag{95}$$

Now, we see that the value $\alpha = \sqrt{2/(TL^2)}$ leads to:

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\eta_t}) \leq \sum_{t=1}^{T}\min_{\theta\in\Theta_0}\left[\sum_{i=1}^{n}\ell_{t,i}(\theta)+\frac{b^2}{8}+b\sqrt{\frac{n\log(M)}{2}}+\log(M)\right]+L\sqrt{2T}. \quad (96)$$

Rearranging terms, and replacing L by its value,

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\eta_t}) \leq \sum_{t=1}^{T}\min_{\theta\in\Theta_0}\sum_{i=1}^{n}\ell_{t,i}(\theta)+bT\sqrt{\frac{n\log(M)}{2}}+\frac{b^2T}{8}+T\log(M)$$
$$+\left(n^2\log M+\frac{nB^2}{8}\right)\sqrt{2T}, \quad (97)$$

that is the statement of the theorem. □

Proof of Theorem 3. An application of Proposition 1 leads to

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\pi_t}) \leq \min_{\pi\in\Lambda}\left\{\sum_{t=1}^{T}\left[\sum_{i=1}^{n}\ell_{t,i}(\theta_t^*)+C\log\frac{1}{\pi(\theta_t^*)}\right]+\frac{\alpha TL^2}{2}+\frac{\|\pi-\pi_1\|^2}{2\alpha}\right\} \quad (98)$$

and so

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\pi_t}) \leq \min_{\pi\in\Lambda}\left\{\sum_{t=1}^{T}\left[\sum_{i=1}^{n}\ell_{t,i}(\theta_t^*)+C\log\frac{1}{\pi(\theta_t^*)}\right]+\frac{\alpha TL^2}{2}+\frac{1}{2\alpha}\right\} \quad (99)$$

define π_{I^*} such that $\pi_{I^*}(\theta_j) = 1/(2m^*)$ if $j \in I^*$ and $\pi_{I^*}(\theta_j) = 1/(2(M-m^*))$ otherwise. We have $\pi_I^* \in \Lambda$ and thus

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\ell_{t,i}(\theta_{t,i}^{\pi_t}) \leq \sum_{t=1}^{T}\left[\sum_{i=1}^{n}\ell_{t,i}(\theta_t^*)+C\log(2m^*)\right]+\frac{\alpha TL^2}{2}+\frac{1}{2\alpha}. \quad (100)$$

Replace L and α by their values to get the theorem. □

7. Conclusions

We proposed two simple meta-learning strategies together with their theoretical analysis. Our results clearly show an improvement on learning in isolation if the tasks are similar enough. These theoretical findings are confirmed by our numerical experiments. Important questions remain open. In [27], a purely online method is proposed, in the sense that it does not require storing all of the information of the current task. In the case of OGA, this method allows us to learn the starting point. However, its application to learn the step size is not direct [28]. An important question is, then: is there a purely online method that would provably improve on learning in isolation in this case? Another important question is the automatic calibration of Γ. However, as mentioned in Section 5, we believe that a very general and efficient meta-learning method for learning priors in Bayesian statistics (or in generalized Bayesian inference) would be extremely valuable in practice.

Author Contributions: Investigation, D.M. and P.A.; Software, D.M.; Writing—original draft, D.M. and P.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: This project was initiated as Dimitri Meunier's internship project at RIKEN AIP, in the Approximate Bayesian Inference team. The internship was cancelled because of the pandemic. We would like to thank Arnak Dalalyan (ENSAE Paris), who provided fundings so that the internship could take place at ENSAE Paris instead. We would like to thank Emtiyaz Khan (RIKEN AIP), Sébastien Gerchinovitz (IRT Saint-Exupéry, Toulouse), Vianney Perchet (ENSAE Paris) and all the members of the Approximate Bayesian Inference team for valuable feedback.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Thrun, S.; Pratt, L. *Learning to Learn*; Kluwer Academic Publishers: New York, NY, UK, 1998.
2. Chollet, F. On the measure of intelligence. *arXiv* **2019**, arXiv:1911.01547.
3. Cesa-Bianchi, N.; Lugosi, G. *Prediction, Learning, and Games*; Cambridge University Press: Cambridge, UK, 2006.
4. Hazan, E. Introduction to online convex optimization. *arXiv* **2019**, arXiv:1909.05207.
5. Orabona, F. A modern introduction to online learning. *arXiv* **2019**, arXiv:1912.13213.
6. Shalev-Shwartz, S. Online learning and online convex optimization. *Found. Trends Mach. Le.* **2012**, *4*, 107–194. [CrossRef]
7. Maurer, A. Bounds for linear multi-task learning. *J. Mach. Learn. Res.* **2006**, *7*, 117–139.
8. Romera-Paredes, B.; Aung, H.; Bianchi-Berthouze, N.; Pontil, M. Multilinear multitask learning. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 1444–1452.
9. Yamada, M.; Koh, T.; Iwata, T.; Shawe-Taylor, J.; Kaski, S. Localized lasso for high-dimensional regression. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, PMLR, Fort Lauderdale, FL, USA, 20–22 April 2017; Volume 54, pp. 325–333.
10. Alquier, P.; Mai, T.T.; Pontil, M. Regret Bounds for Lifelong Learning. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017; Volume 54, pp. 261–269.
11. Amit, R.; Meir, R. Meta-learning by adjusting priors based on extended PAC-Bayes theory. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 205–214.
12. Baxter, J. Theoretical models of learning to learn. In *Learning to Learn*; Springer: Berlin, Germany, 1998; pp. 71–94.
13. Jose, S.T.; Simeone, O.; Durisi, G. Transfer meta-learning: Information-theoretic bounds and information meta-risk minimization. *arXiv* **2020**, arXiv:2011.02872.
14. Maurer, A.; Pontil, M.; Romera-Paredes, B. The benefit of multitask representation learning. *J. Mach. Learn. Res.* **2016**, *17*, 1–32.
15. Pentina, A.; Lampert, C. A PAC-Bayesian bound for lifelong learning. In Proceedings of the 31st International Conference on Machine Learning, Bejing, China, 22–24 June 2014; pp. 991–999.
16. Rothfuss, J.; Fortuin, V.; Krause, A. Pacoh: Bayes-optimal meta-learning with pac-guarantees. *arXiv* **2020**, arXiv:2002.05551.
17. Andrychowicz, M.; Denil, M.; Gomez, S.; Hoffman, M.W.; Pfau, D.; Schaul, T.; Shillingford, B.; De Freitas, N. Learning to learn by gradient descent by gradient descent. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 3981–3989.
18. Ruvolo, P.; Eaton, E. Ella: An efficient lifelong learning algorithm. In Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, 17–19 June 2013; pp. 507–515.
19. Balcan, M.-F.; Khodak, M.; Talwalkar, A. Provable guarantees for gradient-based meta-learning. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 424–433.
20. Denevi, G.; Ciliberto, C.; Stamos, D.; Pontil, M. Learning to learn around a common mean. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018; pp. 10169–10179.
21. Denevi, G.; Ciliberto, C.; Grazzi, R.; Pontil, M. Learning-to-learn stochastic gradient descent with biased regularization. *arXiv* **2019**, arXiv:1903.10399.
22. Denevi, G.; Pontil, M.; Ciliberto, C. The advantage of conditional meta-learning for biased regularization and fine tuning. In Proceedings of the Advances in Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33.
23. Fallah, A.; Mokhtari, A.; Ozdaglar, A. On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Online, 26–28 August 2020; pp. 1082–1092.
24. Finn, C.; Rajeswaran, A.; Kakade, S.; Levine, S. Online meta-learning. *arXiv* **2019**, arXiv:1902.08438.
25. Konobeev, M.; Kuzborskij, I.; Szepesvári, C. On optimality of meta-learning in fixed-design regression with weighted biased regularization. *arXiv* **2020**, arXiv:2011.00344.
26. Zhou, P.; Yuan, X.; Xu, H.; Yan, S.; Feng, J. Efficient meta learning via minibatch proximal update. In Proceedings of the 2019 Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 1534–1544.
27. Denevi, G.; Stamos, D.; Ciliberto, C.; Pontil, M. Online-within-online meta-learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 13110–13120.
28. Meunier, D. Meta-Learning Meets Variational Inference: Learning Priors with Guarantees. Master's Thesis, Université Paris Saclay, Paris, France, 2020. Available online: https://dimitri-meunier.github.io/files/RikenReport.pdf (accessed on 26 September 2021).
29. Khodak, M.; Balcan, M.-F.; Talwalkar, A. Adaptive Gradient-Based Meta-Learning Methods. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 5917–5928.

30. Li, L.; Jamieson, K.; DeSalvo, G.; Rostamizadeh, A.; Talwalkar, A. Hyperband: A novel bandit-based approach to hyperparameter optimization. *J. Mach. Learn. Res.* **2017**, *18*, 6765–6816.
31. Shang, X.; Kaufmann, E.; Valko, M. A simple dynamic bandit algorithm for hyper-parameter tuning. In Proceedings of the 6th ICML Workshop on Automated Machine Learning, Long Beach, CA, USA, 14–15 June 2019.
32. Kivinen, J.; Warmuth, M.K. Exponentiated gradient versus gradient descent for linear predictors. *Inf. Comput.* **1997**, *132*, 1–63. [CrossRef]
33. Kulis, B.; Bartlett, P.L. Implicit online learning. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 575–582.
34. Parikh, N.; Boyd, S. Proximal algorithms. *Found. Trends Optim.* **2014**, *1*, 127–239. [CrossRef]
35. Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*; Springer Science & Business Media: Berlin, Germany, 2004; Volume 87.
36. Alquier, P. Approximate Bayesian Inference. *Entropy* **2020**, *22*, 1272. [CrossRef] [PubMed]
37. Mai, T.T. On continual single index learning. *arXiv* **2021**, arXiv:2102.12961.
38. Lin, W.; Khan, M.E.; Schmidt, M. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Volume 97, pp. 3992–4002.
39. Chérief-Abdellatif, B.-E.; Alquier, P.; Khan, M.E. A generalization bound for online variational inference. In Proceedings of the Eleventh Asian Conference on Machine Learning, PMLR, Nagoya, Japan, 17–19 November 2019; Volume 101, pp. 662–677.
40. Domke, J. Provable smoothness guarantees for black-box variational inference. In Proceedings of the 37th International Conference on Machine Learning, Online, 12–18 July 2021; Volume 119, pp. 2587–2596.
41. Alquier, P. Non-exponentially weighted aggregation: Regret bounds for unbounded loss functions. In Proceedings of the 38th International Conference on Machine Learning, Online, 18–24 July 2021; Volume 139, pp. 207–218.
42. Knoblauch, J.; Jewson, J.; Damoulas, T. Generalized variational inference: Three arguments for deriving new posteriors. *arXiv* **2019**, arXiv:1904.02063.
43. Campolongo, N.; Orabona, F. Temporal variability in implicit online learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020; Volume 33.

entropy

MDPI

Article

Fast Compression of MCMC Output

Nicolas Chopin *,† and Gabriel Ducrocq †

Institut Polytechnique de Paris, ENSAE Paris, CEDEX, 92247 Malakoff, France; gabriel.ducrocq@ensae.fr
* Correspondence: nicolas.chopin@ensae.fr
† These authors contributed equally to this work.

Abstract: We propose cube thinning, a novel method for compressing the output of an MCMC (Markov chain Monte Carlo) algorithm when control variates are available. It allows resampling of the initial MCMC sample (according to weights derived from control variates), while imposing equality constraints on the averages of these control variates, using the cube method (an approach that originates from survey sampling). The main advantage of cube thinning is that its complexity does not depend on the size of the compressed sample. This compares favourably to previous methods, such as Stein thinning, the complexity of which is quadratic in that quantity.

Keywords: control variates; Markov chain Monte Carlo; thinning

Citation: Chopin, N.; Ducrocq, G. Fast Compression of MCMC Output. *Entropy* **2021**, *23*, 1017. https://doi.org/10.3390/e23081017

Academic Editor: Cathy W. S. Chen

Received: 6 July 2021
Accepted: 3 August 2021
Published: 6 August 2021

1. Introduction

MCMC (Markov chain Monte Carlo) remains, to this day, the most popular approach to sampling from a target distribution p, in particular in Bayesian computations [1].

Standard practice is to run a single chain, $X_1, \dots, X_N$ according to a Markov kernel that leaves invariant p. It is also common to discard part of the simulated chain, either to reduce its memory footprint, or to reduce the CPU cost of later post-processing operations, or more generally for the user's convenience. Historically, the two common recipes for compressing an MCMC output are:

- burn-in, which allows discarding the b first states;
- thinning, which allows retaining only one out of t (post burn-in) states.

The impact of either recipes on the statistical properties of the subsampled estimates are markedly different. Burn-in reduces the bias introduced by the discrepancy between p and the distribution of the initial state X_1 (since $X_b \approx p$ for b large enough). On the other hand, thinning always increases the (asymptotic) variance of MCMC estimators [2].

Practitioners often choose b (the burn-in period) and t (the thinning frequency) separately, in a somewhat ad hoc fashion (i.e., through visual inspection of the initial chain), or using convergence diagnosis such as, e.g., those reviewed in [3].

Two recent papers [4,5] cast a new light on the problem of compressing an MCMC chain by considering, more generally, the problem, for a given M, of selecting the subsample of size M that best represents (according to a certain criterion) the target distribution p. We focus for now on [5], for reasons we explain below.

Stein thinning, the method developed in [5], chooses the subsample $\mathcal{S}$ of size M which minimises the following criterion:

$$D(\mathcal{S}) := \frac{1}{M^2} \sum_{m,n \in \mathcal{S}} k_p(X_m, X_n), \quad \mathcal{S} \subset \{1, \dots, N\}, \quad |\mathcal{S}| = M \tag{1}$$

where k_p is a p-dependent kernel function derived from another kernel function k: $\mathcal{X} \times \mathcal{X} \to \mathbb{R}$, as follows:

$$k_p(x,y) = \nabla_x \cdot \nabla_y k(x,y) + \langle \nabla_x k(x,y), s_p(y) \rangle + \langle \nabla_y k(x,y), s_p(x) \rangle + k(x,y) \langle s_p(x), s_p(y) \rangle$$

with $\langle\cdot,\cdot\rangle$ being the Euclidean inner product, $s_p(x) := \nabla \log p(x)$ is the so-called score function (gradient of the log target density), and ∇ is the gradient operator.

The rationale behind criterion (1) is that it may be interpreted as the KSD (kernel Stein discrepancy) between the true distribution p and the empirical distribution of subsample S. We refer to [5] for more details on the theoretical background of the KSD, and its connection to Stein's method.

Stein thinning is appealing, as it seems to offer a principled, quasi-automatic way to compress an MCMC output. However, closer inspection reveals the following three limitations.

First, it requires computing the gradient of the log-target density, $s_p(x) = \nabla \log p(x)$. This restricts the method to problems where this gradient exists and is tractable (and, in particular, to $\mathcal{X} = \mathbb{R}^d$).

Second, its CPU cost is $\mathcal{O}(NM^2)$. This makes it nearly impossible to use Stein thinning for $M \gg 100$. This cost stems from the greedy algorithm proposed in [5], see their Algorithm 1, which adds at iteration t the state X_i which minimises $k_p(X_i, X_i) + \sum_{j\in S_{t-1}} k_p(X_i, X_j)$, where S_{t-1} is the sample obtained from the $t-1$ previous iterations.

Third, its performance seems to depend in a non-trivial way on the original kernel function k; the authors of [5] propose several strategies for choosing and scaling k, but none of them seem to perform uniformly well in their numerical experiments.

We propose a different approach in this paper, which we call cube thinning, and which addresses these shortcomings to some extent. Assuming the availability of J control variates (that is, of functions h_j with known expectation under p), we cast the problem of MCMC compression as that of resampling the initial chain under constraints based on these control variates. The main advantage of cube thinning is that its complexity is $\mathcal{O}(NJ^3)$; in particular, it does not depend on M. That makes it possible to use it for much larger values of M. We shall discuss the choice of J, but, by and large, J should be of the same order as d, the dimension of the sampling space. The name stems from the cube method of [6], which plays a central part in our approach, as we explain in the body of the paper.

The availability of control variates may seem like a strong requirement. However, if we assume we are able to compute $s_p(x) = \nabla \log p(x)$, then (for a large class of functions $\phi : \mathbb{R}^d \to \mathbb{R}^d$, which we define later)

$$\mathbb{E}_p\left[\phi(x) s_p(x) + \nabla_x \cdot \phi(x)\right] = 0$$

where $\nabla_x \cdot \phi$ denotes the divergence of ϕ. In other words, the availability of the score function implies, automatically, the availability of control variates. The converse is not true: there exists control variates, e.g., [7], that are not gradient-based. One of the examples we consider in our numerical examples feature such non gradient-based control variates; as a result, we are able to apply cube thinning, although Stein thinning is not applicable.

The supporting methods of [4] do not require control variates. It is thus more generally applicable than either cube thinning or Stein thinning. On the other hand, when gradients (and thus control variates) are available, the numerical experiments of [5] suggest that Stein thinning outperforms support points. From now on, we focus on situations where control variates are available.

This paper is organised as follows. Section 2 recalls the concept of control variates, and explains how control variates may be used to reweight an MCMC sample. Section 3 describes the cube method of [6]. Section 4 explains how to combine control variates and the cube method to perform cube thinning. Section 5 assesses the statistical performance of cube thinning through two numerical experiments.

We use the following notations throughout: p denotes both the target distribution and its probability density; $p(f)$ is a short-hand for the expectation of $f(X)$ under p. The gradient of a function f is denoted by $\nabla_x f(x)$, or simply $\nabla f(x)$ when there is no ambiguity. The $i-$-th component of a vector $v \in \mathbb{R}^d$ is denoted by $v[i]$, and it is transposed by v^t. The vectors of the canonical basis of $\mathbb{R}^d$ are denoted by e_i, i.e., $e_i[j] = 1$ if $j = i$, 0 otherwise. Matrices are written in upper-case; the kernel (null space) of matrix A is denoted by ker$A$. The set of functions $f : \Omega \to \mathbb{R}^d$ that are continuously differentiable is denoted by $C^1(\Omega, \mathbb{R}^d)$.

2. Control Variates

2.1. Definition

Control variates are a very well known way to reduce the variance of Monte Carlo estimates—see, e.g., the books of [1,8,9].

Suppose we want to estimate the quantity $p(f) = \mathbb{E}_p[f(X)]$ for a suitable $f : \mathbb{R}^d \to \mathbb{R}$, based on an IID (independent and identically distributed) sample $\{X_1, \ldots, X_N\}$ from distribution p. The generalisation of control variates to MCMC will be discussed in Section 4.

The usual Monte Carlo estimator of $p(f)$ is

$$\hat{p}(f) = \frac{1}{N} \sum_{n=1}^{N} f(X_n). \tag{2}$$

Assume we know $J \in \mathbb{N}^\star$ functions $h_j : \mathbb{R}^d \to \mathbb{R}$ for $j \in \{1, \ldots, J\}$ such that $p(h_j) = 0$. Functions with this property are called control variates. We can use this property to build an estimate with a lower variance: let us denote $h(X) = (h_1(X), \ldots, h_J(X))^t$ and write our new estimate:

$$\hat{p}_\beta(f) = \frac{1}{N} \sum_{n=1}^{N} f(X_n) + \beta^t h(X_n) \tag{3}$$

with $\beta \in \mathbb{R}^J$. Then it is straightforward to show that $\mathbb{E}[\hat{p}_\beta(f)] = \mathbb{E}[\hat{p}(f)] = p(f)$. Depending on the choice of β, we may have $\text{Var}[\hat{p}_\beta(f)] \leq \text{Var}[\hat{p}(f)]$. The next section discusses how to choose such a β.

2.2. Control Variates as a Weighting Scheme

The standard approach to choose β consists of two steps. First, one shows easily that the value the minimises the variance of estimator (3) is:

$$\beta^\star(f) = \text{Var}(h(X))^{-1}\text{Cov}(h(X), f(X)) \tag{4}$$

where $\text{Var}(h(X))$ is the $J \times J$ variance matrix of the vector $h(X)$ and $\text{Cov}(h(X), f(X))$ is the $J \times 1$ vector such that $\text{Cov}(h(X), f(X))_{i,1} = \text{Cov}(f(X), h_i(X))$.

Second, one realises that this quantity may be estimated from the sample $X_1, \ldots, X_N$ through a simple linear regression model, where the $f(X_n)$s are the outcome, and the $h_j(X_n)$s are the predictors:

$$f(X_n) \approx \mu + \beta^t h(X_n) + \epsilon_n, \quad \mathbb{E}[\epsilon_n] = 0. \tag{5}$$

More precisely, let $\gamma \in R^{J+1}$ be the vector such that $\gamma^t = (\mu, \beta^t)$, $H = (H_{ij})$ the design matrix such that $H_{i1} = 1$, $H_{i(j+1)} = h_j(X_i)$, and $F = (f(X_1), \ldots, f(X_N))$. Then, the OLS (ordinary least squares) estimate of γ is

$$\widehat{\gamma}_{\text{OLS}} = (H^t H)^{-1} H^t F. \tag{6}$$

Since $\mathbb{E}[f(X_n)] = \mu$ in this artificial regression model, the first component of $\widehat{\gamma}_{\text{OLS}}$:

$$\hat{p}_\star(f) := \widehat{\gamma}_{\text{OLS}} \times e_1, \tag{7}$$

actually corresponds to estimate (3) when $\beta = \widehat{\beta}_{\text{OLS}}$.

At first glance, the approach described above seems to require implementing a different linear regression for each function f of interest. Ref. [9] noted, however, that one may re-express (7) as a weighted average:

$$\hat{p}_\star(f) = \sum_{n=1}^{N} w_n f(X_n) \tag{8}$$

where the weights w_n sum to one, and do not depend on f. It is thus possible to compute these weights once from a given sample (given a certain choice of control variates), and then quickly compute $\hat{p}_\star(f)$ for any function f of interest.

The exact expression of the weights are easily deduced from (7) and (6): $w = (w_n)$ with

$$w = H(H^t H)^{-1} e_1.$$

2.3. Gradient-Based Control Variates

In this section and the next, we recall generic methods to construct control variates. This section specifically considers control variates that are derived from the score function, $s_p(x) = \nabla \log p(x)$. (We therefore assume that this quantity is tractable.)

Under the following two conditions:

1. The probability density $p \in C^1(\Omega, \mathbb{R})$ where $\Omega \subseteq \mathbb{R}^d$ is an open set;
2. Function $\phi \in C^1(\Omega, \mathbb{R}^d)$ is such that $\oint_{\partial\Omega} p(x)\phi(x) \cdot n(x) S(dx) = 0$ where $\oint_{\partial\Omega}$ denotes the integral over the boundary of Ω, and $S(dx)$ is the surface element at $x \in \partial\Omega$.

The following function:

$$h(x) = \nabla_x \cdot \phi(x) + \phi(x) \cdot s_p(x) \tag{9}$$

is a control variate: $p(h) = 0$, see, e.g., [10] or [11] for further details. To gain some insight, note that in dimension 1 and assuming the domain of integration is an interval $]a, b[\subset \mathbb{R}$, this amounts to an integration by parts with the condition that $h(b)p(b) - h(a)p(a) = 0$.

Thus, whenever the score function is available (and the conditions above hold), we are able to construct an infinite number of control variates (one for each function ϕ). For simplicity, we shall focus on the following standard classes of such functions. First, for $i = 1, \ldots, d$,

$$\begin{aligned} \phi_i \colon \mathbb{R}^d &\to \mathbb{R}^d \\ x &\mapsto e_i \end{aligned}$$

which leads to the following d control variates:

$$h_i(x) = s_p(x)[i]. \tag{10}$$

For a Gaussian target, $N(\mu, \Sigma)$, the score is $s_p(x) = -\Sigma^{-1}(x - \mu)$, and the control variates above make it possible to reweight the Monte Carlo sample to make it have the same expectation as the target distribution.

Second, we consider, for $i, j = 1, \ldots, d$:

$$\begin{aligned} \phi_{ij} \colon \mathbb{R}^d &\to \mathbb{R}^d \\ x &\mapsto x[i] e_j \end{aligned}$$

which leads to the following d^2 control variates:

$$h_{ij}(x) = \mathbb{1}\{i = j\} + x[i] s_p(x)[j]. \tag{11}$$

Again, for a Gaussian target $N(\mu, \Sigma)$, this makes it possible to fix the empirical covariance matrix to true covariance Σ.

In our simulations, we consider two sets of control variates: the 'full' set, consisting of the d control variates defined by (10), and the d^2 control variates defined by (11), and a 'diagonal' set of $2d$ control variates, where for (11), we only consider the cases where $i = j$. Of course, the former set should lead to a better performance (lower variance), but since the complexity of our approach will be $\mathcal{O}(J^3)$, where J is the number of control variates, taking $J = \mathcal{O}(d^2)$ may be too expensive whenever the dimension d is large. In fact, when d is very large, one might even consider considering only control variates that depend on a few components of x of interest.

2.4. MCMC-Based Control Variates

We mention in passing other ways to construct control variates, in particular in the context of MCMC.

For instance, [7] noted that, for a Markov chain $\{X_n\}$, the quantity

$$\phi(X_n) - \mathbb{E}[\phi(X_n)|X_{n=1}]$$

has zero expectations. In particular, if the MCMC kernel is a Gibbs sampler, it is likely that one is able to compute the conditional expectation of each component, i.e., $\phi(x) = x[i]$ for $i = 1, \ldots, d$.

See also [12,13] for other ways to construct control variates when the X_ns are simulated from a Metropolis kernel.

3. The Cube Method

We review in this section the cube method of [6]. This method originated from survey sampling and is a way to sample from a finite population under constraints. The first subsection gives some definitions, the second one explains the flight phase of the cube method and the third subsection discusses the landing phase of the method.

3.1. Definitions

Suppose we have a finite population $\{1, \ldots, N\}$ of N individuals and that to each individual $n = 1, \ldots, N$ is associated a variable of interest y_n and J auxiliary variables, $v_n = (v_{n1}, \ldots, v_{nJ})$. Without loss of generality, suppose also that the J vectors $(v_{1j}, \ldots, v_{Nj})$ are linearly independent. We are interested in estimating the quantity $Y = \sum_{n=1}^{N} y_n$ using a subsample of $\{1, \ldots, N\}$. Furthermore, we know the exact value of each sum $V_j = \sum_{n=1}^{N} v_{nj}$, and we wish to use this auxiliary information to better estimate Y.

We assign, to each individual n, a sampling probability $\pi_n \in [0, 1]$. We consider random variables S_n such that, marginally, $\mathbb{P}(S_n = 1) = \pi_n$. We may then define the Horvitz–Thompson estimator of Y as

$$\hat{Y} = \sum_{n=1}^{N} \frac{S_n y_n}{\pi_n} \tag{12}$$

which is unbiased, and which depends only on selected individuals (i.e., $S_n = 1$).

We define similarly the Horvitz–Thompson estimator of V_j as

$$\hat{V}_j = \sum_{n=1}^{N} \frac{S_n v_{nj}}{\pi_n}. \tag{13}$$

Our objective is to construct a joint distribution ξ for the inclusion variables S_n such that $\mathbb{P}_\xi(S_n = 1) = \pi_n$ for all $n = 1, \ldots, N$, and

$$\hat{V} = V \quad \xi\text{-almost surely.} \tag{14}$$

where $V = (V_1, \ldots, V_J)$, $\hat{V} = (\hat{V}_1, \ldots, \hat{V}_J)$. Such a probability distribution is called a balanced sampling design.

3.2. Subsamples as Vertices

We can view all the possible samples from $\{1, \dots, N\}$ as the vertices of the hypercube $\mathcal{C} = [0,1]^N$ in $\mathbb{R}^N$. A sampling design with inclusion probabilities $\pi_n = \mathbb{P}_\xi(S_n = 1)$ is then a distribution over the set of these vertices such that $\mathbb{E}[S] = \pi$, where $S = (S_1, \dots, S_N)^t$, and $\pi = (\pi_1, \dots, \pi_N)^t$ is the vector of inclusion probabilities. Hence, π is expressed as a convex combination of the vertices of the hypercube.

We can think of a sampling algorithm as finding a way to reach any vertex of the cube, starting at π, while satisfying the balancing Equation (14). However, before we describe such a sampling algorithm, we may wonder if it is possible to find a vertex such that (14) is satisfied.

3.3. Existence of a Solution

The balancing equation, Equation (14), defines a linear system. Indeed, we can re-express (14) as S, as a solution to $As = V$, where $A = (A_{jn})$ is of dimension $J \times N$, $A_{jn} = v_{kn}/\pi_n$. This system defines a hyperplane Q of dimension $N - J$ in $\mathbb{R}^N$.

What we want is to find vertices of the hypercube $\mathcal{C}$ that also belong to the hyperplane Q. Unfortunately, it is not necessarily possible, as it depends on how the hyperplane Q intersects cube $\mathcal{C}$. In addition, there is no way to know beforehand if such a vertex exists. Since $\pi \in Q$, we know that $\mathcal{K} := \mathcal{C} \cap Q \neq \emptyset$ and is of dimension $N - J$. The only thing we can say is stated Proposition 1 in [6]: if r is a vertex of $\mathcal{K}$, then in general $q = \mathrm{card}(\{n : 0 < r[n] < 1\}) \leq J$.

The next section describes the flight phase of the cube algorithm, which generates a vertex in $\mathcal{K}$ when such vertices exist, or which, alternatively, returns a point in $\mathcal{K}$ with most (but not all) components set to zero or one. In the latter case, one needs to implement a landing phase, which is discussed in Section 3.5.

3.4. Flight Phase

The flight phases simulates a process $\pi(t)$ which takes values in $\mathcal{K} = \mathcal{C} \cap Q$, and starts at $\pi(0) = \pi$. At every time t, one selects a unit vector $u(t)$, then one chooses randomly between one of the two points that are in the intersection of the hypercube $\mathcal{C}$ and the line parallel to $u(t)$ that passes through $\pi(t-1)$. The probability of selecting these two points are set to ensure that $\pi(t)$ is a martingale; in that way, we have $\mathbb{E}[\pi_t] = \pi$ at every time step. The random direction $u(t)$ must be generated to fulfil the following two requirements: (a) that the two points are in Q, i.e., $u(t) \in \ker A$, and (b) whenever $\pi(t)$ reaches one of the faces of the hypercube, it must stay within that face; thus, $u(t)[k] = 0$ if $\pi(t-1)[k] = 0$ or 1.

Algorithm 1 describes one step of the flight phase.

Algorithm 1: Flight phase iteration

```
    Input: $\pi(t-1)$
    Output: $\pi(t)$
1   Sample $u(t)$ in $\ker A$ with $u_k(t) = 0$ if the $k$-th component of $\pi(t-1)$ is an integer.
2   Compute $\lambda_1^\star$ and $\lambda_2^\star$, the largest values of $\lambda_1 > 0$ and $\lambda_2 > 0$ such that:
      $0 \leq \pi(t-1) + \lambda_1 u(t) \leq 1$ and $0 \leq \pi(t-1) - \lambda_2 u(t) \leq 1$.
3   With probability $\lambda_2^\star/(\lambda_1^\star + \lambda_2^\star)$, set $\pi(t) \leftarrow \pi(t-1) + \lambda_1 u(t)$; otherwise, set
      $\pi(t) \leftarrow \pi(t-1) - \lambda_2 u(t)$.
```

The flight phase stops when Step 1 of Algorithm 1 cannot be performed (i.e., no vector $u(t)$ fulfils these conditions). Until this happens, each iteration increases by at least one the number of components in $\pi(t)$ that are either zero or one. Thus, the flight phases completes at most in N steps.

In practice, to generate $u(t)$, one may proceed as follows: first generate a random vector $v(t) \in \mathbb{R}^N$, then project it in the constraint hyperplane: $u(t) = I(t)v(t) - I(t)A^t(AI(t)A^t)^-$

$AI(t)v(t)$, where $I(t)$ is a diagonal matrix such that $I_{kk}(t)$ is 0 if $\pi_k(t)$ is an integer and 1 otherwise, and M^- denotes the pseudo-inverse of the matrix M.

The authors of [14] propose a particular method to generate vector $v(t)$, which ensures that the complexity of a single iteration of the flight phase is $\mathcal{O}(J^3)$. This leads to an overall complexity of $\mathcal{O}(NJ^3)$ for the flight phase, since it terminates in at most N iterations.

3.5. Landing Phase

Denote by $\pi^\star$ the value of process $\pi(t)$ when the flight phase terminates. If $\pi^\star$ is a vertex of $\mathcal{C}$ (i.e., all its components are either zero or one), one may stop and return $\pi^\star$ as the output of the cube algorithm. If $\pi^\star$ is not a vertex, this informs us that no vertex belongs to $\mathcal{K}$. One may implement a landing phase, which aims at choosing randomly a vertex which is close to $\pi^\star$, and such that the variance of the components of $\hat{V}$ is small.

Appendix A gives more details on the landing phase. Note that its worst-case complexity is $\mathcal{O}(2^J)$. However, in practice, it is typically either much faster, or not required (i.e., $\pi^\star$ is already a vertex) as soon as $J \ll N$.

4. Cube Thinning

We now explain how the previous ingredients (control variates, and the cube method) may be combined in order to thin a Markov chain, $X_1, \ldots, X_N$, into a subsample of size M. As before, the invariant distribution of the chain is denoted by p, and we assume we know of J control variates h_j, i.e., $p(h_j) = 0$ for $j = 1, \ldots, J$.

4.1. First Step: Computing the Weights

The first step of our method is to use the J control variates to compute the N weights w_n, as defined at the end of Section 2.2. Recall that these weights sum to one, and that they automatically fulfil the constraints:

$$\sum_{n=1}^{N} w_n h_j(X_n) = 0 \tag{15}$$

for $j = 1, \ldots, J$, and that we use them to compute

$$\hat{p}_\star(f) = \sum_{n=1}^{N} w_n f(X_n) \tag{16}$$

as a low-variance estimate for $p(f)$ for any f.

Recall that the control variates procedure we described in Section 2 assume that the input variables, $X_1, \ldots, X_N$, are IID. This is obviously not the case in an MCMC context; however, we follow the common practice [10,11] of applying the procedure to MCMC points as if they were IID points. This implies that the weighted estimate above corresponds to a value of β in (3) that does not minimise the (asymptotic) variance of estimator (3). It is actually possible to estimate the value of β that minimises the asymptotic variance of an MCMC estimate [7,15]. However, this type of approach is specific to certain MCMC samplers, and, critically for us, it cannot be cast as a weighting scheme. Thus, we stick to this standard approach.

We note in passing that, in our experiments (see Figure 1 and the surrounding discussion), the weights w_n make it easy to visually assess the convergence (and thus the burn-in) of the Markov chain. In fact, since the MCMC points of the burn-in phase are far from the mass of the target distribution, the procedure must assign a small or negative weight to these points in order to respect the constraints based on the control variates. Again, see Section 5.2 for more discussion on this issue. The fact that control variates may be used to assess MCMC convergence has been known for a long time (e.g., [16]), but the visualisation of weights makes this idea more expedient.

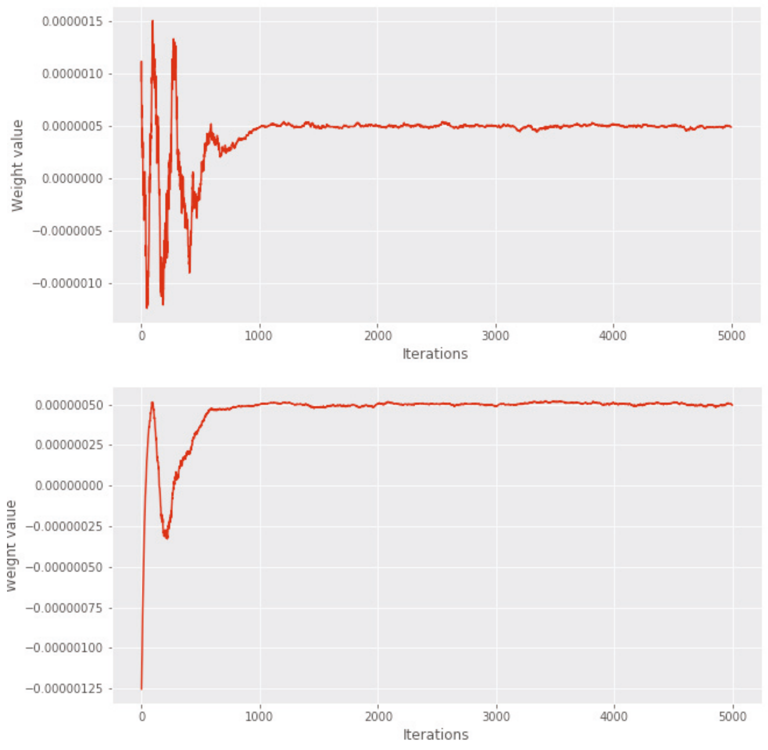


Figure 1. Lotka–Volterra example: first 5000 weights of the cube methods, based on full (**top**) or diagonal (**bottom**) set of covariates.

4.2. Second Step: Cube Resampling

The second step consists in resampling the weighted sample $(w_n, X_n)_{n=1,\dots,N}$, to obtain a subsample $\mathcal{S} = \{X_n : S_n = 1\}$ where S_n are random variables such that (a) $\mathbb{E}[S_n] = w_n$; (b) $\sum_{n=1}^N S_n = M$, and (c) for $j = 1, \dots, J$:

$$\sum_{S_n=1} h_j(X_n) = 0.$$

Condition (a) ensures that the procedure does not introduce any bias:

$$\mathbb{E}\left[\frac{1}{M}\sum_{S_n=1} f(X_n)\middle| X_{1:N}\right] = \sum_{n=1}^N w_n f(X_n).$$

Condition (b) ensures that the subsample is exactly of size M.

We would like to use the cube method in order to generate the S_n's. Specifically, we would like to assign the inclusion probabilities π_n to w_n, and impose the $(J+1)$ constraints defined above by conditions (b) and (c). There is one caveat, however: the weights w_n do not necessarily lie in $[0, 1]$.

4.3. Dealing with Weights Outside of $[0, 1]$

We rewrite (16) as:

$$\hat{p}_\star(f) = \frac{\Omega}{M} \times \sum_{n=1}^N W_n \times \operatorname{sgn}(w_n) f(X_n) \tag{17}$$

where $\Omega = \sum_{n=1}^{N} |w_n|$ and $W_n = M|w_n|/\Omega$. We now have $W_n \geq 0$, and $\sum_{n=1}^{N} W_n = M$, which is required for condition (b) in the previous section. We might have a few points such that $W_n > 1$. In that case, we replace them by $\lfloor W_n \rfloor$ copies, with adjusted weights $W_n / \lfloor W_n \rfloor$.

It then becomes possible to implement the cube method, using as inclusion probabilities the W_ns, and as the matrix A that defines the $J+1$ constraints, the matrix $A = (A_{jn})$ such that $A_{1n} = 1$, $A_{(j+1)n} = \mathrm{sgn}(w_n) h_j(X_n)$. The cube method samples variables S_n, which may be used to compute the subsampled estimate

$$\hat{\nu}(f) = \frac{\Omega}{M} \sum_{S_n=1} \mathrm{sgn}(w_n) f(X_n). \tag{18}$$

More generally, in our numerical experiments, we shall evaluate to which extent the random signed measure:

$$\hat{\nu} = \frac{\Omega}{M} \sum_{S_n=1} \mathrm{sgn}(w_n) \delta_{X_n}(\mathrm{d}x). \tag{19}$$

is a good approximation of the target distribution p.

5. Experiments

We consider two examples. The first example is taken from [5], and is used to compare cube thinning with KSD thinning. The second example illustrates cube thinning when used in conjunction with control variates that are not gradient-based. We also include standard thinning in our comparisons.

Note that there is little point in comparing these methods in terms of CPU cost, as KSD thinning is considerably slower than cube thinning and standard thinning whenever $M \gg 100$. (In one of our experiments, for $M = 1000$, KSD took close to 7 h to run, while cube thinning with all the covariates took about 30 s.) Thus, our comparison will be in terms of statistical error, or, more precisely, in terms of how representative of p is the selected subsample.

In the following (in particular in the plots), "cubeFull" (resp. "cubeDiagonal") will refer to our approach based on the full (resp. diagonal) set of control variates, as discussed in Section 2.3. "NoBurnin" means that burn-in has been discarded manually (hence, no burn-in in the inputs). Finally, "thinning" denotes the usual thinning approach, "SMPCOV", "MED" and "SCLMED" are the same names used in [5] for KSD thinning, based on three different kernels.

To implement the cube method, we used R package `BalancedSampling`.

5.1. Evaluation Criteria

We could compare the three different methods in terms of variance of the estimates of $p(f)$ for certain functions f. However, it is easy to pick functions f that are strongly correlated with the chosen control variates; this would bias the comparison in favour of our approach. In fact, as soon as the target is Gaussian-like, the control variates we chose in Section 2.3 should be strongly correlated with the expectation of any polynomial function of order two, as we discussed in that section.

Rather, we consider criteria that are indicative of the performance of the methods for a general class of function. Specifically, we consider three such criteria. The first one is the kernel Stein discrepency (KSD) as defined in [5] and recalled in the introduction—see (1). Note that this criterion is particularly favourable for KSD thinning, since this approach specifically minimises this quantity. (We use the particular version based on the median kernel in Riabiz et al. [5].)

The second criterion is the energy distance (ED) between p and the empirical distribution defined by the thinning method, e.g., (19) for cube thinning. Recall that the ED between two distributions F and G is:

$$ED(F,G) = 2\mathbb{E}||Z - X||_2 - \mathbb{E}||Z - Z'||_2 - \mathbb{E}||X - X'||_2 \tag{20}$$

where $Z', Z \overset{iid}{\sim} F$ and $X', X \overset{iid}{\sim} G$, and that this quantity is actually a pseudo-distance: $ED(F,G) \geq 0$, $ED(F,G) = 0 \Rightarrow F = G$, $ED(F,G) = ED(G,F)$, but ED does not fulfil the triangle inequality [17,18].

One technical difficulty is that (19) is a signed measure, not a probability measure; see Appendix B on how we dealt with this issue.

Our third criterion is inspired by the star discrepancy, a well-known measure of the uniformity of N points $u_n \in [0,1]^d$ in the context of quasi-Monte Carlo sampling [9] (Chapter 15). Specifically, we consider the quantity

$$d^\star(\hat{P}, \hat{\nu}) = \sup_{B \in \mathcal{B}} \left| \hat{P}_\psi(B) - \hat{\nu}_\psi(B) \right|$$

where $\psi : \mathbb{R}^d \to [0,1]^d$, $\hat{P}_\psi$ and $\hat{\nu}_\psi$ are the push-forward measures associated to empirical distributions $\hat{P} = (N-b)^{-1} \sum_{n=b+1}^{N} \delta_{X_n}(dx)$, and $\hat{\nu}$ as defined in (19), and $\mathcal{B}$ is the set of hyper-rectangles $B = \prod_{i=1}^{d} [0, b_i]$. In practice, we defined function ψ as follows: we apply the linear transform that makes the considered sample to have zero mean and unit variance, and then we applied the inverse CDF (cumulative distribution function) of a unit Gaussian to each component.

Additionally, since the sup above is not tractable, we replace it by a maximum over a finite number of b_i (simulated uniformly).

5.2. Lotka–Volterra Model

This example is taken from [5]. The Lotka–Volterra model describes the evolution of a prey–predator system in a closed environment. We denote the number of prey by u_1 and the number of predators by u_2. The growth rate of the prey is controlled by a parameter $\theta_1 > 0$ and its death rate—due to the interactions with the predators—is controlled by a parameter $\theta_2 > 0$. In the same way, the predator population has a death rate of $\theta_3 > 0$ and a growth rate of $\theta_4 > 0$. Given these parameters, the evolution of the system is described by a system of ODEs:

$$\begin{aligned} \frac{du_1}{dt} &= \theta_1 u_1 - \theta_2 u_1 u_2 \\ \frac{du_2}{dt} &= \theta_4 u_1 u_2 - \theta_3 u_2 \end{aligned}$$

Ref. [5] set $\theta = (\theta_1, \theta_2, \theta_3, \theta_4) = (0.67, 1.33, 1, 1)$, the initial condition $u_0 = (1,1)$, and simulate synthetic data. They assume they observe the populations of prey and predator at times $t_i, i = 1, \ldots, 2400$ where the t_i are taken uniformly on $[0, 25]$ and that these observations are corrupted with a centered Gaussian noise with a covariance matrix $C = \text{diag}(0.2^2, 0.2^2)$. Finally, the model is parametrised in terms of $x = (\log\theta_1, \log\theta_2, \log\theta_3, \log\theta_4) \in \mathbb{R}^4$ and a standard normal distribution as a prior on x is used.

The authors have provided their code as well as the sampled values they obtained by running different MCMC chains for a long time. We use the exact same experimental set-up, and we do not run any MCMC chain on our own, but use the ones they provide instead, specifically the simulated chain, of length 2×10^6, from preconditionned MALA.

We compress this chain into a subsample of size either $M = 100$ or $M = 1000$. For each value of M, we run different variations of our cube method 50 times and make a comparison with the usual thinning method and with the KSD thinning method with different kernels, see [5]. In Figure 1, we show the first 5000 weights of the cube method. We can see that after 1000 iterations, the weights seem to stabilise. Based on visual examination of these weights, we chose a conservative burn-in period of 2000 iterations for the variants where burn-in is removed manually.

We plot the results of the experiment in Figures 2–4.

First, we see that regarding the kernel Stein discrepancy metric, Figure 2, the KSD method performs better than the standard thinning procedure and the cube method. This is not surprising since, even if this method does not properly minimise the Kernel–Stein Discrepency, this is still its target. We also see that, for $M = 1000$, the KSD method performs a bit better than our cube method which in turn performs better than the standard thinning procedure. Note that the relative performance of the KSD method to our cube methods depends on the kernel that is being used and that there is no way to determine which kernel will perform best before running any experiment.

The picture is different for $M = 100$: KSD thinning outperforms standard thinning, which in turn outperforms all of our cube thinning variations. Once again, the fact that the KSD method performs better than any other method seems reasonable: since it regards minimizing the Kernel–Stein Discrepancy, the KSD method is "playing at home" on this metric.

If we look at Figure 4, we see that all of our cube methods outperform the KSD method with any kernel. Interestingly, the standard thinning methods has a similar energy distance as the cube methods with "diagonal" control variates. These observations are true for both $M = 100$ and $M = 1000$. We can also note that the cube method with the full set of control variates tends to perform much better than its "diagonal" counterpart, whatever the value of M.

Finally, looking at Figure 3, it is clear that the KSD method—with any kernel—performs worse than any cube method in terms of star discrepancy.

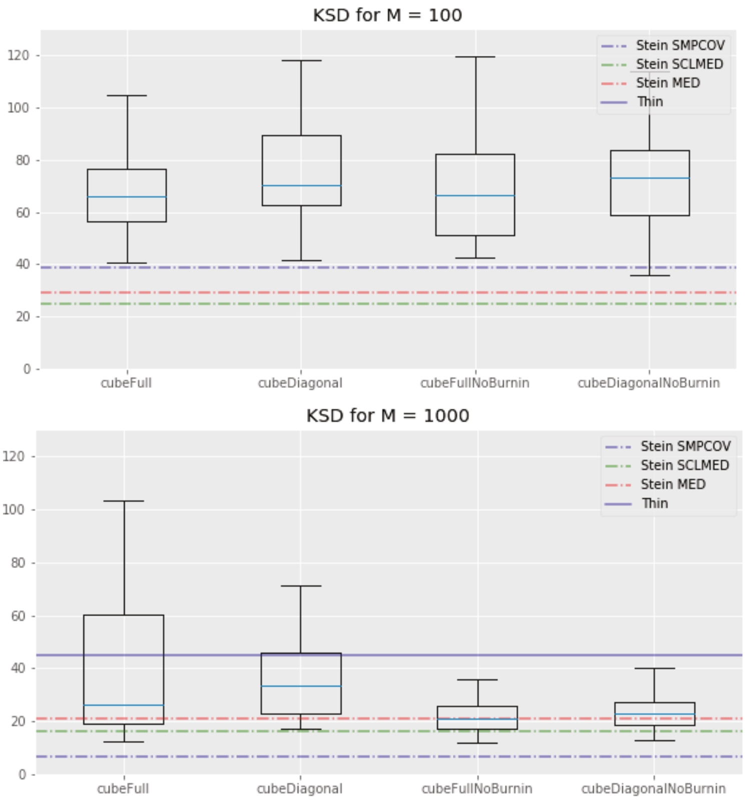
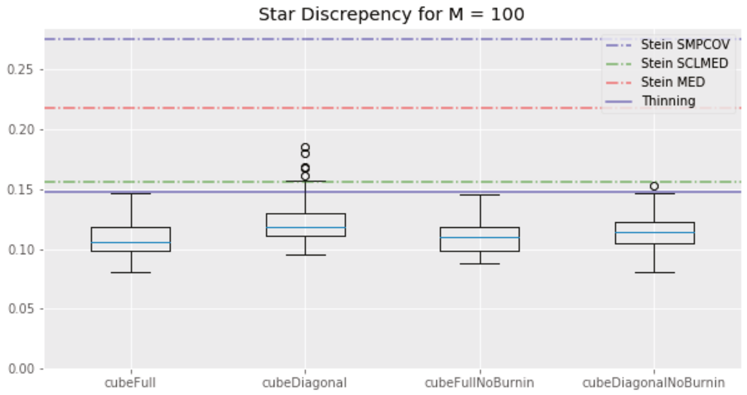


Figure 2. Lotka–Volterra example: box-plots of the kernel Stein discrepency for all the cube method variations, compared with the KSD method for three kernels and the usual thinning method (horizontal lines). **Top**: $M = 100$. **Bottom**: $M = 1000$. (In the top plot, standard thinning is omitted to improve clarity, as corresponding value is too high.)



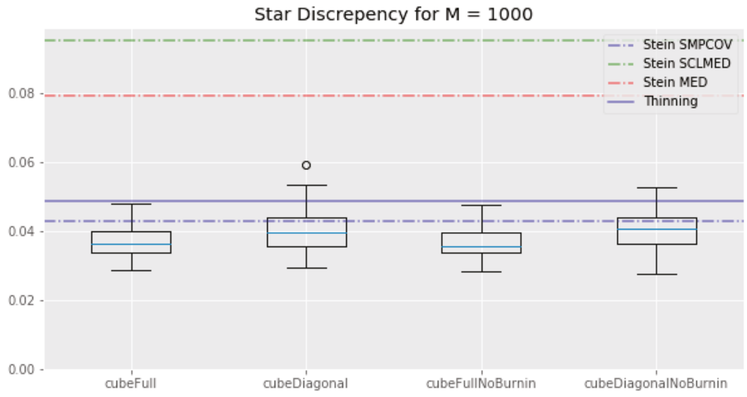


Figure 3. Lotka–Volterra example: box-plots of the star discrepency for all the cube method variations, compared with the KSD method for three kernels and the usual thinning method (horizontal lines). **Top**: $M = 100$. **Bottom**: $M = 1000$.

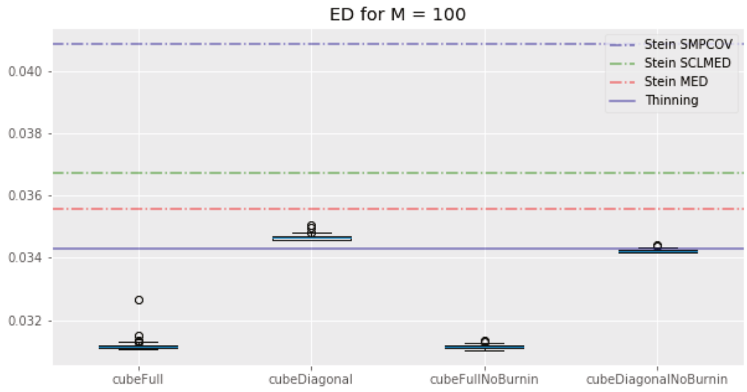


Figure 4. *Cont*.

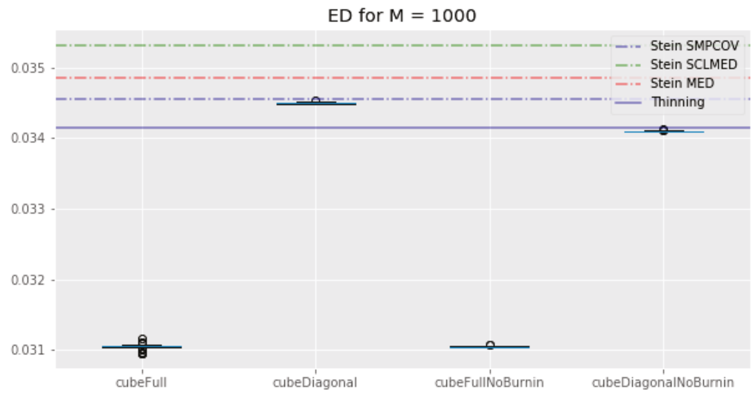


Figure 4. Lotka–Volterra example: boxplots of the energy distance for all the cube method variations, compared with the KSD method for three kernels and the usual thinning method (horizontal lines). **Top**: $M = 100$. **Bottom**: $M = 1000$.

Overall, the relative performance of the cube methods and KSD methods can change a lot depending on the metric being used and the number of points we keep. In addition, while all the cube methods tend to perform roughly the same, this is not the case of the KSD method, whose performances depend on the kernel we use. Unfortunately, we have no way to determine beforehand which kernel will perform best. This is a problem since the KSD method is computationally expensive for subsamples of cardinality $M \gg 100$.

Thus, by and large, cube thinning seems much more convenient to use (both in terms of CPU time and sensitivity to tuning parameters) while offering, roughly, the same level of statistical performance.

5.3. Truncated Normal

In this example, we use the (random-scan version of) the Gibbs sampler of [1] to sample from 10-dimensional multivariate normal truncated to $[0, \infty)^{10}$. We generated the parameters of this truncated normal as follows: the mean was set as the realisation of a 10-dimensional standard normal distribution, while for the covariance matrix Σ, we first generated a matrix $M \in \mathcal{M}_{10,10}(\mathbb{R})$ for which each entry was the realisation of a standard normal distribution. Then, we set $\Sigma = M^T M$.

Since we used a Gibbs sampler, we have access to the Gibbs control variates of [7], based on the expectation of each update (which amounts to simulating from a univariate Gaussian). Thus, we consider 10 control variates.

The Gibbs sampler was run for $N = 10^5$ iterations and no burn-in was performed. We compare the following estimators of the expectation of the target distribution the standard estimator, based on the whole chain ("usualEstim" in the plots), the estimator based on standard thinning ("thinEstim" in the plots), the control variate estimator based on the whole chain, i.e., (7) ("regressionEstim" in the plots), and finally our cube estimator described in Section 4 ("cubeEstim" in the plots). For standard thinning and cube thinning, the thinning sample size was set to $M = 100$, which corresponds to a compression factor of 10^3.

The results are shown in Figure 5. First, we can see that the control variates we chose led to a substantial decrease in the variance of the estimates for regressionEstim compared to usualEstim. Second, the cube estimator performed worse than the regression estimator in terms of variance, but this was expected, as explained in Section 4. More interestingly, if we cannot say that the cube estimator performs better than the usual MCMC estimator in general, we can see that for some components it performed as well or even better, even though the cube estimator used only $M = 100$ points while the usual estimator used 10^5 points. This is largely due to the good choice of the control variates. Finally, the cube estimator outperformed the regular thinning estimator for every component, sometimes significantly.

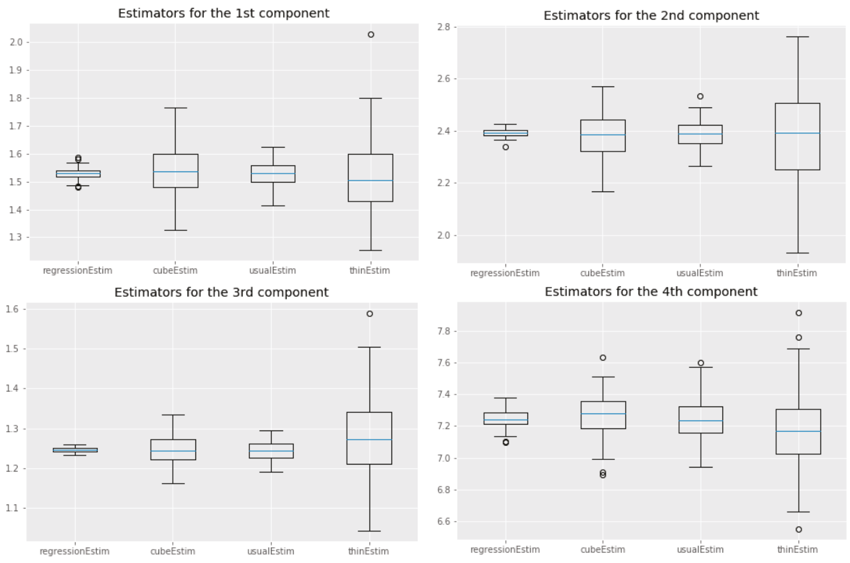


Figure 5. Truncated normal example: box-plots over 100 independent replicates of each estimator; see text for more details.

Author Contributions: Conceptualization, N.C.; Formal analysis, N.C. and G.D.; Investigation, G.D.; Methodology, G.D.; Software, G.D.; Writing—original draft, G.D.; Writing—review and editing, N.C. All authors have read and agreed to the published version of the manuscript.

Funding: The PhD grant of the second author is funded by the French National Research Agency (ANR) contract ANR-17-C23-0002-01 (project B3DCMB).

Data Availability Statement: The data that support the findings of the first numerical experiment are openly available in stein.thinning at https://github.com/wilson-ye-chen/stein.thinning (accessed on 2 August 2021).

Acknowledgments: We are grateful to the editor and the referees for their supportive and useful comments.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Details on the Landing Phase

The landing phase seeks to generate a random vector S in $\{0,1\}^N$, with expectation $\pi^\star$ (the output of the flight phase), which minimises the criterion $\mathrm{tr}(M\mathrm{Var}(\hat{V}|\pi^\star))$ for a certain matrix M. (The notation $\cdot|\pi^\star$ refers to the distribution of S conditional on $\pi(t) = \pi^\star$ at the end of the flight phase.)

Since $\mathrm{Var}(S) = \mathrm{Var}(\mathbb{E}[S|\pi^\star]) + \mathbb{E}[\mathrm{Var}(S|\pi^\star)]$ by the law of total variance, and since the first term is zero (as $\mathbb{E}[S|\pi^\star] = \pi^\star$), we have

$$\mathrm{Var}(\hat{V}) = \mathbb{E}[\mathrm{Var}(\hat{V}|\pi^\star)] = \mathbb{E}[A\mathrm{Var}(S|\pi^\star)A^t]. \tag{A1}$$

and thus:

$$\mathrm{tr}(M\mathrm{Var}(\hat{V}|\pi^\star)) = \sum_{s\in\{0,1\}^N} p(s|\pi^\star)(s-\pi^\star)^t A^t MA(s-\pi^\star). \tag{A2}$$

Choosing $M = (AA^t)^{-1}$, as recommended by [6], amounts to minimising the distance to the hyperplane 'on average'. Let $C(s) = (s - \pi^\star)^t A^t (AA^t)^{-1} A^t (s - \pi^\star)$, then the minimisation program is equivalent to the following linear programming problem over q variables only:

$$\min_{\xi^\star(.)} \sum_{s^\star \in \mathcal{S}^\star} C(s^\star)\xi^\star(s^\star) \tag{A3}$$

with constraints $\sum_{s^\star \in \mathcal{S}^\star} \xi^\star(s^\star) = 1$, $0 \leq \xi^\star(s^\star) \leq 1$, $\sum_{s^\star \in \mathcal{S}^\star | s^\star_k = 1} \xi^\star(s^\star) = \pi^\star_k$ for every $k \in U^\star$ and $\mathcal{S}^\star = \{0,1\}^q$ where $q = \text{card}(U^\star)$ and $U^\star = \{k \in U : 0 < \pi^\star[k] < 1\}$. Here, $\xi^\star$ denotes the marginal distribution of the components $U^\star$ of the sampling design ξ and $C(s^\star)$ must be understood as $C(s)$ with the components of $s \notin U^\star$ being fixed by the result of the flight phase; thus, in this minimisation problem, C is in fact dependent on the components of s that are in $U^\star$ only.

The constraints define a bounded polyhedron. By the fundamental theorem of linear programming, this optimisation problem has at least one solution on a minimal support—see [6].

The flight phase ends on a vertex of $\mathcal{K}$ and, by Proposition 1 in [6], $q \leq J$—typically $J \ll N$. This means that we are solving a linear programming problem in a dimension q potentially much lower than the population size N, and if we do not have too many auxiliary variables, this optimisation problem will not be computationally too expensive. In practice, a simplex algorithm is used to find the solution.

Appendix B. Estimation of the Energy Distance

There are two difficulties with computing (20). First, it involves intractable expectations. Second, as pointed out at the end of Section 4.3, the empirical distribution generated by cube thinning, (19), is actually a signed measure.

Regarding the first issue, we can approximate (20) from our MCMC sample $X_1, \dots, X_N$. That is, if our subsampled empirical measure writes $\hat{\nu} = \sum_{m=1}^{M} w_m \delta_{Z_m}$ and that we approximate the distribution associated with p by $\hat{P} = (N-b)^{-1} \sum_{n=b+1}^{N} \delta_{X_n}$ where $1 \leq b \leq N$ is the burn-in of the chain; then, we can estimate $ED(\hat{\mu}, p)$ with $ED(\hat{\mu}, \hat{P})$.

Regarding the second issue, we can generalize the energy distance to finite measures: suppose we have two finite and potentially signed measures ν_1 and ν_2, both defined on the same measurable space $(\Omega, \mathcal{P}(\Omega)\}$ where $\Omega = \{X_1, \dots, X_N\}$ and $\mathcal{P}(\Omega)$ denote the set of parts of Ω. Suppose, in addition, that $\nu_1(\Omega) = \alpha_1$ and $\nu_2(\Omega) = \alpha_2$ with $\alpha_1 \neq 0$ and $\alpha_2 \neq 0$. We define the generalized energy distance as:

$$\begin{aligned} ED^\star(\nu_1, \nu_2) = & \frac{2}{\alpha_1\alpha_2} \int_\Omega ||x-y||_2 d\nu_1(x) d\nu_2(y) \\ & - \frac{1}{\alpha_1^2} \int_\Omega ||x-x'||_2 d\nu_1(x) d\nu_1(x') \\ & - \frac{1}{\alpha_2^2} \int_\Omega ||y-y'||_2 d\nu_2(y) d\nu_2(y'). \end{aligned}$$

Then, by negative definiteness of the application $\phi(x,y) = ||x-y||_2$ on $\mathbb{R}^N \times \mathbb{R}^N$, $ED^\star(\nu_1, \nu_2) \geq 0$ with equality if and only if $\dfrac{1}{\alpha_1}\nu_1 = \dfrac{1}{\alpha_2}\nu_2$. This means that the generalized energy distance is zero if and only if the two measures are equal up to a non-zero multiplicative constant—see [17] for a demonstration. This generalized energy distance is also symmetric, but the triangle inequality does not hold. It is a pseudo-distance.

Thus, we will use the following criterion, which we will call the energy distance:

$$\begin{aligned} ED^{\star}(\hat{v},\hat{P}) = & \frac{2}{(N-b)\alpha_1}\sum_{k=1}^{N}\sum_{n=b+1}^{N}\frac{\Omega}{M}sgn(w_k)||X_k - X_n||_2\mathbf{1}_{\{S_k=1\}} \\ & - \frac{1}{\alpha_1^2}\sum_{n=1}^{N}\sum_{k=1}^{N}\left(\frac{\Omega}{M}\right)^2 sgn(w_n)sgn(w_k)||Z_k - Z_n||_2\mathbf{1}_{\{S_k=1\}}\mathbf{1}_{\{S_n=1\}} \end{aligned}$$

where $\hat{v}$ is defined in (19) and we dropped the last term because it does not depend on $\hat{v}$ and it is a potentially expensive sum of $(N-b)^2$ terms.

Note that the probability of $\hat{v}(\Omega)$ being zero is non-null and then there is a non-negligible probability of $ED^{\star}(\hat{v},\hat{P})$ being undefined. However, this event is unlikely to happen.

References

1. Robert, C.P.; Casella, G. *Monte Carlo Statistical Methods*; Springer: New York, NY, USA, 2004. [CrossRef]
2. Geyer, C.J. Practical Markov Chain Monte Carlo. *Stat. Sci.* **1992**, *7*, 473–483. [CrossRef]
3. Cowles, M.K.; Carlin, B.P. Markov chain Monte Carlo convergence diagnostics: A comparative review. *J. Am. Statist. Assoc.* **1996**, *91*, 883–904. [CrossRef]
4. Mak, S.; Joseph, V.R. Support points. *Ann. Stat.* **2018**, *46*, 2562–2592. [CrossRef]
5. Riabiz, M.; Chen, W.; Cockayne, J.; Swietach, P.; Niederer, S.A.; Mackey, L.; Oates, C.J. Optimal Thinning of MCMC Output. *arXiv* **2020**, arXiv:2005.03952.
6. Deville, J.C. Efficient balanced sampling: The cube method. *Biometrika* **2004**, *91*, 893–912. [CrossRef]
7. Dellaportas, P.; Kontoyiannis, I. Control variates for estimation based on reversible Markov chain Monte Carlo samplers. *J. R. Stat. Soc. Ser. B (Stat. Methodol.)* **2011**, *74*, 133–161. [CrossRef]
8. Glasserman, P. *Monte Carlo Methods in Financial Engineering*; Springer: New York, NY, USA, 2004; Volume 53, pp. xiv+596.
9. Owen, A.B. *Monte Carlo Theory, Methods and Examples*; in progress, 2013. Available online: https://statweb.stanford.edu/~{}owen/mc/ (accessed on 2 August 2021).
10. Oates, C.J.; Girolami, M.; Chopin, N. Control functionals for Monte Carlo integration. *J. R. Stat. Soc. Ser. B (Stat. Methodol.)* **2016**, *79*, 695–718. [CrossRef]
11. Hammer, H.; Tjelmeland, H. Control variates for the Metropolis-Hastings algorithm. *Scand. J. Stat.* **2008**, *35*, 400–414. [CrossRef]
12. Mijatović, A.; Vogrinc, J. On the Poisson equation for Metropolis-Hastings chains. *Bernoulli* **2018**, *24*, 2401–2428. [CrossRef]
13. Chauvet, G.; Tillé, Y. A fast algorithm for balanced sampling. *Comput. Stat.* **2006**, *21*, 53–62. [CrossRef]
14. Brosse, N.; Durmus, A.; Meyn, S.; Moulines, E.; Radhakrishnan, A. Diffusion approximations and control variates for MCMC. *arXiv* **2019**, arXiv:1808.01665.
15. Brooks, S.; Gelman, A. Some issues for monitoring convergence of iterative simulations. *Comput. Sci. Stat.* **1998**, *1998*, 30–36.
16. Székely, G.J.; Rizzo, M.L. A new test for multivariate normality. *J. Multivar. Anal.* **2005**, *93*, 58–80. [CrossRef]
17. Székely, G.J.; Rizzo, M.L. A new test for multivariate normality. *J. Multivar. Anal.* **2005**, *93*, 58–80. [CrossRef]
18. Klebanov, L.B. *N-Distances and Their Applications*; The Karolinum Press, Charles University: Prague, Czech Republic, 2006.

Article

Accelerated Diffusion-Based Sampling by the Non-Reversible Dynamics with Skew-Symmetric Matrices

Futoshi Futami [1,*], Tomoharu Iwata [1], Naonori Ueda [1] and Issei Sato [2]

[1] Communication Science Laboratories, NTT, Hikaridai, Seika-cho, "Keihanna Science City", Kyoto 619-0237, Japan; tomoharu.iwata.gy@hco.ntt.co.jp (T.I.); naonori.ueda.fr@hco.ntt.co.jp (N.U.)
[2] Department of Computer Science, Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan; sato@g.ecc.u-tokyo.ac.jp
[*] Correspondence: futoshi.futami.uk@hco.ntt.co.jp

Abstract: Langevin dynamics (LD) has been extensively studied theoretically and practically as a basic sampling technique. Recently, the incorporation of non-reversible dynamics into LD is attracting attention because it accelerates the mixing speed of LD. Popular choices for non-reversible dynamics include underdamped Langevin dynamics (ULD), which uses second-order dynamics and perturbations with skew-symmetric matrices. Although ULD has been widely used in practice, the application of skew acceleration is limited although it is expected to show superior performance theoretically. Current work lacks a theoretical understanding of issues that are important to practitioners, including the selection criteria for skew-symmetric matrices, quantitative evaluations of acceleration, and the large memory cost of storing skew matrices. In this study, we theoretically and numerically clarify these problems by analyzing acceleration focusing on how the skew-symmetric matrix perturbs the Hessian matrix of potential functions. We also present a practical algorithm that accelerates the standard LD and ULD, which uses novel memory-efficient skew-symmetric matrices under parallel-chain Monte Carlo settings.

Keywords: Markov Chain Monte Carlo; Langevin dynamics; Hamilton Monte Carlo; non-reversible dynamics

Citation: Futami, F.; Iwata, T.; Ueda, N.; Sato, I. Accelerated Diffusion-Based Sampling by the Non-Reversible Dynamics with Skew-Symmetric Matrices. *Entropy* **2021**, *23*, 993. https://doi.org/10.3390/e23080993

Academic Editor: Pierre Alquier

Received: 21 June 2021
Accepted: 27 July 2021
Published: 30 July 2021

1. Introduction

Sampling is one of the most widely used techniques for the approximation of posterior distribution in Bayesian inference [1]. Markov Chain Monte Carlo (MCMC) is widely used to obtain samples. In MCMC, Langevin dynamics (LD) is a popular choice for sampling from high-dimensional distributions. Each sample in LD moves toward a gradient direction with added Gaussian noise. LD efficiently explore around a mode of a target distribution using the gradient information without being trapped by local minima thanks to added Gaussian noise. Many previous studies theoretically and numerically proved LD's superior performance [2–5]. Since non-reversible dynamics generally improves mixing performance [6,7], research on introducing non-reversible dynamics to LD for better sampling performance is attracting attention [8].

There are two widely known non-reversible dynamics for LD. One is underdamped Langevin dynamics (ULD) [9], which uses second-order dynamics. The other introduces perturbation, which consists of multiplying the skew-symmetric matrix by a gradient [8]. Here, we refer to the matrix as skew matrices for simplicity and this perturbation technique as skew acceleration. Much research has been done on ULD theoretically [9–11] and ULD is widely used in practice, which is also known as stochastic gradient Hamilton Monte Carlo [12]. In contrast, the application of the skew acceleration for standard Bayesian models is quite limited even though it is expected to show superior performance theoretically [8].

For example, skew acceleration has been analyzed focusing on sampling from Gaussian distributions [13–17], although assuming Gaussian distributions in Bayesian models is restrictive in practice. A recent study [8] theoretically showed that skew acceleration accelerates the dynamics around the local minima and saddle points for non-convex functions. Another work [18] clarified that the skew acceleration theoretically and numerically improves mixing speed when used as interactions between chains in parallel sampling schemes for non-convex Bayesian models.

Compared to ULD, what seems to be lacking for skew acceleration is a theoretical understanding of issues that are important to practitioners. The most significant problem is that no theory exists for selecting skew matrices. In existing studies, introducing a skew matrix into LD results in equal or faster convergence, denoting that a bad choice of skew matrix results in no acceleration. Thus, choosing appropriate skew matrices is critical. Furthermore, although ULD's acceleration has been analyzed quantitatively, existing studies have only analyzed skew acceleration qualitatively. Thus, it is difficult to justify the usefulness of skew acceleration in practice compared to ULD. Another issue is that introducing skew matrices requires a vast memory cost in many practical Bayesian models.

The purpose of this study is to solve these problems from theoretical and numerical viewpoints and establish a practical algorithm for skew acceleration. The following are the two major contributions of this work.

Our contribution 1: We present a convergence analysis of skew acceleration for standard Bayesian model settings, including non-convex potential functions using Poincaré constants [19]. The major advantage of Poincaré constants is that we can analyze skew acceleration through a Hessian matrix and its eigenvalues and develop a practical theory about the selection of J and the quantitative assessment of skew acceleration.

Furthermore, we propose skew acceleration for ULD and present convergence analysis for the first time. Since ULD shows faster convergence than LD, combining skew acceleration with ULD is promising.

Our contribution 2: We develop a practical skew accelerated sampling algorithm for a parallel sampling setting with novel memory-efficient skew matrices. Since a naive implementation of skew acceleration requires a large memory cost to store skew matrices, memory-efficiency is critical in practice. We also present a non-asymptotic theoretical analysis for our algorithm in both LD and ULD settings under a stochastic gradient and Euler discretization. We clarify that introducing skew matrices accelerates the convergence of continuous dynamics, although it increases the discretization and stochastic gradient error. Then to the best of our knowledge, we propose the first algorithm that adaptively controls this trade-off using the empirical distribution of the parallel sampling scheme.

Finally, we verify our algorithm and theory in practical Bayesian problems and compare it with other sampling methods.

Notations: I_d denotes a $d \times d$ identity matrix. Capital letters such as X represent random variables, and lowercase letters such as x represent non-random real values. $\cdot$, $\|\cdot\|$ and $|\cdot|$ denote Euclidean inner products, distances and absolute values.

2. Preliminaries

In this section, we briefly introduce the basic settings of LD and non-reversible dynamics for the posterior distribution sampling in Bayesian inference.

2.1. LD and Stochastic Gradient LD

First, we introduce the notations and the basic settings of LD and stochastic gradient LD (SGLD), which is a practical extension of LD. Here z_i denotes a data point in space $\mathbb{Z}$, $|\mathcal{Z}|$ denotes the total number of data points, and $x \in \mathbb{R}^d$ corresponds to the parameters of a given model, which we want to sample. Our goal is to sample from the target distribution with density $d\pi(x) \propto e^{-\beta U(x)}dx$, where potential function $U(x)$ is the summation of $u : \mathbb{R}^d \times \mathbb{Z} \to \mathbb{R}$, i.e., $U(x) = \frac{1}{|\mathcal{Z}|}\sum_{i=1}^{|\mathcal{Z}|} u(x, z_i)$. Function $u(\cdot,\cdot)$ is continuous and non-convex. The explicit assumptions made for it are discussed in Section 3.1. The SGLD algorithm [2,3] is given as a recursion:

$$X_{k+1} = X_k - h\nabla\hat{U}(X_k) + \sqrt{2h\beta^{-1}}\epsilon_k, \tag{1}$$

where $h \in \mathbb{R}^+$ is a step size, $\epsilon_k \in \mathbb{R}^d$ is a standard Gaussian random vector, β is a temperature parameter of π, and $\nabla\hat{U}(X_k)$ is a conditionally unbiased estimator of true gradient $\nabla U(X_k)$. This unbiased estimate of the true gradient is suitable for large-scale data set since we can use not the full gradient, but a stochastic version obtained through a randomly chosen subset of data at each time step. This means that we can reduce the computational cost to calculate the gradient at each time step.

The discrete time Markov process in Equation (1) is the discretization of the continuous-time LD [2]:

$$dX_t = -\nabla U(X_t)dt + \sqrt{2\beta^{-1}}dw_t, \tag{2}$$

where w_t denotes the standard Brownian motion in $\mathbb{R}^d$. The stationary measure of Equation (2) is $d\pi(x) \propto e^{-\beta U(x)}dx$.

2.2. Poincaré Inequality and Convergence Speed

In sampling, we are interested in the convergence speed to the stationary measure. The speed is often characterized by the *the generator* associated with Equation (2) and defined as:

$$\begin{aligned}\mathcal{L}f(X_t) :&= \lim_{s\to 0^+} \frac{\mathbb{E}(f(X_{t+s})|X_t) - f(X_t)}{s} \\ &= \Big(-\nabla U(X_t)\cdot\nabla + \beta^{-1}\Delta\Big) f(X_t),\end{aligned} \tag{3}$$

where Δ denotes a standard Laplacian on $\mathbb{R}^d$ and $f \in \mathcal{D}(\mathcal{L})$ and $\mathcal{D}(\mathcal{L}) \subset \mathrm{L}^2(\pi)$ denote the $\mathcal{L}$ domain. This $-\mathcal{L}$ is a self-adjoint operator, which has only discrete spectrums (eigenvalues). π with $\mathcal{L}$ has a *spectral gap* if the smallest eigenvalue of $-\mathcal{L}$ (other than 0) is positive. We refer to it as ρ_0(>0). This spectral gap is closely related to Poincaré inequality. Internal energy is defined:

$$\mathcal{E}(f) := -\int_{\mathbb{R}^d} f\mathcal{L}f d\pi. \tag{4}$$

Please note that $\mathcal{E}(f) > 0$ is satisfied. Then π with $\mathcal{L}$ satisfies the Poincaré inequality with constant c, if for any $f \in \mathcal{D}(\mathcal{L})$, π with $\mathcal{L}$ satisfies:

$$\int f^2 d\pi - \left(\int f d\pi\right)^2 \leq c\mathcal{E}(f). \tag{5}$$

The spectral gap characterizes this constant $c \leq \frac{1}{\rho_0}$, which holds (see Appendix A.2 for details). We refer to best constant c as the Poincaré constant [19]. For notational simplicity, we define $m_0 := \frac{1}{c}$ and refer to this m_0 as the Poincaré constant.

In sampling, crucially, Poincaré inequality dominates the convergence speed in χ^2 divergence:

$$\int \left(\frac{d\mu_t}{d\pi} - 1\right)^2 d\pi := \chi^2(\mu_t\|\pi) \leq e^{-\frac{2m_0}{\beta}t}\chi^2(\mu_0\|\pi), \tag{6}$$

where μ_t denotes the measure at time t induced by Equation (2) and μ_0 is the initial measure (see Appendix A.3 for details). Thus, the larger Poincaré constant m_0 is, the faster convergence we have.

2.3. Non-Reversible Dynamics

In this section, we introduce the non-reversible dynamics. π with $\mathcal{L}$ is reversible if for any test function $f, g \in \mathcal{D}(\mathcal{L})$, π with $\mathcal{L}$ satisfies

$$\int_{\mathbb{R}^d} f\mathcal{L}g d\pi = \int_{\mathbb{R}^d} g\mathcal{L}f d\pi. \tag{7}$$

If this is not satisfied, π with $\mathcal{L}$ is non-reversible [19].

We introduce two non-reversible dynamics for LD. The first is ULD, which is given as

$$\begin{aligned} dX_t &= \Sigma^{-1}V_t dt, \\ dV_t &= -\nabla U(X_t)dt - \gamma\Sigma^{-1}V_t dt + \sqrt{2\gamma\beta^{-1}}dw_t, \end{aligned} \tag{8}$$

where $V \in \mathbb{R}^d$ is an auxiliary random variable, $\gamma \in \mathbb{R}$ is a positive constant, and Σ is the variance of the stationary distribution of auxiliary random variable V. The stationary distribution is $\tilde{\pi} := \pi \otimes \mathcal{N}(0, \Sigma) \propto e^{-\beta U(x) - \frac{1}{2}\Sigma^{-1}\|v\|^2}$, where $\mathcal{N}$ denotes a Gaussian distribution. The superior performance of ULD compared with LD has been studied rigorously [9–11]. ULD's convergence speed is also characterized by the Poincaré constant [20]. In practice, we use discretization and the stochastic gradient for ULD, which is called the stochastic gradient Hamilton Monte Carlo (SGHMC) [10]. The second non-reversible dynamics is the skew acceleration given as

$$dX_t = -(I + \alpha J)\nabla U(X_t)dt + \sqrt{2\beta^{-1}}dw_t, \tag{9}$$

where J is a real value skew matrix and $\alpha \in \mathbb{R}^+$ is a positive constant. We call this dynamics S-LD. The stationary distribution of S-LD is still π, and S-LD shows faster convergence and smaller asymptotic variance [13–15,18].

3. Theoretical Analysis of Skew Acceleration

In this section, we present a theoretical analysis of skew acceleration in LD and ULD in standard Bayesian settings. We analyze acceleration through the Poincaré constant and connect it with the eigenvalues of the Hessian matrix, which allows us to obtain a practical criterion to choose skew matrices and quantitatively evaluate acceleration. We focus on a setting where a continuous SDE and a full gradient of the potential function is used in this section. The discretized SDE and stochastic gradient are discussed in Section 4.

3.1. Acceleration Characterization by the Poincaré Constant

First, we introduce the same four assumptions as a previous work [2], which showed the existence of the Poincaré constant about m_0 for LD (see Appendix C for details).

Assumption 1. *(Upper bound of the potential function at the origin) Function u takes nonnegative real values and is twice continuously differentiable on $\mathbb{R}^d$, and constants A and B exist such that for all $z \in \mathbb{Z}$,*

$$|u(0, z)| \leq A, \quad \|\nabla u(0, z)\| \leq B. \tag{10}$$

Assumption 2. *(Smoothness) Function u has Lipschitz continuous gradients; for all $z \in \mathbb{Z}$, positive constant M exists for all $x, y \in \mathbb{R}^d$,*

$$\|\nabla u(x, z) - \nabla u(y, z)\| \leq M\|x - y\|. \tag{11}$$

Assumption 3. *(Dissipative condition) Function u satisfies the (m,b)-dissipative condition for all $z \in \mathbb{Z}$; for all $x \in \mathbb{R}^d$, $m > 0$ and $b \geq 0$ exist such that*

$$-x \cdot \nabla u(x, z) \leq -m\|x\|^2 + b. \tag{12}$$

Assumption 4. *(Initial condition) Initial probability distribution μ_0 of X_0 has a bounded and strictly positive density p_0, and for all $x \in \mathbb{R}^d$,*

$$\kappa_0 := \log \int_{\mathbb{R}^d} e^{\|x\|^2} p_0(x) dx < \infty. \tag{13}$$

Please note that these assumptions allow us to consider the non-convex potential functions, which are common in practical Bayesian models. Furthermore, we make the following assumption about J.

Assumption 5. *The operator norm of J is bounded:*

$$\|J\|_2 \leq 1. \tag{14}$$

This means that the largest singular value of J is below 1.

Under these assumptions, we present the convergence behavior of skew acceleration using the Poincaré constant. First, we present the following S-LD result.

Theorem 1. *Under Assumptions 1–5, the S-LD of Equation (9) has exponential convergence,*

$$\chi^2(\mu_t^\alpha \| \pi) \leq e^{-\frac{2m(\alpha)}{\beta} t} \chi^2(\mu_0 \| \pi), \tag{15}$$

where μ_t^α is the measure at time t induced by S-LD and $m(\alpha)$ is the Poincaré constant of S-LD defined by its generator

$$\mathcal{L}_\alpha f(x) := \Big(-(I + \alpha J) \nabla U(x) \cdot \nabla + \beta^{-1} \Delta \Big) f(x). \tag{16}$$

Furthermore, $m(\alpha)$ satisfies $m(\alpha) \geq m_0$.

The proof is shown in Appendix C. This theorem states that introducing the skew matrices accelerates the convergence of LD by improving the convergence rate from m_0 to $m(\alpha)$. Although [18] obtained a similar result, we used the Poincaré constant and derived an explicit criterion when $m(\alpha) = m_0$ holds, as we discuss below.

Next, we also introduce skew acceleration in ULD. Since ULD shows faster convergence than LD in standard Bayesian settings [10,11], it is promising to combine skew acceleration with ULD to obtain a more efficient sampling algorithm. For that purpose, we propose the following SDE:

$$dX_t = \Sigma^{-1} V_t dt + \alpha_1 J_1 \nabla U(X_t) dt, \tag{17}$$

$$dV_t = -\nabla U(X_t) dt - \gamma(\Sigma^{-1} + \alpha_2 J_2) V_t dt + \sqrt{2\gamma\beta^{-1}} dw_t, \tag{18}$$

where J_1 and J_2 are real value skew matrices and α_1 and α_2 are positive constants. We assume that J_1 and J_2 satisfy Assumption 5. We refer to this method as skew underdamped Langevin dynamics (S-ULD) whose stationary distribution is $\tilde{\pi} = \pi \otimes \mathcal{N}(0, \Sigma) \propto e^{-\beta U(x) - \frac{1}{2}\Sigma^{-1}\|v\|^2}$. See Appendix B for details, which include discussions on other combinations of skew matrices. As for S-ULD, we need an additional assumption about the initial condition of V_0:

Assumption 6. *(Initial condition) Initial probability distribution $\mu_0(x,v)$ of (X_0, V_0) has a bounded and strictly positive density p_0 that satisfies,*

$$\kappa_0 := \log \int_{\mathbb{R}^{2d}} e^{\|x\|^2 + \|v\|^2} p_0((x,v))dxdv < \infty. \tag{19}$$

We then provide the following convergence theorem that resembles S-LD.

Theorem 2. *Under Assumptions 1–3, 5, 6, S-ULD has exponential convergence in χ^2 divergence and its convergence rate is also characterized by $m(\alpha)$ as defined in Theorem 1. S-ULD's convergence equals or exceeds ULD, of which convergence rate is characterized by m_0.*

See Appendix C.2 for details. From these theorems, we confirmed that skew acceleration is effective in both S-LD and S-ULD, and the convergence speed is characterized by Poincaré constant $m(\alpha)$ defined by Equation (16).

3.2. *Skew Acceleration from the Hessian Matrix*

Our goal is to clarify what choices of J induce $m(\alpha) > m_0$, which leads to acceleration. Therefore, we discuss how Poincaré constant $m(\alpha)$ is connected to the eigenvalues and eigenvectors of the perturbed Hessian matrix $(I + \alpha J)\nabla^2 U(x)$. Next, we introduce the notations. We express the Hessian of $U(x)$ as $H(x)$ and the perturbed Hessian matrix as $H'(x) := (I + \alpha J)H(x)$. Please note that H is a real symmetric matrix, which has real eigenvalues and diagonalizable. On the other hand, since H' is not symmetric, it has complex eigenvalues, although diagonalization is not assured (see Appendix E). We express pairs of eigenvectors and eigenvalues of $H'(x)$ as $\{(v_i^\alpha(x), \lambda_i^\alpha(x))\}_{i=1}^d$, which are ordered as $\mathrm{Re}(\lambda_1^\alpha(x))) \leq \cdots \leq \mathrm{Re}(\lambda_d^\alpha(x))$. Here, $\mathrm{Re}(\lambda_1^\alpha(x))$ expresses the real part of complex value λ_1^α and Im denotes the imaginary part. We express those of $H(x)$ as $\{(v_i^0(x), \lambda_i^0(x))\}_{i=1}^d$ and order them as $\lambda_1^0(x) \leq \cdots \leq \lambda_d^0(x)$.

3.2.1. Strongly Convex Potential Function

Assume that U is an m-strongly convex function, where for all $x \in \mathbb{R}^d$, $m \leq \lambda_1^0(x)$ holds. Poincaré constant m_0 of LD satisfies $m_0 = m$ [19]. For the skew acceleration, since Poincaré constant satisfies $m(\alpha) = m'(\alpha)$, where $m'(\alpha)$ is the best constant that satisfies, for all x, $m'(\alpha) \leq \mathrm{Re}(\lambda_1^\alpha(x))$ (see Appendix D.1). Therefore, studying the Poincaré constant is equivalent to studying the smallest (real part of the) eigenvalue of the Hessian matrix. Thus, the relation between $\lambda_1^0(x)$ and $\mathrm{Re}(\lambda_1^\alpha(x))$ must be studied. The following theorem describes how the skew matrices change the smallest eigenvalue.

Theorem 3. *For all $x \in \mathbb{R}^d$, the real parts of the eigenvalues of H' satisfy*

$$m \leq \lambda_1^0(x) \leq \mathrm{Re}(\lambda_1^\alpha(x)) \leq \cdots \leq \mathrm{Re}(\lambda_d^\alpha(x)) \leq \lambda_d^0(x). \tag{20}$$

The condition of $\lambda_1^0(x) = \mathrm{Re}(\lambda_1^\alpha(x))$ is shown in Remark 1.

Remark 1. *Denote the set of the eigenvectors of eigenvalue $\lambda_1^0(x)$ as V_1^0. If $V_1^0 = \{v\}$ and $Jv = 0$, then $\lambda_1^0(x) = \mathrm{Re}(\lambda_1^\alpha(x))$ holds. If the cardinality of set V_1^0 is larger than 1, and vectors $v, v' \in V_1^0$ exist, such that $\lambda_1^0 \alpha J v = (\mathrm{Im}(\lambda_1^\alpha))v'$ and $\lambda_1^0 \alpha J v' = -(\mathrm{Im}(\lambda_1^\alpha))v$, then $\lambda_1^0(x) = \mathrm{Re}(\lambda_1^\alpha(x))$ holds.*

Refer to Appendix F for the proof. This is an extension of previous work [8,13]. If $\lambda_1^0(x) < \mathrm{Re}(\lambda_1^\alpha(x))$ is satisfied for all x, we have $m_0 < m(\alpha)$, i.e., acceleration occurs. We discuss how to construct J such that $\lambda_1^0(x) < \mathrm{Re}(\lambda_1^\alpha(x))$ holds in Section 3.3.

3.2.2. Non-Convex Potential Function

The previous work [21] clarified that the Poincaré constant of the non-convex function is characterized by the negative eigenvalue of the saddle point. As shown in Figure 1, denote x_1 as the global minima, and x_2 is the local minima which has the second smallest value in $U(x)$. We express the saddle point with index one, i.e., there is only one negative eigenvalue at the point, between x_1 and x_2 as x^*. This means that the eigenvalues of $H(x^*)$ satisfies $\lambda_1^0(x^*) < 0 < \lambda_2^0(x^*) < \cdots < \lambda_d^0(x^*)$. Ref. [21] clarified that the saddle point x^* characterizes the Poincaré constant as

$$m_0^{-1} \propto \frac{1}{|\lambda_1(x^*)|} e^{\beta(U(x^*)-U(x_1)-U(x_2))}. \tag{21}$$

When skew matrices are introduced, [8] clarified the following relation:

Theorem 4. *([8]) $\lambda_1^\alpha(x^*) \leq \lambda_1^0(x^*) < 0$ and equality holds only if $Jv_1^\alpha(x^*) = 0$.*

Note $\lambda_1^\alpha(x^*)$ is not a complex number. Thus, the skew acceleration reduces the negative eigenvalue and leads to a larger Poincaré constant (see Appendix D.2) and results in faster convergence.

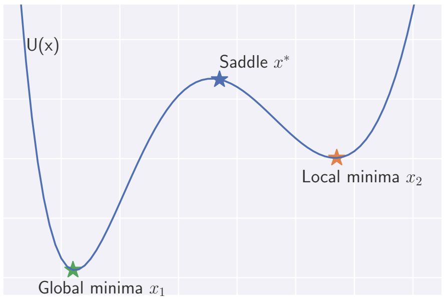


Figure 1. Double-potential example: Poincaré constant is related to the eigenvalue at x^*.

In conclusion, introducing the skew matrix changes the Hessian's eigenvalues and increase the Poincaré constant. If $\lambda_1^0(x) \neq \mathrm{Re}(\lambda_1^\alpha(x))$ is satisfied, this leads to faster convergence for both convex and non-convex potential functions.

3.3. Choosing J

In this section, we present a method for choosing J that leads to $\lambda_1^0(x) \neq \mathrm{Re}(\lambda_1^\alpha(x))$ to ensure the acceleration based on the equality conditions in Theorems 3 and 4. Combining these theorems, we obtain the following criterion:

Remark 2. *Given a point x, $\lambda_1^0(x) \neq \mathrm{Re}(\lambda_1^\alpha(x))$ holds if either the following conditions are satisfied: (i) when $V_1^0 = \{v\}$, $Jv \neq 0$ is satisfied. (ii) when $|V_1^0| > 1$, $Jv \neq 0$ holds for any $v \in V_1^0$, and for any $v, v' \in V_1^0$, $\lambda_1^0 \alpha Jv = (\mathrm{Im}(\lambda_1^\alpha))v'$ and $\lambda_1^0 \alpha Jv' = -(\mathrm{Im}(\lambda_1^\alpha))v$ are not satisfied.*

The first condition (i) is easily satisfied if we choose J such that $\mathrm{Ker}J = \{0\}$. On the other hand, the second condition (ii) is difficult to verify since H and its eigenvalues and eigenvectors generally depend on the current position of X_t. Instead of evaluating eigenvalues and eigenvectors of H and H' directly, we use the random matrix property shown in the next theorem.

Theorem 5. *Suppose the upper triangular entries of J follow a probability distribution that is absolutely continuous with respect to the Lebesgue measure. If* $\mathrm{Ker}J = \{0\}$ *is satisfied, then given a point* $x \in \mathbb{R}^d$, $\lambda_1^0(x) \neq \mathrm{Re}\big(\lambda_1^\alpha(x)\big)$ *holds with probability 1.*

The proof is given in Appendix G.1. From this theorem, we simply generate J from some probability distribution, such as the Gaussian distribution. Then, we check whether $\mathrm{Ker}J = \{0\}$ holds. If $\mathrm{Ker}J = \{0\}$ does not hold, we generate a random matrix J again.

The above theorem is valid only at a given evaluation point x. We can extend the above theorem to all the points over the path of the discretized dynamics (see Appendix G.3). With this procedure, we can theoretically ensure that acceleration occurs with probability one for discretized dynamics.

3.4. Qualitative Evaluation of The Acceleration

So far, we have discussed skew acceleration qualitatively but not quantitatively. Although acceleration's quantitative evaluation is critical for practical purposes, to the best of our knowledge, no existing work has addressed it. In this section, we present a formula that quantitatively assesses skew acceleration by analyzing the eigenvalues of the Hessian matrix.

Theorem 6. *With the identical notation as in Theorem 3, for all x, we have*

$$\mathrm{Re}(\lambda_1^\alpha(x)) = \lambda_1^0(x) + \alpha^2 \sum_{k=2}^{d} \frac{\lambda_1^0(x)\lambda_k^0(x)|v_k^0(x)Jv_1^0(x)|^2}{\lambda_k^0(x) - \lambda_1^0(x)} + \mathcal{O}(\alpha^3). \tag{22}$$

In particular, at saddle point x^*, we have

$$\lambda_1^\alpha(x^*) = \lambda_1^0(x^*) + \alpha^2 \sum_{k=2}^{d} \frac{\lambda_1^0(x^*)\lambda_k^0(x^*)|v_k^0(x^*)Jv_1^0(x^*)|^2}{\lambda_k^0(x^*) - \lambda_1^0(x^*)} + \mathcal{O}(\alpha^3). \tag{23}$$

The proofs are shown in Appendix H. When focusing on Equation (22), if $U(x)$ is a strongly convex function, since for all $k > 1$, $\lambda_k(x) > \lambda_1(x) > 0$ holds and the second term in Equation (22) is positive. From this, $\mathrm{Re}\big(\lambda_1^\alpha(x)\big) > \lambda_1^0(x)$ holds. A similar relation holds for $\mathrm{Re}(\lambda_d^\alpha(x))$. In Equation (23), $\lambda_1^\alpha(x^*) < \lambda_1^0(x^*) < 0$ holds. Thus, the changes of the Poincaré constants are proportional to α^2. With these formulas, we can quantitatively evaluate the acceleration. We present numerical experiments to confirm our theoretical findings in Section 6.1.

4. Practical Algorithm for Skew Acceleration

In this section, we discuss skew acceleration in more practical settings compared to Section 3. First, we discuss the memory issue for storing J and the discretization of SDE and the stochastic gradient, which are widely used techniques in Bayesian inference. Finally, we present a practical algorithm for skew acceleration.

4.1. Memory Issue of Skew Acceleration and Ensemble Sampling

For d-dimensional Bayesian models, we need $\mathcal{O}(d^2)$ memory space to store skew matrices Js, and this is difficult for high-dimensional models. Instead of storing J, we can randomly generate Js at each time step following Theorem 5. However, we experimentally confirmed that using different Js at each step does not accelerate the convergence (see Section 6). Thus, we need to use a fixed J during the iterations.

As discussed below, we found that the previously proposed accelerated parallel sampling [18] can be a practical algorithm to resolve this memory issue. In that method, we simultaneously updated N samples of the model's parameters with correlation. In such a parallel sampling scheme, a correlation exists among multiple Markov chains, it is more efficient than a naive parallel-chain MCMC, where the samples are independent.

We express the n-th sample at time t as $X_t^{(n)} \in \mathbb{R}^d$ and the joint state of all samples at time t as $X_t^{\otimes N} := (X_t^{(1)}, \dots, X_t^{(N)})^\top \in \mathbb{R}^{dN}$. We express the joint stationary measure as $\pi^{\otimes N} := \pi \otimes \cdots \otimes \pi(x^{\otimes N}) \propto e^{-\beta \sum_{i=1}^N U(x^{(i)})}$. We express the sum of the potential function as $U^{\otimes N} := \sum_{i=1}^N U(x^{(i)})$. We then consider the following dynamics:

$$dX_t^{\otimes N} = -(I_{dN} + \alpha J)\nabla U^{\otimes N}(X_t^{\otimes N})dt + \sqrt{2\beta^{-1}}dw_t, \tag{24}$$

$$\nabla U^{\otimes N}(X_t^{\otimes N}) := \left(\nabla U(X_t^{(1)}), \dots, \nabla U(X_t^{(N)})\right)^\top. \tag{25}$$

We call this dynamics skew parallel LD (S-PLD). N-independent parallel LD (PLD) is coupled with the skew matrix. Since each chain in PLD is independent of the other, the Poincaré constant of PLD is also m_0. Ref. [18] argued that the Poincaré constant of S-PLD, $m(\alpha, N)$, satisfies $m(\alpha, N) \geq m_0$. This means S-PLD shows faster convergence than PLD. As discussed in Section 3.2, these Poincaré constants are characterized by the smallest eigenvalue of the Hessian matrix $\nabla^2 U^{\otimes N}(x^{\otimes N})$ and $(I_{dN} + \alpha J)\nabla^2 U^{\otimes N}(x^{\otimes N})$ where $x^{\otimes N} \in \mathbb{R}^{dN}$. We denote these smallest eigenvalues as $\lambda_1^0(x^{\otimes N})$ and $\mathrm{Re}\lambda_1^\alpha(x^{\otimes N})$. As discussed in Section 3.2, acceleration occurs if $\lambda_1^0(x^{\otimes N}) \neq \mathrm{Re}\lambda_1^\alpha(x^{\otimes N})$ is satisfied.

In [18], they failed to specify the choice of J whose naive construction of J requires $\mathcal{O}(d^2N^2)$ memory cost. To reduce the memory cost, we propose the following skew matrix:

$$J := J_0 \otimes I_d, \tag{26}$$

where J_0 is a $N \times N$ skew matrix and $\otimes$ is a Kronecker product. We then have the following lemma:

Lemma 1. *If J_0 is generated based on Theorem 5 and* $\mathrm{Ker}J_0 = \{0\}$ *is satisfied, then given a point $x^{\otimes N}$, J does not satisfy the equality condition in Theorems 3,4, which means $\lambda_1^0(x^{\otimes N}) \neq$ $\mathrm{Re}\lambda_1^\alpha(x^{\otimes N})$ with probability 1.*

See Appendix G.2 for the proof. Thus, from this lemma, we only need to prepare and store J_0, which requires $\mathcal{O}(N^2)$ memory, which does not depend on d. In practical settings, this is a significant reduction of the memory size since the number of parallel chains is smaller than the dimension of models. Please note that we can ensure the acceleration with this J.

Lemma 2. *Under Assumptions 1–5, assume J satisfies the condition of Lemma 1. Then S-PLD shows*

$$\chi^2(\mu_t^{\alpha,\otimes N} \| \pi^{\otimes N}) \leq e^{-\frac{2m(\alpha,N)}{\beta}t}\chi^2(\mu_0^{\otimes N} \| \pi^{\otimes N}), \tag{27}$$

where $\mu_t^{\alpha,\otimes N}$ is the measure at time t induced by S-PLD, and $\mu_0^{\otimes N}$ is the initial measure defined as the product measure of μ_0.

See Appendix I.1 for the proofs. Thus, combined with Lemma 2, S-PLD converges faster than PLD. We also considered the ensemble version of ULD (parallel ULD (PULD)) and its skew accelerated version:

$$\begin{aligned} dX_t^{\otimes N} &= \Sigma^{-1}V_t^{\otimes N}dt + \alpha_1 J_1 \nabla U^{\otimes N}(X_t^{\otimes N})dt, \\ dV_t^{\otimes N} &= -\nabla U^{\otimes N}(X_t^{\otimes N})dt - \gamma(\Sigma^{-1} + \alpha_2 J_2)V_t^{\otimes N}dt + \sqrt{2\gamma\beta^{-1}}dw_t, \end{aligned} \tag{28}$$

where J_1 and $J_2 \in \mathbb{R}^{dN \times dN}$ are real-valued skew-symmetric matrices, and α_1 and $\alpha_2 \in \mathbb{R}_+$ are positive constants and $V_t^{\otimes N} = \left(V_t^{(1)}, \dots, V_t^{(N)}\right)^\top \in \mathbb{R}^{dN}$. We refer to this dynamics as skew PULD (S-PULD) whose faster convergence can be assured similar to Lemma 2 as shown in Appendix I.2.

4.2. Discussion of the Discretization of SDE and Stochastic Gradient and Practical Algorithm

In this section, we further consider practical settings for S-PLD and S-PULD. We discretize these continuous dynamics, e.g., by the Euler-Maruyama method, and approximate the gradient by the stochastic gradient. Although introducing skew matrices accelerates the convergence of continuous dynamics, it simultaneously increases the discretization and stochastic gradient error, resulting in a trade-off. We present a practical algorithm that controls this trade-off.

4.2.1. Trade-Off Caused by Discretization and Stochastic Gradient

We consider the following discretization and stochastic gradient for S-PLD and S-PULD:

$$X_{k+1}^{\otimes N} = X_k^{\otimes N} - h(I_{dN} + \alpha J)\nabla\hat{U}^{\otimes N}(X_k^{\otimes N}) + \sqrt{2h\beta^{-1}}\epsilon_k, \tag{29}$$

and

$$\begin{aligned} X_{k+1}^{\otimes N} &= X_k^{\otimes N} + \Sigma^{-1}V_k^{\otimes N}h + \alpha J\nabla\hat{U}^{\otimes N}(X_k^{\otimes N})h \\ V_{k+1}^{\otimes N} &= V_k^{\otimes N} - \nabla\hat{U}^{\otimes N}(X_k^{\otimes N})h - \gamma\Sigma^{-1}V_k^{\otimes N}h + \sqrt{2\gamma\beta^{-1}h}\epsilon_k, \end{aligned} \tag{30}$$

where $\epsilon_k \in \mathbb{R}^{dN}$ is a standard Gaussian random vector. $\nabla\hat{U}^{\otimes N}(X^{\otimes N})$ is an unbiased estimator of the gradient $\nabla U^{\otimes N}(X^{\otimes N})$. We refer to Equation (29) as skew-SGLD and Equation (30) as skew-SGHMC. For skew-SGHMC, we dropped J_2 of S-PULD to decrease the parameters, shown in Appendix B. Please note that skew-SGLD is the identical as the previous dynamics [18]. We introduce an assumption about the stochastic gradient:

Assumption 7. *(Stochastic gradient) There exists a constant $\delta \in [0,1)$ such that*

$$\mathbb{E}[\|\nabla\hat{U}(x) - \nabla U(x)\|^2] \le 2\delta\left(M^2\|x\|^2 + B^2\right). \tag{31}$$

Given a test function f with L_f lipschitzness, we approximate $\int f d\pi$ by skew-SGLD or skew-SGHMC, with estimator $\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)})$. The bias of skew-SGLD is upper-bounded as

Theorem 7. *Under Assumptions 1–7, for any $k \in \mathbb{N}$ and any $h \in (0, 1 \wedge \frac{m}{4M^2})$ obeying $kh \ge 1$ and $\beta m \ge 2$, we have*

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \le L_f(\underbrace{C_1(\alpha)kh}_{(i)} + \underbrace{C_2 e^{-\beta^{-1}m(\alpha,N)kh}}_{(ii)}) \tag{32}$$

and C_1 and C_2 depends on the constants of Assumptions 1–7, for the details see Appendix J.

We present a tighter bias bound in Section 4.3 under a stronger assumption. We can show a similar upper bound for the skew-SGHMC using the same proof strategy. This bound resembles of a previous one [18]; ours shows improved dependency on kh. The previous results of [18] are also limited to LD, not including skew-SGHMC.

Please note that (i) corresponds to the discretization and stochastic gradient error and (ii) corresponds to the convergence behavior of S-PLD, which is continuous dynamics. Since $C_1(\alpha) \ge C_1(\alpha = 0)$, skew acceleration increases the discretization and stochastic gradient error. On the other hand, since $m(\alpha, N) \ge m_0$, the convergence of the continuous dynamics is accelerated. Thus, skew acceleration causes a trade-off. When α is sufficiently small, we derive the explicit dependency of α for this trade-off from an asymptotic expansion. Using the quantitative evaluation of skew acceleration in Theorem 6, we obtain

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq \underbrace{(d_1\alpha + d_2\alpha^2)kh}_{(i)} - \underbrace{\alpha^2 d_0 e^{-\beta^{-1}m_0 kh}}_{(ii)} + \mathcal{O}(\alpha^3) + \text{const}, \tag{33}$$

where d_0 to d_2 are positive constants obtained by the asymptotic expansion. See Appendix K for the details. In the above expression, (i) and (ii) correspond to (i) and (ii) of Equation (32). Thus, by choosing appropriate α, we can control the trade-off.

4.2.2. Practical Algorithm Controlling the Trade-Off

Since calculating the optimal α that minimizes Equation (33) at each step is computationally demanding, we adaptively tune the value of α by measuring the acceleration with kernelized Stein discrepancy (KSD) [22]. Our idea is to update samples under different α and $\alpha+\eta$, and compare KSD between the stationary and empirical distributions of these different interaction strengths. Here, $\eta \in \mathbb{R}^+$ is a small increment of α. We denote the samples at the $(k+1)$th step, which is obtained by Equation (29) as $X_{k+1,\alpha}^{\otimes N} := X_{k,\alpha}^{\otimes N} - h(I_{dN} + \alpha J)\nabla\hat{U}^{\otimes N}(X_{k,\alpha}^{\otimes N}) + \sqrt{2h\beta^{-1}}\epsilon_k$, (or (30) as $X_{k+1,\alpha}^{\otimes N} := X_k^{\otimes N} + \Sigma^{-1}V_k^{\otimes N}h + \alpha J\nabla\hat{U}^{\otimes N}(X_k^{\otimes N})h$). We denote the samples, which are obtained by replacing the above α by $\alpha+\eta$, as $X_{k+1,\alpha+\eta}^{\otimes N}$. We denote the KSD between the measure of $X_{k+1,\alpha}^{\otimes N}$ and stationary measure π as $KSD(k+1,\alpha)$ and estimate the differences of empirical KSD:

$$\Delta := K\hat{S}D(k+1,\alpha) - K\hat{S}D(k+1,\alpha+\eta), \tag{34}$$

where KSD is estimated by

$$K\hat{S}D(k,\alpha) = \frac{1}{N(N-1)}\sum_{i=1}^{N} u_q(X_{k,\alpha}^{(i)}, X_{k,\alpha}^{(j)}), \tag{35}$$

$$u_q(x,x') := \nabla_x \log\pi(x)^\top l(x,x')\nabla_x \log\pi(x') + \nabla_x \log\pi(x)^\top \nabla_{x'} l(x,x') + \nabla_x l(x,x')^\top \nabla_x \log\pi + \mathrm{Tr}\nabla_{x,x'} l(x,x'), \tag{36}$$

where l denotes a kernel and we use an RBF kernel. If $\Delta > 0$, which indicates that the empirical distribution of $X_{k+1,\alpha+\eta}^{\otimes N}$ is closer to the stationary distribution than that of $X_{k+1,\alpha}^{\otimes N}$. Thus, we should increase the interaction strength from α to $\alpha+\eta$. If $\Delta < 0$, we decrease it to $\alpha-\eta$. We also update η to $c\eta$ where $c \in (0,1]$. The overall process is shown in Algorithm 1. Detailed discussions of the algorithm including how to select α_0, η_0, and c are shown in Appendix L.

Algorithm 1 Tuning α

```
Input: $X_k^{\otimes N}, \eta_k, \alpha_k, c$
Output: $\alpha_{k+1}, \eta_{k+1}$
1: Calculate $X_{k+1,\alpha_k}^{\otimes N}$ and $X_{k+1,\alpha_k+\eta_k}^{\otimes N}$.
2: Calculate $\Delta := K\hat{S}D(k+1,\alpha_k) - K\hat{S}D(k+1,\alpha_k+\eta_k)$
3: if $\Delta > 0$ then
4:     Update $\alpha_{k+1} = \alpha_k + \eta_k$
5:     Update $\eta_{k+1} = \eta_k$
6: else
7:     Update $\alpha_{k+1} = |\alpha_k - \eta_k|$
8:     Update $\eta_{k+1} = c\eta_k$
9: end if
```

Finally, we present Algorithm 2, which describes the whole process. We update the value of α once every k' step. Please note that its computational cost is not much larger than that of Equation (30). We only calculate the eigenvalues of J once, which requires $\mathcal{O}(N^3)$. The calculation of different KSDs is computationally inexpensive since we can re-use the gradient, which is the most computationally demanding part.

Algorithm 2 Proposed algorithm

```
Input: $X_0^{\otimes N}, h, \alpha_0, \eta, k', K, c, (V_0^{\otimes N}, \gamma, \Sigma^{-1})$
Output: $X_K^{\otimes N}$
1: Make a $N \times N$ random matrix $J_0$ and check $\ker J_0 = \{0\}$
2: Set $J = J_0 \otimes I_d$
3: for $k = 0$ to $K$ do
4:     if $\lfloor \frac{k}{k'} \rfloor = 0$ then
5:         Update $\alpha$ by Algorithm 1
6:     end if
7:     Update $X_k^{\otimes N}$ by Equation (29) (for skew-SGLD)
8:     (Update $(X_k^{\otimes N}, V_k^{\otimes N})$ by Equation (30) for skew-SGHMC)
9: end for
```

4.3. Refined Analysis for the Bias of Skew-SGLD

When using a constant step size for skew-SGLD, the bound in Theorem 7 is meaningless since the first term of Equation (32) will diverge. Here, following [23], we present a tighter bound for the bias of skew-SGLD under a stronger assumption.

Theorem 8. *Under Assumptions 1–7, for any $k \in \mathbb{N}$ and any $h \in (0, 1 \wedge \frac{\lambda(\alpha,N)}{4\sqrt{2}M^2} \wedge \frac{m}{4M^2})$ obeying $kh \geq 1$ and $\beta m \geq 2$, we have*

$$\left| \mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi \right| \leq L_f \sqrt{\frac{2}{\lambda(\alpha,N)}} \sqrt{e^{-\lambda(\alpha,N)kh}\mathrm{KL}(\mu_0|\pi) + \frac{C_3(\alpha)}{\lambda(\alpha,N)}}, \tag{37}$$

where

$$\lambda(\alpha,N) := \left(\frac{1}{(1+m(\alpha,N)^{-1}\beta C(m_0))2\pi e^2} + \frac{3}{2}m(\alpha,N)^{-1} \right)^{-1} \tag{38}$$

and constants $C_3(\alpha)$ and $C(m_0)$ depend on the constants of Assumptions 1–7. Moreover, $\lambda(\alpha,N)$ satisfies $\lambda(\alpha,N) \geq \lambda(\alpha=0,N)$. For the details, see Appendix M.

Proof is shown in Appendix M. Please note that even if we use a constant step size for skew-SGLD, the bound in Theorem 8 will not diverge. Here we need the stronger assumption about a step size compared to Theorem 7. From Equation (37), the convergence behavior is characterized by $\lambda(\alpha,N)$ and the bias bound become smaller when $\lambda(\alpha,N)$ become larger. From the definition of $\lambda(\alpha,N)$, the larger $m(\alpha,N)$ is, the larger $\lambda(\alpha,N)$ we obtain. Thus, as we had seen so far, introducing the skew matrices leads to the larger Poincaré constant, and thus, this leads to larger $\lambda(\alpha,N)$.

Previous work [18] clarified that if α is sufficiently small, introducing skew matrices improves the Poincaré constant by a constant factor, which means that we have $m(\alpha,N) - m_0 \approx \mathcal{O}(\alpha^2)$, where $\mathcal{O}(\alpha^2)$ depends on the eigenvector and eigenvalues of the generator $\mathcal{L}$. On the other hand, from Theorem 8, for any $\xi > 0$, to achieve the bias smaller than ξ, it suffice to run skew-SGLD at least for $k \geq \frac{2}{\lambda(\alpha,N)h} \ln \frac{L_f}{\xi}\sqrt{\frac{\mathrm{KL}(\mu_0|\pi)}{2\lambda(\alpha,N)}}$ iterations using the appropriate step size h and under the assumption that δ and α are small enough (see Appendix M.2 for details). Combined with these observations, introducing skew matrices into SGLD improves the computational complexity for a constant order. Our numerical experiments show that even constant improvement results in faster convergence in practical Bayesian models.

5. Related Work

In this section, we discuss the relationship between our method and other sampling methods.

5.1. Relation to Non-Reversible Methods

As we discussed in Section 1, our work extends the existing analysis of non-reversible dynamics [8,18] and presents a practical algorithm. Compared to those previous works, we focus on the practical setting of Bayesian sampling and derive the explicit condition about J for acceleration. We also derived a formula to quantitatively evaluate skew acceleration based on the asymptotic expansion of the eigenvalues of the perturbed Hessian matrix. A previous work [24], which derived the optimal skew matrices when the target distribution is Gaussian, requires $\mathcal{O}(d^3)$ computational cost to derive optimal skew matrices, and it is unclear whether it works for non-convex potential functions. On the other hand, our construction method for skew matrices is simple, computationally cheap, and can be applied to general Bayesian models.

Our work analyzes skew acceleration for ULD, which is more effective than LD in practical problems. Another work [8,18] only analyzed skew acceleration for LD. A previous work [17] combined a non-reversible drift term with ULD. Unlike our method, this work's purpose was to reduce the asymptotic variance of the expectation of a test function and is mainly focusing on sampling from Gaussian distribution.

To the best of our knowledge, our work is the first to focus on the memory issue of skew acceleration and develop a memory-efficient skew matrix for ensemble sampling. Our work also presents an algorithm that controls the trade-off for the first time. Another work [18] identified the trade-off and handled it by cross-validation, which is computationally inefficient, unfortunately.

Finally, we point out an interesting connection between our skew-SGHMC and the magnetic HMC (M-HMC) [25]. M-HMC accelerates HMC's mixing time by introducing a "magnetic" term into the Hamiltonian. That magnetic term is expressed by special skew matrices. Although a previous work [25] argued that M-HMC is numerically superior to a standard HMC, its theoretical property remains unclear. Thus, our work can analyze the theoretical behavior of magnetic HMC.

5.2. Relation to Ensemble Methods

Our proposed algorithm is based on ensemble sampling [26]. Ensemble sampling, in which multiple samples are simultaneously updated with interaction, has been attracting attention numerically and theoretically because of improvements in memory size, computational power, and parallel processing computation schemes [26]. There are successful, widely used ensemble methods, including SVGD [27] and SPOS [28], with which we compare our proposed method numerically in Section 6. Although both show numerically good performance, it is unclear how the interaction term theoretically accelerates the convergence since they are formulated as a McKean–Vlasov process, which is non-linear dynamics, complicating establishing a finite sample convergence rate. Our algorithm is an extension of another work [18], where the interaction was composed of a skew-acceleration term and can be rigorously analyzed. Compared to that previous work [18], we analyzed skew acceleration, focused on the Hessian matrix, and developed practical algorithms, as discussed in Section 4.2, and derived the explicit condition when acceleration occurs, which was unclear [18].

Another difference among SPOS, SVGD, and [18] is that they use first-order methods; our approach uses the second-order method. Little work has been done on ensemble sampling for second-order dynamics. Recently a second-order ensemble method was proposed [29], based on gradient flow analysis. Although its method showed good numerical performance, its theoretical property for finite samples remains unclear since it proposed a scheme as a finite sample approximation of the gradient flow. In contrast, our proposed method is a valid sampling scheme with a non-asymptotic guarantee.

6. Numerical Experiments

The purpose of our numerical experiments is to confirm the acceleration of our algorithm proposed in Section 4 in various commonly used Bayesian models including Gaussian distribution (toy data), latent Dirichlet allocation (LDA), and Bayesian neural net regression and classification (BNN). We compared our algorithm's performance with other ensemble sampling methods: SVGD, SPOS, standard SGLD, and SGHMC. In all the experiments, the values and the error bars are the mean and the standard deviation of repeated trials. For all the experiments we set $\gamma = 1$ and $\Sigma^{-1} = 300$ for SGHMC and Skew-SGHMC. As for the hyperparameters of our proposed algorithm, the selection criterion is discussed in Appendix L.

6.1. Toy Data Experiment

The target distribution is the multivariate Gaussian distribution, $\pi = N(\mu, \Omega)$ where we generated $\Omega^{-1} = A^\top A$ and each element of $A \in \mathbb{R}^{2d\times d}$ is drawn from the standard Gaussian distribution. The dimension of the target distribution is $d = 50$, we approximate by 20 samples using the proposed ensemble methods. We tested these toy data because the LD for this target distribution is known as the Ornstein–Uhlenbeck process, and its theoretical properties have been studied extensively e.g., [30]. Thus, by studying the convergence behavior of these toy data, we can understand our proposed method more clearly.

First, we confirmed how the skew-symmetric matrix affects the eigenvalues of the Hessian matrix, as discussed in Section 3, where we only showed the asymptotic expansion for the smallest real part of the eigenvalues and saddle point. Here we can show a similar expansion for the largest real part:

$$\mathrm{Re}(\lambda_{dN}^{\alpha}) = \lambda_{dN}^{0} + \alpha^2 \sum_{k=1}^{dN-1} \frac{\lambda_d N^0 \lambda_k^0 |v_k^0 J v_{dN}^0|^2}{\lambda_k^0 - \lambda_{dN}^0} + \mathcal{O}(\alpha^3). \tag{39}$$

$\mathrm{Re}\big(\lambda_{dN}^{\alpha}\big) \leq \lambda_{dN}^{\alpha}$ holds.

Then we observed how the largest and smallest real parts of the eigenvalues of $(I + \alpha J)\Omega^{-1}$ depend on α. The results are shown in Figure 2, where we averaged 10 trials over a randomly made J with fixed A. The upper-left, upper-right, and lower figures show $\mathrm{Re}(\lambda_1(\alpha))$, $\mathrm{Re}(\lambda_{dN}(\alpha))$, and $\mathrm{Re}(\lambda_1(\alpha))/\mathrm{Re}(\lambda_{dN}(\alpha))$. These behaviors are consistent with Theorem 3. When α is small, its behavior is close to the quadratic function proved in Theorem 3.

Next, we observed the convergence behavior of skew-SGLD and skew-SGHMC. We measured the convergence by maximum mean discrepancy (MMD) [31] between the empirical and stationary distributions. For MMD, we used 2000 samples for the target distribution, and we used the Gaussian kernel whose bandwidth is set to the median distance of these 2000 samples. We used gradient descent (GD), with step size $h = 1 \times 10^{-4}$. The results are shown in Figure 3. The proposed method shows faster convergence than naive parallel sampling, which is consistent with Table 2.

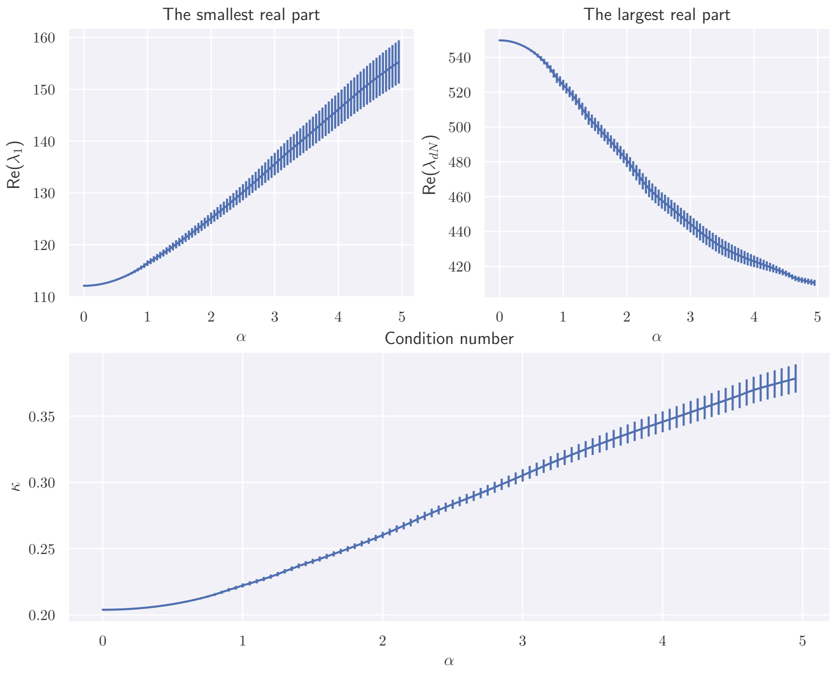


Figure 2. Eigenvalue changes (averaged over ten trials).

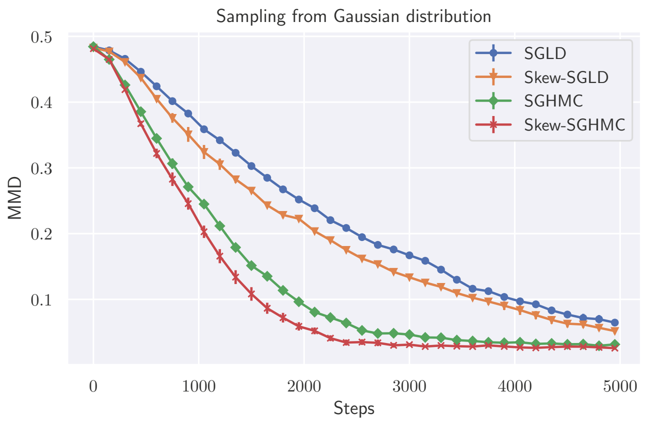


Figure 3. Convergence behavior of toy data in MMD (averaged over ten trials).

6.2. LDA Experiment

We tested with an LDA model using the ICML dataset [32] following the same setting as [33]. We used 20 samples for all the methods. Minibatch size is 100. We used step size $h = 5 \times 10^{-4}$. First, we confirmed the effectiveness of our proposed Algorithm 1, which adaptively tunes α values. For that purpose, we compared the final performance obtained by our methods with a previous method [18], in which α is selected by cross-validation (CV). Here instead of CV, we just fixed α during the sampling and refer to it as fixed α. We also tested the case when J is generated randomly at each step with fixed α, as discussed in Section 4.1. We refer to it as random J. The results are shown in Figure 4 where skew-SGLD was used. We found that our method showed competitive performance with the best performance of fixed α. For the computational cost, we used $k' = 2$ in Algorithm 2, and our method needed twice the wall clock time than each fixed α. This means that our algorithm greatly reduces the total computational time since we tried more than two αs in the fixed α for CV. We also found that since using different Js at each step did not accelerate the performance, we need to store and fix J during the sampling for acceleration. Next, we compared our method with other ensemble sampling schemes and observed the convergence speed. The result is shown in Figure 5. Skew-SGLD and skew-SGHMC outperformed SGLD and SGHMC, which is consistent with our theory.

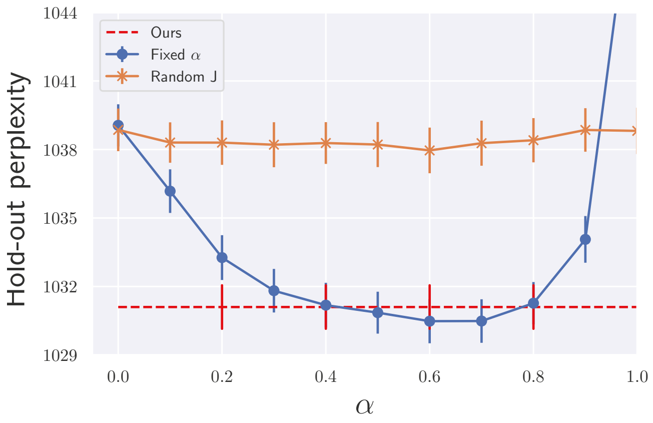


Figure 4. Final performances of LDA under different values of α (averaged over ten trials).

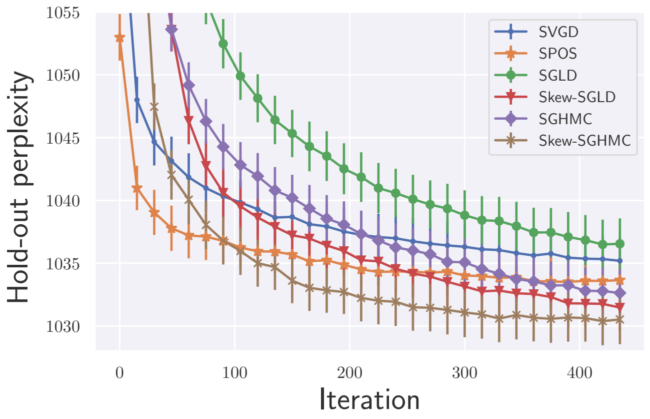


Figure 5. LDA experiments (Averaged over 10 trials).

6.3. BNN Regression and Classification

We tested with the BNN regression task using the UCI dataset [34], following a previous setting Liu and Wang [27]. We used one hidden layer neural network model with ReLU activation and 100 hidden units. We used 10 samples for all the methods. We used the minibatch size 100. We used step size $h = 5 \times 10^{-5}$. The results are shown in Tables 1 and 2. We also tested on BNN classification task using the MNIST dataset. The result is shown in Figure 6. We used one hidden layer neural network model with ReLU activation and 100 hidden units. Batchsize is 500 and we set step size $h = 5 \times 10^{-5}$. Our proposed methods outperformed other ensemble methods. Please note that skew-SGHMC and skew-SGLD consistently outperformed SGHMC and SGLD.

Table 1. Benchmark results on test RMSE for regression task.

Dataset	Avg. Test RMSE					
	SVGD	SPOS	SGLD	Skew-SGLD	SGHMC	Skew-SGHMC
Concrete	5.709 ± 0.040	5.239 ± 0.199	5.009 ± 0.091	4.973 ± 0.057	4.949 ± 0.144	4.790 ± 0.081
Kin8nm	0.0731 ± 0.0006	0.0688 ± 0.0003	0.0693 ± 0.0006	0.0689 ± 0.0005	0.0687 ± 0.0001	0.0683 ± 0.0003
Energy	0.520 ± 0.060	0.456 ± 0.030	0.428 ± 0.045	0.412 ± 0.045	0.406 ± 0.019	0.403 ± 0.008
Bostonhousing	3.306 ± 0.005	3.107 ± 0.173	2.948 ± 0.084	2.930 ± 0.095	3.053 ± 0.093	2.986 ± 0.143
Winequality	0.619 ± 0.001	0.618 ± 0.007	0.641 ± 0.003	0.634 ± 0.004	0.614 ± 0.004	0.613 ± 0.004
PowerPlant	4.219 ± 0.012	4.160 ± 0.009	4.129 ± 0.002	4.118 ± 0.006	4.112 ± 0.009	4.105 ± 0.008
Yacht	0.475 ± 0.049	0.467 ± 0.110	0.464 ± 0.058	0.442 ± 0.046	0.464 ± 0.078	0.432 ± 0.051

Table 2. Benchmark results on test negative log likelihood for regression task.

Dataset	Avg. Test Negative Log Likelihood					
	SVGD	SPOS	SGLD	Skew-SGLD	SGHMC	Skew-SGHMC
Concrete	-3.157 ± 0.008	-3.124 ± 0.025	-3.052 ± 0.009	-3.049 ± 0.012	-3.046 ± 0.025	-3.033 ± 0.021
Kin8nm	1.153 ± 0.0084	1.212 ± 0.008	1.223 ± 0.002	1.223 ± 0.005	1.230 ± 0.0015	1.235 ± 0.0025
Energy	-0.816 ± 0.102	-0.976 ± 0.079	-0.867 ± 0.056	-0.845 ± 0.021	-0.843 ± 0.045	-0.844 ± 0.041
Bostonhousing	-2.98 ± 0.000	-2.644 ± 0.027	-2.548 ± 0.016	-2.539 ± 0.002	-2.574 ± 0.019	-2.561 ± 0.017
Winequality	-1.012 ± 0.000	-0.959 ± 0.007	-0.976 ± 0.006	-0.968 ± 0.005	-0.941 ± 0.007	-0.938 ± 0.005
PowerPlant	-2.871 ± 0.004	-2.850 ± 0.004	-2.844 ± 0.002	-2.842 ± 0.001	-2.838 ± 0.004	-2.835 ± 0.003
Yacht	-1.184 ± 0.06	-1.372 ± 0.07	-1.077 ± 0.066	-1.078 ± 0.030	-1.083 ± 0.030	-1.079 ± 0.051

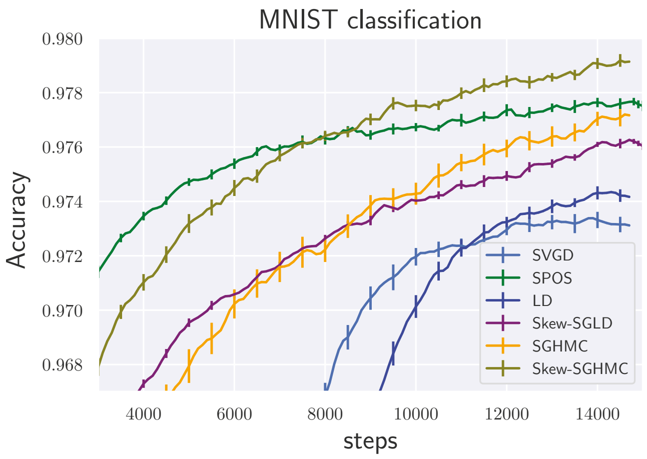


Figure 6. MNIST classification (Averaged over ten trials).

7. Conclusions

We studied skew acceleration for LD and ULD from practical viewpoints and concluded that the improved eigenvalues of the perturbed Hessian matrix caused acceleration and derived the explicit condition for acceleration. We described a novel ensemble sampling method, which couples multiple SGLD or SGHMC with memory-efficient skew matrices. We also proposed a practical algorithm that controls the trade-off of faster convergence and larger discretization and stochastic gradient error and numerically confirmed the effectiveness of our proposed algorithm.

Author Contributions: Conceptualization, F.F. and T.I.; methodology, F.F. and T.I.; software, F.F.; validation, F.F., T.I., N.U. and I.S.; formal analysis, F.F. and I.S.; writing—original draft preparation, F.F.; project administration, F.F.; funding acquisition, F.F. All authors have read and agreed to the published version of the manuscript.

Funding: JST ACT-X: Grant Number JPMJAX190R.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: http://archive.ics.uci.edu/ml (accessed on 21 June 2021).

Acknowledgments: FF was supported by JST ACT-X Grant Number JPMJAX190R.

Conflicts of Interest: The authors declare no conflict of interest.The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

LD	Langevin Dynamics
MCMC	Markov Chain Monte Carlo
ULD	Underdamped Langevin Dynamics
SGLD	Stochastic Gradient Langevin Dynamics
SGHMC	Stochastic Gradient Hamilton Monte Carlo
PLD	Parallel Langevin Dynamics
PULD	Parallel Underdamped Langevin Dynamics
SLD	Skew Langevin Dynamics
S-ULD	Skew Underdamped Langevin Dynamics
S-PLD	Skew Parallel Langevin Dynamics
S-PULD	Skew Parallel Underdamped Langevin Dynamics
KSD	Kernelized Stein Discrepancy

Appendix A. Additional Backgrounds

We introduce additional backgrounds which are used in our Proof.

Appendix A.1. Wasserstein Distance and Kullback–Leibler Divergence

In this paper, we use the Wasserstein distance. Let us define the Wasserstein distance. Let (E, d) be a metric space (appropriate space such as Polish space) with σ field $\mathcal{A}$, where $d(\cdot,\cdot)$ is $\mathcal{A}\times\mathcal{A}$-measurable. Let μ, ν are probability measures on E, and $p \geq 1$. The Wasserstein distance of order p with cost function d between μ and ν is defined as

$$W_p^d(\mu,\nu) = \inf_{\pi\in\Pi(\mu,\nu)} \left(\int\int d(x,y)^p d\pi(x,y) \right)^{1/p}, \tag{A1}$$

where $\Pi(\mu,\nu)$ is the set of all joint probability measures on $E \times E$ with marginals μ and ν. In this paper, we work on the space $\mathbb{R}^d$. As for the distance, we use the Euclidean distance, $\|\cdot\|$. For simplicity, we express the p-Wasserstein distance with the Euclidean distance as W_p. The various properties of Wasserstein distance are summarized in [35]. We define the Kullback–Leibler (KL) divergence as

$$\mathrm{KL}(\nu\|\mu) = \begin{cases} \int \log \frac{d\nu}{d\mu} d\nu, & \nu \ll \mu, \\ +\infty, & \text{otherwise}. \end{cases} \tag{A2}$$

Appendix A.2. Markov Diffusion and Generator

Here we introduce the additional explanation about the generator of the Markov diffusion process. Given an SDE,

$$dX_t = -\nabla U(X_t)dt + \sqrt{2\beta^{-1}}dw(t), \tag{A3}$$

and we denote the corresponding Markov semigroup as $P = \{P_t\}_{t>0}$ and define the Kolmogorov operator as P_s which is defined as $P_s f(X_t) = \mathbb{E}[f(X_{t+s})|X(t)]$, where $f : \mathbb{R}^d \to \mathbb{R}$ is some bounded test function in $\mathrm{L}^2(\mu)$. A property $P_{s+t} = P_s \circ P_t$ is called Markov property. A probability measure π is the stationary distribution when it satisfies for all measurable bounded function f and t, $\int_{\mathbb{R}^d} P_t f d\pi = \int_{\mathbb{R}^d} f d\pi$.

We denote the infinitesimal generator of the associated Markov group as $\mathcal{L}$ and we call it a generator for simplicity. The linearity of the operators of P_t with the semigroup property indicates that $\mathcal{L}$ is the derivative of P_t as

$$\frac{1}{h}(P_{t+h} - P_t) = P_t \frac{1}{h}(P_h - Id) = \frac{1}{h}(P_h - Id)P_t, \tag{A4}$$

where Id is the identity map. In addition, taking $h \to 0$, we have $\partial P_t = \mathcal{L}P_t = P_t\mathcal{L}$. From the Hille–Yoshida theory [19], there exists a dense linear subspace of $\mathrm{L}^2(\pi)$ on which $\mathcal{L}$ exists. We refer it as $\mathcal{D}(\mathcal{L})$. If the Markov semigroup is associated with the SDE of Equation (A3), the generator can be written as

$$\mathcal{L}f(X_t) := \lim_{h \to 0^+} \frac{\mathbb{E}(f(X_{t+h})|X_t) - f(X_t)}{h} = \Big(-\nabla U(X_t) \cdot \nabla + \beta^{-1}\Delta\Big) f(X_t), \tag{A5}$$

where Δ is the Laplacian in the standard Euclidean space. The generator satisfies $\mathcal{L}1 = 0, \quad \int_{\mathbb{R}^d} \mathcal{L} f d\pi = 0$.

Appendix A.3. Poincaré Inequality

We use the Poincaré inequality to measure the speed of convergence to the stationary distribution. In this section, we summarize definitions and useful properties of them and see [19] for more details. We define the Dirichlet form $\mathcal{E}(f)$ for all bounded functions $f \in \mathcal{D}(\mathcal{L})$ where $\mathcal{D}(\mathcal{L})$ denotes the domain of $\mathcal{L}$ as

$$\mathcal{E}(f) := -\int_{\mathbb{R}^d} f\mathcal{L}f d\pi. \tag{A6}$$

$\mathcal{E}(f) > 0$ is satisfied. By the partial integration, we have $\mathcal{E}(f) = -\int_{\mathbb{R}^d} f\mathcal{L}f d\pi = \frac{1}{\beta}\int_{\mathbb{R}^d} \|\nabla f\|^2 d\pi$. We define a Dirichlet domain, $\mathcal{D}(\mathcal{E})$, which is the set of functions $f \in L^2(\pi)$ and satisfies $\mathcal{E}(f) < \infty$.

We say that π with $\mathcal{L}$ satisfies *a Poincaré inequality* with a positive constant c if for any $f \in \mathcal{D}(\mathcal{E})$, π with $\mathcal{L}$ satisfies,

$$\int f^2 d\pi - \left(\int f d\pi\right)^2 \leq c\mathcal{E}(f). \tag{A7}$$

This constant c is closely related to a spectral gap. If the smallest eigenvalue of $\mathcal{L}$, λ, is greater than 0, then it is called the spectral gap. If the spectral gap $\lambda > 0$ exists, then it is written as

$$\lambda := \inf_{f \in \mathcal{D}(\mathcal{E})} \left\{ \frac{\mathcal{E}(f)}{\int f^2 d\pi} : f \neq 0, \int f d\pi = 0 \right\}. \tag{A8}$$

From this, a constant c which satisfies $c \geq 1/\lambda$, can also satisfy the Poincaré inequality. To check the existence of the spectral gap, one approach is to use the Lyapunov function, which is developed by Bakry et al. [36].

We can also express the Poincaré inequality via chi divergence. Let us define the χ^2 divergence for $\mu \ll \pi$ as

$$\chi^2(\mu\|\pi) := \left\| \frac{d\mu}{d\pi} - 1 \right\|^2_{L^2_\pi} = \int_{\mathbb{R}^d} \left| \frac{d\mu}{d\pi} - 1 \right|^2 d\pi. \tag{A9}$$

Then, we express the Poincaré inequality with a constant c for all $\mu \ll \pi$ as

$$\chi^2(\mu\|\pi) \leq c\,\mathcal{E}\left(\sqrt{\frac{d\mu}{d\pi}}\right). \tag{A10}$$

We obtain the following exponential convergence results from the above functional inequalities for measures.

Theorem A1. *(Exponential convergence in the variance, Theorem 4.2.5 in [19]) When π satisfies the Poincaré inequality with a constant c, it implies the exponential convergence in the variance with a rate $2/c$, i.e., for every bounded function $f : \mathbb{R}^d \to \mathbb{R}$,*

$$\mathrm{Var}_\pi(P_t f) \leq e^{-2t/c}\mathrm{Var}_\pi(f), \tag{A11}$$

where $\mathrm{Var}_\pi(f) := \int_{\mathbb{R}^d} f^2 d\pi - \left(\int_{\mathbb{R}^d} f d\pi\right)^2$.

We also introduce the important property of Poincaré inequality as for the product measures. These relations play important roles in our analysis.

Theorem A2. *(Stability under the product, Proposition 4.3.1 in [19]) If μ_1 and μ_2 on $\mathbb{R}^d$ satisfy the Poincaré inequalities with a constant c_1 and c_2, then the product $\mu_1 \otimes \mu_2$ on $\mathbb{R}^d \otimes \mathbb{R}^d$ satisfies the Poincaré inequality with the constant* $\max(c_1, c_2)$.

Appendix B. Generator of the Underdamped Langevin Dynamics (ULD)

Following [10], we define the infinitesimal generator of the ULD as

$$\mathcal{L}f(x,v) := -(\gamma v + \nabla U(x))\nabla_v f(x,v) + \gamma\beta^{-1}\Delta f(x,v) + v\nabla_x f(x,v). \tag{A12}$$

Then, we define the generator of S-ULD as

$$\begin{aligned}\mathcal{L}f(x,v) := &-(\gamma v + \nabla U(x))\nabla_v f(x,v) + \gamma\beta^{-1}\Delta f(x,v) \\ &+ v\nabla_x f(x,v) + \alpha_1 J_1 \nabla U(x)\nabla_x f(x,v) + \alpha_1 J_2 \Sigma^{-1} v \nabla_v f(x,v),\end{aligned} \tag{A13}$$

where the second line corresponds to the interaction terms. Then it is easily to confirm $\int_{\mathbb{R}^{2d}} \mathcal{L}f(x,v)d\tilde{\pi} = 0$, where $\tilde{\pi} := \pi \otimes \mathcal{N}(0,\Sigma) \propto e^{-\beta U(x) - \frac{1}{2}\Sigma^{-1}\|v\|^2}$. Thus, the stationary distribution of S-ULD is $\tilde{\pi}$. We can prove this by simply using the partial integral and using the property of the skew-symmetric matrix. Thus, the stationary distribution of S-ULD is $\tilde{\pi}$.

We consider other combinations the skew matrices with ULD. For example, we can consider the following more general combination;

$$\begin{aligned}dX_t &= \Sigma^{-1}V_t dt + \alpha_1 J_1 \nabla U(X_t)dt + \alpha_2 \Sigma^{-1} J_2 V_t dt \\ dV_t &= -\nabla U(X_t)dt - \gamma\Sigma^{-1}V_t dt + \alpha_3 J_3 V_t dt + \alpha_4 J_4 \nabla U(X_t)dt + \sqrt{2\gamma\beta^{-1}}dw_t,\end{aligned} \tag{A14}$$

compared to S-ULD, there are new two terms are included. We can also derive the infinitesimal generator of this Markov process. We express it as $\tilde{\mathcal{L}}$. Then we calculate the infinitesimal change of the expectation of f

$$\int_{\mathbb{R}^{2d}} \tilde{\mathcal{L}} f(x,v) d\tilde{\pi} \neq 0, \tag{A15}$$

which suggests that the stationary distribution of Equation (A14) is different form $\tilde{\pi}$.

It is widely known that underdamped Langevin dynamics converges to (overdamped) Langevin dynamics. Here we observe that S-ULD converges to Skew-LD in [18]. The limiting procedure is widely known, for example, see [17,37,38]. We cite Proposition 1 in [17]; given a stochastic process

$$\begin{aligned} dX_t &= \Sigma^{-1} V_t dt + \alpha_1 J_1 \nabla U(X_t) dt, \\ dV_t &= -\nabla U(X_t) dt - \gamma \Sigma^{-1} V_t dt - \alpha_2 \Sigma^{-1} J_2 V_t dt + \sqrt{2\gamma} dw_t, \end{aligned} \tag{A16}$$

and we rescale it by introducing ϵ which expresses the small mass limit as

$$\begin{aligned} dX_t &= \frac{1}{\epsilon} \Sigma^{-1} V_t dt + \alpha_1 J_1 \nabla U(X_t) dt, \\ dV_t &= -\frac{1}{\epsilon} \nabla U(X_t) dt - \frac{1}{\epsilon^2} \gamma \Sigma^{-1} V_t dt - \frac{1}{\epsilon}^2 \alpha_2 \Sigma^{-1} J_2 V_t dt + \frac{1}{\epsilon} \sqrt{2\gamma} dw_t, \end{aligned} \tag{A17}$$

and by taking the limit $\epsilon \to 0$, the dynamics converges to

$$dX_t = -(\alpha_2 J_2 + \gamma)^{-1} \nabla U(X_t) dt - \alpha_1 J_1 \nabla U(X_t) + (\alpha_2 J_2 + \gamma)^{-1} \sqrt{2\gamma} dw_t. \tag{A18}$$

See Proposition 1 in [17], for the precise statements. Please note that the term related J_2 works as preconditioning. Thus, if we set $\alpha_2 J_2 = 0$, the obtained dynamics are equivalent to the continuous dynamics of skew-SGLD. Thus, our skew-SGHMC is the natural extension of skew-SGLD.

Appendix C. Proof of Theorem 1

Appendix C.1. Proof for S-LD

First, under Asuumptions 1–5, LD has a spectral gap, and its Poincaré constant is upper bounded as

$$\frac{1}{m_0} \leq \frac{2C(d+b\beta)}{m\beta} \exp\left(\frac{2}{m}(M+B)(b\beta+d) + \beta(A+B)\right) + \frac{1}{m\beta(d+b\beta)}. \tag{A19}$$

and this is derived in [2].

Next, we introduce the generator of S-LD

$$\mathcal{L}_\alpha f(x) = \left(-\nabla U_\alpha(x) \cdot \nabla + \beta^{-1} \Delta\right) f(x),$$

where $\nabla U_\alpha(x) := \nabla U(x) + \alpha J \nabla U(x)$.

The proof is almost similar to [18] of Theorem 12.

Proof of Theorem 1. Since the generator $\mathcal{L}_{\alpha=0}$ is self-adjoint, and the suitable growth condition, the spectral of $\mathcal{L}_{\alpha=0}$ is discrete [19]. We denote the spectrum of $\mathcal{L}_{\alpha=0}$ as $\{\lambda_k\}_{k=0}^{\infty} \in \mathbb{R}$ and corresponding normalized eigenvectors as $\{e_k\}_{k=0}^{\infty}$, which are the real functions. We order the spectrum as $0 > \lambda_0 > \lambda_1 > \dots$. Thus, $m_0 = -\lambda_0$.

As for $\mathcal{L}_\alpha$, although it is not a self-adjoint operator, from Proposition 1 in Franke et al. [39], it has discrete complex spectrums. We denote the spectrum of $\mathcal{L}_\alpha$ as $\lambda + i\mu \in \mathbb{C}$ where

$\lambda, \mu \in \mathbb{R}$ and corresponding normalized eigenvector as $u + iv$ where u, v are the real functions and then we have

$$\mathcal{L}_\alpha(u + iv) = (\lambda + i\mu)(u + iv). \tag{A20}$$

From this definition, by checking the real parts and complex parts, following relations are derived

$$\mathcal{L}_\alpha u = \lambda u - \mu v, \tag{A21}$$
$$\mathcal{L}_\alpha v = \lambda v + \mu u. \tag{A22}$$

Due to the divergence-free drift property, for any bounded real value test function $g(x)$,

$$\int g(\mathcal{L}_{\alpha=0} - \mathcal{L}_\alpha)g d\pi = \int \alpha g\gamma \cdot \nabla g d\pi = -\int \alpha g\gamma \cdot \nabla g d\pi, \tag{A23}$$

where we used the partial integral. This means that for any bounded real function $g(x)$,

$$\int g\mathcal{L}_{\alpha=0} g d\pi = \int g\mathcal{L}_\alpha g d\pi. \tag{A24}$$

(This only holds for real functions.) Then, we can evaluate the real part of the eigenvalue λ as follows,

$$\int u\mathcal{L}_{\alpha=0} u d\pi + \int v\mathcal{L}_{\alpha=0} v d\pi = \int u\mathcal{L}_\alpha u d\pi + \int v\mathcal{L}_\alpha v d\pi = \lambda\left(\int u^2 d\pi + \int v^2 d\pi\right) = \lambda. \tag{A25}$$

Then, by expanding the eigenfunction u, v by the eigenfunction $\{e_k\}$,

$$\begin{aligned}\lambda = \int u\mathcal{L}_{\alpha=0} u d\pi + \int v\mathcal{L}_{\alpha=0} v d\pi &= \sum_k \lambda_k \left(\left(\int u e_k d\pi\right)^2 + \left(\int v e_k d\pi\right)^2\right) \\ &\leq \lambda_0 \sum_k \left(\left(\int u e_k d\pi\right)^2 + \left(\int v e_k d\pi\right)^2\right) \leq \lambda_0.\end{aligned} \tag{A26}$$

Thus, the real part of the eigenvalue of $\mathcal{L}_\alpha$ is smaller than the smallest eigenvalue of $\mathcal{L}_\alpha$. This means that the spectral gap of $\mathcal{L}_\alpha$ is larger than that of $\mathcal{L}_{\alpha=0}$, i.e., $m(\alpha) \geq m_0$ holds. □

Appendix C.2. Proof of Theorem 2 (S-ULD)

Proof of Theorem 2 . To prove the S-ULD, we use the result of [20], which characterize the convergence of ULD via the Poincaré constant. Let us denote $\tilde{\mu}_t$ as the measure induced by ULD. Then from Theorem 1 of [20], if π with $\mathcal{L}$ has the Poincaré constant m_0, we have

$$\chi^2(\tilde{\mu}_t \| \tilde{\pi}) \leq \frac{1 + \bar{\epsilon}}{1 - \bar{\epsilon}} e^{-\lambda_\gamma t} \chi^2(\tilde{\mu}_t \| \tilde{\pi}). \tag{A27}$$

where $\bar{\epsilon}$ and λ_γ is given as follows.

$$\lambda_\gamma = \frac{\Lambda(\gamma, \bar{\epsilon} \min(\gamma, \gamma^{-1}))}{1 + \bar{\epsilon} \min(\gamma, \gamma^{-1})}, \tag{A28}$$

where

$$\Lambda(\gamma, \epsilon) = \frac{\gamma\Sigma^{-1} - \frac{1}{1+\frac{m_0\Sigma^{-1}}{\beta}}}{2} - \frac{1}{2}\sqrt{(S_{--} - S_{++})^2 + (S_{-+})^2}, \tag{A29}$$

$$S_{--} = \epsilon\lambda_{ham}, \tag{A30}$$

$$S_{-+} = -\epsilon(R_{ham} + \gamma\Sigma^{-1}/2), \tag{A31}$$

$$S_{++} = \gamma\Sigma^{-1} - \epsilon, \tag{A32}$$

$$\lambda_{ham} = 1 - \left(1 + \frac{m_0\Sigma^{-1}}{\beta}\right)^{-1}, \tag{A33}$$

$$\epsilon = \bar{\epsilon}\min(\gamma, \gamma^{-1}), \tag{A34}$$

where $\bar{\epsilon}$ is arbitrary sufficiently small positive value such that $\Lambda(\gamma, \bar{\epsilon}\min(\gamma, \gamma^{-1})) > 0$ is satisfies. As for R_{ham}, if there exists a positive constant K, such that $\nabla^2 U \geq -KI$, then $R_{ham} \leq \sqrt{\max\{K, 2\}}$. In our assumption, this corresponds to βM, thus $R_{ham} \leq \sqrt{\max\{\beta M, 2\}}$. From the above definitions, we can see that the larger m_0 is, i.e., the larger the Poincaré constant is the faster convergence ULD shows.

This can also be confirmed numerically, see Figure A1, which shows how the Λ changes under different m_0. We set $\Sigma^{-1} = 100$. From the figure, the larger the Poincaré constant is, the larger Λ becomes.

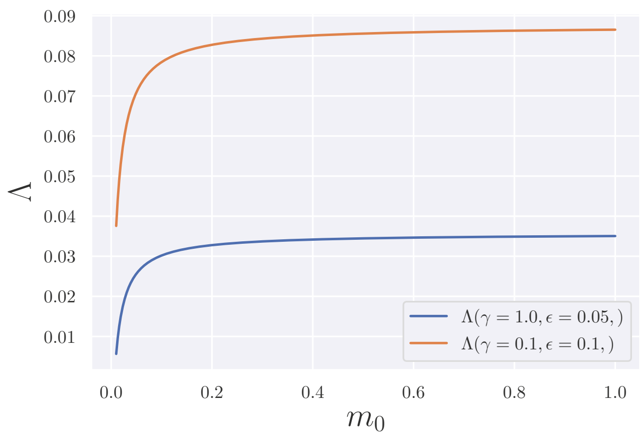


Figure A1. The convergence rate of ULD under the different Poincaré constants.

So far, we confirmed that the convergence speed of S-ULD is characterized by the Poincaré constant of $\mathcal{L}$. When we consider S-ULD, we simply add the skew matrices term to the generator of the ULD in the proof of Proposition 1 in [20]. This means that we simply replace the Poincaré constant from m_0 to $m(\alpha)$ in the proof of Proposition 1 in [20]. Then, m_0 will be replaced with $m(\alpha)$ that indicates the faster convergence. $\square$

Appendix D. Eigenvalue and Poincaré Constant

In this section, we discuss the relation between eigenvalues of the Hessian matrix and Poincaré constant.

Appendix D.1. Strongly Convex Potential Function

When we consider LD with m-strongly convex potential function, then the Poincaré constant is m, this means exponential convergence with rate m (See [19] for the detail).

We then consider the S-LD with m-strongly convex function. In this setting, by considering the synchronous coupling technique [11], we can show that the variance decays exponentially with the rate of the smallest real part of the eigenvalue. This is because that by preparing two S-LD (X_t, Y_t) given as

$$dX_t = -(I+\alpha J)\nabla U(X_t)dt + \sqrt{2\beta^{-1}}dw_t, \qquad dY_t = -(I+\alpha J)\nabla U(Y_t)dt + \sqrt{2\beta^{-1}}dw'_t. \tag{A35}$$

Then we evaluate the behavior of $\|X_t - Y_t\|^2$. From Ito lemma and considering the synchronous coupling, we obtain

$$\frac{d}{dt}\|X_t - Y_t\|^2 = -(X_t - Y_t)\cdot\frac{(I+\alpha J)}{\beta}(\nabla U(X_t) - \nabla U(Y_t)) \leq -\frac{2m(\alpha)}{\beta}\|X_t - Y_t\|^2, \tag{A36}$$

where $m(\alpha)$ is the constant that satisfies $m(\alpha) \leq \mathrm{Re}\lambda_1^\alpha(x)$ for all x, see Appendix E for details. This means that variance decays exponentially with the rate $\frac{2m(\alpha)}{\beta}$. From the fundamental property of the Poincaré constant (Theorem 4.2.5 in [19]), $m(\alpha)$ is the Poincaré constant. Thus the imaginary part has no effect on the continuous dynamics. Thus, the Poincaré inequality is the smallest real part of the perturbed Hessian matrix.

Appendix D.2. Non-Convex Potential Function

As we discussed in Section 3.1, [21] derived the sharper estimation for the Poincaré constant for the non-convex potential function. It is easy to verify that their assumptions are satisfied under our assumption 1–5. Following the main paper, we denote x_1 global minima, and x_2 is the local minima which have the second smallest value in $U(x)$. We express the saddle point between x_1 and x_2 as x^*. To be more precise, the saddle point that characterizes the Poincaré constant is known as the critical point with index one defined as

$$U(x^*) = \inf\left\{\max_{s\in[0,1]} U(\gamma(s)) : \gamma \in C([0,1],\mathbb{R}^d), \gamma(0) = x_1, \gamma(1) = x_2\right\}, \tag{A37}$$

and the eigenvalue of $\nabla^2 U(x^*)$ has one negative eigenvalue and $d-1$ positive eigenvalues. We express them as $\lambda_1(x^*) < 0 < \lambda_2(x^*) < \ldots, \lambda_d(x^*)$.

Ref. [21] studied the Poincaré constant by decomposing the non-convex potential focusing on attractors. By focusing on attractors, they showed that the non-convex potential can be decomposed into the sum of *approximately* Gaussian distributions. They proved that the Poincaré constant is characterized by the local Poincaré constants, these are derived by the approximate Gaussian distribution on the attractors and their surrounding regions. In addition, they proved that the dominant term of the Poincaré constant is specified by the saddle points between the global minima and the point which takes the second smallest value for $U(x)$. From Theorem 2.12 and Corollary 2.15 in [21], the Poincaré constant is characterized by

$$m_0^{-1} \approx \frac{\sqrt{\mathrm{det}H(x^*)}}{\sqrt{Z|\lambda_1(x^*)|\mathrm{det}H(x_1)}\sqrt{\mathrm{det}H(x_2)}}e^{\beta(U(x^*)-U(x_1)-U(x_2))} \propto \frac{1}{|\lambda_1(x^*)|}e^{\beta(U(x^*)-U(x_1)-U(x_2))}, \tag{A38}$$

where Z is the normalizing constant of $e^{-\beta U(x)}$.

Next, we discuss how this estimate changes when skew matrices are applied. When the skew matrices are introduced, from lemma A.1 in [40], at the saddle point, there exists a unique negative real eigenvalue $\lambda_1^\alpha(x^*) < 0$ for the perturbed Hessian matrix even if $(I+\alpha J)H$ is not a symmetric matrix.

Then from Proposition 5 in [8], that negative eigenvalue of the perturbed Hessian is smaller than that of the un-perturbed Hessian matrix at the saddle point. This means that $\lambda_1^\alpha(x^*) \leq \lambda_1(x^*) < 0$ holds.

Finally, from Theorem 5.1 in [41] and Theorem 2.12 in [21], this improvement of the negative eigenvalue of the saddle point directly leads to the larger Poincaré constant.

Appendix E. Properties of a Skew-Symmetric Matrix

Here, we introduce the basic properties of the skew-symmetric matrices. Let us consider assume that $d \times d$ matrix $H' = (I + \alpha J)H$ is diagonalizable. Then assume that matrix H' has l real eigenvalues $\lambda_1, \dots, \lambda_l$ and $2m$ complex eigenvalues, $\mu_1 = \alpha_1 \pm i\beta_1, \dots, \mu_m = \alpha_m \pm i\beta_m$. Thus, $d = l + 2m$. We denote the corresponding eigenvectors as $\{v_j\}_{j=1}^{l}$ for real eigenvalues and $\{w_j = a_j + ib_j\}_{j=1}^{m}$ for complex eigenvalues $\{\mu_j\}_{j=1}^{m}$ and $\{\bar{w}_j\}$ for corresponding conjugate eigenvalues. Then, let us define a $d \times d$ matrix V as

$$V = [v_1, \dots, v_l, a_1, b_1, \dots, a_m, b_m]. \tag{A39}$$

Then, we can decompose H' into a block diagonal matrix [42];

$$H'V = VD \tag{A40}$$

$$D = \underbrace{\begin{pmatrix} \lambda_1 &&&&&&& \\ & \ddots &&&&&& \\ && \lambda_l &&&&& \\ &&& \alpha_1 & 0 &&& \\ &&& 0 & \alpha_1 &&& \\ &&&&& \ddots && \\ &&&&&& \alpha_m & 0 \\ &&&&&& 0 & \alpha_m \end{pmatrix}}_{:=A} + \underbrace{\begin{pmatrix} 0 &&&&&&& \\ & \ddots &&&&&& \\ && 0 &&&&& \\ &&& 0 & \beta_1 &&& \\ &&& -\beta_1 & 0 &&& \\ &&&&& \ddots && \\ &&&&&& 0 & \beta_m \\ &&&&&& -\beta_m & 0 \end{pmatrix}}_{:=B}. \tag{A41}$$

Thus, $D := A + B$. Then, from the Taylor expansion and expressing its residual by integral, by defining $H(x) := \nabla^2 U(x)$ we have

$$(x-y)^\top (I+\alpha J)(\nabla U(x) - \nabla U(y)) = (x-y)^\top \left(\int_0^1 (I+\alpha J)H(y+\tau(x-y))(x-y)d\tau \right). \tag{A42}$$

Then, let us apply the Jordan canonical form here. If $(I + \alpha J)H$ is diagonalizable, and it is decomposable by the Jordan canonical form shown in Equation (A40). Then, we can decompose $(I + \alpha J)H$ as

$$(I+\alpha J)H(x^* + \tau(x(t) - x^*)) = VDV^{-1}. \tag{A43}$$

Then, we obtain

$$\begin{aligned} (x-y)^\top (I+\alpha J)(\nabla U(x) - \nabla U(y)) &= (x-y)^\top \left(\int_0^1 (I+\alpha J)H(y+\tau(x-y))(x-y)d\tau \right) \\ &= \left(\int_0^1 (x-y)^\top V(A+B)V^{-1}(x-y)dt \right) \\ &= \left(\int_0^1 (x-y)^\top VAV^{-1}(x(t)-x^*)dt \right) \\ &\le m(\alpha)\|x(t) - x^*\|^2. \end{aligned} \tag{A44}$$

where $m(\alpha)$ is the constant that satisfies $m(\alpha) \le \min\{\lambda_1, \dots, \lambda_l, \alpha_1, \dots, \alpha_m\}$ for all x. Thus, the imaginary part never appears to the upper bound and we only need to focus on the largest real part of the eigenvalues, if the matrix is diagonalizable. Next subsection describes when the non-symmetric matrix H' is diagonalizable by focusing on the random matrix.

Appendix F. Proof of Theorem 3

Proof. Since the potential function is m-strongly convex, the smallest eigenvalue of the Hessian matrix H is m, which is larger than 0. Thus, H and $H^{1/2}$ are regular matrices. With this in mind, we consider $H + H^{1/2}JH^{1/2}$ as a similar matrix of $H' := (I+J)H$. This is easily confirmed by

$$H^{-1/2}(H + H^{1/2}JH^{1/2})H^{1/2} = H'. \tag{A45}$$

This means that to study the eigenvalues of H', we only need to study the similar matrix $A := H + H^{1/2}JH^{1/2}$. By doing this, A is composed of symmetric and skew-symmetric matrices, which are easy to treat compared to H', where the term JH is difficult to analyze. For simplicity, we omit the dependency of H and H' on x in this section.

Remark A1. *Please note that we can eliminate the strong convexity of* U*, if* H *is a regular matrix. This means that* H *does not have* 0 *as an eigenvalue.*

For simplicity, we assume that the dimension d is an even number. We assume that the eigenvalues and eigenvectors of A are expressed as

$$Aw_j = \mu_j w_j \Leftrightarrow A(a_j + ib_j) = (\alpha_j + i\beta_j)(a_j + ib_j). \tag{A46}$$

and α_j is ordered as $\alpha_1 \leq \alpha_2, \ldots$. In this section, we only consider the setting where all the eigenvalue and eigenvector are imaginary for notational simplicity. The extension to the general settings similar to Appendix E and the setting when is d is odd is straightforward.

We denote the eigenvalues and eigenvectors of H as $\{\lambda_j, v_j\}_{j=1}^d$ and v_js are linearly independent. In addition, we assume that $\lambda_1 \leq, \ldots, \lambda_d$. From this definition, by checking the real parts and complex parts, the following relations are derived

$$Aa_j = \alpha_j a_j - \beta b_j, \tag{A47}$$
$$Ab_j = \alpha_j b_j + \beta a_j. \tag{A48}$$

thus, by the skew-symmetric property

$$a_j^\top Aa_j + b_j^\top Ab_j = \alpha_j(\|a_j\|^2 + \|b_j\|^2) = \alpha_j \tag{A49}$$
$$= a_j^\top Ha_j + b_j^\top Hb_j, \tag{A50}$$

and in the third equality, we used the property

$$a_j^\top H^{1/2}JH^{1/2}a_j = b_j^\top H^{1/2}JH^{1/2}b_j = 0, \tag{A51}$$

since $H^{1/2}JH^{1/2}$ is a skew-symmetric matrix. Then, we expand a_j and b_j by v_j as

$$a_k = \sum_{j=1}^d a_k^\top v_j \tag{A52}$$

$$b_k = \sum_{j=1}^d b_k^\top v_j v_j, \tag{A53}$$

since v_js are eigenvalues of H, which can be used as the basis for $\mathbb{R}^d$. Then we substitute this into Equation (A50) and we have

$$\alpha_k = \sum_{j=1}^d \lambda_j (a_k^\top v_j)^2 + \sum_{j=1}^d \lambda_j (b_k^\top v_j)^2 \geq \lambda_1 \sum_{j=1}^d (a_k^\top v_j)^2 + (b_k^\top v_j)^2) = \lambda_1. \tag{A54}$$

This means that any real part of the eigenvalue of A is larger than λ_1 which is the smallest eigenvalue of H. Thus, if the α_1 is the smallest real part of the eigenvalue of A, that is larger than the smallest eigenvalue of H. This concludes the proof.

In the same way,

$$\alpha_k = \sum_{j=1}^{d} \lambda_j (a_k^\top v_j)^2 + \sum_{j=1}^{d} \lambda_j (b_k^\top v_j)^2 \leq \lambda_d \sum_{j=1}^{d} (a_k^\top v_j)^2 + (b_k^\top v_j)^2) = \lambda_d, \tag{A55}$$

which means any real part of the eigenvalues of A is smaller than the largest eigenvalue of H. Thus, if α is the largest real part of the eigenvalues of A, it is smaller than the largest eigenvalue of H.

Equality condition:

Next, we discuss when the equality holds for $\alpha_1 = \lambda_1$. First, we assume that eigenvalues of H are distinct, thus, there is only one eigenvector for λ_1. Later, we discuss if eigenvalues are not distinct. From Equation (A54), we have

$$\alpha_1 = \sum_{j=1}^{d} \lambda_j (a_1^\top v_j)^2 + \sum_{j=1}^{d} \lambda_j (b_1^\top v_j)^2 \geq \lambda_1 \sum_{j=1}^{d} (a_1^\top v_j)^2 + (b_1^\top v_j)^2) = \lambda_1, \tag{A56}$$

in general. Please note that if a_1 and b_1 does not correspond to v_1, then $\lambda_{j\neq 1} > \lambda_1$ must appear in the summation and equality never holds. So, the condition is

$$a_1, b_1 \propto v_1, \tag{A57}$$

must hold for the equality.

Based on this, let us assume that $w_1 = ca_1 + ic'b_1$ where $c^2 + c'^2 = 1$. We consider the case $a_1 = b_1 = v_1$. Then we need to solve the simultaneous equations

$$A(ca_1 + ic'b_1) = (\lambda_1 + i\beta_1)(ca_1 + ic'b_1) = (\lambda_1 c - c'\beta_1)v_1 + i(c\beta_1 + \lambda_1 c')v_1, \tag{A58}$$

this is obtained by the definition of the eigenvalue of A and

$$A(ca_1 + ic'b_1) = \lambda_1^{1/2} c(I\lambda_1^{1/2} + \alpha H^{1/2} J)v_1 + i\lambda_1^{1/2} c'(I\lambda_1^{1/2} + \alpha H^{1/2} J)v_1, \tag{A59}$$

this is obtained from the definition of eigenvalues of H. Then multiplying v_1 from the left, we obtain $c\beta_1 = 0$ and $c'\beta_1 = 0$. Thus, $\beta_1 = 0$. $\beta_1 = 0$ means$b_1 = 0$ from the property of the complex eigenvectors. Thus, we obtain $w_1 = a_1 = v_1$ for $\lambda_1 = \alpha_1$. Then, the following relation holds,

$$\lambda_1 v_1 = Av_1 = Hv_1 + \alpha H^{1/2} J H^{1/2} v_1 = \lambda_1 v_1 + \alpha \lambda_1^{1/2} H^{1/2} J v_1. \tag{A60}$$

Since $\lambda_1 \neq 0$ and $H^{1/2}$ has the inverse matrix, this condition indicates that

$$\alpha J v_1 = 0. \tag{A61}$$

This is the condition that $\lambda_1 = \alpha_1$ holds. The same relation can be derived for $\lambda_d = \alpha_d$.

Next, we assume that eigenvalues of H are not distinct. Let us denote the set of eigenvectors of the eigenvalue λ_1^0 as $\{v_1^0\}$. Please note that if a_1 and b_1 does not included in V_1^0, then $\lambda_{j\neq 1} > \lambda_1$ must appear and equality never holds. Thus

$$a_1, b_1 \in V_1^0 \tag{A62}$$

must hold for equality. Based on this, let us assume that $w_1 = ca_1 + ic'b_1$ where $c^2 + c^{'2} = 1$. We consider the case $a_1 \neq b_1$. Then

$$H^{-1/2}A(ca_1 + ic'b_1) = \lambda_1^{-1/2}(\lambda_1 + i\beta_1)(ca_1 + ic'b_1)$$
$$H^{-1/2}(H + \alpha H^{1/2}JH^{1/2})(ca_1 + ic'b_1) = \lambda_1^{1/2}c(I + \alpha J)a_1 + i\lambda_1^{1/2}c'(I + \alpha J)b_1, \quad \text{(A63)}$$

then we obtain the condition

$$\lambda_1 c\alpha J a_1 = -\beta_1 c' b_1 \quad \text{(A64)}$$
$$\lambda_1 c'\alpha J b_1 = \beta_1 c a_1. \quad \text{(A65)}$$

□

Appendix G. Proofs of Random Matrices

Appendix G.1. Proof of Theorem 5

Proof. The proof is the straightforward consequence of lemma in [43], that is

Lemma in ([43]) *If $f(x_1, \dots, x_m)$ is a polynomial in real variables $x_1, \dots, x_m$, which is not identically zero, then the subset $N_m = \{(x_1, \dots, x_m) | f(x_1, \dots, x_m) = 0\}$ of the Euclidean m-space $\mathbb{R}^m$ has the Lebesgue measure zero.*

We use this lemma to prove that the probability of $\lambda_1 = \alpha_1$ is 0 by showing that the probability mass of $\lambda_1 = \alpha_1$ has Lebesgue measure zero.

We use the same notation as in Appendix F. Recall Equation (A64), which is the condition of equality about $\lambda_1 = \alpha_1$. We express the elements of a_1 and b_1 as $a_1 = (a_1^1, \dots, a_1^d)^\top$ and $b_1 = (b_1^1, \dots, b_1^d)^\top$. Then the equality condition can be written as

$$\sum_{i=1}^{d}(\sum_{j=1}^{d} \lambda_1 c\alpha J_{i,j} a_1^j + \beta_1 c' b_1^i))^2 + \sum_{i=1}^{d}(\sum_{j=1}^{d} \lambda_1 c'\alpha J_{i,j} b_1^j - \beta_1 c a_1^i))^2 = 0. \quad \text{(A66)}$$

Then we define the polynomial about $\{J_{i.j}\}$

$$f(J_{1,2}, \dots, J_{d-1,d}) = \sum_{i=1}^{d}(\sum_{j=1}^{d} \lambda_1 c\alpha J_{ij} a_1^j + \beta_1 c' b_1^i))^2 + \sum_{i=1}^{d}(\sum_{j=1}^{d} \lambda_1 c'\alpha J_{ij} b_1^j - \beta_1 c a_1^i))^2. \quad \text{(A67)}$$

To apply lemma of [43], we must confirm that $f(J_{1,2}, \dots, J_{d-1,d})$ is not always 0. This is clear from the definition of f since we generate $J_{1,2}, \dots, J_{d-1,d}$ randomly from the distribution that is absolutely continuous with respect to Lebesgue measure and $\lambda_1 \neq 0$ and $c^2 + c'^2 = 1$ and either $a_1, b_1 \neq 0$.

Then, given an evaluation point x, from lemma of [43], the subset of $\{J_{i,j}\} \in \mathbb{R}^{d(d-1)/2}$ that satisfies $f(J_{1,2}, \dots, J_{d-1,d}) = 0$ has Lebesque measure zero. Thus, if we generate $\{J_{i,j}\}$ from the probability measure which is absolutely continuous with respect to Lebesque measure, (such as Gaussian distribution), $f(J_{1,2}, \dots, J_{d-1,d}) = 0$ holds probability 0. This concludes the proof. □

Appendix G.2. Proof of Lemma 1

Proof. We first discuss the condition about $\mathrm{Ker}J_0 = \{0\}$. Since $J = J_0 \otimes I_d$, and we denote the set of eigenvalues of J_0 as $\{\omega_i\}$. In general, the eigenvalues of the matrix that is composed of the Kronecker product with two matrices, e.g., A and B, are given as the product of each eigenvalue of A and B [44]. Thus, since J is the Kronecker product of J_0 and I_d, if J_0 does not have 0 as an eigenvalue, J does not have 0 as an eigenvalue.

Next, we discuss another equality condition. We use the similar notation as in Appendix F, but now the dimension of the matrix J is dN. We express the eigenvalue which has the smallest real part as λ_1^α and its eigenvector as $\omega_1^\alpha = a_1 + ib_1$. The elements of a_1 and

b_1 as $a_1 = (a_1^1, \dots, a_1^d, a_1^{d+1}, \dots, a_1^{dN})^\top \in \mathbb{R}^{dN}$ and $b_1 = (b_1^1, \dots, b_1^d, \dots, b_1^{d+1}, \dots, b_1^{dN})^\top$. We also express these as $a_1 = (a_1^{(1)}, \dots, a_1^{(N)})^\top \in \mathbb{R}^{dN}$ where $a_1^{(i)} = (a_1^{(i-1)d+1}, \dots, a_1^{id})^\top \in \mathbb{R}^d$.

We use the Kronecker product property:

$$Ja_1 = (J_0 \otimes I_d)a_1 = \left(\sum_{i=1}^N J_{0|i,1} a_1^{(i)}, \dots, \sum_{i=1}^N J_{0|i,N} a_1^{(i)} \right)^\top, \tag{A68}$$

where $J_{0|i,j}$ indicates the element of i-th row and j-th column of J_0 where we use the property of the Kronecker product and the Vec operator in the second equality [44].

The proof is almost similar to Appendix G.1. Then the equality condition can be written as

$$\sum_{n=1}^N \left\| \lambda_1 c\alpha \sum_i J_{0|i,n} a_1^{(i)} + \beta_1 c' b_1^{(n)} \right\|^2 + \sum_{n=1}^N \left\| \lambda_1 c'\alpha \sum_i J_{0|i,n} b_1^{(i)} + \beta_1 c a_1^{(n)} \right\|^2 = 0, \tag{A69}$$

where $\| \cdot \|$ is the d-dimensional Euclidean norm since $a_1^{(n)}, b_1^{(n)} \in \mathbb{R}^d$. Then we define the polynomial about $\{J_{i.j}\}$

$$f(J_{1,2}, \dots, J_{N-1,N}) = \sum_{n=1}^N \left\| \lambda_1 c\alpha \sum_i J_{0|i,n} a_1^{(i)} + \beta_1 c' b_1^{(n)} \right\|^2 + \sum_{n=1}^N \left\| \lambda_1 c'\alpha \sum_i J_{0|i,n} b_1^{(i)} + \beta_1 c a_1^{(n)} \right\|^2. \tag{A70}$$

In a similar discussion with Appendix G.1, it is clear that f is not always 0. Thus, given an evaluation point x, from lemma of [43], the subset of $\{J_{i,j}\} \in \mathbb{R}^{N(N-1)/2}$ that satisfies $f(J_{1,2}, \dots, J_{N-1,d}) = 0$ has Lebesque measure zero. Thus, if we generate $\{J_{i,j}\}$ from the probability measure which is absolutely continuous with respect to Lebesque measure, (such as Gaussian distribution), $f(J_{1,2}, \dots, J_{N-1,N}) = 0$ holds probability 0. This concludes the proof. □

Appendix G.3. Extending the Theorem to the Path

About Theorem 5 and Lemma 1, the statement holds true when we fix an evaluation point x. To ensure the acceleration, we need to extend Theorem 5 and Lemma 1 from a single evaluation point to the path of the stochastic process for S-LD, S-PLD, S-ULD, and S-PULD.

First, the condition of $\mathrm{Ker} J_0 = \{0\}$ is not related to the evaluation point. Thus, we need to consider the equality condition for $\mathrm{Re}\lambda_1^\alpha = \lambda_1^0$. As for this condition, as we had seen in Theorem 5 and Lemma 1, if we generate the random matrix J which is absolutely continuous with respect to Lebesgue measure, then the equality condition is not satisfied with probability 1 at the given evaluation point. The important point in those proof is to prove that the event when the equality holds has Lebesgue measure 0 at the given evaluation point using the lemma of [43].

Let us consider when two evaluation points are given (e.g., x_1, x_2), and we check whether the random matrix J satisfies the above equality condition or not. We can easily prove that at each evaluation point, such an event (we express them as S_1 and S_2) has Lebesgue measure 0 using the lemma of [43] (We refer to this as $P(S_1) = 0$ and $P(S_2) = 0$ where P is the law induced by generating the random matrix that has independent $d(d-1)/2$ elements). So, the volume of the event of sum of S_1 and S_2 are also 0 ($P(S_1 \bigcup S_2 = 0$). By repeating this procedure, when given a finite number of evaluation points, $(x_1, \dots, x_k)$, the sum of such probability is 0 (this indicates $P(S_1 \bigcup S_2, \dots, \bigcup S_k) = 0$).

When we consider the discretized dynamics of S-LD, S-PLD, and so on, and update samples up to k-iterations, then there exist k evaluation points. So, by applying the above discussion, we can ensure that along the path of the discretized dynamics, the equality condition does not hold with probability 1. On the other hand, as for the continuous dynamics, the evaluation point is infinite, thus when we cannot conclude that the probability that the equality does not hold is 1.

Appendix H. Proof of Theorem 6

We use the same notation as in Appendix F. We consider the expansion concerning α and we consider the following setting,

$$w_j := v_j + \delta v_j \tag{A71}$$

$$\mu_j := \lambda_j + \delta\lambda_j, \tag{A72}$$

which indicates that by introducing the skew-acceleration terms, the pairs of eigenvalues and eigenvectors of H' are expressed by the small perturbation for the eigenvalues and eigenvectors of H. Since $\{v_j\}_{j=1}^d$ are the eigenvalues of H and they can be used as an orthogonal basis, thus we expand δv by this basis. We obtain

$$\delta v_j = \sum_{k \neq j}^{d} c_{jk} v_k, \tag{A73}$$

where $c_{jk} = \delta v_j^\top v_k$.

Appendix H.1. Asymptotic Expansion When the Smallest Eigenvalue of $H(x)$ Is Positive

We work on the similar matrix of H', that is $H + \alpha V$ where $V := H^{1/2} J H^{1/2}$. See Appendix G.1 for the detail. Please note that this similar matrix only exists when the smallest eigenvalue of $H(x)$ is positive. Thus, the following discussion cannot apply to the case at the saddle point, where negative eigenvalues appear. We discuss the saddle point expansion later.

From the definition, we have

$$H' w_j = H w_j + \alpha V w_j = \mu_j w_j = (\lambda_j + \delta\lambda_j)(v_j + \delta v_j), \tag{A74}$$

We rearrange this equation as

$$H v_j + H\delta v_j + \alpha V v_j + \alpha V \delta v_j = \lambda_j v_j + \delta\lambda_j v_j + \lambda_j \delta v_j + \delta\lambda_j \delta v_j. \tag{A75}$$

First, we focus on the first-order expansion. This means we neglect high-order terms. Then, we have

$$H v_j + H\delta v_j + \alpha V v_j = \lambda_j v_j + \delta\lambda_j v_j + \lambda_j \delta v_j. \tag{A76}$$

By multiplying v_j to Equation (A76) from the left-hand side, we have

$$\lambda_j + \lambda_j v_j^\top \delta v_j + \alpha v_j^\top V v_j = \lambda_j + \delta\lambda_j + \lambda_j v_j^\top \delta v_j, \tag{A77}$$

Since $v_j^\top V v_j = 0$ due to the skew-symmetric property of V. Thus, we have

$$\delta\lambda_j = 0, \tag{A78}$$

up to the first-order expansion. Then we substitute this into Equation (A76) and multiplying v_i where $i \neq j$, we have

$$\lambda_i c_{ji} + \alpha v_i^\top V v_j = \lambda_j c_{ji}. \tag{A79}$$

Then we have

$$c_{ji} = \frac{\alpha v_i^\top V v_j}{\lambda_j - \lambda_i}. \tag{A80}$$

Then we obtain

$$\delta v_j = \alpha \sum_{i \neq j}^{d} \frac{v_i^\top V v_j}{\lambda_j - \lambda_i} v_i. \tag{A81}$$

We substitute this into Equation (A75), and multiplying $v_j^\top$, we have

$$\begin{aligned} & v_j^\top H \alpha \sum_{i \neq j}^{d} \frac{v_i^\top V v_j}{\lambda_j - \lambda_i} v_i + \alpha v_j^\top V v_j + \alpha v_j^\top V \alpha \sum_{i \neq j}^{d} \frac{v_i^\top V v_j}{\lambda_j - \lambda_i} v_i \\ & = \delta\lambda_j v_j^\top v_j + \lambda_j v_j^\top \alpha \sum_{i \neq j}^{d} \frac{v_i^\top V v_j}{\lambda_j - \lambda_i} v_i + \delta\lambda_j v_j^\top \alpha \sum_{i \neq j}^{d} \frac{v_i^\top V v_j}{\lambda_j - \lambda_i} v_i. \end{aligned} \tag{A82}$$

Since $v_j^\top V v_j = 0$ and $v_j^\top v_i = 0$ and $v_j^\top v_j = 1$, we have

$$\alpha^2 \sum_{i \neq j}^{d} \frac{v_i^\top V v_j}{\lambda_j - \lambda_i} v_j^\top V v_i = \delta\lambda_j. \tag{A83}$$

Thus, we have

$$\mu_j - \lambda_j = \alpha_j + i\beta_j - \lambda_j = -\alpha^2 \sum_{i \neq j}^{d} \frac{(v_i^\top V v_j)^2}{\lambda_j - \lambda_i}. \tag{A84}$$

Thus, by taking the real part, and note that $\mathrm{Re}\lambda_j(\alpha) = \alpha_j$, we have

$$\mathrm{Re}\lambda_j(\alpha) - \lambda_j = \alpha^2 \mathrm{Re} \sum_{i \neq j}^{d} \frac{(v_i^\top V v_j)^2}{\lambda_i - \lambda_j} + \mathcal{O}(\alpha^3) = \alpha^2 \sum_{i \neq j}^{d} \frac{\lambda_i \lambda_j (v_i^\top J v_j)^2}{\lambda_i - \lambda_j} + \mathcal{O}(\alpha^3). \tag{A85}$$

This concludes the proof.

Appendix H.2. Expansion of the Eigenvalue at the Saddle Point

Here we derive the formula of the expansion of the eigenvalue at the saddle point. Since the smallest eigenvalue is negative, we cannot use the similar matrix as shown above. Instead, we use the relation,

$$\mu_j H w_j = H \mu_j w_j = H(I + \alpha J) H w_j \tag{A86}$$

where we used the definition of the eigenvalues and eigenvectors. Here, we express $H' := (I + \alpha J)H$ and its pairs of eigenvalues and eigenvectors as $\{(\mu_i, w_i)\}_{i=1}^d$. As introduced in the above, we substitute the expansion to Equation (A86), then we obtain

$$(\lambda_j + \delta\lambda_j) H (v_j + \delta v_j) = H(I + \alpha J) H (v_j + \delta v_j) \tag{A87}$$

Then, in the same way as above, since $\{v_j\}_{j=1}^d$ are the eigenvalues of H and they can be used as an orthogonal basis, we expand δv by this basis. This means

$$\delta v_j = \sum_{k=1}^{d} c_{jk} v_k, \tag{A88}$$

where $c_{jk} = \delta v_j^\top v_k$. By multiplying v_i to Equation (A87) where $i \neq j$ from left-hand side and neglecting high-order terms, we have

$$c_{ji} = \frac{\lambda_j}{\lambda_j - \lambda_i} (v_i^\top \alpha J v_i). \tag{A89}$$

Next, Then by multiplying v_j to Equation (A87) from left-hand side, we have

$$v_j H(\alpha J) H \delta v_j = (\delta\lambda_j)(\lambda_j + \lambda_j v_j^\top \delta v_j) \tag{A90}$$

Then by substituting δv_j with coefficient Equation (A89), we have

$$\delta\lambda_j = \alpha^2 \sum_{i \neq j}^{d} \frac{\lambda_i \lambda_j (v_i^\top J v_j)^2}{\lambda_i - \lambda_j} + \mathcal{O}(\alpha^3) \tag{A91}$$

This concludes the proof.

Appendix I. Convergence Rate of Parallel Sampling Schemes

Appendix I.1. Proof of Lemma 2

First, we introduce the notations. We express the random variables of S-PLD as $Y_t^{\otimes N}$. We express the measure induced by S-PLD as $\mu_t^{\otimes N}(\alpha)$, which uses the αJ as an interaction term. Thus, we express the measure of PLD as $\mu_{kh}^{\otimes N}(0)$, we can decompose the measure as marginals. We also denote the marginal measure of S-PLD for $Y_t^{(n)}$ $\nu_t^{(n)}(\alpha)$. Please note that initial distribution is $\mu_0^{\otimes N}$ and its marginals are μ_0 as defined in Assumption 4.

Please note that the marginal measure of PLD is the same as those of LD if the initial measures are all the same, thus each marginal satisfy the Poincaré constant m_0. This is also the result of the tensorization property of the spectral gap (Proposition 4.3.1 in Bakry et al. [19]).

As for the initial condition, from the fact that χ^2 divergence is the special case of Renyi divergence ($\alpha = 4$), and from the tensorization property of the Renyi divergence (see Theorem 28 in [45]), we have

$$\chi^2(\mu_t^{\otimes N}(0), \pi^{\otimes N}) \leq e^{-2\beta^{-1} m_0 t} \chi^2(\mu_0^{\otimes N}, \pi^{\otimes N}) = \sum_{n=1}^{N} e^{-2\beta^{-1} m_0 t} \chi^2(\mu_0, \pi). \tag{A92}$$

Then we have

$$\chi^2(\mu_t^{\otimes N}(0), \pi^{\otimes N}) \leq e^{-2\beta^{-1} m_0 t} \chi^2(\mu_0^{\otimes N}, \pi^{\otimes N}) = N e^{-2\beta^{-1} m_0 t} \chi^2(\mu_0, \pi). \tag{A93}$$

If the skew acceleration is applied, from the same discussion as S-LD (see Appendix C.1), S-PLD has the Poincaré constant which is larger than m_0. We express it as $m(\alpha, N) (\geq m_0)$. Then we have

$$\chi^2(\mu_t^{\otimes N}(\alpha), \pi^{\otimes N}) \leq N e^{-2\beta^{-1} m(\alpha,N) t} \chi^2(\mu_0, \pi). \tag{A94}$$

At first, since there exists a constant N in the convergence bound, this bound seems not useful. However, as we discussed below, when we bound the bias or variance, these bound is meaningful. For example, let us consider approximating the true expectation $\int f(x) d\pi(x)$ by the ensemble samples $\frac{1}{N}\sum_{n=1}^{N} f(X_t^{(n)})$. Then we are interested in bounding the error

$$\left| \mathbb{E} \frac{1}{N} \sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi \right|. \tag{A95}$$

For this purpose, we can bound this by 2-Wasserstein distance as

$$\left| \mathbb{E} \frac{1}{N} \sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi \right|, \leq \frac{L_f}{\sqrt{N}} W_2(\mu_{kh}^{\otimes N}(\alpha), \pi^{\otimes N}) \tag{A96}$$

where we assumed that f shows L_f lipschitzness and used the fact that $\frac{1}{N}\sum_{n=1}^{N} f(x^{(n)})$ shows $L_f/\sqrt{N}$ lipschitzness.

To bound the distance, we use the basic relation

$$W_2^2(\nu_{kh}(\alpha), \pi^{\otimes N}) \leq 2\frac{1}{m(\alpha,N)}\chi^2(\mu_{kh}^{\otimes N}(\alpha), \pi^{\otimes N}), \tag{A97}$$

where $m(\alpha, N)$ is the Poincaré constant. This is established by the definition of Wasserstein distance and χ^2-divergence, see [46] for the detail. Then combined with above relations, we obtain the bias bound of S-PLD as

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq L_f\sqrt{\frac{2}{m(\alpha,N)}}e^{-\beta^{-1}m(\alpha,N)kh}\chi^2(\mu_0,\pi)^{1/2}. \tag{A98}$$

In the same way, we obtain the bias bound of PLD as

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq L_f\sqrt{\frac{2}{m_0}}e^{-\beta^{-1}m_0 kh}\chi^2(\nu_0,\pi)^{1/2}. \tag{A99}$$

Thus, while the explicit dependency on N disappeared, but S-PLD shows faster convergence through the relation of $m(\alpha, N) \geq m_0$. Moreover, if we use the skew matrices, which does not satisfy the equality condition, we have $m(\alpha, N) > m_0$.

Appendix I.2. Proof for S-ULD

We can characterize the convergence rate almost in the same way as Appendix C.2. The derivation is the same above, thus we only show the result

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq L_f\sqrt{\frac{2}{m(\alpha,N)}}\sqrt{\frac{1+\bar{\epsilon}}{1-\bar{\epsilon}}}e^{-\lambda_\gamma/2kh}\chi^2(\nu_0^0,\pi)^{1/2}. \tag{A100}$$

where $\bar{\epsilon}$ and λ_γ is given as follows.

$$\lambda_\gamma = \frac{\Lambda(\gamma, \bar{\epsilon}\min(\gamma,\gamma^{-1}))}{1+\bar{\epsilon}\min(\gamma,\gamma^{-1})}, \tag{A101}$$

and

$$\Lambda(\gamma,\epsilon) = \frac{\gamma\Sigma^{-1} - \frac{1}{1+\frac{m_0\Sigma^{-1}}{\beta}}}{2} - \frac{1}{2}\sqrt{(S_{--}-S_{++})^2 + (S_{-+})^2}, \tag{A102}$$

$$S_{--} = \epsilon\lambda_{ham}, \tag{A103}$$

$$S_{-+} = -\epsilon(R_{ham} + \gamma\Sigma^{-1}/2), \tag{A104}$$

$$S_{++} = \gamma\Sigma^{-1} - \epsilon, \tag{A105}$$

$$\lambda_{ham} = 1 - \left(1 + \frac{m(\alpha,N)\Sigma^{-1}}{\beta}\right)^{-1}, \tag{A106}$$

$$\epsilon = \bar{\epsilon}\min(\gamma,\gamma^{-1}), \tag{A107}$$

where $\bar{\epsilon}$ is arbitrary sufficiently small positive value such that $\Lambda(\gamma, \bar{\epsilon}\min(\gamma,\gamma^{-1})) > 0$ is satisfies. and

$$R_{ham} \leq \sqrt{\max\{M,2\}}. \tag{A108}$$

Appendix J. Proof of Theorem 7

We show our theorem again with explicit constants

Theorem A3. *Under Assumptions 1–7, for any $k \in \mathbb{N}$ and any $h \in (0, 1 \wedge \frac{m}{4M^2})$ obeying $kh \geq 1$ and $\beta m \geq 2$, we have*

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq L_f\sqrt{\tilde{C}_0^2\sqrt{\delta} + \tilde{C}_1^2\sqrt{h}}k\eta + L_f\sqrt{\frac{2}{m(\alpha,N)}}\chi^2(\mu_0,\pi)^{1/2}e^{-\beta^{-1}m(\alpha,N)kh}. \quad \text{(A109)}$$

where

$$\tilde{C}_0^2 = \left(12 + 8\left(\kappa_0 + 2b + \frac{2d}{\beta}\right)\right)\left(\beta C_0 + \sqrt{\beta C_0}\right), \quad \text{(A110)}$$

$$\tilde{C}_1^2 = \left(12 + 8\left(\kappa_0 + 2b + \frac{2d}{\beta}\right)\right)\left(C_1 + \sqrt{C_1}\right) \quad \text{(A111)}$$

$$C_0 = (1+\alpha)^2\left(M^2\left(\kappa_0 + 2\left(1 \vee \frac{1}{m}\right)\left(b + 2(1+\alpha)^2B^2 + \frac{d}{\beta}\right)\right) + B^2\right), \quad \text{(A112)}$$

$$C_1 = 6(1+\alpha^2)M^2(\beta C_0 + d), \quad \text{(A113)}$$

Then obtained bound is $\mathcal{O}(kh \cdot h^{1/4})$, which is independent of N. Thus, this result is much better than those in [18]. Additionally, note that we can derive the similar bias bound for skew-SGHMC in the same way as skew-SGLD.

Proof. For notational simplicity, we express the random variables of skew-SGLD which uses the αJ as an interaction term as $X_k^{\otimes N}$ and those of S-PLD as $Y_k^{\otimes N}$. In this section, for simplicity, we express them as X_k and Y_k. We denote the measure of X_k and Y_k as $\nu_{kh}^{\otimes N}$ and $\mu_{kh}^{\otimes N}$. We also denote the marginal measure of $X_k^{(n)}$ and $Y_k^{(n)}$ as $\mu_{kh}^{(n)}$ and $\nu_{kh}^{(n)}$.

Then, we first decompose the bias as

$$\begin{aligned}
&\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \\
&= \left|\mathbb{E}\frac{\sum_{n=1}^{N} f(X_k^{(n)})}{N} - \mathbb{E}\frac{\sum_{n=1}^{N} f(Y_k^{(n)})}{N} + \mathbb{E}\frac{\sum_{n=1}^{N} f(Y_k^{(n)})}{N} - \int_{\mathbb{R}^d} f d\pi\right| \\
&\leq \left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(Y_k^{(n)})\right| + \left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(Y_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \\
&\leq \frac{L_f}{N}\sum_{i=1}^{N} W_2(\nu_{kh}^{(n)}(\alpha), \mu_{kh}^{(n)}(\alpha)) + \frac{L_f}{\sqrt{N}}\underbrace{W_2(\mu_{kh}^{\otimes N}(\alpha), \pi^{\otimes N})}_{(i)},
\end{aligned} \quad \text{(A114)}$$

where we used the Jensen inequality for the first term in the last inequality and we move $\frac{1}{N}\sum_{i=1}^{N}$ outside the $|\cdot|$. In addition, each expectation only depends on the marginal measures $\mu^{(i)}$ in the first term and we use the property of the 2-Wasserstein (2-W) distance. Furthermore, we decompose the first term as

$$\frac{L_f}{N}\sum_{n=1}^{N} W_2(\mu_{kh}^{(n)}(\alpha), \nu_{kh}^{(n)}(\alpha)) \leq \frac{L_f}{N}\left(\sum_{n=1}^{N}\underbrace{W_2(\nu_{kh}^{(n)}(\alpha), \mu_{kh}^{(n)}(0))}_{(ii)} + \underbrace{W_2(\mu_{kh}^{(n)}(\alpha), \mu_{kh}^{(n)}(0))}_{iii)}\right), \quad \text{(A115)}$$

where $\mu_{kh}^{(n)}(0)$ denotes the measure induced by PLD, which is the naive parallel sampling without a skew-symmetric interaction.

In conclusion, our task is to bound each (i), (ii), (iii) terms in the above. Bounding (i) is already discussed in Appendix I.1.

Next, we work on (ii) and(iii). Following [10], we use weighted CKP inequality to bound the 2-W distance. From Bolley and Villani [47], using the weighted CKP inequality, we can bound each 2-W distance by the relative entropy (KL divergence). This weighted CKP inequality indicates that

$$W_2(\nu_{kh}^{(n)}(\alpha),\mu_{kh}^{(n)}(0))\leq C_{\mu_{kh}^{(n)}(0)}\left(\mathrm{KL}(\nu_{kh}^{(n)}(\alpha)|\mu_{kh}^{(n)}(0))^{1/2}+\left(\frac{\mathrm{KL}(\nu_{kh}^{(n)}(\alpha)|\mu_{kh}^{(n)}(0))}{2}\right)^{1/4}\right),\quad \text{(A116)}$$

with

$$C_{\mu_{kh}^{(n)}(0)}=2\inf_{\lambda>0}\left(\frac{1}{\lambda}\left(\frac{3}{2}+\log\int_{\mathbb{R}^d}e^{\lambda\|x^{(n)}\|^2}d\mu_{kh}^{(n)}(0)\right)\right)^{1/2}.\quad \text{(A117)}$$

and

$$W_2(\mu_{kh}^{(n)}(\alpha),\mu_{kh}^{(n)}(0))\leq C_{\mu_{kh}^{(n)}(0)}\left(\mathrm{KL}(\mu_{kh}^{(n)}(\alpha)|\mu_{kh}^{(n)}(0))^{1/2}+\left(\frac{\mathrm{KL}(\mu_{kh}^{(n)}(\alpha)|\mu_{kh}^{(n)}(0))}{2}\right)^{1/4}\right),\quad \text{(A118)}$$

with

$$C_{\mu_{kh}^{(n)}(0)}=2\inf_{\lambda>0}\left(\frac{1}{\lambda}\left(\frac{3}{2}+\log\int_{\mathbb{R}^d}e^{\lambda\|x^{(n)}\|^2}d\mu_{kh}^{(n)}(0)\right)\right)^{1/2}.\quad \text{(A119)}$$

We point out that using $C_{\mu_{kh}^{(i)}(0)}$ not $C_{\nu_{kh}^{(i)}(\alpha)}$ and $C_{\mu_{kh}^{(i)}(\alpha)}$ in weighted CKP inequality is important. This is because since $\mu_{kh}^{(i)}(0)$ is the constant based on the parallel-chain Monte Carlo without skew-symmetric term, thus the parallel chain can be decomposed each independent chains. Thus, $C_{\mu_{kh}^{(i)}}$ actually does not depend on i and it does not depend on N and shows $\mathcal{O}(d)$ dependency. However, $C_{\nu_{kh}^{(i)}(\alpha)}$ and $C_{\mu_{kh}^{(i)}(\alpha)}$ show $\mathcal{O}(dN)$ which shows linear dependency on N since there is an interaction term between parallel chains and we cannot decompose the parallel chain easily. Thus, this results in unsatisfactory dependency on N. This is the reason we introduced $\mu_{kh}^{(i)}(0)$ in our theoretical analysis.

Please note that since $\mu_{kh}^{(n)}(0)$ is induced by the naive parallel chain, each marginal is independent with each other and takes the same measure if the initial measure is the same. Thus, $\mu_{kh}^{(1)}(0)=\cdots=\mu_{kh}^{(N)}(0)$. From now on, we express the marginal as $\mu_{kh}(0)$ for simplicity. Thus, $C_{\mu_{kh}^{(1)}(0)}=\cdots=C_{\mu_{kh}^{(N)}(0)}=C_{\mu_{kh}(0)}$.

Then substituting the above WKP inequalities and using the Jensen inequality, we obtain

$$\begin{aligned}&\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N}f(X_k^{(n)})-\mathbb{E}\frac{1}{N}\sum_{n=1}^{N}f(Y_k^{(n)})\right|\\&\leq L_fC_{\mu_{kh}(0)}\frac{1}{N}\sum_{n=1}^{N}\left(\mathrm{KL}(\nu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))^{1/2}+\left(\frac{\mathrm{KL}(\nu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))}{2}\right)^{1/4}\right.\\&\qquad\left.+\mathrm{KL}(\mu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))^{1/2}+\left(\frac{\mathrm{KL}(\mu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))}{2}\right)^{1/4}\right)\end{aligned}$$

$$\leq L_f C_{\mu_{kh}(0)} \left(\left(\sum_{n=1}^{N} \frac{\mathrm{KL}(\nu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))}{N} \right)^{\frac{1}{2}} + \left(\sum_{n=1}^{N} \frac{\mathrm{KL}(\nu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))}{2N} \right)^{\frac{1}{4}} \right.$$
$$\left. + \left(\sum_{n=1}^{N} \frac{\mathrm{KL}(\mu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))}{N} \right)^{\frac{1}{2}} + \left(\sum_{n=1}^{N} \frac{\mathrm{KL}(\mu_{kh}^{(n)}(\alpha)|\mu_{kh}(0))}{2N} \right)^{\frac{1}{4}} \right). \quad \text{(A120)}$$

To analyze the discretization error, we use the following key lemma:

Lemma A1. *Assume that there exist random variables* $\{X_i \in \Omega_i\}_{i=1}^N$ *and* $\{Y_i \in \Omega_i\}_{i=1}^N$*. We denote the product space as* $\Omega^{\otimes N} := \Omega_1 \times \dots \Omega_N$*. Let us introduce* $X = (X_1, \dots, X_N) \in \Omega^{\otimes N}$ *and* $Y = (Y_1, \dots, Y_N) \in \Omega^{\otimes N}$*. Let us express their joint probability measures as expressed as* $P(X) := P(X_1, \dots, X_N)$*,* $Q(Y) := Q(Y_1, \dots, Y_N)$*, let us denote the marginal measures of each Xs and Ys as* $\{P_i(X_i)\}_{i=1}^N$ *and* $\{Q_i(Y_i)\}_{i=1}^N$*. If* $P_i << Q_i$ *holds, we have*

$$\sum_{i=1}^{N} \mathrm{KL}(P_i(X_i)\|Q_i(Y_i)) \leq \mathrm{KL}(P(X)\|Q(Y)), \quad \text{(A121)}$$

A proof is given in Appendix J.1. We apply this lemma as

$$\sum_{n=1}^{N} \mathrm{KL}(\mu_{kh}^{(n)}|\mu_{kh}(0)) \leq \mathrm{KL}(\nu_{kh}^{\otimes N}|\mu_{kh}^{\otimes N}(0)), \quad \text{(A122)}$$
$$\sum_{n=1}^{N} \mathrm{KL}(\mu_{kh}^{(n)}(\alpha)|\mu_{kh}(0)) \leq \mathrm{KL}(\mu_{kh}^{\otimes N}(\alpha))|\mu_{kh}^{\otimes N}(0))). \quad \text{(A123)}$$

Combining these results with the above bias bound, we obtain

$$\left| \mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(Y_k^{(n)}) \right|$$
$$\leq L_f C_{\mu_{kh}(0)} \left(\left(\frac{\mathrm{KL}(\nu_{kh}^{\otimes N}(\alpha)|\mu_{kh}^{\otimes N}(0))}{N} \right)^{\frac{1}{2}} + \left(\frac{\mathrm{KL}(\nu_{kh}^{\otimes N}(\alpha)|\mu_{kh}^{\otimes N}(0))}{2N} \right)^{\frac{1}{4}} \right.$$
$$\left. + \left(\frac{\mathrm{KL}(\mu_{kh}^{\otimes N}(\alpha)(\alpha)|\mu_{kh}^{\otimes N}(0))}{N} \right)^{\frac{1}{2}} + \left(\frac{\mathrm{KL}(\mu_{kh}^{\otimes N}(\alpha)|\mu_{kh}^{\otimes N}(0))}{2N} \right)^{\frac{1}{4}} \right). \quad \text{(A124)}$$

Thus, we need to bound $\mathrm{KL}(\mu_{kh}^{(i)}(\alpha)|\mu_{kh}^{\otimes N}(0))$ and $\mathrm{KL}(\nu_{kh}^{\otimes N}(\alpha)|\mu_{kh}^{\otimes N}(0))$ and $C_{\mu_{kh}(0)}$. We can upper-bound them using the results of [2]. For that purpose, we need to replace the constants in [2] as we show in the below. Here, we discuss how the constants in the assumption are changed in the ensemble scheme. We define

$$\nabla u^{\otimes N}(x^{\otimes N}) := (\nabla u(x^{(1)}), \dots, \nabla u(x^{(N)})) \quad \text{(A125)}$$

First, we focus on the smoothness condition. From Assumption 2 and lemma 8 in [18], we have

$$\|(I+\alpha J)\nabla u^{\otimes N}(x^{\otimes N}, z) - (I+\alpha J)\nabla u^{\otimes N}(y^{\otimes N}, z))\| \leq M(1+\alpha)\|x^{\otimes N} - y^{\otimes N}\|. \quad \text{(A126)}$$

where the norm in the right-hand side is the Euclidean norm in $\mathbb{R}^{dN}$.

Next, we discuss the smoothness condition. Define $\nabla U_\alpha(x^{\otimes N}) := \nabla U^{\otimes N}(x^{\otimes N}) + \alpha J \nabla U^{\otimes N}(x^{\otimes N})$. Then, Let $x^{\otimes N} \in \mathbb{R}^{dN}$ and under the assumptions 1 to 6, we have

$$x^{\otimes N} \cdot \nabla U_\alpha(x^{\otimes N}) \geq m\|x^{\otimes N}\|^2 - bN. \quad \text{(A127)}$$

Next, we check about the condition of the drift function at the origin: $\|\nabla u(0,z)\| \leq B$. We can calculate in the same way as the smoothness condition. Then we have

$$\|(I+\alpha J)\nabla U^{\otimes N}(0^{\otimes N})\| \leq B\sqrt{N}(1+\alpha). \tag{A128}$$

Next, we study the condition about the stochastic gradient: $\mathbb{E}[\|\nabla \hat{U}(x) - \nabla U(x)\|^2] \leq 2\delta\big(M^2\|x\|^2 + B^2\big)$. This can be easily modified to

$$\begin{aligned}
&\mathbb{E}[\|(I+\alpha J)\nabla \hat{U}^{\otimes N}(x^{\otimes N}) - (I+\alpha J)\nabla U^{\otimes N}(x^{\otimes N})\|^2] \\
&\leq (1+\alpha)^2\mathbb{E}[\nabla \hat{U}^{\otimes N}(x^{\otimes N}) - \nabla U^{\otimes N}(x^{\otimes N})\|^2] \\
&\leq (1+\alpha)^2\sum_{i=1}^{N}\mathbb{E}[\nabla \hat{U}(x^{(i)}) - \nabla U(x^{(i)})\|^2] \\
&\leq (1+\alpha)^2\sum_{i=1}^{N}2\delta\Big(M^2\|x^{(i)}\|^2 + B^2\Big) \\
&\leq 2\delta(1+\alpha)^2\Big(M^2\|x^{\otimes N}\|^2 + NB^2\Big).
\end{aligned} \tag{A129}$$

Finally, we discuss about the initial condition: $\kappa_0 := \log\int_{\mathbb{R}^d} e^{\|x\|^2}p_0(x)dx < \infty$. We assume that the initial probability distribution is $\mu_0^{\otimes N}(X_0^{\otimes N}) = \mu_0(X_0^{(1)}) \times \cdots \times \mu_0(X_0^{(N)})$, which means that all the marginal probability is the same. Then

$$\kappa_0^{\otimes N} := \log\int_{\mathbb{R}^{dN}} e^{\|x^{\otimes N}\|^2}\mu_0^{\otimes N}(x^{\otimes N})dx^{\otimes N} = \log\prod_{n=1}^{N}\Big(\int_{\mathbb{R}^d} e^{\|x^{(n)}\|^2}\mu_0(x^{(n)})dx\Big) = N\kappa_0. \tag{A130}$$

In this way, the constants in the assumptions are modified and expressed with N and α. Then combined with the results of [2], we can derive the following relations

$$C_{\nu_{kh}(0)} \leq 12 + 8\Big(\kappa_0 + 2b + \frac{2d}{\beta}\Big), \tag{A131}$$

$$\mathrm{KL}(\nu_{kh}^{\otimes N}|\mu_{kh}^{\otimes N}(0)) \leq N(C_0\beta\delta + C_1\eta)k\eta, \tag{A132}$$

$$\mathrm{KL}(\mu_{kh}^{\otimes N}(\alpha)|\mu_{kh}^{\otimes N}(0)) \leq N\frac{\beta}{2}\alpha^2M^2(\kappa_0 + \frac{b+d/\beta}{m})k\eta, \tag{A133}$$

where

$$C_0 = (1+\alpha)^2\bigg(M^2\bigg(\kappa_0 + 2\Big(1\vee\frac{1}{m}\Big)\Big(b + 2(1+\alpha)^2B^2 + \frac{d}{\beta}\Big)\bigg) + B^2\bigg), \tag{A134}$$

$$C_1 = 6(1+\alpha^2)M^2(\beta C_0 + d). \tag{A135}$$

This concludes the proof. □

Appendix J.1. Proof of Lemma A1

Proof. We prove this lemma using the Donsker–Varadhan representation of the relative entropy [48]. The relative entropy admits the dual representation as:

$$\mathrm{KL}(P(X)\|Q(Y)) = \sup_{T:\Omega^{\otimes N}\to\mathbb{R}} \mathbb{E}_{P(X)}[T] - \log\mathbb{E}_{Q(Y)}[e^T], \tag{A136}$$

where supremum is taken over all function T of which the expectation of e^T and T are finite. We then restrict the function class into a class $\mathcal{F}(T) = \{T(X)|\exists T_i : \Omega_i \to \mathbb{R}, s.t. T(X) = \sum_{i=1}^N T_i(X_i)\}$ where each expectation of e^{T_i} and T_i are finite. Then by definition,

$$\mathrm{KL}(P(X)\|Q(Y)) = \sup_{T:\Omega\to\mathbb{R}} \mathbb{E}_{P(X)}[T] - \log\mathbb{E}_{Q(Y)}[e^T] \geq \sup_{T\in\mathcal{F}}\mathbb{E}_{P(X)}\Big[\sum_i T_i\Big] - \log\mathbb{E}_{Q(Y)}\Big[e^{\sum_i T_i}\Big]. \tag{A137}$$

Then we have

$$\begin{aligned}
\mathrm{KL}(P(X)\|Q(Y)) &\geq \sup_{T\in\mathcal{F}} \sum_i \mathbb{E}_{P_i(X_i)}[T_i] - \log \prod_i \mathbb{E}_{Q_i(Y_i)}\left[e^{T_i}\right] \\
&= \sup_{T\in\mathcal{F}} \sum_i \Big(\mathbb{E}_{P_i(X_i)}[T_i] - \log \mathbb{E}_{Q_i(Y_i)}\left[e^{T_i}\right]\Big) \\
&= \sum_i \sup_{T_i:\Omega_i\to\mathbb{R}} \mathbb{E}_{P_i(X_i)}[T_i] - \log \mathbb{E}_{Q_i(Y_i)}\left[e^{T_i}\right] \\
&= \sum_{i=1}^N \mathrm{KL}(P_i(X_i)\|Q_i(Y_i)). \qquad \text{(A138)}
\end{aligned}$$

□

Appendix K. Order Expansion

Bias Expansion for S-PLD

Recall that the bias of S-PLD is

$$\begin{aligned}
&\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^N f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \\
&\leq L_f\sqrt{\tilde{C}_0^2\sqrt{\delta} + \tilde{C}_1^2\sqrt{h}}k\eta + L_f\sqrt{\frac{2}{m(\alpha,N)}}\chi^2(\mu_0),\pi)^{1/2} e^{-\beta^{-1}m(\alpha,N)kh}. \qquad \text{(A139)}
\end{aligned}$$

where

$$\tilde{C}_0^2 = \left(12 + 8\left(\kappa_0 + 2b + \frac{2d}{\beta}\right)\right)\left(\beta C_0 + \sqrt{\beta C_0}\right), \qquad \text{(A140)}$$

$$\tilde{C}_1^2 = \left(12 + 8\left(\kappa_0 + 2b + \frac{2d}{\beta}\right)\right)\left(C_1 + \sqrt{C_1}\right) \qquad \text{(A141)}$$

$$C_0 = (1+\alpha)^2\left(M^2\left(\kappa_0 + 2\left(1\vee\frac{1}{m}\right)\left(b + 2(1+\alpha)^2B^2 + \frac{d}{\beta}\right)\right) + B^2\right), \qquad \text{(A142)}$$

$$C_1 = 6(1+\alpha^2)M^2(\beta C_0 + d), \qquad \text{(A143)}$$

First, we discuss the convergence of the continuous dynamics. Using the eigenvalue expansion in Theorem 6 , with some positive constant d_0, we have

$$m(\alpha, N) \approx m_0 + \alpha^2 d_0 + \mathcal{O}(\alpha^3). \qquad \text{(A144)}$$

Then by assuming α^2 is small enough and considering the Tayler expansion, we have

$$L_f\sqrt{\frac{2}{m(\alpha,N)}}\chi^2(\mu_0,\pi)^{1/2}e^{-\beta^{-1}m(\alpha,N)t} \approx L_f\chi^2(\mu_0,\pi)^{1/2}\sqrt{2}\left(\frac{1}{\sqrt{m_0}} - \frac{d_0}{2m_0^{3/2}}\alpha^2\right)e^{-\beta^{-1}m_0 t}. \qquad \text{(A145)}$$

As for the discretization and stochastic gradient error, using the Taylor expansion, there exists a positive constant d_1 and d_2, such that

$$L_f\sqrt{\tilde{C}_0^2\sqrt{\delta} + \tilde{C}_1^2\sqrt{h}}k\eta \approx (d_1\alpha + d_2\alpha^2 + \mathrm{Const})kh. \qquad \text{(A146)}$$

Combining these terms, we have

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^N f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq (d_1\alpha + d_2\alpha^2)kh - \alpha^2 L_f\chi^2(\mu_0,\pi)^{1/2}\frac{1}{\sqrt{2}m_0^{3/2}}e^{-\beta^{-1}m_0 t} + \mathrm{Const}. \qquad \text{(A147)}$$

Thus, there exists an optimal α^*, which minimizes the bias. Please note that at $k = 0$, acceleration always occurs. As k goes to infinity, the second third terms 0, thus the first term will be dominant, which means we have larger discretization and stochastic gradient error.

Appendix L. Hyperparameters of the Proposed Algorithm

Here we discuss how to set hyperparameters in the algorithm. There are three hyperparameters, α_0, η, and c. We numerically found that setting $c = 0.95$ work well for real dataset including LDA experiment, and Bayesian neural network regression and classification. For toy dataset, we set $c = 0.9$.

As for α_0 and η, we empirically found that using the following scaling trick works well for real dataset including LDA experiment, and Bayesian neural network regression and classification,

$$\alpha_0 \approx \frac{1}{\sqrt{\frac{1}{N^2}\sum_n \nabla U(x_0^{(n)})^2}} Nh. \tag{A148}$$

and using $\eta \approx 0.1\alpha_0$. The intuition is that the magnitude of the gradient can be very different in each dimension, so we introduce the scaling by the gradient. We also multiply h so that the stochastic gradient and discretization error of the skew term will not be dominant compared to usual gradient term. Finally, we multiply some constant so that α_0 will not be too small.

Appendix M. Proof of Theorem 8

In this section, we derive the upper-bound of the bias of skew-SGLD based on [23]. This approach requires us to use the logarithmic Sobolev inequality [19], which is stronger than the Poincaré inequality. First, we present the definition of the logarithmic Sobolev inequality. We say that π on $\mathbb{R}^d$ with $\mathcal{L}$ satisfies the logarithmic Sobolev inequality with constant λ in case for all function f on $\mathbb{R}^d$ with $\int_{\mathbb{R}^d} u^2 d\pi = 1$,

$$\int_{\mathbb{R}^d} f^2 \ln f^2 d\pi \leq \frac{2}{\lambda} \int_{\mathbb{R}^d} -f\mathcal{L}f d\pi. \tag{A149}$$

This logarithmic Sobolev inequality is stronger than the Poincaré inequality and induces the convergence in KL divergence. See [19] for details. It was proved in [2,18] that our dynamics, LD, SLD, PLD, S-PLD, and skew-SGLD satisfy the logarithmic Sobolev inequalities under our assumptions. We express the constant of the logarithmic Sobolev inequality for skew-SGLD as $\lambda(\alpha, N)$. This constant depends on the skew matrices and the Poincaré constant. We estimate this constant in Appendix M.1.

To upper-bound the bias, here we control the KL divergence. We denote the law of skew-SGLD at iteration k with interaction strength α as $\mu_{kh}^{\otimes N}(\alpha)$. We upper-bound the bias by 2-Wasserstein distance

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right|, \leq \frac{L_f}{\sqrt{N}} W_2(\mu_{kh}^{\otimes N}(\alpha), \pi^{\otimes N}). \tag{A150}$$

Then, from the transportation inequality [19],

$$W_2(\mu_k, \pi) \leq \sqrt{\frac{2}{\lambda(\alpha, N)} \mathrm{KL}(\mu_{kh}^{\otimes N}(\alpha)|\pi^{\otimes N})}. \tag{A151}$$

Thus, we will upper bound the KL divergence using the technique in [23]. However, in the original proof, a full gradient ∇U is used so we replace it with the stochastic gradient. Moreover, we introduce the skew interaction term.

First, Lemma 11 in [23] is modified to

$$\mathbb{E}_{\pi^{\otimes N}}\|\nabla U^{\otimes N}\|^2 \leq \frac{dNM}{\beta}. \tag{A152}$$

Then Lemma 12 in [23] is modified to

$$\mathbb{E}_{\mu}\|\nabla U^{\otimes N}\|^2 \leq 4M^2\lambda \mathrm{KL}(\mu|\pi^{\otimes N}) + \frac{2dNM}{\beta}, \tag{A153}$$

for any integrable μ.

Herein after, we drop $\otimes N$ from $X^{\otimes N}$, $\nabla U^{\otimes N}$, and $\nabla \hat{U}^{\otimes N}$ for notational simplicity. We focus on skew-SGLD at iteration k, we consider the following SDE for $t \in (kh, (k+1)h]$

$$dX_t = -(I+\alpha J)\nabla \tilde{U}(X_k)dt + \sqrt{2\beta^{-1}}dw_t, \tag{A154}$$

where $\nabla \tilde{U}(X_k)$ is the stochastic gradient conditioned on X_k. The solution of this SDE is

$$X_{(k+1)} = X_k - (I+\alpha J)\nabla \tilde{U}(X_k)h + \sqrt{2\beta^{-1}}\epsilon. \tag{A155}$$

We would like to derive the continuity equation correspond to Equation (A154). Following [23], we express X_t as x_t and X_k as x_0 for simplicity. Let $\rho_{0t}(x_0, x_t)$ denote the joint distribution of (x_0, x_t). Then, the conditional and marginal relations are written as

$$\rho_{0t}(x_0, x_t) = \rho_0(x_0)\rho_{t|0}(x_t|x_0) = \rho_t(x_t)\rho_{0|t}(x_0|x_t). \tag{A156}$$

The conditional density $\rho_{t|0}(x_t|x_0)$ follows the FP equation

$$\frac{\partial \rho_{t|0}(x_t|x_0)}{\partial t} = \nabla \cdot (\rho_{t|0}(x_t|x_0)(I+\alpha J)\nabla \tilde{U}(x_0)) + \beta^{-1}\Delta \rho_{t|0}(x_t|x_0), \tag{A157}$$

Then following [23], to derive the evolution of ρ_t, we take the expectation over $\rho_0(x_0)$

$$\begin{aligned}\frac{\partial \rho_t(x)}{\partial t} &= \int_{\mathbb{R}^d} \frac{\partial \rho_{t|0}(x_t|x_0)}{\partial t}\rho_0(x_0)dx_0 \\ &= \nabla \cdot (\rho_t(x_t)\mathbb{E}_{\rho_{0|t}}[(I+\alpha J)\nabla \tilde{U}(x_0)|x_t = x]) + \beta^{-1}\Delta \rho_t(x).\end{aligned} \tag{A158}$$

Then, we take the expectation regarding for the stochastic gradient in the above equation and include it into $\mathbb{E}_{\rho_{0|t}}$ for notational simplicity. Then following the discussion of Lemma 3 in [23], we obtain

$$\begin{aligned}\frac{\partial \mathrm{KL}(\mu_t|\pi)}{\partial t} \leq& -\frac{3}{4}I(\mu_t^{\otimes N}|\pi^{\otimes N}) + 2\mathbb{E}_{\rho_{0t}}[\|\nabla U(X_t) - \nabla U(X_0)\|^2] \\ &+ 2(1+\alpha)^2\mathbb{E}_{\rho_{0t}}[\|\nabla U(X_0) - \nabla \tilde{U}(X_0)\|^2] + 2\alpha^2\mathbb{E}_{\rho_{0t}}[\nabla U(x_0)\|^2],\end{aligned} \tag{A159}$$

where $t \in (kh, (k+1)h]$ and

$$X_t = X_k - t(I+\alpha J)\nabla U(X_k) + \sqrt{2t\beta^{-1}}\epsilon. \tag{A160}$$

Then, from [18], we can upper-bound the second term by

$$\mathbb{E}_{\rho_{0t}}[\|\nabla U(X_0) - \nabla \tilde{U}(X_0)\|^2] \leq NC_0'\delta, \tag{A161}$$

$$C_0' := 2\left(M^2\left(\kappa_0 + 2\left(1 \vee \frac{1}{m}\right)\left(b + 2(1+\alpha)^2B^2 + \frac{d}{\beta}\right)\right) + B^2\right) \tag{A162}$$

and the third term is upper-bounded by

$$\mathbb{E}_{\rho_{0t}}[\|\nabla U(X_0)-\nabla\mathbb{E}_{\rho_{0t}}[\nabla U(x_0)\|^2]\leq 2M^2\|x_0\|^2+2NB^2 \\ \leq NC_0', \tag{A163}$$

where we used lemma 2 and 7 in [2]. Finally, from the original proof of [23] we obtain

$$2\mathbb{E}_{\rho_{0t}}[\|\nabla U(X_t)-\nabla U(X_0)\|^2]\leq 8t^2M^4\lambda\mathrm{KL}(\mu_k^{\otimes N}|\pi^{\otimes N})+\frac{4t^2dNM^3}{\beta}+\frac{4tdNM^2}{\beta}. \tag{A164}$$

Then, in conclusion, under $h\in(0,1\wedge\frac{m}{4M^2})$ obeying $kh\geq 1$ and $\beta m\geq 2$, we obtain

$$\frac{d}{dt}\mathrm{KL}(\mu_t^{\otimes N}|\pi^{\otimes N})\leq -\frac{3}{4}I(\mu_t^{\otimes N}|\pi^{\otimes N})+8t^2M^4\lambda(\alpha,N)\mathrm{KL}(\mu_k^{\otimes N}|\pi^{\otimes N}) \\ +\frac{4t^2dNM^3}{\beta}+\frac{4tdNM^2}{\beta}+2NC_0'(\delta(1+\alpha)^2+\alpha^2). \tag{A165}$$

For simplicity, we assume that $h\in(0,\frac{m}{4M^2})$ and $\frac{m}{4M^2}<1$, then we obtain

$$\frac{d}{dt}\mathrm{KL}(\mu_t^{\otimes N}|\pi^{\otimes N})\leq -\frac{3}{4}I(\mu_t^{\otimes N}|\pi^{\otimes N})+8t^2M^4\lambda(\alpha,N)\mathrm{KL}(\mu_k^{\otimes N}|\pi^{\otimes N}) \\ +\frac{t^2dNM}{\beta}(m+4M)+2NC_0'(\delta(1+\alpha)^2+\alpha^2). \tag{A166}$$

Then using $t\in(kh,(k+1)h]$, we obtain

$$\mathrm{KL}(\mu_{k+1}^{\otimes N}|\pi^{\otimes N})\leq e^{-\frac{3}{2}\lambda(\alpha,N)h}\Big(1+16h^3M^4\lambda\Big)\mathrm{KL}(\mu_k^{\otimes N}|\pi^{\otimes N}) \\ +e^{-\frac{3}{2}\lambda(\alpha,N)h}\left(\frac{2hdNM}{\beta}(m+4M)+8hNC_0'(\delta(1+\alpha)^2+\alpha^2)\right). \tag{A167}$$

If $h\in(0,\frac{\lambda(\alpha,N)}{4\sqrt{2}M^2})$, we obtain

$$\mathrm{KL}(\mu_{k+1}^{\otimes N}|\pi^{\otimes N})\leq e^{-\lambda(\alpha,N)h}\mathrm{KL}(\mu_k^{\otimes N}|\pi^{\otimes N})+\frac{2h^2dNM}{\beta}(m+4M)+8hNC_0'(\delta(1+\alpha)^2+\alpha^2). \tag{A168}$$

From this one step inequality, we obtain

$$\begin{aligned}&\mathrm{KL}(\mu_k^{\otimes N}|\pi^{\otimes N})\\ &\leq e^{-\lambda(\alpha,N)kh}\mathrm{KL}(\mu_0^{\otimes N}|\pi^{\otimes N})+\frac{1}{1-e^{-\lambda(\alpha,N)h}}\left(\frac{2h^2dNM}{\beta}(m+4M)+8hNC_0'(\delta(1+\alpha)^2+\alpha^2)\right)\\ &\leq e^{-\lambda(\alpha,N)kh}\mathrm{KL}(\mu_0^{\otimes N}|\pi^{\otimes N})+\frac{2N}{\lambda(\alpha,N)}\left(\frac{hdM}{\beta}(m+4M)+4C_0'(\delta(1+\alpha)^2+\alpha^2)\right).\end{aligned} \tag{A169}$$

Then, finally we obtain

$$\begin{aligned}&\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^N f(X_k^{(n)})-\int_{\mathbb{R}^d}fd\pi\right|\\ &\leq\frac{L_f}{\sqrt{N}}\sqrt{\frac{2}{\lambda(\alpha,N)}\mathrm{KL}(\mu_{kh}^{\otimes N}(\alpha)|\pi^{\otimes N})}\\ &\leq L_f\sqrt{\frac{2}{\lambda(\alpha,N)}}\sqrt{e^{-\lambda(\alpha,N)kh}\mathrm{KL}(\mu_0|\pi)+\frac{2}{\lambda(\alpha,N)}\left(\frac{hdM}{\beta}(m+4M)+4C_0'(\delta(1+\alpha)^2+\alpha^2)\right)}\\ &\leq L_f\sqrt{\frac{2}{\lambda(\alpha,N)}}\sqrt{e^{-\lambda(\alpha,N)kh}\mathrm{KL}(\mu_0|\pi)+\frac{C_3(\alpha)}{\lambda(\alpha,N)}},\end{aligned} \tag{A170}$$

where

$$C_3(\alpha) := 2\frac{hdM}{\beta}(m+4M) + 8C_0'(\delta(1+\alpha)^2+\alpha^2), \tag{A171}$$

$$C_0' := 2\left(M^2\left(\kappa_0 + 2\left(1 \vee \frac{1}{m}\right)\left(b + 2(1+\alpha)^2B^2 + \frac{d}{\beta}\right)\right) + B^2\right). \tag{A172}$$

Moreover, from Appendix M.1, the logarithmic Sobolev constant is

$$\lambda(\alpha, N) := \left(\frac{1}{(1+\beta m(\alpha,N)^{-1}|C(m_0)|)2\pi e^2} + \frac{3}{2m(\alpha,N)}\right), \tag{A173}$$

where

$$-C(m_0) := \mathbb{E}_{\pi^{\otimes N}}[\|\nabla U^{\otimes N}(x)\|]^{1/2} + \sqrt{\frac{8}{m_0}}\mathbb{E}_{\pi^{\otimes N}}[\|\nabla U^{\otimes N}(x)\|^2]^{1/2}. \tag{A174}$$

Appendix M.1. Estimation of the Logarithmic Sobolev Constant

In this section, we estimate the logarithmic Sobolev constants using the technique of restricted logarithmic Sobolev inequality, which was introduced in [49].

The technique of [49] estimates the constant of the logarithmic Sobolev inequality as follows. Assume that π on $\mathbb{R}^d$ with $\mathcal{L}$ satisfies the Poincaré inequality with constant m. Then, for any function u on $\mathbb{R}^d$ that satisfies

$$\int_{\mathbb{R}^d} u d\pi = 0 \quad \text{and} \quad \int_{\mathbb{R}^d} u^2 d\pi = 1, \tag{A175}$$

we find a constant b that satisfies

$$\int_{\mathbb{R}^d} u^2 \ln u^2 d\pi \leq b\int_{\mathbb{R}^d} -u\mathcal{L}u d\pi. \tag{A176}$$

Then the logarithmic constant is larger than $2(b+\frac{3}{m})^{-1}$. Thus, we only need to focus on the restricted function class to estimate a constant b. We slightly change the Lemma 3.2 of [49] that estimate the constant b in Equation (A176) to apply it in our setting. In Lemma 3.2 of [49], it was proved that if u on $\mathbb{R}^d$ satisfies the conditions in Equation (A175), then for any $t \in (0,1)$, we have

$$\int_{\mathbb{R}^d} -u\mathcal{L}u d\pi - t\pi e^2\int_{\mathbb{R}^d} u^2 \ln u^2 d\pi \geq (1-t)m + t\beta\int_{\mathbb{R}^d}(-\frac{1}{2}\mathcal{L}U(x) - \pi e^2 U(x))u^2 d\pi, \tag{A177}$$

where we assume that $\pi \propto e^{-\beta U(x)}$ satisfies the Poincaré inequality with constant m. If there exists a constant C such that

$$-C \geq \beta\int_{\mathbb{R}^d}(-\frac{1}{2}\mathcal{L}U(x) - \pi e^2 U(x))u^2 d\pi > -\infty, \tag{A178}$$

then by setting $t = m/(m+|C|)$, we can show that

$$\int_{\mathbb{R}^d} -u\mathcal{L}u d\pi - m/(m+|C|)\pi e^2\int_{\mathbb{R}^d} u^2 \ln u^2 d\pi > 0. \tag{A179}$$

Thus, the constant b in Equation (A176) is $b = t = m/(m+|C|)$ and the logarithmic constant is $2(m/(m+|C|) + \frac{3}{m})^{-1}$.

Thus, We analyze the constant C. The first term of the integral in Equation (A178) is lower-bounded bounded by

$$-\mathbb{E}_\pi[\mathcal{L}U(x)u^2] \geq -|\mathbb{E}_\pi[U(x)\mathcal{L}U(x)]|^{1/2}|\mathbb{E}_\pi[u^2\mathcal{L}u^2]|^{1/2} \geq -2\mathbb{E}_\pi[\|\nabla U(x)\|^2]^{1/2}, \tag{A180}$$

where we used the property of $\mathcal{L}$, see [19] for details. As for the second term, it is lower-bounded by

$$-|\mathbb{E}_\pi[U(x)u^2] \geq -\sqrt{|\mathbb{E}_\pi[U^2(x)u^2]} \geq -\sqrt{\frac{1}{m}|\mathbb{E}_\pi[(U(x)|u|)\mathcal{L}(U(x)|u|)]|}$$
$$\geq -\sqrt{\frac{8}{m}\mathbb{E}_\pi[\|\nabla U(x)\|^2]^{1/2}}. \tag{A181}$$

Thus, by setting

$$-C := \mathbb{E}_\pi[\|\nabla U(x)\|]^{1/2} + \sqrt{\frac{8}{m_0}\mathbb{E}_\pi[\|\nabla U(x)\|^2]^{1/2}}, \tag{A182}$$

we can estimate the logarithmic constant as $2(m/(m+|C|)+\frac{3}{m})^{-1}$.

In our setting, this is modified to

$$\lambda(\alpha, N) = \left(\frac{1}{(1+\beta m(\alpha,N)^{-1}|C(m_0)|)2\pi e^2} + \frac{3}{2m(\alpha,N)}\right)^{-1}. \tag{A183}$$

where

$$-C(m_0) := \mathbb{E}_{\pi^{\otimes N}}[\|\nabla U^{\otimes N}(x)\|]^{1/2} + \sqrt{\frac{8}{m_0}\mathbb{E}_{\pi^{\otimes N}}[\|\nabla U^{\otimes N}(x)\|^2]^{1/2}}. \tag{A184}$$

Finally, if we increase $m(\alpha, N)$, $\lambda(\alpha, N)$ increases. Thus, since $m(\alpha, N) \geq m(\alpha = 0, N)$, we obtain $\lambda(\alpha, N) \geq \lambda(\alpha = 0, N)$.

Appendix M.2. Computational Complexity

To derive the computational complexity, for simplicity, we assume that $\delta \leq h$ and We also set $\alpha^2 \leq h$ for simplicity. This means that the variance of the stochastic gradient is small enough and we use small α. Then the bias is

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq L_f\sqrt{\frac{2}{\lambda(\alpha,N)}}\sqrt{e^{-\lambda(\alpha,N)kh}\mathrm{KL}(\mu_0|\pi) + \frac{C_3(\alpha)}{\lambda(\alpha,N)}}$$
$$\leq L_f\sqrt{\frac{2}{\lambda(\alpha,N)}}\left(\sqrt{e^{-\lambda(\alpha,N)kh}\mathrm{KL}(\mu_0|\pi)} + \sqrt{\frac{C_3(\alpha)}{\lambda(\alpha,N)}}\right), \tag{A185}$$

where

$$C_3(\alpha) := h\left(2\frac{dM}{\beta}(m+4M) + 8C_0'((1+h^{1/2})^2+1)\right), \tag{A186}$$
$$C_0' := 2\left(M^2\left(\kappa_0 + 2\left(1 \vee \frac{1}{m}\right)\left(b + 2(1+h^{1/2})^2B^2 + \frac{d}{\beta}\right)\right) + B^2\right). \tag{A187}$$

Then we define

$$C_3' := 2\frac{dM}{\beta}(m+4M) + 8C_0'((1+h^{1/2})^2+1), \tag{A188}$$

and use the step size that satisfies $h = \frac{\lambda(\alpha,N)\xi}{2\sqrt{2}C_3'L_f}$. Then when we use

$$k \geq \frac{2}{\lambda(\alpha,N)h}\ln\frac{L_f}{\xi}\sqrt{\frac{\mathrm{KL}(\mu_0|\pi)}{2\lambda(\alpha,N)}}, \tag{A189}$$

we have

$$\left|\mathbb{E}\frac{1}{N}\sum_{n=1}^{N} f(X_k^{(n)}) - \int_{\mathbb{R}^d} f d\pi\right| \leq \frac{\xi}{2} + \frac{\xi}{2} \leq \xi. \tag{A190}$$

References

1. Murphy, K.P. *Machine Learning: A Probabilistic Perspective*; MIT Press: Cambridge, MA, USA, 2012.
2. Raginsky, M.; Rakhlin, A.; Telgarsky, M. Non-convex learning via Stochastic Gradient Langevin Dynamics: A nonasymptotic analysis. In Proceedings of the Conference on Learning Theory, Amsterdam, The Netherlands, 7–10 July 2017; pp. 1674–1703.
3. Welling, M.; Teh, Y.W. Bayesian learning via stochastic gradient Langevin dynamics. In Proceedings of the International Conference on Machine Learning, Washington, DC, USA, 28 June–2 July 2011; pp. 681–688.
4. Livingstone, S.; Girolami, M. Information-Geometric Markov Chain Monte Carlo Methods Using Diffusions. *Entropy* **2014**, *16*, 3074–3102. [CrossRef]
5. Hartmann, C.; Richter, L.; Schütte, C.; Zhang, W. Variational Characterization of Free Energy: Theory and Algorithms. *Entropy* **2017**, *19*, 626. [CrossRef]
6. Neal, R.M. Improving asymptotic variance of MCMC estimators: Non-reversible chains are better. *arXiv* **2004**, arXiv:math/0407281.
7. Neklyudov, K.; Welling, M.; Egorov, E.; Vetrov, D. Involutive mcmc: a unifying framework. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 13–18 July 2020; pp. 7273–7282.
8. Gao, X.; Gurbuzbalaban, M.; Zhu, L. Breaking Reversibility Accelerates Langevin Dynamics for Non-Convex Optimization. In Proceedings of the Advances in Neural Information Processing Systems, Online, 6–12 December 2020; pp. 17850–17862.
9. Eberle, A.; Guillin, A.; Zimmer, R. Couplings and quantitative contraction rates for Langevin dynamics. *Ann. Probab.* **2019**, *47*, 1982–2010. [CrossRef]
10. Gao, X.; Gürbüzbalaban, M.; Zhu, L. Global convergence of stochastic gradient Hamiltonian Monte Carlo for non-convex stochastic optimization: Non-asymptotic performance bounds and momentum-based acceleration. *arXiv* **2018**, arXiv:1809.04618.
11. Cheng, X.; Chatterji, N.S.; Abbasi-Yadkori, Y.; Bartlett, P.L.; Jordan, M.I. Sharp convergence rates for Langevin dynamics in the nonconvex setting. *arXiv* **2018**, arXiv:1805.01648.
12. Chen, T.; Fox, E.; Guestrin, C. Stochastic gradient hamiltonian monte carlo. In Proceedings of the International conference on machine learning, Beijing, China, 21–26 June 2014; pp. 1683–1691.
13. Hwang, C.R.; Hwang-Ma, S.Y.; Sheu, S.J. Accelerating gaussian diffusions. *Ann. Appl. Probab.* **1993**, 3, 897–913. [CrossRef]
14. Hwang, C.R.; Hwang-Ma, S.Y.; Sheu, S.J. Accelerating diffusions. *Ann. Appl. Probab.* **2005**, *15*, 1433–1444. [CrossRef]
15. Hwang, C.R.; Normand, R.; Wu, S.J. Variance reduction for diffusions. *Stoch. Process. Their Appl.* **2015**, *125*, 3522–3540. [CrossRef]
16. Duncan, A.B.; Lelièvre, T.; Pavliotis, G.A. Variance Reduction Using Nonreversible Langevin Samplers. *J. Stat. Phys.* **2016**, *163*, 457–491. [CrossRef] [PubMed]
17. Duncan, A.B.; Nüsken, N.; Pavliotis, G.A. Using Perturbed Underdamped Langevin Dynamics to Efficiently Sample from Probability Distributions. *J. Stat. Phys.* **2017**, *169*, 1098–1131. [CrossRef]
18. Futami, F.; Sato, I.; Sugiyama, M. Accelerating the diffusion-based ensemble sampling by non-reversible dynamics. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 13–18 July 2020; pp. 3337–3347.
19. Bakry, D.; Gentil, I.; Ledoux, M. *Analysis and Geometry of Markov Diffusion Operators*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013; Volume 348.
20. Roussel, J.; Stoltz, G. Spectral methods for Langevin dynamics and associated error estimates. *ESAIM Math. Model. Numer. Anal.* **2018**, *52*, 1051–1083. [CrossRef]
21. Menz, G.; Schlichting, A. Poincaré and logarithmic Sobolev inequalities by decomposition of the energy landscape. *Ann. Probab.* **2014**, *42*, 1809–1884. [CrossRef]
22. Liu, Q.; Lee, J.; Jordan, M. A kernelized Stein discrepancy for goodness-of-fit tests. In Proceedings of the International Conference on Machine Learning, New York , NY, USA, 24–26 June 2016; pp. 276–284.
23. Vempala, S.; Wibisono, A. Rapid convergence of the unadjusted langevin algorithm: Isoperimetry suffices. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 8094–8106.
24. Lelièvre, T.; Nier, F.; Pavliotis, G.A. Optimal non-reversible linear drift for the convergence to equilibrium of a diffusion. *J. Stat. Phys.* **2013**, *152*, 237–274. [CrossRef]
25. Tripuraneni, N.; Rowland, M.; Ghahramani, Z.; Turner, R. Magnetic Hamiltonian Monte Carlo. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 3453–3461.
26. Nusken, N.; Pavliotis, G. Constructing sampling schemes via coupling: Markov semigroups and optimal transport. *SIAM/ASA J. Uncertain. Quantif.* **2019**, *7*, 324–382. [CrossRef]
27. Liu, Q.; Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. In Proceedings of the Advances In Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 2378–2386.
28. Zhang, J.; Zhang, R.; Chen, C. Stochastic particle-optimization sampling and the non-asymptotic convergence theory. *arXiv* **2018**, arXiv:1809.01293.
29. Wang, Y.; Li, W. Information Newton's flow: second-order optimization method in probability space. *arXiv* **2020**, arXiv:2001.04341.

30. Wibisono, A. Sampling as optimization in the space of measures: The Langevin dynamics as a composite optimization problem. In Proceedings of the Conference On Learning Theory, Stockholm, Sweden, 6–9 July 2018; pp. 2093–3027.
31. Gretton, A.; Borgwardt, K.M.; Rasch, M.J.; Schölkopf, B.; Smola, A. A kernel two-sample test. *J. Mach. Learn. Res.* **2012**, *13*, 723–773.
32. Ding, N.; Fang, Y.; Babbush, R.; Chen, C.; Skeel, R.D.; Neven, H. Bayesian sampling using stochastic gradient thermostats. In Proceedings of the Advances in neural information processing systems, Montreal, QC, Canada, 8–11 December 2014; pp. 3203–3211.
33. Patterson, S.; Teh, Y.W. Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–8 December 2013; pp. 3102–3110.
34. Dua, D.; Graff, C. UCI Machine Learning Repository. Available online: http://archive.ics.uci.edu/ml (accessed on 21 July 2021).
35. Villani, C. Optimal transportation, dissipative PDE's and functional inequalities. In *Optimal Transportation and Applications*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 53–89.
36. Bakry, D.; Barthe, F.; Cattiaux, P.; Guillin, A. A simple proof of the Poincaré inequality for a large class of probability measures including the log-concave case. *Electron. Commun. Probab* **2008**, *13*, 21. [CrossRef]
37. Nelson, E. *Dynamical Theories of Brownian Motion*; Princeton University Press: Princeton, NJ, USA, 1967; Volume 3.
38. Pavliotis, G.A. *Stochastic Processes and Applications: Diffusion Processes, the Fokker-Planck and Langevin Equations*; Springer: Berlin/Heidelberg, Germany, 2014; Volume 60.
39. Franke, B.; Hwang, C.R.; Pai, H.M.; Sheu, S.J. The behavior of the spectral gap under growing drift. *Trans. Am. Math. Soc.* **2010**, *362*, 1325–1350. [CrossRef]
40. Landim, C.; Seo, I. Metastability of Nonreversible Random Walks in a Potential Field and the Eyring-Kramers Transition Rate Formula. *Commun. Pure Appl. Math.* **2018**, *71*, 203–266. [CrossRef]
41. Landim, C.; Mariani, M.; Seo, I. Dirichlet's and Thomson's principles for non-selfadjoint elliptic operators with application to non-reversible metastable diffusion processes. *Arch. Ration. Mech. Anal.* **2019**, *231*, 887–938. [CrossRef]
42. Golub, G.H.; Van Loan, C.F. *Matrix Computations*; JHU Press: Baltimore, MD, USA, 2012; Volume 3.
43. Okamoto, M. Distinctness of the Eigenvalues of a Quadratic form in a Multivariate Sample. *Ann. Statist.* **1973**, *1*, 763–765. [CrossRef]
44. Petersen, K.B.; Pedersen, M.S. *The Matrix Cookbook*; Technical University of Denmark: Lynby, Denmark, 2012. Available online: http://www2.compute.dtu.dk/pubdb/pubs/3274-full.html (accessed on 21 July 2021).
45. Van Erven, T.; Harremos, P. Rényi divergence and Kullback-Leibler divergence. *IEEE Trans. Inf. Theory* **2014**, *60*, 3797–3820. [CrossRef]
46. Chewi, S.; Le Gouic, T.; Lu, C.; Maunu, T.; Rigollet, P.; Stromme, A. Exponential ergodicity of mirror-Langevin diffusions. In Proceedings of the Advances in Neural Information Processing Systems, Online, 6–12 December 2020; 2020; pp. 19573–19585.
47. Bolley, F.; Villani, C. Weighted Csiszár-Kullback-Pinsker inequalities and applications to transportation inequalities. In *Annales de la Faculté des Sciences de Toulouse: Mathématiques*; Université Paul Sabatier: Toulouse, France, 2005; Volume 14, pp. 331–352.
48. Donsker, M.D.; Varadhan, S.S. Asymptotic evaluation of certain Markov process expectations for large time. IV. *Commun. Pure Appl. Math.* **1983**, *36*, 183–212. [CrossRef]
49. Carlen, E.; Loss, M. Logarithmic Sobolev inequalities and spectral gaps. *Contemp. Math.* **2004**, *353*, 53–60.

entropy

MDPI

Article

Flexible and Efficient Inference with Particles for the Variational Gaussian Approximation

Théo Galy-Fajou [1,*], Valerio Perrone [2] and Manfred Opper [1,3]

1 Artificial Intelligence Group, Technische Universität Berlin, 10623 Berlin, Germany; manfred.opper@tu-berlin.de
2 Amazon Web Services, 10969 Berlin, Germany; vperrone@amazon.com
3 Centre for Systems Modelling and Quantitative Biomedicine, University of Birmingham, Birmingham B15 2TT, UK
* Correspondence: galy-fajou@tu-berlin.de

Abstract: Variational inference is a powerful framework, used to approximate intractable posteriors through variational distributions. The de facto standard is to rely on Gaussian variational families, which come with numerous advantages: they are easy to sample from, simple to parametrize, and many expectations are known in closed-form or readily computed by quadrature. In this paper, we view the Gaussian variational approximation problem through the lens of gradient flows. We introduce a flexible and efficient algorithm based on a linear flow leading to a particle-based approximation. We prove that, with a sufficient number of particles, our algorithm converges linearly to the exact solution for Gaussian targets, and a low-rank approximation otherwise. In addition to the theoretical analysis, we show, on a set of synthetic and real-world high-dimensional problems, that our algorithm outperforms existing methods with Gaussian targets while performing on a par with non-Gaussian targets.

Keywords: variational inference; Gaussian; particle flow; variable flow

Citation: Galy-Fajou, T.; Perrone, V.; Opper, M. Flexible and Efficient Inference with Particles for the Variational Gaussian Approximation. *Entropy* **2021**, *23*, 990. https://doi.org/10.3390/e23080990

Academic Editor: Pierre Alquier

Received: 22 June 2021
Accepted: 21 July 2021
Published: 30 July 2021

1. Introduction

Representing uncertainty is a ubiquitous problem in machine learning. Reliable uncertainties are key for decision making, especially in contexts where the trade-off between exploitation and exploration plays a central role, such as Bayesian optimization [1], active learning [2], and reinforcement learning [3]. While Bayesian inference is a principled tool to provide uncertainty estimation, computing posterior distributions is intractable for many problems of interest. Most sampling methods struggle to scale up to large datasets [4], while the diagnosis of convergence is not always straightforward [5]. On the other hand, *Variational Inference* **(VI)** methods can rely on well-understood optimization techniques and scale well to large datasets, at the cost of an approximation quality depending heavily on the assumptions made. The Gaussian family is by far the most popular variational approximation used in VI [6,7]. This is for several reasons. First, Gaussian variational families are easy to sample from, reparametrize, and marginalize. Second, they are easily amenable to diagonal covariance approximations, making them scalable to high dimensions. Third, most expectations are either easily computable by quadrature or Monte Carlo integration, or known in closed-form.

A large body of work covers different approaches to optimize the *Variational Gaussian Approximation* **(VGA)**, with the speed of convergence and the scalability in dimensions as the main concerns. From the perspective of convergence speed, the major bottleneck when computing gradients with stochastic estimators is the estimator variance [8]. *Particle-based methods* with deterministic paths do not have this issue, and have been proven to be highly successful in many applications [9–11]. However, can we use a particle-based algorithm to compute a VGA? If so, what are its properties and is it competitive with other VGA methods?

In this paper, we attempt to answer these questions by introducing the *Gaussian Particle Flow* **(GPF)**, a framework to approximate a Gaussian variational distribution with particles. GPF is derived from a continuous-time flow, where the necessary expectations over the evolving densities are approximated by particles. The complexity of the method grows quadratically with the number of particles but linearly with the dimension, remaining compatible with other approximations such as structured mean-field approximations. Using the same dynamics, we also derive a stochastic version of the algorithm, *Gaussian Flow* **(GF)**. To show convergence, we prove the decrease in an empirical version of the free energy that is valid for a finite number of particles. For the special case of D–dimensional Gaussian target densities, we show that $D+1$ particles are enough to obtain convergence to the true distribution. We also find, for this case, that convergence is exponentially fast. Finally, we compare our approach with other VGA algorithms, both in fully controlled synthetic settings and on a set of real-world problems.

2. Related Work

The goal of Bayesian inference is to carry out computations with the posterior distribution of a latent variable $x \in R^D$ given some observations y. By Bayes theorem, the posterior distribution is $p(x|y) = \frac{p(y|x)p(x)}{p(y)}$, where $p(y|x)$ and $p(x)$ are, respectively, the likelihood and the prior distribution. Even if the likelihood and the prior are known analytically, marginalizing out high-dimensional variables in the product $p(y|x)p(x)$ in order to compute quantities such as $p(y)$ is typically intractable. *Variational Inference* **(VI)** aims to simplify this problem by turning it into an optimization one. The intractable posterior is approximated by the closest distribution within a tractable family, with closeness being measured by the *Kullback-Leibler* **(KL)** divergence, defined by

$$\mathrm{KL}\,[q(x)||p(x)] = \mathbb{E}_q[\log q(x) - \log p(x)],$$

where $\mathbb{E}_q[f(x)] = \int f(x)q(x)dx$ denotes the expectation of f over q. Denoting by $\mathcal{Q}$ a family of distributions, we look for

$$\arg\min_{q\in\mathcal{Q}} \mathrm{KL}\,[q(x)||p(x|y)].$$

Since $p(y)$ is not computable in an efficient way, we equivalently minimize the upper bound $\mathcal{F}$:

$$\mathrm{KL}[q(x)||p(x|y)] \leq \mathcal{F}[q] = -\mathbb{E}_q[\log p(y|x)p(x)] - \mathbb{H}_q, \tag{1}$$

where $\mathbb{H}_q$ is the entropy of q ($-\mathbb{E}_q[\log q(x)]$). Here, $\mathcal{F}$ is known as the variational free energy and $-\mathcal{F}$ is known as the Evidence Lower BOund (ELBO). A diverse set of approaches to perform VI with Gaussian families $\mathcal{Q}$ have been developed in the literature, which we review in the following.

2.1. The Variational Gaussian Approximation

The VGA is the restriction of $\mathcal{Q}$ to be the family of multivariate Gaussian distributions $q(x) = \mathcal{N}(m, C)$, where $m \in \mathbb{R}^D$ is the mean and $C \in \{A \in \mathbb{R}^{D\times D} | x^\top Ax \geq 0, \forall x \in \mathbb{R}^D\}$ is the covariance matrix, for which the free energy is found to be

$$\mathcal{F}[q] = -\frac{1}{2}\log|C| + \mathbb{E}_q[\varphi(x)]. \tag{2}$$

where $\varphi(x) = -\log(p(y|x)p(x))$. A standard descent algorithm based on gradients of Equation (2) with respect to variational parameters m, C give rise to some issues. First, naively computing the gradient of the expectation with respect to the covariance matrix

C involves unwanted second derivatives of $\varphi(x)$ [12], which may not be available or may be computationally too expensive in a *black-box* setting. Second, the gradient of the entropy term $\mathbb{H}_q$ entails inverting a non-sparse matrix, which we would like to avoid for higher-dimensional cases. Finally, the positive-definiteness of the covariance matrix leads to non-trivial constraints on parameter updates, which can lead to a slowdown of convergence or, if ignored, to instabilities in the algorithm.

To solve these issues, a variety of approaches have been proposed in the literature. If we focus on factorizable models, we can make a simplification: for problems with likelihoods that can be rewritten as $p(y|x) = \prod_{d=1}^{D} p(y|x_d)$, the number of independent variational parameters is reduced to $2D$ [12,13]. In this special case, the Gaussian expectations in the free energy (2) split into a sum of 1-dimensional integrals, which can be efficiently computed by using numerical quadrature methods. To extend to the general case, gradients of the free energy are estimated by a stochastic sampling approach, which also forms the starting point of our method. This relies on the so-called *reparametrization trick*, where the expectation over the parameter-dependent variational density q_θ is replaced by an expectation over a fixed density q^0 instead. This facilitates the gradient computation because unwanted derivatives of the type $\nabla_\theta q_\theta(x)$ are avoided. For the Gaussian case, the reparametrization trick is a linear transformation of an arbitrary D dimensional Gaussian random variable $x \sim q_\theta(x)$ in terms of a D-dimensional Gaussian random variable $x^0 \sim q^0 = \mathcal{N}(m^0, C^0)$:

$$x = \Gamma(x^0 - m^0) + m, \tag{3}$$

where $\Gamma \in \mathbb{R}^{D\times D}$ and $m \in \mathbb{R}^D$ are the variational parameters. We assume that the covariance C^0 is not degenerate and, for simplicity, we set it as the identity. For instance, the gradient of the expectation given q over a function f given the mean m becomes $\nabla_m \mathbb{E}_q[f(x)] = \mathbb{E}_{q^0}\left[\nabla_m f(\Gamma(x^0 - m^0) + m)\right]$. This can be simply proved by using the reparametrization (3) inside the integral and passing the gradient inside; for more details, see [14].

Given this representation, the free energy is easily obtained as a function of the variational parameters:

$$\mathcal{F}(q) = -\log|\Gamma| + \mathbb{E}_{q^0}\left[\varphi(\Gamma(x^0 - m^0) + m)\right]. \tag{4}$$

Other representations are possible. Challis and Barber [13] and Ong et al. [15] use a different reparametrization with a factorized structure of the covariance $C = \Gamma^\top\Gamma + \operatorname{diag}(d)$, where $\Gamma \in \mathbb{R}^{D\times P}$ and $d \in \mathbb{R}^D$, with $P \leq D$ is the rank of $\Gamma^\top\Gamma$. Other representations assume special structures of the precision matrix $\Lambda = C^{-1}$, which allow you to enforce special properties, such as sparsity in [16,17].

In general, these methods tend to scale poorly with the number of dimensions, as one needs to optimize $D(D+3)/2$ parameters. The (structured) *Mean-Field* **(MF)** [18,19] approach imposes independence between variables in the variational distribution. The number of variational parameters is then $2D$, but covariance information between dimensions is lost.

2.2. Natural Gradients

Besides the issue of expectations, more efficient optimizations directions, beyond ordinary gradient descent, have been considered. These can help to deal with constraints such as those given for the covariance matrix. Natural gradients [20] are a special case of Riemannian gradients and utilize the specific Riemannian manifold structure of variational parameters. They can often deal with constraints of parameters (such as the positive definiteness of the covariance), accelerate inference, and improve the convergence of algorithms. The application of such advanced gradient methods typically requires an estimate of the inverse Fisher information matrix as a preconditioner of ordinary gradients. Khan and Nielsen [21] and Lin et al. [22] propose a solution that requires extra second derivatives of the log–posteriors. Salimbeni et al. [23] developed an automatic process to compute these without the second derivatives but with instability issues. Lin et al. [17] solved these issues by using geodesics on the manifold of parameters, at the price of having to compute inverse matrices as well as Hessians.

2.3. Particle-Based VI

Stochastic gradient descent methods compute expectations (and gradients) at each time step with new independent Monte Carlo samples drawn from the current approximation of the variational density. Particle-based methods for variational inference *draw samples only once* at the beginning of the algorithm instead. They iteratively construct transformations of an initial random variable (having a simple tractable density) where the transformed density leads to the decrease and finally to the minimum of the variational free energy. The iterative approach induces a deterministic temporal flow of random variables which depends on the current density of the variable itself. Using an approximation by the empirical density (which is represented by the positions of a set of 'particles') one obtains a flow of interacting particles which converges asymptotically to an empirical approximation of the desired optimal variational density.

The most popular approach is *Stein Variational Gradient Descent* **(SVGD)** [24], which computes a nonparametric transformation based on the kernelized Stein discrepancy [9]. SVGD has the advantage of not being restricted to a parametric form of the variational distribution. However, using standard distance-based kernels like the squared exponential kernel ($k(x,y) = \exp(-\|x-y\|_2^2/2)$) can lead to underestimated covariances and poor performance in high dimensions [11,25]. Hence, it is interesting to develop particle approaches that approximate the VGA. We provide a more thorough comparison between our method and SVGD in Section 3.6.

2.4. GVA in Bayesian Neural Networks

There has been increased interest in making *Bayesian Neural Networks* **(BNN)** by adding priors to Neural Networks parameters. The true form of the posterior is unknown but VGA has been used due to its ease of use and scalability with the number of dimensions (typically $D \gg 10^5$). Most of the aforementioned methods apply to BNN, but techniques have been specifically tailored with BNN in mind. [26] use the low-rank structure of [13] but exploit the *Local Reparametrization Trick*, where each datapoint y_i gets a different sample from q in order to reduce the stochastic gradient estimator variance. *Stochastic Weight Averaging-Gaussian* **(SWAG)** [27], in which a set of particles obtained via stochastic gradient descent represent a low-rank Gaussian distribution, approximating the true posterior with a prior posterior produced by the network's regularization. While easy to implement, SWAG does not allow you to incorporate an explicit prior, and the resulting distribution does not derive from a principled Bayesian approach.

2.5. Related Approaches

The closest approach to our proposed method is the *Ensemble Kalman Filter* **(EKF)** [28]. It assumes that the posterior is computed in a sequential way, where, at each time step, only single (or smaller batches) of data observations, represented by their likelihoods, become available. An ensemble of particles, representing a Gaussian distribution is iteratively updated with every new batch of observations. EKF allows us to work on high-dimensional problems with a limited amount of particles but is restricted to factorizable likelihoods for which a sequential representation is possible. While EKF maintains a representation of a Gaussian posterior, it is not clear how this relates to the goal of minimizing the free energy or the KL divergence.

3. Gaussian (Particle) Flow

We introduce *Gaussian Particle Flow* **(GPF)** and *Gaussian Flow* **(GF)**, two computationally tractable approaches, to obtain a *Variational Gaussian Approximation* **(VGA)**. In the following, we derive deterministic linear dynamics, which decreases the variational free energy. We additionally give some variants with a *Mean-Field* **(MF)** approach and prove theoretical convergence guarantees.

In the following, $\frac{d(\cdot)}{dt}$ indicates the total derivative given time, $\frac{\partial(\cdot)}{\partial t}$ partial derivatives given time, $\nabla_x(\cdot)$ gradients given a vector x.

3.1. Gaussian Variable Flows

We next discuss an alternative approach to generate the desired transformation of random variables, leading from a simple (prior) Gaussian density to a more complex Gaussian, which minimizes the variational free energy. It is based on the idea of *variable flows*, i.e., recursive deterministic transformations of the random variables defined by a mapping $x^{n+1} = x^n + \epsilon f^n(x^n)$ where $f^n : \mathbb{R}^D \to \mathbb{R}^D$. Well-known examples of flows are *Normalizing Flows* [29], where f^n are bijections, or *Neural ODEs* [30] where $f^n = f$ is defined by a neural network and x^0 is the input. For simplicity, we will consider small changes $\epsilon \to 0$ and work with flows in the continuous-time limit ($t = n\epsilon$), which follow a system of *Ordinary Differential Equation* **(ODE)**. For the Gaussian case, in the spirit of the reparametrization trick (3), we choose a linear corresponding map f and write

$$\frac{dx^t}{dt} = f^t(x^t) = A^t(x^t - m^t) + b^t, \tag{5}$$

where A^t is a matrix and $m^t \doteq \mathbb{E}_{q^t}[x]$ (which is no longer interpreted as an independent variational parameter). When the initial random variable x^0 is Gaussian distributed, the vectors x^t are also Gaussian for any t. To construct a flow that decreases the free energy over time, we can either compute the time derivative of the specific free energy (2) induced by the ODE (5), or simply derive the general result valid for smooth maps f (see, e.g., [24]). To be self contained, we briefly repeat the main steps: We first compute the change of the free energy in terms of the time derivative of q^t:

$$\begin{aligned}
\frac{d\mathcal{F}[q^t]}{dt} =& \frac{d}{dt}\int q^t(x)\big(\log q^t(x) + \varphi(x)\big)dx \\
=& \int \frac{\partial q^t(x)}{\partial t}\big(\log q^t(x) + \varphi(x)\big)dx + \int q^t(x)\left(\frac{\partial q^t(x)}{\partial t}\frac{1}{q^t(x)} + \frac{\partial \varphi(x)}{\partial t}\right)dx \\
=& \int \frac{\partial q^t(x)}{\partial t}\big(\log q^t(x) + \varphi(x)\big)dx
\end{aligned}$$

where we have used the fact that $\int \frac{\partial q^t(x)}{\partial t}dx = \frac{d}{dt}\int q^t(x)dx = 0$ and $\frac{\partial \varphi(x)}{\partial t} = 0$. We next use the *continuity equation* for the density

$$\frac{\partial q^t(x)}{\partial t} = -\nabla_x \cdot \big(q^t(x) f^t(x)\big),$$

related to the deterministic flow to obtain

$$\begin{aligned}
\frac{d\mathcal{F}[q^t]}{dt} &= \int \nabla_x \cdot \big(q^t(x) f^t(x)\big)\big(\log q^t(x) + \varphi(x)\big)dx \\
&= -\int \big(q^t(x) f^t(x)\big) \cdot \nabla_x\big(\log q^t(x) + \varphi(x)\big)dx \\
&= \int \big(\nabla_x \cdot \big(q^t(x) f^t(x)\big) + q^t(x) f^t(x) \cdot \nabla_x \varphi(x)\big)dx \\
&= \int \nabla_x q^t(x) \cdot f^t(x) + q^t(x) f^t(x) \cdot \nabla_x \varphi(x) dx \\
&= -\mathbb{E}_{q^t}\left[\nabla_x \cdot f^t(x) - f^t(x) \cdot \nabla_x \varphi(x)\right]
\end{aligned}$$

where we have applied Green's identity twice and used the fact that $\lim_{x\to\infty} q_t(x) = 0$. Specializing to the linear flow (5), we obtain

$$\frac{d\mathcal{F}[q^t]}{dt} = -\mathrm{tr}[A^t(A^t_\star)^\top] - (b^t)^\top b^t_\star, \tag{6}$$

where

$$\begin{aligned} A^t_\star &\doteq I - \mathbb{E}_{q^t}\left[\nabla_x \varphi(x)(x-m^t)^\top\right] \\ b^t_\star &\doteq -\mathbb{E}_{q^t}[\nabla_x \varphi(x)] \end{aligned} \tag{7}$$

Equation (6) represents the change in the free energy $\mathcal{F}$ for an infinitesimal change in the variables x given by the flow (5). Obviously, the simplest choices

$$A^t \equiv A^t_\star \qquad b^t \equiv b^t_\star \tag{8}$$

lead to a decrease in the free energy $\frac{d\mathcal{F}[q^t]}{dt} \leq 0$. More detailed derivations are given in Appendix A. Additionally, *equality* only happens, when

$$\begin{aligned} I - \mathbb{E}_q\left[\nabla_x \varphi(x)(x-m)^\top\right] &= 0 \\ \mathbb{E}_q[\nabla_x \varphi(x)] &= 0 \end{aligned} \tag{9}$$

Using Stein's lemma [31], we can show that these fixed-point solutions are equal to the conditions for the optimal variational Gaussian distribution solution given in [12]. In Appendix C, we show that our parameter updates can be interpreted as a Riemannian gradient descent method for the free energy (4). This is based on the metric introduced by ([20], Theorem 7.6) as an efficient technique for learning the mixing matrix in models of blind source separation. This gradient should not be confused with the so-called *natural gradient* obtained by pre-multiplying with the inverse Fischer-information matrix.

Of course, there are other choices for A^t and b^t, which lead to a decrease in the free energy and the same fixed-point equations. In Section 3.6, we discuss how SVGD, with a linear kernel, can lead to the same fixed points but with different dynamics.

3.2. From Variable Flows to Parameter Flows

Before we introduce the particle algorithm, we show that the results for the variable flow can also be converted into a temporal change of the parameters Γ^t, m^t, as defined for Equation (3). From this, a corresponding *Gaussian Flow* **(GF)** algorithm can be easily derived. By differentiating the parametrisation $x^t = \Gamma^t(x^0 - m^0) + m^t$ (with m^t now considered as free variational parameter) with respect to time t and using (5), we obtain

$$\frac{dx^t}{dt} = \frac{d\Gamma^t}{dt}(x^0 - m^0) + \frac{dm^t}{dt} = A^t(x^t - m^t) + b^t \tag{10}$$

By inserting $x^t = \Gamma^t(x^0 - m^0) + m^t$ into the right hand side of (10), and using the optimal parameters from (7), we obtain

$$\begin{aligned} \frac{d\Gamma^t}{dt} &= \Gamma^t - \mathbb{E}_{q^0}\left[\nabla_x \varphi(x^t)(x^0 - m^0)^\top\right]\Gamma^t(\Gamma^t)^\top \\ \frac{dm^t}{dt} &= -\mathbb{E}_{q^0}\left[\nabla_x \varphi(x^t)\right] \end{aligned} \tag{11}$$

Note that the expectations are over the probability distribution of the initial random variable x^0. Discretizing Equations (11) in time, and estimating the expectations by drawing independent samples from the fixed Gaussian q^0 at each time step, we obtain our GF algorithm to minimize the variational free energy in the space of Gaussian densities. We summarize the steps of GF in Algorithm 1. Remarkably, this scheme differs from previous VGA algorithms with Riemannian gradients based on the Fisher information metric (see, e.g., [17,32]) because no *matrix inversions* or *second order derivatives* of the function φ are required.

GF also allows for the computation of a low-rank VGA by enforcing $\Gamma \in \mathbb{R}^{D\times K}$ and $x^0 \in \mathbb{R}^K$. This algorithm scales linearly in the number of dimensions and quadratically in the rank K of the covariance.

It is interesting to note that the reverse construction of a variable flow from a parameter flow is, in general, not possible. This would require the ability to eliminate all variational parameters and the initial variables x^0 in the resulting differential equation for x^t, and replace them with functions of x^t alone. For instance, if we eliminate the initial variables x^0 in terms of $(\Gamma^t)^{-1}$ and x^t the algorithm of [14], the resulting expression still depends on Γ^t.

3.3. Particle Dynamics

The main idea of the particle approach is to approximate the Gaussian density q^t in (7) by the empirical distribution

$$\hat{q}^t \doteq \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i^t) \tag{12}$$

computed from N samples x_i^t, $i = 1, \ldots, N$. These are initially sampled from the density q^0 at time $t = 0$ and are then propagated using the discretized dynamics of the ODE (5):

$$\frac{dx_i^t}{dt} = -\eta_1^t \mathbb{E}_{\hat{q}^t}[\nabla_x \varphi(x)] - \eta_2^t \hat{A}^t (x_i^t - \hat{m}^t) \tag{13}$$

where

$$\hat{A}^t = I - \frac{1}{N}\sum_{i=1}^{N}\nabla_x\varphi(x)(x_i^t - \hat{m}^t)^\top$$

$$\hat{b}^t = \frac{1}{N}\sum_{i=1}^{N}\nabla_x\varphi(x_i^t), \qquad \hat{m}^t = \frac{1}{N}\sum_{i=1}^{N}x_i^t$$

where η_1^t and η_2^t are learning rates (We further comment on the use of different optimization schemes in Section 4.4). Note that although $\mathbb{E}_{\hat{q}^t}\left[\nabla_x\varphi(x)(x - \hat{m}^t)^\top\right]$ is a $D \times D$ matrix, changing the matrix multiplication order leads to a computational complexity of $\mathcal{O}(N^2 D)$ with a storage complexity of $\mathcal{O}(N(N + D))$, since neither the empirical covariance matrix or A^t need to be explicitly computed.

Relaxation of Empirical Free Energy and Convergence

We have shown that the continuous-time dynamics (10) of the random variables leads to a decay of the free energy $\mathcal{F}(q^t)$ with time t. Assuming that the free energy is bounded from below, one might conjecture that this property would imply the convergence of the particle algorithm to a fixed point when learning rates are sufficiently small such that the discrete-time dynamics are approximated well by the continuous limit. Unfortunately, the finite number N of particles poses an extra problem. The definition of the free energy $\mathcal{F}(q)$ by the KL–divergence (1) for continuous random variables such as assumes that both $q(\cdot)$ and $p(\cdot|y)$ are densities with respect to the Lebesgue measure. Hence, $\mathcal{F}(\hat{q})$ is not defined if we take $q \equiv \hat{q}$, (12) as the empirical distribution of the finite particle approximation. Nevertheless, we define a finite N *approximation* to the Gaussian free energy, which is also then found to decay under the finite N dynamics. Let us first assume that $N > D$ and define

$$\tilde{\mathcal{F}}(\hat{q}^t) \doteq -\frac{1}{2}\log|\hat{C}^t| + \mathbb{E}_{\hat{q}^t}[\varphi(x)] \tag{14}$$

with the empirical covariance matrix

$$\hat{C}^t = \frac{1}{N}\sum_{i=1}^{N}\left(x_i^t - m^t\right)\left(x_i^t - m^t\right)^\top \tag{15}$$

The definition (14) is chosen in such way that in the large N limit, when the empirical distribution $\hat{q}^t$ converges to a Gaussian distribution q^t, we will also obtain the convergence of the approximation (14) to $\mathcal{F}(q^t)$. It can be shown (see Appendix B) that $\frac{d\tilde{\mathcal{F}}(\hat{q}^t)}{dt} \le 0$, with equality only at the fixed points of the dynamics.

In applications of our particle method to high-dimensional problems, the limitations of computational power may force us to restrict particle numbers to be smaller than the dimensionality D. For $N < D+1$, the empirical covariance C^t will be singular, and typically contain only $N-1$ non-zero eigenvalues, which leads to the $-\log\left|\hat{C}\right| = \infty$ and makes Equation (14) meaningless. We resolve this issue through a regularisation of the log–determinant term in (14), replacing all zero eigenvalues of $\hat{C}$ by the values 1, i.e., $\lambda_i = 0 \to \tilde{\lambda}_i = 1$. We show in Appendix B that the free energy still decays, provided that the dynamics of the particles stay the same. This regularisation step can be formally stated as a replacement of the empirical covariance (15) in (14) by

$$\hat{C}^t \to \hat{C}^t + \sum_{i:\lambda_i^t=0} e_i^t (e_i^t)^\top$$

where $e_i^t = i$th eigenvector of $\hat{C}^t$.

3.4. Algorithm and Properties

The algorithm we propose is to sample N particles $\{x_1^0, \dots, x_N^0\}$ where $x_i^0 \in \mathbb{R}^D$ from q^0 (which can be centered around the MAP for example), and iteratively optimize their positions using Equation (13). Once convergence is reached, i.e., $\frac{d\mathcal{F}}{dt} = 0$, we can easily make predictions using the converged empirical distribution $\hat{q}(x) = \frac{1}{N}\sum_{i=1}^N \delta(x - x_i)$, where δ is the Dirac delta function, or, alternatively, the Gaussian density it represents, i.e., $q(x) = \mathcal{N}(m, C)$, where $m = \frac{1}{N}\sum_{i=1}^N x_i$ and $C = \frac{1}{N}\sum_{i=1}^N (x_i - m)(x_i - m)^\top$. To draw samples from $\hat{q}$, no inversions of the empirical covariance C are needed, as we can obtain new samples by computing:

$$x = \frac{1}{\sqrt{N}}\sum_{i=1}^{N}(x_i - m)\circ \xi_i + m, \tag{16}$$

where ξ_i are i.i.d. normal variables: $\xi_i \sim \mathcal{N}(0, \mathbb{I}_D)$. This can be shown by defining D, the deviation matrix, a matrix which columns equal to $D_i = \frac{x_i - m}{\sqrt{N}}$. We naturally have $DD^\top = C$ which makes D the Cholesky decomposition of C.

All the inference steps are summarized in Algorithm 2 and an illustration in two dimensions is provided in Figure 1.

We summarize the principal points of our approach:

- Gradients of expectations have zero variance, at the cost of a bias decreasing with the number of particles and equal to zero for Gaussian target (see Theorem 1);
- It works with noisy gradients (when using subsampling data, for example);
- The rank of the approximated covariance C is $\min(N-1, D)$. When $N \le D$, the algorithm can be used to obtain a low-rank approximation.
- The complexity of our algorithm is $\mathcal{O}(N^2 D)$ and storing complexity is $\mathcal{O}(N(N+D))$. By adjusting the number of particles used, we can control the performance trade-off;
- GPF (and GF) are also compatible with any kind of structured MF (see Section 3.5);
- Despite working with an empirical distribution ,we can compute a surrogate of the free energy $\mathcal{F}(q)$ to optimize hyper-parameters, compute the lower bound of the log-evidence, or simply monitor convergence.

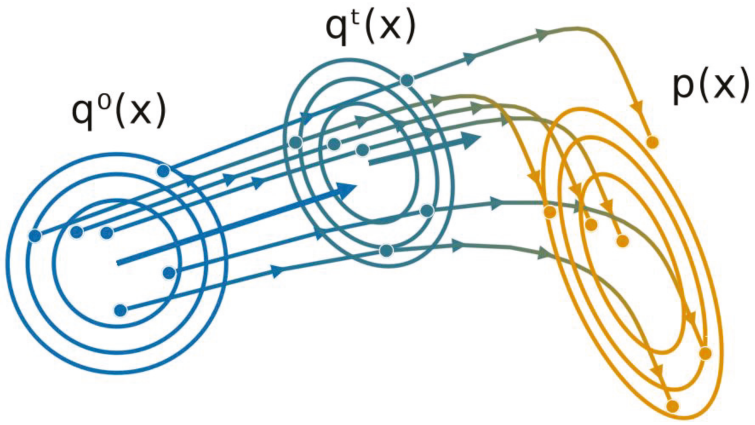


Figure 1. Illustration of the Gaussian Particle Flow algorithm, with $q^0(x)$ and $p(x)$ representing the initial and target distribution respectively. Particles are iteratively moved according to the gradient flow starting from $q^0(x)$, approximating a new Gaussian distribution $q^t(x)$ at each iteration t.

```
Algorithm 1: Gaussian Flow (GF)
Input: Number of samples $N$, initial distribution $q^0 = \mathcal{N}(\mu^0, \Gamma^0(\Gamma^0)^\top)$, target
       $p(x) \propto e^{-\varphi(x)}$, learning rates $\eta_1^t, \eta_2^t$
Output: Variational dist. $q(x) = \mathcal{N}(\mu, \Gamma\Gamma^\top)$
for $t$ in $0:T$ do
    $\{x_i^0\}_{i=1}^N \sim q^0$                                   # Sample $N$ initial particles from $q^0$
    $x_i = \Gamma^t(x_i^0 - \mu^0) + \mu^t,\ \forall i$                         # Reparametrize
    $g_i = \nabla_x \varphi(x_i),\ \forall i$                                 # Compute gradients
    $\mu^{t+1} = \mu^t - \eta_1^t \frac{1}{N}\sum_{i=1}^N \varphi(x_i)$                        # Update $\mu$
    $A = \frac{1}{N}\sum_i g_i(x_i^0 - \mu^0)^\top (\Gamma^t)^\top$                      # Compute matrix
    $\Gamma^{t+1} = \Gamma^t - \eta_2^t A \Gamma^t$                                   # Update $\Gamma$
```

```
Algorithm 2: Gaussian Particle Flow (GPF)
Input: Number of particles $N$, initial distribution $q^0$, target $p(x) \propto e^{-\varphi(x)}$, learning
       rates $\eta_1^t, \eta_2^t$
Output: Empirical dist. $q(x) = \frac{1}{N}\sum_{i=1}^N \delta_{x,x_i}$
Init: Sample N particles from $q^0$ : $\{x_i^0\}_{i=1}^N$
for $t$ in $0:T$ do
    $g_i = \nabla_x \varphi(x_i^t),\ \forall i$                              # Compute gradients
    $m = \frac{1}{N}\sum_i x_i, \quad \overline{g} = \frac{1}{N}\sum_i g_i$                      # Compute means
    $A = \frac{1}{N}\sum_i g_i(x_i^t - m)^\top - I$                          # Compute matrix
    $x_i^{t+1} = x_i^t - \eta_1^t \overline{g} - \eta_2^t A(x_i^t - m),\ \forall i$            # Update particles
```

3.4.1. Relaxation of Empirical Free Energy

The definition of the free energy $\mathcal{F}(q)$ from the KL–divergence (1) for a continuous random variables assumes that both $q(\cdot)$ and $p(\cdot|y)$ are densities with respect to the Lebesgue measure. Hence, it is not *a priori* clear that a specific *approximation* $\mathcal{F}(\hat{q}^t)$, based on an empirical distribution $\hat{q}^t(x) \doteq \frac{1}{N}\sum_{i=1}^N \delta(x - x_i^t)$ with a *finite number of particles N*, will decrease under the particle flow. Thus we may not be able to guarantee convergence to a fixed point for finite N. Luckily, as we show in Appendix D, we find that:

$$\frac{d\mathcal{F}(\hat{q}_t)}{dt} = \frac{d(\mathbb{E}_{\hat{q}_t}[\varphi(x)] - \frac{1}{2}\log|C^t|)}{dt} \leq 0. \tag{17}$$

For $N < D+1$, the empirical covariance C^t will typically contain $N-1$ non-zero eigenvalues and lead to $-\log|C| = \infty$, making Equation (17) meaningless. We resolve this issue by introducing a *regularized free energy* $\widetilde{\mathcal{F}}$ where $\log|C^t|$ is replaced by $\sum_{i:\lambda_i>0}\log\lambda_i$ where $\{\lambda_i\}_{i=1}^D$ are the eigenvalues of C^t. We show in Appendix D that, given the dynamics from Equation (5), $\widetilde{\mathcal{F}}$ is also guaranteed to not increase over time. It can, therefore, be used as a regularized proxy for the true $\mathcal{F}$ and used to optimize over hyper-parameters or to monitor convergence. Note that similar proofs exist for SVGD [33] and were proven to be highly non-trivial.

3.4.2. Dynamics and Fixed Points for Gaussian Targets

We illustrate our method by some exact theoretical results for the dynamics and the fixed points of our algorithm when *the target is a multivariate Gaussian density*. While such targets may seem like a trivial application, our analysis could still provide some insight into the performance for more complicated densities.

Theorem 1. *If the target density $p(x)$ is a D-dimensional multivariate Gaussian, only $D+1$ particles are needed for Algorithm 2 to converge to the exact target parameters.*

Proof. The proof is given in Appendix E. □

Theorem 2. *For a target $p(x) = \mathcal{N}(x \mid \mu, \Lambda^{-1})$, i.e., with precision matrixΛ, where $x \in \mathbb{R}^D$, and $N \geq D+1$ particles, the continuous time limit of Algorithm 2 will converge exponentially fast for both the mean and the trace of the precision matrix:*

$$\begin{aligned} m^t - \mu =& e^{-\Lambda t}(m^0 - \mu), \\ \mathrm{tr}((C^t)^{-1} - \Lambda) =& e^{-2t}\mathrm{tr}\left(\left(C^0\right)^{-1} - \Lambda\right), \end{aligned}$$

where m^t and C^t are the empirical mean and covariance matrix at time t and $\exp(-\Lambda t)$ is the matrix exponential.

Proof. The proof is given in Appendix F. □

Our result shows that convergence of the mean m^t directly depends on Λ. However, we can also precondition the gradient on m by C^t, i.e., using the natural gradient approximation in the Fisher sense, and eventually get rid of the dependency on Λ when $(C^t)^{-1} \approx \Lambda$.

The exponential relaxation of fluctuations also manifests itself in the decay of the free energy towards its minimum. For the Gaussian target, the free energy exactly separates into two terms corresponding to the mean and fluctuations. We can write $\mathcal{F}(m^t, C^t) = \frac{1}{2}(m^t - \mu)^\top\Lambda(m^t - \mu) + \frac{D}{2} + \mathcal{F}_{fl}(C^t)$, where the nontrivial fluctuation part (subtracted by its minimum) is given by

$$\mathcal{F}_{fl}(C^t) = -\frac{1}{2}\log|C^t| + \frac{1}{2}\mathrm{tr}(\Lambda C^t - I).$$

We can show that

$$-\lim_{t\to\infty}\frac{d\ln\mathcal{F}_{fl}(C^t)}{dt} \geq 4,$$

indicating an asymptotic decrease in $\mathcal{F}_{fl}(C^t)$ faster than e^{-4t}, independent of the target. We can also prove the finite time bound

$$\mathcal{F}_{fl}(C^t) \leq \mathcal{F}_{fl}(C^0) e^{-\left[\frac{2t}{\mathrm{tr}(\Lambda^{-1})(\mathrm{tr}(\Lambda)+|\mathrm{tr}((C^0)^{-1}-\Lambda)|)}\right]}.$$

The degenerate case $\mathbf{N} < \mathbf{D} + \mathbf{1}$

Additionally, we can show the following result for the fixed points:

Theorem 3. *Given a D-dimensional multivariate Gaussian target density $p(x) = \mathcal{N}(x|\mu, \Sigma)$, using Algorithm 2 with $N < D + 1$ particles, the empirical mean converges to the exact mean μ. The $N - 1$ non-zero eigenvalues of C^t converge to a subset of the target covariance Σ spectrum. Furthermore, the* ***global minimum*** *of the regularised version $\widetilde{\mathcal{F}}$ of the free energy (17) corresponds to the* ***largest*** *eigenvalues of Σ.*

Proof. The proof is given in Appendix G. □

This result suggests that C^t might typically converge to an optimal low-rank approximation of Σ. We show an empirical confirmation in Section 4.2 for this conjecture. This suggests that it makes sense to apply our algorithm to high-dimensional problems even when the number of particles is not large. If the target density has significant support close to a low-dimensional submanifold, we might still obtain a reasonable approximation.

3.5. Structured Mean-Field

For high-dimensional problems, it may be useful to restrict the variational Gaussian approximation to the posterior to a specific structure via a structured mean-field approximation. In this way, spurious dependencies between variables that are caused by finite-sample effects could be explicitly removed from the algorithms. This is most easily incorporated in our approach by splitting a given collection of latent variables x into M disjoint subsets $x^{(i)}$. We reorder the vector indices in such a way that the first components correspond to $x^{(1)}, x^{(2)}$, and so on. Hence, we obtain $x = \{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$. A structured mean-field approach is enforced by imposing a block matrix structure for the update matrix $A_{MF} = A_{(1)} \oplus \cdots \oplus A_{(M)}$, where $\oplus$ is the direct sum operator. It is easy to see that this construction corresponds to a related block structure of the Γ matrix in Equation (3). This means that the subsets of the random vectors are modeled as independent. Hence, when the number of particles grows to infinity, one recovers the fixed-point equations for the optimal MF structured Gaussian variational approximation from our approach. As previously, as the number of particles grows to infinity, we recover the optimal MF Gaussian variational approximation. Note that using a structured MF does not change the complexity of the algorithm but requires fewer particles to obtain a full-rank solution.

3.6. Comparison with SVGD

Given the similarities with the SVGD methods [24],one could question the differences of our approach. The model proposed by [10] using a *linear kernel* $k(x, x') = x^\top x' + 1$ has similar properties to our approach. The variable update becomes:

$$\begin{aligned}\frac{dx}{dt} &= \frac{1}{N}\sum_{i=1}^{N}(-k(x_i, x)\nabla\varphi(x_i) + \nabla_{x_i}K(x_l, x_i)) \\ &= \mathbb{E}_{\hat{q}}\left[I - \nabla\varphi(x)x^\top\right]x - \mathbb{E}_{\hat{q}}[\nabla\varphi(x)]\end{aligned}$$

The fixed points are

$$\begin{aligned}0 =& \mathbb{E}_{\hat{q}}[\nabla\varphi(x)] \\ I =& \mathbb{E}_{\hat{q}}\left[\nabla\varphi(x)x^\top\right] = \mathbb{E}_{\hat{q}}\left[\nabla\varphi(x)(x - m)^\top\right]\end{aligned}$$

where the last equality holds since $\mathbb{E}_{\hat{q}}[\nabla\varphi(x)] = 0$. This is the same as our algorithm fixed points (9). Similarly to Theorem 1, $D+1$ particles will converge to the exact D-dimensional multivariate Gaussian target. However, the generated flows are different. The main difference is that *we normalize our flow via the L_2 norm*, whereas [10] rely on the *reproducing kernel Hilbert space (RKHS) norm*, i.e., $\|\varphi\|_k^2 = \varphi^\top K^{-1}\varphi$ where $\varphi_i = \varphi(x_i)$ and $K_{ij} = k(x_i, x_j)$. For a full introduction on RKHS, we recommend [34]. Remarkably, centering the particles on the mean, namely, using the modified linear kernel $k(x,x') = (x-m)^\top(x'-m)+1$, leads to the same dynamics. Additionally, when using SVGD, there is no direct possibility of computing the current KL divergence between the variational distribution and the target, unless some values are accumulated [35]. There is also no clear theory explaining what happens when the number of particles is smaller than the number of dimensions, for both distance-based kernels and the linear kernel.

4. Experiments

We now evaluate the efficiency of GPF and GF. First, given a Gaussian target, we compare the convergence of our approach with popular VGA methods, which are all described in Section 2. Second, we evaluate the effect of varying the number of particles for both Gaussian targets and non-Gaussian targets, especially with a low-rank covariance. Then, we evaluate the efficiency of our algorithm on a range of real-world binary classification problems through a Bayesian logistic regression model and a series of BNN on the MNIST dataset.

All the Julia [36] code and data used to reproduce the experiments are available at the Github repository: https://github.com/theogf/ParticleFlow_Exp (accessed on 27 July 2021).

4.1. Multivariate Gaussian Targets

We consider a 20-dimensional multivariate Gaussian target distribution. The mean is sampled from a normal Gaussian $\mu \sim \mathcal{N}(0, I_D)$ and the covariance is a dense matrix defined as $\Sigma = U\Lambda U^\top$, where U is a unitary matrix and Λ is a diagonal matrix. Λ is constructed as $\log_{10}(\Lambda_{ii}) = \frac{\log_{10}(\kappa)(i-1)}{D-1} - 1$ where κ is the condition number, i.e., $\kappa = \Lambda_{\max}/\Lambda_{\min}$. This means that, for $\kappa = 1$, we obtain a $\Sigma = 0.1\mathbb{I}$, and for $\kappa = 100$, we obtain eigenvalues ranging uniformly from 0.1 to 10 in log-space.

We compare GPF and GF to the state-of-the art methods for VGA described in Section 2, namely *Doubly Stochastic VI* **(DSVI)** [14], *Factor Covariance Structure* **(FCS)** [15] with rank $p = D$, *iBayes Learning Rule* **(IBLR)** [17] with a full-rank covariance and their Hessian approach, and Stein Variational Gradient Descent with both a linear kernel (**Linear SVGD**) [10] and a squared-exponential kernel (**Sq. Exp. SVGD**) [24]. For all methods, we set the number of particles or, alternatively, the number of samples used by the estimator, as $D+1$, and use standard gradient descent ($x^{t+1} = x^t + \eta\varphi^t(x^t)$) with a learning rate of $\eta = 0.01$ for all particle methods. We use RMSProp [37] with a learning rate of 0.01 for all stochastic methods. We run each experiment 10 times with 30,000 iterations, and plot the average error on the mean and the covariance with one standard deviation. For GPF, we additionally evaluate the method with and without using natural gradients for the mean (i.e., pre-multiplying the averaged gradient with C^t), indicated, respectively, with a dashed and solid line. Figure 2 reports the L_2 norm of the difference between the mean and covariance with the true posterior over time for the target condition number $\kappa \in \{1, 10, 100\}$.

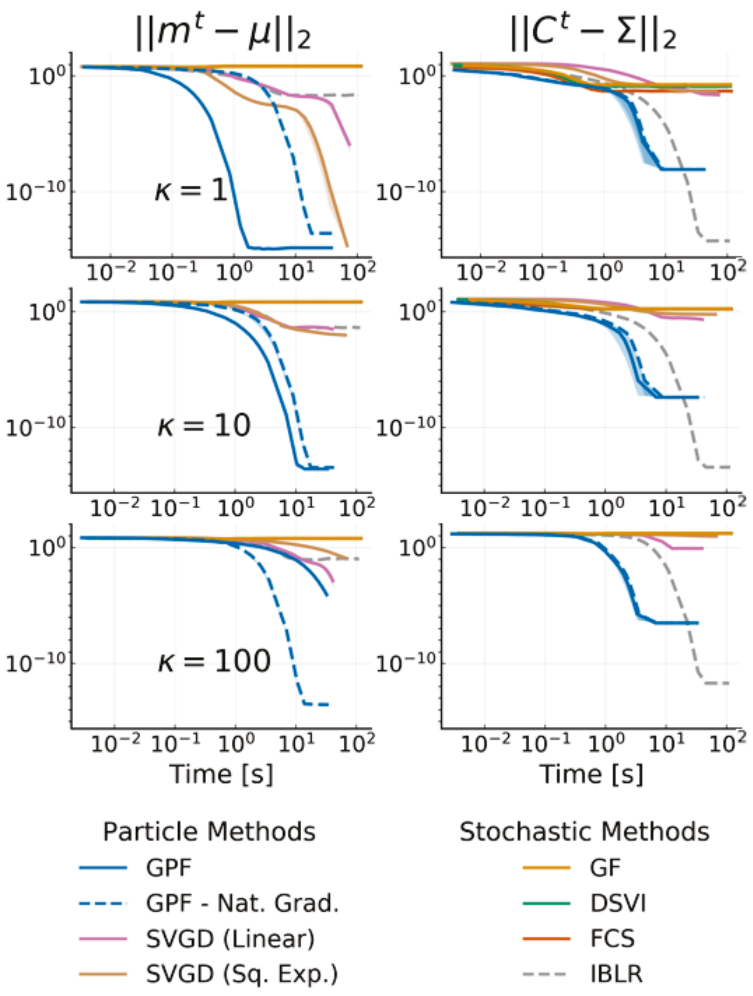


Figure 2. L^2 norm of the difference between the target mean μ (left side) and target covariance Σ (right side) with the inferred variational parameters m^t and C^t against time for 20-dimensional Gaussian targets with condition number κ. We use $D + 1$ particles/samples and show the mean over 10 runs as well as the 68% credible interval. Methods with dashed curves use natural gradients on the mean. Note that DSVI, GF and FCS are overlapping and are, at this scale, indistinguishable from one another.

As Theorem 1 predicts, GPF converges exactly to the true distribution, regardless of the target. GF and other methods based on stochastic estimators cannot obtain the same precision as their accuracy is penalized by the gradient noise. IBLR approximate the covariance perfectly, despite the stochasticity of its estimator; however IBLR needs to compute the true Hessian at each step. When using a Hessian approximation instead, IBLR performed just like DSVI; the true benefit of IBLR appears when second-order functions are computed, which is naturally intractable in high-dimensions. SVGD with a linear kernel, achieves a good performance but is highly unstable: most of the runs (ignored here) diverge. This is due to the dot computation $x^\top x$ which can become extremely high, especially for non-centered data. For this reason, we do not consider this method for the later experiments. SVGD with a sq. exp. kernel obtains a good estimate for the mean but fails to approximate the covariance.

Perhaps surprisingly, GF does not perform much better than DSVI or FCS. This is potentially due to *the benefit of Riemannian gradients being canceled by the gradient noise* [38] providing a strong argument for particle-based methods over stochastic estimators.

Remarkably, we also confirm Theorem 2, that the convergence speed of C^t is independent of the target Σ, while the convergence speed of m^t has this dependency unless the natural gradient is used (see the dashed curves). The case $\kappa = 1$ highlights that *natural gradient do not necessarily improve convergence speed*.

4.2. Low-Rank Approximation for Full Gaussian Targets

We explore the effect of the number of particles for both Gaussian and non-Gaussian targets. We use the same Gaussian target from the previous experiment in 50 dimensions with a full-rank covariance determined by their condition number $\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$. The covariance eigenvalues λ_i in log-space range uniformly from 0.1 to 0.1κ. For a given target multivariate Gaussian, we vary the number of particles from 2 to $D+1$ and look at the absolute difference of $|\text{tr}(C - \Sigma)|$. The results in $D = 50$, as well as the corresponding predictions (in dashed-black), from Theorem 3, are shown on Figure 3.

The empirical results perfectly match the theoretical predictions, confirming that, for Gaussian targets, the particles determine a low-rank approximation whose spectrum is equal to the largest eigenvalues from the target.

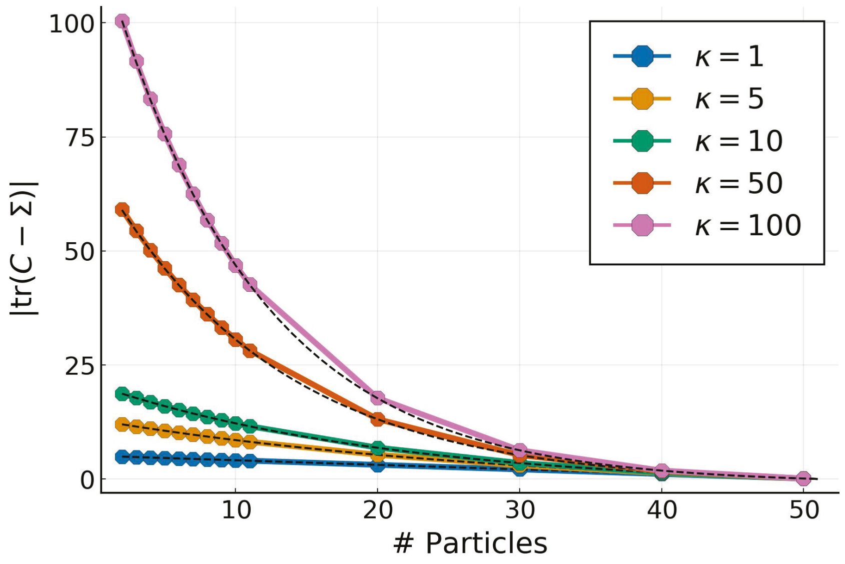


Figure 3. Trace error for a Gaussian target with $D = 50$ and condition numbers κ for a varying number of particles with GPF. Predictions from Theorem 3 are shown in dashed-black.

4.3. High-Dimensional Low-Rank Gaussian Targets

We consider a typical low-rank target case where the dimensionality is high but the effective rank of the covariance is unknown. The target is given by $p(x) = \mathcal{N}(\mu, \Sigma)$ where $\mu \sim \mathcal{N}(0, \mathbb{I}_D)$, the covariance is defined by $\Sigma = U\Lambda U^\top$, where U is a $D \times D$ unitary matrix and Λ is a diagonal matrix defined by

$$\Lambda_{ii} = \begin{cases} \mathcal{N}(2,1), & \text{if } i \leq K \\ 10^{-8}, & \text{otherwise} \end{cases}$$

where K is the effective rank of the target. We pick $D = 500$ and vary $K \in \{10, 20, 30\}$ to simulate a true problem where the correct K is not known. We test all methods allowing for low-rank structure, namely, GPF, GF, FCS and SVGD (Linear and Sq. Exp.). We fix the rank (or the number of particles) to be 20; therefore, we obtain three cases where the rank is exact, under-estimated, and over-estimated. For all methods, we use RMSProp [37] for the stochastic methods, or a diagonal version of it (see Section 4.4) for the particle ones. The error of the mean and the covariance is shown in Figure 4. Note that the difference in the initial error on the covariance is due to the difficulty of starting with the same covariance between particle and stochastic methods.

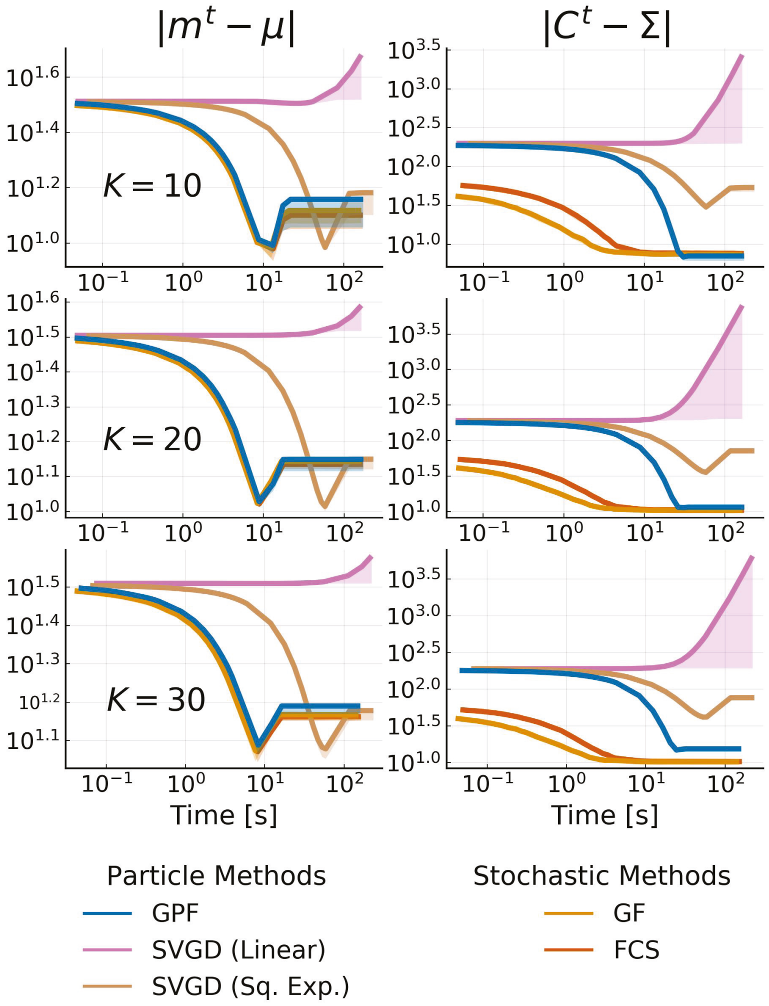


Figure 4. Convergence plot of low-rank methods for a 500-dimensional multivariate Gaussian target with effective rank $K \in \{10, 20, 30\}$. The rank of each method is fixed as 20. The difference in the starting point for the covariance is due to the initialization difference between each method. We show the mean over 10 runs for each method with shadowed areas representing the 68% credible interval.

We observe once again that the SVGD with a linear kernel fails to converge due to the large gradients. All methods perform equally in the estimation of the mean while being non-influenced by the rank of the target. As expected, the approximation quality for the covariance degrades when the rank gets bigger, but all algorithms still converge to good approximations. SVGD with a sq. exp. kernel performs much worse than the rest of the methods. This is a known phenomenon where, for high dimensions, the covariance SVGD is either over- or underestimated.

4.4. Non-Gaussian Target

We now investigate the behavior of our algorithm with non-Gaussian target distributions. We built a two-dimensional banana distribution: $p(x) \propto \exp(-0.5(0.01x_1^2 + 0.1(x_2 + 0.1x_1^2 - 10)^2))$, varied the number of particles used for GPF in $\{3, 5, 10, 20, 50\}$ and compared it with a standard full-rank VGA approach. We also showed the impact of replacing a fixed η with the Adam [39] optimizer for 50 particles. The results are shown in Figure 5. As expected, increasing the number of particles madesthe distribution obtained via GPF increasingly closer to the optimal standard VGA, even in a non-Gaussian setting. However, using a momentum-based optimizer such as Adam breaks the linearity assumption of the original flow (5) and leads to a twisted representation of the particles. (We observed the same behavior with other momentum-based optimizers). A simple modification of the most known optimizers allows the linearity to be maintained while correctly adapting the learning rate to the shape of the problem. Most optimisers accumulate momentum or gradients element-wise, and end up modifying the updates as $x^{t+1} = x^t + P^t \odot \varphi^t(x^t)$, where $P^t \in \mathbb{R}^{D \times D}$ is the preconditioner obtained via the optimiser and $\odot$ is the Hadamard product. By instead taking the average over each dimensions, we obtained the updates $x^{t+1} = x^t + P^t \varphi^t(x^t)$, where P^t is a $D \times D$ diagonal matrix. The details of the dimension-wise conditioners for ADAM, AdaGrad and AdaDelta are given in Appendix H.

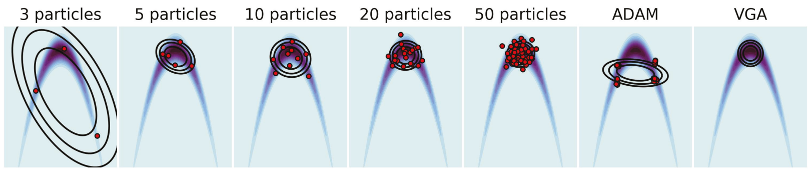
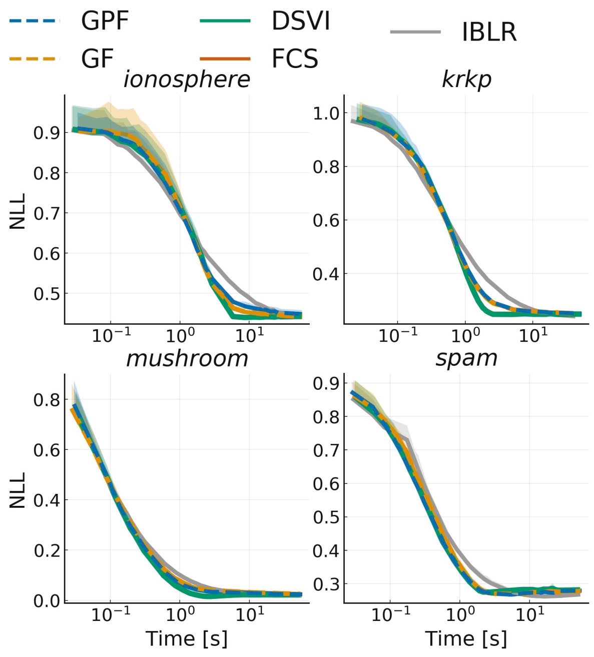


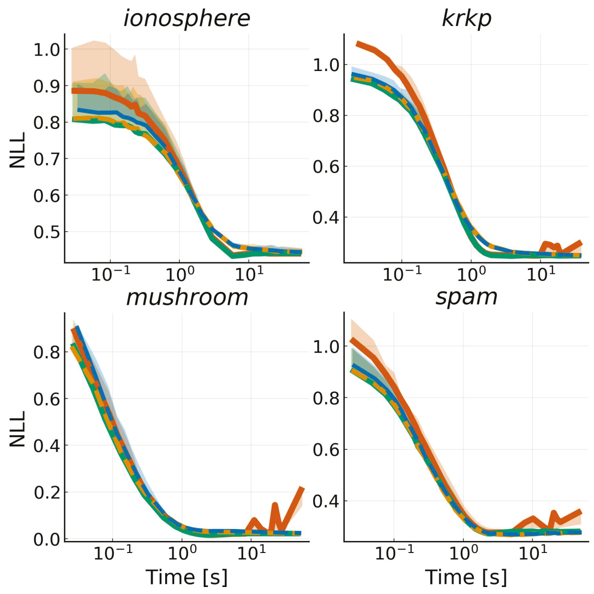
Figure 5. Two-dimensional Banana distribution. Comparison of GPF using an increasing number of particles and a different optimizer (ADAM) with the standard VGA (rightmost plot).

4.5. Bayesian Logistic Regression

Finally, we considered a range of real-world binary classification problems modeled with a Bayesian logistic regression. Given some data $\{(x_i, y_i)\}_{i=1}^N$ where $x_i \in \mathbb{R}^D$ and $y \in \{-1, 1\}$, we defined the model $y_i \sim \text{Bernoulli}(\sigma(w^\top x_i))$ with weight $w \in \mathbb{R}^D$, and with σ being the logistic function. We set a prior on w: $w\ \mathcal{N}(0, 10\mathbb{I}_D)$. We benchmarked the competing approaches over four datasets from the UCI repository [40]: `spam` ($N = 4601, D = 104$), `krkp` ($N = 351, D = 111$), `ionosphere` ($N = 3196, D = 37$) and `mushroom` ($N = 8124, D = 95$). We ran all algorithms discussed in Section 4.1, both with and without a mean-field approximation; SVGD was omitted since it is too unstable. All algorithms were run with a fixed learning rate $\eta = 10^{-4}$, and we used mini-batches of size 100. We show alternative training settings in Appendix I. Note that FCS, for mean-field, simplifies to DSVI Additionally, we did not consider full-rank IBLR, as it is too expensive, and we used their reparametrized gradient version for the Hessian. Figure 6 shows the average negative log-likelihood on 10-fold cross-validation with one standard deviation for each dataset. While, as expected, the advantages shown for Gaussian targets do not transfer to non-Gaussian targets, GPF and GF are consistently on par with competitors. On the other hand, IBLR tends to be outperformed. It is also interesting to note that mean-field does not seem to have a negative impact on these problems, and performance remains the same even with a full-rank matrix.



(**a**) Mean-field approximation



(**b**) No mean-field approximation

Figure 6. Average negative log-likelihood vs. time on a test-set over 10 runs against training time for a Bayesian logistic regression model applied to different datasets. Top plots use a mean-field approximation, while bottom plots use a low-rank structure for the covariance with rank $L = 100$.

4.6. Bayesian Neural Network

We ran our algorithm on a standard network with two hidden layers each, with $L = 200$ neurons and tanh activation functions (we additionally tried ReLU [41], but some baselines failed to converge). We trained on the MNIST dataset [42] ($N = 60{,}000$, $D = 784$) and used an isotropic prior on the weights $p(w) = \mathcal{N}(0, \alpha I_D)$ with $\alpha = 1.0$. We additionally compared these with *Stochastic Weight Averaging-Gaussian* **(SWAG)** [27] with an SGD learning rate of 10^{-6} (selected empirically) and *Efficient Low-Rank Gaussian Variational Inference* **(ELRGVI)** [26]. We varied the assumptions on the covariance matrix to be diagonal (**Mean-Field**), or to have rank $L \in \{5, 10\}$. Additionally, we showed, for GPF, the effect of using a structured mean-field assumption by imposing the independence of the weights between each layer (**GPF (Layers)**).

We trained each algorithm for 5000 iterations with a batchsize of 128($\sim$10 epochs) and reported the final average negative log-likelihood, accuracy and expected calibration error [43] on the test set ($N = 10{,}000$) on Table 1. The predictive distribution is given by

$$p(y = k|x^*, \mathcal{D}) = \int p(y = k|x^*, w)p(w|\mathcal{D})dw \approx \int p(y = k|x^*, w)q(w)dw,$$

where $\mathcal{D}$ is the training data, and x^* is a test sample. We computed the accuracy and the average negative test log-likelihood as:

$$\text{Acc} = \frac{1}{N}\sum_{i=1}^{N} 1_{y_i}(\arg_k \max p(y = k|x_i^*, \mathcal{D}))$$
$$\text{NLL} = -\frac{1}{N}\sum_{i=1}^{N} \log p(y = y_i|x_i^*, \mathcal{D})$$

where $1_y(x)$ is the indicator function (equal to 1 for $y = x$, 0 otherwise). For the definition of expected calibrated error, we refer the reader to [43]. Additional convergence and uncertainty calibration plots can be found in Appendix I.

Table 1. Negative Log-Likelihood (NLL), Accuracy (Acc), and Expected Calibration Error (ECE) for a *Bayesian Neural Networks* **(BNN)** on the MNIST dataset. We varied the rank of the variational covariance from mean-field (all variables are independent) to a low-rank structure with $L \in \{5, 10\}$. Bold numbers indicated the best performance, and italic bold numbers indicate the best performance when restricted to VGA methods. Convergence and calibration plots can be found in Appendix I.

Alg.	Mean-Field			$L = 5$			$L = 10$		
	NLL	**Acc**	**ECE**	**NLL**	**Acc**	**ECE**	**NLL**	**Acc**	**ECE**
GPF	0.183	0.95	0.0384	0.166	***0.96***	0.0918	0.172	0.955	0.0869
GPF (Layers)	-	-	-	***0.147***	0.958	**0.0181**	0.178	0.952	0.0395
GF	0.178	0.953	0.0706	0.185	0.956	0.136	0.171	0.952	0.0455
DSVI	0.204	0.945	0.11	-	-	-	-	-	-
SVGD (Sq. Exp)	-	-	-	0.139	0.965	0.0732	**0.133**	**0.967**	0.0879
SWAG	-	-	-	0.257	0.957	0.0662	0.287	0.956	0.0878
ELRGVI	-	-	-	0.453	0.901	0.53	0.537	0.882	0.777

Overall, the SVGD method performed best in terms of both accuracy and negative log-likelihood. However, SVGD is not in the same category as others, since it is not a VGA. For VGAs, we observed that a low-rank approximation improves upon mean-field methods. In particular, assuming independence between layers provides a large advantage to GPF. GPF and GF generally perform equally or better than all the other VGA methods. Note that, although not reported here, all methods needed approximately the same time for the 5000 iterations, except for SWAG, which only needed the MAP and a few thousand iterations of SGD afterward, making it generally faster but also less controlled (a grid search was needed to find the appropriate learning for SGD).

5. Discussion

We introduced GPF, a general-purpose and theoretically grounded, particle-based approach, to perform inference with variational Gaussians as well as GF its parameter version. We were able to show the convergence of the particle algorithm based on an empirical approximation of the free energy. We also showed that we can approximate high-dimensional targets by allowing for low-rank approximations with a small number of particles. The results for Gaussian targets suggest that the convergence of posterior covariance approximation may relax asymptotically fast, with small dependence on the target. This work is the first step in analyzing convergence speed and guarantees in inference with variational Gaussians, and future work could extend guarantees to non-Gaussian problems. One could also take advantage of existing particle-based VI methods to accelerate inference further or reach a better optima [44,45].

Author Contributions: Conceptualization, T.G.-F. and M.O.; methodology, T.G.-F., V.P. and M.O.; software, T.G.-F.; validation, T.G.-F.; formal analysis, T.G.-F.; investigation, T.G.-F.; resources, T.G.-F. and V.P.; data curation, T.G.-F.; writing—original draft preparation, T.G.-F., V.P. and M.O.; writing—review and editing, T.G.-F., V.P. and M.O.; visualization, T.G.-F.; supervision, M.O.; project administration, T.G.-F.; funding acquisition, M.O. All authors have read and agreed to the published version of the manuscript.

Funding: We acknowledge the support of the German Research Foundation and the Open Access Publication Fund of TU Berlin.

Data Availability Statement: Datasets can be found on the UCI dataset website [40] and the MNIST dataset can be found on Yann Lecun website [42].

Acknowledgments: We thank Fela Winkelmolen for his initial help on computations, Jannik Thümmel for his work on the linear SVGD and the reviewers for their insightful comments.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Derivation of the Optimal Parameters

In Section 3, we considered the optimization problem:

$$\min_{A^t, b^t \in \mathcal{B}} \frac{d\mathcal{F}[q^t]}{dt} \text{ where } \mathcal{B} = \{A^t, b^t : \|A^t\|_F^2 = 1,\ \|b^t\|^2 = 1\},$$

where we have introduced $\|A^2\|_F^2 = \text{tr}(AA^\top)$, the Froebius norm and $\|b^t\|$, the L_2 norm and

$$\frac{d\mathcal{F}[q^t]}{dt} = -\text{tr}\left[A^t (A_\star^t)^\top\right] - (b^t)^\top b_\star^t \tag{A1}$$

To solve this problem, we used the Lagrange multiplier method. We write the Lagrangian as:

$$\mathcal{L}(A^t, b^t) = \frac{d\mathcal{F}[q^t]}{dt} - \lambda_A g(A^t) - \lambda_b h(b^t),$$

where $g(A) = \text{tr}(AA^\top) - 1$ and $h(b) = \|b\|_2^2 - 1$. For simplicity we can divide the problem as:

$$\mathcal{L}(A^t) = -\text{tr}\left[A^t (A_\star^t)^\top\right] - \lambda_A g(A^t)$$
$$\mathcal{L}(b^t) = -(b^t)^\top b_\star^t - \lambda_b h(b^t)$$

For A^t, we have the constraints:

$$\begin{aligned}\nabla_{A^t}\mathrm{tr}\left[A^t(A^t_\star)^\top\right] =& \lambda_A \nabla_{A^t} g(A^t)\\ g(A^t) =& 0\end{aligned}$$

Computing the gradients is straightforward:

$$\begin{aligned}A^t_\star =& 2\lambda_A A^t\\ \Rightarrow A^t =& \frac{A^t_\star}{2\lambda_A}\\ \Rightarrow \frac{1}{4\lambda_A^2}\mathrm{tr}(A^t_\star(A^t_\star)^\top) =& 1\\ \Rightarrow \lambda_A =& \sqrt{\frac{\mathrm{tr}(A^t_\star(A^t_\star)^\top)}{4}}.\end{aligned}$$

which gives us the result $A^t = \frac{A^t_\star}{\|A^t_\star\|_F}$. Similarly for b^t:

$$\begin{aligned}\nabla_{b^t}(b^t)^\top b^t_\star =& \lambda_b \nabla_{b^t} h(b^t)\\ h(b^t) =& 0.\end{aligned}$$

Replacing the gradients gives:

$$\begin{aligned}b^t_\star =& 2\lambda_b b^t\\ \Rightarrow b^t =& \frac{b^t_\star}{2\lambda_b}\\ \Rightarrow \frac{1}{4\lambda_b^2}\|b^t_\star\|_2^2 =& 1\\ \Rightarrow \lambda_b =& \frac{2}{\|b^t_\star\|_2}\end{aligned}$$

which gives us the result $b^t = \frac{b^t_\star}{\|b^t_\star\|_2}$.

Appendix B. Relaxation of the Empirical Free Energy

We prove the decrease in the empirical free energy (17) under the particle flow when the covariance C is nonsingular. We define the empirical distribution $\hat{q}(x) = \frac{1}{N}\sum_{i=1}^N \delta_{x,x_i}$ with a finite number N of particles. The empirical free energy is defined as

$$\mathcal{F}[\hat{q}] = \mathbb{E}_{\hat{q}}[\varphi(x)] - \frac{1}{2}\log|C|.$$

We are interested in the temporal change of the free energy, when particles move under a general linear dynamics

$$\frac{dx_i}{dt} = b + A(x_i - m).$$

The induced dynamics for $\mathcal{F}$ are:

$$\frac{d\mathcal{F}}{dt} = \mathbb{E}_{q^t}\left[\nabla_x\varphi(x)^\top \frac{dx}{dt}\right] - \frac{1}{2}\mathrm{tr}(C^{-1}\frac{dC}{dt})$$

For notational simplicity, we introduce $g(x) = \nabla_x\varphi(x)$ and $\dot{x} = \frac{dx}{dt}$ (similarly $\dot{m} = \frac{dm}{dt}$).

$$\begin{aligned}\frac{dC}{dt} =& \frac{d}{dt}\mathbb{E}_q\left[(x-m)(x-m)^\top\right] \\ =& \mathbb{E}_q\left[(\dot{x}-\dot{m})(x-m)^\top\right] + \mathbb{E}_q\left[(x-m)(\dot{x}-\dot{m})^\top\right] \\ =& \mathbb{E}_q\left[\dot{x}x^\top + x\dot{x}^\top - \dot{m}m^\top - m\dot{m}^\top\right] \\ =& \mathbb{E}_q\left[\dot{x}(x-m)^\top\right] + \mathbb{E}_q\left[(x-m)\dot{x}^\top\right]\end{aligned}$$

$$\begin{aligned}\frac{d\mathcal{F}}{dt} =& \mathbb{E}_q\left[g(x)^\top\dot{x}\right] - \\ & \frac{1}{2}\mathbb{E}_q\left[\mathrm{tr}(C^{-1}\dot{x}(x-m)^\top) + \mathrm{tr}(C^{-1}(x-m)^\top\dot{x}^\top)\right] \\ =& \mathbb{E}_q\left[\dot{x}^\top\left(g(x) - C^{-1}(x-m)\right)\right]\end{aligned} \tag{A2}$$

where we used the permutation properties of the trace.

Plugging the dynamics into Equation (A2), we obtain:

$$\begin{aligned}\frac{d\mathcal{F}}{dt} =& b^\top\mathbb{E}_q[g(x)] + \mathbb{E}_q\left[(x-m)^\top A^\top g(x)\right] \\ & - \mathbb{E}_q\left[(x-m)^\top A^\top C^{-1}(x-m)\right]\end{aligned} \tag{A3}$$

where we used the fact that $b^\top C^{-1}\mathbb{E}_q[x-m] = 0$.

We next look for conditions on b and A, under which $\frac{d\mathcal{F}}{dt} < 0$, i.e., the dynamics will lead to a decrease in the free energy. We pick $b = -\beta_1\mathbb{E}_q[g(x)]$, where $\beta_1 > 0$, and we obtain, for the first term in (A3):

$$-\beta_1\|\mathbb{E}_q[g(x)]\|^2 \leq 0.$$

For A, let us first define $\psi = \mathbb{E}_q\left[g(x)(x-m)^\top\right]$ and rewrite the second and last term of the Equation (A3) as:

$$\begin{aligned}\mathbb{E}_q\left[(x-m)^\top A^\top g(x)\right] =& \mathrm{tr}\left(\mathbb{E}_q\left[A^\top g(x)(x-m)^\top\right]\right) \\ =& \mathrm{tr}\left(A^\top\psi\right) \\ \mathbb{E}_q\left[(x-m)^\top A^\top C^{-1}(x-m)\right] =& \mathrm{tr}\left(A^\top C^{-1}C\right) \\ =& \mathrm{tr}(A)\end{aligned}$$

Combining both, we get $\mathrm{tr}\left(A^\top(\psi - I)\right)$. Similarly to the previous step, we pick $A = -\beta_2(\psi - I)$, where $\beta_2 \geq 0$, which leads to another negative term:

$$-\beta_2\mathrm{tr}((\psi-I)^\top(\psi-I)) \leq 0,$$

where we use the fact that $X^\top X$ is a positive semi-definite matrix for any real valued X.

Note that different forms of A (e.g., β_2 are replaced by a positive definite matrix) could be used, as long as the trace of the product stays positive. Inserting b and A, the free energy dynamics become

$$\frac{d\mathcal{F}}{dt} = -\beta_1\|\mathbb{E}_q[g(x)]\|^2 - \beta_2\mathrm{tr}((\psi-I)^\top(\psi-I))$$

The variable dynamics are given by

$$\begin{aligned}\frac{dx}{dt} &= -\beta_1 \mathbb{E}_q[g(x)] - \beta_2(\psi - I)(x-m) \\ &= -\beta_1 \mathbb{E}_q[g(x)] \\ &\quad - \beta_2\Big(\mathbb{E}_q\Big[g(x)(x-m)^\top\Big] - I\Big)(x-m),\end{aligned}$$

which is equivalent to Equation (5), for $\beta_1 = \beta_2 = 1$. Our result shows that the empirical approximation of the free energy decreases under the particle flow.

Appendix C. Riemannian Gradient for Matrix Parameter Γ

The parameter flow for the matrix Γ in (11) is given by

$$\frac{d\Gamma^t}{dt} = \Gamma^t - \mathbb{E}_{q^0}\Big[\nabla_x \varphi(x^t)(x^0 - m^0)^\top\Big]\Gamma^t(\Gamma^t)^\top.$$

This is easily rewritten in terms of the parameter gradient as $\frac{d\Gamma^t}{dt} = \frac{\partial \mathcal{F}}{\partial \Gamma}\Gamma\Gamma^\top$

Similar to natural gradients, which are defined by the metric, which is induced by the Fisher–matrix, we can rewrite the parameter change in terms of a different *Riemannian* gradient. This gradient is the direction of change $d\Gamma = \Gamma(t+dt) - \Gamma(t)$, which yields the steepest descent of the free energy over a small time interval dt. As an extra condition, one keeps the length of $d\Gamma$ (measured by a 'natural' metric, which has specific invariance properties) fixed. This is defined by an inner product (the squared length) $\langle d\Gamma, d\Gamma\rangle_\Gamma$ in the tangent space of small deviations $d\Gamma$ from the matrix Γ. Hence, $d\Gamma$ is found by minimising $\mathcal{F}(\Gamma(t) + d\Gamma, m)$ (for small $d\Gamma$) under the condition that $\langle d\Gamma, d\Gamma\rangle_{\Gamma(t)}$ is fixed. Following [20] (Theorem 6), a natural metric in the space of symmetric nonsingular matrices can be defined as

$$\langle d\Gamma, d\Gamma\rangle_\Gamma \doteq \mathrm{tr}\Big((d\Gamma\,\Gamma^{-1})^\top d\Gamma\,\Gamma^{-1}\Big).$$

This metric is invariant against multiplications of Γ and $d\Gamma$ by matrices Y, i.e., $\langle d\Gamma, d\Gamma\rangle_\Gamma = \langle d\Gamma\, Y, d\Gamma\, Y\rangle_{\Gamma Y}$ and reduces to the Euclidian metric at the unit matrix $\Gamma = I$.

The direction of the natural gradient is obtained by expanding the free energy for small $d\Gamma$ and introducing a Lagrange–multiplier λ for the constraint. One ends up with the quadratic form

$$\frac{\partial \mathcal{F}}{\partial \Gamma} d\Gamma + \lambda \mathrm{tr}\Big((d\Gamma\,\Gamma^{-1})^\top d\Gamma\,\Gamma^{-1}\Big)$$

to be minimised by $d\Gamma$. By taking the derivative with respect to $d\Gamma$, one finds that the direction of $d\Gamma$ agrees with the right equation of the flow (11).

Appendix D. Regularised Free Energy for $N \le D$

The problem of defining an empirical approximation for $N \le D$ particles is that the empirical covariance becomes singular and typically has $N-1$ nonzero eigenvalues, and thus $|C| = 0$. Note that the extra 0 eigenvalue is derived from the fact that the empirical sum of fluctuations must be zero, which provides an additional linear constraint.

We can regularise the log determinant term by replacing the zero eigenvalues of C: $\lambda_i = 0 \to \tilde{\lambda}_i = 1$. The new covariance $\tilde{C}$ becomes

$$\log|\widetilde{C}| = \sum_{i:\lambda_i > 0} \log \lambda_i,$$

since $\log 1 = 0$. The dynamics of the particles stays the same. To rewrite this formally in terms of matrices, we define

$$\widetilde{C} = C + C_\perp$$

where

$$C_\perp = \sum_{i:\lambda_i=0} e_i e_i^\top$$

and $e_i = i$th eigenvector of C. This replaces all 0 eigenvalues by 1. $C_\perp$ is a projector: $C_\perp^2 = C_\perp$ and $C_\perp(I - C_\perp) = 0$. We also have $\mathrm{tr}(C_\perp) = D - (N-1)$. In the following, it is useful to introduce the $D \times N$ matrix of fluctuations Z, such that $C = ZZ^\top/N$. The column vectors of Z span the subspace of eigenvectors e_i with $\lambda_i > 0$. Hence, it follows that $C_\perp Z = 0$.

We want to show that the regularised free energy $\widetilde{F}$ decreases under the particle dynamics for $N \le D$. Since the part of the time derivative of $\widetilde{F}$ that depends on $\frac{dm}{dt}$ is not changed, we will only discuss the fluctuation part in the following.

It is useful to introduce the matrix:

$$\widetilde{A} \doteq I - C_\perp - gZ^\top/N = A - C_\perp,$$

with $g = \nabla_x \varphi(x)$ is the $D \times N$ matrix of the gradient.

$$\begin{aligned}\mathbb{E}_q\left[g(x)^\top \frac{dx}{dt}\right] =& \mathrm{tr}(A) - \mathrm{tr}(A^\top A)\\ =& \mathrm{tr}(\widetilde{A} + C_\perp) - \mathrm{tr}((\widetilde{A} + C_\perp)^\top(\widetilde{A} + C_\perp))\\ =& \mathrm{tr}(\widetilde{A}) - \mathrm{tr}(\widetilde{A}^\top \widetilde{A}).\end{aligned}$$

To obtain this result, we need

$$\begin{aligned}\mathrm{tr}(C_\perp \widetilde{A}) =& \mathrm{tr}(C_\perp \widetilde{A}^\top)\\ =& \mathrm{tr}(C_\perp(I - C_\perp) - C_\perp Z g^\top/N) = 0.\end{aligned}$$

We need to work out

$$\begin{aligned}-\frac{1}{2}\frac{d\ln|\widetilde{C}|}{dt} =& -\frac{1}{2}\mathrm{tr}\left(\frac{d\widetilde{C}}{dt}\widetilde{C}^{-1}\right)\\ =& -\frac{1}{2}\mathrm{tr}\left(\frac{dC}{dt}\widetilde{C}^{-1}\right)\end{aligned}$$

where we have used the fact that the eigenvalues $\lambda_i = 1$ of $\widetilde{C}$ have a zero time derivative and can be omitted. We use the linear dynamics $\frac{dZ}{dt} = AZ$ to obtain:

$$\begin{aligned}\frac{dC}{dt} =& = CA^\top + AC\\ =& (\widetilde{C} - C_\perp)(\widetilde{A}^\top + C_\perp) + (\widetilde{A} + C_\perp)(\widetilde{C} - C_\perp)\\ =& \widetilde{C}\widetilde{A}^\top + \widetilde{A}\widetilde{C} + C_\perp\widetilde{C} + \widetilde{C}C_\perp - \widetilde{A}C_\perp - C_\perp\widetilde{A}^\top - 2C_\perp\\ =& \widetilde{C}\widetilde{A}^\top + \widetilde{A}\widetilde{C},\end{aligned}$$

where we have used $C_\perp^2 = C_\perp$ and $C_\perp \widetilde{A}^\top = 0$. Hence

$$-\frac{1}{2}\mathrm{tr}\left(\frac{d\widetilde{C}}{dt}\widetilde{C}^{-1}\right) = -\,\mathrm{tr}(\widetilde{A}).$$

Finally, the temporal change in the free energy due to the fluctuations is given by

$$\frac{d\widetilde{\mathcal{F}}}{dt} = -\mathrm{tr}(\widetilde{A}^\top \widetilde{A}) \leq 0.$$

Note that this proof is not only valid for $N \leq D$, but also for $N > D$, as the overall computations are simplified with $C_\perp = 0$. A more detailed proof for $N > D$ is, furthermore, given in Appendix B.

Efficient Computation of $\log\left|\widetilde{C}\right|$

A practical way to compute $\log|\widetilde{C}|$ without performing an eigenvector expansion is to define the $N \times N$ matrix

$$R \doteq Z^\top Z/N + J_{N,N}/N,$$

where $J_{N,N}$ is the $N \times N$ *all-ones* matrix. $Z^\top Z/N$ shares the $N-1$ nonzero eigenvalues with C and has an additional eigenvalue 0 corresponding to the constant eigenvector $(e_N)_i = 1/\sqrt{N}$. Adding an all-ones matrix preserves all existing eigenvalues while replacing the 0 one with a constant. This leads to the following result:

$$-\frac{1}{2}\log|R| = -\frac{1}{2}\sum_{i=1}^{N-1}\log\lambda_i.$$

Appendix E. Proof of Theorem 1: Fixed Points for a Gaussian Model ($N > d$)

Theorem A1 (1)**.** *If the target density $p(x)$ is a D-dimensional multivariate Gaussian, only $D+1$ particles are needed for Algorithm 2 to converge to the exact target parameters.*

The general fixed-point condition for the dynamics (13) of the position x_i for particle i is given by:

$$(I - \mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right])(x_i - m) - \mathbb{E}_{\hat{q}}[g(x)] = 0.$$

for $i = 1, \ldots, N$. By taking the expectation over all particles, we obtain:

$$\mathbb{E}_{\hat{q}}[g(x)] = 0, \tag{A4}$$

where $\hat{q}$ is the empirical distributions of particles at the the fixed point. Note that this result is independent of N, i.e., it is also valid for $N = 1$.

For a D-dimensional Gaussian target $p(x) = \mathcal{N}(\mu, \Sigma)$, we will show that empirical mean and covariance given by the particle algorithm converge to the true mean and covariance matrix of the Gaussian when we use $N \geq D+1$ particles. In this setting, we have $\varphi(x) = \frac{1}{2}x^\top\Sigma^{-1}x - x^\top\Sigma^{-1}\mu$. For simplification, we use the precision matrix $\Lambda = \Sigma^{-1}$ and get

$$\varphi(x) = \frac{1}{2}x^\top\Lambda x - x^\top\Lambda\mu.$$

The gradient $g(x)$ becomes:

$$g(x) = \Lambda(x - \mu)$$

At the fixed points, we have that $\frac{dm}{dt}$ and $\frac{d\Gamma}{dt}$ are equal to 0. For the mean m:

$$\begin{aligned}\frac{dm}{dt} = \mathbb{E}_{\hat{q}}[g(x)] =&0\\ \Lambda\mathbb{E}_{\hat{q}}[x-\mu] =&0\\ \Lambda m =&\Lambda\mu\\ m =&\mu\end{aligned}$$

For the matrix Γ, we have

$$\begin{aligned}\frac{d\Gamma}{dt} = -A\Gamma =&0\\ \Gamma - \mathbb{E}_{q_0}\left[g(x)(x-m)^\top\right]\Gamma =&0\\ \mathbb{E}_{q_0}\left[\Lambda(x-\mu)(x-m)^\top\right]\Gamma =&\Gamma\\ -2\eta_2\mathbb{E}_{q_0}\left[(x-m)(x-m)^\top\right]\Gamma =&\Gamma\\ \Lambda C\Gamma =&\Gamma\\ \Lambda C^2 =&C\end{aligned}$$

where we use the result for the mean $m = \mu$ and right multiplied by $\Gamma^\top$ as $C = \Gamma\Gamma^\top$. Now, we can only simplify, as $C = \Lambda^{-1} = \Sigma$ if C is not singular. This is true only if its rank is equal to D, needing $D+1$ particles.

Appendix F. Proof of Theorem 2: Rates of Convergence for Gaussian Targets

Theorem A2 (2)**.** *For a target $p(x) = \mathcal{N}(x \mid \mu, \Lambda^{-1})$, where $x \in \mathbb{R}^D$, and $N \geq D+1$ particles, the continuous time limit of Algorithm 2 will converge exponentially fast for both the mean and the trace of the precision matrix:*

$$\begin{aligned}m^t - \mu =&e^{-\Lambda t}(m^0 - \mu),\\ \mathrm{tr}\left(\left(C^t\right)^{-1} - \Lambda\right) =&e^{-2t}\mathrm{tr}\left(\left(C^0\right)^{-1} - \Lambda\right),\end{aligned}$$

where m^t and C^t are the empirical mean and covariance matrix at time t and $\exp(-\Lambda t)$ is the matrix exponential.

In the following, we assume the target $p(x) = \mathcal{N}(\mu, \Sigma)$ We use the notation $\Lambda \doteq \Sigma^{-1}$ and $\delta C^t = C^t - \Sigma$.

Appendix F.1. Convergence of the Mean

Given our target $p(x)$, similarly to Appendix E we have $g(x) = \Lambda(x-\mu)$, where $\eta_1 = \Sigma^{-1}\mu$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. This transform the first of Equations (11) into

$$\begin{aligned}\frac{dm}{dt} =&-\Lambda(\mathbb{E}_{\hat{q}}[x] - \mu)\\ =&-\Lambda(m-\mu)\end{aligned}$$

If now consider the error on $m : \delta m = m - \mu$ we obtain:

$$\begin{aligned}\frac{d\delta m}{dt} =&\frac{dm}{dt} = -\Lambda(m-\mu)\\ =&-\Lambda\delta m.\end{aligned}$$

Therefore, the mean converges exponentially fast to the true mean. The asymptotic rate is governed by the largest eigenvalue of Λ, i.e., the inverse of the smallest eigenvalue of Σ, $\lambda_{\min}$.

Appendix F.2. Convergence of the Covariance Matrix

Let $z = x - m$, we have from Equation (5), that

$$\frac{dz}{dt} = -Az$$

where $A = \mathbb{E}_{q_0}\left[g(x)z^\top\right] - I$. This expectation can further be simplified as

$$\mathbb{E}_{\hat{q}}\left[\Lambda(x-\mu)z^\top\right] = \Lambda C, \tag{A5}$$

where $q \sim \mathcal{N}(m, C)$. Hence, we have the exact result

$$\frac{dC}{dt} = (I - \Lambda C)C + C(I - C\Lambda). \tag{A6}$$

We know that the optimal target is $C = \Sigma$. Therefore, we define the error $\delta C = C - \Sigma$. Linearizing Equation (A6) gives us

$$\begin{aligned}\frac{d\delta C}{dt} = \frac{dC}{dt} =& (I - \Lambda(\delta C + \Sigma))(\delta C + \Sigma) \\ &+ (\delta C + \Sigma)(I - (\delta C + \Sigma)\Lambda) \\ =& -\Lambda\delta C(\delta C + \Sigma) - (\delta C + \Sigma)\delta C\Lambda \\ \approx& -\Lambda\delta C\Sigma - \Sigma\delta C\Lambda\end{aligned}$$

We were not yet able to find a general solution of this equation, but we can obtain a simple result for the trace $y^t \doteq \mathrm{tr}(\delta C)$ at time t:

$$\frac{dy^t}{dt} \simeq -2y^t.$$

We, therefore, have a asymptotic linear convergence: $y^t \propto e^{-2t}y^0$ which is independent of the parameters of the Gaussian model.

We can also equivalently obtain a non-asymptotic estimate of a specific error measure for the precision matrix. Using equation (A6), we have the following dynamics for the precision C^{-1}:

$$\begin{aligned}\frac{dC^{-1}}{dt} =& -C^{-1}\frac{dC}{dt}C^{-1} \\ =& -C^{-1}(I - \Lambda C) - (I - \Lambda C)C^{-1}\end{aligned}$$

Taking the trace

$$\begin{aligned}\frac{d\mathrm{tr}(C^{-1})}{dt} =& -2\mathrm{tr}(C^{-1}) - 2\mathrm{tr}(\Lambda) \\ \frac{d\mathrm{tr}(C^{-1} - \Lambda)}{dt} =& -2\mathrm{tr}(C^{-1} - \Lambda)\end{aligned}$$

Hence we get the following exact result:

$$\mathrm{tr}((C^t)^{-1} - \Lambda) = e^{-2t}\mathrm{tr}((C^0)^{-1} - \Lambda)$$

which is again independent of the parameters of the Gaussian model.

Additionally, this tells us that if the covariance C is non-singular at time $t = 0$, it will remain non-singular for all t ($\mathrm{tr}(C^{-1})$ would be infinite). Hence, if we start with $N > d$ particles with a proper empirical covariance, they cannot collapse to make C singular.

Appendix F.3. Convergence of the Trace of the Covariance

The asymptotic result on traces obtained previously can be turned into an exact inequality. We have

$$\frac{d\delta C}{dt} = -\Lambda\delta C\Sigma - \Sigma\Lambda\delta C - \Lambda(\delta C)^2 - (\delta C)^2\Lambda$$

Taking the trace, we get

$$\frac{d\mathrm{tr}(\delta C)}{dt} = -2\mathrm{tr}(\delta C) - 2\mathrm{tr}(\delta C\Lambda\delta C)$$

Since $\delta C\Lambda\delta C$ is positive definite, we have $-2\mathrm{tr}(\delta C\Lambda\delta C) \leq 0$ and thus

$$\frac{d\mathrm{tr}(\delta C)}{dt} \leq -2\mathrm{tr}(\delta C)$$

leading to:

$$\mathrm{tr}(\delta C^t) \leq \mathrm{tr}(\delta C^0)e^{-2t}$$

by using by Grönwall's lemma [46]:

Lemma A1 (Grönwall)**.** *For an interval $I_0 = [0, \infty)$ and a given function f differentiable everywhere in I_0 and satisfying:*

$$f'(t) \leq \beta(t)f(t), \quad t \in I_0$$

then f is bounded by the corresponding differential equation $g'(t) = \beta(t)g(t)$:

$$f(t) \leq f(0)\int_0^t \beta(s)ds, \quad t \in I_0$$

The bound is nontrivial only if $\mathrm{tr}(\delta C) \geq 0$. This would be natural assumption for a Bayesian model, if C^0 is the prior covariance and the eigenvalues of C^t at $t = \infty$ (corresponding to the posterior) are reduced by the data.

Appendix F.4. Decay of Fluctuation Part of the Free Energy

Still focusing on the Gaussian model, we can further derive a bound on the free energy. It is easy to see that for the Gaussian case, the free energy in Equation (4) separates into a sum of two terms. The first one depends on the mean m^t only and the second one on only the fluctuations (i.e., C^t).

We will consider the second, nontrivial part only. We assume that the covariance matrix is nonsingular (corresponding to $N > D$). The fluctuation part of the free energy (minus its minimum) is given by

$$\mathcal{F}_{fl} = -\frac{1}{2}\ln|I - B| - \frac{1}{2}\mathrm{tr}(B)$$

where we have introduced the matrix $B \doteq I - \Lambda C$. One can show that its eigenvalues are real and are upper bounded by 1. First, we can show from the equations of motion that

$$\frac{d\mathcal{F}_{fl}}{dt} = -\mathrm{tr}(BB^\top) \tag{A7}$$

Second, using the elementary bound $-\ln(1-u) \leq \frac{u}{1-u}$ valid for $u \leq 1$ and applied to the eigenvalues of B yields

$$\begin{aligned}\mathcal{F}_{fl} \leq& \frac{1}{2}\mathrm{tr}(B(I-B)^{-1} - B) \\ =& \frac{1}{2}\mathrm{tr}(B(I-B)^{-1} - B(I-B)(I-B)^{-1}) \\ =& \frac{1}{2}\mathrm{tr}(B^2(I-B)^{-1}) \\ =& \frac{1}{2}\mathrm{tr}(B^2C^{-1}\Lambda^{-1}) \leq \frac{1}{2}\mathrm{tr}(B^\top\Lambda^{-1}BC^{-1})\end{aligned}$$

The last two equalities used the definition $B = I - \Lambda C$. Since $B^\top\Lambda^{-1}B$ and C^{-1} are both positive definite, we can bound the last term by (see ([47], Theorem 6.5))

$$\begin{aligned}\mathcal{F}_{fl} \leq \frac{1}{2}\mathrm{tr}(B^\top\Lambda^{-1}B)\mathrm{tr}(C^{-1}) \leq \\ \frac{1}{2}\mathrm{tr}(BB^\top)\mathrm{tr}(\Lambda^{-1})\mathrm{tr}(C^{-1})),\end{aligned}$$

where, in the last line, we have bounded the trace of a product of p.d. matrices a second time.

Combining with Equation (A7) we show that

$$\frac{d\mathcal{F}_{fl}}{dt} \leq -\frac{2\mathcal{F}_{fl}}{\mathrm{tr}(\Lambda^{-1})\mathrm{tr}(C^{-1})}$$

We can plug in our result from Theorem 2:

$$\begin{aligned}\mathrm{tr}(C^{-1}) =& \mathrm{tr}(\Lambda) + \mathrm{tr}(C^{-1} - \Lambda) \\ =& \mathrm{tr}(\Lambda) + e^{-2t}\mathrm{tr}((C^0)^{-1} - \Lambda) \\ \leq& \mathrm{tr}(\Lambda) + e^{-2t}|\mathrm{tr}((C^0)^{-1} - \Lambda)| \\ \leq& \mathrm{tr}(\Lambda) + |\mathrm{tr}((C^0)^{-1} - \Lambda)|\end{aligned}$$

We can plug this in and use Grönwall's Lemma A1 to get an exponential bound

$$\mathcal{F}_{fl}(C^t) \leq \mathcal{F}_{fl}(C^0)e^{-\left[\frac{2t}{\mathrm{tr}(\Lambda^{-1})(\mathrm{tr}(\Lambda)+|\mathrm{tr}((C^0)^{-1}-\Lambda)|)}\right]}.$$

Appendix F.5. Asymptotic Decay of the Free Energy:

For large times t, we can do better. Let us analyse the asymptotic decay constant $\mathcal{F}_{fl} \simeq e^{-\lambda_{free} t}$ defined by

$$\begin{aligned}\lambda_{free} \doteq -\lim_{t\to\infty} \frac{d\ln(\mathcal{F}_{fl})}{dt} &= -\lim \frac{\frac{d\mathcal{F}_{fl}}{dt}}{\mathcal{F}_{fl}} \\ = \lim \frac{\mathrm{tr}(BB^\top)}{-\frac{1}{2}\ln|I-B| - \frac{1}{2}\mathrm{tr}(B)} &\geq \\ \lim \frac{\mathrm{tr}(B^2)}{-\frac{1}{2}\ln|I-B| - \frac{1}{2}\mathrm{tr}(B)}\end{aligned}$$

In the last inequality, we used $\mathrm{tr}(BB^\top) \geq \mathrm{tr}(B^2)$. Everything is expressed by traces of functions of B, and thus by its eigenvalues. Since $B \to 0$ as $t \to \infty$ (this applies also to its eigenvalues u), we can use Taylor's expansion $\ln(1-u) + u = -u^2/2 + O(u^3)$ to show that

$$\lambda_{free} \geq 4$$

which is independent of Λ.

Appendix G. Proof of Theorem 3: Fixed-Points for Gaussian Model ($N \leq D$)

Theorem A3 (3)**.** *Given a D-dimensional multivariate Gaussian target density $p(x) = \mathcal{N}(x|\mu, \Sigma)$, using Algorithm 2 with $N < D + 1$ particles, the empirical mean converges to the exact mean μ. The $N-1$ non-zero eigenvalues of C^t converge to a subset of the target covariance Σ spectrum. Furthermore, the **global minimum** of the regularised version $\widetilde{\mathcal{F}}$ of the free energy (17) corresponds to the **largest** eigenvalues of Σ.*

Applying Equation (A4) to our fixed point equation, we obtain

$$(I - \mathbb{E}_{\hat{q}}\Big[g(x)(x-m)^\top\Big])(x_i - m) = 0,\ \forall i = 1, \ldots, N$$

Hence, the set of centered positions of the particles $S = \{x_i - m\}_{i=1}^N$, are all eigenvectors of the matrix $\mathbb{E}_{\hat{q}}[g(x)(x-m)^\top]$ with eigenvalue 1. S spans a $N-1$ dimensional space (we have $\sum_{i=1}^N (x_i - m) = 0$).

If we specialise to a Gaussian target $p(x) = \mathcal{N}(x \mid \mu, \Sigma)$, (and $\Lambda = \Sigma^{-1}$ we have $g(x) = \Lambda(x-\mu)$ and can reuse the result from Equation (A5):

$$\begin{aligned}\mathbb{E}_{\hat{q}}\Big[g(x)(x-m)^\top\Big] &= \Lambda \mathbb{E}_{\hat{q}}\Big[(x-m)(x-m)^\top\Big] \\ &= \Lambda C.\end{aligned}$$

Using the equality above, we get:

$$\begin{aligned}\Lambda C(x_i - m) &= (x_i - m) \\ C(x_i - m) &= \Sigma(x_i - m),\ \forall i = 1, \ldots, N\end{aligned}$$

which shows that the obtained low-rank covariance C and the target covariance Σ have $N-1$ eigenvectors and eigenvalues in common.

However, are these the largest ones? We look at the modified free energy (17) (ignoring the contribution of the mean):

$$\min \widetilde{\mathcal{F}} = \min \left\{ -\frac{1}{2} \sum_{i:\lambda_i>0} \ln \lambda_i + \operatorname{tr}(\Lambda C) \right\}$$

where λ_i are the eigenvalues of the empirical covariance C. We first note that $\operatorname{tr}(\Lambda C) = N-1$, independent of which eigenvalues are obtained at the fixed point. This is easily seen by the following argument: If we use the index–set $\mathcal{I}$ for the common eigenvectors e_i and eigenvalues λ_i, $i \in \mathcal{I}$, we can write

$$C = \sum_{i\in\mathcal{I}} e_i \lambda_i e_i^\top$$
$$\Sigma = \sum_{i} e_i \lambda_i e_i^\top$$

From this we obtain

$$\operatorname{tr}(\Lambda C) = \operatorname{tr}(\sum_{i\in\mathcal{I}} e_i \lambda_i^{-1} \lambda_i e^\top) = N-1$$

From this result we obtain

$$\min \widetilde{\mathcal{F}} = \max \frac{1}{2} \sum_{i:\lambda_i>0} \ln \lambda_i - (N-1),$$

The term $N-1$ is a constant, but the first term makes a difference: The **absolute minimum** of $\widetilde{\mathcal{F}}$ is achieved, when the λ_i are $N-1$ **largest** eigenvalues of Σ. Our simulations empirically show that the algorithm usually converges to the absolute minimum.

Appendix H. Dimension-Wise Optimizers

Here, we list some of the most populars optimizers used and their dimension-wise versions. In all algorithms, we consider φ the matrix created by the concatenation of the flow of each particle : $\varphi = [\varphi_1, \ldots, \varphi_N]$, where $\varphi_n = \varphi(x_n)$ We additionally use the notation $\varphi_{n,i}$ for the i-th dimension of the flow of the n-th particle. The main differences between the original algorithms and their modified version were put in red.

Appendix H.1. ADAM

The ADAM algorithm is given by:

Algorithm A1: ADAM

```
**Input:** $\varphi^t, m^{t-1}, v^{t-1}, \beta_1, \beta_2, \eta$
**Output:** $\Delta$
$m^t_{n,d} = \beta_1 m^{t-1}_{n,d} + (1-\beta_1)\varphi^t_{n,d}$
$v^t_{n,d} = \beta_2 v^{t-1}_{n,d} + (1-\beta_2)\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{m^t_{n,d}}{(1-\beta_1^t)\left(\sqrt{v^t_{n,d}(1-\beta_2^t)^{-1}}+\epsilon\right)}$
```

Algorithm A2: Dimension-wise ADAM

```
**Input:** $\varphi^t, m^{t-1}, v^{t-1}, \beta_1, \beta_2, \eta$
**Output:** $\Delta$
$m^t_{n,d} = \beta_1 m^{t-1}_{n,d} + (1-\beta_1)\varphi^t_{n,d}$;
$v^t_d = \beta_2 v^{t-1}_d + (1-\beta_2)\frac{1}{N}\sum_{n=1}^{N}\left(\varphi^t_{n,d}\right)^2$;
$\Delta_{n,d} = \eta \frac{m^t_{n,d}}{(1-\beta^t_1)\left(\sqrt{v^t_d(1-\beta^t_2)^{-1}}+\epsilon\right)}$;
```

Appendix H.2. AdaGrad

The AdaGrad algorithm is given by:

Algorithm A3: AdaGrad

```
**Input:** $\varphi^t, v^{t-1}, \eta$
**Output:** $\Delta$
$v^t_{n,d} = v^{t-1}_{n,d} + \left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_{n,d}}+\epsilon}$
```

Algorithm A4: Dimension-wise AdaGrad

```
**Input:** $\varphi^t, v^{t-1}, \eta$
**Output:** $\Delta$
$v^t_d = v^{t-1}_d + \frac{1}{N}\sum_{n=1}^{N}\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_d}+\epsilon}$
```

Appendix H.3. RMSProp

The RMSProp algorithm is given by:

Algorithm A5: RMSProp

```
**Input:** $\varphi^t, v^{t-1}, \rho, \eta$
**Output:** $\Delta$
$v^t_{n,d} = \rho v^{t-1}_{n,d} + (1-\rho)\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_{n,d}}+\epsilon}$
```

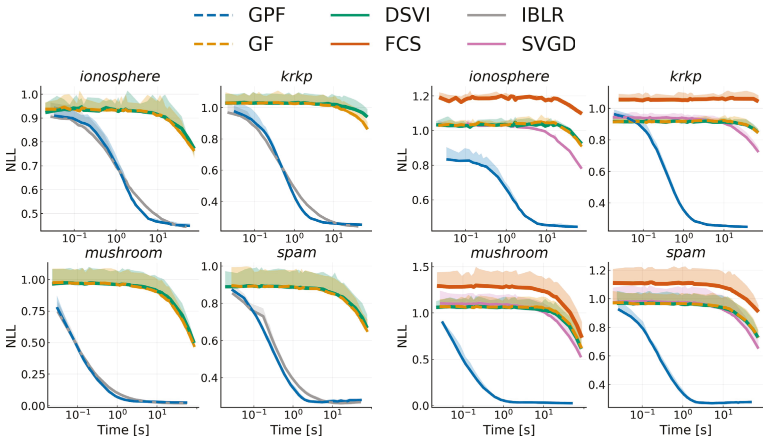
Algorithm A6: Dimension-wise RMSProp

```
**Input:** $\varphi^t, v^{t-1}, \rho, \eta$
**Output:** $\Delta$
$v^t_d = \rho v^{t-1}_d + (1-\rho)\frac{1}{N}\sum_{n=1}^{N}\left(\varphi^t_{n,d}\right)^2$
$\Delta_{n,d} = \eta \frac{\varphi^t_{n,d}}{\sqrt{v^t_d}+\epsilon}$
```

Appendix I. Additional Figures

Appendix I.1. Bayesian Logistic Regression

Similarly to the previous section, we also show results with the RMSProp optimizer with learning rate 1×10^{-4}.



(**a**) Mean-field approximation (**b**) No mean-field approximation

Figure A1. Similarly to Figure 6, we show the average negative log-likelihood on a test-set over 10 runs against training time on different datasets for a Bayesian logistic regression problem. The dashed curve represents the low-rank approximation with RMSProp for methods based on stochastic estimators.

Appendix I.2. Bayesian Neural Network

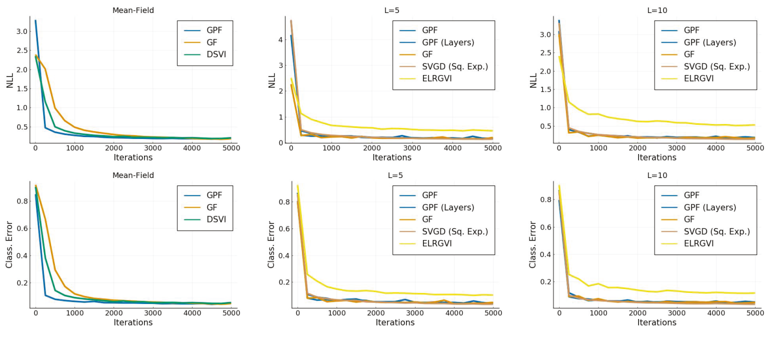


Figure A2. Convergence of the classification error and average negative log-likelihood as a function of time.

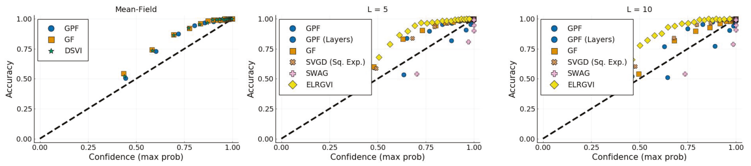


Figure A3. Accuracy vs confidence. Every test sample is clustered in function of its highest predictive probability. The accuracy of this cluster is then computed. A perfectly calibrated estimator would return the identity.

References

1. Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R.P.; de Freitas, N. Taking the human out of the loop: A review of Bayesian optimization. *Proc. IEEE* **2016**, *104*, 148–175. [CrossRef]
2. Settles, B. *Active Learning Literature Survey*; Computer Sciences Technical Report 1648; University of Wisconsin–Madison: Madison, WI, USA, 2009.
3. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; The MIT Press: Cambridge, MA, USA, 2018.
4. Bardenet, R.; Doucet, A.; Holmes, C. On Markov chain Monte Carlo methods for tall data. *J. Mach. Learn. Res.* **2017**, *18*, 1515–1557.
5. Cowles, M.K.; Carlin, B.P. Markov chain Monte Carlo convergence diagnostics: A comparative review. *J. Am. Stat. Assoc.* **1996**, *91*, 883–904. [CrossRef]
6. Barber, D.; Bishop, C.M. Ensemble learning for multi-layer networks. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 1998; pp. 395–401.
7. Graves, A. Practical Variational Inference for Neural Networks. In Proceedings of the 24th International Conference on Neural Information Processing Systems, Granada, Spain, 12–15 December 2011; Volume 24, pp. 2348–2356.
8. Ranganath, R.; Gerrish, S.; Blei, D. Black box variational inference. In Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, Reykjavik, Iceland, 22–25 April 2014; pp. 814–822.
9. Liu, Q.; Lee, J.; Jordan, M. A kernelized Stein discrepancy for goodness-of-fit tests. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 276–284.
10. Liu, Q.; Wang, D. Stein variational gradient descent as moment matching. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018; Volume 32, pp. 8868–8877
11. Zhuo, J.; Liu, C.; Shi, J.; Zhu, J.; Chen, N.; Zhang, B. Message Passing Stein Variational Gradient Descent. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 6018–6027.
12. Opper, M.; Archambeau, C. The variational Gaussian approximation revisited. *Neural Comput.* **2009**, *21*, 786–792. [CrossRef] [PubMed]
13. Challis, E.; Barber, D. Gaussian kullback-leibler approximate inference. *J. Mach. Learn. Res.* **2013**, *14*, 2239–2286.
14. Titsias, M.; Lázaro-Gredilla, M. Doubly stochastic variational Bayes for non-conjugate inference. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1971–1979.
15. Ong, V.M.H.; Nott, D.J.; Smith, M.S. Gaussian variational approximation with a factor covariance structure. *J. Comput. Graph. Stat.* **2018**, *27*, 465–478. [CrossRef]
16. Tan, L.S.; Nott, D.J. Gaussian variational approximation with sparse precision matrices. *Stat. Comput.* **2018**, *28*, 259–275. [CrossRef]
17. Lin, W.; Schmidt, M.; Khan, M.E. Handling the Positive-Definite Constraint in the Bayesian Learning Rule. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; Volume 119, pp. 6116–6126.
18. Hinton, G.E.; van Camp, D. Keeping the Neural Networks Simple by Minimizing the Description Length of the Weights. In Proceedings of the Sixth Annual Conference on Computational Learning Theory, Santa Cruz, CA, USA, 26–28 July 1993; COLT '93; Association for Computing Machinery: New York, NY, USA, 1993; pp. 5–13.
19. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877. [CrossRef]
20. Amari, S.I. Natural Gradient Works Efficiently in Learning. *Neural Comput.* **1998**, *10*, 251–276. [CrossRef]
21. Khan, M.E.; Nielsen, D. Fast yet simple natural-gradient descent for variational inference in complex models. In Proceedings of the International Symposium on Information Theory and Its Applications (ISITA), Singapore, 28–31 October 2018; pp. 31–35.
22. Lin, W.; Khan, M.E.; Schmidt, M. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 3992–4002.
23. Salimbeni, H.; Eleftheriadis, S.; Hensman, J. Natural Gradients in Practice: Non-Conjugate Variational Inference in Gaussian Process Models. In Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics, Lanzarote, Canary Islands, 9–11 April 2018; pp. 689–697.
24. Liu, Q.; Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. *arXiv* **2016**, arXiv:1608.04471.
25. Ba, J.; Erdogdu, M.A.; Ghassemi, M.; Suzuki, T.; Sun, S.; Wu, D.; Zhang, T. Towards Characterizing the High-dimensional Bias of Kernel-based Particle Inference Algorithms. In Proceedings of the 2nd Symposium on Advances in Approximate Bayesian Inference, Vancouver, BC, Canada, 8 December 2019.
26. Tomczak, M.; Swaroop, S.; Turner, R. Efficient Low Rank Gaussian Variational Inference for Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–12 December 2020; Volume 33.
27. Maddox, W.J.; Izmailov, P.; Garipov, T.; Vetrov, D.P.; Wilson, A.G. A simple baseline for bayesian uncertainty in deep learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 13153–13164.
28. Evensen, G. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *J. Geophys. Res. Oceans* **1994**, *99*, 10143–10162. [CrossRef]

29. Rezende, D.; Mohamed, S. Variational inference with normalizing flows. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 1530–1538.
30. Chen, R.T.; Rubanova, Y.; Bettencourt, J.; Duvenaud, D. Neural ordinary differential equations. In Proceedings of the 32nd International Conference on Neural Information Processing, Montréal, QC, Canada, 3–8 December 2018; pp. 6572–6583.
31. Ingersoll, J.E. *Theory of Financial Decision Making*; Rowman & Littlefield: Lanham, MD, USA, 1987; Volume 3.
32. Barfoot, T.D.; Forbes, J.R.; Yoon, D.J. Exactly sparse gaussian variational inference with application to derivative-free batch nonlinear state estimation. *Int. J. Robot. Res.* **2020**, *39*, 1473–1502. [CrossRef]
33. Korba, A.; Salim, A.; Arbel, M.; Luise, G.; Gretton, A. A Non-Asymptotic Analysis for Stein Variational Gradient Descent. In Proceedings of the 32nd International Conference on Neural Information Processing, Virtual, 6–12 December 2020; Volume 33. pp. 4672–4682.
34. Berlinet, A.; Thomas-Agnan, C. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.
35. Zaki, N.; Galy-Fajou, T.; Opper, M. Evidence Estimation by Kullback-Leibler Integration for Flow-Based Methods. In Proceedings of the Third Symposium on Advances in Approximate Bayesian Inference, Virtual Event, January–February 2021.
36. Bezanson, J.; Edelman, A.; Karpinski, S.; Shah, V.B. Julia: A fresh approach to numerical computing. *SIAM Rev.* **2017**, *59*, 65–98. [CrossRef]
37. Tieleman, T.; Hinton, G. *Lecture 6.5-rmsprop, Coursera: Neural Networks for Machine Learning*; Technical Report; University of Toronto: Toronto, ON, USA, 2012.
38. Zhang, G.; Li, L.; Nado, Z.; Martens, J.; Sachdeva, S.; Dahl, G.; Shallue, C.; Grosse, R.B. Which Algorithmic Choices Matter at Which Batch Sizes? Insights From a Noisy Quadratic Model. In *Advances in Neural Information Processing Systems*; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA 2019; Volume 32, pp. 8196–8207.
39. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
40. Dua, D.; Graff, C. UCI Machine Learning Repository. 2017. Available online: https://archive.ics.uci.edu/ml/datasets.php (accessed on 28 July 2021).
41. Agarap, A.F. Deep learning using rectified linear units (relu). *arXiv* **2018**, arXiv:1803.08375.
42. LeCun, Y. The MNIST Database of Handwritten Digits. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 20 July 2021).
43. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1321–1330.
44. Liu, C.; Zhuo, J.; Cheng, P.; Zhang, R.; Zhu, J. Understanding and accelerating particle-based variational inference. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 4082–4092.
45. Zhu, M.H.; Liu, C.; Zhu, J. Variance Reduction and Quasi-Newton for Particle-Based Variational Inference. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020.
46. Gronwall, T.H. Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Ann. Math.* **1919**, *20*, 292–296. [CrossRef]
47. Zhang, F. *Matrix Theory: Basic Results and Techniques*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.

entropy

Article

ABCDP: Approximate Bayesian Computation with Differential Privacy

Mijung Park [1,*], Margarita Vinaroz [2,3] and Wittawat Jitkrittum [4]

1 Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, Canada
2 Max Planck Institute for Intelligent Systems, 72076 Tübingen, Germany; mvinaroz@tuebingen.mpg.de
3 Department of Computer Science, University of Tübingen, 72076 Tübingen, Germany
4 Google Research, 80636 Munich, Germany; wittawat@google.com
* Correspondence: mijungp@cs.ubc.ca

Abstract: We developed a novel approximate Bayesian computation (ABC) framework, *ABCDP*, which produces differentially private (DP) and approximate posterior samples. Our framework takes advantage of the sparse vector technique (SVT), widely studied in the differential privacy literature. SVT incurs the privacy cost only when a condition (whether a quantity of interest is above/below a threshold) is met. If the condition is sparsely met during the repeated queries, SVT can drastically reduce the cumulative privacy loss, unlike the usual case where every query incurs the privacy loss. In ABC, the quantity of interest is the distance between observed and simulated data, and only when the distance is below a threshold can we take the corresponding prior sample as a posterior sample. Hence, applying SVT to ABC is an organic way to transform an ABC algorithm to a privacy-preserving variant with minimal modification, but yields the posterior samples with a high privacy level. We theoretically analyzed the interplay between the noise added for privacy and the accuracy of the posterior samples. We apply ABCDP to several data simulators and show the efficacy of the proposed framework.

Keywords: approximate Bayesian computation (ABC); differential privacy (DP); sparse vector technique (SVT)

Citation: Park, M.; Vinaroz, M.; Jitkrittum, W. ABCDP: Approximate Bayesian Computation with Differential Privacy. *Entropy* **2021**, *23*, 961. https://doi.org/10.3390/e23080961

Academic Editor: Pierre Alquier

Received: 19 May 2021
Accepted: 20 July 2021
Published: 27 July 2021

1. Introduction

Approximate Bayesian computation (ABC) aims to identify the posterior distribution over simulator parameters. The posterior distribution is of interest as it provides the mechanistic understanding of the stochastic procedure that directly generates data in many areas such as climate and weather, ecology, cosmology, and bioinformatics [1–4].

Under these complex models, directly evaluating the likelihood of data is often intractable given the parameters. ABC resorts to an approximation of the likelihood function using simulated data that are *similar* to the actual observations.

In the simplest form of ABC called *rejection ABC* [5], we proceed by sampling multiple model parameters from a prior distribution π: $\theta_1, \theta_2, \ldots \sim \pi$. For each θ_t, a pseudo dataset Y_t is generated from a simulator (the forward sampler associated with the intractable likelihood $\mathrm{P}(y|\theta)$). The parameter θ_t for which the generated Y_t are similar to the observed Y^*, as decided by $\rho(Y_t, Y^*) < \epsilon_{abc}$, is accepted. Here, ρ is a notion of distance, for instance, L2 distance between Y_t and Y^* in terms of a pre-chosen summary statistic. Whether the distance is small or large is determined by ϵ_{abc}, a *similarity threshold*. The result is samples $\{\theta_t\}_{t=1}^M$ from a distribution, $\tilde{\mathrm{P}}_\epsilon(\theta|Y^*) \propto \pi(\theta)\tilde{\mathrm{P}}_\epsilon(Y^*|\theta)$, where $\tilde{\mathrm{P}}_\epsilon(Y^*|\theta) = \int_{B_\epsilon(Y^*)} \mathrm{P}(Y|\theta)dY$ and $B_\epsilon(Y^*) = \{Y : \rho(Y, Y^*) < \epsilon_{abc}\}$. As the likelihood computation is approximate, so is the posterior distribution. Hence, this framework is named by *approximate* Bayesian computation, as we do not compute the likelihood of data explicitly.

Most ABC algorithms evaluate the data similarity in terms of summary statistics computed by an aggregation of individual datapoints [6–11]. However, this seemingly innocuous step of similarity check could impose a privacy threat, as aggregated statistics could still reveal an individual's participation to the dataset with the help of combining other publicly available datasets (see [12,13]). In addition, in some studies, the actual observations are privacy-sensitive in nature, e.g., Genotype data for estimating tuberculosis transmission parameters [14]. Hence, it is necessary to privatize the step of similarity check in ABC algorithms.

In this light, we introduce an ABC framework that obeys the notion of *differential privacy*. The differential privacy definition provides a way to quantify the amount of information that the distance computed on the privacy-sensitive data contains, whether or not a single individual's data are included (or modified) in the data [15]. Differential privacy also provides rigorous privacy guarantees in the presence of *arbitrary side information* such as similar public data.

A common form of applying DP to an algorithm is by adding noise to outputs of the algorithm, called *output perturbation* [16]. In the case of ABC, we found that *adding noise to the distance* computed on the real observations and pseudo-data suffices for the privacy guarantee of the resulting posterior samples. However, if we choose to simply add noise to the distance in every ABC inference step, this DP-ABC inference imposes an additional challenge due to the *repeated* use of the real observations. The *composition* property of differential privacy states that the privacy level degrades over the repeated use of data. To overcome this challenge, we adopt the *sparse vector technique* (SVT) [17], and apply it to the rejection ABC paradigm. The SVT outputs *noisy* answers of whether or not a stream of queries is above a certain threshold, where privacy cost incurs only when the SVT outputs at most c "above threshold" answers. This is a significant saving in privacy cost, as arbitrarily many "below threshold" answers are privacy cost free.

We name our framework, which combines ABC with SVT, as *ABCDP* (approximate Bayesian computation with differential privacy). Under ABCDP, we theoretically analyze the effect of noise added to the distance in the resulting posterior samples and the subsequent posterior integrals. Putting together, we summarize our main contributions:

1. We provide a novel ABC framework, ABCDP, which combines the *sparse vector technique* (SVT) [17] with the rejection ABC paradigm. The resulting ABCDP framework can improve the trade-off between the privacy and accuracy of the posterior samples, as the privacy cost under ABCDP is a function of the number of *accepted* posterior samples only.
2. We theoretically analyze ABCDP by focusing on the effect of noisy posterior samples in terms of two quantities. The first quantity provides the probability of an output of ABCDP being different from that of ABC at any given time during inference. The second quantity provides the convergence rate, i.e., how fast the posterior integral using ABCDP's noisy samples' approaches that using non-private ABC's samples. We write both quantities as a function of added noise for privacy to better understand the characteristics of ABCDP.
3. We validate our theory in the experiments using several simulators. The results of these experiments are consistent with our theoretical findings on the flip probability and the average error induced by the noise addition for privacy.

Unlike other existing ABC frameworks that typically rely on a pre-specified set of summary statistics, we use a kernel-based distance metric called *maximum mean discrepancy* following K2-ABC [18] to eliminate the necessity of pre-selecting a summary statistic. Using a kernel for measuring similarity between two empirical distributions was also proposed in K-ABC [19]. K-ABC formulates ABC as a problem of estimating a conditional mean embedding operator mapping (induced by a kernel) from summary statistics to corresponding parameters. However, unlike our algorithm, K-ABC still relies on a particular choice of summary statistics. In addition, K-ABC is a soft-thresholding ABC algorithm, while ours is a rejection-ABC algorithm.

To avoid the necessity of pre-selecting summary statistics, one could resort to methods that automatically or semi-automatically learn the best summary statistics given in a dataset, and use the learned summary statistics in our ABCDP framework. An example is semi-auto ABC [6], where the authors suggest using the posterior mean of the parameters as a summary statistic. Another example is the indirect-score ABC [20], where the authors suggest using an auxiliary model which determines a score vector as a summary statistic. However, the posterior mean of the parameters in semi-auto ABC as well as the parameters of the auxiliary model in indirect-score ABC need to be estimated. The estimation step can incur a further privacy loss if the real data need to be used for estimating them. Our ABCDP framework does not involve such an estimation step and is more economical in terms of privacy budget to be spent than semi-auto ABC and indirect-score ABC.

2. Background

We start by describing relevant background information.

2.1. Approximate Bayesian Computation

Given a set Y^* containing observations, **rejection ABC** [5] yields samples from an approximate posterior distribution by repeating the following three steps:

$$\theta \sim \pi(\theta), \tag{1}$$

$$Y = \{y_1, y_2, \ldots\} \sim \mathrm{P}(y|\theta), \tag{2}$$

$$\mathrm{P}_{\epsilon_{abc}}(\theta|Y^*) \sim \mathrm{P}_{\epsilon_{abc}}(Y^*|\theta)\pi(\theta), \tag{3}$$

where the pseudo dataset Y is compared with the observations Y^* via:

$$\mathrm{P}_{\epsilon_{abc}}(Y^*|\theta) = \int_{B_{\epsilon_{abc}}(Y^*)} \mathrm{P}(Y|\theta)dY,$$

$$B_{\epsilon_{abc}}(Y^*) = \{Y|\rho(Y, Y^*) \leq \epsilon_{abc}\}, \tag{4}$$

where ρ is a divergence measure between two datasets. Any distance metric can be used for ρ. For instance, one can use the L2 distance under two datasets in terms of a pre-chosen set of summary statistics, i.e., $\rho(Y, Y^*) = D(S(Y), S(Y^*))$, with an L2 distance measure D on the statistics computed by S.

A more statistically sound choice for ρ would be *maximum mean discrepancy* (MMD, [21]) as used in [18]. Unlike a pre-chosen finite dimensional summary statistic typically used in ABC, MMD compares two distributions in terms of all the possible moments of the random variables described by the two distributions. Hence, ABC frameworks using the MMD metric such as [18] can avoid the problem of non-sufficiency of a chosen summary statistic that may occur in many ABC methods. For this reason, in this paper, we demonstrate our algorithm using the MMD metric. However, other metrics can be used as we illustrated in our experiments.

Maximum Mean Discrepancy

Assume that the data $Y \subset \mathcal{X}$ and let $k\colon \mathcal{X} \times \mathcal{X}$ be a positive definite kernel. MMD between two distributions P, Q is defined as

$$\mathrm{MMD}^2(P, Q) = \mathbb{E}_{x,x' \sim P} k(x, x') + \mathbb{E}_{y,y' \sim Q} k(y, y') - 2\mathbb{E}_{x \sim P}\mathbb{E}_{y \sim Q} k(x, y). \tag{5}$$

By following the convention in kernel literature, we call MMD^2 simply MMD.

The Moore–Aronszajn theorem states that there is a unique Hilbert space $\mathcal{H}$ on which k defines an inner product. As a result, there exists a feature map $\phi\colon \mathcal{X} \to \mathcal{H}$ such that $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}} = \langle \cdot, \cdot \rangle$ denotes the inner product on $\mathcal{H}$. The MMD in (5) can be written as

$$\mathrm{MMD}^2(P, Q) = \|\mathbb{E}_{x \sim P}[\phi(x)] - \mathbb{E}_{y \sim Q}[\phi(y)]\|^2_{\mathcal{H}},$$

where $\mathbb{E}_{x\sim P}[\phi(x)] \in \mathcal{H}$ is known as the (kernel) mean embedding of P, and exists if $\mathbb{E}_{x\sim P}\sqrt{k(x,x)} < \infty$ [22]. The MMD can be interpreted as the distance between the mean embeddings of the two distributions. If k is a characteristic kernel [23], then $P \mapsto \mathbb{E}_{x\sim P}[\phi(x)]$ is injective, implying that $\mathrm{MMD}(P,Q) = 0$, if and only if $P = Q$. When P,Q are observed through samples $X_m = \{x_i\}_{i=1}^m \sim P$ and $Y_n = \{y_i\}_{i=1}^n \sim Q$, MMD can be estimated by empirical averages [21] (Equation (3)): $\widehat{\mathrm{MMD}}^2(X_m, Y_n) = \frac{1}{m^2}\sum_{i,j=1}^m k(x_i,x_j) + \frac{1}{n^2}\sum_{i,j=1}^n k(y_i,y_j) - \frac{2}{mn}\sum_{i=1}^m\sum_{j=1}^n k(x_i,y_j)$. When applied in the ABC setting, one input to $\widehat{\mathrm{MMD}}$ is the observed dataset Y^* and the other input is a pseudo dataset $Y_t \sim p(\cdot|\theta_t)$ generated by the simulator given $\theta_t \sim \pi(\theta)$.

2.2. Differential Privacy

An output from an algorithm that takes in sensitive data as input will naturally contain some information of the sensitive data $\mathcal{D}$. The goal of differential privacy is to augment such an algorithm so that useful information about the population is retained, while sensitive information such as an individual's participation in the dataset cannot be learned [17]. A common way to achieve these two seemingly paradoxical goals is by deliberately injecting a controlled level of random noise to the to-be-released quantity. The modified procedure, known as a DP mechanism, now gives a stochastic output due to the injected noise. In the DP framework, a higher level of noise provides stronger privacy guarantee at the expense of less accurate population-level information that can be derived from the released quantity. Less noise added to the output thus reveals more about an individual's presence in the dataset.

More formally, given a mechanism $\mathcal{M}$ (a *mechanism* takes a dataset as input and produces stochastic outputs) and neighboring datasets $\mathcal{D}, \mathcal{D}'$ differing by a single entry (either by replacing one's datapoint with another, or by adding/removing a datapoint to/from $\mathcal{D}$), the *privacy loss* of an outcome o is defined by

$$L^{(o)} = \log \frac{\mathrm{P}(\mathcal{M}(\mathcal{D}) = o)}{\mathrm{P}(\mathcal{M}(\mathcal{D}') = o)}. \tag{6}$$

The mechanism $\mathcal{M}$ is called ϵ-DP if and only if $|L^{(o)}| \le \epsilon$, for all possible outcomes o and for all possible neighboring datasets $\mathcal{D}, \mathcal{D}'$. The definition states that a single individual's participation in the data does not change the output probabilities by much; this limits the amount of information that the algorithm reveals about any one individual. A weaker or an *approximate* version of the above notion is (ϵ,δ)-DP: $\mathcal{M}$ is (ϵ,δ)-DP if $|L^{(o)}| \le \epsilon$, with probability $1-\delta$, where δ is often called a failure probability which quantifies how often the DP guarantee of the mechanism fails.

Output perturbation is a commonly used DP mechanism to ensure the outputs of an algorithm to be differentially private. Suppose a deterministic function $h : \mathcal{D} \mapsto \mathbb{R}^p$ computed on sensitive data $\mathcal{D}$ outputs a p-dimensional vector quantity. In order to make h private, we can add noise to the output of h, where the level of noise is calibrated to the *global sensitivity* [24], Δ_h, defined by the maximum difference in terms of some norm $||h(\mathcal{D}) - h(\mathcal{D}')||$ for neighboring $\mathcal{D}$ and $\mathcal{D}'$ (i.e., differ by one data sample).

There are two important properties of differential privacy. First, the *post-processing invariance* property [24] tells us that the composition of any arbitrary data-independent mapping with an (ϵ,δ)-DP algorithm is also (ϵ,δ)-DP. Second, the *composability* theorem [24] states that the strength of privacy guarantee degrades with the repeated use of DP-algorithms. Formally, given an ϵ_1-DP mechanism $\mathcal{M}_1$ and an ϵ_2-DP mechanism $\mathcal{M}_2$, the mechanism $\mathcal{M}(\mathcal{D}) := (\mathcal{M}_1(\mathcal{D}), \mathcal{M}_2(\mathcal{D}))$ is $(\epsilon_1+\epsilon_2)$-DP. This composition is often-called *linear* composition, under which the total privacy loss linearly increases with the number of repeated use of DP-algorithms. The *strong* composition [17] [Theorem 3.20] improves the linear composition, while the resulting DP guarantee becomes weaker (i.e., approximate (ϵ,δ)-DP). Recently, more refined methods further improve the privacy loss (e.g., [25]).

2.3. AboveThreshold and Sparse Vector Technique

Among the DP mechanisms, we will utilize *AboveThreshold* and *sparse vector technique* (SVT) [17] to make the rejection ABC algorithm differentially private. AboveThreshold outputs 1 when a query value exceeds a pre-defined threshold, or 0 otherwise. This resembles rejection ABC where the output is 1 when the distance is less than a chosen threshold. To ensure the output is differentially private, AboveThreshold adds noise to both the threshold and the query value. We take the same route as AboveThreshold to make our ABCDP outputs differentially private. Sparse vector technique (SVT) consists of c calls to AboveThreshold, where c in our case determines how many posterior samples ABCDP releases.

Before presenting our ABCDP framework, we first describe the privacy setup we consider in this paper.

3. Problem Formulation

We assume a *data owner* who owns sensitive data Y^* and is willing to contribute to the posterior inference.

We also assume a *modeler* who aims to learn the posterior distribution of the parameters of a simulator. Our ABCDP algorithm proceeds with the two steps:

1. *Non-private step:* The modeler draws a parameter sample $\theta_t \sim \pi(\theta)$; then generates a pseudo-dataset Y_t, where $Y_t \sim \mathrm{P}(y|\theta_t)$ for $t = 1, \cdots, T$ for a large T. We assume these parameter-pseudo-data pairs $\{\theta_t, Y_t\}_{t=1}^{T}$ are publicly available (even to an adversary).
2. *Private step:* the data owner takes the whole sequence of parameter-pseudo-data pairs $\{(\theta_t, Y_t)\}_{t=1}^{T}$ and runs our ABCDP algorithm in order to output a set of *differentially private* binary indicators determining whether or not to accept each θ_t.

Note that T is the maximum number of parameter-pseudo-data pairs that are publicly available. We will run our algorithm for T steps, while our algorithm can terminate as soon as we output the c number of accepted posterior samples. So, generally, $c \ll T$. The details are then introduced.

4. ABCDP

Recall that the only place where the real data Y^* appear in the ABC algorithm is when we judge whether the simulated data are similar to the real data, i.e., as in (4). Our method hence adds noise to this step. In order to take advantage of the privacy analysis of SVT, we also add noise to the ABC threshold and to the ABC distance. Consequently, we introduce two perturbation steps.

Before we introduce them, we describe the global sensitivity of the distance, as this quantity tunes the amount of noise we will add in the two perturbation steps. For $\rho(Y^*, Y) = \widehat{\mathrm{MMD}}(Y^*, Y)$ with a bounded kernel, then the sensitivity of the distance is $\Delta_\rho = O(1/N)$ as shown in Lemma 1.

Lemma 1 ($\Delta_\rho = O(1/N)$ for MMD)**.** *Assume that Y^* and each pseudo dataset Y_t are of the same cardinality N. Set $\rho(Y^*, Y) = \widehat{\mathrm{MMD}}(Y^*, Y)$ with a kernel k bounded by $B_k > 0$, i.e., $\sup_{x,y \in \mathcal{X}} k(x, y) \leq B_k < \infty$. Then:*

$$\sup_{(Y^*, Y^{*'}), Y} |\rho(Y^*, Y) - \rho(Y^{*'}, Y)| \leq \Delta_\rho := \frac{2}{N}\sqrt{B_k}$$

and $\sup_{Y^*, Y} \rho(Y^*, Y) \leq 2\sqrt{B_k}$.

A proof is given in Appendix B. For $\rho = \widehat{\mathrm{MMD}}$ using a Gaussian kernel, $k(x, y) = \exp\left(-\frac{\|x-y\|^2}{2l^2}\right)$ where $l > 0$ is the bandwidth of the kernel, $B_k = 1$ for any $l > 0$.

Now, we introduce the two perturbation steps used in our algorithm summarized in Algorithm 1.

Algorithm 1 Proposed c-sample ABCDP

```
Require: Observations $Y^*$, Number of accepted posterior sample size $c$, privacy tolerance $\epsilon_{total}$, ABC threshold $\epsilon_{abc}$, distance $\rho$, and parameter-pseudo-data pairs $\{(\boldsymbol{\theta}_t, Y_t)\}_{t=1}^T$, and option RESAMPLE.
Ensure: $\epsilon_{total}$-DP indicators $\{\tilde{\tau}_t\}_{t=1}^T$ for corresponding samples $\{\boldsymbol{\theta}_t\}_{t=1}^T$
 1: Calculate the noise scale $b$ by Theorem 1.
 2: Privatize ABC threshold: $\hat{\epsilon}_{abc} = \epsilon_{abc} + m_t$ via (7)
 3: Set count=0
 4: for $t = 1, \ldots, T$ do
 5:   Privatize distance: $\hat{\rho}_t = \rho(Y^*, Y_t) + \nu_t$ via (8)
 6:   if $\hat{\rho}_t \leq \hat{\epsilon}_{abc}$ then
 7:     Output $\tilde{\tau}_t = 1$
 8:     count = count+1
 9:     if RESAMPLE then
10:       $\hat{\epsilon}_{abc} = \epsilon_{abc} + m_t$ via (7)
11:     end if
12:   else
13:     Output $\tilde{\tau}_t = 0$
14:   end if
15:   if count $\geq$ c then
16:     Break the loop
17:   end if
18: end for
```

Step 1: Noise for privatizing the ABC threshold.

$$\hat{\epsilon}_{abc} = \epsilon_{abc} + m_t \tag{7}$$

where $m_t \sim \text{Lap}(b)$, i.e., drawn from the zero-mean Laplace distribution with a scale parameter b.

Step 2: Noise for privatizing the distance.

$$\hat{\rho}_t = \rho(Y^*, Y_t) + \nu_t \tag{8}$$

where $\nu_t \sim \text{Lap}(2b)$.

Due to these perturbations, Algorithm 1 runs with the privatized threshold and distance. We can choose to perturb the threshold only once, or every time we output 1 by setting RESAMPLE to false or true. After outputting c number of 1's, the algorithm is terminated. How do we calculate the resulting privacy loss under the different options we choose?

We formally state the relationship between the noise scale and the final privacy loss ϵ_{tot} for the Laplace noise in Theorem 1.

Theorem 1 (Algorithm 1 is ϵ_{total}-DP)**.** *For any neighboring datasets $Y^*, Y^{*'}$ of size N and any dataset Y, assume that ρ is such that $0 < \sup_{(Y^*, Y^{*'}), Y} |\rho(Y^*, Y) - \rho(Y^{*'}, Y)| \leq \Delta_\rho < \infty$. Algorithm 1 is ϵ_{total}-DP, where:*

$$\epsilon_{total} = \begin{cases} \frac{(c+1)\Delta_\rho}{b} & \text{if RESAMPLE is False,} \\ \frac{2c\Delta_\rho}{b} & \text{if RESAMPLE is True.} \end{cases} \tag{9}$$

A proof is given in Appendix A. The proof uses linear composition, i.e., the privacy level linearly degrading with c. However, using the strong composition or more advanced compositions can reduce the resulting privacy loss, while these compositions turn pure-DP into a weaker, approximate-DP. In this paper, we focus on the pure-DP. For the case of RESAMPLE = True, the proof directly follows the proof of the standard SVT algorithm using the linear composition method [17], with an exception that we utilize the quantity representing the minimum noisy value of any query evaluated on Y^*, as opposed to the maximum utilized in SVT. For the case of RESAMPLE= False, the proof follows the proof of Algorithm 1 in [26].

Note that the DP analysis in Theorem 1 holds for other types of distance metrics and not limited to only MMD, as long as there is a bounded sensitivity Δ_ρ under the chosen metric. When there is no bounded sensitivity, one could impose a clipping bound C to the distance by taking the distance from $\min[\rho(Y_t, Y^*), C]$, such that the resulting distance between any pseudo data Y_t and $Y^{*'}$ with modifying one datapoint in Y^* cannot exceed that clipping bound. In fact, we use this trick in our experiments when there is no bounded sensitivity.

4.1. Effect of Noise Added to ABC

Here, we would like to analyze the effect of noise added to ABC. In particular, we are interested in analyzing the probability that the output of ABCDP differs from that of ABC: $\mathbb{P}[\tilde{\tau}_t \neq \tau_t | \tau_t]$ at any given time t. To compute this probability, we first compute the probability density function (PDF) of the random variables $m_t - \nu_t$ in the following Lemma.

Lemma 2. *Recall $m_t \sim Lap(b)$, $\nu_t \sim Lap(2b)$. The subtraction of these yields another random variable $Z = m_t - \nu_t$, where the PDF of Z is given by*

$$f_Z(z) = \frac{1}{6b}\left[2\exp\left(-\frac{|z|}{2b}\right) - \exp\left(-\frac{|z|}{b}\right)\right]. \tag{10}$$

Furthermore, for $a \geq 0$, $G_b(a) := \int_a^\infty f_Z(z)\,\mathrm{d}z = \frac{1}{6}\left[4\exp\left(-\frac{a}{2b}\right) - \exp\left(-\frac{a}{b}\right)\right]$, and the CDF of Z is given by $F_Z(a) = H[a] + (1 - 2H[a])G_b(|a|)$ where $H[a]$ is the Heaviside step function.

See Appendix C for the proof. Using this PDF, we now provide the following proposition:

Proposition 1. *Denote the output of Algorithm 1 at time t by $\tilde{\tau}_t \in \{0, 1\}$ and the output of ABC by $\tau_t \in \{0, 1\}$. The flip probability, the probability that the outputs of ABCDP and ABC differ given the output of ABC, is given by $P[\tilde{\tau}_t \neq \tau_t | \tau_t] = G_b(|\rho_t - \epsilon_{abc}|)$, where $G_b(a)$ is defined in Lemma 2, and $\rho_t := \rho(Y^*, Y_t)$.*

See Appendix D for proof.

To provide an intuition of Proposition 1, we visualize the flip probability in Figure 1. This flip probability provides a guideline for choosing the accepted sample size c given the datasize N and the desired privacy level ϵ_{total}. For instance, if a given dataset is extremely small, e.g., containing datapoints on the order of 10, c has to be chosen such that the flip probability of each posterior sample remains low for a given privacy guarantee (ϵ_{total}). If a higher number of posterior samples are needed, then one has to reduce the desired privacy level for the posterior sample of ABCDP to be similar to that of ABC. Otherwise, with a small ϵ_{total} with a large c, the accepted posterior samples will be poor. On the other hand, if the dataset is bigger, then a larger c can be taken for a reasonable level of privacy.

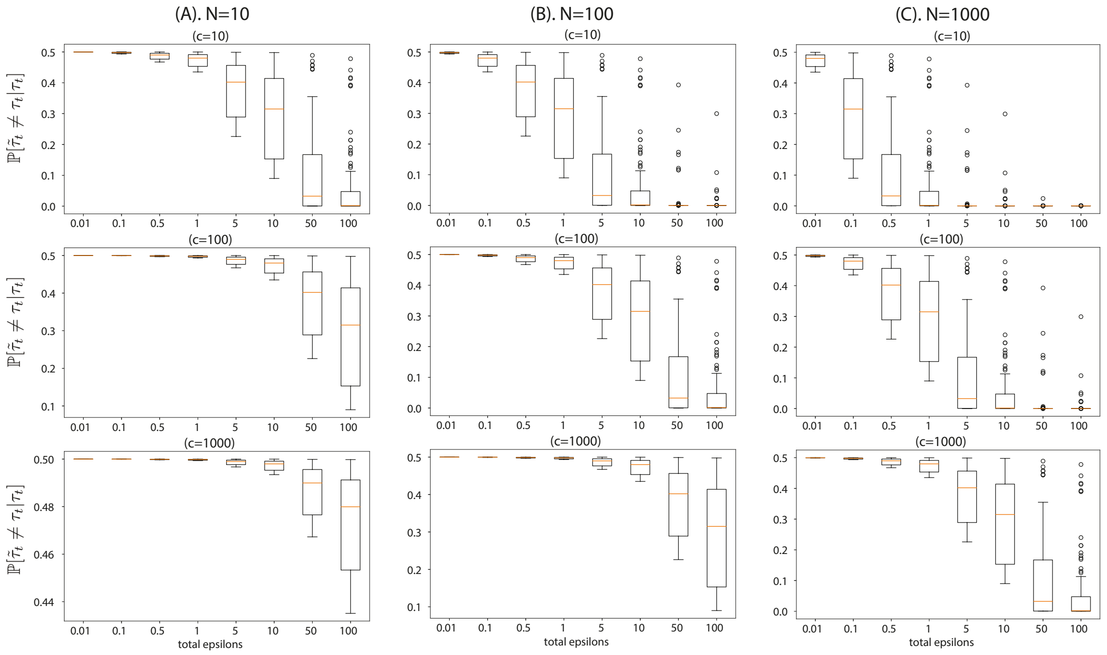


Figure 1. Visualization of flip probability derived in Proposition 1, the probability that the outputs of ABCDP and ABC differ given an output of ABC, with different dataset size N and accepted posterior sample size c. We simulated $\rho \sim \text{Uniform}[0,1]$ (drew 100 values for ρ) and used $\epsilon_{abc} = 0.2$: (**A**) This column shows the flip probability at a regime of extremely small datasets, $N = 10$. Top plot shows the probability at $c = 10$, middle plot at $c = 100$, and bottom plot at $c = 1000$. In this regime, even $\epsilon_{total} = 100$ cannot reduce the flip probability to perfectly zero when $c = 10$. The flip probability remains high when we accept more samples, i.e., $c = 1000$; (**B**) the flip probability at $N = 100$; (**C**) the flip probability at $N = 1000$. As we increase the dataset size N (moving from the left to right columns), the flip probability approaches zero at a smaller privacy loss ϵ_{total}.

4.2. Convergence of Posterior Expectation of Rejection-ABCDP to Rejection-ABC.

The flip probability studied in Section 4.1 only accounts for the effect of noise added to a single output of ABCDP. Building further on this result, we analyzed the discrepancy between the posterior expectations derived from ABCDP and from the rejection ABC. This analysis requires quantifying the effect of noise added to the whole sequence of outputs of ABCDP. The result is presented in Theorem 2.

Theorem 2. *Given Y^* of size N, and $\{(\boldsymbol{\theta}_t, Y_t)\}_{t=1}^T$ as input, let $\tilde{\tau}_t \in \{0,1\}$ be the output from Algorithm 1 where $\tilde{\tau}_t = 1$ indicates that $(\boldsymbol{\theta}_t, Y_t)$ is accepted, for $t = 1, \ldots, T$. Similarly, let τ_t denote the output from the traditional rejection ABC algorithm, for $t = 1, \ldots, T$. Let f be an arbitrary vector-valued function of $\boldsymbol{\theta}$. Assume that the numbers of accepted samples from Algorithm 1, and the traditional rejection ABC algorithm are $c := \sum_{t=1}^T \tilde{\tau}_t \geq 1$ and $c' := \sum_{t=1}^T \tau_t \geq 1$, respectively. Let $b = \frac{4c\sqrt{B_k}}{\epsilon_{total}N}$ if RESAMPLE=True, and $b = \frac{2(c+1)\sqrt{B_k}}{\epsilon_{total}N}$ if RESAMPLE=False (see Theorem 1). Define $K_T := \max_{t=1,\ldots,T} \|f(\boldsymbol{\theta}_t)\|_2$. Then, the following statements hold for both RESAMPLE options:*

1. $\mathbb{E}_{\tilde{\tau}_1,\ldots,\tilde{\tau}_T}\left\| \frac{1}{c}\sum_{t=1}^T f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'}\sum_{t=1}^T f(\boldsymbol{\theta}_t)\tau_t \right\|_2 \leq \frac{2K_T}{c'}\sum_{t=1}^T G_b(|\rho_t - \epsilon_{abc}|)$, *where the decreasing function $G_b(x) \in (0, \frac{1}{2}]$ for any $x \geq 0$ is defined in Lemma 2;*
2. $\mathbb{E}_{\tilde{\tau}_1,\ldots,\tilde{\tau}_T}\left\| \frac{1}{c}\sum_{t=1}^T f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'}\sum_{t=1}^T f(\boldsymbol{\theta}_t)\tau_t \right\|_2 \to 0$ *as $N \to \infty$;*

3. For any $a > 0$:

$$P\left(\left\|\frac{1}{c}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t\right\|_2 \leq a\right) \geq 1 - \frac{4K_T}{3ac'}\sum_{t=1}^{T}\exp\left(-\frac{|\rho_t - \epsilon_{abc}|}{2b}\right)$$

where the probability is taken with respect to $\tilde{\tau}_1, \ldots, \tilde{\tau}_T$.

Theorem 2 contains three statements. The first states that the expected error between the two posterior expectations of an arbitrary function f is bounded by a constant factor of the sum of the flip probability in each rejection/acceptance step. As we have seen in Section 4.1, the flip probability is determined by the scale parameter b of the Laplace distribution. Since $b = O(1/N)$ (see Theorem 1 and Lemma 1), it follows that the expected error decays as N increases, giving the second statement.

The third statement gives a probabilistic bound on the error. The bound guarantees that the error decays exponentially in N. Our proof relies on establishing an upper bound on the error as a function of the total number of flips $\sum_{t=1}^{T} |\tilde{\tau}_t - \tau_t|$ which is a random variable. Bounding the error of interest then amounts to characterizing the tail behavior of this quantity. Observe that in Theorem 2, we consider ABCDP and rejection ABC with the same computational budget, i.e., the same total number of iterations T performed. However, the number of accepted samples may be different in each case (c for ABCDP and c' for reject ABC). The fact that c itself is a random quantity due to injected noise presents its own technical challenge in the proof. Our proof can be found in Appendix E.

5. Related Work

Combining DP with ABC is relatively novel. The only related work is [27], which states that a rejection ABC algorithm produces posterior samples from the exact posterior distribution given perturbed data, when the kernel and bandwidth of rejection ABC are chosen in line with the data perturbation mechanism. The focus of [27] is to identify the condition when the posterior becomes exact in terms of the kernel and bandwidth of the kernel through the lens of data perturbation. On the other hand, we use the sparse vector technique to reduce the total privacy loss. The resulting theoretical studies including the flip probability and the error bound on the posterior expectation are new.

6. Experiments

6.1. Toy Examples

We start by investigating the interplay between ϵ_{abc} and ϵ_{total}, in a synthetic dataset where the ground truth parameters are known. Following [18], we also consider a symmetric Dirichlet prior π and a likelihood $p(y|\theta)$ given by a mixture of uniform distributions as

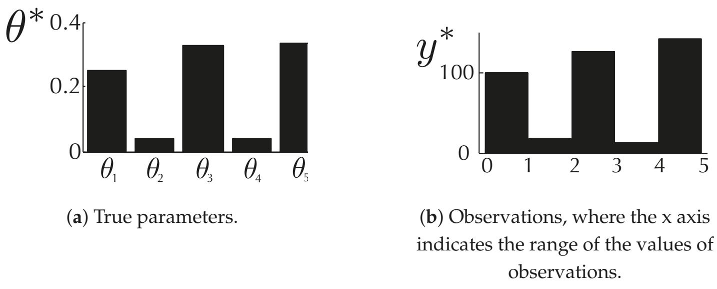
$$\begin{aligned} \pi(\theta) &= \text{Dirichlet}(\theta; \mathbf{1}), \\ \text{P}(y|\theta) &= \sum_{i=1}^{5} \theta_i \text{Uniform}(y; [i-1, i]). \end{aligned} \tag{11}$$

A vector of mixing proportions is our model parameters θ, where the ground truth is $\theta^* = [0.25, 0.04, 0.33, 0.04, 0.34]^\top$ (see Figure 2). The goal is to estimate $\mathbb{E}[\theta|Y^*]$ where Y^* is generated with θ^*.

We first generated 5000 samples for Y^* drawn from (11) with true parameters θ^*. Then, we tested our two ABCDP frameworks with varying ϵ_{abc} and ϵ_{total}. In these experiments, we set $\rho = \widehat{\text{MMD}}$ with a Gaussian kernel. We set the bandwidth of the Gaussian kernel using the median heuristic computed on the simulated data (i.e., we did not use the real data for this, hence there is no privacy violation in this regard).

We drew 5000 pseudo-samples for Y_t at each time. We tested various settings, as shown in Figure 3, where we vary the number of posterior samples, $c = \{10, 100, 1000\}$,

$\epsilon_{abc} = \{0.05, 0.1, 0.2, 0.5\}$ and $\epsilon_{total} = \{0.5, 1.0, 10, \infty\}$. We showed the result of ABCDP for both RESAMPLE options in Figure 3.



(**a**) True parameters. (**b**) Observations, where the x axis indicates the range of the values of observations.

Figure 2. Synthetic data. (**a**): 5-dimensional true parameters; (**b**): observations sampled from the mixture of uniform distributions in (11) with the true parameters.

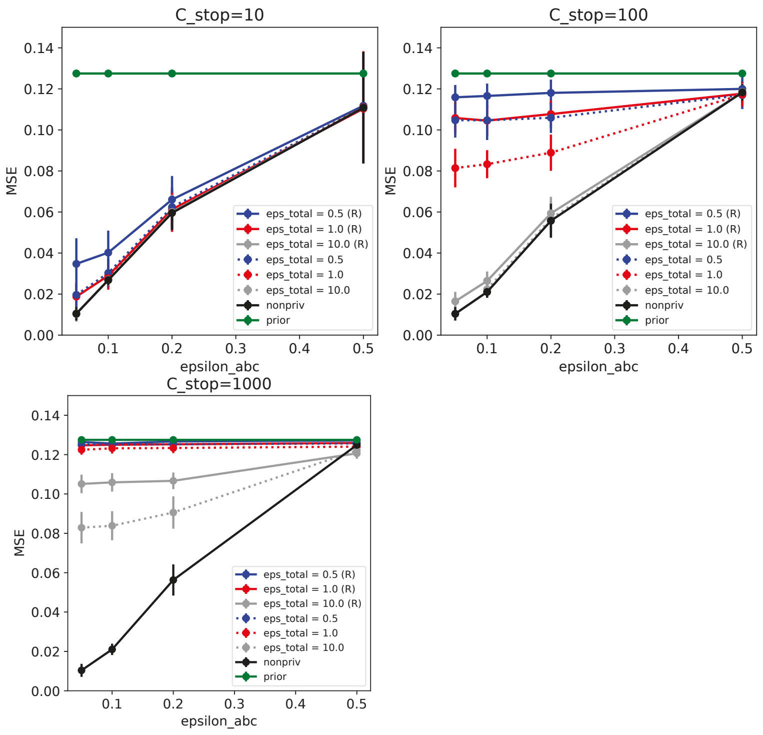


Figure 3. ABCDP on synthetic data. Mean-squared error (between true parameters and posterior mean) as a function of similarity threshold ϵ_{abc} given each privacy level. We ran ABCDP with the following options: *RESAMPLE = True* (denoted by R and solid line); or *RESAMPLE = False* (without R and dotted line) for 60 independent runs. (**Top Left**) When $c_{stop} = 10$ at different values of ϵ_{abc}, ABCDP and non-private ABC (black trace) achieved the highest accuracy (lowest MSE) at the smallest ϵ_{abc} ($\epsilon_{abc} = 0.01$). Notice that ABCDP *RESAMPLE = False* (dotted) outperformed ABCDP *RESAMPLE=True* (solid) for the same privacy tolerance (ϵ_{total}) at small values of ϵ_{abc}. (**Top Right**) MSE for $c_{stop} = 100$ at different values of ϵ_{abc}; (**Bottom Left**) MSE for $c_{stop} = 1000$ at different values of ϵ_{abc}. We can observe when ϵ_{abc} is large, ABCDP (gray) marginally outperforms non-private ABC (black) due to the excessive noise added in ABCDP.

6.2. Coronavirus Outbreak Data

In this experiment, we modelled coronavirus outbreak in the Netherlands using a polynomial model consisting of four parameters a_0, a_1, a_2, a_3, which we aimed to infer, where:

$$y(t) = a_3 + a_2 t + a_1 t^2 + a_0 t^3. \tag{12}$$

The observed (https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-COVID-19-cases-worldwide, accessed on 10 October 2020) data are the number of cases of the coronavirus outbreak from 27 February to 17 March 2020, which amounts to 18 datapoints ($N = 18$). The presented experiment imposes privacy concern as each datapoint is a count of the individuals who are COVID positive at each time. The goal is to identify the approximate posterior distribution $\tilde{\mathrm{P}}(a_0, a_1, a_2, a_3|y^*)$ over these parameters, given a set of observations.

Recalling from Figure 1 that the small size of data worsens the privacy and accuracy trade-off, the inference is restricted to a small number of posterior samples (we chose $c = 5$) since the number of datapoints is extremely limited in this dataset. We used the same prior distributions for the four parameters as $a_i \sim \mathcal{N}(0,1)$ for all $i = 0, 1, 2, 3$. We drew 50,000 samples from the Gaussian prior, and performed our ABCDP algorithm with $\epsilon_{total} = \{13, 22, 44\}$ and $\epsilon_{abc} = 0.1$, as shown in Figure 4.

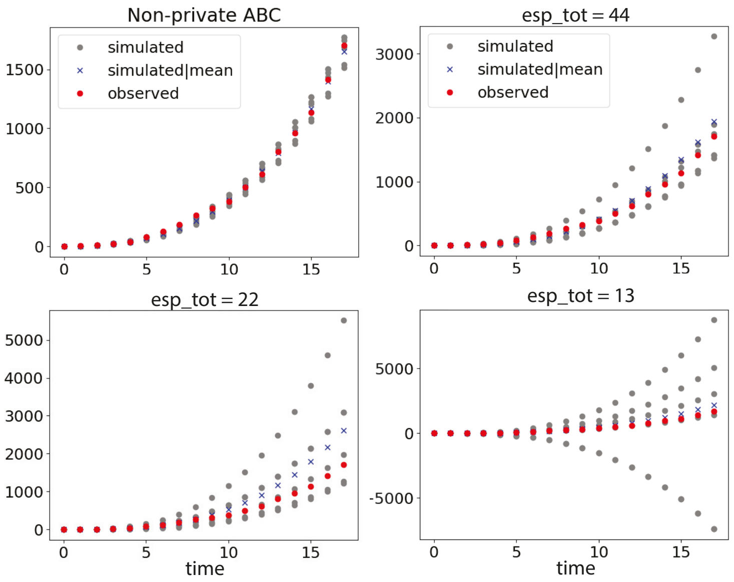


Figure 4. COVID-19 outbreak data ($N = 18$) and simulated data under different privacy guarantees. Red dots show observed data, and gray dots show simulated data drawn from 5 posterior samples accepted in each case. The blue crosses are simulated data given the posterior mean in each case: (**Top left**) simulated data by non-private ABC; (**Top right**) simulated data by ABCDP with $\epsilon_{total} = 44$ are relatively well aligned with regard to the extremely small size of the data. Note that we use a different scale for left and right plots for better visibility. If we use the same y scale in both plots, the simulated and observed points are not distinguishable on the left plot: (**Bottom left**) the simulated data given 5 posterior samples exhibit a large variance when $\epsilon_{total} = 22$; and (**Bottom right**) when $\epsilon_{total} = 13$, the simulated data exhibit an excessively large variance.

6.3. Modeling Tuberculosis (TB) Outbreak Using Stochastic Birth–Death Models

In this experiment, we used the stochastic birth–death models to model Tuberculosis (TB) outbreak. There are four parameters that we aim to infer, which go into the communicable disease outbreak simulator as inputs: burden rate β, transmission rate t_1, reproductive numbers R_1 and R_2. The goal is to identify the approximate posterior distribution $\tilde{p}(R_1, t_1, R_2, \beta|y^*)$ over these parameters given a set of observations. Please refer to Section 3 in [28] for the description of the birth–death process of the model. We used the same prior distributions for the four parameters as in [28]: $\beta \sim \mathcal{N}(200, 30)$, $R_1 \sim \text{Unif}(1.01, 20)$, $R_2 \sim \text{Unif}(0.01, (1 - 0.05R_1)/0.95)$, $t_1 \sim \text{Unif}(0.01, 30)$.

To illustrate the privacy and accuracy trade-off, we first generated two sets of observations y^* ($n = 100$ and $n = 1000$) by some *true* model parameters (shown as black bars in Figure 5). We then tested our ABCDP algorithm with a privacy level $\epsilon = 1$. We used the summary statistic described in Table 1 in [28] and used a weighted L2 distance as ρ as done in [28], together with $\epsilon_{abc} = 150$. Since there is no bounded sensitivity in this case, we impose an artificial boundedness by clipping the distance by C (we set $C = 200$) when the distance goes beyond C.

As an error metric, we computed the mean absolute distance between each posterior mean and the true parameter values. The top row in Figure 5 shows that the mean of the prior (red) is far from the true value (black) that we chose. As we increase the data size from $n = 100$ (middle) to $n = 1000$ (bottom), the distance between true values and estimates reduces, as reflected in the error from 4.71 to 2.20 for RESAMPLE = True; and from 4.51 to 2.10 for RESAMPLE=False.

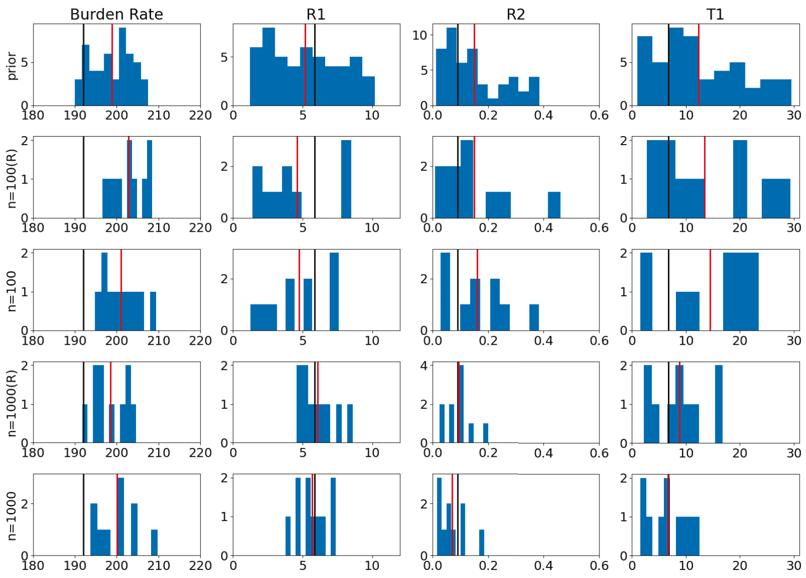


Figure 5. Posterior samples for modeling tuberculosis (TB) outbreak. In all ABCDP methods, we set $\epsilon_{total} = 1$. True values in black. Mean of samples in red: (R) indicates ABCDP with Resampling = True. (**1st row**): Histogram of 50 samples drawn from the prior (we used the same prior as [28]); (**2nd row**): 10 posterior samples from ABCDP with (R) given $n = 100$ observations; (**3rd row**): 10 posterior samples from ABCDP without (R) given $n = 100$ observations; (**4th row**): 10 posterior samples from ABCDP with (R) given $n = 1000$ observations; and (**5th row**): 10 posterior samples from ABCDP without (R) given $n = 1000$ observations. The distance between the black bar (true) and red bar (estimate) reduces as the size of data increases from 100 to 1000. ABCDP with Resampling=False performs better regardless of the data size.

7. Summary and Discussion

We presented the ABCDP algorithm by combining DP with ABC. Our method outputs differentially private binary indicators, yielding differentially private posterior samples.

To analyze the proposed algorithm, we derived the probability of flip from the rejection ABC's indicator to the ABCDP's indicator, as well as the average error bound of the posterior expectation.

We showed experimental results that output a relatively small number of posterior samples. This is due to the fact that the cumulative privacy loss increases linearly with the number of posterior samples (i.e., c) that our algorithm outputs. For a large-sized dataset (i.e., N is large), one can still increase the number of posterior samples while providing a reasonable level of privacy guarantee. However, for a small-sized dataset (i.e., N is small), a more refined privacy composition (e.g., [29]) would be necessary to keep the cumulative privacy loss relatively small, at the expense of providing an *approximate* DP guarantee rather than the pure DP guarantee that ABCDP provides.

When we presented our work to the ABC community, we often received the question of whether we could apply ABCDP to other types of ABC algorithms such as the sequential Monte Carlo algorithm which outputs the significance of each proposal sample, as opposed to its acceptance or rejection as in the rejection ABC algorithm. Directly applying the current form of ABCDP to these algorithms is not possible, while applying the Gaussian mechanism to the significance of each proposal sample can guarantee differential privacy for the output of the sequential Monte Carlo algorithm. However, the cumulative privacy loss will be relatively large, as now it is a function of the number of proposal samples, whether they are taken as good posterior samples or not.

A natural by-product of ABCDP is differentially private synthetic data, as the simulator is a public tool that anybody can run and hence differentially private posterior samples suffice for differentially private synthetic data without any further privacy cost. Applying ABCDP to generate complex datasets is an intriguing future direction.

Author Contributions: M.P. and W.J. contributed to conceptualization and methodology development. M.P. and M.V. contributed to software development. All three authors contributed to writing the paper. All authors have read and agreed to the published version of the manuscript.

Funding: Gips Schüle Foundation and the Institutional Strategy of the University of Tübingen (ZUK63).

Data Availability Statement: A publicly available dataset was analyzed in this study. This data can be found here: https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-covid-19-cases-worldwide.

Conflicts of Interest: The authors declare no conflict of interest.

Note: Our code is available at https://github.com/ParkLabML/ABCDP.

Appendix A. Proof of Theorem 1

Proof of Theorem 1. *Case I: RESAMPLE = True.* We prove the case of $c = 1$ first. The case of $c > 1$ is a c-composition of the case of $c = 1$, where the privacy loss linearly increases with c.

Given any neighboring datasets Y^* and $Y^{*'}$ of size N and any dataset Y, assume that ρ is such that $0 < \sup_{(Y^*,Y^{*'}),Y} \mid \rho(Y^*, Y) - \rho(Y^{*'}, Y) \mid < \Delta_\rho < \infty$ and ρ is bounded by B_ρ.

Let A denote the random variable that represents the outputs Algorithm 1 given $(\{(\boldsymbol{\theta}_t, Y_t)\}_{t=1}^T, Y^*, \rho, \epsilon_{abc}, \epsilon)$ and A' the random variable that represents the outputs given $(\{(\boldsymbol{\theta}_t, Y_t)\}_{t=1}^T, Y^{*'}, \rho, \epsilon_{abc}, \epsilon)$. The output of the algorithm is some realization of these variables, $\tau \in \{1,0\}^k$ where $0 < k \leq T$ and for all $t < k$, $\tau_t = 0$ and $\tau_k = 1$. For the rest of the proof, we will fix the arbitrary values of $\nu_1, ..., \nu_{k-1}$ and take probabilities over the randomness of ν_k and ϵ_{abc}. We define the deterministic quantity ($\nu_1, ..., \nu_{k-1}$ are fixed):

$$g(Y^*) = \min_{t<k}(\rho(Y_t, Y^*) + \nu_t) \tag{A1}$$

that represents the minimum noised value of the distance evaluated on any dataset Y^*.

Let $P[\hat{\epsilon}_{abc} = a]$ be the pdf of $\hat{\epsilon}_{abc}$ evaluated on a and $P[\nu_k = v]$ the pdf of ν_k evaluated on v, and $\mathbb{1}[x]$ the indicator function of event x. We have:

$$\mathrm{P}_{\hat{\epsilon}_{abc},\nu_k}[A = \tau_k] = \mathrm{P}[\hat{\epsilon}_{abc} < g(Y^*)$$

and:

$$\begin{aligned}\rho(Y_k, Y^*) + \nu_k \leq \hat{\epsilon}_{abc}] &= \mathrm{P}[\hat{\epsilon}_{abc} \in [\rho(Y_k, Y^*) + \nu_k, g(Y^*))] \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathrm{P}[\nu_k = v]\mathrm{P}[\hat{\epsilon}_{abc} = a]\mathbb{1}[a \in [\rho(Y_k, Y^*) + \nu_k, g(Y^*))]dvda\end{aligned}$$

Now, we define the following variables:

$$\hat{a} = a + g(Y^*) - g(Y^{*'})$$
$$\hat{v} = vs. + g(Y^*) - g(Y^{*'}) + \rho(Y_k, Y^{*'}) - \rho(Y_k, Y^*)$$

We know that for each $Y^*, Y^{*'}$, ρ is Δ_ρ-sensitive and hence, the quantity $g(Y^*)$ is Δ_ρ-sensitive as well. In this way, we obtain that $\mid \hat{a} - a \mid \leq \Delta_\rho$ and $\mid \hat{v} - vs. \mid \leq 2\Delta_\rho$. Applying these changes of variables, we have:

$$\begin{aligned}&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathrm{P}[\nu_k = \hat{v}]\mathrm{P}[\hat{\epsilon}_{abc} = \hat{a}]\mathbb{1}[a + g(Y^*) - g(Y^{*'}) \in [v + g(Y^*) - g(Y^{*'}) + \\ &\qquad \rho(Y_k, Y^{*'}), g(Y^*))]dvda \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathrm{P}[\nu_k = \hat{v}]\mathrm{P}[\hat{\epsilon}_{abc} = \hat{a}]\mathbb{1}[a \in [v + \rho(Y_k, Y^{*'}), g(Y^{*'}))]dvda \\ &\leq \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \exp\left(\frac{\epsilon}{2}\right)\mathrm{P}[\nu_k = v] \exp\left(\frac{\epsilon}{2}\right)\mathrm{P}[\hat{\epsilon}_{abc} = a]\mathbb{1}[a \in [v + \rho(Y_k, Y^{*'}), g(Y^{*'}))]dvda \\ &\leq \exp(\epsilon) \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathrm{P}[\nu_k = v]\mathrm{P}[\hat{\epsilon}_{abc} = a]\mathbb{1}[a \in [v + \rho(Y_k, Y^{*'})), g(Y^{*'}))]dvda \\ &= \exp(\epsilon)\mathrm{P}[\hat{\epsilon}_{abc} < g(Y^{*'}) \text{ and } \rho(Y_k, Y^{*'}) + \nu_k \leq \hat{\epsilon}_{abc}] = \exp(\epsilon)\mathrm{P}_{\hat{\epsilon}_{abc},\nu_k}[A' = \tau_k]\end{aligned}$$

where the inequality comes from the bounds considered throughout the proof (i.e., $\mid \hat{a} - a \mid \leq \Delta_\rho$ and $\mid \hat{v} - vs. \mid \leq 2\Delta_\rho$) and the form of the cdf for the Laplace distribution.

Case II: RESAMPLE = False. In this case, the proof follows the proof of Algorithm 1 in [26], with an exception that positive events for [26] become negative events for us and vice versa as we find the value below a threshold, where [26] finds the value above a threshold. □

Appendix B. Proof of Lemma 1

Proof of Lemma 1. We will establish Δ_ρ when ρ is MMD. Recall that $(Y^*, Y^{*'})$ is a pair of neighboring datasets, and Y is an arbitrary dataset. Without loss of generality, assume that $Y^* = \{x_1, \ldots, x_N\}$, $Y^{*'} = \{x'_1, \ldots, x'_N\}$ such that $x_i = x'_i$ for all $i = 1, \ldots, N-1$, and $Y = \{y_1, \ldots, y_m\}$. We start with:

$$\begin{aligned}
&\sup_{(Y^*,Y^{*'}),Y} \quad |\rho(Y^*,Y) - \rho(Y^{*'},Y)| \\
&= \sup_{(Y^*,Y^{*'}),Y} \quad |\widehat{\mathrm{MMD}}(Y^*,Y) - \widehat{\mathrm{MMD}}(Y^{*'},Y)| \\
&= \sup_{(Y^*,Y^{*'}),Y} \left| \left\| \frac{1}{N}\sum_{i=1}^{N}\phi(x_i) - \frac{1}{m}\sum_{j=1}^{m}\phi(y_j) \right\|_{\mathcal{H}} - \left\| \frac{1}{N}\sum_{i=1}^{N}\phi(x_i') - \frac{1}{m}\sum_{j=1}^{m}\phi(y_j) \right\|_{\mathcal{H}} \right| \\
&\overset{(a)}{\leq} \sup_{(X,X')} \left\| \frac{1}{N}\sum_{i=1}^{N}\phi(x_i) - \frac{1}{N}\sum_{i=1}^{N}\phi(x_i') \right\|_{\mathcal{H}} \\
&= \sup_{(x_N,x_N')} \left\| \frac{1}{N}\phi(x_N) - \frac{1}{N}\phi(x_N') \right\|_{\mathcal{H}} \\
&= \sup_{(x_N,x_N')} \frac{1}{N}\sqrt{k(x_N,x_N) + k(x_N',x_N') - 2k(x_N,x_N')} \\
&\leq \frac{2}{N}\sqrt{B_k},
\end{aligned}$$

where at (a), we use the reverse triangle inequality. Furthermore:

$$\begin{aligned}
&\sup_{Y^*,Y} \rho(Y^*,Y) \\
&\leq \sup_{Y^*,Y} \sqrt{\left\| \frac{1}{N}\sum_{i=1}^{N}\phi(x_i) - \frac{1}{m}\sum_{i=1}^{m}\phi(y_i) \right\|_{\mathcal{H}}^2} \\
&= \sup_{Y^*,Y} \sqrt{\frac{1}{N^2}\sum_{i,j=1}^{N}k(x_i,x_j) + \frac{1}{m^2}\sum_{i,j=1}^{m}k(y_i,y_j) - \frac{2}{mn}\sum_{i=1}^{N}\sum_{j=1}^{m}k(x_i,y_j)} \\
&= \sqrt{B_k + B_k + 2B_k} = 2\sqrt{B_k}.
\end{aligned}$$

□

Appendix C. Proof of Lemma 2

Proof of Lemma 2. The PDF is computed from the convolution of two PDFs:

$$f_{m_t - v_t}(z) = \int_{-\infty}^{\infty} f_{m_t}(x) f_{v_t}(x - z)\,dx, \tag{A2}$$

where $f_{m_t}(x) = \frac{1}{2b}\exp(-\frac{|x|}{b})$ and $f_{v_t}(y) = \frac{1}{4b}\exp(-\frac{|y|}{2b})$:

$$f_{m_t - v_t}(z) = \frac{1}{8b^2}\int_{-\infty}^{\infty} \exp\left(-\frac{|x|}{b} - \frac{|x-z|}{2b}\right) dx \tag{A3}$$

For case $z \geq 0$:

$$f_{m_t-\nu_t}(z) = \frac{1}{8b^2}\int_{-\infty}^{0}\exp\left(\frac{x}{b}+\frac{x-z}{2b}\right)dx + \frac{1}{8b^2}\int_{0}^{z}\exp\left(-\frac{x}{b}+\frac{x-z}{2b}\right)dx$$
$$+\frac{1}{8b^2}\int_{z}^{\infty}\exp\left(-\frac{x}{b}-\frac{x-z}{2b}\right)dx, \tag{A4}$$
$$= \frac{1}{8b^2}\int_{-\infty}^{0}\exp\left(\frac{3x-z}{2b}\right)dx + \frac{1}{8b^2}\int_{0}^{z}\exp\left(\frac{-x-z}{2b}\right)dx$$
$$+\frac{1}{8b^2}\int_{z}^{\infty}\exp\left(\frac{-3x+z}{2b}\right)dx \tag{A5}$$
$$= \frac{\exp\left(\frac{-z}{2b}\right)}{8b^2}\int_{-\infty}^{0}\exp\left(\frac{3x}{2b}\right)dx + \frac{\exp\left(\frac{-z}{2b}\right)}{8b^2}\int_{0}^{z}\exp\left(\frac{-x}{2b}\right)dx$$
$$+\frac{\exp\left(\frac{z}{2b}\right)}{8b^2}\int_{z}^{\infty}\exp\left(\frac{-3x}{2b}\right)dx \tag{A6}$$
$$= \frac{\exp\left(\frac{-z}{2b}\right)}{8b^2}\frac{2b}{3} - \frac{\exp\left(\frac{-z}{2b}\right)}{8b^2}2b\left(\exp\left(\frac{-z}{2b}\right)-1\right) + \frac{\exp\left(\frac{z}{2b}\right)}{8b^2}\frac{2b}{3}\exp\left(\frac{-3z}{2b}\right), \tag{A7}$$
$$= \frac{1}{12b}\left[\exp\left(\frac{-z}{2b}\right)+3\exp\left(\frac{-z}{2b}\right)\left(1-\exp\left(\frac{-z}{2b}\right)\right)+\exp\left(\frac{-z}{b}\right)\right], \tag{A8}$$
$$= \frac{1}{12b}\left[4\exp\left(\frac{-z}{2b}\right)-2\exp\left(\frac{-z}{b}\right)\right], \tag{A9}$$
$$= \frac{1}{6b}\left[2\exp\left(\frac{-z}{2b}\right)-\exp\left(\frac{-z}{b}\right)\right] \tag{A10}$$

For case $z < 0$:

$$f_{m_t-\nu_t}(z) = \frac{1}{8b^2}\int_{-\infty}^{z}\exp\left(\frac{x}{b}+\frac{x-z}{2b}\right)dx + \frac{1}{8b^2}\int_{z}^{0}\exp\left(\frac{x}{b}-\frac{x-z}{2b}\right)dx$$
$$+\frac{1}{8b^2}\int_{0}^{\infty}\exp\left(-\frac{x}{b}-\frac{x-z}{2b}\right)dx, \tag{A11}$$
$$= \frac{1}{8b^2}\int_{-\infty}^{z}\exp\left(\frac{3x-z}{2b}\right)dx + \frac{1}{8b^2}\int_{z}^{0}\exp\left(\frac{x+z}{2b}\right)dx$$
$$+\frac{1}{8b^2}\int_{0}^{\infty}\exp\left(\frac{-3x+z}{2b}\right)dx \tag{A12}$$
$$= \frac{\exp\left(\frac{-z}{2b}\right)}{8b^2}\int_{-\infty}^{z}\exp\left(\frac{3x}{2b}\right)dx + \frac{\exp\left(\frac{z}{2b}\right)}{8b^2}\int_{z}^{0}\exp\left(\frac{x}{2b}\right)dx$$
$$+\frac{\exp\left(\frac{z}{2b}\right)}{8b^2}\int_{0}^{\infty}\exp\left(\frac{-3x}{2b}\right)dx \tag{A13}$$
$$= \frac{\exp\left(\frac{-z}{2b}\right)}{8b^2}\frac{2b}{3}\exp\left(\frac{3z}{2b}\right) + \frac{\exp\left(\frac{z}{2b}\right)}{8b^2}2b\left(1-\exp\left(\frac{z}{2b}\right)\right) + \frac{\exp\left(\frac{z}{2b}\right)}{8b^2}\frac{2b}{3}, \tag{A14}$$
$$= \frac{1}{12b}\left[\exp\left(\frac{z}{b}\right)-3\exp\left(\frac{z}{2b}\right)\left(\exp\left(\frac{z}{2b}\right)-1\right)+\exp\left(\frac{z}{2b}\right)\right], \tag{A15}$$
$$= \frac{1}{12b}\left[-2\exp\left(\frac{z}{b}\right)+4\exp\left(\frac{z}{2b}\right)\right], \tag{A16}$$
$$= \frac{1}{6b}\left[2\exp\left(\frac{z}{2b}\right)-\exp\left(\frac{z}{b}\right)\right]. \tag{A17}$$

With the obtained PDF $f_Z(z) = \frac{1}{6b}\left[2\exp\left(-\frac{|z|}{2b}\right)-\exp\left(-\frac{|z|}{b}\right)\right]$. for $Z := m_t - \nu_t$, it is straightforward to compute $G_b(a) := \int_a^\infty f_Z(z)\,\mathrm{d}z = \frac{1}{6}\left[4\exp\left(-\frac{a}{2b}\right)-\exp\left(-\frac{a}{b}\right)\right]$ for $a \geq 0$. In other words, $G_b(a) = 1 - F_Z(a)$ for $a \geq 0$ where F_Z denotes the CDF of Z.

To show that the CDF of Z is $F_Z(a) = H[a] + (1-2H[a])G_b(|a|)$ where $H[a]$ is the Heaviside step function, we note that the density $f_Z(z)$ is an even function, i.e., $f_Z(z) = f_Z(-z)$ for any z. It follows that if $a < 0$, $1 - F_z(a) = 1 - G_b(-a)$. This means that:

$$1 - F_Z(a) = \begin{cases} G_b(a) & \text{if } a \geq 0, \\ 1 - G_b(-a) & \text{if } a < 0, \end{cases}$$

or equivalently:

$$F_Z(a) = \begin{cases} 1 - G_b(a) & \text{if } a \geq 0, \\ G_b(-a) & \text{if } a < 0. \end{cases}$$

More concisely:

$$\begin{aligned} F_Z(a) &= (1 - G_b(|a|))\mathbb{I}[a \geq 0] + G_b(|a|)\mathbb{I}[a < 0] \\ &= \mathbb{I}[a \geq 0] + (\mathbb{I}[a < 0] - \mathbb{I}[a \geq 0])G_b(|a|) \\ &\overset{(a)}{=} H[a] + (1 - 2H[a])G_b(|a|), \end{aligned}$$

where at (a), we use $(\mathbb{I}[a < 0] - \mathbb{I}[a \geq 0]) = (1 - 2H[a])$. □

Appendix D. Proof of Proposition 1

Proof of Proposition 1. Using this pdf above, we can compute the following probabilities:

$$\begin{aligned} &\mathrm{P}[\tilde{\tau}_t = 1 | \tau_t = 0] && \text{(A18)} \\ &= \mathrm{P}[0 \leq \rho_t - \epsilon_{abc} \leq Z], && \text{(A19)} \\ &= \int_{\rho_t - \epsilon_{abc}}^{\infty} f(z)dz, \quad \text{where } \rho_t - \epsilon_{abc} \geq 0 && \text{(A20)} \\ &= \int_{\rho_t - \epsilon_{abc}}^{\infty} \frac{1}{6b}\left[2\exp\left(-\frac{|z|}{2b}\right) - \exp\left(-\frac{|z|}{b}\right)\right]dz, \text{ by definition of } f(z) && \text{(A21)} \\ &= \int_{\rho_t - \epsilon_{abc}}^{\infty} \frac{1}{6b}\left[2\exp\left(-\frac{z}{2b}\right) - \exp\left(-\frac{z}{b}\right)\right]dz, \quad \text{because } \rho_t - \epsilon_{abc} \geq 0 && \text{(A22)} \\ &= \frac{1}{6b}\left[4b\exp\left(-\frac{\rho_t - \epsilon_{abc}}{2b}\right) - b\exp\left(-\frac{\rho_t - \epsilon_{abc}}{b}\right)\right], && \text{(A23)} \\ &= \frac{1}{6}\left[4\exp\left(-\frac{\rho_t - \epsilon_{abc}}{2b}\right) - \exp\left(-\frac{\rho_t - \epsilon_{abc}}{b}\right)\right], \text{where } \rho_t - \epsilon_{abc} \geq 0, && \text{(A24)} \end{aligned}$$

and:

$$\begin{aligned} &\mathrm{P}[\tilde{\tau}_t = 0 | \tau_t = 1] \\ &= \mathrm{P}[Z \leq \rho_t - \epsilon_{abc} \leq 0], && \text{(A25)} \\ &= \int_{-\infty}^{\rho_t - \epsilon_{abc}} f(z)dz, \text{ where } \rho_t - \epsilon_{abc} \leq 0, && \text{(A26)} \\ &= \int_{-\infty}^{\rho_t - \epsilon_{abc}} \frac{1}{6b}\left[2\exp\left(\frac{z}{2b}\right) - \exp\left(\frac{z}{b}\right)\right]dz, && \text{(A27)} \\ &= \frac{1}{6b}\left[4b\exp\left(\frac{\rho_t - \epsilon_{abc}}{2b}\right) - b\exp\left(\frac{\rho_t - \epsilon_{abc}}{b}\right)\right], && \text{(A28)} \\ &= \frac{1}{6}\left[4\exp\left(\frac{\rho_t - \epsilon_{abc}}{2b}\right) - \exp\left(\frac{\rho_t - \epsilon_{abc}}{b}\right)\right], \text{ where } \rho_t - \epsilon_{abc} \leq 0. && \text{(A29)} \end{aligned}$$

So:

$$\begin{aligned} \mathrm{P}[\tilde{\tau}_t \neq \tau_t | \tau_t] &= \begin{cases} \mathrm{P}[\tilde{\tau}_t = 1 | \tau_t = 0], & \text{if } \rho_t \geq \epsilon_{abc} \\ \mathrm{P}[\tilde{\tau}_t = 0 | \tau_t = 1], & \text{otherwise} \end{cases} && \text{(A30)} \\ &= \begin{cases} \frac{1}{6}\left[4\exp\left(-\frac{\rho_t - \epsilon_{abc}}{2b}\right) - \exp\left(-\frac{\rho_t - \epsilon_{abc}}{b}\right)\right], \text{if } \rho_t \geq \epsilon_{abc} \\ \frac{1}{6}\left[4\exp\left(\frac{\rho_t - \epsilon_{abc}}{2b}\right) - \exp\left(\frac{\rho_t - \epsilon_{abc}}{b}\right)\right], \text{otherwise.} \end{cases} \end{aligned}$$

The two cases can be combined with the use of an absolute value to give the result. □

Appendix E. Proof of Theorem 2

Proof of Theorem 2. Let $H(x)$ be the Heaviside step function. Recall from our algorithm that each accepted sample $(\boldsymbol{\theta}, Y)$ is associated with two independent noise realizations: $m_t \sim \text{Lap}(b)$ (i.e., $\hat{\epsilon}_{abc} = \epsilon_{abc} + m_t$) and $\nu_t \sim \text{Lap}(2b)$ (added to $\rho(Y^*, Y_t)$). With this notation, we have $\tilde{\tau}_t = H[\epsilon_{abc} - \rho(Y_t, Y^*) + m_t - \nu_t]$ for $t = 1, \dots, T$. Similarly, $\tau_t := H[\epsilon_{abc} - \rho(Y_t, Y^*)]$. For brevity, we define $\rho_t := \rho(Y_t, Y^*)$. It follows that $\tilde{\tau}_t \sim \text{Bernoulli}(p_t)$ where $p_t := \text{P}(m_t - \nu_t > \rho_t - \epsilon_{abc}) = \text{P}(\tilde{\tau} = 1)$.

Proof of the first claim: we start by establishing an upper bound for:

$$\begin{aligned}
&\left\| \frac{1}{c} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t \right\|_2 \\
&= \left\| \frac{1}{c} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t + \frac{1}{c'} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t \right\|_2 \\
&\leq \left| \frac{1}{c} - \frac{1}{c'} \right| \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t \right\| + \frac{1}{c'} \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t \right\| \\
&= \frac{1}{c'} |c' - c| \frac{1}{c} \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t \right\| + \frac{1}{c'} \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)(\tilde{\tau}_t - \tau_t) \right\| \\
&\leq \frac{1}{c'} \left| \sum_{t=1}^{T} \tau_t - \sum_{t=1}^{T} \tilde{\tau}_t \right| \frac{1}{c} \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t \right\| + \frac{1}{c'} \sum_{t=1}^{T} \|f(\boldsymbol{\theta}_t)\|_2 |\tilde{\tau}_t - \tau_t| \\
&\leq \frac{1}{c'} \sum_{t=1}^{T} |\tilde{\tau}_t - \tau_t| \frac{1}{c} \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t \right\| + \frac{K_T}{c'} \sum_{t=1}^{T} |\tilde{\tau}_t - \tau_t|,
\end{aligned} \tag{A31}$$

where $K_T := \max_{t=1,\dots,T} \|f(\boldsymbol{\theta}_t)\|_2$. Consider $\frac{1}{c} \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t \right\|$. We can show that it is bounded by K_T by

$$\frac{1}{c} \left\| \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t \right\| \leq \frac{1}{c} \sum_{t=1}^{T} \|f(\boldsymbol{\theta}_t)\|_2 \tilde{\tau}_t \leq \frac{K_T}{c} \sum_{t=1}^{T} \tilde{\tau}_t = K_T.$$

Combining this bound with (A31), we have:

$$\left\| \frac{1}{c} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'} \sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t \right\|_2 \leq \frac{2K_T}{c'} \sum_{t=1}^{T} |\tilde{\tau}_t - \tau_t| \tag{A32}$$

We will need to characterize the distribution of $|\tilde{\tau}_t - \tau_t|$. Let $Z_t := m_t - \nu_t$. By Lemma 2, we have:

$$\begin{aligned}
p_t = \text{P}(\tilde{\tau}_t = 1) = \text{P}(Z_t > \rho_t - \epsilon_{abc}) &= 1 - F_Z(\rho_t - \epsilon_{abc}) \\
&= 1 - H[\rho_t - \epsilon_{abc}] + (2H[\rho_t - \epsilon_{abc}] - 1)G_b(|\rho_t - \epsilon_{abc}|) \\
&= \tau_t + (1 - 2\tau_t)G_b(|\rho_t - \epsilon_{abc}|),
\end{aligned}$$

where the decreasing function $G_b(x) \in (0, \frac{1}{2}]$ for any $x \geq 0$ is defined in Lemma 2. We observe that $|\tilde{\tau}_t - \tau_t| \sim \text{Bernoulli}(q_t)$ where $q_t := \text{P}(\tilde{\tau}_t \neq \tau_t) = (1 - p_t)\tau_t + p_t(1 - \tau_t)$. We can rewrite q_t as

$$\begin{aligned}
q_t &= \tau_t + p_t(1 - 2\tau_t) \\
&= \tau_t + [\tau_t + (1 - 2\tau_t)G_b(|\rho_t - \epsilon_{abc}|)](1 - 2\tau_t) \\
&= G_b(|\rho_t - \epsilon_{abc}|).
\end{aligned}$$

To prove the first claim, we take the expectation on both sides of (A32):

$$\mathbb{E}_{\tilde{\tau}_1,\dots,\tilde{\tau}_T}\left\|\frac{1}{c}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t\right\|_2 \le \frac{2K_T}{c'}\mathbb{E}_{\tilde{\tau}_t}\left[\sum_{t=1}^{T}|\tilde{\tau}_t - \tau_t|\right]$$
$$= \frac{2K_T}{c'}\mu_T,$$

where $\mu_T = \mathbb{E}_{\tilde{\tau}_t}\left[\sum_{t=1}^{T}|\tilde{\tau}_t - \tau_t|\right] = \sum_{t=1}^{T} G_b(|\rho_t - \epsilon_{abc}|)$ and we use the fact that $\mathbb{E}_{\tilde{\tau}_t}|\tilde{\tau}_t - \tau_t| = q_t$. Note that these are T independent, marginal expectations, i.e., they do not depend on the condition that noise is added to the ABC threshold.

Proof of the second claim: observe that $G_b(|\rho_t - \epsilon_{abc}|) \to 0$ as $b \to 0$. The claim follows by noting that $b = O(1/N)$.

Proof of the third claim: based on (A32), characterizing the tail bound of $\left\|\frac{1}{c}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t\right\|_2$ amounts to establishing a tail bound on $S_T := \sum_{t=1}^{T}|\tilde{\tau}_t - \tau_t|$. By Markov's inequality:

$$\begin{aligned}
\mathrm{P}(S_T \le s) &\ge 1 - \mathbb{E}[S_T]/s \\
&= 1 - \frac{1}{s}\sum_{t=1}^{T} G_b(|\rho_t - \epsilon_{abc}|) \\
&= 1 - \frac{1}{s}\sum_{t=1}^{T}\frac{1}{6}\left[4\exp\left(-\frac{|\rho_t - \epsilon_{abc}|}{2b}\right) - \exp\left(-\frac{|\rho_t - \epsilon_{abc}|}{b}\right)\right] \\
&\ge 1 - \frac{2}{3s}\sum_{t=1}^{T}\exp\left(-\frac{|\rho_t - \epsilon_{abc}|}{2b}\right).
\end{aligned}$$

Applying this bound to (A32) gives:

$$\mathrm{P}\left(\frac{2K_T}{c'}S_T \le \frac{2K_T}{c'}s\right) = \mathrm{P}(S_T \le s).$$

With a reparametrization $a := \frac{2K_T}{c'}s$ so that $s = \frac{ac'}{2K_T}$, we have:

$$\mathrm{P}\left(\frac{2K_T}{c'}S_T \le a\right) \ge 1 - \frac{4K_T}{3ac'}\sum_{t=1}^{T}\exp\left(-\frac{|\rho_t - \epsilon_{abc}|}{2b}\right),$$

Since $\left\|\frac{1}{c}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t\right\|_2 \le \frac{2K_T}{c'}S_T$, we have:

$$\mathrm{P}\left(\left\|\frac{1}{c}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tilde{\tau}_t - \frac{1}{c'}\sum_{t=1}^{T} f(\boldsymbol{\theta}_t)\tau_t\right\|_2 \le a\right) \ge \mathrm{P}\left(\frac{2K_T}{c'}S_T \le a\right).$$

which gives the result in the third claim. □

References

1. Tavaré, S.; Balding, D.J.; Griffiths, R.C.; Donnelly, P. Inferring coalescence times from DNA sequence data. *Genetics* **1997**, *145*, 505–518. [CrossRef] [PubMed]
2. Ratmann, O.; Jørgensen, O.; Hinkley, T.; Stumpf, M.; Richardson, S.; Wiuf, C. Using Likelihood-Free Inference to Compare Evolutionary Dynamics of the Protein Networks of H. pylori and P. falciparum. *PLoS Comput. Biol.* **2007**, *3*, e230. [CrossRef] [PubMed]
3. Bazin, E.; Dawson, K.J.; Beaumont, M.A. Likelihood-Free Inference of Population Structure and Local Adaptation in a Bayesian Hierarchical Model. *Genetics* **2010**, *185*, 587–602. [CrossRef] [PubMed]
4. Schafer, C.M.; Freeman, P.E. Likelihood-Free Inference in Cosmology: Potential for the Estimation of Luminosity Functions. In *Statistical Challenges in Modern Astronomy V*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 3–19.

5. Pritchard, J.; Seielstad, M.; Perez-Lezaun, A.; Feldman, M. Population growth of human Y chromosomes: A study of Y chromosome microsatellites. *Mol. Biol. Evol.* **1999**, *16*, 1791–1798. [CrossRef]
6. Fearnhead, P.; Prangle, D. Constructing summary statistics for approximate Bayesian computation: Semi-automatic approximate Bayesian computation. *J. R. Stat. Soc. Ser.* **2012**, *74*, 419–474. [CrossRef]
7. Joyce, P.; Marjoram, P. Approximately Sufficient Statistics and Bayesian Computation. *Stat. Appl. Genet. Molec. Biol.* **2008**, *7*, 1544–6115. [CrossRef] [PubMed]
8. Robert, C.P.; Cornuet, J.; Marin, J.; Pillai, N.S. Lack of confidence in approximate Bayesian computation model choice. *Proc. Natl. Acad. Sci. USA* **2011**, *108*, 15112–15117. [CrossRef] [PubMed]
9. Nunes, M.; Balding, D. On Optimal Selection of Summary Statistics for Approximate Bayesian Computation. *Stat. Appl. Genet. Molec. Biol.* **2010**, *9*. [CrossRef] [PubMed]
10. Aeschbacher, S.; Beaumont, M.A.; Futschik, A. A Novel Approach for Choosing Summary Statistics in Approximate Bayesian Computation. *Genetics* **2012**, *192*, 1027–1047. [CrossRef] [PubMed]
11. Drovandi, C.; Pettitt, A.; Lee, A. Bayesian Indirect Inference Using a Parametric Auxiliary Model. *Statist. Sci.* **2015**, *30*, 72–95. [CrossRef]
12. Homer, N.; Szelinger, S.; Redman, M.; Duggan, D.; Tembe, W.; Muehling, J.; Pearson, J.V.; Stephan, D.A.; Nelson, S.F.; Craig, D.W. Resolving Individuals Contributing Trace Amounts of DNA to Highly Complex Mixtures Using High-Density SNP Genotyping Microarrays. *PLoS Genet.* **2008**, *4*, e1000167. [CrossRef] [PubMed]
13. Johnson, A.; Shmatikov, V. Privacy-preserving Data Exploration in Genome-wide Association Studies. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; pp. 1079–1087.
14. Tanaka, M.M.; Francis, A.R.; Luciani, F.; Sisson, S.A. Using approximate Bayesian computation to estimate tuberculosis transmission parameters from genotype data. *Genetics* **2006**, *173*, 1511–1520. [CrossRef] [PubMed]
15. Dwork, C.; McSherry, F.; Nissim, K.; Smith, A. Calibrating noise to sensitivity in private data analysis. In Proceedings of the TCC, New York, NY, USA, 4–7 March 2006; Springer: Berlin/Heidelberg, Germany, 2006; Volume 3876, pp. 265–284.
16. Chaudhuri, K.; Monteleoni, C.; Sarwate, A.D. Differentially Private Empirical Risk Minimization. *J. Mach. Learn. Res.* **2011**, *12*, 1069–1109. [PubMed]
17. Dwork, C.; Roth, A. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* **2014**, *9*, 211–407. [CrossRef]
18. Park, M.; Jitkrittum, W.; Sejdinovic, D. K2-ABC: Approximate Bayesian Computation with Infinite Dimensional Summary Statistics via Kernel Embeddings. In Proceedings of the AISTATS, Cadiz, Spain, 9–11 May 2016.
19. Nakagome, S.; Fukumizu, K.; Mano, S. Kernel approximate Bayesian computation in population genetic inferences. *Stat. Appl. Genet. Mol. Biol.* **2013**, *12*, 667–678. [CrossRef] [PubMed]
20. Gleim, A.; Pigorsch, C. *Approximate Bayesian Computation with Indirect Summary Statistics*; University of Bonn: Bonn, Germany, 2013.
21. Gretton, A.; Borgwardt, K.; Rasch, M.; Schölkopf, B.; Smola, A. A Kernel Two-Sample Test. *J. Mach. Learn. Res.* **2012**, *13*, 723–773.
22. Smola, A.; Gretton, A.; Song, L. ; Schölkopf, D. A Hilbert space embedding for distributions. In *Algorithmic Learning Theory, Proceedings of the 18th International Conference, Sendai, Japan, 1–4 October* 2007; Springer: Berlin/Heidelberg, Germany, 2007.
23. Sriperumbudur, B.; Fukumizu, K.; Lanckriet, G. Universality, characteristic kernels and RKHS embedding of measures. *J. Mach. Learn. Res.* **2011**, *12*, 2389–2410.
24. Dwork, C.; Kenthapadi, K.; McSherry, F.; Mironov, I.; Naor, M. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *Advances in Cryptology—EUROCRYPT 2006, Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, 28 May–1 June 2006*; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4004, pp. 486–503.
25. Mironov, I. Rényi Differential Privacy. In Proceedings of the 30th IEEE Computer Security Foundations Symposium (CSF), Santa Barbara, CA, USA, 21–25 August 2017; pp. 263–275.
26. Lyu, M.; Su, D.; Li, N. Understanding the Sparse Vector Technique for Differential Privacy. *Proc. VLDB Endow.* **2017**, *10*, 637–648. [CrossRef]
27. Gong, R. Exact Inference with Approximate Computation for Differentially Private Data via Perturbations. *arXiv* **2019**, arXiv:stat.CO/1909.12237.
28. Lintusaari, J.; Blomstedt, P.; Rose, B.; Sivula, T.; Gutmann, M.; Kaski, S.; Corander, J. Resolving outbreak dynamics using approximate Bayesian computation for stochastic birth?death models [version 2; peer review: 2 approved]. *Wellcome Open Res.* **2019**, *4*. [CrossRef]
29. Zhu, Y.; Wang, Y.X. Improving Sparse Vector Technique with Renyi Differential Privacy. In Proceedings of the 2020 Conference on Neural Information Processing Systems, Virtual, 6–12 December 2020; Volume 33.

entropy

Article

Variational Message Passing and Local Constraint Manipulation in Factor Graphs

İsmail Şenöz [1,*], Thijs van de Laar [1], Dmitry Bagaev [1] and Bert de Vries [1,2]

1 Department of Electrical Engineering, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands; T.W.v.d.Laar@tue.nl (T.v.d.L.); d.v.bagaev@tue.nl (D.B.); bert.de.vries@tue.nl (B.d.V.)
2 GN Hearing, JF Kennedylaan 2, 5612 AB Eindhoven, The Netherlands
* Correspondence: i.senoz@tue.nl

Abstract: Accurate evaluation of Bayesian model evidence for a given data set is a fundamental problem in model development. Since evidence evaluations are usually intractable, in practice variational free energy (VFE) minimization provides an attractive alternative, as the VFE is an upper bound on negative model log-evidence (NLE). In order to improve tractability of the VFE, it is common to manipulate the constraints in the search space for the posterior distribution of the latent variables. Unfortunately, constraint manipulation may also lead to a less accurate estimate of the NLE. Thus, constraint manipulation implies an engineering trade-off between tractability and accuracy of model evidence estimation. In this paper, we develop a unifying account of constraint manipulation for variational inference in models that can be represented by a (Forney-style) factor graph, for which we identify the Bethe Free Energy as an approximation to the VFE. We derive well-known message passing algorithms from first principles, as the result of minimizing the constrained Bethe Free Energy (BFE). The proposed method supports evaluation of the BFE in factor graphs for model scoring and development of new message passing-based inference algorithms that potentially improve evidence estimation accuracy.

Keywords: Bayesian inference; Bethe free energy; factor graphs; message passing; variational free energy; variational inference; variational message passing

Citation: Şenöz, İ.; van de Laar, T.; Bagaev, D.; de Vries, B. Variational Message Passing and Local Constraint Manipulation in Factor Graphs. *Entropy* **2021**, *23*, 807. https://doi.org/10.3390/e23070807

Academic Editor: Pierre Alquier

Received: 19 May 2021
Accepted: 22 June 2021
Published: 24 June 2021

1. Introduction

Building models from data is at the core of both science and engineering applications. The search for good models requires a performance measure that scores how well a particular model m captures the hidden patterns in a data set D. In a Bayesian framework, that measure is the *Bayesian evidence* $p(D|m)$, i.e., the probability that model m would generate D if we were to draw data from m. The art of modeling is then the iterative process of proposing new model specifications, evaluating the evidence for each model and retaining the model with the most evidence [1].

Unfortunately, Bayesian evidence is intractable for most interesting models. A popular solution to evidence evaluation is provided by *variational* inference, which describes the process of Bayesian evidence evaluation as a (free energy) minimization process, since the variational free energy (VFE) is a tractable upper bound on Bayesian (negative log-)evidence [2]. In practice, the model development process then consists of proposing various candidate models, minimizing VFE for each model and selecting the model with the lowest minimized VFE.

The difference between VFE and negative log-evidence (NLE) is equal to the Kullback–Leibler divergence (KLD) [3] from the (perfect) Bayesian posterior distribution to the variational distribution for the latent variables in the model. The KLD can be interpreted as the cost of conducting variational rather than Bayesian inference. Perfect (Bayesian) inference would lead to zero inference costs (KLD $= 0$), and the KLD increases as the variational posterior diverges further from the Bayesian posterior. As a result, model development in a variational inference context is a balancing act, where we search for models that have both large amounts of evidence for the data and small inference costs (small KLD). In other words, in a variational inference context, the researcher has two knobs to tune models. The first knob alters the model specification, which affects model evidence. The second knob relates to constraining the search space for the variational posterior, which may affect the inference costs.

In this paper, we are concerned with developing algorithms for tuning the second knob. How do we constrain the range of variational posteriors so as to make variational inferences both tractable and accurate (resulting in low KLD)? We present our framework in the context of a (Forney-style) factor graph representation of the model [4,5]. In that context, variational inference can be understood as an automatable and efficient message passing-based inference procedure [6–8].

Traditional constraints include mean-field [6] and Bethe approximations [9,10]. However, more recently it has become clear how alternative local constraints, such as posterior factorization [11], expectation and chance constraints [12,13], and local Laplace approximation [14], may impact both tractability and inference accuracy, and thereby potentially lead to lower VFE. The main contribution of the current work lies in unifying the various ideas on local posterior constraints into a principled method for deriving variational message passing-based inference algorithms. The proposed method derives existing message passing algorithms, but also supports the development of new message passing variants.

Section 2 reviews Forney-style Factor Graphs (FFGs) and variational inference by minimizing the Bethe Free Energy (BFE). This review is continued in Section 3, where we discuss BFE optimization from a Lagrangian optimization viewpoint. In Appendix A, we include an example to illustrate that the Bayes rule can be derived from Lagrangian optimization with data constraints. Our main contribution lies in Section 4, which provides a rigorous treatment of the effects of imposing local constraints on the BFE and the resulting message update rules. We build upon several previous works that describe how manipulation of (local) constraints and variational objectives can be employed to improve variational approximations in the context of message passing. For example, ref. [12] shows how inference algorithms can be unified in terms of hybrid message passing by Lagrangian constraint manipulation. We extend this view by bringing form (Section 4.2) and factorization constraints (Section 4.1) into a constrained optimization framework. In [15], a high-level recipe for generating message passing algorithms from divergence measures is described. We apply their general recipe in the current work, where we adhere to the view on local stationary points for region-based approximations on general graphs [16]. In Appendix B, we also show that locally stationary solutions are also the global stationary solutions. In Section 5, we develop an algorithm for VFE evaluation in an FFG. In previous work, ref. [17] describes a factor softening approach to evaluate the VFE for models with deterministic factors. We extend this work in Section 5, and show how to avoid factor softening for both free energy evaluation and inference of posteriors. We show an example of how to compute VFE for a deterministic node in Appendix C. A more detailed comparison to related work is given in Section 7.

In the literature, proofs and descriptions of message passing-based inference algorithms are scattered across multiple papers and varying graphical representations, including Bayesian networks [6,18], Markov random fields [16], bi-partite (Tanner) factor graphs [12,17,19] and Forney-style factor graphs (FFGs) [5,11]. In Appendix D, we provide first-principle proofs for a large collection of familiar message passing algorithms in the context of Forney-style factor graphs, which is the preferred framework in the information and communication theory communities [4,20].

2. Factor Graphs and the Bethe Free Energy

2.1. Terminated Forney-Style Factor Graphs

A Forney-style factor graph (FFG) is an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. We denote the neighboring edges of a node $a \in \mathcal{V}$ by $\mathcal{E}(a)$. Vice versa, for an edge $i \in \mathcal{E}$, the notation $\mathcal{V}(i)$ collects all neighboring nodes. As a notational convention, we index nodes by a, b, c and edges by i, j, k, unless stated otherwise. We will mainly use a and i as summation indices and use the other indices to refer to a node or edge of interest.

In this paper, we will frequently refer to the notion of a subgraph. We define an edge-induced subgraph by $\mathcal{G}(i) = (\mathcal{V}(i), i)$, and a node-induced subgraph by $\mathcal{G}(a) = (a, \mathcal{E}(a))$. Furthermore, we denote a local subgraph by $\mathcal{G}(a, i) = (\mathcal{V}(i), \mathcal{E}(a))$, which collects all local nodes and edges around i and a, respectively.

An FFG can be used to represent a factorized function,

$$f(\boldsymbol{s}) = \prod_{a \in \mathcal{V}} f_a(\boldsymbol{s}_a)\,, \tag{1}$$

where $\boldsymbol{s}_a$ collects the argument variables of factor f_a. We assumed that all the factors are positive. In an FFG, a node $a \in \mathcal{V}$ corresponds to a factor f_a, and the neighboring edges $\mathcal{E}(a)$ correspond to the variables $\boldsymbol{s}_a$ that are the arguments of f_a.

As an example model, the following factorization (2), the corresponding FFG of which is shown in Figure 1.

$$f(s_1, \ldots, s_5) = f_a(s_1)\, f_b(s_1, s_2, s_3)\, f_c(s_2)\, f_d(s_3, s_4, s_5)\, f_e(s_5)\,. \tag{2}$$

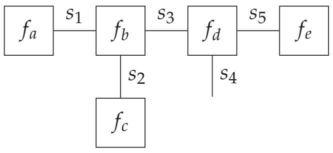


Figure 1. Example Forney-style factor graph for the model of (2).

The FFG of Figure 1 consists of five nodes $\mathcal{V} = \{a, \ldots, e\}$, as annotated by their corresponding factor functions, and five edges $\mathcal{E} = \{(a, b), \ldots, (d, e)\}$ as annotated by their corresponding variables. An edge that connects to only one node (e.g., the edge for s_4) is called a half-edge. In this example, the neighborhood $\mathcal{E}(b) = \{(a, b), (b, c), (b, d)\}$ and $\mathcal{V}((b, c)) = \{b, c\}$.

In the FFG representation, a node can be connected to an arbitrary number of edges, while an edge can only be connected to at most two nodes. Therefore, FFGs often contain "equality nodes" that constrain connected edges to carry identical beliefs, with the implication that these beliefs can be made available to more than two factors. An equality node has the factor function

$$f_a(s_i, s_j, s_k) = \delta(s_j - s_i)\, \delta(s_j - s_k)\,, \tag{3}$$

for which the node-induced subgraph $\mathcal{G}(a)$ is drawn in Figure 2.

If every edge in the FFG has exactly two connected nodes (including equality nodes), then we designate the graph as a terminated FFG (TFFG). Since multiplication of a function $f(\boldsymbol{s})$ by 1 does not alter the function, any FFG can be terminated by connecting any half-edge i to a node a that represents the unity factor $f_a(s_i) = 1$.

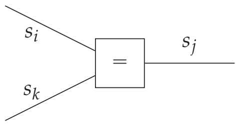


Figure 2. Visualization of the node-induced subgraph for an equality node. If the node function f_a is known, a symbol representing the node function is often substituted within the node ("=" in this case).

In Section 4.2 we discuss form constraints on posterior distributions. If such a constraint takes on a Dirac-delta functional form, then we visualize the constraint on the FFG by a small circle in the middle of the edge. For example, the small shaded circle in Figure 11 indicates that the variable has been observed. In Section 4.3.2 we consider form constraints in the context of optimization, in which case the circle annotation will be left open (see, e.g., Figure 14).

2.2. Variational Free Energy

Given a model $f(\boldsymbol{s})$ and a (normalized) probability distribution $q(\boldsymbol{s})$, we can define a Variational Free Energy (VFE) functional as

$$F[q,f] \triangleq \int q(\boldsymbol{s}) \log \frac{q(\boldsymbol{s})}{f(\boldsymbol{s})} \,\mathrm{d}\boldsymbol{s}\,. \tag{4}$$

Variational inference is concerned with finding solutions to the minimization problem

$$q^*(\boldsymbol{s}) = \arg\min_{q\in\mathcal{Q}} F[q,f]\,, \tag{5}$$

where $\mathcal{Q}$ imposes some constraints on q.

If q is unconstrained, then the optimal solution is obtained for $q^*(\boldsymbol{s}) = p(\boldsymbol{s})$, with $p(\boldsymbol{s}) = \frac{1}{Z} f(\boldsymbol{s})$ being the exact posterior, and $Z = \int f(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}$ a normalizing constant that is commonly referred to as the evidence. The minimum value of the free energy then follows as the negative log-evidence (NLE),

$$F[q^*,f] = -\log Z\,,$$

which is also known as the surprisal. The NLE can be interpreted as a measure of model performance, where low NLE is preferred.

As an unconstrained search space for q grows exponentially with the number of variables, the optimization of (5) quickly becomes intractable beyond the most basic models. Therefore, constraints and approximations to the variational free energy (4) are often utilized. As a result, the *constrained* variational free energy with $q^* \in \mathcal{Q}$ bounds the NLE by

$$F[q^*,f] = -\log Z + \int q^*(\boldsymbol{s}) \log \frac{q^*(\boldsymbol{s})}{p(\boldsymbol{s})} \,\mathrm{d}\boldsymbol{s}\,, \tag{6}$$

where the latter term expresses the divergence from the (intractable) exact solution to the optimal variational belief.

In practice, the functional form of $q(\boldsymbol{s}) = q(\boldsymbol{s};\boldsymbol{\theta})$ is often parameterized, such that gradients of F can be derived w.r.t. the parameters $\boldsymbol{\theta}$. This effectively converts the variational optimization of $F[q,f]$ to a parametric optimization of $F(\boldsymbol{\theta})$ as a function of $\boldsymbol{\theta}$. This problem can then be solved by a (stochastic) gradient descent procedure [21,22].

In the context of variational calculus, while form constraints may lead to interesting properties (see Section 4.2), they are generally not required. Interestingly, in a variational optimization context, the functional form of q is often not an *assumption*, but rather a *result* of optimization (see Section 4.3.1). An example of variational inference is provided in Appendix A.

2.3. Bethe Free Energy

The Bethe approximation enjoys a unique place in the landscape of $\mathcal{Q}$, because the Bethe free energy (BFE) defines the fundamental objective of the celebrated belief propagation (BP) algorithm [17,23]. The origin of the Bethe approximation is rooted in tree-like approximations to subgraphs (possibly containing cycles) by enforcing local consistency conditions on the beliefs associated with edges and nodes [24].

Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ for a factorized function $f(\boldsymbol{s}) = \prod_{a\in\mathcal{V}} f_a(\boldsymbol{s}_a)$ (1), the Bethe free energy (BFE) is defined as [25]:

$$F[q, f] \triangleq \sum_{a\in\mathcal{V}} \underbrace{\int q_a(\boldsymbol{s}_a) \log \frac{q_a(\boldsymbol{s}_a)}{f_a(\boldsymbol{s}_a)} \,\mathrm{d}\boldsymbol{s}_a}_{F[q_a, f_a]} + \sum_{i\in\mathcal{E}} \underbrace{\int q_i(s_i) \log \frac{1}{q_i(s_i)} \,\mathrm{d}s_i}_{H[q_i]} \tag{7}$$

such that the factorized beliefs

$$q(\boldsymbol{s}) = \prod_{a\in\mathcal{V}} q_a(\boldsymbol{s}_a) \prod_{i\in\mathcal{E}} q_i(s_i)^{-1} \tag{8}$$

satisfy the following constraints:

$$\int q_a(\boldsymbol{s}_a) \,\mathrm{d}\boldsymbol{s}_a = 1\,, \quad \text{for all } a \in \mathcal{V} \tag{9a}$$

$$\int q_a(\boldsymbol{s}_a) \,\mathrm{d}\boldsymbol{s}_{a\setminus i} = q_i(s_i)\,, \quad \text{for all } a \in \mathcal{V} \text{ and all } i \in \mathcal{E}(a)\,. \tag{9b}$$

Together, the normalization constraint (9a) and marginalization constraint (9b) imply that the edge marginals are also normalized:

$$\int q_i(s_i) \,\mathrm{d}s_i = 1\,, \quad \text{for all } i \in \mathcal{E}\,. \tag{10}$$

The Bethe free energy (7) includes a local free energy term $F[q_a, f_a]$ for each node $a \in \mathcal{V}$, and an entropy term $H[q_i]$ for each edge $i \in \mathcal{E}$. Note that the local free energy also depends on the node function f_a, as specified in the factorization of f (1), whereas the entropy only depends on the local belief q_i.

The Bethe factorization (8) and constraints are summarized by the local polytope [26]

$$\mathcal{L}(\mathcal{G}) = \{q_a \text{ for all } a \in \mathcal{V} \text{ s.t. (9a), and } q_i \text{ for all } i \in \mathcal{E}(a) \text{ s.t. (9b)}\}\,, \tag{11}$$

which defines the constrained search space for the factorized variational distribution (8).

2.4. Problem Statement

In this paper, the problem is to find the beliefs in the local polytope that minimize the Bethe free energy

$$q^*(\boldsymbol{s}) = \arg \min_{q\in\mathcal{L}(\mathcal{G})} F[q, f]\,, \tag{12}$$

where q is defined by (8), and where $q \in \mathcal{L}(\mathcal{G})$ offers a shorthand notation for optimizing over the individual beliefs in the local polytope. In the following sections, we will follow the Lagrangian optimization approach to derive various message passing-based inference algorithms.

2.5. Sketch of Solution Approach

The problem statement of Section 2.4 defines a global minimization of the beliefs in the Bethe factorization. Instead of solving the global optimization problem directly, we employ the factorization of the variational posterior and local polytope to subdivide the global problem statement in multiple *interdependent* local objectives.

From the BFE objective (12) and local polytope of (11), we can construct the Lagrangian

$$L[q,f] = \sum_{a\in\mathcal{V}} F[q_a, f_a] + \sum_{a\in\mathcal{V}} \psi_a \left[\int q_a(\boldsymbol{s}_a)\, \mathrm{d}\boldsymbol{s}_a - 1 \right] + \sum_{a\in\mathcal{V}} \sum_{i\in\mathcal{E}(a)} \int \lambda_{ia}(s_i) \left[q_i(s_i) - \int q_a(\boldsymbol{s}_a)\, \mathrm{d}\boldsymbol{s}_{a\setminus i} \right] \mathrm{d}s_i$$
$$+ \sum_{i\in\mathcal{E}} H[q_i] + \sum_{i\in\mathcal{E}} \psi_i \left[\int q_i(s_i)\, \mathrm{d}s_i - 1 \right], \tag{13}$$

where the Lagrange multipliers ψ_a, ψ_i and λ_{ia} enforce the normalization and marginalization constraints of (9). It can be seen that this Lagrangian contains local beliefs q_a and q_i, which are coupled through the λ_{ia} Lagrange multipliers. The Lagrange multipliers λ_{ia} are doubly indexed, because there is a multiplier associated with each marginalization constraint. The Lagrangian method then converts a constrained optimization problem of $F[q,f]$ to an unconstrained optimization problem of $L[q,f]$. The total variation of the Lagrangian (13) can then be approached from the perspective of variations of the individual (coupled) local beliefs.

More specifically, given a locally connected pair $b \in \mathcal{V}, j \in \mathcal{E}(b)$, we can rewrite the optimization of (12) in terms of the local beliefs q_b, q_j, and the constraints in the local polytope

$$\mathcal{L}(\mathcal{G}(b,j)) = \left\{ q_b \text{ s.t. (9a), and } q_j \text{ s.t. (9b)} \right\}, \tag{14}$$

that pertains to these beliefs. The problem then becomes finding local stationary solutions

$$\{q_b^*, q_j^*\} = \arg \min_{\mathcal{L}(\mathcal{G}(b,j))} F[q,f]\,. \tag{15}$$

Using (13), the optimization of (15) can then be written in the Lagrangian form

$$q_b^* = \arg\min_{q_b} L_b[q_b, f_b]\,, \tag{16a}$$
$$q_j^* = \arg\min_{q_j} L_j[q_j]\,, \tag{16b}$$

where the Lagrangians L_b and L_j include the local polytope of (14) to rewrite (13) as an explicit functional of beliefs q_b and q_j (see, e.g., Lemmas 1 and 2). The combined stationary solutions to the local objectives then also comprise a stationary solution to the global objective (Appendix B).

The current paper shows how to identify stationary solutions to local objectives of the form (15), with the use of variational calculus, under varying constraints as imposed by the local polytope (14). Interestingly, the resulting fixed-point equations can be interpreted as message passing updates on the underlying TFFG representation of the model. In the following Sections 3 and 4, we derive the local stationary solutions under a selection of constraints and show how these relate to known message passing update rules (Table 1). It then becomes possible to derive novel message updates and algorithms by simply altering the local polytope.

Table 1. Relation between local constraints and derived message updates. The rows refer to different constraints that relate to factor–variable combinations, factors, and variables, respectively. Note that each message passing algorithm combines a set of constraints. Abbreviations: Sum-Product (SP), Structured Variational Message Passing (SVMP), Mean-Field Variational Message Passing (MFVMP), Data Constraint (DC), Laplace Propagation (LP), Mean-Field Variational Laplace (MFVLP), Expectation Maximization (EM), and Expectation Propagation (EP).

Local Constraint	SP	SVMP	MFVMP	DC	LP	MFVLP	EM	EP
Normalization	✓	✓	✓	✓	✓	✓	✓	✓
Marginalization	✓	✓	✓	✓	✓	✓	✓	✓
Moment-Matching								✓
Structured Mean-Field		✓					✓	
Naive Mean-Field			✓			✓		
Laplace Approximation					✓	✓		
Dirac-delta				✓			✓	
Estimation							✓	

3. Bethe Lagrangian Optimization by Message Passing

3.1. Stationary Points of the Bethe Lagrangian

We wish to minimize the Bethe free energy under variations of the variational density. As the Bethe free energy factorizes over factors and variables (7), we first consider variations on separate node- and edge-induced subgraphs.

Lemma 1. *Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the node-induced subgraph $\mathcal{G}(b)$ (Figure 3). The stationary points of the Lagrangian (16a) as a functional of q_b,*

$$L_b[q_b, f_b] = F[q_b, f_b] + \psi_b\left[\int q_b(\boldsymbol{s}_b)\, \mathrm{d}\boldsymbol{s}_b - 1\right] + \sum_{i\in\mathcal{E}(b)} \int \lambda_{ib}(s_i)\left[q_i(s_i) - \int q_b(\boldsymbol{s}_b)\, \mathrm{d}\boldsymbol{s}_{b\setminus i}\right] \mathrm{d}s_i + C_b\,, \tag{17}$$

where C_b collects all terms that are independent of q_b, which are of the form

$$q_b(\boldsymbol{s}_b) = \frac{f_b(\boldsymbol{s}_b) \prod\limits_{i\in\mathcal{E}(b)} \mu_{ib}(s_i)}{\int f_b(\boldsymbol{s}_b) \prod\limits_{i\in\mathcal{E}(b)} \mu_{ib}(s_i) \mathrm{d}\boldsymbol{s}_b}\,. \tag{18}$$

Proof. See Appendix D.1. □

The $\mu_{ib}(s_i)$ are any set of positive functions that makes (18) satisfy (9b), and will be identified in Theorem 1.

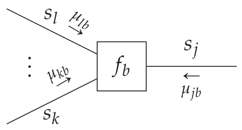


Figure 3. The subgraph around node b with indicated messages. Ellipses indicate an arbitrary (possibly zero) amount of edges.

Lemma 2. *Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider an edge-induced subgraph $\mathcal{G}(j)$ (Figure 4). The stationary points of the Lagrangian (16b) as a functional of q_j,*

$$L_j[q_j] = H[q_j] + \psi_j\left[\int q_j(s_j)\,\mathrm{d}s_j - 1\right] + \sum_{a\in\mathcal{V}(j)} \int \lambda_{ja}(s_j)\left[q_j(s_j) - \int q_a(\boldsymbol{s}_a)\,\mathrm{d}\boldsymbol{s}_{a\setminus j}\right]\mathrm{d}s_j + C_j\,, \tag{19}$$

where C_j collects all terms that are independent of q_j, are of the form

$$q_j(s_j) = \frac{\mu_{jb}(s_j)\mu_{jc}(s_j)}{\int \mu_{jb}(s_j)\mu_{jc}(s_j)\mathrm{d}s_j}\,. \tag{20}$$

Proof. See Appendix D.2. □

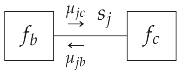


Figure 4. An edge-induced subgraph $\mathcal{G}(j)$ with indicated messages.

3.2. Minimizing the Bethe Free Energy by Belief Propagation

We now combine Lemmas 1 and 2 to derive the sum-product message update.

Theorem 1 (Sum-Product Message Update)**.** *Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the induced subgraph $\mathcal{G}(b, j)$ (Figure 5). Given the local polytope $\mathcal{L}(\mathcal{G}(b, j))$ of (14), then the local stationary solutions to (15) are given by*

$$q_b^*(\boldsymbol{s}_b) = \frac{f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)} \mu_{ib}^*(s_i)}{\int f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)} \mu_{ib}^*(s_i)\mathrm{d}\boldsymbol{s}_b} \tag{21a}$$

$$q_j^*(s_j) = \frac{\mu_{jb}^*(s_j)\mu_{jc}^*(s_j)}{\int \mu_{jb}^*(s_j)\mu_{jc}^*(s_j)\mathrm{d}s_j}\,, \tag{21b}$$

with messages $\mu_{jc}^*(s_j)$ corresponding to the fixed points of

$$\mu_{jc}^{(k+1)}(s_j) = \int f_b(\boldsymbol{s}_b) \prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \mu_{ib}^{(k)}(s_i)\mathrm{d}\boldsymbol{s}_{b\setminus j}\,, \tag{22}$$

with k representing an iteration index.

Proof. See Appendix D.3. □

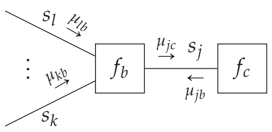


Figure 5. Visualization of a subgraph with indicated sum-product messages.

The sum-product algorithm has proven to be useful in many engineering applications and disciplines. For example, it is widely used for decoding in communication systems [4,20,27]. Furthermore, for a linear Gaussian state space model, Kalman filtering and smoothing can be expressed in terms of sum-product message passing for state inference on a factor graph [28,29]. This equivalence has inspired applications ranging from localization [30] to estimation [31].

The sum-product algorithm with updates (22) obtains the exact Bayesian posterior when the underlying graph is a tree [24,25,32]. Application of the sum-product algorithm to cyclic graphs is not guaranteed to converge and might lead to oscillations in the BFE over iterations. Theorems 3.1 and 3.2 in [33] show that the BFE of a graph with a single cycle is convex, which implies that the sum-product algorithm will converge in this case. Moreover, ref. [19] shows that it is possible to obtain a double-loop message passing algorithm if the graph has a cycle such that the stable fixed points will correspond to local minima of the BFE.

Example 1. *A Linear Dynamical System Considering a Linear Gaussian state space model specified by the following factors:*

$$g_0(x_0) = \mathcal{N}(x_0|m_{x_0}, V_{x_0}) \tag{23a}$$
$$g_t(x_{t-1}, z_t, A_t) = \delta(z_t - A_t x_{t-1}) \tag{23b}$$
$$h_t(x'_t, z_t, Q_t) = \mathcal{N}(x'_t|z_t, Q_t^{-1}) \tag{23c}$$
$$n_t(x_t, x'_t, x''_t) = \delta(x_t - x'_t)\delta(x_t - x''_t) \tag{23d}$$
$$m_t(o_t, x''_t, B_t) = \delta(o_t - B_t x''_t) \tag{23e}$$
$$r_t(y_t, o_t, R_t) = \mathcal{N}(y_t|o_t, R_t^{-1})\,. \tag{23f}$$

The FFG corresponding to the one time segment of the state space model is given in Figure 6. We assumed that we know the following matrices that are used to generate the data:

$$\hat{A}_t = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix},\ \hat{Q}_t^{-1} = \begin{bmatrix} 3 & 0.1 \\ 0.1 & 2 \end{bmatrix},\ \hat{B}_t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \hat{R}_t^{-1} = \begin{bmatrix} 10 & 2 \\ 2 & 20 \end{bmatrix} \tag{24}$$

with $\theta = \pi/8$. Given a collection of observations $\hat{\boldsymbol{y}} = \{\hat{y}_1, \dots, \hat{y}_T\}$, we constrain the latent states $\boldsymbol{x} = \{x_0, \dots, x_T\}$ by local marginalization and normalization constraints (for brevity we omit writing the normalization constraints explicitly) in accordance with Theorem 1, i.e.,

$$\int q(x_{t-1}, z_t, A_t)\mathrm{d}x_{t-1}\mathrm{d}z_t = q(A_t),\ \int q(x_{t-1}, z_t, A_t)\mathrm{d}A_t = q(z_t|x_{t-1})q(x_{t-1}) \tag{25a}$$
$$\int q(x'_t, z_t, Q_t)\mathrm{d}x'_t\mathrm{d}z_t = q(Q_t),\ \int q(x'_t, z_t, Q_t)\mathrm{d}z_t\mathrm{d}Q_t = q(x'_t),\ \int q(x'_t, z_t, Q_t)\mathrm{d}x'_t\mathrm{d}Q_t = q(z_t) \tag{25b}$$
$$q(x_t, x'_t, x''_t) = q(x_t)\delta(x_t - x'_t)\delta(x_t - x''_t) \tag{25c}$$
$$\int q(o_t, x''_t, B_t)\mathrm{d}o_t, \mathrm{d}x''_t = q(B_t),\ \int q(o_t, x''_t, B_t)\mathrm{d}B_t = q(o_t|x''_t)q(x''_t) \tag{25d}$$
$$\int q(o_t, y_t, R_t)\mathrm{d}o_t\mathrm{d}y_t = q(R_t),\ \int q(o_t, y_t, R_t)\mathrm{d}R_t\mathrm{d}o_t = q(y_t),\ \int q(o_t, y_t, R_t)\mathrm{d}R_t\mathrm{d}y_t = q(o_t) \tag{25e}$$

Moreover, we use data constraints in accordance with Theorem 3 (explained in Section 4.2.1) for the observations, state transition matrices and precision matrices, i.e.,

$$q(y_t) = \delta(y_t - \hat{y}_t),\ q(A_t) = \delta(A_t - \hat{A}_t),\ q(B_t) = \delta(B_t - \hat{B}_t),\ q(Q_t) = \delta(Q_t - \hat{Q}_t),\ q(R_t) = \delta(R_t - \hat{R}_t)\,.$$

Computation of sum-product messages by (22) is analytically tractable and detailed algebraic manipulation can be found in [31]. If the backwards messages are not passed, then the resulting sum-product message passing algorithm is equivalent to Kalman filtering and if both forward and backward messages are propagated, then the Rauch–Tung–Striebel smoother is obtained [34] (Ch. 8).

We generated $T = 100$ observations $\hat{\boldsymbol{y}}$ using the matrices specified in (24) and the initial condition $\hat{x}_0 = [5, -5]^\top$. Due to (23a), we have $\mu_{x_0 g_1} = \mathcal{N}(m_{x_0}, V_{x_0})$. We chose $V_{x_0} = 100 \cdot \mathrm{I}$ and $m_{x_0} = \hat{x}_0$. Under these constraints, the results of sum-product message passing and Bethe free *energy evaluation is given in Figure 6. As the underlying graph is a tree, sum-product message passing results are exact and the evaluated BFE corresponds to negative log-evidence. In the follow-up Example 2, we will modify the constraints and give a comparative free energy plot for the examples in Figures 10 and 16.*

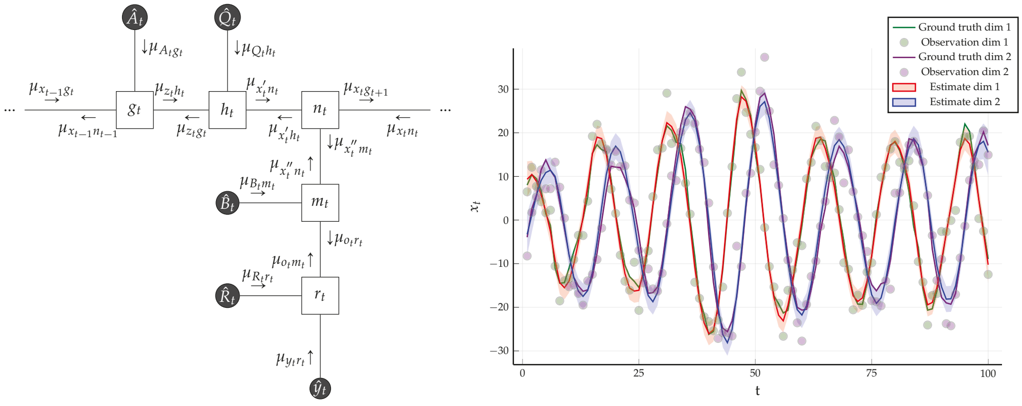


Figure 6. (**Left**) One time segment of the FFG corresponding to the linear Gaussian state space model specified in Example 1, with the sum-product messages computed according to (22). The three small dots at both sides of the graph indicate identical continuation of the graph over time. (**Right**) The small dots indicate the noisy observations that are synthetically generated by the linear state space model of (23) using parameter matrices as specified in (24). The posterior distribution for the hidden states are inferred by sum-product message passing and are drawn with shaded regions, indicating plus and minus the variance. The Bethe free energy evaluates to $F[q, f] = 580.698$.

4. Message Passing Variations through Constraint Manipulation

For generic node functions with arbitrary connectivity, there is no guarantee that the sum-product updates can be solved analytically. When analytic solutions are not possible, there are two ways to proceed. One way is to try to solve the sum-product update equations numerically, e.g., by Monte Carlo methods. Alternatively, we can add additional constraints to the BFE that leads to simpler update equations at the cost of inference accuracy. In the remainder of the paper, we explore a variety of constraints that have proven to yield useful inference solutions.

4.1. Factorization Constraints

Additional factorizations of the variational density $q_a(\boldsymbol{s}_a)$ are often assumed to ease computation. In particular, we assumed a *structured mean-field factorization* such that

$$q_b(\boldsymbol{s}_b) \triangleq \prod_{n \in l(b)} q_b^n(\boldsymbol{s}_b^n)\,, \tag{26}$$

where n indicates a local cluster as a set of edges. To define a local cluster rigorously, let us first denote by $\mathcal{P}(a)$ the power set of an edge set $\mathcal{E}(a)$, where the power set is the set of all subsets of $\mathcal{E}(a)$. Then, a mean-field factorization $l(a) \subseteq \mathcal{P}(a)$ can be chosen such that all elements in $\mathcal{E}(a)$ are included in $l(a)$ exactly once. Therefore, $l(a)$ is defined as a set of one or multiple sets of edges. For example, if $\mathcal{E}(a) = \{i, j, k\}$, then $l(a) = \{\{i\}, \{j, k\}\}$ is allowed, as is $l(a) = \{\{i, j, k\}\}$ itself, but $l(a) = \{\{i, j\}, \{j, k\}\}$ is not allowed, since the element j occurs twice. More formally, in (26), the intersection of the super- and subscript collects the required variables, see Figure 7 for an example. The special case of a fully factorized $l(b)$ for all edges $i \in \mathcal{E}(b)$ is known as the *naive mean-field factorization* [11,24].

We will analyze the effect of a structured mean-field factorization (26) on the Bethe free energy (7) for a specific factor node $b \in \mathcal{V}$. Substituting (26) in the local free energy for factor b yields

$$F[q_b, f_b] = F[\{q_b^n\}, f_b] = \sum_{n \in l(b)} \int q_b^n(\boldsymbol{s}_b^n) \log q_b^n(\boldsymbol{s}_b^n)\, \mathrm{d}\boldsymbol{s}_b^n - \int \Big\{ \prod_{n \in l(b)} q_b^n(\boldsymbol{s}_b^n) \Big\} \log f_b(\boldsymbol{s}_b)\, \mathrm{d}\boldsymbol{s}_b\,. \tag{27}$$

We are then interested in

$$q_b^{m,*} = \arg\min_{q_b^m} L_b^m[q_b^m, f_b]\,, \tag{28}$$

where the Lagrangian L_b^m (Lemma 3) enforces the normalization and marginalization constraints

$$\int q_b^m(\boldsymbol{s}_b^m)\, \mathrm{d}\boldsymbol{s}_b^m = 1\,, \tag{29a}$$

$$\int q_b^m(\boldsymbol{s}_b^m)\, \mathrm{d}\boldsymbol{s}_{b\setminus i}^m = q_i(s_i), \text{ for all } i \in m\,, m \in l(b)\,. \tag{29b}$$

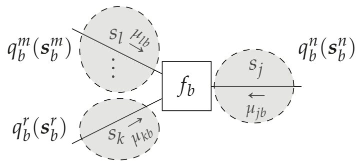

$q_b^m(\boldsymbol{s}_b^m)$
s_l
μ_{lb}
f_b
s_j
μ_{jb}
$q_b^n(\boldsymbol{s}_b^n)$
$q_b^r(\boldsymbol{s}_b^r)$
s_k
μ_{kb}

Figure 7. A node-induced subgraph $\mathcal{G}(b)$ with shaded sections that enclose the edges of an exemplary structured mean-field factorization $l(b) = \{m, n, r\}$. Note that, in this example, the cluster n only encompasses the single edge j, such that $q_b^n(s_b^n) = q_j(s_j)$. In general, the assignment and number of edges in a cluster can be arbitrary.

Lemma 3. *Given a terminated FFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider a node-induced subgraph $\mathcal{G}(b)$ with a structured mean-field factorization $l(b)$ (e.g., Figure 7). Then, local stationary solutions to the Lagrangian*

$$\begin{aligned} L_b^m[q_b^m] = &\int q_b^m(\boldsymbol{s}_b^m) \log q_b^m(\boldsymbol{s}_b^m)\, \mathrm{d}\boldsymbol{s}_b^m - \int \Big\{ \prod_{n \in l(b)} q_b^n(\boldsymbol{s}_b^n) \Big\} \log f_b(\boldsymbol{s}_b)\, \mathrm{d}\boldsymbol{s}_b + \\ &\psi_b^m \Big[\int q_b^m(\boldsymbol{s}_b^m)\, \mathrm{d}\boldsymbol{s}_b^m - 1 \Big] + \sum_{i \in m} \int \lambda_{ib}(s_i) \Big[q_i(s_i) - \int q_b^m(\boldsymbol{s}_b^m)\, \mathrm{d}\boldsymbol{s}_{m\setminus i} \Big] \mathrm{d}s_i + C_b^m\,, \end{aligned} \tag{30}$$

where C_b^m collects all terms independent of q_b^m, which are of the form

$$q_b^m(\boldsymbol{s}_b^m) = \frac{\tilde{f}_b^m(\boldsymbol{s}_b^m) \prod_{i \in m} \mu_{ib}(s_i)}{\int \tilde{f}_b^m(\boldsymbol{s}_b^m) \prod_{i \in m} \mu_{ib}(s_i) \mathrm{d}\boldsymbol{s}_b^m}\,, \tag{31}$$

where

$$\tilde{f}_b^m(\boldsymbol{s}_b^m) = \exp\Bigg(\int \Big\{ \prod_{\substack{n \in l(b) \\ n \neq m}} q_b^n(\boldsymbol{s}_b^n) \Big\} \log f_b(\boldsymbol{s}_b)\, \mathrm{d}\boldsymbol{s}_b^{\setminus m} \Bigg)\,. \tag{32}$$

Proof. See Appendix D.4. □

4.1.1. Structured Variational Message Passing

We now combine Lemmas 2 and 3 to derive the structured variational message passing algorithm.

Theorem 2. *Structured variational message passing: Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the induced subgraph $\mathcal{G}(b, j)$ with a structured mean-field factorization $l(b) \subseteq \mathcal{P}(b)$, with local clusters $n \in l(b)$. Let $m \in l(b)$ be the cluster where $j \in m$ (see, e.g., Figure 8). Given the local polytope*

$$\mathcal{L}(\mathcal{G}(b,j)) = \left\{ q_b^n \text{ for all } n \in l(b) \text{ s.t. (29a), and } q_j \text{ s.t. (29b)} \right\}, \tag{33}$$

then local stationary solutions to

$$\{q_b^{m,*}, q_j^*\} = \arg \min_{\mathcal{L}(\mathcal{G}(b,j))} F[q,f], \tag{34}$$

are given by

$$q_b^{m,*}(\boldsymbol{s}_b^m) = \frac{\tilde{f}_b^{m,*}(\boldsymbol{s}_b^m) \prod_{i \in m} \mu_{ib}^*(s_i)}{\int \tilde{f}_b^{m,*}(\boldsymbol{s}_b^m) \prod\limits_{i \in m} \mu_{ib}^*(s_i) \mathrm{d}\boldsymbol{s}_b^m} \tag{35a}$$

$$q_j^*(s_j) = \frac{\mu_{jb}^*(s_j)\mu_{jc}^*(s_j)}{\int \mu_{jb}^*(s_j)\mu_{jc}^*(s_j)\mathrm{d}s_j}, \tag{35b}$$

with messages $\mu_{jc}^*(s_j)$ corresponding to the fixed points of

$$\mu_{jc}^{(k+1)}(s_j) = \int \tilde{f}_b^{m,(k)}(\boldsymbol{s}_b^m) \prod_{\substack{i \in m \\ i \neq j}} \mu_{ib}^{(k)}(s_i) \mathrm{d}\boldsymbol{s}_{b \setminus j}^m, \tag{36}$$

with iteration index k, and where

$$\tilde{f}_b^{m,(k)} = \exp\left(\int \left\{ \prod_{\substack{n \in l(b) \\ n \neq m}} q_b^{n,(k)}(\boldsymbol{s}_b^n) \right\} \log f_b(\boldsymbol{s}_b) \, \mathrm{d}\boldsymbol{s}_b^{\setminus m} \right). \tag{37}$$

Proof. See Appendix D.5. □

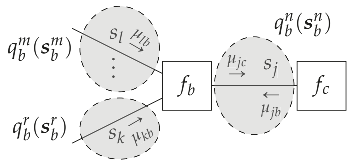

$q_b^m(\boldsymbol{s}_b^m)$
s_l
μ_{lb}
$q_b^r(\boldsymbol{s}_b^r)$
s_k
μ_{kb}
f_b
μ_{jc}
s_j
μ_{jb}
$q_b^n(\boldsymbol{s}_b^n)$
f_c

Figure 8. An example subgraph corresponding to $\mathcal{G}(b, j)$. Dashed ellipses enclose the edges of an exemplary exact cover $l(b) = \{m, n, r\}$. In general, the assignment and number of edges in a cluster can be arbitrary.

The structured mean-field factorization applies the marginalization constraint only to the local cluster beliefs, as opposed to the joint node belief. As a result, computation for the local cluster beliefs might become tractable [24] (Ch.5). The practical appeal of Variational Message Passing (VMP) based inference becomes evident when the underlying model is composed of conjugate factor pairs from the exponential family. When the underlying factors are conjugate exponential family distributions, the message passing updates (36) amounts to adding natural parameters [35] of the underlying exponential family distributions. Structured variational message passing is popular in acoustic signal modelling, e.g., [36], as it allows one to be able to keep track of correlations over time. In [37], a stochastic variant of structured variational inference is utilized for Latent Dirichlet Allocation. Structured approximations are also used to improve inference in auto-encoders. In [38], inference involving non-parametric Beta-Bernoulli process priors is improved by developing a structured approximation to variational auto-encoders. When the data being modelled are time series, structured approximations reflect the transition structure over time. In [39], an efficient structured black-box variational inference algorithm for fitting Gaussian variational models to latent time series is proposed.

Example 2. *Consider the linear Gaussian state space model of Example 1. Let us assume that the precision matrix for latent-state transitions Q_t is not known and can not be constrained by data. Then, we can augment state space model by including a prior for Q_t and try to infer a posterior over Q_t from the observations. Since Q_t is the precision of a normal factor, we chose a conjugate Wishart prior and assumed that Q_t is time-invariant by adding the following factors*

$$w_0(Q_0, V, \nu) = \mathcal{W}(Q_0|V, \nu) \tag{38a}$$

$$w_t(Q_{t-1}, Q_t, Q_{t+1}) = \delta(Q_{t-1} - Q_t)\delta(Q_t - Q_{t+1}), \textit{ for every } t = 1, \dots, T\,. \tag{38b}$$

It is certainly possible to assume a time-varying structure for Q_t; however, our purpose is to illustrate a change in constraints rather than analyzing time-varying properties. This is why we assume time-invariance.

In this setting, the sum-product equations around the factor h_t are not analytically tractable. Therefore, we changed the constraints associated with h_t (25b) to those given in Theorem 2 as follows

$$\int q(x'_t, z_t, Q_t)\mathrm{d}x'_t\mathrm{d}z_t = q(Q_t),\ \int q(x'_t, z_t, Q_t)\mathrm{d}Q_t = q(x'_t, z_t) \tag{39a}$$

$$\int q(Q_t)\mathrm{d}Q_t = 1,\ \int q(x'_t, z_t)\mathrm{d}x'_t\mathrm{d}z_t = 1\,. \tag{39b}$$

We removed the data constraint on $q(Q_t)$ and instead included data constraints on the hyper-parameters

$$q(V) = \delta(V - \hat{V}),\ \ q(\nu) = \delta(\nu - \hat{\nu})\,. \tag{40}$$

With the new set of constraints ((39a) and (39b)), we obtained a hybrid of the sum-product and structured VMP algorithm, where structured messages around the factor h_t are computed by (36) and the rest of the messages are computed by the sum-product (22). One time segment of the modified FFG along with the messages is given Figure 9. We used the same observations $\hat{\boldsymbol{y}}$ that were generated in Example 1 and the same initialization for the hidden states. For the hyper-parameters of the Wishart prior, we chose $\hat{V} = 0.1 \cdot \mathrm{I}$ and $\hat{\nu} = 2$. Under these constraints, the result of structured variational message passing results along with the Bethe free energy evaluation is given in Figure 9.

4.1.2. Naive Variational Message Passing

As a corollary of Theorem 2, we can consider the special case of a naive mean-field factorization, which is defined for node b as

$$q_b(\boldsymbol{s}_b) = \prod_{i \in \mathcal{E}(b)} q_i(s_i)\,. \tag{41}$$

The naive mean-field constraint (41) transforms the local free energy into

$$\begin{aligned} F[q_b, f_b] &= F[\{q_i\}, f_b] \\ &= \sum_{i \in \mathcal{E}(b)} \int q_i(s_i) \log q_i(s_i)\,\mathrm{d}s_i - \int \Big\{ \prod_{i \in \mathcal{E}(b)} q_i(s_i) \Big\} \log f_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_b\,. \end{aligned} \tag{42}$$

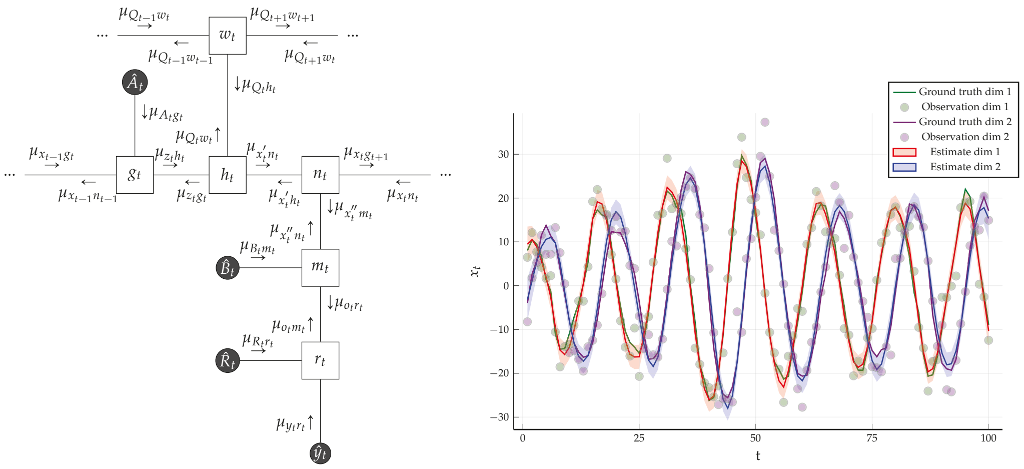


Figure 9. (**Left**) One time segment of the FFG corresponding to the linear Gaussian state space model specified in Example 2 with the sum-product messages computed according to (36). (**Right**) The small dots indicate the noisy observations that are synthetically generated by the linear state space model of (23) using matrices specified in (24). The posterior distribution of the hidden states inferred by structured variational message passing is depicted with shaded regions representing plus and minus one variances. The minimum of the evaluated Bethe free energy over all iterations is $F[q, f] = 586.178$ (compared to $F[q, f] = 580.698$ in Example 1). The posterior distribution for the precision matrix is given by $Q \sim \mathcal{W}\left(\begin{bmatrix} 0.00266 & 0.000334 \\ 0.00034 & 0.00670 \end{bmatrix}, 102.0\right)$.

Corollary 1. *Naive Variational Message Passing: Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the induced subgraph $\mathcal{G}(b, j)$ with a naive mean-field factorization $l(b) = \{i$such that for all $i \in \mathcal{E}(b)\}$. Let $m \in l(b)$ be the cluster where $j = m$. Given the local polytope of* (33)*, the local stationary solutions to* (34) *are given by*

$$q_b^{m,*}(\boldsymbol{s}_b^m) = q_j^*(s_j) = \frac{\mu_{jb}^*(s_j)\mu_{jc}^*(s_j)}{\int \mu_{jb}^*(s_j)\mu_{jc}^*(s_j)\,\mathrm{d}s_j},$$

where the messages $\mu_{jc}^*(s_j)$ are the fixed points of the following iterations

$$\mu_{jc}^{(k+1)}(s_j) = \exp\left(\int \Big\{ \prod_{\substack{i \in \mathcal{E}(b) \\ i \neq j}} q_i^{(k)}(s_i) \Big\} \log f_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_{b\setminus j}\right), \tag{43}$$

where k is an iteration index.

Proof. See Appendix D.6. □

The naive mean-field factorization limits the search space of beliefs by imposing strict constraints on the variational posterior. As a result, the variational posterior also loses flexibility. To improve inference performance for sparse Bayesian learning, the authors of [40] proposes a hybrid mechanism by augmenting naive mean-field VMP with sum-product updates. This hybrid scheme reduces the complexity of the sum-product algorithm, while improving the accuracy of the naive VMP approach. In [41], naive VMP is applied to semi-parametric regression and allows for scaling of regression models to large data sets.

Example 3. *As a follow up on Example 2, we relaxed the constraints in ((39a) and (39b)) to the following constraints presented in Corollary 1 as*

$$\int q(x'_t, z_t, Q_t)\mathrm{d}x'_t\mathrm{d}z_t = q(Q_t), \int q(x'_t, z_t, Q_t)\mathrm{d}Q_t = q(x'_t, z_t) = q(x'_t)q(z_t) \tag{44a}$$

$$\int q(Q_t)\mathrm{d}Q_t = 1, \int q(x'_t)\mathrm{d}x'_t = 1, \int q(z_t)\mathrm{d}z_t = 1\,. \tag{44b}$$

The FFG remains the same and we use identical data constraints as in Example 2. Together with constraint (44), we obtained a hybrid of naive variational message passing and sum-product message passing algorithm where the messages around the factor h_t are computed by (43) and the rest of the messages by sum-product (22). Using the same data as in Example 1, the results for naive VMP are given in Figure 10 along with the evaluated Bethe free energy.

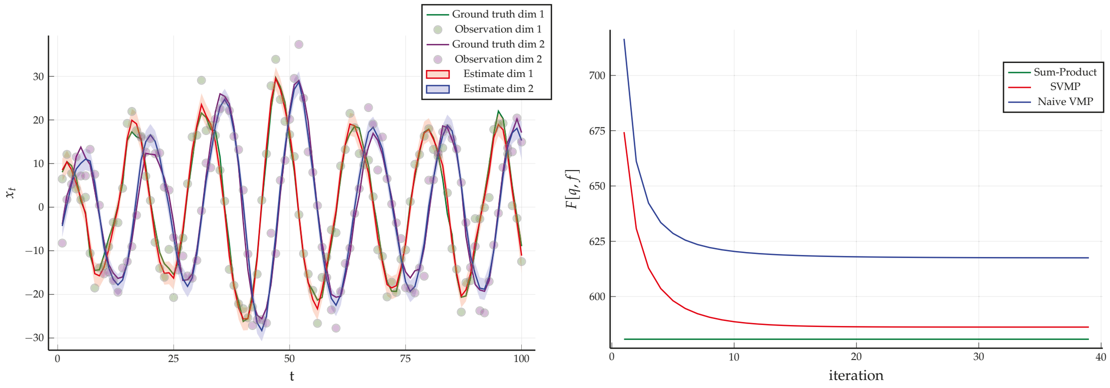


Figure 10. (**Left**) The small dots indicate the noisy observations that were synthetically generated by the linear state space model of (23) using matrices specified in (24). The posterior distribution for the hidden states inferred by naive variational message passing is depicted with shaded regions representing plus and minus one variances. The minimum of the evaluated Bethe free energy over all iterations is $F[q, f] = 617.468$, which is more than for the less-constrained Example 2 (with $F[q, f] = 586.178$) and Example 1 (with $F[q, f] = 580.698$). The posterior for the precision matrix is given by $Q \sim \mathcal{W}\left(\begin{bmatrix} 0.00141 & -6.00549e^{-5} \\ -6.00549e^{-5} & 0.00187 \end{bmatrix}, 102.0\right)$. (**Right**) A comparison of the Bethe free energies for sum-product, structured and naive variational message passing algorithms for the data generated in Example 1.

4.2. Form Constraints

Form constraints limit the functional form of the variational factors $q_a(\mathbf{s}_a)$ and $q_i(s_i)$. One of the most widely used form constraints, the data constraint, is also illustrated in Appendix A.

4.2.1. Data Constraints

A data constraint can be viewed as a special case of (9b), where the belief q_j is constrained to be a Dirac-delta function [42], such that

$$\int q_a(\mathbf{s}_a)\, \mathrm{d}\mathbf{s}_{a\setminus j} = q_j(s_j) = \delta(s_j - \hat{s}_j)\,, \tag{45}$$

where $\hat{s}_j$ is a known value, e.g., an observation.

Lemma 4. *Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the node-induced subgraph $\mathcal{G}(b)$ (Figure 3). Then local stationary solutions to the Lagrangian*

$$L_b[q_b, f_b] = F[q_b, f_b] + \psi_b\left[\int q_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_b - 1\right] + \sum_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \int \lambda_{ib}(s_i)\left[q_i(s_i) - \int q_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_{b\setminus i}\right]\mathrm{d}s_i +$$
$$\int \lambda_{jb}(s_j)\left[\delta(s_j - \hat{s}_j) - \int q_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_{b\setminus j}\right]\mathrm{d}s_j + C_b\,. \tag{46}$$

where C_b collects all terms that are independent of q_b, are of the form

$$q_b(\boldsymbol{s}_b) = \frac{f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)}\mu_{ib}(s_i)}{\int f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)}\mu_{ib}(s_i)\mathrm{d}\boldsymbol{s}_b}\,. \tag{47}$$

Proof. See Appendix D.7. □

Theorem 3. *Data-Constrained Sum-Product: Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the induced subgraph $\mathcal{G}(b, j)$ (Figure 11). Given the local polytope*

$$\mathcal{L}(\mathcal{G}(b, j)) = \{q_b \text{ s.t. } (45)\}\,, \tag{48}$$

the local stationary solutions to

$$q_b^* = \arg\min_{\mathcal{L}(\mathcal{G}(b,j))} F[q, f]\,,$$

are of the form

$$q_b^*(\boldsymbol{s}_b) = \frac{f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)}\mu_{ib}^*(s_i)}{\int f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)}\mu_{ib}^*(s_i)\mathrm{d}\boldsymbol{s}_b}\,, \tag{49}$$

with message

$$\mu_{jb}^*(s_j) = \delta(s_j - \hat{s}_j)\,. \tag{50}$$

Proof. See Appendix D.8. □

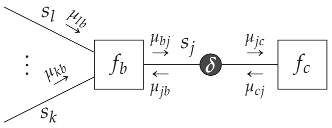


Figure 11. Visualization of a subgraph $\mathcal{G}(b, j)$ with indicated messages, where the dark circled delta indicates a data constraint—i.e., the variable s_j is constrained to have a distribution of the form $\delta(s_j - \hat{s}_j)$.

Note that the resulting message $\mu_{jb}^*(s_j)$ to node b does not depend on messages from node c, as would be the case for a sum-product update. By the symmetry of Theorem 3 for the subgraph $\mathcal{L}\{\mathcal{G}(c, j)\}$, (A32) identifies

$$\mu_{cj}(s_j) = \int f_c(\boldsymbol{s}_c) \prod_{\substack{i \in \mathcal{E}(c) \\ i \neq j}} \mu_{ic}(s_i) \, \mathrm{d}\boldsymbol{s}_{c \setminus j} \neq \delta(s_j - \hat{s}_j) \, .$$

This implies that messages incoming to a data constraint (such as μ_{cj}) are not further propagated through the data constraint. The data constraint thus effectively introduces a conditional independence between the variables of neighboring factors (conditioned on the shared constrained variable). Interestingly, this is similar to the notion of an intervention [43], where a decision variable is externally forced to a realization.

Data constraints allow information from data sets to be absorbed into the model. Essentially, (variational) Bayesian machine learning is an application of inference in a graph with data constraints. In our framework, data are a constraint, and machine learning via Bayes rule follows naturally from the minimization of the Bethe free energy (see also Appendix A).

4.2.2. Laplace Propagation

A second type of form constraint we consider is the Laplace constraint, see also [14]. Consider a second-order Taylor approximation on the local log-node function

$$\mathcal{L}_a(\boldsymbol{s}_a) = \log f_a(\boldsymbol{s}_a) \, , \tag{51}$$

around an approximation point $\hat{\boldsymbol{s}}_a$, as

$$\tilde{\mathcal{L}}_a(\boldsymbol{s}_a; \hat{\boldsymbol{s}}_a) = \mathcal{L}_a(\hat{\boldsymbol{s}}_a) + \nabla^{\top} \mathcal{L}_a(\hat{\boldsymbol{s}}_a)(\boldsymbol{s}_a - \hat{\boldsymbol{s}}_a) + \frac{1}{2}(\boldsymbol{s}_a - \hat{\boldsymbol{s}}_a)^{\top} \nabla^2 \mathcal{L}_a(\hat{\boldsymbol{s}}_a)(\boldsymbol{s}_a - \hat{\boldsymbol{s}}_a) \, . \tag{52}$$

From this approximation, we define the Laplace-approximated node function as

$$\tilde{f}_a(\boldsymbol{s}_a; \hat{\boldsymbol{s}}_a) \triangleq \exp\big(\tilde{\mathcal{L}}_a(\boldsymbol{s}_a; \hat{\boldsymbol{s}}_a)\big) \, , \tag{53}$$

which is substituted in the local free energy to obtain the Laplace-encoded local free energy as

$$F[q_a, \tilde{f}_a; \hat{\boldsymbol{s}}_a] = \int q_a(\boldsymbol{s}_a) \log \frac{q_a(\boldsymbol{s}_a)}{\tilde{f}_a(\boldsymbol{s}_a; \hat{\boldsymbol{s}}_a)} \, \mathrm{d}\boldsymbol{s}_a \, . \tag{54}$$

It follows that the Laplace-encoded optimization of the local free energy becomes

$$q_a^* = \arg\min_{q_a} L_a[q_a, \tilde{f}_a; \hat{\boldsymbol{s}}_a] \, , \tag{55}$$

where the Lagrangian L_a imposes the marginalization and normalization constraints of (9) on (54).

Lemma 5. *Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the node-induced subgraph $\mathcal{G}(b)$ (Figure 12). The stationary points of the Laplace-approximated Lagrangian (55) as a functional of q_b,*

$$\begin{aligned} L_b[q_b, \tilde{f}_b; \hat{\boldsymbol{s}}_b] = F[q_b, \tilde{f}_b; \hat{\boldsymbol{s}}_b] + \psi_b \left[\int q_b(\boldsymbol{s}_b) \, \mathrm{d}\boldsymbol{s}_b - 1 \right] + \\ \sum_{i \in \mathcal{E}(b)} \int \lambda_{ib}(s_i) \left[q_i(s_i) - \int q_b(\boldsymbol{s}_b) \, \mathrm{d}\boldsymbol{s}_{b \setminus i} \right] \mathrm{d}s_i + C_b \, , \end{aligned} \tag{56}$$

where C_b collects all terms that are independent of q_b, which are of the form

$$q_b(\boldsymbol{s}_b) = \frac{\tilde{f}_b(\boldsymbol{s}_b;\hat{\boldsymbol{s}}_b) \prod_{i\in\mathcal{E}(b)} \mu_{ib}(s_i)}{\int \tilde{f}_b(\boldsymbol{s}_b;\hat{\boldsymbol{s}}_b) \prod_{i\in\mathcal{E}(b)} \mu_{ib}(s_i)\mathrm{d}\boldsymbol{s}_b}\,. \tag{57}$$

Proof. See Appendix D.9. □

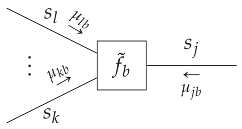


Figure 12. The subgraph around a Laplace-approximated node b with indicated messages.

We can now formulate Laplace propagation as an iterative procedure, where the approximation point $\hat{\boldsymbol{s}}_b$ is chosen as the mode of the belief $q_b(\boldsymbol{s}_b)$.

Theorem 4. *Laplace Propagation: Given a TFFG $\mathcal{G} = (\mathcal{V},\mathcal{E})$, consider the induced subgraph $\mathcal{G}(b,j)$ (Figure 13) with the Laplace-encoded factor $\tilde{f}_b$ as per (53). We write the model (1) with the Laplace-encoded factor $\tilde{f}_b$ substituted for f_b, as $\tilde{f}$. Given the local polytope $\mathcal{L}(\mathcal{G}(b,j))$ of (14), the local stationary solutions to*

$$\{q_b^*, q_j^*\} = \arg \min_{\mathcal{L}(\mathcal{G}(b,j))} F[q,\tilde{f};\hat{\boldsymbol{s}}_b]\,, \tag{58}$$

are given by

$$q_b^*(\boldsymbol{s}_b) = \frac{\tilde{f}_b(\boldsymbol{s}_b;\hat{\boldsymbol{s}}_b^*) \prod_{i\in\mathcal{E}(b)} \mu_{ib}^*(s_i)}{\int \tilde{f}_b(\boldsymbol{s}_b;\hat{\boldsymbol{s}}_b^*) \prod_{i\in\mathcal{E}(b)} \mu_{ib}^*(s_i)\mathrm{d}\boldsymbol{s}_b}$$

$$q_j^*(s_j) = \frac{\mu_{jb}^*(s_j)\mu_{jc}^*(s_j)}{\int \mu_{jb}^*(s_j)\mu_{jc}^*(s_j)\mathrm{d}s_j}\,,$$

with $\hat{\boldsymbol{s}}_b^*$ and the messages $\mu_{jc}^*(s_j)$ the fixed points of

$$\hat{\boldsymbol{s}}_b^{(k)} = \arg\max_{\boldsymbol{s}_b} \log q_b^{(k)}(\boldsymbol{s}_b)$$

$$q_b^{(k+1)}(\boldsymbol{s}_b) = \frac{\tilde{f}_b(\boldsymbol{s}_b;\hat{\boldsymbol{s}}_b^{(k)}) \prod_{i\in\mathcal{E}(b)} \mu_{ib}^{(k)}(s_i)}{\int \tilde{f}_b(\boldsymbol{s}_b;\hat{\boldsymbol{s}}_b^{(k)}) \prod_{i\in\mathcal{E}(b)} \mu_{ib}^{(k)}(s_i)\mathrm{d}\boldsymbol{s}_b}$$

$$\mu_{jc}^{(k+1)}(s_j) = \int \tilde{f}_b(\boldsymbol{s}_b;\hat{\boldsymbol{s}}_b^{(k)}) \prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \mu_{ib}^{(k)}(s_i)\mathrm{d}\boldsymbol{s}_{b\setminus j}\,.$$

Proof. See Appendix D.10. □

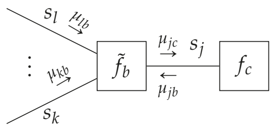


Figure 13. Visualization of a subgraph with indicated Laplace propagation messages. The node function f_b is denoted by $\tilde{f}_b$ according to (53).

A Laplace propagation is introduced in [14] as an algorithm that propagates mean and variance information when exact updates are expensive to compute. Laplace propagation has found applications in the context of Gaussian processes and support vector machines [14]. In the jointly normal case, Laplace propagation coincides with sum-product and expectation propagation [14,18].

4.2.3. Expectation Propagation

Expectation propagation can be derived in terms of constraint manipulation by relaxing the marginalization constraints to expectation constraints. Expectation constraints are of the form

$$\int q_a(\boldsymbol{s}_a) T_i(s_i) \mathrm{d}\boldsymbol{s}_a = \int q_i(s_i) T_i(s_i) \mathrm{d}s_i \,, \tag{59}$$

for a given function (statistic) $T_i(s_i)$. Technically, the statistic $T_i(s_i)$ can be chosen arbitrarily. Nevertheless, they are often chosen as sufficient statistics of an exponential family distribution. An exponential family distribution is defined by

$$q_i(s_i) = h(s_i) \exp\Big(\eta_i^\top T_i(s_i) - \log Z(\eta_i)\Big)\,, \tag{60}$$

where η_i is the natural parameter, $Z(\eta_i)$ is the partition function, $T_i(s_i)$ is the sufficient statistics and $h(s_i)$ is a base measure [24]. The reason $T_i(s_i)$ is a sufficient statistic is because if there are observed values of the random variable s_i, then the parameter η_i can be estimated by using only the statistics $T_i(s_i)$. This means that the estimator of η_i will depend only on the statistics.

The idea behind expectation propagation [18] is to relax the marginalization constraints with moment-matching constraints by choosing sufficient statistics from exponential family distributions [12]. Relaxation allows approximating the marginals of the sum-product algorithm with exponential family distributions. By keeping the marginals within the exponential family, the complexity of the resulting computations is reduced.

Lemma 6. *Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the node-induced subgraph $\mathcal{G}(b)$ (Figure 3). The stationary points of the Lagrangian*

$$\begin{aligned} L_b[q_b, f_b] = F[q_b, f_b] + \psi_b\left[\int q_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_b - 1\right] + \sum_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \int \lambda_{ib}(s_i)\left[q_i(s_i) - \int q_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_{b\setminus i}\right]\mathrm{d}s_i + \\ \eta_{jb}^\top\left[\int q_j(s_j)T_j(s_j)\mathrm{d}s_j - \int q_b(\boldsymbol{s}_b)T_j(s_j)\mathrm{d}\boldsymbol{s}_b\right] + C_b\,, \end{aligned} \tag{61}$$

with sufficient statistics T_j, and where C_b collects all terms that are independent of q_b, are of the form

$$q_b(\boldsymbol{s}_b) = \frac{f_b(\boldsymbol{s}_b) \prod\limits_{i\in\mathcal{E}(b)} \mu_{ib}(s_i)}{\int f_b(\boldsymbol{s}_b) \prod\limits_{i\in\mathcal{E}(b)} \mu_{ib}(s_i) \mathrm{d}\boldsymbol{s}_b}\,, \tag{62}$$

with incoming exponential family message

$$\mu_{jb}(s_j) = \exp\Big(\eta_{jb}^\top T_j(s_j)\Big). \tag{63}$$

Proof. See Appendix D.11. □

Lemma 7. *Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider an edge-induced subgraph $\mathcal{G}(j)$ (Figure 4). The stationary solutions of the Lagrangian*

$$L_j[q_j] = H[q_j] + \psi_j\left[\int q_j(s_j)\,\mathrm{d}s_j - 1\right] + \sum_{a\in\mathcal{V}(j)} \eta_{ja}^\top\left[\int q_j(s_j)T_j(s_j)\mathrm{d}s_j - \int q_a(\boldsymbol{s}_a)T_j(s_j)\mathrm{d}\boldsymbol{s}_a\right] + C_j\,,$$

with sufficient statistics $T_j(s_j)$, and where C_j collects all terms that are independent of q_j, are of the form

$$q_j(s_j) = \frac{\exp\Big([\eta_{jb} + \eta_{jc}]^\top T_j(s_j)\Big)}{\int \exp\Big([\eta_{jb} + \eta_{jc}]^\top T_j(s_j)\Big)\mathrm{d}s_j}. \tag{64}$$

Proof. See Appendix D.12. □

Theorem 5. *Expectation Propagation: Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the induced subgraph $\mathcal{G}(b, j)$ (Figure 5). Given the local polytope*

$$\mathcal{L}(\mathcal{G}(b,j)) = \{q_b \text{ s.t. (9a), and } q_j \text{ s.t. (59) and (10)}\}\,, \tag{65}$$

and $\mu_{jb}(s_j) = \exp\Big(\eta_{jb}^\top T_j(s_j)\Big)$ an exponential family message (from Lemma 6). Then, the local stationary solutions to (15) are given by

$$q_b^*(\boldsymbol{s}_b) = \frac{f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)}\mu_{ib}^*(s_i)}{\int f_b(\boldsymbol{s}_b)\prod_{i\in\mathcal{E}(b)}\mu_{ib}^*(s_i)\mathrm{d}\boldsymbol{s}_b} \tag{66a}$$

$$q_j^*(s_j) = \frac{\exp\Big([\eta_{jb}^* + \eta_{jc}^*]^\top T_j(s_j)\Big)}{\int \exp\Big([\eta_{jb}^* + \eta_{jc}^*]^\top T_j(s_j)\Big)\mathrm{d}s_j}, \tag{66b}$$

with η_{jb}^*, η_{jc}^* and $\mu_{jc}^*(s_j)$ being the fixed points of the iterations

$$\tilde{\mu}_{jc}^{(k)}(s_j) = \int f_b(\boldsymbol{s}_b)\prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}}\mu_{ib}^{(k)}(s_i)\,\mathrm{d}\boldsymbol{s}_{b\setminus j}$$

$$\tilde{q}_j^{(k)}(s_j) = \frac{\mu_{jb}^{(k)}(s_j)\tilde{\mu}_{jc}^{(k)}(s_j)}{\int \mu_{jb}^{(k)}(s_j)\tilde{\mu}_{jc}^{(k)}(s_j)\mathrm{d}s_j}.$$

By moment-matching on $\tilde{q}_j^{(k)}(s_j)$, we obtain the natural parameter $\tilde{\eta}_j^{(k)}$. The message update then follows from

$$\eta_{jc}^{(k)} = \tilde{\eta}_j^{(k)} - \eta_{jb}^{(k)}$$

$$\mu_{jc}^{(k+1)}(s_j) = \exp\Big(T_j(s_j)^\top \eta_{jc}^{(k)}\Big).$$

Proof. See Appendix D.13. □

Moment-matching can be performed by solving [24] (Proposition 3.1)

$$\nabla_{\eta_j} \log Z_j(\eta_j) = \int \tilde{q}_j(s_j)\, T_j(s_j)\, \mathrm{d}s_j$$

for η_j, where

$$Z_j(\eta_j) = \int \exp\Big(\eta_j^\top T_j(s_j)\Big) \mathrm{d}s_j\,.$$

In practice, for a Gaussian approximation, the natural parameters can be obtained by converting the matched mean and variance of $\tilde{q}_j(s_j)$ to the canonical form [18]. Computing the moments of $\tilde{q}_j(s_j)$ is often challenging due to lack of closed form solutions of the normalization constant. In order to address the computation of moments in EP, Ref. [44] proposes to evaluate challenging moments by quadrature methods. For multivariate random variables, moment-matching by spherical radial cubature would be advantageous as it will reduce the computational complexity [45]. Another popular way of evaluating the moments is through importance sampling [46] (Ch. 7) and [47].

Expectation propagation has been utilized in various applications ranging from time series estimation with Gaussian processes [48] to Bayesian learning with stochastic natural gradients [49]. When the likelihood functions for Gaussian process classification are not Gaussian, EP is often utilized [50] (Chapter 3). In [51], a message passing-based expectation propagation algorithm is developed for models that involve both continuous and discrete random variables. Perhaps the most practical applications of EP are in the context of probabilistic programming [52], where it is heavily used in real-world applications.

4.3. Hybrid Constraints

In this section, we consider hybrid methods that combine factorization and form constraints, and formalize some well-known algorithms in terms of message passing.

4.3.1. Mean-Field Variational Laplace

Mean-field variational Laplace applies the mean-field factorization to the Laplace-approximated factor function. The appeal of this method is that all messages outbound from the Laplace-approximated factor can be represented by Gaussians.

Theorem 6. *Mean-field variational Laplace: Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the induced subgraph $\mathcal{G}(b, j)$ (Figure 13) with the Laplace-encoded factor $\tilde{f}_b$ as per (53). We write the model (1) with substituted Laplace-encoded factor $\tilde{f}_b$ for f_b, as $\tilde{f}$. Furthermore, assume a naive mean-field factorization $l(b) = \{\{i\}$ for all $i \in \mathcal{E}(b)\}$. Let $m \in l(b)$ be the cluster where $j = m$. Given the local polytope of (33), the local stationary solutions to*

$$\{q_b^{m,*}, q_j^*\} = \arg \min_{\mathcal{L}(\mathcal{G}(b,j))} F[q, \tilde{f}; \hat{\boldsymbol{s}}_b]\,, \tag{67}$$

are given by

$$q_b^{m,*}(\boldsymbol{s}_b^m) = q_j^*(s_j) = \frac{\mu_{jb}^*(s_j)\mu_{jc}^*(s_j)}{\int \mu_{jb}^*(s_j)\mu_{jc}^*(s_j)\, \mathrm{d}s_j}\,,$$

where μ_{jc}^* represents the fixed points of the following iterations

$$\mu_{jc}^{(k+1)}(s_j) = \exp\left(\int \Big(\prod_{\substack{i \in \mathcal{E}(b) \\ i \neq j}} q_i^{(k)}(s_i)\Big) \log \tilde{f}_b(\boldsymbol{s}_b; \hat{\boldsymbol{s}}_b^{(k)})\, \mathrm{d}\boldsymbol{s}_{b\setminus j}\right), \tag{68}$$

with

$$\hat{\boldsymbol{s}}_b^{(k)} = \arg\max_{\boldsymbol{s}_b} \; \log q_b^{(k)}(\boldsymbol{s}_b)\,.$$

Proof. See Appendix D.14. □

Conveniently, under these constraints, every outbound message from node b will be proportional to a Gaussian. Substituting the Laplace-approximated factor function, we obtain:

$$\log \mu_{jc}^{(k)}(s_j) = \int \Big(\prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} q_i^{(k)}(s_i) \Big) \tilde{\mathcal{L}}_b(\boldsymbol{s}_b; \hat{\boldsymbol{s}}_b^{(k)})\, \mathrm{d}\boldsymbol{s}_{b\setminus j} + C\,.$$

Resolving this expectation yields a quadratic form in s_j, which after completing the square leads to a proportionally Gaussian message $\mu_{jc}(s_j)$. This argument holds for any edge adjacent to b, and therefore for all outbound messages from node b. Moreover, if the incoming messages are represented by Gaussians as well (e.g., because these are also computed under the mean-field variational Laplace constraint), then all beliefs on the adjacent edges to b will also be Gaussian. This significantly simplifies the procedure of computing the expectations, which illustrates the computational appeal of mean-field variational Laplace.

Mean-field variational Laplace is widely used in dynamic causal modeling [53] and more generally in cognitive neuroscience, partly because the resulting computations are deemed neurologically plausible [54–56].

4.3.2. Expectation Maximization

Expectation Maximization (EM) can be viewed as a hybrid algorithm that combines a structured variational factorization with a Dirac-delta constraint, where the constrained value itself is optimized. Given a structured mean-field factorization $l(a) \subseteq \mathcal{P}(a)$, with a single-edge cluster $m = j$, then expectation maximization considers local factorizations of the form

$$q_a(\boldsymbol{s}_a) = \delta(s_j - \theta_j) \prod_{\substack{n\in l(a)\\ n\neq m}} q_a^n(\boldsymbol{s}_a^n), \tag{69}$$

where the belief for s_j is constrained by a Dirac-delta distribution, similar to Section 4.2.1. In (69), however, the variable s_j represents a random variable with (unknown) value $\theta_j \in \mathbb{R}^d$, where d is the dimension of the random variable s_j. We explicitly use the notation θ_j (as opposed to $\hat{s}_j$ for the data constraint in Section 4.2.1) to clarify that this value is a parameter for the constrained belief over s_j that will be optimized—that is, θ_j does not represent a model parameter in itself. To make this distinction even more explicit, in the context of optimization, we will refer to Dirac-delta constraints as point-mass constraints.

The factor-local free energy $F[q_a, f_a; \theta_j]$ then becomes a function of the θ_j parameter.

Theorem 7. *Expectation maximization: Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider the induced subgraph $\mathcal{G}(b, j)$ (Figure 14) with a structured mean-field factorization $l(b) \subseteq \mathcal{P}(b)$, with local clusters $n \in l(b)$. Let $m \in l(b)$ be the cluster where $j = m$. Given the local polytope*

$$\mathcal{L}(\mathcal{G}(b,j)) = \left\{ q_b^n \text{ for all } n \in l(b) \text{ s.t. } (29a) \right\}, \tag{70}$$

the local stationary solutions to

$$\theta_j^* = \arg\min_{\mathcal{L}(\mathcal{G}(b,j))} F[q, f; \theta_j]\,,$$

are given by the fixed points of

$$\mu_{bj}^{(k+1)}(s_j) = \exp\left(\int \left\{ \prod_{\substack{n \in l(b) \\ n \neq m}} q_b^{n,(k)}(\boldsymbol{s}_b^n) \right\} \log f_b(\boldsymbol{s}_b)\, \mathrm{d}\boldsymbol{s}_{b \setminus j}\right) \tag{71a}$$

$$\theta_j^{(k+1)} = \arg\max_{s_j} \left(\log \mu_{bj}^{(k+1)}(s_j) + \log \mu_{cj}^{(k+1)}(s_j) \right). \tag{71b}$$

Proof. See Appendix D.15. □

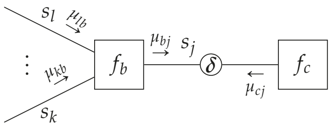


Figure 14. Visualization of a subgraph $\mathcal{G}(b, j)$ with indicated messages. The open circle indicates a point-mass constraint of the form $\delta(s_j - \theta_j)$, where the value θ_j is optimized.

Expectation maximization was formulated in [57] as an iterative method that optimizes log-expectations of likelihood functions, where each EM iteration is guaranteed to increase the expected log-likelihood. Moreover, under some differentiability conditions, the EM algorithm is guaranteed to converge [57] (Theorem 3). A detailed overview of EM for exponential families is available in [24] (Ch. 6). A formulation of EM in terms of message passing is given by [58], where message passing for EM is applied in a filtering and system identification context. In [58], derivations are based on [57] (Theorem 1), whereas our derivations directly follow from variational principles.

Example 4. *Now suppose we do not know the angle θ for the state transition matrix A_t in Example 2 and would like to estimate the value of θ. Moreover, further suppose that we are interested in estimating the hyper-parameters for the prior m_{x_0} and V_{x_0}, as well as the precision matrix for the state transitions Q_t. For this purpose, we changed the constraints of (25a) into EM constraints in accordance with Theorem 7:*

$$q(x_{t-1}, z_t, A_t(\theta)) = \delta(A_t(\theta) - A_t(\hat{\theta}))q(z_t|x_{t-1}, A_t(\theta))q(x_{t-1}) \tag{72a}$$

$$q(x_0, m_{x_0}, V_{x_0}) = q(x_0)\delta(m_{x_0} - \hat{m}_{x_0})\delta(V_{x_0} - \hat{V}_{x_0}), \tag{72b}$$

where we optimize $\hat{\theta}$, $\hat{V}_{x_0}$ and $\hat{m}_{x_0}$ with EM ($\hat{V}_{x_0}$ is further constrained to be positive definite during the optimization procedure). With the addition of the new EM constraints, the resulting FFG is given in Figure 15. The hybrid message passing algorithm consists of structured variational messages around the factor h_t, and sum-product messages around w_t, n_t, m_t and r_t, and EM messages around g_0 and g_t. We used identical observations as in the previous examples. The results for the hybrid SVMP-EM-SP algorithm are given in Figure 16 along with the evaluated Bethe free energy over all iterations.

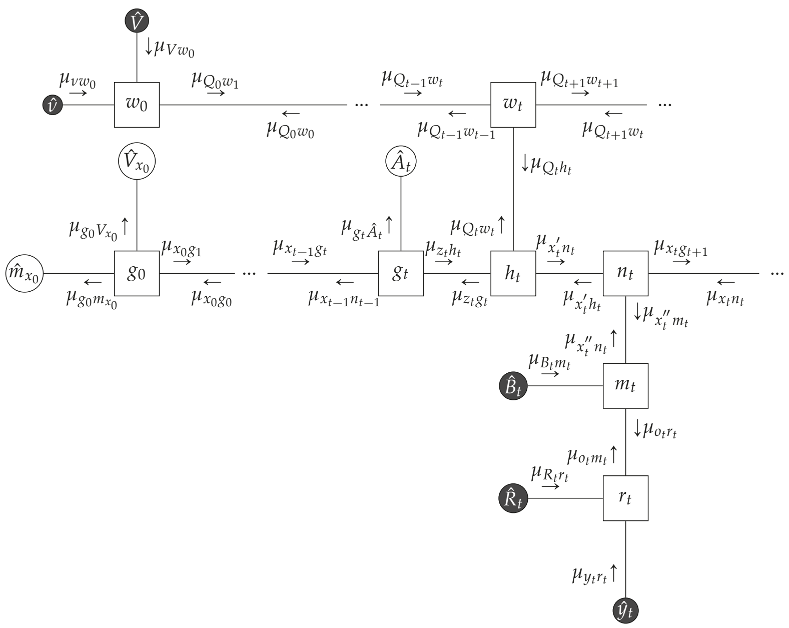


Figure 15. The FFG of the linear Gaussian state space model augmented with the EM constraints in Example 4.

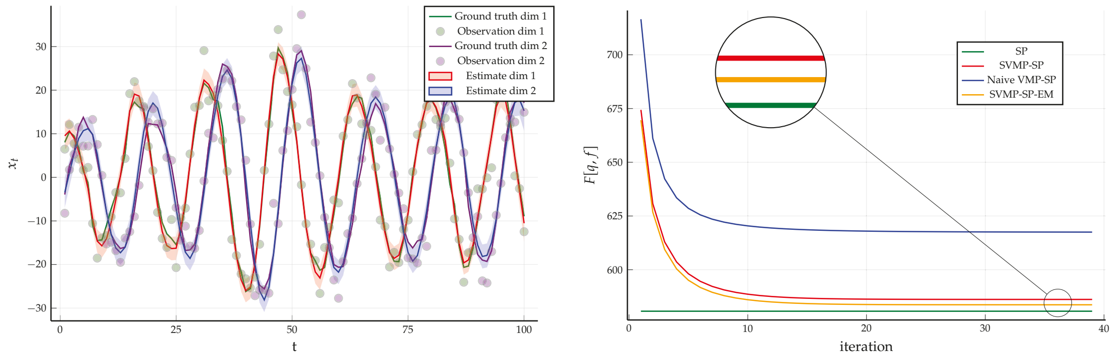


Figure 16. (**Left**) The small dots indicate the noisy observations that are synthetically generated by the linear state space model of (23) using matrices specified in (24). The posterior distribution of the hidden states inferred by structured variational message passing is depicted with shaded regions representing plus and minus one variances. The minimum of the evaluated Bethe free energy over iterations is $F[q, f] = 583.683$. Moreover, the posterior distribution for the precision matrix is given by $Q \sim \mathcal{W}\left(\begin{bmatrix} 0.00286 & 0.00038 \\ 0.00038 & 0.0.00691 \end{bmatrix}, 102.0\right)$. The EM estimates are $\theta = \pi/7.821$, $\hat{m}_{x_0} = [7.23, -7.016]$ and $\hat{V}_{x_0} = \begin{bmatrix} 11.028 & -1.926 \\ -1.926 & 10.918 \end{bmatrix}$. (**Right**) Free energy plots of the 4 algorithms discussed in Examples 1–4 on the same data set.

4.4. Overview of Message Passing Algorithms

In Sections 4.1–4.3, following a high-level recipe pioneered by [15], we presented first-principle derivations of some of the popular message passing-based inference algorithms by manipulating the local constraints of the Bethe free energy. The results are summarized in Table 1.

Crucially, the method of constrained BFE minimization goes beyond the reviewed algorithms. Through creating a new set of local constraints and following similar derivations based on variational calculus, one can obtain new message passing-based inference algorithms that better match the specifics of the generative model or application.

5. Scoring Models by Minimized Variational Free Energy

As discussed in Section 2.2, the variational free energy is an important measure of model performance. In Sections 5.1 and 5.2, we discuss some problems that occur when evaluating the BFE on a TFFG. In Section 5.3, we propose an algorithm that evaluates the constrained BFE as a summation of local contributions on the TFFG.

5.1. Evaluation of the Entropy of Dirac-Delta Constrained Beliefs

For continuous variables, data and point-mass constraints, as discussed in Sections 4.2.1 and 4.3.2 and Appendix A, collapse the information density to infinity, which leads to singularities in entropy evaluation [59]. More specifically, for a continuous variable s_j, the entropies for beliefs of the form $q_j(s_j) = \delta(s_j - \hat{s}_j)$ and $q_a(\boldsymbol{s}_a) = q_{a|j}(\boldsymbol{s}_{a\setminus j}|s_j)\delta(s_j - \hat{s}_j)$ both evaluate to $-\infty$.

In variational inference, it is common to define the VFE only with respect to the latent (unobserved) variables [2] (Section 10.1). In contrast, in this paper, we explicitly define the BFE in terms of an iteration over all nodes and edges (7), which also includes non-latent beliefs in the BFE definition. Therefore, we define

$$
\begin{aligned}
q_j(s_j) = \delta(s_j - \hat{s}_j) &\Rightarrow H[q_j] \triangleq 0\,,\\
q_a(\boldsymbol{s}_a) = q_{a|j}(\boldsymbol{s}_{a\setminus j}|s_j)\delta(s_j - \hat{s}_j) &\Rightarrow H[q_a] \triangleq H[q_{a\setminus j}]\,,
\end{aligned}
$$

where $q_{a|j}(\boldsymbol{s}_{a\setminus j}|s_j)$ indicates the conditional belief and $q_{a\setminus j}(\boldsymbol{s}_{a\setminus j})$ is the joint belief. These definitions effectively remove the entropies for observed variables from the BFE evaluation. Note that although $q_{a\setminus j}(\boldsymbol{s}_{a\setminus j})$ is technically not a part of our belief set (7), it can be obtained by marginalization of $q_a(\boldsymbol{s}_a)$ (9b).

5.2. Evaluation of Node-Local Free Energy for Deterministic Nodes

Another difficulty arises with the evaluation of the node-local free energy $F[q_a]$ for factors of the form

$$
f_a(\boldsymbol{s}_a) = \delta(h_a(\boldsymbol{s}_a))\,. \tag{73}
$$

This type of node function reflects deterministic operations, e.g., $h(x, y, z) = z - x - y$ corresponds to the summation $z = x + y$. In this case, directly evaluating $F[q_a]$ again leads to singularities.

There are (at least) two strategies available in the literature that resolve this issue. The first strategy "softens" the Dirac-delta by re-defining:

$$
f_a(\boldsymbol{s}_a) \triangleq \frac{1}{\sqrt{2\pi\epsilon}} \exp\left(-\frac{1}{2\epsilon} h_a(\boldsymbol{s}_a)^2\right),
$$

with $0 < \epsilon \ll 1$ [17]. A drawback of this approach is that it may alter the model definition in a numerically unstable way, leading to a different inference solution and variational free energy than originally intended.

The second strategy combines the deterministic factor f_a with a neighboring stochastic factor f_b into a new *composite* factor f_c, by marginalizing over a shared variable s_j, leading to [60]

$$
f_c(\boldsymbol{s}_c) \triangleq \int \delta(h_a(\boldsymbol{s}_a))\, f_b(\boldsymbol{s}_b)\, \mathrm{d}s_j\,,
$$

where $\boldsymbol{s}_c = \{\boldsymbol{s}_a \cup \boldsymbol{s}_b\} \backslash s_j$. This procedure has drawbacks for models that involve many deterministic factors—namely, the convenient model modularity and resulting distributed compatibility are lost when large groups of factors are compacted in model-specific composite factors. We propose here a third strategy.

Theorem 8. *Let $f_a(\boldsymbol{s}_a) = \delta(h_a(\boldsymbol{s}_a))$, with $h_a(\boldsymbol{s}_a) = s_j - g_a(\boldsymbol{s}_{a\backslash j})$, and node-local belief $q_a(\boldsymbol{s}_a) = q_{j|a}(s_j|\boldsymbol{s}_{a\backslash j})\, q_{a\backslash j}(\boldsymbol{s}_{a\backslash j})$. Then, the node-local free energy evaluates to*

$$F[q_a, f_a] = \begin{cases} -H[q_{a\backslash j}] & \text{if } q_{j|a}(s_j|\boldsymbol{s}_{a\backslash j}) = \delta(s_j - g_a(\boldsymbol{s}_{a\backslash j})) \\ \infty & \text{otherwise.} \end{cases}$$

Proof. See Appendix D.16. □

An example that evaluates the node-local free energy for a non-trivial deterministic node can be found in Appendix C.

The equality node is a special case deterministic node, with a node function of the form (3). The argument of (Theorem 8) does not directly apply to this node. As the equality node function comprises two Dirac-delta functions, it can not be written in the form of Theorem 8. However, we can still reduce the node-local free energy contribution.

Theorem 9. *Let $f_a(\boldsymbol{s}_a) = \delta(s_j - s_i)\,\delta(s_j - s_k)$, with node-local belief $q_a(\boldsymbol{s}_a) = q_{ik|j}(s_i, s_k|s_j)\, q_j(s_j)$. Then, the node-local free energy evaluates to*

$$F[q_a, f_a] = \begin{cases} -H[q_j] & \text{if } q_{ik|j}(s_i, s_k|s_j) = \delta(s_j - s_i)\,\delta(s_j - s_k) \\ \infty & \text{otherwise.} \end{cases}$$

Proof. See Appendix D.17. □

5.3. Evaluating the Variational Free Energy

We propose here an algorithm that evaluates the BFE on a TFFG representation of a factorized model. The algorithm is based on the following results:

- The definitions for the computation of data-constrained entropies ensure that only variables with associated stochastic beliefs count towards the Bethe entropy. This makes the BFE evaluation consistent with Theorems 3 and 7, where the single-variable beliefs for observed variables are excluded from the BFE definition;
- We assume that a local mean-field factorization $l(a)$ is available for each $a \in \mathcal{V}$ (Section 4.1). If the mean-field factorization is not explicitly defined, we assume $l(a) = \{a\}$ is the unfactored set;
- Deterministic nodes are accounted for by Theorem 8, which reduces the joint entropy to an entropy over the "inbound" edges. Although the belief over the "inbounds" $q_{a\backslash j}(\boldsymbol{s}_{a\backslash j})$ is not a term in the Bethe factorization (8), it can simply be obtained by marginalization of $q_a(\boldsymbol{s}_a)$;
- The equality node is a special case, where we let the node entropy discount the degree of the associated variable in the original model definition. While the BFE definition on a TFFG (7) does not explicitly account for edge degrees, this mechanism implicitly corrects for "double-counting" [17]. In this case, edge selection for counting is arbitrary, because all associated edges are (by definition) constrained to share the same belief (Section 2.1, Theorem 9).

The decomposition of (7) shows that the BFE can be computed by an iteration over the nodes and edges of the graph. As some contributions to the BFE might cancel each other, the algorithm first tracks counting numbers u_a for the average energies

$$U_a[q_a] = -\int q_a(\boldsymbol{s}_a) \log f_a(\boldsymbol{s}_a)\, \mathrm{d}\boldsymbol{s}_a\,,$$

and counting numbers h_k for the (joint) entropies

$$H[q_k] = -\int q_k(\boldsymbol{s}_k) \log q_k(\boldsymbol{s}_k) \, \mathrm{d}\boldsymbol{s}_k \,,$$

which are ultimately combined and evaluated. We used an index k to indicate that the entropy computation may include not only the edges but a generic set of variables. We will give the definition of the set that k belongs to in Algorithm 1.

Algorithm 1 Evaluation of the Bethe free energy on a Terminated Forney-style factor graph.

```
given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
given a local mean-field factorization $l(a)$ for all $a \in \mathcal{V}$
define $q_j(s_j) = \delta(s_j - \hat{s}_j) \Rightarrow H[q_j] \triangleq 0$    ▷ Ignore entropy of Dirac-delta constrained beliefs
define $q_a(\boldsymbol{s}_a) = q_{a|j}(\boldsymbol{s}_{a\setminus j}|s_j)\delta(s_j - \hat{s}_j) \Rightarrow H[q_a] \triangleq H[q_{a\setminus j}]$ ▷ Reduce entropy of Dirac-delta constrained joint beliefs
define $\mathcal{K} = \{a, a\setminus i, n,$ for all $a \in \mathcal{V}, i \in \mathcal{E}(a), n \in l(a)\}$ the set of (joint) belief indices
initialize counting numbers $u_a = 0$ for all $a \in \mathcal{V}$, $h_k = 0$ for all $k \in \mathcal{K}$

for all nodes $a \in \mathcal{V}$ do
    if $a$ is a stochastic node then
        $u_a$ += 1                                    ▷ Count the average energy
        for all clusters $n \in l(a)$ do
            $h_n$ += 1                                ▷ Count the (joint) cluster entropy
        end for
    else if $a$ is an equality node then
        Select an edge $j \in \mathcal{E}(a)$
        $h_j$ += 1                                    ▷ Count the variable entropy
    else                                              ▷ Deterministic node $a$
        Obtain the node function $f_a(\boldsymbol{s}_a) = \delta(s_j - g_a(\boldsymbol{s}_{a\setminus j}))$
        $h_{a\setminus j}$ += 1                       ▷ Count the (joint) entropy of the inbounds
    end if
end for

for all edges $i \in \mathcal{E}$ do
    $h_i$ −= 1                                        ▷ Discount the variable entropy
end for

$U = \sum_{a\in\mathcal{V}} u_a U_a[q_a]$
$H = \sum_{k\in\mathcal{K}} h_k H[q_k]$
return $F = U - H$
```

6. Implementation of Algorithms and Simulations

We have developed a probabilistic programming toolbox *ForneyLab.jl* in the Julia language [61,62]. The majority of algorithms that are reviewed in Table 1 have been implemented in ForneyLab along with variety of demos (https://github.com/biaslab/ForneyLab.jl/tree/master/demo, accessed on 23 June 2021). ForneyLab is extendable and supports postulating new local constraints of the BFE for the creation of custom message passing-based inference algorithms.

In order to limit the length of this paper, we refer the reader to the demonstration folder of ForneyLab and to several of our previous papers with code. For instance, our previous work in [63] implemented a mean-field variational Laplace propagation for the hierarchical Gaussian filter (HGF) [64]. In the follow-up work [65], inference results improved by changing to structured factorization and moment-matching local constraints. In that case, modification of local constraints created a hybrid EP-VMP algorithm that better suited the model. Moreover, in [13], we formulated the idea of *chance constraints* in the form of violation probabilities leading to a new message passing algorithm that supports goal-directed behavior within the context of active inference. A similar line of reasoning led to improved inference procedures for auto-regressive models [66].

7. Related Work

Our work is inspired by the seminal work [17], which discusses the equivalence between the fixed points of the belief propagation algorithm [32] and the stationary points of the Bethe free energy. This equivalence is established through a Lagrangian formalism, which allows for the derivation of Generalized Belief Propagation (GBP) algorithms by introducing region-based graphs and the region-based (Kikuchi) free energy [16].

Region graph-based methods allows for overlapping clusters (Section 4.1) and thus offer a more generic message passing approach. The selection of appropriate regions (clusters), however, proves to be difficult, and the resulting algorithms may grow prohibitively complex. In this context, Ref. [67] addresses how to manipulate regions and manage the complexity of GBP algorithms. Furthermore, Ref. [68] also establishes a connection between GBP and expectation propagation (EP) by introducing structured region graphs.

The inspirational work of [15] derives message passing algorithms by minimization of α-divergences. The stationary points of α-divergences are obtained by a fixed point projection scheme. This projection scheme is reminiscent of the minimization scheme of the expectation propagation (EP) algorithm [18]. Compared to [15], our work focuses on a single divergence objective (namely, the VFE). The work of [12] derives the EP algorithm by manipulating the marginalization and factorization constraints of the Bethe free energy objective (see also Section 4.2.3). The EP algorithm is, however, not guaranteed to converge to a minimum of the associated divergence metric.

To address the convergence properties of the algorithms that are obtained by region graph methods, the outstanding work of [33] derives conditions on the region counting numbers that guarantee the convexity of the underlying objective. In general, however, the constrained Bethe free energy is not guaranteed to be convex and therefore the derived message passing updates are not guaranteed to converge.

8. Discussion

The key message in this paper is that a (variational) Bayesian model designer may tune the tractability-accuracy trade-off for evidence and posterior evaluation through constraint manipulation. It is interesting to note that the technique to derive message passing algorithms is always the same. We followed the recipe pioneered in [15] to derive a large variety of message passing algorithms solely through minimizing constrained Bethe free energy. This minimization leads to local fixed-point equations, which we can interpret as message passing updates on a (terminated) FFG. The presented lemmas showed how the constraints affect the Lagrangians locally. The presented theorems determined the stationary solutions of the Lagrangians and obtained the message passing equations. Thus, if a designer proposes a new set of constraints, then the first place to start is to analyze the effect on the Lagrangian. Once the effect of the constraint on the Lagrangian is known, then variational optimization may result in stationary solutions that can be obtained by a fixed-point iteration scheme.

In this paper, we selected the Forney-style factor graph framework to illustrate our ideas. FFGs are mathematically comparable to the more common bi-partite factor graphs that associate round nodes with variables and square nodes with factors [20]. Bi-partite factor graphs require two distinct types of message updates (one leaving variable nodes and one leaving factor nodes), while message passing on a (T)FFG requires only a single type of message update [69]. The (T)FFG paradigm thus substantially simplifies the derivations and resulting message passing update equations.

The message passing update rules in this paper are presented without guarantees on convergence of the (local) minimization process. In practice, however, algorithm convergence can be easily checked by evaluating the BFE (Algorithm 1) after each belief update.

In future work, we plan on extending the treatment of constraints to formulate sampling-based algorithms such as importance sampling and Hamiltonian Monte Carlo in a message passing framework. While introducing SVMP, we have limited the discussion to local clusters that are not overlapping. We plan to extend variational algorithms to include local clusters that are overlapping without altering the underlying free-energy objective or the graph structure [16,67].

9. Conclusions

In this paper, we formulated a message-passing approach to probabilistic inference by identifying local stationary solutions of a constrained Bethe free energy objective (Sections 3 and 4). The proposed framework constructs a graph for the generative model and specifies local constraints for variational optimization in a local polytope. The constraints are then imposed on the variational objective by a Lagrangian construct. Unconstrained optimization of the Lagrangian then leads to local expressions of stationary points, which can be obtained by iterative execution of the resulting fixed point equations, which we identify with message passing updates.

Furthermore, we presented an approach to evaluate the BFE on a (terminated) Forney-style factor graph (Section 5). This procedure allows an algorithm designer to readily assess the performance of algorithms and models.

We have included detailed derivations of message passing updates (Appendix D) and hope that the presented formulation inspires the discovery of novel and customized message passing algorithms.

Author Contributions: Conceptualization: İ.Ş., T.v.d.L. and B.d.V.; methodology: İ.Ş. and T.v.d.L.; formal analysis, İ.Ş. and T.v.d.L.; investigation: İ.Ş., T.v.d.L. and B.d.V.; software, İ.Ş. and D.B.; validation, İ.Ş.; resources: İ.Ş., T.v.d.L. and B.d.V.; writing—original draft preparation: İ.Ş. and T.v.d.L.; writing—review and editing: T.v.d.L., D.B. and B.d.V.; visualizations: İ.Ş., T.v.d.L. and B.d.V.; supervision: T.v.d.L. and B.d.V.; project administration, B.d.V. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partly financed by GN Hearing A/S.

Acknowledgments: The authors would like to thank the BIASlab team members for many very interesting discussions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

BFE	Bethe Free Energy
BP	Belief Propagation
DC	Data Constraint
EM	Expectation Maximization
EP	Expectation Propagation
FFG	Forney-style Factor Graph
GBP	Generalized Belief Propagation
LP	Laplace Propagation
MFVLP	Mean-Field Variational Laplace
MFVMP	Mean-Field Variational Message Passing

NLE	Negative Log-Evidence
TFFG	Terminated Forney-style Factor Graph
VFE	Variational Free Energy
VMP	Variational Message Passing
SVMP	Structured Variational Message Passing
SP	Sum-Product

Appendix A. Free Energy Minimization by Variational Inference

In this section, we present a pedagogical example of inductive inference. After we establish an intuition, we apply the same principles to a more general context in the further sections. We follow Caticha [42,70], who showed that a constrained free energy functional can be interpreted as a principled objective measure for inductive reasoning, see also [71,72]. The calculus of variations offers a principled method for optimizing this free energy functional.

In this section, we assume an example model

$$f(\boldsymbol{y}, \theta) = f_{\boldsymbol{y}}(\boldsymbol{y}, \theta)\, f_\theta(\theta)\,, \tag{A1}$$

with observed variables $\boldsymbol{y}$ and a single parameter θ.

We define the (variational) free energy (VFE) as

$$F[q, f] = \iint q(\boldsymbol{y}, \theta) \log \frac{q(\boldsymbol{y}, \theta)}{f(\boldsymbol{y}, \theta)}\, \mathrm{d}\boldsymbol{y}\, \mathrm{d}\theta\,. \tag{A2}$$

The goal is to find a posterior

$$q^* = \arg\min_{q \in \mathcal{Q}} F[q, f] \tag{A3}$$

that minimizes the free energy subject to some pre-specified constraints. These constraints may include form or factorization constraints on q (to be discussed later) or relate to observations of a signal $\boldsymbol{y}$.

As an example, assume that we obtained some measurements $\boldsymbol{y} = \hat{\boldsymbol{y}}$ and wish to obtain a posterior marginal belief $q^*(\theta)$ over the parameter. We can then incorporate the data in the form of a data constraint

$$\int q(\boldsymbol{y}, \theta)\, \mathrm{d}\theta = \delta(\boldsymbol{y} - \hat{\boldsymbol{y}})\,, \tag{A4}$$

where δ defines a Dirac-delta. The *constrained* free energy can be rewritten by including Lagrange multipliers as

$$L[q, f] = F[q, f] + \gamma\left(\iint q(\boldsymbol{y}, \theta)\, \mathrm{d}\boldsymbol{y}\, \mathrm{d}\theta - 1\right) + \int \lambda(\boldsymbol{y})\left(\int q(\boldsymbol{y}, \theta)\, \mathrm{d}\theta - \delta(\boldsymbol{y} - \hat{\boldsymbol{y}})\right) \mathrm{d}\boldsymbol{y}\,, \tag{A5}$$

where the first term specifies the (to be minimized) free energy objective, the second term a normalization constraint, and the third term the data constraint. Optimization of (A5) can be performed using variational calculus.

Variational calculus considers the impact of a variation in $q(\boldsymbol{y}, \theta)$ on the Lagrangian $L[q, f]$. We define the variation as

$$\delta q(\boldsymbol{y}, \theta) \triangleq \epsilon \phi(\boldsymbol{y}, \theta)\,,$$

where $\epsilon \to 0$, and ϕ is a continuous and differentiable "test" function. The fundamental theorem of variational calculus states that the stationary solutions q^* are obtained by setting $\delta L/\delta q = 0$, where the functional derivative $\delta L/\delta q$ is implicitly defined by Appendix D in [2]:

$$\left.\frac{\mathrm{d}L[q+\epsilon\phi, f]}{\mathrm{d}\epsilon}\right|_{\epsilon=0} = \iint \frac{\delta L}{\delta q}(\boldsymbol{y},\theta)\,\phi(\boldsymbol{y},\theta)\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta\,. \tag{A6}$$

Equation (A6) provides a way to derive the functional derivative through ordinary differentiation. For example, we take the Lagrangian defined by (A5) and work out the left hand side of (A6):

$$\begin{aligned}
\left.\frac{\mathrm{d}L[q+\epsilon\phi, f]}{\mathrm{d}\epsilon}\right|_{\epsilon=0} &= \left.\frac{\mathrm{d}F[q+\epsilon\phi, f]}{\mathrm{d}\epsilon}\right|_{\epsilon=0} + \frac{\mathrm{d}}{\mathrm{d}\epsilon}\gamma\left.\iint (q+\epsilon\phi)\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta\right|_{\epsilon=0} + \frac{\mathrm{d}}{\mathrm{d}\epsilon}\left.\int \lambda(\boldsymbol{y})\int (q+\epsilon\phi)\,\mathrm{d}\theta\,\mathrm{d}\boldsymbol{y}\right|_{\epsilon=0} && \text{(A7a)}\\
&= \iint \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\left((q+\epsilon\phi)\log\frac{(q+\epsilon\phi)}{f}\right)\right|_{\epsilon=0}\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta + \gamma\iint \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}(q+\epsilon\phi)\right|_{\epsilon=0}\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta\\
&\quad + \int \lambda(\boldsymbol{y})\int \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}(q+\epsilon\phi)\right|_{\epsilon=0}\mathrm{d}\theta\,\mathrm{d}\boldsymbol{y} && \text{(A7b)}\\
&= \iint \Big[\underbrace{\log\frac{q(\boldsymbol{y},\theta)}{f(\boldsymbol{y},\theta)} + 1 + \gamma + \lambda(\boldsymbol{y})}_{\delta L[q,f]/\delta q}\Big]\phi(\boldsymbol{y},\theta)\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta\,. && \text{(A7c)}
\end{aligned}$$

Note that, since (A7c) has been written in similar form as (A6), it is easy to identify the functional derivative. This procedure is one of many ways to obtain the functional derivatives [73].

Setting $\delta L[q,f]/\delta q = 0$ we find the stationary solution as

$$\begin{aligned}
q^*(\boldsymbol{y},\theta) &= \exp(-1-\gamma-\lambda(\boldsymbol{y}))\,f(\boldsymbol{y},\theta) && \text{(A8a)}\\
&= \frac{1}{Z}\exp(-\lambda(\boldsymbol{y}))f(\boldsymbol{y},\theta)\,, && \text{(A8b)}
\end{aligned}$$

with $Z = \iint \exp(-\lambda(\boldsymbol{y}))f(\boldsymbol{y},\theta)\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta = \exp(\gamma+1)$. In order to determine the Lagrange multipliers γ and $\lambda(\boldsymbol{y})$, we must substitute the stationary solution (A8b) back into the constraints. The normalization constraint evaluates to

$$\frac{1}{Z}\iint \exp(-\lambda(\boldsymbol{y}))f(\boldsymbol{y},\theta)\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta = 1. \tag{A9}$$

We find that (A9) is always satisfied since $Z = \iint \exp(-\lambda(\boldsymbol{y}))f(\boldsymbol{y},\theta)\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\theta$ by definition. Note, however, that the computation of the normalization constant still depends on the undetermined Lagrange multiplier $\lambda(\boldsymbol{y})$.

The data constraint evaluates to

$$\int q^*(\boldsymbol{y},\theta)\,\mathrm{d}\theta = \frac{1}{Z}\exp(-\lambda(\boldsymbol{y}))\int f(\boldsymbol{y},\theta)\,\mathrm{d}\theta = \delta(\boldsymbol{y}-\hat{\boldsymbol{y}}) \tag{A10}$$

which can be rewritten as

$$\frac{\exp(-\lambda(\boldsymbol{y}))}{Z} = \frac{\delta(\boldsymbol{y}-\hat{\boldsymbol{y}})}{\int f(\boldsymbol{y},\theta)\,\mathrm{d}\theta}\,. \tag{A11}$$

Equation (A11) shows that $\lambda(\boldsymbol{y})$ can satisfy this constraint only if it is proportional to $\delta(\boldsymbol{y}-\hat{\boldsymbol{y}})$. Indeed, substitution of (A11) into (A8b) gives

$$q^*(\boldsymbol{y},\theta) = \frac{f(\boldsymbol{y},\theta)}{\int f(\boldsymbol{y},\theta)\,\mathrm{d}\theta}\delta(\boldsymbol{y}-\hat{\boldsymbol{y}})\,,$$

and the posterior for the parameters evaluates to

$$\begin{aligned} q^*(\theta) &= \int q^*(\boldsymbol{y}, \theta) \mathrm{d}\boldsymbol{y} \\ &= \int \frac{f(\boldsymbol{y}, \theta)}{\int f(\boldsymbol{y}, \theta)\, \mathrm{d}\theta} \delta(\boldsymbol{y} - \hat{\boldsymbol{y}}) \mathrm{d}\boldsymbol{y} \\ &= \frac{f(\hat{\boldsymbol{y}}, \theta)}{\int f(\hat{\boldsymbol{y}}, \theta)\, \mathrm{d}\theta} \\ &= \frac{f_{\boldsymbol{y}}(\hat{\boldsymbol{y}}, \theta) f_\theta(\theta)}{\int f_{\boldsymbol{y}}(\hat{\boldsymbol{y}}, \theta) f_\theta(\theta)\, \mathrm{d}\theta} , \end{aligned}$$

which we recognize as the Bayes rule.

Note that the Bayes rule was derived here as a special case of constrained variational free energy minimization when data constraints are present. This derivation of the Bayes rule seems unnecessarily tedious but the value of this approach to inductive inference is that the same principle applies when other (not data) constraints on q are present.

Appendix B. Lagrangian Optimization and the Dual Problem

With the addition of Lagrange multipliers to the Bethe functional, the resulting Lagrangian depends both on the variational distribution $q(\boldsymbol{s})$ and the Lagrange multipliers $\Psi(\boldsymbol{s})$. Formally, the introduction of the Lagrange multipliers allows us to rewrite the constrained optimization on the local polytope as an unconstrained optimization. We follow [33], and write

$$\min_{q \in \mathcal{L}(\mathcal{G})} F[q] = \min_q \max_\Psi L[q, \Psi] .$$

Weak duality [74] (Chapter 5) then states that

$$\min_q \max_\Psi L[q, \Psi] \geqslant \max_\Psi \min_q L[q, \Psi] .$$

The minimization with respect to q then yields a solution that depends on the Lagrange multipliers, as

$$q^*(\boldsymbol{s}; \Psi) = \arg\min_q L[q, \Psi] .$$

For any given q the Lagrangian is concave in Ψ. Therefore, substituting q^* in the Lagrangian, the maximization over $L[q^*, \Psi]$ yields the unique solution

$$\Psi^*(\boldsymbol{s}) = \arg\max_\Psi L[q^*, \Psi] .$$

Stationary solutions are then given by

$$q^*(\boldsymbol{s}; \Psi^*) = \arg \min_{q \in \mathcal{L}(\mathcal{G})} F[q] .$$

In the current paper, we consider factorized q's (e.g., (8)), and consider variations with respect to the individual factors. We then need to show that the combined stationary points of the individual factors also constitute a stationary point of the total objective.

Consider a Lagrangian having multiple arguments, i.e.,

$$L[\boldsymbol{q}] = L[q_1, \dots, q_n, \dots, q_N] \tag{A12}$$

$$\boldsymbol{q} \triangleq [q_1, \dots, q_N]^\top . \tag{A13}$$

We want to determine the first total variation of the Lagrangian given by

$$\delta L = L[\boldsymbol{q} + \epsilon \boldsymbol{\phi}] - L[\boldsymbol{q}] \tag{A14}$$

$$\boldsymbol{\phi}(\boldsymbol{s}) \triangleq [\phi_1(\boldsymbol{s}), \dots, \phi_N(\boldsymbol{s})]^\top . \tag{A15}$$

By a Taylor series expansion on ϵ we obtain [73] (A.14) and [75] (Equation (23.2))

$$L[\boldsymbol{q} + \epsilon \boldsymbol{\phi}] - L[\boldsymbol{q}] = \sum_{k=1}^{K} \frac{1}{k!} \frac{\mathrm{d}}{\mathrm{d}\epsilon^k} \Big(L^k[\boldsymbol{q} + \epsilon \boldsymbol{\phi}] \Big) \epsilon^k + \mathcal{O}(\epsilon^{K+1}) . \tag{A16}$$

Omitting all terms higher than the first order, we obtain the first variation as

$$\delta L = \frac{\mathrm{d}}{\mathrm{d}\epsilon} (L[\boldsymbol{q} + \epsilon \boldsymbol{\phi}]) \epsilon . \tag{A17}$$

Rearranging the terms and letting ϵ vanish, we obtain the following expression

$$\lim_{\epsilon \to 0} \frac{\delta L}{\epsilon} = \frac{\mathrm{d}}{\mathrm{d}\epsilon} (L[\boldsymbol{q} + \epsilon \boldsymbol{\phi}]) \Big|_{\epsilon = 0} . \tag{A18}$$

Let us assume that the Frechet derivative exists [73] such that we can obtain the following integral representation (It should be noted that this integral expression is not always possible for a generic Lagrangian. That is why we need to assume that the Frechet derivative exists)

$$\frac{\mathrm{d}}{\mathrm{d}\epsilon} (L[\boldsymbol{q} + \epsilon \boldsymbol{\phi}]) \Big|_{\epsilon = 0} = \int \phi(\boldsymbol{s})^\top \frac{\delta L}{\delta \boldsymbol{q}} \mathrm{d}\boldsymbol{s} \tag{A19}$$

where $\frac{\delta L}{\delta \boldsymbol{q}}$ is the variational derivative

$$\frac{\delta L}{\delta \boldsymbol{q}} = \left[\frac{\delta L}{\delta q_1}, \dots, \frac{\delta L}{\delta q_N} \right]^\top \tag{A20}$$

$$\delta q_n = \epsilon \phi_n(\boldsymbol{s}) . \tag{A21}$$

This means that (A19) can be written as [75] (Equation (22.5)) (Here, we use a more generic Lagrangian and our notation is different than in [75]; howeverm the expression is motivated again by a Taylor series expansion on ϵ)

$$\lim_{\epsilon \to 0} \frac{\delta L}{\epsilon} = \frac{\mathrm{d}}{\mathrm{d}\epsilon} (L[\boldsymbol{q} + \epsilon \boldsymbol{\phi}]) \Big|_{\epsilon = 0} = \sum_n \int \phi(\boldsymbol{s}) \frac{\delta L}{\delta q_n} \mathrm{d}\boldsymbol{s} . \tag{A22}$$

Fundamental theorem of variational calculus states that in order for a point to be stationary, the first variation needs to vanish. In order for the first variation to vanish, it is sufficient to have vanishing of the variational derivatives

$$\frac{\delta L}{\delta q_n} = 0 \;\text{ for every }\; n = 1, \dots, N . \tag{A23}$$

Vanishing of individual variational derivatives will mean that that the local stationary points will also correspond to a global stationary point.

Appendix C. Local Free Energy Example for a Deterministic Node

Theorem 8 tells us how to evaluate the node-local free energy for a deterministic node. As an example, consider the node function $f_a(y, x) = \delta(y - \mathrm{sgn}(x))$, with $y \in \{-1, 1\}$ and $x \in \mathbb{R}$ as depicted in Figure A1.

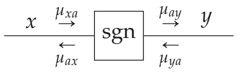


Figure A1. Messages around a "sign" node.

Interestingly, there is information loss in this node because the "sign" mapping is not bijective. Given an incoming Bernoulli distributed message $\mu_{ya}(y) = \mathcal{B}er(y|p)$, the backward outgoing message is derived as

$$\begin{aligned}\mu_{ax}(x) &= \int \mu_{ya}(y)\, \delta(y - \operatorname{sgn}(x))\, \mathrm{d}y \\ &= \begin{cases} p & \text{if } x \geqslant 0 \\ 1-p & \text{if } x < 0\,. \end{cases}\end{aligned}$$

Given a Gaussian distributed incoming message $\mu_{xa}(x) = \mathcal{N}(x|m, \vartheta)$, the resulting belief then becomes

$$\begin{aligned}q_x(x) &= \frac{\mu_{xa}(x)\, \mu_{ax}(x)}{\int \mu_{xa}(x)\, \mu_{ax}(x)\, \mathrm{d}x} \\ &= \begin{cases} \frac{p}{p+\Phi-2p\Phi}\, \mathcal{N}(x|m,\vartheta) & \text{if } x \geqslant 0 \\ \frac{1-p}{p+\Phi-2p\Phi}\, \mathcal{N}(x|m,\vartheta) & \text{if } x < 0\,, \end{cases}\end{aligned}$$

with $\Phi = \int_{-\infty}^{0} \mathcal{N}(x|m,\vartheta)\, \mathrm{d}x$. We define a truncated Gaussian distribution as

$$\mathcal{T}(x|m,\vartheta,a,b) = \begin{cases} \frac{1}{\Phi(a,b;m,\vartheta)} \mathcal{N}(x|m,\vartheta) & \text{if } a \leqslant x \leqslant b\,, \\ 0 & \text{otherwise,} \end{cases}$$

with $\Phi(a,b;m,\vartheta) = \int_a^b \mathcal{N}(x|m,\vartheta)\, \mathrm{d}x$. This leads to

$$q_x(x) = \underbrace{\frac{p(1-\Phi)}{p+\Phi-2p\Phi}}_{K} \mathcal{T}(x|m,\vartheta,-\infty,0) + \underbrace{\frac{(1-p)\Phi}{p+\Phi-2p\Phi}}_{1-K} \mathcal{T}(x|m,\vartheta,0,\infty)\,,$$

as a truncated Gaussian mixture.

The node-local free energy then evaluates to

$$\begin{aligned}F[q_a, f_a] &= -H[q_x] \\ &= \int_{-\infty}^{0} q_x(x) \log q_x(x)\, \mathrm{d}x + \int_{0}^{\infty} q_x(x) \log q_x(x)\, \mathrm{d}x \\ &= -KH[\mathcal{T}(m,\vartheta,-\infty,0)] + K\log K - (1-K)H[\mathcal{T}(m,\vartheta,0,\infty)] + (1-K)\log(1-K) \\ &= -KH[\mathcal{T}(m,\vartheta,-\infty,0)] - (1-K)H[\mathcal{T}(m,\vartheta,0,\infty)] - H[\mathcal{B}er(K)]\,,\end{aligned}$$

as a weighted sum of entropies, which can be computed analytically.

Appendix D. Proofs

Appendix D.1. Proof of Lemma 1

Proof. We apply the variation $\epsilon\phi_b$ to q_b and, as discussed in Appendix A, we can identify the functional derivative $\delta L_b/\delta q_b$ through ordinary differentiation as

$$\left.\frac{\mathrm{d}L_b[q_b + \epsilon\phi_b, f_b]}{\mathrm{d}\epsilon}\right|_{\epsilon=0} = \int \Bigg(\overbrace{\log \frac{q_b(s_b)}{f_b(s_b)} + 1 + \psi_b - \sum_{i\in\mathcal{E}(b)} \lambda_{ib}(s_i)}^{\delta L_b/\delta q_b} \Bigg) \phi_b(s_b)\, \mathrm{d}s_b\,.$$

Setting the functional derivative to zero and identifying

$$\mu_{ib}(s_i) = \exp(\lambda_{ib}(s_i)) \tag{A24}$$

$$\psi_b = \log \int f_b(\boldsymbol{s}_b) \prod_{i \in \mathcal{E}(b)} \mu_{ib}(s_i) \mathrm{d}\boldsymbol{s}_b - 1 \tag{A25}$$

yields the stationary solutions (18) in terms of Lagrange multipliers that are to be determined. □

Appendix D.2. Proof of Lemma 2

Proof. We follow the same procedure as in Appendix D.1, where we apply a variation $\epsilon\phi_j$ to q_j (instead of q_b), and identify the functional derivative $\delta L_j/\delta q_j$ through

$$\left.\frac{\mathrm{d}L_j[q_j + \epsilon\phi_j]}{\mathrm{d}\epsilon}\right|_{\epsilon=0} = \int \left(\overbrace{-\log q_j(s_j) - 1 + \psi_j + \sum_{a \in \mathcal{V}(j)} \lambda_{ja}(s_j)}^{\delta L_j/\delta q_j} \right) \phi_j(s_j)\, \mathrm{d}s_j\,.$$

As the TFFG is terminated, each edge has 2 degrees and the node-induced edge set has only 2 factors, which we denote by f_b and f_c. Then, setting the functional derivative to zero and identifying

$$\mu_{ja}(s_j) = \exp\big(\lambda_{ja}(s_j)\big) \tag{A26}$$

$$\psi_j = -\log \int \mu_{jb}(s_j)\mu_{jc}(s_j)\mathrm{d}s_j + 1 \tag{A27}$$

yields the stationary solution of (20) in terms of the Lagrange multipliers. □

Appendix D.3. Proof of Theorem 1

Proof. The local polytope of (14) constructs the Lagrangians of (17) and (19). Substituting the stationary solutions from Lemmas 1 and 2 in the marginalization constraint,

$$q_j(s_j) = \int q_b(\boldsymbol{s}_b)\, \mathrm{d}\boldsymbol{s}_{b\setminus j}\,,$$

we obtain the following relation

$$\frac{\mu_{jb}(s_j)\mu_{jc}(s_j)}{Z_j} = \frac{1}{Z_b} \int f_b(\boldsymbol{s}_b) \prod_{i \in \mathcal{E}(b)} \mu_{ib}(s_i)\, \mathrm{d}\boldsymbol{s}_{b\setminus j}\,,$$

where we defined the following normalization constants to ensure that the computed marginals are normalized:

$$Z_j = \int \mu_{jb}(s_j)\mu_{jc}(s_j)\mathrm{d}s_j$$

$$Z_b = \int f_b(\boldsymbol{s}_b) \prod_{i \in \mathcal{E}(b)} \mu_{ib}(s_i)\mathrm{d}\boldsymbol{s}_b\,.$$

Extracting μ_{jb} from the integral

$$\frac{\mu_{jb}(s_j)\mu_{jc}(s_j)}{Z_j} = \frac{\mu_{jb}(s_j)}{Z_b} \int f_b(\boldsymbol{s}_b) \prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \mu_{ib}(s_i)\,\mathrm{d}\boldsymbol{s}_{b\setminus j}\,,$$

$$\mu_{jc}(s_j) = \frac{Z_j}{Z_b} \int f_b(\boldsymbol{s}_b) \prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \mu_{ib}(s_i)\,\mathrm{d}\boldsymbol{s}_{b\setminus j} \tag{A28}$$

and cancelling μ_{jb} on both sides then yields the condition on the functional form of the message μ_{jc}.

We now need to show that the fixed points of (22) satisfy (A28). Let us assume that the fixed points exist, such that $\mu_{jc}^{(k)} = \mu_{jc}^{(k+1)}$ for some k. Then, we want to show that at the fixed points the following equality holds:

$$\mu_{jc}^{(k)}(s_j) = \frac{Z_j^{(k)}}{Z_b^{(k)}} \int f_b(\boldsymbol{s}_b) \prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \mu_{ib}^{(k)}(s_i)\,\mathrm{d}\boldsymbol{s}_{b\setminus j}\,.$$

Substituting (22), we need to show that

$$\mu_{jc}^{(k)}(s_j) = \frac{Z_j^{(k)}}{Z_b^{(k)}} \mu_{jc}^{(k+1)}(s_j)\,.$$

Since $\mu_{jc}^{(k)} = \mu_{jc}^{(k+1)}$, we can rearrange

$$\mu_{jc}^{(k)}\left(1 - \frac{Z_j^{(k)}}{Z_b^{(k)}}\right) = 0\,.$$

From Z_b, we obtain

$$\begin{aligned}
Z_b^{(k)} &= \int \mu_{jb}^{(k)}(s_j) \int f_b(\boldsymbol{s}_b) \prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \mu_{ib}^{(k)}(s_i)\,\mathrm{d}\boldsymbol{s}_{b\setminus j}\mathrm{d}s_j \\
&= \int \mu_{jb}^{(k)}(s_j)\mu_{jc}^{(k+1)}(s_j)\mathrm{d}s_j \\
&= \int \mu_{jb}^{(k)}(s_j)\mu_{jc}^{(k)}(s_j)\mathrm{d}s_j \\
&= Z_j^{(k)}\,,
\end{aligned}$$

which implies that the fixed points satisfy the desired condition. This proves that the stationary solutions to the BFE within the local polytope can be obtained as fixed points of the sum-product update equations. □

Appendix D.4. Proof of Lemma 3

Proof. Substituting the definition of (32), we can re-write the second term of Lagrangian (30) as

$$\begin{aligned}
\int \Big\{ \prod_{n\in l(b)} q_b^m(\boldsymbol{s}_b^m) \Big\} \log f_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_b &= \int q_b^m(\boldsymbol{s}_b^m) \bigg(\int \Big\{ \prod_{\substack{n\in l(b)\\ n\neq m}} q_b^n(\boldsymbol{s}_b^n) \Big\} \log f_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_b^{\setminus m} \bigg)\,\mathrm{d}\boldsymbol{s}_b^m \\
&= \int q_b^m(\boldsymbol{s}_b^m) \log \tilde{f}_b^m(\boldsymbol{s}_b^m)\,\mathrm{d}\boldsymbol{s}_b^m\,.
\end{aligned}$$

We apply the variation $\epsilon\phi_b^m$ to q_b^m and identify the functional derivative $\delta L_b^m/\delta q_b^m$, as

$$\frac{\mathrm{d}L_b^m[q_b^m + \epsilon\phi_b^m]}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} = \int \Bigg(\overbrace{\log \frac{q_b^m(\boldsymbol{s}_b^m)}{\tilde{f}_b^m(\boldsymbol{s}_b^m)} + 1 + \psi_b^m - \sum_{i\in m} \lambda_{ib}(s_i)}^{\delta L_b^m/\delta q_b^m} \Bigg) \phi_b^m(\boldsymbol{s}_b^m)\, \mathrm{d}\boldsymbol{s}_b^m\,,$$

whose functional form we recognize from Appendix D.1. Setting the functional derivative to zero and again identifying $\mu_{ib}(s_i) = \exp \lambda_{ib}(s_i)$, yields the stationary solutions of (31). □

Appendix D.5. Proof of Theorem 2

Proof. The local polytope of (33) constructs the Lagrangians L_b^m and L_j as (30) and (19), respectively. We substitute the stationary solutions of Lemmas 2 and 3 in the local marginalization constraint (29b), which yields

$$q_j(s_j) = \int q_b^m(\boldsymbol{s}_b^m)\, \mathrm{d}\boldsymbol{s}_{b\setminus j}^m\,.$$

Following the structure of the proof in Appendix D.3, we obtain the following condition for the stationary solutions in terms of messages:

$$\begin{aligned} \frac{\mu_{jb}(s_j)\mu_{jc}(s_j)}{Z_j} &= \frac{\mu_{jb}(s_j)}{Z_b^m} \int \tilde{f}_b^m(\boldsymbol{s}_b^m) \prod_{\substack{i\in m\\ i\neq j}} \mu_{ib}(s_i) \mathrm{d}\boldsymbol{s}_{b\setminus j}^m \\ \frac{\mu_{jc}(s_j)}{Z_j} &= \frac{1}{Z_b^m} \int \tilde{f}_b^m(\boldsymbol{s}_b^m) \prod_{\substack{i\in m\\ i\neq j}} \mu_{ib}(s_i) \mathrm{d}\boldsymbol{s}_{b\setminus j}^m\,. \end{aligned} \tag{A29}$$

Now we want to show that the fixed points of the message updates (36) satisfy (A29). Let us assume that the fixed points exists for some k such that $\mu_{jc}^{(k+1)} = \mu_{jc}^{(k)}$. Then, we will show that the fixed points satisfy

$$\frac{\mu_{jc}^{(k)}(s_j)}{Z_j^{(k)}} = \frac{1}{Z_b^{m,(k)}} \int \tilde{f}_b^{m,(k)}(\boldsymbol{s}_b^m) \prod_{\substack{i\in m\\ i\neq j}} \mu_{ib}^{(k)}(s_i) \mathrm{d}\boldsymbol{s}_{b\setminus j}^m\,. \tag{A30}$$

Similar to Appendix D.3, it will suffice to show that $Z_b^{m,(k)} = Z_j^{(k)}$ at the fixed points. Arranging the order of integration in normalization constant $Z_b^{m,(k)}$, we obtain

$$\begin{aligned} Z_b^{m,(k)} &= \int \mu_{jb}^{(k)}(s_j) \int \tilde{f}_b^{m,(k)}(\boldsymbol{s}_b^m) \prod_{\substack{i\in m\\ i\neq j}} \mu_{ib}^{(k)}(s_i) \mathrm{d}\boldsymbol{s}_{b\setminus j}^m \mathrm{d}s_j \\ &= \int \mu_{jb}^{(k)}(s_j) \mu_{jc}^{(k+1)}(s_j) \mathrm{d}s_j \\ &= \int \mu_{jb}^{(k)}(s_j) \mu_{jc}^{(k)}(s_j) \mathrm{d}s_j \\ &= Z_j^{(k)}\,. \end{aligned}$$

By the same line of reasoning as in Appendix D.3, this shows that the fixed points of the message updates (36) leads to stationary distributions of the Bethe free energy with structured factorization constraints. □

Appendix D.6. Proof of Corollary 1

Proof. For a fully factorized local variational distribution (41), the augmented node function $\tilde{f}_b^m(\mathbf{s}_b^m)$ of (32) reduces to

$$\tilde{f}_j(s_j) = \exp\left(\int \left\{ \prod_{\substack{i \in \mathcal{E}(b) \\ i \neq j}} q_i(s_i) \right\} \log f_b(\mathbf{s}_b)\, \mathrm{d}\mathbf{s}_{b\setminus j}\right). \tag{A31}$$

The message of (36) then reduces to

$$\mu_{jc}(s_j) \;=\; \tilde{f}_j(s_j)\,,$$

which, after substitution, recovers (43). □

Appendix D.7. Proof of Lemma 4

Proof. When we apply the variation $\epsilon\phi_b$ to q_b and identify the functional derivative $\delta L_b/\delta q_b$, we recover the result from Appendix D.1, which leads to a solution of the form (47). □

Appendix D.8. Proof of Theorem 3

Proof. We construct the Lagrangian of (46), which by Lemma 4 leads to a solution of the form (47). Substituting this solution in the constraint of (45) leads to

$$\left[\overbrace{\int f_b(\mathbf{s}_b) \prod_{\substack{i \in \mathcal{E}(b) \\ i \neq j}} \mu_{ib}(s_i)\, \mathrm{d}\mathbf{s}_{b\setminus j}}^{\mu_{bj}(s_j)}\right] \mu_{jb}(s_j) \;=\; \delta(s_j - \hat{s}_j)\,. \tag{A32}$$

This equation is then satisfied by (50), which proves the theorem. □

Appendix D.9. Proof of Lemma 5

Proof. The proof follows directly from Appendix D.1, with $\tilde{f}_b(\mathbf{s}_b; \hat{\mathbf{s}}_b)$ substituted for $f_b(\mathbf{s}_b)$. □

Appendix D.10. Proof of Theorem 4

Proof. Given the result of Lemma 5, the proof follows Appendix D.3, where Laplace propagation chooses the expansion point to be the fixed point $\hat{\mathbf{s}}_b = \arg\max \log q_b(\mathbf{s}_b)$.

For all second-order fixed points of the Laplace iterations, it holds that $\hat{\mathbf{s}}_b$ is a fixed point if and only if it is a local optimum of q_b. The proof is then concluded by Lemma 1 in [76]. □

Appendix D.11. Proof of Lemma 6

Proof. We note that the Lagrange multiplier η_{jb} does not depend on s_j because the expectation removes all the functional dependencies on s_j. Furthermore, the expectations of $T_j(s_j)$ have the same dimension as the function $T_j(s_j)$. This means that the dimension of η_{jb} needs to be compatible with that of $T_j(s_j)$ so that we can write the constraint as an inner product.

We apply the variation $\epsilon\phi_b$ to q_b and identify the functional derivative $\delta L_b/\delta q_b$, as

$$\left.\frac{\mathrm{d}L_b[q_b + \epsilon\phi_b, f_b]}{\mathrm{d}\epsilon}\right|_{\epsilon=0} = \int \left(\overbrace{\log\frac{q_b(\mathbf{s}_b)}{f_b(\mathbf{s}_b)} + 1 + \psi_b - \sum_{\substack{i \in \mathcal{E}(b) \\ i \neq j}} \lambda_{ib}(s_i) - \eta_{jb}^\top T_j(s_j)}^{\delta L_b/\delta q_b}\right) \phi_b(\mathbf{s}_b)\, \mathrm{d}\mathbf{s}_b\,.$$

Setting the functional derivative to zero and identifying $\mu_{ib}(s_i) = \exp \lambda_{ib}(s_i)$ for $i \neq j$ and identifying $\mu_{jb}(s_j) = \exp\Big(\eta_{jb}^\top T_j(s_j)\Big)$ yields the functional form of the stationary solution as (62). $\square$

Appendix D.12. Proof of Lemma 7

Proof. We follow a similar procedure as in Appendix D.11 and apply the variation $\epsilon\phi_j$ to q_j, which identifies the functional derivative $\delta L_j/\delta q_j$, as

$$\frac{\mathrm{d}L[q_j + \epsilon\phi_j]}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} = \int \Bigg(\overbrace{-\log q_j(s_j) - 1 + \psi_j + \sum_{a\in\mathcal{V}(j)} \eta_{ja}^\top T_j(s_j)}^{\delta L_j/\delta q_j}\Bigg)\phi_j(s_j)\,\mathrm{d}s_j\,.$$

Setting the functional derivative to zero and following the same procedure as in Appendix D.2 yields (64). $\square$

Appendix D.13. Proof of Theorem 5

Proof. By substituting the stationary solutions given by Lemmas 6 and 7 into the moment-matching constraint (59), we obtain the following condition:

$$\int T_j(s_j) q_j(s_j)\,\mathrm{d}s_j = \int T_j(s_j) q_b(\boldsymbol{s}_b)\,\mathrm{d}\boldsymbol{s}_b$$

$$\frac{1}{Z_j}\int T_j(s_j)\exp\Big([\eta_{jb} + \eta_{jc}]^\top T_j(s_j)\Big)\mathrm{d}s_j = \frac{1}{\tilde{Z}_j}\int T_j(s_j)\,\overbrace{\exp\Big(\eta_{jb}^\top T_j(s_j)\Big)}^{\mu_{jb}(s_j)}\Bigg[\overbrace{\int f_b(\boldsymbol{s}_b)\prod_{\substack{i\in\mathcal{E}(b)\\ i\neq j}} \mu_{ib}(s_i)\,\mathrm{d}\boldsymbol{s}_{b\setminus j}}^{\tilde{\mu}_{jc}(s_j)}\Bigg]\mathrm{d}s_j$$

$$= \int T_j(s_j)\tilde{q}_j(s_j)\,\mathrm{d}s_j\,,$$

where we recognize the sum-product message $\tilde{\mu}_{jc}(s_j)$, which we multiply by the incoming exponential family message $\mu_{jb}(s_j)$ and normalize to obtain $\tilde{q}_j(s_j)$. By defining $\eta_j = \eta_{jb} + \eta_{jc}$, normalization constants are given by

$$Z_j(\eta_j) = \int \exp\Big(\eta_j^\top T_j(s_j)\Big)\mathrm{d}s_j$$
$$\tilde{Z}_j = \int \exp\Big(\eta_{jb}^\top T_j(s_j)\Big)\tilde{\mu}_{jc}(s_j)\mathrm{d}s_j\,.$$

Computing the moments allows us to determine the exponential family parameter by solving the following equation [24] (Proposition 3.1)

$$\nabla_{\eta_j}\log Z_j(\eta_j) = \int \tilde{q}_j(s_j)\,T_j(s_j)\,\mathrm{d}s_j\,.$$

Suppose you obtain a solution to this equation denoted by $\tilde{\eta}_j$, this allows us to approximate the sum-product message $\tilde{\mu}_{jc}(s_j)$ by an exponential family message whose parameter is given by

$$\eta_{jc} = \tilde{\eta}_j - \eta_{jb}\,.$$

Now let us assume that the fixed points of the sum-product iterations $\tilde{\mu}_{jc}^{(k)}(s_j) = \tilde{\mu}_{jc}^{(k+1)}(s_j)$ and the incoming exponential family messages $\mu_{jb}^{(k)}(s_j) = \mu_{jb}^{(k+1)}(s_j)$ exist for some k. Then, we need to show that the existence of these fixed points implies the existence of the fixed points of $\mu_{jc}^{(k+1)} = \mu_{jc}^{(k)}$.

By moment-matching, we have

$$\begin{aligned}\eta_{jc}^{(k+1)} &= \tilde{\eta}_j^{(k+1)} - \eta_{jb}^{(k+1)} \\ &= \tilde{\eta}_j^{(k)} - \eta_{jb}^{(k)} \\ &= \eta_{jc}^{(k)}\,,\end{aligned}$$

which proves the existence of the fixed point of μ_{jc} if $\tilde{\mu}_{jc}$ and $\mu_{jb}(s_j)$ have fixed points. □

Appendix D.14. Proof of Theorem 6

Proof. The proof follows directly from substituting the Laplace-approximated factor-function (53) in the naive mean-field result of Corollary. 1. □

Appendix D.15. Proof of Theorem 7

Proof. In order to obtain the optimal parameter value θ_j^*, we view the free energy as a function of θ_j. As there are two node-local free energies that depend upon θ_j, this leads to

$$\begin{aligned}\theta_j^* &= \arg\min_{\theta_j} \Big(F[q_b, f_b; \theta_j] + F[q_c, f_c; \theta_j] \Big) \\ &= \arg\max_{\theta_j} \left(\int \Big\{ \delta(s_j - \theta_j) \prod_{\substack{n \in l(b) \\ n \neq m}} q_b^n(\mathbf{s}_b^n) \Big\} \log f_b(\mathbf{s}_b)\, \mathrm{d}\mathbf{s}_b + \int \Big\{ \delta(s_j - \theta_j) \prod_{\substack{n \in l(c) \\ n \neq m}} q_c^n(\mathbf{s}_c^n) \Big\} \log f_c(\mathbf{s}_c)\, \mathrm{d}\mathbf{s}_c \right) \\ &= \arg\max_{\theta_j} \left(\int \Big\{ \prod_{\substack{n \in l(b) \\ n \neq m}} q_b^n(\mathbf{s}_b^n) \Big\} \log f_b(\mathbf{s}_{b\setminus j}, \theta_j)\, \mathrm{d}\mathbf{s}_{b\setminus j} + \int \Big\{ \prod_{\substack{n \in l(c) \\ n \neq m}} q_c^n(\mathbf{s}_c^n) \Big\} \log f_c(\mathbf{s}_{c\setminus j}, \theta_j)\, \mathrm{d}\mathbf{s}_{c\setminus j} \right) \\ &= \arg\max_{s_j} \Big(\log \mu_{bj}(s_j) + \log \mu_{cj}(s_j) \Big)\,,\end{aligned}$$

where in the last step we replaced θ_j with s_j for convenience. Here, we recognize μ_{bj} and μ_{cj} as the structured variational updates of Theorem 2. Identification of the fixed points can then be obtained by [57] (Corollary 2). For a rigorous discussion on convergence of the EM algorithm, we refer to [77] (Corollary 32), [24] (Chapter 6) and [57] (Section 3). □

Appendix D.16. Proof of Theorem 8

Proof. Substituting for $q_a(\mathbf{s}_a)$, the node-local free energy becomes

$$\begin{aligned}F[q_a, f_a] &= \int q_a(\mathbf{s}_a) \log \frac{q_a(\mathbf{s}_a)}{f_a(\mathbf{s}_a)}\, \mathrm{d}\mathbf{s}_a \\ &= \int q_a(\mathbf{s}_a) \log \frac{q_{j|a}(s_j|\mathbf{s}_{a\setminus j})}{f_a(\mathbf{s}_a)}\, \mathrm{d}\mathbf{s}_a + \int q_a(\mathbf{s}_a) \log q_{a\setminus j}(\mathbf{s}_{a\setminus j})\, \mathrm{d}\mathbf{s}_a \\ &= \int q_{a\setminus j}(\mathbf{s}_{a\setminus j}) q_{j|a}(s_j|\mathbf{s}_{a\setminus j}) \log \frac{q_{j|a}(s_j|\mathbf{s}_{a\setminus j})}{f_a(\mathbf{s}_a)}\, \mathrm{d}\mathbf{s}_a + \int q_{a\setminus j}(\mathbf{s}_{a\setminus j}) q_{j|a}(s_j|\mathbf{s}_{a\setminus j}) \log q_{a\setminus j}(\mathbf{s}_{a\setminus j})\, \mathrm{d}\mathbf{s}_a \\ &= \int q_{a\setminus j}(\mathbf{s}_{a\setminus j}) \left[\int q_{j|a}(s_j|\mathbf{s}_{a\setminus j}) \log \frac{q_{j|a}(s_j|\mathbf{s}_{a\setminus j})}{f_a(\mathbf{s}_a)}\, \mathrm{d}s_j \right] \mathrm{d}\mathbf{s}_{a\setminus j} + \int q_{a\setminus j}(\mathbf{s}_{a\setminus j}) \log q_{a\setminus j}(\mathbf{s}_{a\setminus j})\, \mathrm{d}\mathbf{s}_{a\setminus j} \\ &= \mathbb{E}_{q_{a\setminus j}}\Big[D\big[q_{j|a} \| f_a \big] \Big] - H[q_{a\setminus j}]\,,\end{aligned}$$

where the first term expresses an expected Kullback–Leibler divergence, and the second term is a negative entropy. The only possibility for the local free energy to becomes finite, is when $q_{j|a}(s_j|\boldsymbol{s}_{a\setminus j}) = f_a(\boldsymbol{s}_a) = \delta(s_j - g_a(\boldsymbol{s}_{a\setminus j}))$. We then have:

$$F[q_a, f_a] = \begin{cases} -H[q_{a\setminus j}] & \text{if } q_{j|a}(s_j|\boldsymbol{s}_{a\setminus j}) = \delta(s_j - g_a(\boldsymbol{s}_{a\setminus j})) \\ \infty & \text{otherwise.} \end{cases}$$

□

Appendix D.17. Proof of Theorem 9

Proof. The proof is similar to Appendix D.16. Substituting for $q_a(\boldsymbol{s}_a)$, the node-local free energy becomes

$$\begin{aligned} F[q_a, f_a] &= \int q_a(\boldsymbol{s}_a) \log \frac{q_a(\boldsymbol{s}_a)}{f_a(\boldsymbol{s}_a)} \mathrm{d}\boldsymbol{s}_a \\ &= \int q_a(s_i, s_j, s_k) \log \frac{q_{ik|j}(s_i, s_k|s_j)}{f_a(s_i, s_j, s_k)} \mathrm{d}s_i \,\mathrm{d}s_j \,\mathrm{d}s_k + \int q_j(s_j) \log q_j(s_j) \,\mathrm{d}s_j \\ &= \mathbb{E}_{q_j}\Big[D\Big[q_{ik|j} \| f_a\Big]\Big] - H[q_j] \,. \end{aligned}$$

In contrast to Appendix D.16, here we have a joint belief within the divergence with a single conditioning variable. Conditioning on s_j (or by symmetry s_i or s_k) determines the realization of the other variables. Therefore, we have:

$$F[q_a, f_a] = \begin{cases} -H[q_j] & \text{if } q_{ik|j}(s_i, s_k|s_j) = \delta(s_j - s_i)\,\delta(s_j - s_k) \\ \infty & \text{otherwise.} \end{cases}$$

□

References

1. Blei, D.M. Build, Compute, Critique, Repeat: Data Analysis with Latent Variable Models. *Annu. Rev. Stat. Appl.* **2014**, *1*, 203–232. [CrossRef]
2. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006.
3. Kullback, S.; Leibler, R.A. On Information and Sufficiency. *Ann. Math. Stat.* **1951**, *22*, 79–86. [CrossRef]
4. Forney, G. Codes on graphs: Normal realizations. *IEEE Trans. Inf. Theory* **2001**, *47*, 520–548. [CrossRef]
5. Loeliger, H.A. An introduction to factor graphs. *IEEE Signal Process. Mag.* **2004**, *21*, 28–41.
6. Winn, J.; Bishop, C.M. Variational message passing. *J. Mach. Learn. Res.* **2005**, *6*, 661–694.
7. Yedidia, J.S.; Freeman, W.T.; Weiss, Y. *Understanding Belief Propagation and Its Generalizations*; Mitsubishi Electric Research Laboratories, Inc.: Cambridge, MA, USA, 2001.
8. Cox, M.; van de Laar, T.; de Vries, B. A factor graph approach to automated design of Bayesian signal processing algorithms. *Int. J. Approx. Reason.* **2019**, *104*, 185–204. [CrossRef]
9. Yedidia, J.S. An Idiosyncratic Journey beyond Mean Field Theory. In *Advanced Mean Field Methods*; The MIT Press: Cambridge, MA, USA, 2000; pp. 37–49.
10. Yedidia, J.S.; Freeman, W.T.; Weiss, Y. *Bethe Free Energy, Kikuchi Approximations, and Belief Propagation Algorithms*; Mitsubishi Electric Research Laboratories, Inc.: Cambridge, MA, USA, 2001 ; p. 24.
11. Dauwels, J. On Variational Message Passing on Factor Graphs. In Proceedings of the IEEE International Symposium on Information Theory, Nice, France, 24–29 June 2007; pp. 2546–2550. [CrossRef]
12. Zhang, D.; Wang, W.; Fettweis, G.; Gao, X. Unifying Message Passing Algorithms under the Framework of Constrained Bethe Free Energy Minimization. *arXiv* **2017**, arXiv:1703.10932.
13. van de Laar, T.; Şenöz, I.; Özçelikkale, A.; Wymeersch, H. Chance-Constrained Active Inference. *arXiv* **2021**, arXiv:2102.08792.
14. Smola, A.J.; Vishwanathan, S.V.N.; Eskin, E. Laplace propagation. In *NIPS*; The MIT Press: Cambridge, MA, USA, 2004; pp. 441–448.
15. Minka, T. *Divergence Measures and Message Passing*. Available online: https://www.microsoft.com/en-us/research/publication/divergence-measures-and-message-passing/ (accessed on 24 June 2021).
16. Yedidia, J.S. Generalized Belief Propagation and Free Energy Minimization. Available online: http://cba.mit.edu/events/03.11.ASE/docs/Yedidia.pdf (accessed on 24 June 2021).

17. Yedidia, J.S.; Freeman, W.; Weiss, Y. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Inf. Theory* **2005**, *51*, 2282–2312. [CrossRef]
18. Minka, T.P. Expectation Propagation for Approximate Bayesian Inference. In Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence, Seattle, WA, USA, 2–5 August 2001 ; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2001; pp. 362–369.
19. Heskes, T. Stable fixed points of loopy belief propagation are local minima of the bethe free energy. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2003; pp. 359–366.
20. Kschischang, F.R.; Frey, B.J.; Loeliger, H.A. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory* **2001**, *47*, 498–519.
21. Hoffman, M.; Blei, D.M.; Wang, C.; Paisley, J. Stochastic Variational Inference. *arXiv* **2012**, arXiv:1206.7051.
22. Archer, E.; Park, I.M.; Buesing, L.; Cunningham, J.; Paninski, L. Black box variational inference for state space models. *arXiv* **2015**, arXiv:1511.07367.
23. Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1988.
24. Wainwright, M.J.; Jordan, M.I. Graphical Models, Exponential Families, and Variational Inference. *Found. Trends® Mach. Learn.* **2008**, *1*, 1–305. [CrossRef]
25. Chertkov, M.; Chernyak, V.Y. Loop Calculus in Statistical Physics and Information Science. *Phys. Rev. E* **2006**, *73* . [CrossRef]
26. Weller, A.; Tang, K.; Jebara, T.; Sontag, D.A. Understanding the Bethe approximation: When and how can it go wrong? In Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence, Quebec City, QC, Canada, 23–27 July 2014; pp. 868–877.
27. Sibel, J.C. Region-Based Approximation to Solve Inference in Loopy Factor Graphs: Decoding LDPC Codes by Generalized Belief Propagation. Available online: https://hal.archives-ouvertes.fr/tel-00905668 (accessed on 24 June 2021).
28. Minka, T. *From Hidden Markov Models to Linear Dynamical Systems*; Technical Report 531; VIsion and Modeling Group, Media Lab, MIT: Cambridge, MA, USA, 1999.
29. Loeliger, H.A.; Dauwels, J.; Hu, J.; Korl, S.; Ping, L.; Kschischang, F.R. The Factor Graph Approach to Model-Based Signal Processing. *Proc. IEEE* **2007**, *95*, 1295–1322. [CrossRef]
30. Loeliger, H.A.; Bolliger, L.; Reller, C.; Korl, S. Localizing, forgetting, and likelihood filtering in state-space models. In Proceedings of the 2009 Information Theory and Applications Workshop, La Jolla, CA, USA, 8–13 February 2009; pp. 184–186. [CrossRef]
31. Korl, S. A Factor Graph Approach to Signal Modelling, System Identification and Filtering. Ph.D. Thesis, Swiss Federal Institute of Technology, Zurich, Switzerland, 2005.
32. Pearl, J. Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach. In Proceedings of the Second AAAI Conference on Artificial Intelligence, Pittsburgh, PA, USA, 18–20 August 1982; AAAI Press: Pittsburgh, PA, USA, 1982; pp. 133–136.
33. Heskes, T. Convexity arguments for efficient minimization of the Bethe and Kikuchi free energies. *J. Artif. Intell. Res.* **2006**, *26*, 153–190.
34. Särkkä, S. *Bayesian Filtering and Smoothing*; Cambridge University Press: London, UK; New York, NY, USA, 2013.
35. Khan, M.E.; Lin, W. Conjugate-Computation Variational Inference: Converting Variational Inference in Non-Conjugate Models to Inferences in Conjugate Models. *arXiv* **2017**, arXiv:1703.04265.
36. Logan, B.; Moreno, P. Factorial HMMs for acoustic modeling. In Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, Seattle, WA, USA, 15 May 1998; Volume 2, pp. 813–816. [CrossRef]
37. Hoffman, M.D.; Blei, D.M. Structured Stochastic Variational Inference. *arXiv* **2014**, arXiv:1404.4114.
38. Singh, R.; Ling, J.; Doshi-Velez, F. Structured Variational Autoencoders for the Beta-Bernoulli Process. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; p. 9.
39. Bamler, R.; Mandt, S. Structured Black Box Variational Inference for Latent Time Series Models. *arXiv* **2017**, arXiv:1707.01069.
40. Zhang, C.; Yuan, Z.; Wang, Z.; Guo, Q. Low Complexity Sparse Bayesian Learning Using Combined BP and MF with a Stretched Factor Graph. *Signal Process.* **2017**, *131*, 344–349. [CrossRef]
41. Wand, M.P. Fast Approximate Inference for Arbitrarily Large Semiparametric Regression Models via Message Passing. *J. Am. Stat. Assoc.* **2017**, *112*, 137–168. [CrossRef]
42. Caticha, A. Entropic Inference and the Foundations of Physics. In Proceedings of the 11th Brazilian Meeting on Bayesian Statistics, Amparo, Brazil, 18–22 March 2012.
43. Pearl, J. A Probabilistic Calculus of Actions. Available online: https://arxiv.org/ftp/arxiv/papers/1302/1302.6835.pdf (accessed on 24 June 2021).
44. Zoeter, O.; Heskes, T. Gaussian Quadrature Based Expectation Propagation. In Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, Bridgetown, Barbados, 6–8 January 2005; p. 9.
45. Arasaratnam, I.; Haykin, S. Cubature Kalman Filters. *IEEE Trans. Autom. Control* **2009**, *54*, 1254–1269. [CrossRef]
46. Sarkka, S. Bayesian Estimation of Time-Varying Systems: Discrete-Time Systems. Available online: https://users.aalto.fi/~ssarkka/course_k2011/pdf/course_booklet_2011.pdf (accessed on 24 June 2021).
47. Gelman, A.; Vehtari, A.; Jylänki, P.; Robert, C.; Chopin, N.; Cunningham, J.P. Expectation propagation as a way of life. *arXiv* **2014**, arXiv:1412.4869.

48. Deisenroth, M.P.; Mohamed, S. Expectation Propagation in Gaussian Process Dynamical Systems: Extended Version. *arXiv* **2012**, arXiv:1207.2940.
49. Teh, Y.W.; Hasenclever, L.; Lienart, T.; Vollmer, S.; Webb, S.; Lakshminarayanan, B.; Blundell, C. Distributed Bayesian Learning with Stochastic Natural-gradient Expectation Propagation and the Posterior Server. *arXiv* **2015**, arXiv:1512.09327.
50. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006.
51. Cox, M. Robust Expectation Propagation in Factor Graphs Involving Both Continuous and Binary Variables. In Proceedings of the 26th European Signal Processing Conference (EUSIPCO), Rome, Italy, 3–7 September 2018; p. 5.
52. Minka, T.; Winn, J.; Guiver, J.; Webster, S.; Zaykov, Y.; Yangel, B.; Spengler, A.; Bronskill, J. Infer.NET 2.6. 2014. Available online: http://research.microsoft.com/infernet (accessed on 23 June 2021).
53. Friston, K.J.; Harrison, L.; Penny, W. Dynamic causal modelling. *Neuroimage* **2003**, *19*, 1273–1302.
54. Mathys, C.D.; Daunizeau, J.; Friston, K.J.; Klaas, S.E. A Bayesian foundation for individual learning under uncertainty. *Front. Hum. Neurosci.* **2011**, *5*. [CrossRef]
55. Friston, K.; Kilner, J.; Harrison, L. A free energy principle for the brain. *J. Physiol.* **2006**, *100*, 70–87. [CrossRef]
56. Friston, K. The free-energy principle: A rough guide to the brain? *Trends Cogn. Sci.* **2009**, *13*, 293–301. [CrossRef]
57. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. R. Stat. Soc. Ser. B Methodol.* **1977**, *39*, 1–38.
58. Dauwels, J.; Eckford, A.; Korl, S.; Loeliger, H.A. Expectation maximization as message passing—Part I: Principles and gaussian messages. *arXiv* **2009**, arXiv:0910.2832.
59. Bouvrie, P.; Angulo, J.; Dehesa, J. Entropy and complexity analysis of Dirac-delta-like quantum potentials. *Phys. A Stat. Mech. Appl.* **2011**, *390*, 2215–2228. [CrossRef]
60. Dauwels, J.; Korl, S.; Loeliger, H.A. Expectation maximization as message passing. In Proceedings of the International Symposium on Information Theory 2005, (ISIT 2005), Adelaide, Australia, 4–9 September 2005; pp. 583–586. [CrossRef]
61. Cox, M.; van de Laar, T.; de Vries, B. ForneyLab.jl: Fast and flexible automated inference through message passing in Julia. In Proceedings of the International Conference on Probabilistic Programming, Boston, MA, USA, 4–6 October 2018.
62. Bezanson, J.; Edelman, A.; Karpinski, S.; Shah, V. Julia: A Fresh Approach to Numerical Computing. *SIAM Rev.* **2017**, *59*, 65–98. [CrossRef]
63. Şenöz, I.; de Vries, B. Online Variational Message Passing in the Hierarchical Gaussian Filter. In Proceedings of the 2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP), Aalborg, Denmark, 17–20 September 2018; pp. 1–6. [CrossRef]
64. Mathys, C.D. *Uncertainty, Precision, and Prediction Errors*; UCL Computational Psychiatry Course; UCL: London, UK, 2014.
65. Şenöz, I.; de Vries, B. Online Message Passing-based Inference in the Hierarchical Gaussian Filter. In Proceedings of the 2020 IEEE International Symposium on Information Theory (ISIT), Los Angeles, CA, USA, 21–26 June 2020; pp. 2676–2681. [CrossRef]
66. Podusenko, A.; Kouw, W.M.; de Vries, B. Online Variational Message Passing in Hierarchical Autoregressive Models. In Proceedings of the 2020 IEEE International Symposium on Information Theory (ISIT), Los Angeles, CA, USA, 21–26 June 2020; pp. 1337–1342. [CrossRef]
67. Welling, M. On the Choice of Regions for Generalized Belief Propagation. *arXiv* **2012**, arXiv:1207.4158.
68. Welling, M.; Minka, T.P.; Teh, Y.W. Structured Region Graphs: Morphing EP into GBP. *arXiv* **2012**, arXiv:1207.1426.
69. Loeliger, H.A. Factor Graphs and Message Passing Algorithms—Part 1: Introduction, 2007. Available online: http://www.crm.sns.it/media/course/1524/Loeliger_A.pdf (accessed on 3 April 2019).
70. Caticha, A. Relative Entropy and Inductive Inference. *AIP Conf. Proc.* **2004**, *707*, 75–96. [CrossRef]
71. Ortega, P.A.; Braun, D.A. A Minimum Relative Entropy Principle for Learning and Acting. *J. Artif. Intell. Res.* **2010**, *38*, 475–511.
72. Shore, J.; Johnson, R. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Trans. Inf. Theory* **1980**, *26*, 26–37. [CrossRef]
73. Engel, E.; Dreizler, R.M. *Density Functional Theory: An Advanced Course*; Theoretical and Mathematical Physics; Springer: Berlin/Heidelberg, Germany, 2011. [CrossRef]
74. Boyd, S.P.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK; New York, NY, USA, 2004.
75. Lanczos, C. *The Variational Principles of Mechanics*; Courier Corporation: North Chelmsford, MA, USA, 2012.
76. Ahn, S.; Chertkov, M.; Shin, J. Gauging Variational Inference. Available online: https://dl.acm.org/doi/10.5555/3294996.3295048 (accessed on 24 June 2021).
77. Tran, V.H. Copula Variational Bayes inference via information geometry. *arXiv* **2018**, arXiv:1803.10998.

entropy

MDPI

Article

Understanding the Variability in Graph Data Sets through Statistical Modeling on the Stiefel Manifold

Clément Mantoux [1,2,3,*], Baptiste Couvy-Duchesne [1,2], Federica Cacciamani [1,2], Stéphane Epelbaum [1,2,4], Stanley Durrleman [1,2] and Stéphanie Allassonnière [5,6]

1 ARAMIS Project Team, Inria, 75013 Paris, France; baptiste.couvy@icm-institute.org (B.-C.D.); federica.cacciamani@icm-institute.org (F.C.); stephane.epelbaum@icm-institute.org (S.E.); stanley.durrleman@inria.fr (S.D.)
2 ARAMIS Lab, Brain and Spine Institute, ICM, INSERM UMR 1127, CNRS UMR 7225, Sorbonne University, Hôpital de la Pitié-Salpêtrière, 75013 Paris, France
3 CMAP, École Polytechnique, 91120 Palaiseau, France
4 Institute of Memory and Alzheimer's Disease (IM2A), Centre of Excellence of Neurodegenerative Disease (CoEN), CIC Neurosciences, AP-HP, Department of Neurology, Hôpital de la Pitié-Salpêtrière, 75013 Paris, France
5 Centre de Recherche des Cordeliers, Université de Paris, INSERM UMR 1138, Sorbonne Université, 75006 Paris, France; stephanie.allassonniere@parisdescartes.fr
6 HEKA Project Team, Inria, 75006 Paris, France
* Correspondence: clement.mantoux@inria.fr

Citation: Mantoux, C.; Couvy-Duchesne, B.; Cacciamani, F.; Epelbaum, S.; Durrleman, S.; Allassonnière, S. Understanding the Variability in Graph Data Sets through Statistical Modeling on the Stiefel Manifold. *Entropy* **2021**, *23*, 490. https://doi.org/10.3390/e23040490

Academic Editor: Pierre Alquier

Received: 13 March 2021
Accepted: 14 April 2021
Published: 20 April 2021

Abstract: Network analysis provides a rich framework to model complex phenomena, such as human brain connectivity. It has proven efficient to understand their natural properties and design predictive models. In this paper, we study the variability within groups of networks, i.e., the structure of connection similarities and differences across a set of networks. We propose a statistical framework to model these variations based on manifold-valued latent factors. Each network adjacency matrix is decomposed as a weighted sum of matrix patterns with rank one. Each pattern is described as a random perturbation of a dictionary element. As a hierarchical statistical model, it enables the analysis of heterogeneous populations of adjacency matrices using mixtures. Our framework can also be used to infer the weight of missing edges. We estimate the parameters of the model using an Expectation-Maximization-based algorithm. Experimenting on synthetic data, we show that the algorithm is able to accurately estimate the latent structure in both low and high dimensions. We apply our model on a large data set of functional brain connectivity matrices from the UK Biobank. Our results suggest that the proposed model accurately describes the complex variability in the data set with a small number of degrees of freedom.

Keywords: network modeling; network variability; Stiefel manifold; MCMC-SAEM; data imputation

1. Introduction

Network science is at the core of an ever-growing range of applications. Network analysis [1] aims at studying the natural properties of complex systems of interacting components or individuals through their connections. It provides a large number of tools to detect communities [2], predict unknown connections [3] and covariates [4], measure population characteristics [5,6] or build unsupervised low-dimensional representations [7]. The need to understand and model networks arises in multiple fields, such as social networks analysis [8], recommender systems [9], gene interactions networks [10], neuroscience [11] or chemistry [12]. Network analysis allows accounting for very diverse phenomenons in similar mathematical frameworks, which lend themselves to theoretical and statistical analysis [13]. In this paper, we are interested in groups of undirected networks that are defined on the same set of nodes. This situation describes the longitudinal evolution of a given network throughout time or the case where the nodes define a standard structure identical across the networks. The former is of interest in computational social science, which studies the evolution of interactions within a fixed population [14]. The latter arises naturally in neuroscience, where the connections between well-defined brain regions are studied on large groups of subjects. The analysis of brain networks is the main application of the present study. It has proven an efficient tool to discover new aspects of the anatomy and function of the human brain [15] and remains a very active research topic [16].

In this study, we are interested in the variability of undirected graph data sets, i.e., how graphs defined on a common set of nodes vary from one network to another. Accounting for this variability is a crucial issue in neuroscience: predicting neurodegenerative diseases or understanding the complex mechanisms of aging requires robust, coherent statistical frameworks that model the diversity among a population. Working on such graphs sharing the same nodes allows comparing them to one another through their adjacency matrices.

The comparison and statistical modeling of such matrices are difficult problems. If all the graphs have n nodes, a Gaussian model on the $n \times n$ adjacency matrices has a covariance matrix with n^4 coefficients, which is hard to interpret and difficult to estimate from a reasonable number of observations. Considering adjacency matrices as large vectors allows using classical statistical methods, such as Principal Component Analysis (PCA), but does not take advantage of the strong structures underlying the interactions between the nodes. Tailored kernel methods can be employed to evaluate distances between networks, but many theoretically interesting graph kernels require solving NP-hard problems [17]. In the field of brain network analysis, graphs are often modeled and summarized by features like the average shortest path length, which only partially characterize their structure [6]. Recent methods relying on graphs neural networks often consider the nodes of the network to be permutation invariant, whereas nodes in brain networks play a specific role likely to remain stable across subjects [15,18].

In this paper, we propose a generative statistical model to express the variability in undirected graph data sets. We decompose the network adjacency matrices as a weighted sum of orthonormal matrix patterns with rank one. The patterns and their weights vary around their mean values. Using rank-one patterns allows understanding each decomposition term, while using only a small number of parameters. This is comparable to PCA where each observation is decomposed onto orthogonal elements, which in this case would be matrices. The orthogonal patterns are seen as elements of the Stiefel manifold of rectangular matrices X such that $X^\top X$ is the identity matrix [19]. This model allows us to use known distributions and perform a statistical estimation of the mean patterns and weights. We use a restricted number of patterns to get a robust model, which captures the main structures and their variations. This low-dimensional parametric representation provides a simple interpretation of the structure and the variability of the distribution. Our model accounts for two sources of variability: the perturbations of the patterns and their weight. In contrast, current approaches in the literature only consider one of them, as with dictionary-based models and graph auto-encoders.

The proposed framework is expressed as a generative statistical model so that it can easily be generalized to analyze heterogeneous populations. This corresponds to a mixture of several copies of the former model where each cluster has its own center and variance parameters.

In Section 2, we recall relevant literature references for network modeling and statistics on the Stiefel manifold. Section 3 defines our model and further motivates its structure. Section 4 proposes an algorithm based on Expectation-Maximization (EM) to perform Maximum Likelihood Estimation of the model parameters. In Section 5, we present numerical experiments on synthetic and real data. We use our model to predict missing links using the parameters given by the algorithm. We show how our model can be used to perform clustering on network data sets, allowing to distinguish different modes of variability better than a classical clustering algorithm. Applying our method to the UK Biobank collection of brain functional connectivity networks, we demonstrate that our model is able to capture a complex variability with a limited number of parameters. Note that the tools we present here could also be used on any type of network, such as the ones we mentioned above or gene interaction networks.

2. Background

2.1. Statistical Modeling for Graphs Data Sets

The analysis of graph data sets is a wide area of research that overlaps with many application domains. In this section, we review the principal trends of this field that are used in statistics and machine learning.

The first category of statistical models characterizes graphs in a data set (with possibly varying number of nodes) by a set of features that can be compared across networks, rather than matching the nodes of one graph to those of another. These features can be, for example, the average shortest path length, the clustering coefficient, or the occurrence number of certain patterns. Two examples of such models are Exponential Random Graphs Models [20] and graph kernel methods [17]. Other models are defined by a simple, interpretable generative procedure that allows testing hypotheses on complex networks. The Erdős–Rényi model [21] assumes that each node has an equal probability of connecting with one another. The Stochastic Block Model (SBM, [22]) extends this model and introduces communities organized in distinct clusters with simple interactions. In the limit of a large number of nodes, the same idea gives rise to the graphon model, which has also recently been used to model graph data sets [23]. Finally, recent machine learning models leverage the power of graph neural networks [24] to perform classification or regression tasks. They are used, for instance, in brain network analysis to predict whether a patient is affected by Alzheimer's disease or how the disease will evolve [25,26].

In this paper, we consider undirected graphs on a fixed given set of n nodes connected by weighted or binary edges. This situation arises when studying the evolution of a given network across time [27] or when considering several subjects whose networks have the same structure, for instance, brain networks and protein or gene interaction networks. This constraint allows building models based on the ideas of mean and covariance of adjacency matrices, otherwise ill-defined when the nodes change across networks. In particular, little work has been done in the literature so far on the analysis of the variability of graphs in a data set sharing a common set of nodes. Dictionary-based graph analysis models [28] and graph auto-encoders [25,29] are interesting frameworks in that regard. They allow concisely representing a network in a form that compresses the $O(n^2)$ adjacency matrix representation into a smaller space of dimension $O(p)$ or $O(np)$ (where p is the encoding dimension that characterizes the model). However, they each focus on one aspect of the variability of graph data sets, either the variations of patterns for graph auto-encoders or the variations of patterns weights for dictionary-based models. The model proposed in Section 3 builds on these ideas and accounts for both sources of variability in two latent variables that are combined to obtain the adjacency matrices. These variables are the dominant eigenvalues and the related eigenvectors.

These eigenvectors are regrouped in matrices with orthonormal columns, which makes them points on the Stiefel manifold introduced in the next section. Statistical modeling of these matrices requires taking their geometry into account with manifold-valued distributions.

2.2. Models and Algorithms on the Stiefel Manifold

2.2.1. Compact Stiefel Manifolds of Orthonormal Frames

In this paper, we will be considering latent variables belonging to the compact Stiefel manifold $\mathcal{V}_{n,p}$, defined as the set of n-dimensional orthonormal p-frames (with $p \leq n$): $\mathcal{V}_{n,p} = \{X \in \mathbb{R}^{n\times p} \mid X^\top X = I_p\}$. Since an element of $\mathcal{V}_{n,p}$ can be obtained by taking the p first columns of an orthogonal matrix, the Stiefel manifold can be seen as a quotient manifold from the orthogonal group, and thus inherits a canonical Riemannian manifold structure. A detailed and clear introduction to algorithms for optimization and geodesic path computation on the Stiefel Manifold can be found in [30]. More recently, Zimmermann [31] proposed an algorithm to compute the Riemannian logarithm associated with the canonical metric, solving the inverse problem of the geodesic computation.

2.2.2. Von Mises–Fisher Distributions

Various difficulties arise when dealing with statistical distributions on Riemannian manifolds: for instance, computing the barycenter of a set of points can be a difficult problem, if not even ill-posed. The normalizing constant of a distribution is often impossible to compute analytically from its non-normalized density, so Maximum Likelihood Estimation cannot be performed by standard optimization.

Luckily, tractable distributions on the Stiefel manifolds circumventing some of these problems have been brought up and studied over the last decades in the research field of directional statistics. The most well-studied of them is the von Mises–Fisher (vMF) distribution (also called the Matrix Langevin distribution in some papers) first introduced in [32], which is the one we will be using in this paper. Given a matrix-valued parameter $F \in \mathbb{R}^{n\times p}$, the von Mises–Fisher distribution on the Stiefel Manifold is defined by its density: $p_{\mathrm{vMF}}(X) \propto \exp(\mathrm{Tr}(F^\top X))$. Written differently, if we denote by $f_1, ..., f_p$ the columns of F and by $x_1, ..., x_p$ those of X, we have

$$p_{\mathrm{vMF}}(X) \propto \exp(\langle f_1, x_1\rangle + ... + \langle f_p, x_p\rangle)\,.$$

In this expression, each x_i is drawn toward $f_i/|f_i|$ (up to the orthogonality constraint). The norm $|f_i|$ can be interpreted as a concentration parameter that determines the strength of the attraction toward $f_i/|f_i|$. The von Mises–Fisher distribution can be considered analogous to a Euclidean Gaussian distribution with a diagonal covariance matrix: the density imposes no interaction between the components of X, so that the only dependency between the columns is the orthogonality constraint. The equivalent of the Gaussian mode (which is the same as the Gaussian mean) is given by the following lemma:

Lemma 1. *The von Mises–Fisher distribution with parameter F reaches its maximum density value at $X = \pi_V(F)$, where π_V is an orthogonal projection onto the Stiefel manifold.*

Proof. From the definition of the von Mises–Fisher density, we have:

$$\begin{aligned}\mathrm{argmax}_{X^\top X = I_p}\, \mathrm{Tr}(F^\top X) &= \mathrm{argmax}_{X^\top X=I_p} -\frac{1}{2}\mathrm{Tr}(F^\top F) + \mathrm{Tr}(F^\top X) - \frac{1}{2}\mathrm{Tr}(X^\top X)\\ &= \mathrm{argmin}_{X^\top X=I_p} \frac{1}{2}\|F - X\|^2\,,\end{aligned}$$

with $\|\cdot\|$ the Frobenius norm. Hence, by definition, $\pi_V(F)$ maximizes the von Mises–Fisher density. Note that the projection onto the Stiefel manifold is not uniquely defined, as $\mathcal{V}_{n,p}$ is not convex. □

The following lemma allows us to compute such a projection.

Lemma 2. *Let $M \in \mathbb{R}^{n\times p}$, and $M = UDV^\top$ ($U \in \mathbb{R}^{n\times p}, D \in \mathbb{R}^{p\times p}, V \in \mathbb{R}^{p\times p}$) the Singular Value Decomposition of M. If M has full rank, then $UV^\top$ is the unique projection of M onto the Stiefel manifold $\mathcal{V}_{n,p}$.*

Proof. Let us consider the Lagrangian related to the constrained optimization problem $\pi_V(M) \in \mathrm{argmin}_{X^\top X=I_p} \frac{1}{2}\|M - X\|^2$:

$$\mathcal{L}(X, \Lambda) = \frac{1}{2}\|M - X\|^2 - \mathrm{Tr}(\Lambda^\top(I_p - X^\top X))\,.$$

Then the Karush–Kuhn–Tucker theorem [33] shows that, if X^* is a local extremum of $X \mapsto \frac{1}{2}\|X - M\|^2$ over $\mathcal{V}_{n,p}$, then there exists Λ^* such that $\nabla_X \mathcal{L}(X^*, \Lambda^*) = 0$. This gradient writes:

$$\begin{aligned}\nabla_X \mathcal{L}(X^*, \Lambda^*) &= X^* - M + X^*(\Lambda^* + \Lambda^{*\top}) \\ &= X^*(I + \Lambda^* + \Lambda^{*\top}) - M = 0\,.\end{aligned}$$

Since $X \in \mathcal{V}_{n,p}$ and M has full rank, the symmetric matrix $\Omega = I + \Lambda^* + \Lambda^{*\top}$ must be invertible, so that $X^* = M\Omega^{-1}$. Hence

$$I_p = X^{*\top} X^* = \Omega^{-1} M^\top M \Omega^{-1} \iff \Omega^2 = M^\top M = V D^2 V^\top\,.$$

The matrix square roots of $M^\top M$ are exactly given by the Ω's of the form $VRV^\top$, with $R = \mathrm{Diag}(\pm D_{11}, \ldots, \pm D_{pp})$. We get $X^* = M\Omega^{-1} = UDR^{-1}V^\top$, which gives the following objective function:

$$\|M - X^*\|^2 = \left\|U(D - DR^{-1})V^\top\right\|^2 = \left\|D - DR^{-1}\right\|^2.$$

As D has a positive diagonal, this function is globally minimized by $R = D$, so that the unique projection is $X^* = UV^\top$. □

The simple, interpretable density of the von Mises–Fisher distribution comes with several important advantages. First, it allows using classical Markov Chain Monte Carlo (MCMC) methods to sample efficiently from the distribution (see Figure 1 for examples of distributions over $\mathcal{V}_{3,2}$). Next, the form of the density makes it a member of the exponential family, which is a key requirement to perform latent variable inference with the MCMC-Stochastic Approximation Expectation-Maximization algorithm (MCMC-SAEM, [34]) used in this paper. Finally, reasonably efficient algorithms exist to perform Maximum Likelihood Estimation (MLE) of the parameter F. This point will be further developed in Section 4.

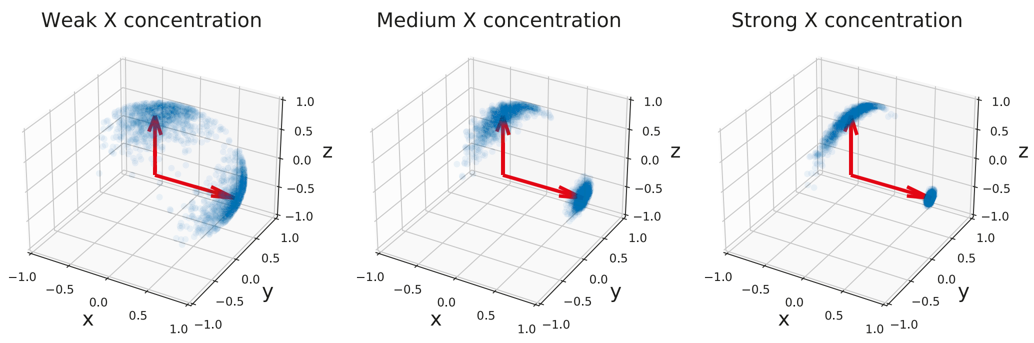


Figure 1. One thousand samples of three von Mises–Fisher distributions on $\mathcal{V}_{3,2}$. The mode of the distribution is represented by two red arrows along the x and z axes, and the two vectors in each matrix by two blue points. The concentration parameters are set to $|f_z| = 10$ and $|f_x| \in [10, 100, 500]$ (from **left** to **right**). Samples are drawn with an adaptive Metropolis–Hastings sampler using the transition kernel described in Section 4. A stronger concentration of the x vector impacts the spread of the z vector.

2.2.3. Application to Network Modeling

Statistical modeling on the Stiefel manifold has proven relevant to analyze networks. By considering the matrix of the p eigenvectors associated with the largest eigenvalues of an adjacency matrix as an element of $\mathcal{V}_{n,p}$, Hoff and colleagues [35–38] showed that probabilistic modeling of the eigenvector matrix on the Stiefel manifold provides a robust representation while allowing to quantify the uncertainty of each edge and estimate the probability of missing links. In these papers, the eigenvectors follow a uniform prior distribution. In the present study, we propose to model the eigenvectors of several networks as samples of a common distribution on $\mathcal{V}_{n,p}$ concentrated around a mode.

3. A Latent Variable Model for Graph Data Sets

3.1. Motivation

We model graphs in a data set by studying the eigendecomposition of their adjacency matrices. Given such a symmetric weighted adjacency matrix $A \in \mathbb{R}^{n \times n}$, the spectral theorem grants the existence of a unique decomposition $A = X\Lambda X^\top = \sum_{i=1}^{r} \lambda_i x_i x_i^\top$, where r is the rank of A, and $\lambda_1 \geq ... \geq \lambda_r$ and $x_1, ..., x_r$ are the eigenvalues and the orthonormal eigenvectors of the matrix. This decomposition is unique up to the sign of the eigenvectors, as long as the non-zero eigenvalues values have multiplicity-one, which always holds in practice. The interest of this decomposition for graph adjacency matrices is threefold.

First, the eigendecomposition of the adjacency matrix reflects the modularity of a network, i.e., the extent to which its nodes can be divided into separate communities. For instance, in the case of the Stochastic Block Model (SBM), each node i is randomly assigned to one cluster $c(i)$ among p possible ones. Nodes in clusters c, c' are connected independently with probability $P_{cc'}$. In expectation, the adjacency matrix is equal to the matrix $(P_{c(i)c(j)})$, which has the rank of p at most. In samples of the SBM as well as real modular networks, the decay of the eigenvalues allows estimating the number of clusters. The eigenvectors related to non-zero eigenvalues are used to perform clustering on the nodes to retrieve their labels.

Furthermore, this decomposition provides a natural expression of A as a sum of rank-one patterns $x_i x_i^\top$. Modeling vectors as a weighted sum of patterns is at the core of dictionary learning-based and mixed effects models, which have proven of great interest to the statistics and machine learning research communities. In the specific case of graph data sets, such a model was recently proposed by D'Souza et al. [28] in the context of brain networks analysis. The authors learn a set of rank-one patterns without orthogonality constraints, and estimate the adjacency matrices as weighted sums of these patterns, in order to use the weights as regression variables. However, they consider the patterns as population-level variables only. This choice prevents taking into account potential individual-level variations.

Finally, the dominant eigenvectors yield strong patterns that are likely to remain stable among various networks in a data set, up to a certain variability. In other words, given N adjacency matrices $A^{(1)}, ..., A^{(N)}$ and their eigendecompositions $(X^{(1)}, \Lambda^{(1)})$, ..., $(X^{(N)}, \Lambda^{(N)})$, the first columns of the $X^{(k)}$'s should remain stable among subjects (up to a column permutation and/or change of sign). On the contrary, smaller eigenvalues should be expected to correspond to eigenvectors with greater variability. The recent work of Chen et al. [39] takes stock of this remark to analyze the Laplacian matrices of brain networks (the Laplacian is a positive matrix that can be computed from the adjacency matrix). The authors propose to compute the L^1 mean of the $X^{(k)}$'s first p columns in order to get a robust average X representative of the population. As the $X^{(k)}$'s are composed of p orthonormal vectors, their average should have the same property: it ensures that the obtained matrix can be interpreted as a point that best represents the distribution. Its definition thus formulates as an optimization problem over the Stiefel manifold $\mathcal{V}_{n,p}$. The authors show that taking this geometric consideration into account leads to better results than computing a Euclidean mean.

In the next section, we introduce our statistical analysis framework. We model the perturbations of the adjacency matrix eigendecomposition to account for the variability within a network data set.

3.2. Model Description

We propose to account for the variability in a set of networks by considering the random perturbation of both the patterns (X variable) that compose the networks and their weight (λ variable). In this study, we consider each pattern x_i (column of X) and each weight λ_i to be independent of one another. This assumption, although a first approximation, leads to a tractable inference problem and interpretable results. Future works could consider interactions between the x_i's or the λ_i's, as well as the dependency between both.

The model decomposition of each adjacency matrix $A^{(k)}$ in a data set writes

$$A^{(k)} = X^{(k)}\mathrm{Diag}(\lambda^{(k)})X^{(k)\top} + \varepsilon^{(k)} \tag{1}$$

with $X^{(k)}$ a pattern matrix, $\lambda^{(k)}$ the pattern weight vector and $\varepsilon^{(k)}$ the symmetric residual noise. The $X^{(k)}$ and $\lambda^{(k)}$ are independent unobserved variables that determine the individual-level specificity of network k. We model these variables as follows:

$$\begin{cases} X^{(k)} \overset{\text{i.i.d}}{\sim} \mathrm{vMF}(F) \\ \lambda^{(k)} \overset{\text{i.i.d}}{\sim} \mathcal{N}(\mu, \sigma_\lambda^2 I_p) \\ \varepsilon^{(k)} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma_\varepsilon^2 I_{n(n+1)/2}). \end{cases} \tag{2}$$

The matrix $F \in \mathbb{R}^{n\times p}$ parametrizes a von Mises–Fisher distribution for the eigenvectors matrix $X^{(k)}$, and the eigenvalues $\lambda^{(k)}$ follow a Gaussian distribution with mean $\mu \in \mathbb{R}^p$ and independent components with variance σ_λ^2. We further impose that the columns of F are orthogonal: this constraint ensures that the maximum of the log-density $\langle f_1, x_1\rangle + ... + \langle f_p, x_p\rangle$ is reached at $\pi_V(F) = (f_1/|f_1|, ..., f_p/|f_p|)$. In this model, the matrix $\pi_V(F)$ is the mode of the distribution of patterns and plays a role similar to the mean of a Gaussian distribution. The mode of the full distribution of latent variables thus refers to $(\pi_V(F), \mu)$. In the particular case where F has orthogonal columns, the column norms of F correspond to its singular values. In the remainder of the paper we call them the *concentration parameters* of the distribution. The variability of the adjacency matrices is thus fully characterized by σ_ε, σ_λ and the concentration parameters. The pattern weights $\lambda^{(k)}$ are the eigenvalues of the $X^{(k)}\mathrm{Diag}(\lambda^{(k)})X^{(k)\top}$ term, and we thus call them eigenvalues even though they are not the actual spectrum of the real adjacency matrices $A^{(k)}$. Our model is summarized in Figure 2.

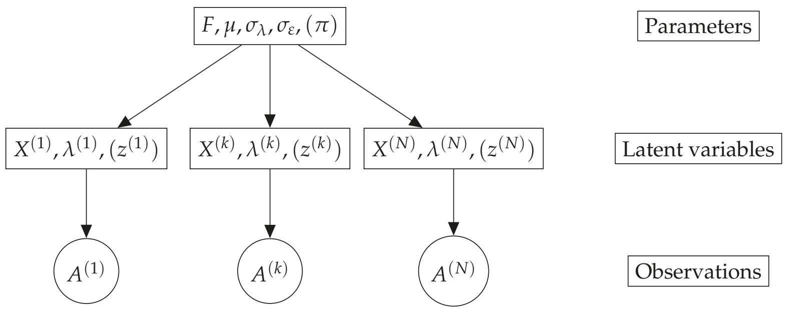

$F, \mu, \sigma_\lambda, \sigma_\varepsilon, (\pi)$
Parameters
$X^{(1)}, \lambda^{(1)}, (z^{(1)})$
$X^{(k)}, \lambda^{(k)}, (z^{(k)})$
$X^{(N)}, \lambda^{(N)}, (z^{(N)})$
Latent variables
$A^{(1)}$
$A^{(k)}$
$A^{(N)}$
Observations

Figure 2. Graphical model for a data set of adjacency matrices $A_1, ..., A_N$. The variables π and $z^{(k)}$ can be added to get a mixture model.

Note that this model may be adapted to deal with other types of adjacency matrices. The distribution for $\lambda^{(k)}$ can be effortlessly changed to a log-normal distribution to model data sets of positive matrices like covariance matrices. Binary networks can be modeled by removing the $\varepsilon^{(k)}$ noise and adding a Bernoulli sampling step, considering $X^{(k)}\lambda^{(k)}X^{(k)\top}$ as a logit. Adjacency matrices with positive coefficients are considered by adding the softplus function $x \mapsto \log(1 + e^x)$ in Equation (1). These extensions bring a wide range of possible statistical models for adjacency matrices for which the estimation procedure is the same as the one developed below.

Equation (1) theoretically requires each $A^{(k)}$ to be close to a rank p matrix. While this assumption is reasonable for well-clustered networks like samples of an SBM, some real-life networks exhibit heavy eigenvalue tails and cannot be approximated accurately using low rank matrices. While our model should not be expected to provide a perfect fit on general networks data sets, its main goal is to retrieve the principal modes of variability and their weight in an interpretable way, comparable to probabilistic Principal Component Analysis (PCA) or probabilistic Independent Component Analysis (ICA) [40]. An important difference with these methods is that our model expresses each of the p components using only an n-dimensional vector, whereas PCA and ICA require an $n \times n$ matrix per component to model adjacency matrices.

In the case of well clustered networks, our model can be seen as a refinement of the SBM better suited to data sets of networks. The SBM is designed to handle one single network and mainly addresses the problem of identifying the communities. In the case of network data sets, all subjects share the same node labels and the communities can be more easily identified by averaging the edge weights over the subjects. The main assumption of the SBM that the connections between the nodes are independent of one another prevents from further analyzing individual-level variability. In contrast, our model can account for the impact of a node variation on its connections, as well as pattern variations affecting the whole network. In the limit where the concentration parameters become very large and the weight variance is small, the patterns become constant and our model becomes equivalent to an SBM for networks organized in distinct clusters.

Another remark can be made on the identifiability of the model: the manifold of matrices of the form $X\mathrm{Diag}(\lambda)X^\top$ with $X \in \mathcal{V}_{n,p}, \lambda \in \mathbb{R}^p$ (also known as the non-compact Stiefel manifold) has a tangent space T with dimension $\dim(\mathcal{V}_{n,p}) + p = np - p(p-1)/2$ at $X^{(k)}\mathrm{Diag}(\lambda^{(k)})X^{(k)\top}$. The noise $\varepsilon^{(k)}$ can be decomposed into components in T and its orthogonal complement $T^\top$ with dimension $n^2 - np + p(p-1)/2$. The component in T thus induces an implicit source of variability on X and λ, which depends on σ_ε. We show in the experiment section that it may lead to underestimating the concentration parameters $(|f_1|, ..., |f_p|)$. While aware of this phenomenon, we consider it an acceptable trade-off regarding the simple formulation of Equation (2).

3.3. Mixture Model

The matrix distribution introduced in the previous section can be integrated in a mixture model to account for heterogeneous populations with a multi-modal distribution and variability. It amounts to considering K clusters with, for each cluster, a probability π^c and a parameter $\theta^c = (F^c, \mu^c, \sigma^c_\varepsilon, \sigma^c_\lambda)$. The mixture model writes hierarchically:

$$\begin{cases} z^{(k)} \sim \mathrm{Categorical}(\pi) \\ (X^{(k)} \mid z^{(k)} = c) \sim \mathrm{vMF}(F^c) \\ (\lambda^{(k)} \mid z^{(k)} = c) \sim \mathcal{N}(\mu^c, (\sigma^c_\lambda)^2 I_p) \\ (A^{(k)} \mid X^{(k)}, \lambda^{(k)}, z^{(k)} = c) \sim \mathcal{N}(X^{(k)}\mathrm{Diag}(\lambda^{(k)})X^{(k)\top}, (\sigma^c_\varepsilon)^2 I_{n(n+1)/2}). \end{cases} \quad (3)$$

We show in the next section on parameter estimation that the mixture layer only comes at a small algorithmic cost.

4. A Maximum Likelihood Estimation Algorithm

We now turn to the problem of estimating the model parameters $\theta = (F, \mu, \sigma_\lambda, \sigma_\varepsilon)$ given a set of observations $(A^{(k)})_{k=1}^N$. Let us denote $\lambda \cdot X = X\mathrm{Diag}(\lambda)X^\top$. The complete likelihood is expressed as:

$$p((A^{(k)}),(X^{(k)}),(\lambda^{(k)});\theta) = \prod_{k=1}^{N} \frac{1}{K(\theta)} p(A^{(k)} \mid X^{(k)},\lambda^{(k)};\theta) p(X^{(k)};\theta) p(\lambda^{(k)};\theta)$$

with

$$\begin{cases} p(A^{(k)} \mid X^{(k)},\lambda^{(k)};\theta) = \frac{1}{|\sigma_\varepsilon|^{n^2}(2\pi)^{n^2/2}} \exp\left[-\frac{1}{2\sigma_\varepsilon^2}\|A^{(k)} - \lambda^{(k)} \cdot X^{(k)}\|^2\right] \\ p(X^{(k)};\theta) = \frac{1}{\mathcal{C}_{n,p}(F)} \exp\left[\mathrm{Tr}(F^\top X^{(k)})\right] \\ p(\lambda^{(k)};\theta) = \frac{1}{|\sigma_\lambda|^{p}(2\pi)^{p/2}} \exp\left[-\frac{1}{2\sigma_\lambda^2}\|\lambda^{(k)} - \mu\|^2\right] \end{cases}$$

We compute the maximum of the observed likelihood $p((A^{(k)});\theta)$ using the MCMC-SAEM algorithm introduced in the next section. The MLE is not unique, as a permutation or a change of sign in the columns of X (together with a permutation of λ) yield the same model. This invariance can be broken by sorting the eigenvalues μ in increasing order as long as they are sufficiently spread. However, in practice, several eigenvalues may be close, and imposing such an order hinders the convergence of the algorithm. We thus choose to leave the optimization problem unchanged and deal with the permutation invariance by adding a supplementary step to the MCMC-SAEM algorithm.

4.1. Maximum Likelihood Estimation for Exponential Models with the MCMC-SAEM Algorithm

When dealing with latent variable models, the standard tool for MLE is the Expectation-Maximization (EM) algorithm [41]. Given a general parametric model $p(y,z;\theta)$ with y an observed variable and z a latent variable, performing MLE amounts to maximizing $\log p(y;\theta) = \log \int p(y,z;\theta)\mathrm{d}z$, which is intractable in practice with classical optimization routines. The EM algorithm allows indirectly maximizing this objective by looping over two alternating steps:

1. *E-step*: Using the current value of the parameter θ_t, compute the expectation

$$Q_t(\theta) = \mathbb{E}_{p(z|y;\theta_t)}[\log p(y,z;\theta)]\,;$$

2. *M-step*: Find $\theta_{t+1} \in \mathrm{argmax}_\theta\, Q_t(\theta)$.

While the EM algorithm proves efficient to deal with simple models like mixtures of Gaussian distributions, it requires adaptation for the cases of more complicated models where the expectation in the $Q_t(\theta)$ function is intractable, and the distribution $p(z \mid y,\theta_n)$ cannot be explicitly sampled from to approximate the expectation.

The Markov Chain Monte Carlo–Stochastic Approximation EM algorithm (MCMC-SAEM) developed by [34] aims at overcoming these hurdles in the case of models belonging to the Curved Exponential Family. For such models, the log-density expresses as $\log p(y,z;\theta) = \langle S(y,z), \varphi(\theta)\rangle + \psi(\theta)$, where $S(y,y)$ is a sufficient statistic. The Q_t function then simply rewrites $Q_t(\theta) = \langle \mathbb{E}_{p(z|y;\theta_t)}[S(y,z)], \varphi(\theta)\rangle + \psi(\theta)$. In the MCMC-SAEM algorithm, the expectation of sufficient statistics is computed throughout iterations using Stochastic Approximation. The samples from $p(z \mid y;\theta_t)$ are drawn using a MCMC kernel $q(z \mid z_t;\theta_t)$ with invariant distribution $p(z \mid y;\theta_t)$. The procedure is recalled in Algorithm 1. Under additional assumptions on the model and the Markov kernel, the MCMC-SAEM algorithm converges toward a critical point of the initial objective $\log p(y;\theta)$ [42,43].

In the case of the model proposed in this paper, the MCMC-SAEM is well suited to the problem at hand as we have to deal with a latent variable model. In a setting with manifold-valued latent variables, the E-step of the SAEM algorithm becomes intractable; using the MCMC-SAEM allows overcoming this hurdle. Following the outline of Algorithm 1, we need to draw samples from $p(X^{(k)},\lambda^{(k)} \mid A^{(k)};\theta)$ and perform the maximization step using the stochastic approximation of sufficient statistics.

Algorithm 1: The MCMC-SAEM Algorithm

```
Initialize $\theta_0$, $z_0$ and $S_0$
repeat
    Simulate $z_{t+1} \sim q(\cdot \mid z_t; \theta_t)$ using MCMC
    Update $\bar{S}_{t+1} = (1 - \alpha_t)\bar{S}_t + \alpha_t S(y, z_{t+1})$
    Find $\theta_{t+1} \in \text{argmax}_\theta \langle \bar{S}_{t+1}, \varphi(\theta) \rangle + \psi(\theta)$
until convergence
return $\theta_T$, $(z_t)_{t=1}^T$
```

4.2. E-Step with Markov Chain Monte Carlo

4.2.1. Transition Kernel

The target density $p(X^{(k)}, \lambda^{(k)} \mid A^{(k)}; \theta)$ is known up to a normalizing constant, and it is sufficient to use MCMCs based on the Metropolis–Hastings acceptance rule [44]. The MCMC is structured as a Gibbs sampler alternating simulations of $X^{(k)}$ and $\lambda^{(k)}$ for each individual. Note that conditional density $p(\lambda^{(k)} \mid X^{(k)}, A^{(k)}; \theta)$ is a Gaussian distribution. However, when experimenting with the MCMC-SAEM, we find that using Metropolis–Hastings-based transitions rather than sampling directly from the true conditional distribution accelerates the Markov chain convergence. This is why we perform a Metropolis–Hastings within Gibbs sampler for both variables [45]. We generate proposals for λ with a symmetric Gaussian kernel with adaptive variance in order to reach a target acceptance rate. We also use a Metropolis Hastings transition for X, with the constraint that the variable stays on the Stiefel manifold. Several techniques can be used to generate such proposals. The most natural equivalent of the symmetric random walk consists of a geodesic random walk generated by normally distributed tangent vectors. This method can be employed as the exponential map on the Stiefel manifold has a closed-form expression relying on the matrix exponential [30]. Another option is to use the curves given by the Cayley transform as in [46]: Cayley curves can be considered a fast first-order approximation of the exponential map. Finally, a more direct approach consists of making non-manifold Gaussian transitions and projecting the result back onto the manifold using Lemma 2. In our experiments these three approaches turn out to give very similar performances, and in practice we use the last method, which is also the fastest.

Remark 1. *Our numerical implementation offers the possibility to use the Metropolis Adjusted Langevin Algorithm (MALA) instead of Metropolis–Hastings, as the gradient of the log-likelihood can be computed explicitly. While the experiments we have presented rely on the Metropolis–Hastings kernel, which is faster overall, we find that in some cases where the dimensions n and p grow large the MALA kernel allows accelerating the convergence.*

4.2.2. Permutation Invariance Problem

The non-uniqueness of the MLE translates into a practical hurdle to the convergence of the MCMC: if two eigenvalues μ_i, μ_j are close, we get $(\mu_i, \mu_j) \cdot (x_i, x_j) \simeq (\mu_j, \mu_i) \cdot (x_i, x_j)$. As a consequence, the distribution $p(X^{(k)}, \lambda^{(k)} \mid A^{(k)}; \theta)$ is multi-modal in $X^{(k)}$, with a dominant mode close to $\pi_V(F)$ and other modes corresponding to column sign variations and permutations among similar eigenvalues. These modes are numerical artifacts rather than likely locations for the true value of $X^{(k)}$. Exploring them in the MCMC-SAEM hinders the global convergence: they encourage the samples to spread over the Stiefel manifold, which in turn yields a very bad estimation of F by inducing a bias toward the uniform distribution.

We address the permutation invariance problem by adding a column matching step every five SAEM iterations for the first third of the SAEM iterations. This step is a greedy algorithm that aims at finding the column permutation of a sample $X^{(k)}$ that makes it closest to $M = \pi_V(F)$. It proceeds recursively by choosing the columns m_i, x_j with the greatest absolute correlation. The steps are summarized in Algorithm 2. The greedy permutation algorithm causes the MCMC samples to stabilize around a single mode, allowing estimation of the F parameter.

Algorithm 2: Greedy column matching

```
input $F \in \mathbb{R}^{n \times p}, X \in \mathcal{V}_{n,p}$
Compute $M = \pi_V(F), D = (\langle m_i, x_j \rangle)_{i,j=1}^p$
Let $I = J = \{1, ..., p\}$
Let $\sigma = (0, ..., 0)$ (column order), $\eta = (0, ..., 0)$ (column sign)
for $t \in [1, ..., n]$ do
    Find $i_t, j_t \in \mathrm{argmax}_{i \in I, j \in J} |D_{ij}|$
    Set $\sigma(j_t) = i_t, \eta(i_t) = \mathrm{sign}(D_{i_t j_t})$
    Set $I = I \backslash \{i_t\}, J = J \backslash \{j_t\}$
end
return $\sigma, \eta$
```

4.3. M-Step with Saddle-Point Approximations

The maximization step of the MCMC-SAEM algorithm has a closed form expression, except for the parameter F. In this section, we recall a method to estimate F in a general setting and apply this method to get the optimal model parameters given sufficient statistics.

4.3.1. Maximum Likelihood Estimation of Von Mises–Fisher Distributions

The main obstacle to retrieving the parameter F given samples $X_1, ..., X_N$ is the normalizing constant of the distribution: though analytically known, it is hard to compute in practice (see Pal et al. [47] for a computation procedure when $n = 2$). Jupp and Mardia [48] proved that the MLE exists and is unique as long as $p < n$ and $N \geq 2$, or $p = n$ and $N \geq 3$. Khatri and Mardia [32], who first studied the properties of the MLE, showed the following result:

Theorem 1 ([32])**.** *Let $X_1, ..., X_N$ be N samples from a von Mises–Fisher distribution and $\overline{X} = \frac{1}{n} \sum_{i=1}^n X_i$. Let $\overline{X} = \overline{U}\overline{D}\overline{V}^\top$ be the Singular Value Decomposition (SVD) of $\overline{X}$. Then the Maximum Likelihood Estimator can be written under the form $\hat{F} = \overline{U}\mathrm{Diag}(\hat{s})\overline{V}^\top$, with $\hat{s} \in \mathbb{R}_+^p$.*

Maximizing the log-likelihood of samples $X_1, ..., X_N$ is thus equivalent to solving the optimization problem

$$\mathrm{argmax}_{s \in \mathbb{R}^p} \mathrm{Tr}[\overline{V}\mathrm{Diag}(s)\overline{U}^\top \overline{X}] - \log \mathcal{C}_{n,p}(\overline{U}\mathrm{Diag}(s)\overline{V}^\top), \tag{4}$$

where $\mathcal{C}_{n,p}(F)$ is the normalizing constant of the vMF distribution.

Several methods were proposed to solve this problem: the authors of [32] provide approximate formulas when the singular values of F are all either very large or very small. The authors of [49] propose a method to approximate the normalizing constant, which in turn yields a surrogate objective for the MLE giving satisfactory results. Finally, in [50], a different formula is proposed, which applies when the singular values are small. When experimenting with von Mises–Fisher distributions, we found that the method proposed by [49] gives the most robust results for a wide range of singular values of F, even in a high-dimensional setting.

4.3.2. Application to the Proposed Model

Computational details for the likelihood rearrangement are deferred to Appendix A. The model belongs to the curved exponential family, and its sufficient statistics are:

$$S(A, X, \lambda) = \begin{cases} S^1 = \frac{1}{N} \sum_{k=1}^N X^{(k)} \\ S^2 = \frac{1}{N} \sum_{k=1}^N \lambda^{(k)} \\ S^3 = \frac{1}{N} \sum_{k=1}^N \left\| \lambda^{(k)} \right\|^2 \\ S^4 = \frac{1}{N} \sum_{k=1}^N \left\| A^{(k)} - \lambda^{(k)} \cdot X^{(k)} \right\|^2 . \end{cases}$$

These sufficient statistics are updated using the MCMC samples $(X_t^{(k)}, \lambda_t^{(k)})_{k=1}^N$ with the stochastic approximation $\bar{S}_{t+1} = (1-\alpha_t)\bar{S}_t + \alpha_t S(A, X_t, \lambda_t)$. The optimization problem defined by the M-step of the SAEM algorithm gives the following results:

$$\hat{\theta}_t = \begin{cases} \hat{F} & = \hat{F}(\bar{S}_t^1) \\ \hat{\mu} & = \bar{S}_t^2 \\ \hat{\sigma}_\lambda^2 & = \frac{1}{p}\left(\|\hat{\mu}\|^2 - 2\langle \hat{\mu}, \bar{S}_t^2 \rangle + \bar{S}_t^3\right) \\ \hat{\sigma}_\varepsilon^2 & = \frac{1}{n^2}\bar{S}_t^4 , \end{cases} \tag{5}$$

where $\hat{F}(\bar{S}_t^1)$ denotes the MLE of the von Mises–Fisher distribution. As explained in the section above, the method proposed by Kume et al. [49] allows estimating the normalizing constant of general Fisher–Bingham distributions. The approximation relies on rewriting the constant to make it depend on a density that fits into the framework of Saddle-Point Approximations [51]. We recall the main steps of the computation procedure for this approximation in Appendix A for the specific, simple case of vMF distributions.

In the definition of our model, we impose that the columns of F are orthogonal. As recalled in Section 2.2, the MLE for the vMF mode is $\overline{M} = \overline{U}\,\overline{V}^\top$, where $\overline{X} = \overline{U}\,\overline{D}\,\overline{V}^\top$ is the SVD of the empirical arithmetic mean of samples. Since the column norms correspond to the singular values when the columns are orthogonal, the MLE under this constraint can be sought under the form $\overline{M}\mathrm{Diag}(s)$. Hence, the optimization problem is used to estimate F:

$$\operatorname{argmax}_{s\in\mathbb{R}^p} \operatorname{Tr}[\mathrm{Diag}(s)\overline{M}^\top \overline{X}] - \log \widehat{\mathcal{C}}_{n,p}(\overline{M}\mathrm{Diag}(s)) , \tag{6}$$

with $\widehat{\mathcal{C}}_{n,p}$ the approximation of the normalizing constant. We solve this optimization problem using the open source optimization library `scipy.optimize`.

The complete procedure is summarized in Algorithm 3.

4.4. Algorithm for the Mixture Model

The mixture model adds a cluster label $z^{(k)}$ for each subject and a list π of cluster probabilities. The model still remains in the curved exponential family, and the MCMC-SAEM algorithm can still be used. The Gibbs sampler now also updates $z^{(k)}$: it consists of sampling from the probabilities $p(z^{(k)} \mid X^{(k)}, \lambda^{(k)}, A^{(k)}; \pi, \theta)$, which can be computed explicitly. The sufficient statistics S^1, S^2, S^3, S^4 are defined and stored for each cluster. The statistics of cluster c are updated using only the indices k such that $z^{(k)} = c$. The variable π adds new sufficient statistics: $S^\pi = (\#\{k \mid z^{(k)} = c\}/N)_{c=1}^K$. The related MLE estimate of π is $\hat{\pi} = S^\pi$.

In our implementation, we initialize the clusters using the K-Means algorithm. We use the tempering proposed by [52] for the z sampling step in order to encourage points moving between clusters at the beginning of the algorithm. The vMF parameters F^c are aligned every 5 SAEM iterations using Algorithm 2 in order to allow the latent variables to move between the regions of influence of different clusters through small Metropolis–Hastings steps. The resulting algorithm is detailed in Appendix C.

4.5. Numerical Implementation Details

We initialize the algorithm by taking the first eigenvectors and eigenvalues of each adjacency matrix. Algorithm 2 is used to align the eigenvectors between samples. In order to accelerate the convergence, we perform a small number of hybrid MCMC-SAEM steps at the start of the algorithm, where the MCMC step on X is replaced with a gradient ascent step on the log-likelihood. These first steps move the $X^{(k)}$'s to an area of $\mathcal{V}_{n,p}$ with high posterior probability, which accelerate the convergence of the MCMC, as the X variable is the slowest to evolve along the MCMC-SAEM iterations. The Riemannian gradient ascent is detailed in Appendix B.

Algorithm 3: Maximum Likelihood Estimation algorithm for $\theta = (F, \mu, \sigma_\varepsilon, \sigma_\lambda)$

```
Initialize $\theta_0$, $X_0$, $\lambda_0$ and $S_0$
for $t = 1$ to $T$ do
    if $t \leq T/3$ and $(t \bmod 5) = 0$ then
        for $k = 1$ to $N$ do
            Use Algorithm 2 to align $X_t^{(k)}$ with $\pi_V(F_t)$.
            Permute $\lambda_t^{(k)}$ accordingly.
        end
    end
    Set $\widetilde{X}_0^{(k)} = X_t^{(k)}$ and $\widetilde{\lambda}_0^{(k)} = \lambda_t^{(k)}$
    for $\ell = 1$ to $n_{\mathrm{MCMC}}$ do
        for $k = 1$ to $N$ do
            Sample $\widetilde{X}_\ell^{(k)}$ from the Metropolis kernel $q_X(\cdot \mid \widetilde{X}_{\ell-1}^{(k)}, \widetilde{\lambda}_{\ell-1}^{(k)}; \theta_t)$ targetting
              $p(X^{(k)} \mid A^{(k)}, \widetilde{\lambda}_{\ell-1}^{(k)}; \theta_t)$
            Sample $\widetilde{\lambda}_\ell^{(k)}$ from the Metropolis kernel $q_\lambda(\cdot \mid \widetilde{X}_\ell^{(k)}, \widetilde{\lambda}_{\ell-1}^{(k)}; \theta_t)$ targetting
              $p(\lambda^{(k)} \mid A^{(k)}, \widetilde{X}_{\ell-1}^{(k)}; \theta_t)$
        end
    end
    Set $X_{t+1}^{(k)} = \widetilde{X}_{n_{\mathrm{MCMC}}}^{(k)}$ and $\lambda_{t+1}^{(k)} = \widetilde{\lambda}_{n_{\mathrm{MCMC}}}^{(k)}$
    Update the sufficient statistics $\bar{S}_{t+1} = (1 - \alpha_t)\bar{S}_t + \alpha_t S(A, X_{t+1}, \lambda_{t+1})$
    Compute $\mu_{t+1}$, $(\sigma_\varepsilon)_{t+1}$ and $(\sigma_\lambda)_{t+1}$ using Equation (5).
    Compute $F_{t+1}$ by solving problem (6).
end
return $\theta_T$, $(X_t, \lambda_t)_{t=1}^T$
```

The Metropolis–Hastings transition variance is selected adaptively throughout the iterations using stochastic approximation. At SAEM step $t + 1$, the proportion of accepted Metropolis transitions is computed. The logarithm of the variance is then incremented according to the rule $\log \sigma_{MH}^{t+1} = \log \sigma_{MH}^t + \ell_t / 2t^{0.6}$, with $\ell_t = \pm 1$ depending on whether the proportion of accepted jumps should be increased or decreased.

During the first half of the T iterations we set $\alpha_t = 1$ in order to minimize the impact of poor initializations. Then α_t decreases as $1/(t - T/2)^{0.6}$, which ensures the theoretical convergence of the algorithm.

The algorithms as well as all the experiments presented in this paper are implemented with Python 3.8.6. The package Numba [53] is used to accelerate the code. We provide a complete implementation (https://github.com/cmantoux/graph-spectral-variability, accessed on 19 April 2021), which allows reproducing the experiments on synthetic data and running the algorithm on new data sets.

5. Experiments

5.1. Experiments on Synthetic Data

5.1.1. Parameters Estimation Performance

First we investigate the ability of the algorithm to retrieve the correct parameters when the data are simulated according to Equations (1) and (2). We test the case $(n = 3, p = 2)$, referred to as low-dimensional, where X can be visualized in three dimensions as well as the case $(n = 40, p = 20)$, referred to as high-dimensional.

Small Dimension

We choose F with two orthogonal columns uniformly in $\mathcal{V}_{3,2}$ with column norms $(25, 10)$. Using these low concentration parameters makes the results simple to visualize. We set $\mu = (20, 10)$ and $\sigma_\lambda = 2$, and generate $N = 100$ matrices $A^{(k)}$ with $\sigma_\varepsilon = 0.1$ and 100 other matrices with the same $X^{(k)}$'s and $\lambda^{(k)}$'s but a much stronger noise standard deviation $\sigma_\varepsilon = 4$. We run the MCMC-SAEM algorithm for 100 iterations with 20 MCMC steps for each maximization step. The results are shown in Figure 3. In both cases, the mode of the vMF distribution $\pi_V(F)$ is well recovered. In the small noise case, the posterior X samples closely match the true X samples, and the estimated concentration parameters $(23.7, 8.0)$ remain close to ground truth. In the strong noise case, the posterior samples spread much farther around $\hat{F}$ than the true samples: the estimated concentration is $(9.9, 2.8)$. This result highlights the remark in Section 3.2 on the bias induced by the Gaussian noise on the latent variable spread: the best X variable to estimate the matrix $A^{(k)}$ is moved apart from the true $X^{(k)}$ in a random direction because of the noise $\varepsilon^{(k)}$ living outside the manifold.

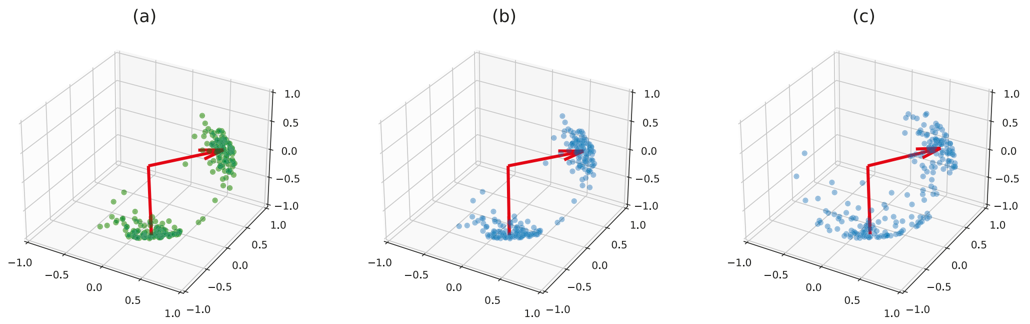


Figure 3. True latent variables $X^{(k)}$ and their posterior MCMC mean estimation. The red arrows represent the true $\pi_V(F)$ parameter and its estimate $\pi_V(\hat{F})$. (**a**) The true mode and samples. (**b**) Mode and samples estimates when $\sigma_\varepsilon = 0.1$. (**c**) Mode and samples estimates when $\sigma_\varepsilon = 4$. The columns are rearranged using Algorithm 2 to ease visualization. The latent variables are accurately estimated when the noise is small. A stronger noise causes the estimated latent variables to spread over the Stiefel manifold.

High Dimension

We now consider a synthetic data set of $N = 200$ samples generated from 20 latent patterns in dimension 40 and $\sigma_\varepsilon = 1, \sigma_\lambda = 2$, with various sizes of concentration parameters and eigenvalues, pairing large eigenvalues together with high concentrations. We run the MCMC-SAEM algorithm for 100 iterations with 20 MCMC steps per SAEM iteration to obtain convergence. The convergence of the parameters is shown in Figure 4. For both the concentration parameters and the eigenvalues, the algorithm starts by finding the highest values, only identifying lower values progressively afterward. The lowest values are associated to patterns with low weight, hence their recovery is naturally more difficult. As in the previous sections, the concentration parameters tend to be underestimated, indicating wider spreading around the mode vectors $f_i/|f_i|$ than the original latent variable. However, the ordering and orders of magnitude of the concentrations stay coherent, which, in practice, allows interpreting them and comparing them to each other. The estimation $\hat{F}$ matches the true parameter with a relative Root Mean Square Error (rRMSE) of 28%. As can be seen in Figure 5, the estimated normalized columns closely correspond to the original ones except when the concentration parameters get too small to allow for a good estimation, as explained above.

We use this example to illustrate the role of the algorithm hyperparameters on the practical convergence, namely the number of MCMC steps per SAEM iteration and the column matching step. We consider the same data set, but we initialize the MCMC-SAEM algorithm with random latent variables instead of the method described in Section 4: this worst-case initialization highlights the differences between the settings more easily. It is also closer to the case of real data sets: the MCMC and model parameters are slower to converge on real data, as the adjacency matrices are not actual samples of the theoretical model distribution. For different numbers of MCMC steps per SAEM iterations, we run the MCMC-SAEM algorithm for 200 iterations 10 times to average out the random initialization dependency, with and without the column matching step. Then we compute the relative RMSE of the parameters F and μ at the end of the algorithm. The rRMSE averaged over the 10 runs is shown in Figure 6. It can be seen that when the column matching step is used, increasing the number of MCMC steps at a fixed number of SAEM iterations improves the estimation. It allows accelerating the convergence, as MCMC steps are faster than the maximization step (which requires repeated vMF normalizing constant computations). However, when the number of MCMC steps gets too large, the performance improvement stagnates while the execution time increases. We find that, in practice, using between 20 and 40 MCMC steps per SAEM iterations is a good compromise in terms of convergence speed. Figure 6 also illustrates the need for the column matching step proposed in Section 4: when not used, the parameters hardly converge to the right values, even with a large number of MCMC steps per SAEM iteration. When the eigenvectors are permuted differently across the samples, the related eigenvalues cannot be estimated accurately, as they mix together when averaged in the maximization step. The abscence of permutations also spreads the eigenvectors over the Stiefel manifold, which prevents estimating the von Mises–Fisher parameter. Since Algorithm 2 is very fast to execute, it is not a computational bottleneck. In our experiments, the number of SAEM iterations between successive column permutation steps did not have a significant impact as long as it was not too high: values between 5 and 20 produced similar results.

Model Selection

In all the experiments on simulated data presented in this paper, we use the correct number of columns p, which we assume to be known. However, when studying real data sets, classical model selection procedures like the Bayesian Information Criterion cannot be applied to our model: they require computing the complete probability of the observations $p(A \mid \theta_m) = \int_{\mathcal{V}_{n,p}} \int_{\mathbb{R}^{pm}} p(A \mid X, \lambda, \theta_m)\, \mathrm{d}X\, \mathrm{d}\lambda$ for each model θ_m. This probability cannot be computed explicitly, as it requires integrating over the Stiefel manifold, which results in intractable expressions using the matrix hypergeometric function [49].

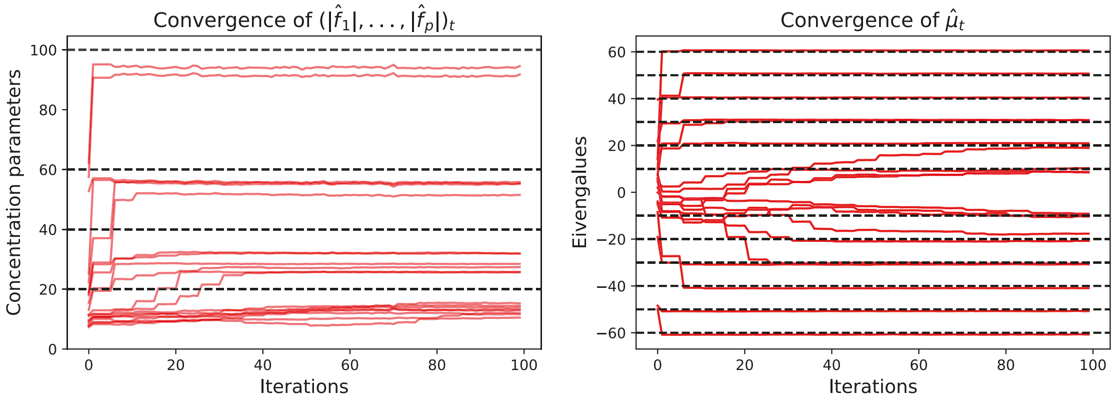


Figure 4. Convergence of the concentration parameters $(|f_1|, \ldots, |f_p|)$ (**left**) and the mean eigenvalues μ (**right**) over the SAEM iterations. The red lines represent the values of the parameters along the iterations. The black dotted lines represent the true values, which are grouped in batches to ease visualization. The convergence is fastest for the large eigenvalues and concentration parameters. At the start of the algorithm, the biggest changes in the parameters come from the greedy permutation performed every 5 iterations. As explained in the text, the concentration parameters are underestimated. However, they keep the right order of magnitude, which allows interpreting the output of the algorithm in practice.

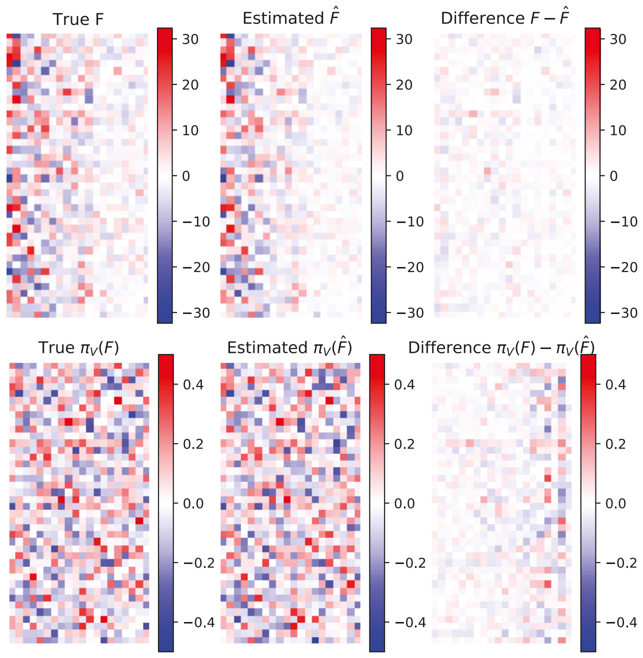


Figure 5. Von Mises-Stiefel distribution parameter F and its estimation $\hat{F}$. (**Top row**): the two parameters and their difference. (**Bottom row**): mode of the true distribution (given by $\pi_V(F)$), mode of the estimated distribution $\pi_V(\hat{F})$ and their difference. The images show each matrix as an array of coefficients, with pixel color corresponding to coefficient amplitude. Since the matrix columns are orthonormal, the projection just consists of normalizing the columns. The columns are sorted by decreasing the concentration parameter. The normalized columns of F corresponding to the smallest concentration parameters are estimated with less precision.

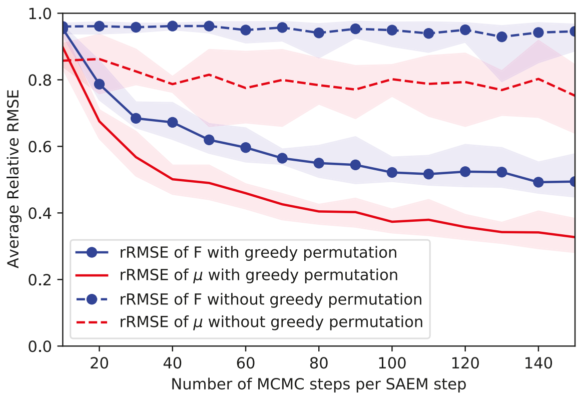


Figure 6. Relative RMSE of parameters F and μ after 100 MCMC-SAEM iterations depending on the number of MCMC steps per SAEM iteration. Results are averaged over 10 experiments to reduce the variance. The shaded areas indicate the extremal values across the repeated experiments. When using the greedy permutation, the rRMSE decreases rapidly when the number of MCMC steps increases before stabilizing. On the other hand, without the permutation step, the performance stays poor for any number of MCMC steps per maximization, as the parameters cannot be estimated correctly. In this experiment only, the latent variables are initialized at random to highlight the result.

In practice, several tools can be used to choose the number of latent patterns. First, the marginal likelihood $p(A \mid X, \lambda; \theta)$ or the error $\|A - \lambda \cdot X\|$ can be used to evaluate the model expressiveness. As p increases, the error will naturally diminish and should be very small for $p = n$. As with linear models, the proportion of the variance captured by $\lambda \cdot X$ can be computed to evaluate the improvement gained by adding new patterns. The concentration parameters of the von Mises–Fisher distribution also give important information on pattern relevance: if a pattern has a very low concentration parameter, it means that the related eigenvectors are widely spread across the Stiefel manifold. Smaller concentrations are thus related to overfitting, as they do not correspond to actual patterns contributing to the data set variability. The relative importance of concentration parameters can be compared numerically with the vMF concentration obtained on samples from the uniform distribution gathered with Algorithm 2.

Remark 2. *In this paper, we approximate the posterior mean of MCMC samples of $X^{(k)}$ by projecting their arithmetic mean over the Stiefel manifold. We find this procedure a very convenient alternative to computing the Fréchet mean (i.e., the Riemannian center of mass) over the manifold for two reasons. First, computing the Fréchet mean requires an extensive use of the Riemannian logarithm. Although a recent paper [31] allows computing this logarithm, the proposed algorithm heavily relies on matrix logarithm computations and requires points to remain very close to the mean. Similar iterative algorithms to compute the mean based on other retraction and lifting maps than the Riemannian exponential and logarithm were proposed and analyzed in [54], but in our experiments, these alternatives also turn out to require samples close to the mean point, especially in high dimensions. Second, projecting the mean sample onto the Stiefel manifold amounts to computing the mode of a vMF distribution. As shown in Appendix D, the vMF distribution is symmetric around its mode, which makes this mode a summary variable similar to the Gaussian mean.*

5.1.2. Missing Links Imputation

Once the parameters $\hat{\theta}$ are estimated from adjacency matrices $A_1, ..., A_N$, missing links can be inferred on a new adjacency matrix A. Suppose that only a subset Ω of the edge weights is known: the weights of masked edges $\overline{\Omega}$ can be obtained by considering the posterior distribution $p(A_{\overline{\Omega}} \mid A_\Omega; \theta)$. This distribution is obtained as a marginal of the full posterior $p(A_{\overline{\Omega}}, X, \lambda \mid A_\Omega; \theta)$. Sampling from this distribution yields a posterior mean as well as confidence intervals for the value of missing links. In the case of binary networks, the posterior distribution gives the probability of a link existing for each masked edge. Samples are obtained by Gibbs sampling using the same method as in Section 4. We also compute the Maximum A Posteriori (MAP) by performing gradient ascent on the posterior density of $(A_{\overline{\Omega}}, X, \lambda)$ given A_Ω.

We generate a synthetic data set of $N = 200$ adjacency matrices with $n = 20$ nodes and $p = 5$. The noise level σ_ε is chosen such that the average relative difference between the coefficients of $A^{(k)}$ and $\lambda^{(k)} \cdot X^{(k)}$ is 25%. We estimate the model parameters using the MCMC-SAEM algorithm. Then, we generate another 200 samples from the same model. We mask 16% of the edge weights corresponding to the interactions between the last eight nodes. The posterior estimation is compared with the ground truth for one matrix in Figure 7. Both the MAP and posterior mean allow to estimate the masked coefficients better than the mean sample $(A_1 + ... + A_N)/N$, which is the base reference for missing data imputation. They achieve, respectively, 58% (±28%) and 57% (±24%) rRMSE on average, whereas the mean sample has an 85% (±10% over the data set) relative difference to the samples on average. Finally, we perform the same experiment except we select the masked edges uniformly at random, masking 40% of the edges. This problem is easier than the former despite the larger amount of hidden coefficients because the missing connections are not aligned with each other. The posterior mean and the MAP achieve, respectively, 34% (±9% over the data set) and 35% (±7%) rRMSE, against 75% (±5%) for the mean sample.

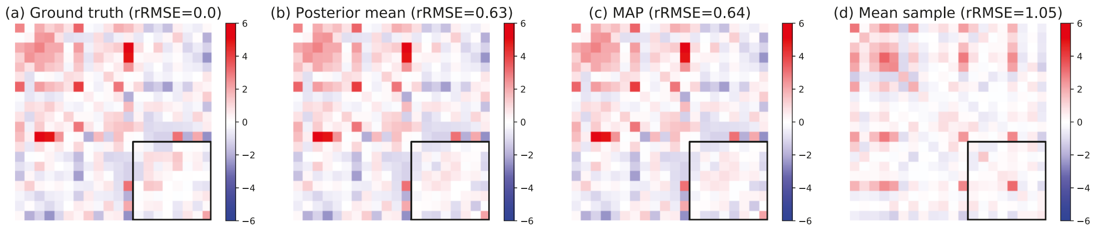


Figure 7. Result for missing link inference using the posterior distribution. (**a**) Ground truth input matrix A. (**b**) Posterior mean of the masked coefficients. (**c**) MAP estimator. (**d**) Mean of model samples for comparison. The area of masked edges is highlighted by a black square. Above each matrix is the rRMSE with the ground truth. Both the posterior mean and the MAP give a reasonable estimation for the missing weights, significantly better than the empirical mean of all adjacency matrices, which is the base reference for missing data imputation. The images show each matrix as an array of coefficients, with pixel color corresponding to coefficient amplitude.

Link prediction has been a very active research topic in network analysis for several decades, and numerous methods can be employed to address this problem depending on the setting [3,55,56]. However, the most commonly used approaches are designed to perform inference on a single network or consider the nodes as permutation invariant. In turn, the new approach we propose allows for population-informed prediction and uncertainty quantification. It could be used in practice to compare specific connection weights of a new subject with their distribution given the other coefficients and the population parameters. This comparison provides a tool to detect anomalies in the subject's connectivity network stepping out of the standard variability.

Remark 3. *The error uncertainties reported in this paper refer to the variance of the estimation error across the adjacency matrices.*

5.1.3. Clustering on Synthetic Data

As explained in Section 3.3, our model can be used within a mixture to account for multi-modal distributions of networks. When experimenting with the clustering version of our algorithm on data sets with distinctly separated clusters, we noticed that the algorithm provides results similar to running K-Means and estimating the parameters on each K-Means cluster separately. However, the clusters in complex populations often overlap, and the ideal case where all groups are well separated rarely occurs. In this section, we show two examples of simulated data sets where the variabilities of the clusters makes them hard to distinguish with the sole application of the K-Means algorithm.

Small Dimension

We test the mixture model estimation in the small dimensional case $(n = 3, p = 2)$ where results can be visualized. We simulate three clusters of matrices as in Section 5.1.1 with $N = 500$ samples overall. In order to make the problem difficult, we use the same mean eigenvalues for two clusters. We set the Stiefel modes of these clusters to be very close, differing mainly by their concentration parameters. We run the tempered MCMC-SAEM for 1000 iterations with a decreasing temperature profile $T_t = 1 + 50/t^{0.6}$. Once the convergence is achieved, the estimated clusters are mapped to the true clusters. The eigenvalue parameters are estimated accurately with 2% rRMSE. The original and estimated von Mises–Fisher distributions are compared in Figure 8. We can see that each cluster distribution is well recovered. In particular, the overlapping distributions of cluster 1 and 2 are separated, and the higher concentration of cluster 1 is recovered in the estimation. This example also highlights the relevance of the MCMC-SAEM clustering procedure compared with its K-Means initialization: up to a label permutation, 50.4% of the K-Means proposed labels are correct, whereas the posterior distribution $p(z^{(k)} \mid A^{(k)}; \hat{\theta})$ computed with the final MCMC samples predicts the correct answer for 79.6% of the model samples.

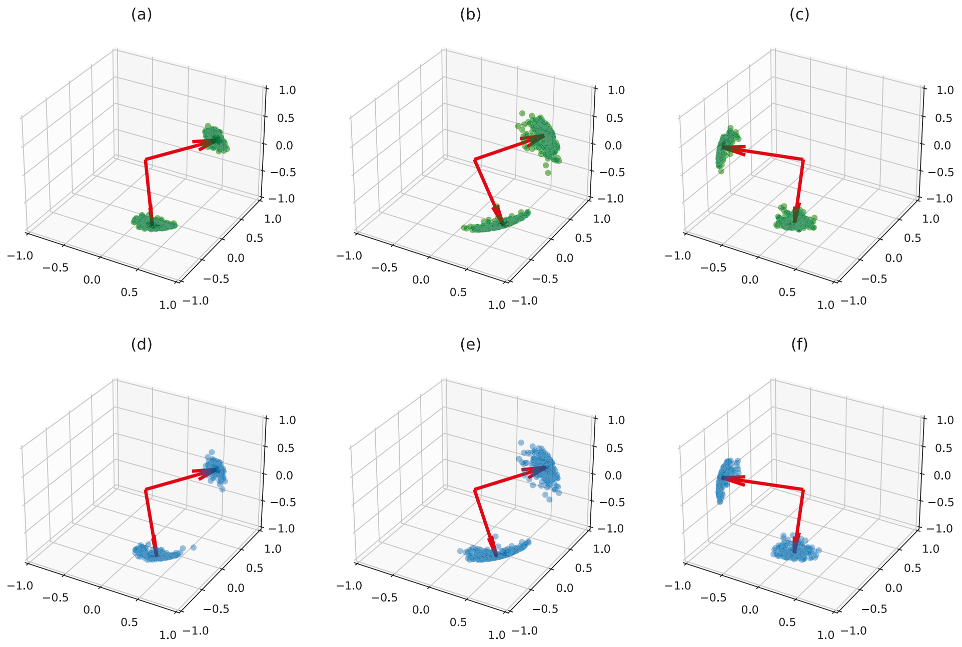


Figure 8. True latent variables $X^{(k)}$ and their posterior mean estimation for the clustering problem. (**Top row**): the plots (**a**–**c**) represent the true vMF modes (in red), as well as the true $X^{(k)}$ samples (in green) in their true class. (**Bottom row**): the plots (**d**–**f**) represent the three estimated vMF central modes (in red) and the estimated $X^{(k)}$ in their estimated class (in blue). The cluster centers are well recovered, as well as the concentration parameters. In particular, the two first clusters, which mainly differ by their concentration parameters, are correctly separated.

Larger Dimension

We now test the mixture model on a synthetic data set of 500 samples in dimension $(n = 20, p = 10)$. We generate four clusters with Stiefel modes close to one another, with equal concentration parameters. The modes mainly differ by their mean eigenvalues μ^c. The eigenvalue standard deviation σ_λ is set to be of the same order of magnitude as the means μ, larger than most of its coefficients. The resulting data set is hard to estimate with classical clustering: the K-Means algorithm retrieves 53.6% of correct labels at best. In contrast, running the tempered MCMC-SAEM algorithm for 1000 iterations yields 99.4% of correct labels. The algorithm achieves this result by identifying the template patterns of each cluster despite the large variation in their weights. Once these template patterns are learned, the proportion of correctly classified samples increases and the mean eigenvalues of each cluster converge to a good estimation.

Model Selection

Selecting the number of clusters K is a known problem adressed for general mixture models [57]. Although it is well understood for simple Gaussian mixture models or for low dimensional data, other cases remain challenging problems. For the model proposed in this paper, likelihood-based procedures cannot be applied, as the complete likelihood is an integral over the Stiefel manifold (see Section 5.1.1). As with the selection of parameter p, the concentration parameters and the reconstruction errors could be used to choose the number of clusters. Using a K that is too small will result in stretching the latent von

Mises–Fisher distributions toward low concentration parameters and large reconstruction errors. The reconstruction error should decrease slower once the right number of clusters has been reached.

Remark 4. *The link prediction procedure described in Section 5.1.2 could also be applied in the mixture model to infer the coefficients of new networks of which the class is unknown.*

5.2. Experiments on Brain Connectivity Networks

We test our model on the UK Biobank data repository [58]. The UK Biobank is a large scale data collection project, gathering brain imaging data on more than 37,000 subjects. In this paper, we are interested more specifically in the resting-state functional Magnetic Resonance Imaging data (rs-fMRI). The rs-fMRI measures the variations of blood oxygenation levels (BOLD signals) across the whole brain while the subject is in a resting state, i.e., receives no stimulation. The brain is then divided into regions through a spatial ICA that maximizes the signal coherence within each region [59]. Smaller regions give more detail on the brain structure but are less consistent across individuals. Finally, the raw imaging data are processed to obtain a matrix that gathers the temporal correlations between the mean blood oxygenation levels in each region. This matrix thus represents the way brain regions activate and deactivate with one another. It is called the *functional connectivity network* of the brain, as it provides information on the role of the regions rather than their physical connections. In the UK Biobank data used in the present study, the connectivity matrices are defined on a parcellation of the brain into $n = 21$ regions. These connectivity matrices illustrate our purpose well: as shown in Figure 9, the data set has a very large diversity of networks that express in patterns with varying weights.

5.2.1. Parameter Estimation

We run our algorithm on $N = 1000$ subjects for 1000 SAEM iterations with 20 MCMC steps per SAEM iteration. Working on a restricted number of samples allows for a fast convergence toward the final values. Indeed, we noticed that, while most of the parameters stabilize relatively fast, the time to convergence of the concentration parameters grows with the number of samples. Apart from these concentration parameters, we obtained very similar results when taking all the UK Biobank subjects. In this section, we consider a decomposition into $p = 5$ patterns. In Appendix E.1, we show the results obtained by taking different values of p.

In Figure 10, we show the p normalized patterns $f_i f_i^\top / \|f_i\|^2$ obtained once the algorithm has converged. Patterns 3 and 5 have very high concentration parameters and only use a small subset of the nodes. The three other patterns have smaller concentration parameters. However, these concentrations are still high enough for the related columns of X to be significantly more concentrated than a uniform distribution: the average Euclidean distance between these three columns of $X^{(k)}$ and the related mode columns is 1.1 ($\pm$0.2 over the data set). Comparatively, the average distance between two points drawn uniformly on the Stiefel manifold is 2.4 ($\pm$0.2) (over 10,000 uniform samples).

Figure 11 displays data set matrices $A^{(k)}$ alongside the respective mean posterior estimates of $\lambda^{(k)} \cdot X^{(k)}$. For comparison purpose, we also compute the approximation given by the projection onto the subspace of the first five PCA components of the full data set, where each component has been vectorized. The $\lambda \cdot X$ matrices capture the main structure, whereas the PCA approximation relying on the same number of base components provides a less accurate reconstitution. Quantitatively, the $\lambda \cdot X$ term has a 47% ($\pm$5% over the data set) relative distance to A, whereas the PCA approximation has a 92% ($\pm$12%) relative distance to A. The $\lambda \cdot X$ representation accounts for 60% of the total variance, whereas the corresponding PCA representation only accounts for 35%. This difference highlights the benefits of taking into account the variations of the patterns across individuals. In a classical dictionary-based representation model, the patterns do not vary among individuals. In contrast, accounting for the pattern variability only adds a small number of parameters (one per pattern) and increases the representation power.

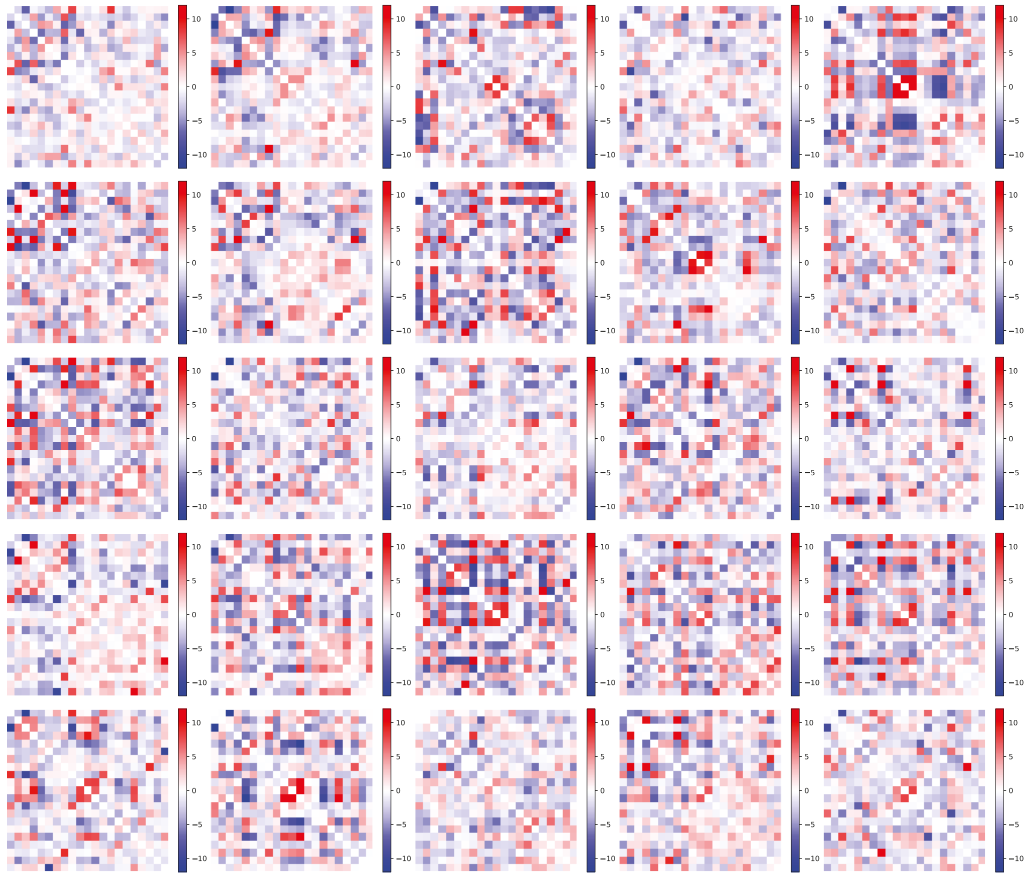


Figure 9. Functional connectivity matrices (21 × 21) of 25 UK Biobank subjects. The connectivity structure changes a lot depending on the subject, with various patterns expressing with different weights. The matrices in the data set have no diagonal coefficients; hence, the diagonals are shown as zero.

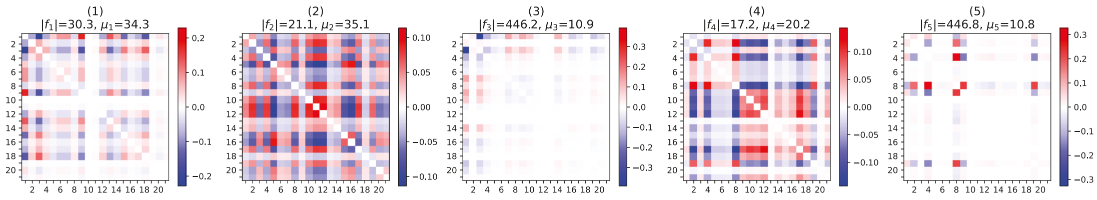


Figure 10. Normalized rank-one connectivity patterns. The matrix i represents $\text{sign}(\mu_i) f_i f_i^\top / \|f_i\|^2$. The caption above each pattern gives the related concentration parameter and mean eigenvalue. The diagonal coefficients are set to zero, as they do not correspond to values in the data set. The images show each matrix as an array of coefficients, with pixel color corresponding to coefficient amplitude.

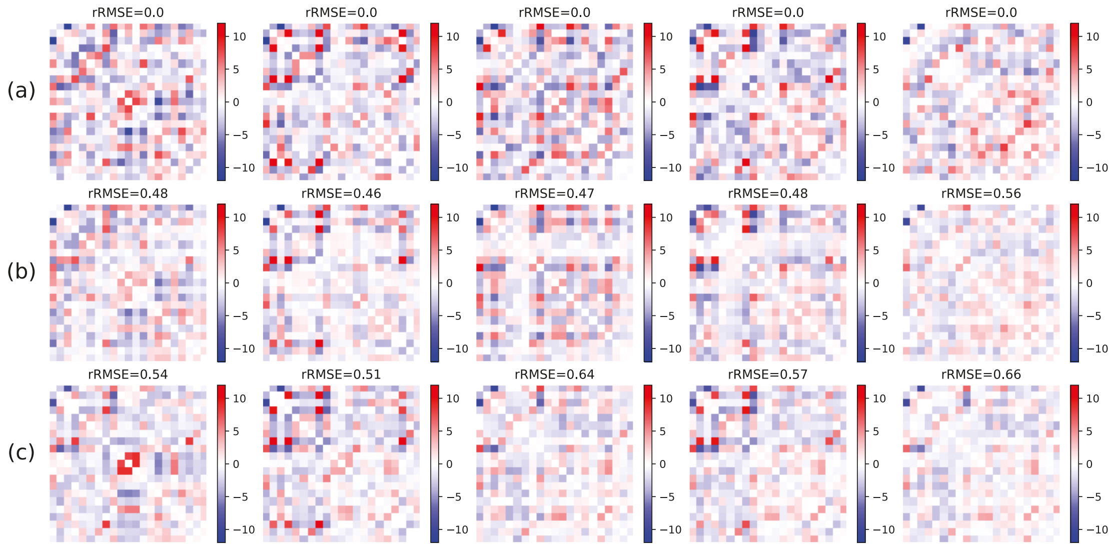


Figure 11. (**a**) UK Biobank connectivity matrices for 5 subjects. (**b**) Corresponding posterior mean value of $\lambda \cdot X$ estimated by the MCMC-SAEM. (**c**) Projection of the true connectivity matrices onto the subspace of the first five PCA components. The posterior mean matrix achieves a better rRMSE than PCA by capturing the main patterns of each individual matrix. As in Figure 10, the diagonal cofficients are set to zero.

5.2.2. Pattern Interpretation

Once the patterns are identified, they can be interpreted based on the function of the related involved brain regions. All brain regions can be found on a web page of the UK Biobank project (https://www.fmrib.ox.ac.uk/datasets/ukbiobank/group_means/rfMRI_ICA_d25_good_nodes.html, accessed on 19 April 2021). The regions analyzed in this section can be visualized on brain cuts in Appendix E.2.

Pattern 3 mainly represents the anti-correlation between regions 1 and 3. Region 1 comprises, among others, the inner part of the orbitofrontal cortex and the precuneus. These regions are parts of the Default Mode Network (DMN) of the brain, which is a large-scale functional brain network known to be active when the subject is at rest or mind-wandering [60]. Region 3 comprises part of the insular cortex and the post-central gyrus, which both play a role in primary sensory functions. The anti-correlation between regions 1 and 3 is a consequence of external sensations activating the sensory areas and decreasing the DMN activity. This anti-correlation is also one of the strongest coefficients in pattern 1.

Pattern 5 mainly features the dependency between nodes 2, 4, 8, 9, and 19, which are all related to the visual functions. Node 2 represents the parts of the occipital and temporal lobes forming the ventral and dorsal streams: they are theorized to process the raw sensory vision and hearing to answer the questions “what?” and “where?” [61]. Region 4 features the cuneus, which is a primary visual area in the occipital lobe. Region 8 spans over the whole occipital lobe, covering primary visual functions and associative functions like the recognition of color or movement. Region 9 comprises the continuation of the ventral and dorsal streams of region 2 in the parietal and medial temporal areas. Finally, Region 19 represents the V1 area that processes the primary visual information. Pattern 5 involving these regions has a very high concentration parameter, which means that this structure remains very stable among the subjects.

Considering that the subject’s activity in the MRI scanner mainly consists of looking around and laying still, it is coherent that the most stable patterns (i.e., with highest concentration parameters) during the resting-state fMRI measurement are the activity of the vision system and the anti-correlation between the DMN and sensory areas.

Pattern 4 also shows the interaction between the visual areas 2, 4, 8, and 19. It also includes the strong correlation between nodes 9, 10, 11, 12, and 17. Regions 10, 11, and 12 are involved in motor functions. Region 10 features part of the pre-central gyrus, which is central in the motor control function, and part of the post-central gyrus, which is involved in sensory information processing. Region 11 encompasses the entire pre-central gyrus. Region 12 includes a part of the motor and pre-motor cortex in the frontal lobe and the insular cortex. It also includes the cerebellum, which plays an important role in motor control, and the insular cortex, which also acts on the motor control, for instance, in the face and hands motion control [62]. Region 17 comprises the medial face of the superior temporal gyrus and the hippocampus, which are involved in short and long-term memory and spatial navigation.

Pattern 2 combines, to some extent, the structure contained in patterns 4 and 5. It features, among others, interactions between the visual areas and the correlation between the motor function areas.

Remark 5. *The results and interpretation we present in this experiment depend on the state of the subjects—in this case, a resting state—and the brain parcellation used to obtain the definition of the regions. If we were to analyze another data set of subjects performing a different task, the connectivity patterns X would likely differ from their resting-state counterpart. It follows from the fact that two different phenomenons naturally require two different base dictionaries. Analyzing the pattern difference would thus provides a way to interpret the structure difference between the two settings. For instance, the role of the occipital lobe in the vision-involved patterns would likely change for tasks related to vision. However, if the brain regions are defined differently in the two experiments, the comparison can only be made in a qualitative way.*

5.2.3. Link Prediction

We evaluate the relevance of our model on fMRI data by testing the missing link imputation method introduced in the previous section. First we fit the model on $N = 1000$ subjects. Then we take 1000 other test subjects and mask the edges corresponding to the interactions between the last nine nodes (except the diagonal coefficients, which are unknown and thus considered null). We compute the MAP estimator of the masked coefficients. For comparison purposes, we perform a linear regression to predict the masked coefficients given the visible ones. Finally, we truncate the matrix with masked coefficients to only keep the p dominant eigenvalues. This technique is at the core of low-rank matrix completion methods [63], and it relates naturally with the estimation derived from our model relying on low-rank variability. The result is shown for one sample in Figure 12. The linear model and the MAP estimator give comparable estimates, both close to the true masked coefficients. Over the 1000 test subjects, these estimators achieve on average 58% (±14% over the samples) rRMSE for the linear model and 65% (±15%) rRMSE for the MAP. Interestingly, our model uses only $np + p + 2 = 112$ degrees of freedom, whereas the linear prediction model has dimension 26,640 and was specifically trained for the regression purpose.

Our model captures a faithful representation of the fMRI data set and uses far fewer coefficients than other models like PCA and linear regression by accounting for the structure of the interactions between the network nodes. It provides an explanation of the network variability using simple interpretable patterns, which correspond to known specific functions and structures of the brain. The variations of these patterns and their weight allow for a representation rich enough to explain a significant proportion of the variance and impute the value of missing coefficients.

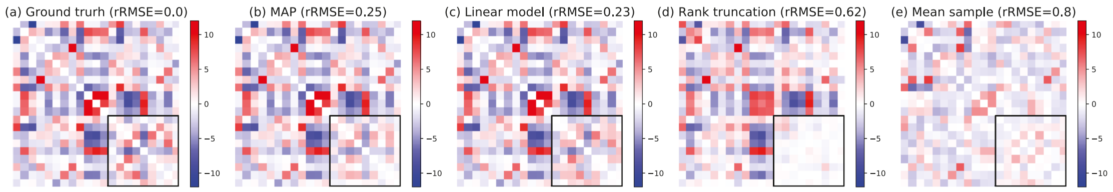


Figure 12. From left to right: (**a**) True connectivity matrix A. (**b**) MAP estimator for the masked coefficients framed in a black square. (**c**) Linear model prediction for the masked coefficients. (**d**) Rank 5 truncation of the matrix A with masked coefficients set to zero. (**e**) Mean of all data set matrices. Above each matrix is the rRMSE with the ground truth.

6. Conclusions

This paper introduces a new model for the analysis of undirected graph data sets. The adjacency matrices are expressed as a weighted sum of rank-one matrix patterns. The individual-level deviations from the population average translate into variations of the patterns and their weight. Sample graphs are characterized by these variations in a way similar to PCA. The form of the decomposition allows for a simple interpretation: each pattern corresponds to a matrix with rank one and is thus represented by a vector of node coefficients. The variability of this decomposition is captured within a small number of variance and concentration parameters.

We use the MCMC-SAEM algorithm to estimate the model parameters as well as the individual-level variable. The parameter of von Mises–Fisher distributions is recovered by estimating the vMF normalizing constant, which allows retrieving both the mode and its concentration parameters. Future work could further investigate the role of the approximation error induced by the use of saddle-point approximations, comparing its performance with a recently proposed alternative method [64]. The impact of noise on the underestimation of the vMF distribution concentration also requires further analysis.

Experiments on synthetic data show that the algorithm yields good approximations of the true parameters and covers the posterior distributions of the latent variables. Our model can be used to infer the value of masked or unknown edge weights once the parameters are estimated. In practice, the posterior distribution could be compared to the real connections to detect anomalous connections that step out of the expected individual variability.

The model we introduce is a hierarchical generative statistical model, which easily extends to mixture models. We show that a mixture of decomposition models can be estimated with a similar algorithmic procedure and allow disentangling between modes of variability that are indistinguishable by a traditional clustering method.

We demonstrate the relevance of the proposed approach for the modeling of functional brain networks. Using few parameters, it explains the main components of the variability. The induced posterior representation is more accurate than PCA and gives a link prediction performance similar to a linear model, which has a comparably simple structure, but requires far more coefficients and was trained specifically to that purpose. The estimated connectivity patterns have a simple structure and lead to an interpretable representation of the functional networks. We show that our model identifies specific patterns for the visual information processing system or the motor control. The related concentration parameters allow measuring the variability of the function of the related brain regions across the subjects.

This work focuses on cross-sectional network data sets, i.e., populations where each adjacency matrix belongs to a different subject and is independent of the others. Our model could also be used as a base framework for longitudinal network modeling using the tools proposed by Schiratti et al. [65]. This would consist of considering time-dependent latent variables X and λ for each subject, evolving close to a population-level reference trajectory in the latent space.

Future work could investigate the dependencies between the latent variables of the model. Correlation can be introduced between the patterns by using Fisher–Bingham distributions on the Stiefel manifold [38] and between pattern weights with full Gaussian covariance matrices. Another direction to develop is the quantification of the uncertainty: by adding prior distributions on F and μ, a Bayesian analysis would naturally provide posterior confidence regions for the model parameters [47]. Finally, our framework could be adapted to model graph Laplacian matrices instead of adjacency matrices. The analysis of the eigenvalues and eigenvectors of the graph Laplacian has proven of great theoretical [66] and practical [67] interest in network analysis. Understanding the variability of the eigendecomposition of graph Laplacians could help to design robust models relying on spectral graph theory.

Author Contributions: Conceptualization, C.M. and S.A.; methodology, C.M. and S.A.; software, C.M.; data curation, B.C.-D. and C.M.; validation, C.M. and S.A.; visualization, C.M. and B.C.-D.; result analysis, F.C., S.E., S.D., S.A. and C.M.; writing—original draft preparation, C.M.; writing—review and editing, C.M., S.D. and S.A.; supervision, S.D. and S.A. All authors have read and agreed to the published version of the manuscript.

Funding: The research leading to these results has received funding from the European Research Council (ERC) under grant agreement No. 678304, European Union's Horizon 2020 research and innovation program under grant agreement No 666992 (Euro-POND) and No. 826421 (TVB-Cloud). It was also funded by in part by the program "Investissements d'avenir" ANR-10-IAIHU-06 and the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Informed consent was obtained by the UK Biobank from all subjects involved in the study.

Data Availability Statement: The data used in this paper come from the UK Biobank repository. The website is hosted at https://www.ukbiobank.ac.uk/ (accessed on 19 April 2021). The data are accessed upon application and cannot be made available publicly.

Acknowledgments: This research has been conducted using the UK Biobank resource under application 53185.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A. SAEM Maximization Step

Appendix A.1. Maximum Likelihood Estimates for μ, σ_λ^2, σ_ε^2

Up to a constant normalization term c, the complete log-likelihood of the model in the Gaussian case writes:

$$\begin{aligned}\log p((A^{(k)}),(X^{(k)}),(\lambda^{(k)});\theta) &= \sum_{k=1}^{N} \log p(A^{(k)},X^{(k)},\lambda^{(k)};\theta)\\ &= \sum_{k=1}^{N}\left[-\frac{1}{2\sigma_\varepsilon^2}\left\|A^{(k)}-\lambda^{(k)}\cdot X^{(k)}\right\|^2-\frac{1}{2\sigma_\lambda^2}\left\|\lambda^{(k)}-\mu\right\|^2+\mathrm{Tr}(F^\top X^{(k)})\right] \\ &\quad - Nn^2\log\sigma - Np\log\sigma_\lambda - N\log\mathcal{C}_{n,p}(F)+c \\ &= N\left[\mathrm{Tr}(F^\top S_1)+\langle S_2,\frac{1}{2\sigma_\lambda^2}\mu\rangle - S_3\frac{1}{2\sigma_\lambda^2}-S_4\frac{1}{2\sigma_\varepsilon^2}+\Psi(\theta)\right]\end{aligned} \tag{A1}$$

with $\Psi(\theta) = -n^2 \log \sigma_\varepsilon - p \log \sigma_\lambda - \log \mathcal{C}_{n,p}(F) + c$, and

$$\begin{cases} S^1 = \frac{1}{N}\sum_{k=1}^N X^{(k)} \\ S^2 = \frac{1}{N}\sum_{k=1}^N \lambda^{(k)} \\ S^3 = \frac{1}{N}\sum_{k=1}^N \left\|\lambda^{(k)}\right\|^2 \\ S^4 = \frac{1}{N}\sum_{k=1}^N \left\|A^{(k)} - \lambda^{(k)} \cdot X^{(k)}\right\|^2 \end{cases}$$

The model thus belongs to the curved exponential family, and its sufficient statistics are given by (S^1, S^2, S^3, S^4). The log-likelihood is componentwise convex in μ, σ_λ^2 and σ_ε^2. Computing its gradient yields one single critical point, which is thus the maximum value.

In the case of the binary model, the log-likelihood writes:

$$\begin{aligned} \log p((A^{(k)}), (X^{(k)}), (\lambda^{(k)}); \theta) = & \sum_{k=1}^N \sum_{i,j=1}^n \left[A_{ij}^{(k)} \log h(\lambda^{(k)} \cdot X^{(k)})_{ij} + (1 - A_{ij}^{(k)}) \log(1-h)(\lambda^{(k)} \cdot X^{(k)})_{ij} \right] \\ & + \sum_{k=1}^N -\frac{1}{2\sigma_\lambda^2}\left\|\lambda^{(k)} - \mu\right\|^2 + \mathrm{Tr}(F^\top X^{(k)}) \\ & - Np \log \sigma_\lambda - N \log \mathcal{C}_{n,p}(F) + c \end{aligned}$$

with $h(x) = 1/(1+\exp(-x))$ the sigmoid function, which applies component-wise on matrices. The distribution $(A \mid \lambda, X)$ is non parametric and needs no estimation. Hence, for all the model parameters F, μ, σ_λ the MLE remains unchanged.

Appendix A.2. Saddle-Point Approximation of $\mathcal{C}_{n,p}(F)$

We recall the main steps to compute the approximation of $\mathcal{C}_{n,p}(F)$ proposed by Kume et al. [49]. For more details on the justification of the approximation and applications to more general distributions, we refer the reader to the original paper. Our implementation provides a function `spa.log_vmf`, which computes this approximation. The approximation $\widehat{\mathcal{C}}_{n,p}(F)$ for von Mises–Fisher distributions is written in Equation (16) of [49]:

$$\widehat{\mathcal{C}}_{n,p}(F) = \frac{2^p (2\pi)^{np/2 - p(p+1)/4}}{|\hat{K}''|^{1/2} |\hat{C}|^{1/2}} \exp\left\{ \frac{1}{2}\mathrm{vec}(F)^\top \hat{C}^{-1} \mathrm{vec}(F) - \sum_{i=1}^p \hat{\vartheta}_{ii} \right\} \exp(T - p/2). \tag{A2}$$

Using the following definitions:

- $C(\vartheta) = -2I_{np} - 2\sum_{1 \le i \le j \le p} \vartheta_{ij}(J_{ij} + J_{ji})$. The matrix J_{ij} is composed of p^2 blocks. Block (i, j) is the identity I_n and all the other blocks are set to zero.
- $K(\vartheta) = -\frac{1}{2}\log|C(\vartheta)| - \frac{1}{2}\mu^\top C(\vartheta)^{-1}\mu - \frac{1}{2}\mathrm{vec}(\mu)^\top \mathrm{vec}(\mu)$. In this formula, μ is the $n \times p$ diagonal matrix with diagonal p singular values ω of F. The function $K(\vartheta)$ is the cumulant generating function.
- $\hat{\vartheta}$ is the unique solution of the so-called saddle-point equation $K'(\vartheta) = \vartheta$. It has the explicit expression $\hat{\vartheta} = -1/(2\mathrm{Diag}(\hat{\phi}))$, with $\hat{\phi}_r = \left(n + \sqrt{n^2 + 4\omega_r^2}\right)/(2\omega_r^2)$
- $\hat{C}$ is given by $C(\hat{\vartheta})$ and $\hat{K}''$ by $K''(\hat{\vartheta})$.
- $\hat{K}''$ can be computed explicitly:

$$\hat{K}''_{(r_1,s_1),(r_2,s_2)} = \begin{cases} 0 & r_1 \neq r_2 \text{ or } s_1 \neq s_2, \\ n\hat{\phi}_r\hat{\phi}_s + \hat{\phi}_r\hat{\phi}_s(\omega_r^2\hat{\phi}_r + \omega_s^2\hat{\phi}_s) & r_1 = r_2 < s_1 = s_2, \\ 2n\hat{\phi}_r^2 + 4\omega_r^2\hat{\phi}_r^3 & r_1 = r_2 = s_1 = s_2 \end{cases}$$

- The parameter T is defined in Equation (8) of [49] and computed in the supplementary material of the paper in the case of vMF distributions. In first approximation, T can be considered zero.

As in the original paper, we validate our implementation by comparing the result with the Monte Carlo estimate of the normalizing constant produced by uniform sampling on the Stiefel manifold.

Remark A1. *The $-p/2$ factor comes from using $B = -I_{n\times p}/2$ (and thus $V = I_{n\times p}$) and compensating with Equation (22) of [49] to handle vMF distributions, which otherwise have $B = 0$. This point is not stated explicitly in the main text of the paper but it is explained in the related MATLAB implementation provided by the authors.*

Appendix B. Gradient Formulas

The MCMC-SAEM initialization heuristic, as well as the MALA transition kernel, require the gradients of the log-likelihood with respect to the latent variables. In this section, we compute these gradients for the model with Gaussian perturbation and the model with binary coefficients.

Appendix B.1. Model with Gaussian Perturbation

Consider the log-likelihood for the variables of only one subject (X, λ, A). Using the formula in Equation (A1), we can compute its gradients with respect to X and λ. For λ, it writes:

$$\nabla_\lambda \log p(X,\lambda,A;\theta) = -\left(\frac{1}{\sigma_\varepsilon^2} + \frac{1}{\sigma_\lambda^2}\right)\lambda + \frac{1}{\sigma^2}(x_i^\top A x_i)_{i=1}^p + \frac{1}{\sigma_\lambda^2}\mu\,,$$

with $(x_i)_{i=1}^p$ the columns of X. Similarly, the Euclidean gradient for X is given by

$$\nabla_X \log p(X,\lambda,A;\theta) = \frac{1}{\sigma_\varepsilon^2} AX\mathrm{Diag}(\lambda) + F + 4X\mathrm{Diag}(\lambda)X^\top X\mathrm{Diag}(\lambda)$$

Following Edelman et al. [30], the Riemannian gradient on the Stiefel manifold is then given by:

$$\nabla_X^{\mathcal{V}} \log p(X,\lambda,A;\theta) = \nabla_X \log p(X,\lambda,A;\theta) X^{(k)\top} - X\nabla_X \log p(X,\lambda,A;\theta)$$

Appendix B.2. Binary Model

Similarly, the log-likelihood gradients can be derived for the binary model. Let $\tilde{x}_i$ be the i-th *row* of X and $\odot$ denote the entrywise product. We have:

$$\begin{aligned}
\nabla_\lambda \log p(X,\lambda,A;\theta) &= -\frac{1}{\sigma_\lambda^2}(\lambda-\mu) + \sum_{i,j=1}^n [A_{ij}h(-(\lambda\cdot X)_{ij}) - (1-A_{ij})h((\lambda\cdot X)_{ij})](\tilde{x}_i \odot \tilde{x}_j) \\
\nabla_X \log p(X,\lambda,A;\theta) &= F + \sum_{i\neq j}[A_{ij}h(-(\lambda\cdot X)_{ij}) - (1-A_{ij})h((\lambda\cdot X)_{ij})]H_{ij} \\
&\quad + \sum_{i=1}^n [A_{ii}h(-(\lambda\cdot X)_{ii}) - (1-A_{ii})h((\lambda\cdot X)_{ii})]K_i\,.
\end{aligned}$$

In the latter formula, H_{ij} is a $n\times p$ matrix with zeros everywhere except the i-th row equal to $\lambda\odot\tilde{x}_j$ and the j-th row equal to $\lambda\odot\tilde{x}_i$. K_i is the $n\times p$ matrix with zeros everywhere except the i-th row equal to $2\lambda\odot\tilde{x}_i$.

Appendix C. Algorithm for the Clustering Model

We summarize in Algorithm A1 the procedure to estimate the MLE of a mixture model.

Algorithm A1: Maximum Likelihood Estimation of $\theta = (F, \mu, \sigma_\varepsilon, \sigma_\lambda, \pi)$ for the mixture model

```
Initialize $\theta_0$ and $S_0$.
Initialize $X_0$, $\lambda_0$ and $z_0$ using the K-Means algorithm.
for $t = 1$ to $T$ do
    if $(t \bmod 5) = 0$ then
        Align together the parameters $(F^c, \mu^c)_{c=1}^K$ of each cluster using Algorithm 2.
    end
    if $t \leq T/3$ and $(t \bmod 5) = 0$ then
        for $k = 1$ to $N$ do
            Use Algorithm 2 to align $X_t^{(k)}$ with $\pi_V\left(F_t^{z_t^{(k)}}\right)$.
            Permute $\lambda_t^{(k)}$ accordingly.
        end
    end
    Set $\widetilde{X}_0^{(k)} = X_t^{(k)}, \widetilde{\lambda}_0^{(k)} = \lambda_t^{(k)}, \widetilde{z}_0^{(k)} = z_t^{(k)}$
    for $\ell = 1$ to $n_{\mathrm{MCMC}}$ do
        for $k = 1$ to $N$ do
            Sample $\widetilde{X}_\ell^{(k)}$ from the Metropolis kernel $q_X(\cdot \mid \widetilde{X}_{\ell-1}^{(k)}, \widetilde{\lambda}_{\ell-1}^{(k)}, \widetilde{z}_{\ell-1}^{(k)}; \theta_t)$ targetting
             $p(X^{(k)} \mid A^{(k)}, \widetilde{\lambda}_{\ell-1}^{(k)}, \widetilde{z}_{\ell-1}^{(k)}; \theta_t)$.
            Sample $\widetilde{\lambda}_\ell^{(k)}$ from the Metropolis kernel $q_\lambda(\cdot \mid \widetilde{X}_\ell^{(k)}, \widetilde{\lambda}_{\ell-1}^{(k)}, \widetilde{z}_{\ell-1}^{(k)}; \theta_t)$ targetting
             $p(\lambda^{(k)} \mid A^{(k)}, \widetilde{X}_\ell^{(k)}, \widetilde{z}_{\ell-1}^{(k)}; \theta_t)$.
            Sample $\widetilde{z}_\ell^{(k)}$ from the distribution $p(z^{(k)} \mid A^{(k)}, \widetilde{X}_\ell^{(k)}, \widetilde{\lambda}_\ell^{(k)}; \theta_t)$.
        end
    end
    Set $X_{t+1}^{(k)} = \widetilde{X}_{n_{\mathrm{MCMC}}}^{(k)}, \lambda_{t+1}^{(k)} = \widetilde{\lambda}_{n_{\mathrm{MCMC}}}^{(k)}$ and $z_{t+1}^{(k)} = \widetilde{z}_{n_{\mathrm{MCMC}}}^{(k)}$.
    Update the sufficient statistics $\bar{S}_{t+1} = (1 - \alpha_t)\bar{S}_t + \alpha_t S(A, X_{t+1}, \lambda_{t+1})$.
    Compute $\pi_{t+1}$ using the proportion of samples $z_{t+1}^{(k)}$ belonging to each cluster.
    for $c = 1$ to $K$ do
        Compute $\mu_{t+1}^c$, $(\sigma_\varepsilon^c)_{t+1}$ and $(\sigma_\lambda^c)_{t+1}$ with Equation (5) using only the $k$ such that $z_{t+1}^{(k)} = c$.
        Compute $F_{t+1}^c$ by solving problem (6), using only the $k$ such that $z_{t+1}^{(k)} = c$.
    end
end
return $\theta_T$, $(X_t, \lambda_t, z_t)_{t=1}^T$
```

Appendix D. Symmetry of Von Mises–Fisher Distributions

Let F be the parameter of a von Mises–Fisher distribution. Let $\exp_X$ be the Riemannian exponential map at X. We have the following result:

Proposition A1. *Suppose that the columns of F are orthogonal. Let $M = \pi_V(F)$ be the mode of the vMF distribution and $D \in T_M\mathcal{V}_{n,p}$ a tangent vector at M. Then $p_{\mathrm{vMF}}(\exp_M(D)) = p_{\mathrm{vMF}}(\exp_M(-D))$, i.e., the vMF distribution is symmetric around its mode.*

Proof. Since the columns of F are orthogonal, we can write $F = M\Lambda$ with $M = \pi_V(F) \in \mathcal{V}_{n,p}$ and $\Lambda = \mathrm{Diag}(\lambda)$. Let $D \in T_M\mathcal{V}_{n,p}$. As proven in [30], the geodesic X_t starting at M with $X'(0) = D$ is then given by

$$X_t = (M, M_\perp)\exp(tK_M(D))I_{n,p}\,,$$

where exp is the matrix exponential, $M_\perp \in \mathcal{V}_{n,n-p}$ is such that $M^\top M_\perp = 0$ and $K_M(D)$ is skew-symmetric: $K_M(D)^\top = -K_M(D)$. Therefore, the von Mises–Fisher log-density along X_t writes as:

$$\begin{aligned}
\operatorname{Tr}(F^\top X_t) &= \operatorname{Tr}(\Lambda M^\top (M, M_\perp) \exp(tK_M(D)) I_{n,p}) \\
&= \operatorname{Tr}(\Lambda I_{p,n} \exp(tK_M(D)) I_{n,p}) \\
&= \operatorname{Tr}(I_{n,p}^\top \exp(tK_M(D))^\top I_{p,n}^\top \Lambda^\top) \\
&= \operatorname{Tr}(I_{p,n} \exp(tK_M(D)^\top) I_{n,p} \Lambda) \\
&= \operatorname{Tr}(\Lambda I_{p,n} \exp(-tK_M(D)) I_{n,p}) \\
&= \operatorname{Tr}(F^\top X_{-t})
\end{aligned}$$

Therefore the von Mises–Fisher density is symmetric around its mode. □

Appendix E. Additional Details on the UK Biobank Experiment

Appendix E.1. Impact of the Number p of Patterns

We perform the same experiment as in Section 5.2 with different numbers of patterns, $p \in \{2, 5, 10\}$, always running the MCMC-SAEM for 1000 iterations with 20 MCMC steps per SAEM iteration. We call the related models M2, M5, and M10. The normalized patterns of M2 and M10 are reproduced in Figures A1 and A2. The patterns of M2 correspond to patterns 1 and 2 of M5 and M10. Similarly, the patterns of M5 correspond to patterns 1 to 5 of M10. This result confirms that our model acts in a way comparable to PCA, selecting first the dominant patterns with the largest eigenvalues. Figure A3 compares the posterior means of $\lambda \cdot X$ given by M2, M5 and M10 for 5 subjects. Coherently, the approximation $\lambda^{(k)} \cdot X^{(k)}$ refines and gets closer to $A^{(k)}$ as p increases. Over the 1000 subjects, these posterior means achieve, respectively, 57% (±7%), 47% (±5%) and 35% (±4%) relative RMSE.

However, this observation does not assess whether higher values of p provide additional relevant features to represent the network structure. The following result illustrates this idea. We repeat the experiment of missing link MAP imputation on models M2 and M10. We find that both M2 and M10 yield a worse prediction than M5 on this task: model M2 gets 70% (±16%) rRMSE and M10 gets 76% (±16%) rRMSE, whereas model M5 gets 65% (±15%) rRMSE. While the prediction performance of M2 is expected to be worse than M5's, observing a worse prediction performance in M10 means that the information captured by the additional components does not help infer the network structure. As with PCA, the components with lesser amplitude are less relevant to perform regression tasks; this idea is at the core of Partial Least Square Regression [68].

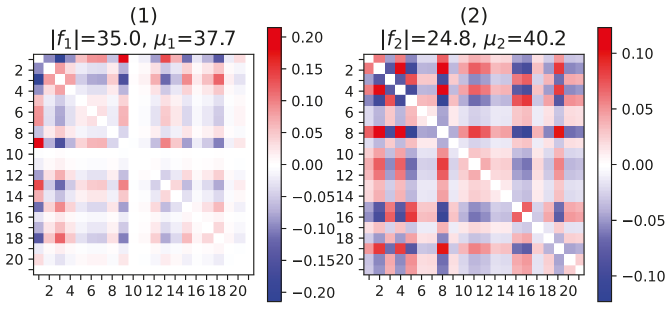


Figure A1. Normalized connectivity patterns when $p = 2$, computed as in Figure 10.

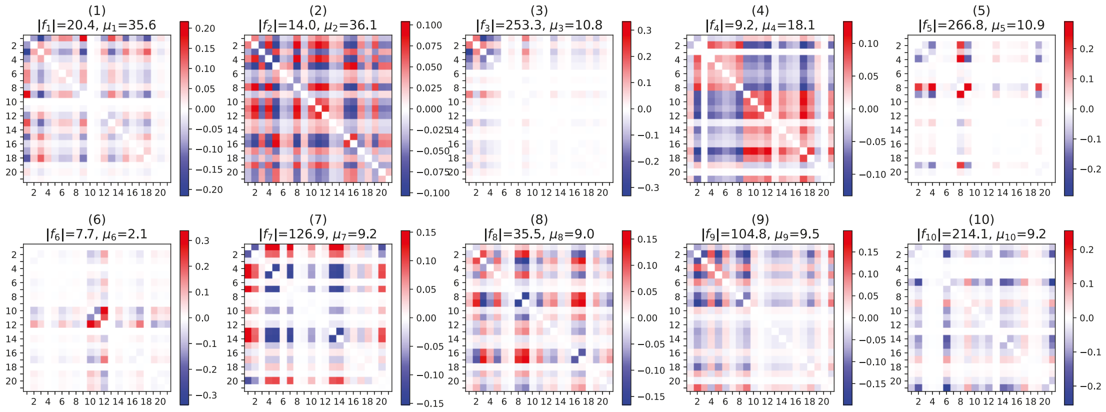


Figure A2. Normalized connectivity patterns when $p = 10$, computed as in Figure 10.

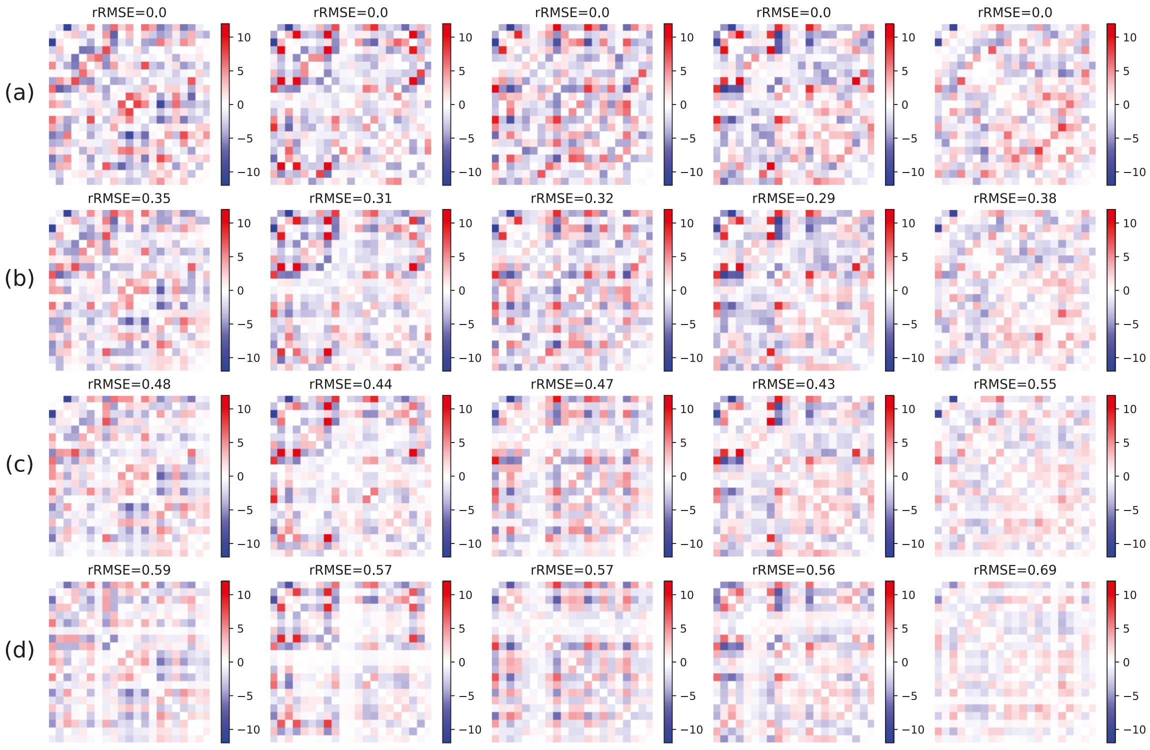


Figure A3. (**a**) UK Biobank connectivity matrices for 5 subjects. (**b**) M10 posterior mean value of $\lambda \cdot X$. (**c**) M5 posterior mean value of $\lambda \cdot X$. (**d**) M2 posterior mean value of $\lambda \cdot X$. The rRMSE coherently increases as p decreases.

Therefore, the parameter p should be chosen with care when using our model for predictive purposes. The experiment presented above can be used to quantify the relevance of the obtained representation, but other methods could be explored. Future work could investigate the question of parameter selection by adapting Bayesian model selection methods to our method, as well as likelihood ratio tests or criteria like BIC and AIC.

Appendix E.2. Brain Regions of the UK Biobank fMRI Correlation Networks

As explained in Section 5.2, the Regions Of Interest (ROIs) that define the correlation networks are detected automatically using a spatial ICA [59]. Each component of the ICA attributes a weight to each brain voxel. The brain regions are visualized by selecting the voxels with weight above a certain threshold. The obtained level set may be scattered over the brain, which sometimes makes their interpretation difficult. In Figure A4, we show the brain regions mentioned in the interpretation of the patterns identified by our model, namely regions 1, 2, 3, 4, 8, 9, 10, 11, 12, 17, 19. In this figure, as well as online, the ICA weight threshold value is set to 5.

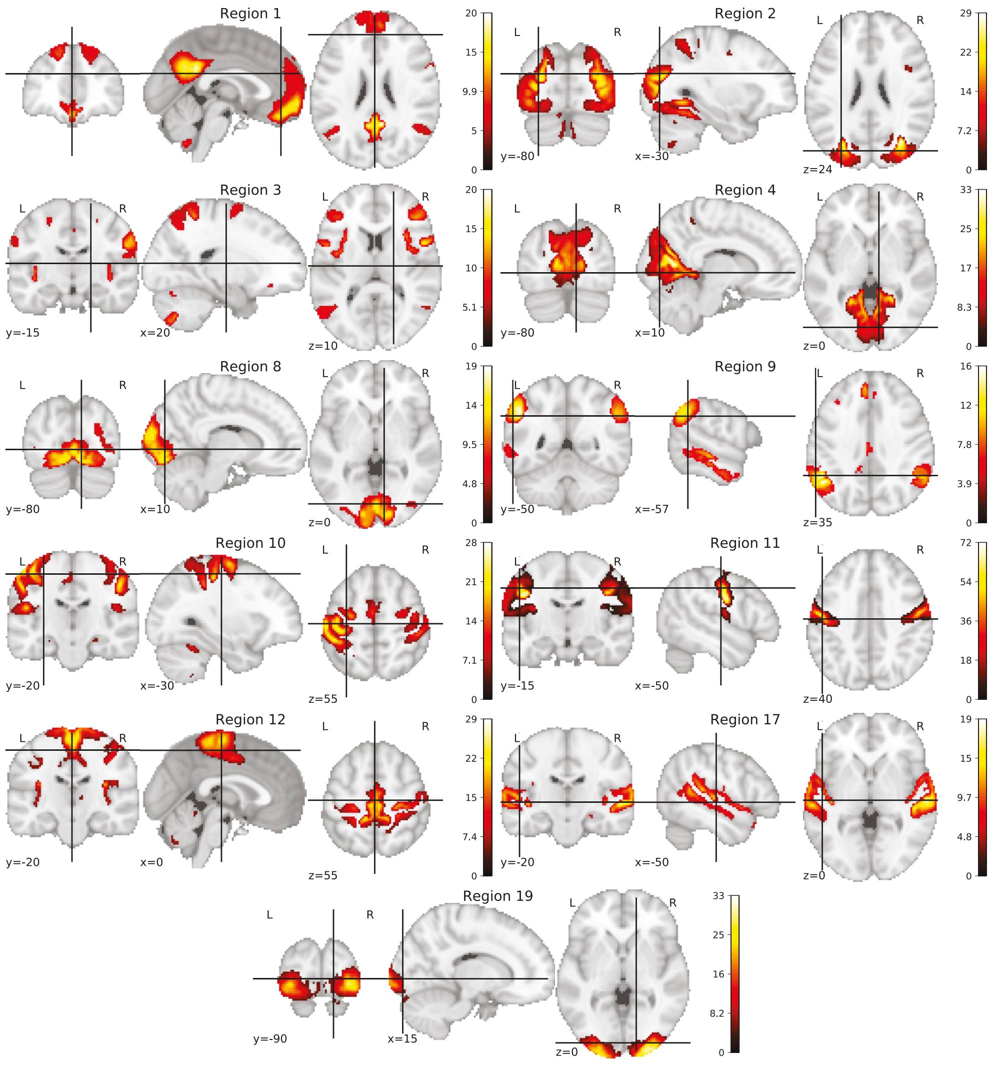


Figure A4. Frontal, sagittal, and transverse cuts of the brain for the UK Biobank fMRI brain regions analyzed in this paper. As explained in Section 5.2, region 1 comprises part of the Default Mode Network of the brain, which characterizes its activity at rest. Region 3, which is anti-correlated to region 1, is related to sensory functions. Regions 2, 4, 8, 9, and 19 are involved in the visual functions. Regions 10, 11, 12 correspond to motor control. Region 17 is involved in memory and spatial navigation. The L/R letters distinguish the left and right hemispheres. The black axes on each view give the three-dimensional position of the cut. The color strength corresponds to the truncated ICA weight.

References

1. Newman, M.E.J. *Networks—An Introduction*; Oxford University Press: Oxford, UK, 2012.
2. Ni, C.C.; Lin, Y.Y.; Luo, F.; Gao, J. Community Detection on Networks with Ricci Flow. *Sci. Rep.* **2019**, *9*, 9984. [CrossRef]
3. Martínez, V.; Berzal, F.; Cubero, J.C. A Survey of Link Prediction in Complex Networks. *ACM Comput. Surv.* **2016**, *49*, 1–33. [CrossRef]
4. Shen, X.; Finn, E.S.; Scheinost, D.; Rosenberg, M.D.; Chun, M.M.; Papademetris, X.; Constable, R.T. Using Connectome-Based Predictive Modeling to Predict Individual Behavior from Brain Connectivity. *Nat. Protoc.* **2017**, *12*, 506–518. [CrossRef]
5. Banks, D.; Carley, K. Metric Inference for Social Networks. *J. Classif.* **1994**, *11*, 121–149. [CrossRef]
6. Rubinov, M.; Sporns, O. Complex Network Measures of Brain Connectivity: Uses and Interpretations. *NeuroImage* **2010**, *52*, 1059–1069. [CrossRef] [PubMed]
7. Simonovsky, M.; Komodakis, N. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. In *Artificial Neural Networks and Machine Learning—ICANN 2018*; Lecture Notes in Computer Science; Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 412–422._41. [CrossRef]
8. Pozzi, F.A.; Fersini, E.; Messina, E.; Liu, B. *Sentiment Analysis in Social Networks*; Morgan Kaufmann: Burlington, MA, USA, 2016.
9. Monti, F.; Bronstein, M.; Bresson, X. Geometric Matrix Completion with Recurrent Multi-Graph Neural Networks. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 3697–3707.
10. Narayanan, T.; Subramaniam, S. Community Structure Analysis of Gene Interaction Networks in Duchenne Muscular Dystrophy. *PLoS ONE* **2013**, *8*. [CrossRef] [PubMed]
11. He, Y.; Evans, A. Graph Theoretical Modeling of Brain Connectivity. *Curr. Opin. Neurol.* **2010**, *23*, 341–350. doi:10.1097/WCO.0b013e32833aa567. [CrossRef]
12. Duvenaud, D.K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R.P. Convolutional Networks on Graphs for Learning Molecular Fingerprints. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 2224–2232.
13. Lovász, L. *Large Networks and Graph Limits*; Colloquium Publications; American Mathematical Society: Providence, RI, USA, 2012; Volume 60. [CrossRef]
14. Hanneke, S.; Fu, W.; Xing, E.P. Discrete Temporal Models of Social Networks. *Electron. J. Stat.* **2010**, *4*, 585–605. doi:10.1214/09-EJS548. [CrossRef]
15. Fornito, A.; Zalesky, A.; Bullmore, E. *Fundamentals of Brain Network Analysis*; Academic Press: Cambridge, MA, USA, 2016.
16. Zheng, W.; Yao, Z.; Li, Y.; Zhang, Y.; Hu, B.; Wu, D.; Alzheimer's Disease Neuroimaging Initiative. Brain Connectivity Based Prediction of Alzheimer's Disease in Patients With Mild Cognitive Impairment Based on Multi-Modal Images. *Front. Hum. Neurosci.* **2019**, *13*. [CrossRef]
17. Ghosh, S.; Das, N.; Gonçalves, T.; Quaresma, P.; Kundu, M. The Journey of Graph Kernels through Two Decades. *Comput. Sci. Rev.* **2018**, *27*, 88–111. [CrossRef]
18. Damoiseaux, J.S. Effects of Aging on Functional and Structural Brain Connectivity. *NeuroImage* **2017**, *160*, 32–40. [CrossRef] [PubMed]
19. Chikuse, Y. *Statistics on Special Manifolds*; Lecture Notes in Statistics; Springer: New York, NY, USA, 2003; doi:10.1007/978-0-387-21540-2. [CrossRef]
20. Harris, J.K. *An Introduction to Exponential Random Graph Modeling*; Number 173 in Quantitative Applications in the Social Sciences; SAGE: Thousand Oaks, CA, USA, 2014.
21. Erdős, P.; Rényi, A. On Random Graphs. *Publ. Math.* **1959**, *6*, 290–297.
22. Peixoto, T.P. Bayesian Stochastic Blockmodeling. In *Advances in Network Clustering and Blockmodeling*; Doreian, P., Batagelj, V., Ferligoj, A., Eds.; Wiley Series in Computational and Quantitative Social Science; Wiley: New York, NY, USA, 2020; pp. 289–332. Available online: http://arxiv.org/abs/1705.10225 (accessed on 19 April 2021).
23. Chandna, S.; Maugis, P.A. Nonparametric Regression for Multiple Heterogeneous Networks. *arXiv* **2020**, arXiv:2001.04938.
24. Zhang, Z.; Cui, P.; Zhu, W. Deep Learning on Graphs: A Survey. *arXiv* **2020**, arXiv:1812.04202.
25. Banka, A.; Rekik, I. Adversarial Connectome Embedding for Mild Cognitive Impairment Identification Using Cortical Morphological Networks. In *Connectomics in NeuroImaging*; Lecture Notes in Computer Science; Schirmer, M.D., Venkataraman, A., Rekik, I., Kim, M., Chung, A.W., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 74–82. [CrossRef]
26. Ma, J.; Zhu, X.; Yang, D.; Chen, J.; Wu, G. Attention-Guided Deep Graph Neural Network for Longitudinal Alzheimer's Disease Analysis. In *Medical Image Computing and Computer Assisted Intervention —MICCAI 2020*; Martel, A.L., Abolmaesumi, P., Stoyanov, D., Mateus, D., Zuluaga, M.A., Zhou, S.K., Racoceanu, D., Joskowicz, L., Eds.; Springer International Publishing: Cham, Switzerland, 2020; Volume 12267, pp. 387–396._38. [CrossRef]
27. Westveld, A.H.; Hoff, P.D. A Mixed Effects Model for Longitudinal Relational and Network Data, with Applications to International Trade and Conflict. *Ann. Appl. Stat.* **2011**, *5*, 843–872. [CrossRef]

28. D'Souza, N.S.; Nebel, M.B.; Wymbs, N.; Mostofsky, S.; Venkataraman, A. Integrating Neural Networks and Dictionary Learning for Multidimensional Clinical Characterizations from Functional Connectomics Data. In *Medical Image Computing and Computer Assisted Intervention—MICCAI 2019*; Shen, D., Liu, T., Peters, T.M., Staib, L.H., Essert, C., Zhou, S., Yap, P.T., Khan, A., Eds.; Springer International Publishing: Cham, Switzerland, 2019; Volume 11766, pp. 709–717._79. [CrossRef]
29. Liu, M.; Zhang, Z.; Dunson, D.B. Auto-Encoding Graph-Valued Data with Applications to Brain Connectomes. *arXiv* **2019**, arXiv:1911.02728.
30. Edelman, A.; Arias, T.A.; Smith, S.T. The Geometry of Algorithms with Orthogonality Constraints. *SIAM J. Matrix Anal. Appl.* **1998**, *20*, 303–353. [CrossRef]
31. Zimmermann, R. A Matrix-Algebraic Algorithm for the Riemannian Logarithm on the Stiefel Manifold under the Canonical Metric. *SIAM J. Matrix Anal. Appl.* **2017**, *38*, 322–342. [CrossRef]
32. Khatri, C.G.; Mardia, K.V. The von Mises–Fisher Matrix Distribution in Orientation Statistics. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 95–106. [CrossRef]
33. Karush, W. Minima of Functions of Several Variables with Inequalities as Side Constraints. Masters's Thesis, Department of Mathematics, University of Chicago, Chicago, IL, USA, 1939.
34. Kuhn, E.; Lavielle, M. Coupling a Stochastic Approximation Version of EM with an MCMC Procedure. *ESAIM Probab. Stat.* **2004**, *8*, 115–131.:2004007. [CrossRef]
35. Hoff, P.D.; Ward, M.D. Modeling Dependencies in International Relations Networks. *Political Anal.* **2004**, *12*, 160–175. [CrossRef]
36. Hoff, P.D. Modeling Homophily and Stochastic Equivalence in Symmetric Relational Data. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*; NIPS'07; Curran Associates Inc.: Red Hook, NY, USA, 2007; pp. 657–664.
37. Hoff, P.D. Model Averaging and Dimension Selection for the Singular Value Decomposition. *J. Am. Stat. Assoc.* **2007**, *102*, 674–685. [CrossRef]
38. Hoff, P.D. Simulation of the Matrix Bingham—von Mises—Fisher Distribution, With Applications to Multivariate and Relational Data. *J. Comput. Graph. Stat.* **2009**, *18*, 438–456. [CrossRef]
39. Chen, J.; Han, G.; Cai, H.; Ma, J.; Kim, M.; Laurienti, P.; Wu, G. Estimating Common Harmonic Waves of Brain Networks on Stiefel Manifold. In *Medical Image Computing and Computer Assisted Intervention—MICCAI 2020*; Lecture Notes in Computer Science; Martel, A.L., Abolmaesumi, P., Stoyanov, D., Mateus, D., Zuluaga, M.A., Zhou, S.K., Racoceanu, D., Joskowicz, L., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 367–376._36. [CrossRef]
40. Allassonnière, S.; Younes, L. A Stochastic Algorithm for Probabilistic Independent Component Analysis. *Ann. Appl. Stat.* **2012**, *6*, 125–160. [CrossRef]
41. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum Likelihood from Incomplete Data Via the EM Algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 1–22. [CrossRef]
42. Allassonnière, S.; Kuhn, E.; Trouvé, A. Construction of Bayesian Deformable Models via a Stochastic Approximation Algorithm: A Convergence Study. *Bernoulli* **2010**, *16*, 641–678. [CrossRef]
43. Debavelaere, V.; Durrleman, S.; Allassonnière, S. On the Convergence of Stochastic Approximations under a Subgeometric Ergodic Markov Dynamic. *Electron. J. Stat.* **2021**, *15*, 1583–1609.
44. Hastings, W.K. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika* **1970**, *57*, 97–109. [CrossRef]
45. Robert, C.P.; Casella, G. *Monte Carlo Statistical Methods*; Springer: New York, NY, USA, 2010.
46. Li, J.; Fuxin, L.; Todorovic, S. Efficient Riemannian Optimization On The Stiefel Manifold Via The Cayley Transform. *arXiv* **2020**, arXiv:2002.01113.
47. Pal, S.; Sengupta, S.; Mitra, R.; Banerjee, A. Conjugate Priors and Posterior Inference for the Matrix Langevin Distribution on the Stiefel Manifold. *Bayesian Anal.* **2020**, *15*, 871–908. [CrossRef]
48. Jupp, P.E.; Mardia, K.V. Maximum Likelihood Estimators for the Matrix Von Mises-Fisher and Bingham Distributions. *Ann. Stat.* **1979**, *7*, 599–606. [CrossRef]
49. Kume, A.; Preston, S.P.; Wood, A.T.A. Saddlepoint Approximations for the Normalizing Constant of Fisher–Bingham Distributions on Products of Spheres and Stiefel Manifolds. *Biometrika* **2013**, *100*, 971–984. [CrossRef]
50. Ali, M.; Gao, J. Classification of Matrix-Variate Fisher–Bingham Distribution via Maximum Likelihood Estimation Using Manifold Valued Data. *Neurocomputing* **2018**, *295*, 72–85. [CrossRef]
51. Butler, R.W. *Saddlepoint Approximations with Applications*; Cambridge University Press: Cambridge, UK, 2007.
52. Debavelaere, V.; Durrleman, S.; Allassonnière, S.; Alzheimer's Disease Neuroimaging Initiative. Learning the Clustering of Longitudinal Shape Data Sets into a Mixture of Independent or Branching Trajectories. *Int. J. Comput. Vis.* **2020**, *128*, 2794–2809. [CrossRef]
53. Lam, S.K.; Pitrou, A.; Seibert, S. Numba: A LLVM-Based Python JIT Compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*; LLVM '15; Association for Computing Machinery: New York, NY, USA, 2015; pp. 1–6. [CrossRef]
54. Kaneko, T.; Fiori, S.; Tanaka, T. Empirical Arithmetic Averaging Over the Compact Stiefel Manifold. *IEEE Trans. Signal Process.* **2013**, *61*, 883–894. [CrossRef]
55. Lu, L.; Zhou, T. Link Prediction in Complex Networks: A Survey. *Phys. A Stat. Mech. Its Appl.* **2011**, *390*, 1150–1170. [CrossRef]
56. Zhang, M.; Chen, Y. Link Prediction Based on Graph Neural Networks. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 5171–5181.

57. Celeux, G.; Frühwirth-Schnatter, S.; Robert, C.P. Model Selection for Mixture Models—Perspectives and Strategies. In *Handbook of Mixture Analysis*, 1st ed.; Frühwirth-Schnatter, S., Celeux, G., Robert, C.P., Eds.; Chapman and Hall: London, UK; CRC Press: Boca Raton, FL, USA, 2019; pp. 117–154. [CrossRef]
58. Sudlow, C.; Gallacher, J.; Allen, N.; Beral, V.; Burton, P.; Danesh, J.; Downey, P.; Elliott, P.; Green, J.; Landray, M.; et al. UK Biobank: An Open Access Resource for Identifying the Causes of a Wide Range of Complex Diseases of Middle and Old Age. *PLoS Med.* **2015**, *12*, e1001779. [CrossRef] [PubMed]
59. Kiviniemi, V.; Kantola, J.H.; Jauhiainen, J.; Hyvärinen, A.; Tervonen, O. Independent Component Analysis of Nondeterministic fMRI Signal Sources. *NeuroImage* **2003**, *19*, 253–260. [CrossRef]
60. Horn, A.; Ostwald, D.; Reisert, M.; Blankenburg, F. The Structural–Functional Connectome and the Default Mode Network of the Human Brain. *NeuroImage* **2014**, *102*, 142–151. [CrossRef] [PubMed]
61. Eysenck, M.W. *Cognitive Psychology: A Student's Handbook*; Psychology Press: New York, NY, USA, 2010.
62. Purves, D.; Augustine, G.J.; Fitzpatrick, D.; Hall, W.C.; LaMantia, A.S.; Mooney, R.D.; Platt, M.L.; White, L.E. (Eds.) *Neuroscience*, 6th ed.; Sinauer Associates is an Imprint of Oxford University Press: New York, NY, USA, 2017.
63. Nguyen, L.T.; Kim, J.; Shim, B. Low-Rank Matrix Completion: A Contemporary Survey. *IEEE Access* **2019**, *7*, 94215–94237. [CrossRef]
64. Kume, A.; Sei, T. On the Exact Maximum Likelihood Inference of Fisher–Bingham Distributions Using an Adjusted Holonomic Gradient Method. *Stat. Comput.* **2018**, *28*, 835–847. [CrossRef]
65. Schiratti, J.B.; Allassonniere, S.; Colliot, O.; Durrleman, S. Learning Spatiotemporal Trajectories from Manifold-Valued Longitudinal Data. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 2404–2412.
66. Hammond, D.K.; Vandergheynst, P.; Gribonval, R. Wavelets on Graphs via Spectral Graph Theory. *Appl. Comput. Harmon. Anal.* **2011**, *30*, 129–150. [CrossRef]
67. Atasoy, S.; Donnelly, I.; Pearson, J. Human Brain Networks Function in Connectome-Specific Harmonic Waves. *Nat. Commun.* **2016**, *7*. [CrossRef]
68. Wold, S.; Sjöström, M.; Eriksson, L. PLS-Regression: A Basic Tool of Chemometrics. *Chemom. Intell. Lab. Syst.* **2001**, *58*, 109–130. [CrossRef]

Article

“Exact” and Approximate Methods for Bayesian Inference: Stochastic Volatility Case Study

Yuliya Shapovalova

Institute for Computing and Information Sciences, Radboud University Nijmegen, Toernooiveld 212, 6525 EC Nijmegen, The Netherlands; yuliya.shapovalova@ru.nl

Abstract: We conduct a case study in which we empirically illustrate the performance of different classes of Bayesian inference methods to estimate stochastic volatility models. In particular, we consider how different particle filtering methods affect the variance of the estimated likelihood. We review and compare particle Markov Chain Monte Carlo (MCMC), RMHMC, fixed-form variational Bayes, and integrated nested Laplace approximation to estimate the posterior distribution of the parameters. Additionally, we conduct the review from the point of view of whether these methods are (1) easily adaptable to different model specifications; (2) adaptable to higher dimensions of the model in a straightforward way; (3) feasible in the multivariate case. We show that when using the stochastic volatility model for methods comparison, various data-generating processes have to be considered to make a fair assessment of the methods. Finally, we present a challenging specification of the multivariate stochastic volatility model, which is rarely used to illustrate the methods but constitutes an important practical application.

Keywords: Bayesian inference; Markov Chain Monte Carlo; Sequential Monte Carlo; Riemann Manifold Hamiltonian Monte Carlo; integrated nested laplace approximation; fixed-form variational Bayes; stochastic volatility

Citation: Shapovalova, Y. ‘Exact’ and Approximate Methods for Bayesian Inference: Stochastic Volatility Case Study. *Entropy* **2021**, *23*, 466. https://doi.org/10.3390/e23040466

Academic Editor: Pierre Alquier

Received: 28 February 2021
Accepted: 12 April 2021
Published: 15 April 2021

1. Introduction

The field of Bayesian statistics and machine learning has advanced in recent years quite rapidly. The methods that have been developed do not often find fast assimilation across different fields. In this review, we aim to provide the reader with methodologies that try to solve the estimation problem in models with latent variables and intractable likelihoods. We are in particular interested in the methods that can be used to estimate nonlinear state-space models and in particular stochastic (latent) volatility models. There are multiple studies that conducted review and comparison of the methods of estimation of the stochastic volatility models [1–3]. We briefly mention some of the methods that have been reviewed; however, most of the methods considered in this paper have not entered those reviews. In this paper, we focus in particular on comparing methods that target posterior distribution exactly and the methods that try to approximate it. We also conduct the review from the point of view of estimating multivariate models with these methods and discuss what the bottleneck is in each of them when extending to higher-dimensional stochastic volatility (SV) models. We consider different data-generating processes for simulating data in the empirical studies and conclude that the choice of the data-generating process can heavily affect performance of a method. Thus, illustrating the performance of a method on just one data generating process or one real-world data set is not sufficient.

In financial econometrics literature, GARCH-type models prevail since they are much simpler to estimate. Stochastic (latent) volatility models, however, can be more natural frameworks for modeling asset returns. They can provide flexible and intuitive tools for applications in financial econometrics as well as some other disciplines. In particular, multivariate stochastic volatility models offer an attractive framework for detection and measuring *volatility spillover effects*. Volatility spillovers in this framework can be defined through Granger-causal links in the latent (unobservable) volatility process, which is modeled with a Vector Autoregressive model (VAR(p)). Insights about the causal structure can help to identify the relationship (Granger-causality or/and contemporaneous correlation) between the financial markets. Such information can be insightful and helpful in the decision-making process of portfolio managers and policymakers. These models are, however, rarely considered in practice. Multiple Bayesian inference methods have been proposed for the estimation of this class of models in recent years. In this paper, we identify the bottlenecks in different classes of methods for the estimation of these models in the multivariate case.

One of the stepping stones of estimation of the nonlinear state-space models in general (and stochastic volatility models in particular) lies in the intractability of the likelihood, which is the result of the presence of an unobservable process in the model and nonlinear dependence between this process and the observed data. The likelihood can be estimated with particle filter methods, also known as Sequential Monte Carlo. This is a computationally intensive procedure; however, depending on the problem and the data, it can provide excellent results. The second stepping stone of the estimation is the intractable posterior distribution. A standard starting point for sampling from the posterior distribution is the Metropolis-Hastings algorithm, which is a general method and can be applied straightforwardly to different models. It works well when the number of parameters in the model is small. However, the convergence of the algorithm can be slow in larger models, due to inefficiency of the sampling with random walk proposals. Particle Metropolis-Hastings [4] combines Sequential Monte Carlo for the likelihood estimation with Metropolis-Hastings for the sampling from the posterior, which results in a state-of-the-art method in terms of the estimation quality since it targets the exact posterior. The downside of this method is that it is computationally extremely demanding. Note that, while we consider particle Metropolis-Hastings in this paper, the class of methods from [4] is more general.

Two main downsides of particle Metropolis-Hastings are random walk behavior of the proposals and computational burden. One of the possible solutions to the first problem are the algorithms that use gradient information for the construction of the proposal distribution and thus explore the parameter space more efficiently. An additional step in improving these algorithms is defining them on a Riemann manifold instead of Euclidean space as proposed in [5]. The resulting algorithm, which we consider for the comparison in this paper, is Riemann Manifold Hamiltonian Monte Carlo. For extensive comparison of the methods that exploit gradient information and Langevin dynamics—such as Metropolis-adjusted Langevin algorithm, Hamiltonian Monte Carlo, Riemann manifold Metropolis adjusted Langevin algorithm, andRiemann Manifold Hamiltonian Monte Carlo—we refer to [5].

Thus far, we have discussed the methods that target the posterior distribution exactly and have a high computational burden, which makes empirical investigation of their performance in high-dimensional cases infeasible. In the last decade, a large number of methods have been published on approximate posterior inference thaat allow much faster computations, but lose in terms of precision of the estimation. In this paper, we consider two such methods that deal with different types of approximation. Fixed-form variational Bayes, proposed in [6], assumes hierarchical factorization of the prior and posterior distributions, and the factorized distributions are approximated by an analytically tractable distribution from a certain family of distributions $q(\cdot)$. Then, instead of solving integration problem, one solves the optimization problem of minimizing the Kullback–Leibler divergence between $q(\cdot)$ and $p(\cdot)$, where $p(\cdot)$ is the target distribution. The second approximate method that we consider is the integrated nested Laplace approximation (INLA) [7]. The method relies on the nested version of the classical Laplace approximation. It became very popular in recent years and made computations in many models feasible.

In this review paper, we focus our attention on the following methodologies and provide a comparison for some of the methods via a simulation study. We consider how the variance of the estimated likelihood is affected by choosing different particle-filtering algorithms. Unlike previous studies, we consider the variance of the estimated likelihood over the whole parameter space and notice that it is affected by some parameters of the model more than by the others. We compare particle Metropolis-Hastings with Riemann Manifold Hamiltonian Monte Carlo as two state-of-the-art sampling methods for this type of problem. We asses how well the INLA method performs in the task of the estimation of the parameters of stochastic volatility model and finally, compare fixed-form variational Bayes methods with sampling by RMHMC. All the between-methods comparisons are performed on multiple simulated data sets with different underlying parameters. We illustrate that, for fair comparison and performance assessment, illustration only on data sets is not sufficient.

The paper is organized as follows. In Section 2, we introduce the model and its different specifications. While in simulation studies we use univariate model, we do introduce multivariate stochastic volatility models with Granger-causal feedback as the model of interest for high-dimensional inference. In Section 3, we review the methods that can be used for the estimation of this class of models. We introduce major ideas behind these methods, and for the details of the derivations we refer to the original papers. In Section 4, we perform empirical case study on different simulated data sets and compare the methods on two real-world time series. We in particular focus on the precision loss of parameter estimation when using approximate methods and how adaptable the methods are to perform multivariate estimation and estimation of various model specifications.

2. Model

2.1. Univariate Stochastic Volatility Model

In this section, we introduce the model of interest that we will use in the simulation studies. Stochastic volatility (SV) models are concerned with modeling asset prices or asset returns depending on how the model is formulated. Let P_t be the price of the asset at time t or the exchange rate at time t (we consider two applications to real data in Section 3.5: one to exchange rate and one to log-returns). Then the log-return y_t is

$$y_t = \log(1 + R_t) = \log \frac{P_t}{P_{t-1}}. \tag{1}$$

Stochastic volatility models are built in such a way that they can mimic *stylized facts* about financial markets and log-returns y_t. Stylized facts are empirically observed statistical properties of asset prices and asset returns. Typical examples of stylized facts are

- *Volatility clustering and persistence*: the big changes in asset returns tend to be followed by big changes, and small changes in asset returns tend to be followed by small changes; in other words, there are periods of large fluctuations and small fluctuations [8].
- *Leverage effect*: the changes in stock prices may be negatively related to the changes in volatility [9].
- *Co-movements:* different stocks tend to exhibit co-movements, which means that if the volatility of one stock changes in a specific direction, volatilities of the other stocks tend to change in the same direction [9].

One of the earlier works that received much attention in the financial literature and proposed a mathematical model that tried to explain the dynamics of financial markets is [10]. Numerous continuous-time stochastic volatility models have been proposed since then, and among the first ones, multiple variants should be mentioned [11–14]. The model we will be considering in this chapter can be viewed as a discrete version of the model in [13] derived by using Euler–Maruyama approximation. The stochastic volatility model in continuous time can be written as

$$ds(t) = \sigma(t) dB_1(t), \tag{2}$$

$$\ln \sigma^2(t) = \mu + \beta \ln \sigma^2(t) dt + \sigma_\eta dB_2(t), \tag{3}$$

where $s(t)$ is log of asset price, $\sigma^2(t)$ is the volatility, $B_1(t)$ and $B_2(t)$ are Brownian motions that satisfy $corr(B_1(t), B_2(t)) = \rho$. If $\rho < 0$, there is leverage effect present. Thus, log of asset price follows diffusion and its volatility parameter also follows diffusion [15]. As we often get the data in discrete time, usually the discrete time approximation of the model is used in practice. The discrete model then follows by using Euler–Maruyama approximation

$$y_t = \sigma_t \epsilon_t, \tag{4}$$
$$\ln \sigma^2_{t+1} = \mu + \phi \ln \sigma^2_t + \sigma_\eta \eta_{t+1}, \tag{5}$$

where y_t is logarithmic return, $\epsilon_t = B_1(t+1) - B_1(t)$, $\eta_{t+1} = B_2(t+1) - B_2(t)$, $\phi = 1 + \beta$. Further, $\epsilon_t \sim N(0,1)$ and $\eta_t \sim N(0,1)$, $corr(\epsilon_t, \eta_{t+1}) = \rho$.

We get state-space representation of the model that is commonly used by defining $h_t = \ln \sigma^2_t$ and $\sigma^2_t = \exp(h_2)$

$$y_t = \exp(h_t/2)\epsilon_t, \tag{6}$$
$$h_{t+1} = \mu + \phi h_t + \eta_{t+1}, \tag{7}$$

where y_t are log-returns that are observed and volatility h_t is latent and drives the dynamics of y_t. Figure 1 illustrates this structure of the model. Note that the latent volatility process has an autoregressive form. However, unlike in the standard autoregressive model, the latent volatility is not observed and thus has to be estimated together with the model parameters μ, ϕ, and σ_η, which are the scale, the volatility persistence and the noise variance of the latent volatility process, respectively. The persistence parameter ϕ reflects one of the stylized facts of financial returns, namely volatility persistence. The intuition is as follows: if $\phi > 0$ and $\exp(h_{t-1}/2)$ is large, then $\exp(h_t/2)$ will tend to be large too. Hence, the model can account for volatility clustering. In this paper, we consider stationary volatility cases with $|\phi| < 1$. Finally, one can also incorporate leverage effects by defining negative correlation between noise terms ϵ_t and η_{t+1}. Intuitive interpretation of the leverage effect goes as follows: bad news tends to decrease the price, which means that financial leverage increases, the firm becomes riskier, and thus expected volatility also increases. The leverage effects in this model have been studied in [16]. The stochastic volatility model can be parametrized in multiple ways; often, the following alternatives are considered [2]. Other ways to parametrize this model are presented in Equarions (8) and (9). The left-hand side version of the model corresponds to that of [17]. The right-hand side version is a different way to define the scaling parameter; in this case, it is β. For identifiability reasons, only β or μ as in Equation (7) should be included in the model.

$$y_t = \sqrt{h_t}\epsilon_t \qquad\qquad y_t = \beta \exp(h_t/2)\epsilon_t \tag{8}$$
$$\log h_t = \mu + \phi \log h_{t-1} + \eta_t \qquad\qquad h_t = \phi h_{t-1} + \eta_t. \tag{9}$$

Note that the authors of [17] define the leverage effect as correlation between ϵ_t and η_t, so the correlation between noise terms is contemporaneous while [16] model correlation between ϵ_t and η_{t+1}, which corresponds to correlation of the returns with one-step-ahead volatility. Reference Yu [18] shows that the approach of [16] is preferable. In particular, while in case of [16] the model is a martingale difference sequence, i.e., the past does not help to predict the future of the time series, in the case of [17], it is not. Hence, in the latter case, the efficient market hypothesis is violated.

In the remainder of this manuscript, we will work with either specification of the model defined in Equations (6) and (7) or in right-hand side of Equations (8) and (9). These models are equivalent, and we interchange the representation either for the convenience of using some of the methods or for comparison with other work. In the literature, both specifications are frequently used, and in some papers (for example, ref. [19]) the transition from one specification to another is conducted by observing that $\beta = \exp(\mu/2)$.

Under the assumption that $|\phi| < 1$, the unconditional first and second moments of the latent process h_t are

$$\mathbb{E}(h_t) = \frac{\mu}{1-\phi}, \qquad Var(h_t) = \frac{\sigma_\eta^2}{1-\phi^2}. \tag{10}$$

The challenge of the estimation of the model lies in the intractability of the likelihood and posterior distribution. The likelihood factorizes as

$$L(y|\theta) = \prod_{t=1}^{T} p(y_t|y_{1:t-1}, \theta), \tag{11}$$

where the terms in the product can be computed recursively, and it becomes clear that the likelihood is a high-dimensional integral

$$p(y_t|y_{1:t-1}, \theta) = \int p(y_t|h_t, \theta)p(h_t|y_{1:t-1}, \theta)dh_t. \tag{12}$$

There is no analytical solution to the integral in Equation (12), and in this paper, we consider methods to estimate it using sequential Monte Carlo methods.

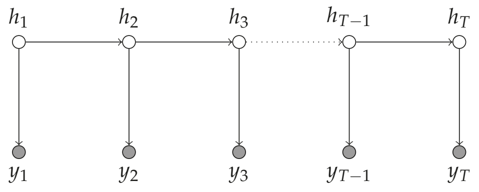


Figure 1. Graphical representation of stochastic volatility model. Observations y_t represented by shaded edges depend at each time point on the state of the latent volatility process h_t.

2.2. Multivariate Stochastic Volatility Model

In this section, we introduce the multivariate stochastic volatility model, which is rarely used in practice due to the challenges of estimation. One of the objectives of this paper is to assess whether modern methods in Bayesian inference are capable of the estimation of these models in high-dimensional case. Multivariate or high dimensional application of this class of models can give insightful information to practitioners. We deal with the same set-up as before; however, we now consider multiple time series of logarithmic returns that are interconnected through the latent volatility process

$$\boldsymbol{y}_t = \boldsymbol{\Omega}_t \boldsymbol{\epsilon}_t, \tag{13}$$

where $\boldsymbol{\epsilon}_t \sim N(\boldsymbol{0}, \boldsymbol{R})$ and $\boldsymbol{R}$ is a correlation matrix with entries $r_{ii} = 1, i = 1, \ldots, n$ on the diagonal. Furthermore, $\boldsymbol{\Omega}_t$ is a diagonal matrix that contains time-varying volatilities that are driven by an independent stochastic process $\boldsymbol{h}_t$,

$$\boldsymbol{\Omega}_t = diag(\exp(\boldsymbol{h}_t/2)).$$

The process $\boldsymbol{h}_t$ of log-volatilities follows a VAR(p) process

$$\boldsymbol{h}_t = \boldsymbol{\mu} + \sum_{k=1}^{p} \Phi_k (\boldsymbol{h}_{t-i} - \boldsymbol{\mu}) + \boldsymbol{\eta}_t, \tag{14}$$

where $\Phi_k = \left(\phi_{ij,k}\right)_{i,j=1,\ldots,n}$ are $n \times n$ coefficient matrices. Introducing the matrices $\Phi_k = \left(\phi_{ij,k}\right)_{i,j=1,\ldots,n}$ allows us to model connectivity in financial time series through the concept of Granger-causality in latent volatility process. We say that h_i does not Granger-cause h_j

if all $(\phi_{ij,k})_{k=1,\dots,p} = 0$. The standard conditions on stationarity of a vector autoregressive model apply: the root of $|\mathbb{I} - \lambda\Phi| = 0$ should lie outside the unit circle, and the errors η_t are independent and identically normally distributed with mean zero and variance-covariance matrix $\Sigma = diag(\sigma_1^2, \dots, \sigma_n^2)$. Equations (13) and (14) are multivariate extensions of the model described in Equations (6) and (7). The representation from the right-hand side of Equations (8) and (9) can be obtained by including a vector of parameters β into $\boldsymbol{\Omega_t}$ and removing $\boldsymbol{\mu}$ from Equation (14). As before, for identifiability, only one vector of the scale parameters—either $\boldsymbol{\mu}$ or $\boldsymbol{\beta}$—should be included in the model.

The above MSV model can also be viewed as a non-linear state-space model where (14) is the state equation of the latent process h_t and (13) is the observation equation that depends non-linearly on the latent state. Note that, in this model, the time series are interconnected and the relationship between them can be interpreted through the concept of Granger-causality in latent volatility processes.

3. Methods

3.1. Bayesian Inference

In this paper, we review various methods that sample from or approximate the posterior distribution of the parameters of the model $\boldsymbol{\theta}$. The sampling or approximate methods are necessary since we are working in the framework when the posterior distribution and the likelihood are analytically intractable. The Bayes' rule allows us to write posterior distribution in the form

$$p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{\pi(\boldsymbol{\theta})g(\boldsymbol{y}|\boldsymbol{\theta})}{m(\boldsymbol{y})}, \tag{15}$$

where $\pi(\boldsymbol{\theta})$ is the prior distribution of the parameters of the model, $g(\boldsymbol{y}|\boldsymbol{\theta})$ is the likelihood of the data given parameters of the model, and $m(\boldsymbol{y})$ is the marginal density of y, which can be viewed as normalizing constant and which we will ignore in this paper. In the remainder of the paper we will work with the Bayes' rule in proportionality terms:

$$p(\boldsymbol{\theta}|\boldsymbol{y}) \propto \pi(\boldsymbol{\theta})g(\boldsymbol{y}|\boldsymbol{\theta}). \tag{16}$$

Note that in the stochastic volatility model we have to estimate parameters of the model $\boldsymbol{\theta} = (\mu, \phi, \sigma^2)$ and the latent vector of volatilities $\boldsymbol{h}$. Thus, we are interested in the following form of the Bayes' rule

$$p(\boldsymbol{\theta}, \boldsymbol{h}|\boldsymbol{y}) \propto g(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{h})f(\boldsymbol{h}|\boldsymbol{\theta})\pi(\boldsymbol{\theta}). \tag{17}$$

Multiple approaches can be used for the estimation of $p(\boldsymbol{\theta}, \boldsymbol{h}|\boldsymbol{y})$. One of the challenges is that neither posterior $p(\boldsymbol{\theta}, \boldsymbol{h}|\boldsymbol{y})$ nor the likelihood $g(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{h})$ is tractable. We start our review by considering sequential Monte Carlo methods, also known as particle filtering, for the estimation of the likelihood $g(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{h})$. We then discuss Metropolis-Hastings algorithm for sampling from the posterior and how these two algorithm can be combined into particle Metropolis-Hastings for sampling from the posterior distribution. We continue the review of the methods by considering RMHMC method in which the parameters and volatilities are sampled within the same framework. Finally, we review two approximate methods: integrated nested Laplace approdximation and fixed-form variational Bayes, two different ways of approximating posterior distribution.

3.2. Sequential Monte Carlo for the Estimation of the Likelihood

The Sequential Monte Carlo (SMC) method, also known in the literature as *particle filtering*, is considered a state-of-the-art method for estimation of the intractable likelihoods in nonlinear state-space models. The general idea behind this method lies in the estimation of the latent states by drawing multiple samples (particles) and then propagating them in time according to corresponding importance weights. By combining the weights over all time steps, one obtains a marginal likelihood estimate. Standard and well-known schemes are *Bootstrap particle filter* (BPF) [20], *Seqiential Importance Sampling* (SIS), and *Seqiential Importance Resampling* (SIR) [21]. Sequential Monte Carlo methods were elegantly combined with Markov Chain Monte Carlo in [4], and the method was named particle Markov Chain Monte Carlo (PMCMC). This method provides a powerful and coherent approach for Bayesian inference in a wide range of complex models. In the later subsections, we will discuss how sequential Monte Carlo methods are combined with Markov Chain Monte Carlo for fully Bayesian inference in stochastic volatility models. One of the concerns when using and implementing SMC for the likelihood estimation is the variance of the estimated likelihood. Standard SMC techniques such as SIS are prone to have high variance of the estimated likelihood once the dimensionality of the problem increases [22]. A number of studies have tried to address this problem. The common choice of proposal for sample of particles in standard schemes is $f(h_t|h_{t-1})$. Pitt and Shephard [23] propose an *auxiliary particle filter* as a solution that is using proposal for particles which takes into account the current observation $q(h_t|h_{t-1}, y_t)$ and not only the dynamics of the latent process itself. Scharth and Kohn [24] suggest using efficient importance sampling [25] inside the PMCMC procedure. Guarniero et al. [26] use twisted representation of the model and use the look-ahead type of particle filtering to address the issue of high variance of the estimated likelihood. Johansen and Doucet [27] compare sequential importance resampling (SIR) with auxiliary particle filter and find that APF does not always outperform SIR. Often, the variance of the estimated likelihood is analyzed in the true value of the parameters, such as in [24]. However, when using particle Markov Chain Monte Carlo, it is also of interest whether the same conclusions hold in different points of the parameter space. In particular, we never start running the algorithm at the point of the true parameter values. This means that if the variance of the estimated likelihood is much larger in some areas of the parameter space, the convergence of the algorithm can be affected. Having insights into how the variance of the estimated likelihood is different in the parameter space can help to make a more efficient choice of the starting point for the algorithm.

We first review the sequential Monte Carlo methods for the estimation of the likelihood. After that, we discuss Metropolis-Hastings algorithm and how SMC and Metropolis-Hastings can be combined for Bayesian inference in general and stochastic volatility models in particular.

3.2.1. Sequential Monte Carlo

Assume that we are in the framework with an observed time series process y_t and a latent Markovian process h_t. Since we never observe the latent process, we need to infer it. The objective that can be achieved with *Sequential Monte Carlo* (SMC) is also known as *particle filtering*. The method operates in sequential manner with arriving observations y_t. The posterior distribution of the latent process can be computed sequentially

$$p(h_{0:t}|y_{1:t}) = p(h_{0:t-1}|y_{0:t-1})\frac{g(y_{1:t}|h_{0:t})f(h_t|h_{t-1})}{p(y_t|y_{t-1})}. \tag{18}$$

The denominator of Equation (18) is not analytically tractable, which can be also seen from Equation (12) earlier. SMC allows us to estimate the posterior distribution $p(h_{0:t}|y_{1:t})$ and additionally get the estimate of the likelihood

$$\begin{aligned} L(y_{1:T}) &= \int p(y_{1:T}, h_{1:T})dh_{1:T} = \int g(y_{1:T}|h_{1:T})p(h_{1:T})dh_{1:T} \\ &= \int g(y_1|h_1)p(h_1)\prod_{t=2}^{T} g(y_t|h_t)f(h_t|h_{t-1})dh_1 \dots h_T. \end{aligned} \tag{19}$$

The basic procedure of particle filtering in this setting can be summarized by three crucial steps: prediction, updating, and resampling. The outline of a basic particle filter can be summarized in the following way.

- Initialization: given the prior distribution $\pi(\theta_0)$, we draw N independent random samples $\{h_i^0\}_{i=1}^N$; these samples we call *particles*.

- Prediction: we sample particles according to the importance density

$$h_t^{(i)} \sim q(h_t|h_{t-1}^{(i)}, y_t). \tag{20}$$

- Updating: During updating, we assign a weight $w_t^{(i)}$ to every particle

$$w_t^{(i)} = \frac{p(y_t|h_t^{(i)})f(h_t^{(i)}|h_{t-1}^{(i)})}{p(y_t|y_{1:t-1})q_t(h_t^{(i)}|h_{0:t-1}^{(i)})} \tag{21}$$

  and normalize these weights to sum to 1. Every weight can be interpreted as our "confidence" about a particle.
- Resampling: resample the particles if the effective number of particles,

$$N_{eff} = \frac{1}{\sum_{i=1}^{N}(\omega_t^{(i)})^2}, \tag{22}$$

  is too low. In Equation (22), $\omega_t^{(i)}$ is the normalized weight of particle i at the time step k. The threshold for the resampling step is set depending on whether particle degeneracy is a problem. In general, we perform resampling when $N_{eff} < N/c$, where c is a constant.

The resampling step is performed to find the trade-off between two well-documented problems: particle *degeneracy* and particle *impoverishment* [28]. The former happens when the resampling step is ignored or is not performed frequently enough. In this case, one ends up with a particle set that has zero weights. The latter problem happens when the particle set is resampled too frequently; then, eventually one gets one particle with a large weight and hence the particle set lacks the diversity. The way to find the balance between these two problems is resampling when the efficient number of particles is smaller than a certain threshold.

In this paper, we consider two particle filters: bootstrap and auxiliary particle filters. A generic particle filter is presented in Algorithm A1 [28]. The bootstrap filter is a variation of a more general approach—sequential importance sampling (resampling). The distinction of the bootstrap filter is the proposal mechanism for the particles. In the bootstrap particle filter the proposals for the particles are made on the basis of the dynamics of the model $f(h_t|h_{t-1})$. If $q(h_t|y_t, h_{t-1}) = f(h_t|h_{t-1})$, then the term $\frac{f(h_t|h_{t-1})}{q(h_t|y_t,h_{t-1})}$ is equal to 1. In the case of the auxiliary particle filter, we also incorporate the current observation into the proposal mechanism $q(h_t|h_{t-1}, y_t)$. Incorporating the current observation into the proposal for the particles in some cases allows us to reduce the variance of the estimated likelihood. In our case, there is no analytical expression for the proposal density. In the next subsection, we discuss how it can be approximated as proposed in [23].

3.2.2. Auxiliary Particle Filter for SV Model

Incorporating knowledge of y_t into proposals for particles $q(h_t|h_{t-1}, y_t)$ can help to reduce the variance of the estimated likelihood and improve the approximation of the filtering distribution $p(h_t|y_{1:t})$. Note, however, that it is not always the case as has been shown in [27]. Only in the case of linear Gaussian state-space models does the proposal density from Equation (A2) have an analytical expression. Hence, for the stochastic volatility models, this term must be approximated. Pitt and Shephard [23] propose using non-blind proposals for the next generation of particles by first expanding $\log g(y_{t+1}|h_{t+1})$ to a second-order term around μ^k_{t+1} via Taylor expansion

$$\log g(y_{t+1}|h_{t+1}, \mu^k_{t+1}) = \log p(y_{t+1}|\mu^k_{t+1})' \times \frac{\partial \log p(y_{t+1}|\mu^k_{t+1})}{\partial h_{t+1}} + \frac{1}{2} \times (h_{t+1} - \mu^k_{t+1})' \times \frac{\partial^2 \log p(y_{t+1}|\mu^k_{t+1})}{\partial h_{t+1} h'_{t+1}} \times (h_{t+1} - \mu^k_{t+1}) \tag{23}$$

For deriving the expression for $\log g(y_{t+1}|h_{t+1}, \mu^k_{t+1})$, recall that $y_t \sim N(0, \exp(h_t))$ and hence

$$g(y_t|h_t) = \frac{1}{\sqrt{2\pi \exp(h_t)}} \exp\left\{-\frac{y_t^2}{2\exp(h_t)}\right\} = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{y_t^2}{\exp(h_t)} - \frac{h_t}{2}\right\} \tag{24}$$

and further note that $f(h_t|h_{t-1}) = N(\mu + \phi(h_{t-1} - \mu), \sigma^2_\eta)$; thus

$$f(h_t|h_{t-1}) = \frac{1}{\sqrt{2\pi\sigma^2_\eta}} \exp\left\{\frac{(h_t - \mu - \phi(h_{t-1} - \mu))^2}{2\sigma^2_\eta}\right\}. \tag{25}$$

It follows that the proposal for particles at time $t + 1$ when taking into account the observation of the same period is

$$q(h_{t+1} \mid h_t^{(k)}, y_{t+1}; \mu^{(k)}_{t+1}) = N\left(\mu^{(k)}_{t+1} + \frac{\sigma^2}{2}\left(\frac{y_t^2}{\beta^2} \exp(-\mu^{(k)}_{t+1}) - 1\right), \sigma^2\right). \tag{26}$$

3.2.3. Metropolis–Hastings

In this section, we consider the problem of sampling from the posterior distribution and a general algorithm to construct such a sampling scheme. With the Metropolis–Hastings algorithm, we sample from the posterior distribution by proposing a transition $\boldsymbol{\theta} \to \boldsymbol{\theta}^*$ with the density $q(\boldsymbol{\theta}^*|\boldsymbol{\theta})$, which we accept with probability

$$\alpha(\boldsymbol{\theta}, \boldsymbol{\theta}^*) = \min\left\{1, \frac{\tilde{p}(\boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta})} \frac{q(\boldsymbol{\theta}|\boldsymbol{\theta}^*)}{q(\boldsymbol{\theta}^*|\boldsymbol{\theta})}\right\}, \tag{27}$$

where $\tilde{p}(\cdot)$ is a function proportional to our target distribution. A common choice for the proposal distribution is a random-walk, which we also use when applying PMCMC later in this paper, $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}) = N(\boldsymbol{\theta}^*|\boldsymbol{\theta}, \Sigma)$. The Metropolis–Hastings algorithm is one of the off-the-shelf MCMC methods in the statistical community. It is quite general and can be applied to various problems. The implementation of the Metropolis–Hastings algorithm requires specification of multiple quantities. We need to specify a conditional density $q(\boldsymbol{\theta}^*|\boldsymbol{\theta})$ that is a proposal distribution, generally $q(\boldsymbol{\theta}^*|\boldsymbol{\theta})$ should be such that we can easily simulate from it. In many applications, including ours, it is reasonable to take the Gaussian distribution as proposal distribution. In this case, it is also symmetric, meaning $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}) = q(\boldsymbol{\theta}|\boldsymbol{\theta}^*)$. The Metropolis-Hastings iteration is outlined in the Algorithm 1.

In this algorithm, $\alpha(\boldsymbol{\theta}, \boldsymbol{\theta}^*)$ is the Metropolis–Hastings acceptance probability, where $\boldsymbol{\theta}$ is the current state of the chain and $\boldsymbol{\theta}^*$ is the candidate state of the parameter vector. Generally, in the simulations, it is desired to have around 25% of proposed candidate values accepted [29]. The idea is that when the proposal steps are too large (we make a proposal that is far away from the current state, $\boldsymbol{\theta}$, in the Markov chain), we do not explore local regions sufficiently well; moreover many of the candidates are then very likely to be rejected. When the proposal steps are very small, the acceptance rate will be very high, however, then we are not likely to leave regions of the local maximum or the convergence will happen very slowly.

Algorithm 1 Metropolis-Hastings Algorithm.

```
1: Given $\boldsymbol{\theta}^{(t)}$,
2: Generate $\boldsymbol{\theta}_t^* \sim q(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}^{(t)})$,
3: Take
   $$\boldsymbol{\theta}^{(t+1)} = \begin{cases} \boldsymbol{\theta}_t^*, & \text{with probability } \alpha(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}_t^*) \\ \boldsymbol{\theta}^{(t)} & \text{with probability } 1 - \alpha(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}_t^*), \end{cases}$$
   where
   $$\alpha(\boldsymbol{\theta}, \boldsymbol{\theta}^*) = \min\left(1, \frac{\tilde{p}(\boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta}_t)} \frac{q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^*)}{q(\boldsymbol{\theta}^* \mid \boldsymbol{\theta})}\right)$$
```

The performance of Metropolis–Hastings depends on the choice of $q(\cdot)$ proposal distribution. In the simulation studies, we consider random-walk proposals of the form $\boldsymbol{\theta}_{i+1}^* = \boldsymbol{\theta}_i + \boldsymbol{\epsilon}_i$, where i is iteration of the algorithm and $\boldsymbol{\epsilon}_i$ is assumed to be Gaussian. More information on the theoretical properties of this algorithm can be found in [30].

3.2.4. Particle Metropolis-Hastings

Particle Markov Chain Monte Carlo (PMCMC) methods were introduced in [4]. The basic idea is that MCMC methods, and in particular, Metropolis–Hastings algorithm, which is of interest to us, can be combined with Sequential Monte Carlo to make draws from the posterior distributions of the parameters. Algorithm 2 presents the particle Metropolis–Hastings algorithm. The difference from the standard Metropolis-Hastings is in the quantity $\hat{p}_{\boldsymbol{\theta}^*}(y_{1:T})$, which is the estimate of the likelihood obtained with a particle filter conditioning on the parameters vector $\boldsymbol{\theta}$. In this algorithm, $q(\boldsymbol{\theta}(i-1)|\boldsymbol{\theta}^*)$ is the proposal distribution (which cancels out when it is symmetric), and $\pi(\cdot)$ is prior distribution.

Algorithm 2 Particle Metropolis-Hastings.

```
1: Initialize algorithm at $i = 0$ and initialize parameters $\boldsymbol{\theta}(0)$
2: Run an SMC algorithm targeting $p_{\boldsymbol{\theta}^{(0)}}(h_{1:T} \mid y_{1:T})$, sample $h_{1:T}^{(0)} \sim \hat{p}_{\boldsymbol{\theta}^{(0)}}(\cdot \mid y_{1:T})$ and let
   $\hat{p}_{\boldsymbol{\theta}}^{(0)}(y_{1:T})$ denote the marginal likelihood estimate for the initialized parameters
3: for $i = 1, \ldots, M$ do
4:     Generate $\boldsymbol{\theta}^* \sim q(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}^{(i-1)})$,
5:     Run an SMC algorithm targeting $p_{\boldsymbol{\theta}^*}(h_{1:T} \mid y_{1:T})$, sample $h_{1:T}^* \sim \hat{p}_{\boldsymbol{\theta}^*}(\cdot \mid y_{1:T})$ and
   let $p\hat{_{\theta^*}}(y_{1:T})$ denote the marginal likelihood estimate for the proposed parameters $\theta^*$
6:     With probability
       $$\min\left\{1, \frac{\hat{p}_{\boldsymbol{\theta}^*}(y_{1:T})}{\hat{p}_{\boldsymbol{\theta}^{(i-1)}}(y_{1:T})} \frac{\pi(\boldsymbol{\theta}^*)}{\pi(\boldsymbol{\theta}^{(i-1)})} \frac{q(\boldsymbol{\theta}^{(i-1)}|\boldsymbol{\theta}^*)}{q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(i-1)})}\right\} \quad (28)$$
7:     Set $\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^*$, $h_{1:T}^{(i)} = h_{1:T}^*$ and $\hat{p}_{\boldsymbol{\theta}^{(i)}}(y_{1:T}) = \hat{p}_{\boldsymbol{\theta}^*}(y_{1:T})$;
8:     Otherwise set $\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)}$, $h_{1:T}^{(i)} = h_{1:T}^{(i-1)}$ and $\hat{p}_{\boldsymbol{\theta}^{(i)}}(y_{1:T}) = \hat{p}_{\boldsymbol{\theta}^{(i-1)}}(y_{1:T})$.
9: end for
```

3.3. MCMC with Gradient Information

In this section, we discuss Riemann Manifold Langevin Hamiltonial Monte Carlo methods that are introduced in [5] and in particular can be applied to stochastic volatility models.

The method originates in physics statistical literature and provides a tool that allows one to make large transitions with high acceptance probability, something that standard methods such as Metropolis–Hastings fail to achieve. The idea of HMC is based on relation between differential geometry and statistical theory (MCMC in particular). Girolami and Calderhead [5] propose the Metropolis-adjusted Langevin algorithm and Hamiltonian Monte Carlo sampling algorithms that are defined on the Riemann manifold. Their methods allow us to overcome the problem of sampling from high-dimensional densities that may show strong correlation. We further provide the general background and summary of the algorithms together with the necessary quantities for their implementation in the case of stochastic volatility models. It is not our goal to provide theoretical foundations of these methods in this article. For deeper theoretical foundations, see [31–33].

In standard MCMC setting, one uses probability distribution to make a proposal for the next state of the Markov chain. Hamiltonian Monte Carlo methods exploit physical system dynamics to make proposals for the next state. It can improve the mixing drastically and result in a more efficient algorithm. Especially since we are interested in multivariate modeling, a more efficient exploration of the posterior distribution is of interest. Once the dimension of the model grows with standard random walk, it is very hard to make proposals that would be accepted frequently enough and result in a good mixing Markov chain. We first introduce some basic ideas on which Hamiltonian Monte Carlo method is built; for an extensive introduction, we refer to [33].

3.3.1. Metropolis-Adjusted Langevin Algorithm

Previously we have discussed the Metropolis-Hastings algorithm. The idea of the Metropolis-Hastings algorithm is to make a new proposal $\boldsymbol{\theta}^*$ using random walk. Then this proposal is accepted with probability.

$$\alpha(\boldsymbol{\theta}, \boldsymbol{\theta}^*) = \min\left\{1, \frac{\tilde{p}(\boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta})}\frac{q(\boldsymbol{\theta}|\boldsymbol{\theta}^*)}{q(\boldsymbol{\theta}^*|\boldsymbol{\theta})}\right\}. \tag{29}$$

Although this algorithm benefits from desirable theoretical guarantees, the random walk proposal is not efficient, especially when the number of parameters in the model becomes large. Metropolis-adjusted Langevin algorithm (MALA), originally proposed in [34], is designed to solve the same problem—sample from the target distribution. The big advantage of MALA in comparison to Metropolis–Hastings is the construction for the proposal of the candidate parameter $\boldsymbol{\theta}^*$. The proposal mechanism for MALA originates from the stochastic differential equation based on Langevin diffusion; the proposal mechanism reads

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}^n + \epsilon^2 \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^n)/2 + \epsilon \boldsymbol{z}^{\boldsymbol{n}}, \tag{30}$$

where we define $L(\boldsymbol{\theta}^n) = \log(p(\theta))$ and $\boldsymbol{z} \sim N(\boldsymbol{z}|\boldsymbol{0}, \boldsymbol{I})$ and ϵ—integration step size. Convergence for this proposal is not guaranteed unless we employ a Metropolis acceptance probability after every integration step. For convenience, let us define

$$\boldsymbol{\mu}(\boldsymbol{\theta}^n, \epsilon) = \boldsymbol{\theta}^n + \frac{\epsilon^2}{2}\nabla_{\theta} L(\theta^n); \tag{31}$$

then the proposal density can be written as $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^n) = N(\boldsymbol{\theta}^*|\mu(\boldsymbol{\theta}^n, \epsilon), \epsilon^2 I)$. The standard acceptance probability follows

$$\min\{1, p(\boldsymbol{\theta}^*)q(\boldsymbol{\theta}^n|\boldsymbol{\theta}^*)/p(\boldsymbol{\theta}^n)q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^n)\}. \tag{32}$$

The type of proposal in Equation (29) is inefficient for strongly correlated parameters $\boldsymbol{\theta}$. To solve this issue, one can use a preconditioning matrix $\boldsymbol{M}$

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}^n + \epsilon^2 \boldsymbol{M} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^n)/2 + \epsilon\sqrt{\boldsymbol{M}} z^n. \tag{33}$$

Unfortunately there is no principled way to choose matrix $\boldsymbol{M}$. As we will see later, HMC encounters the same problem. Generally, MALA iterates between two general steps.

First, Langevin dynamics is used for the proposals, and it exploits the gradients of the target. Second, the proposals are accepted or rejected similarly to those of the Metropolis–Hastings algorithm.

3.3.2. Hamiltonian Monte Carlo Algorithm

The HMC algorithm [31] also uses gradient information for constructing the proposal of the parameters in the MCMC scheme. In particular, it exploits the ideas from simulating the behavior of the physical systems. Similarly to describing the behavior of the physical system, HMC performs sampling by exploiting *Hamiltonian dynamics*. A conceptual introduction to this class of methods and its relationship to differential geometry can be found in [33]. In this section, we discuss the general idea behind the algorithm without performing detailed derivations. We focus on the final proposal machinery that can be used in practice and investigate which quantities need to be manually computed before implementing the algorithm and which variables need to be calibrated for the successful performance of the algorithm. First, let us consider a general set-up. In Hamiltonian Monte Carlo, we consider a Hamiltonian function

$$H(\boldsymbol{\theta}, \boldsymbol{p}) = -\log p(\boldsymbol{\theta}) + \frac{1}{2}\log\{(2\pi)^D|\boldsymbol{M}|\} + \frac{1}{2}\boldsymbol{p}^T\boldsymbol{M}^{-1}\boldsymbol{p}, \tag{34}$$

which consists of potential energy in the system $E(\boldsymbol{\theta}) = -L(\boldsymbol{\theta})$ and kinetic energy $K(p) = \frac{1}{2}\log\{(2\pi)^D|\boldsymbol{M}|\} + \frac{1}{2}\boldsymbol{p}^T\boldsymbol{M}^{-1}\boldsymbol{p}$; variables $\boldsymbol{p}$ are called momentum variables. The dynamics of the system then evolves according to Hamiltonian equations

$$\frac{d\boldsymbol{\theta}}{d\tau} = \frac{\partial H}{\partial \boldsymbol{p}} = \boldsymbol{M}^{-1}\boldsymbol{p}, \tag{35}$$

$$\frac{d\boldsymbol{p}}{d\tau} = -\frac{\partial H}{\partial \boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}), \tag{36}$$

where by τ in physical interpretation of the system we denote continuous time. Practical implementation requires discretization, and the commonly used scheme for this purpose is the leapfrog discretezation:

$$\boldsymbol{p}(\tau + \epsilon/2) = \boldsymbol{p}(\tau) + \epsilon\nabla_{\boldsymbol{\theta}} L\{\boldsymbol{\theta}(\tau)\}/2, \tag{37}$$

$$\boldsymbol{\theta}(\tau + \epsilon) = \boldsymbol{\theta}(\tau) + \epsilon\boldsymbol{M}^{-1}\boldsymbol{p}(\tau + \epsilon/2), \tag{38}$$

$$\boldsymbol{p}(\tau + \epsilon) = \boldsymbol{p}(\tau + \epsilon/2) + \epsilon\nabla_{\boldsymbol{\theta}} L\{\boldsymbol{\theta}(\tau + \epsilon)\}/2. \tag{39}$$

This scheme does not sample from the target distribution and to correct for that, implementation of Metropolis acceptance probability is necessary. For a proposal $(\boldsymbol{\theta}, \boldsymbol{p}) \to (\boldsymbol{\theta}^*, \boldsymbol{p}^*)$, acceptance probability in this algorithm is defined as

$$\min\{1, \exp\{-H(\boldsymbol{\theta}^*, \boldsymbol{p}^*) + H(\boldsymbol{\theta}, \boldsymbol{p})\}.$$

Thus, HMC iterates between updating momentum variables, proposal for the parameter values, additional update to the momentum variables, and then an acceptance/rejection step. The Gibbs sampler provides a good understanding for the system evolution in this algorithm:

$$\boldsymbol{p}^{n+1}|\boldsymbol{\theta}^n \sim p(\boldsymbol{p}^{n+1}|\boldsymbol{\theta}^n) = p(\boldsymbol{p}^{n+1}) = N(\boldsymbol{p}^{n+1}|0, \boldsymbol{M}), \tag{40}$$

$$\boldsymbol{\theta}^{n+1}|\boldsymbol{p}^{n+1} \sim p(\boldsymbol{\theta}^{n+1}|\boldsymbol{p}^{n+1}) \tag{41}$$

Similarly to MALA, the choice of matrix $\boldsymbol{M}$ is crucial for good performance of HMC. While the choice of the step size and the leapfrog steps can be tuned relatively easily by considering acceptance rate, the choice of the matrix $\boldsymbol{M}$ is challenging, and there is no principled way to define it. Leapfrog step and step size proposal are two variables that need to be calibrated when implementing HMC. Usually, different combinations of these two variables are considered, and the combination leading to the highest acceptance rate is picked.

3.4. Riemann Manifold Hamiltonian Monte Carlo

The further improvement of HMC and MALA can done by defining the algorithms on Riemann manifold instead of Euclidean space. Proposals guided by Riemann metric instead of Euclidean distance have the potential to explore parameter space more efficiently, especially in the cases when the target density is high-dimensional or exhibits strong correlation [5]. The method originally proposed in [5] and multiple algorithms were compared in the paper: MALA, MMALA, HMC, and RMHMC. For detailed comparison between these methods, we refer to [5], while in our simulation studies, we will focus on comparing RMHMC and particle Metropolis–Hastings for the estimation of parameters in stochastic volatility models.

Girolami and Calderhead [5] define HMC methods in the form of Riemann manifold, and this can be seen as generalization of HMC. The Hamiltonian on the Riemann manifold is defined as follows

$$H(\boldsymbol{\theta}, \boldsymbol{p}) = -\log p(\boldsymbol{\theta}) + \frac{1}{2}\log((2\pi)^n \mid G(\boldsymbol{\theta}) \mid) + \frac{1}{2}\boldsymbol{p}^T G(\theta)\boldsymbol{p} \tag{42}$$

with $\exp(-H(\boldsymbol{\theta}, \boldsymbol{p})) = p(\boldsymbol{\theta}, \boldsymbol{p}) = p(\boldsymbol{\theta})p(\boldsymbol{p} \mid \boldsymbol{\theta})$ and the marginal target density

$$\begin{aligned} p(\boldsymbol{\theta}) \propto \int \exp(-H(\boldsymbol{\theta}, \boldsymbol{p}))d\boldsymbol{p} = \frac{\exp\{\log p(\theta)\}}{\sqrt{2\pi^n \mid G(\boldsymbol{\theta}) \mid}} \int \exp\Big\{-\frac{1}{2}\boldsymbol{p}^T G(\boldsymbol{\theta})^{-1}\boldsymbol{p}\Big\}d\boldsymbol{p} \\ = \exp(\log p(\boldsymbol{\theta})). \end{aligned} \tag{43}$$

The general idea behind the updates in RMHMC is similar to that of HMC, and the updates for the momentum variables and parameters of the model are defined in Equations (44)–(46).

$$\boldsymbol{p}(\tau + \frac{\epsilon}{2}) = \boldsymbol{p}(\tau) - \frac{\epsilon}{2}\nabla_{\boldsymbol{\theta}} H\Big\{\boldsymbol{\theta}(\tau), \boldsymbol{p}(\tau + \frac{\epsilon}{2})\Big\}, \tag{44}$$

$$\boldsymbol{\theta}(\tau + \epsilon) = \boldsymbol{\theta}(\tau) + \epsilon/2\Big[\nabla_p H\Big\{\boldsymbol{\theta}(\tau), p(\tau + \frac{\epsilon}{2})\Big\} + \nabla_p H\Big\{\boldsymbol{\theta}(\tau + \epsilon), p(\tau + \frac{\epsilon}{2})\Big\}\Big], \tag{45}$$

$$p(\tau + \epsilon) = p(\tau + \frac{\epsilon}{2}) - \frac{\epsilon}{2}\nabla_{\boldsymbol{\theta}} H\Big\{\boldsymbol{\theta}(\tau + \epsilon), p(\tau + \frac{\epsilon}{2})\Big\} \tag{46}$$

Therefore, as in standard HMC algorithm, we iterate between half-step update of the momentum variables, and then we update position variables, and we finish iteration with additional half-step update of the momentum variables and Metropolis acceptance/rejection step with the probability

$$\min\{1, \exp\{-H(\boldsymbol{\theta}^*, \boldsymbol{p}^*) + H(\boldsymbol{\theta}^n, \boldsymbol{p}^{n+1})\}\}.$$

Similarly to HMC, RMHMC can be viewed as a Gibbs sampling scheme

$$\boldsymbol{p}^{n+1}|\boldsymbol{\theta}^n \sim p(\boldsymbol{p}^{n+1}|\boldsymbol{\theta}^n) = N\{\boldsymbol{p}^{n+1}|0, G(\boldsymbol{\theta}^n)\}, \tag{47}$$

$$\boldsymbol{\theta}^{n+1}|\boldsymbol{p}^{n+1} \sim p(\boldsymbol{\theta}^{n+1}|\boldsymbol{p}^{n+1}). \tag{48}$$

Recall that in the case of MALA and HMC, matrix $\boldsymbol{M}$ has to be chosen manually and there is no principled way to choose it. In RMHMC, matrix $\boldsymbol{G}(\boldsymbol{\theta})$ is defined at each step by underlying geometry; see for more details [5]. Below we discuss quantities that need to be computed for the implementation of RMHMC in the case of stochastic volatility model and in particular $\boldsymbol{G}(\boldsymbol{\theta})$.

Recall stochastic volatility model parametrized through β

$$y_t = \beta \exp(h_t/2)\epsilon_t, \tag{49}$$

$$h_{t+1} = \phi h_t + \eta_{t+1}, \tag{50}$$

$\epsilon_t \sim N(0,1), \eta_t \sim N(0,\sigma^2)$, with $h_1 \sim N(0,\sigma^2/(1-\phi^2))$.
The joint likelihood of the model is

$$p(y,h,\beta,\phi,\sigma) = \prod_{t=1}^{T} p(y_t \mid h_t,\beta) \prod_{t=2}^{T} p(h_t \mid h_{t-1},\phi,\sigma)\pi(\beta)\pi(\phi)\pi(\sigma) \tag{51}$$

The prior distributions are chosen as follows

$$\beta \propto \exp(\beta), \qquad \sigma^2 \sim Inv-\chi^2(10,0.05), \qquad (\phi+1)/2 \sim Beta(20,1.5). \tag{52}$$

Further, following [5], we write the partial derivatives for $L = p(y,h \mid \beta,\phi,\sigma)$

$$\frac{\partial L}{\partial \beta} = -\frac{T}{\beta} + \sum_{t=1}^{T} \frac{y^2}{\beta^3 \exp(h_t)}, \tag{53}$$

$$\frac{\partial L}{\partial \phi} = -\frac{\phi}{(1-\phi^2)} + \frac{\phi h_1^2}{\sigma^2} + \sum_{t=2}^{T} \frac{h_{t-1}(h_t - \phi h_{t-1})}{\sigma^2}, \tag{54}$$

$$\frac{\partial L}{\partial \sigma} = -\frac{T}{\sigma} + \frac{h_1^2(1-\phi^2)}{\sigma^3} + \sum_{t=2}^{T} \frac{(h_t - \phi h_{t-1})^2}{\sigma^3}. \tag{55}$$

To implement the algorithms, we require the expressions for the individual components of the metric tensor for the likelihood. Following [5], the expressions are

$$\mathbb{E}\left\{\frac{\partial L}{\partial \beta}\frac{\partial L}{\partial \beta}\right\} = \frac{2T}{\beta^2}, \qquad \mathbb{E}\left\{\frac{\partial L}{\partial \sigma}\frac{\partial L}{\partial \sigma}\right\} = \frac{2T}{\sigma^2}, \qquad \mathbb{E}\left\{\frac{\partial L}{\partial \beta}\frac{\partial L}{\partial \sigma}\right\} = \mathbb{E}\left\{\frac{\partial L}{\partial \beta}\frac{\partial L}{\partial \phi}\right\} = 0, \tag{56}$$

$$\mathbb{E}\left\{\frac{\partial L}{\partial \sigma}\frac{\partial L}{\partial \phi}\right\} = \frac{2\phi}{\sigma^3(1-\phi^2)}, \qquad \mathbb{E}\left\{\frac{\partial L}{\partial \phi}\frac{\partial L}{\partial \phi}\right\} = \frac{2\phi^2}{(1-\phi^2)^2} + \frac{T-1}{1-\phi^2}. \tag{57}$$

Furthermore, the expressions for the metric tensor for the likelihood and its partial derivatives follow

$$G(\phi,\sigma,\beta) = \begin{bmatrix} \frac{2T}{\beta^2} & 0 & 0 \\ 0 & \frac{2T}{\sigma^2} & \frac{2\phi}{\sigma^3(1-\phi^2)} \\ 0 & \frac{2\phi}{\sigma^3(1-\phi^2)} & \frac{2\phi^2}{(1-\phi^2)^2} + \frac{T-1}{1-\phi^2} \end{bmatrix}, \tag{58}$$

$$\frac{\partial G}{\partial \beta} = \begin{bmatrix} -\frac{4T}{\beta^3} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{59}$$

$$\frac{\partial G}{\partial \sigma} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\frac{4T}{\sigma^3} & -\frac{6\phi}{\sigma^4(1-\phi^2)} \\ 0 & -\frac{6\phi}{\sigma^4(1-\phi^2)} & 0 \end{bmatrix}, \tag{60}$$

$$\frac{\partial G}{\partial \phi} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{2}{\sigma^3(1-\phi^2)} + \frac{4\phi^2}{\sigma^3(1-\phi^2)^2} \\ 0 & \frac{2}{\sigma^3(1-\phi^2)} + \frac{4\phi^2}{\sigma^3(1-\phi^2)^2} & \frac{2\phi(1+T)}{(1-\phi^2)^2} + \frac{6\phi^3}{(1-\phi^2)^3} \end{bmatrix}. \tag{61}$$

The proposal machinery in RMHMC provides advantages for exploring parameter space efficiently. However, it is not easily adaptable for different model specifications, especially when increasing the model's dimensionality, as we discussed in Section 1. In particular, although matrix G can be computed in the multivariate model specified in Equations (12) and (13) exactly, it scales quadratically with the number of parameters. This might be one of the reasons why the method has not been used on multivariate stochastic volatility models we introduced in Section 1. However, probabilistic programming languages [35,36] and automatic differentiation possibilities developed in recent years allow the efficient and adaptable implementation of these algorithms in practice.

3.5. Integrated Nested Laplace Approximation

Integrated Nested Laplace Approximation was introduced in [7]. The method is based on the nested version of the classical Laplace approximation and was introduced for latent Gaussian models (LGMs). It became a popular approach in Bayesian inference due to its good performance in the variety of models in the class of LGMs and its computational advantages over other methods in Bayesian literature. The computational appeal of this method comes from the possibility of exploiting sparse matrix computations when evaluating certain approximations. INLA has found its applications in many fields in the models where high-dimensional problems arise. Stochastic volatility models have been analyzed using INLA in [37,38]. Bivariate stochastic volatility model has been considered in [39], where the authors present and solve some issues that arise in using INLA in the multivariate case of the model. One of the conclusions of this study was that INLA loses its computational advantage with increased dimensionality of the stochastic volatility model. We further discuss the details of the method and the implementation shortcomings in a multivariate case and present the reader with a simulation study that illustrates the discussed approach's performance.

Stochastic volatility model can be written in the form of LGMs

$$y \mid h, \theta_1 \sim \prod_{i \in \mathcal{I}} \pi(y_i \mid h_i, \theta_1), \tag{62}$$

$$h \mid \theta_2 \sim N(\mu(\theta_2), Q^{-1}(\theta_2)). \tag{63}$$

As before, y_t is the data that we observe and h_t is the latent volatility process, and we are interested in the posterior distribution of the parameters of the model θ and the latent process given the data

$$p(h, \theta \mid y) \propto p(\theta) p(h \mid \theta) \prod_{t=1}^{T} p(y \mid h_t, \theta). \tag{64}$$

The outline of the INLA approach can be summarized in the following steps [7,37]

1. Build an approximation $p(\theta \mid y)$
2. Build an approximation to $p(h_t \mid \theta, y)$
3. Compute an approximation to $p(h_t \mid y)$ using the approximations from steps 1 and 2.

The first approximation $p(\theta \mid y)$ relies on the Gaussian approximation of the form

$$p(x \mid y, \theta) \propto \exp\left\{ -\frac{1}{2} x^T Q x + \sum g_t(h_t) \right\}, \tag{65}$$

where $x = (\mu, h)$, $g_t(h_t) = \log p(y_t \mid h_t, \theta)$. By matching the mode and curvature in the mode, we obtain the Gaussian approximation for our model

$$\tilde{p}_G(x \mid y, \theta) = K_1 \exp\left\{ -\frac{1}{2} (x - \mu)^T (Q + diag(c))(x - m) \right\}, \tag{66}$$

where K_1 is a normalizing constant, m is the modal value of $p(x \mid y, \theta)$, the vector c contains the second order terms in the Taylor expansion of $\sum g_t(h_t)$ at the modal value m, and Q is the precision matrix that has the form

$$Q = \begin{bmatrix} 1 & -\phi & & & \\ -\phi & 1+\phi^2 & -\phi & & \\ & \ddots & \ddots & \ddots & \\ & & -\phi & 1+\phi^2 & -\phi \\ & & & -\phi & 1 \end{bmatrix}. \tag{67}$$

The sparsity of the precision matrix above allows one to exploit efficient sparse matrix computational methods and thus gain computational speed. Note that in the multivariate case, this advantage disappears since the matrix Q is not sparse anymore.

When it comes to the estimation of stochastic volatility models, approximation of the marginals $p(h_t \mid \theta, y)$ is always the most challenging task. The solution that is proposed in [7] is (simplified) Laplace approximation of the form

$$\log \tilde{p}_{SLA}(x_t \mid \theta, y) = const - \frac{1}{2}x_t^2 + \gamma_t^{(1)}(\theta)x_t + \frac{1}{6}x_t^3\gamma_t^3(\theta) + \dots, \tag{68}$$

where $\gamma_t^{(1)}$ and $\gamma_t^{(3)}$ are the terms in the Taylor expansion. The final step of the method is to approximate $p(x_t \mid y)$ with the numerical integration scheme

$$\tilde{p}(x_t \mid y) = \sum_k \tilde{p}(x_t \mid \theta^k, y)\tilde{p}(\theta^k \mid y) \vartriangle_k, \tag{69}$$

for some θ^k of θ, where θ^k is selected by creating a grid of points that covers the area of high density for $\tilde{p}(\theta \mid y)$. For more details on implementation of the simplified Laplace approximation and the selection of grid of points for θ^k, see [7,37].

3.6. Fixed-Form Variational Bayes

In this section, we discuss how the posterior distribution can be approximated using the fixed-form variational Bayes method proposed in [6]. The general idea of fixed-form variational inference consists in assuming a certain factorization of the prior distribution, which naturally leads to the factorized structure of the posterior. The factorizing distributions of the posterior are then assumed to come from a certain parametric family of distributions (for example, exponential) and instead of a sampling task, as in the previous section, we would perform the optimization task of minimizing the distance between the approximating distribution and the unknown posterior distribution.

As before, assume we observe a process $\{y_t\}_{t=1}^T$ that is driven by an unobservable or latent process $\{h_t\}_{t=1}^T$. Recall that Bayes' rule gives us the posterior distribution of the parameters of the system

$$p(h \mid y) \propto g(y \mid h)\pi(h). \tag{70}$$

In the Bayesian framework, we formulate our prior beliefs, which we update once we acquire more data. In general, Variational Bayes methods focus on approximating the posterior distribution $p(h \mid y)$ with some distribution $q(h \mid y)$. Further, it is common to choose blocks of the parameters and impose independence for these blocks

$$p(h \mid y) \approx q(h \mid y) = q(h_1 \mid y)q(h_2 \mid y). \tag{71}$$

By construction, the posterior of the blocks of the parameters is independent. In the literature, this is referred to as *the mean-field assumption*. To find the optimal approximation, we minimize the Kullback–Leibler (KL) divergence from $q(h \mid y)$ to $p(h \mid y)$

$$\tilde{p}(h \mid y) = \underset{q(h_1|\cdot)q(h_2|\cdot)}{\arg\min} \; KL(q(h_1 \mid y)q(h_2 \mid y) \mid\mid p(h \mid y)). \tag{72}$$

Distributional approximation can be viewed as an optimization problem; i.e., an optimal distribution has to be chosen from the space of all possible distributions, and the KL divergence is chosen as a loss function [40]. Salimans et al. [6] propose a specific approach to the minimization problem of KL divergence, which is based on the similarities between the optimal solution to the problem and linear regression. The general idea of the approach is summarized as comprising the following steps:

- initialize all the posterior approximations $q(\theta)$;
- iterate over the parameters updating every one of them given the others;
- repeat until convergence.

Consider the stochastic volatility model

$$y_t = \beta \exp(h_t/2)\epsilon_t \tag{73}$$

$$h_{t+1} = \phi h_t + \eta_{t+1}, \tag{74}$$

with $h_1 \sim N(0, \sigma^2/(1-\phi^2))$ and $\epsilon_t \sim N(0,1)$, $\eta_t \sim N(0, \sigma_\eta^2)$. We specify our a priori beliefs in the following manner

$$p(\beta) \propto \beta^{-1} \qquad (\phi+1)/2 \sim Beta(20, 1.5), \qquad \sigma^2 \sim IG(5, 0.25). \tag{75}$$

To apply the Variational Bayes method, we need to specify the posterior approximations $q(\theta)$. It is convenient to assume a hierarchical structure of the prior, in which case it factorizes to

$$p(\phi, \sigma^2, \beta, f) = p(\phi)p(\sigma^2)p(f \mid \phi, \sigma^2)p(y \mid f), \tag{76}$$

where $f = (\log(\beta), h')$. The hierarchical structure of the prior leads to the following factorization of the posterior approximation

$$q_\xi(\sigma_\eta^2, f \mid f) = q_\xi(\sigma_\eta^2)q_\xi(f \mid \phi, \sigma^2) = \frac{q_\xi(\sigma_\eta^2 \mid \phi)p(f \mid \phi, \sigma^2)q_\xi(y \mid f)}{q_\xi(y \mid \phi, \sigma^2)}. \tag{77}$$

Thus, the posterior approximations can be chosen as follows

$$q_\xi((\phi+1)/2) = Beta(\xi_1, \xi_2), \tag{78}$$

$$q_\xi(\sigma^2 \mid \phi) \sim IG(\xi_3, \xi_4 + \xi_5\phi^2), \tag{79}$$

$$q(\log(\beta), h \mid \phi, \sigma^2) = N(m, V), \tag{80}$$

where

$$V^{-1} = P(\phi, \sigma^2) + \xi_6, \qquad m = V^{-1}\xi_7,$$

with $P(\phi, \sigma^2)$ precision matrix of $p(log(\beta), h \mid \phi, \sigma^2)$.

Once the posterior approximations are initialized, we proceed with the next step and iterate over the parameters. The parameters are updated in blocks that correspond to the factorization of the posterior approximations. First, we update the block for the persistence parameter in the latent process $q_\xi(\phi)$

$$\phi^* = s_1(\xi, z_1^*), \text{ with } s_1() \text{ and } z_1^* \text{ such that } \sigma^{2*} \sim q_\xi(\sigma^2 \mid \phi^*), \tag{81}$$

$$\sigma^{2*} = s_2(\xi, z_2^*, \phi^*), \text{ with } s_2() \text{ and } z_2^* \text{ such that } \sigma^{2*} \sim q_\xi(\sigma^2 \mid \phi^*), \tag{82}$$

$$\hat{C}_1 = \nabla_\xi[s_1(\xi, z_1^*)]\nabla_\phi[T_1(\phi^*)], \tag{83}$$

$$\hat{g}_1 \approx \nabla_\xi[s_1(\xi, z_1^*)]\{\nabla_\phi[\log p(\phi^*) + \log q_\xi(y \mid \phi^*, \sigma^{2*}) - \log q_\xi(\sigma^{2*} \mid \phi^*)]\}. \tag{84}$$

Second, we update the block for the variance of the latent process $q_\xi(\sigma^2 \mid \phi)$

$$\hat{C}_2 = \nabla_\xi[s_2(\xi, z_2^*, \phi^*)]\nabla_{\sigma^2}[T_2(\sigma^{2*})] \tag{85}$$

$$\hat{g}_2 \approx \nabla_{\xi}[s_2(\xi, z_2^{2*}, \phi^*)]\nabla_{\sigma^2}[\log p(\sigma^{2*}) + \log q_{\xi}(y \mid \phi^*, \sigma^{2*})], \tag{86}$$

where $T_2(\sigma^{2*})$ are the sufficient statistics of $q_{\xi}(\sigma^2 \mid \phi)$. The last update is the update of the likelihood approximation

$$a_{t+1} = (1-\omega)a_t + \omega\mathbb{E}_{q_{\xi}(f|\phi^*,\sigma^{2*})}[\nabla_f \log p(y \mid f)], \tag{87}$$

$$z_{t+1} = (1-\omega)z_t + \omega\mathbb{E}_{q_{\xi}(f|\phi^*,\sigma^{2*})}[f], \tag{88}$$

$$\xi_{6,t+1} = (1-\omega)\xi_{6,t} - \omega\mathbb{E}_{q_{\xi}(f|\phi^*,\sigma^{2*})}[\nabla_f\nabla_f \log p(y \mid f)], \tag{89}$$

$$\xi_{7,t+1} = a_{t+1} + \xi_{6,t+1}z_{t+1}. \tag{90}$$

For more extensive derivations of the updates, we refer the reader to [6]. Further, one might wonder how the latent process is estimated in this procedure. Salimans et al. [6] propose using the Kalman filter to estimate the filtering distribution. Even though it is a valid approach that is also undertaken in quasi-maximum likelihood method [41], its weakness lies in the linearization of the observation equation which implies that the distribution of the noise process is not longer Gaussian.

4. Results

In this section, we present results for the comparison of the discussed methods. We compare two particle filters (bootstrap and auxiliary particle filters) on the basis of bias, variance and on the estimated effective number of particles. We choose the better performing procedure of the two for using in the particle Metropolis–Hastings algorithm. We compare particle Metropolis–Hastings (PMH), Riemann Manifold Hamiltonian Monte Carlo (RMHMC), integrated nested Laplace approximation (INLA), and fixed-form variational Bayes (VB) on the basis of how well the posterior distributions obtained with these methods capture the ground truth (e.g., true parameter values). The ability of the methods to recover ground truth is assessed based on five simulated data sets with different underlying parameters. We additionally provide effective sample sizes for the comparison of the sampling methods (PMH and RMHMC). For illustration purposes, we also provide comparison on two real-world data sets.

4.1. Variance of the Estimated Likelihood

As we mentioned before, the marginal likelihood can be approximated sequentially through particle filtering. The marginal likelihood approximation of the parameters $\boldsymbol{\theta}$ reads

$$p(y_{1:T}|\boldsymbol{\theta}) \approx \prod_t \hat{p}(y_t|y_{1:t-1}, \boldsymbol{\theta}), \tag{91}$$

where the right hand side is obtained by running particle filter presented in Algorithm A1. In practice, usually the log-likelihood

$$\log p_{\boldsymbol{\theta}}(y_{1:T}) = \log p_{\boldsymbol{\theta}}(y_1) + \sum_{t=2}^{T} \log p_{\boldsymbol{\theta}}(y_t|y_{1:t-1}) \tag{92}$$

is estimated for the purpose of numerical stability (as the product of small weights would lead to unstable results). The estimate of the log-likelihood is the by-product of the particle filtering, as it is the average over log-weights that are assigned to the particles at every time step. In this section, we compare the bootstrap (BPF) and auxiliary particle filters (APF) in terms of bias, variance, and number of effective particles. Both of them can be used for obtaining simulated likelihood estimates, which can be further used in the particle

Metropolis–Hastings algorithm. We denote by $\hat{L}$ the estimate of the likelihood obtained with a particle filter. Then, the bias and the variance can be estimated as follows

$$Bias = 5000^{-1} \sum_{i=1}^{K} \sum_{j=1}^{M} (\log \hat{L}^j - \log \bar{L}(y^i)), \tag{93}$$

$$Variance = 5000^{-1} \sum_{i=1}^{K} \sum_{j=1}^{M} (\log \hat{L}^j - \log \bar{L}(y^i))^2, \tag{94}$$

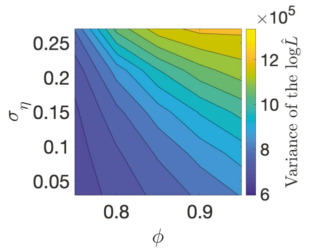
where y_i is the i-th time series, and $\log L$ is the "true" log-likelihood value. For the comparison, we use $K = 50$ different time series generated from the stochastic volatility model and $M = 100$ Monte Carlo iterations. We use $N = 100$, $N = 1000$, and N = 10,000 number of particles for this study. As the true value of the likelihood is not available, we substitute it with an estimate that is obtained with N = 1,000,000 number of particles. First, we conduct the analysis of the variance of the estimated likelihood in true parameter values as discussed in [24]. The authors of, ref. [27] showed theoretically that the asymptotic variance is not always smaller for the APF in comparison to the BPF. We run additional simulation studies to examine whether the variance of the estimated likelihood varies in the parameter space. Since we are interested in using the estimated likelihood in the Markov Chain Monte Carlo setting, it is relevant how the variance behaves in different points of the parameter space. If we start far away from the true value and the variance of the estimated likelihood is larger in that part of the parameter space, it can affect the convergence and calibration of the algorithm. Table 1 shows variance of the estimated likelihood for bootstrap and auxiliary particle filters. N indicates the number of the particles that we used for the estimation of the likelihood. It is clear that, on average, APF performs better in terms of the variance of the estimated likelihood. Table 2 indicates results for a similar experiment, but on the level of individual times series. We consider different data-generating processes and find that, in particular, higher variance of the latent volatility process is associated with higher variance of the estimated likelihood. Finally, in Figures 2–4, we illustrate that the variance of the estimated likelihood changes depending on the location in the parameter space, and these changes can be specific to a data-generating process. These figures correspond to the experiments with time series 2, 3, and 4 from Table 2. The likelihood was estimated with $N = 1000$ particles. We observe that the variance of the latent process has a strong effect on the landscape of the variance of the estimated likelihood in the parameter space. From Figure 4c,d, we see that the variance of the estimated likelihood obtained with the bootstrap particle filter appears to be more strongly affected by the location in the parameter space than the variance of the estimated likelihood obtained with the auxiliary particle filter. In Figure 3, we observe that the variance of the estimated likelihood is affected by the scale parameter β in the case of auxiliary particle filter, but not so much in the case of the bootstrap particle filter. Thus, initialization of PMCMC and the choice of number of particles should be considered with care for the optimal performance of the algorithm as the variance of the estimated likelihood can differ in the parameter space, and these changes can vary across different time series.

Table 1. Variance, bias, and number of effective particles N_{eff} for the estimated likelihood with bootstrap particle filter (BPF) and auxiliary particle filter (APF) averaged over 50 time series. N_{eff} is computed as in Equation (22). Variance and bias are computed as in Equations (93) and (94).

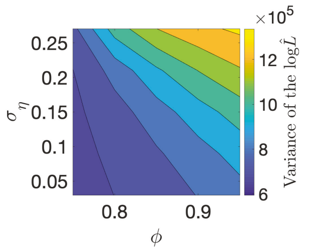
	Variance		**Bias**		N_{eff}	
	BPF	**APF**	**BPF**	**APF**	**BPF**	**APF**
N = 100	778.46	3.36	19.95	−0.68	64.07	23.52
N = 1000	805.79	0.20	20.32	−0.07	640.23	224.65
N = 10,000	808.40	0.02	20.36	−0.01	6402.11	2233.59

Table 2. Variance of the estimated likelihood for 10 different data-generating processes (TS). We consider different settings for the number of particles N.

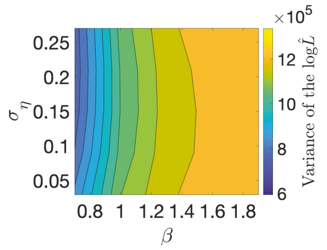
	Bootstrap Particle Filter			**Auxiliary Particle Filter**			**True Parameters**		
TS	***N* = 100**	***N* = 1000**	***N* = 10,000**	***N* = 100**	***N* = 1000**	***N* = 10,000**	$\boldsymbol{\beta}$	$\boldsymbol{\phi}$	$\boldsymbol{\sigma_\eta}$
1	511.26	556.01	559.85	2.436	0.251	0.017	0.36	0.96	0.13
2	128.50	138.05	139.19	0.688	0.040	0.005	1.18	0.95	0.08
3	18,143.0	18,883.0	18,945.0	259.3	9.669	0.993	1.93	0.89	0.43
4	79.085	84.045	83.510	0.517	0.045	0.005	0.9	0.96	0.07
5	51.753	52.112	52.078	0.066	0.005	0.001	0.93	0.76	0.08
6	4.2531	4.2704	4.2961	0.026	0.003	0.000	1.43	0.79	0.04
7	2684.2	2780.2	2790.5	5.076	0.339	0.032	1.58	0.91	0.27
8	137.22	140.65	140.48	0.210	0.018	0.002	0.09	0.85	0.11
9	8.5151	13.541	13.419	0.808	0.067	0.008	0.94	0.98	0.10
10	3601.2	3665.6	3670.4	25.32	2.614	0.291	0.55	0.84	0.21



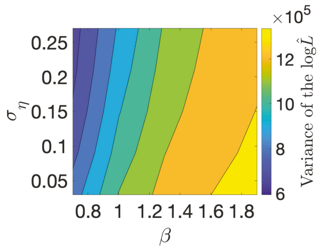
a Variance of $log\hat{L}$ estimated with BPF, with $\beta = 1.0$



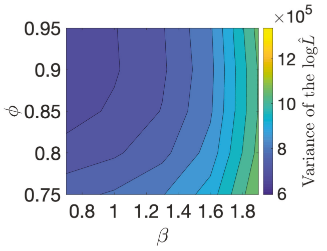
b Variance of $log\hat{L}$ estimated with APF, with $\beta = 1.0$



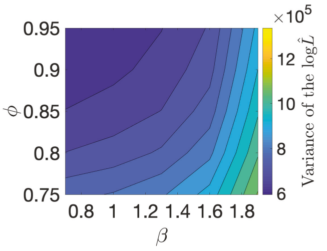
c Variance of $log\hat{L}$ estimated with BPF, with $\phi = 0.95$



d Variance of $log\hat{L}$ estimated with APF, with $\phi = 0.95$

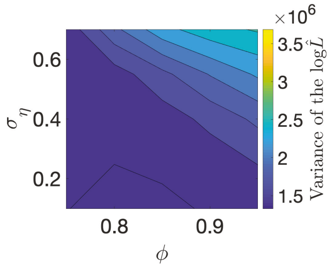


e Variance of $log\hat{L}$ estimated with BPF, with $\sigma_\eta = 0.15$

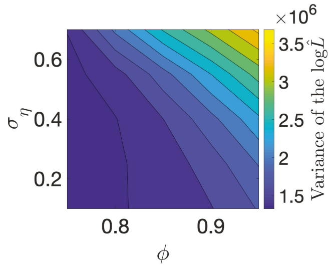


f Variance of $log\hat{L}$ estimated with BPF, with $\sigma_\eta = 0.15$

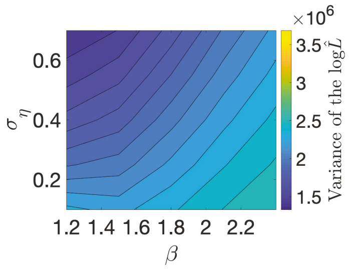
Figure 2. Variance of the estimated likelihood in different points of the parameter space for $TS = 2$ from Table 2.



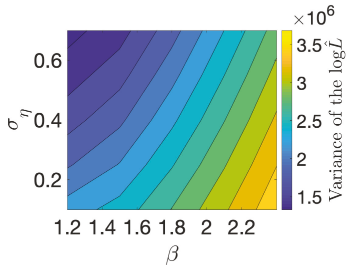
a Variance of $log\hat{L}$ estimated with BPF, with $\beta = 1.9$



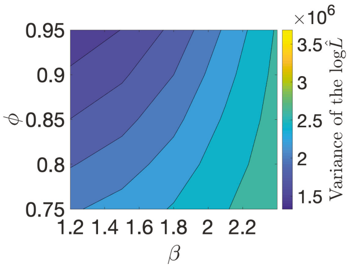
b Variance of $log\hat{L}$ estimated with APF, $\beta = 1.9$



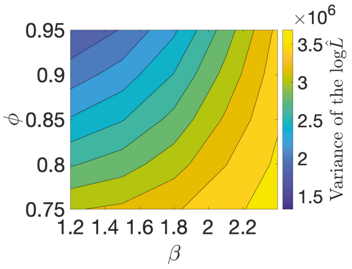
c Variance of $log\hat{L}$ estimated with BPF, $\phi = 0.9$



d Variance of $log\hat{L}$ estimated with APF, $\phi = 0.9$

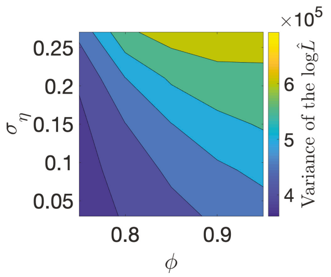


e Variance of $log\hat{L}$ estimated with BPF, $\sigma_\eta = 0.45$

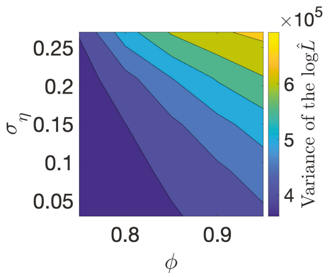


f Variance of $log\hat{L}$ estimated with APF, $\sigma_\eta = 0.45$

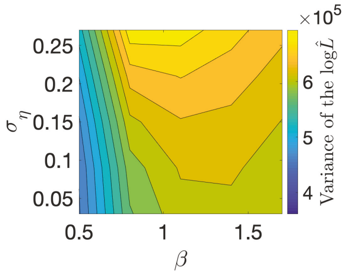
Figure 3. Variance of the estimated likelihood in different points of the parameter space for $TS = 3$ from Table 2.



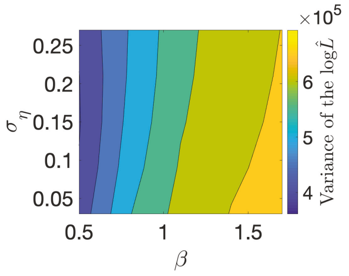
a Variance of $log\hat{L}$ estimated with BPF, with $\beta = 1.0$



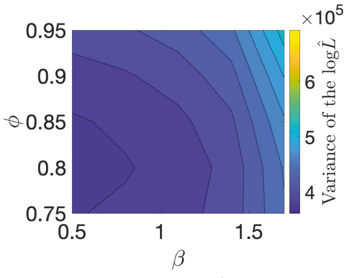
b Variance of $log\hat{L}$ estimated with APF, with $\beta = 1.0$



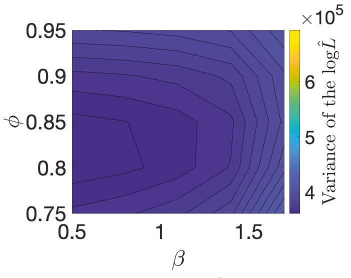
c Variance of $log\hat{L}$ estimated with BPF with $\phi = 0.95$



d Variance of $log\hat{L}$ estimated with APF, with $\phi = 0.95$



e Variance of $log\hat{L}$ estimated with BPF, with $\sigma_\eta = 0.05$



f Variance of $log\hat{L}$ estimated with APF, with $\sigma_\eta = 0.05$

Figure 4. Variance of the estimated likelihood in different points of the parameter space for $TS = 4$ from Table 2.

4.2. Particle Metropolis–Hastings and Riemann Manifold Hamiltonian Monte Carlo

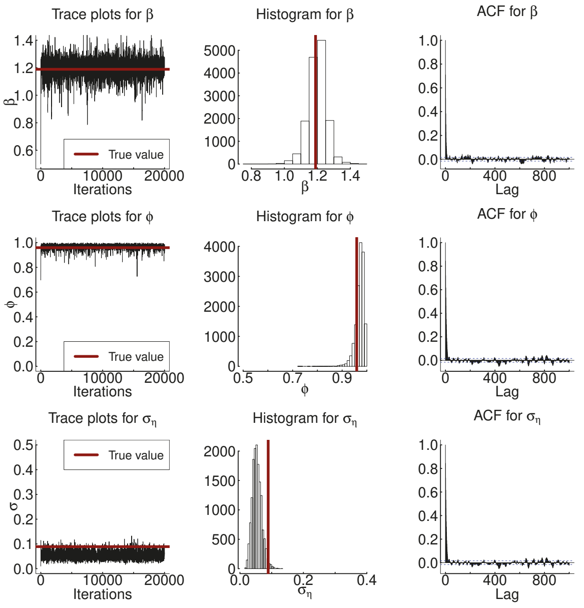
In this section, we compare particle Metropolis–Hastings (PMH) and Riemann Manifold Hamiltonian Monte Carlo. We evaluate the algorithms based on how well they recover the true parameters of the model β, ϕ, and σ_η and on the basis of the effective sample size. We obtained 20,000 samples and discarded the first 1000 as burn-in. Further, Figure 5 and Figures A1–A4 show results for both samplers: trace plots, histograms, and autocorrelation function are depicted. Table 3 presents the moments and highest posterior density intervals for the parameters of the model. The marginal likelihood in PMH was estimated with auxiliary particle filter as discussed in [23]. The Metropolis–Hastings part of the algorithm was calibrated to achieve 20–40% acceptance rate. RMHMC was implemented as in [5] with openly available implementation of the method by the authors. Both PMH and RMHMC require careful calibration of the step-size, and RMHMC additionally needs calibration of the number of the leapfrog steps; thus in Table A1, we additionally present results for the no-u-turn sampler (NUTS). NUTS is an extension of HMC algorithm that allows one to tune the algorithm automatically. From Figure 5 and Figures A1–A4, we observe that autocorrelation of the samples indicated in the third column of the plots decreases faster for PMH than for RMHMC, in particular for the parameters ϕ and σ_η. Effective sample size is similarly high for parameter β for both samplers. Effective sample size for the parameters ϕ and σ_η is lower in the case of RMHMC. However, if we compute the ESS per second as presented in the last column of Table 3, this advantage disappears. This result is not surprising since PMH is the most computationally intensive procedure we are considering. Both the likelihood and the posterior use sequential sampling methods, which makes computations very demanding. Nevertheless, PMH allows us to recover the underlying parameters more accurately. In particular, in most of the presented examples, true variance of the latent volatility process lies inside the 95% highest posterior density interval for PMH. RMHMC tends to overestimate this parameter. As Table A1 indicates, the highest posterior density intervals obtained with NUTS are larger than those obtained with PMH and RMHMC. Number of gradient evaluations for RMHMC are 69718, 70042, 69802, 69801, and 70041 for Experiments 1–5, respectively.

Table 3. Posterior moments for the samples obtained with particle Metropolis–Hastings (PMH) and Riemann Manifold Hamiltonian Monte Carlo (RMHMC) for the parameters β, ϕ and σ_η of the stochastic volatility model. Experiments 1, 2, 3, 4, and 5 correspond to TS 2, 4, 5, 9, and 10 from Table 2.

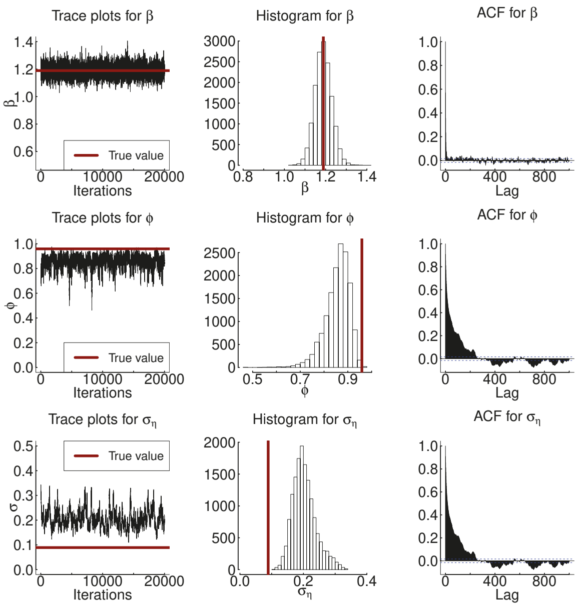
Experiment 1: Posterior Moments Obtained with PMH							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	1.1985	1.2140	1.0771	1.3034	1.189	2783.2	0.054
ϕ	0.9713	0.9732	0.9392	0.9987	0.959	1516.8	0.029
σ_η	0.0537	0.0514	0.0260	0.0828	0.089	2280.7	0.044
Experiment 1: Posterior Moments Obtained with RMHMC							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	1.18628	1.16608	1.10609	1.26526	1.189	3580.7	50.57
ϕ	0.83824	0.78390	0.71272	0.93643	0.959	188.6244	2.66
σ_η	0.21889	0.17203	0.14846	0.30072	0.0828	93.5265	1.32
Experiment 2: Posterior Moments Obtained with PMH							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.8629	0.8780	0.7870	0.9385	0.903	2909.6	0.045
ϕ	0.9644	0.9843	0.9077	0.9987	0.967	1157.2	0.017
σ_η	0.0534	0.0504	0.0233	0.0912	0.076	1694.3	0.026
Experiment 2: Posterior Moments Obtained with RMHMC							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.85071	0.82888	0.79927	0.90418	0.903	4108.8	76.26
ϕ	0.79293	0.73992	0.62454	0.92638	0.967	186.8240	3.46
σ_η	0.22291	0.23163	0.14751	0.30825	0.076	72.6236	1.34

Table 3. *Cont.*

Experiment 3: Posterior Moments Obtained with PMH							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.9579	0.9623	0.9090	1.0043	0.938	3330.7	0.071
ϕ	0.9079	0.9583	0.7974	0.9885	0.764	2178.9	0.046
σ_η	0.0429	0.0317	0.0184	0.0725	0.08	2690.1	0.057
Experiment 3: Posterior Moments Obtained with RMHMC							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.94395	0.90574	0.89572	0.99246	0.938	8112.2	164.83
ϕ	0.67333	0.43245	0.42649	0.88097	0.764	116.9159	2.37
σ_η	0.19759	0.18142	0.13385	0.27098	0.08	98.9095	2.0
Experiment 4: Posterior Moments Obtained with PMH							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.8865	0.8932	0.7297	1.0485	0.942	3046.7	0.057
ϕ	0.9826	0.9892	0.9672	0.9961	0.981	1781.9	0.033
σ_η	0.1100	0.1011	0.0702	0.1503	0.1	1609.9	0.031
Experiment 4: Posterior Moments Obtained with RMHMC							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.90560	0.80476	0.74840	1.07499	0.942	445.7	9.22
ϕ	0.96391	0.96149	0.93865	0.98672	0.981	336.5	6.96
σ_η	0.17588	0.15766	0.13452	0.22005	0.1	130.0	2.69
Experiment 5: Posterior Moments Obtained with PMH							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.5836	0.5893	0.5359	0.6293	0.553	2733.8	0.0552
ϕ	0.9497	0.9970	0.8890	0.9970	0.840	2733.8	0.0552
σ_η	0.0756	0.0525	0.0294	0.1331	0.215	1589.1	0.032
Experiment 5: Posterior Moments Obtained with RMHMC							
	Mean	Mode	95% HPD_l	95% HPD_u	True	ESS	ESS/s
β	0.56805	0.57215	0.53372	0.60258	0.553	3276.6	75.41
ϕ	0.77895	0.79364	0.60134	0.92411	0.840	130.7	3.0
σ_η	0.22905	0.21974	0.14799	0.31755	0.215	64.9	1.49



a Trace plots (left), histograms (middle) and ACF plots (right) obtained with Particle Metropolis-Hastings



b Trace plots (left), histograms (middle) and ACF plots (right) obtained with Riemann manifold Hamiltonian Monte Carlo

Figure 5. Results of the sampling from the posterior distribution with PMH and RMHMC for $TS = 2$ from Table 2. The first column corresponds to the trace plots, the middle column to histograms obtained with the samples from the posterior distribution, and the last column corresponds to autocorrelation function for the samples.

4.3. Integrated Nested Laplace Approximation

We provide two simulation studies for the integrated nested Laplace approximation. First, we replicate and extend the simulation study provided in [38] by analyzing data-generating processes with different values of μ and σ_η since the estimation of the variance parameter appears to be a challenge for existing methods. Table 4 replicates results from [38] with the parametrization of the model with the scale parameter μ, and Table 5 presents results for the parametrization with the scale parameter β and different data-generating processes. Both Monte Carlo studies are conducted with 1000 iterations. Our findings are comparable to those of [38]: the mean of the volatility process and the persistence parameter are estimated quite accurately, while the variance of the latent volatility process estimated with INLAis biased—usually, it is overestimated. Second, we provide the posterior moments for INLA similarly to Table 3 for PMH and RMHMC. These results are presented in Table 6. These results also suggest that the variance of the latent volatility process tends to be overestimated with INLA to a larger degree than with RMHMC, which also overestimates this parameter, as can be seen from the Table 3. Moreover, highest posterior density intervals for the parameter ϕ obtained with INLA are larger than those obtained with PMH and RMHMC.

Table 4. Bias and square root of the mean squared error for integrated nested Laplace approximation (INLA) parametrized with scale parameter μ.

μ_{true}	ϕ_{true}	$\sigma_{\eta true}$	bias (μ)	smse (μ)	bias (ϕ)	smse (ϕ)	bias (σ_η)	smse (σ_η)
0.1366	0.9	0.0186	−0.0672	0.01804	−0.6138	0.5403	0.29183	0.0912
−0.2143	0.9	0.0366	−0.0565	0.0092	−0.5803	0.4953	0.2739	0.0815
−0.0658	0.9	0.0636	−0.0664	0.0151	−0.5817	0.5176	0.2494	0.0704
−0.0289	0.95	0.0186	−0.0675	0.0151	−0.6354	0.5550	0.2883	0.0900
0.0203	0.95	0.0366	−0.0705	0.0207	−0.5796	0.4843	0.2677	0.0795
−0.0630	0.95	0.0636	−0.0931	0.0333	−0.4142	0.3250	0.2129	0.0592
−0.0174	0.98	0.0186	−0.0763	0.0222	−0.6173	0.5458	0.2788	0.0874
0.1343	0.98	0.0366	−0.1534	0.0797	−0.4329	0.3650	0.2228	0.0654
0.0584	0.98	0.0636	−0.2596	0.1379	−0.2095	0.1710	0.1280	0.034

Table 5. Bias and square root of the mean squared error for INLA parametrized with scale parameter β.

β_{true}	ϕ_{true}	$\sigma_{\eta true}$	bias (μ)	smse (μ)	bias (ϕ)	smse (ϕ)	bias (σ_η)	smse (σ_η)
0.367	0.965	0.134	−0.2295	0.0526	−0.0083	0.0001	0.4725	0.2233
1.188	0.959	0.088	0.2000	0.0400	−0.1909	0.0364	0.3466	0.1201
1.937	0.897	0.433	1.6162	2.6122	−0.0021	0.0000	0.5926	0.3512
0.902	0.966	0.075	−0.1888	0.0356	−0.3049	0.0930	0.3153	0.0994
0.938	0.764	0.080	−0.0515	0.0026	−0.4921	0.2421	0.1468	0.0215
1.435	0.793	0.048	0.6014	0.3616	−0.3872	0.1499	0.2155	0.0464
1.588	0.919	0.275	0.2539	0.0644	0.0202	0.0004	0.4786	0.2291
0.092	0.857	0.109	−0.0841	0.0070	−0.7490	0.5610	0.1341	0.0179
0.942	0.980	0.100	−0.3246	0.1054	0.0192	0.0003	95.8384	9185. 0
0.553	0.840	0.214	−0.2384	0.0568	−0.3904	0.1524	0.1918	0.0368

Table 6. Posterior results for estimation of the stochastic volatility (SV) model with INLA. Experiments 1, 2, 3, 4, and 5 correspond to *TS* 2, 4, 5, 9, and 10 from Table 2.

			Experiment 1		
	mean	mode	95% HPD_l	95% HPD_u	true
β	1.1786	1.1753	1.0937	1.2797	1.189
ϕ	0.7717	0.9986	0.2029	0.9945	0.959
σ^η	0.4368	0.4918	0.3048	0.6665	0.089
			Experiment 2		
	mean	mode	95% HPD_l	95% HPD_u	true
β	0.8457	0.8426	0.7807	0.9367	0.903
ϕ	0.7480	0.9993	−0.0844	0.9996	0.967
σ_η	0.4155	0.9111	0.2631	0.7726	0.076
			Experiment 3		
	mean	mode	95% HPD_l	95% HPD_u	true
β	0.9446	0.9403	0.8672	1.0798	0.938
ϕ	0.3827	0.9999	−0.7491	0.9999	0.764
σ_η	0.2328	4.5772	0.1327	0.5202	0.080
			Experiment 4		
	mean	mode	95% HPD_l	95% HPD_u	true
β	0.0892	0.0893	0.0847	0.0939	0.942
ϕ	0.1266	1.0000	−0.9470	0.9994	0.981
σ_η	0.2405	0.3196	0.1407	0.4447	0.100
			Experiment 5		
	mean	mode	95% HPD_l	95% HPD_u	true
β	0.5614	0.5604	0.5286	0.5993	0.553
ϕ	0.4567	0.9980	−0.3055	0.9726	0.840
σ_η	0.4067	0.4538	0.2888	0.5680	0.215

4.4. Fixed-Form Variational Bayes

In this section, we discuss results for the simulation study with fixed-form variational Bayes. We consider the same time series as in the case of comparison between PMH and RMHMC. In Table 7, we present estimated variational parameters and in Figure 6, comparison of the posterior with fixed-form variational Bayes (in blue), RMHMC (histograms from the posterior samples), and INLA (green). It is clear that in some cases, the variational Bayes method performs quite well; in particular, parameter β is very well estimated in most of the cases. Only in Figure 6j is the approximate posterior for β far from the truth. The variance parameter is underestimated in all cases with VB; this is less severe in the cases when the true variance is relatively small. However, when the true variance is relatively large, the discrepancy between VB estimate and the true value increases, as can be seen from Figure 6o. We observe the opposite picture with INLA: it tends to overestimate the variance of the latent volatility process. Overestimation of the variance of the latent volatility process for stochastic volatility models with INLA has been previously reported in [38]. Additionally, it is reported in [38] that this effect decreases with larger values of σ_η. The source of this has to be investigated further. RMHMC overestimates the variance to a lesser degree than INLA, and as can be seen from Figure 6o, this is also connected to the value of the ground truth for σ_η: with larger true value of σ_η, RMHMC provides more accurate results.

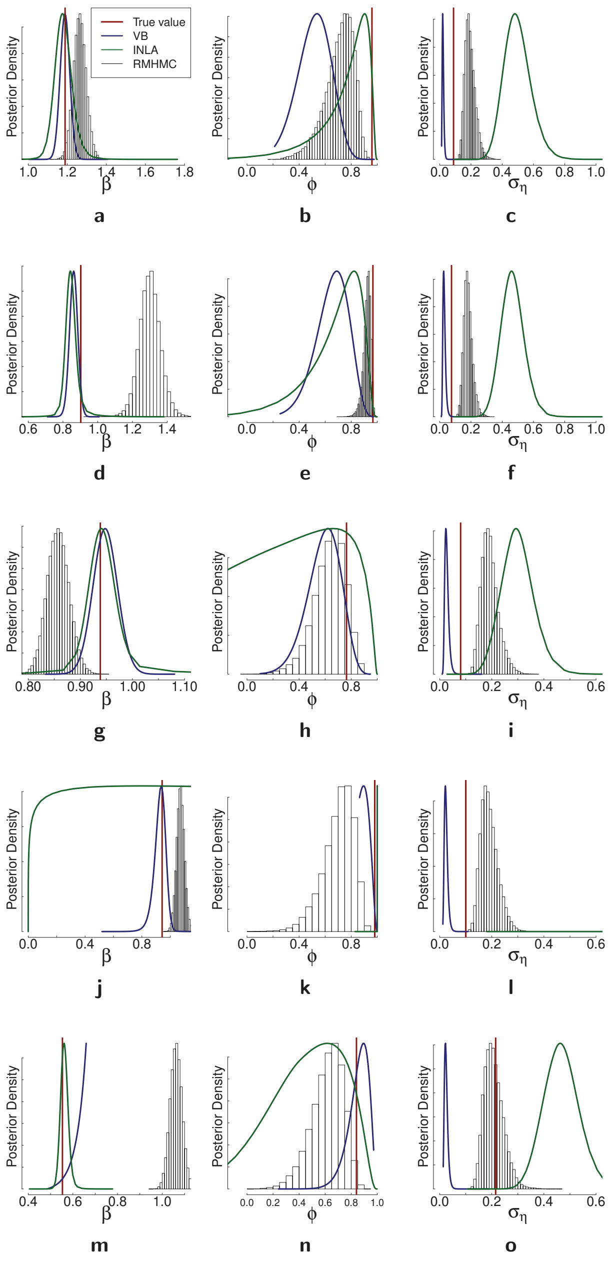


Figure 6. Illustration of the Fixed-form Variational Bayes in comparison to RMHMC and INLA. Subfigures illustrate the posterior distributions estimated with different methods for the different data-generating processes. (**a**–**c**) correspond to Experiment 1 from Tables 2 and 6, (**d**–**f**) correspond to Experiment 2, (**g**–**i**) correspond to Experiment 3, (**j**–**l**) correspond to experiment 4, and (**m**–**o**) correspond to Experiment 5. Red vertical lines indicate true parameter values.

Table 7. Parameters of the posterior distribution obtained with fixed-form Variational Bayes. Exp. 1–5 correspond to the Experiments 1–5 in Tables 3 and 6.

Exp.	ξ_1	ξ_2	ξ_3	ξ_4	ξ_5
1	31.1347	9.3573	19.1852	0.4658	−0.2051
2	28.2720	5.2039	12.6658	0.4510	−0.1812
3	30.3353	7.1038	13.9647	0.4563	−0.2039
4	38.8151	2.1501	17.9035	0.7568	−0.4296
5	38.8151	2.1501	17.9035	0.7568	−0.4296

4.5. Comparison of the Methods on the Real Data

In this section, we present posterior distributions of the parameters estimated with different Bayesian inference methods on two real-world time series. First, we consider the mean corrected log-returns of the Australian dollar against the US dollar. The data range from January 1994 to December 2003 with a total of 519 weekly observations. Resulting posterior distributions obtained with different inference methods are presented in Figure 7. Second, we consider daily log-returns for the DAX index from 3 January 2000 until 17 May 2001, which in total constitute 1000 observations. We provide descriptive statistics for both time series in Table A2. Resulting posterior distributions for this time series are presented in Figure 8. The main discrepancies between the methods are largest in the estimation of the parameter σ_η for both time series, and the results are consistent with the simulation studies in terms of the difference of these discrepancies. As we can see from Figures 7c and 8d, the posterior distribution of σ_η obtained with variational Bayes is concentrated in smaller values in comparison to the other methods. INLA suggests the higher values for σ_η in comparison to the other methods. The posterior samples obtained with RMHMC are concentrated in values higher than the ones obtained with PMH. Both sampling methods appear to give results larger than VB but smaller than INLA for the parameter σ_η.

In Table 8, we present results for efficient sample size (ESS) for both empirical applications and both samplers. Similarly to what is found in simulation studies, ESS is higher in the case of the PMH algorithm. However, if the computational time were taken into account, this advantage would have disappeared, similarly to the results in Table 2.

Table 8. Efficient sample size (ESS) for PMH and RMHMC in real-world time series applications: weekly log-returns for the exchange rate of Australian/US dollars and daily log-returns of DAX index.

	Australian/US Dollars Exchange Rate	
	ESS PMH	ESS RMHMC
β	3373.9	906.2
ϕ	1546.4	481
σ_η	2439	208.3
	DAX Index	
	ESS PMH	ESS RMHMC
β	3868.4	3962.3
ϕ	915.3	134.6
σ_η	2439	79.6

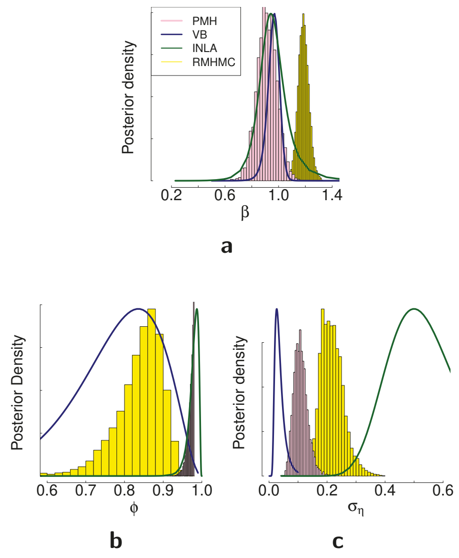


Figure 7. Comparison of PMH (pink), VB (blue), INLA (green), and RMHMC (yellow) on the weekly log-returns of the Australian dollar against the US dollar. Subfigures illustrate the posterior distributions for different parameters of the model obtained with different methods. (**a**) Corresponds to the posterior distribution for the parameter β. (**b**) Corresponds to the posterior distribution of the parameter ϕ. (**c**) Corresponds to the posterior distribution of the parameter σ_η.

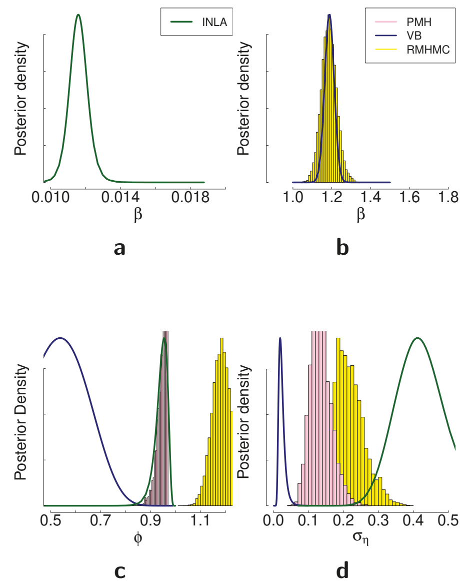


Figure 8. Comparison of PMH (pink), VB (blue), INLA (green), and RMHMC (yellow) on the daily log-returns of DAX index. Subfigures illustrate the posterior distributions for different parameters of the model obtained with different methods. (**a**,**b**) correspond to the posterior distribution of the parameter β. (**c**) corresponds to the posterior distribution of the parameter ϕ. (**d**) corresponds to the posterior distribution of the parameter σ_η.

5. Discussion

This paper reviewed multiple methods for the estimation of nonlinear state-space modes and stochastic volatility modes in particular that appear in Bayesian statistics and machine learning. We in particular focused on representative inference methods from different classes: methods that can recover the posterior distribution ‘exactly’ and the ones that build an approximation. We discussed which methods have the potential to be applied in a multivariate or high-dimensional situation and why they have this potential. Finally, we discovered that while stochastic volatility models are common for use in simulation studies for demonstrating the performance of the methods, usually not enough possible data-generating processes are considered to make a fair comparison. In particular, the performance of the methods is heavily connected to the variance of the latent volatility process.

State-space models can be powerful tools for modeling latent variables in different scientific fields. However, already for univariate time series, they are challenging to estimate. This paper’s main aim was to review and understand the existing classes of methods of estimation (targeting exact posterior or approximating it) and define the direction one can undertake for the estimation of *multivariate* nonlinear state-space models. The challenge arises from both statistical and computational perspectives. By this, we mean it is hard to develop methods that both provide sufficiently good results from the estimation point of view and are computationally feasible. We have reviewed a number of methods that allow a trade-off between these two aspects. In particular, we have considered particle Markov Chain Monte Carlo and reviewed multiple particle filtering approaches for this method, Riemann Manifold Langevin Hamiltonian Monte Carlo, Integrated Nested Laplace Approximation, and Variational Bayes. All these methods are equipped with the ability to estimate models with intractable likelihoods.

5.1. Sequential Monte Carlo

We compared the auxiliary particle filter with the bootstrap particle filter in terms of the variance of the estimated likelihood. We found that the auxiliary particle filter outperformed the bootstrap particle filter for most of the data-generating processes. As discussed in [27], auxiliary particle filter does not always have a smaller variance of the estimated likelihood. Additionally, we looked into how the variance of the estimated likelihood changes in the parameter space. We found that, in particular, the variance of the latent process affects the variance of the estimated likelihood. This implies that one has to find the balance for the number of particles used in Sequential Monte Carlo and a clever way of finding initial parameter values for the sampling from the posterior, especially when considering multivariate models. The advantage of the auxiliary particle filter from the methodological point of view is that it takes into account current observation y_t when constructing the proposal for the particles $q(h_t \mid h_{t-1}, y_t)$. The method that we did not include in our simulation study, but that possibly can solve the problem with the variance of the estimated likelihood, is the iterated auxiliary particle filter (iAPF): for the proposal of the particles, it uses not only current observation y_t, but all observations $q(h_t \mid h_{t-1}, y_{1:T})$. A backward sequential procedure with an optimization step is used in this proposal mechanism for the particles, which makes the algorithm computationally intensive. The multivariate application of the stochastic volatility model in [26] considers only diagonal case of the matrix Φ, and the proposed procedure for the particle proposals does not incorporate such dependence. While this method does introduce an additional computational burden on already computationally intensive method (particle Metropolis–Hastings), it is promising for getting state-of-the-art results for the task of parameter estimation.

5.2. Particle Metropolis-Hastings

Metropolis-Hastings is a general MCMC method that is easy to implement and works well for the univariate model. The estimation results are satisfying when it is properly calibrated, and good mixing of the chains is achieved. It works well in low-dimensional problems but is unlikely to be successful in the case of multivariate stochastic volatility models. Considering the non-diagonal matrix Φ in a five-dimensional case, we would have 45 parameters to be estimated. The random walk proposal would be very inefficient even with a reasonable sparseness assumption on Φ. Nevertheless, in the low-dimensional model, we get the best estimation results with particle Metropolis–Hastings, where the particle filtering scheme is chosen to be an auxiliary particle filter. From the methods considered in this paper, particle Markov Chain Monte Carlo methods are easiest to adapt to different specifications of the model and are easiest to implement.

5.3. Riemann Manifold Hamiltonian Monte Carlo Methods

Hamiltonian Monte Carlo is a very attractive method for high-dimensional problems as it allows us to explore the parameter space efficiently. In particular, the gain in efficiency comes from avoiding random walk behavior in the proposals. The disadvantage comes from the need of careful calibration since there is no principled way of choosing matrix $\boldsymbol{M}$. RMHMC avoids this problem by exploiting underlying geometry in the proposal mechanism. In our study, we notice that RMHMC results in good mixing of the Markov chains, and the method is generally easy to calibrate, but the estimation of the parameters is not very good. In particular, it appears that the variance of the latent volatility process is challenging for the method. It is not surprising that the PMH algorithm performs better in terms of parameter estimation since we use an auxiliary particle filter for the volatility process estimation and thus take current observation y_t for the particle proposals. RMHMC does not benefit from similar information when estimating model parameters. Therefore, improved estimation of the volatility process can be one of the directions for improving the performance of RMHMC for the parameter estimation of stochastic volatility models.

5.4. Variational Bayes

As one can see from the illustrative example, in some cases, variational Bayes performs quite well; however, there are also situations when it is far off from the underlying truth. The challenge with stochastic volatility models remains the same: it is difficult to estimate the latent states. In the approach of [6], this is done via Kalman filtering. Therefore, the drawback of linearization of the model will remain and will show in the final results. In this respect, the possible combination of VB and SMC can be of interest. Some advances in this direction have already been made [42].

5.5. Integrated Nested Laplace Approximation

Integrated Nested Laplace Approximation is another approach that works well considering how fast the method is, but it clearly overestimates the variance of the latent volatility process. Additionally, the sparse matrix computation that is used in univariate models is not applicable to the multivariate case. In the multivariate case, the precision matrix in Equation (67) is not sparse, and thus, the method does not benefit from fast sparse matrix computation. An approach that we have not considered in this paper is the Expectation Propagation algorithm. In particular, the authors of [43] propose a way to improve approximate marginals $p(x_t \mid \theta, y)$ in latent Gaussian fields by using EP. The motivation of the approach builds on the fact that EP can give better approximations than the Laplace approximation in this case. The improvements, however, would come at computational costs. In the univariate case, the extra computational costs do not play a significant role as the algorithm can be parallelized. However, it is hard to say how big the difference would be in the multivariate model, both in terms of improvement in the estimation and loss in computational speed.

6. Conclusions

We reviewed multiple Bayesian inference methods, which both target the exact posterior distribution and approximate it. By comparing methods on various data-generating processes, we notice that variational Bayes tends to underestimate the latent volatility process variance, while INLA and RMHMC, in the cases considered, overestimated this parameter. We also get similar disposition of the results on two real-world data sets. We achieved the best performance with PMH in terms of recovering ground truth and uncertainty quantification. In PMH, the particle filtering step was performed with an auxiliary particle filter. This indicates that filtering with look-ahead approaches, which include current (or future) observations into proposal machinery can improve the performance of the inference method. It is important to note that different data-generating processes for simulation studies would indicate different performance results. Thus, we stress that when using stochastic volatility models, more than one data-generating process should be considered for methods comparison. This practice would allow indicating in which situation a method can fail or perform differently. Our results indicate that fixed-form variational Bayes tends to underestimate the variance of the latent process, while RMHMC and INLA overestimated this parameter. To estimate the stochastic volatility model in the multivariate case, the combination of different strategies appears to be necessary. In a high-dimensional case, the random-walk proposal would become extremely inefficient. At the same time, approximate methods lose their outstanding computational advantage (for example, INLA), and the implementation of these methods in the multivariate case is not straightforward.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The author sincerely thanks anonymous referees for constructive and useful comments. Their suggestions helped improve and clarify this manuscript.

Conflicts of Interest: The author declares no conflict of interest.

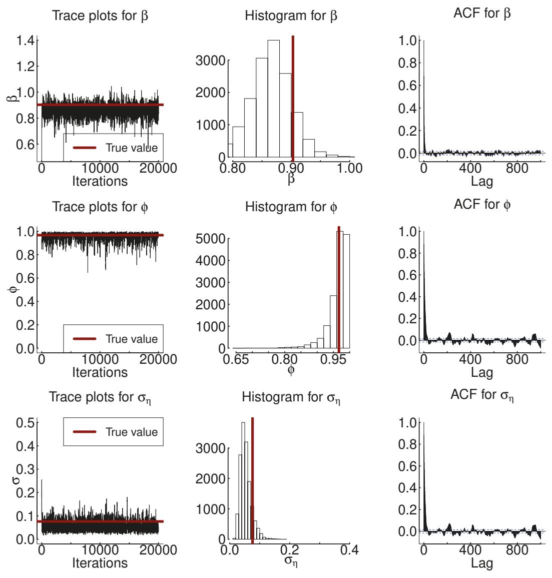
Abbreviations

The following abbreviations are used in this manuscript:

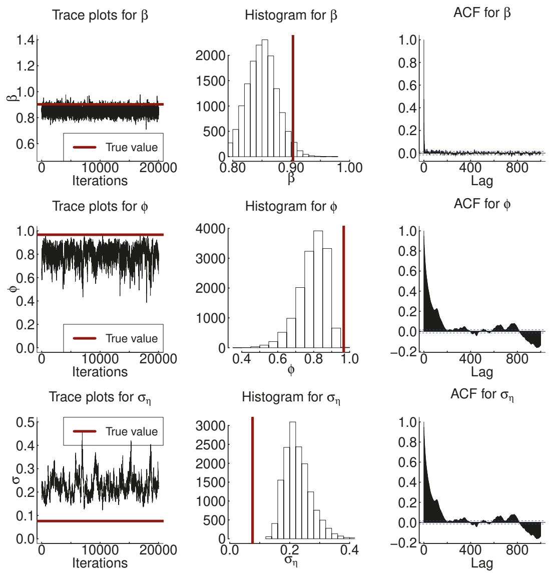
SV	Stochastic volatility
SMC	Sequential Monte Carlo
iAPF	Iterated Auxiliary Particle Filter
PMCMC	Particle Markov Chain Monte Carlo
PMH	Particle Metropolis-Hastings
HMC	Hamiltonian Monte Carlo
RMHMC	Riemann Manifold Hamiltonian Monte Carlo
VB	Variational Bayes
INLA	Integrated Nested Laplace Approximation

Appendix A

Appendix A.1.

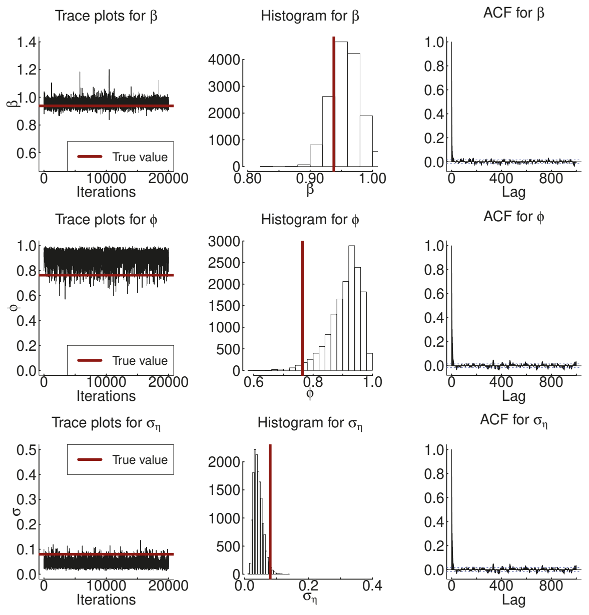


a Trace plots (left), histograms (middle) and ACF plots (right) obtained with Particle Metropolis-Hastings

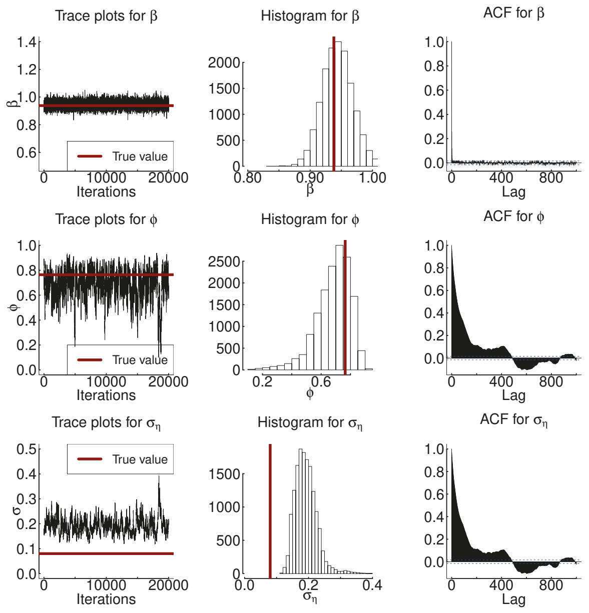


b Trace plots (left), histograms (middle) and ACF plots (right) obtained with Riemann manifold Hamiltonian Monte Carlo

Figure A1. Results of the sampling from the posterior distribution with PMH and RMHMC for $TS = 4$ from Table 2. The first column corresponds to the trace plots, the middle column to histograms obtained with the samples from the posterior distribution, and the last column to autocorrelation function for the samples.

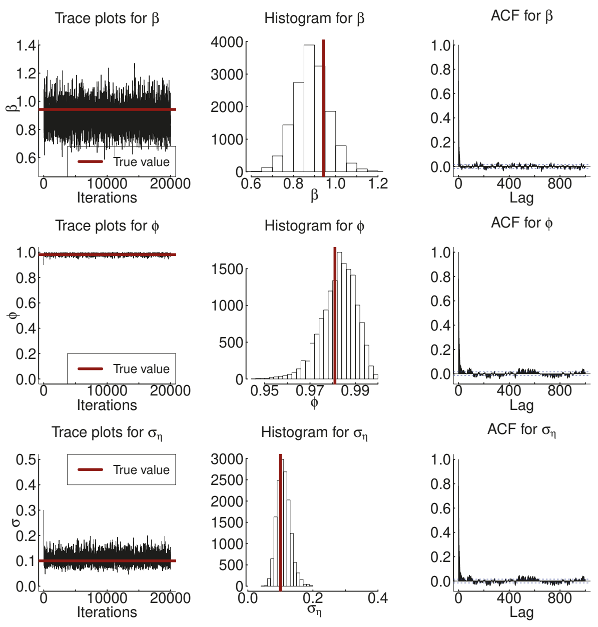


a Trace plots (left), histograms (middle) and ACF plots (right) obtained with Particle Metropolis-Hastings

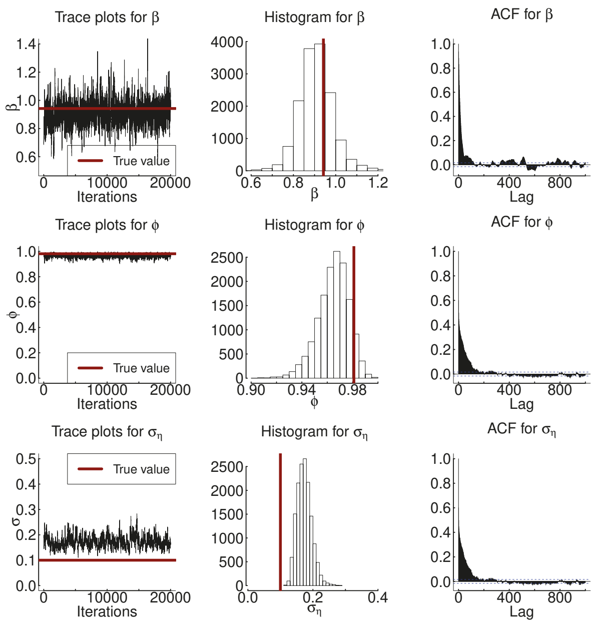


b Trace plots (left), histograms (middle) and ACF plots (right) obtained with Riemann manifold Hamiltonian Monte Carlo

Figure A2. Results of the sampling from the posterior distribution with PMH and RMHMC for $TS = 5$ from Table 2. The first column corresponds to the trace plots, the middle column to histograms obtained with the samples from the posterior distribution, and the last column to autocorrelation function for the samples.

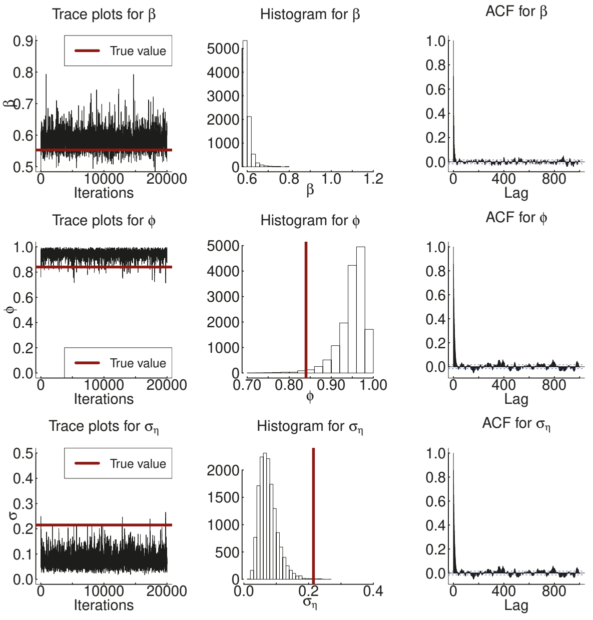


a Trace plots (left), histograms (middle) and ACF plots (right) obtained with Particle Metropolis-Hastings

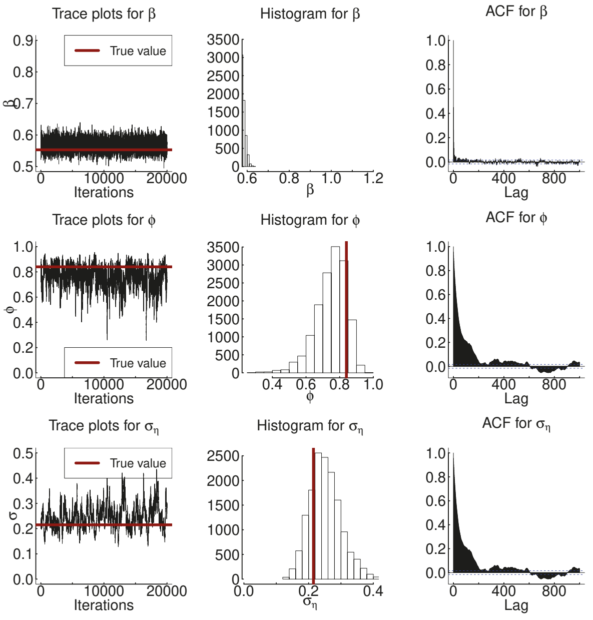


b Trace plots (left), histograms (middle) and ACF plots (right) obtained with Riemann manifold Hamiltonian Monte Carlo

Figure A3. Results of the sampling from the posterior distribution with PMH and RMHMC for $TS = 9$ from Table 2. The first column corresponds to the trace plots, the middle column to histograms obtained with the samples from the posterior distribution, and the last column to autocorrelation function for the samples.



a Trace plots, histograms and ACF plots obtained with Particle Metropolis-Hastings



b Trace plots, histograms and ACF plots obtained with Riemann manifold Hamiltonian Monte Carlo

Figure A4. Results of the sampling from the posterior distribution with PMH and RMHMC for $TS = 10$ from Table 2. The first column corresponds to the trace plots, the middle column to histograms obtained with the samples from the posterior distribution, and the last column to autocorrelation function for the samples.

Appendix A.2.

Although we do not give details for the NUTS sampler [44] in the main text, we present here experiments for this sampler using the same experiments as in the main text. The results in Table A1 are obtained with the sampler implemented in RStan [45]. The method provides large confidence intervals for the parameters ϕ and σ_η. Similarly to RMHMC, the variance of the latent volatility process tends be overestimated based on the mean and the mode of the posterior distribution. The confidence intervals obtained with NUTS appear to be quite large, especially for the parameters ϕ and σ_η. The multivariate version of the stochastic volatility model can provide additional challenges since different parameters might require different step sizes and the sampler can get stuck in the regions of space where a small step size is needed to achieve target acceptance rate. In the univariate case, NUTS appears to be more efficient than both PMH and RMHMC as can be seen from the last two columns of Table A1. The applicability of this particular implementation can be limited due to the large 95% highest posterior intervals as uncertainty about parameters ϕ and σ_η is very large in most cases.

Table A1. Posterior results for estimation of the SV model with INLA. Experiments 1, 2, 3, 4, and 5 correspond to *TS* 2, 4, 5, 9, and 10 from Table 2.

	mean	mode	95% HPD_l	95% HPD_u	true	ESS	ESS/s
Experiment 1							
β	1.1785	1.1779	1.1053	1.2567	1.189	18794	87.31
ϕ	0.6266	0.8366	0.0886	0.9773	0.959	18704	86.89
σ_η	0.3117	0.2985	0.0961	0.5397	0.089	18534	86.10
Experiment 2							
β	0.8498	0.8509	0.7978	0.9004	0.903	18694	42.08
ϕ	0.4033	0.7162	−0.3309	0.9912	0.967	17888	40.26
σ_η	0.2944	0.3064	0.0547	0.5127	0.076	18789	42.29
Experiment 3							
β	0.9510	0.9510	0.9074	0.9960	0.938	19001	42.31
ϕ	0.0188	−0.2263	−0.8129	0.8727	0.764	18168	40.46
σ_η	0.1259	0.0276	0.0000	0.2906	0.080	19001	42.31
Experiment 4							
β	0.9096	0.9042	0.7178	1.1550	0.942	18587	39.88
ϕ	0.9753	0.9784	0.9525	0.9965	0.981	18529	39.75
σ_η	0.1398	0.1317	0.0876	0.1923	0.100	18397	39.47
Experiment 5							
β	0.5631	0.5632	0.5329	0.5958	0.553	18819	42.40
ϕ	0.2804	0.3784	−0.3789	0.8847	0.840	18703	42.14
σ_η	0.3648	0.3910	0.1657	0.5445	0.215	18248	41.11

Appendix A.3.

Algorithm A1 is a generic particle filter. We use auxiliary version of it proposed in [23].

Algorithm A1 Approximation of marginal likelihood with ASIR algorithm.

```
1: Draw $N$ samples $h_0^{(i)}$ from the prior
       $$h_0^{(i)} \sim \pi(h_0 \mid \boldsymbol{\theta}), \ \ i = 1, \ldots, N \qquad \text{(A1)}$$
   and set $w_0^{(i)} = 1/N$, for all $i = 1, \ldots, N$.
2: For each $t = 1, \ldots, T$ do the following
3: Draw samples $h_t^{(i)}$ from the importance distribution
       $$h_t^{(i)} \sim q(h_t \mid h_{t-1}^{(i)}, y_{1:t}, \boldsymbol{\theta}), \ \ i = 1, \ldots, N. \qquad \text{(A2)}$$
4: Compute the following weights
       $$w_t^{(i)} = \frac{g(y_t \mid h_t^{(i)}, \boldsymbol{\theta}) f(h_t^{(i)} \mid h_{t-1}^{(i)}, \boldsymbol{\theta})}{q(h_t^{(i)} \mid h_{t-1}^{(i)}, y_{1:t}, \boldsymbol{\theta})} \qquad \text{(A3)}$$
   and compute the estimate of $p(y_t \mid y_{1:t-1}, \boldsymbol{\theta})$ as
       $$\hat{p}(y_t \mid y_{1:t-1}, \boldsymbol{\theta}) = \sum_i W_{t-1}^{(i)} w_t^{(i)}. \qquad \text{(A4)}$$
5: Compute normalized weights as
       $$W_t^{(i)} \propto W_{t-1}^{(i)} w_t^{(i)}. \qquad \text{(A5)}$$
6: If the effective number of particles is too low, perform resampling.
```

Appendix A.4.

Table A2. Descriptive statistics for time series from the empirical example in Section 3.5: daily log-returns for DAX index and weekly log-returns for Australian/US dollar exchange rate.

	DAX	Australia/US
Mean	−0.001	−0.04
Std.dev.	0.013	1.00
Skewness	0.202	0.06
Kurtosis	3.253	3.37

References

1. Shephard, N.; Torben, G. Stochastic volatility: Origins and overview. In *Handbook of Financial Time Series*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 233–254.
2. Platanioti, K.; McCoy, E.; Stephens, D. *A Review of Stochastic Volatility: Univariate and Multivariate Models*; Technical Report, Working Paper; Imperial College London: London, UK, 2005.
3. Asai, M.; McAleer, M.; Yu, J. Multivariate stochastic volatility: A review. *Econom. Rev.* **2006**, *25*, 145–175. [CrossRef]
4. Andrieu, C.; Doucet, A.; Holenstein, R. Particle Markov chain Monte Carlo methods. *J. R. Stat. Soc. Ser. B* **2010**, *72*, 269–342. [CrossRef]
5. Girolami, M.; Calderhead, B. Riemann Manifold Langevin and Hamiltonian Monte Carlo methods. *J. R. Stat. Soc. Ser. B (Stat. Methodol.)* **2011**, *73*, 123–214. [CrossRef]
6. Salimans, T.; Knowles, D.A. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Anal.* **2013**, *8*, 837–882. [CrossRef]
7. Rue, H.; Martino, S.; Chopin, N. Approximate Bayesian inference for latent Gaussian models by using Integrated Nested Laplace Approximations. *J. R. Stat. Soc. Ser. B* **2009**, *71*, 319–392. [CrossRef]
8. Mandelbrot, B.B. The variation of certain speculative prices. In *Fractals and Scaling in Finance*; Springer: New York, NY, USA, 1997; pp. 371–418.

9. Black, F. Studies of stock price volatility changes. In *Proceedings of the 1976 Meeting of the Business and Economic Statistics Section*; American Statistical Association: Washington, DC, USA, 1976; pp. 177–181.
10. Black, F.; Scholes, M. The pricing of options and corporate liabilities. *J. Political Econ.* **1973**, *81*, 637–654. [CrossRef]
11. Hull, J.; White, A. The pricing of options on assets with stochastic volatilities. *J. Financ.* **1987**, *42*, 281–300. [CrossRef]
12. Johnson, H.; Shanno, D. Option pricing when the variance is changing. *J. Financ. Quant. Anal.* **1987**, *22*, 143–151. [CrossRef]
13. Wiggins, J.B. Option values under stochastic volatility: Theory and empirical estimates. *J. Financ. Econ.* **1987**, *19*, 351–372. [CrossRef]
14. Heston, S.L. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Rev. Financ. Stud.* **1993**, *6*, 327–343. [CrossRef]
15. Campbell, J.Y.; Champbell, J.J.; Campbell, J.W.; Lo, A.W.; Lo, A.W.; MacKinlay, A.C. *The Econometrics of Financial Markets*; Princeton University Press: Princeton, NJ, USA, 1997.
16. Harvey, A.C.; Shephard, N. Estimation of an asymmetric stochastic volatility model for asset returns. *J. Bus. Econ. Stat.* **1996**, *14*, 429–434.
17. Jacquier, E.; Polson, N.G.; Rossi, P.E. Bayesian analysis of stochastic volatility models with fat-tails and correlated errors. *J. Econom.* **2004**, *122*, 185–212. [CrossRef]
18. Yu, J. On leverage in a stochastic volatility model. *J. Econom.* **2005**, *127*, 165–178. [CrossRef]
19. Kim, S.; Shephard, N.; Chib, S. Stochastic volatility: Likelihood inference and comparison with ARCH models. *Rev. Econ. Stud.* **1998**, *65*, 361–393. [CrossRef]
20. Gordon, N.J.; Salmond, D.J.; Smith, A.F. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F (Radar Signal Process.)* **1993**, *140*, 107–113. [CrossRef]
21. Doucet, A.; de Freitas, N.; Gordon, N. *Sequential Monte Carlo Methods in Practice*; Springer: New York, NY, USA, 2001.
22. Chopin, N. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *Ann. Stat.* **2004**, *32*, 2385–2411. [CrossRef]
23. Pitt, M.K.; Shephard, N. Filtering via simulation: Auxiliary particle filters. *J. Am. Stat. Assoc.* **1999**, *94*, 590–599. [CrossRef]
24. Scharth, M.; Kohn, R. Particle efficient importance sampling. *J. Econom.* **2016**, *190*, 133–147. [CrossRef]
25. Richard, J.F.; Zhang, W. Efficient high-dimensional importance sampling. *J. Econom.* **2007**, *141*, 1385–1411. [CrossRef]
26. Guarniero, P.; Johansen, A.M.; Lee, A. The iterated auxiliary particle filter. *J. Am. Stat. Assoc.* **2017**, *112*, 1636–1647. [CrossRef]
27. Johansen, A.M.; Doucet, A. A note on auxiliary particle filters. *Stat. Probab. Lett.* **2008**, *78*, 1498–1504. [CrossRef]
28. Särkkä, S. *Bayesian Filtering and Smoothing*; Cambridge University Press: Cambridge, UK, 2013; Volume 3.
29. Roberts, G.O.; Gelman, A.; Gilks, W.R. Weak convergence and optimal scaling of random walk Metropolis algorithms. *Ann. Appl. Probab.* **1997**, *7*, 110–120. [CrossRef]
30. Robert, C.P.; Casella, G. *Monte Carlo Statistical Methods*; Springer Texts in Statistics; Springer: New York, NY, USA, 2005.
31. Duane, S.; Kennedy, A.D.; Pendleton, B.J.; Roweth, D. Hybrid Monte Carlo. *Phys. Lett. B* **1987**, *195*, 216–222. [CrossRef]
32. Neal, R.M. MCMC Using Hamiltonian Dynamics. In *Handbook of Markov Chain Monte Carlo*; CRC Press: Boca Raton, FL, USA, 2011; Volume 2.
33. Betancourt, M. A Conceptual Introduction to Hamiltonian Monte Carlo. *arXiv* **2017**, arXiv:1701.02434.
34. Roberts, G.O.; Rosenthal, J.S. Optimal scaling of discrete approximations to Langevin diffusions. *J. R. Stat. Soc. Ser. B* **1998**, *60*, 255–268. [CrossRef]
35. Gelman, A.; Lee, D.; Guo, J. Stan: A probabilistic programming language for Bayesian inference and optimization. *J. Educ. Behav. Stat.* **2015**, *40*, 530–543. [CrossRef]
36. Salvatier, J.; Wiecki, T.V.; Fonnesbeck, C. Probabilistic programming in Python using PyMC3. *PeerJ Comput. Sci.* **2016**, *2*, e55. [CrossRef]
37. Martino, S.; Aas, K.; Lindqvist, O.; Neef, L.R.; Rue, H. Estimating stochastic volatility models using integrated nested Laplace approximations. *Eur. J. Financ.* **2011**, *17*, 487–503. [CrossRef]
38. Ehlers, R.; Zevallos, M. Bayesian Estimation and Prediction of Stochastic Volatility Models via INLA. *Commun. Stat.-Simul. Comput.* **2015**, *44*, 683–693. [CrossRef]
39. Martino, S. *Approximate Bayesian Inference for Multivariate Stochastic Volatility Models*; Technical Report; Department of Mathematical Sciences, Norwegian University of Science and Technology: Trondheim, Norway, 2007.
40. Šmídl, V.; Quinn, A. *The Variational Bayes Method in Signal Processing*; Springer Science & Business Media: Dordrecht, The Netherlands, 2006.
41. Ruiz, E. Quasi-maximum likelihood estimation of stochastic volatility models. *J. Econom.* **1994**, *63*, 289–306. [CrossRef]
42. Naesseth, C.; Linderman, S.; Ranganath, R.; Blei, D. Variational Sequential Monte Carlo. In Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics, Playa Blanca, Lanzarote, Canary Islands, Spain, 9–11 April 2018; pp. 968–977.
43. Cseke, B.; Heskes, T. Approximate marginals in latent Gaussian models. *J. Mach. Learn. Res.* **2011**, *12*, 417–454.
44. Hoffman, M.D.; Gelman, A. The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.* **2014**, *15*, 1593–1623.
45. Stan Development Team. RStan: The R Interface to Stan; R Package Version 2.21.2. Available online: http://mc-stan.org/ (accessed on 6 April 2021).

Article

PAC-Bayes Bounds on Variational Tempered Posteriors for Markov Models

Imon Banerjee [1], Vinayak A. Rao [1,*,†] and Harsha Honnappa [2,†]

[1] Department of Statistics, Purdue University, West Lafayette, IN 47907, USA; ibanerj@purdue.edu
[2] School of Industrial Engineering, Purdue University, West Lafayette, IN 47907, USA; honnappa@purdue.edu
[*] Correspondence: varao@purdue.edu
[†] These authors contributed equally to this work.

Abstract: Datasets displaying temporal dependencies abound in science and engineering applications, with Markov models representing a simplified and popular view of the temporal dependence structure. In this paper, we consider Bayesian settings that place prior distributions over the parameters of the transition kernel of a Markov model, and seek to characterize the resulting, typically intractable, posterior distributions. We present a Probably Approximately Correct (PAC)-Bayesian analysis of variational Bayes (VB) approximations to tempered Bayesian posterior distributions, bounding the model risk of the VB approximations. Tempered posteriors are known to be robust to model misspecification, and their variational approximations do not suffer the usual problems of over confident approximations. Our results tie the risk bounds to the mixing and ergodic properties of the Markov data generating model. We illustrate the PAC-Bayes bounds through a number of example Markov models, and also consider the situation where the Markov model is misspecified.

Keywords: ergodicity; Markov chain; probably approximately correct; variational Bayes

Citation: Banerjee, I.; Rao, V. A.; Honnappa, H. PAC-Bayes Bounds on Variational Tempered Posteriors for Markov Models. *Entropy* **2021**, *23*, 313. https://doi.org/10.3390/e23030313

Academic Editor: Pierre Alquier

Received: 8 February 2021
Accepted: 4 March 2021
Published: 6 March 2021

1. Introduction

This paper presents probably approximately correct (PAC)-Bayesian bounds on variational Bayesian (VB) approximations of fractional or tempered posterior distributions for Markov data generation models. Exact computation of either standard or tempered posterior distributions is a hard problem that has, broadly speaking, spawned two classes of computational methods. The first, Markov chain Monte Carlo (MCMC), constructs ergodic Markov chains to approximately sample from the posterior distribution. MCMC is known to suffer from high variance and complex diagnostics, leading to the development of variational Bayesian (VB) [1] methods as an alternative in recent years. VB methods pose posterior computation as a variational optimization problem, approximating the posterior distribution of interest by the 'closest' element of an appropriately defined class of 'simple' probability measures. Typically, the measure of closeness used by VB methods is the Kullback–Leibler (KL) divergence. Excellent introductions to this so-called *KL-VB* method can be found in [2–4]. More recently, there has also been interest in alternative divergence measures, particularly the α-Rényi divergence [5–7], though in this paper, we focus on the KL-VB setting.

Theoretical properties of VB approximations, and in particular asymptotic frequentist consistency, have been studied extensively under the assumption of an independent and identically distributed (i.i.d.) data generation model [4,8,9]. On the other hand, the common setting where data sets display temporal dependencies presents unique challenges. In this paper, we focus on homogeneous Markov chains with parameterized transition kernels, representing a parsimonious class of data generation models with a wide range of applications. We work in the Bayesian framework, focusing on the posterior distribution over the unknown parameters of the transition kernel. Our theory develops PAC bounds that link the ergodic and mixing properties of the data generating Markov chain to the Bayes risk associated with approximate posterior distributions.

Frequentist consistency of Bayesian methods, in the sense of concentration of the posterior distribution around neighborhoods of the 'true' data generating distribution, have been established in significant generality, in both the i.i.d. [10–12] and in the non-i.i.d. data generation setting [13,14]. More recent work [14–16] has studied fractional or tempered posteriors, a class of generalized Bayesian posteriors obtained by combining the likelihood function raised to a fractional power with an appropriate prior distribution using Bayes' theorem. Tempered posteriors are known to be robust against model misspecification: in the Markov setting we consider, the associated stationary distribution as well as mixing properties are sensitive to model parameterization. Further, tempered posteriors are known to be much simpler to analyze theoretically [14,16]. Therefore, following [14–16] we focus on tempered posterior distributions on the transition kernel parameters, and study the rate of concentration of variational approximations to the tempered posterior. Equivalently, as shown in [16] and discussed in Section 1.1, our results also apply to so-called α-variational approximations to standard posterior distributions over kernel parameters. The latter are modifications of the standard KL-VB algorithm to address the well-known problem of overconfident posterior approximations.

While there have been a number of recent papers studying the consistency of approximate variational posteriors [5,8,15] in the large sample limit, rates of convergence have received less attention. Exceptions include [9,15,17], where an i.i.d. data generation model is assumed. [15] establishes PAC-Bayes bounds on the convergence of a variational tempered posterior with fractional powers in the range $[0,1)$, while [9] considers the standard variational posterior case (where the fractional power equals 1). [17], on the other hand, establishes PAC-Bayes bounds for risk-sensitive Bayesian decision making problems in the standard variational posterior setting. The setting in [15] allows for model misspecification and the analysis is generally more straightforward than that in [9,17]. Our work extends [15] to the setting of a discrete-time Markov data generation model.

Our first results in Theorem 1 and Corollary 1 of Section 2 establish PAC-Bayes bounds for sequences with arbitrary temporal dependence. Our resultsgeneralize [15], [Theorem 2.4] to the non-i.i.d. data setting in a straightforward manner. Note that Theorem 1 also recovers ([16], [Theorem 3.3]), which is established under different 'existence of test' conditions. Our objective in this paper is to explicate how the ergodic and mixing properties of the Markov data generating process influences the PAC-Bayes bound. The sufficient conditions of our theorem, bounding the mean and variance of the log-likelihood ratio of the data, allows for developing this understanding, without the technicalities of proving the existence of test conditions intruding on the insights.

In Section 3, we study the setting where the data generating model is a stationary α-mixing Markov chain. Stationarity means that the Markov chain is initialized with the invariant distribution corresponding to the parameterized transition kernel, implying all subsequent states also follow this marginal distribution. The α-mixing condition ensures that the variance of the likelihood ratio of the Markov data does not grow faster than linear in the sample size. Our main results in this setting are applicable when the state space of the Markov chain is either continuous or discrete. The primary requirement on the class of data generating Markov models is for the log-likelihood ratio of the parameterized transition kernel and invariant distribution to satisfy a Lipschitz property. This condition implies a decoupling between the model parameters and the random samples, affording a straightforward verification of the mean and variance bounds. We highlight this main result by demonstrating that it is satisfied by a finite state Markov chain, a birth-death Markov chain on the positive integers, and a one-dimensional Gaussian linear model.

In practice, the assumption that the data generating model is stationary is unlikely to be satisfied. Typically, the initial distribution is arbitrary, with the state distribution of the Markov sequence converging weakly to the stationary distribution. In this setting, we must further assume that the class of data generating Markov chains are geometrically ergodic.

We show that this implies the boundedness of the mean and variance of the log-likelihood ratio of the data generating Markov chain. Alternatively, in Theorem 4 we directly impose a drift condition on random variables that bound the log-likelihood ratio. Again, in this more general nonstationary setting, we illustrate the main results by showing that the PAC-Bayes bound is satisfied by a finite state Markov chain, a birth-death Markov chain on the positive integers, and a one-dimensional Gaussian linear model.

In preparation for our main technical results starting in Section 2 we first note relevant notations and definitions in the next section.

1.1. Notations and Definitions

We broadly adopt the notation in [15]. Let the sequence of random variables $X^n = (X_0, \dots, X_n) \subset \mathbb{R}^{m\times(n+1)}$ represent a dataset of $n+1$ observations drawn from a joint distribution $P^{(n)}_{\theta_0}$, where $\theta_0 \in \Theta \subseteq \mathbb{R}^d$ is the 'true' parameter underlying the data generation process. We assume the state space $S \subseteq \mathbb{R}^m$ of the random variables X_i is either discrete-valued or continuous, and write $\{x_0, \dots, x_n\}$ for a realization of the dataset. We also adopt the convention that $0\log(0/0) = 0$.

For each $\theta \in \Theta$, we will write $p^{(n)}_\theta$ as the probability density of $P^{(n)}_\theta$ with respect to some measure $Q^{(n)}$, i.e., $p^{(n)}_\theta := \frac{dP^{(n)}_\theta}{dQ^{(n)}}$, where $Q^{(n)}$ is either Lebesgue measure or the counting measure. Unless stated otherwise, all probabilities, expectations and variances, which we represent as P, $\mathrm{E}[X]$ and $\mathrm{Var}[X]$, are with respect to the true distribution $P^{(n)}_{\theta_0}$.

Let $\pi(\theta)$ be a *prior* distribution with support Θ. The α^{te}*-fractional posterior* is defined as

$$\pi_{n,\alpha^{te}|X^n}(d\theta) := \frac{e^{-\alpha^{te} r_n(\theta,\theta_0)(X^n)}\pi(d\theta)}{\int e^{-\alpha^{te} r_n(\theta,\theta_0)(X^n)}\pi(d\theta)}, \tag{1}$$

where, for $\theta_0, \theta \in \Theta$, $r_n(\theta,\theta_0)(\cdot) := \log\left(\frac{p^{(n)}_{\theta_0}(\cdot)}{p^{(n)}_{\theta}(\cdot)}\right)$, is the log-likelihood ratio of the corresponding density functions, and $\alpha^{te} \in (0,\infty)$ is a tempering coefficient. Setting $\alpha^{te} = 1$ recovers the standard Bayesian posterior. Note that we will use superscripts to distinguish different quantities that are referred to just as α in the literature.

The *Kullback–Leibler* (KL) divergence between distributions P, Q is defined as

$$\mathcal{K}(P,Q) := \int_{\mathcal{X}} \log\left(\frac{p(x)}{q(x)}\right) p(x)dx,$$

where p, q are the densities corresponding to P, Q on some sample space $\mathcal{X}$. In particular, the KL divergence between the distributions parameterized by θ_0 and θ is

$$\begin{aligned}\mathcal{K}(P^{(n)}_{\theta_0}, P^{(n)}_{\theta}) &:= \int \log\left(\frac{p^{(n)}_{\theta_0}(x_0,\dots,x_n)}{p^{(n)}_{\theta}(x_0,\dots,x_n)}\right) p^{(n)}_{\theta_0}(x_0,\dots,x_n)dx_0\cdots dx_n \\ &= \int r_n(\theta,\theta_0)(x_0,\dots,x_n) p^{n}_{\theta_0}(x_0,\dots,x_n)dx_0\cdots dx_n. \end{aligned} \tag{2}$$

The α^{re}*-Rényi divergence* $D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0})$ is defined as

$$D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0}) := \frac{1}{\alpha^{re}-1}\log\int \exp(-\alpha^{re} r_n(\theta,\theta_0)(x_0,\dots,x_n)) p^{(n)}_{\theta_0}(x_0,\dots,x_n)dx_0\cdots dx_n, \tag{3}$$

where $\alpha^{re} \in (0,1)$. As $\alpha^{re} \to 1$, the α^{re}-Rényi divergence recovers the KL divergence.

Let $\mathcal{F}$ be some class of distributions with support in $\mathbb{R}^d$ and such that any distribution P in $\mathcal{F}$ is absolutely continuous with respect to the tempered posterior: $P \ll \pi_{n,\alpha^{te}|X^n}$.

Many choices of $\mathcal{F}$ exist; for instance (see also [15]), $\mathcal{F}$ can be the set of Gaussian measures, denoted $\mathcal{F}_{id}^{\Phi}$:

$$\mathcal{F}_{id}^{\Phi} = \{\Phi(d\theta; \mu, \Sigma) : \mu \in \mathbb{R}^d, \Sigma_{d\times d} \in \text{P.D.}\}, \tag{4}$$

where P.D. references the class of positive definite matrices. Alternately, $\mathcal{F}$ can be the family of *mean-field* or factored distributions where the components θ_i of θ are independent of each other. Let $\tilde{\pi}_{n,\alpha^{te}|X^n}$ be the variational approximation to the tempered posterior, defined as

$$\tilde{\pi}_{n,\alpha^{te}|X^n} := \operatorname*{arg\,min}_{\rho\in\mathcal{F}} \ \mathcal{K}(\rho, \pi_{n,\alpha^{te}|X^n}) \tag{5}$$

It is easy to see that finding $\tilde{\pi}_{n,\alpha^{te}|X^n}$ in Equation (5) is equivalent to the following optimization problem:

$$\tilde{\pi}_{n,\alpha^{te}|X^n} := \operatorname*{arg\,max}_{\rho\in\mathcal{F}} \left[\int r_n(\theta, \theta_0)(x_0, \ldots, x_n)\rho(d\theta) - \left(\alpha^{te}\right)^{-1}\mathcal{K}(\rho, \pi) \right]. \tag{6}$$

Setting $\alpha^{te} = 1$ again recovers the usual variational solution that seeks to approximate the posterior distribution with the closest element of $\mathcal{F}$ (the right-hand side above is called the evidence lower bound (ELBO)). Other settings of α^{te} constitute α^{te}-variational inference [16], which seeks to regularize the 'overconfident' approximate posteriors that standard variational methods tend to produce.

Our results in this paper focus on parametrized Markov chains. We term a Markov chain as 'parameterized' if the transition kernel $p_\theta(\cdot|\cdot)$ is parametrized by some $\theta \in \Theta \subseteq \mathbb{R}^d$. Let $q^{(0)}(\cdot)$ be the initial density (defined with respect to the Lebesgue measure over $\mathbb{R}^m$) or initial probability mass function. Then, the joint density is $p_\theta^{(n)}(x_0, \ldots, x_n) = q^{(0)}(x_0)\prod_{i=0}^{n-1} p_\theta(x_{i+1}|x_i)$; recall, this joint density $p_\theta^{(n)}(x_0, \ldots, x_n)$ corresponds to the walk probability of a time-homogeneous Markov chain. We assume that corresponding to each transition kernel p_θ, $\theta \in \Theta$, there exists an invariant distribution $q_\theta^{(\infty)} \equiv q_\theta$ that satisfies

$$q_\theta(x) = \int p_\theta(x|y)q_\theta(dy) \quad \forall x \in \mathbb{R}^m, \theta \in \Theta.$$

We will also use q_θ to designate the density of the invariant measure (as before, this is with respect to the Lebesgue or counting measure for continuous or discrete state spaces, respectively). A Markov chain is stationary if its initial distribution is the invariant probability distribution, that is, $X_0 \sim q_\theta$.

Our results in the ensuing sections will be established under strong mixing conditions [18] on the Markov chain. Specifically, recall the definition of the α-mixing coefficients of a Markov chain $\{X_n\}$:

Definition 1 (α-mixing coefficient)**.** *Let $\mathcal{M}_i^j$ denote the σ-field generated by the Markov chain $\{X_k : i \le k \le j\}$ parameterized by $\theta \in \Theta$. Then, the α-mixing coefficient is defined as*

$$\alpha_k = \sup_{t>0} \sup_{(A,B)\in\mathcal{M}_{-\infty}^t \times \mathcal{M}_{t+k}^{\infty}} |P_\theta(A\cap B) - P_\theta(A)P_\theta(B)|. \tag{7}$$

Informally speaking, the α-mixing coefficients $\{\alpha_k\}$ measure the dependence between any two events A (in the 'history' σ-algebra) and B (in the 'future' σ-algebra) with a time lag k. We note that we do not use superscripts to identify these α parameters, since they are the only ones with subscripts, and can be identified through this.

2. A Concentration Bound for the α^{re}-Rényi Divergence

The object of analysis in what follows is the probability measure $\tilde{\pi}_{n,\alpha^{te}|X^n}(\theta)$, the variational approximation to the tempered posterior. Our main result establishes a bound on the Bayes risk of this distribution; in particular, given a sequence of loss functions $\ell_n(\theta,\theta_0)$, we bound $\int \ell_n(\theta,\theta_0)\tilde{\pi}_{n,\alpha^{te}|X^n}(\theta)d\theta$. Following recent work in both the i.i.d. and dependent sequence settings [14–16], we will use $\ell_n(\theta,\theta_0) = D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})$, the α^{re}-Rényi divergence between $P_\theta^{(n)}$ and $P_{\theta_0}^{(n)}$ as our loss function. Unlike loss functions like Euclidean distance, Rényi divergence compares θ and θ_0 through their effect on observed sequences, so that issues like parameter identifiability no longer arise. Our first result generalizes [15], [Theorem 2.1] to a general non-i.i.d. data setting.

Proposition 1. *Let $\mathcal{F}$ be a subset of all probability distributions on Θ. For any $\alpha^{re} \in (0,1)$, $\epsilon \in (0,1)$ and $n \geq 1$, the following probabilistic uniform upper bound on the expected α^{re}-Rényi divergence holds:*

$$P\left[\sup_{\rho\in\mathcal{F}} \int D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})\rho(d\theta) \leq \frac{\alpha^{re}}{1-\alpha^{re}}\int r_n(\theta,\theta_0)\rho(d\theta) + \frac{\mathcal{K}(\rho,\pi)+\log(\frac{1}{\epsilon})}{1-\alpha^{re}}\right] \geq 1-\epsilon. \tag{8}$$

The proof of Proposition 1 follows easily from [15], and we include it in Appendix B.1.1 for completeness. Mirroring the comments in [15], when $\rho = \tilde{\pi}_{n,\alpha^{te}}$ this result is precisely [14, Theorem 3.4]. We also note from [14] that $\forall\ \alpha^{re}, \beta \in (0,1]$ α^{re}-Rényi divergences are all equivalent through the following inequality $\frac{\alpha^{re}(1-\beta)}{\beta(1-\alpha^{re})}D_\beta \leq D_{\alpha^{re}} \leq D_\beta\ \forall\ \alpha^{re} \leq \beta$. Hence, for the subsequent results, we simplify by assuming that $\alpha^{te} = \alpha^{re}$. This probabilistic bound implies the following PAC-Bayesian concentration bound on the model risk computed with respect to the fractional variational posterior:

Theorem 1. *Let $\mathcal{F}$ be a subset of all probability distributions parameterized by Θ, and assume there exist $\epsilon_n > 0$ and $\rho_n \in \mathcal{F}$ such that*

i. $\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) = \int \mathrm{E}[r_n(\theta,\theta_0)]\rho_n(d\theta) \leq n\epsilon_n$,
ii. $\int \mathrm{Var}(r_n(\theta,\theta_0))\rho_n(d\theta) \leq n\epsilon_n$, *and*
iii. $\mathcal{K}(\rho_n,\pi) \leq n\epsilon_n$.

Then, for any $\alpha^{re} \in (0,1)$ and $(\epsilon,\eta) \in (0,1)\times(0,1)$,

$$P\left[\int D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}_{n,\alpha^{re}}(d\theta|X^{(n)}) \leq \frac{(\alpha^{re}+1)n\epsilon_n + \alpha^{re}\sqrt{\frac{n\epsilon_n}{\eta}} - \log(\epsilon)}{1-\alpha^{re}}\right] \geq 1-\epsilon-\eta. \tag{9}$$

The proof of Theorem 1 is a generalization of [15] (Theorem 2.4) to the non-i.i.d. setting, and a special case of [16] (Theorem 3.1), where the problem setting includes latent variables. We include a proof for completeness. As noted in [15], the sufficient conditions follow closely from [13] and we will show that they hold for a variety of Markov chain models.

A direct corollary of Theorem 1 follows by setting $\eta = \frac{1}{n\epsilon_n}$, $\epsilon = e^{-n\epsilon_n}$ and using the fact that $e^{-n\epsilon_n} \geq \frac{1}{n\epsilon_n}$. Note that Equation (9) is vacuous if $\eta + \epsilon > 1$. Therefore, without loss of generality, we restrict ourselves to the condition $\frac{2}{n\epsilon_n} < 1$.

Corollary 1. *Assume $\exists\ \epsilon_n > 0$, $\rho_n \in \mathcal{F}$ such that the following conditions hold:*

i. $\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) = \int E[r_n(\theta,\theta_0)]\rho_n(d\theta) \leq n\epsilon_n$,
ii. $\int \mathrm{Var}(r_n(\theta,\theta_0))\rho_n(d\theta) \leq n\epsilon_n$, *and*
iii. $\mathcal{K}(\rho_n,\pi) \leq n\epsilon_n$.

Then, for any $\alpha^{re} \in (0,1)$,

$$P\left[\int D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}_{n,\alpha^{re}}(d\theta|X^{(n)}) \le \frac{2(\alpha^{re}+1)\epsilon_n}{1-\alpha^{re}}\right] \ge 1 - \frac{2}{n\epsilon_n}. \tag{10}$$

We observe that Theorem 1 and Corollary 1 place no assumptions on the nature of the statistical dependence between data points. However, verification of the sufficient conditions is quite hard, in general. One of our key contributions is to verify that under reasonable assumptions on the smoothness of the transition kernel, the sufficient conditions of Theorem 1 and Corollary 1 are satisfied by ergodic Markov chains.

Observe that the first two conditions in Corollary 1 ensure that the distribution ρ_n concentrates on parameters $\theta \in \Theta$ around the true parameter θ_0, while the third condition requires that ρ_n not diverge from the prior π rapidly as a function of the sample size n. In general, verifying the first and third conditions is relatively straightforward. The second condition, on the other hand, is significantly more complicated in the current setting of dependent data, as the variance of $r_n(\theta, \theta_0)$ includes correlations between the observations $\{X_0, \ldots, X_n\}$. In the next section, we will make assumptions on the transition kernels (and corresponding invariant densities) that 'decouple' the temporal correlations and the model parameters in the setting of strongly mixing and ergodic Markov chain models, and allow for the verification of the conditions in Corollary 1. Towards this, Propositions 2 and 3 below characterize the expectation and variance of the log-likelihood ratio $r_n(\cdot,\cdot)$ in terms of the one-step transition kernels of the Markov chain. First, consider the expectation of $r_n(\cdot,\cdot)$ in condition (i).

Proposition 2. *Fix $\theta_1, \theta_2 \in \Theta$ and consider the parameterized Markov transition kernels p_{θ_1} and p_{θ_2}, and initial distributions $q_{\theta_1}^{(0)}$ and $q_{\theta_2}^{(0)}$. Let $p_{\theta_1}^{(n)}$ and $p_{\theta_2}^{(n)}$ be the corresponding joint probability densities; that is,*

$$p_{\theta_j}^{(n)}(x_0, \ldots, x_n) = q_{\theta_j}^{(0)}(x_0) \prod_{i=1}^{n} p_{\theta_i}(x_i|x_{i-1}) \tag{11}$$

for $j \in \{1,2\}$. Then, for any $n \ge 1$, the log-likelihood ratio $r_n(\theta_2, \theta_1)$ satisfies

$$\mathrm{E}_{\theta_1}[r_n(\theta_2,\theta_1)] = \sum_{i=1}^{n} \mathrm{E}_{\theta_1}\left[\log\left(\frac{p_{\theta_1}(X_i|X_{i-1})}{p_{\theta_2}(X_i|X_{i-1})}\right)\right] + \mathrm{E}_{\theta_1}[Z_0], \tag{12}$$

where $Z_0 := \log\left(\frac{q_{\theta_1}^{(0)}(X_0)}{q_{\theta_2}^{(0)}(X_0)}\right)$. The expectation in the first term is with respect to the joint density function $p_{\theta_1}(y,x) = p_{\theta_1}(y|x)q_{\theta_1}^{(i-1)}(x)$ where the marginal density satisfies

$$q_{\theta_1}^{(i-1)}(x) = \begin{cases} \int p_{\theta_1}^{(i-1)}(x_0, \ldots, x_{i-2}, x)dx_0 \cdots dx_{i-2} & \text{for } i > 1, \text{and} \\ q_{\theta_1}^{(0)}(x) & \text{for } i = 1. \end{cases}$$

If the Markov chain is also stationary under θ_1, then Equation (12) simplifies to

$$\mathrm{E}_{\theta_1}[r_n(\theta_2,\theta_1)] = n\mathrm{E}_{\theta_1}\left[\log\left(\frac{p_{\theta_1}(X_1|X_0)}{p_{\theta_2}(X_1|X_0)}\right)\right] + \mathrm{E}_{\theta_1}[Z_0]. \tag{13}$$

Notice that $\mathrm{E}_{\theta_1}[r_n(\theta_2,\theta_1)]$ is precisely the KL divergence, $\mathcal{K}(P_{\theta_1}^{(n)}, P_{\theta_2}^{(n)})$. Next, the following proposition uses [19] (Lemma 1.3) to upper bound the variance of the log-likelihood ratio.

Proposition 3. *Fix $\theta_1, \theta_2 \in \Theta$ and consider parameterized Markov transition kernels p_{θ_1} and p_{θ_2}, with initial distributions $q_{\theta_1}^{(0)}$ and $q_{\theta_2}^{(0)}$. Let $p_{\theta_1}^{(n)}$ and $p_{\theta_2}^{(n)}$ be the corresponding joint* *probability densities of the sequence $(x_0, \ldots, x_n)$, and $q^{(i)}_{\theta_j}$ the marginal density for $i \in \{1, \ldots, n\}$ and $j \in \{1, 2\}$. Fix $\delta > 0$ and, for each $i \in \{1, \ldots, n\}$, define*

$$C^{(i)}_{\theta_1,\theta_2} := \int \left| \log\left(\frac{p_{\theta_1}(x_i|x_{i-1})}{p_{\theta_2}(x_i|x_{i-1})} \right) \right|^{2+\delta} p_{\theta_1}(x_i|x_{i-1}) q^{(i-1)}_{\theta_1}(x_{i-1}) dx_i dx_{i-1}.$$

Similarly, define $Z_0 := \log\left(\frac{q^{(0)}_{\theta_1}(X_0)}{q^{(0)}_{\theta_2}(X_0)} \right)$, and $D_{1,2} := \mathrm{E}_{\theta_1}|Z_0|^{2+\delta}$. Suppose the Markov chain corresponding to θ_1 is α-mixing with coefficients $\{\alpha_k\}$. Then,

$$\begin{aligned} \mathrm{Var}(r_n(\theta_1, \theta_2)) < \sum_{i,j=1}^{n} & \left(\frac{4}{n} + 2n^{\delta/2} (C^{(i)}_{\theta_1,\theta_2} + C^{(j)}_{\theta_1,\theta_2} + \sqrt{C^{(i)}_{\theta_1,\theta_2} C^{(j)}_{\theta_1,\theta_2}}) \right) \left(\alpha^{\delta/(2+\delta)}_{|i-j|-1} \right) \\ & + \sum_{i=1}^{n} \left(\frac{4}{n} + 2n^{\delta/2} (C^{(i)}_{\theta_1,\theta_2} + D_{1,2} + \sqrt{C^{(i)}_{\theta_1,\theta_2} D_{1,2}}) \right) \left(\alpha^{\delta/(2+\delta)}_{i-1} \right) \quad (14) \\ & + \mathrm{Cov}(Z_0, Z_0). \quad (15) \end{aligned}$$

Note that this result holds for any parameterized Markov chain. In particular, when the Markov chain is stationary, $C^{(i)}_{\theta_1,\theta_2} = C^{(1)}_{\theta_1,\theta_2} \ \forall \ i$ and $\forall \theta \in \Theta$, and Equation (14) simplifies to

$$\begin{aligned} \mathrm{Var}(r_n(\theta_1, \theta_2)) < n & \left(\frac{4}{n} + 6n^{\delta/2} C^{(1)}_{\theta_1,\theta_2} \right) \left(\sum_{k \geq 0} \alpha_k^{\delta/(2+\delta)} \right) \\ & + \left(\frac{4}{n} + 2n^{\delta/2} (C^{(1)}_{\theta_1,\theta_2} + D_{1,2} + \sqrt{C^{(1)}_{\theta_1,\theta_2} D_{1,2}}) \right) \left(\sum_{k \geq 1} \alpha_k^{\delta/(2+\delta)} \right) \\ & + \mathrm{Cov}(Z_0, Z_0). \quad (16) \end{aligned}$$

If the sum $\sum_{k \geq 0} \alpha_k^{\delta/(2+\delta)}$ is infinite, the bound is trivially true. For it to be finite, of course, the coefficients α_k must decay to zero sufficiently quickly. For instance, Theorem A.1.2 shows that if the Markov chain is geometrically ergodic, then the α-mixing coefficients are geometrically decreasing. We will use this fact when the Markov chain is non-stationary, as in Section 4. In the next section, however, we first consider the simpler stationary Markov chain setting where geometric ergodic conditions are not explicitly imposed. We also note that unless only a finite number of α_k are nonzero, the sum $\sum_{k \geq 0} \alpha_k^{\delta/(2+\delta)}$ is infinite when $\delta = 0$, and our results will typically require $\delta > 0$.

3. Stationary Markov Data-Generating Models

Observe that the PAC-Bayesian concentration bound in Corollary 1 specifically requires bounding the mean and variance of the log-likelihood ratio $r_n(\theta, \theta_0)$. We ensure this by imposing regularity conditions on the log-ratio of the one-step transition kernels and the corresponding invariant densities. Specifically, we assume the following conditions that decouple the model parameters from the random samples, allowing us to verify the bounds in Corollary 1.

Assumption 1. *There exist positive functions $M^{(1)}_k(\cdot, \cdot)$ and $M^{(2)}_k(\cdot)$, $k \in \{1, 2, \ldots, m\}$ such that for any parameters $\theta_1, \theta_2 \in \Theta$, the log of the ratio of one-step transition kernels and the log of the ratio of the invariant distributions satisfy, respectively,*

$$|\log p_{\theta_1}(x_1|x_0) - \log p_{\theta_2}(x_1|x_0)| \leq \sum_{k=1}^{m} M^{(1)}_k(x_1, x_0) |f^{(1)}_k(\theta_2, \theta_1)| \ \forall \ (x_0, x_1), \text{ and} \quad (17)$$

$$|\log q_{\theta_1}(x) - \log q_{\theta_2}(x)| \leq \sum_{k=1}^{m} M_k^{(2)}(x)|f_k^{(2)}(\theta_2, \theta_1)| \;\; \forall \;\; x. \tag{18}$$

We further assume that for some $\delta > 0$, the functions $f_k^{(1)}, f_k^{(2)}$ and $M_k^{(1)}$ satisfy the following:

i. *there exist constants $C_k^{(t)}$ and measures $\rho_n \in \mathcal{F}$ such that $\int |f_k^{(t)}(\theta, \theta_0)|^{2+\delta}\rho_n(d\theta) < \frac{C_k^{(t)}}{n}$ for $t \in \{1, 2\}$, $n \geq 1$ and $k \in \{1, 2, \ldots, m\}$, and*
ii. *there exists a constant B such that $\int M_k^{(1)}(x_1, x_0)^{2+\delta} p_{\theta_j}(x_1|x_0) q_{\theta_j}^{(0)}(x_0) dx_1 dx_0 < B$, $k \in \{1, \ldots, m\}$ and $j \in \{1, 2\}$.*

The following examples illustrate Equations (17) and (18) for discrete and continuous state Markov chains.

Example 1. *Suppose $\{X_0, \ldots, X_n\}$ is generated by the birth-death chain with parameterized transition probability mass function,*

$$p_\theta(j|i) = \begin{cases} \theta & \text{if } j = i - 1, \\ 1 - \theta & \text{if } j = i + 1. \end{cases}$$

In this example, the parameter θ denotes the probability of birth. We shall see that, $m = 3$: $M_1^{(1)}(X_1, X_0) = I_{[X_1 = X_0 + 1]}$, $M_2^{(1)}(X_1, X_0) = I_{[X_1 = X_0 - 1]}$, and $M_3^{(1)}(X_1, X_0) = 1$. We also define $M_1^{(2)}(X_0) = 1$, and set $M_2^{(2)}(X_0)$ and $M_3^{(2)}(X_0)$ both to $X_0 - 1$. Let $f_1^{(1)}(\theta, \theta_0) = \log\left[\frac{\theta_0}{\theta}\right]$, $f_2^{(1)}(\theta, \theta_0) = \log\left[\frac{1-\theta_0}{1-\theta}\right]$, $f_3^{(1)}(\theta, \theta_0) = 0$, $f_1^{(2)}(\theta, \theta_0) = -f_3^{(2)}(\theta, \theta_0) = \log\left[\frac{1-\theta_0}{1-\theta}\right]$, and $f_2^{(2)}(\theta, \theta_0) = \log\left[\frac{\theta_0}{\theta}\right]$. The derivation of these terms and that they satisfy the conditions of Assumption 1 is provided in the proof of Proposition 6.

Example 2. *Suppose $\{X_0, \ldots, X_n\}$ is generated by the 'simple linear' Gauss–Markov model*

$$X_n = \theta X_{n-1} + W_n,$$

where $\{W_n\}$ is a sequence of i.i.d. standard Gaussian random variables. Then, $m = 2$, with $M_1^{(1)}(X_n, X_{n-1}) = |X_n X_{n-1}|$, $M_2^{(1)}(X_n, X_{n-1}) = X_n^2$, $M_1^{(2)}(x) = \frac{x^2}{2}$ and $M_2^{(2)}(X) = 0$. Corresponding to these, we have $f_1^{(1)}(\theta, \theta_0) = (\theta - \theta_0)$, $f_2^{(1)}(\theta, \theta_0) = (\theta_0^2 - \theta^2)$, $f_1^{(2)}(\theta_0, \theta_0) = (\theta_0^2 - \theta^2)$ and $f_2^{(2)}(\theta_0, \theta_0) = 0$. The derivation of these quantities and that these satisfy the conditions of Assumption 1 under appropriate choice of ρ_n is shown in the proof of Proposition 10.

Note that assuming the same number m of $M_k^{(1)}$ and $M_k^{(2)}$ involves no loss of generality, since these functions can be set to 0. Both Equations (17) and (18) can be viewed as generalized Lipschitz-smoothness conditions, recovering the usual Lipschitz-smoothness when $m = 1$ and when $f_k^{(t)}$ is Euclidean distance. Our generalized conditions are useful for distributions like the Gaussian, where Lipschitz smoothness does not apply. From Jensen's inequality we have $\int |f_k^{(t)}(\theta, \theta_0)|\rho_n(d\theta)| \leq \left[\int |f_k^{(t)}(\theta, \theta_0)|^{2+\delta}\rho_n(d\theta)\right]^{\frac{1}{2+\delta}}$, and Assumption 1(i) above implies that for some constant $C > 0$ and $k \in \{1, 2, \ldots, m\}, t \in \{1, 2\}$,

$$\int |f_k^{(t)}(\theta, \theta_0)|\rho_n(d\theta) \leq \frac{C}{n^{1/(2+\delta)}} < \frac{C}{\sqrt{n}}. \tag{19}$$

Assumption 1(i) is satisfied in a variety of scenarios, for example, under mild assumptions on the partial derivatives of the functions $f_k^{(t)}$. To illustrate this, we present the following proposition.

Proposition 4. *Let $f(\theta, \theta_0)$ be a function on a bounded domain with bounded partial derivatives with $f(\theta_0, \theta_0) = 0$. Let $\{\rho_n(\cdot)\}$ be a sequence of probability densities on θ such that $\mathrm{E}_{\rho_n}[\theta] = \theta_0$ and $\mathrm{Var}_{\rho_n}[\theta] = \frac{\sigma^2}{n}$ for some $\sigma > 0$. Then, for some $C > 0$,*

$$\int |f(\theta, \theta_0)|^{2+\delta} \rho_n(d\theta) < \frac{C}{n}. \tag{20}$$

Proof. Define $\partial_\theta f(\theta, \theta_0) := \frac{\partial f(\theta, \theta_0)}{\partial \theta}$ as the partial derivative of the function f. By the mean value theorem, $|f(\theta, \theta_0)| = |\theta - \theta_0||\partial_\theta f(\theta^*, \theta_0)|$, for some $\theta^* \in [\min\{\theta, \theta_0\}, \max\{\theta, \theta_0\}]$. Since the partial derivatives are bounded, there exists $L \in \mathbb{R}$ such that $\partial_\theta f(\theta^*, \theta_0) < L$, and $\int |f(\theta, \theta_0)|^{2+\delta} \rho_n(d\theta) < L^{2+\delta} \int |\theta - \theta_0|^{2+\delta} \rho_n(d\theta)$. Choose $G > 0$ be such that $|\theta| < G$, then $\left|\frac{\theta - \theta_0}{2G}\right|^{2+\delta} < \left|\frac{\theta - \theta_0}{2G}\right|^2$. Therefore, $\int |\theta - \theta_0|^{2+\delta} \rho_n(d\theta) < (2G)^{2+\delta} \mathrm{Var}\left[\frac{\theta}{2G}\right] < (2G)^\delta \frac{\sigma^2}{n}$. Now choosing $(2G)^\delta \sigma^2$ as C completes the proof. □

If $\partial_\theta f_k^{(t)}$ is continuous and Θ is compact, then $\partial_\theta f_k^{(t)}$ is always bounded. Furthermore, observe that if $\mathrm{E}\left[M_k^{(1)}(X_1, X_0)^{2+\delta}\right] < B$, without loss of generality we can use Jensen's inequality to conclude that, for all $0 < a < 2 + \delta$, $\mathrm{E}\left[M_k^{(1)}(X_1, X_0)^a\right] < B^{\frac{a}{2+\delta}} < B$.

We can now state the main theorem of this section.

Theorem 2. *Let $\{X_0, \ldots, X_n\}$ be generated by a stationary, α-mixing Markov chain parametrized by $\theta_0 \in \Theta$. Suppose that Assumption 1 holds and that the α-mixing coefficients satisfy $\sum_{k \geq 1} \alpha_k^{\delta/(2+\delta)} < +\infty$. Furthermore, assume that $\mathcal{K}(\rho_n, \pi) \leq \sqrt{n} C$ for some constant $C > 0$. Then, the conditions of Corollary 1 are satisfied with $\epsilon_n = \mathrm{O}\left(\max\left(\frac{1}{\sqrt{n}}, \frac{n^{\delta/2}}{n}\right)\right)$.*

Theorem 2 is satisfied by a large class of Markov chains, including chains with countable and continuous state spaces. In particular, if the Markov chain is geometrically ergodic, then it follows from Equation (A4) (in the appendix) that $\sum_{k \geq 1} \alpha_k^{\delta/(2+\delta)} < +\infty$. Observe that in order to achieve $O(\frac{1}{\sqrt{n}})$ convergence, we need $\delta \leq 1$. Key to the proof of Theorem 2 is the fact that the variance of the log-likelihood ratio can be controlled via the application of Assumption 1 and Proposition 3. Note also that as δ decreases, satisfying the condition $\sum_{k \geq 1} \alpha_k^{\delta/(2+\delta)}$ requires the Markov chain to be faster mixing.

We now illustrate Theorem 2 for a number of Markov chain models. First, consider a birth-death Markov chain on a finite state space.

Proposition 5. *Suppose the data-generating process is a birth-death Markov chain, with one-step transition kernel parametrized by the birth probability $\theta_0 \in \Theta$. Let $\mathcal{F}$ be the set of all Beta distributions. We choose the prior to be a Beta distribution. Then, the conditions of Theorem 2 are satisfied and $\epsilon_n = \mathrm{O}\left(\frac{1}{\sqrt{n}}\right)$.*

Proof. The proof of Proposition 5 follows from the more general Proposition 8, by fixing the initial distribution to the invariant distribution under θ_0. Therefore it has been omitted. We simply refer to the proof of Proposition 8 under a more general setup in Appendix B.3. □

The birth-death chain on the finite state space is, of course, geometrically ergodic and the α-mixing coefficients α_k decay geometrically. Note that the invariant distribution of this Markov chain is uniform over the state space, and consequently this is a particularly simple example. A more complicated and more realistic example is a birth-death Markov chain on the nonnegative integers. We note that if the probability of birth θ in a birth-death Markov chain on positive integers is greater than 0.5, then the Markov chain is transient, and consequently, not ergodic. Hence, our prior should be chosen to have support within $(0, 0.5)$. For that purpose, we define the class of scaled beta distributions.

Definition 2 (Scaled Beta)**.** *If X is a beta distribution on with parameters a and b, then Y is said to be a scaled beta distribution with same parameters on the interval $(c, m+c)$ if*

$$Y = mx + c \;\; ; \;\; (m, c) \in \mathbb{R}^2$$

and in that case, the pdf of Y is obtained as

$$f(y) = \begin{cases} \frac{1}{m\text{Beta}(a,b)} \left(\frac{y-c}{m}\right)^{a-1} \left(1 - \frac{y-c}{m}\right)^{b-1} & \text{if } y \in (c, m+c), \\ 0 & \text{otherwise}. \end{cases}.$$

Here, $\mathrm{E}[Y] = m\frac{a}{a+b} + c$ and $\mathrm{Var}[Y] = m^2 \frac{ab}{(a+b)^2(a+b+1)}$. For the birth-death chain, we set $m = 0.5$ and $c = 0$ giving it support on $(0, \frac{1}{2})$. Setting $m = 2$ and $c = -1$ gives a beta distribution rescaled to have support on $(-1, 1)$.

Proposition 6. *Suppose the data-generating process is a positive recurrent birth-death Markov chain on the positive integers parameterized by the birth probability $\theta_0 \in (0, \frac{1}{2})$. Further let $\mathcal{F}$ be the set of all Beta distributions rescaled to have support $(0, \frac{1}{2})$. We choose the prior to be a scaled Beta distribution on $(0, 1/2)$ with parameters a and b. Then, the conditions of Theorem 2 are satisfied with $\epsilon_n = \mathrm{O}\left(\frac{1}{\sqrt{n}}\right)$.*

Proof. The proof of Proposition 6 (for the stationary case) follows from the more general Proposition 9 (the nonstationary case) by fixing the initial distribution to the invariant distribution under θ_0. We omit the proof and simply refer to the proof of Proposition 9 under a more general setup in Appendix B.3. □

Unlike with the finite state-space, the invariant distribution now depends on the parameter $\theta \in \Theta$, and verification of the conditions of the proposition is more involved. In Appendix A.2, we prove that the class of scaled beta distributions satisfy the condition $\mathcal{K}(\rho_n, \pi) \leq n\epsilon_n$ when the prior π is a beta or an uniform distribution. This fact will allow us to prove the above propositions.

Both Proposition 5 and Proposition 6 assume a discrete state space. The next example considers a strictly stationary simple linear model (as defined in Example 2), which has a continuous, unbounded state space.

Proposition 7. *Suppose the data-generating model is a stationary simple linear model:*

$$X_n = \theta_0 X_{n-1} + W_n, \tag{21}$$

where $\{W_n\}$ are i.i.d. standard Gaussian random variables and $|\theta_0| < 1$. Suppose that $\mathcal{F}$ is the class of all beta distributions rescaled to have the support $(-1, 1)$. Then, the conditions of Theorem 2 are satisfied with $\epsilon_n = \mathrm{O}\left(\frac{1}{\sqrt{n}}\right)$.

Proof. This is a special case of the more general non-stationary simple linear model which is detailed in Proposition 10. Therefore, the proof of the fact that the simple linear model satisfies Assumption 1 when starting from stationarity is deferred to the proof of Proposition 10. The simple linear model with $|\theta_0| < 1$ has geometrically decreasing (and therefore summable) α-mixing coefficients as a consequence of [20] (eq. (15.49)) and Theorem A.1.2. Combining these two facts, it follows that the conditions of Theorem 2 are satisfied. □

Observe that Theorem 1 (and Corollary 1) are general, and hold for *any* dependent data-generating process. Therefore, there can be Markov chains that satisfy these, but do not satisfy Assumption 1 which entails some loss of generality. However, as our examples demonstrate, common Markov chain models do indeed satisfy the latter assumption.

4. Non-Stationary, Ergodic Markov Data-Generating Models

We call a time-homogeneous Markov chain *non-stationary* if the initial distribution $q^{(0)}$ is not the invariant distribution. There are two sets of results in this setting: in Theorem 3 and Theorem 4 we explicitly impose the α-mixing condition, while in Theorem 5 we impose a f-geometric ergodicity condition (Definition A.1.2 in the appendix). As seen in Equation (A4) (in the appendix) if the Markov chain is also geometrically ergodic, then $\forall\ \delta > 0$, $\sum \alpha_k^{\delta/(2+\delta)} < \infty$. This condition can be relaxed, albeit at the risk of more complicated calculations that, nonetheless, mirror those in the geometrically ergodic setting. A common thread through these results is that we must impose some integrability or regularity conditions on the functions $M_k^{(1)}$.

First, in Theorem 3 we assume that the $M_k^{(1)}$ functions in Assumption 1 are uniformly bounded and that the α-mixing condition is satisfied. This result holds for both discrete and continuous state space settings.

Theorem 3. *Let $\{X_0, \dots, X_n\}$ be generated by an α-mixing Markov chain parametrized by $\theta_0 \in \Theta$ with transition probabilities satisfying Assumption 1 and with known initial distribution $q^{(0)}$. Let $\{\alpha_k\}$ be the α-mixing coefficients under θ_0, and assume that $\sum_{k\geq 1} \alpha_k^{\delta/(2+\delta)} < +\infty$. Suppose that there exists $B \in \mathbb{R}$ such that $\sup_{x,y} |M_k^{(1)}(x,y)| < B$ for all $k \in \{1,2,\dots,m\}$ in Assumption 1. Furthermore, assume that there exists $\rho_n \in \mathcal{F}$ such that $\mathcal{K}(\rho_n, \pi) \leq \sqrt{n}C$ for some constant $C > 0$. If the initial distribution $q^{(0)}$ satisfies $E_{q^{(0)}}|M_k^{(2)}(X_0)|^2 < +\infty$ for all $k \in \{1,2,\dots,m\}$, then the conditions of Corollary 1 are satisfied with $\epsilon_n = O\Big(\max(\frac{1}{\sqrt{n}}, \frac{n^{\delta/2}}{n})\Big)$.*

The following result in Proposition 8 illustrates Theorem 3 in the setting of a finite state birth-death Markov chain.

Proposition 8. *Suppose the data-generating process is a finite state birth-death Markov chain, with one-step transition kernel parametrized by the birth probability θ_0. Let $\mathcal{F}$ be the set of all Beta distributions. We choose the prior on θ_0 to be a Beta distribution. Then, the conditions of Theorem 3 are satisfied with $\epsilon_n = O\Big(\frac{1}{\sqrt{n}}\Big)$ for any initial distribution $q^{(0)}$.*

Theorem 3 also applies to data generated by Markov chains with countably infinite state spaces, so long as the class of data-generating Markov chains is strongly ergodic and the initial distribution has finite second moments. The following example demonstrates this in the setting of a birth-death Markov chain on the positive integers, where the initial distribution is assumed to have finite second moments.

Proposition 9. *Suppose the data-generating process is a birth-death Markov chain on the non-negative integers, parameterized by the probability of birth $\theta_0 \in (0, \frac{1}{2})$. Further let $\mathcal{F}$ be the set of all Beta distributions rescaled upon the support $(0, \frac{1}{2})$. Let $q^{(0)}$ be a probability mass function on non-negative integers such that $\sum_{i=1}^{\infty} i^2 q^{(0)}(i) < +\infty$. We choose the prior to be a scaled Beta distribution on $(0, 1/2)$ with parameters a and b. Then, the conditions of Theorem 3 are satisfied with $\epsilon_n = O\Big(\frac{1}{\sqrt{n}}\Big)$.*

Since continuous functions on a compact domain are bounded, we have the following (easy) corollary (stated without proof).

Corollary 2. *Let $\{X_0, \dots, X_n\}$ be generated by an α-mixing Markov chain parametrized by $\theta_0 \in \Theta$ on a compact state space, and with initial distribution $q^{(0)}$. Suppose the α-mixing coefficients satisfy $\sum_{k\geq 1} \alpha_k^{\delta/(2+\delta)} < +\infty$, and that Assumption 1 holds with continuous functions $M_k^{(1)}(\cdot,\cdot)$, $k \in \{1,2,\dots,m\}$. Furthermore, assume that there exists ρ_n such that $\mathcal{K}(\rho_n, \pi) \leq \sqrt{n}C$ for some constant C. Then, Theorem 3 is satisfied with $\epsilon_n = O\Big(\max(\frac{1}{\sqrt{n}}, \frac{n^{\delta/2}}{n})\Big)$.*

In general, the $M_k^{(1)}$ functions will not be uniformly bounded (consider the case of the Gauss–Markov simple linear model in Example 2), and stronger conditions must be imposed on the data-generating Markov chain itself. The following assumption imposes a 'drift' condition from [21]. Specifically, [21] (Theorem 2.3) shows that under the conditions of Assumption 2, the moment generating function of an aperiodic Markov chain $\{X_n\}$ can be upper bounded by a function of the moment generating function of X_0. Together with the α-mixing condition, Assumption 2 implies that this Markov data generating process satisfies Corollary 1.

Assumption 2. *Consider a Markov chain $\{X_n\}$ parameterized by $\theta_0 \in \Theta$. Let $\mathcal{M}_{-\infty}^n$ denote the σ-field generated by $\{X_{-\infty}, \dots, X_{n-1}, X_n\}$. Denote the stochastic process $\{M_n^k\} := \{M_k^{(1)}(X_n, X_{n-1})\}$; recall $M_k^{(1)}$, for each $k = 1, \dots, m_1$, are defined in Assumption 1. For each $k = 1, \dots, m$, assume the process $\{M_n^k\}$ satisfies the following conditions:*

- *The drift condition holds for $\{M_n^k\}$, i.e., $\mathrm{E}\left[M_n^k - M_{n-1}^k | \mathcal{M}_{-\infty}^{n-1}, M_{n-1}^k > a\right] \leq -\epsilon$ for some $\epsilon, a > 0$.*
- *For some $\lambda > 0$ and $\mathcal{D} > 0$, $\mathrm{E}\left[e^{\lambda(M_n^k - M_{n-1}^k)} | \mathcal{M}_{-\infty}^{n-1}\right] \leq \mathcal{D}$.*

Under this drift condition, the next theorem shows that Corollary 1 is satisfied.

Theorem 4. *Let $\{X_0, \dots, X_n\}$ be generated by an aperiodic α-mixing Markov chain parametrized by $\theta_0 \in \Theta$ and initial distribution $q^{(0)}$. Suppose that Assumption 1 and Assumption 2 hold, and that the α-mixing coefficients satisfy $\sum_{k \geq 1} \alpha_k^{\delta/(2+\delta)} < +\infty$. Furthermore, assume $\mathcal{K}(\rho_n, \pi) \leq \sqrt{n}C$ for some constant $C > 0$. If $\int e^{\lambda M_k^{(1)}(y,x)} p_{\theta_0}(y|x) q_1^{(0)}(x) dx < +\infty$ for all $k = 1, \dots, m_1$, then the conditions of Corollary 1 are satisfied with $\epsilon_n = \mathrm{O}\left(\max(\frac{1}{\sqrt{n}}, \frac{n^{\delta/2}}{n})\right)$.*

Verifying the conditions in Theorem 4 can be quite challenging. Instead, we suggest a different approach that requires f-geometric ergodicity. Unlike the drift condition in Assumption 2, f-geometric ergodicity additionally requires the existence of a petite set. As noted before, geometric ergodicity implies α-mixing with geometrically decaying mixing coefficients. As with Theorem 4, we assume for simplicity that the Markov chain is aperiodic.

Theorem 5. *Let $\{X_0, \dots, X_n\}$ be generated by an aperiodic Markov chain parametrized by $\theta_0 \in \Theta$ with known initial distribution $q^{(0)}$, and assumed to be V-geometrically ergodic for some $V : \mathbb{R}^m \to [1, \infty)$. Suppose that Assumption 1 holds and $\int M_k^{(1)}(y,x)^{2+\delta} p_{\theta_0}(y|x) dy < V(x)\ \forall\ k, x$ and some $\delta > 0$. Furthermore, assume that $\mathcal{K}(\rho_n, \pi) \leq \sqrt{n}C$ for some constant $C > 0$. If the initial distribution $q^{(0)}$ satisfies $E_{q^{(0)}}[V(X_0)] < +\infty$, then the conditions of Corollary 1 are satisfied with $\epsilon_n = \mathrm{O}\left(\max(\frac{1}{\sqrt{n}}, \frac{n^{\delta/2}}{n})\right)$.*

The following Proposition 10 shows, the simple linear model satisfies Theorem 5 when the parameter θ_0 is suitably restricted.

Proposition 10. *Consider the simple linear model satisfying the equation*

$$X_n = \theta_0 X_{n-1} + W_n, \tag{22}$$

where $\{W_n\}$ are i.i.d. standard Gaussian random variables and $|\theta_0| < 2^{\frac{1}{4+2\delta} - 1}$ for $\delta > 0$. Let $\mathcal{F}$ be the space of all scaled Beta distributions on $(-1, 1)$ and suppose the prior π is a uniform distribution on $(-1, 1)$. Then, the conditions of Theorem 5 are satisfied with $\epsilon_n = \mathrm{O}\left(\max(\frac{1}{\sqrt{n}}, \frac{n^{\delta/2}}{n})\right)$, if the initial distribution $q^{(0)}$ satisfies $E_{q^{(0)}}[X_0^{4+2\delta}] < +\infty$.

5. Misspecified Models

We show next how our results can be extended to the misspecified model setting. Assume that the true data generating distribution is parametrized by $\theta_0 \notin \Theta$. Let $\theta_n^* := \arg\min_{\theta\in\Theta} \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})$ represent the closest parametrized distribution in the variational family to the data-generating distribution. Further, assume our usual conditions:

i. $\int \mathrm{E}[r_n(\theta,\theta_n^*)]\rho_n(d\theta) \le n\epsilon_n$,
ii. $\int \mathrm{Var}(r_n(\theta,\theta_n^*))\rho_n(d\theta) \le n\epsilon_n$.

Now, since $r_n(\theta,\theta_0) = r_n(\theta,\theta_n^*) + r_n(\theta_n^*,\theta_0)$, we have

$$\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \le \mathrm{E}[r_n(\theta_0,\theta_n^*)] + n\epsilon_n. \tag{23}$$

Similarly, decomposing the variance it follows that

$$\mathrm{Var}[r_n(\theta,\theta_0)] = \mathrm{Var}[r_n(\theta,\theta_n^*)] + \mathrm{Var}[r_n(\theta_n^*,\theta_0)] + 2\mathrm{Cov}[r_n(\theta,\theta_n^*), r_n(\theta_n^*,\theta_0)]. \tag{24}$$

Using the fact that $2ab \le a^2+b^2$ on the covariance term $2\mathrm{Cov}[r_n(\theta,\theta_n^*), r_n(\theta_n^*,\theta_0)] = 2\mathrm{E}[(r_n(\theta,\theta_n^*) - \mathrm{E}[r_n(\theta,\theta_n^*)])(r_n(\theta_n^*,\theta_0) - \mathrm{E}[r_n(\theta_n^*,\theta_0)])]$, we have

$$\mathrm{Var}[r_n(\theta,\theta_0)] \le 2\mathrm{Var}[r_n(\theta,\theta_n^*)] + 2\mathrm{Var}[r_n(\theta_n^*,\theta_0)]. \tag{25}$$

Integrating both sides with respect to $\rho_n(d\theta)$ we get

$$\begin{aligned}\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) &\le 2\int \mathrm{Var}[r_n(\theta,\theta_n^*)]\rho_n(d\theta) + 2\int \mathrm{Var}[r_n(\theta_n^*,\theta_0)]\rho_n(d\theta) \\ &\le 2n\epsilon_n + 2\mathrm{Var}[r_n(\theta_n^*,\theta_0)]. \end{aligned}\tag{26}$$

Consequently, we arrive at the following result:

Theorem 6. *Let $\mathcal{F}$ be a subset of all probability distributions parameterized by Θ. Let $\theta_n^* = \arg\min_{\theta\in\Theta} \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})$ and assume there exist $\epsilon_n > 0$ and $\rho_n \in \mathcal{F}$ such that*

i. $\int \mathrm{E}[r_n(\theta,\theta_n^*)]\rho_n(d\theta) \le n\epsilon_n$,
ii. $\int \mathrm{Var}(r_n(\theta,\theta_n^*))\rho_n(d\theta) \le n\epsilon_n$, *and*
iii. $\mathcal{K}(\rho_n,\pi) \le n\epsilon_n$.

Then, for any $\alpha^{re} \in (0,1)$ and $(\epsilon,\eta) \in (0,1)\times(0,1)$,

$$P\left[\int D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}_{n,\alpha^{re}}(d\theta|X^{(n)}) \le \right.$$
$$\left. \frac{(\alpha^{re}+1)n\epsilon_n + \mathrm{E}[r_n(\theta_0,\theta_n^*)] + \alpha^{re}\sqrt{\frac{2n\epsilon_n + 2\mathrm{Var}[r_n(\theta_n^*,\theta_0)]}{\eta}} - \log(\epsilon)}{1-\alpha^{re}}\right] \ge 1-\epsilon-\eta. \tag{27}$$

The proof of this theorem is straightforward and follows from the proof of Theorem 1 by plugging in the upper bounds for KL-divergence from Equation (23), and variance from Equation (26) to Equation (A13). A sketch of the proof is presented in the appendix.

6. Conclusions

Concentration of the KL-VB model risk, in terms of the expected α^{re}-Rényi divergence, is well established under the i.i.d. data generating model assumption. Here, we extended this to the setting of Markov data generating models, linking the concentration rate to the mixing and ergodic properties of the Markov model. Our results apply to both stationary and non-stationary Markov chains, as well as to the situation with misspecified models. There remain a number of open questions. An immediate one is to extend the current analysis to continuous-time Markov chains and Markov jump processes, possibly using uniformization of the continuous time model. Another direction is to extend this to the setting of non-homogeneous Markov chains, where analogues of notions such as stationarity are less straightforward. Further, as noted in the introduction, [14] establish PAC-Bayes bounds under slightly weaker 'existence of test functions' conditions, while our results are established under the stronger conditions used by [15] for the i.i.d. setting. Weakening the conditions in our analysis is important, but complicated. A possible path is to build on results from [22], who provides conditions form the existence of exponentially powerful test functions exist for distinguishing between two Markov chains. It is also known that there exists a likelihood ratio test separating any two ergodic measures [23]. However, leveraging these to establish the PAC-Bayes bounds for the KL-VB posterior is a challenging effort that we leave to future papers. Finally it is of interest to generalize our PAC-bounds to posterior approximations beyond KL-variational inference, such as α^{re}-Rényi posterior approximations [6], and loss-calibrated posterior approximations [24,25].

Author Contributions: Formal analysis, I.B.; Investigation, I.B.; Methodology, I.B., V.A.R. and H.H.; Resources, V.A.R. and H.H.; Validation, V.A.R. and H.H. All authors have read and agreed to the published version of the manuscript.

Funding: National Science Foundation : IIS-1816499; DMS-1812197.

Acknowledgments: Rao and Honnappa acknowledge support from NSF DMS-1812197. In addition, Rao acknowledges NSF IIS-1816499 for supporting this project.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Technical Desiderata

Appendix A.1. Definitions Related to Markov Chains

As noted before, ergodicity plays an acute role in establishing our results. We consolidate various definitions used throughout the paper in this appendix. Recall that we assume the parameterized Markov chain possesses an invariant probability density or mass function q_θ under parameter $\theta \in \Theta$. Our results in Section 4 also rely on the ergodic properties of the Markov chain, and we assume that the Markov chain is f-geometrically ergodic [20] (Chapter 15). First, refer to the definition of the functional norm $\|\cdot\|_f$, from Definition A.1.1,

Definition A.1.1 (f-norm). *The functional norm in f-metric of a measure v, or the f-norm of v is*

$$\|v\|_f = \sup_{g:|g|<f} \left| \int g dv \right|, \tag{A1}$$

where f and g are any two functions.

An immediate consequence of this definition is that if f_1, f_2 are two functions such that $f_1 < f_2$ (i.e., for all points in the support of the functions), then

$$\|v\|_{f_1} \leq \|v\|_{f_2}. \tag{A2}$$

Now that we have defined the $\|\cdot\|_f$ norm, we can now define f-geometric ergodicity. In the following, we assume the Markov chain is positive Harris; see [20] for a definition. This is a mild and fairly standard assumption in Markov chain theory.

Definition A.1.2 (f-geometric ergodicity)**.** *For any function f, Markov chain $\{X_n\}$ parameterized by $\theta \in \Theta$ is said to be f-geometrically ergodic if it is positive Harris and there exists a constant $r_f > 1$, that depends on f, such that for any $A \in \mathcal{B}(X)$,*

$$\sum_{n=1}^{n} r_f^n \left\| P_\theta(X_n \in A | X_0 = x) - \int_A q_\theta(y) dy \right\|_f < \infty. \tag{A3}$$

It is straightforward to see that this is equivalent to

$$\left\| P_\theta(X_n \in A | X_0 = x) - \int q_\theta(y) dy \right\|_f \leq C r_f^{-n}$$

for an appropriate constant C (which may depend on the state x), that is, the Markov chain approaches steady state at a geometrically fast rate. If a Markov chain is f-geometrically ergodic for $f \equiv 1$, then, it is simply termed as *geometrically ergodic*. It is straightforward to see (via Theorem A.1.2 in the Appendix) that a geometrically ergodic Markov chain is also α-mixing, with mixing coefficients satisfying

$$\sum_{k \geq 0} \alpha_k^v < \infty \ \forall \ v > 0, \tag{A4}$$

showing that, under geometric ergodicity, the α-mixing coefficients raised to any positive power v are finitely summable. We note here that the most standard procedure to establish f-geometric ergodicity for any Markov chain is through the verification of the drift condition. The drift condition is a sufficient condition for a Markov chain to be f-geometrically ergodic, as long as there exists a set (called petite set) towards which the Markov chain drifts to (see Assumption A.1.1 in the appendix). If a Markov chain is f-geometrically ergodic with $f \equiv V$, for some particular function V, then we call it V-geometrically ergodic.

We defined V-geometric ergodicity in the previous sections. In this section, we provide a sufficient condition for a Markov chain to be V-geometrically ergodic. First, we recall the definition of resolvent from [20] (Chapter 5).

Definition A.1.3 (Resolvent)**.** *Let $n \in \{0, 1, 2, \ldots\}$ and q_n be such that $q_n \geq 0 \ \forall \ n$ and $\sum_{n=1}^{\infty} q_n = 1$. Note that q_n can be thought of being a probability mass function for a random variable "q" taking values on non-negative integers. Then, the resolvent of a Markov chain with respect to q is given by $K_q(x, A)$ where,*

$$K_q(x, A) = \sum_{n=0}^{\infty} q_n P(X_n \in A | X_0 = x). \tag{A5}$$

Then, the definition of petite sets follows (see, for Reference, [20] (Chapter 5)).

Definition A.1.4 (Petite Sets)**.** *Let $X_0, \ldots, X_n$ be n samples from a Markov chain taking values on the state space $\mathcal{X}$. Let C be a set. We shall call C to be v_q petite if*

$$K_q(x, B) \geq v_q(B)$$

for all $x \in C$ and $B \in \mathcal{B}(\mathcal{X})$, and a non-trivial measure v_q on $\mathcal{B}(\mathcal{X})$, and a probability mass function q on $\{1, 2, 3, \ldots\}$

Now, let $\Delta V(x) := E[V(X_n)|X_{n-1} = x] - V(x)$ for $V : S \to [1, \infty)$.

Assumption A.1.1 (Drift condition)**.** *[20] (Chapter 5) Suppose the chain $\{X_n\}$ is, aperiodic and ψ-irreducible . Let there exists a petite set C, constants $b < \infty, \beta > 0$, and a non-trivial function $V : S \to [1, \infty)$ satisfying*

$$\Delta V(x) \leq -\beta V(x) + bI_{x \in C} \;\; \forall x \in S. \tag{A6}$$

If a Markov chain drifts towards a petite set then it is V-geometrically ergodic. Suppose, for simplicity, that $V(x) = |X|$. Then, the drift condition becomes $E[|X_n||X_{x-1}] - |X_{n-1}| = -\beta|X_n| + bI_{X_n \in C}$. The left hand side of this equation represents the change in the state of the Markov chain in one time epoch. Thus, the condition in Assumption A.1.1 essentially states that the Markov chain drifts towards a petite set C and then, once it reaches that set, moves to any point in the state space with at least some probability independent of C.

Theorem A.1.1 (Geometrically ergodic theorem)**.** *Suppose that $\{X_n\}$ is satisfies Assumption A.1.1. Then, the set $S_V = \{x : V(x) < \infty\}$ is absorbing, i.e., $P_\theta(X_1 \in S_V | X_0 = x) = 1 \; \forall x \in S_V$, and full, i.e., $\psi(S_V^c) = 0$. Furthermore, $\exists$ constants $r > 1, R < \infty$ such that, for any $A \in \mathcal{B}(S)$,*

$$\left\| P_\theta(X_n \in A | X_0 = x) - \int_A q_\theta(y)dy \right\|_V \leq Rr^{-n}V(x). \tag{A7}$$

Any aperiodic and ψ-irreducible Markov chain satisfying the drift condition is geometrically ergodic. A consequence of Equation (A2) is that if, $\{X_n\}$ is V-geometrically ergodic, then for any other function U, such that $|U| < V$, it is also U-geometrically ergodic. In essence, a geometrically ergodic Markov chain is asymptotically uncorrelated in a precise sense. Recall ρ-mixing coefficients defined as follows. Let $\mathcal{A}$ be a sigma field and $\mathcal{L}^2(A)$ be the set of square integrable, real valued, $\mathcal{A}$ measurable functions.

Definition A.1.5 (ρ-mixing coefficient)**.** *Let $\mathcal{M}_i^j$ denote the sigma field generated by the measures X_k, where $i \leq k \leq j$. Then,*

$$\rho_k = \sup_{t>0} \sup_{(f,g) \in \mathcal{L}^2(\mathcal{M}_{-\infty}^t) \times \mathcal{L}^2(\mathcal{M}_{t+k}^\infty)} |\mathrm{Corr}(f,g)|, \tag{A8}$$

where Corr *is the correlation function.*

Theorem A.1.2. *If X_n is geometrically ergodic, then it is α-mixing. That is, there exists a constant $c > 0$ such that $\alpha_k = \mathrm{O}(e^{-ck})$.*

Proof. By [26] (Theorem 2) it follows that a geometrically ergodic Markov chain is asymptotically uncorrelated with ρ-mixing coefficients (see Definition A.1.5) that satisfy $\rho_k = \mathrm{O}(e^{-ck})$. Furthermore, it is well known that [18,26] $\alpha_k \leq \frac{1}{4}\rho_k$, implying $\alpha_k = \mathrm{O}(e^{-ck})$. □

Appendix A.2. Bounding the KL-Divergence between Beta Distributions

The following results will be utilized in the proofs of Propositions 8–10.

Lemma A.2.1. *Let $\theta_0 \in (0,1)$. Let, ρ_n be a sequence of Beta distributions with parameters $a_n = n\theta_0$ and $b_n = n(1-\theta_0)$. Let π denote an uniform distribution, $U(0,1)$. Then, $\mathcal{K}(\rho_n, \pi) < C + \frac{1}{2}\log(n)$, for some constant $C > 0$.*

Proof. Without loss of generality, we can assume $a_n > 1$ and $b_n > 1$. The same form of the result can be obtained in all the other cases, by appropriate use of the bounds presented in the proof. We write the KL divergence $\mathcal{K}(\rho_n, \pi)$ as $\int \log(\frac{\rho_n}{\pi})\rho_n(d\theta)$. Since π is uniform,

$\pi(\theta) = 1$ whenever $\theta \in (0,1)$. Hence, the KL-divergence can be written as the negative of the entropy of ρ_n $\int_0^1 \log(\rho_n(\theta))\rho_n(d\theta)$, which can be written as

$$\mathcal{K}(\rho_n, \pi) = (a_n - 1)\psi(a_n) + (b_n - 1)\psi(b_n) - (a_n + b_n - 2)\psi(a_n + b_n) \\ - \log \mathrm{Beta}(a_n, b_n), \tag{A9}$$

where ψ is the digamma function. Using Stirling's approximation on $\mathrm{Beta}(a_n, b_n)$ yields,

$$\mathrm{Beta}(a_n, b_n) = \sqrt{2\pi}\frac{a_n^{a_n-1/2} b_n^{b_n-1/2}}{(a_n+b_n)^{a_n+b_n-1/2}}(1+o(1)).$$

Hence, setting $C_1 = \log(2\sqrt{\pi})$, we can write $-\log \mathrm{Beta}(a_n, b_n)$ as,

$$-\log \mathrm{Beta}(a_n, b_n) = C_1 - (a_n - \frac{1}{2})\log(a_n) - (b_n - \frac{1}{2})\log(b_n) \\ + (a_n + b_n - \frac{1}{2})\log(a_n + b_n) + \log(1 + o(1)).$$

From [27] we have that $\log(x) - \frac{1}{x} < \psi(x) < \log(x) - \frac{1}{2x}\ \forall\ x > 0$. Since we assumed $a_n > 1$ and $b_n > 1$, the fact that $\psi(x) < \log(x) - \frac{1}{2x}$ implies

$$(a_n - 1)\psi(a_n) < (a_n - 1)\log(a_n) - \frac{a_n - 1}{2a_n} \text{ and,}$$
$$(b_n - 1)\psi(b_n) < (b_n - 1)\log(b_n) - \frac{b_n - 1}{2b_n}.$$

Finally, using the fact that $\log(x) - \frac{1}{x} < \psi(x)$, we get,

$$-(a_n + b_n - 2)\psi(a_n + b_n) < -(a_n + b_n - 2)\log(a_n + b_n) + \frac{a_n + b_n - 2}{a_n + b_n}.$$

Therefore, after much cancellation, the KL-divergence

$$(a_n - 1)\psi(a_n) + (b_n - 1)\psi(b_n) - (a_n + b_n - 2)\psi(a_n + b_n) - \log \mathrm{Beta}(a_n, b_n)$$

can be upper bounded by

$$-\frac{1}{2}\log(a_n) - \frac{1}{2}\log(b_n) + \frac{3}{2}\log(a_n + b_n) + \frac{a_n + b_n - 2}{a_n + b_n} - \frac{a_n - 1}{2a_n} - \frac{b_n - 1}{2b_n}.$$

Now, plugging in the values of a_n and b_n, we get Plugging in the values of a_n and b_n, we get as upper bound for the KL-divergence as,

$$\begin{aligned}\mathcal{K}(\rho_n, \pi) &< -\frac{1}{2}\log(n\theta_0) - \frac{1}{2}\log(n(1-\theta_0)) + \frac{3}{2}\log(n) + \frac{n-2}{n} - \frac{n\theta_0 - 1}{2n\theta_0} - \frac{n(1-\theta_0) - 1}{2n(1-\theta_0)} \\ &= \frac{1}{2}\log(n) - \frac{1}{2}(\log(\theta_0) + \log(1-\theta_0)) + 3 - \frac{2}{n} - \frac{1}{2n\theta_0} - \frac{1}{2n(1-\theta_0)} \\ &< C + \frac{1}{2}\log(n),\end{aligned}$$

for some large enough positive constant C. This completes our proof. □

Proposition A.2.1. *Let $\theta_0 \in (0,1)$. Let, ρ_n be a sequence of Beta distributions with parameters $a_n = n\theta_0$ and $b_n = n(1-\theta_0)$. Let π denote an Beta distribution, with parameters (a, b). Then, $\mathcal{K}(\rho_n, \pi) < C + \frac{1}{2}\log(n)$, for some constant $C > 0$.*

Proof. Without loss of generality, we assume $a > 1$ and $b > 1$. As mentioned in the proof of Lemma A.2.1, the other cases follows similarly. We write the KL-divergence between ρ_n and π as,

$$\mathcal{K}(\rho_n, \pi) = \int \log\left(\frac{\rho_n}{\pi}\right)\rho_n(d\theta) = \int \log\left(\frac{\rho_n}{U}\right)\rho_n(d\theta) + \int \log\left(\frac{U}{\pi}\right)\rho_n(d\theta),$$

where, U is an uniform distribution on $(0,1)$. We analyze the second term in the above expression. The second term can be written as,

$$\begin{aligned}\int \log\left(\frac{U}{\pi}\right)\rho_n(d\theta) &= \int \log\left(\frac{1}{\frac{1}{\text{Beta}(a,b)}\theta^{a-1}(1-\theta)^{b-1}}\right)\rho_n(d\theta)\\ &= C_1 - (a-1)\int \log(\theta)\rho_n(d\theta) - (b-1)\int \log(1-\theta)\rho_n(d\theta),\end{aligned}$$

where C_1 is $\log(\text{Beta}(a,b))$. Since, ρ_n follows a Beta distribution with parameters $a_n = n\theta_0$ and $b_n = n(1-\theta_0)$, we get that,

$$\int \log\left(\frac{U}{\pi}\right)\rho_n(d\theta) = C_1 - (a-1)[\psi(a_n) - \psi(a_n+b_n)] - (b-1)[\psi(b_n) - \psi(a_n+b_n)]$$

Since, $\log(x) - \frac{1}{x} < \psi(x) < \log(x) - \frac{1}{2x}$, looking at the term $[\psi(a_n) - \psi(a_n+b_n)]$, we get that,

$$\begin{aligned}-[\psi(a_n) - \psi(a_n+b_n)] &= -[\psi(n\theta_0) - \psi(n\theta_0 + n(1-\theta_0))]\\ &= -[\psi(n\theta_0) - \psi(n)].\end{aligned}$$

Using the lower bound on $\psi(n\theta_0)$ and the upper bound on $\psi(n)$, we get

$$\begin{aligned}-[\psi(a_n) - \psi(a_n+b_n)] &< -\log(n\theta_0) + \frac{1}{n\theta_0} + \log(n) - \frac{1}{2n}\\ &= -\log(\theta_0) + \frac{2-\theta_0}{2n\theta_0}.\end{aligned}$$

Furthermore, similarly, we get that,

$$-[\psi(b_n) - \psi(a_n+b_n)] < -\log(1-\theta_0) + \frac{2-(1-\theta_0)}{2n(1-\theta_0)}.$$

Therefore it follows that

$$\begin{aligned}&\max\{-(a-1)[\psi(a_n) - \psi(a_n+b_n)], -(b-1)[\psi(b_n) - \psi(a_n+b_n)]\}\\ &\quad< \max\left\{(a-1)\left[-\log(\theta_0) + \frac{2-\theta_0}{2n\theta_0}\right], (b-1)\left[-\log(1-\theta_0) + \frac{2-(1-\theta_0)}{2n(1-\theta_0)}\right]\right\}\\ &\quad< C,\end{aligned}$$

for a large positive constant C. Using the above bounds, we finally show that,

$$\begin{aligned}C_1 - (a-1)[\psi(a_n) - \psi(a_n+b_n)] - (b-1)[\psi(b_n) - \psi(a_n+b_n)]&\\ < C_1 + 2C,&\end{aligned}$$

which can be upper bounded by C' for some large constant C'. Finally, we upper bound $\int \log(\frac{\rho_n}{U})\rho_n(d\theta)$ by Lemma A.2.1 thereby completing the proof. □

Appendix B. Proofs of Main Results

Appendix B.1. Proofs for A Concentration Bound for the α^{re}-Rényi Divergence

Appendix B.1.1. Proof of Proposition 1

We start by recalling the variational formula of Donsker and Varadhan [28].

Lemma B.1.1 (Donsker-Varadhan). *For any probability distribution function π on Θ, and for any measurable function $h : \Theta \to \mathbb{R}$, if $\int e^h d\pi < \infty$, then*

$$\log \int e^h d\pi = \sup_{\rho \in \mathcal{M}^+(\Theta)} \left\{ \int h d\rho - \mathcal{K}(\rho, \pi) \right\} \tag{A10}$$

Now, fix $\alpha^{re} \in (0,1)$, and $\theta \in \Theta$. First, observe that by the definition of the α^{re}-Rényi divergence we have

$$\mathrm{E}^{(n)}_{\theta_0}[\exp(-\alpha^{re} r_n(\theta, \theta_0))] = \exp[-(1-\alpha^{re}) D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0})]$$

Multiplying both sides of the equation by $\exp[(1-\alpha^{re}) D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0})$ and integrating with respect to (w.r.t.) $\pi(\theta)$ it follows that

$$\int \mathrm{E}^{(n)}_{\theta_0}\left[\exp\left(-\alpha^{re} r_n(\theta,\theta_0) + (1-\alpha^{re}) D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0})\right)\right]\pi(d\theta) = 1, \text{or}$$
$$\mathrm{E}^{(n)}_{\theta_0}\left[\int \exp\left(-\alpha^{re} r_n(\theta,\theta_0) + (1-\alpha^{re}) D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0})\right)\pi(d\theta)\right] = 1.$$

Define $h(\theta) := -\alpha^{re} r_n(\theta,\theta_0) + (1-\alpha^{re}) D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0})$. Then, applying Lemma B.1.1 to the integrand on the left hand side (l.h.s.) above, it follows that

$$\mathrm{E}^{(n)}_{\theta_0}\left[\exp\left(\sup_{\rho\in\mathcal{M}^+(\Theta)}\left[\int h(\theta)\rho(d\theta) - \mathcal{K}(\rho,\pi)\right]\right)\right] = 1.$$

Multiply both sides of this equation by $\epsilon > 0$ to obtain

$$\mathrm{E}^{(n)}_{\theta_0}\left[\exp\left(\sup_{\rho\in\mathcal{M}^+(\Theta)}\left[\int h(\theta)\rho(d\theta) - \mathcal{K}(\rho,\pi) + \log(\epsilon)\right]\right)\right] = \epsilon.$$

Now, by Markov's inequality, we have

$$P^{(n)}_{\theta_0}\left[\sup_{\rho\in\mathcal{M}^+(\Theta)} \int (-\alpha^{re} r_n(\theta,\theta_0) + (1-\alpha^{re}) D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0}))\rho(d\theta) - \mathcal{K}(\rho,\pi) + \log(\epsilon) \ge 0\right] \le \epsilon. \tag{A11}$$

Thus, it follows via complementation that

$$P^{(n)}_{\theta_0}\left[\forall \rho \in \mathcal{F}(\Theta) \int D_{\alpha^{re}}(P^{(n)}_\theta, P^{(n)}_{\theta_0})\rho(d\theta) \le \frac{\alpha^{re}}{(1-\alpha^{re})} \int r_n(\theta,\theta_0)\rho(d\theta) + \frac{\mathcal{K}(\rho,\pi) - \log(\epsilon)}{1-\alpha^{re}}\right]$$
$$\ge 1-\epsilon,$$

thereby completing the proof. □

Appendix B.1.2. Proof of Theorem 1

Recall the definition of the fractional posterior and the VB approximation,

$$\pi_{n,\alpha^{re}|X^n} = \frac{\exp^{-\alpha^{re} r_n(\theta,\theta_0)(X^n)}\,\pi(d\theta)}{\int \exp^{-\alpha^{re} r_n(\gamma,\theta_0)(X^n)}\,\pi(d\gamma)}, \quad \tilde{\pi}_{n,\alpha^{re}|X^n} = \arg\min_{\rho\in\mathcal{F}} \mathcal{K}(\rho, \pi_{n,\alpha^{re}|X^{(n)}}).$$

It follows by definition of the KL divergence that

$$\tilde{\pi}_{n,\alpha^{re}|X^n} = \arg\min_{\rho \in \mathcal{F}} \left\{ -\alpha^{re} \int r_n(\theta,\theta_0)\rho(d\theta) + \mathcal{K}(\rho,\pi) \right\}, \tag{A12}$$

where π is the prior distribution. Following Proposition 1 it follows that for any $\epsilon > 0$

$$\int D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}(d\theta|X^n) \le \frac{\alpha^{re}}{(1-\alpha^{re})} \int r_n(\theta,\theta_0)\rho(d\theta) + \frac{\mathcal{K}(\rho,\pi) - \log(\epsilon)}{1-\alpha^{re}},$$

with probability $1-\epsilon$. We fix an $\eta \in (0,1)$. Using Chebychev's inequality, we have

$$\begin{aligned}
&P_{\theta_0}^{(n)}\Bigg[\frac{\alpha^{re}}{1-\alpha^{re}}\int r_n(\theta,\theta_0)\rho_n(d\theta) \ge \frac{\alpha^{re}}{1-\alpha^{re}}\int \mathrm{E}[r_n(\theta,\theta_0)]\rho_n(d\theta) \\
&\qquad + \frac{\alpha^{re}}{1-\alpha^{re}}\sqrt{\frac{\mathrm{Var}[\int r_n(\theta,\theta_0)\rho_n(d\theta)]}{\eta}} + \frac{\mathcal{K}(\rho_n,\pi)}{1-\alpha^{re}}\Bigg] \\
&= P_{\theta_0}^{(n)}\Bigg[\frac{\alpha^{re}}{1-\alpha^{re}}\int r_n(\theta,\theta_0)\rho_n(d\theta) - \frac{\alpha^{re}}{1-\alpha^{re}}\int \mathrm{E}[r_n(\theta,\theta_0)]\rho_n(d\theta) - \frac{\mathcal{K}(\rho_n,\pi)}{1-\alpha^{re}} \\
&\qquad \ge \frac{\alpha^{re}}{1-\alpha^{re}}\sqrt{\frac{\mathrm{Var}[\int r_n(\theta,\theta_0)\rho_n(d\theta)]}{\eta}}\Bigg] \\
&\le \frac{\mathrm{Var}\left[\frac{\alpha^{re}}{1-\alpha^{re}}\int r_n(\theta,\theta_0)\rho_n(d\theta) - \frac{\alpha^{re}}{1-\alpha^{re}}\int \mathrm{E}[r_n(\theta,\theta_0)]\rho_n(d\theta) - \frac{\mathcal{K}(\rho_n,\pi)}{1-\alpha^{re}}\right]}{\frac{(\alpha^{re})^2}{(1-\alpha^{re})^2}\frac{\mathrm{Var}[\int r_n(\theta,\theta_0)\rho_n(d\theta)]}{\eta}}.
\end{aligned}$$

Note that $\frac{\alpha^{re}}{1-\alpha^{re}}\int E(r_n(\theta,\theta_0))\rho_n(d\theta)$ and $\frac{\mathcal{K}(\rho_n,\pi)}{1-\alpha^{re}}$ are constants with respect to the data, implying

$$\begin{aligned}
&\mathrm{Var}\left[\frac{\alpha^{re}}{1-\alpha^{re}}\int r_n(\theta,\theta_0)\rho_n(d\theta) - \frac{\alpha^{re}}{1-\alpha^{re}}\int \mathrm{E}[r_n(\theta,\theta_0)]\rho_n(d\theta) - \frac{\mathcal{K}(\rho_n,\pi)}{1-\alpha^{re}}\right] \\
&\qquad = \frac{(\alpha^{re})^2}{(1-\alpha^{re})^2}\mathrm{Var}\left[\int r_n(\theta,\theta_0)\rho_n(d\theta)\right].
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
&P_{\theta_0}^{(n)}\Bigg[\frac{\alpha^{re}}{1-\alpha^{re}}\int r_n(\theta,\theta_0)\rho_n(d\theta) \ge \frac{\alpha^{re}}{1-\alpha^{re}}\int \mathrm{E}[r_n(\theta,\theta_0)]\rho_n(d\theta) \\
&\qquad + \frac{\alpha^{re}}{1-\alpha^{re}}\sqrt{\frac{\mathrm{Var}[\int r_n(\theta,\theta_0)\rho_n(d\theta)]}{\eta}} + \frac{\mathcal{K}(\rho_n,\pi)}{1-\alpha}\Bigg] \le \eta.
\end{aligned}$$

From Proposition 1, with probability $1-\epsilon$ the following holds

$$\int D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}_{n,\alpha^{re}|X^n}(d\theta) \le \frac{\alpha^{re}\int r_n(\theta,\theta_0)\rho_n(d\theta) + \mathcal{K}(\rho_n,\pi) - \log(\epsilon)}{1-\alpha^{re}}.$$

Therefore, with probability $1-\eta-\epsilon$ the following statement holds

$$\int D_{\alpha^{re}}(P_{\theta}^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}_{n,\alpha^{re}|X^n}(d\theta) \leq \frac{\alpha^{re}}{1-\alpha^{re}}\int \mathcal{K}(P_{\theta_0}^{(n)}, P_{\theta}^{(n)})\rho_n(d\theta) + \frac{\alpha^{re}}{1-\alpha^{re}}\sqrt{\frac{\mathrm{Var}[\int r_n(\theta,\theta_0)\rho_n(d\theta)]}{\eta}} + \frac{\mathcal{K}(\rho_n,\pi)-\log(\epsilon)}{1-\alpha^{re}}. \tag{A13}$$

Next, we observe that

$$\mathrm{Var}\left[\int r_n(\theta,\theta_0)\rho_n(d\theta)\right] = E_{\theta_0}^{(n)}\left[\left|\int r_n(\theta,\theta_0)\rho_n(d\theta) - E\left[\int r_n(\theta,\theta_0)\rho_n(d\theta)\right]\right|^2\right] \leq \int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta),$$

by a straightforward application of Jensen's inequality to the inner integral on the left hand side. Finally, following the hypotheses (i), (ii) and (iii), we have,

$$\begin{aligned}\int D_{\alpha^{re}}(P_{\theta}^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}_{n,\alpha^{re}|X^n}(d\theta) &\leq \frac{\alpha^{re}}{1-\alpha^{re}}\int\left(\mathcal{K}(P_{\theta_0}^{(n)}, P_{\theta}^{(n)}) + \sqrt{\frac{\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta)}{\eta}}\right)\rho_n(d\theta) \\ &\quad + \frac{1}{\alpha^{re}}(\mathcal{K}(\rho_n,\pi)-\log(\epsilon)) \\ &\leq \frac{\alpha^{re}(\epsilon_n + \sqrt{\frac{n\epsilon_n}{\eta}})}{1-\alpha^{re}} + \frac{n\epsilon_n - \log(\epsilon)}{1-\alpha^{re}},\end{aligned}$$

thereby concluding the proof. □

Appendix B.1.3. Proof of Proposition 2

We define $Y_i := \log\left(\frac{p_{\theta_1}(X_i|X_{i-1})}{p_{\theta_2}(X_i|X_{i-1})}\right)$ for $i=1,\ldots,n$, and $Z_0 = \log\left(\frac{q_1^{(0)}(X_0)}{q_2^{(0)}(X_0)}\right)$. Then, using the Markov property we can see that the Kullback–Leibler divergence between the joint distributions $P_{\theta_1}^{(n)}$ and $P_{\theta_2}^{(n)}$ satisfies $\mathcal{K}\left(P_{\theta_1}^{(n)}, P_{\theta_2}^{(n)}\right) = \sum_{i=1}^{n} \mathrm{E}_{\theta_1}[Y_i] + \mathrm{E}_{\theta_1}[Z_0]$. If the Markov chain $\{X_i\}$ is stationary under θ_1, so is $\{Y_i\}$. Hence $Y_i \overset{d}{=} Y_1$ and the above equation reduces to,

$$\mathcal{K}\left(P_{\theta_1}^{(n)}, P_{\theta_2}^{(n)}\right) = n\mathrm{E}_{\theta_1}[Y_1] + \mathrm{E}_{\theta_1}[Z_0]. \tag{A14}$$

□

Appendix B.1.4. Proof of Proposition 3

First, recall the following result from [19].

Lemma B.1.2. *[19] (Lemma 1.2) Let $X_{-\infty},\ldots,X_1,X_2,\ldots$ be an α-mixing Markov chain with α-mixing coefficients given by α_k. Let $\mathcal{M}_a^b$ be the sigma-field generated by the subsequence $(X_a,X_{a+1},\ldots,X_b)$. Let $\eta_t \in \mathcal{M}_{-\infty}^t$ and $\tau_t \in \mathcal{M}_{t+k}^{\infty}$ be adapted random variables such that $|\eta_t| \leq 1, |\tau_t| \leq 1$. Then,*

$$\sup_t \sup_{\eta_t,\tau_t} |\mathrm{E}[\eta_t\tau_t] - \mathrm{E}[\eta_t]\mathrm{E}[\tau_t]| \leq 4\alpha_k. \tag{A15}$$

This lemma provides an upper bound on the covariance of events η and τ, as shown next.

Lemma B.1.3. *Let $\eta \in \mathcal{M}_{-\infty}^{t}$ $\tau \in \mathcal{M}_{t+k}^{\infty}$ be such that, $E|\eta|^{2+\delta} \leq C_1, E|\tau|^{2+\delta} \leq C_2$ for some $\delta > 0$. Then, for a fixed $n < +\infty$, we have*

$$|\mathrm{E}\eta\tau - \mathrm{E}\eta\mathrm{E}\tau| \leq \left(\frac{4}{n} + 2n^{\delta/2}(C_1 + C_2) + 2n^{\delta/2}\sqrt{C_1C_2}\right)\alpha_k^{2\delta/(2+\delta)}. \tag{A16}$$

Proof. Let $N < +\infty$ be a fixed number. We get from the triangle inequality that

$$\begin{aligned} |\mathrm{E}\eta\tau - \mathrm{E}\eta\mathrm{E}\tau| \leq\ & |\mathrm{E}\eta\tau I_{[|\eta|\leq N,|\tau|\leq N]} - \mathrm{E}\eta I_{[|\eta|\leq N]}\mathrm{E}\tau I_{[|\tau|\leq N]}| \\ & + |\mathrm{E}\eta\tau I_{[|\eta|\geq N,|\tau|\leq N]} - \mathrm{E}\eta I_{[|\eta|\geq N]}\mathrm{E}\tau I_{[|\tau|\leq N]}| \\ & + |\mathrm{E}\eta\tau I_{[|\eta|\leq N,|\tau|\geq N]} - \mathrm{E}\eta I_{[|\eta|\leq N]}\mathrm{E}\tau I_{[|\tau|\geq N]}| \\ & + |\mathrm{E}\eta\tau I_{[|\eta|\geq N,|\tau|\geq N]} - \mathrm{E}\eta I_{[|\eta|\geq N]}\mathrm{E}\tau I_{[|\tau|\geq N]}|. \end{aligned} \tag{A17}$$

Multiplying and dividing the first term by N^2 and applying Lemma B.1.2, we get $|\mathrm{E}\eta\tau I_{[|\eta|\leq N,|\tau|\leq N]} - \mathrm{E}\eta I_{[|\eta|\leq N]}\mathrm{E}\tau I_{[|\tau|\leq N]}| \leq 4N^2\alpha_k$. For the second term, if $|\tau| \leq N$, then $\tau \leq N$ and $\tau \geq -N$. Plugging this in the second term we get,

$$|\mathrm{E}\eta\tau I_{[|\eta|\geq N,|\tau|\leq N]} - \mathrm{E}\eta I_{[|\eta|\geq N]}\mathrm{E}\tau I_{[|\tau|\leq N]}| \leq \left|N\mathrm{E}\eta I_{[|\eta|\geq N} + N\left[\mathrm{E}\eta I_{[|\eta|\geq N]}\right]\right| \tag{A18}$$

$$= 2N|\mathrm{E}\eta I_{[|\eta|\geq N]}|. \tag{A19}$$

Since $|\eta| \geq N$, we have $1 \leq \frac{|\eta|^{1+\delta}}{N^{1+\delta}}$. Following this,

$$|2N\mathrm{E}\eta I_{[|\eta|\geq N]}| \leq 2N\left|\mathrm{E}\left[\frac{|\eta|^{2+\delta}}{N^{1+\delta}} I_{[|\eta|\geq N]}\right]\right| \tag{A20}$$

$$\leq 2N\frac{1}{N^{1+\delta}}|\mathrm{E}\eta^{2+\delta}| \leq 2\frac{C_1}{N^{\delta}}. \tag{A21}$$

Similarly, we can also write for the third term, $|\mathrm{E}\eta\tau I_{[|\eta|\leq N,|\tau|\geq N]} - \mathrm{E}\eta I_{[|\eta|\leq N]}\mathrm{E}\tau I_{[|\tau|\geq N]}| \leq 2\frac{C_2}{N^{\delta}}$. Finally, for the last term we get that by Cauchy-Schwarz inequality,

$$|\mathrm{E}\eta\tau I_{[|\eta|\geq N,|\tau|\geq N]} - \mathrm{E}\eta I_{[|\eta|\geq N]}\mathrm{E}\tau I_{[|\tau|\geq N]}| \leq \sqrt{\mathrm{Var}\left[\eta I_{[|\eta|\geq N]}\right]\mathrm{Var}\left[\tau I_{[|\tau|\geq N]}\right]} \tag{A22}$$

$$< 2\sqrt{\mathrm{Var}\left[\eta I_{[|\eta|\geq N]}\right]\mathrm{Var}\left[\tau I_{[|\tau|\geq N]}\right]} \tag{A23}$$

$$\leq 2\sqrt{\mathrm{E}\left[\eta^2 I_{[|\eta|\geq N]}\right]\mathrm{E}\left[\tau^2 I_{[|\tau|\geq N]}\right]}. \tag{A24}$$

Since $|\eta| > N$, $1 < \frac{|\eta|^{\delta}}{N^{\delta}}$. Similarly, $1 < \frac{|\tau|^{\delta}}{N^{\delta}}$. Plugging these in the previous equation, we get,

$$\sqrt{\mathrm{E}\left[\eta^2 I_{[|\eta|\geq N]}\right]\mathrm{E}\left[\tau^2 I_{[|\tau|\geq N]}\right]} \leq \sqrt{\frac{1}{N^{2\delta}}\mathrm{E}\left[|\eta|^{2+\delta} I_{[|\eta|\geq N]}\right]\mathrm{E}\left[|\tau|^{2+\delta} I_{[|\tau|\geq N]}\right]} \tag{A25}$$

$$\leq \frac{1}{N^{\delta}}\sqrt{C_1C_2}. \tag{A26}$$

Combining the four upper bounds above, we get,

$$|\mathrm{E}\eta\tau - \mathrm{E}\eta\mathrm{E}\tau| \leq 4N^2\alpha_k + \frac{2}{N^{\delta}}(C_1 + C_2) + \frac{2}{N^{\delta}}\sqrt{C_1C_2}. \tag{A27}$$

Now, in particular, setting $N = n^{-1/2}\alpha_k^{-1/(2+\delta)}$ it follows that

$$|\mathrm{E}\eta\tau - \mathrm{E}\eta\mathrm{E}\tau| \leq \frac{4}{n}\alpha_k^{\delta/(2+\delta)} + 2n^{\delta/2}\alpha_k^{\delta/(2+\delta)}(C_1 + C_2) + 2n^{\delta/2}\alpha_k^{\delta/(2+\delta)}\sqrt{C_1C_2} \tag{A28}$$

$$= \left(\frac{4}{n} + 2n^{\delta/2}(C_1 + C_2) + 2n^{\delta/2}\sqrt{C_1C_2}\right)\alpha_k^{\delta/(2+\delta)}. \tag{A29}$$

□

Lemma B.1.4. *Let $\{X_t\}$ be an α-mixing Markov chain with mixing coefficient α_k. Further assume that $\mathrm{E}|X_t|^{2+\delta} \leq C_1$ and $\mathrm{E}|X_{t+k}|^{2+\delta} \leq C_2$ for some $\delta > 0$. Then, for any t and any $n > 0$*

$$|\mathrm{Cov}(X_t, X_{t+k})| \leq \left(\frac{4}{n} + 2n^{\delta/2}(C_1 + C_2) + 2n^{\delta/2}\sqrt{C_1C_2}\right)\alpha_k^{\delta/(2+\delta)}. \tag{A30}$$

Proof. Set $\eta = X_t, \tau = X_{t+k}$ in Lemma B.1.3. □

We also need to establish the following technical lemma.

Lemma B.1.5. *Let $\{X_t\}$ be an α-mixing Markov Chain with mixing coefficients $\{\alpha_t\}$. Then the process $\{Y_t\}$ where $Y_t := \log\left(\frac{p_{\theta_0}(X_t|X_{t-1})}{p_\theta(X_t|X_{t-1})}\right)$ is also α-mixing with mixing coefficients $\{\tilde{\alpha}_t\}$ where $\tilde{\alpha}_t = \alpha_{t-1}$.*

Proof. By Z_i denote the paired random measure (X_i, X_{i-1}). Let $\mathcal{M}_i^j$ denote the sigma field generated by the measures X_k, where $i \leq k \leq j$. By $\mathcal{G}_i^j$ denote the sigma field generated by the measures Z_k, where $i \leq k \leq j$. Let $C \in \mathcal{M}_{i-1}^j$. Then, C can be expressed as $(C_{i-1} \times C_i \times \cdots \times C_j)$. for $C_{i-1} \in \mathcal{M}_{i-1}^{i-1}, C_i \in \mathcal{M}_i^i \ldots$ and so on. Now, consider a map. $T_i^j : (C_{i-1} \times C_i \times \cdots \times C_j) \to (C_{i-1} \times C_i \times C_i \times \cdots \times C_{j-1} \times C_{j-1} \times C_j)$. Note that, $T_i^j(C) \in \mathcal{G}_i^j$. It is easy to see that $\mathcal{G}_i^j = T_i^j(\mathcal{M}_{i-1}^j) \cup \mathcal{M}_{i-1}^{*j}$, where $T_i^j(\mathcal{M}_{i-1}^j)$ is obtained by applying the map T_i^j to each element of $\mathcal{M}_{i-1}^j$. If we assume this latter set to be the range and $\mathcal{M}_{i-1}^j$ to be the domain, then, by construction, T_i^j is a bijection. Furthermore, the two classes are made of disjoint sets, i.e., if $A \in T_i^j(\mathcal{M}_{i-1}^j)$ and $A^* \in \mathcal{M}_{i-1}^{*j}$, then $A \cap A^* = \phi$. Furthermore, note that $\mathcal{M}_{i-1}^{j*}$ is made of impossible sets. i.e., $P(A^*) = 0 \ \forall \ A^* \in \mathcal{M}_{i-1}^{j*}$. Now consider the α-mixing coefficients for Z_i. By definition, it is given by

$$\begin{aligned}\alpha_k^z &= \sup_i \sup_{A \in \mathcal{G}_{-\infty}^i, B \in \mathcal{G}_{i+k}^\infty} |P(A \cap B) - P(A)P(B)| \\ &= \sup_i \sup_{A \in \mathcal{G}_{-\infty}^i, B \in \mathcal{G}_{i+k}^\infty} |P((A^o \cup A^*) \cap (B^o \cup B^*)) - P((A^o \cup A^*))P((B^o \cup B^*))|.\end{aligned}$$

where,

$$\begin{array}{ll} A = (A^o \cup A^*) & B = (B^o \cup B^*) \\ A^o \in \mathcal{T}_{-\infty}^i(M_{-\infty}^i) & A^* \in \mathcal{M}_{-\infty}^{*i} \\ B^o \in T_{i+k-1}^\infty(\mathcal{M}_{j+k-1}^\infty) & B^* \in \mathcal{M}_{j+k-1}^{*\infty}. \end{array}$$

Then, the expression for the α-mixing coefficient can be reduced into

$$\alpha_k^z = \sup_i \sup_{A^o \in T_{-\infty}^i(\mathcal{M}_{-\infty}^i), B^o \in T_{i+k-1}^\infty(\mathcal{M}_{i+k-1}^\infty)} |P(A^o \cap B^o) - P(A^o)P(B^o)|.$$

Note that, by bijection property of T_i^j, we can find $A' \in \mathcal{M}_{-\infty}^i$ and $B' \in \mathcal{M}_{i+k-1}^\infty$ such that

$$\begin{aligned}\alpha_k^z &= \sup_i \sup_{A' \in \mathcal{M}_{-\infty}^i, B' \in \mathcal{M}_{i+k-1}^\infty} |P(T_{-\infty}^i(A') \cap T_{i+k-1}^\infty(B')) - P(T_{-\infty}^i(A'))P(T_{i+k-1}^\infty(B'))|. \\ &= \alpha_{k-1}.\end{aligned}$$

Now, $\log\left(\frac{p_{\theta_0}(X_n|X_{n-1})}{p_\theta(X_n|X_{n-1})}\right)$ is just a function of the paired Markov chain Z_i, therefore it has α-mixing coefficient α_{k-1}. □

We now proceed to the proof of Proposition 3. Let $\{X_k\}$ be a stationary α-mixing Markov chain under θ_1 with mixing coefficients $\{\alpha_k\}$. Observe that the log-likelihood can be expressed as

$$\begin{aligned} r_n(\theta_2, \theta_1) &= \sum_{i=1}^n \log\left(\frac{p_{\theta_1}(X_i|X_{i-1})}{p_{\theta_2}(X_i|X_{i-1})}\right) + \log\left(\frac{q_1^{(0)}(X_0)}{q_2^{(0)}(X_0)}\right) \\ &\equiv \sum_{i=1}^n Y_i + Z_0. \end{aligned}$$

Therefore, the variance of the log-likelihood ratio is simply

$$\begin{aligned} \mathrm{Var}_{\theta_1}[r_n(\theta_2, \theta_1)] &= \mathrm{Var}_{\theta_1}\left[\sum_{i=1}^n Y_i + Z_0\right] \\ &= \sum_{i,j=1}^n \mathrm{Cov}_{\theta_1}(Y_i, Y_j) + \sum_{i=1}^n \mathrm{Cov}_{\theta_1}(Y_i, Z_0) + \mathrm{Cov}_{\theta_1}(Z_0, Z_0). \end{aligned}$$

It follows from Lemma B.1.5 that $\{Y_k\}$ is a stochastic process with α-mixing coefficients α_{k-1}. Therefore, using Lemma B.1.4 we have

$$\begin{aligned} |\mathrm{Cov}_{\theta_1}(Y_i, Y_j)| &= |\mathrm{E}_{\theta_1} Y_i Y_j - \mathrm{E}_{\theta_1} Y_i \mathrm{E}_{\theta_1} Y_j| \\ &< \Bigg(\frac{4}{n} + 2n^{\delta/2}(\mathrm{E}_{\theta_1}|Y_i|^{2+\delta} + \mathrm{E}_{\theta_1}|Y_j|^{2+\delta} \\ &\quad + \sqrt{\mathrm{E}_{\theta_1}|Y_i|^{2+\delta}\mathrm{E}_{\theta_1}|Y_j|^{2+\delta}})\Bigg)\alpha_{|j-i|-1}^{\delta/(2+\delta)} \\ &= \left(\frac{4}{n} + 2n^{\delta/2}(C_{\theta_1,\theta_2}^{(i)} + C_{\theta_1,\theta_2}^{(j)} + \sqrt{C_{\theta_1,\theta_2}^{(i)} C_{\theta_1,\theta_2}^{(j)}})\right)\alpha_{|j-i|-1}^{\delta/(2+\delta)}. \end{aligned}$$

Similarly, as above we can also say

$$|\mathrm{Cov}_{\theta_1}(Y_i, Z_0)| < \left(\frac{4}{n} + 2n^{\delta/2}(C_{\theta_1,\theta_2}^{(i)} + D_{1,2} + \sqrt{C_{\theta_1,\theta_2}^{(i)} D_{1,2}})\right)\left(\alpha_{i-1}^{\delta/(2+\delta)}\right)$$

Combining, the two upper bounds above, we get the first result:

$$\begin{aligned} \mathrm{Var}_{\theta_1}\Big[r_n(\theta_2, \theta_1)\Big] &< \sum_{i,j=1}^n \left(\frac{4}{n} + 2n^{\delta/2}(C_{\theta_1,\theta_2}^{(i)} + C_{\theta_1,\theta_2}^{(j)} + \sqrt{C_{\theta_1,\theta_2}^{(i)} C_{\theta_1,\theta_2}^{(j)}})\right)\left(\alpha_{|i-j|-1}^{\delta/(2+\delta)}\right) \\ &\quad + \sum_{i=1}^n \left(\frac{4}{n^2} + 2n^{\delta/2}(C_{\theta_1,\theta_2}^{(i)} + D_{1,2} + \sqrt{C_{\theta_1,\theta_2}^{(i)} D_{1,2}})\right)\left(\alpha_{i-1}^{\delta/(2+\delta)}\right) \\ &\quad + \mathrm{Var}[Z_0, Z_0]. \end{aligned}$$

If $\{X_i\}$ is stationary under θ_1, so is $\{Y_i\}$. Therefore, $\mathrm{E}_{\theta_1}|Y_i|^{2+\delta} = \mathrm{E}_{\theta_1}|Y_1|^{2+\delta} = C^{(1)}_{\theta_1,\theta_2}\ \forall\ i$, and

$$\begin{aligned}\sum_{i,j=1}^{n} \mathrm{Cov}_{\theta_1}(Y_i, Y_j) &\leq \sum_{i,j=1}^{n} \left(\frac{4}{n} + 6n^{\delta/2} C^{(1)}_{\theta_1,\theta_2} \right) \alpha_{|j-i|-1}^{\delta/(2+\delta)} \\ &\leq n \left(\frac{4}{n} + 6n^{\delta/2} C^{(1)}_{\theta_1,\theta_2} \right) \left(\sum_{h\geq 1} \alpha_{h-1}^{\delta/(2+\delta)} \right). \end{aligned} \tag{A31}$$

Again, using Lemma B.1.4 on $\mathrm{Cov}_{\theta_1}(Y_i, Z_0)$, yields

$$\sum_{i=1}^{n} \mathrm{Cov}_{\theta_1}(Y_i, Z_0) \leq \left(\frac{4}{n} + 2n^{\delta/2}(C_\theta + D_{1,2} + \sqrt{C_\theta D_{1,2}}) \right) \left(\sum_{h\geq 1} \alpha_h^{\delta/(2+\delta)} \right). \tag{A32}$$

Finally, using Equations (A31) and (A32) we have

$$\begin{aligned}\mathrm{Var}_{\theta_1}[r_n(\theta_2, \theta_1)] \leq n &\left(\frac{4}{n} + 6n^{\delta/2} C^{(1)}_{\theta_1,\theta_2} \right) \left(\sum_{h\geq 1} \alpha_{h-1}^{\delta/(2+\delta)} \right) + \\ &\left(\frac{4}{n} + 2n^{\delta/2}\left(C^{(1)}_{\theta_1,\theta_2} + D_{1,2} + \sqrt{C^{(1)}_{\theta_1,\theta_2} D_{1,2}}\right) \right) \left(\sum_{h\geq 1} \alpha_h^{\delta/(2+\delta)} \right) \\ &+ \mathrm{Cov}_{\theta_1}(Z_0, Z_0). \end{aligned}$$

□

Appendix B.2. Proofs for Stationary Markov Data-Generating Models

Proof of Theorem 2

Part 1: Verifying condition (i) of Corollary 1.

We substitute the true parameter θ_0 for θ_1 and θ for θ_2. We also set $q_1^{(0)}$ to be the invariant distribution of the Markov chain under θ_0, q_0, and $q_2^{(0)}$ as the invariant distribution of the Markov chain under θ, q_θ. Applying the fact that these Markov chains are stationary to Proposition 2, we have

$$\begin{aligned}\mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)}) &= n\mathrm{E}\left[\log\left(\frac{p_{\theta_0}(X_1|X_0)}{p_\theta(X_1|X_0)} \right) \right] + \mathrm{E}[Z_0], \\ &\leq n \sum_{j=1}^{m} \mathrm{E}\left[M_j^{(1)}(X_1, X_0) \right] |f_j^{(1)}(\theta, \theta_0)| + \sum_{k=1}^{m} \mathrm{E}[M_k^{(2)}(X_0)] |f_k^{(2)}(\theta, \theta_0)|, \end{aligned} \tag{A33}$$

where the inequality follows from Assumption 1. Therefore, it follows that

$$\begin{aligned}\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)}) \rho_n(d\theta) \leq n &\sum_{j=1}^{m} \mathrm{E}\left[M_j^{(1)}(X_1, X_0) \right] \int |f_j^{(1)}(\theta, \theta_0)| \rho_n(d\theta) \\ &+ \sum_{k=1}^{m} \mathrm{E}[M_k^{(2)}(X_0)] | \int f_k^{(2)}(\theta, \theta_0) | \rho_n(d\theta). \end{aligned}$$

By Assumption 1(i), it follows that

$$\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)}) \rho_n(d\theta) \leq n \sum_{j=1}^{m} \mathrm{E}\left[M_j^{(1)}(X_1, X_0) \right] \frac{C}{\sqrt{n}} + \sum_{k=1}^{m} \mathrm{E}[M_k^{(2)}(X_0)] \frac{C}{\sqrt{n}} \leq n\epsilon_n^{(1)},$$

where $\epsilon_n^{(1)} = O\Big(\frac{1}{\sqrt{n}}\Big)$.

Part 2: Verifying condition (ii) of Corollary 1. Again, using Proposition 3 along with the fact that the Markov chain is stationary we have

$$\begin{aligned}\operatorname{Var}[r_n(\theta,\theta_0)] \le n\left(\frac{4}{n} + 6n^{\delta/2}C^{(1)}_{\theta_0,\theta}\right)\left(\sum_{k\ge 0}\alpha_k^{\delta/(2+\delta)}\right)\\ + \left(\frac{4}{n^2} + 2n^{\delta/2}(C^{(1)}_{\theta_0,\theta} + D_{\theta_0,\theta} + \sqrt{C^{(1)}_{\theta_0,\theta}D_{\theta_0,\theta}})\right)\left(\sum_{k\ge 1}\alpha_k^{\delta/(2+\delta)}\right)\\ + \operatorname{Var}[Z_0].\end{aligned}$$

It then follows that

$$\begin{aligned}\int \operatorname{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) \le n\left(\frac{4}{n} + 6n^{\delta/2}\int C^{(1)}_{\theta_0,\theta}\rho_n(d\theta)\right)\left(\sum_{k\ge 1}\alpha_{k-1}^{\delta/(2+\delta)}\right) + \int \operatorname{Var}[Z_0]\rho_n(d\theta)\\ + \left(\frac{4}{n^2} + 2n^{\delta/2}(\int C^{(1)}_{\theta_0,\theta}\rho_n(d\theta)\right.\\ \left. + \int D_{\theta_0,\theta}\rho_n(d\theta) + \int \sqrt{C^{(1)}_{\theta_0,\theta}D_{\theta_0,\theta}}\rho_n(d\theta))\right)\left(\sum_{k\ge 1}\alpha_k^{\delta/(2+\delta)}\right).\end{aligned}$$

First, consider the term $\int C^{(1)}_{\theta_0,\theta}\rho_n(\theta)$, and observe that

$$\int C^{(1)}_{\theta_0,\theta}\rho_n(d\theta) = \int \mathrm{E}\log\left|\frac{p_{\theta_0}(X_1|X_0)}{p_\theta(X_1|X_0)}\right|^{2+\delta}\rho_n(d\theta).$$

By Assumption 1, we have

$$\int \mathrm{E}\log\left|\frac{p_{\theta_0}(X_1|X_0)}{p_\theta(X_1|X_0)}\right|^{2+\delta}\rho_n(d\theta) \le \int \mathrm{E}\left[\sum_{j=1}^m M_j^{(1)}(X_1,X_0)|f_k^{(1)}(\theta,\theta_0)|\right]^{2+\delta}\rho_n(d\theta).$$

Since the function $x \mapsto x^{2+\delta}$ is convex, we can apply Jensen's inequality to obtain,

$$\left(\sum_{j=1}^m M_j^{(1)}(X_1,X_0)|f_k^{(1)}(\theta,\theta_0)|\right)^{2+\delta} \le m^{1+\delta}\sum_{k=1}^m M_j^{(1)}(X_1,X_0)^{2+\delta}|f_k^{(1)}(\theta,\theta_0)|^{2+\delta}.$$

Therefore, it follows that

$$\begin{aligned}\int \mathrm{E}\log\left|\frac{p_{\theta_0}(X_1|X_0)}{p_\theta(X_1|X_0)}\right|^{2+\delta}\rho_n(d\theta) \le m^{1+\delta}\sum_{k=1}^m \mathrm{E}[M_k^{(1)}(X_1,X_0)^{2+\delta}]\\ \times \int |f_k^{(1)}(\theta,\theta_0)|^{2+\delta}\rho_n(d\theta).\end{aligned}$$

By Assumption 1, $\int |f_k(\theta,\theta_0)|^{2+\delta}\rho_n(d\theta) < \frac{C}{n}$ and $\mathrm{E}[M_k^{(1)}(X_1,X_0)^{2+\delta}] < B$, implying that

$$\int C^{(1)}_{\theta_0,\theta}\rho_n(d\theta) \le m^{1+\delta}\sum_{k=1}^m B\frac{C}{n} = m^{2+\delta}\frac{BC}{n}.$$

Since $\left(\sum_{k\geq 0}\alpha_k^{\delta/(2+\delta)}\right) < \infty$, it follows that $\left(\frac{4}{n} + 6n^{\delta/2}\int C^{(1)}_{\theta_0,\theta}\rho_n(d\theta)\right)\left(\sum_{k\geq 1}\alpha_{k-1}^{\delta/(2+\delta)}\right) = O(\frac{n^{\delta/2}}{n})$. Similarly, we can show that $\int D_{\theta_0,\theta}\rho_n(d\theta) = O(\frac{1}{n})$, and $\int \mathrm{Var}[Z_0]\rho_n(d\theta) = O(\frac{1}{n})$.

For the final term $\int\sqrt{C^{(1)}_{\theta_0,\theta}D_{\theta_0,\theta}\rho_n(d\theta)}$, use the Cauchy-Schwarz inequality to obtain the upper bound $\left(\int C^{(1)}_{\theta_0,\theta}\rho_n(d\theta)\int D_{\theta_0,\theta}\rho_n(d\theta)\right)^{1/2}$ which is also of order $O(\frac{1}{n})$. Combining all of these together we have

$$\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) \leq n\epsilon_n^{(2)},$$

for some $\epsilon_n^{(2)} = O(\frac{n^{\delta/2}}{n})$.

Since $\mathcal{K}(\rho_n,\pi) < \sqrt{n}C = n\frac{C}{\sqrt{n}}$, it follows that $\mathcal{K}(\rho_n,\pi) < n\epsilon_n^{(3)}$, where $\epsilon_n^{(3)} = O(1/\sqrt{n})$ as before. Finally, by choosing $\epsilon_n = \max(\epsilon_n^{(1)},\epsilon_n^{(2)},\epsilon_n^{(3)})$, our theorem is proved. □

Appendix B.3. Proofs for Non-Stationary, Ergodic Markov Data-Generating Models

Appendix B.3.1. Proof of Theorem 3

Part 1: Verifying condition (i) of Corollary 1: As in the proof of Theorem 2 substitute the true parameter θ_0 for θ_1 and θ for θ_2 in . We also set $q_1^{(0)}$ and $q_2^{(0)}$ to the distribution $q^{(0)}$. Applying Proposition 2 to the corresponding transition kernels and initial distribution we have,

$$\begin{aligned}\mathcal{K}(P^{(n)}_{\theta_0},P^{(n)}_{\theta}) &= \sum_{i=1}^{n}\mathrm{E}\left[\log\left(\frac{p_{\theta_0}(X_i|X_{i-1})}{p_\theta(X_i|X_{i-1})}\right)\right] + \mathrm{E}\left[\log\left(\frac{D(X_0)}{D(X_0)}\right)\right] \\ &= \sum_{i=1}^{n}\mathrm{E}\left[\log\left(\frac{p_{\theta_0}(X_i|X_{i-1})}{p_\theta(X_i|X_{i-1})}\right)\right].\end{aligned} \quad \text{(A34)}$$

Now, applying Assumption 1, we can bound the previous equation as follows,

$$\begin{aligned}\mathcal{K}(P^{(n)}_{\theta_0},P^{(n)}_{\theta}) &\leq \sum_{i=1}^{n}\mathrm{E}\left[\sum_{k=1}^{m}M_k^{(1)}(X_i,X_{i-1})|f_k^{(1)}(\theta,\theta_0)|\right] \\ &= \sum_{i=1}^{n}\sum_{k=1}^{m}\mathrm{E}\left[M_k^{(1)}(X_i,X_{i-1})\right]|f_k^{(1)}(\theta,\theta_0)|.\end{aligned} \quad \text{(A35)}$$

Since $M_k^{(1)}$'s are bounded there exists a constant Q so that,

$$\begin{aligned}\int\mathcal{K}(P^{(n)}_{\theta_0},P^{(n)}_{\theta})\rho_n(d\theta) &\leq Q\int\sum_{i=1}^{n}\sum_{k=1}^{m}|f_k^{(1)}(\theta,\theta_0)|\rho_n(d\theta) \\ &= Qn\sum_{k=1}^{m}\int|f_k^{(1)}(\theta,\theta_0)|\rho_n(d\theta).\end{aligned}$$

By Assumption 19 in Assumption 1, it follows that

$$\int\mathcal{K}(P^{(n)}_{\theta_0},P^{(n)}_{\theta})\rho_n(d\theta) \leq Qn\sum_{k=1}^{m}\frac{C}{\sqrt{n}} = nmQ\frac{C}{\sqrt{n}} = n\epsilon_n^{(1)},$$

for some $\epsilon_n^{(1)} = O(\frac{1}{\sqrt{n}})$.

Part 2: Verifying condition (ii) of Corollary 1: As in the previous part, $Z_0 = 0$, implying that D_{θ,θ_0}. Applying Proposition 3 and integrating with respect to ρ_n, we obtain

$$\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) \le \sum_{i=1}^{n}\left(\frac{4}{n} + 2n^{\delta/2}\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta)\right)\left(\alpha_{i-1}^{\delta/(2+\delta)}\right)$$
$$+ \sum_{i,j=1}^{n}\left(\frac{4}{n} + 2n^{\delta/2}(\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta) + \int C^{(j)}_{\theta_0,\theta}\rho_n(d\theta) + \int \sqrt{C^{(i)}_{\theta_0,\theta}C^{(j)}_{\theta_0,\theta}}\rho_n(d\theta))\right)$$
$$\times \left(\alpha_{|i-j|-1}^{\delta/(2+\delta)}\right). \tag{A36}$$

First, consider the term $\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta)$. Using Assumption 1, we can upper bound $C^{(i)}_{\theta_0,\theta}$ as,

$$C^{(i)}_{\theta_0,\theta} \le \mathrm{E}\left[\sum_{k=1}^{m} M_k^{(1)}(X_i, X_{i-1})|f_k^{(1)}(\theta,\theta_0)|\right]^{2+\delta}$$
$$\le \sum_{k=1}^{m} m^{1+\delta}\mathrm{E}\left[\left(M_k^{(1)}(X_i, X_{i-1})|f_k^{(1)}(\theta,\theta_0)|\right)^{2+\delta}\right] \text{(by Jensen's inequality)}$$
$$= \sum_{k=1}^{m} m^{1+\delta}\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})^{2+\delta}\right]|f_k^{(1)}(\theta,\theta_0)|^{2+\delta}.$$

Since $M_k^{(1)}$'s are upper bounded by Q, it follows from the previous expression that, $C^{(i)}_{\theta_0,\theta} \le \sum_{k=1}^{m} m^{1+\delta}Q^{2+\delta}|f_k^{(1)}(\theta,\theta_0)|^{2+\delta}$.

Hence, from Assumption 1, we get,

$$\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta) \le \sum_{k=1}^{m} m^{1+\delta}Q^{2+\delta}\int |f_k^{(1)}(\theta,\theta_0)|^{2+\delta}\rho_n(d\theta) \le (mQ)^{2+\delta}\frac{C}{n}.$$

Using the upper bound above, we can say for an L large enough, $\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta) \le \frac{L}{n}$. Next, by the Cauchy-Schwarz inequality, we have that $\int \sqrt{C^{(i)}_{\theta_0,\theta}C^{(j)}_{\theta_0,\theta}\rho_n(d\theta))} < \sqrt{\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta)\int C^{(j)}_{\theta_0,\theta}\rho_n(d\theta))} \le \frac{L}{n}$. Thus, we have the following upper bound.

$$\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) \le \sum_{i=1}^{n}\left(\frac{4}{n} + 2n^{\delta/2}\frac{L}{n}\right)\left(\alpha_{i-1}^{\delta/(2+\delta)}\right)$$
$$+ \sum_{i,j=1}^{n}\left(\frac{4}{n} + 2n^{\delta/2}(\frac{L}{n} + \frac{L}{n} + \frac{L}{n})\right)\left(\alpha_{|i-j|-1}^{\delta/(2+\delta)}\right)$$
$$= \left(\frac{4}{n} + 2n^{\delta/2}\frac{L}{n}\right)\left(\sum_{i=1}^{n}\alpha_{i-1}^{\delta/(2+\delta)}\right)$$
$$+ \left(\frac{4}{n} + 6n^{\delta/2}\frac{L}{n}\right)\left(\sum_{i,j=1}^{n}\alpha_{|i-j|-1}^{\delta/(2+\delta)}\right).$$

Since $\sum_{i,j=1}^{n}\alpha_{|i-j|-1}^{\delta/(2+\delta)} < n\sum_{k\ge1}\alpha_{k-1}^{\delta/(2+\delta)} < \infty$, we have that for some $\epsilon_n^{(2)} = O(\frac{n^{\delta/2}}{n})$,

$$\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) < n\epsilon_n^{(2)}.$$

Since $\mathcal{K}(\rho_n,\pi) \le \sqrt{n}C$, following the concluding argument in Theorem 2 completes the proof. $\square$

Appendix B.3.2. Proof of Proposition 8

We verify Assumption 1 and the proof follows from Theorem 3. For $i \in \{1, 2, \ldots, K-1\}$,

$$p_\theta(j|i) = \begin{cases} \theta & \text{if } j = i-1, \\ 1-\theta & \text{if } j = i+1. \end{cases}$$

If $i = 0$ or $i = K$, then the Markov chain goes back to 1 or $K-1$, respectively, with probability 1. With the convention $\log \frac{0}{0} = 0$, the log ratio of the transition probabilities becomes,

$$|\log p_{\theta_0}(X_1|X_0) - \log p_\theta(X_1|X_0)| = I_{[X_1 = X_0+1]} \log\left(\frac{\theta_0}{\theta}\right) + I_{[X_1 = X_0-1]} \log\left(\frac{1-\theta_0}{1-\theta}\right).$$

In this case, $m = 2$. $M_1^{(1)}(X_1, X_0) = I_{[X_1 = X_0+1]}$ and $M_2^{(1)}(X_1, X_0) = I_{[X_1 = X_0-1]}$, both of which are bounded. Let $f_1^{(1)}(\theta, \theta_0) := \log\left(\frac{\theta_0}{\theta}\right)$ suppose $f_2^{(1)}(\theta, \theta_0) := \log\left(\frac{1-\theta_0}{1-\theta}\right)$.

The stationary distribution $q_\theta(i) = \frac{1}{K}\ \forall\ i \in 1, 2, \ldots, K$. Hence the log of the ratio of the invariant distribution becomes

$$\log q_0(x) - \log q_\theta(x) = 0, \tag{A37}$$

and we can set $M_i^{(2)}(\cdot) := 1$ and $f_i^{(2)}(\cdot,\cdot) := 0$ for $i \in \{1, 2\}$. Thus, to prove the concentration bound for this Markov chain it is enough to assume that $\delta = 1$ and show that $\int [f_1^{(1)}(\theta, \theta_0)]^3 \rho_n(d\theta) < \frac{C}{n}$ and $\int [f_2^{(1)}(\theta, \theta_0)]^3 \rho_n(d\theta) < \frac{C}{n}$ for some constant $C > 0$.

As given, $\{\rho_n\}$ is a sequence of beta probability distribution functions, with parameters a_n, b_n that satisfy the constraint $\frac{a_n}{a_n+b_n} = \theta_0$. Specifically, we choose $a_n = n\theta_0$ and (therefore) $b_n = n(1-\theta_0)$. Thus, we get the following,

$$\begin{aligned} \int |f_1^{(1)}(\theta, \theta_0)|^3 \rho_n(d\theta) &= \int \left|\log\left(\frac{\theta_0}{\theta}\right)\right|^3 \rho_n(d\theta) \\ &< \int \left|\frac{\theta_0}{\theta} - 1\right|^3 \rho_n(d\theta) \\ &= \frac{1}{\mathrm{Beta}(a_n, b_n)} \int_0^1 \left|\frac{\theta_0 - \theta}{\theta}\right|^3 \theta^{a_n - 1}(1-\theta)^{b_n - 1} d\theta. \end{aligned}$$

Since $\theta_0, \theta \in (0, 1)$, so is $\frac{|\theta_0 - \theta|}{2}$, giving $|\theta_0 - \theta|^3 < 2(\theta_0 - \theta)^2$. We use that fact to arrive at

$$\begin{aligned} \int |f_1^{(1)}(\theta, \theta_0)|^3 \rho_n(d\theta) &\le \frac{2}{\mathrm{Beta}(a_n, b_n)} \int_0^1 (\theta_0 - \theta)^2 \theta^{a_n - 4}(1-\theta)^{b_n - 1} d\theta \\ &= \frac{2\mathrm{Beta}(a_n - 3, b_n)}{\mathrm{Beta}(a_n, b_n)} \frac{(a_n - 3)(b_n)}{(a_n + b_n - 3)^2(a_n + b_n - 2)}. \end{aligned}$$

From our choice of a_n and b_n, $\frac{2\mathrm{Beta}(a_n - 3, b_n)}{\mathrm{Beta}(a_n, b_n)} = O(1)$, and plugging the values of a_n and b_n into $\frac{(a_n-3)(b_n)}{(a_n+b_n-3)^2(a_n+b_n-2)}$, we get $\frac{(a_n-3)(b_n)}{(a_n+b_n-3)^2(a_n+b_n-2)} = \frac{1}{n} \frac{(\theta_0 - \frac{3}{n})(1-\theta_0)}{(1-\frac{3}{n})^2(1-\frac{2}{n})}$, which is upper bounded by $\frac{C_1}{n}$ for some constant $C_1 > 0$. Hence,

$$\int |f_1^{(1)}(\theta, \theta_0)|^3 \rho_n(d\theta) < \frac{C_1}{n}.$$

Similarly, we can also show that,

$$\int |f_2^{(1)}(\theta, \theta_0)|^3 \rho_n(d\theta) < \frac{C_2}{n}.$$

Finally, from Proposition A.2.1, we get that $\mathcal{K}(\rho_n, \pi) < C + \frac{1}{2}\log(n)$ for some large constant C. Hence, $\mathcal{K}(\rho_n, \pi) < C_3\sqrt{n}$ for some constant $C_3 > 0$. Choosing $C = \max(C_1, C_2, C_3)$, we satisfy all the conditions of Assumption 1 and Theorem 3. □

Appendix B.3.3. Proof of Proposition 9

For the purpose of this proof, we choose ρ_n's with scaled Beta distribution with parameters $a_n = n(\theta_0/2)$ and $b_n = n(1-\theta_0/2)$. Since, ρ_n is a scaled Beta distribution with the scaling factors $m = 0.5$ and $c = 0$, the pdf of ρ_n is given by

$$\rho_n(\theta) = \frac{2}{\mathrm{Beta}(a_n, b_n)}(2\theta)^{a_n}(1-2\theta)^{b_n}$$

Since this is a scaled distribution, $E_{\rho_n}[\theta] = 2\frac{a_n}{a_n+b_n} = \theta_0$ and there exists a constant $\sigma > 0$, $\mathrm{Var}_{\rho_n}[\theta] = \frac{\sigma^2}{n}$. Now, we analyse the transition probabilities. For $i \in \{1, 2, \dots\}$, the Birth-Death process has transition probabilities

$$p_\theta(j|i) = \begin{cases} \theta & \text{if } j = i-1, \\ 1-\theta & \text{if } j = i+1. \end{cases}$$

If $i = 0$, then the Markov chain goes to 1 with probability 1. Hence with the convention $\log\frac{0}{0} = 0$ the ratio of the log of the transition probabilities becomes,

$$|\log p_{\theta_0}(X_1|X_0) - \log p_\theta(X_1|X_0)| = I_{[X_1=X_0+1]}\log\left[\frac{\theta_0}{\theta}\right] + I_{[X_1=X_0-1]}\log\left[\frac{1-\theta_0}{1-\theta}\right].$$

In this case, $m = 3$. $M_1^{(1)}(X_1, X_0) = I_{[X_1=X_0+1]}$ and $M_2^{(1)}(X_1, X_0) = I_{[X_1=X_0-1]}$. Define $M_3^{(1)}(X_1, X_0) := 1$. All these random variables are bounded. Define $f_1^{(1)}(\theta, \theta_0) := \log\left[\frac{\theta_0}{\theta}\right]$, $f_2^{(1)}(\theta, \theta_0) := \log\left[\frac{1-\theta_0}{1-\theta}\right]$ and $f_3^{(1)}(\theta, \theta_0) := 0$. Similarly as in the proof on Proposition 8,

$$\int [f_1^{(1)}(\theta, \theta_0)]^3 \rho_n(d\theta) < \frac{C_1}{n}, \text{ and}$$
$$\int [f_2^{(1)}(\theta, \theta_0)]^3 \rho_n(d\theta) < \frac{C_2}{n}.$$

The stationary distribution is given by $q_\theta(i) = (\frac{\theta}{1-\theta})^{i-1} q_\theta(1)\ \forall\ i \in 1, 2, \dots$, so that $q_\theta(i) = (1-\theta)(\frac{\theta}{1-\theta})^{i-1}$ Hence the log of the ratio of the invariant distribution becomes

$$\log q_0(i) - \log q_\theta(i) = \log\left[\frac{1-\theta_0}{1-\theta}\right] + (i-1)\log\left[\frac{\theta_0}{\theta}\right] - (i-1)\log\left[\frac{1-\theta_0}{1-\theta}\right] \tag{A38}$$

We define $M_1^{(2)}(X_0) := 1$, and $M_2^{(2)}(X_0) = M_3^{(2)}(X_0) := X_0 - 1$. We can write $\mathrm{E}_{q^{(0)}}[M_2^{(2)}(X_0)]^2 = \sum_{i=1}^{\infty}(i-1)^2 q^{(0)}(i) < \sum_{i=1}^{\infty} i^2 q^{(0)}(i)$. We have chosen $q^{(0)}$ such that $\sum_{i=1}^{\infty} i^2 q^{(0)}(i)$ is bounded. Hence, $\mathrm{E}_{q^{(0)}}[M_2^{(2)}(X_0)]^2 < \infty$. To verify Assumption i define, $f_1^{(2)}(\theta, \theta_0) = -f_3^{(2)}(\theta, \theta_0) := \log\left[\frac{1-\theta_0}{1-\theta}\right]$, and define $f_2^{(2)}(\theta, \theta_0) := \log\left[\frac{\theta_0}{\theta}\right]$. Therefore following the proof of Proposition 8,

$$\int |f_1^{(2)}(\theta, \theta_0)|^3 \rho_n(d\theta) = \int |f_3^{(2)}(\theta, \theta_0)|^3 \rho_n(d\theta) = \int |f_2^{(1)}(\theta, \theta_0)|^3 \rho_n(d\theta) < \frac{C_2}{n}, \text{ and ,}$$
$$\int |f_2^{(2)}(\theta, \theta_0)|^3 \rho_n(d\theta) = \int |f_1^{(1)}(\theta, \theta_0)|^3 \rho_n(d\theta) < \frac{C_1}{n}.$$

Finally, we take the KL-divergence $\mathcal{K}(\rho_n, \pi)$. ρ_n follows a scaled Beta distribution on $(0, 1/2)$ with parameters $a_n = n(\theta_0/2)$ and $b_n = n(1 - \theta_0/2)$, while π follows a scaled Beta distribution on $(0, 1/2)$ with parameters a and b. Thus,

$$\mathcal{K}(\rho_n, \pi) = \int_0^{\frac{1}{2}} \log\left(\frac{\rho_n(\theta)}{\pi(\theta)}\right)\rho_n(d\theta),$$

which, by substituting $t = 2\theta$, we get,

$$\mathcal{K}(\rho_n, \pi) = 2\int_0^1 \log\left(\frac{\rho_n(t)}{\pi(t)}\right)\rho_n(dt).$$

$\int_0^1 \log\left(\frac{\rho_n(t)}{\pi(t)}\right)\rho_n(dt)$ is the KL-divergence between a Beta distribution with parameters a_n and b_n and a Beta distribution with parameters a and b. An application of Proposition A.2.1 gives us for a constant $C_1 > 0$,

$$\int_0^1 \log\left(\frac{\rho_n(t)}{\pi(t)}\right)\rho_n(dt) < C_1 + \frac{1}{2}\log(n).$$

Hence we can say, $\mathcal{K}(\rho_n, \pi) < 2\left[C_1 + \frac{1}{2}\log(n)\right]$. Thus, we now get that for some constant $C_3 > 0$,

$$\mathcal{K}(\rho_n, \pi) < C_3\sqrt{n}.$$

Choosing $C = \max(C_1, C_2, C_3)$ we satisfy all of the conditions of Assumption 1 and thus by Theorem 3, we are complete the proof. □

Appendix B.3.4. Proof of Theorem 4

Part 1: Verifying condition (i) of Corollary 1 As in the proof of Theorem 2 substitute the true parameter θ_0 for θ_1 and θ for θ_2. We also set our initial distributions $q_1^{(0)}$ and $q_2^{(0)}$ to the known initial distribution $q^{(0)}$. A method similar to Equation (A35), yields

$$\mathcal{K}(P_{\theta_0}^{(n)}, P_{\theta}^{(n)}) \le \sum_{i=1}^{n}\sum_{k=1}^{m} \mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right]|f_k^{(1)}(\theta, \theta_0)|.$$

Because $M_k^{(1)}$s satisfy Assumption 2, it follows by the application of Theorem 2.3, [21] that $\exists\ \lambda > 0$ such that for any $0 < \kappa \le \lambda$, and for some $\zeta \in (0,1)$ possibly depending upon λ,

$$\mathrm{E}\left[e^{\kappa M_k^{(1)}(X_i, X_{i-1})}\middle| X_1, X_0\right] \le \zeta^{i-1}e^{\kappa M_k^{(1)}(X_1, X_0)} + \frac{1-\zeta^i}{1-\zeta}\mathcal{D}e^{\kappa a} \quad \text{for all } i > 1.$$

We rewrite $\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})|X_1, X_0\right]$ as follows:

$$\begin{aligned}\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})|X_1, X_0\right] &= \frac{\mathrm{E}[\kappa M_k^{(1)}(X_i, X_{i-1})|X_1, X_0]}{\kappa} \\ &\le \frac{\mathrm{E}[e^{\kappa M_k^{(1)}(X_i, X_{i-1})}|X_1, X_0]}{\kappa}.\end{aligned}$$

Therefore, $\sum_{i=1}^n \mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right]$ can be upper bounded as,

$$\begin{aligned}\sum_{i=1}^n \mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right] &= \sum_{i=1}^n \mathrm{E}\left[\kappa M_k^{(1)}(X_i, X_{i-1})|X_1, X_0\right]\kappa^{-1} \\ &\leq \sum_{i=1}^n \left[\zeta^{i-1}\mathrm{E}e^{\kappa M_k^{(1)}(X_1,X_0)} + \frac{1-\zeta^i}{1-\zeta}\mathcal{D}e^{\kappa a}\right]\kappa^{-1}.\end{aligned}$$

Since, $\zeta \in (0,1)$, $\zeta^i < 1$. Hence, we can write that,

$$\begin{aligned}\sum_{i=1}^n \left[\zeta^{i-1}\mathrm{E}e^{\kappa M_k^{(1)}(X_1,X_0)} + \frac{1-\zeta^i}{1-\zeta}\mathcal{D}e^{\kappa a}\right]\kappa^{-1} &\leq \sum_{i=1}^n \left[\zeta^{i-1}\mathrm{E}e^{\kappa M_k^{(1)}(X_1,X_0)} + \frac{1}{1-\zeta}\mathcal{D}e^{\kappa a}\right]\kappa^{-1} \\ &= \left[\frac{1-\zeta^n}{1-\zeta}\mathrm{E}e^{\kappa M_k^{(1)}(X_1,X_0)} + \frac{n}{1-\zeta}\mathcal{D}e^{\kappa a}\right]\kappa^{-1} \\ &\leq nL,\end{aligned}$$

for a large constant L. Therefore $\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta)$ can be upper bounded as follows,

$$\begin{aligned}\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) &\leq \int \sum_{k=1}^m nL|f_k^{(1)}(\theta,\theta_0)|\rho_n(d\theta) \\ &= \sum_{k=1}^m nL \int |f_k^{(1)}(\theta,\theta_0)|\rho_n(d\theta).\end{aligned}$$

By Assumption 1, $\int |f_k^{(1)}(\theta,\theta_0)|\rho_n(d\theta) < \frac{C}{n}$, hence,

$$\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \leq nL\frac{C}{\sqrt{n}}.$$

Hence, for some $\epsilon_n^{(1)} = \mathrm{O}(\frac{1}{\sqrt{n}})$, we have obtained that, $\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \leq n\epsilon_n^{(1)}$.

Part 2: Verifying condition (ii) of Corollary 1: Similar to as in the proof of Theorem 3, we upper bound $\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta)$ by

$$\begin{aligned}\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) \leq \sum_{i,j=1}^n \Bigg(\frac{4}{n} + 2n^{\delta/2}\Bigg(\int C_{\theta_0,\theta}^{(i)}\rho_n(d\theta) + \int C_{\theta_0,\theta}^{(j)}\rho_n(d\theta) \qquad \text{(A39)} \\ + \int \sqrt{C_{\theta_0,\theta}^{(i)}C_{\theta_0,\theta}^{(j)}}\rho_n(d\theta)\Bigg)\Bigg)\left(\alpha_{|i-j|-1}^{\delta/(2+\delta)}\right) \qquad \text{(A40)} \\ + \sum_{i=1}^n \left(\frac{4}{n} + 2n^{\delta/2}\int C_{\theta_0,\theta}^{(i)}\rho_n(d\theta)\right)\left(\alpha_{i-1}^{\delta/(2+\delta)}\right),\end{aligned}$$

where $C_{\theta_0,\theta}$ is upper bounded as

$$C_{\theta_0,\theta}^{(i)} \leq \sum_{k=1}^m m^{1+\delta}\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right]^{2+\delta}|f_k^{(1)}(\theta,\theta_0)|^{2+\delta}.$$

There exists a constant C_δ depending upon δ such that,

$$\begin{aligned}[M_k^{(1)}]^{2+\delta}(X_i, X_{i-1}) &= \frac{\kappa^{2+\delta}[M_k^{(1)}]^{2+\delta}(X_i, X_{i-1})^{2+\delta}}{\kappa^{2+\delta}} \\ &\leq \frac{e^{\kappa M_k^{(1)}(X_i,X_{i-1})} + C_\delta}{\kappa^{2+\delta}}.\end{aligned}$$

By expressing $\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})^{2+\delta}\right] = \mathrm{E}\left[\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})^{2+\delta}|X_1, X_0\right]\right]$ and following a method similar to the previous part, we get,

$$\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})^{2+\delta}\right] \leq \frac{\left[\zeta^i \mathrm{E}e^{\kappa M_k^{(1)}(X_1,X_0)} + \frac{1-\zeta^i}{1-\zeta}\mathcal{D}e^{\kappa a}\right] + C_\delta}{\kappa^{2+\delta}}.$$

The fact that $0 < \zeta < 1$ implies that $0 < \zeta^i < \zeta$. This gives us the following,

$$\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})^{2+\delta}\right] \leq \frac{\left[\zeta \mathrm{E}e^{\kappa M_k^{(1)}(X_1,X_0)} + \frac{1}{1-\zeta}\mathcal{D}e^{\kappa a}\right] + C_\delta}{\kappa^{2+\delta}}.$$

Since $\kappa < \lambda$, by the application of Jensen's inequality, we get

$$\begin{aligned}\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})^{2+\delta}\right] &\leq \frac{\left[\zeta \mathrm{E}e^{\lambda M_k^{(1)}(X_1,X_0)} + \frac{1}{1-\zeta}\mathcal{D}e^{\kappa a}\right] + C_\delta}{\kappa^{2+\delta}} \\ &= \frac{\left[\zeta \int e^{\lambda M_k^{(1)}(x_1,x_0)} p_{\theta_0}(x_1|x_0)D(x_0)dx_1dx_0 + \frac{1}{1-\zeta}\mathcal{D}e^{\kappa a}\right] + C_\delta}{\kappa^{2+\delta}}.\end{aligned}$$

We know that $\int |f_k^{(1)}(\theta,\theta_0)|^{2+\delta}\rho_n(d\theta) < \frac{C}{n}$. Thus, following Assumption 1 we can say that, for a large constant L, $\int C_{\theta_0,\theta}^{(i)}\rho_n(d\theta) \leq \frac{L}{n}$. The rest of the proof follows similarly as in the proof of Theorem 3, and we obtain an $\epsilon_n^{(2)} = \mathrm{O}(\frac{n^{\delta/2}}{n})$, such that,

$$\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) < n\epsilon_n^{(2)}.$$

Since, $\mathcal{K}(\rho_n, \pi) \leq \sqrt{n}C$, similar arguments as in the proof of Theorem 2 holds. The theorem is thus proved.

Appendix B.3.5. Proof of Theorem 5

Part 1: Verifying condition (i) of Corollary 1 As in the proof of Theorem 2 substitute the true parameter θ_0 for θ_1 and θ for θ_2. We also set $q_1^{(0)}$ and $q_2^{(0)}$ to the known initial distribution $q^{(0)}$. Similar to the steps leading to Equation (A35), we get

$$\mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)}) \leq \sum_{i=1}^{n}\sum_{k=1}^{m} \mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right]|f_k^{(1)}(\theta,\theta_0)|.$$

Consider the term $\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right]$. With $q_{\theta_0}^{(i-1)}$ the marginal distribution of X_{i-1}, we have

$$\begin{aligned}\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right] &= \int M_k^{(1)}(x_i, x_{i-1})p_{\theta_0}(x_i|x_{i-1})q_{\theta_0}^{(i-1)}(x_{i-1})dx_idx_{i-1}. \\ \mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right] &= \int M_k^{(1)}(x_i, x_{i-1})p_{\theta_0}(x_i|x_{i-1})p_{\theta_0}^{i-1}(x_{i-1}|x_0)q_{\theta_0}^{(0)}(x_0)dx_0dx_idx_{i-1}\end{aligned}$$

Recall that the marginal density satisfies $q_{\theta_0}^{(i-1)}(x_{i-1}) = \int p_{\theta_0}^{i-1}(x_{i-1}|x_0)q_{\theta_0}^{(0)}(x_0)d(x_0)$, where $p_{\theta_0}^i(\cdot|x_0)$ is the i-step transition probability. Then

$$\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right] = \int \mathrm{E}\left[M_k^{(1)}(X_i, x_{i-1})|x_{i-1}\right]p_{\theta_0}^{i-1}(x_{i-1}|x_0)q_{\theta_0}^{(0)}(x_0)dx_0dx_{i-1}.$$

Since the Markov chain $\{X_n\}$ satisfies Assumption A.1.1, we know by the application of Theorem A.1.1 that $\{X_n\}$ is V-geometrically ergodic. Hence, $\exists\ \tau < 1$, $R < \infty$ such that $\forall\ |f| < V$

$$|\int f(x_{i-1})p_{\theta_0}^{i-1}(x_{i-1}|x_0)dx_{i-1} - \int f(x_{i-1})q_{\theta_0}(x_{i-1})dx_{i-1}| < RV(x_0)\tau^{i-1},$$

where q_{θ_0} is the stationary distribution, implying that

$$\int f(x_{i-1})p_{\theta_0}^{i-1}(x_{i-1}|x_0)dx_{i-1} < \int f(x_{i-1})q_{\theta_0}(x_{i-1})dx_{i-1} + RV(x_0)\tau^{i-1}.$$

By the application of Jensen's inequality we get $\left(\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})|X_{i-1}\right]\right)^{2+\delta} \le \mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})^{2+\delta}|X_{i-1}\right] < V(X_{i-1})$. Since $V(\cdot) \ge 1$, it follows from the previous expression that $\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})|X_{i-1}\right] < V(X_{i-1})^{1/(2+\delta)} \le V(X_{i-1})$. Thus, setting $f(x) = \mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})|X_{i-1} = x\right]$, we obtain

$$\begin{aligned}\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})\right] &< \int \left[\mathrm{E}\left[M_k^{(1)}(X_i, X_{i-1})|X_{i-1}\right]q_{\theta_0}(x_i)dx_{i-1} + RV(x_0)\tau^{i-1}\right]q^{(0)}(x_0)dx_0 \\ &= \mathrm{E}[M_k^{(1)}(X_1, X_0)] + \tau^{i-1}\int RV(x_0)q^{(0)}(x_0)dx_0.\end{aligned}$$

Summing from $i = 1$ to n, we get

$$\begin{aligned}\sum_{i=1}^{n}\mathrm{E}\left[M_k^{(1)}(x_i, x_{i-1})\right] &< n\mathrm{E}[M_k^{(1)}(X_1, X_0)] + \sum_{i=1}^{n}\tau^{i-1}\int RV(x_0)q^{(0)}(x_0)dx_0 \\ &= n\mathrm{E}[M_k^{(1)}(X_1, X_0)] + \frac{1-\tau^n}{1-\tau}\int RV(x_0)q^{(0)}(x_0)dx_0.\end{aligned}$$

This gives us the following bound on $\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta)$:

$$\begin{aligned}\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \le \sum_{k=1}^{m}&\left[n\mathrm{E}[M_k^{(1)}(X_1, X_0)] + \frac{1-\tau^n}{1-\tau}\int RV(x_0)D(x_0)dx_0\right] \\ &\times \int |f_k^{(1)}(\theta, \theta_0)|\rho_n(d\theta).\end{aligned}$$

By Assumption 1, $\int |f_k^{(1)}(\theta, \theta_0)|\rho_n(d\theta) < \frac{C}{\sqrt{n}}$. Hence, we can rewrite the previous expression as

$$\begin{aligned}\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) &\le \sum_{k=1}^{m}\left[n\mathrm{E}[M_k^{(1)}(X_1, X_0)] + \frac{1-\tau^n}{1-\tau}\int RV(x_1)D(x_1)dx_1\right]\frac{C}{\sqrt{n}} \\ &= nm\left[\mathrm{E}[M_k^{(1)}(X_1, X_0)] + \frac{1-\tau^n}{n(1-\tau)}\int RV(x_0)D(x_0)dx_0\right]\frac{C}{\sqrt{n}}.\end{aligned}$$

Since, $\tau < 1, 0 < 1 - \tau^n < 1$, and we rewrite the previous equation as,

$$\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \le nm\left[\mathrm{E}[M_k^{(1)}(X_1, X_0)] + \frac{1}{n(1-\tau)}\int RV(x_0)D(x_0)dx_0\right]\frac{C}{\sqrt{n}}.$$

Hence, there exists an $\epsilon_n^{(1)} = \mathrm{O}(\frac{1}{\sqrt{n}})$ such that $\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \le n\epsilon_n^{(1)}$.

Part 2: Verifying condition (ii) of Corollary 1: Similar to as in the proof of Theorem 3, we upper bound $\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta)$ by

$$\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) \le \sum_{i,j=1}^{n}\left(\frac{4}{n} + 2n^{\delta/2}\left(\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta) + \int C^{(j)}_{\theta_0,\theta}\rho_n(d\theta)\right.\right. \tag{A41}$$

$$\left.\left. + \int \sqrt{C^{(i)}_{\theta_0,\theta}C^{(j)}_{\theta_0,\theta}}\rho_n(d\theta)\right)\right)\left(\alpha_{|i-j|-1}^{\delta/(2+\delta)}\right) \tag{A42}$$

$$+ \sum_{i=1}^{n}\left(\frac{4}{n} + 2n^{\delta/2}\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta)\right)\left(\alpha_{i-1}^{\delta/(2+\delta)}\right),$$

where $C_{\theta_0,\theta}$ is upper bounded as

$$C^{(i)}_{\theta_0,\theta} \le \sum_{k=1}^{m} m^{1+\delta}\mathrm{E}\left[M_k^{(1)}(X_i,X_{i-1})\right]^{2+\delta}|f_k^{(1)}(\theta,\theta_0)|^{2+\delta}.$$

Since $\mathrm{E}\left[M_k^{(1)}(X_i,X_{i-1})^{2+\delta}|X_{i-1}\right] < V(X_{i-1})$, by a similar application of V-geometric ergodicity, we can say that, $\exists\ 0<\tau<1$, such that

$$\mathrm{E}\left[M_k^{(1)}(X_i,X_{i-1})\right]^{2+\delta} \le n\mathrm{E}[M_k^{(1)}(X_1,X_0)]^{2+\delta} + \tau^{i-1}\int RV(x_0)D(x_0)dx_0,$$

which, by the fact that $\tau^{i-1}<\tau$, gives us,

$$\mathrm{E}\left[M_k^{(1)}(X_i,X_{i-1})\right]^{2+\delta} \le \mathrm{E}[M_k^{(1)}(X_1,X_0)]^{2+\delta} + \tau\int RV(x_0)D(x_0)dx_0.$$

By Assumption 1, we know that, $\int |f_k^{(1)}(\theta,\theta_0)|^{2+\delta}\rho_n(d\theta) < \frac{C}{n}$. Hence, for a large constant L, $\int C^{(i)}_{\theta_0,\theta}\rho_n(d\theta) \le \frac{L}{n}$. We also see that since the chain is geometrically ergodic, by the application of Equation (A4), $\sum_{k\ge1}\alpha_k^{\delta/(2+\delta)} < +\infty$. The rest of the proof follows similarly as in the proof of Theorem 3, and we obtain an $\epsilon_n^{(2)} = \mathrm{O}(\frac{n^{\delta/2}}{n})$, such that,

$$\int \mathrm{Var}[r_n(\theta,\theta_0)]\rho_n(d\theta) < n\epsilon_n^{(2)}.$$

Since, $\mathcal{K}(\rho_n,\pi) \le \sqrt{n}C$, similar arguments as in the proof of Theorem 2 holds. The theorem is thus proved. □

Appendix B.3.6. Proof of Proposition 10

For the purpose of the proof, we choose ρ_n's with scaled Beta distribution with parameters $a_n = n\frac{1+\theta_0}{2}$ and $b_n = n\frac{1-\theta_0}{2}$. Since, ρ_n is a scaled Beta distribution with the scaling factors $m=2$ and $c=-1$, the pdf of ρ_n is given by

$$\rho_n(\theta) = \frac{1}{2\mathrm{Beta}(a_n,b_n)}\left(\frac{1+\theta}{2}\right)^{a_n}\left(\frac{1-\theta}{2}\right)^{b_n}$$

Since this is a scaled distribution, $E_{\rho_n}[\theta] = 2\frac{a_n}{a_n+b_n} - 1 = \theta_0$ and there exists a constant $\sigma>0$, $\mathrm{Var}_{\rho_n}[\theta] = \frac{\sigma^2}{n}$. We now analyse the log-ratio of the transition probabilities for the Markov chain,

$$\log p_{\theta_0}(X_n|X_{n-1}) - \log p_\theta(X_n|X_{n-1}) = 2X_nX_{n-1}(\theta-\theta_0) + X_{n-1}^2(\theta_0^2-\theta^2).$$

Observe that in this setting, $M_1^{(1)}(X_n, X_{n-1}) = |X_n X_{n-1}|$ and $M_2^{(1)}(X_n, X_{n-1}) = X_n^2$. Next, using the fact that

$$E[|X_n|^{2+\delta}|X_{n-1}] = E[|X_n - \theta_0 X_{n-1} + \theta_0 X_{n-1}|^{2+\delta}|X_{n-1}],$$

and by an application of triangle inequality, we obtain

$$\begin{aligned}E[|X_n|^{2+\delta}|X_{n-1}] &\le E\Big[(|X_n - \theta_0 X_{n-1}| + |\theta_0 X_{n-1}|)^{2+\delta}|X_{n-1}\Big]\\ &= E\left[\left(2\frac{|X_n - \theta_0 X_{n-1}| + |\theta_0 X_{n-1}|}{2}\right)^{2+\delta}|X_{n-1}\right]\\ &= E\left[2^{2+\delta}\left(\frac{|X_n - \theta_0 X_{n-1}| + |\theta_0 X_{n-1}|}{2}\right)^{2+\delta}|X_{n-1}\right].\end{aligned}$$

Now by using Jensen's inequality we get,

$$\begin{aligned}E[|X_n|^{2+\delta}|X_{n-1}] &\le E\left[2^{2+\delta}\left(\frac{|X_n - \theta_0 X_{n-1}|^{2+\delta} + |\theta_0 X_{n-1}|^{2+\delta}}{2}\right)|X_{n-1}\right]\\ &= 2^{1+\delta}E\Big[|X_n - \theta_0 X_{n-1}|^{2+\delta}|X_{n-1}\Big] + 2^{1+\delta}|\theta_0 X_{n-1}|.\end{aligned}$$

We know if $Y \sim N(\mu, \sigma^2)$, then $E|Y-\mu|^p = \sigma^p \frac{2^{\frac{p}{2}}\Gamma(\frac{p+1}{2})}{\sqrt{\pi}}$. Consequently,

$$E[|X_n|^{2+\delta}|X_{n-1}] \le 2^{1+\delta}\left[\frac{2^{\frac{2+\delta}{2}}\Gamma(\frac{3+\delta}{2})}{\sqrt{\pi}}\right] + 2^{1+\delta}|\theta_0 X_{n-1}|^{2+\delta}. \tag{A43}$$

It follows that,

$$\begin{aligned}E[M_1^{(1)}(X_n, X_{n-1})^{2+\delta}|X_{n-1}] &\le 2^{1+\delta}\left[\frac{2^{\frac{2+\delta}{2}}\Gamma(\frac{3+\delta}{2})}{\sqrt{\pi}}\right]|X_{n-1}|^{2+\delta} + 2^{1+\delta}|\theta_0|^{2+\delta}|X_{n-1}|^{4+2\delta}\\ &\le \left(2^{1+\delta}\left[\frac{2^{\frac{2+\delta}{2}}\Gamma(\frac{3+\delta}{2})}{\sqrt{\pi}}\right] + 2^{1+\delta}|\theta_0|^{2+\delta}\right)(|X_{n-1}|^{4+2\delta} + 1).\end{aligned}$$

Since $\theta_0 < 1$, we can say,

$$E[M_1^{(1)}(X_n, X_{n-1})^{2+\delta}|X_{n-1}] \le \left(2^{1+\delta}\left[\frac{2^{\frac{2+\delta}{2}}\Gamma(\frac{3+\delta}{2})}{\sqrt{\pi}}\right] + 2^{1+\delta}\right)(|X_{n-1}|^{4+2\delta} + 1).$$

Define a constant $C_\delta := \left(2^{1+\delta}\left[\frac{2^{\frac{2+\delta}{2}}\Gamma(\frac{3+\delta}{2})}{\sqrt{\pi}}\right] + 2^{1+\delta}\right)$. The above term then becomes,

$$E[M_1^{(1)}(X_n, X_{n-1})^{2+\delta}|X_{n-1}] \le C_\delta(|X_{n-1}|^{4+2\delta} + 1).$$

Next we analyse the term $M_2^{(1)}(X_n, X_{n-1})$.

$$\begin{aligned}E\Big[M_2^{(1)}(X_n, X_{n-1})^{2+\delta}|X_{n-1}\Big] &= E[X_{n-1}^{4+2\delta}|X_{n-1}]\\ &= X_{n-1}^{4+2\delta}\\ &\le C_\delta(X_{n-1}^{4+2\delta} + 1).\end{aligned}$$

Then, defining $V(x) := C_\delta(x^{4+2\delta}+1)$ it follows that,

$$\mathrm{E}[V(X_n)|X_{n-1}] = \mathrm{E}\Big[C_\delta(X_n^{4+2\delta}+1)|X_{n-1}\Big].$$

Using a technique similar to Equation (A43) we get,

$$\mathrm{E}\Big[C_\delta(X_n^{4+2\delta}+1)|X_{n-1}\Big] \leq \left[C_\delta(2^{3+2\delta}\left[\frac{2^{\frac{4+2\delta}{2}}\Gamma(\frac{5+2\delta}{2})}{\sqrt{\pi}}\right] + 2^{3+2\delta}|\theta_0 X_{n-1}|^{4+2\delta}+1)\right].$$

Define another constant $C'_\delta := C_\delta\left(2^{3+2\delta}\left[\frac{2^{\frac{4+2\delta}{2}}\Gamma(\frac{5+2\delta}{2})}{\sqrt{\pi}}\right] - 2^{3+2\delta}|\theta_0|^{4+2\delta}+1\right)$. Since $\delta > 0$, $\frac{2^{\frac{4+2\delta}{2}}\Gamma(\frac{5+2\delta}{2})}{\sqrt{\pi}} > 1$. Furthermore, since $|\theta_0| < 1$, so is $|\theta_0|^{4+2\delta}$. Hence,

$$2^{3+2\delta}\left[\frac{2^{\frac{4+2\delta}{2}}\Gamma(\frac{5+2\delta}{2})}{\sqrt{\pi}}\right] - 2^{3+2\delta}|\theta_0|^{4+2\delta} > 0.$$

Hence, we have shown that,

$$\mathrm{E}[V(X_n)|X_{n-1}] \leq (2^{3+2\delta}|\theta_0|^{4+2\delta})C_\delta(X_{n-1}^{4+2\delta}+1) + C'_\delta.$$

Since $|\theta_0| < 2^{\frac{1}{4+2\delta}-1}$, $2^{3+2\delta}|\theta_0|^{4+2\delta} < 1$, and we can express the above equation as,

$$\mathrm{E}[V(X_n)|X_{n-1}] \leq V(X_{n-1}) + C'_\delta.$$

Define the set $C(m) := \{x : |x|^{4+2\delta}+1 \leq m\}$. From Proposition 11.4.2, [20], for a large enough m, $C(m)$ forms a petite set. Thus, we have proved that $V(x)$ as defined in this example satisfies Assumption A.1.1, and $\{X_n\}$ is V-geometrically ergodic. The $f_j^{(1)}$'s corresponding to Assumption 1 are given by $f_1^{(1)}(\theta,\theta_0) = (\theta-\theta_0)$ and $f_2^{(1)}(\theta,\theta_0) = (\theta_0^2-\theta^2)$. Therefore, it follows that,

$$\begin{aligned}\partial_\theta f_1^{(1)} &= 1,\\ \partial_\theta f_2^{(1)} &= -2\theta \text{ and}\\ -2 < -2\theta &< 2.\end{aligned}$$

Since $f_1^{(1)}(\theta_0,\theta_0) = f_2^{(1)}(\theta_0,\theta_0) = 0$, We just showed that they also have bounded partial derivatives. We also know that $|\theta| < 1$. Hence, by Proposition 4 $f_j^{(1)}$'s satisfy the conditions of Assumption 1.

The invariant distribution for the simple linear model Markov-chain under parameter θ is given by a gaussian distribution with mean 0 and variance $\frac{1}{1-\theta^2}$. In other words,

$$q_\theta(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{1-\theta^2}{2}x^2}.$$

Analyzing the log likelihood yields,

$$\begin{aligned}\log q_0(x) - \log q_\theta(x) &= -\frac{x^2}{2}(1-\theta_0^2) + \frac{x^2}{2}(1-\theta^2)\\ &= \frac{x^2}{2}(\theta_0^2-\theta^2).\end{aligned}$$

Let $f_1^{(2)}(\theta_0, \theta_0) = (\theta_0^2 - \theta^2)$ and $f_1^{(2)}(\theta_0, \theta_0) = 0$. Since $f_1^{(2)}(\theta_0, \theta_0) = f_2^{(1)}(\theta_0, \theta_0)$, by following arguments similar as before, can conclude that $f_1^{(2)}(\theta_0, \theta_0)$ also satisfies the requirements of Assumption 1. Let $M_1^{(2)}(x) = \frac{x^2}{2}$ and define $M_2^{(2)}(x) := 1$. Let $X_0 \sim q_1^{(0)}$. As long as $\int x^{4+2\delta} q_1^{(0)}(x)dx < \infty$, we satisfy all the conditions required for Theorem 5. Finally we need to verify the condition that $\mathcal{K}(\rho_n, \pi) < C\sqrt{n}$ for some constant $C > 0$. The KL-divergence $\int \log\left(\frac{\rho_n(\theta)}{\pi(\theta)}\right)\rho_n(d\theta)$ becomes,

$$\mathcal{K}(\rho_n, \pi) = \int_{-1}^{1} \log\left(\frac{1}{2\text{Beta}(a_n, b_n)}\left(\frac{1+\theta}{2}\right)^{a_n}\left(\frac{1-\theta}{2}\right)^{b_n}\right) \times \frac{1}{2\text{Beta}(a_n, b_n)}\left(\frac{1+\theta}{2}\right)^{a_n}\left(\frac{1-\theta}{2}\right)^{b_n} d\theta.$$

Substituting, $y = \frac{1+\theta}{2}$, we get,

$$\begin{aligned}\mathcal{K}(\rho_n, \pi) &= \int_0^1 \log\left(\frac{1}{2\text{Beta}(a_n, b_n)}(y)^{a_n}(1-y)^{b_n}\right)\frac{1}{2\text{Beta}(a_n, b_n)}(y)^{a_n}(1-y)^{b_n}dy \\ &= \int_0^1 \log\left(\frac{1}{2}\right)\frac{1}{\text{Beta}(a_n, b_n)}(y)^{a_n}(1-y)^{b_n}dy \\ &\quad + \int_0^1 \log\left(\frac{1}{\text{Beta}(a_n, b_n)}(y)^{a_n}(1-y)^{b_n}\right)\frac{1}{\text{Beta}(a_n, b_n)}(y)^{a_n}(1-y)^{b_n}.\end{aligned}$$

The first term integrates up to $\log(1/2)$. The second term is the KL-divergence between a Uniform and Beta distribution with parameters $a_n = n\frac{1+\theta_0}{2}$ and $b_n = n(1 - \frac{1+\theta_0}{2})$ and support $[0, 1]$. Following Lemma A.2.1 it follows that $\mathcal{K}(\rho_n, \pi)$ is upper bounded by,

$$\mathcal{K}(\rho_n, \pi) < \log(1/2) + C_1 + \frac{1}{2}\log(n) < C\sqrt{n},$$

for some large constant C. This completes the proof. □

Appendix B.4. Proofs for Misspecified Models

Proof of Theorem 6

As in the proof of Theorem 1, following Equation (A13), we note that,

$$\begin{aligned}\int D_{\alpha^{re}}(P_\theta^{(n)}, P_{\theta_0}^{(n)})\tilde{\pi}_{n,\alpha^{re}|X^n}(d\theta) \le& \frac{\alpha^{re}}{1-\alpha^{re}}\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \\ &+ \frac{\alpha^{re}}{1-\alpha^{re}}\sqrt{\frac{\text{Var}[\int r_n(\theta,\theta_0)\rho_n(d\theta)]}{\eta}} + \frac{\mathcal{K}(\rho_n,\pi) - \log(\epsilon)}{1-\alpha^{re}}.\end{aligned} \tag{A44}$$

Following from Equations (23) and (26), we get that,

$$\int \mathcal{K}(P_{\theta_0}^{(n)}, P_\theta^{(n)})\rho_n(d\theta) \le \text{E}[r_n(\theta_0, \theta_n^*)] + n\epsilon_n,$$

and

$$\int \text{Var}[r_n(\theta, \theta_0)]\rho_n(d\theta) \le 2n\epsilon_n + 2\text{Var}[r_n(\theta_n^*, \theta_0)].$$

Plugging these into Equation (A44), we are done. □

References

1. Wainwright, M.J.; Jordan, M.I. Introduction to Variational Methods for Graphical Models. *Found. Trends Mach. Learn.* **2008**, *1*, 1–103. [CrossRef]
2. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: Berlin, Germany, 2006.
3. Ormerod, J.T.; Wand, M.P. Explaining variational approximations. *Am. Stat.* **2010**, *64*, 140–153. [CrossRef]
4. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877. [CrossRef]
5. Jaiswal, P.; Rao, V.; Honnappa, H. Asymptotic Consistency of α-Rényi-Approximate Posteriors. *J. Mach. Learn. Res.* **2020**, *21*, 1–42.
6. Li, Y.; Turner, R.E. Rényi divergence variational inference. In Proceedings of the 30th Annual Conference on Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; Volume 29, pp. 1073–1081.
7. Dieng, A.B.; Tran, D.; Ranganath, R.; Paisley, J.; Blei, D. Variational Inference via χ Upper Bound Minimization. In Proceedings of the 31th Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
8. Wang, Y.; Blei, D.M. Frequentist consistency of variational Bayes. *J. Am. Stat. Assoc.* **2019**, *114*, 1147–1161. [CrossRef]
9. Zhang, F.; Gao, C. Convergence rates of variational posterior distributions. *Ann. Stat.* **2020**, *48*, 2180–2207. [CrossRef]
10. Ghosal, S.; Ghosh, J.K.; Van Der Vaart, A.W. Convergence rates of posterior distributions. *Ann. Stat.* **2000**, *28*, 500–531. [CrossRef]
11. Shen, X.; Wasserman, L. Rates of convergence of posterior distributions. *Ann. Stat.* **2001**, *29*, 687–714.
12. Rousseau, J. On the frequentist properties of Bayesian nonparametric methods. *Annu. Rev. Stat. Its Appl.* **2016**, *3*, 211–231. [CrossRef]
13. Ghosal, S.; Van Der Vaart, A.W. Entropies and rates of convergence for maximum likelihood and Bayes estimation for mixtures of Normal densities. *Ann. Stat.* **2001**, 1233–1263.
14. Bhattacharya, A.; Pati, D.; Yang, Y. Bayesian fractional posteriors. *Ann. Stat.* **2019**, *47*, 39–66. [CrossRef]
15. Alquier, P.; Ridgway, J. Concentration of tempered posteriors and of their variational approximations. *Ann. Stat.* **2020**, *48*, 1475–1497. [CrossRef]
16. Yang, Y.; Pati, D.; Bhattacharya, A. α-variational inference with statistical guarantees. *Ann. Stat.* **2020**, *48*, 886–905. [CrossRef]
17. Jaiswal, P.; Honnappa, H.; Rao, V.A. Risk-sensitive variational Bayes: Formulations and bounds. *arXiv* **2019**, arXiv:1903.05220.
18. Bradley, R.C. Basic Properties of Strong Mixing Conditions. A Survey and Some Open Questions. *Probab. Surv.* **2005**, *2*, 107–144. [CrossRef]
19. Ibragimov, I.A. Some limit theorems for stationary processes. *Theory Probab Appl.* **1962**, *7*, 349–382. [CrossRef]
20. Meyn, S.P.; Tweedie, R.L. *Markov Chains and Stochastic Stability*; Springer: Berlin, Germany, 2012.
21. Hajek, B. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Adv. Appl. Probab.* **1982**, 502–525. [CrossRef]
22. Birgé, L. Robust testing for independent non identically distributed variables and Markov chains. In *Specifying Statistical Models*; Springer: Berlin, Germany, 1983; pp. 134–162.
23. Ryabko, D. Testing statistical hypotheses about ergodic processes. In Proceedings of the IEEE Region 8 International Conference on Computational Technologies in Electrical and Electronics Engineering, Novosibirsk, Russia, 21–25 July 2008.
24. Lacoste-Julien, S.; Huszár, F.; Ghahramani, Z. Approximate inference for the loss-calibrated Bayesian. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Ft. Lauderdale, FL, USA, 11–13 April 2011.
25. Jaiswal, P.; Honnappa, H.; Rao, V.A. Asymptotic consistency of loss-calibrated variational Bayes. *Stat* **2020**, *9*, e258. [CrossRef]
26. Jones, G.L. On the Markov chain central limit theorem. *Probab. Survey.* **2004**, *1*, 299–320. [CrossRef]
27. Alzer, H. On some inequalities for the gamma and psi functions. *Math. Comput.* **1997**, *66*, 373–389. [CrossRef]
28. Donsker, M.D.; Varadhan, S.S. Asymptotic evaluation of certain Markov process expectations for large time, I.*Commun. Pure Appl. Math.* **1975**, *28*, 1–47. [CrossRef]

entropy

MDPI

Article

Approximate Bayesian Computation for Discrete Spaces

Ilze A. Auzina † and Jakub M. Tomczak *,†

Department of Computer Science, Faculty of Science, Vrije Universiteit Amsterdam, De Boelelaan 1111, 1081 HV Amsterdam, The Netherlands; ilze.amanda.auzina@gmail.com
* Correspondence: jmk.tomczak@gmail.com
† These authors contributed equally to this work.

Abstract: Many real-life processes are black-box problems, i.e., the internal workings are inaccessible or a closed-form mathematical expression of the likelihood function cannot be defined. For continuous random variables, likelihood-free inference problems can be solved via Approximate Bayesian Computation (ABC). However, an optimal alternative for discrete random variables is yet to be formulated. Here, we aim to fill this research gap. We propose an adjusted population-based MCMC ABC method by re-defining the standard ABC parameters to discrete ones and by introducing a novel Markov kernel that is inspired by differential evolution. We first assess the proposed Markov kernel on a likelihood-based inference problem, namely discovering the underlying diseases based on a QMR-DTnetwork and, subsequently, the entire method on three likelihood-free inference problems: (i) the QMR-DT network with the unknown likelihood function, (ii) the learning binary neural network, and (iii) neural architecture search. The obtained results indicate the high potential of the proposed framework and the superiority of the new Markov kernel.

Keywords: Approximate Bayesian Computation; differential evolution; MCMC; Markov kernels; discrete state space

Citation: Auzina, I.A.; Tomczak, J.M. Approximate Bayesian Computation for Discrete Spaces. *Entropy* **2021**, *23*, 312. https://doi.org/10.3390/e23030312

Academic Editor: Pierre Alquier

Received: 18 February 2021
Accepted: 2 March 2021
Published: 6 March 2021

1. Introduction

In various scientific domains, an accurate simulation model can be designed, yet formulating the corresponding likelihood function remains a challenge. In other words, there is a simulator of a process available that, when provided an input, returns an output, but the inner workings of the process are not analytically available [1–5]. Thus far, the existing tools for solving such problems are typically limited to continuous random variables. Consequently, many discrete problems are reparameterized to continuous ones via, for example, the Gumbel-softmax trick [6] rather than being solved directly. In this paper, we aim at providing a solution to this problem by translating the existing likelihood-free inference methods to discrete space applications.

Commonly, likelihood-free inference problems for continuous data are solved via a group of methods known under the term Approximate Bayesian Computation (ABC) [2,7]. The main idea behind ABC methods is to model the posterior distribution by approximating the likelihood as a fraction of accepted simulated data points from the simulator model, by the use of a distance measure δ and a tolerance value ϵ. The first approach, known as the ABC-rejection scheme, has been successfully applied in biology [8,9], and since, then many alternative versions of the algorithm have been introduced, with the three main groups represented by Markov Chain Monte Carlo (MCMC) ABC [10], Sequential Monte Carlo (SMC) ABC [11], and neural network-based ABC [12,13]. In the current paper, we focus on the MCMC-ABC version [14] for discrete data application, as it can be more readily implemented and the computational costs are lower [15]. Thus, the efficiency of our newly proposed likelihood-free inference method will depend on two parts, namely (i) on the design of the proposal distribution for the MCMC algorithm and (ii) the selected hyperparameter values for the ABC algorithm.

Our main focus is on optimal proposal distribution design as there is no "natural" notion of the search direction and scale for discrete data spaces. Hence, the presented solution is inspired by Differential Evolution (DE) [16], which has been shown to be an effective optimization technique for many likelihood-free (or black-box) problems [17,18]. We propose to define a probabilistic DE kernel for discrete random variables that allows us to traverse the search space without specifying any external parameters. We evaluate our approach on four test-beds: (i) we verify our proposal on a benchmark problem of the QMR-DTnetwork presented by [19]; (ii) we modify the first problem and formulate it as a likelihood-free inference problem; (iii) we assess the applicability of our method for high-dimensional data, namely training binary neural networks on MNIST data; (iv) we apply the proposed approach to Neural Architecture Search (NAS) using the benchmark dataset proposed by [20].

The contribution of the present paper is as follows. First, we introduce an alternative version of the MCMC-ABC algorithm, namely a population-based MCMC-ABC method, that is applicable to likelihood-free inference tasks with discrete random variables. Second, we propose a novel Markov kernel for likelihood-based inference methods in a discrete state space. Third, we present the utility of the proposed approach on three binary problems.

2. Likelihood-Free Inference and ABC

Let $x \in \mathcal{X}$ be a vector of parameters or decision variables, where $\mathcal{X} = \mathbb{R}^D$ or $\mathcal{X} = \{0,1\}^D$, and $y \in \mathbb{R}^M$ is a vector of observable variables. Typically, for a given collection of observations of y, $y_{data} = \{y_n\}_{n=1}^N$, we are interested in solving the following optimization problem (we note that the logarithm does not change the optimization problem, but it is typically used in practice):

$$x^* = \arg\max \ln p(y_{data}|x), \tag{1}$$

where $p(y_{data}|x)$ is the likelihood function. Sometimes, it is more advantageous to calculate the posterior:

$$\ln p(x|y_{data}) = \ln p(y_{data}|x) + \ln p(x) - \ln p(y_{data}), \tag{2}$$

where $p(x)$ denotes the prior over x and $p(y_{data})$ is the marginal likelihood. The posterior $p(x|y_{data})$ could be further used in Bayesian inference.

In many practical applications, the likelihood function is unknown, but it is possible to obtain (approximate) samples from $p(y|x)$ through a simulator. Such a problem is referred to as likelihood-free inference [3] or a black-box optimization problem [1]. If the problem is about finding the posterior distribution over x while only a simulator is available, then it is considered as an Approximate Bayesian Computation (ABC) problem, meaning that $p(y_{data}|x)$ is assumed to be given represented as the simulator.

3. Population-Based MCMC

Typically, a likelihood-free inference problem or an ABC problem is solved through sampling. One of the most well-known sampling methods is the Metropolis–Hastings algorithm [21], where the samples are generated from an ergodic Markov chain, and the target density is estimated via Monte Carlo sampling. In order to speed up the computations, it is proposed to run multiple chains in parallel rather than sampling from a single chain. This approach is known as population-based MCMC methods [22]. A population-based MCMC method operates over a joint state space with the following distribution:

$$p(x_1, \ldots, x_C) = \prod_{c \in \mathcal{C}} p_c(x_c) \tag{3}$$

where $\mathcal{C}$ denotes the population of chains and at least one of $p_c(x_c)$ is equivalent to the original distribution we want to sample from (e.g., the posterior distribution $p(x|y_{data})$).

Given a population of chains, a question of interest is what is the best proposal distribution for an efficient sampling convergence. One approach is parallel tempering. It introduces an additional temperature parameter and initializes each chain at a different temperature [23,24]. However, the performance of the algorithm highly depends on an appropriate cooling schedule rather than a smart interaction between the chains. A different approach proposed by [25] relies on a suitable proposal that is able to adapt the shape of the population at a single temperature. We further expand on this idea by formulating population-based proposal distributions that are inspired by evolutionary algorithms.

3.1. Continuous Case

Reference [26] successfully formulated a new proposal called Differential Evolution Markov Chain (DE-MC) that combines the ideas of differential evolution and population-based MCMC. In particular, he redefined the DE-1 equation [16] by adding noise, ε, to it:

$$x_{new} = x_i + \gamma(x_j - x_k) + \varepsilon, \tag{4}$$

where ε is sampled from a Gaussian distribution, $\gamma \in \mathbb{R}_+$. The created proposal automatically implies the invariance of the underlying distribution, as the reversibility condition is satisfied:

- Reversibility is met, because the suggested proposal could be inverted to obtain x_i.

Furthermore, the created Markov chain is ergodic, as the following two conditions are met:

- Aperiodicity is met, because the Markov chain follows a random walk.
- Irreducibility is solved by applying the noise.

Hence, the resulting Markov chain has a unique stationary distribution. The results presented by [26] indicate an advantage of DE-MC over conventional MCMC with respect to the speed of calculations, convergence, and applicability to multimodal distributions, therefore positioning DE as an optimal method for choosing an appropriate scale and orientation of the jumping distribution for a population-based MCMC.

3.2. Discrete Case

In this paper, we focus on binary variables, because categorical variables could always be transformed to a binary representation. Hence, the most straightforward proposal for binary variables is the independent sampler that utilizes the product of Bernoulli:

$$q(x) = \prod_d B(\theta_d), \tag{5}$$

where $B(\theta_d)$ denotes the Bernoulli distribution with a parameter θ_d. However, the above proposal does not utilize the information available across the population; hence, the performance could be improved by allowing the chains to interact. Exactly this possibility we investigate in the following section.

4. Our Approach

4.1. Markov Kernels

We propose to utilize the ideas outlined by [26], but in a discrete space. For this purpose, we need to relate the DE-1 equation to logical operators, as now the vector x is represented by a string of bits, $\mathcal{X} = \{0,1\}^D$, and properly defined noise. Following [19], we propose to use the *xor*operator between two bits b_1 and b_2:

$$b_1 \otimes b_2 = \begin{cases} 1, & b_1 \neq b_2 \\ 0, & b_1 = b_2 \end{cases} \tag{6}$$

instead of the subtraction in (4). Next, we define a difference between two chains x_i and x_j as $\delta_k = x_i \otimes x_j$ and a set of all possible differences between two chains, $\Delta = \{\delta_k : \forall_{x_i,x_j \in \mathcal{C}}\ \delta_k = x_i \otimes x_j\}$ (a similar construction could be done for the continuous case). We can construct a distribution over δ_k as a uniform distribution:

$$q(\delta|\mathcal{C}) = \frac{1}{|\Delta|} \sum_{\delta_k \in \Delta} \mathbb{I}\Big[\delta_k = \delta\Big], \tag{7}$$

where $|\Delta|$ denotes the cardinality of Δ and $\mathbb{I}[\cdot]$ is an indicator function such that $\mathbb{I}[\delta_k = \delta] = 1$ if $\delta_k = \delta$ and zero otherwise. Now, we can formulate a binary equivalence of the DE-1 equation by adding a difference drawn from $q(\delta|\mathcal{C})$:

$$x_{new} = x_i \otimes \delta_k. \tag{8}$$

However, the proposal defined in (8) is not a valid ergodic Markov kernel, as is shown in the following Proposition.

Remark 1. *The proposal defined in (8) fulfills reversibility and aperiodicity, but it does not meet the irreducibility requirement.*

Proof. Reversibility is met, as x_i can be re-obtained by applying the difference to the left side of (8). Aperiodicity is met because the general setup of the Markov chain is kept unchanged (it resembles a random walk). However, the operation in (8) is deterministic; thus, it violates the irreducibility assumption. □

The missing property of (8) could be fixed by including the following mutation (*mut*) operation:

$$x_l = \begin{cases} 1 - x_l & \text{if } p_{flip} \geq u \\ x_l & \text{otherwise} \end{cases} \tag{9}$$

where $p_{flip} \in (0, 1)$ corresponds to an independent probability of flipping a bit and $U(0, 1)$ denotes the uniform distribution. Then, the following proposal could be formulated [19] as in Proposition 1.

Proposition 1. *The proposal defined as a mixture $q_{mut+xor}(x|\mathcal{C}) = \pi q_{mut}(x|\mathcal{C}) + (1 - \pi) q_{xor}$ $(x|\mathcal{C})$, where $\pi \in (0, 1)$, $q_{mut}(x|\mathcal{C})$ is defined by (9) and $q_{xor}(x|\mathcal{C})$ is defined by (8), is a proper Markov kernel.*

Proof. Reversibility and aperiodicity were shown in Proposition 1. The irreducibility is met, because the *mut* proposal assures that there is a positive transition probability across the entire search space. □

However, we notice that there are two potential issues with the mixture proposal *mut+xor*. First, it introduces another hyperparameter, π, that needs to be determined. Second, improperly chosen π could negatively affect the convergence speed, i.e., a fixed value that is either too frequent or scarce would drastically halt the convergence.

In order to overcome these issues, we propose to apply the *mut* operation in (9) directly to δ_k, in a similar manner as the Gaussian noise is added to $\gamma(x_i - x_j)$ in the proposition of [26]. As a result, we obtain the following proposal:

$$x_{new} = x_i \otimes (mut(\delta_k)). \tag{10}$$

Importantly, this proposal fulfills all requirements for an ergodic Markov kernel.

Proposition 2. *The proposal defined in (10) is a valid ergodic Markov kernel.*

Proof. Reversibility and aperiodicity are met in the same manner as shown in Proposition 1. Adding the mutation operation directly to δ_k allows obtaining all possible states in the discrete space; thus, the irreducibility requirement is met. □

We refer to this new Markov kernel for discrete random variables as the discrete differential evolution Markov chain (*dde-mc*).

4.2. Population-MCMC-ABC

Since we formulated a proposal distribution that utilizes a population of chains, we propose to use a population-based MCMC algorithm for the discrete ABC problems. The core of the MCMC-ABC algorithm is to use a proxy of the likelihood-function defined as an ϵ-ball from the observed data, i.e., $\|y - y_{data}\| \le \epsilon$, where $\epsilon > 0$ and $\| \cdot \|$ is a chosen metric. The convergence speed and the acceptance rate highly depend on the value of ϵ [27–29]. In this paper, we consider two approaches to determine the ϵ value: (i) by setting a fixed value and (ii) by sampling $\epsilon \sim Exp(\tau)$ [30]. See the Appendix A for details.

A single step of the population-MCMC-ABC algorithm is presented in Algorithm 1. Notice that in Line 5, we take advantage of the symmetricity of all the proposal. Moreover, in the procedure, we skip an outer loop over all chains for clarity. Without loss of generality, we assume a simulator to be a probabilistic program denoted by $\tilde{p}(y|x)$.

Algorithm 1 Population-MCMC-ABC.

```
1: Given $x \in \{0,1\}^D$
2: $x' \sim q(x|\mathcal{C})$                                   ▷ Either (5), *mut+xor* or *dde-mc*.
3: Simulate $y \sim \tilde{p}(y|x')$.
4: **if** $\|y - y_{data}\| \le \epsilon$ **then**
5:     $\alpha = \min\{1, \frac{p(x')}{p(x)}\}$
6:     $u \sim U(0,1)$
7:     **if** $u \le \alpha$ **then**
8:         $x = x'$
9: **return** $x$
```

5. Experiments

In order to verify our proposed approach, we use four test-beds:

1. QMR-DT network (likelihood-based case): First, we validate the novel proposal, *dde-mc*, on a problem when the likelihood is known.
2. QMR-DT network (likelihood-free case): Second, we verify the performance of the presented proposal by modifying the first test-bed as a likelihood-free problem.
3. Binarized Neural Network Learning: Third, we investigate the performance of the proposed approach on a high-dimensional problem, namely learning binary neural networks.
4. Neural architecture search: Lastly, we consider a problem of Neural Network Architecture Search (NAS).

With each test-bed, we increase the complexity of the problem. Hence, the number of iterations chosen varies per experiment. The code of the methods and all experiments is available at the following link: https://github.com/IlzeAmandaA/ABCdiscrete (accessed on 5 March 2021).

5.1. A Likelihood-Based QMR-DT Network

5.1.1. Implementation Details

The overall setup was designed as described by [19], i.e., we considered a QMR-DT network model. The architecture of the network follows a two-level or bipartite graphical model, where the top level of the graph contains nodes for the diseases and the bottom level contains nodes for the findings [31]. The following density model captures the relations between the diseases (x) and findings (y):

$$p(y_i = 1|x) = 1 - (1 - q_{i0}) \prod_{l} (1 - q_{il})^{x_l} \tag{11}$$

where y_i is an individual bit of string y and q_{i0} is the corresponding leak probability, i.e., the probability that the finding is caused by means other than the diseases included in the

QMR-DT model [31]. q_{il} is the association probability between disease l and finding i, i.e., the probability that the disease l alone could cause the finding i to have a positive outcome. For a complete inference, the prior $p(x)$ is specified. We follow the assumption made by [19] that the diseases are independent:

$$p(x) = \prod_{l} p_l^{x_l}(1 - p_l)^{(1-x_l)} \tag{12}$$

where p_l is the prior probability for disease l.

We compare the performance of the *dde-mc* kernel to the *mut* proposal, the *mut-xor* proposal, the *mut+crx* proposal (see [19] for details), and the independent sampler (*ind-samp*) as in (5) with sampling probability $\theta_d = 0.5$. We expect the DE-inspired proposals to outperform *ind-samp*, and *dde-mc* to perform similarly, if not surpass, *mut+xor*. Out of the possible parameter settings we investigate, the following population sizes $C = \{8, 12, 24, 40, 60\}$, as well as bit-flipping probabilities $p_{flip} = \{0.1, 0.05, 0.01, 0.005\}$. All experiments were run for 10,000 iterations, as in earlier work by [19], it was observed that the performance differences after 10,000 steps were negligible, and initial experiments revealed that in the current work, all proposals approximately converged at this mark. Furthermore, the performance was validated over 80 random problem instances, and the resulting mean and its standard error are reported.

In this experiment, we used the error that is defined as the average Hamming distance between the real values of x and the most probable values found by the population-MCMC with different proposals. The number of diseases was set to $m = 20$, and the number of findings was $n = 80$.

5.1.2. Results and Discussion

DE-inspired proposals, *dde-mc* and *mut+xor*, are superior to kernels stemming from genetic algorithms or random search, i.e., *mut+crx*, *mut*, and *ind-samp* (Figure 1). In particular, *dde-mc* converged the fastest (see the first 4000 evaluations in Figure 1), suggesting that an update via a single operator rather than a mixture is most effective. As expected, *ind-samp* requires many evaluations to obtain a reasonable performance. Even more so, the obtained difference in wall-clock time between *dde-mc* and *ind-samp* was negligible, 148 versus 117 min, respectively, even though the computational complexity of the new method is theoretically higher: given a search space of $\{0, 1\}^D$, the *dde-mc* proposal costs O(D), while the time complexity of *ind-samp* is O(1).

Based on the obtained results, the subsequent experiments were carried out only with *dde-mc*, *mut+xor*, and *ind-samp* as a baseline. *mut+crx* and *mut* were not selected due to to their very slow convergence with high-dimensional problems.

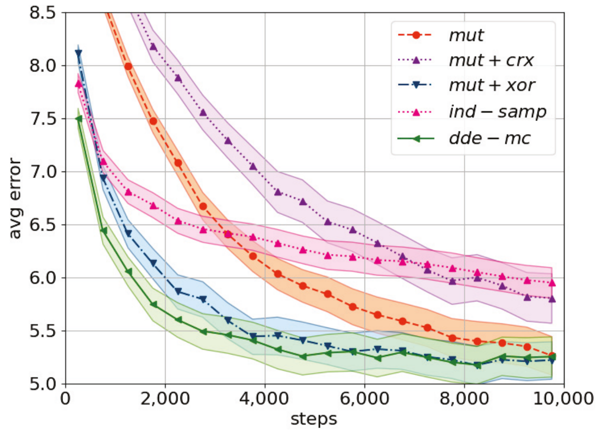


Figure 1. A comparison of the considered proposals using the population average error. The obtained mean and its corresponding standard error (shaded area) across 80 random problem instances are plotted. The following settings were used: $C = 24$, $p_{flip} = 0.01$, $p_{cross} = 0.5$. The corresponding equations for each proposal are as follows: *mut* as in (9), *ind-samp* as in (5), *dde-mc* as in (10), and *mut+xor*, *mut+crx* as in [19].

5.2. A Likelihood-Free QMR-DT Network

5.2.1. Implementation Details

In this test-bed, the QMR-DT network is redefined as a simulator model, i.e., the likelihood is assumed to be intractable. The Hamming distance is selected as the distance metric, but due to its equivocal nature for high-dimensional data, the dimensionality of the problem is reduced. In particular, the number of diseases and observations (i.e., findings) are decreased to 10 and 20, respectively, while the probabilities of the network are sampled from a beta distribution, $Beta(0.15, 0.15)$. The resulting network is more deterministic as the underlying density distributions are more peaked; thus, the stochasticity of the simulator is reduced. Multiple tolerance values are investigated to find the optimal settings, $\epsilon = \{0.5, 0.8, 1., 1.2, 1.5, 2.\}$, respectively. The minimal value is chosen to be 0.5 due to variability across the observed data y_{data}. Additionally, we checked sampling ϵ from the exponential distribution. All experiments were cross-evaluated 80 times, and each experiment was initialized with different underlying parameter settings.

5.2.2. Results and Discussion

First, for the fixed value of ϵ, we notice that *dde-mc* converged faster and to a better (local) optimum than *mut+xor*. However, this effect could be explained by a lower dimensionality of the problem compared to the first experiment. Second, utilizing the exponential distribution had a profound positive effect on the convergence rate of both *dde-mc* and *mut+xor* (Figure 2). This confirmed the expectation that an adjustable ϵ has a better balance between exploration and exploitation. In particular, $\epsilon \sim Exp(2)$ brought the best results with *dde-mc* converging the fastest, followed by *mut+xor* and *ind-samp*. This is in line with the corresponding acceptance rates for the first 10,000 iterations (Table 1), i.e., the use of a smarter proposal allows increasing the acceptance probability, as the search space is investigated more efficiently.

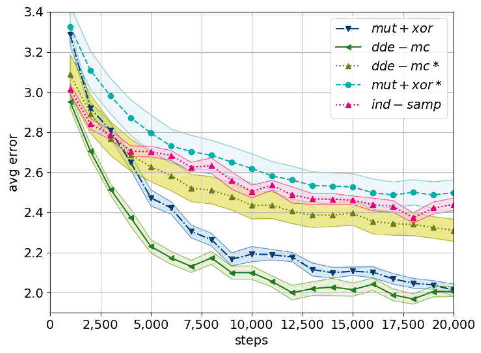


Figure 2. A comparison of the considered proposals using the population error for exponentially adjusted ϵ and the fixed ϵ (indicated by *). The shaded area corresponds to the standard error across 80 random problem instances. The parameter settings are as follows: $C = 24$, $p_{flip} = 0.01$, $\epsilon = 2.0$. The following equations describe the proposal distributions utilized in Algorithm 1: *ind-samp* as in (5), *dde-mc* as in (10), and *mut+xor* as in [19].

Table 1. Percentage of acceptance ratio, α.

Proposal	Mean (std)
dde-mc	24.47 (1.66)
mut+xor	25.81 (1.38)
ind-samp	13.14 (0.33)

Furthermore, the final error obtained by the likelihood-free inference approach is comparable with the results reported for the likelihood-based approach (Figures 1 and 2). This is a positive outcome as any approximation of the likelihood will always be inferior to an exact solution. In particular, the final error obtained by the *dde-mc* proposal is lower; however, this is accounted for by the reduced dimensionality of the problem. Interestingly, despite approximating the likelihood, the computational time only increased twice, while the best performing chain was already identified after 4000 evaluations (Figure 3).

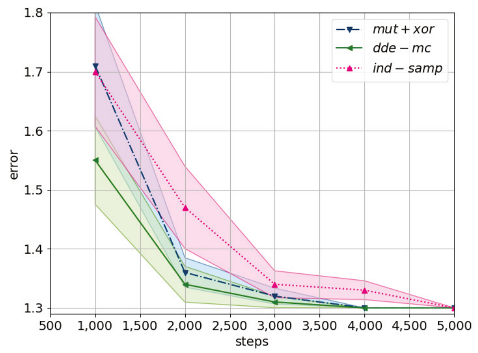


Figure 3. A comparison of the considered proposal using the minimum average error (i.e., the lowest error found by the population) on QMR-DTwith adjusted ϵ. The shaded area corresponds to the standard error across 80 random problem instances. The parameter settings are as follows: $C = 24$, $p_{flip} = 0.01$, $\epsilon = 2.0$. The corresponding equations represent the proposal distributions: *ind-samp* in (5), *dde-mc* in (10), and *mut+xor* as in [19].

Lastly, the obtained results were validated by comparing the true approximate posterior distribution to the approximate posterior distribution of the last five generations of the multi-chain ensemble. In Figure 4, the negative logarithm of the posterior distribution is plotted. The main conclusion is that all proposals converge towards the approximate posterior, yet the obtained distributions are more dispersed.

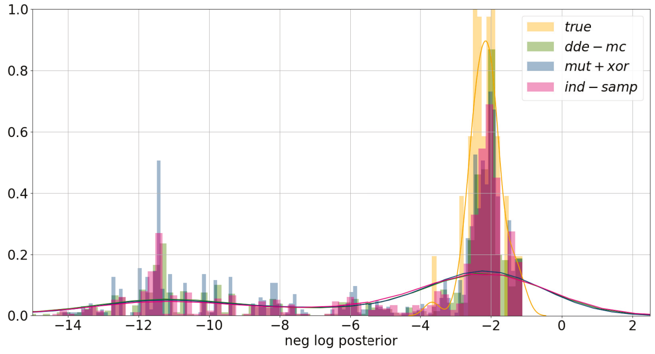


Figure 4. Approximate posterior distribution. The approximate posterior distribution, $p(x|y_{data}) \approx p(y_{data}|x) * p(x)$, was computed using the last population of each chain for all 80 random problem instances. To reconstruct the true posterior, the true underlying parameters were used.

5.3. Binary Neural Networks

5.3.1. Implementation Details

In the following experiment, we aimed at evaluating our approach on a high-dimensional optimization problem. We trained a Binary Neural Network (BinNN) with a single fully-connected hidden layer on the image dataset of ten handwritten digits (MNIST [32]). We used 20 hidden units, and the image was resized from 28px × 28px to 14px × 14px. Furthermore, the image was converted to polar values of +1 or −1, while the network was created in accordance to [33], where the weights and activations of the network were binary, meaning that they were constrained to +1 or −1 as well. We simplified the problem to a binary classification by only selecting two digits from the dataset. As a result, the total number of weights equaled 3940. We used the *tanh*activation function for the hidden units and the sigmoid activation function for the outputs. Consequently, the distance metric becomes the classification error:

$$\|y_{data} - y\| = 1 - \frac{1}{N} \sum_{n=1}^{N} \mathbb{I}[y_n = y_n(x)], \tag{13}$$

where N denotes the number of images, $\mathbb{I}[\cdot]$ is an indicator function, y_n is the true label for the n-th image, and $y_n(x)$ is the n-th label predicted by the binary neural net with weights x.

For the Metropolis acceptance rule, we define a Boltzmann distribution over the prior distribution of the weights x inspired by the work of [34]:

$$p(x) = \frac{h(x)}{\sum_i h(x_i)}, \tag{14}$$

where $h(x) = exp(-\frac{1}{D} \sum_{i=1}^{D} x_i)$ and D denotes the dimensionality of x. As a result, the prior distribution acts as a regularization term as it favors parameter settings with fewer active weights. The distribution is independent of the data y thus, the partition function $\sum_i h(x_i)$ cancels out in the computation of the Metropolis ratio:

$$\alpha = \frac{p(x')}{p(x)} = \frac{h(x')}{h(x)}. \tag{15}$$

The original dataset consists of 60,000 training examples and 10,000 test examples. For our experiment, we selected the digits 0 and 1; hence, the dataset size was reduced to 12,665 training and 2115 test examples. Different tolerance values were investigated to obtain the best convergence, ranging from 0.03 to 0.2, and each experiment was run for at least 200,000 iterations. All experiments were cross-evaluated five times. Lastly, we evaluated the performance by computing both the minimum test error obtained by the final population, as well as the test error obtained by using a Bayesian approach, i.e., we computed the true predictive distribution via majority voting by utilizing an ensemble of models. In particular, we selected the five last updated populations, resulting in $5 \times 24 \times 5 = 600$ models per run, and we repeated this with different seeds 10 times.

Because the classification error function in (13) is non-differentiable, the problem could be treated as a black-box objective. However, we want to emphasize that we do not propose our method as an alternative to gradient-based learning methods. In principle, any gradient-based approach will be superior to a derivative-free method, as what a derivative-free method tries to achieve is to implicitly approximate the gradient [1]. Therefore, the purpose of the presented experiment is not to showcase a state-of-the-art classification accuracy, as that already has been done with gradient-based approaches for BinNN [33], but rather showcase the population-MCMC-ABC applicability to a high-dimensional optimization problem.

5.3.2. Results and Discussion

For the high-dimensional data problem, the *mut+xor* proposal converged the fastest towards the optimal solution in the search space (Figure 5). In particular, the minimum error on the training set was already found after 100,000 iterations, and a tolerance threshold of 0.05 had the best trade-off between the Markov chain error and the likelihood approximation bias.

With respect to the error within the entire population (Figure 6), *dde-mc* converged the fastest, although its performance was on par with *ind-samp*. In general, the drop in performance with respect to the convergence rate of the entire population could be explained by the high dimensionality of the problem, i.e., the higher the dimensionality, the more time is needed for every chain to explore the search space. This observation was confirmed by computing the test error via utilizing all the population members in a majority-voting setting. In particular, the test error based on the ensemble approach was alike across all three proposals, yet the minimum error (i.e., for a single best model) was better for *dde-mc* and *mut+xor* compared to *ind-samp* (Table 2). This result suggests that there seems to be an added advantage of utilizing DE-inspired proposals in faster convergence towards a local optimal solution.

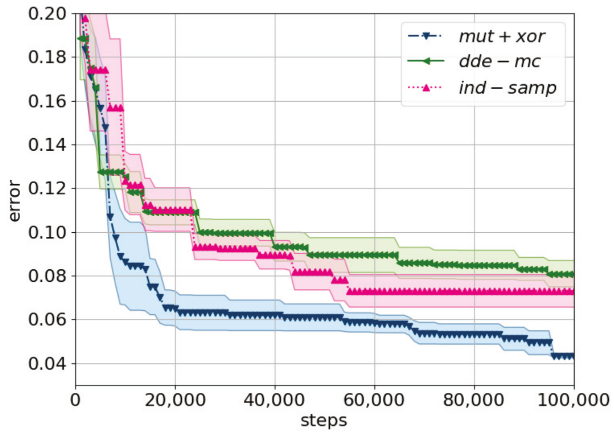


Figure 5. A comparison of the considered proposals using the minimum training error on MNIST. The mean minimum error across five cross-evaluations is plotted with the shaded area corresponding to the standard error. Tolerance is set to $\epsilon = Exp(0.05)$, with the prior and the Metropolis ratio as described in (14) and (15). The following equations describe the proposals: *ind-samp* in (5), *dde-mc* in (10), and *mut+xor* as in [19].

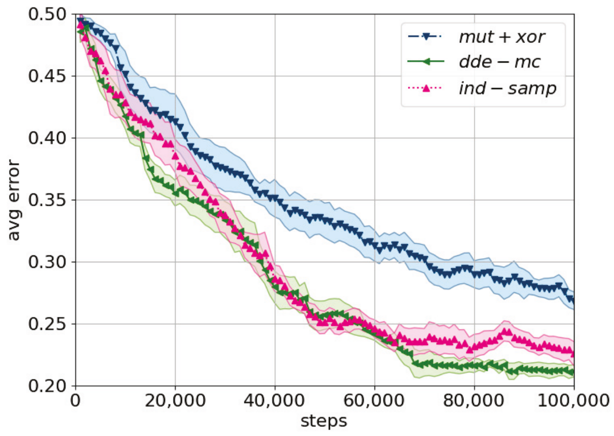


Figure 6. A comparison of the considered proposals using the avg. training error on MNIST. The mean population error across five cross-evaluations is plotted with the shaded area corresponding to the standard error. Tolerance is set to $\epsilon = Exp(0.05)$, with the prior and the Metropolis ratio as described in (14) and (15). The following equations describe the proposals: *ind-samp* in (5), *dde-mc* in (10), and *mut+xor* as in [19].

Table 2. Test error of BinNN on MNIST.

Proposal	**Error (ste)**	
	Single Best	***Ensemble***
dde-mc	0.045 (0.002)	0.013 (0.001)
mut+xor	0.046 (0.002)	0.014 (0.002)
ind-samp	0.051 (0.002)	0.012 (0.001)

5.4. Neural Architecture Search

5.4.1. Implementation Details

In the last experiment, we aimed at investigating whether the proposed approach is applicable for efficient neural architecture search. In particular, we made use of the NAS-Bench-101 dataset, the first public architecture dataset for NAS research [20]. The dataset is represented as a table, which maps neural architectures to their training and evaluations metrics, and as such, it represents an efficient solution for querying different neural topologies. Each topology is captured by a directed acyclic graph represented by an adjacency matrix. The number of vertices was set to seven, while the maximum amount of edges was nine. Apart from these restrictions, we limited the search space by constricting the possible operations for each vertex. Consequently, the simulator was captured by querying the dataset, while the distance metric now was simply the validation error. The prior distribution was kept the same as for the previous experiment.

Every experiment was run for at least 120,000 iterations, with five cross-evaluations. To find the optimal performance, the following tolerance threshold values were investigated $\epsilon = \{0.01, 0.1, 0.2, 0.3\}$. As we are approaching the problem as an optimization task, the aim is to find a chain with the lowest test error, rather than covering the entire distribution. Therefore, to evaluate the performance, we plot the minimum error obtained through the training process, as well as the lowest test error obtained by the final population.

5.4.2. Results and Discussion

dde-mc identified the best solution the fastest with ϵ set to $\epsilon \sim Exp(0.2)$ (Figure 7). The corresponding test error is reported in Table 3, and it follows the same pattern, namely *dde-mc* is superior. Interestingly, here, the *mut+xor* proposal performed almost on par with the *ind-samp* proposal for the first 10,000 iterations, and then, both methods converged to almost the same result. Our proposed Markov kernel obtained again the best result, and also it was the fastest.

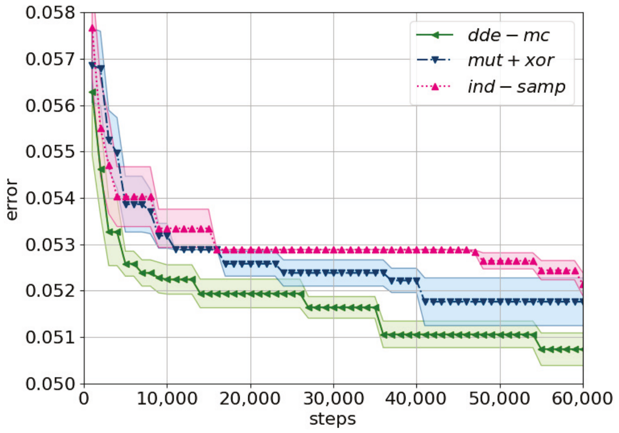


Figure 7. A comparison of the considered proposals using the minimum training error on NAS-Bench-101. The mean minimum error with its corresponding standard error (shaded area) across five cross-evaluations is plotted. Tolerance is set to $\epsilon = Exp(0.2)$. The prior distribution is as described in (14), with the corresponding Metropolis ratio (15). The following equations describe the proposals: *ind-samp* in (5), *dde-mc* in (10), and *mut+xor* as in [19].

Table 3. Test error on NAS-Bench-101.

Proposal	Error (ste)
dde-mc	0.058 (0.001)
mut+xor	0.060 (<0.001)
ind-samp	0.062 (<0.001)

6. Conclusions

In this paper, we note that there is a gap in the available methods for likelihood-free inference on discrete problems. We propose to utilize ideas known from evolutionary computing similarly to [26], in order to formulate a new Markov kernel, *dde-mc*, for a population-based MCMC-ABC algorithm. The obtained results suggest that the newly designed proposal is a promising and effective solution for intractable problems in a discrete space.

Furthermore, Markov kernels based on differential evolution are also effective to traverse a discrete search space. Nonetheless, great attention has to be paid to the choice of the tolerance threshold for the MCMC-ABC methods. In other words, if the tolerance is set too high, then the performance of the DE-based proposals drops to that of an independent sampler, i.e., the error of the Markov chain is high. For high-dimensional problems, the proposed kernel seems to be most promising; however, its population error becomes similar to that of *ind-samp*. This is accounted for by the fact that for high dimensions, it takes more time for the entire population to converge.

In conclusion, we would like to highlight that the present work offers new research directions:

- Alternative ABC algorithms like SMC should be further investigated.
- In this work, we focused on calculating distances in the data space. However, utilizing summary statistics is almost an obvious direction for future work.
- As the whole algorithm is based on logical operators and the input variables are also binary, the algorithm could be encoded using only bits, thus saving considerable amounts of memory storage. Consequently, any matrix multiplication could be replaced by an XNORoperation followed by a sum, thus reducing the computation costs and possibly allowing implementing the algorithm on relatively simple devices. Therefore, a natural consequence of this work would be a direct hardware implementation of the proposed methods.
- In this paper, we outline a number of potential applications of the presented methodology and indicate that the obtained results are of great practical potential. From the optimization perspective, a discrete ABC gives an opportunity to solve a problem in a principled manner. This is extremely important for applications associated with deep learning, e.g., NAS [20,35], neural network quantization, and learning binary neural networks, but also in other domains like topology or relationship discovery in biological networks (e.g., Boolean networks) [36]. Moreover, ABC as a Bayesian framework allows calculating model evidence that is crucial for model selection. In practice, very often, a problem is of combinatorial (discrete) nature, e.g., contamination control or pest control [35]. Therefore, our approach could be seemingly applied without the necessity of dequantizing a problem.

Author Contributions: Conceptualization, J.M.T.; methodology, I.A.A. and J.M.T.; software, I.A.A.; validation, I.A.A.; formal analysis, I.A.A. and J.M.T.; investigation, I.A.A. and J.M.T.; writing—original draft preparation, I.A.A. and J.M.T.; writing—review and editing, I.A.A. and J.M.T.; visualization, I.A.A.; supervision, J.M.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The code used in this paper is available at: https://github.com/IlzeAmandaA/ABCdiscrete (accessed on 5 March 2021).

Acknowledgments: We would like to thank [37] for the usage of the Distributed ASCI Supercomputer (DAS).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

ABC	Approximate Bayesian Computation
SMC	Sequential Monte Carlo
DE	Differential Evolution
MCMC	Markov Chain Monte Carlo
DE-MC	Differential Evolution Markov Chain
mut+xor	a mixture of a mutation-based proposal and an xor-based proposal
dde-mc	discrete differential evolution Markov chain
Population-MCMC-ABC	a population-based MCMC ABC
NAS	Neural Network Architecture Search
ind-samp	independent sampler
mut+crx	a mixture of a mutation-based proposal and a cross-over-based proposal
BinNN	a binary neural network

Appendix A. ϵ Determination

The choice of ϵ defines which data points are going to be accepted; as such, it implicitly models the likelihood. Setting the value too high will result in a biased estimate; however, it will improve the performance of Monte Carlo as more samples are utilized per unit time. Hence, as [4] already has stated: "the goal is to find a good balance between the bias and the Monte Carlo error".

Appendix A.1. Fixed ϵ

The first group of tolerance selection methods are all based on a fixed ϵ value. The possible approaches are summarized as follows:

- Determine a desirable acceptance ratio: For example, define a proportion, 1%, of the simulated samples that should be accepted ([2]).
- Re-use the generated samples: Determine the optimal cutoff value by a leave-one-out cross-validation approach of the underlying parameters of the generated simulations. In particular, minimize the Root Mean Squared Error (RMSE) for the validation parameter values [28].
- Use a pilot run to tune: Based on the rates of convergence [27], define fixed alterations to the initial tolerance value in order to either increase the number of accepted samples, reduce the mean-squared error, or increase the (expected) running time.
- Set ϵ to be proportional to $N_s^{-1/(d+5)}$: where d is the number of dimensions (for a complete overview, see [4]).

Nonetheless, setting ϵ to a fixed value hinders the convergence as it clearly is a sub-optimal approach due to its static nature. Ideally, we want to promote exploration at the beginning of the algorithm and, subsequently, move towards exploitation, hence alluding to the second group of tolerance selection methods: adaptive ϵ.

Appendix A.2. Adaptive ϵ

In general, the research on adaptive tolerance methods for MCMC-ABC is very limited as traditionally, adaptive tolerance is seen as part of SMC-ABC. In the current literature, two adaptive tolerance methods for MCMC-ABC are mentioned:

- An exponential cooling scheme: Reference [29] suggested using an exponential temperature scheme combined with a cooling scheme for the covariance matrix $\sum_t$.
- Sample from the exponential distribution: Similarly, Reference [30] assumed a pseudo-prior for $\epsilon : \pi(\epsilon)$, where $\pi(\epsilon) \sim Exp(\tau)$ and $\tau = 1/10$, thus allowing occasionally generating larger tolerance values to adjust mixing.

In order to establish a clear baseline for MCMC-ABC in a discrete space, we decided to implement both fixed and adaptive ϵ. Such an approach allows us to evaluate what is the effect of an adaptive ϵ in comparison to a fixed ϵ in a discrete space, as well as to compare how well our observations are in line with the observations drawn in a continuous space.

References

1. Audet, C.; Hare, W. *Derivative-Free and Blackbox Optimization*; Springer: Berlin/Heisenberg, Germany, 2017.
2. Beaumont, M.A.; Zhang, W.; Balding, D.J. Approximate Bayesian computation in population genetics. *Genetics* **2002**, *162*, 2025–2035.
3. Cranmer, K.; Brehmer, J.; Louppe, G. The frontier of simulation-based inference. *Proc. Natl. Acad. Sci. USA* **2020**, 117, 30055–30062. [CrossRef]
4. Lintusaari, J.; Gutmann, M.U.; Dutta, R.; Kaski, S.; Corander, J. Fundamentals and recent developments in approximate Bayesian computation. *Syst. Biol.* **2017**, *66*, e66–e82. [CrossRef]
5. Toni, T.; Welch, D.; Strelkowa, N.; Ipsen, A.; Stumpf, M.P. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *J. R. Soc. Interface* **2009**, *6*, 187–202. [CrossRef]
6. Jang, E.; Gu, S.; Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv* **2016**, arXiv:1611.01144.
7. Alquier, P. Approximate Bayesian Inference. *Entropy* **2020**, *22*, 1272. [CrossRef]
8. Pritchard, J.K.; Seielstad, M.T.; Perez-Lezaun, A.; Feldman, M.W. Population growth of human Y chromosomes: A study of Y chromosome microsatellites. *Mol. Biol. Evol.* **1999**, *16*, 1791–1798. [CrossRef]
9. Tavaré, S.; Balding, D.J.; Griffiths, R.C.; Donnelly, P. Inferring coalescence times from DNA sequence data. *Genetics* **1997**, *145*, 505–518. [CrossRef]
10. Marjoram, P.; Molitor, J.; Plagnol, V.; Tavaré, S. Markov chain Monte Carlo without likelihoods. *Proc. Natl. Acad. Sci. USA* **2003**, *100*, 15324–15328. [CrossRef] [PubMed]
11. Beaumont, M.A.; Cornuet, J.M.; Marin, J.M.; Robert, C.P. Adaptive approximate Bayesian computation. *Biometrika* **2009**, *96*, 983–990. [CrossRef]
12. Papamakarios, G. Neural density estimation and likelihood-free inference. *arXiv* **2019**, arXiv:1910.13233.
13. Papamakarios, G.; Sterratt, D.; Murray, I. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In Proceedings of the The 22nd International Conference on Artificial Intelligence and Statistics, Okinawa, Japan, 16–19 April 2019; pp. 837–848.
14. Andrieu, C.; Roberts, G.O. The pseudo-marginal approach for efficient Monte Carlo computations. *Ann. Stat.* **2009**, *37*, 697–725. [CrossRef]
15. Jasra, A.; Stephens, D.A.; Holmes, C.C. On population-based simulation for static inference. *Stat. Comput.* **2007**, *17*, 263–279. [CrossRef]
16. Storn, R.; Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
17. Vesterstrom, J.; Thomsen, R. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753), Portland, OR, USA, 19–23 June 2004; Volume 2, pp. 1980–1987.
18. Maučec, M.S.; Brest, J.; Bošković, B.; others. Improved differential evolution for large-scale black-box optimization. *IEEE Access* **2018**, *6*, 29516–29531.
19. Strens, M. Evolutionary MCMC sampling and optimization in discrete spaces. In Proceedings of the 20th International Conference on Machine Learning (ICML-03), Washington, DC, USA, 21–24 August 2003; pp. 736–743.
20. Ying, C.; Klein, A.; Christiansen, E.; Real, E.; Murphy, K.; Hutter, F. Nas-bench-101: Towards reproducible neural architecture search. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 7105–7114.
21. Metropolis, N.; Ulam, S. The monte carlo method. *J. Am. Stat. Assoc.* **1949**, *44*, 335–341. [CrossRef] [PubMed]
22. Iba, Y. Population monte carlo algorithms. *Trans. Jpn. Soc. Artif. Intell.* **2001**, *16*, 279–286. [CrossRef]

23. Hukushima, K.; Nemoto, K. Exchange Monte Carlo method and application to spin glass simulations. *J. Phys. Soc. Jpn.* **1996**, *65*, 1604–1608. [CrossRef]
24. Liang, F.; Wong, W.H. Evolutionary Monte Carlo: Applications to C p model sampling and change point problem. *Stat. Sin.* **2000**, *10*, 317–342.
25. Strens, M.J.; Bernhardt, M.; Everett, N. Markov Chain Monte Carlo Sampling Using Direct Search Optimization. In Proceedings of the Nineteenth International Conference on Machine Learning, ICML, Sydney, Australia, 8–12 July 2002; pp. 602–609.
26. Ter Braak, C.J. A Markov Chain Monte Carlo version of the genetic algorithm Differential Evolution: Easy Bayesian computing for real parameter spaces. *Stat. Comput.* **2006**, *16*, 239–249. [CrossRef]
27. Barber, S.; Voss, J.; Webster, M. The rate of convergence for approximate Bayesian computation. *Electron. J. Stat.* **2015**, *9*, 80–105. [CrossRef]
28. Faisal, M.; Futschik, A.; Hussain, I. A new approach to choose acceptance cutoff for approximate Bayesian computation. *J. Appl. Stat.* **2013**, *40*, 862–869. [CrossRef]
29. Ratmann, O.; Jørgensen, O.; Hinkley, T.; Stumpf, M.; Richardson, S.; Wiuf, C. Using likelihood-free inference to compare evolutionary dynamics of the protein networks of H. pylori and P. falciparum. *PLoS Comput. Biol.* **2007**, *3*, e230. [CrossRef] [PubMed]
30. Bortot, P.; Coles, S.G.; Sisson, S.A. Inference for stereological extremes. *J. Am. Stat. Assoc.* **2007**, *102*, 84–92. [CrossRef]
31. Jaakkola, T.S.; Jordan, M.I. Variational probabilistic inference and the QMR-DT network. *J. Artif. Intell. Res.* **1999**, *10*, 291–322. [CrossRef]
32. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
33. Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv* **2016**, arXiv:1602.02830.
34. Tomczak, J.M.; Zieba, M. Probabilistic combination of classification rules and its application to medical diagnosis. *Mach. Learn.* **2015**, *101*, 105–135. [CrossRef]
35. Oh, C.; Tomczak, J.M.; Gavves, E.; Welling, M. Combinatorial Bayesian optimization using the graph cartesian product. In Proceedings of the Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019.
36. Friedman, N.; Linial, M.; Nachman, I.; Pe'er, D. Using Bayesian networks to analyze expression data. *J. Comput. Biol.* **2000**, *7*, 601–620. [CrossRef] [PubMed]
37. Bal, H.; Epema, D.; de Laat, C.; van Nieuwpoort, R.; Romein, J.; Seinstra, F.; Snoek, C.; Wijshoff, H. A medium-scale distributed system for computer science research: Infrastructure for the long term. *Computer* **2016**, *49*, 54–63. [CrossRef]

Article

Variationally Inferred Sampling through a Refined Bound

Víctor Gallego [1,2,*] and David Ríos Insua [1,3]

1 Institute of Mathematical Sciences (ICMAT), 28049 Madrid, Spain; david.rios@icmat.es
2 Statistical and Applied Mathematical Sciences Institute, Durham, NC 7333, USA
3 School of Management, University of Shanghai for Science and Technology, Shanghai 201206, China
* Correspondence: victor.gallego@icmat.es

Abstract: In this work, a framework to boost the efficiency of Bayesian inference in probabilistic models is introduced by embedding a Markov chain sampler within a variational posterior approximation. We call this framework "refined variational approximation". Its strengths are its ease of implementation and the automatic tuning of sampler parameters, leading to a faster mixing time through automatic differentiation. Several strategies to approximate evidence lower bound (ELBO) computation are also introduced. Its efficient performance is showcased experimentally using state-space models for time-series data, a variational encoder for density estimation and a conditional variational autoencoder as a deep Bayes classifier.

Keywords: variational inference; MCMC; stochastic gradients; neural networks

Citation: Gallego, V.; Ríos Insua, D. Variationally Inferred Sampling through a Refined Bound. *Entropy* **2021**, *23*, 123. https://doi.org/10.3390/e23010123

Received: 24 December 2020
Accepted: 13 January 2021
Published: 19 January 2021

1. Introduction

Bayesian inference and prediction in large, complex models, such as in deep neural networks or stochastic processes, remains an elusive problem [1–3]. Variational approximations (e.g., automatic differentiation variational inference (ADVI) [4]) tend to be biased and underestimate uncertainty [5]. On the other hand, depending on the target distribution, Markov Chain Monte Carlo (MCMC) [6] methods, such as Hamiltonian Monte Carlo (HMC) [7]), tend to be exceedingly slow [8] in large scale settings with large amounts of data points and/or parameters. For this reason, in recent years, there has been increasing interest in developing more efficient posterior approximations [9–11] and inference techniques that aim to be as general and flexible as possible so that they can be easily used with any probabilistic model [12,13].

It is well known that the performance of a sampling method depends heavily on the parameterization used [14]. This work proposes a framework to automatically tune the parameters of a MCMC sampler with the aim of adapting the shape of the posterior, thus boosting the Bayesian inference efficiency. We deal with a case in which the latent variables or parameters are continuous. Our framework can also be regarded as a principled way to enhance the flexibility of variational posterior approximation in search of an optimally tuned MCMC sampler; thus the proposed name of our framework is the variationally inferred sampler (VIS).

The idea of preconditioning the posterior distribution to speed up the mixing time of a MCMC sampler has been explored recently in [15,16], where a parameterization was learned before sampling via HMC. Both papers extend seminal work in [17] by learning an efficient and expressive deep, non-linear transformation instead of a polynomial regression. However, they do not account for tuning the parameters of the sampler, as introduced in Section 3, where a fully, end-to-end differentiable sampling scheme is proposed.

The work presented in [18] introduced a general framework for constructing more flexible variational distributions, called normalizing flows. These transformations are one of the main techniques used to improve the flexibility of current variational inference (VI) approaches and have recently pervaded the approximate Bayesian inference literature with developments such as continuous-time normalizing flows [19] (which extend an initial simple variational posterior with a discretization of Langevin dynamics) or householder flow for mixtures of Gaussian distributions [20]. However, they require a generative adversarial network (GAN) [21] to learn the posterior, which can be unstable in high-dimensional spaces. We overcome this problem with our novel formulation; moreover, our framework is also compatible with different optimizers, rather than only those derived from Langevin dynamics [22]. Other recent proposals create more flexible variational posteriors based on implicit approaches typically requiring a GAN, as presented in [23] and including unbiased implicit variational inference (UIVI) [24] or semi-implicit variational inference (SIVI) [25]. Our variational approximation is also implicit but uses a sampling algorithm to drive the evolution of the density, combined with a Dirac delta approximation to derive an efficient variational approximation, as reported through extensive experiments in Section 5.

Closely related to our framework is the work presented in [26], where a variational autoencoder (VAE) is learned using HMC. We use a similar compound distribution as the variational approximation, yet our approach allows any stochastic gradient MCMC to be embedded, as well as facilitating the tuning of sampler parameters via gradient descent. Our work also relates to the recent idea of sampler amortization [27]. A common problem with these approaches is that they incur in an additional error—the amortization gap [28]—which we alleviate by evolving a set of particles through a stochastic process in the latent space after learning a good initial distribution, meaning that the initial approximation bias can be significantly reduced. A recent related article was presented in [29], which also defined a compound distribution. However, our focus is on efficient approximation using the reverse KL divergence, which allows sampler parameters to be tuned and achieves superior results. Apart from optimizing this kind of divergence, the main point is that we can compute the gradients of sampler parameters (Section 3.3), whereas in [29] the authors only consider a parameterless sampler: thus, our framework allows for greater flexibility, helping the user to tune sampler hyperparameters. In the Coupled Variational Bayes (CVB) [30] approach, optimization is in the dual space, whereas we optimize the standard evidence lower bound (ELBO). Note that even if the optimization was exact, the solutions would coincide, and it is not clear yet what happens in the truncated optimization case,other than performing empirical experiments on given datasets. We thus feel that there is room for implicit methods that perform optimization in the primal space (besides this, they are easier to implement). Moreover, the previous dual optimization approach requires the use of an additional neural network (see the paper on the Coupled Variational Bayes (CVB) approach or [31]). This adds a large number of parameters and requires another architecture decision. With VIS, we do not need to introduce an auxiliary network, since we perform a "non-parametric" approach by back-propagating instead through several iterations of SGLD. Moreover, the lack of an auxiliary network simplifies the design choices.

Thus, our contributions include a flexible and consistent variational approximation to the posterior, embedding an initial variational approximation within a stochastic process; an analysis of its key properties; the provision of several strategies for ELBO optimization using the previous approximation; and finally, an illustration of its power through relevant complex examples.

2. Background

Consider a probabilistic model $p(x|z)$ and a prior distribution $p(z)$, where x denotes the observations and $z \in \mathbb{R}^d$ the unobserved latent variables or parameters, depending on the context. Whenever necessary for disambiguation purposes, we shall distinguish between z for latent variables and θ for parameters. Our interest is in performing inference regarding the unobserved z by approximating its posterior distribution

$$p(z|x) = \frac{p(z)p(x|z)}{\int p(z)p(x|z)dz} = \frac{p(x,z)}{p(x)}.$$

The integral assessing the evidence $p(x) = \int p(z)p(x|z)dz$ is typically intractable. Thus, several techniques have been proposed to perform approximate posterior inference [3].

2.1. Inference as Optimization

Variational inference (VI) [4] tackles the problem of approximating the posterior $p(z|x)$ with a tractable parameterized distribution $q_\phi(z|x)$. The goal is to find the parameters ϕ so that the variational distribution $q_\phi(z|x)$ (also referred to as variational guide or variational approximation) can be as close as possible to the actual posterior. Closeness is typically measured through the Kullback–Leibler divergence $KL(q_\phi||p)$, reformulated into the ELBO as follows:

$$\text{ELBO}(q) = \mathbb{E}_{q_\phi(z|x)}\left[\log p(x,z) - \log q_\phi(z|x)\right], \tag{1}$$

This is the objective to be optimized, typically through stochastic gradient descent techniques. To enhance flexibility, a standard choice for $q_\phi(z|x)$ is a Gaussian distribution $\mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$, with the mean and covariance matrix defined through a deep, non-linear model conditioned on observation x.

2.2. Inference as Sampling

HMC [7] is an effective sampling method for models whose probability is pointwise computable and differentiable. When scalability is an issue, as proposed by the authors in [32], a formulation of a continuous-time Markov process that converges to the target distribution $p(z|x)$ can be used, which is based on the Euler–Maruyama discretization of Langevin dynamics

$$z_{t+1} \leftarrow z_t + \eta_t \nabla_z \log p(x, z_t) + \mathcal{N}(0, 2\eta_t I), \tag{2}$$

where η_t is the step size at time period t, and I is the identity matrix. The required gradient $\nabla \log p(z_t, x)$ can be estimated using mini-batches of data. Several extensions of the original Langevin sampler have been proposed to increase its mixing speed, such as in [33–36]. We refer to these extensions as stochastic gradient MCMC samplers (SG-MCMC) [37].

3. A Variationally Inferred Sampling Framework

In standard VI, the variational approximation is analytically tractable and typically chosen as a factorized Gaussian, as mentioned above. However, it is important to note that other distributions can be adopted as long as they are easily sampled and their log-density and entropy values evaluated. However, in the rest of this paper, we focus on the Gaussian case, as the usual choice in the Bayesian deep learning community. Stemming from this variational approximation, we introduce several elements to construct the VIS.

Our first major modification of standard VI proposes the use of a more flexible distribution, approximating the posterior by embedding a sampler through

$$q_{\phi,\eta}(z|x) = \int Q_{\eta,T}(z|z_0) q_{0,\phi}(z_0|x) dz_0, \tag{3}$$

where $q_{0,\phi}(z|x)$ is the initial and tractable density $q_\phi(z|x)$ (i.e., the starting state for the sampler). We designate this as refined variational approximation. The conditional distribution $Q_{\eta,T}(z|z_0)$ refers to a stochastic process parameterized by η and used to evolve the original density $q_{0,\phi}(z|x)$ for T periods, so as to achieve greater flexibility. Specific forms for $Q_{\eta,T}(z|z_0)$ are described in Section 3.1. Observe that when $T = 0$, no refinement steps are performed and the refined variational approximation coincides with the original one; on the other hand, as T increases, the approximation will be closer to the exact posterior, assuming that $Q_{\eta,T}$ is a valid MCMC sampler in the sense of [37].

We next maximize a refined ELBO objective, replacing in Equation (1) the original q_ϕ by $q_{\phi,\eta}$:

$$\text{ELBO}(q_{\phi,\eta}) = \mathbb{E}_{q_{\phi,\eta}(z|x)}\left[\log p(x,z) - \log q_{\phi,\eta}(z|x)\right] \tag{4}$$

This is done to optimize the divergence $KL(q_{\phi,\eta}(z|x)||p(z|x))$. The first term of Equation (4) requires only being able to sample from $q_{\phi,\eta}(z|x)$; however, the second term, the entropy $-\mathbb{E}_{q_{\phi,\eta}(z|x)}\left[\log q_{\phi,\eta}(z|x)\right]$, also requires the evaluation of the evolving, implicit density. Section 3.2 describes efficient methods to approximate this evaluation. As a consequence, performing variational inference with the refined variational approximation can be regarded as using the original variational guide while optimizing an alternative, tighter ELBO, as Section 4.2 shows.

The above facilitates a framework for learning the sampler parameters ϕ, η using gradient-based optimization, with the help of automatic differentiation [38]. For this, the approach operates in two phases. First, in a refinement phase, the sampler parameters are learned in an optimization loop that maximizes the ELBO with the new posterior. After several iterations, the second phase, focused on inference, starts. We allow the tuned sampler to run for sufficient iterations, as in SG-MCMC samplers. This is expressed algorithmically as follows.

Refinement phase:

Repeat the following until convergence:

1. Sample an initial set of particles, $z_0 \sim q_{0,\phi}(z|x)$.
2. Refine the particles through the sampler, $z_T \sim Q_{\eta,T}(z|z_0)$.
3. Compute the ELBO objective from Equation (4).
4. Perform automatic differentiation on the objective wrt parameters ϕ, η to update them.

Inference phase:

Once good sampler parameters ϕ^*, η^* are learned,

1. Sample an initial set of particles, $z_0 \sim q_{0,\phi^*}(z|x)$.
2. Use the MCMC sampler $z_T \sim Q_{\eta^*,T}(z|z_0)$ as $T \to \infty$.

Since the sampler can be run for a different number of steps depending on the phase, we use the following notation when necessary: VIS-X-Y denotes $T = X$ iterations during the refining phase and $T = Y$ iterations during the inference phase.

Let us specify now the key elements.

3.1. The Sampler $Q_{\eta,T}(Z|Z_0)$

As the latent variables z are continuous, we evolve the original density $q_{0,\phi}(z|x)$ through a stochastic diffusion process [39]. To make it tractable, we discretize the Langevin dynamics using the Euler–Maruyama scheme, arriving at the stochastic gradient Langevin dynamics (SGLD) sampler (2). We then follow the process $Q_{\eta,T}(z|z_0)$, which represents T iterations of the MCMC sampler.

As an example, for the SGLD sampler $z_t = z_{t-1} + \eta\nabla \log p(x, z_{t-1}) + \xi_t$, where t iterates from 1 to T. In this case, the only parameter is the learning rate η and the noise is $\xi_t \sim \mathcal{N}(0, 2\eta I)$. The initial variational distribution $q_{0,\phi}(z|x)$ is a Gaussian parameterized by a deep neural network (NN). Then, after T iterations of the sampler Q are parameterized by η, we arrive at $q_{\phi,\eta}$.

An alternative arises by ignoring the noise ξ [22], thus refining the initial variational approximation using only the stochastic gradient descent (SGD). Moreover, we can use Stein variational gradient descent (SVGD) [40] or a stochastic version [36] to apply repulsion between particles and promote more extensive explorations of the latent space.

3.2. Approximating the Entropy Term

We propose four approaches for the ELBO optimization which take structural advantage of the refined variational approximation.

3.2.1. Particle Approximation (VIS-P)

In this approach, we approximate the posterior $q_{\phi,\eta}(z|x)$ by a mixture of Dirac deltas (i.e., we approximate it with a finite set of particles), by sampling $z^{(1)}, \dots, z^{(M)} \sim q_{\phi,\eta}(z|x)$ and setting

$$q_{\phi,\eta}(z|x) = \frac{1}{M} \sum_{m=1}^{M} \delta(z - z^{(m)}).$$

In this approximation, the entropy term in (4) is set to zero. Consequently, the sample converges to the maximum posterior (MAP). This may be undesirable when training generative models, as the generated samples usually have little diversity. Thus, in subsequent computations, we add to the refined ELBO the entropy of the initial variational approximation, $\mathbb{E}_{q_{0,\phi}(z|x)}\left[\log q_{0,\phi}(z|x)\right]$, which serves as a regularizer alleviating the previous problem. When using SGD as the sampler, the resulting ELBO is tighter than that without refinement, as shown in Section 4.2.

3.2.2. MC Approximation (VIS-MC)

Instead of performing the full marginalization in Equation (3), we approximate it with $q_{\phi,\eta}(z_T, \dots, z_0|x) = \prod_{t=1}^{T} q_\eta(z_t|z_{t-1}) q_{0,\phi}(z_0|x)$; i.e., we consider the joint distribution for the refinement. However, in inference we only keep the z_T values. The entropy for each factor in this approximation is straightforward to compute. For example, for the SGLD case, we have

$$z_t = z_{t-1} + \eta \nabla \log p(x, z_{t-1}) + \mathcal{N}(0, 2\eta I), \qquad t = 1, ..., T.$$

This approximation tracks a better estimate of the entropy than VIS-P, as we are not completely discarding it; rather, for each t, we marginalize out the corresponding z_t using one sample.

3.2.3. Gaussian Approximation (VIS-G)

This approach is targeted at settings in which it could be helpful to have a posterior approximation that places density over the whole z space. In the specific case of using SGD as the inner kernel, we have

$$\begin{aligned} z_0 &\sim q_{0,\phi}(z_0|x) = \mathcal{N}(z_0|\mu_\phi(x), \sigma_\phi(x)) \\ z_t &= z_{t-1} + \eta \nabla \log p(x, z_{t-1}), \qquad t = 1, \dots, T. \end{aligned}$$

By treating the gradient terms as points, the refined variational approximation can be computed as $q_{\phi,\eta}(z|x) = \mathcal{N}(z|z_T, \sigma_\phi(x))$. Observe that there is an implicit dependence on η through z_T.

3.2.4. Fokker–Planck Approximation (VIS-FP)

Using the Fokker–Planck equation, we derive a deterministic sampler via iterations of the form

$$z_t = z_{t-1} + \eta(\nabla \log p(x, z_{t-1}) - \nabla \log q_t(z_{t-1})), \qquad t = 1, ..., T.$$

Then, we approximate the density $q_{\phi,\eta}(z|x)$ using a mixture of Dirac deltas. A detailed derivation of this approximation is given in Appendix A.

3.3. Back-Propagating through the Sampler

In standard VI, the variational approximation $q(z|x; \phi)$ is parameterized by ϕ. The parameters are learned employing SGD, or variants such as Adam [41], using the gradient $\nabla_\phi \text{ELBO}(q)$. We have shown how to embed a sampler inside the variational guide. It is therefore also possible to compute a gradient of the objective with respect to the sampler parameters η (see Section 3.1). For instance, we can compute a gradient $\nabla_\eta \text{ELBO}(q)$ with respect to the learning rate η from the SGLD or SGD processes to search for an optimal step size at every VI iteration. This is an additional step apart from using the gradient $\nabla_\phi \text{ELBO}(q)$ which is used to learn a good initial sampling distribution.

4. Analysis of Vis

Below, we highlight key properties of the proposed framework.

4.1. Consistency

The VIS framework is geared towards SG-MCMC samplers, where we can compute the gradients of sampler hyperparameters to speed up mixing time (a common major drawback in MCMC [42]). After back-propagating for a few iterations through the SG-MCMC sampler and learning a good initial distribution, one can resort to the learned sampler in the second phase, so standard consistency results from SG-MCMC apply as $T \to \infty$ [43].

4.2. Refinement of ELBO

Note that, for a refined guide using the VIS-P approximation and $M = 1$ samples, the refined objective function can be written as

$$\mathbb{E}_{q(z_0|x)}[\log p(x, z_0 + \eta \nabla \log p(x, z_0)) - \log q(z_0|x)]$$

noting that $z = z_0 + \eta \nabla \log p(x, z_0)$ when using SGD for $T = 1$ iterations. This is equivalent to the refined ELBO in (4). Since we are perturbing the latent variables in the steepest direction, we show easily that, for a moderate η, the previous bound is tighter than $\mathbb{E}_{q(z_0|x)}[\log p(x, z_0) - \log q(z_0|x)]$, the one for the original variational guide $q(z_0|x)$. This reformulation of ELBO is also convenient since it provides a clear way of implementing our refined variational inference framework in any probabilistic programming language (PPL) supporting algorithmic differentiation.

Respectively, for the VIS-FP case, we find that its deterministic flow follows the same trajectories as SGLD: based on standard results of MCMC samplers [44], we have

$$KL(q_{\phi,\eta}(z|x)||p(z|x)) \leq KL(q_{0,\phi}(z|x)||p(z|x)).$$

A similar reasoning applies to the VIS-MC approximation; however, it does not hold for VIS-G since it assumes that the posterior is Gaussian.

4.3. Taylor Expansion

This analysis applies only to VIS-P and VIS-FP. As stated in Section 4.2, within the VIS framework, optimizing the ELBO resorts to the performance of $\max_z \log p(x, z + \Delta z)$, where Δz is one iteration of the sampler; i.e., $\Delta z = \eta \nabla \log p(x, z)$ in the SGD case (VIS-P), or $\Delta z = \eta \nabla(\log p(x, z) - \log q(z))$ in the VIS-FP case. For notational clarity, we consider the case $T = 1$, although a similar analysis follows in a straightforward manner if more refinement steps are performed.

Consider a first-order Taylor expansion of the refined objective

$$\log p(x, z + \Delta z) \approx \log p(x, z) + (\Delta z)^\intercal \nabla \log p(x, z).$$

Taking gradients with respect to the latent variables z, we arrive at

$$\nabla_z \log p(x, z + \Delta z) \approx \nabla_z \log p(x, z) + \eta \nabla_z \log p(x, z)^\intercal \nabla_z^2 \log p(x, z),$$

where we have not computed the gradient through the Δz term (i.e., we treated it as a constant for simplification). Then, the refined gradient can be deemed to be the original gradient plus a second order correction. Instead of being modulated by a constant learning rate, this correction is adapted by the chosen sampler. The experiments in Section 5.4 show that this is beneficial for the optimization as it typically takes fewer iterations than the original variant to achieve lower losses.

By further taking gradients through the Δz term, we may tune the sampler parameters such as the learning rate as presented in Section 3.3. Consequently, the next subsection describes two differentiation modes.

4.4. Two Automatic Differentiation Modes for Refined ELBO Optimization

For the first variant, remember that the original variant can be rewritten (which we term Full AD) as

$$\mathbb{E}_q[\log p(x, z + \Delta z) - \log q(z + \Delta z | x)]. \tag{5}$$

We now define a stop gradient operator $\perp$ (which corresponds to `detach` in Pytorch or `stop_gradient` in tensorflow) that sets the gradient of its operand to zero—i.e., $\nabla_x \perp(x) = 0$—whereas in a forward pass, it acts as the identity function—that is, $\perp(x) = x$. With this, a variant of the ELBO objective (which we term Fast AD) is

$$\mathbb{E}_q[\log p(x, z + \perp(\Delta z)) - \log q(z + \perp(\Delta z)|x)]. \tag{6}$$

Full AD ELBO enables a gradient to be computed with respect to the sampler parameters inside Δz at the cost of a slight increase in computational burden. On the other hand, the Fast AD variant may be useful in numerous scenarios, as illustrated in the experiments.

Complexity

Since we need to back propagate through T iterations of an SG-MCMC scheme, using standard results of meta-learning and automatic differentiation [45], the time complexity of our more intensive approach (Full-AD) is $\mathcal{O}(mT)$, where m is the dimension of the hyperparameters (the learning rate of SG-MCMC and the latent dimension). Since for most use cases, the hyperparameters lie in a low-dimensional space, the approach is therefore scalable.

5. Experiments

The following experiments showcase the power of our approach as well as illustrating the the impact of various parameters on its performance, guiding their choice in practice. We also present a comparison with standard VIS and other recent variants, showing that the increased computational complexity of computing gradients through sampling steps is worth the gains in flexibility. Moreover, the proposed framework is compatible with other structured inference techniques, such as the sum–product algorithm, as well as serving to support other tasks such as classification.

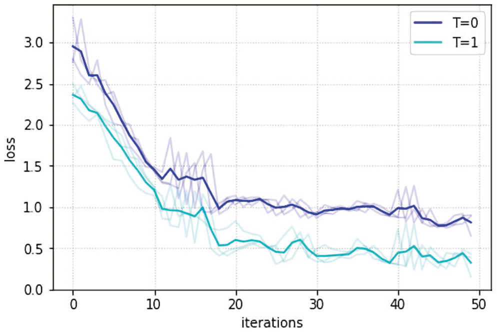
Within the spirit of reproducible research, the code for VIS has been released at https://github.com/vicgalle/vis. The VIS framework is implemented with Pytorch [46], although we have also released a notebook for the first experiment using Jax to highlight the simple implementation of VIS. In any case, we emphasize that the approach facilitates rapid iterations over a large class of models.

5.1. Funnel Density

We first tested the framework on a synthetic yet complex target distribution. This experiment assessed whether VIS is suitable for modeling complex distributions. The target bi-dimensional density was defined through

$$z_1 \sim \mathcal{N}(0, 1.35)$$
$$z_2 \sim \mathcal{N}(0, \exp(z_1)).$$

We adopted the usual diagonal Gaussian distribution as the variational approximation. For VIS, we used the VIS-P approximation and refined it for $T = 1$ steps using SGLD. Figure 1 top shows the trajectories of the lower bound for up to 50 iterations of variational optimization with Adam: our refined version achieved a tighter bound. The bottom figures present contour curves of the learned variational approximations. Observe that the VIS variant was placed closer to the mean of the true distribution and was more disperse than the original variational approximation, illustrating the fact that the refinement step helps in attaining more flexible posterior approximations.



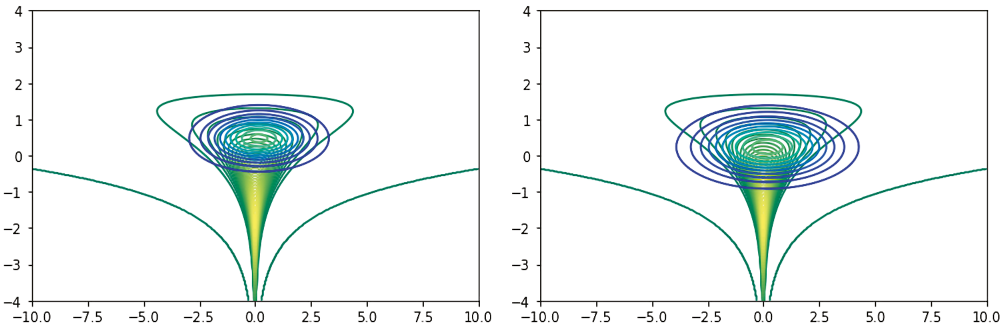


Figure 1. Top: Evolution of the negative evidence lower bound (ELBO) loss objective over 50 iterations. Darker lines depict means along different seeds (lighter lines). **Bottom left**: Contour curves (blue–turquoise) of the variational approximation with no refinement ($T = 0$) at iteration 30 (loss of 1.011). **Bottom right**: Contour curves (blue–turquoise) of refined variational approximation ($T = 1$) at iteration 30 (loss of 0.667). Green–yellow curves denote target density.

5.2. State-Space Markov Models

We tested our variational approximation on two state-space models: one for discrete data and another for continuous observations. These experiments also demonstrated that the framework is compatible with standard inference techniques such as the sum–product scheme from the Baum–Welch algorithm or Kalman filter. In both models, we performed inference on their parameters θ. All the experiments in this subsection used the Fast AD version (Section 4.4) as it was not necessary to further tune the sampler parameters to obtain competitive results. Full model implementations can be found in Appendix B.1, based on `funsor` (https://github.com/pyro-ppl/funsor/), a PPL on top of the `Pytorch` autodiff framework.

Hidden Markov Model (HMM): The model equations are

$$p(x_{1:\tau}, z_{1:\tau}, \theta) = \prod_{t=1}^{\tau} p(x_t|z_t, \theta_{em}) p(z_t|z_{t-1}, \theta_{tr}) p(\theta), \tag{7}$$

where each conditional is a categorical distribution taking five different classes. The prior is $p(\theta) = p(\theta_{em})p(\theta_{tr})$ based on two Dirichlet distributions that sample the observation and state transition probabilities, respectively.

Dynamic Linear Model (DLM): The model equations are as in (7), although the conditional distributions are now Gaussian and the parameters θ refer to the observation and transition variances.

For each model, we generated a synthetic dataset and used the refined variational approximation with $T = 0, 1, 2$. For the original variational approximation to the parameters θ, we used a Dirac delta. Performing VI with this approximation corresponded to MAP estimation using the Baum–Welch algorithm in the HMM case [47] and the Kalman filter in the DLM case [48], as we marginalized out the latent variables $z_{1:\tau}$. We used the VIS-P variant since it was sufficient to show performance gains in this case.

Figure 2 shows the results. The first row reports the experiments related to the HMM, the second row those for the DLM. We report the evolution of the log-likelihood during inference in all graphs; the first column reports the number of ELBO iterations, and the second column portrays clock times as the optimization takes place. They confirm that VIS ($T > 0$) achieved better results than standard VI ($T = 0$) for a comparable amount of time. Note also that there was not as much gain when changing from $T = 1$ to $T = 2$ as there is from $T = 0$ to $T = 1$, suggesting the need to carefully monitor this parameter. Finally, the top-right graph for the case $T = 0$ is shorter as it requires less clock time.

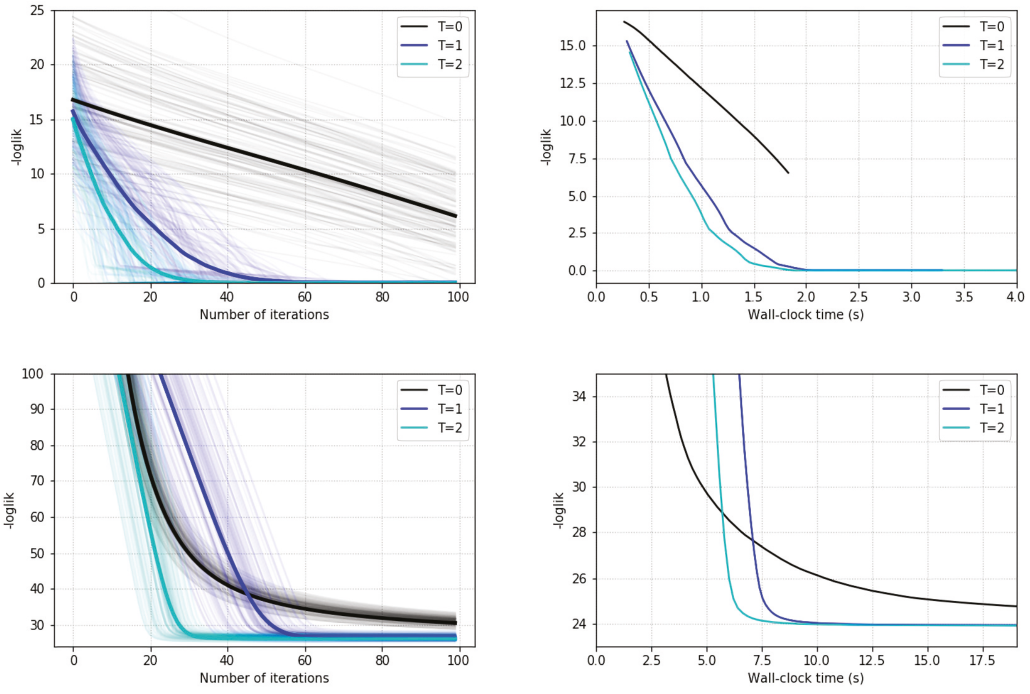


Figure 2. Results of ELBO optimization for state-space models. **Top-left** (Hidden Markov Model (HMM)): Log-likelihood against the number of ELBO gradient iterations. **Top-right** (HMM): Log-likelihood against clock time. **Bottom-left** (Dynamic Linear Model (DLM)): Log-likelihood against number of ELBO gradient iterations. **Bottom-right** (DLM): Log-likelihood against against clock time.

5.2.1. Prediction with an HMM

With the aim of assessing whether ELBO optimization helps in attaining better auxiliary scores, results in a prediction task are also reported. We generated a synthetic time series of alternating values of 0 and 1 for $\tau = 105$ timesteps. We trained the previous HMM model on the first 100 points and report in Table 1 the accuracy of the predictive distribution $p(y_t)$ averaged over the final five time-steps. We also report the predictive entropy as it helps in assessing the confidence of the model in its predictions, as a strictly proper scoring rule [49]. To guarantee the same computational budget time and a fair comparison, the model without refinement was run for 50 epochs (an epoch was a full iteration over the training dataset), whereas the model with refinement was run for 20 epochs. It can be observed that the refined model achieved higher accuracy than its counterpart. In addition, it was more correctly confident in its predictions.

Table 1. Prediction metrics for the HMM.

	$T = 0$	$T = 1$
accuracy	0.40	0.84
predictive entropy	1.414	1.056
logarithmic score	-1.044	-0.682

5.2.2. Prediction with a DLM

We tested the VIS framework on Mauna Loa monthly CO_2 time-series data [50]. We used the first 10 years as a training set, and we tested over the next 2 years. We used a DLM composed of a local linear trend plus a seasonal block of periodicity 12. Data were standardized to a mean of zero and standard deviation of one. To guarantee the same computational budget time, the model without refining was run for 10 epochs, whereas the model with refinement was run for 4 epochs. Table 2 reports the mean absolute error (MAE) and predictive entropy. In addition, we computed the interval score [49], as a strictly proper scoring rule. As can be seen, for similar clock times, the refined model not only achieved a lower MAE, but also its predictive intervals were narrower than the non-refined counterpart.

Table 2. Prediction metrics for the DLM.

	$T = 0$	$T = 1$
MAE	0.270	0.239
predictive entropy	2.537	2.401
interval score ($\alpha = 0.05$)	15.247	13.461

5.3. Variational Autoencoder

The third batch of experiments showed that VIS was competitive with respect to other algorithms from the recent literature, including unbiased implicit variational inference (UIVI [24]), semi-implicit variational inference (SIVI [25]), variational contrastive divergence (VCD [29]), and the HMC variant from [26], showing that our framework can outperform those approaches in similar experimental settings.

To this end, we tested the approach with a variational autoencoder (VAE) model [51]. The VAE defines a conditional distribution $p_\theta(x|z)$, generating an observation x from a latent variable z using parameters θ. For this task, our interest was in modeling the 28×28 image distributions underlying the MNIST [52] and the fashion-MNIST [53] datasets. To perform inference (i.e., to learn the parameters θ) the VAE introduces a variational approximation $q_\phi(z|x)$. In the standard setting, this distribution is Gaussian; we instead used the refined variational approximation comparing various values of T. We used the VIS-MC approximation (although we achieved similar results with VIS-G) with the Full AD variant given in Section 4.4.

For the experimental setup, we reproduced the setting in [24]. For $p_\theta(x|z)$, we used a factorized Bernoulli distribution parameterized by a two layer feed-forward network with 200 units in each layer and relu activations, except for a final sigmoid activation. As a variational approximation $q_\phi(z|x)$, we used a Gaussian with mean and (diagonal) covariance matrix parameterized by two distinct neural networks with the same structure as previously used, except for sigmoid activation for the mean and a softplus activation for the covariance matrix.

Results are reported in Table 3. To guarantee fair comparison, we trained the VIS-5-10 variant for 10 epochs, whereas all the other variants were trained for 15 epochs (fMNIST) or 20 epochs (MNIST), so that the VAE's performance was comparable to that reported in [24]. Although VIS was trained for fewer epochs, by increasing the number T of MCMC iterations, we dramatically improved the test log-likelihood. In terms of computational complexity, the average time per epoch using $T = 5$ was 10.46 s, whereas with no refinement ($T = 0$), the time was 6.10 s (which was the reason behind our decision to train the refined variant for fewer epochs): a moderate increase in computing time may be worth the dramatic increase in log-likelihood while not introducing new parameters into the model, except for the learning rate η.

Table 3. Test log-likelihood on binarized MNIST and fMNIST. Bold numbers indicate the best results. UIVI: unbiased implicit variational inference; SIVI: semi-implicit variational inference; VAE: variational autoencoder; VCD: variational contrastive divergence; HMC-DLGM: Hamiltonian Monte Carlo for Deep Latent Gaussian Models; VIS: variationally inferred sampler.

Method	MNIST	fMNIST
	Results from [24]	
UIVI	−94.09	−110.72
SIVI	−97.77	−121.53
VAE	−98.29	−126.73
	Results from [29]	
VCD	−95.86	−117.65
HMC-DLGM	−96.23	−117.74
	This paper	
VIS-5-10	**−82.74 ± 0.19**	**−105.08 ± 0.34**
VIS-0-10	−96.16 ± 0.17	−120.53 ± 0.59
VAE (VIS-0-0)	−100.91 ± 0.16	−125.57 ± 0.63

Finally, as a visual inspection of the VAE reconstruction quality trained with VIS, Figures 3 and 4, respectively, display 10 random samples of each dataset.

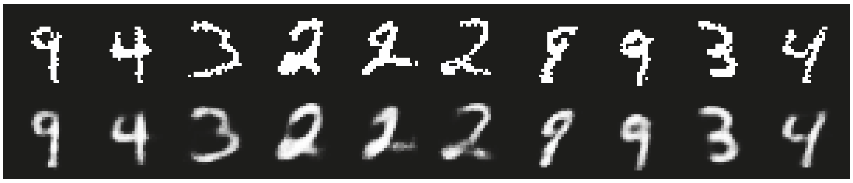


Figure 3. Top: original images from MNIST. Bottom: reconstructed images using VIS-5-10 at 10 epochs.

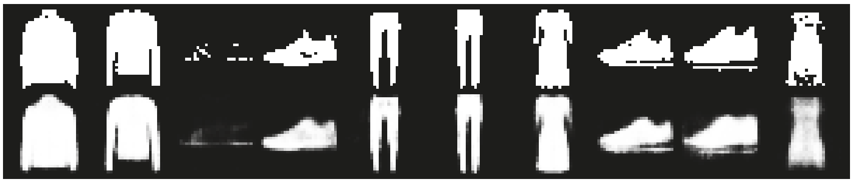


Figure 4. Top: original images from fMNIST. Bottom: reconstructed images using VIS-5-10 at 10 epochs.

5.4. Variational Autoencoder as a Deep Bayes Classifier

In the final experiments, we investigated whether VIS can deal with more general probabilistic graphical models and also perform well in other inference tasks such as classification. We explored the flexibility of the proposed scheme to solve inference problems in an experiment with a classification task in a high-dimensional setting with the MNIST dataset. More concretely, we extended the VAE model, conditioning it on a discrete variable $y \in \mathcal{Y} = \{0, 1, \ldots, 9\}$, leading to a conditional VAE (cVAE). The cVAE defined a decoder distribution $p_\theta(x|z, y)$ on an input space $x \in \mathbb{R}^D$ given a class label $y \in \mathcal{Y}$, latent variables $z \in \mathbb{R}^d$ and parameters θ. Figure 5 depicts the corresponding probabilistic graphic model. Additional details regarding the model architecture and hyperparameters are given in Appendix B.

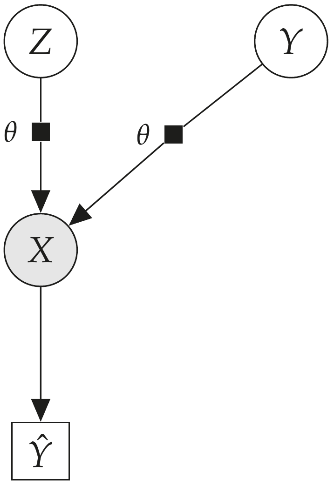


Figure 5. Probabilistic graphical model for the deep Bayes classifier.

To perform inference, a variational posterior was learned as an encoder $q_\phi(z|x, y)$ from a prior $p(z) \sim \mathcal{N}(0, I)$. Leveraging the conditional structure on y, we used the generative model as a classifier using the Bayes rule,

$$p(y|x) \propto p(y)p(x|y) = p(y) \int p_\theta(x|z, y)q_\phi(z|x, y)dz \approx \frac{1}{M} \sum_{m=1}^{M} p_\theta(x|z^{(m)}, y)p(y), \tag{8}$$

where we used M Monte Carlo samples $z^{(m)} \sim q_\phi(z|x, y)$. In the experiments, we set $M = 5$. Given a test sample x, the label $\hat{y}$ with the highest probability $p(y|x)$ is predicted.

For comparison, we performed several experiments changing T in the transition distribution $Q_{\eta,T}$ of the refined variational approximation. The results are given in Table 4, which reports the test accuracy at end of the refinement phase. Note that we are comparing different values of T depending on their use in refinement or inference phases (in the latter, the model and variational parameters were kept frozen). The model with $T_{ref} = 5$ was trained for 10 epochs, whereas the other settings were for 15 epochs, to give all settings a similar training time. Results were averaged over three runs with different random seeds. In all settings, we used the VIS-MC approximation for the entropy term. From the results, it is clear that the effect of using the refined variational approximation (the cases when $T > 0$) is crucially beneficial to achieve higher accuracy. The effect of learning a good initial distribution and inner learning rate by using the gradients $\nabla_{\phi}\text{ELBO}(q)$ and $\nabla_{\eta}\text{ELBO}(q)$ has a highly positive impact in the accuracy obtained.

On a final note, we have not included the case of only using an SGD or an SGLD sampler (i.e., without learning an initial distribution $q_{0,\phi}(z|x)$) since the results were much worse than those in Table 4 for a comparable computational budget. This strongly suggests that, for inference in high-dimensional, continuous latent spaces, learning a good initial distribution through VIS may accelerate mixing time dramatically.

Table 4. Results on digit classification task using a deep Bayes classifier.

T_{ref}	T_{inf}	**Acc. (Test)**
0	0	96.5 ± 0.5 %
0	10	97.7 ± 0.7 %
5	10	$\mathbf{99.8} \pm \mathbf{0.2}$ %

6. Conclusions

In this work, we have proposed a flexible and efficient framework to perform large-scale Bayesian inference in probabilistic models. The scheme benefits from useful properties and can be employed to efficiently perform inference with a wide class of models such as state-space time series, variational autoencoders and variants such as the conditioned VAE for classification tasks, defined through continuous, high-dimensional distributions.

The framework can be seen as a general approach to tuning MCMC sampler parameters, adapting the initial distributions and learning rate. Key to the success and applicability of the VIS framework are the ELBO approximations based on the introduced refined variational approximation, which are computationally cheap but convenient.

Better estimates of the refined density and its gradient may be a fruitful line of research, such as the spectral estimator used in [54]. Another alternative is to use a deterministic flow (such as SGD or SVGD), keeping track of the change in entropy at each iteration using the change of the variable formula, as in [55]. However, this requires a costly Jacobian computation, making it unfeasible to combine with our approach of back-propagation through the sampler (Section 3.3) for moderately complex problems. We leave this for future exploration. Another interesting and useful line of further research would be to tackle the case in which the latent variables z are discrete. This would entail adapting the automatic differentiation techniques to be able to back-propagate the gradients through the sequences of acceptance steps necessary in Metropolis–Hastings samplers.

In order to deal with the implicit variational density, it may be worthwhile to consider optimizing the Fenchel dual of the KL divergence, as in [31]. However, this requires the use of an auxiliary neural network, which may entail a large computational price compared with our simpler particle approximation.

Lastly, probabilistic programming offers powerful tools for Bayesian modeling. A PPL can be viewed as a programming language extended with random sampling and Bayesian conditioning capabilities, complemented with an inference engine that produces answers to inference, prediction and decision-making queries. Examples include WinBUGS [56], Stan [57] or the recent Edward [58] and Pyro [59] languages. We plan to adapt VIS into several PPLs to facilitate the adoption of the framework.

Author Contributions: Conceptualization, V.G. and D.R.I.; methodology, V.G.; software, V.G.; investigation, V.G.; writing—original draft preparation, V.G. and D.R.I.; writing—review and editing, V.G. and D.R.I. All authors have read and agreed to the published version of the manuscript.

Funding: VG acknowledges support from grant FPU16-05034. DRI is grateful to the MINECO MTM2017-86875-C3-1-R project and the AXA-ICMAT Chair in Adversarial Risk Analysis. Both authors acknowledge support from the Severo Ochoa Excellence Program CEX2019-000904-S. This material was based upon work partially supported by the National Science Foundation under Grant DMS-1638521 to the Statistical and Applied Mathematical Sciences Institute as well as a BBVA Foundation project.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Fokker-Planck Approximation (Vis-Fp)

The Fokker–Planck equation is a PDE that describes the temporal evolution of the density of a random variable under a (stochastic) gradient flow [39]. For a given SDE

$$dz = \mu(z,t)dt + \sigma(z,t)dB_t,$$

the corresponding Fokker–Planck equation is

$$\frac{\partial}{\partial t}q_t(z) = -\frac{\partial}{\partial z}[\mu(z,t)q_t(z)] + \frac{\partial^2}{\partial z^2}\left[\frac{\sigma^2(z,t)}{2}q_t(z)\right].$$

We are interested in converting the SGLD dynamics to a deterministic gradient flow.

Proposition A1. *The SGLD dynamics, given by the SDE*

$$dz = \nabla \log p(z)dt + \sqrt{2}dB_t,$$

have an equivalent deterministic flow, written as the ODE

$$dz = (\nabla \log p(z) - \nabla \log q_t(z))dt.$$

Proof. Let us write the Fokker–Planck equation for the respective flows. For the Langevin SDE, it is

$$\frac{\partial}{\partial t}q_t(z) = -\frac{\partial}{\partial z}\left[\nabla \log p(z)q_t(z)\right] + \frac{\partial^2}{\partial z^2}\left[q_t(z)\right].$$

On the other hand, the Fokker–Planck equation for the deterministic gradient flow is given by

$$\frac{\partial}{\partial t}q_t(z) = -\frac{\partial}{\partial z}\left[\nabla \log p(z)q_t(z)\right] + \frac{\partial}{\partial z}\left[\nabla \log q_t(z)q_t(z)\right].$$

The result immediately follows since $\frac{\partial}{\partial z}[\nabla \log q_t(z)q_t(z)] = \frac{\partial^2}{\partial z^2}[q_t(z)]$. □

Given that both flows are equivalent, we restrict our attention to the deterministic flow. Its discretization leads to iterations of the form

$$z_t = z_{t-1} + \eta(\nabla \log p(z_{t-1}) - \nabla \log q_{t-1}(z_{t-1})). \tag{A1}$$

In order to tackle the last term, we make the following particle approximation. Using a variational formulation, we have

$$-\nabla \log q(z) = \nabla\left(-\frac{\delta}{\delta q}\mathbb{E}_q[\log q]\right).$$

Then, we smooth the true density q convolving it with a kernel K, typically the rbf kernel, $K(z,z') = \exp\{-\gamma\|z-z'\|^2\}$, where γ is the bandwidth hyperparameter, leading to

$$\begin{aligned}\nabla\left(-\frac{\delta}{\delta q}\mathbb{E}_q[\log q]\right) &\approx \nabla\left(-\frac{\delta}{\delta q}\mathbb{E}_q[\log(q*K)]\right)\\ &= \nabla\log(q*K) - \nabla\left(\frac{q}{(q*K)}*K\right).\end{aligned}$$

If we consider a mixture of Dirac deltas, $q(z) = \frac{1}{M}\sum_{m=1}^{M}\delta(z-z_m)$, then the approximation is given by

$$-\nabla\log q(z) \approx -\frac{\sum_k \nabla_{z_m}K(z_m,z_n)}{\sum_n K(z_m,z_n)} - \sum_l \frac{\nabla_{z_m}K(z_m,z_l)}{\sum_n K(z_n,z_l)},$$

which can be inserted into Equation (A1). Finally, note that it is possible to back-propagate through this equation; i.e., the gradients of K can be explicitly computed.

Appendix B. Experiment Details

Appendix B.1. State-Space Models

Appendix B.1.1. Initial Experiments

For the HMM, both the observation and transition probabilities are categorical distributions, taking values in the domain $\{0,1,2,3,4\}$.

The equations of the DLM are

$$\begin{aligned}z_{t+1} &\sim \mathcal{N}(0.5z_t + 1.0, \sigma_{tr})\\ x_t &\sim \mathcal{N}(3.0z_t + 0.5, \sigma_{em}).\end{aligned}$$

with $z_0 = 0.0$.

Appendix B.1.2. Prediction Task in a DLM

The DLM model comprises a linear trend component plus a seasonal block with a period of 12. The trend is specified as

$$\begin{aligned}x_t &= z_{level,t} + \epsilon_t \qquad \epsilon_t \sim \mathcal{N}(0,\sigma_{obs})\\ z_{level,t} &= z_{level,t-1} + z_{slope,t-1} + \epsilon'_t \qquad \epsilon'_t \sim \mathcal{N}(0,\sigma_{level})\\ z_{slope,t} &= z_{slope,t-1} + \epsilon''_t \qquad \epsilon''_t \sim \mathcal{N}(0,\sigma_{slope}).\end{aligned}$$

With respect to the seasonal component, we specify it through

$$\begin{aligned}x_t &= Fz_t + v_t \qquad v_t \sim \mathcal{N}(0,\sigma_{obs})\\ z_t &= Gz_{t-1} + w_t \qquad w_t \sim \mathcal{N}(0,\sigma_{seas})\end{aligned}$$

where F is a 12-dimensional vector $(1,0,\ldots,0,0)$ and G is the 12×12 matrix

$$G = \begin{bmatrix} 0 & 0 & \ldots & 0 & 1\\ 1 & 0 & & 0 & 0\\ 0 & 1 & & 0 & 0\\ & & \ddots & & \\ 0 & 0 & & 1 & 0\end{bmatrix}.$$

Further details are in [60].

Appendix B.2. Vae

Appendix B.2.1. Model Details

The prior distribution $p(z)$ for the latent variables $z \in \mathbb{R}^{10}$ is a standard factorized Gaussian. The decoder distribution $p_\theta(x|z)$ and the encoder distribution (initial variational approximation) $q_{0,\phi}(z|x)$ are parameterized by two feed-forward neural networks, as detailed in Figure A1.

Appendix B.2.2. Hyperparameter Settings

The optimizer Adam is used in all experiments, with la earning rate of $\lambda = 0.001$. We also set $\eta = 0.001$. We train for 15 epochs (fMNIST) and 20 epochs (MNIST) to achieve a performance similar to the VAE in [24]. For the VIS-5-10 setting, we train only for 10 epochs to allow a fair computational comparison in terms of similar computing times.

Appendix B.3. cVAE

Appendix B.3.1. Model Details

The prior distribution $p(z)$ for the latent variables $z \in \mathbb{R}^{10}$ is a standard factorized Gaussian. The decoder distribution $p_\theta(x|y, z)$ and the encoder distribution (initial variational approximation) $q_{0,\phi}(z|x, y)$ are parameterized by two feed-forward neural networks whose details can be found in Figure A2. Equation (8) is approximated with one MC sample from the variational approximation in all experimental settings, as it allowed fast inference times while offering better results.

```
class VAE(nn.Module):
    def __init__(self):
        super(VAE, self).__init__()

        self.z_d = 10
        self.h_d = 200
        self.x_d = 28*28

        self.fc1_mu = nn.Linear(self.x_d, self.h_d)
        self.fc1_cov = nn.Linear(self.x_d, self.h_d)
        self.fc12_mu = nn.Linear(self.h_d, self.h_d)
        self.fc12_cov = nn.Linear(self.h_d, self.h_d)
        self.fc2_mu = nn.Linear(self.h_d, self.z_d)
        self.fc2_cov = nn.Linear(self.h_d, self.z_d)

        self.fc3 = nn.Linear(self.z_d, self.h_d)
        self.fc32 = nn.Linear(self.h_d, self.h_d)
        self.fc4 = nn.Linear(self.h_d, self.x_d)

    def encode(self, x):
        h1_mu = F.relu(self.fc1_mu(x))
        h1_cov = F.relu(self.fc1_cov(x))
        h1_mu = F.relu(self.fc12_mu(h1_mu))
        h1_cov = F.relu(self.fc12_cov(h1_cov))
        # we work in the logvar-domain
        return self.fc2_mu(h1_mu),
        torch.log(F.softplus(self.fc2_cov(h1_cov)))

    def decode(self, z):
        h3 = F.relu(self.fc3(z))
        h3 = F.relu(self.fc32(h3))
        return torch.sigmoid(self.fc4(h3))
```

Figure A1. Model architecture for the VAE.

```
class cVAE(nn.Module):
    def __init__(self):
        super(cVAE, self).__init__()

        self.z_d = 10
        self.h_d = 200
        self.x_d = 28*28
        num_classes = 10

        self.fc1_mu = nn.Linear(self.x_d + num_classes, self.h_d)
        self.fc1_cov = nn.Linear(self.x_d + num_classes, self.h_d)
        self.fc12_mu = nn.Linear(self.h_d, self.h_d)
        self.fc12_cov = nn.Linear(self.h_d, self.h_d)
        self.fc2_mu = nn.Linear(self.h_d, self.z_d)
        self.fc2_cov = nn.Linear(self.h_d, self.z_d)

        self.fc3 = nn.Linear(self.z_d + num_classes, self.h_d)
        self.fc32 = nn.Linear(self.h_d, self.h_d)
        self.fc4 = nn.Linear(self.h_d, self.x_d)

    def encode(self, x, y):
        h1_mu = F.relu(self.fc1_mu(torch.cat([x, y], dim=-1)))
        h1_cov = F.relu(self.fc1_cov(torch.cat([x, y], dim=-1)))
        h1_mu = F.relu(self.fc12_mu(h1_mu))
        h1_cov = F.relu(self.fc12_cov(h1_cov))
        # we work in the logvar-domain
        return self.fc2_mu(h1_mu),
        torch.log(F.softplus(self.fc2_cov(h1_cov)))

    def decode(self, z, y):
        h3 = F.relu(self.fc3(torch.cat([z, y], dim=-1)))
        h3 = F.relu(self.fc32(h3))
        return torch.sigmoid(self.fc4(h3))
```

Figure A2. Model architecture for the cVAE.

Appendix B.3.2. Hyperparameter Settings

The optimizer Adam was used in all experiments, with a learning rate of $\lambda = 0.01$. We set the initial $\eta = 5 \times 10^{-5}$.

References

1. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877. [CrossRef]
2. Insua, D.; Ruggeri, F.; Wiper, M. *Bayesian Analysis of Stochastic Process Models*; John Wiley & Sons: New York, NY, USA, 2012; Volume 978.
3. Alquier, P. Approximate Bayesian Inference. *Entropy* **2020**, *22*, 1272. [CrossRef] [PubMed]
4. Kucukelbir, A.; Tran, D.; Ranganath, R.; Gelman, A.; Blei, D.M. Automatic differentiation variational inference. *J. Mach. Learn. Res.* **2017**, *18*, 430–474.
5. Riquelme, C.; Johnson, M.; Hoffman, M. Failure modes of variational inference for decision making. In Proceedings of the Prediction and Generative Modeling in RL Workshop (AAMAS, ICML, IJCAI), Stockholm, Sweden, 15 July 2018.
6. Andrieu, C.; Doucet, A.; Holenstein, R. Particle Markov chain Monte Carlo methods. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2010**, *72*, 269–342. [CrossRef]
7. Neal, R.M. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*; CRC Press: Boca Raton, FL, USA, 2011; Volume 2, p. 2.
8. Van Ravenzwaaij, D.; Cassey, P.; Brown, S.D. A simple introduction to Markov Chain Monte–Carlo sampling. *Psychon. Bull. Rev.* **2018**, *25*, 143–154. [CrossRef] [PubMed]
9. Nalisnick, E.; Hertel, L.; Smyth, P. Approximate inference for deep latent gaussian mixtures. In Proceedings of the NIPS Workshop on Bayesian Deep Learning, Barcelona, Spain, 10 December 2016.
10. Salimans, T.; Kingma, D.; Welling, M. Markov chain Monte Carlo and variational inference: Bridging the gap. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1218–1226.
11. Tran, D.; Ranganath, R.; Blei, D.M. The variational Gaussian process. In Proceedings of the 4th International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016.
12. Wood, F.; Meent, J.W.; Mansinghka, V. A new approach to probabilistic programming inference. In Proceedings of the Artificial Intelligence and Statistics, Reykjavik, Iceland, 22–25 April 2014; pp. 1024–1032.
13. Ge, H.; Xu, K.; Ghahramani, Z. Turing: a language for flexible probabilistic inference. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Lanzarote, Spain, 9–11 April 2018; pp. 1682–1690.
14. Papaspiliopoulos, O.; Roberts, G.O.; Sköld, M. A general framework for the parametrization of hierarchical models. *Stat. Sci.* **2007**, *22*, 59–73. [CrossRef]

15. Hoffman, M.; Sountsov, P.; Dillon, J.V.; Langmore, I.; Tran, D.; Vasudevan, S. Neutra-lizing bad geometry in hamiltonian Monte Carlo using neural transport. *arXiv* **2019**, arXiv:1903.03704.
16. Li, S.H.; Wang, L. Neural Network Renormalization Group. *Phys. Rev. Lett.* **2018**, *121*, 260601. [CrossRef]
17. Parno, M.; Marzouk, Y. Transport map accelerated markov chain monte carlo. *arXiv* **2014**, arXiv:1412.5492.
18. Rezende, D.; Mohamed, S. Variational Inference with Normalizing Flows. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1530–1538.
19. Chen, C.; Li, C.; Chen, L.; Wang, W.; Pu, Y.; Carin, L. Continuous-Time Flows for Efficient Inference and Density Estimation. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 25–31 July 2018.
20. Liu, G.; Liu, Y.; Guo, M.; Li, P.; Li, M. Variational inference with Gaussian mixture model and householder flow. *Neural Netw.* **2019**, *109*, 43–55. [CrossRef]
21. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, USA, 8–13 December 2014; pp. 2672–2680.
22. Mandt, S.; Hoffman, M.D.; Blei, D.M. Stochastic Gradient Descent as Approximate Bayesian Inference. *J. Mach. Learn. Res.* **2017**, *18*, 4873–4907.
23. Huszár, F. Variational inference using implicit distributions. *arXiv* **2017**, arXiv:1702.08235.
24. Titsias, M.K.; Ruiz, F. Unbiased Implicit Variational Inference. In Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics, Naha, Japan, 16–18 April 2019; pp. 167–176.
25. Yin, M.; Zhou, M. Semi-Implicit Variational Inference. *arXiv* **2018**, arXiv:1805.11183.
26. Hoffman, M.D. Learning deep latent Gaussian models with Markov chain Monte Carlo. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 22–31 July 2017; pp. 1510–1519.
27. Feng, Y.; Wang, D.; Liu, Q. Learning to draw samples with amortized stein variational gradient descent. *arXiv* **2017**, arXiv:1707.06626.
28. Cremer, C.; Li, X.; Duvenaud, D. Inference suboptimality in variational autoencoders. *arXiv* **2018**, arXiv:1801.03558.
29. Ruiz, F.; Titsias, M. A Contrastive Divergence for Combining Variational Inference and MCMC. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 5537–5545.
30. Dai, B.; Dai, H.; He, N.; Liu, W.; Liu, Z.; Chen, J.; Xiao, L.; Song, L. Coupled variational bayes via optimization embedding. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, USA, 3–8 December 2018; pp. 9690–9700.
31. Fang, L.; Li, C.; Gao, J.; Dong, W.; Chen, C. Implicit Deep Latent Variable Models for Text Generation. *arXiv* **2019**, arXiv:1908.11527.
32. Welling, M.; Teh, Y.W. Bayesian learning via stochastic gradient Langevin dynamics. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Montreal, QC, USA, 11–13 June 2014; pp. 681–688.
33. Li, C.; Chen, C.; Carlson, D.; Carin, L. Preconditioned stochastic gradient Langevin dynamics for deep neural networks. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
34. Li, C.; Chen, C.; Fan, K.; Carin, L. High-order stochastic gradient thermostats for Bayesian learning of deep models. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
35. Abbati, G.; Tosi, A.; Osborne, M.; Flaxman, S. Adageo: Adaptive geometric learning for optimization and sampling. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Canary Islands, Spain, 9–11 April 2018; pp. 226–234.
36. Gallego, V.; Insua, D.R. Stochastic Gradient MCMC with Repulsive Forces. *arXiv* **2018**, arXiv:1812.00071.
37. Ma, Y.A.; Chen, T.; Fox, E. A complete recipe for stochastic gradient MCMC. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 2917–2925.
38. Baydin, A.G.; Pearlmutter, B.A.; Radul, A.A.; Siskind, J.M. Automatic differentiation in machine learning: A survey. *J. Mach. Learn. Res.* **2017**, *18*, 5595–5637.
39. Pavliotis, G. Stochastic Processes and Applications: Diffusion Processes, the Fokker-Planck and Langevin Equations. In *Texts in Applied Mathematics*; Springer: New York, NY, USA, 2014.
40. Liu, Q.; Wang, D. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In Proceedings of the Advances In Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 2378–2386.
41. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
42. Graves, T.L. Automatic step size selection in random walk Metropolis algorithms. *arXiv* **2011**, arXiv:1103.5986.
43. Brooks, S.; Gelman, A.; Jones, G.; Meng, X.L. *Handbook of Markov Chain Monte Carlo*; CRC Press: Boca Raton, FL, USA, 2011.
44. Murray, I.; Salakhutdinov, R. Notes on the KL-Divergence between a Markov Chain and Its Equilibrium Distribution; 2008. Available online: http://www.cs.toronto.edu/~rsalakhu/papers/mckl.pdf (accessed on 12 June 2020).
45. Franceschi, L.; Donini, M.; Frasconi, P.; Pontil, M. Forward and reverse gradient-based hyperparameter optimization. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 22–31 July 2017; pp. 1165–1173.
46. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Granada, Spain, 2019; pp. 8024–8035.
47. Rabiner, L.R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* **1989**, *77*, 257–286. [CrossRef]

48. Zarchan, P.; Musoff, H. *Fundamentals of Kalman filtering: A Practical Approach*; American Institute of Aeronautics and Astronautics, Inc.: Washington, DC, USA, 2013.
49. Gneiting, T.; Raftery, A.E. Strictly proper scoring rules, prediction, and estimation. *J. Am. Stat. Assoc.* **2007**, *102*, 359–378. [CrossRef]
50. Keeling, C.D. *Atmospheric Carbon Dioxide Record from Mauna Loa*; Scripps Institution of Oceanography, The University of California: La Jolla, CA, USA, 2005.
51. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. *arXiv* **2013**, arXiv:1312.6114.
52. LeCun, Y.; Cortes, C. MNIST handwritten Digit Database. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 12 May 2020).
53. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv* **2017**, arXiv:1708.07747.
54. Shi, J.; Sun, S.; Zhu, J. A Spectral Approach to Gradient Estimation for Implicit Distributions. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 25–31 July 2018; pp. 4651–4660.
55. Duvenaud, D.; Maclaurin, D.; Adams, R. Early stopping as nonparametric variational inference. In Proceedings of the Artificial Intelligence and Statistics, Cadiz, Spain, 9–11 May 2016; pp. 1070–1077.
56. Lunn, D.J.; Thomas, A.; Best, N.; Spiegelhalter, D. WinBUGS-a Bayesian modelling framework: Concepts, structure, and extensibility. *Stat. Comput.* **2000**, *10*, 325–337. [CrossRef]
57. Carpenter, B.; Gelman, A.; Hoffman, M.D.; Lee, D.; Goodrich, B.; Betancourt, M.; Brubaker, M.; Guo, J.; Li, P.; Riddell, A. Stan: A probabilistic programming language. *J. Stat. Softw.* **2017**, *76*. [CrossRef]
58. Tran, D.; Hoffman, M.W.; Moore, D.; Suter, C.; Vasudevan, S.; Radul, A. Simple, distributed, and accelerated probabilistic programming. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018; pp. 7609–7620.
59. Bingham, E.; Chen, J.P.; Jankowiak, M.; Obermeyer, F.; Pradhan, N.; Karaletsos, T.; Singh, R.; Szerlip, P.; Horsfall, P.; Goodman, N.D. Pyro: Deep Universal Probabilistic Programming. *arXiv* **2018**, arXiv:1810.09538.
60. West, M.; Harrison, J. *Bayesian Forecasting and Dynamic Models*; Springer: New York, NY, USA, 2006.

entropy

MDPI

Article

Dynamics of Coordinate Ascent Variational Inference: A Case Study in 2D Ising Models

Sean Plummer *, Debdeep Pati and Anirban Bhattacharya

Department of Statistics, Texas A&M University, College Station, TX 77843, USA; debdeep@stat.tamu.edu (D.P.); anirbanb@stat.tamu.edu (A.B.)
* Correspondence: snplmmr@stat.tamu.edu

Received: 3 September 2020; Accepted: 3 November 2020; Published: 6 November 2020

Abstract: Variational algorithms have gained prominence over the past two decades as a scalable computational environment for Bayesian inference. In this article, we explore tools from the dynamical systems literature to study the convergence of coordinate ascent algorithms for mean field variational inference. Focusing on the Ising model defined on two nodes, we fully characterize the dynamics of the sequential coordinate ascent algorithm and its parallel version. We observe that in the regime where the objective function is convex, both the algorithms are stable and exhibit convergence to the unique fixed point. Our analyses reveal interesting discordances between these two versions of the algorithm in the region when the objective function is non-convex. In fact, the parallel version exhibits a periodic oscillatory behavior which is absent in the sequential version. Drawing intuition from the Markov chain Monte Carlo literature, we empirically show that a parameter expansion of the Ising model, popularly called the Edward–Sokal coupling, leads to an enlargement of the regime of convergence to the global optima.

Keywords: bifurcation; dynamical systems; Edward–Sokal coupling; mean-field; Kullback–Leibler divergence; variational inference

1. Introduction

Variational Bayes (VB) is now a standard tool to approximate computationally intractable posterior densities. Traditionally this computational intractability has been circumvented using sampling techniques such as Markov chain Monte Carlo (MCMC). MCMC techniques are prone to be computationally expensive for high dimensional and complex hierarchical Bayesian models, which are prolific in modern applications. VB methods, on the other hand, typically provide answers orders of magnitude faster, as they are based on optimization. Introduction to VB can be found in chapter 10 of [1] and chapter 33 of [2]. Excellent recent surveys can be found in [3,4].

The objective of VB is to find the best approximation to the posterior distribution from a more tractable class of distributions on the latent variables that is well-suited to the problem at hand. The best approximation is found by minimizing a divergence between the posterior distribution of interest and a class of distributions that are computationally tractable. The most popular choices for the discrepancy and the approximating class are the Kullback–Leibler (KL) divergence and the class of product distributions, respectively. This combination is popularly known as mean field variational inference, originating from mean field theory in physics [5]. Mean-field inference has percolated through a wide variety of disciplines, including statistical mechanics, electrical engineering, information theory, neuroscience, cognitive sciences [6] and more recently deep neural networks [7]. While computing the KL divergence is intractable for a large class of distributions, reframing the minimization problem for maximizing the evidence lower bound (ELBO) leads to efficient algorithms. In particular, for conditionally conjugate-exponential family models, the optimal distribution for mean field variational inference can be computed by iteration of closed form updates. These updates form a coordinate ascent algorithm known as coordinate ascent variational inference (CAVI) [1].

Research into the theoretical properties of variational Bayes has exploded in the last few years. Recent theoretical work focuses on statistical risk bounds for variational estimate obtained from VB [8–11], asymptotic normality of VB posteriors [12] and extension to model misspecification [8,13]. While much of the recent theoretical work focuses on statistical optimality guarantees, there has been less work studying the convergence of the CAVI algorithms employed in practice. Convergence of CAVI to the global optima is only known in special cases that depend heavily on model structure for normal mixture models [14,15]; stochastic block models [16–19]; topic models [20]; and under special restrictions of the parameter regime, Ising models [21,22]. The convergence properties of the CAVI algorithm still largely constitute an open problem.

The goal of this work is to suggest a general systematic framework for studying convergence properties of CAVI algorithms. By viewing CAVI as a discrete time dynamical system, we can leverage dynamical systems theory to analyze the convergence behavior of the algorithm and bifurcation theory to study the types of changes that solutions can undergo as the various parameters are varied. For sake of concreteness, we focus on the 2D Ising model. While dynamical systems theory possesses the tools [23–25] necessary to analyze higher dimensional systems, they were mainly developed for non-sequential systems. The general theory for n-dimensional discrete dynamical systems is dependent on having the evolution function in the form $x_{n+1} = F(x_n)$. Deriving this F is typically not possible for densely connected higher dimensional sequential systems. The 2D Ising model has the special property that both the sequential and parallel updates in the two variables case can be written as two separate one variable dynamical systems, allowing for a simplified analysis. Our contributions to the literature are as follows: We provide a complete classification of the dynamical properties of the the traditional sequential update CAVI algorithm, and a parallelized version of the algorithm using dynamical systems and bifurcation theory on the Ising models. Our findings show that the sequential CAVI algorithm and the parallelized version have different convergence properties. Additionally, we numerically investigated the convergence of the CAVI algorithm on the Edward–Sokal coupling, a generalization of the Ising model. Our findings suggest that couplings/parameter expansion may provide a powerful way of controlling the convergence behavior of the CAVI algorithm, beyond the immediate example considered here.

2. Mean-Field Variational Inference and the Coordinate Ascent Algorithm

In this section, we briefly introduce mean-field variational inference for a target distribution in the form of a Boltzmann distribution with potential function Ψ,

$$p(\mathbf{x}) = \frac{\exp\{\Psi(\mathbf{x})\}}{\mathcal{Z}}, \quad \mathbf{x} \in \mathcal{X},$$

where $\mathcal{Z}$ denotes the intractable normalizing constant. The above representation encapsulates both posterior distributions that arise in Bayesian inference, where Ψ is the log-posterior up to constants, and probabilistic graphical models such as the Ising and Potts models. For instance, $\Psi(x) = \beta \sum_{u \sim v} J_{uv} x_u x_v + \beta \sum_u h_u x_u$ for the Ising model; see the next section for more details. Many of the complications in inference arise from the intractability of the normalizing constant $\mathcal{Z}$, which is commonly referred to as the free energy in probabilistic graphical models, and the marginal likelihood or evidence in Bayesian statistics. Variational inference aims to mitigate this problem by using optimization to find the best approximation q^* to the target density p from a class $\mathcal{F}$ of variational distributions over the parameter vector $\mathbf{x}$,

$$q^* = \arg\min_{q \in \mathcal{F}} D(q \,||\, p) \tag{1}$$

where $D(q\,||\,p)$ denotes the Kullback–Leibler (KL) divergence between q and p. The complexity of this optimization problem is largely determined by the choice of variational family $\mathcal{F}$. The objective function of the above optimization problem is intractable because it also involves the evidence $\mathcal{Z}$. We can work around this issue by rewriting the KL divergence as

$$D(q\,||\,p) = \mathbb{E}_q[\log q] - \mathbb{E}_q[\Psi] + \log \mathcal{Z} \tag{2}$$

where $\mathbb{E}_q$ denotes the expectation with respect to $q(\mathbf{x})$. Rearranging terms,

$$\begin{aligned} \log \mathcal{Z} &= D(q\,||\,p) + \mathbb{E}_q[\Psi] - \mathbb{E}_q[\log q] && (3)\\ &\geq \mathbb{E}_q[\Psi] - \mathbb{E}_q[\log q] := \text{ELBO}(q). && (4)\end{aligned}$$

The acronym ELBO stands for evidence lower bound and the nomenclature is now apparent from the above inequality. Notice from Equation (2) that maximizing the ELBO is equivalent to minimizing the KL divergence. By maximizing the ELBO we can solve the original variational problem while by-passing the computational intractability of the evidence.

As mentioned above, the choice of variational family controls both the complexity and accuracy of approximation. Using a more flexible family achieves a tighter lower bound but at the cost of having to solve a more complex optimization problem. A popular choice of family that balances both flexibility and computability is the mean-field family. Mean-field variational inference refers to the situation when q is restricted to the product family of densities over the parameters,

$$\mathcal{F}_{\text{MF}} := \{q(\mathbf{x}) = q_1(x_1) \otimes \cdots \otimes q_n(x_n) \text{ for probability measures } q_j, j = 1, \ldots, n\}, \tag{5}$$

The coordinate ascent variational inference (CAVI) algorithm (refer to Algorithm 1) is a learning algorithm that optimizes the ELBO over the mean-field family $\mathcal{F}_{\text{MF}}$. At each time step $t \geq 1$, the CAVI algorithm iteratively updates the current mean field marginal distribution $q_j^{(t)}(x_j)$ by maximizing the ELBO over that marginal while keeping the other marginals $\{q_\ell^{(t)}(x_\ell)\}_{\ell \neq j}$ fixed at their current values. Formally, we update the current distribution $q^{(t)}(\mathbf{x})$ to $q^{(t+1)}(\mathbf{x})$ by the updates,

$$\begin{aligned} q_1^{(t+1)}(x_1) &= \arg\max_{q_1} \text{ELBO}(q_1 \otimes q_2^{(t)} \otimes \cdots \otimes q_n^{(t)}) \\ q_2^{(t+1)}(x_2) &= \arg\max_{q_2} \text{ELBO}(q_1^{(t+1)} \otimes q_2 \otimes q_3^{(t)} \otimes \cdots \otimes q_n^{(t)}) \\ &\vdots \\ q_n^{(t+1)}(x_n) &= \arg\max_{q_n} \text{ELBO}(q_1^{(t+1)} \otimes \cdots \otimes q_{n-1}^{(t+1)} \otimes q_n). \end{aligned}$$

Algorithm 1 Coordinate ascent variational inference (CAVI).

```
Input: Model $p(\mathbf{x}) = \exp(\Psi(\mathbf{x}) - \log \mathcal{Z})$
Output: A variational density $q(\mathbf{x}) = \prod_{j=1}^n q_j(x_j)$
Initialize: variational densities $q_j(x_j)$
while $ELBO(q)$ not converged do
    for $j \in \{1, \ldots, n\}$ do
        $q_j(x_j) \propto \exp\{\mathbb{E}_{-j}[\Psi(\mathbf{x})]\}$
    end
    Compute $\text{ELBO}(q) = \mathbb{E}_q[\Psi(\mathbf{x})] - \mathbb{E}_q[\log q(\mathbf{x})]$
end
return $q(x)$
```

The objective function $\text{ELBO}(q_1 \otimes \cdots \otimes q_n)$ is concave in each of the arguments individually (although it is rarely jointly concave), so these individual maximization problems have unique solutions. The optimal update for the jth mean field variational component of the model has the closed form,

$$q_j^*(\mathbf{x}_j) \quad \propto \quad \exp\left\{\mathbb{E}_{-j}\left[\Psi(\mathbf{x})\right]\right\}$$

where the expectations $\mathbb{E}_{-j}$ are taken with respect to the distribution $\prod_{i \neq j} q_i(\mathbf{x}_i)$. Furthermore, the update step of the algorithm is monotonous, as each step of the CAVI increases the objective function

$$\begin{aligned}\text{ELBO}(q_1^{(t+1)} \otimes q_2^{(t+1)} \otimes \cdots \otimes q_n^{(t+1)}) \quad &\geq \text{ELBO}(q_1^{(t+1)} \otimes q_2^{(t+1)} \otimes \cdots \otimes q_{n-1}^{(t+1)} \otimes q_n^{(t)}) \geq \cdots \geq \\ &\text{ELBO}(q_1^{(t)} \otimes q_2^{(t)} \otimes \cdots \otimes q_n^{(t)}).\end{aligned}$$

For parametric models, the sequential updates of the variational marginal distributions in the CAVI algorithm is done by a sequential update of the variational parameters of these distributions. The CAVI algorithm updates for parametric models induce a discrete time dynamical system of the parameters. Clearly, convergence of the CAVI algorithm can be framed in terms of this induced discrete time dynamical system. As discussed before, the ELBO is generally a non-convex function, and hence the CAVI algorithm is only guaranteed to converge to a local optimum of the system. It is also not clear how many local optima (or fixed points) the system has, nor whether the algorithm always settles on a single fixed point, diverges away from the fixed point or cycles between multiple fixed points. These questions translate to questions about the existence and stability of fixed points of the induced dynamical system. We are also interested in how the behavior of the CAVI algorithm could possibly change as we vary the parameters of the model. This translates to questions about the possible bifurcations of the induced dynamical system. In Section 3, we formally introduce the Ising model and its mean-field variational inference.

3. CAVI in Ising Model

We first briefly review the definition of an Ising model. The Ising model was first introduced as a model for magnetization in statistical physics, but has found many applications in other fields; see [26] and references therein. The Ising model is a probability distribution on the hypercube $\{\pm 1\}^n$ given by

$$p(\mathbf{x}) \quad \propto \quad \exp\left[\beta \sum_{u \sim v} J_{uv} x_u x_v + \beta \sum_u h_u x_u\right], \tag{6}$$

where the interaction matrix J is a symmetric real $n \times n$ matrix with zeros on the diagonal, h is a real n-vector that represents the external magnetic field, and β is the inverse temperature parameter. The model is said to be ferromagnetic if $J_{uv} \geq 0$ for all u, v and anti-ferromagnetic if $J_{uv} < 0$ for all u, v. The normalizing constant or the partition function of the Ising model is

$$\mathcal{Z} \quad = \sum_{\mathbf{x} \in \{\pm 1\}^n} \exp\left[\beta \sum_{u \sim v} J_{uv} x_u x_v + \beta \sum_u h_u x_u\right].$$

Refer to Chapter 31 of [2] for an excellent review of Ising models.

Mean Field Variational Inference in Ising Model

Here we provide a derivation of the CAVI update function for the Ising model, focusing on the two nodes ($n = 2$) case for simplicity and analytic tractability.

Notice $\log p(\mathbf{x}) := \beta\mathcal{H}(\mathbf{x}) = \beta\sum_{u\sim v} J_{uv}x_u x_v + \beta\sum_u h_u x_u$. In this case, we have the Ising model on two spins with $\mathbf{x} = (x_1, x_2)$ and influence matrix J with off diagonal term J_{12} and external magnetic field $h = (h_1, h_2) = (0, 0)$. From the general framework in Section 2, the CAVI updates are given by,

$$q_j^*(x_j) \propto \exp\left\{\mathbb{E}_{-j}\left[\beta(J_{12}x_1x_2 + h_1x_1 + h_2x_2)\right]\right\}.$$

Equivalently, the same updates are obtained by setting the gradient of the ELBO as a function of (x_1, x_2) equal to the $(0, 0)'$ vector. Illustrations of the ELBO and the gradient functions for various values of β are in Figures 1 and 2 respectively.

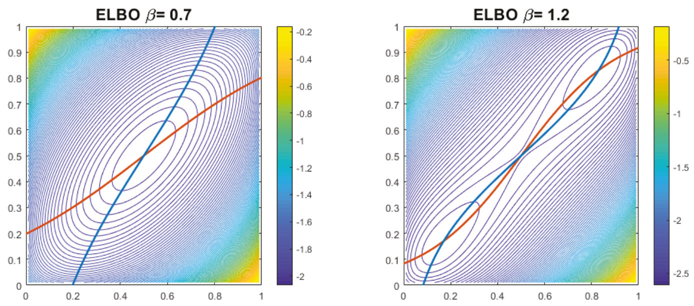


Figure 1. A contour plot of the ELBO as a function of x_1 and x_2 for $\beta = 0.7$ (**left**) and $\beta = 1.2$ (**right**) together with the optimal update functions for x_1 (**orange**) and x_2 (**blue**) given in Equation (8). For $\beta = 0.7$ the ELBO is a convex function and has exactly one optima, the global maximum, at $(0.5, 0.5)$. For $\beta = 1.2$ the ELBO is now a nonconvex function and has three optima at $(0.5, 0.5)$, $(0.17071, 0.17071)$ and $(0.82928, 0.82928)$.

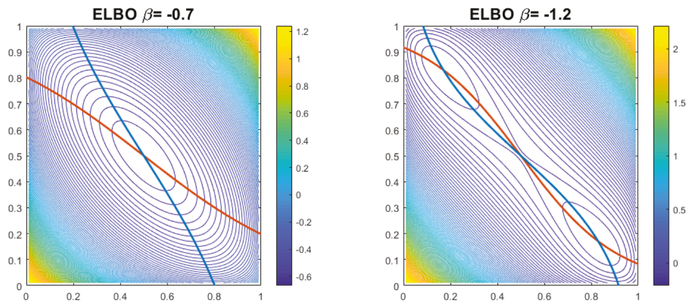


Figure 2. A contour plot of the ELBO as a function of x_1 and x_2 for $\beta = -0.7$ (**left**) and $\beta = -1.2$ (**right**) together with the optimal update functions for x_1 (**orange**) and x_2 (**blue**) given in Equation (8). For $\beta = -0.7$ the ELBO is a convex function and has exactly one optima, the global maximum, at $(0.5, 0.5)$. For $\beta = -1.2$ the ELBO is now a nonconvex function and has three optima at $(0.5, 0.5)$, $(0.17071, 0.82928)$ and $(0.82928, 0.17071)$.

Since q_1^* and q_2^* are two point distributions, it is sufficient to keep track of the mass assigned to 1. Simplifying,

$$\begin{aligned}
q_1^*(x_1) &\propto \exp\left\{\mathbb{E}_2\left[\log p(x_1, x_2)\right]\right\} \\
&= \exp\left\{\beta\mathcal{H}(x_1, x_2 = 1)q_2(x_2 = 1) + \beta\mathcal{H}(x_1, x_2 = -1)q_2(x_2 = -1)\right\} \\
&= \exp\left\{(\beta J_{12}x_1 + \beta h_1 x_1 + \beta h_2)\xi + (-\beta J_{12}x_1 + \beta h_1 x_1 - \beta h_2)(1 - \xi)\right\} \\
&= \exp\left\{(2\xi - 1)(\beta J_{12}x_1 + \beta h_2) + \beta h_1 x_1\right\},
\end{aligned}$$

where $\xi = q_2(x_2 = 1)$. Therefore

$$\begin{aligned} q_1^*(x_1 = 1) &= \frac{\exp\{(2\xi - 1)(\beta J_{12} + \beta h_2) + \beta h_1\}}{\exp\{(2\xi - 1)(\beta J_{12} + \beta h_2) + \beta h_1\} + \exp\{(2\xi - 1)(-\beta J_{12} + \beta h_2) - \beta h_1\}} \\ &= \frac{1}{1 + \exp\{-2\beta J_{12}(2\xi - 1) - 2\beta h_1\}}. \end{aligned}$$

Similarly denoting $\zeta = q_1(x_1 = 1)$,

$$\begin{aligned} q_2^*(x_2 = 1) &= \frac{\exp\{(2\zeta - 1)(\beta J_{12} + \beta h_1) + \beta h_2\}}{\exp\{(2\zeta - 1)(\beta J_{12} + \beta h_1) + \beta h_2\} + \exp\{(2\zeta - 1)(-\beta J_{12} + \beta h_1) - \beta h_2\}} \\ &= \frac{1}{1 + \exp\{-2\beta J_{12}(2\zeta - 1) - 2\beta h_2\}}. \end{aligned}$$

Let ζ_k (resp. ξ_k) denote the kth iterate of $q_1(x_1 = 1)$ (resp. $q_2(x_2 = 1)$) from the CAVI algorithm. To succinctly represent these updates, define the logistic sigmoid function

$$\sigma(u, \beta) = \frac{1}{1 + e^{-\beta u}}, \quad u \in [0, 1], \quad \beta \in \mathbb{R}. \tag{7}$$

With this notation, we have, for any $k \in \mathbb{Z}_+$,

$$\begin{aligned} \zeta_{k+1} &= \sigma(J_{12}(2\xi_k - 1) + h_1, 2\beta) \\ \xi_{k+1} &= \sigma(J_{12}(2\zeta_{k+1} - 1) + h_2, 2\beta). \end{aligned} \tag{8}$$

Without loss of generality we henceforth set $J_{12} = 1$. Under this choice the model is in the ferromagnetic regime for $\beta > 0$ and the anti-ferromagnetic regime for $\beta < 0$.

4. Why the Ising Model: A Summary of Our Contributions

There are exactly two cases of the Ising model that have a full analytic solution for the free energy. They are (i) the one dimensional line graph solved by Ernst Ising in his thesis [27] and (ii) the two dimensional case on the anisotropic square lattice when the magnetic field $h = 0$ by [28]. Comparison with the mean field solution for the same models highlights the poor approximation quality of the mean field solution in low dimensions. To the best knowledge of the authors, there are no results in the literature detailing the properties of the mean field solution to the anti-ferromagnetic Ising model. Readers not familiar with the physics may wonder why this is the case. To explain this, there are two cases in the anti-ferromagnetic regime: one of the two regions is equivalent to the ferromagnetic case and in the other the mean field approximation is not a good approximation of the system. The first case occurs in a bipartite graph where a transformation of variables makes the antiferromagnetic regime equivalent to the ferromagnetic one [29]. The other case can be seen on the triangle graph. By fixing the spin of one vertex as 1 and the other as -1, the third vertex becomes geometrically frustrated and neither choice of spin lowers the energy level of the system and the two configurations are equivalent [30]. In this case the mean field approximation gives a completely incorrect answer and does not merit further investigation from a qualitative point of view. The physics literature is primarily concerned with using the mean field solutions to the Ising model to estimate important physical constants of the systems. These constants are only meaningful when the mean field solution provides a good approximation to the behavior of the system in large dimensions. It is known, however, that under certain conditions the mean field approximation does indeed converge to the true free energy of the system as the dimension increases [21,31].

Our work is focused on providing a rigorous methodology to analyze dynamics of the CAVI algorithm that can be applied to any model structure. All of the interesting behaviors exhibited by the CAVI algorithm fit into the classical mathematical framework of discrete dynamical systems and bifurcation theory. Specifically, we use the Ising model as a simple and yet rich example to illustrate the potential of dynamical systems theory to analyze CAVI updates for mean field variational inference. The bifurcation of the ferromagnetic Ising model at the boundary of the Dobrushin regime is known [2,26]; however, a rigorous proof in terms of dynamical systems theory is missing in the literature.

There are several features that make the CAVI algorithm on the Ising model a nontrivial example worth investigating. The optimization problem arising from mean field variational inference on the Ising model is, in general, non-convex [21]. However, it is straightforward to obtain sufficient conditions to guarantee the existence of a global optima. One such condition is that the inverse temperature β is inside the Dobrushin regime, $|\beta| < 1$ [21]. Inside the Dobrushin regime, the CAVI update equations form a contraction mapping guaranteeing a unique global optima [21]. Outside of this regime the behavior of the CAVI algorithm is nontrivial. The CAVI solution to the Ising model with zero external magnetic field exhibits multiple local optima outside of the Dobrushin regime [2].

Our contributions to the literature are as follows. We utilize tools from dynamical systems theory to rigorously classify the full behavior of Ising model for the full parameter regime in dimension $n = 2$ for both the sequential and parallel versions of CAVI algorithm. We show that the dynamical behavior of the sequential CAVI is not equivalent to the behavior of the parallel CAVI. Lastly we derive a variational approximation to the Edward-Sokal parameter expansion of the Potts and Random Cluster models and numerically study its convergence behavior under the CAVI algorithm. Our numerical results reveal that the parameter expansion leads to an enlargement of the regime of convergence to the global optima. In particular the Dobrushin regime is strictly contained in the expanded regime. This is compatible with the analogous results in Markov chain literature. See the introduction of [32] for a well written summary of Markov chain mixing in the Ising model.

Statistical Significance of Our Results

Although mean-field variational inference has been routinely used in applications [3] for computational efficiency, it may not yield statistically optimal estimators. A statistically optimal estimator should correctly recover the statistical properties of the true distribution. Ideally, we would like the estimate to recover the true mean and true covariance of the distribution. It is well known that mean-field variational inference produces estimators that underestimate the posterior covariance [14]. More recently, it was shown that the mean-field estimators for certain topic models and stochastic block models may not even be correlated with the true distribution [17,20]. For these reasons, it is important to see if the mean field estimators can at least recover the true mean for various $\beta \in \mathbb{R}$.

Mean field inference approximates the joint probability mass function in (6) for $n = 2$ by product of two distributions on $\{-1, 1\}$ in the sense of Kullback–Leibler divergence. As discussed in Section 3, minimizing this divergence is equivalent to maximizing an objective function, called the Evidence Lower Bound (ELBO). Our objective is to better understand the relation between the CAVI estimate and the global maximum of ELBO in (6) when $n = 2$ and $h = 0$. Ideally, we want the global maximum of the ELBO to be a statistically reliable estimate. To understand this, let us denote $2 \times \text{Bernoulli}(p) - 1$ by $\langle 1, -1; p\rangle$. As the marginal distributions of (6) are both equal to $\langle 1, -1; 0.5\rangle$, we want the ELBO to be maximized at this value. From an algorithmic perspective, we would like to ensure that the CAVI iterates converge to this global maximum. The synergy of these two phenomena leads to a successful variational inference method. We show in this article that both these conditions can be violated in a certain regime of the parameter space in the context of Ising model on two nodes. Inside the Dobrushin regime ($-1 \leq \beta \leq 1$), the global optima of the ELBO obtained from a mean field inference occurs at $(\langle 1, -1; 0.5\rangle, \langle 1, -1; 0.5\rangle)$ which is qualitatively the optimal solution. In this regime, the CAVI system converges to this global optimum irrespective of where the system is initialized. Thus, in the Dobrushin regime, the mean field inference yields the statistically optimal estimate. Additionally, the CAVI algorithm is stable and convergent at this value. Unfortunately, this property deteriorates outside of the Dobrushin regime. Outside of the regime, the global maxima occur at two symmetric points which are different from $(\langle 1, -1; 0.5\rangle, \langle 1, -1; 0.5\rangle)$. These two symmetric points are equivalent under label switching. For example, when $\beta = 1.2$ one of the optima is $(\langle 1, -1; 0.17071\rangle, \langle 1, -1; 0.17071\rangle)$ and the other is $(\langle 1, -1; 0.82928\rangle, \langle 1, -1; 0.82928\rangle)$. Notice this second optima is equivalent to the sign swapped version $(\langle -1, 1; 0.17071\rangle, \langle -1, 1; 0.17071\rangle)$.

The original optima $(\langle 1, -1; 0.5\rangle, \langle 1, -1; 0.5\rangle)$ is actually a local minimum of the ELBO outside the Dobrushin regime. We illustrate in our theory that the CAVI system returns one of two global maxima of the objective function depending on the initialization of the algorithm. Although it is widely known that the statistical quality of the mean field inference is poor outside the regime, we show in addition that the algorithm itself exhibits erratic behavior and may not converge to the global maximizer of the ELBO for all initializations. Interestingly, outside the Dobrushin regime, the statistically optimal solution $(\langle 1, -1; 0.5\rangle, \langle 1, -1; 0.5\rangle)$ is a repelling fixed point of the CAVI system. This means that as the system is iterated, the current value of the system is pulled away from $(\langle 1, -1; 0.5\rangle, \langle 1, -1; 0.5\rangle)$ to the global maximum.

A common technique to further improve computational time is the use of block updates in the CAVI algorithm, meaning groups of parameters are updated simultaneously. We refer to this as the parallelized CAVI algorithm. This has been shown to work well in certain models [17], but has not been investigated in a general setting. However, it turns out that block updating in the Ising model can lead to new problematic behaviors. Outside the Dobrushin regime, block updates can exhibit non-convergence in the form of cycling. As the system updates, it eventually switches back and forth between two points that yield the same value in the objective function.

Parameter expansions (coupling) is another method of improving the convergence properties of algorithms. In the Markov chain theory for Ising models, it is well-known that mixing and convergence time are typically improved by using the Edward–Sokal coupling, a parameter expansion of the ferromagnetic Ising model [33]. Our preliminary investigation reveals that the convergence properties of the CAVI algorithm also exhibit a similar phenomenon.

5. Main Results

In this section, we analyze the behavior of the dynamical systems that one can form using the CAVI update equations and show that the behaviors of the systems differ. Our results are heavily dependent on well-known techniques in dynamical systems. For readers unfamiliar with some of technical terminology below, we have included a primer on the basics of dynamical systems in Appendix A.

Recall the system of sequential updates, which are the updates used in CAVI:

$$\zeta_{k+1} = \sigma(2\xi_k - 1, 2\beta), \quad \xi_{k+1} = \sigma(2\zeta_{k+1} - 1, 2\beta), \tag{9}$$

and the parallel updates:

$$\zeta_{k+1} = \sigma(2\xi_k - 1, 2\beta), \quad \xi_{k+1} = \sigma(2\zeta_k - 1, 2\beta). \tag{10}$$

We will show that these two systems are not topologically conjugate. We first state and prove some results on the dynamics of the sigmoid function (7). These results will be used as building blocks to study the dynamics of (9) and (10). Phase change behavior of dynamical systems using the sigmoid and RELU activation functions are known in the literature in the context of generalization performance of deep neural networks [34,35]. In this section we present a complete proof of the bifurcation analysis of non-linear dynamical systems involving sigmoid activation function despite its connections with [34,35]. Our results in Section 5.1 provide a more complete picture of the behavior of the dynamics in all regimes and can be readily exploited to analyze the dynamics of (9) and (10).

5.1. Sigmoid Function Dynamics

In this section we provide a full classification for the dynamics of the following sigmoid function and its second iterate,

$$\sigma(2x-1,2\beta), \tag{11}$$
$$\sigma(2\sigma(2x-1,2\beta)-1,2\beta). \tag{12}$$

To the best of our knowledge, we could not find a formal proof of the full classification of the dynamics of the sigmoid function (or its second iterate) for all $\beta \in \mathbb{R}$ in the literature. Additionally, it provides an introductory example to demonstrate the concepts and techniques of dynamical systems. We begin by using numerical techniques to determine the number of fixed points in the system and its possible periodic behavior. We then proceed by providing a formal proof of the full dynamical properties of (11) in Theorem 1 and the full dynamical properties of (12) in Theorem 2.

Using numerical techniques, we solve for the number of fixed points of the system. The number of fixed points the function (11) depends on the magnitude of the parameter. For $\beta > 0$, there is no periodic behavior, so there are no additional fixed points in (12) that are not fixed points in (11). For $-1 \leq \beta \leq 1$, there is a single fixed point at $x_* = 1/2$ and for $\beta > 1$, there are 3 fixed points $c_0(\beta), 1/2, c_1(\beta)$ in the interval $[0,1]$. These fixed points satisfy $0 \leq c_0(\beta) < 1/2 < c_1(\beta) \leq 1$, $c_0(\beta) \to 0$ and $c_1(\beta) \to 1$ as $\beta \to \infty$. For $\beta < 0$, we see periodic behavior in the system; there are fixed points of (12) that are not fixed points of (11). For $\beta < -1$, the function (11) has one fixed point at $x_* = 1/2$ and a periodic cycle $C = \{c_0(\beta), c_1(\beta)\}$. Both $c_0(\beta), c_1(\beta)$ are fixed points of (12) and these points are the same fixed points from the $\beta > 0$ regime as (12) is an even function with respect to β.

Table 1 denotes the values of the derivatives at the fixed point $1/2$ for $\beta = \pm 1$.

Table 1. Partial derivatives of (11) and (12) at fixed point $x_* = 1/2$ for parameter value $\beta = \pm 1$. The derivatives of the the function (11) are denoted using σ and the derivatives for (12) are denoted using σ^2.

	σ_x	σ_{xx}	σ_{xxx}	σ_β	$\sigma_{\beta x}$	σ^2_x	σ^2_{xx}	σ^2_{xxx}	σ^2_β	$\sigma^2_{\beta x}$
$\beta = 1$	1	0	-8	0	$1/2$	1	0	-16	0	1
$\beta = -1$	-1	0	8	0	$1/2$	1	0	-16	0	-1

We now have enough information to provide a complete classification of the dynamics of the sigmoid function.

Theorem 1 (Dynamics of sigmoid function). *Consider the discrete dynamical system generated by (11)*

$$x \mapsto \sigma(2x-1,2\beta) = \frac{1}{1+e^{-2\beta(2x-1)}}.$$

The full dynamics of the system (11) are as follows

1. *For $-1 \leq \beta \leq 1$, the system has a single hyperbolic fixed point $x_* = 1/2$ which is a global attractor and there are no p-periodic points for $p \geq 2$.*
2. *For $\beta > 1$, the system has one repelling hyperbolic fixed point $x_* = 1/2$ and two hyperbolic stable fixed points c_0, c_1, with $0 < c_0 < 1/2 < c_1 < 1$, and stable sets $W^s(c_0) = [0,1/2)$, $W^s(c_1) = (1/2,1]$. There are no p-periodic points for $p \geq 2$.*
3. *For $\beta < -1$, the system has one unstable hyperbolic fixed point $x_* = 1/2$, and a stable 2-cycle $\mathcal{C} = \{c_0, c_1\}$ with stable set $W^s(\mathcal{C}) = [0,1/2) \cup (1/2,1]$, with $0 < c_0 < 1/2 < c_1 < 1$. There are no p-periodic points for $p > 2$.*

4. *For $|\beta| = 1$, the system has one non-hyperbolic fixed point at $x_* = 1/2$ which is asymptotically stable and attracting.*

   The system undergoes a PD bifurcation at $\beta = -1$ and a pitchfork bifurcation at $\beta = 1$.

Proof. We will break the proof up into three parts. The first part of the proof is a linear stability analysis of the system, the second part is a stability analysis of the periodic points in the system and the third part is an analysis of the bifurcations of the system. We begin with a linear stability analysis of the system at each fixed point. For $\beta \leq 1$ the system has one fixed point $x_* = 1/2$ and for $\beta > 1$ the system has three fixed points $c_0, 1/2, c_1$. The derivative of $\sigma(2x-1, 2\beta)$ is $\sigma_x(2x-1, 2\beta) = -4\beta\sigma(2x-1, 2\beta)(1-\sigma(2x-1, 2\beta))$.

Fixed point $x_* = 1/2$ **:** The Jacobian of the system at the fixed point $x_* = 1/2$ is $\sigma_x(2x_*-1, 2\beta) = \beta$. For $\beta \neq 1$, the fixed point $x_* = 1/2$ is hyperbolic and for $\beta = \pm 1$ the fixed point is non-hyperbolic. We classify the stability of the hyperbolic fixed point $x_* = 1/2$ using Theorem A2. For $|\beta| < 1$ the fixed point $x_* = 1/2$ is globally attracting as $|\sigma_x(2x_*-1, 2\beta)| < 1$ and for $|\beta| > 1$ the fixed point $x_* = 1/2$ is globally repelling as $|\sigma_x(2x_*-1, 2\beta_*)| > 1$. For $\beta = \pm 1$ we invoke Theorem A3 to check for stability of the fixed point. At $\beta = -1$ we have $\sigma_x(2x_*-1, 2\beta) = -1$ and we need to check the Schwarzian derivative. The fixed point $x_* = 1/2$ is asymptotically stable for $\beta = -1$ by Theorem A3, as $\mathcal{S}\sigma(2\sigma(2x-1, 2\beta) - 1, 2\beta) \mid_{x=x_*} = -8$. For $\beta = 1$ we have $\sigma_x(2x_*-1, 2\beta) = 1$ and we need to check the second and third derivatives at the fixed point. The fixed point $x_* = 1/2$ is asymptotically stable when $\beta = 1$ by Theorem A3 as $\sigma_{xx}(2x_*-1, 2\beta) = 0$ and $\sigma_{xxx}(2x_*-1, 2\beta) = -8$.

Fixed points c_0, c_1 **:** These fixed points have the same behavior so we have grouped them together in the analysis. When $\beta > 1$ there are two additional fixed points c_0, c_1 of the system, both are attracting fixed points by Theorem A2 as $|\sigma_x(2c_i - 1, 2\beta)| < 1$ for each $i = 0, 1$ and all $\beta > 1$. The stable sets are $W^s(c_0) = [0, 1/2)$ and $W^s(c_1) = (1/2, 1]$.

Periodic points: For $\beta < -1$ we see the two cycle $\mathcal{C} = \{c_0, c_1\}$. Notice $\sigma(2c_0 - 1, 2\beta) = c_1$ and $\sigma(2c_1 - 1, 2\beta) = c_0$. This two cycle is stable since c_0 and c_1 are both stable fixed points of (12). The stable set is $W^s(\mathcal{C}) = [0, 1/2) \cup (1/2, 1]$, $\quad 0 < c_0 < 1/2 < c_1 < 1$.

At $(x_*, \beta_*) = (1/2, 1)$ the system under goes a pitchfork bifurcation as it satisfies the conditions in Theorem A5:

$$\begin{aligned}
&\sigma(2x_*-1, 2\beta_*) = 1/2 \quad \sigma_x(2x_*-1, 2\beta_*) = 1 \quad \sigma_{xx}(2x_*-1, 2\beta_*) = 0, \\
&\sigma_\beta(2x_*-1, 2\beta_*) = 0 \quad \sigma_{x\beta}(2x_*-1, 2\beta_*) \neq 0 \quad \sigma_{xxx}(2x_*-1, 2\beta_*) \neq 0.
\end{aligned}$$

Similarly at $(x_*, \beta_*) = (1/2, -1)$ the system under goes a period doubling bifurcation as it satisfies the conditions in Theorem A4

$$\begin{aligned}
&\sigma(2x_*-1, 2\beta_*) = 1/2 \quad \sigma_x(2x_*-1, 2\beta_*) = -1 \quad \sigma_{xx}(2x_*-1, 2\beta_*) = 0, \\
&\sigma_\beta(2x_*-1, 2\beta_*) = 0 \quad \sigma_{x\beta}(2x_*-1, 2\beta_*) \neq 0 \quad \sigma_{xxx}(2x_*-1, 2\beta_*) \neq 0.
\end{aligned}$$

□

We can fully classify the dynamics of (12) using the above theorem. We omit the proof as it is similar to the proof of Theorem 1.

Theorem 2. *The full dynamics of the system (12) are as follows*

1. *For $-1 \leq \beta \leq 1$, the system has a single hyperbolic fixed point $x_* = 1/2$ which is a global attractor and there are no p-periodic points for $p \geq 2$.*
2. *For $|\beta| > 1$, the system has one repelling hyperbolic fixed point $x_* = 1/2$ and two hyperbolic stable fixed points c_0, c_1, with $0 < c_0 < 1/2 < c_1 < 1$, and stable sets $W^s(c_0) = [0, 1/2)$, $W^s(c_1) = (1/2, 1]$.*
3. *For $|\beta| = 1$, the system has one non-hyperbolic fixed point at $x_* = 1/2$ which is asymptotically stable and attracting.*

The system undergoes a pitchfork bifurcation at $\beta = \pm 1$. There are no p-periodic points for $p \geq 2$.

5.2. Sequential Dynamics

To fully understand the dynamics of the equations defining the updates to q_1^* and q_2^* it suffices to track the evolution of the points $q_1^*(1) = \zeta$ and $q_2^*(1) = \xi$. The CAVI algorithm updates terms sequentially, using the new values of the variables to calculate the others. We initialize the CAVI algorithm at points ζ_0, ξ_0. The CAVI algorithm is a dynamical system formed by sequential iterations of $\sigma(2x-1, 2\beta)$ starting from ζ_0, ξ_0. We can decouple the CAVI updates for ξ_k and ζ_k by looking at the second iterations. This decoupling is visualized in the diagram (14) below. The system formed the sequential updates is equivalent to the following decoupled system

$$\begin{aligned}
\zeta_1 &= \sigma(2\xi_0 - 1, 2\beta), \\
\zeta_{k+1} &= \sigma(2\sigma(2\zeta_k - 1, 2\beta) - 1, 2\beta), \quad k \geq 1, \\
\xi_{k+1} &= \sigma(2\sigma(2\xi_k - 1, 2\beta) - 1, 2\beta), \quad k \geq 0.
\end{aligned} \tag{13}$$

We propose to investigate the dynamics of the sequential system (9) by studying the dynamics of individual subsequences ζ_{k+1} and ξ_{k+1} of the decoupled system (13). The dynamical properties of the individual subsequences follow from a combination of Theorem 1, Theorem 2 and other methods from Appendix A.

$$\begin{tikzcd}
\zeta_0 & \zeta_1 \arrow[r, "\sigma^2"] \arrow[d, "\sigma"] & \zeta_2 \arrow[r, "\sigma^2"] \arrow[d, "\sigma"] & \zeta_3 \cdots \\
\xi_0 \arrow[r, "\sigma^2"] \arrow[ur, "\sigma"] & \xi_1 \arrow[r, "\sigma^2"] \arrow[ur, "\sigma"] & \xi_2 \arrow[r, "\sigma^2"] \arrow[ur, "\sigma"] & \xi_3 \cdots
\end{tikzcd} \tag{14}$$

Illustrations of the evolution of the dynamics of the sequential updates for various initializations and values of β are in Figures 3–6.

Theorem 3 (CAVI dynamics)**.** *The Dynamics of the CAVI System (9) Are Given by*

1. *For $\beta < -1$, the system has the system has one locally asymptotically unstable fixed point $(1/2, 1/2)$ and two locally asymptotically stable fixed points (c_0, c_1) and (c_1, c_0), with stable sets $W^s((c_0, c_1)) = [0,1] \times [0, 1/2)$ and $W^s((c_1, c_0)) = [0,1] \times (1/2, 1]$ respectively.*
2. *For $|\beta| \leq 1$, the system has a global asymptotically stable fixed point $(1/2, 1/2)$.*
3. *For $\beta > 1$ the system has the system has one locally asymptotically unstable fixed point $(1/2, 1/2)$ and two locally asymptotically stable fixed points (c_0, c_0) and (c_1, c_1), with domains of attraction $W^s((c_0, c_0)) = [0,1] \times [0, 1/2)$ and $W^s((c_1, c_1)) = [0,1] \times (1/2, 1]$ respectively.*

where $0 \leq c_0 < 1/2 < c_1 \leq 1$ are the fixed points of (11) in $[0,1]$. The system undergoes a super-critical pitchfork bifurcation at $\beta = -1$ and again at $\beta = 1$. Furthermore the system has no p-periodic points for $p \geq 2$.

Proof. We will proceed to construct the dynamics of the system (9) by tracing the behavior of the dynamics in the equivalent system (13). The dynamics of each of these subsequences is governed by the Functions (11) and (12) and dependent on the initialization ξ_0. The behavior for each of the subsequence ξ_{k+1}, for $k \geq 0$ is governed by Theorem 2. Similarly the behavior of the subsequence ζ_{k+1}, for $k \geq 1$ is governed by Theorem 2 with the additional point $\zeta_1 = \sigma(2\xi_0 - 1, 2\beta)$ dependent on Theorem 1. For $|\beta| < 1$, (11) has a globally stable fixed point at $x_* = 1/2$ and thus for all ξ_0, $\zeta_1 = \sigma(2\xi_0 - 1, 2\beta) \in W^s(1/2)$. It now follows from Theorem 2 that the only fixed point in the sequential system is $(1/2, 1/2)$ which must be globally stable. For $\beta = \pm 1$, the fixed point $x_0 = 1/2$ is asymptotically stable by Theorem A3. The system undergoes a super-critical pitchfork bifurcation at $\beta = -1$ and again at $\beta = 1$ as a consequnece from its relation to (12). For $\beta > 1$, (11) bifurcates. We have the unstable fixed point $x_* = 1/2$, and the two locally stable fixed points, c_0 with stable set

$W^s(c_0) = [0, 1/2)$, and c_1 with stable set $W^s(c_1) = (1/2, 1]$. For $\xi_0 \in W^s(c_0)$ we have $\zeta_1 \in W^s(c_0)$ and $\xi_1 \in W^s(c_0)$. It now follows from Theorem 2 that the system will converge to (c_0, c_0) and that $W^s((c_0, c_0)) = [0, 1] \times [0, 1/2)$. A similar argument shows the system converges to (c_1, c_1) for $\xi_0 \in W^s(c_1)$ and $W^s((c_1, c_1)) = [0, 1] \times (1/2, 1]$. Lastly, $(1/2,\ 1/2)$ is a repelling fixed point of the systems since $x_* = 1/2$ is a repelling fixed point for both (11) and (12). For $\beta < -1$, (11) bifurcates. We have the unstable fixed point $x_* = 1/2$, and the stable two cycle, $\mathcal{C} = \{c_0, c_1\}$ with stable set $W^s(\mathcal{C}) = [0, 1/2) \cup (1/2, 1]$. For any $\xi_0 < 1/2$ we have, $\zeta_1 > 1/2$ and $\xi_1 < 1/2$. It now follows from Theorem 2 that the system will converge to (c_1, c_0) and that $W^s((c_1, c_0)) = [0, 1] \times [0, 1/2)$. A similar argument shows the system converges to (c_0, c_1) for $\xi_0 > 1/2$ and $W^s((c_0, c_1)) = [0, 1] \times [0, 1/2)$. Lastly, $(1/2,\ 1/2)$ is a repelling fixed point of the systems since $x_* = 1/2$ is a repelling fixed point for both (11) and (12). The dynamics of (13) lack any p-period point and cycles for $p > 2$ as a consequence of its construction from (12). $\square$

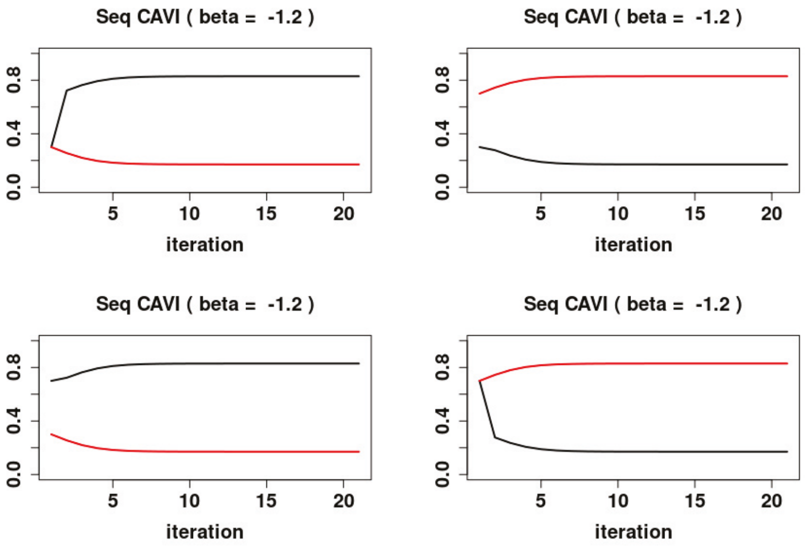


Figure 3. A plot of the first 20 iterations of the CAVI algorithm at various initializations for $\beta = -1.2$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.3$; we see that ζ_k converges to the local fixed point $c_1(1.2) = 0.82928$ and ξ_k converges to the local fixed point $c_0(1.2) = 0.17071$. The upper right is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that ζ_k converges to the local fixed point $c_0(1.2) = 0.17071$ and ξ_k converges to the local fixed point $c_1(1.2) = 0.82928$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that ζ_k converges to the local fixed point $c_1(1.2) = 0.82928$ and ξ_k converges to the local fixed point $c_0(1.2) = 0.17071$. The upper left plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that ζ_k converges to the local fixed point $c_0(1.2) = 0.17071$ and ξ_k converges to the local fixed point $c_1(1.2) = 0.82928$.

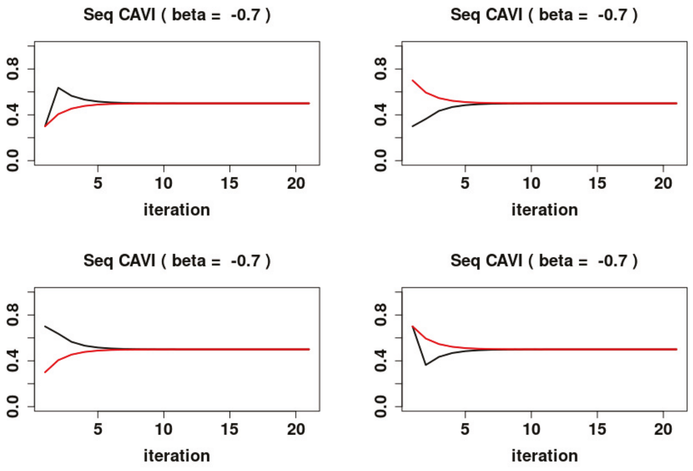


Figure 4. A plot of the first 20 iterations of the CAVI algorithm at various initializations for $\beta = -0.7$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.3$; we see that both of these converge to the global fixed point $1/2$. The upper right is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that this initialization converges to the global fixed point $1/2$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that this initialization converges to the global fixed point $1/2$. The upper left plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that both of these converge to the global fixed point $1/2$.

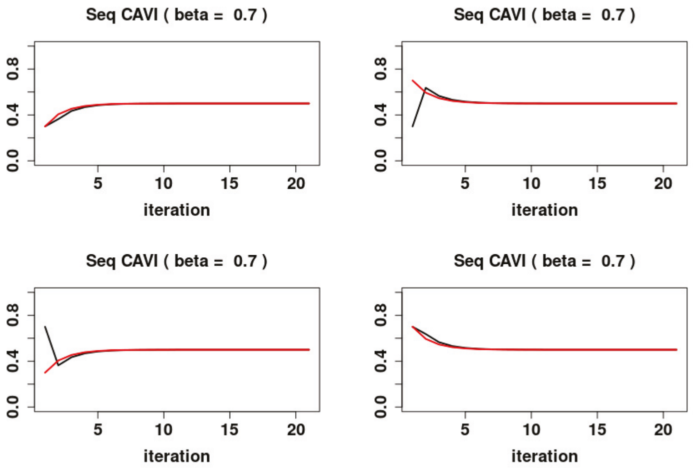


Figure 5. A plot of the first 20 iterations of the CAVI algorithm at various initializations for $\beta = 0.7$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.3$; we see that both of these converge to the global fixed point $1/2$. The upper right is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that this initialization converges to the global fixed point $1/2$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that this initialization converges to the global fixed point $1/2$. The upper left plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that both of these converge to the global fixed point $1/2$.

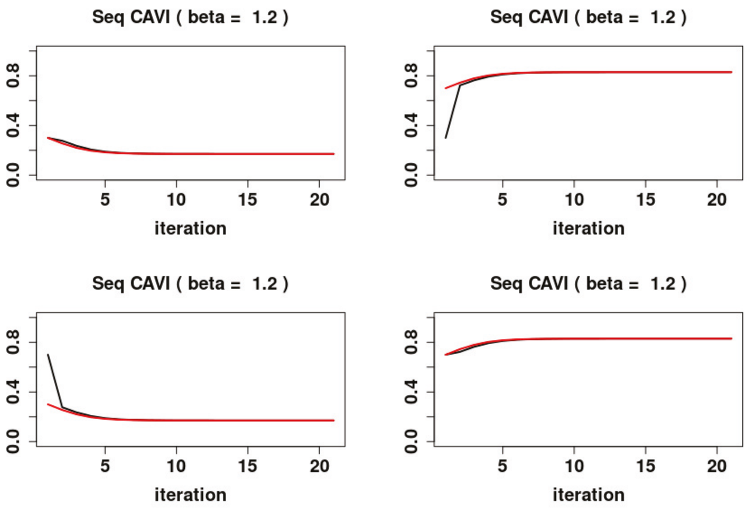


Figure 6. A plot of the first 20 iterations of the CAVI algorithm at various initializations for $\beta = 1.2$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.3$; we see that both of these converge to the local fixed point $c_0(1.2) = 0.17071$. The upper right is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that this initialization converges to the local fixed point $c_1(1.2) = 0.82928$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that this initialization converges to the local fixed point $c_0(1.2) = 0.17071$. The upper left plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that both of these converge to the local fixed point $c_1(1.2) = 0.82928$.

5.3. Parallel Updates

The system of parallel updates is defined by the one-step map $F : \mathbb{R}^2 \to \mathbb{R}^2$

$$\begin{pmatrix} \zeta \\ \xi \end{pmatrix} \mapsto F(\zeta, \xi) = \begin{pmatrix} \sigma(2\xi - 1, 2\beta) \\ \sigma(2\zeta - 1, 2\beta). \end{pmatrix} \tag{15}$$

The dynamics of the parallel system are similar to the system studied in [36]. As we shall show below, the parallel system exhibits periodic behavior that the sequential system does not and it follows as a corollary that the systems are not locally topologically conjugate.

The parallelized CAVI algorithm is a dynamical system formed by iterations of F defined in (15). We shall decouple the parallelized CAVI updates for sequences ξ_k and ζ_k by looking at iterations of (12) acting on the sequences individually. This decoupling is visualized in diagram form

$$\begin{tikzcd} \zeta_0 \arrow[rd] & \zeta_1 \arrow[rd] & \zeta_2 \arrow[rd] & \zeta_3 \cdots \\ \xi_0 \arrow[ru] & \xi_1 \arrow[ru] & \xi_2 \arrow[ru] & \xi_3 \cdots \end{tikzcd} \tag{16}$$

where each cross is an application of F. The system formed the parallel updates is equivalent to the following decoupled systems of even subsequences and odd subsequences. The even subsequences are

$$\zeta_{2k} = \sigma(2\sigma(2\zeta_{2(k-1)} - 1, 2\beta) - 1, 2\beta), \quad k \geq 1 \tag{17}$$
$$\xi_{2k} = \sigma(2\sigma(2\xi_{2(k-1)} - 1, 2\beta) - 1, 2\beta), \quad k \geq 1. \tag{18}$$

The odd subsequences are

$$\zeta_{2k+1} = \begin{cases} \sigma(2\xi_0, 2\beta) & k = 0 \\ \sigma(2\sigma(2\zeta_{2k-1}, 2\beta), 2\beta) & k \geq 1 \end{cases} \tag{19}$$

$$\xi_{2k+1} = \begin{cases} \sigma(2\zeta_0, 2\beta) & k = 0 \\ \sigma(2\sigma(2\xi_{2k-1}, 2\beta), 2\beta) & k \geq 1. \end{cases} \tag{20}$$

Following a similar approach to the one used to study the sequential dynamics, we investigate the dynamics of the parallel system (15) by studying the dynamics of four individual subsequences (17)–(20) of the decoupled system given by diagram (16). The dynamical properties of the individual subsequences follow from a combination of Theorem 1, Theorem 2 and other methods from Appendix A. Illustrations of the evolution of the dynamics of the parallel updates for various initializations and values of β are in Figures 7–12.

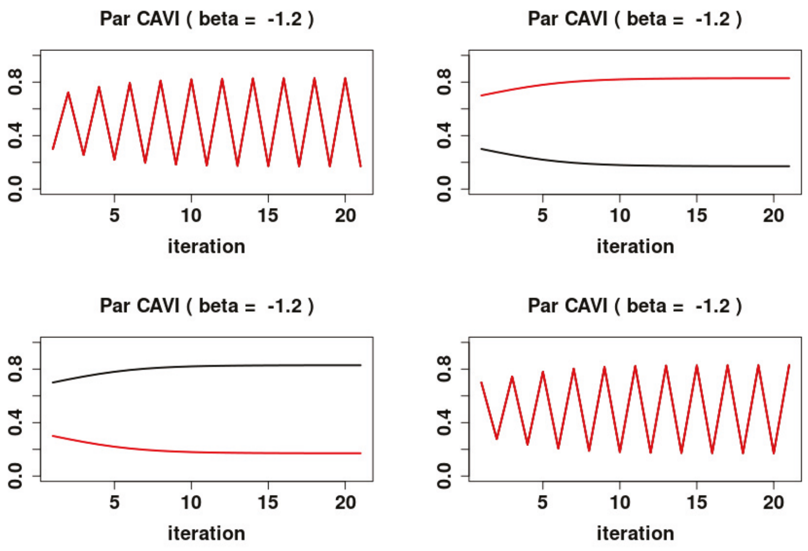


Figure 7. A plot of the first 20 iterations of the parallel update CAVI algorithm at various initializations for $\beta = -1.2$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that this initialization converges to the two cycle $\mathcal{C}_0 = \{(c_0, c_0), (c_1, c_1)\}$. The upper right plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that both of these converge to $c_0(1.2) \approx 0.17071$. The lower left plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that both of these converge to $c_1(1.2) \approx 0.82928$. The lower right is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that this initialization converges to the two cycle $\mathcal{C}_0 = \{(c_0, c_0), (c_1, c_1)\}$.

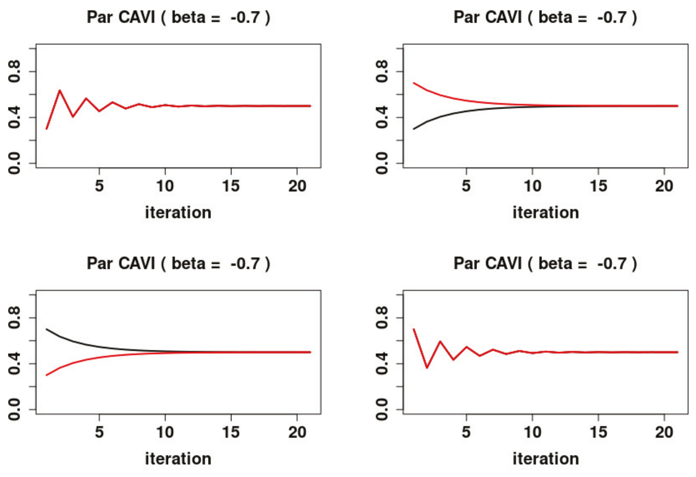


Figure 8. A plot of the first 20 iterations of the parallel update CAVI algorithm at various initializations for $\beta = -0.7$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.3$; we see that both of these converge to the global fixed point $1/2$. The upper right is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that this initialization converges to the global fixed point $1/2$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that this initialization converges to the global fixed point $1/2$. The upper left plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that both of these converge to the global fixed point $1/2$.

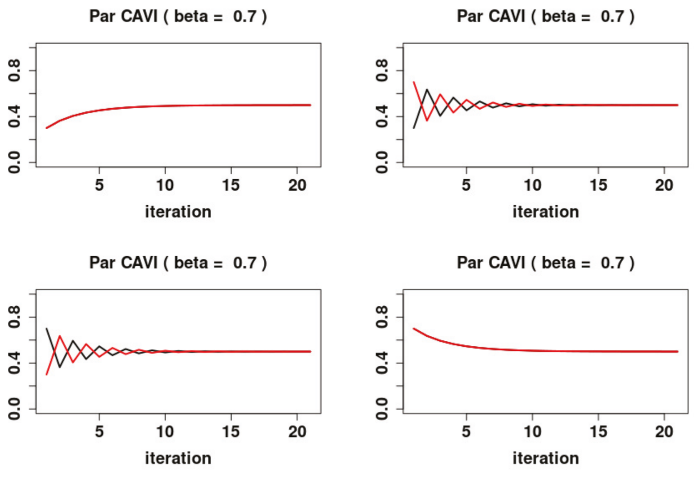


Figure 9. A plot of the first 20 iterations of the parallel update CAVI algorithm at various initializations for $\beta = 0.7$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.3$; we see that both of these converge to the global fixed point $1/2$. The upper right is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that this initialization converges to the global fixed point $1/2$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that this initialization converges to the global fixed point $1/2$. The upper left plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that both of these converge to the global fixed point $1/2$.

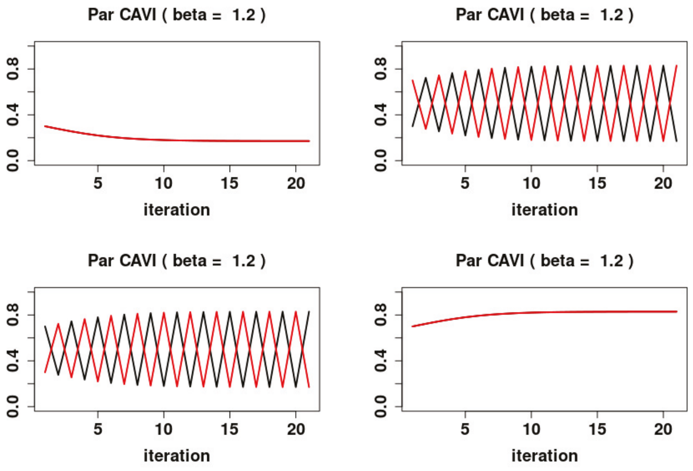


Figure 10. A plot of the first 20 iterations of the parallel update CAVI algorithm at various initializations for $\beta = 1.2$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.3$; we see that both of these converge to $c_0(1.2) \approx 0.17071$. The upper right is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.7$; we see that this initialization converges to the two cycle $\mathcal{C}_1 = \{(c_1, c_0), (c_0, c_1)\}$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.3$; we see that this initialization converges to the two cycle $\mathcal{C}_1 = \{(c_1, c_0), (c_0, c_1)\}$. The lower right plot is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.7$; we see that both of these converge to $c_1(1.2) \approx 0.82928$.

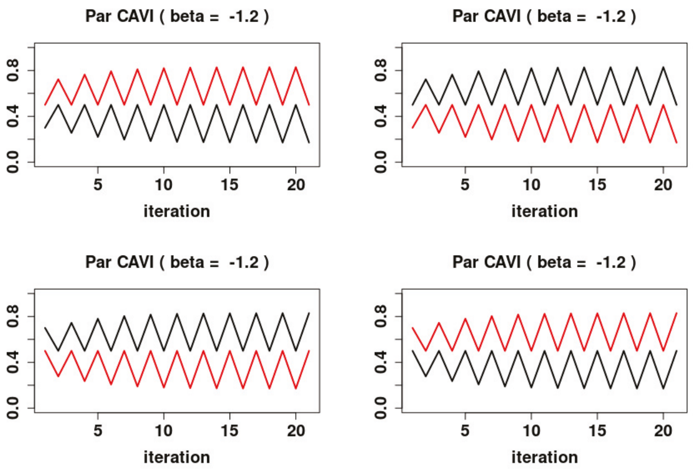


Figure 11. A plot of the first 20 iterations of the parallel update CAVI algorithm at various initializations for $\beta = -1.2$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.5$; we see that this converges to the two-cycle $\mathcal{C}_2 = \{(c_0, 1/2), (1/2, c_1)\}$. The upper right is an initialization of $\zeta_0 = 0.5$ and $\xi_0 = 0.3$; we see that this initialization converges to the two cycle $\mathcal{C}_3 = \{(c_1, 1/2), (1/2, c_0)\}$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.5$; we see that this initialization converges to the two cycle $\mathcal{C}_3$. The lower right plot is an initialization of $\zeta_0 = 0.5$ and $\zeta_0 = 0.7$; we see that this converges to the two-cycle $\mathcal{C}_2$.

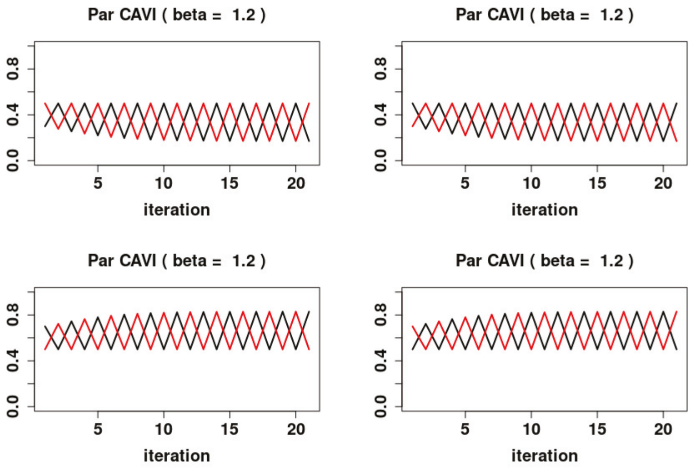


Figure 12. A plot of the first 20 iterations of the parallel update CAVI algorithm at various initializations for $\beta = 1.2$. In each of the plots the ζ updates are black and the ξ updates are red. The upper left plot is an initialization of $\zeta_0 = 0.3$ and $\xi_0 = 0.5$; we see that this converges to the two-cycle $\mathcal{C}_4 = \{(c_0, 1/2), (1/2, c_0)\}$. The upper right is an initialization of $\zeta_0 = 0.5$ and $\xi_0 = 0.3$; we see that this initialization converges to the two cycle $\mathcal{C}_4$. The lower left is is an initialization of $\zeta_0 = 0.7$ and $\xi_0 = 0.5$; we see that this initialization converges to the two cycle $\mathcal{C}_5 = \{(c_1, 1/2), (1/2, c_1)\}$. The lower right plot is an initialization of $\zeta_0 = 0.5$ and $\xi_0 = 0.7$; we see that this converges to the two-cycle $\mathcal{C}_5$.

We now present the main result for the parallel dynamics.

Theorem 4 (Parallel Dynamics)**.** *The Dynamics of the Parallel System (10) Are As Follows*

1. *For $\beta < -1$, the system has two locally asymptotically stable fixed points (c_1, c_0) and (c_0, c_1), and one locally asymptotically unstable fixed point $(1/2, 1/2)$, where c_0 and c_1 are the fixed points of (11). Furthermore the system exhibits periodic behavior in the form of 2-cycles. The asymptotically stable 2-cycle, $\mathcal{C}_1 = \{(c_0, c_0), (c_1, c_1)\}$ and asymptotically unstable 2-cycles,*

$$\mathcal{C}_2 = \{(1/2, c_1), (c_0, 1/2)\} \text{ and } \mathcal{C}_3 = \{(1/2, c_0), (c_1, 1/2)\}.$$

The stable sets are

$$\begin{aligned} W^s(c_0, c_1) &= [0, 1/2) \times (1/2, 1] \\ W^s(c_1, c_0) &= (1/2, 1] \times [0, 1/2) \\ W^s(\mathcal{C}_1) &= ([0, 1/2) \times [0, 1/2)) \cup ((1/2, 1] \times (1/2, 1]) \\ W^s(\mathcal{C}_2) &= ([0, 1/2) \times \{1/2\}) \cup (\{1/2\} \times (1/2, 1]) \\ W^s(\mathcal{C}_3) &= ([0, 1/2) \times \{1/2\}) \cup (\{1/2\} \times (1/2, 1]). \end{aligned}$$

2. *For $-1 \le \beta \le 1$, the system has a global attracting fixed point $(1/2, 1/2)$.*
3. *For $\beta > 1$, the system has two locally asymptotically stable fixed points (c_0, c_0) and (c_1, c_1), and one locally asymptotically unstable fixed point $(1/2, 1/2)$, where c_0 and c_1 are the fixed points of (11). Furthermore the system exhibits periodic behavior in the form of 2-cycles. The asymptotically stable 2-cycle, $\mathcal{C}_3 = \{(c_0, c_0), (c_1, c_1)\}$ and asymptotically unstable 2-cycles, $\mathcal{C}_4 = \{(1/2, c_0), (c_1, 1/2)\}$ and $\mathcal{C}_5 = \{(1/2, c_1), (c_1, 1/2)\}$. The stable sets are*

$$\begin{aligned}
W^s(c_0, c_1) &= [0, 1/2) \times (1/2, 1] \\
W^s(c_1, c_0) &= (1/2, 1] \times [0, 1/2) \\
W^s(\mathcal{C}_3) &= ([0, 1/2) \times [0, 1/2)) \cup ((1/2, 1] \times (1/2, 1]) \\
W^s(\mathcal{C}_4) &= ([0, 1/2) \times \{1/2\}) \cup (\{1/2\} \times [0, 1/2)) \\
W^s(\mathcal{C}_5) &= (\{1/2\} \times (1/2, 1]) \cup ((1/2, 1] \times \{1/2\}).
\end{aligned}$$

The system has no p-periodic points for $p > 2$. The system under goes a PD bifurcation at $\beta = -1$ and a pitchfork bifurcation at $\beta = 1$.

Proof. The dynamics of the system defined by F in (15) are equivalent to the dynamics of the system generated by the subsequences (17)–(20). The dynamics of each of these subsequences are governed by the functions (11) and (12). By Theorem 1, we have the behavior for each of the subsequences (17)–(20). For $|\beta| < 1$, (11) has a globally stable fixed point at $x_* = 1/2$ and thus the only fixed point in the parallel system is $(1/2, 1/2)$ which must be globally stable. For $\beta = \pm 1$, the fixed point $x_0 = 1/2$ is asymptotically stable by Theorem A3.

For $\beta > 1$, (11) bifurcates. We have the unstable fixed point $x_* = 1/2$, and the two locally stable fixed points, c_0 with stable set $W^s(c_0) = [0, 1/2)$, and c_1 with stable set $W^s(c_1) = (1/2, 1]$. Returning to the system generated by F, if we consider the initialization $(\zeta_0, \xi_0) = (c_0, c_0)$ then by the sequence construction of ζ_n, given in (17) and (19), we see that $\zeta_n = c_0$ for $n \geq 1$, as c_0 is a fixed point of (11) for $\beta > 1$. Similarly, using the sequence construction of ξ_n, given in (18) and (20), we see that $\xi_n = c_0$ for $n \geq 1$, as c_0 is a fixed point of (11) for $\beta > 1$. Therefore, (c_0, c_0) is a fixed point. An analogous argument shows that (c_1, c_1) is also a fixed point. The parallel system has the stable fixed points (c_0, c_0) with stable set $W^s(c_0, c_0) = W^s(c_0) \times W^s(c_0)$ and (c_1, c_1) with stable set $W^s(c_1, c_1) = W^s(c_1) \times W^s(c_1)$. After the bifurcation at $\beta = 1$ the parallel system also contains 2-cycles. Using the sequence construction we see that $\mathcal{C}_3 = \{(c_1, c_0), (c_0, c_1)\}$ is an asymptotically stable 2-cycle in the parallel system, with stable subspace $W^s(\mathcal{C}_3) = (1/2, 1] \times [0, 1/2) \cup [0, 1/2) \times (1/2, 1]$. Additionally, we have two asymptotically unstable 2-cycles $\mathcal{C}_4 = \{(c_0, 1/2), (1/2, c_0)\}$ and $\mathcal{C}_5 = \{(c_1, 1/2), (1/2, c_1)\}$. Perturbing the 1/2 coordinate in the unstable cycle pushes it into the basin of attraction for one of the fixed points or the asymptotically stable 2-cycle. The stable sets are $W^s(\mathcal{C}_4) = ([0, 1/2) \times \{1/2\}) \cup (\{1/2, 1\} \times [0, 1/2))$, $W^s(\mathcal{C}_5) = (\{1/2\} \times (1/2, 1]) \cup ((1/2, 1] \times \{1/2\})$. The dynamics of F lack any p-period point and cycles for $p > 2$ as a consequence of its construction from (12).

For $\beta < -1$, (11) bifurcates. We have the unstable fixed point $x_* = 1/2$, and the stable two cycle, $\mathcal{C} = \{c_0, c_1\}$ with stable set $W^s(\mathcal{C}) = [0, 1/2) \cup (1/2, 1]$. Returning to the system generated by F, if we consider the initialization $(\zeta_0, \xi_0) = (c_0, c_1)$ then by the sequence construction of ζ_n, given in (17) and (19), we see that $\zeta_n = c_0$ for $n \geq 1$, as $\mathcal{C}$ is a 2-cycle of (11) for $\beta < -1$. Similarly, using the sequence construction of ξ_n, given in (18) and (20) we see that $\xi_n = c_1$ for $n \geq 1$, as $\mathcal{C}$ is a 2-cycle of (11) for $\beta < -1$. Therefore, (c_0, c_1) is a fixed point. An analogous argument shows that (c_1, c_0) is also a fixed point. The parallel system has the stable fixed points (c_0, c_1) with stable set $W^s(c_0, c_1) = W^s(c_0) \times W^s(c_1)$ and (c_1, c_0) with stable set $W^s(c_1, c_0) = W^s(c_1) \times W^s(c_0)$, where $W^s(c_0) = [0, 1/2)$ and $W^s(c_1) = (1/2, 1]$. After the bifurcation at $\beta = -1$ the parallel system also contains 2-cycles. Using the sequence construction we see that $\mathcal{C}_1 = \{(c_0, c_0), (c_1, c_1)\}$ is an asymptotically stable 2-cycle in the parallel system, with stable subspace $W^s(\mathcal{C}_1) = W^s(c_0) \times W^s(c_0) \cup W^s(c_1) \times W^s(c_1)$. Additional we have two asymptotically unstable 2-cycles $\mathcal{C}_2 = \{(c_0, 1/2), (1/2, c_1)\}$ and $\mathcal{C}_3 = \{(c_1, 1/2), (1/2, c_0)\}$. Perturbing the 1/2 coordinate in the unstable cycle pushes it into the basin of attraction for one of the fixed points or the asymptotically stable 2-cycle. The stable sets are $W^s(\mathcal{C}_3) = ([0, 1/2) \times [0, 1/2)) \cup ((1/2, 1] \times (1/2, 1])$, $W^s(\mathcal{C}_4) = ([0, 1/2) \times \{1/2\}) \cup (\{1/2\} \times [0, 1/2))$, $W^s(\mathcal{C}_5) = (\{1/2\} \times (1/2, 1]) \cup ((1/2, 1] \times \{1/2\})$.

The dynamics of F lack any p-period point and cycles for $p > 2$ as a consequence of its construction from (12).

This completes the characterization of the dynamics of F for $\beta \in \mathbb{R}$. $\square$

5.4. A Comparison of the Dynamics

We end the section by providing a comparison of the dynamical properties of the sequential system in Theorem 3 and the parallel system in Theorem 4. The main difference between the sequential system and the parallel system is the presence of two-cycles that can be found in the parallel system when $|\beta| > 1$. This behavior stems from the difference between the sequential and parallel implementations of the CAVI. Looking closely at the update diagrams for the two systems reveals the key difference that produces these two-cycles. The decoupled sequential system is

$$\begin{tikzcd}
\zeta_0 & \zeta_1 \arrow[r, "\sigma^2"] \arrow[d, "\sigma"] & \zeta_2 \arrow[r, "\sigma^2"] \arrow[d, "\sigma"] & \zeta_3 \cdots \\
\xi_0 \arrow[r, "\sigma^2"] \arrow[ur, "\sigma"] & \xi_1 \arrow[r, "\sigma^2"] \arrow[ur, "\sigma"] & \xi_2 \arrow[r, "\sigma^2"] \arrow[ur, "\sigma"] & \xi_3 \cdots
\end{tikzcd}$$

and the decoupled parallel system is

$$\begin{tikzcd}
\zeta_0 \arrow[dr] & \zeta_1 \arrow[dr] & \zeta_2 \arrow[dr] & \zeta_3 \cdots \\
\xi_0 \arrow[ur] & \xi_1 \arrow[ur] & \xi_2 \arrow[ur] & \xi_3 \cdots
\end{tikzcd}$$

The major difference between these diagrams is how the individual update sequences begin. Notice ζ_0 plays no role in updating the sequential system as both the ζ_k update sequence and the ξ_k update sequence are dependent only on the choice of ξ_0. Even after rewriting the sequential updates in terms of individual sequences the system is not truly decoupled as both sequences depend on a common starting point. This precisely prescribes the behavior that we see in the system relative to the sigmoid function dynamics in Theorem 1 and Theorem 2. Compare this to the parallel system. Here ζ_0 is involved in updating both the odd ξ_{2k+1} subsequence and the even ζ_{2k} subsequence. Furthermore, ξ_0 remains involved by controlling the updates for the even ξ_{2k} subsequence and the odd ζ_{2k+1} subsequence. This additional flexibility allows the parallel system to develop periodic behavior outside of the Dobrushin regime $(1 \leq \beta \leq 1)$.

As an example, we will consider initializing the sequential algorithm to the parallel algorithm for $\beta = 1.2$. We begin with the sequential algorithm. For $\beta = 1.2$, consider initializing the sequential system at $(\zeta_0, \xi_0) = (0.7, 0.3)$. The sequential system updates are fully determined by ξ_0, so for $\xi_0 = 0.3$ it follows from Theorem 1 that an application of the function (11) will cause $\zeta_1 \in W^s(c_0)$. At this point, the system can be evolved by applying (12) to the independent sequences for ζ and ξ as given in (13). The dynamics of the system are now controlled by the function (12). From this initialization the system will converge to the fixed point $(c_0, c_0) = (0.17071, 0.17071)$ as shown in Figure 6.

Contrast this with the behavior of the parallel system in which the updates are determined by both ξ_0 and ζ_0. For $\beta = 1.2$, consider initializing the parallel system at $(\zeta_0, \xi_0) = (0.7, 0.3)$. It follows from Theorem 1 that an application of the function (11) will cause $\zeta_1 \in W^s(c_0)$ and $\xi_1 \in W^s(c_1)$. Successive updates will cause the sequences ζ_k and ξ_k to flip back and forth between the domains $W^s(c_0)$ and $W^s(c_1)$, until the system settles into the two cycle $\mathcal{C}_1 = \{(c_0, c_1), (c_1, c_0)\} = \{(0.17071, 0.82928), (0.82928, 0.17071)\}$ as seen in Figure 10.

This simple example highlights the danger of naively parallelizing the CAVI algorithm. The convergence properties of a parallel version of the CAVI algorithm will heavily depend on the models CAVI update equations. In the case of the Ising model we have demonstrated that for certain parameter regimes the parallel implementation of the algorithm can fail to converge due to the dependence of the algorithm on both ζ_0 and $\tilde{\zeta}_0$.

6. Edward–Sokal Coupling

One method of improving convergence in Markov chains is through the use of probabilistic couplings. The Edward–Sokal (ES) coupling is a coupling of two statistical physics models, the random cluster model and the Potts model (a generalization of the Ising model) [37]. Running a Markov chain on the ES coupling leads to improved mixing properties compared to the equivalent Potts model and random cluster models [33]. Motivated by these findings in the Markov chain literature, we ask a similar question: Can the convergence properties of mean-field VI be improved by using the ES coupling in place of the Ising model? In this section we investigate this idea numerically. We first introduce the Edward–Sokal coupling following [37]. We introduce a variational family for the Edward–Sokal coupling and derive the variational updates for this model. Our findings suggests the variational updates converge to a unique solution in a larger range than the equivalent Dobrushin regime for the corresponding Ising measure.

6.1. Random Cluster Model

Let $G = (V, E)$ be a finite graph. Let $e = \langle x, y \rangle \in E$ denote an edge in G with endpoints $x, y \in V$. $\Sigma = \{1, 2, \ldots, q\}^V$, $\Omega = \{0, 1\}^E$ and $\mathcal{F}$ denotes the powerset of Ω. The random cluster model is a 2 parameter probability measure with an edge weight parameter $p \in [0, 1]$ and a cluster weight parameter $q \in \{2, 3, \ldots\}$ on $(\Omega, \mathcal{F})$ given by

$$\phi_{p,q}(\omega) \propto \left\{ \prod_{e \in E} p^{\omega(e)} (1-p)^{(1-\omega(e))} \right\} q^{\kappa(\omega)},$$

where $\kappa(\omega)$ denoted the number of connected components in the subgraph corresponding to ω. The partition function for the random cluster model is

$$\mathcal{Z}_{RC} = \sum_{\omega \in \Omega} \left\{ \prod_{e \in E} p^{\omega(e)} (1-p)^{(1-\omega(e))} \right\} q^{\kappa(\omega)}.$$

For $q = 2$ the the random cluster model reduces to the Ising model on G.

The Edward–Sokal Coupling is a probability measure μ on $\Sigma \times \Omega$ given by

$$\mu(\sigma, \omega) \propto \prod_{e \in E} \left[(1-p)\delta_{\omega(e),0} + p\delta_{\omega(e),1}\delta_e(\sigma) \right], \tag{21}$$

where $\delta_{a,b} = 1(a = b)$, and $\delta_e(\sigma) = 1(\sigma_x = \sigma_y)$, for $e = (x, y) \in E$.

It is well known that in the special case, $p = 1 - e^{-\beta}$ and $q = 2$ the Σ-marginal of the ES coupling is the Ising model, the Ω-marginal is the random cluster model [37]. We are interested in better understanding how the convergence of the CAVI algorithm on the ES coupling compares to the convergence of the CAVI algorithm on the Ising model.

6.2. VI Objective Function

To calculate the VI updates for each variable we may need to make use of the alternative characterization of the ES coupling

$$\mu(\sigma, \omega) \propto \psi(\sigma)\phi_{p,1}(\omega)1_F(\sigma, \omega)$$

where ψ is uniform measure on Σ and $\phi_{p,1}(\omega)$ is a product measure on Ω

$$\phi_{p,1}(\omega) = \prod_{e \in E} p^{\omega(e)}(1-p)^{(1-\omega(e))} \tag{22}$$

and

$$F = \{(\sigma, \omega) : \delta_\omega(e) = 1 \implies \delta_e(\sigma) = 1\} \tag{23}$$

The variational family that we will be optimizing over is

$$q(\sigma, \omega) = q_1(\sigma_1)q_2(\sigma_2)q_0(\omega)1_F(\sigma, \omega). \tag{24}$$

We have added the indicator on the set F to eliminate the configurations (σ, ω) that are not well defined in the variational objective. We will use the convention that $0\log(0) = 0$.

6.3. VI Updates

The ELBO that corresponds to the variational family (24) is

$$\begin{aligned}
\text{ELBO}(x_1, x_2, y, p) &= x_1x_2y\log(x_1x_2y) - x_1x_2y\log(1-p) \\
&+ (1-x_1)x_2y\log((1-x_1)x_2y) - (1-x_1)x_2y\log(1-p) \\
&+ x_1(1-x_2)y\log(x_1(1-x_2)y) - x_1(1-x_2)y\log(1-p) \\
&+ (1-x_1)(1-x_2)y\log((1-x_1)(1-x_2)y) - (1-x_1)(1-x_2)y\log(1-p) \\
&+ x_1x_2(1-y)\log(x_1x_2(1-y)) - x_1x_2(1-y)\log(p) \\
&+ (1-x_1)(1-x_2)(1-y)\log((1-x_1)(1-x_2)(1-y)) - (1-x_1)(1-x_2)(1-y)\log(p).
\end{aligned}$$

Taking the derivative with respect to x_1 and simplifying gives us

$$\begin{aligned}
\text{ELBO}_1(x_1, x_2, y, p) &= y\log\left(\frac{x_1}{1-x_1}\right) + (1-y)\log\left(\frac{1}{1-x_1}\right) \\
&+ x_2(1-y)\log(x_1(1-x_1)) + x_2(1-y)\log\left(\frac{x_2(1-x_2)(1-y)^2}{p^2}\right) \\
&+ \log\left(\frac{p}{(1-x_2)(1-y)}\right) + (2x_2-1)(1-y).
\end{aligned}$$

Taking the derivative with respect to x_2 and simplifying gives us

$$\begin{aligned}
\text{ELBO}_2(x_1, x_2, y, p) &= y\log\left(\frac{x_2}{1-x_2}\right) + (1-y)\log\left(\frac{1}{1-x_2}\right) \\
&+ x_1(1-y)\log(x_2(1-x_2)) + x_1(1-y)\log\left(\frac{x_1(1-x_1)(1-y)^2}{p^2}\right) \\
&+ \log\left(\frac{p}{(1-x_1)(1-y)}\right) + (2x_1-1)(1-y).
\end{aligned}$$

Taking the derivative with respect to y and simplifying gives us

$$\begin{aligned}
\text{ELBO}_y(x_1, x_2, y, p) &= x_1x_2\log\left(\frac{y}{1-y}\right) + x_1x_2\log\left(\frac{p}{1-p}\right) \\
&+ (1-x_1)(1-x_2)\log\left(\frac{y}{1-y}\right) + (1-x_1)(1-x_2)\log\left(\frac{p}{1-p}\right) \\
&+ (1-x_1)x_2\log\left(\frac{(1-x_1)x_2y}{1-p}\right) + x_1(1-x_2)\log\left(\frac{x_1(1-x_2)y}{1-p}\right) \\
&+ (1-x_1)x_2 + x_1(1-x_2).
\end{aligned}$$

Absence of closed form updates for any of the variables limits our ability to study the convergence of the system with classical dynamical systems techniques. Instead we look at the long evolution behavior of the system by plotting 100 iterations of the CAVI updates which are generated from the following system

$$\begin{aligned} x_1(t+1) &= \text{argmin}_{z\in(0,1)}|\text{ELBO}_1(z, x_2(t), y(t), p)|, \\ x_2(t+1) &= \text{argmin}_{z\in(0,1)}|\text{ELBO}_2(x_1(t+1), z, y(t), p)|, \\ y(t+1) &= \text{argmin}_{z\in(0,1)}|\text{ELBO}_y(x_1(t+1), x_2(t+1), z, p)|. \end{aligned}$$

We generate the argmin of the free variable z from a line search with a step size of $\Delta = 10^{-6}$. Running these simulations we find that the iterations of $x_1(t), x_2(t), y(t)$ converge to a global solution within about $T = 20$ time steps from any initialization $x_1(0), x_2(0), y(0) \in (0,1)$ and any $\beta > 0$. It is evident that using the ES coupling, we get global convergence of the algorithm outside of the Dobrushin regime of the corresponding paramagnetic Ising model. The figures depicting the simulation results of convergence of the variational inference algorithm in the Edward–Sokal coupling can be found below in Figures 13–16.

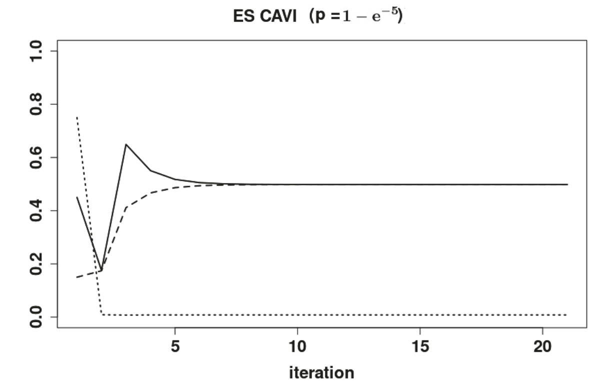


Figure 13. A plot of the 20 iterations of the ES updates for $p = 1 - e^{-5}$ from a uniformly random initialization. Each of the lines represents a different parameter. The solid line is x_1, the dashed line is x_2 and the dotted line is y. We see convergence to a unique fixed point for each of the variables.

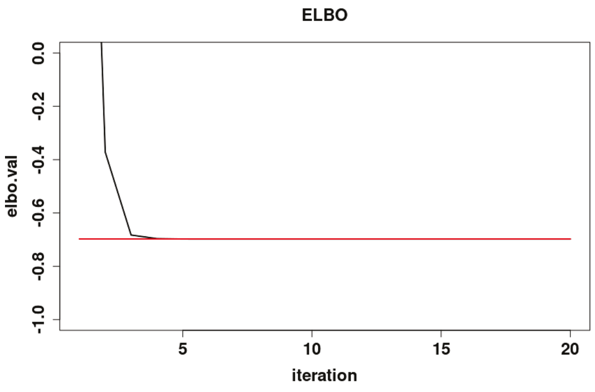


Figure 14. A plot of the ELBO of the ES coupling for $p = 1 - e^{-5}$. The red line denotes the global minimum ELBO value.

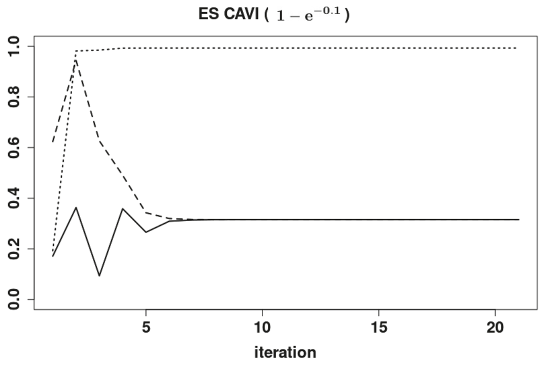


Figure 15. A plot of the 20 iterations of the ES updates for $p = 1 - e^{-0.1}$ from a uniformly random initialization. Each of the lines represents a different parameter. The solid line is x_1, the dashed line is x_2 and the dotted line is y. We see convergence to a unique fixed point.

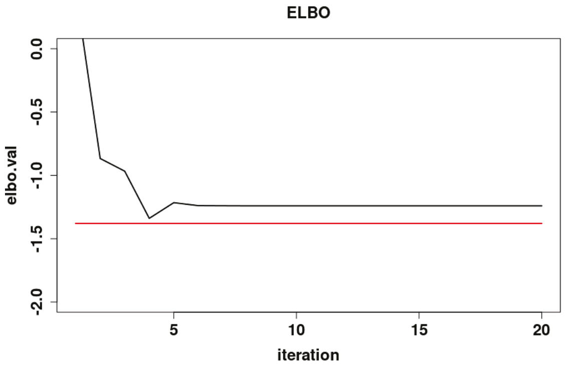


Figure 16. A plot of the ELBO of the ES coupling for $p = 1 - e^{-0.1}$. The red line denotes the global minimum ELBO value.

7. Conclusions

This paper demonstrates the use of classical dynamical systems and bifurcation theory to study the convergence properties of the CAVI algorithm of the Ising model on two nodes. In our simple model we are able to provide the complete dynamical behavior for the Ising model on two nodes. Interestingly, we find that the sequential CAVI algorithm and parallelized CAVI algorithm are not topologically conjugate owing to the presence of periodic behavior in the parallelized CAVI. This behavior originates from the added flexibility of the initialization in the parallelized CAVI when compared to the sequential CAVI. The erratic behavior we see in the Ising model for $|\beta| > 1$ is due to a combination of the existence of multiple fixed points of the systems update function and the instability of these fixed points. In this parameter regime, the fixed point that produces the optimal solution (0.5, 0.5) is a repelling fixed point. Unless we initialize the algorithm exactly at (0.5, 0.5), the CAVI system cannot converge to this point. The other two suboptimal fixed points are both asymptotically stable. This suggests that the main problem that the CAVI algorithm experiences is centered around the existence of multiple fixed points. Recent work on stochastic block models (SBM) and topic models (TM) models shows that mean field VI leads to suboptimal estimators [17–20]. It is not clear if this property comes from the mean field variational inferences construction using product distributions or if this is a consequence of structure among latent variables. A minor difference of the stochastic block model (SBM) or topic model (TM) with the Ising model is that the former contain parameters (e.g., the cluster labels) that are identifiable only up to permutations. That being said, in the SBM or TM, if the cluster means are not well-separated, then it is not possible to identify the labels even up to permutations. This is somewhat related to having multiple fixed points of the objective function and we conjecture similar behavior to what we have found in the Ising model will be exhibited in the SBM or TM outside the Dobrushin regime. Interestingly, a close look at the BCAVI updates in [17,18] reveals a similar sigmoid update function $1/(1+e^{-x})$. Applying the tools and techniques from dynamical systems theory to study the CAVI algorithm in the SBM, TM and other models will provide a better understanding of the issues that come with using mean field variational inference and is important to developing better variational inference techniques.

Most of the research into the theoretical properties of variational inference has focused on the mean field family due to its computational simplicity. This computational simplicity comes at the cost of limited expressive power. Can we make due with this limited expressive power in practical applications? More specifically, is there an equivalent parameter regime to the Dobrushin regime ($1 \leq \beta \leq 1$) for other similar models like the SBM and TM inside which the CAVI produces statistically optimal estimators? The answer to this question provides researchers with stable parameter regimes for the model. The non-existenceof such a region would indicate the need for more expressive variational methods for the model beyond mean field methods. Recent work [19,20] suggests that this adding some structure to algorithms may fix the problems that arise from mean field VI. How much structure is needed to recover statistically optimal estimators? Could adding in a simple structure of pair-wise dependence to the mean field VI in the Ising model, similarly to [19], be enough to recover the optimal estimator outside of the Dobrushin regime? Is the amount of additional structure that is needed somehow related to the latent structure of the models? Tools from dynamical systems theory can be used to study these questions.

Using dynamical systems to study the convergence properties of the CAVI algorithm is not without its challenges. While dynamical systems theory can provide the answers to many of the above questions, applying these tools to higher dimensional sequential systems is a challenging problem. As mentioned previously, the general theory for n-dimensional discrete dynamical systems is dependent on writing the evolution function in the form $x_{n+1} = F(x_n)$. Deriving this F is typically not possible for densely connected higher dimensional sequential systems like the n-dimensional Ising model CAVI. This is not the only challenging aspect to the problem. These systems typically possess multiple fixed points which can only be found numerically. Multiple fixed points will lead to more complicated partitions of the space into domains of attraction. Furthermore, higher dimensional systems can possess bifurcations of multiple codimensions, which as significantly more difficult to study. Bifurcations of codimension 3 are so exotic that they are not well studied [23,24]. Software to handle such calculations has only recently been developed [24]. In practical terms this means that the convergence properties can only be studied numerically for models with a small number of parameters. Furthermore, most of the numerical techniques work under the assumption of differentiability of the evolution operator and will fail to be applicable to many systems of practical interest in statistics such as the Edward–Sokal CAVI. Applying tools from dynamical systems to the study of variational inference algorithms will require developing new theory for high dimensional and well connected sequential dynamical systems.

Author Contributions: Conceptualization, S.P., D.P. and A.B.; formal analysis, S.P.; supervision, D.P. and A.B.; writing—original draft, S.P., D.P. and A.B. All authors have read and agreed to the published version of the manuscript.

Funding: Pati and Bhattacharya acknowledge support from NSF DMS (1854731, 1916371) and NSF CCF 1934904 (HDR-TRIPODS). In addition, Bhattacharya acknowledges the NSF CAREER 1653404 award for supporting this project.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. An Overview of One Dimensional Dynamical Systems

The main focus of discrete dynamical systems is the asymptotic behavior of iterated systems (8). Bifurcation theory studies how the dynamical behavior of a system changes as the parameter J_{12} changes. We study the behavior of convergence of the CAVI algorithm by studying the autonomous discrete time dynamical system formed by the update Equation (8). This allows us to utilize tools from dynamical systems theory to study the behavior of the algorithm with respect to its parameters. In this section we provide a brief overview of the necessary dynamical systems and bifurcation theory in dimension 1 used in Section 5.

Appendix A.1. Notation

Our focus will be on parametric dynamical systems defined by a functions $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$. We will call elements $\mathbf{x} \in \mathbb{R}^n$ elements in the state space (phase space) and elements $\alpha \in \mathbb{R}^p$ as parameters. We denote real numbers $x \in \mathbb{R}$ and real vectors in $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ with bold. We denote the inverse of an invertible function f by f^{-1}. The k-fold composition of a function f with itself at a point $(\mathbf{x}, \alpha)$ will be denoted by $f^k(\mathbf{x}, \alpha)$. The k-fold composition of the inverse function f^{-1} will be denoted f^{-k}. The identity function will be denoted id. We use the convention $f^0 = \text{id}$. We denote the tensors of derivatives of f by $f_{\mathbf{x}}(\mathbf{x}, \alpha) = (\partial f_i / \partial x_j)$, $f_{\mathbf{xx}}(\mathbf{x}, \alpha) = (\partial^2 f_i / \partial x_j \partial x_k)$, $f_{\mathbf{xx}}(\mathbf{x}, \alpha) = (\partial^2 f_i / \partial x_j \partial x_k)$, $f_{\mathbf{xxx}}(\mathbf{x}, \alpha) = (\partial^3 f_i / \partial x_j \partial x_k \partial x_\ell)$, $f_\alpha(\mathbf{x}, \alpha) = (\partial f_i / \partial \alpha_j)$.

Appendix A.2. Dynamical Systems

Dynamical systems is a classical approach to studying the convergence properties of non-linear iterative systems. These systems can be continuous in time, for example a differential equation, or discrete in time, for example iterations of a function from an initial point. A dynamical system is called autonomous if the function governing the system is independent of time and non-autonomous otherwise. The coordinate ascent variational inference for the Ising model is a discrete-time autonomous dynamical system. Before giving a complete proof of the dynamical properties of the CAVI algorithm for the Ising model in dimension 2, we first give a basic introduction to the theory of discrete time dynamical systems and bifurcations following [23–25,38].

Formally, a dynamical system is triple $\{T, X, \phi^t\}$ where T is a time set, X is the state space and $\phi^t : X \to X$ is a family of evolution operators parameterized by $t \in T$ satisfying $\phi^0 = id$ and $\phi^{s+t} = \phi^t \circ \phi^s$ for all $x \in X$. For a discrete time system the evolution operator is fully specified by the one-step map $\phi^1 = f$, since the composition rule then defines $\phi^k = f^k$ for $k \in \mathbb{Z}$. We restrict the further discussion to discrete time dynamical systems defined by the one-step map

$$\mathbf{x} \mapsto f(\mathbf{x}, \alpha), \quad \mathbf{x} \in \mathbb{R}^n, \alpha \in \mathbb{R}^p, \tag{A1}$$

where f is a diffeomorphism, a smooth function with smooth inverse, of the state space $\mathbb{R}^n$ and α are the parameters of the system.

The basic geometric objects of a dynamical system are orbits in the state space and the phase portrait, defined as follows. The phase portrait is the partition of the state space induced by the orbits. The orbit starting at a point $\mathbf{x}$ is an ordered subset of the state space $\mathbb{R}^n$ denoted $\text{orb}(\mathbf{x}) = \{f^k(\mathbf{x}) : k \in \mathbb{Z}\}$. There are two special types of orbits, fixed points and cycles, defined below.

A fixed point $\mathbf{x}_*$ of the system are points that remain fixed under the evolution of the system, ones that satisfies $\mathbf{x}_* = f(\mathbf{x}_*)$. We can classify fixed points of the system by studying the local behavior of the system near the fixed point. To do this we consider small perturbations of the system near the fixed point. A fixed point $\mathbf{x}_*$ is said to be locally stable if points that are near the fixed point do not move too far away from the fixed point as the system evolves. Formally, if for any $\varepsilon > 0$ there exists $\delta > 0$ such that for all x with $|\mathbf{x} - \mathbf{x}_*| < \delta$ we have $|f^k(\mathbf{x}) - \mathbf{x}_*| < \varepsilon$ for all $k > 0$. A fixed point is called semi-stable from the right if for any $\varepsilon > 0$ there exists $\delta > 0$ such that for all x with $0 < \mathbf{x} - \mathbf{x}_* < \delta$ we have $|f^k(\mathbf{x}) - \mathbf{x}_*| < \varepsilon$ for all $k > 0$ (semi-stable from the left is defined analogously). It is said to be locally unstable otherwise. A fixed point $\mathbf{x}_*$ is locally attracting if all points in a small neighborhood converge to the fixed point as we let the system evolve. Formally, if there exists an $\eta > 0$ such that $|\mathbf{x} - \mathbf{x}_*| < \eta$ implies $f^n(\mathbf{x}) \to \mathbf{x}_*$ as $n \to \infty$. A fixed point $\mathbf{x}_*$ is locally asymptotically stable if it is both locally stable and attracting. A fixed point $\mathbf{x}_*$ is locally semi-asymptotically stable from the right if it is both locally semi-stable from the right and $\lim_n f^n(x) = x_*$ for $0 < \mathbf{x} - \mathbf{x}_* < \eta$ for some η. It is globally asymptotically stable if the point is attracting for all $\mathbf{x}$ in the state space.

A cycle is a periodic orbit of distinct points $C = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{K-1}\}$, where $\mathbf{x}_0 = f(\mathbf{x}_{K-1})$ for some $K > 0$. The minimal K generating the cycle is called the period of the cycle. A subset $S \subset \mathbb{R}^n$ is called invariant if $f^k(S) \subset S, \quad k \in \mathbb{Z}$. An invariant set S is called asymptotically stable if there exists a neighborhood U of S such that for any point in U is eventually inside the set S. The stable set of $S \subset \mathbb{R}^n$ is $W^s(S) = \left\{ \mathbf{x} \in \mathbb{R}^n : \lim_{k\to\infty} f^k(\mathbf{x}) \in S \right\}$. If f is invertible, we define the unstable set of $S \subset \mathbb{R}^n$ is $W^u(S) = \left\{ \mathbf{x} \in \mathbb{R}^n : \lim_{k\to\infty} f^{-k}(\mathbf{x}) \in S \right\}$. The unstable set of S for the forward system $f^k, k > 0$ is the stable set of S for the backward system $f^{-k}, k > 0$. It is possible to study the behavior of points that diverge by studying points that converge under the inverse map. We can also classify the stability of K-cycles. We classify the stability of the cycle as a fixed point in the map f^K.

Consider a discrete time dynamical system defined by a diffeomorphism $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. Let x_* be a fixed point of $f(x, \alpha)$ and consider a nearby point x, $|x - x_*| = \epsilon$. Taking a Taylor expansion of the system about the fixed point gives us

$$f(x,\alpha) - x_* \quad = \quad f_x(x_*,\alpha)(x - x_*) + f_{xx}(x_*,\alpha)(x - x_*)^2 + O(|x - x_*|^3).$$

If the Jacobian does not have modulus one and ϵ is small enough, then the contribution by the terms of $O(|x - x_*|^2)$ will be negligible, in which case the behavior of the system is governed by the the behavior of the linearization of the system $f_x(x_*, \alpha)$. We now introduce the idea of a hyperbolic fixed point. Assume that the Jacobian $A := f_x(x_*, \alpha)$ of the system (A1) at a fixed point x_* is non-singular. The fixed point x_* is called hyperbolic if $|f_x(x_*, \alpha)| \neq 1$ and non-hyperbolic if $|f_x(x_*, \alpha)| = 1$. The notion of hyperbolic fixed and non-hyperbolic fixed points generalizes to higher dimensions where it involves the eigenvalues of the Jacobian; see [23,25,38] for more details.

Near a hyperbolic fixed point a non-linear dynamical system behaves its first order Taylor approximation (also known as the linearization of the system). To make this argument rigorous we need to discuss what it means for two dynamical systems to be equivalent. Two systems are topologically equivalent if we can map orbits of one system to orbits of another system in a continuous way that preserves the order of time. The dynamical system (A1) is called topologically equivalent to the system

$$\mathbf{y} \mapsto g(\mathbf{y}, \beta), \quad \mathbf{y} \in \mathbb{R}^n, \quad \beta \in \mathbb{R}^p, \tag{A2}$$

if there exists a homeomorphism of the parameter space $h_p : \mathbb{R}^p \to \mathbb{R}^p$, $\beta = h_p(\alpha)$ and a parameter dependent state space homeomorphism, continuous in the first argument, $h : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ such that, $\mathbf{y} = h(\mathbf{x}, \alpha)$, mapping orbits of the system (A1) at parameter value α onto orbits of the system (A2) at parameter $\beta = h_p(\alpha)$ preserving the direction of time. If h is a diffeomorphism then the systems are called smoothly equivalent.

Let (A1) and (A2) be two topologically equivalent invertible dynamical systems. Consider the orbit of the system under the mapping $f(\mathbf{x}, \alpha)$, $\text{orb}(\mathbf{x}; f, \alpha)$ and the orbit of the system $g(\mathbf{y}, \beta)$, $\text{orb}(\mathbf{y}; g, \beta)$. Topological equivalence means that the homeomorphism $(h(\mathbf{x}, \alpha), h_p(\alpha))$ maps $\text{orb}(\mathbf{x}; f, \alpha)$ to $\text{orb}(\mathbf{y}; g, \beta)$ preserving the order of time. This gives us the following commutative diagram

$$\begin{tikzcd}
\cdots \arrow[r, "f"] \arrow[d, "h"] & f^{-1}(\mathbf{x},\alpha) \arrow[r, "f"] \arrow[d, "h"] & \mathbf{x} \arrow[r, "f"] \arrow[d, "h"] & f(\mathbf{x},\alpha) \arrow[r, "f"] \arrow[d, "h"] & \cdots \\
\cdots \arrow[r, "g"] & g^{-1}(\mathbf{y},\beta) \arrow[r, "g"] & \mathbf{y} \arrow[r, "g"] & g(\mathbf{y},\beta) \arrow[r, "g"] & \cdots .
\end{tikzcd}$$

The orbits being topologically equivalent means that orbit $\mathbf{x}$ under the mapping h should produce the same orbit as mapping $\mathbf{x}$ to $\mathbf{y} = h(\mathbf{x},\alpha)$ computing the orbit of $\mathbf{y}$ under $g(\cdot,\beta)$ and mapping back to $f(\mathbf{x},\alpha)$ by h^{-1}, $f(\mathbf{x},\alpha) = h^{-1} \circ g \circ h(\mathbf{x},\alpha)$. We shall primarily be interested in the behavior of the system in a small neighborhood of an equilibrium point. A system (A1) is called locally topologically equivalent near an equilibrium $\mathbf{x}_*$ to a system (A2) near an equilibrium $\mathbf{y}_*$ if there exists a homeomorphism $h : \mathbb{R}^n \to \mathbb{R}^n$ defined in a small neighborhood U of $\mathbf{x}_*$ with $\mathbf{y}_* = h(\mathbf{x}_*)$ that maps orbits of (A1) in U onto orbits of (A2) in $V = h(U)$, preserving the direction of time.

We now have enough terminology to introduce the following theorem, which shows that the dynamics of a smooth system in the neighborhood of a hyperbolic fixed point are equivalent to the dynamics of the linearization of the system,

Theorem A1 (Grobman–Hartman)**.** *Consider a smooth map*

$$\boldsymbol{x} \mapsto A\boldsymbol{x} + F(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^n, \tag{A3}$$

where A is an $n \times n$ matrix and $F(\boldsymbol{x}) = O(\|\boldsymbol{x}\|^2)$. If $\boldsymbol{x}_* = 0$ is a hyperbolic fixed point of (A3), then (A3) is topologically equivalent near this point to its linearization

$$\boldsymbol{x} \mapsto A\boldsymbol{x}, \quad \boldsymbol{x} \in \mathbb{R}^n.$$

Note Theorem A1 is true for a general n-dimensional system. Theorem A1 provides sufficient conditions to determine the stability of a hyperbolic fixed point of a general discrete time system,

Theorem A2. *Consider a discrete time dynamical systems (A1) where f is a smooth map. Suppose for a fixed point x_* that the eigenvalues of Jacobian $f_x(x_*,\alpha)$ all satisfy $|\lambda| < 1$ then the fixed point is stable. Alternatively, suppose for a fixed point x_* that the eigenvalues of Jacobian $f_x(x_*,\alpha)$ all satisfy $|\lambda| > 1$ then the fixed point is unstable.*

The linearization of the system near a non-hyperbolic fixed point is not sufficient to determine stability of the fixed point and we need to investigate higher order terms. The following theorem provides sufficient condition to check the stability of a smooth one dimensional system at a non-hyperbolic fixed point,

Theorem A3. *Let $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. Suppose that $f(\cdot,\alpha) \in C^3(\mathbb{R};\mathbb{R})$ and x_* is a non-hyperbolic fixed point of f, $x_* = f(x_*,\alpha)$. We have the following cases:*

Case 1: If $f_x(x_*,\alpha) = 1$, then

1. *If $f_{xx}(x_*,\alpha) \neq 0$ then x_* is semi-asymptotically stable from the left if $f_{xx}(x_*,\alpha) > 0$ and semi-asymptotically stable from the right if $f_{xx}(x_*,\alpha) < 0$;*
2. *if $f_{xx}(x_*,\alpha) = 0$ and $f_{xxx}(x_*,\alpha) < 0$ then x_* is asymptotically stable;*
3. *if $f_{xx}(x_*,\alpha) = 0$ and $f_{xxx}(x_*,\alpha) > 0$ then x_* is unstable.*

Case 2: If $f_x(x_*,\alpha) = -1$, then

1. *If $\mathcal{S}f(x_*,\alpha) < 0$, then x_* is asymptotically stable;*

2. *If $\mathcal{S}f(x_*; , \alpha) > 0$, then x_* is unstable.*

where $\mathcal{S}f(x)$ is the Schwarzian derivative of f

$$\mathcal{S}f(x,\alpha) = \frac{f_{xxx}(x,\alpha)}{f_x(x,\alpha)} - \frac{3}{2}\left[\frac{f_{xx}(x,\alpha)}{f_x(x,\alpha)}\right]^2.$$

The Schwarzian derivative controls the higher order behavior in oscillatory systems.

Appendix A.3. Codimension 1 Bifurcations

Until now we have kept the parameter of the system fixed. The study of the change in behavior of a dynamical system as the parameters are varied is called bifurcation theory. A bifurcation occurs when the dynamics of the system at a parameter value α_1 differ from the dynamics of the system at a different parameter value α_2. Changing the parameter in a system may cause a stable fixed point to become unstable, the fixed point may split into multiple fixed points, or a new orbit may form. Each of these is an example of a bifurcation, although these are not the only things that can happen. The point at which a bifurcation occurs is called a bifurcation point. More formally, the parameter α_* is called a bifurcation point if arbitrarily close to it there is α such that $\mathbf{x} \mapsto f(\mathbf{x},\alpha), \mathbf{x} \in \mathbb{R}^n$ is not topologically equivalent to $\mathbf{x} \mapsto f(\mathbf{x},\alpha_*), \mathbf{x} \in \mathbb{R}^n$ in some domain $U \subset \mathbb{R}^n$.

A necessary, but not sufficient condition for bifurcation of a fixed point to occur is for the fixed point to be nonhyperbolic. Theorem A1 together with the implicit function theorem show that in a sufficiently small neighborhood of a hyperbolic fixed point $(\mathbf{x}_*, \alpha_*)$, for each α there is another unique fixed point with the same stability properties as $(\mathbf{x}_*, \alpha)$. So hyperbolic fixed points do not undergo local bifurcations. In the context of discrete systems, a local bifurcation can occur only at a fixed point $(\mathbf{x}_*, \alpha_*)$ when the Jacobian of the system at $(\mathbf{x}_*, \alpha_*)$ has an eigenvalue with modulus one.

Perhaps surprisingly, there are only three types of generic bifurcations that can happen in a discrete system with one parameter. They are the limit point (LP), period doubling (PD) and Neimark–Sacker (NS) bifurcations. The reason for this is fairly simple. It turns out that there is a generic system, called the topological normal form, which undergoes this bifurcation at the origin in the $(\mathbf{x},\alpha)$-plane. For any other system that undergoes the same bifurcation and satisfies certain non-degeneracy conditions there is a local change of coordinates that transforms the system into the topological normal form.

In general the types of bifurcations that can occur are connected to the number of parameters in the system. The minimal number of parameters that must be changed in order for a particular bifurcation to occur in $f(\mathbf{x},\alpha)$ is called the codimension of the bifurcation. A bifurcation is called local if it can be detected in any small neighborhood of the fixed point, otherwise its called global. Global bifurcations are much harder to analyze and since we do not attempt to investigate them in this paper we will not expand upon them further. More detailed results on bifurcations in codimension 1 and 2 can be found in [23,24].

We will now formally define the sufficient conditions for a system to undergo a period doubling or a pitchfork bifurcation. The period doubling bifurcation occurs when a system with a non-hyperbolic fixed point with multiplier $\lambda_1 = -1$ satisfies certain non-degeneracy conditions. There are two types of PD bifurcations. In the super-critical case, a stable 2-cycle is generated when a fixed point becomes unstable. In the sub-critical case, a stable fixed point turns unstable when it coalesces with an unstable 2-cycle (This is true for a general k-cycle. In the super-critical case, a stable $2k$-cycle is generated when a k-cycle becomes unstable. In the sub-critical case, a stable k-cycle turns unstable when it coalesces with an unstable $2k$-cycle). The conditions for a PD bifurcation to occur are given as follows

Theorem A4 (Period Doubling Bifurcation)**.** *Suppose That A One-Dimensional System*

$$x \mapsto f(x,\alpha), \quad x,\alpha \in \mathbb{R},$$

with smooth f, has at $\alpha = 0$ the fixed point $x_* = 0$, and let $\lambda = f_x(0,0) = -1$. Assume the following non-degeneracy conditions are satisfied

1. $1/2(f_{xx}(0,0))^2 + 1/3 f_{xxx}(0,0) \neq 0$
2. $f_{x\alpha}(0,0) \neq 0$

Then there are smooth invertible coordinate and parameter changes transforming the system into

$$\eta \mapsto -(1+\beta) \pm \eta^3 + O(\eta^4). \tag{A4}$$

An classical example of a period doubling bifurcation can be seen in the logistic map $f(x,\mu) = \mu x(1-x)$, for $x \in [0,1]$. The bifurcation occurs at the point $(x_*, \mu_*) = (2/3, 3)$. The logistic map has two fixed points. One fixed point is at $x = 0$ and the other is at $x = (\mu - 1)/\mu$. We will ignore the fixed point at $x = 0$ since it is repelling for $\mu > 1$. We look at the behavior of the system in a small neighborhood of $\mu_* = 3$. For $\mu = 2.9$, the fixed point $x_* = (\mu-1)/\mu$ is a hyperbolic attracting fixed point since $|f_x(x_*, 2.9)| = |2 - \mu| < 1$. For $\mu = 3$ the fixed point $x_* = (\mu-1)/\mu$ is a non-hyperbolic fixed point since $f_x(x_*, 2.9) = 2 - \mu = -1$. Checking the Schwarzian derivative shows that the fixed point is asymptotically stable. For $\mu = 3.1$, $x_* = (\mu-1)/\mu$ becomes a repelling fixed point. The points in $(0, x_*) \cup (x_*, 1)$ converge to the attracting 2-cycle $C = \{0.558014, 0.7645665\}$. A super-critical period doubling bifurcation has occurred in the system formed by the logistic map. As the parameter μ increases we see a stable fixed point degenerate and a stable 2-cycle is formed.

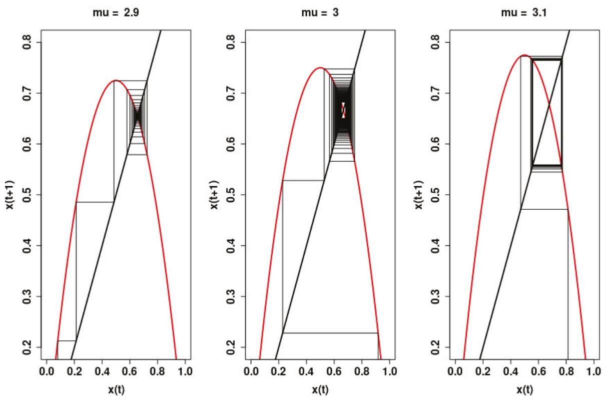


Figure A1. The above plots are cobweb diagrams for the logistic map $f(x,\mu) = \mu x(1-x)$, for $x \in [0,1]$, with parameters $\mu = 2.9$, $\mu = 3$ and $\mu = 3.1$, respectively. For $\mu = 2.9$ the system has one stable fixed point $x_* = (\mu-1)/\mu$. For $\mu = 3$, the system has one non-hyperbolic fixed point $x_* = (\mu-1)/\mu$ which is asymptotically stable attracting; the plot was not iterated long enough to see convergence. For $\mu = 3.1$, the system has a hyperbolic repelling fixed point $x_* = (\mu-1)/\mu$ and an asymptotically stable attracting two cycle $C = \{0.558014, 0.7645665\}$.

The second iterate of a map that undergoes a PD bifurcation undergoes a bifurcation know as the pitchfork bifurcation. A system that undergoes a super-critical pitchfork bifurcation when a stable fixed point becomes unstable and two stable fixed points appear in the system. A system that undergoes a sub-critical pitchfork bifurcation when two stable fixed points coalesce with an unstable fixed point, the unstable fixed point becomes stable as the parameter crosses the bifurcation point. Below we present extra details pertaining to the period doubling bifurcation and its relation to the pitchfork bifurcation.

Consider the one-dimensional system

$$x \mapsto -(1+\alpha)x + x^3 = f(x,\alpha).$$

The map $f(x, \alpha)$ is invertible in a small neighborhood of $(0, 0)$. The system has a fixed point at $x_* = 0$ for all α, with eigenvalue $-(1 + \alpha)$. For small $\alpha < 0$ the fixed point is hyperbolic stable and for $\alpha > 0$ is it hyperbolic unstable. For $\alpha = 0$ the fixed point is non-hyperbolic, but is asymptotically stable.

Consider the second iterate of $f(x, \alpha)$

$$\begin{aligned} f^2(x, \alpha) &= -(1+\alpha) f(x, \alpha) + (f(x, \alpha))^3 \\ &= (1+\alpha)^2 x - \Big[(1+\alpha)(2 + 2\alpha + \alpha^2)\Big] x^3 + O(x^5). \end{aligned}$$

The second iterate has a trivial fixed point at $x_* = 0$ and for $\alpha > 0$ it has two non-trivial stable fixed points $x_1 = (\sqrt{\alpha} + O(\alpha))$, $x_1 = -(\sqrt{\alpha} + O(\alpha))$ that form a two cycle

$$x_2 = f(x_1, \alpha), \quad x_1 = f(x_2, \alpha).$$

The conditions for a generic pitchfork bifurcation can be found in [25]

Theorem A5 (Pitchfork Bifurcation)**.** *For A System*

$$x \mapsto f(x, \alpha), \quad x, \alpha \in \mathbb{R}$$

having non-hyperbolic fixed point at $x_* = 0$, $\alpha_* = 0$ with $f_x(0, 0) = 1$ undergoes a pitchfork bifurcation at $(x_*, \alpha_*) = (0, 0)$ if

$$f_\alpha(0,0) = 0, \quad f_{xx}(0,0) = 0, \quad f_{xxx}(0,0) \neq 0, \quad f_{x\alpha}(0,0) \neq 0.$$

A pitchfork bifurcation is super-critical if $-f_{xxx}(x_*, \alpha_*)/f_{\alpha x}(x_*, \alpha_*) > 0$ and sub-critical if $-f_{xxx}(x_*, \alpha_*)/f_{\alpha x}(x_*, \alpha_*) < 0$

An example of a pitchfork bifurcation can be seen in the second iteration of the logistic map $f^2(x, \mu) = \mu^2 x(1-x)(1 - \mu x(1-x))$, for $x \in [0, 1]$. The bifurcation occurs at the point $(x_*, \mu_*) = (2/3, 3)$. For $\mu \leq 3$, the second iteration of the logistic map has the same fixed points as the first iteration. One fixed point is at $x = 0$ and the other is at $x = (\mu - 1)/\mu$. We will ignore the fixed point at $x = 0$ since it is repelling for $\mu > 1$. We look at the behavior of the system in a small neighborhood of $\mu_* = 3$. For $\mu = 2.9$, the fixed point $x_* = (\mu - 1)/\mu$ is a hyperbolic attracting fixed point since $|f_x^2(x_*, 2.9)| < 1$. For $\mu = 3$ the fixed point $x_* = (\mu - 1)/\mu$ is non-hyperbolic since $f_x^2(x_*, 2.9) = 2 - \mu = 1$. Checking the higher order derivative shows that the fixed point is asymptotically stable. For $\mu = 3.1$, $x_* = (\mu - 1)/\mu$ becomes a repelling fixed point. Using numerical methods we find two additional fixed points, $x_1 = 0.558014$ and $x_2 = 0.7645665$, both of which are attracting. A super-critical pitchfork bifurcation has occurred in the system formed by the logistic map. As the parameter μ increases we see a stable fixed point degenerates to an unstable fixed point and two stable fixed points.

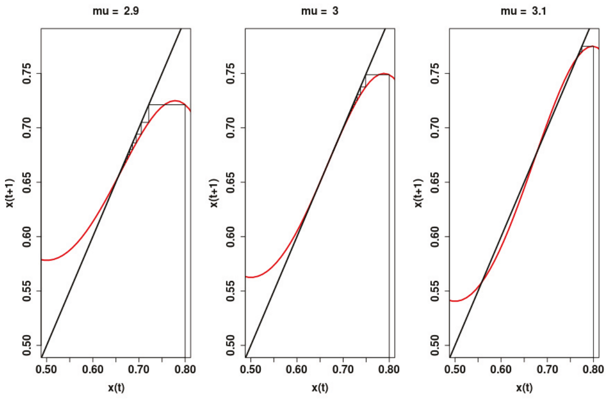


Figure A2. The above plots are cobweb diagrams for the second iterate of the logistic map $f(x,\mu) = \mu x(1-x)$, for $x \in [0,1]$, with parameters $\mu = 2.9$ and $\mu = 3.1$, respectively. For $\mu = 2.9$ the system has one stable fixed point $x_* = (\mu-1)/\mu$. For $\mu = 3.1$, the system has a hyperbolic repelling fixed point $x_* = (\mu-1)/\mu$ and two asymptotically stable attracting fixed points $x_1 = 0.0558014$ and $x_2 = 0.7645665$.

References

1. Bishop, C. *Pattern Recognition and Machine Learning*; Information Science and Statistics; Springer: Berlin/Heidelberger, Germany, 2006.
2. MacKay, D.J.; Mac Kay, D.J. *Information Theory, Inference and Learning Algorithms*; Cambridge University Press: Cambridge, UK, 2003.
3. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D. Variational Inference: A Review for Statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877. [CrossRef]
4. Zhang, C.; Bütepage, J.; Kjellström, H.; Mandt, S. Advances in variational inference. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 2008–2026. [CrossRef] [PubMed]
5. Parisi, G. *Statistical Field Theory*; Frontiers in Physics; Addison-Wesley: Boston, MA, USA, 1988.
6. Opper, M.; Saad, D. *Advanced Mean Field Methods: Theory and Practice*; MIT Press: Cambridge, MA, USA, 2001.
7. Gabrié, M. Mean-field inference methods for neural networks. *J. Phys. A Math. Theor.* **2020**, *53*, 223002. [CrossRef]
8. Alquier, P.; Ridgway, J.; Chopin, N. On the properties of variational approximations of Gibbs posteriors. *J. Mach. Learn. Res.* **2016**, *17*, 1–41.
9. Pati, D.; Bhattacharya, A.; Yang, Y. On statistical optimality of variational Bayes. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Canary Islands, Spain, 9–11 April 2018; pp. 1579–1588.
10. Yang, Y.; Pati, D.; Bhattacharya, A. α-Variational inference with statistical guarantees. *Ann. Stat.* **2020**, *48*, 886–905. [CrossRef]
11. Chérief-Abdellatif, B.E.; Alquier, P. Consistency of variational Bayes inference for estimation and model selection in mixtures. *Electron. J. Stat.* **2018**, *12*, 2995–3035. [CrossRef]
12. Wang, Y.; Blei, D.M. Frequentist consistency of variational Bayes. *J. Am. Stat. Assoc.* **2019**, *114*, 1147–1161. [CrossRef]
13. Wang, Y.; Blei, D.M. Variational Bayes under Model Misspecification. *arXiv* **2019**, arXiv:1905.10859.
14. Wang, B.; Titterington, D. *Inadequacy of Interval Estimates Corresponding to Variational Bayesian Approximations*; AISTATS; Citeseer: Princeton, NJ, USA, 2005.
15. Wang, B.; Titterington, D. Convergence properties of a general algorithm for calculating variational Bayesian estimates for a normal mixture model. *Bayesian Anal.* **2006**, *1*, 625–650. [CrossRef]
16. Zhang, A.Y.; Zhou, H.H. Theoretical and Computational Guarantees of Mean Field Variational Inference for Community Detection. *arXiv* **2017**, arXiv:math.ST/1710.11268.

17. Mukherjee, S.S.; Sarkar, P.; Wang, Y.R.; Yan, B. Mean field for the stochastic blockmodel: Optimization landscape and convergence issues. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2018; pp. 10694–10704.
18. Sarkar, P.; Wang, Y.; Mukherjee, S.S. When random initializations help: A study of variational inference for community detection. *arXiv* **2019**, arXiv:1905.06661.
19. Yin, M.; Wang, Y.X.R.; Sarkar, P. A Theoretical Case Study of Structured Variational Inference for Community Detection. *Proc. Mach. Learn. Res.* **2020**, *108*, 3750–3761.
20. Ghorbani, B.; Javadi, H.; Montanari, A. An Instability in Variational Inference for Topic Models. *arXiv* **2018**, arXiv:stat.ML/1802.00568.
21. Jain, V.; Koehler, F.; Mossel, E. The Mean-Field Approximation: Information Inequalities, Algorithms, and Complexity. *arXiv* **2018**, arXiv:cs.LG/1802.06126.
22. Koehler, F. Fast Convergence of Belief Propagation to Global Optima: Beyond Correlation Decay. *arXiv* **2019**, arXiv:cs.LG/1905.09992.
23. Kuznetsov, Y. *Elements of Applied Bifurcation Theory*; Applied Mathematical Sciences; Springer: New York, NY, USA, 2008.
24. Kuznetsov, Y.; Meijer, H. *Numerical Bifurcation Analysis of Maps*; Cambridge Monographs on Applied and Computational Mathematics; Cambridge University Press: Cambridge, UK, 2019.
25. Wiggins, S. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*; Texts in Applied Mathematics; Springer: New York, NY, USA, 2003.
26. Friedli, S.; Velenik, Y. *Statistical Mechanics of Lattice Systems: A Concrete Mathematical Introduction*; Cambridge University Press: Cambridge, UK, 2017.
27. Ising, E. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik* **1925**, *31*, 253–258. [CrossRef]
28. Onsager, L. Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition. *Phys. Rev.* **1944**, *65*, 117–149. [CrossRef]
29. Toda, M.; Toda, M.; Saito, N.; Kubo, R.; Saito, N. *Statistical Physics I: Equilibrium Statistical Mechanics*; Springer Series in Solid-State Sciences; Springer: Berlin/Heidelberg, Germany, 2012.
30. Moessner, R.; Ramirez, A.P. Geometrical frustration. *Phys. Today* **2006**, *59*, 24. [CrossRef]
31. Basak, A.; Mukherjee, S. Universality of the mean-field for the Potts model. *Probab. Theory Relat. Fields* **2017**, *168*, 557–600. [CrossRef]
32. Blanca, A.; Chen, Z.; Vigoda, E. Swendsen-Wang dynamics for general graphs in the tree uniqueness region. *Random Struct. Algorithms* **2019**, *56*, 373–400. [CrossRef]
33. Guo, H.; Jerrum, M. Random cluster dynamics for the Ising model is rapidly mixing. *Ann. Appl. Probab.* **2018**, *28*, 1292–1313. [CrossRef]
34. Oostwal, E.; Straat, M.; Biehl, M. Hidden Unit Specialization in Layered Neural Networks: ReLU vs. Sigmoidal Activation. *arXiv* **2019**, arXiv:1910.07476.
35. Çakmak, B.; Opper, M. A Dynamical Mean-Field Theory for Learning in Restricted Boltzmann Machines. *arXiv* **2020**, arXiv:2005.01560.
36. Blum, E.; Wang, X. Stability of fixed points and periodic orbits and bifurcations in analog neural networks. *Neural Netw.* **1992**, *5*, 577–587. [CrossRef]
37. Grimmett, G. *The Random-Cluster Model*; Grundlehren der Mathematischen Wissenschaften; Springer: Berlin/Heidelberg, Germany, 2006.
38. Elaydi, S. *Discrete Chaos: With Applications in Science and Engineering*; CRC Press: New York, NY, USA, 2007.

MDPI
St. Alban-Anlage 66
4052 Basel
Switzerland

Tel: +41 61 683 77 34
Fax: +41 61 302 89 18

www.mdpi.com

ISBN 978-3-0365-3790-0