Computing Entropies with Nested Sampling

Brendon J. Brewer

Department of Statistics, The University of Auckland, Auckland 1142, New Zealand; bj.brewer@auckland.ac.nz

Abstract: The Shannon entropy, and related quantities such as mutual information, can be used to quantify uncertainty and relevance. However, in practice, it can be difficult to compute these quantities for arbitrary probability distributions, particularly if the probability mass functions or densities cannot be evaluated. This paper introduces a computational approach, based on Nested Sampling, to evaluate entropies of probability distributions that can only be sampled. I demonstrate the method on three examples: a simple Gaussian example where the key quantities are available analytically; (ii) an experimental design example about scheduling observations in order to measure the period of an oscillating signal; and (iii) predicting the future from the past in a heavy-tailed scenario.

Keywords: information theory; entropy; mutual information; Monte Carlo; nested sampling; Bayesian inference

1. Introduction

If an unknown quantity x has a discrete probability distribution $p(x)$, the Shannon entropy [1,2] is defined as

$$H(x) = -\sum_x p(x) \log p(x), \tag{1}$$

where the sum is over all of the possible values of x under consideration. The entropy quantifies the degree to which the issue "what is the value of x, precisely?" remains unresolved [3]. More generally, the 'Kullback–Leibler divergence' quantifies the degree to which a probability distribution $p(x)$ departs from a 'base' or 'prior' probability distribution $q(x)$ [4,5]:

$$D_{\mathrm{KL}}(p \,||\, q) = \sum_x p(x) \log \left[\frac{p(x)}{q(x)} \right]. \tag{2}$$

If $q(x)$ is taken to be uniform, this is equivalent (up to an additive constant and a change of sign) to the Shannon entropy $H(x)$. With the negative sign reinstated, Kullback–Leibler divergence is sometimes called relative entropy, and is more fundamental than the Shannon entropy [4]. However, this paper focuses on the Shannon entropy for simplicity and because the resulting algorithm calculates the expected value of a log-probability.

The Shannon entropy can be interpreted straightforwardly as a measure of uncertainty. For example, if there is only one possible value that has a probability of one, $H(x)$ is zero. Conventionally, $0 \times \log 0$ is defined to be equal to $\lim_{x \to 0^+} (x \log x)$, which is zero; i.e., 'possibilities' with probability zero do not contribute to the sum. If there are N possibilities with equal probabilities $1/N$ each, then the entropy is $H(x) = \log(N)$. If the space of possible x-values is continuous so that $p(x)$ is a probability density function, the differential entropy

$$H(x) = -\int p(x) \log p(x) \, dx \tag{3}$$

quantifies uncertainty by generalising the log-volume of the plausible region, where volume is defined with respect to dx. Crucially, in continuous spaces, the differential entropy is not invariant under changes of coordinates, so any value given for a differential entropy must be understood with respect to the coordinate system that was used. Even in discrete cases, using the Shannon entropy assumes that each possibility contributes equally when counting possibilities—if this is inappropriate, the relative entropy should be used.

Entropies, relative entropies, and related quantities tend to be analytically available only for a few families of probability distributions. On the numerical side, if $\log p(x)$ can be evaluated for any x, then simple Monte Carlo will suffice for approximating $H(x)$. On the other hand, if $p(x)$ can only be *sampled* but not evaluated (for example, if $p(x)$ is a marginal or conditional distribution, or the distribution of a quantity derived from x), then this will not work. Kernel density estimation or the like e.g., [6] may be effective in this case, but is unlikely to generalise well to high dimensions.

This paper introduces a computational approach to evaluating the Shannon entropy $H(x)$ of probability distributions that can only be sampled using Markov Chain Monte Carlo (MCMC). The key idea is that Nested Sampling can create an unbiased estimate of a *log*-probability, and entropies are averages of log-probabilities.

Notation and Conventions

Throughout this paper, I use the compact 'overloaded' notation for probability distributions favoured by many Bayesian writers [7,8], writing $p(x)$ for either a probability mass function or a probability density function, instead of the more cumbersome $P(X = x)$ or $f_X(x)$. In the compact notation, there is no distinction between the 'random variable' itself (X) and a similarly-named dummy variable (x). Probability distributions are implicitly conditional on some prior information, which is omitted from the notation unless necessary. All logarithms are written log and any base can be used, unless otherwise specified (for example by writing ln or $\log_{10}$). Any numerical values given for the value of specific entropies are in *nats*, i.e., the natural logarithm was used.

Even though the entropy is written as $H(x)$, it is imperative that we remember that it is not a property of the value of x itself, but a property of the probability distribution used to describe a state of ignorance about x. Throughout this paper, $H(x)$ is used as notation for both Shannon entropies (in discrete cases) and differential entropies (in continuous cases). Which one it is should be clear from the context of the problem at hand. I sometimes write general formulae in terms of sums (i.e., as they would appear in a discrete problem), and sometimes as integrals, as they would appear in a continuous problem.

2. Entropies in Bayesian Inference

Bayesian inference is the use of probability theory to describe uncertainty, often about unknown quantities ('parameters') $\boldsymbol{\theta}$. Some data $\boldsymbol{d}$, initially unknown but thought to be relevant to $\boldsymbol{\theta}$, is obtained. The prior information leads the user to specify a prior distribution $p(\boldsymbol{\theta})$ for the unknown parameters, along with a conditional distribution $p(\boldsymbol{d}|\boldsymbol{\theta})$ describing prior knowledge of how the data is related to the parameters (i.e., if the parameters were known, what data would likely be observed?). By the product rule, this yields a *joint prior*

$$p(\boldsymbol{\theta}, \boldsymbol{d}) = p(\boldsymbol{\theta})p(\boldsymbol{d}|\boldsymbol{\theta}), \qquad (4)$$

which is the starting point for Bayesian inference [5,9]. In practical applications, the following operations are usually feasible and have a low computational cost:

1. Samples can be generated from the prior $p(\boldsymbol{\theta})$;
2. Simulated datasets can be generated from $p(\boldsymbol{d}|\boldsymbol{\theta})$ for any given value of $\boldsymbol{\theta}$;
3. The likelihood, $p(\boldsymbol{d}|\boldsymbol{\theta})$, can be evaluated cheaply for any $\boldsymbol{d}$ and $\boldsymbol{\theta}$. Usually, it is the log-likelihood that is actually implemented, for numerical reasons.

Throughout this paper, I assume that these operations are available and inexpensive.

2.1. The Relevance of Data

The entropy of the prior $p(\boldsymbol{\theta})$ describes the degree to which the question "what is the value of $\boldsymbol{\theta}$, precisely?" remains unanswered, while the entropy of the joint prior $p(\boldsymbol{\theta}, \boldsymbol{d})$ describes the degree to which the question "what is the value of the pair $(\boldsymbol{\theta}, \boldsymbol{d})$, precisely?" remains unanswered. The degree to which the question "what is the value of $\boldsymbol{\theta}$?" would remain unresolved if $\boldsymbol{d}$ were resolved is given by the conditional entropy

$$H(\boldsymbol{\theta}|\boldsymbol{d}) = -\sum_{\boldsymbol{d}} p(\boldsymbol{d}) \sum_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\boldsymbol{d}) \log p(\boldsymbol{\theta}|\boldsymbol{d}), \tag{5}$$

which is the expected value of the entropy of the posterior, averaged over all possible datasets that might be observed.

One might wish to compute $H(\boldsymbol{\theta}|\boldsymbol{d})$, perhaps to compare it to $H(\boldsymbol{\theta})$ and quantify how much might be learned about $\boldsymbol{\theta}$. This would be difficult because the expression for the posterior distribution

$$p(\boldsymbol{\theta}|\boldsymbol{d}) = \frac{p(\boldsymbol{\theta})p(\boldsymbol{d}|\boldsymbol{\theta})}{p(\boldsymbol{d})} \tag{6}$$

contains the marginal likelihood:

$$p(\boldsymbol{d}) = \sum_{\boldsymbol{\theta}} p(\boldsymbol{\theta})p(\boldsymbol{d}|\boldsymbol{\theta}), \tag{7}$$

also known as the 'evidence', which tends to be computable but costly, especially when the summation is replaced by an integration over a high-dimensional parameter space.

It is important to distinguish between $H(\boldsymbol{\theta}|\boldsymbol{d})$ and the entropy of $\boldsymbol{\theta}$ given a particular value of $\boldsymbol{d}$, which might be written $H(\boldsymbol{\theta}|\boldsymbol{d} = \boldsymbol{d}_{\mathrm{obs}})$ ('obs' for observed). The former measures the degree to which one question answers another ex ante, and is a function of two questions. The latter measures the degree to which a question remains unresolved after conditioning on a specific statement (ex post), and is a function of a question and a statement.

2.2. Mutual Information

The mutual information is another way of describing the relevance of the data to the parameters. Its definition, and relation to other quantities, is

$$I(\boldsymbol{\theta}; \boldsymbol{d}) = \sum_{\boldsymbol{\theta}} \sum_{\boldsymbol{d}} p(\boldsymbol{\theta}, \boldsymbol{d}) \log \left[\frac{p(\boldsymbol{\theta}, \boldsymbol{d})}{p(\boldsymbol{\theta})p(\boldsymbol{d})} \right] \tag{8}$$

$$= H(\boldsymbol{\theta}) + H(\boldsymbol{d}) - H(\boldsymbol{\theta}, \boldsymbol{d}) \tag{9}$$

$$= H(\boldsymbol{d}) - H(\boldsymbol{d}|\boldsymbol{\theta}) \tag{10}$$

$$= H(\boldsymbol{\theta}) - H(\boldsymbol{\theta}|\boldsymbol{d}). \tag{11}$$

The mutual information can also be written as the expected value (with respect to the prior over datasets $p(\boldsymbol{d})$) of the Kullback–Leibler divergence from prior to posterior:

$$I(\boldsymbol{\theta}; \boldsymbol{d}) = \sum_{\boldsymbol{d}} p(\boldsymbol{d}) D_{\mathrm{KL}} \big(p(\boldsymbol{\theta}|\boldsymbol{d}) \,\|\, p(\boldsymbol{\theta}) \big). \tag{12}$$

In terms of the prior, likelihood, and evidence, it is

$$I(\boldsymbol{\theta}; \boldsymbol{d}) = \sum_{\boldsymbol{\theta}} \sum_{\boldsymbol{d}} p(\boldsymbol{\theta})p(\boldsymbol{d}|\boldsymbol{\theta}) \left[\log p(\boldsymbol{d}|\boldsymbol{\theta}) - \log p(\boldsymbol{d}) \right], \tag{13}$$

i.e., the mutual information is the prior expected value of the log likelihood minus the log evidence. As with the conditional entropy, the computational difficulty appears in the form of the log evidence, $\log p(\boldsymbol{d}) = \log\left(\sum_{\boldsymbol{\theta}} p(\boldsymbol{\theta})p(\boldsymbol{d}|\boldsymbol{\theta})\right)$, which must be evaluated or estimated for many possible datasets.

For experimental design purposes, maximising the expected amount of information obtained from the data is a sensible goal. Formally, either maximising $I(\boldsymbol{\theta};\boldsymbol{d})$ or minimising $H(\boldsymbol{\theta}|\boldsymbol{d})$ will produce the same result because the prior $p(\boldsymbol{\theta})$ does not vary with the experimental design. Reference priors [10] also maximise $I(\boldsymbol{\theta};\boldsymbol{d})$ but vary the prior $p(\boldsymbol{\theta})$ in the maximisation process while keeping the experimental design fixed.

3. Nested Sampling

Nested Sampling (NS), introduced by Skilling [11], is an algorithm whose aim is to calculate the evidence

$$p(\boldsymbol{\theta}) = \int p(\boldsymbol{\theta})p(\boldsymbol{d}|\boldsymbol{\theta})\,d\boldsymbol{\theta}, \tag{14}$$

or, in the simplified notation common when discussing computational matters,

$$Z = \int \pi(\boldsymbol{\theta})L(\boldsymbol{\theta})\,d\boldsymbol{\theta}, \tag{15}$$

where $\boldsymbol{\theta}$ is the unknown parameter(s), π is the prior distribution, and L is the likelihood function. The original NS algorithm, and variations within the same family [12–14], have become popular tools in Bayesian data analysis [15–17] and have also found use in statistical physics [18–20], which was another of its aims. NS also estimates the Kullback–Leibler divergence from the prior $\pi(\boldsymbol{\theta})$ to the posterior $\pi(x)L(x)/Z$, which Skilling calls the information. This is a measure of how much was learned about $\boldsymbol{\theta}$ from the specific dataset, and is also useful for defining a termination criterion for NS.

While Z is a simple expectation with respect to π, the distribution of L-values implied by π tends to be very heavy-tailed, which is why simple Monte Carlo does not work. Equivalently, the integral over $\boldsymbol{\theta}$ is dominated by very small regions where $L(\boldsymbol{\theta})$ is high.

To overcome this, NS evolves a population of N particles in the parameter space towards higher likelihood regions. The particles are initially initialised from the prior $\pi(\boldsymbol{\theta})$, and the particle with the lowest likelihood, L_1^*, is found and its details are recorded as output. This worst particle is then discarded and replaced by a new particle drawn from the prior π but subject to the constraint that its likelihood must be above L_1^*. This is usually achieved by cloning a surviving particle and evolving it with MCMC to explore the prior distribution but with a constraint on likelihood value, $L(\boldsymbol{\theta}) > L_1^*$. This 'constrained prior' distribution has probability density proportional to

$$\pi(\boldsymbol{\theta})\mathbb{1}\left(L(\boldsymbol{\theta}) > L_1^*\right), \tag{16}$$

where $\mathbb{1}()$ is an indicator function that returns one if the argument is true and zero if it is false.

This process is repeated, resulting in an increasing sequence of likelihoods

$$L_1^*, L_2^*, L_3^*, ... \tag{17}$$

Defining $X(\ell)$ as the amount of prior mass with likelihood greater than some threshold ℓ,

$$X(\ell) = \int \pi(\boldsymbol{\theta})\mathbb{1}\left(L(\boldsymbol{\theta}) > \ell\right)\,d\boldsymbol{\theta}, \tag{18}$$

and each discarded point in the sequence can be assigned an X-value, which transforms Z to a one-dimensional integral. Skilling's [11] key insight was that the X-values of the discarded points are approximately known from the nature of the algorithm. Specifically, each iteration reduces the prior mass by approximately a factor $e^{-1/N}$. More accurately, the conditional distribution of X_i given X_{i-1}

(and the definition of the algorithm) is obtainable by letting $t_i \sim \text{Beta}(N, 1)$ and setting $X_{i+1} := t_i X_{i-1}$ (where $X_1 = t_1$). The t_i quantities are the proportion of the remaining prior mass that is *retained* at each iteration, after imposing the constraint of the latest L^* value.

The Sequence of X-Values

The sequence of *X*-values is defined by

$$X_1 = t_1, \tag{19}$$

$$X_2 = t_2 X_1, \tag{20}$$

$$X_3 = t_3 X_2, \tag{21}$$

and so on, i.e., X_i can be written as

$$X_i = \prod_{k=1}^{i} t_k. \tag{22}$$

Taking the log and multiplying both sides by –1, we get

$$-\ln X_i = \sum_{k=1}^{i} \left(-\ln t_k\right). \tag{23}$$

Consider t_i, the fraction of the remaining prior mass that is retained in each NS iteration. By a change of variables, the distribution for these compression factors $t_i \sim \text{Beta}(N, 1)$ corresponds to an exponential distribution for $-\ln(t_i)$:

$$-\ln t_i \sim \text{Exponential}(N). \tag{24}$$

From Equations 23 and 24, we see that the sequence of $-\ln X$ values produced by a Nested Sampling run can be considered as a Poisson process with a rate equal to N, the number of particles. This is why separate NS runs can be simply merged [11,21]. Walter [22] showed how this view of the NS sequence of points can be used to construct a version of NS that produces unbiased (in the frequentist sense) estimates of the evidence. However, it can also be used to construct an unbiased estimator of a log-probability, which is more relevant to information theoretic quantities discussed in the present paper.

Consider a particular likelihood value ℓ whose corresponding *X*-value we would like to know. Since the $-\ln X_i$ values have the same distribution as the arrival times of a Poisson process with rate N, the probability distribution for the number of points in an interval of length w is Poisson with expected value Nw. Therefore, the expected number of NS discarded points with likelihood below ℓ is $-N \ln X(\ell)$:

$$\langle n(L(\boldsymbol{\theta}_i) \leq \ell) \rangle = -N \ln X(\ell). \tag{25}$$

We can therefore take $n(L(\boldsymbol{\theta}_i) \leq \ell)/N$, the number of points in the sequence with likelihood below ℓ divided by the number of NS particles, as an unbiased estimator of $-\ln X(\ell)$. The standard deviation of the estimator is $\sqrt{-\ln X(\ell)/N}$.

4. The Algorithm

The above insight, that the number of NS discarded points with likelihood below ℓ has expected value $-N \ln X(\ell)$, is the basis of the algorithm. If we wish to measure a log-probability $-\ln X(\ell)$, we can use NS to do it. $\pi(\boldsymbol{\theta})$ and $L(\boldsymbol{\theta})$ need not be the prior and likelihood, respectively, but can be

any probability distribution (over any space) and any function whose expected value is needed for a particular application.

See Algorithm 1 for a step-by-step description of the algorithm. The algorithm is written in terms of an unknown quantity $\boldsymbol{\theta}$ whose entropy is required. In applications, $\boldsymbol{\theta}$ may be parameters, a subset of the parameters, data, some function of the data, parameters and data jointly, or whatever.

Algorithm 1 The algorithm which estimates the expected value of the depth:
$-\int p(\boldsymbol{\theta}_{\text{ref}}) \int p(\boldsymbol{\theta}) \log \left[P(d(\boldsymbol{\theta}; \boldsymbol{\theta}_{\text{ref}}) < r \mid \boldsymbol{\theta}_{\text{ref}}) \right] d\boldsymbol{\theta} \, d\boldsymbol{\theta}_{\text{ref}}$,
that is, minus the expected value of the log-probability of a small region near $\boldsymbol{\theta}_{\text{ref}}$, which can be converted to an estimate of an entropy or differential entropy. The part highlighted in green is standard Nested Sampling with quasi-prior $p(\boldsymbol{\theta})$ and quasi-likelihood given by minus a distance function $d(\boldsymbol{\theta}; \boldsymbol{\theta}_{\text{ref}})$. The final result, $\widehat{h_{\text{final}}}$, is an estimate of the expected depth.

Set the numerical parameters:

$N \in \{1, 2, 3, ...\}$ ▷ the number of Nested Sampling particles to use

$r \geq 0$ ▷ the tolerance

$\widehat{\mathbf{h}} \leftarrow \{\}$ ▷ Initialise an empty list of results

while more iterations desired **do**

     $k \leftarrow 0$ ▷ Initialise counter

     Generate $\boldsymbol{\theta}_{\text{ref}}$ from $p(\boldsymbol{\theta})$ ▷ Generate a reference point

     Generate $\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ from $p(\boldsymbol{\theta})$ ▷ Generate initial NS particles

     Calculate $d_i \leftarrow d(\boldsymbol{\theta}_i; \boldsymbol{\theta}_{\text{ref}})$ for all i ▷ Calculate distance of each particle from the reference point

     $i^* \leftarrow \text{argmin}(\lambda i \rightarrow d_i)$ ▷ Find the worst particle (greatest distance)

     $d_{\text{max}} \leftarrow d_{i^*}$ ▷ Find the greatest distance

     while $d_{\text{max}} > r$ **do**

         Replace $\boldsymbol{\theta}_{i^*}$ with $\boldsymbol{\theta}_{\text{new}}$ from $p\left(\boldsymbol{\theta} \mid d(\boldsymbol{\theta}) < d_{\text{max}}\right)$ ▷ Replace worst particle

         Calculate $d_{i^*} \leftarrow d(\boldsymbol{\theta}_{\text{new}}; \boldsymbol{\theta}_{\text{ref}})$ ▷ Calculate distance of new particle from reference point

         $i^* \leftarrow \text{argmin}(\lambda i \rightarrow d_i)$ ▷ Find the worst particle (greatest distance)

         $d_{\text{max}} \leftarrow d_{i^*}$ ▷ Find the greatest distance

         $k \leftarrow k + 1$ ▷ Increment counter k

     $\widehat{\mathbf{h}} \leftarrow \text{append}\left(\widehat{\mathbf{h}}, k/N\right)$ ▷ Append latest estimate to results

$\widehat{h_{\text{final}}} \leftarrow \frac{1}{\text{num\_iterations}} \sum \widehat{\mathbf{h}}$ ▷ Average results

---

The idea is to generate a reference particle $\boldsymbol{\theta}_{\text{ref}}$ from the distribution $p(\boldsymbol{\theta})$ whose entropy is required. Then, a Nested Sampling run evolves a set of particles $\{\boldsymbol{\theta}\}_{i=1}^{N}$, initially representing $p(\boldsymbol{\theta})$, towards $\boldsymbol{\theta}_{\text{ref}}$ in order to measure the log-probability near $\boldsymbol{\theta}_{\text{ref}}$ (see Figure 1). Nearness is defined using a distance function $d(\boldsymbol{\theta}; \boldsymbol{\theta}_{\text{ref}})$, and the number of NS iterations taken to reduce the distance to below some threshold r (for 'radius') provides an unbiased estimate of

$$(\text{depth}) = -\log \left[P(d(\boldsymbol{\theta}; \boldsymbol{\theta}_{\text{ref}}) < r) \right], \tag{26}$$

which I call the 'depth'. For example, if $N = 10$ Nested Sampling particles are used, and it takes 100 iterations to reduce the distance to below r, then the depth is 10 nats.
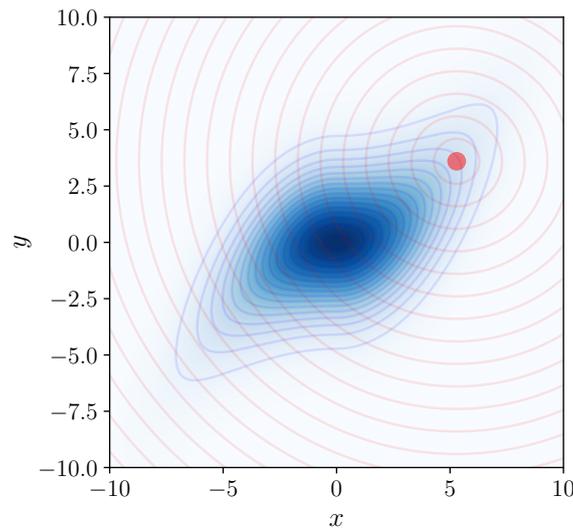
Figure 1. To evaluate the log-probability (or density) of the blue probability distribution near the red point, Nested Sampling can be used, with the blue distribution playing the role of the "prior" in NS, and the Euclidean distance from the red point (illustrated with red contours) playing the role of the negative log-likelihood. Averaging over selections of the red point gives an estimate of the entropy of the blue distribution.

If we actually want the differential entropy

$$H(\boldsymbol{\theta}) = -\int p(\boldsymbol{\theta}) \log p(\boldsymbol{\theta})\, d\boldsymbol{\theta}, \tag{27}$$

we can use the fact that density equals mass divided by volume. Assume r is small, so that

$$P(d(\boldsymbol{\theta}; \boldsymbol{\theta}_{\text{ref}}) < r \,|\, \boldsymbol{\theta}_{\text{ref}}) \approx p(\boldsymbol{\theta}_{\text{ref}}) \int_{d(\boldsymbol{\theta}; \boldsymbol{\theta}_{\text{ref}}) < r} d\boldsymbol{\theta} \tag{28}$$

$$= p(\boldsymbol{\theta}_{\text{ref}})\, V, \tag{29}$$

where V is the volume of the region where $d(\boldsymbol{\theta}; \boldsymbol{\theta}_{\text{ref}}) < r$. Then, $H(\boldsymbol{\theta}) = \langle (\text{depth}) \rangle_{p(\boldsymbol{\theta})} + \log V(r)$. It is important to remember that the definition of volume here is in terms of the coordinates $\boldsymbol{\theta}$, and that a change of coordinates (or a change of the definition of the distance function) would result in a different value of the differential entropy. This is equivalent to counting possibilities or volumes with a different choice of base measure $q(\boldsymbol{\theta})$.

Reducing r towards zero ought to lead to a more accurate result because the probability density at a point will be more accurately approximated by the probability of a small region containing the point, divided by the volume of that region. On the other hand, smaller values of r will lead to more Nested Sampling iterations required, and hence less accuracy in the depth estimates. Thus, there is a an awkward trade-off involved in choosing the value of r, which will need to be investigated more thoroughly in the future. However, the algorithm as it currently stands ought to be useful for calculating the entropies of any low-to-moderate dimensional functions of an underlying high-dimensional space.

If the distance function $d(\boldsymbol{\theta}_{\text{ref}}, \boldsymbol{\theta})$ is chosen to be Euclidean, the constraint $d(\boldsymbol{\theta}_{\text{ref}}, \boldsymbol{\theta}) < r$ corresponds to a ball in the space of possible $\boldsymbol{\theta}_{\text{ref}}$ values. The volume of such a ball, of radius r in n dimensions, is

$$V = \frac{(\pi r^2)^{n/2}}{\left(\frac{n}{2}\right)!}. \tag{30}$$

In one dimension, if the threshold r is not small, we might think we are calculating the entropy associated with the question 'what is θ, to within a tolerance of $\pm r$?'. This is not quite correct. See Appendix A for a discussion of this type of question.

In the following sections, I demonstrate the algorithm on three example problems. For each of the examples, I used 1000 repetitions of NS (i.e., 1000 reference particles), and used 10 NS particles and 10,000 MCMC steps per NS iteration.

5. Example 1: Entropy of a Prior for the Data

This example demonstrates how the algorithm may be used to determine the entropy for a quantity whose distribution can only be sampled, in this case, a prior distribution over datasets.

Consider the basic statistics problem of inferring a quantity μ from 100 observations $x = \{x_1, ..., x_{100}\}$ whose probability distribution (conditional on μ) is

$$p(x_i|\mu) \sim \text{Normal}(\mu, 1). \tag{31}$$

I assumed a $\text{Normal}(0, 10^2)$ prior for μ. Before getting the data, the prior information is described by the joint distribution

$$p(\mu, x) = p(\mu)p(x|\mu). \tag{32}$$

The prior over datasets is given by the marginal distribution

$$p(x) = \int p(\mu, x) \, d\mu. \tag{33}$$

This is the distribution whose differential entropy I seek in this section. Its true value is available analytically in this case, by noting that the posterior is available in closed form for any dataset:

$$\mu|x \sim \text{Normal} \left(\frac{1}{100.01} \sum_{i=1}^{100} x_i, \left[\frac{10}{\sqrt{10001}} \right]^2 \right), \tag{34}$$

which enables the calculation of the mutual information and hence $H(x)$. The true value of $H(x)$ is 146.499 nats. However, in most situations, the entropy of a marginal distribution such as this is not available in closed form.

To use the algorithm, a distance function $d(;)$ must be defined, to quantify how far the NS particles are from the reference particle. To calculate the entropy of the data, I applied the algorithm in the joint space of possible parameters and datasets, i.e., distribution of reference particles followed Equation 32. For the distance function, which helps the NS particles approach the reference particle, I used a simple Euclidean metric in the space of datasets:

$$d\left(\theta; \theta_{\text{ref}}\right) = \sqrt{\sum_{i=1}^{100} (x_i - x_{\text{ref},i})^2}. \tag{35}$$

Since the distance function refers only to the data x and not the parameter μ, the sampling is effectively done in the space of possible datasets only — the parameter μ can be thought of as merely a latent variable allowing $p(x)$ to be explored conveniently.

I ran the algorithm and computed the average depth using a tolerance of $r = 10^{-3}\sqrt{100}$, so that the root mean square difference between points in the two datasets was about 10^{-3}. From the algorithm, the estimate of the expected depth was

$$\langle (\text{depth})_x \rangle = 698.25 \pm 0.34. \tag{36}$$

The uncertainty here is the standard error of the mean, i.e., the standard deviation of the depth estimates divided by square root of the number of NS repetitions (the number of reference particles whose depth was evaluated).

The log-volume of a 100-dimensional ball of radius $r = 10^{-3}\sqrt{100}$ is -551.76. Therefore, the differential entropy is estimated to be

$$H(x) \approx 146.49 \pm 0.34, \tag{37}$$

which is very close to the true value (perhaps suspiciously close, but this was just a fluke).

To carry out the sampling for this problem, Metropolis–Hastings moves that explore the joint prior distribution $p(\mu, x)$ had to be implemented. Four basic kinds of proposals were used: (i) those which change μ while leaving x fixed; (ii) those which change μ and shift x correspondingly; (iii) those which resample a subset of the xs from $p(x|\mu)$; and (iv) those which move a single x slightly.

6. Example 2: Measuring the Period of an Oscillating Signal

In physics and astronomy, it is common to measure an oscillating signal at a set of times $\{t_1, ..., t_n\}$, in order to infer the amplitude, period, and phase of the signal. Here, I demonstrate the algorithm on a toy version of Bayesian experimental design: at what times should the signal be observed in order to obtain as much information as possible about the period? To answer this, we need to be able to calculate the mutual information between the unknown period and the data.

As the algorithm lets us calculate the entropy of any distribution that can be sampled by MCMC, there are several options. The mutual information can be written in various ways, such as the following:

$$I(\theta; d) = H(\theta) + H(d) - H(\theta, d), \tag{38}$$

$$I(\theta; d) = H(d) - H(d|\theta), \tag{39}$$

$$I(\theta; d) = H(\theta) - H(\theta|d), \tag{40}$$

$$I(\theta; d) = \int p(d)\, D_{\mathrm{KL}}\left(p(\theta|d)\,||\,p(\theta)\right)\, dd, \tag{41}$$

$$I(\theta; d) = D_{\mathrm{KL}}\left(p(\theta, d)\,||\,p(\theta)p(d)\right). \tag{42}$$

In a Bayesian problem where θ is an unknown parameter and d is data, the first and second formulas would be costly to compute because the high dimensional probability distribution for the dataset would require a large number of NS iterations to compute $H(d)$ and $H(\theta, d)$. The fourth involves the Kullback–Leibler divergence from the prior to the posterior, averaged over all possible datasets. This is straightforward to approximate with standard Nested Sampling, but the estimate for a given dataset may be biased. It is quite plausible that the bias is negligible compared to the variance from averaging over many datasets. This is probably true in many situations.

However, more significantly, the above method would not work if we want $I(\theta; d)$ for a single parameter or a subset of them, rather than for all of the parameters. The prior-to-posterior KL divergence measured by standard NS is the amount learned about *all* parameters, including nuisance parameters. Therefore, I adopted the third formula as the best way to compute $I(\theta; d)$. In the sinusoidal example, the parameter of interest is the period, whereas the model will contain other parameters as well. Since the period will have its prior specified explicitly, $H(\theta)$ will be available analytically. I then use the algorithm to obtain $H(\theta|d)$ as follows:

$$H(\theta|d) = \int p(d_{\mathrm{ref}})H(\theta|d = d_{\mathrm{ref}})\, dd_{\mathrm{ref}}. \tag{43}$$

This is an expected value over datasets, of the entropy of the posterior given each dataset. I ran the algorithm by generating reference particles from $p(\theta, d)$, i.e., generating parameter values from the prior, along with corresponding simulated datasets. Given each simulated dataset, I used NS

to compute an unbiased estimate of $H(\boldsymbol{\theta}|\boldsymbol{d} = \boldsymbol{d}_{\text{ref}})$, by evolving a set of particles (initially from the posterior) towards the reference particle. The role of the "prior" in the NS is actually taken by the posterior, and the "likelihood" was minus a distance function defined in terms of the parameter(s) of interest.

To generate the initial NS particles from the posterior, I used standard MCMC (targeting the posterior), initialised at the $\boldsymbol{\theta}$-value that produced the simulated dataset $\boldsymbol{d}_{\text{ref}}$, which is a perfect sample from the posterior.

6.1. Assumptions

I consider two possible observing strategies, both involving $n = 101$ measurements. The 'even' strategy has observations at times

$$t_i = \frac{i-1}{n-1} \tag{44}$$

for $i \in \{1, 2, ..., 101\}$; that is, the observation times are $0, 0.01, 0.02, ..., 0.99, 1$—evenly spaced from $t = 0$ to $t = 1$, including the endpoints. The second, 'uneven' observing strategy schedules the observations according to

$$t_i = \left(\frac{i - \frac{1}{2}}{n}\right)^3, \tag{45}$$

which schedules observations close together initially, and further apart as time goes on.

A purely sinusoidal signal has the form

$$y(t) = A \sin\left(\frac{2\pi t}{T} + \phi\right), \tag{46}$$

where A is the amplitude, T is the period, and ϕ is the phase. Throughout this section, I parameterise the period by its logarithm, $\tau = \log_{10} T$. I assumed the following priors:

$$\ln A \sim \text{Normal}\left(0, 0.1^2\right), \tag{47}$$

$$\tau \sim \text{Uniform}(-1, 0), \tag{48}$$

$$\phi \sim \text{Uniform}(0, 2\pi), \tag{49}$$

and the following conditional prior for the data:

$$Y_i | A, T, \phi \sim \text{Normal}\left(y(t_i), 0.1^2\right). \tag{50}$$

That is, the data is just the signal $y(t)$ observed at particular times $\{t_1, ..., t_n\}$, with Gaussian noise of standard deviation 0.1. The amplitude of the sinusoid is very likely to be around 10 times the noise level, and the period is between 0.1 times and 1 time the duration of the data. An example signal observed with the even and uneven observing schedules is shown in Figure 2.

To carry out the sampling for this problem, Metropolis–Hastings moves that explore the posterior distribution $p(A, \tau, \phi|\boldsymbol{Y})$ had to be implemented. I used heavy-tailed proposals (as in Section 8.3 of Brewer and Foreman-Mackey [23]), which modified one parameter at a time.
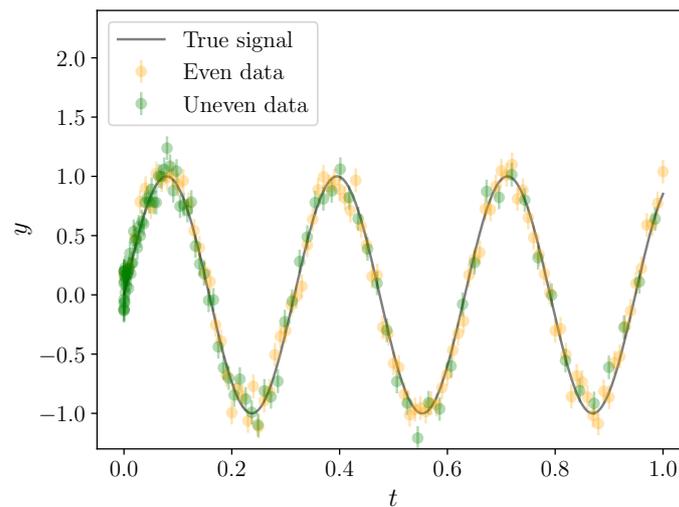
Figure 2. A signal with true parameters $A = 1$, $\tau = -0.5$, and $\phi = 0$, observed with noise standard deviation 0.1 with the even (gold points) and uneven (green points) observing strategies.

6.2. Results

Letting $\tau = \log_{10} T$, and treating A and ϕ are nuisance parameters, I calculated the conditional entropy $H(\tau|Y)$ using the algorithm with a tolerance of $r = 10^{-5}$. The distance function was just the absolute difference between τ and τ_{ref}.

The result was

$$\langle(\text{depth})\rangle = 5.379 \pm 0.038. \tag{51}$$

Converting to a differential entropy by adding $\ln(2r) = -10.820$, the conditional entropy is

$$H(\tau|Y) = -5.441 \pm 0.038 \text{ nats}. \tag{52}$$

Since the Uniform(0,1) prior for τ has a differential entropy of zero, the mutual information is $I(\tau;Y) = H(\tau) - H(\tau|Y) = 5.441 \pm 0.038$ nats.

The results for the uneven observing schedule were

$$\langle(\text{depth})\rangle = 5.422 \pm 0.038, \tag{53}$$
$$H(\tau|Y) = -5.398 \pm 0.038, \tag{54}$$
$$I(\tau;Y) = 5.398 \pm 0.038. \text{ nats}. \tag{55}$$

The difference in mutual information between the two observation schedules is trivial. The situation may be different if we had allowed for shorter periods to be possible, as irregular observing schedules are known to reduce aliasing and ambiguity in the inferred period [24]. When inferring the period of an oscillating signal, multimodal posterior distributions for the period are common [17,25]. The multimodality of the posterior here raises an interesting issue. Is the question we really want answered "what is the value of T **precisely**?", to which the mutual information relates? Most practicing scientists would not feel particularly informed to learn that the vast majority of possibilities had been ruled out, if the posterior still consisted of several widely separated modes! Perhaps, in some applications, a more appropriate question is "what is the value of T to within $\pm 10\%$", or something along these lines. See Appendix A for more about this issue.

In a serious experimental design situation, the relevance is not the only consideration (or the answer to every experimental design problem would be to obtain more data without limit), but it is a very important one.

7. Example 3: Data with Pareto Distribution

Consider a situation where the probability distribution for some data values $x = \{x_1, x_2, ..., x_n\}$ is a Pareto distribution, conditional on a minimum value $x_{\min} = 1$ and a slope α:

$$p(x|\alpha) = \prod_{i=1}^{n} \frac{\alpha}{x_i^{\alpha+1}}, \tag{56}$$

where all of the x_i are greater than $x_{\min} = 1$. If α is relatively low (close to zero), then this distribution is very heavy-tailed, and as α increases, the probability density for the xs concentrates just above $x = 1$.

With heavy-tailed distributions, it can be difficult to predict future observations from past ones, a notion popularised by Taleb [26]. This occurs for three main reasons. First, the observations may not provide much information about the parameters (in this case, α). Second, even conditional on the parameters, the predictive distribution for future data can be heavy-tailed, implying a lot of uncertainty. Finally, one may always doubt the probability assignment on which such a calculation is based. However, the data clearly provides *some* nonzero amount of information about the future.

Consider $n = 100$ data values, partitioned into two halves, so $y = \{x_1, ..., x_{50}\} = \{y_1, ..., y_{50}\}$ is the first half of the data and $z = \{x_{51}, ..., x_{100}\} = \{z_1, ..., z_{50}\}$ is the second half. Let y_{tot} and z_{tot} be the totals of the two subsets of the data:

$$y_{\text{tot}} = \sum_{i=1}^{50} y_i, \tag{57}$$

$$z_{\text{tot}} = \sum_{i=1}^{50} z_i. \tag{58}$$

In this section, I use the algorithm to answer the questions: (i) How uncertain is y_{tot}?; (ii) How uncertain is z_{tot}?; and (iii) How informative is y_{tot} about z_{tot}? These are quantified by $H(y_{\text{tot}})$, $H(z_{\text{tot}})$, and $I(y_{\text{tot}}; z_{\text{tot}})$, respectively. This can be done despite the fact that α is unknown.

The prior for α was chosen to be lognormal such that the prior median of α is 1 and the prior standard deviation of $\ln \alpha$ is also 1. Due to the heavy-tailed nature of the Pareto distribution, I ran the algorithm in terms of $\ln y_{\text{tot}}$ and $\ln z_{\text{tot}}$. This affects the differential entropies but not the mutual information (which is invariant under changes of parameterisation). The joint distribution for $\ln y_{\text{tot}}$ and $\ln z_{\text{tot}}$, whose properties I aimed to obtain, is shown in Figure 3.

Three types of Metropolis–Hastings proposals were used for this example: (i) a proposal that changes α and modifies all of the xs accordingly; (ii) a proposal that resamples a subset of the xs from the Pareto distribution; and (iii) a proposal that moves a single x-value slightly. A fourth proposal that moves α while keeping x fixed, i.e., a posterior-sampling move, was not implemented.

To compute the entropy of $\ln y_{\text{tot}}$, the distance function was just the absolute value of the distance between the NS particles and the reference particle in terms of $\ln y_{\text{tot}}$. The result, using a tolerance of $r = 10^{-3}$, was:

$$\left\langle (\text{depth})_{\ln y_{\text{tot}}} \right\rangle = 8.851 \pm 0.057. \tag{59}$$

Correcting for the volume factor yields a differential entropy estimate of $H(\ln y_{\text{tot}}) = 2.366 \pm 0.057$ nats. By the symmetry of the problem specification, $H(\ln z_{\text{tot}}) = H(\ln y_{\text{tot}})$.
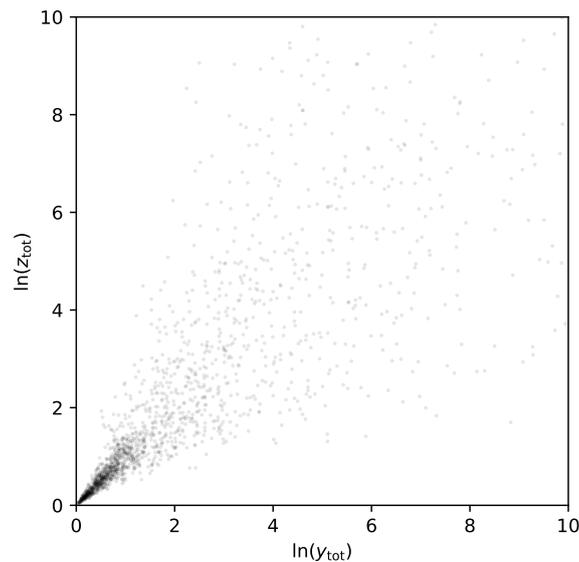
Figure 3. The joint distribution for the log-sum of the first half of the 'Pareto data' and the second half. The goal is to calculate the entropy of the marginal distributions, the entropy of the joint distribution, and hence the mutual information. The distribution is quite heavy tailed despite the logarithms, and extends beyond the domain shown in this plot.

The joint entropy $H(\ln y_{\text{tot}}, \ln z_{\text{tot}})$ was also estimated by using a Euclidean distance function and a tolerance of 10^{-3}. The result was

$$\left\langle (\text{depth})_{(\ln y_{\text{tot}}, \ln z_{\text{tot}})} \right\rangle = 16.03 \pm 0.11. \tag{60}$$

Applying the log-volume correction yields a differential entropy estimate of $H(\ln y_{\text{tot}}, \ln z_{\text{tot}}) = 3.40 \pm 0.11$ nats. Therefore, the mutual information is

$$I(y_{\text{tot}}; z_{\text{tot}}) = H(\ln y_{\text{tot}}) + H(\ln z_{\text{tot}}) - H(\ln y_{\text{tot}}, \ln z_{\text{tot}}) \tag{61}$$

$$= 1.130 \pm 0.082 \text{ nats.} \tag{62}$$

The variance of this mutual information estimate was reduced by using a common sequence of reference points for the marginal and joint entropy runs. This is an instance of the 'common random numbers' technique (http://demonstrations.wolfram.com/ TheMethodOfCommonRandomNumbersAnExample/). In short, for those reference particles with a large depth in terms of the $\ln y_{\text{tot}}$ distance function, the depth in terms of the $(\ln y_{\text{tot}}, \ln z_{\text{tot}})$ distance function was also large. Since the mutual information involves a difference of the averages of these depths, it is more accurate to compute the average difference (i.e., compute the two depths using the same reference particle, to compute the difference between the two depths) than the difference of the averages.

8. Computational Cost and Limitations

The computational resources needed to compute these quantities was quite large, as Nested Sampling appears in the inner loop of the algorithm. However, it is worth reflecting on the complexity of the calculation that has been done.

In a Bayesian inference problem with parameters $\boldsymbol{\theta}$ and data $\boldsymbol{d}$, the mutual information is

$$I(\boldsymbol{\theta}; \boldsymbol{d}) = \iint p(\boldsymbol{\theta}, \boldsymbol{d}) \ln \left[\frac{p(\boldsymbol{\theta}, \boldsymbol{d})}{p(\boldsymbol{\theta}) p(\boldsymbol{d})} \right] d\boldsymbol{\theta} \, d\boldsymbol{d}, \tag{63}$$

and this measures the dependence between $\boldsymbol{\theta}$ and $\boldsymbol{d}$. A Monte Carlo strategy to evaluate this is to sample from $p(\boldsymbol{\theta}, \boldsymbol{d})$ and average the value of the logarithm, which involves computing the log-evidence

$$\ln p(\boldsymbol{d}) = \ln \left[\int p(\boldsymbol{\theta}) p(\boldsymbol{d}|\boldsymbol{\theta}) \, d\boldsymbol{\theta} \right] \tag{64}$$

for each possible data set. Computing $p(\boldsymbol{d})$ for even a single data set has historically been considered difficult.

However, if we want the mutual information between the data and *a subset of the parameters* (i.e., there are nuisance parameters we do not care about), things become even more tricky. Suppose we partition the parameters $\boldsymbol{\theta}$ into important parameters $\boldsymbol{\phi}$ and nuisance parameters $\boldsymbol{\eta}$, such that we want to calculate the mutual information $I(\boldsymbol{\phi}; \boldsymbol{d})$. This is given by

$$I(\boldsymbol{\phi}; \boldsymbol{d}) = \iint p(\boldsymbol{\phi}, \boldsymbol{d}) \ln \left[\frac{p(\boldsymbol{\phi}, \boldsymbol{d})}{p(\boldsymbol{\phi}) p(\boldsymbol{d})} \right] \, d\boldsymbol{\phi} \, d\boldsymbol{d}, \tag{65}$$

which is equivalent to Equation 63 but with $\boldsymbol{\theta}$ replaced throughout by $\boldsymbol{\phi}$. However, this is an even more difficult calculation than before, as $p(\boldsymbol{d}|\boldsymbol{\phi})$, the marginal likelihood function with the nuisance parameters integrated out, is typically unavailable in closed form. If we were to marginalise out the nuisance parameters $\boldsymbol{\eta}$ explicitly, this would give us Equation 65 with every probability distribution written as an explicit integral over $\boldsymbol{\eta}$:

$$I(\boldsymbol{\phi}; \boldsymbol{d}) = \iint \left(\int p(\boldsymbol{\phi}, \boldsymbol{\eta}) p(\boldsymbol{d}|\boldsymbol{\phi}, \boldsymbol{\eta}) \, d\boldsymbol{\eta} \right) \ln \left[\frac{\int p(\boldsymbol{\phi}, \boldsymbol{\eta}) p(\boldsymbol{d}|\boldsymbol{\phi}, \boldsymbol{\eta}) \, d\boldsymbol{\eta}}{p(\boldsymbol{\phi}) \int p(\boldsymbol{\phi}, \boldsymbol{\eta}) p(\boldsymbol{d}|\boldsymbol{\phi}, \boldsymbol{\eta}) \, d\boldsymbol{\phi} \, d\boldsymbol{\eta}} \right] \, d\boldsymbol{\phi} \, d\boldsymbol{d}. \tag{66}$$

It should not be surprising that this is costly. Indeed, it seems possible to me that any *general* algorithm for computing entropies will have the property that NS (or a similar algorithm capable of measuring small volumes) appears in an inner loop, since entropies are sums of terms, each of which relates to the downset of a given statement — the downset of a statement *S* being the set of all statements that imply *S* [3].

The current implementation of the algorithm (see Appendix B) is in C++ and has not been parallelised. Given the high computational costs, parallelisation should be attempted. This could follow the method of [21], or alternatively, different threads could work on different reference particles simultaneously.

The algorithm as presented will tend to become less accurate on high dimensional problems, as the required depth of the runs will increase correspondingly, since many NS iterations will be required to get a high dimensional vector of quantities to be close to the reference vector. The accuracy of the $\ln(\text{depth})$ estimates should degrade in proportion to $\sqrt{(\text{depth})}$, due to the Poisson-process nature of the $-\ln X$ sequence.

Throughout this paper, the question of how to sample from the constrained distributions required by NS has not been discussed in detail. For the examples, I used the original Skilling [11] suggestion of using a short MCMC run initialised at a surviving point. In some problems, this will not be effective, and this will cause the resulting estimates to be incorrect, as in standard Nested Sampling applications. Testing the sensitivity of the results to changes in the number of MCMC steps used is a simple way to increase one's confidence in the results, but is not a guarantee (no general guarantee can exist, because any algorithm based on sampling a subset of a space might miss a crucial but hard-to-locate dominant spike). Despite this weakness, I hope the algorithm will be useful in a wide range of applications.

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

Appendix A. Precisional Questions

The central issue about the value of a parameter θ asks "what is the value of θ, precisely?". However, in practice, we often don't need or want to know θ to arbitrary precision. Using the central issue can lead to counterintuitive results if you don't keep its specific definition in mind. For example, suppose x could take any integer value from 1 to 1 billion. If we learned the last digit of x, we will have ruled out nine tenths of the possibilities and therefore obtained a lot of information about the central issue. However, this information might be useless *for practical purposes*. If we learned the final digit was a 9, x could still be 9, or 19, or 29, any number ending in 9 up to 999,999,999.

In practice, we may want to ask a question that is different from the central issue. For example, suppose $x \in \{1, ..., 10\}$ and we want to know the value of x to within a tolerance of ± 1. Any of the following statements would resolve the issue:

- $x \in \{1, 2, 3\}$ and anything that implies it,
- $x \in \{2, 3, 4\}$ and anything that implies it,
- $x \in \{3, 4, 5\}$ and anything that implies it,
- and so on.

According to Knuth [3], the entropy of a question is computed by applying the sum rule over all statements that would answer the question.

We can write the question of interest, Q, as a union of ideal questions (the ideal questions are downsets of statements: the set of all statements that imply S is the downset of S and is denoted $\downarrow S$):

$$Q = [\downarrow (x \in \{1, 2, 3\})] \cup [\downarrow (x \in \{2, 3, 4\})] \cup [\downarrow (x \in \{3, 4, 5\})] \cup ... \tag{A1}$$

The entropy of this question is

$$H(Q) = h_{1,2,3} \tag{A2}$$
$$+ (h_{2,3,4} - h_{2,3}) \tag{A3}$$
$$+ (h_{3,4,5} - h_{3,4}) \tag{A4}$$
$$+ ... \tag{A5}$$
$$+ (h_{8,9,10} - h_{8,9}), \tag{A6}$$

where $h_x = -P(x) \log P(x)$.

Continuous Case

Consider a probability density function $f(x)$, defined on the real line. The probability contained in an interval $[x_0, x_0 + r]$, which has length r, is

$$\int_{x_0}^{x_0+r} f(x) \, dx = F(x_0 + r) - F(x_0), \tag{A7}$$

where $F(x)$ is the cumulative distribution function (CDF). This probability can be considered as a function of two variables, x_0 and r, which I will denote by $P(,)$:

$$P(x_0, r) = F(x_0 + r) - F(x_0). \tag{A8}$$

The contribution of such an interval to an entropy expression is $-P \log P$, i.e.,

$$Q(x_0, r) = -P(x_0, r) \log P(x_0, r). \tag{A9}$$

Consider the rate of change of Q as the interval is shifted to the right, but with its width r held constant:

$$\frac{\partial Q}{\partial x_0} = \frac{\partial}{\partial x_0} [-P \log P] \tag{A10}$$

$$= -[1 + \log P(x_0, r)] \frac{\partial P}{\partial x_0} \tag{A11}$$

$$= -[1 + \log P(x_0, r)] [f(x_0 + r) - f(x_0)]. \tag{A12}$$

Consider also the rate of change of Q as the interval is expanded to the right, while keeping its left edge fixed:

$$\frac{\partial Q}{\partial r} = \frac{\partial}{\partial r} [-P(x_0, r) \log P(x_0, r)] \tag{A13}$$

$$= -[1 + \log P(x_0, r)] \frac{\partial P}{\partial r} \tag{A14}$$

$$= -[1 + \log P(x_0, r)] f(x_0 + r). \tag{A15}$$

The entropy of the precisional question is built up from Q terms. The extra entropy from adding the interval $[x_0, x_0 + r]$ and removing the overlap $[x_0, x_0 + r - h]$, for small h, is

$$\delta H = Q(x_0, r) - Q(x_0, r - h) \tag{A16}$$

$$= h \frac{\partial Q}{\partial r} \tag{A17}$$

$$= -h [1 + \log P(x_0, r)] f(x_0 + r). \tag{A18}$$

Therefore, the overall entropy is

$$H = -\int_{-\infty}^{\infty} [1 + \log P(x - r, r)] f(x) \, dx \tag{A19}$$

$$= -\int_{-\infty}^{\infty} (1 + \log [F(x) - F(x - r)]) f(x) \, dx. \tag{A20}$$

Interestingly, calculating the log-probability of an interval to the right of x gives an equivalent result:

$$H' = -\int_{-\infty}^{\infty} (1 + \log [F(x + r) - F(x)]) f(x) \, dx \tag{A21}$$

$$= H. \tag{A22}$$

This is not quite the same as the expected log-probability calculated by the version of the algorithm proposed in this paper (when the tolerance is not small). However, the algorithm can be made to estimate the entropy of the precisional question, by redefining the distance function. For a one-dimensional quantity of interest, the distance function can be defined such that $d(x; x_{\text{ref}})$ can only ever be below r if $x < x_{\text{ref}}$ (or alternatively, if $x > x_{\text{ref}}$).

Appendix B. Software

A C++ implementation of the algorithm is available in a `git` repository located at

https://github.com/eggplantbren/InfoNest

and can be obtained using the following `git` command, executed in a terminal:

```
git clone https://github.com/eggplantbren/InfoNest
```

The following will compile the code and execute the first example from the paper:

```
cd InfoNest/cpp
make
./main
```

The algorithm will run for 1000 'reps', i.e., 1000 samples of θ_{ref}, which is time consuming. Output is saved to `output.txt`. At any time, you can execute the Python script `postprocess.py` to get an estimate of the depth:

```
python postprocess.py
```

By default, `postprocess.py` estimates the depth with a tolerance of $r = 10^{-3}$. This value can be changed by calling the `postprocess` function with a different value of its argument `tol`. E.g., `postprocess.py` can be edited so that its last line is `postprocess(tol=0.01)` instead of `postprocess()`.

The numerical parameters, and the choice of which problem is being solved, are specified in `main.cpp`. The default problem is the first example from the paper. For Example 2, since it is a conditional entropy that is being estimated (which requires a slight modification to the algorithm), an additional argument `InfoNest::Mode::conditional_entropy` must be passed to the `InfoNest::execute` function.

References

1. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423.
2. Cover, T.M.; Thomas, J.A. *Elements of Information Theory*; Wiley & Sons, Hoboken, NJ, USA, 2012.
3. Knuth, K.H. Toward question-asking machines: The logic of questions and the inquiry calculus. In Proceedings of the 10th International Workshop On Artificial Intelligence and Statistics, Barbados, 6–8 January 2005.
4. Knuth, K.H.; Skilling, J. Foundations of inference. *Axioms* **2012**, *1*, 38–73.
5. Caticha, A.; Giffin, A. Updating probabilities. *AIP Conf. Proc.* **2006**, *31*, doi:10.1063/1.2423258.
6. Szabó, Z. Information theoretical estimators toolbox. *J. Mach. Learn. Res.* **2014**, *15*, 283–287.
7. Jaynes, E.T. *Probability theory: The logic of science*; Cambridge university press: New York, NY, USA, 2003.
8. MacKay, D.J. *Information Theory, Inference and Learning Algorithms*; Cambridge university press: New York, NY, USA, 2003.
9. Caticha, A. Lectures on probability, entropy, and statistical physics. *arXiv* **2008**, arXiv:0808.0012.
10. Bernardo, J.M. Reference analysis. *Handb. Stat.* **2005**, *25*, 17–90.
11. Skilling, J. Nested sampling for general Bayesian computation. *Bayesian anal.* **2006**, *1*, 833–859.
12. Feroz, F.; Hobson, M.; Bridges, M. MultiNest: An efficient and robust Bayesian inference tool for cosmology and particle physics. *Mon. Not. R. Astron. Soc.* **2009**, *398*, 1601–1614.
13. Brewer, B.J.; Pártay, L.B.; Csányi, G. Diffusive nested sampling. *Stat. Comput.* **2011**, *21*, 649–656.
14. Handley, W.; Hobson, M.; Lasenby, A. PolyChord: Next-generation nested sampling. *Mon. Not. R. Astron. Soc.* **2015**, *453*, 4384–4398.
15. Knuth, K.H.; Habeck, M.; Malakar, N.K.; Mubeen, A.M.; Placek, B. Bayesian evidence and model selection. *Digit. Signal Process.* **2015**, *47*, 50–67.
16. Pullen, N.; Morris, R.J. Bayesian model comparison and parameter inference in systems biology using nested sampling. *PLoS ONE* **2014**, *9*, e88419.
17. Brewer, B.J.; Donovan, C.P. Fast Bayesian inference for exoplanet discovery in radial velocity data. *Mon. Not. R. Astron. Soc.* **2015**, *448*, 3206–3214.
18. Pártay, L.B.; Bartók, A.P.; Csányi, G. Efficient sampling of atomic configurational spaces. *J. Phys. Chem. B* **2010**, *114*, 10502–10512.
19. Baldock, R.J.; Pártay, L.B.; Bartók, A.P.; Payne, M.C.; Csányi, G. Determining pressure-temperature phase diagrams of materials. *Phys. Rev. B* **2016**, *93*, 174108.

20. Martiniani, S.; Stevenson, J.D.; Wales, D.J.; Frenkel, D. Superposition enhanced nested sampling. *Phys. Rev. X* **2014**, *4*, 031034.

21. Henderson, R.W.; Goggans, P.M.; Cao, L. Combined-chain nested sampling for efficient Bayesian model comparison. *Digit. Signal Process.* **2017**, *70*, 84–93.

22. Walter, C. Point process-based Monte Carlo estimation. *Stat. Comput.* **2017**, *27*, 219–236.

23. Brewer, B.J.; Foreman-Mackey, D. DNest4: Diffusive Nested Sampling in C++ and Python. *J. Stat. Softw.* **2016**, in press.

24. Bretthorst, G.L. Nonuniform sampling: Bandwidth and aliasing. *AIP Conf. Proc.* **2001**, *567*, doi:10.1063/1.1381847.

25. Gregory, P.C. A Bayesian Analysis of Extrasolar Planet Data for HD 73526. *Astrophys. J.* **2005**, *631*, 1198–1214.

26. Taleb, N.N. *The Black Swan: The Impact of the Highly Improbable.* Random House: New York, NY, USA, 2007.