mathematics

Article

Effects of Exploration Weight and Overtuned Kernel Parameters on Gaussian Process-Based Bayesian Optimization Search Performance

Yuto Omae

College of Industrial Technology, Nihon University, 1-2-1, Izumi, Narashino, Chiba 275-8575, Japan; oomae.yuuto@nihon-u.ac.jp

Abstract: Gaussian process-based Bayesian optimization (GPBO) is used to search parameters in machine learning, material design, etc. It is a method for finding optimal solutions in a search space through the following four procedures. (1) Develop a Gaussian process regression (GPR) model using observed data. (2) The GPR model is used to obtain the estimated mean and estimated variance for the search space. (3) The point where the sum of the estimated mean and the weighted estimated variance (upper confidence bound, UCB) is largest is the next search point (in the case of a maximum search). (4) Repeat the above procedures. Thus, the generalization performance of the GPR is directly related to the search performance of the GPBO. In procedure (1), the kernel parameters (KPs) of the GPR are tuned via gradient descent (GD) using the log-likelihood as the objective function. However, if the number of iterations of the GD is too high, there is a risk that the KPs will overfit the observed data. In this case, because the estimated mean and variance output by the GPR model are inappropriate, the next search point cannot be properly determined. Therefore, overtuned KPs degrade the GPBO search performance. However, this negative effect can be mitigated by changing the parameters of the GPBO. We focus on the weight of the estimated variances (exploration weight) of the UCB as one of these parameters. In a GPBO with a large exploration weight, the observed data appear in various regions in the search space. If the KP is tuned using such data, the GPR model can estimate the diverse regions somewhat correctly, even if the KP overfits the observed data, i.e., the negative effect of overtuned KPs on the GPR is mitigated by setting a larger exploration weight for the UCB. This suggests that the negative effect of overtuned KPs on the GPBO search performance may be related to the UCB exploration weight. In the present study, this hypothesis was tested using simple numerical simulations. Specifically, GPBO was applied to a simple black-box function with two optimal solutions. As parameters of GPBO, we set the number of KP iterations of GD in the range of 0–500 and the exploration weight as $\{1, 5\}$. The number of KP iterations expresses the degree of overtuning, and the exploration weight expresses the strength of the GPBO search. The results indicate that, in the overtuned KP situation, GPBO with a larger exploration weight has better search performance. This suggests that, when searching for solutions with a small GPBO exploration weight, one must be careful about overtuning KPs. The findings of this study are useful for successful exploration with GPBO in all situations where it is used, e.g., machine learning hyperparameter tuning.

Keywords: machine learning; Bayesian optimization; Gaussian process; overfitting

MSC: 68T01; 62J02

check for updates

Citation: Omae, Y. Effects of Exploration Weight and Overtuned Kernel Parameters on Gaussian Process-Based Bayesian Optimization Search Performance. *Mathematics* **2023**, *11*, 3067. https://doi.org/10.3390/math11143067

Academic Editors: Florin Leon, Mircea Hulea and Marius Gavrilescu

Received: 7 June 2023
Revised: 4 July 2023
Accepted: 7 July 2023
Published: 11 July 2023

1. Introduction

Gaussian process-based Bayesian optimization (GPBO) optimizes black-box functions and is adopted to save time and/or reduce costs. For example, it is used for concrete design [1,2], material design [3–5], and tuning hyperparameters in machine learning (for support vector machines [6–8], random forest models [9,10] and neural networks [9,11,12], etc.). Appropriate parameters can be obtained more rapidly using GPBO compared with simple methods such as grid search. For example, Wu et al. [9] reported that the computation time of hyperparameter tuning in machine learning can be reduced significantly using GPBO. Snoek et al. [11] reported that when GPBO was used for tuning, the developed convolutional neural networks had higher generalization scores than networks with parameters tuned by an expert of machine learning.

To perform GPBO, a Gaussian process regression (GPR) model must be developed, which is computationally expensive. The computational cost of the kernel inverse matrix required in GPBO is $\mathcal{O}(n^3)$ for a data size of n [13]. Due to the fact that n increases with each successive GPBO iteration, the computational cost increases. Therefore, sparse matrix methods [13–16] and mini-batch methods [17] have been proposed for reducing the computational cost.

In the log-likelihood-based objective function, the kernel inverse matrix is used to tune the hyperparameters of the GPR model [18]. Gradient-based methods [17,19,20], evolutionary algorithms [21,22], and Markov chain Monte Carlo methods [23] have been adopted for hyperparameter tuning of the GPR model. Cross-validation [18] is adopted for tuning the kernel parameters (KPs) to avoid overfitting to the observed data. However, because the size of the observed data gradually increases, performing cross-validation at the beginning of GPBO is difficult. Therefore, we cannot perform cross-validation at the beginning of GPBO to tune the hyperparameters of the GPR model. Moreover, cross-validation increases the computational cost [18].

In the early stages of GPBO, the sample size is insufficient. Additionally, it is difficult to properly tune a GPR model by a small sample size [24,25]. Therefore, consider the situation wherein the KPs are tuned using all samples instead of using a method that reduces the number of data, such as cross-validation. In this case, because there are no data for validation, it is not known to what extent the KPs should be fitted to the observed data. Thus, there is a risk of overtuning the GPR model. Overtuned GPR models can correctly estimate observed regions but cannot properly estimate unobserved regions [26]. GPBO is an algorithm that searches for the optimal solution in unobserved regions. Therefore, if unobserved areas cannot be correctly estimated, a proper search cannot be performed. From this viewpoint, the search performance of Bayesian optimization using an overfitted GPR model is expected to be poor.

The negative effect of overtuned KPs on the search performance depends on other parameters of GPBO. We focus on the exploration weight of the upper confidence bound (UCB) [27,28], which is a GPBO parameter. Generally, when the exploration weight is set to a large value, the next search point tends to be selected from regions with insufficient observations; therefore, the observed samples appear in various areas. In this case, even if the KPs are overfitted to the observed samples, the GPR model can correctly estimate the various input domains. In contrast, with a small exploration weight, because the observed samples appear in only limited regions, the overfitted GPR model only estimates limited regions; therefore, its generalization score is worse.

This implies that the risk of overtuned KPs degrading the search performance depends on the exploration weight. We verified this hypothesis by analyzing the relationships among the overtuned KPs, exploration weight, and GPBO search performance. The results indicated that, for GPBO, more attention must be paid to the overtuning of the KPs in the case of smaller exploration weights. Additionally, it is necessary to pay attention to avoid overtuning KPs when searching for the solution via GPBO with a small exploration weight. These findings are useful for successful exploration with GPBO in all situations where it is used, e.g., hyperparameter tuning in machine learning.

As indicated by previous studies [21,24,25], the likelihood function in GPR may have multiple minimal solutions, making it difficult to find a globally optimal solution. Therefore, we focused on gradient descent (GD), which can rapidly obtain a local minimal solution, as a method for tuning the KPs. This method is widely used for tuning the KPs of GPR models [17,19,20].

2. Gaussian Process-Based Bayesian Optimization

2.1. Surrogate Model

In this study, we use the GPR model as a surrogate model for Bayesian optimization, which we call GPBO. This method outputs the estimated average and variance by assuming a Gaussian distribution from the dataset $\mathcal{D}$ consisting of pairs of observed input and output values.

Here, the output value y is obtained as

$$y = f(\boldsymbol{x}) + \epsilon, \tag{1}$$

where $\boldsymbol{x} = [x_1 \ \cdots \ x_D]^\top$ is the D-dimensional input vector, $f(\boldsymbol{x})$ is a black-box function, and ϵ denotes observation noise. In addition, we consider a situation in which a dataset

$$\mathcal{D} = \{(\boldsymbol{x}_n, y_n) | n = 1, \cdots, N\} \tag{2}$$

is collected via N observations. In the case of GPR, assuming that the average of y is zero, the average and variance values of the output y' for the new input data $\boldsymbol{x}'$ are given as follows:

$$\mathbb{E}[y'|\boldsymbol{x}', \mathcal{D}, \boldsymbol{\theta}] = \boldsymbol{k}'(\boldsymbol{\theta})^\top \boldsymbol{K}(\boldsymbol{\theta})^{-1} \boldsymbol{y}, \tag{3}$$

$$\mathbb{V}[y'|\boldsymbol{x}', \mathcal{D}, \boldsymbol{\theta}] = k''(\boldsymbol{\theta}) - \boldsymbol{k}'(\boldsymbol{\theta})^\top \boldsymbol{K}(\boldsymbol{\theta})^{-1} \boldsymbol{k}'(\boldsymbol{\theta}) \tag{4}$$

where $\boldsymbol{y} = [y_1 \cdots y_N]^\top$. $\boldsymbol{K}(\boldsymbol{\theta})$ is a kernel matrix defined as follows:

$$\boldsymbol{K}(\boldsymbol{\theta}) = \left[k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\theta})\right] \in \mathbb{R}^{N \times N}_{>0} \tag{5}$$

where $k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\theta})$ denotes the kernel function. For Gaussian kernels, the kernel function is defined as

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\theta}) = \theta_1 \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2}{\theta_2}\right) + \theta_3 \delta(i,j), \ \ \delta(i,j) = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases}, \ \ \boldsymbol{\theta} \in \mathbb{R}^3_{>0} \tag{6}$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ is a vector comprising three KPs. Due to the fact that the estimation accuracy depends on the KPs, appropriate tuning is important. $k''(\boldsymbol{\theta})$ and $\boldsymbol{k}'(\boldsymbol{\theta})$ are the kernels related to the observed samples $\boldsymbol{x}_i$ and new sample $\boldsymbol{x}'$, respectively, and are defined as follows:

$$k''(\boldsymbol{\theta}) = k(\boldsymbol{x}', \boldsymbol{x}'; \boldsymbol{\theta}), \ \ \boldsymbol{k}'(\boldsymbol{\theta}) = \left[k(\boldsymbol{x}_i, \boldsymbol{x}'; \boldsymbol{\theta})\right] \in \mathbb{R}^N_{>0}. \tag{7}$$

2.2. Tuning Kernel Parameters

As stated in the Introduction, the KP vector $\boldsymbol{\theta}$ is tuned using GD. Due to the fact that the $\boldsymbol{\theta}$ that maximizes the generation probability of the observed output $\boldsymbol{y}$ is desirable, using the log-likelihood, the objective function L is defined as

$$\begin{aligned} L(\boldsymbol{\theta}) &= \log p(\boldsymbol{y}|\boldsymbol{\theta}) \\ &= \log \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}(\boldsymbol{\theta})) \\ &= \log\left(\frac{1}{\sqrt{(2\pi)^N |\boldsymbol{K}(\boldsymbol{\theta})|}} \exp\left(-\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}(\boldsymbol{\theta})^{-1} \boldsymbol{y}\right)\right) \\ &\propto -\log|\boldsymbol{K}(\boldsymbol{\theta})| - \boldsymbol{y}^\top \boldsymbol{K}(\boldsymbol{\theta})^{-1} \boldsymbol{y}. \end{aligned} \tag{8}$$

We assume that $p(\boldsymbol{y}|\boldsymbol{\theta})$ is a Gaussian distribution consisting of the average $\boldsymbol{0}$ and the covariance matrix $\boldsymbol{K}(\boldsymbol{\theta})$. When we adopt the gradient method for the objective function, $\boldsymbol{\theta}$

may be negative. Due to the fact that the Gaussian KP $\boldsymbol{\theta}$ requires a plus (see Equation (6)), $\boldsymbol{\theta}$ is redefined as follows:

$$\begin{aligned}\boldsymbol{\theta} &= g(\boldsymbol{\theta}') \\ &= \exp(\boldsymbol{\theta}'), \ \ \boldsymbol{\theta}' = [\theta'_1 \ \theta'_2 \ \theta'_3]^\top\end{aligned} \tag{9}$$

where the map g is

$$g : \mathbb{R}^3 \mapsto \mathbb{R}^3_{>0}. \tag{10}$$

Therefore, when the gradient method is run as the search target $\boldsymbol{\theta}'$ and the obtained parameter is transformed into $\boldsymbol{\theta}$ using Equation (9), $\boldsymbol{\theta}$ will certainly be positive. Therefore, we use the update equation for the KPs, as follows:

$$\boldsymbol{\theta}'^{(t_\mathrm{g}+1)} = \boldsymbol{\theta}'^{(t_\mathrm{g})} + \gamma \nabla L\Big(\exp(\boldsymbol{\theta}'^{(t_\mathrm{g})})\Big) \tag{11}$$

where t_g represents the iteration count, $\boldsymbol{\theta}'^{(t_\mathrm{g})}$ is the t_g-th specific value of $\boldsymbol{\theta}'$, and γ represents the learning rate. This partial differentiation is as follows:

$$\begin{aligned}\nabla L\big(\exp(\boldsymbol{\theta}')\big) = &- \mathrm{tr}\Big(\boldsymbol{K}(\exp(\boldsymbol{\theta}'))^{-1} \nabla \boldsymbol{K}(\exp(\boldsymbol{\theta}'))\Big) \\ &+ \Big(\boldsymbol{K}(\exp(\boldsymbol{\theta}'))^{-1}\boldsymbol{y}\Big)^\top \nabla \boldsymbol{K}(\exp(\boldsymbol{\theta}'))\Big(\boldsymbol{K}(\exp(\boldsymbol{\theta}'))^{-1}\boldsymbol{y}\Big).\end{aligned} \tag{12}$$

Due to the fact that $\boldsymbol{K}$ is the matrix consisting of kernel functions, i.e., $\nabla \boldsymbol{K}(\exp(\boldsymbol{\theta}'))$, the partial derivative of $k(\boldsymbol{x}_i, \boldsymbol{x}_j; \exp(\boldsymbol{\theta}'))$ with respect to $\theta'_1, \theta'_2, \theta'_3$ is required, i.e.,

$$\frac{\partial k(\boldsymbol{x}_i, \boldsymbol{x}_j; \exp(\boldsymbol{\theta}'))}{\partial \theta'_1} = \exp(\theta'_1) \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2}{\exp(\theta'_2)}\right), \tag{13}$$

$$\frac{\partial k(\boldsymbol{x}_i, \boldsymbol{x}_j; \exp(\boldsymbol{\theta}'))}{\partial \theta'_2} = \frac{\exp(\theta'_1)}{\exp(\theta'_2)} ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2 \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2}{\exp(\theta'_2)}\right), \tag{14}$$

$$\frac{\partial k(\boldsymbol{x}_i, \boldsymbol{x}_j; \exp(\boldsymbol{\theta}'))}{\partial \theta'_3} = \exp(\theta'_3)\delta(i,j). \tag{15}$$

This calculation technique was described by Mochihashi et al. [29].

After the parameter is updated T_g times, we obtain $\boldsymbol{\theta}'^{(T_\mathrm{g})}$. Substituting this into Equation (9) yields the tuned KP $\boldsymbol{\theta}^{(T_\mathrm{g})}$. According to Equation (10), the obtained parameter $\boldsymbol{\theta}^{(T_\mathrm{g})}$ satisfies the condition of $\mathbb{R}^3_{>0}$.

2.3. Optimization Algorithm for Experiments

Using Equations (3) and (4), the acquisition function of GPBO is defined as

$$A(\boldsymbol{x}'; \beta, \boldsymbol{\theta}) = \mathbb{E}[y'|\boldsymbol{x}', \mathcal{D}, \boldsymbol{\theta}] + \beta\sqrt{\mathbb{V}[y'|\boldsymbol{x}', \mathcal{D}, \boldsymbol{\theta}]} \tag{16}$$

and it is called the UCB [27,28]. Here, with N observed samples, the $N+1$-th (next) observation sample is determined as follows:

$$\boldsymbol{x}_{N+1} = \underset{\boldsymbol{x}' \in \boldsymbol{\Psi}}{\mathrm{argmax}}\, A(\boldsymbol{x}'; \beta, \boldsymbol{\theta}), \tag{17}$$

$$y_{N+1} = f(\boldsymbol{x}_{N+1}) + \epsilon \tag{18}$$

where $\boldsymbol{\Psi}$ represents the domain of the input $\boldsymbol{x}'$. The second term of the acquisition function, which is defined in Equation (16), controls the search weight. When β is set to a large value, exploration is emphasized. Subsequently, the observation dataset is updated as

$$\mathcal{D} = \mathcal{D} \cup \{(\boldsymbol{x}_{N+1}, y_{N+1})\}. \tag{19}$$

By performing this process T_b times, we obtain the maximum value $y_{\max}$ and the approximate solution $\boldsymbol{x}_{\max}$ as follows:

$$y_{\max} = \max \mathcal{D}_y, \quad \boldsymbol{x}_{\max} = \operatorname*{argmax}_{\boldsymbol{x} \in \mathcal{D}_x} \mathcal{D}_y \tag{20}$$

where $\mathcal{D}_y$ and $\mathcal{D}_x$ are the sets consisting of the output and input values, respectively, of the set $\mathcal{D}$. This procedure is presented in Algorithm 1.

Algorithm 1 Verification-targeted optimization algorithm

Input:
Initial observation dataset $\mathcal{D}$, maximum number of BO iterations T_b,
maximum number of GD iterations T_g, GD learning rate γ, exploration weight β,
black-box function $f(\boldsymbol{x})$, observation noise ϵ,
search space $\boldsymbol{\Psi}$, initial KP $\boldsymbol{\theta}^{(0)} = \exp(\boldsymbol{\theta}'^{(0)})$

Output:
Solution $\boldsymbol{x}_{\max}$ and its value $y_{\max}$

```
1:  for $t_\mathrm{b} = 1$ to $T_\mathrm{b}$ do
2:      for $t_\mathrm{g} = 0$ to $T_\mathrm{g} - 1$ do
3:          $\boldsymbol{\theta}'^{(t_\mathrm{g}+1)} \leftarrow \boldsymbol{\theta}'^{(t_\mathrm{g})} + \gamma \nabla L\left(\exp(\boldsymbol{\theta}'^{(t_\mathrm{g})})\right)$
4:      end for
5:      $\boldsymbol{\theta}^{(T_\mathrm{g})} \leftarrow \exp\left(\boldsymbol{\theta}'^{(T_\mathrm{g})}\right)$
6:      $\boldsymbol{x}_{N+1} \leftarrow \operatorname*{argmax}_{\boldsymbol{x}' \in \boldsymbol{\Psi}} A(\boldsymbol{x}'; \beta, \boldsymbol{\theta}^{(T_\mathrm{g})})$
7:      $y_{N+1} \leftarrow f(\boldsymbol{x}_{N+1}) + \epsilon$
8:      $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{x}_{N+1}, y_{N+1})\}$
9:  end for
10: $y_{\max} \leftarrow \max \mathcal{D}_y$, $\boldsymbol{x}_{\max} \leftarrow \operatorname*{argmax}_{\boldsymbol{x} \in \mathcal{D}_x} \mathcal{D}_y$
11: return $\boldsymbol{x}_{\max}, y_{\max}$
```

Notes:
· Lines 2–4: Tuning the KPs via GD;
· Lines 6–8: Decisions regarding the next search point and observation;
· Lines 10–11: Obtain an approximate solution.
"GD": gradient descent, "BO": Bayesian optimization

2.4. Indices

The GPBO search performance depends on the training and generalization errors of the surrogate model. Therefore, the training and generalization errors are, respectively, defined as

$$E_\mathrm{t}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} (y_n - \mathbb{E}[y_n | \boldsymbol{x}_n, \mathcal{D}, \boldsymbol{\theta}])^2, \tag{21}$$

$$E_\mathrm{v}(\boldsymbol{\theta}) = \frac{1}{|\boldsymbol{\Psi}|} \sum_{\boldsymbol{x} \in \boldsymbol{\Psi}} (f(\boldsymbol{x}) - \mathbb{E}[y | \boldsymbol{x}, \mathcal{D}, \boldsymbol{\theta}])^2. \tag{22}$$

Moreover, the GPBO search performance depends on whether the next search point $\boldsymbol{x}_{N+1}$ and past observed points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N$ are close, i.e., if the search is performed only near past observation points, the search performance is poor. Therefore, we determine whether they are close to each other as follows:

$$\min\{||\boldsymbol{x}_{N+1} - \boldsymbol{x}_n||_2 \mid n = 1, \cdots, N\} < \omega \tag{23}$$

where ω denotes the threshold value. When the minimum distance on the left side is less than ω, the next search point $\boldsymbol{x}_{N+1}$ is close to the previously observed points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N$.

Furthermore, whether the KP vector $\boldsymbol{\theta}'^{(T_\mathrm{g})}$ obtained as T_g times GD has converged is determined as follows:

$$\frac{||\boldsymbol{\theta}'^{(T_\mathrm{g})} - \boldsymbol{\theta}'^{(T_\mathrm{g}-1)}||_2}{||\boldsymbol{\theta}'^{(T_\mathrm{g}-1)}||_2} < \tau \tag{24}$$

where τ is the threshold for the convergence criterion. The term on the left side is called the "relative change in parameters," and it is widely used as a convergence criterion [30–32].

Table 1. Input values for Algorithm 1. $\{\cdot\}$ is a set consisting of multiple elements, i.e., multiple patterns were adopted.

Parameters	**Value(s)**
Maximum number of GD iterations T_g	$\{0, 50, 100, \cdots, 500\}$
GD learning rate γ	0.01
Maximum number of BO iterations T_b	50
Exploration weight β	$\{1, 5\}$
Initial KP $\boldsymbol{\theta}^{(0)}$	$[1\ 1\ 1]^\top$
Search space $\boldsymbol{\Psi}$	Equation (25)
Black-box function $f(\boldsymbol{x})$	Equation (25)
Observation noise ϵ	0
Initial observation dataset $\mathcal{D}$	Four points randomly selected in $\boldsymbol{\Psi}$

3. Experiments

3.1. Objective and Outline

From the definition of $A(\boldsymbol{x}'; \beta, \boldsymbol{\theta})$, the search performance of GPBO depends on the KP $\boldsymbol{\theta}$. Due to the fact that the KP is obtained via GD, the maximum number of iterations for updating T_g significantly affects the GPBO search performance. When the number of iterations T_g is too high, the surrogate model is overfitted to the observation samples. In such cases, a relatively poor search performance is expected. When the exploration weight is set to a large value, because the observed samples occur in various regions in the input domain, the generalization error caused by overfitting may be mitigated. In contrast, when a small value is used for the exploration weight, because observation samples only occur in limited regions, the generalization error can be larger. Due to the fact that the generalization performance of the surrogate model affects the GPBO search performance, we consider that the negative effect of the overfitted KPs on the search performance depends on the exploration weight β. This hypothesis was verified through a simple numerical simulation.

We adopted the black-box function $f(\boldsymbol{x})$ and its search space $\boldsymbol{\Psi}$ as follows:

$$f(\boldsymbol{x}) = \frac{1}{3}\sin\frac{x_1}{3}\sin\frac{x_2}{3} - \frac{(x_1)^2}{300} - \frac{(x_2)^2}{300} + \frac{5}{6}, \quad \boldsymbol{x} \in \boldsymbol{\Psi} := [-10, 10] \times [-10, 10]. \tag{25}$$

For simplicity, we set the observation noise to $\epsilon = 0$. The black-box function is shown in Figure 1. The adopted black-box function had two optimal solutions.

We used GPBO based on Algorithm 1, and the values of the input parameters are presented in Table 1. To verify the aforementioned hypothesis, the effects of the degree of kernel-parameter tuning and the exploration weight on the GPBO search performance were analyzed. Therefore, we adopted multiple values for the maximum number of GD iterations T_g and exploration weight β. Moreover, the initial observation dataset $\mathcal{D}$ comprised four randomly selected points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_4$ from the search space $\boldsymbol{\Psi}$. To enhance the reliability of the results, we performed 20 experiments in which the same parameter conditions were used but the random seed identification was changed.

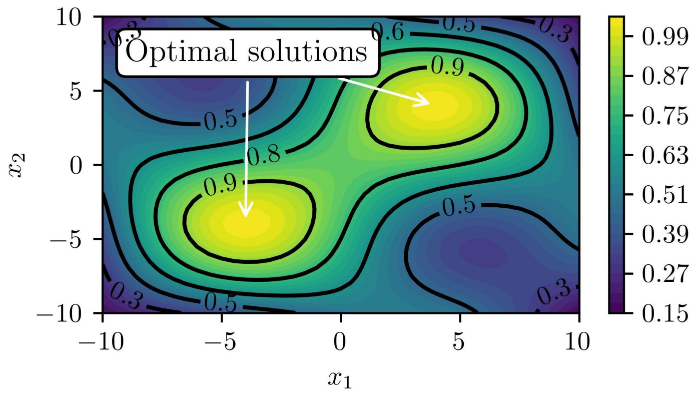

Figure 1. Adopted black-box function $f(\boldsymbol{x})$ defined by Equation (25).

3.2. Results and discussions

The relationship between the maximum number of GD iterations and the kernel-parameter tuning is presented in Figure 2A. The figure shows the rate at which the KPs $\boldsymbol{\theta}^{(T_\mathrm{g})}$ obtained via the GD of each Bayesian optimization step of 50 iterations converged. We investigated whether the convergence was determined by Equation (24) for $\tau = 0.01$. The results represent the averages of 20 trials with different seeds. For $\beta = 1$ and 5, Figure 2A suggests that the log-likelihood converged when the maximum number of GD iterations T_g was set as more than approximately 300. Therefore, we regarded the KPs as overtuned at >300 GD iterations.

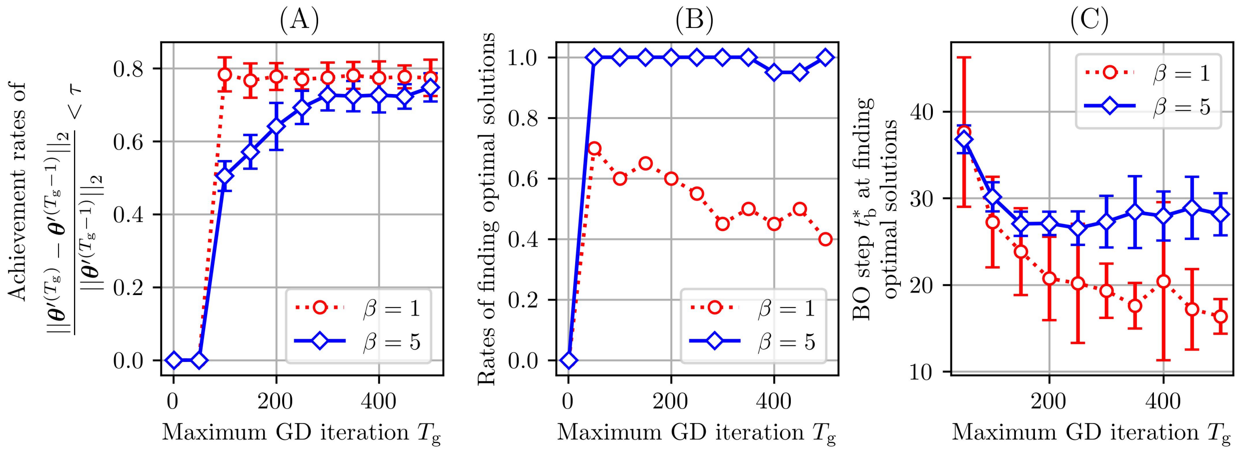

Figure 2. Relationships between the maximum number of GD iterations and other indices. Figure (**A**) presents the convergence achievement rates for Equation (24). (First, we averaged the results of all the Bayesian optimization steps ($T_\mathrm{b} = 50$). Then, the averages and standard deviations of the results for 20 seeds were calculated.) Figure (**B**) presents the rates of finding two optimal solutions (total of 20 seeds). Figure (**C**) presents the average number of optimization steps needed for finding two optimal solutions calculated with only the results where two optimal solutions were successfully found (total of 20 seeds). The error bars indicate the standard deviations.

Figure 2B shows the rate of finding the two optimal solutions in 20 trials with different seeds. The maximum number of GD iterations was the degree of kernel-parameter tuning via GD. For $\beta = 1$, a larger maximum number of GD iterations corresponded to worse search performance. In contrast, for $\beta = 5$, even when the maximum number of GD iterations was large, the rates of finding solutions remained high. Therefore, we consider that even if KPs are overtuned to the observation dataset, in cases of a large exploration weight, GPBO can adequately search for solutions (of course, it is desirable to avoid overfitting). Figure 2C presents the average number of Bayesian optimization steps needed for finding two optimal solutions calculated using only the results where two optimal solutions were successfully found. As shown, for a smaller value of β and a larger number of KPs tuned, the optimal solutions were found faster. However, as shown in Figure 2B, the rate of correctly finding the optimal solutions was lower. Therefore, for a small β, the KPs should not be excessively tuned. In contrast, for a large β, the negative effect caused by the overtuned KPs was not observed.

Next, we verified the training error $E_{\mathrm{t}}(\boldsymbol{\theta})$ and generalization error $E_{\mathrm{v}}(\boldsymbol{\theta})$ defined by Equations (21) and (22) for analyzing the effect of the exploration weight β on the GPBO search performance. Figure 3A,B show the training errors. Figure 3C,D show the generalization errors. We transformed them into log errors, that is, using $\log E_{\mathrm{t}}(\boldsymbol{\theta})$ instead of $E_{\mathrm{t}}(\boldsymbol{\theta})$. As the maximum number of GD iterations T_{g} and number of Bayesian optimization steps t_{b} increased, the training errors decreased. In contrast, in the latter half of the Bayesian optimization process, a higher number of tuned KPs (i.e., larger T_{g}) corresponded to larger generalization errors. When the generalization error of the surrogate model was large, because the reliability of the first term of the acquisition function defined by Equation (16) was low, the GPBO search performance was poor. From these results, for $\beta = 1$, we attribute the degradation of the GPBO search performance to the overtuning of the KPs, which increased the generalization error. For $\beta = 5$, even when the KPs were overtuned, there were many regions with small generalization errors. Therefore, in this case, we consider that the GPBO search performance was not degraded.

When observation samples occur in a limited region, the generalization error is large, because the surrogate model based on these data cannot correctly estimate the output values of various regions in the search space. To verify this hypothesis, we used Equation (23) with $\omega = 0.5$ and calculated the rate at which the next search point determined by Equation (17) and the observed points were close. The results are presented in Figure 3E,F. Figure 3E suggests that, when the exploration weight was set as $\beta = 1$, the GPBO searched for areas close to previously observed samples. This was particularly true when large numbers of KPs were tuned. For example, in the case of $\beta = 1$ and $T_{\mathrm{g}} = 500$, when the number of Bayesian optimization steps exceeded approximately 35 ($t_{\mathrm{b}} > 35$), the search area of GPBO remained close to the previously observed samples. This trend was weaker for a smaller value of T_{g}. Thus, there was a stronger tendency to search close to the previously observed samples when the KPs were overtuned. Therefore, we considered the GPBO search performance to be poor. Figure 3F suggests that, for $\beta = 5$, the GPBO searched various areas even if the KPs were overtuned. In summary, from Figure 3E,F, the degree of kernel-parameter tuning affected whether GPBO searched areas close to previously observed samples. Moreover, a smaller exploration weight β corresponded to a higher risk. Thus, for a smaller value of β, there should be more focus on kernel-parameter tuning.

Next, we present the changes in the surrogate model for each Bayesian optimization step in Figures 4 and 5. These values were $\beta = 1, 5$, respectively. Due to the fact that we cannot show the results for all the seeds, the results for a specific seed are shown. In the case where no KPs were tuned, that is, $T_{\mathrm{g}} = 0$, appropriate searches were not performed regardless of β. For $T_{\mathrm{g}} = 0$, Figure 3A–D indicate that the training and generalization errors were large. We believe that, because the surrogate model was inappropriate, the GPBO search was inappropriate.

With a small number of KPs tuned, i.e., $T_{\mathrm{g}} = 50$, the average $\mathbb{E}$ generated from the surrogate model nearly succeeded in reproducing the black-box function $f(\boldsymbol{x})$ regardless of β. Therefore, the rate of correctly finding the optimal solution was high (see Figure 2B).

Figures 4 and 5 suggest that, when the number of iterations of KP tuning was $T_{\mathrm{g}} = 500$, the search results depended on the exploration weight β. For $\beta = 1$, because the GPBO searched only the neighborhood of the best point from the initial solutions, even if the search progressed, the output from the surrogate model $\mathbb{E}$ could not reproduce the black-box function $f(\boldsymbol{x})$. In contrast, for $\beta = 5$, because observation samples appeared in various areas, even if the KPs were overtuned, the output of the surrogate model $\mathbb{E}$ reproduced the black-box function $f(\boldsymbol{x})$. The results indicate that the negative effect of overtuning the KPs on the GPBO search performance can be mitigated by increasing the exploration weight β.

However, in both the $\beta = 1$ and 5 cases, when the number of iterations of KP tuning increased ($T_g = 500$), the surrogate model output was inappropriate when the number of BO steps t_b was approximately 30–35 (the cases of $T_g = 500$ are shown in Figures 4 and 5). In general, overfitting parameters to observed data increases the risk of reduced estimation performance in areas where there are no observed data. Therefore, overfitting can easily occur with high T_g values. Due to the fact that this occurs in both the cases of $\beta = 1$ and 5, it is important not to overtune the KPs, regardless of the exploration weight. Although the negative effects of overtuned KPs can be mitigated by increasing the exploration weight, they cannot be completely eliminated.

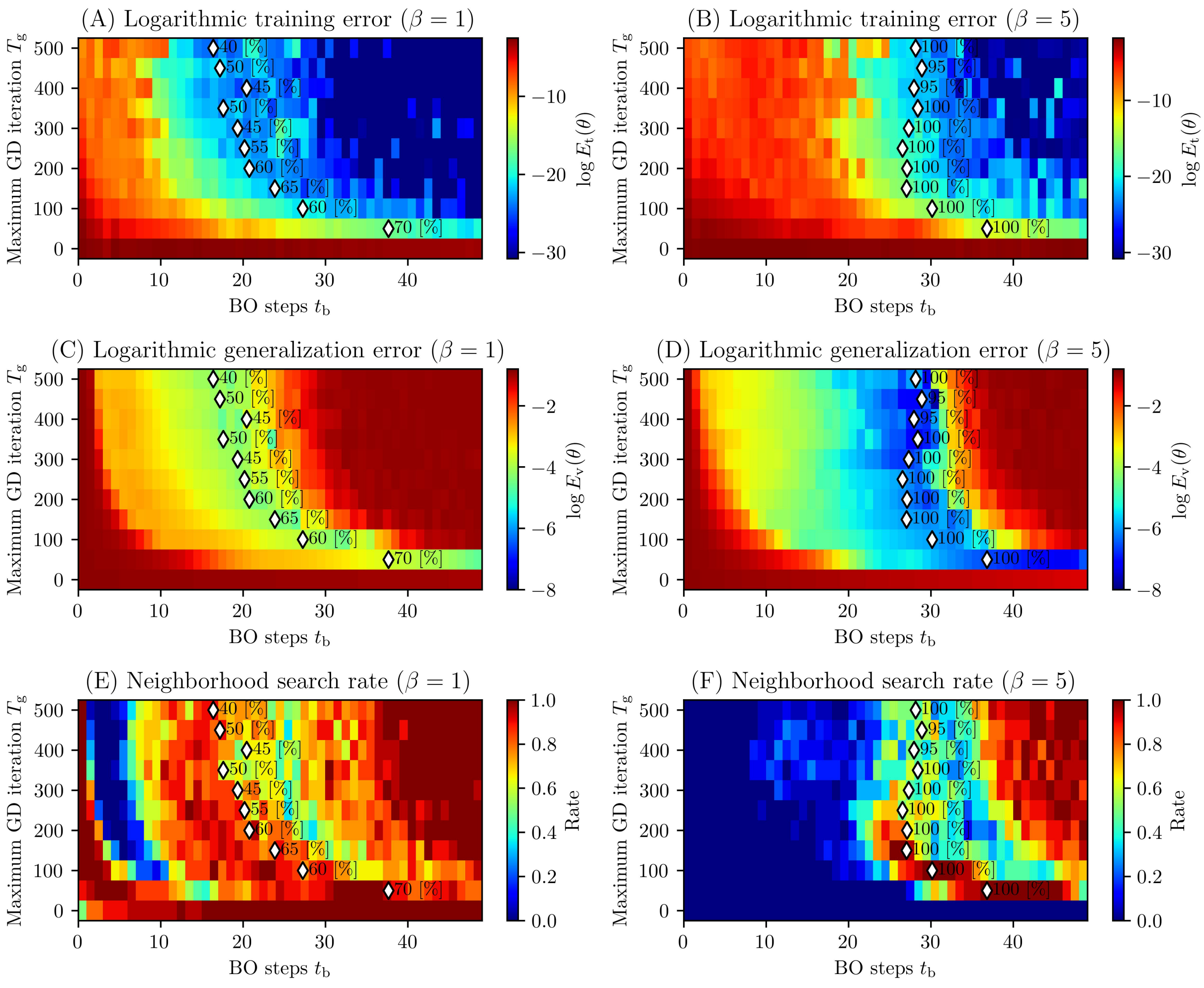

Figure 3. (**A**,**B**) Training errors of surrogate models calculated using Equation (21) for each maximum number of GD iterations; (**C**,**D**) generalization errors calculated using Equation (22); (**E**,**F**) rates of exploring the neighborhood of past observed samples. The values of (**A**–**D**) are the averages of 20 seeds, and those of (**E**,**F**) are the rates of 20 seeds. The white markers indicate the timings of finding two optimal solutions, and the percentages are the success rates of finding them in 20 trials with different seeds. These results are presented in Figure 2B,C. In the cases of $\beta = 1$, with a larger maximum number of GD iterations, although a solution was found faster, the success rate was worse. At $\beta = 5$, no such trend was observed.

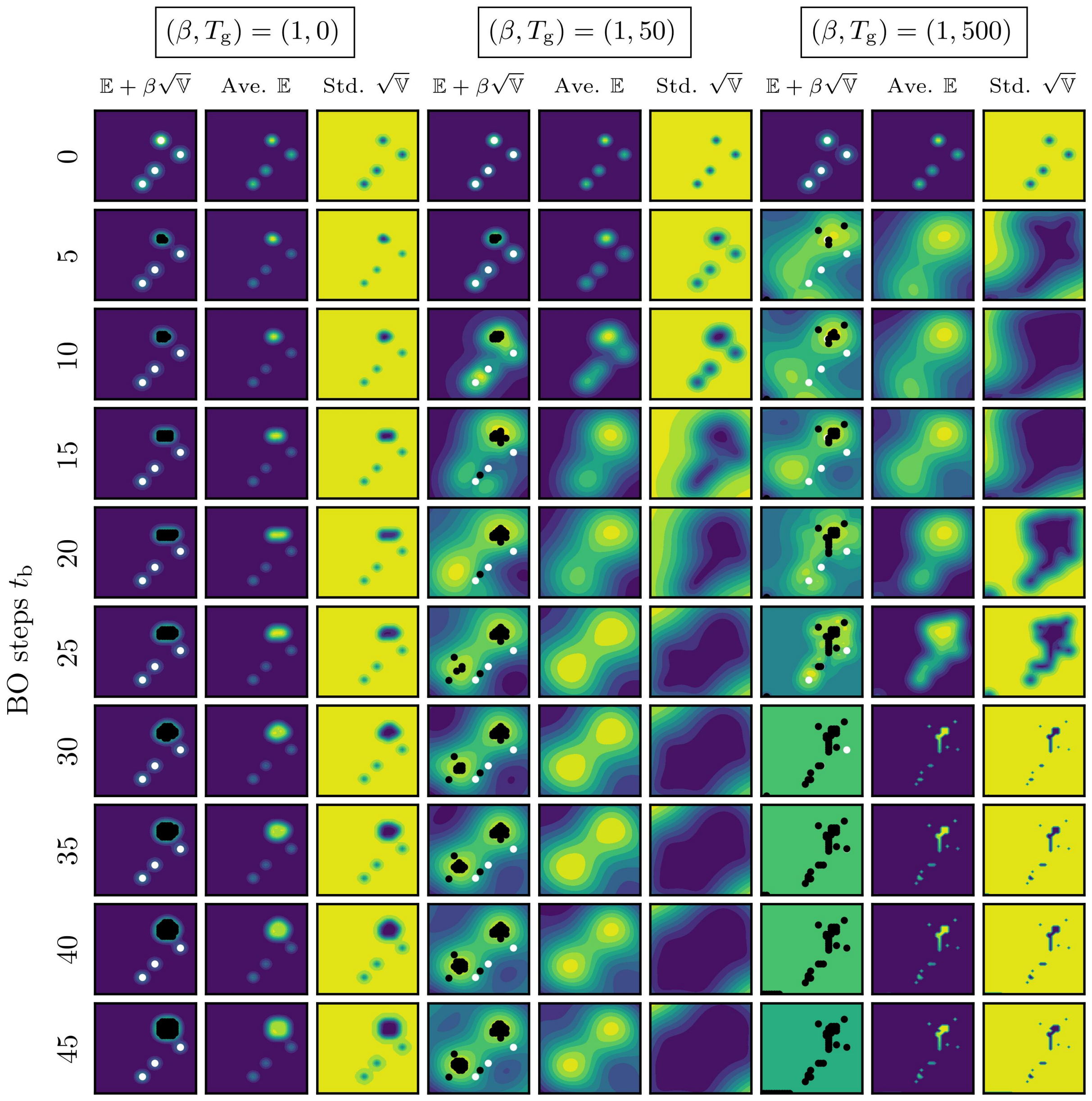

Figure 4. Average $\mathbb{E}$, standard deviation $\sqrt{\mathbb{V}}$ of the surrogate model, and the acquisition function $\mathbb{E} + \beta\sqrt{\mathbb{V}}$ for the maximum number of GD iterations T_g and exploration weight $\beta = 1$ calculated using Equations (3), (4) and (16). The white circles represent initial points, and the black circles represent observation points selected by the acquisition function.

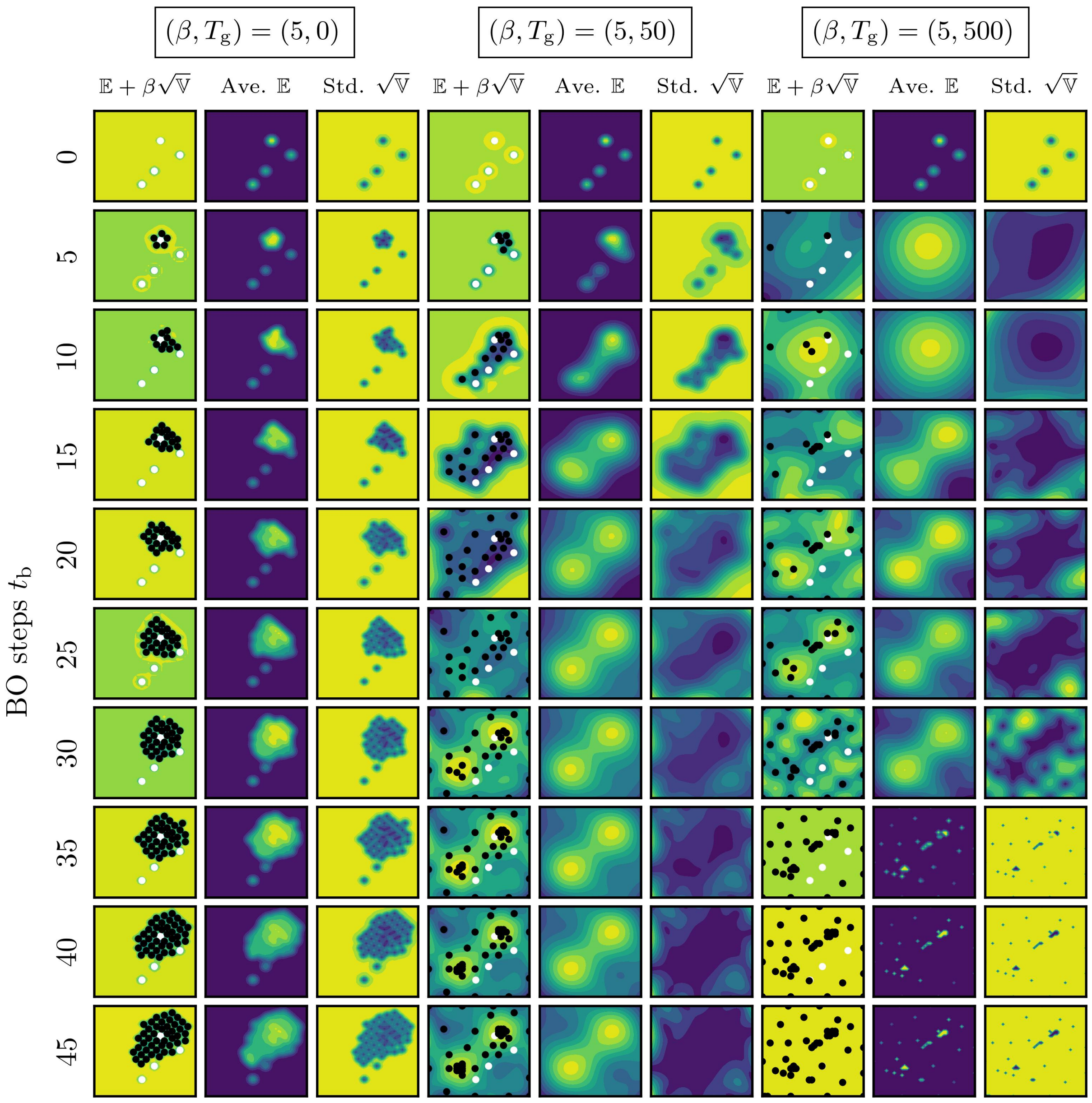

Figure 5. Average $\mathbb{E}$, standard deviation $\sqrt{\mathbb{V}}$ of the surrogate model, and the acquisition function $\mathbb{E} + \beta\sqrt{\mathbb{V}}$ for the maximum number of GD iterations T_{g} and exploration weight $\beta = 5$ calculated using Equations (3), (4) and (16). The white circles represent initial points, and the black circles represent observation points selected by the acquisition function.

4. Conclusions

The exploration weight affects the degree of exploration of GPBO. Therefore, the GPBO search performance depends on the exploration weight. In this study, we analyzed the mitigation of the negative effect of overtuning KPs as another effect of the exploration weight. The results indicate that we should pay attention to overtuning the KPs in the case of Bayesian optimization with a small exploration weight. It is preferable to use methods for avoiding the overtuning of KPs, e.g., early stopping of the GD. In contrast, for large exploration weights, the solution discovery rate is high even when overtuning the KPs. These findings are useful in all situations wherein GPBO is used, e.g., hyperparameter tuning in machine learning.

The results and discussions presented in this paper are entirely based on the black-box function defined by Equation (25). The function has two optimal solutions clearly visible in the domain. However, it is not known whether the results of this study would be reproduced if flat functions such as the Beale function [33] or the Goldstein–Price function [34] (the minimum search) were adopted. As shown in Figure 6, the structure of these functions is apparent after log transformation. A similar result can be obtained with a flat function, or a logarithmic transformation may have to be performed. These points are unclear and will be discussed in a future work.

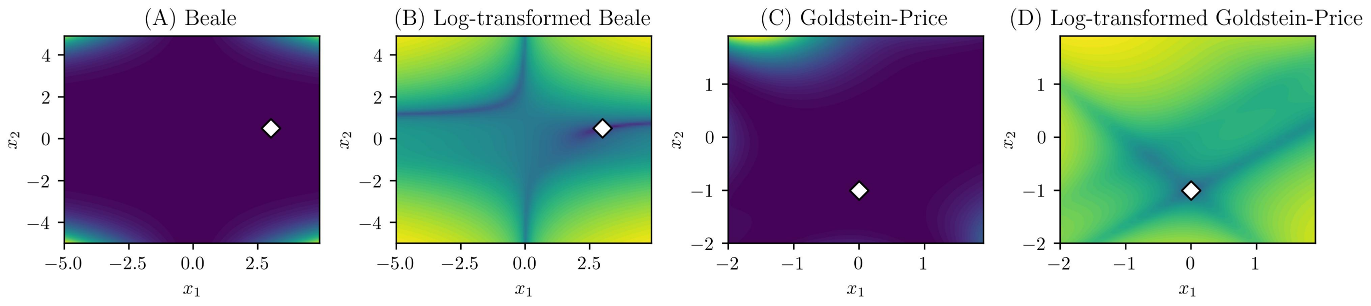

Figure 6. (**A**) Beale function [33], (**B**) log-transformed Beale function, (**C**) Goldstein–Price function [34], and (**D**) log-transformed Goldstein–Price function. The white diamond mark represents the global optimal solution.

Funding: This work was supported in part by JSPS Grants-in-Aid for Scientific Research (C) (Grant Nos. 21K04535 and 23K11310) and a JSPS Grant-in-Aid for Young Scientists (Grant No. 19K20062).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Saleh, E.; Tarawneh, A.; Naser, M.Z.; Abedi, M.; Almasabha, G. You only design once (YODO): Gaussian Process-Batch Bayesian optimization framework for mixture design of ultra high performance concrete. *Constr. Build. Mater.* **2022**, *330*, 127270. [CrossRef]
2. Mathern, A.; Steinholtz, O.S.; Sjöberg, A.; Önnheim, M.; Ek, K.; Rempling, R.; Gustavsson, E.; Jirstrand, M. Multi-objective constrained Bayesian optimization for structural design. *Struct. Multidiscip. Optim.* **2021**, *63*, 689–701. [CrossRef]
3. Frazier, P.I.; Wang, J. Bayesian optimization for materials design. *Springer Ser. Mater. Sci.* **2015**, *225*, 45–75. [CrossRef]
4. Ohno, H. Empirical studies of Gaussian process based Bayesian optimization using evolutionary computation for materials informatics. *Expert Syst. Appl.* **2018**, *96*, 25–48. [CrossRef]
5. Ueno, T.; Rhone, T.D.; Hou, Z.; Mizoguchi, T.; Tsuda, K. COMBO: An efficient Bayesian optimization library for materials science. *Mater. Discov.* **2016**, *4*, 18–21. [CrossRef]
6. Elsayad, A.M.; Nassef, A.M.; Al-Dhaifallah, M. Bayesian optimization of multiclass SVM for efficient diagnosis of erythemato-squamous diseases. *Biomed. Signal Process. Control* **2022**, *71*, 103223. [CrossRef]
7. Agrawal, A.K.; Chakraborty, G. On the use of acquisition function-based Bayesian optimization method to efficiently tune SVM hyperparameters for structural damage detection. *Struct. Control. Health Monit.* **2021**, *28*, e2693. . [CrossRef]
8. Xie, W.; Nie, W.; Saffari, P.; Robledo, L.F.; Descote, P.Y.; Jian, W. Landslide hazard assessment based on Bayesian optimization–support vector machine in Nanping City, China. *Nat. Hazards* **2021**, *109*, 931–948. [CrossRef]
9. Wu, J.; Chen, X.Y.; Zhang, H.; Xiong, L.D.; Lei, H.; Deng, S.H. Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization. *J. Electron. Sci. Technol.* **2019**, *17*, 26–40. [CrossRef]
10. Kumar, P.; Nair, G.G. An efficient classification framework for breast cancer using hyper parameter tuned Random Decision Forest Classifier and Bayesian Optimization. *Biomed. Signal Process. Control* **2021**, *68*, 102682. [CrossRef]
11. Snoek, J.; Larochelle, H.; Adams, R.P. Practical Bayesian Optimization of Machine Learning Algorithms. In Proceedings of the 25th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012.
12. Kolar, D.; Lisjak, D.; Pajak, M.; Gudlin, M. Intelligent Fault Diagnosis of Rotary Machinery by Convolutional Neural Network with Automatic Hyper-Parameters Tuning Using Bayesian Optimization. *Sensors* **2021**, *21*, 2411. [CrossRef] [PubMed]
13. Snelson, E.; Ghahramani, Z. Local and global sparse Gaussian process approximations. In Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics, San Juan, Puerto Rico, 21–24 March 2007; pp. 524–531.
14. Snelson, E.; Ghahramani, Z. Sparse Gaussian Processes using Pseudo-inputs. In Proceedings of the 18th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada 5–8 December 2005.
15. Csató, L.; Opper, M. Sparse On-Line Gaussian Processes. *Neural Comput.* **2002**, *14*, 641–668. [CrossRef] [PubMed]

16. Seeger, M.W.; Williams, C.K.I.; Lawrence, N.D. Fast Forward Selection to Speed Up Sparse Gaussian Process Regression. In Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, Key West, FL, USA, 3–6 January 2003; pp. 254–261.
17. Chen, H.; Zheng, L.; Kontar, R.A.; Raskutti, G. Gaussian Process Parameter Estimation Using Mini-batch Stochastic Gradient Descent: Convergence Guarantees and Empirical Benefits. *J. Mach. Learn. Res.* **2022**, *23*, 1–59.
18. Martino, L.; Laparra, V.; Camps-Valls, G. Probabilistic cross-validation estimators for Gaussian Process regression. In Proceedings of the 25th European Signal Processing Conference, EUSIPCO 2017, Kos, Greece, 28 August–2 September 2017; pp. 823–827. [CrossRef]
19. Zhang, R.; Zhao, X. Inverse Method of Centrifugal Pump Blade Based on Gaussian Process Regression. *Math. Probl. Eng.* **2020**, *2020*, 4605625. [CrossRef]
20. Senanayake, R.; O'callaghan, S.; Ramos, F. Predicting Spatio-Temporal Propagation of Seasonal Influenza Using Variational Gaussian Process Regression. *Proc. AAAI Conf. Artif. Intell.* **2016**, *30*, 3901–3907. [CrossRef]
21. Petelin, D.; Filipic, B.; Kocijan, J. Optimization of Gaussian process models with evolutionary algorithms. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6593, pp. 420–429. [CrossRef]
22. Ouyang, Z.L.; Zou, Z.J. Nonparametric modeling of ship maneuvering motion based on Gaussian process regression optimized by genetic algorithm. *Ocean Eng.* **2021**, *238*, 109699. [CrossRef]
23. Cheng, L.; Ramchandran, S.; Vatanen, T.; Lietzen, N.; Lahesmaa, R.; Vehtari, A.; Lähdesmäki, H. LonGP: An additive Gaussian process regression model for longitudinal study designs. *bioRxiv* **2018**, 259564. [CrossRef]
24. Israelsen, B.; Ahmed, N.; Center, K.; Green, R.; Bennett, W., Jr. Adaptive Simulation-Based Training of Artificial-Intelligence Decision Makers Using Bayesian Optimization. *J. Aerosp. Comput. Inf. Commun.* **2018**, *15*, 38–56. . [CrossRef]
25. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006.
26. Deringer, V.L.; Bartók, A.P.; Bernstein, N.; Wilkins, D.M.; Ceriotti, M.; Csányi, G. Gaussian Process Regression for Materials and Molecules. *Chem. Rev.* **2021**, *121*, 10073–10141. . [CrossRef]
27. Oliveira, R.; Ott, L.; Ramos, F. Bayesian optimisation under uncertain inputs. In Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics, Naha, Japan, 16–18 April 2019; Volume 89, pp. 1177–1184.
28. Ath, G.D.; Everson, R.M.; Rahat, A.A.M.; Fieldsend, J.E. Greed is Good: Exploration and Exploitation Trade-offs in Bayesian Optimisation. *ACM Trans. Evol. Learn. Optim.* **2021**, *1*, 1–22. [CrossRef]
29. Mochihashi, D.; Oba, S. *Gaussian Process and Machine Learning*; Kodansha Scientific, Tokyo, Japan, 2019.
30. Blonigen, B.A.; Knittel, C.R.; Soderbery, A. Keeping it Fresh: Strategic Product Redesigns and Welfare. *Int. J. Ind. Organ.* **2017**, *53*, 170–214. [CrossRef]
31. Mareček, R.; ŘÍha, P.; Bartoňová, M.; Kojan, M.; Lamoš, M.; Gajdoš, M.; VojtÍšek, L.; Mikl, M.; Bartoň, M.; Doležalová, I.; et al. Automated fusion of multimodal imaging data for identifying epileptogenic lesions in patients with inconclusive magnetic resonance imaging. *Hum. Brain Mapp.* **2021**, *42*, 2921–2930. [CrossRef] [PubMed]
32. Che, K.; Chen, X.; Guo, M.; Wang, C.; Liu, X. Genetic Variants Detection Based on Weighted Sparse Group Lasso. *Front. Genet.* **2020**, *11*, 155. [CrossRef] [PubMed]
33. Surjanovic, S.; Bingham, D. Beale Function. 2013. Available online: https://www.sfu.ca/~ssurjano/beale.html (accessed on 27 June 2023).
34. Surjanovic, S.; Bingham, D. Goldstein-Price Function. 2013. Available online: https://www.sfu.ca/~ssurjano/goldpr.html (accessed on 27 June 2023).