

ENSEMBLE NEURAL NETWORK-BASED SURROGATE OPTIMIZATION WITH ADJUSTABLE LOGARITHMIC SEARCH WEIGHT IN AN UPPER CONFIDENCE BOUND

KYOYA SASAKI, YUTO OMAE AND JUN TOYOTANI

College of Industrial Technology
Nihon University
1-2-1, Izumi, Narashino, Chiba 275-8575, Japan
ciky25019@g.nihon-u.ac.jp; { oomae.yuuto; toyotani.jun }@nihon-u.ac.jp

Received December 2025; revised April 2026

ABSTRACT. *One promising approach for black-box optimization is Ensemble Neural Network-based Surrogate Optimization (ENN-SO). In this method, the next evaluation point is selected using the predictive mean and variance estimated by multiple networks. The optimization performance is influenced by the weight assigned to the predictive variance. In SO, it is desirable to emphasize exploration in the early steps and exploitation in the later steps. To achieve this, previous studies commonly employed search weights based on logarithmic decay. Among these, the search weight based on logarithmic decay has demonstrated superior performance compared to the constant weight. However, it suffers from several drawbacks: 1) the initial weight cannot be freely specified, 2) the maximum value of the weight is fixed, and 3) the final weight cannot be adjusted arbitrarily. To address these limitations, we propose a novel search weighting scheme that overcomes the issues associated with logarithmic decay. To evaluate the effectiveness of the proposed method, we conducted performance experiments using six benchmark functions. The results showed that, after 40 search iterations, the proposed method outperformed the existing method on four of the six functions and achieved comparable performance to the existing method on the remaining two functions. Therefore, employing the proposed method in search experiments has the potential to yield superior performance compared to conventional approaches, even under limited resource conditions.*

Keywords: Logarithmic search weight, Upper confidence bound, Surrogate optimization

1. Introduction. Algorithms such as the gradient method and genetic algorithms have been widely used to find approximate maximum values of an objective function $f(\mathbf{x})$ with respect to input conditions $\mathbf{x}$. However, these methods typically assume that $f(\mathbf{x})$ is either not a black-box function or can be evaluated at a low cost. When $f(\mathbf{x})$ is a black-box function with high evaluation costs, Surrogate Optimization (SO) becomes an effective approach. SO uses a low cost regression model approximating the expensive objective function $f(\mathbf{x})$. Because the surrogate model closely approximates the behavior of $f(\mathbf{x})$, it allows one to estimate $f(\mathbf{x})$ based solely on inexpensive evaluations, without the need to perform expensive physical experiments. Using this surrogate model, we search for points $\mathbf{x}$ which are insufficiently observed and expected to have high $f(\mathbf{x})$. The next evaluation point $\mathbf{x}_{\text{new}}$ is then selected based on this criterion. This process is repeated, and the model is updated iteratively. The acquisition function is constructed based on a surrogate model trained on data obtained through experiments. SO has been used across various fields. For example, Raul and Leifsson [1] applied SO to reducing the computational cost of physical

simulations in the field of fluid dynamics, while Ren et al. [2] employed SO for materials fabrication. Several studies have demonstrated that the application of SO enables the automation of the entire process – from the input stage, including material fabrication, to the acquisition and evaluation of experimental results [3, 4]. SO using Neural Network (NN) has been employed in studies such as fluid dynamics [5] and crystal plasticity simulations [6]. Among various types of acquisition functions, this study employed the Upper Confidence Bound (UCB) [7]. This approach has been adopted in multiple studies and has demonstrated high effectiveness [8, 9, 10]. Liu et al. [11] applied SO to the simultaneous optimization of 3D packing design and motion planning in robotics, reporting a substantial improvement in computational efficiency compared to conventional methods. In addition, Chen et al. [12] introduced SO into the wind farm layout optimization problem, demonstrating its effectiveness in multi-objective optimization that balances power generation performance and installation cost. Passmore et al. [13] employed SO for optimizing thin-film coating process parameters and showed that high-quality experimental conditions can be efficiently identified within manufacturing processes.

UCB is defined as $E + \beta V^{1/2}$, where E and V are estimated mean and variance, respectively. E represents exploitation, while V represents exploration. Exploitation and exploration are in a trade-off relationship, which is balanced using β .

In UCB, the previous studies [14, 15] employed a constant value of β . However, using a constant search weight poses several issues. When β is fixed and small, exploration proceeds without accounting for the influence of V , causing the process to repeatedly explore the same regions of the search space. On the other hand, when β is fixed and large, exploration proceeds without accounting for the influence of E . This causes the process to focus excessively on highly uncertain regions in the search space, making it difficult to obtain good scores. Therefore, it is desirable to adjust β dynamically – large in early steps and smaller later – to balance exploration and exploitation.

The previous studies [16, 17] used $\beta = \sqrt{\log t/t}$, which is a logarithmic decay function for β . Here, t represents the search iterations. This allows the search to emphasize V in the early steps and E in the later steps of exploration. However, $\beta = \sqrt{\log t/t}$ has several issues. These are Problem 1 – The search weight function exhibits an extremum; Problem 2 – The initial weight cannot be freely determined; Problem 3 – The terminal weight cannot be freely determined. These issues are illustrated in Figure 1. As seen on the left side of Figure 1 (Existing method), the search weight is low at the start of exploration and then increases, exhibiting an extremum (Problem 1). Even if the function is shifted to the left to solve Problem 1, the maximum value is determined by the shape of the function,

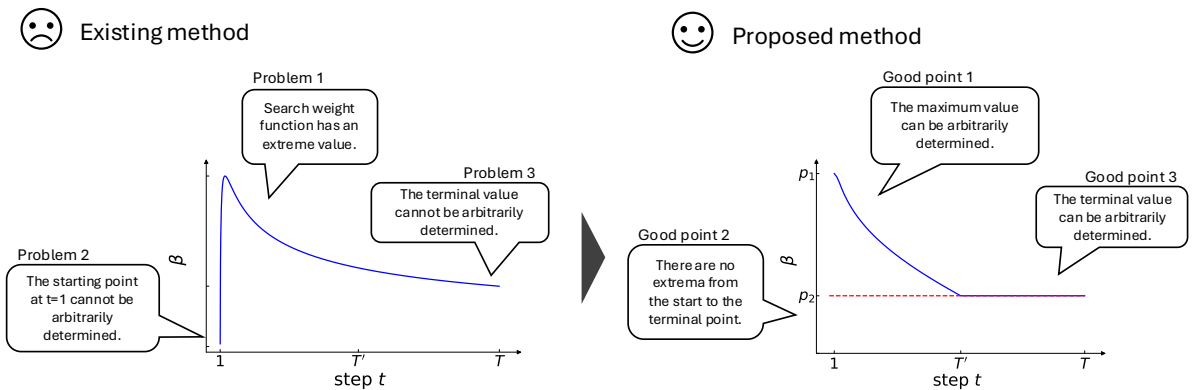


FIGURE 1. Search weights for $E + \beta V^{1/2}$ (Left: the existing method; Right: the proposed method)

preventing free adjustment of the initial search weight (Problem 2). Although the search weight decays according to the number of search iterations, the terminal value of the search weight is determined by the shape of the function and cannot be freely adjusted (Problem 3). In this research, we propose a new logarithmic decay-based search weight for UCB exploration that calculates E and V using Ensemble Neural Network-based Surrogate Optimization (ENN-SO) [18] as the surrogate model, building upon the existing method, specifically the UCB exploration with a constant search weight. Existing research [18] investigated the relationship between the number of hidden-layer units in ENN-SO and search performance. In contrast, although this study also uses ENN-SO as a surrogate model, it focuses on the design of the search weight β in UCB. We propose an adjustable logarithmic search weight that resolves the limitations of the existing logarithmic decay weight: 1) the initial weight cannot be freely specified, 2) the maximum value of the weight is fixed, and 3) the final weight cannot be adjusted arbitrarily. Observation values versus the number of searches in the existing method and the proposed method will be compared using key benchmark functions in Section 3. This paper is organized as follows. Section 2 presents the proposed method, Section 3 reports the experimental settings and results, and Section 4 concludes the paper.

2. Proposed Method. In the proposed method, SO iteratively selects the next observation point using the acquisition function UCB. Therefore, Section 2.1 describes the process of calculating E and V using NNs, and the overall SO procedure. Furthermore, Section 2.2 provides details of the proposed β formulation.

2.1. ENN-SO algorithm. In this study, the objective function is defined such that $f(\mathbf{x}) \in [-1, 1]$. The input is defined as $\mathbf{x} = [x_1, \dots, x_d]^\top$, where $\mathbf{x} \in \Psi$. The ENN-SO aims to acquire $\max_{\mathbf{x} \in \Psi} f(\mathbf{x})$ while exploring space with high uncertainty. In this work, the uncertainty of $f(\mathbf{x})$ is quantified using the variance. Since quantifying variance requires multiple predictive models, K regression-based NNs are constructed to approximate $f(\mathbf{x})$. Let θ_k denote the parameters of the k -th NN (where $k = 0, 1, \dots, K-1$) obtained through training. The parameters θ_k are learned based on the observed dataset D_t at step t . The initial observation dataset D_0 at step 0 consists of N samples from the initial solutions.

The predicted value of an NN with parameters for an input coordinate $\mathbf{x}$ is denoted by $b(\mathbf{x}, \theta_k)$. Let $\Theta = \{\theta_k \mid k = 0, \dots, K-1\}$ be the set of parameters for all K neural networks. Then, the estimated mean $E(\mathbf{x}, \Theta)$ and estimated variance $V(\mathbf{x}, \Theta)$ of the objective function for an experimental condition $\mathbf{x}$ are defined as

$$E(\mathbf{x}, \Theta) = \frac{1}{K} \sum_{k=0}^{K-1} b(\mathbf{x}, \theta_k), \quad V(\mathbf{x}, \Theta) = \frac{1}{K} \sum_{k=0}^{K-1} (E(\mathbf{x}, \Theta) - b(\mathbf{x}, \theta_k))^2. \quad (1)$$

$E(\mathbf{x}, \Theta)$ represents the expected value of the objective function at $\mathbf{x}$ while $V(\mathbf{x}, \Theta)$ indicates the level of uncertainty associated with that outcome. Accordingly, the UCB can be defined as $E(\mathbf{x}, \Theta) + \beta \sqrt{V(\mathbf{x}, \Theta)}$. This definition is based on the assumption that the experimental outcome follows a Gaussian distribution. Since it is desirable to explore regions with high expected mean and large uncertainty, the next observation point is defined as

$$\mathbf{x}_{\text{new}} = \underset{\mathbf{x} \in \Psi}{\operatorname{argmax}} \left(E(\mathbf{x}, \Theta) + \beta \sqrt{V(\mathbf{x}, \Theta)} \right). \quad (2)$$

In existing methods, a constant value is used for β [19, 20]. In this study, instead of using a constant weight, we propose a logarithmic function $q(t, p_1, p_2, T', \lambda^*)$ with tunable weights (see Algorithm 1, line 11). The details of $q(t, p_1, p_2, T', \lambda^*)$ are described in Section 2.2.

Algorithm 1 Ensemble Neural Network-based Surrogate Optimization (ENN-SO)

Input: Initial observation set $D_0 = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, Surrogate iterations T , The number of NN K , NN iterations M , NN learning rate η , Search space Ψ , Initial weight p_1 , Final weight p_2 , The timing when β reaches the final weight T'

Output: Solution $\mathbf{x}^*$

```

1:  $D \leftarrow D_0$ 
2: for  $t = 0$  to  $T - 1$  do
3:    $\Theta \leftarrow \{\}$ 
4:   for  $k = 0$  to  $K - 1$  do
5:      $\boldsymbol{\theta}_k^{(0)} \leftarrow \text{Randomly Generated}$ 
6:     for  $m = 0$  to  $M - 1$  do
7:        $\boldsymbol{\theta}_k^{(m+1)} \leftarrow \boldsymbol{\theta}_k^{(m)} - \eta \frac{\partial L}{\partial \boldsymbol{\theta}_k^{(m)}}$ 
8:     end for
9:      $\Theta \leftarrow \Theta \cup \{\boldsymbol{\theta}_k^{(M)}\}$ 
10:  end for
11:   $\beta = q(t, p_1, p_2, T', \lambda^*)$ , Equation (13)
12:   $\mathbf{x}_{|D_t|+1} \leftarrow \underset{\mathbf{x} \in \Psi}{\operatorname{argmax}} \left( E(\mathbf{x}, \Theta) + \beta \sqrt{V(\mathbf{x}, \Theta)} \right)$ , Equation (2)
13:   $y_{|D_t|+1} \leftarrow f(\mathbf{x}_{|D_t|+1})$ 
14:   $D_{t+1} \leftarrow D_t \cup \{(\mathbf{x}_{|D_t|+1}, y_{|D_t|+1})\}$ 
15: end for
16:  $\mathbf{x}^* = \mathbf{x}_{i^*}$  s.t.  $i^* = \underset{i \in \{1, \dots, |D_0|+T\}}{\operatorname{argmax}} y_i$ 
17: return  $\mathbf{x}^*$ 

```

2.2. Proposed logarithmic search weight.

2.2.1. *Addressing Problem 1.* Typically, the logarithmic search weight is defined as

$$w(t, a) = \left(\frac{\log(t + a)}{t + a} \right)^{\frac{1}{2}} \quad (3)$$

with $a = 0$ being the standard choice [17]. However, as shown on the left side of Figure 1 (Problem 1), using $a = 0$ causes the weight to be low at the beginning of the exploration, increase sharply, and then gradually decrease. To address this issue, we introduce a shift parameter a that moves the entire function to the left. This makes it possible to set a such that $w(t, a)$ reaches its maximum at $t = 0$. This leads to the problem of determining the value of a so that the maximum of the function occurs at $dw(t, a)/dt = 0$, $t \in [1, T]$.

The derivative of the function is given by

$$\frac{dw(t, a)}{dt} = \frac{\sqrt{\frac{\log(a+t)}{a+t}} (1 - \log(a+t))}{2(a+t) \log(a+t)} = 0. \quad (4)$$

Therefore, solving $1 - \log(t + a) = 0$ yields $t = e - a$. This implies that the peak of $w(t)$ occurs at $t = e - a$. Since we want to start the exploration at $t = 1$ and avoid any extrema within the search period, we set $a = e - 1$. From this point onward, we fix $a = e - 1$, and define the function as

$$w(t) = \left(\frac{\log(t + e - 1)}{t + e - 1} \right)^{\frac{1}{2}}. \quad (5)$$

2.2.2. *Addressing Problem 2.* As shown on the left side of Figure 1 (Problem 2), the user cannot arbitrarily set the initial search weight at $t = 1$. To resolve this issue, we define

$$g(t, p_1) = p_1 \frac{w(t)}{w(1)} = p_1 e^{\frac{1}{2}} \left(\frac{\log(t + e - 1)}{t + e - 1} \right)^{\frac{1}{2}}, \quad (6)$$

so that the weight at $t = 1$ becomes $g(1, p_1) = p_1$, allowing the user to freely control the initial search weight.

2.2.3. *Addressing Problem 3.* As shown on the left side of Figure 1 (Problem 3), the weight at an arbitrary point T' , i.e., $g(T', p_1)$, depends on the shape of the function and cannot be freely adjusted. As a result, it becomes difficult to satisfy the desired condition of reducing the search weight toward the end of the search. To address this, we introduce an auxiliary function $h(t, p_2, T', \lambda)$ such that the weight at $t = T'$ can be controlled by p_2 . The overall search weight is then defined as

$$q(t, p_1, p_2, T', \lambda) = g(t, p_1) + h(t, p_2, T', \lambda). \quad (7)$$

To ensure that the weight q equals p_2 at $t = T'$, we define the auxiliary function h as

$$h(t, p_2, T', \lambda) = C(p_2, T') (1 - e^{-\lambda(t-1)}), \quad \lambda > 0. \quad (8)$$

This is an exponential-shaped function whose curvature is controlled by the parameter λ . Since a positive λ makes h a converging function, we constrain $\lambda > 0$. The shape of the overall function $q(t, p_1, p_2, T', \lambda)$ depends on the value of $C(p_2, T')$. We require that the search weight be p_1 at $t = 1$ and p_2 at $t = T'$, which leads to the following conditions:

$$q(1, p_1, p_2, T', \lambda) = p_1, \quad q(T', p_1, p_2, T', \lambda) = p_2. \quad (9)$$

Therefore, we must determine $C(p_2, T')$ such that these conditions are satisfied.

Here, since $g(1, p_1) = p_1$ and $h(1, p_2, T', \lambda) = 0$ hold true for any value of $C(p_2, T')$, the condition $q(1, p_1, p_2) = p_1$ is always satisfied. Therefore, we only need to consider the condition $q(T', p_1, p_2, \lambda) = p_2$. Therefore, we solve the following equation for $C(p_2, T')$:

$$p_2 = q(T', p_1, p_2, T', \lambda) = g(T', p_1) + C(p_2, T') (1 - e^{-\lambda(T'-1)}). \quad (10)$$

Solving for $C(p_2, T')$, we obtain

$$C(p_2, T') = \frac{p_2 - g(T', p_1)}{1 - e^{-\lambda(T'-1)}}. \quad (11)$$

Substituting this back into Equation (8), the final form of $h(t, p_2, T', \lambda)$ becomes

$$h(t, p_2, T', \lambda) = \frac{p_2 - g(T', p_1)}{1 - e^{-\lambda(T'-1)}} (1 - e^{-\lambda(t-1)}), \quad \lambda > 0. \quad (12)$$

2.2.4. *Proposed search weight.* Based on the above discussion, the proposed search weight is defined as

$$q(t, p_1, p_2, T', \lambda) = \begin{cases} p_1 e^{\frac{1}{2}} \left(\frac{\log(t + e - 1)}{t + e - 1} \right)^{\frac{1}{2}} + \frac{p_2 - g(T', p_1)}{1 - e^{-\lambda(T'-1)}} (1 - e^{-\lambda(t-1)}), & 1 \leq t < T', \\ p_2, & T' \leq t \leq T \end{cases}, \quad \lambda > 0, \quad (13)$$

where $\beta = q(t, p_1, p_2, T', \lambda)$ is substituted into Equation (2) to perform ENN-SO.

Here, the parameter λ controls the curvature of the function. It is desirable for the proposed function $q(t, p_1, p_2, T', \lambda)$ to resemble the shape of $g(t, p_1)$, which does not allow

direct control over the weight at $t = T$. To achieve this, λ is determined by minimizing the squared error between q and g from $t = 1$ to T' :

$$\lambda^* = \underset{\lambda}{\operatorname{argmin}} \sum_{t=1}^{T'} (q(t, p_1, p_2, T', \lambda) - g(t, p_1))^2. \quad (14)$$

However, to prevent q from having undesirable extrema, we impose one of the following conditions before solving the optimization problem in Equation (14):

$$\begin{aligned} \text{Cond. 1: } \frac{dh(t)}{dt} < 0, \quad \text{Cond. 2: } \left(\frac{dh(t)}{dt} > 0 \right) \wedge \left(\left| \frac{dh(t)}{dt} \right| < \left| \frac{dg(t)}{dt} \right| \right), \\ \forall t \in \{1, \dots, T'\}. \end{aligned} \quad (15)$$

Condition 1 ensures that q is a monotonically decreasing function. Specifically, when h is also monotonically decreasing, and since g itself is monotonically decreasing, their sum $q = g + h$ inherits this monotonicity and thus avoids having any local extrema. In contrast, condition 2 is more intricate and requires detailed explanation. The term $|dg(t)/dt|$ for the monotonically decreasing function g represents the “downward force” of g at time t , i.e., how rapidly it is decreasing. Similarly, when $|dh(t)/dt| > 0$, the function h is monotonically increasing, and the derivative magnitude corresponds to the “upward force” of h at step t . If the upward force of h exceeds the downward force of g , i.e., $|dh(t)/dt| > |dg(t)/dt|$, then $q = g + h$ may exhibit a local extremum. To avoid this, we impose the condition $|dh(t)/dt| < |dg(t)/dt|$. Since these conditions must be satisfied not just locally, but across the entire domain of interest, they must hold $\forall t \in \{1, \dots, T'\}$.

3. Experiment.

3.1. Purpose and outline of the experiment. An experiment was conducted to compare the existing method (constant search weight) and the proposed method (logarithmic search weight) in terms of search performance. In this experiment, benchmark functions – Beale, Goldstein-Price, Griewank, Himmelblau, Shubert, and Six-Hump Camel [21, 22] – were used as black-box functions, and the global optima were searched for. Each search space is $\Psi = [-5, 5]^2$ (Beale), $\Psi = [-2, 2]^2$ (Goldstein-Price), $\Psi = [-5, 5]^2$ (Griewank), $\Psi = [-6, 6]^2$ (Himmelblau), $\Psi = [-2, 2]^2$ (Shubert), $\Psi = [-2, 2]^2$ (Six Hump Camel). While benchmark functions are designed for minimization problems, Equation (2) is formulated for maximization. Therefore, by multiplying benchmark functions by -1 , we transformed them into functions suitable for maximum value search. The objective function was normalized with respect to $f(\mathbf{x}) \in [-1, 1]$. The specific parameters used in Algorithm 1 are as follows: the number of initial observation dataset $N = 5$, the number of repetitions for surrogate $T = 100$, the number of individuals NN $K = 30$, training iterations $M = 1000$, NN learning rate $\eta = 0.01$. As described in Algorithm 1 line 5, the initial parameters of NN $\theta_k^{(0)}$ are randomly generated. The initial observation dataset D_0 is determined randomly, which may lead to variations in performance due to randomness. Since a single trial is insufficient to accurately compare the performance between the existing and proposed methods, we performed 20 experiments in total by generating the initial parameters of NN and initial observation dataset $\theta_k^{(0)}$ using different random seeds. Initial observation dataset included in dataset D_0 were randomly selected from the subset of search space Ψ for which the value of $f(\mathbf{x})$ was below a specified threshold. For the existing method (constant search weight), we used β values of 0 and 3, and for the proposed method, we set $p_1 = 3$, $p_2 = 0$, and $T' \in \{25, 50, 100\}$.

3.2. Results and discussion. The results obtained from the experiment described in Section 3.1 are shown in Table 1 and Figure 2. The values in Table 1 represent the evaluation scores at step t , and since this is a maximum value search task, higher values indicate better performance. As shown in Figure 2, for the Goldstein-Price, Griewank, Himmelblau, Shubert, and Six Hump Camel functions, $\beta = 0$ yielded high scores in the early step ($t = 10$), but its performance declined in the middle ($t = 40$) and late ($t = 80$) steps. This is because the influence of the exploration term $V(\mathbf{x})$ is absent. While the case of $\beta = 0$ is effective when the number of search iterations is small, it yields lower scores than other methods as t increases. Therefore, in the case of $\beta = 0$, it cannot be considered an effective method when a moderate number of search iterations can be performed. To resolve this issue related to the existing method, $\beta = 3$ is set as the search weight to amplify the effect of the exploration term. Nevertheless, because $\beta = 3$ consistently shows a strong bias toward exploration, its scores in the middle steps of the search are inferior to those

TABLE 1. Evaluation scores at step t for both the existing and proposed methods. The bold figures represent the highest score. Each value corresponds to the mean over 20 runs with different random seeds.

Benchmark	Method	Steps			
		$t = 10$	$t = 20$	$t = 40$	$t = 80$
Beale [21]	Existing ($\beta = 0$)	0.443	0.721	0.828	0.873
	Existing ($\beta = 3$)	0.450	0.538	0.700	1.000
	Ours: $\beta = q(t, 3, 0, 25, \lambda^*)$	0.506	0.661	0.967	0.975
	Ours: $\beta = q(t, 3, 0, 50, \lambda^*)$	0.495	0.741	0.973	0.975
	Ours: $\beta = q(t, 3, 0, 100, \lambda^*)$	0.479	0.723	0.993	1.000
Goldstein-Price [21]	Existing ($\beta = 0$)	0.309	0.583	0.938	0.938
	Existing ($\beta = 3$)	0.062	0.349	1.000	1.000
	Ours: $\beta = q(t, 3, 0, 25, \lambda^*)$	0.083	0.673	1.000	1.000
	Ours: $\beta = q(t, 3, 0, 50, \lambda^*)$	0.107	0.526	0.965	1.000
	Ours: $\beta = q(t, 3, 0, 100, \lambda^*)$	0.141	0.580	0.938	0.966
Griewank [21]	Existing ($\beta = 0$)	0.841	0.922	0.935	0.953
	Existing ($\beta = 3$)	0.637	0.778	0.865	0.984
	Ours: $\beta = q(t, 3, 0, 25, \lambda^*)$	0.716	0.914	0.966	0.966
	Ours: $\beta = q(t, 3, 0, 50, \lambda^*)$	0.736	0.929	0.974	0.974
	Ours: $\beta = q(t, 3, 0, 100, \lambda^*)$	0.758	0.883	0.991	0.992
Himmelblau [22]	Existing ($\beta = 0$)	-0.059	0.494	0.789	0.856
	Existing ($\beta = 3$)	-0.404	-0.113	0.180	0.898
	Ours: $\beta = q(t, 3, 0, 25, \lambda^*)$	-0.295	0.272	0.868	0.898
	Ours: $\beta = q(t, 3, 0, 50, \lambda^*)$	-0.233	0.285	0.832	0.847
	Ours: $\beta = q(t, 3, 0, 100, \lambda^*)$	-0.266	0.238	0.802	0.847
Shubert [21]	Existing ($\beta = 0$)	0.458	0.566	0.623	0.667
	Existing ($\beta = 3$)	0.278	0.442	0.618	0.909
	Ours: $\beta = q(t, 3, 0, 25, \lambda^*)$	0.276	0.425	0.683	0.721
	Ours: $\beta = q(t, 3, 0, 50, \lambda^*)$	0.299	0.452	0.635	0.668
	Ours: $\beta = q(t, 3, 0, 100, \lambda^*)$	0.351	0.489	0.640	0.673
Six Hump Camel [21]	Existing ($\beta = 0$)	0.576	0.968	0.984	1.000
	Existing ($\beta = 3$)	0.051	0.347	0.839	1.000
	Ours: $\beta = q(t, 3, 0, 25, \lambda^*)$	0.197	0.925	0.984	1.000
	Ours: $\beta = q(t, 3, 0, 50, \lambda^*)$	0.269	0.984	0.984	1.000
	Ours: $\beta = q(t, 3, 0, 100, \lambda^*)$	0.282	0.869	0.984	1.000

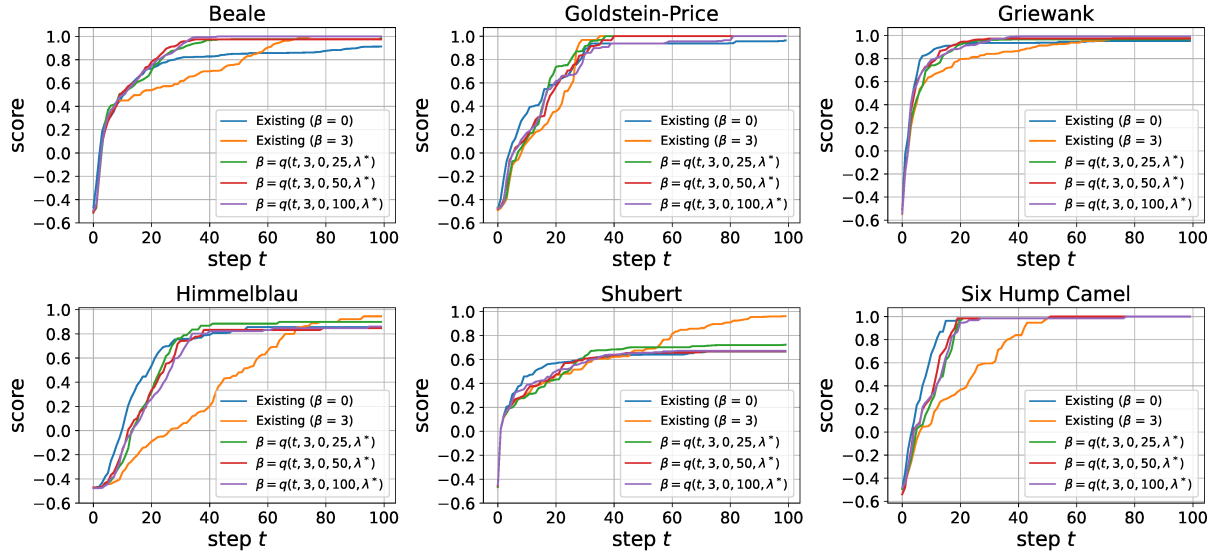


FIGURE 2. (color online) Performances of the existing and proposed methods on the 6 benchmark functions. The average of 20 trials on different random seeds.

of other methods (e.g., see the values of Himmelblau and Beale in the case $(\beta, t) = (3, 40)$ in Table 1). To solve these problems, we adopt a logarithmic search weight approach as proposed in our method. As shown in Figure 2 at $t = 20$ and $t = 40$, the proposed method consistently outperformed other methods across nearly all benchmark functions. This is because adopting logarithmic weights results in a stronger tendency toward exploration $V(\mathbf{x})$ in the early steps and a stronger tendency toward exploitation $E(\mathbf{x})$ in the later steps. Since this behavior aligns well with an effective strategy for solving exploration problems, it is considered that the proposed method achieved better results than the existing method.

4. Conclusion. SO based on UCB is defined as $E + \beta V^{1/2}$, where the parameter β controls the strength of exploration. When the value of β is constant and small, the influence of the exploration term V on the search space is not considered; when it is constant and large, the influence of the exploitation term E is neglected. This poses a problem. In the existing methods [16, 17], logarithmic search weights exhibit three problems: 1) the initial weight at $t = 1$ cannot be freely chosen; 2) the maximum value of the weight is fixed; and 3) the final weight cannot be freely set. In this research, we proposed a new logarithmic search weight approach (see Equation (13)) to address these issues, and demonstrated favorable results through experimental evaluations using benchmark functions. However, the experiments were conducted on a 2-dimensional search space, and the benchmark functions may be relatively simple or monotonic compared to real-world black-box functions. We used NN as the surrogate model, but alternatives such as Gaussian process regression [23] and radial basis function [24] also exist. In the future, we plan to propose methods to address these issues and conduct performance evaluation experiments.

Acknowledgements. This work was supported by JSPS Grants-in-Aid for Scientific Research (C) (Grant Nos.: 23K11310 and 21K04535).

REFERENCES

- [1] V. Raul and L. Leifsson, Surrogate-based aerodynamic shape optimization for delaying airfoil dynamic stall using Kriging regression and infill criteria, *Aerosp. Sci. Technol.*, vol.111, Article No.106555, 2021.
- [2] F. Ren, L. Ward et al., Accelerated discovery of metallic glasses through iteration of machine learning and high-throughput experiments, *Sci. Adv.*, vol.4, no.4, 2018.
- [3] B. P. MacLeod, F. G. L. Parlane et al., Self-driving laboratory for accelerated discovery of thin-film materials, *Sci. Adv.*, vol.6, no.20, Article No.eaaz8867, 2020.
- [4] B. Burger, P. M. Maffettone et al., A mobile robotic chemist, *Nature*, vol.583, no.7815, pp.237-241, 2020.
- [5] L. Sun, H. Gao, S. Pan and J. Wang, Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data, *Comput. Methods Appl. Mech. Eng.*, vol.361, no.23, Article No.112732, 2020.
- [6] R. Kamijo, A. Ishii, S. Coppieters and A. Yamanaka, Bayesian texture optimization using deep neural network-based numerical material test, *Int. J. Mech. Sci.*, vol.223, Article No.107285, 2022.
- [7] Y. Omae, Y. Kakimoto, M. Sasaki and M. Mori, EVI-GPBO: Estimated variance integration-based Gaussian process Bayesian optimization, *IEEE Access*, vol.13, no.12, pp.26208-26224, 2025.
- [8] W. Jouini, D. Ernst, C. Moy and J. Palicot, Upper confidence bound based decision making strategies and dynamic spectrum access, *2010 Proc. of IEEE Int. Conf. Commun. (ICC)*, pp.1-5, 2010.
- [9] A. Carpentier, A. Lazaric, M. Ghavamzadeh, R. Munos and P. Auer, Upper-confidence-bound algorithms for active learning in multi-armed bandits, *Algorithmic Learn. Theory*, pp.189-203, 2011.
- [10] T. Imagawa and T. Kaneko, Estimating the maximum expected value through upper confidence bound of likelihood, *2017 Conf. Technol. Appl. Artif. Intell.*, pp.202-207, 2017.
- [11] Z. Liu, T. Kawabe, T. Nishi, S. Ito and T. Fujiwara, Surrogate-assisted multi-objective optimization for simultaneous three-dimensional packing and motion planning problems using the sequence-triple representation, *Appl. Artif. Intell.*, vol.38, no.1, Article No.2398895, 2024.
- [12] Y. Chen, L. Wang and H. Huang, An effective surrogate model assisted algorithm for multi-objective optimization: Application to wind farm layout design, *Front. Energy Res.*, vol.11, Article No.1239332, 2023.
- [13] C. Passmore, K. E. Wu, J. R. Howse, G. Panoutsos and S. J. Ebbens, Surrogate-assisted optimization of roll-to-roll slot die coating, *Sci. Rep.*, vol.15, Article No.11279, 2025.
- [14] D. D. Cox and S. John, A statistical method for global optimization, *Proc. of IEEE Int. Conf. Syst., Man, Cybern.*, pp.1241-1246, 1992.
- [15] H. Ohno, Empirical studies of Gaussian process based Bayesian optimization using evolutionary computation for materials informatics, *Expert Syst. Appl.*, vol.96, pp.25-48, 2018.
- [16] D. Mochihashi and O. Shigeyuki, *Gaussian Process and Machine Learning*, Machine Learning Professional Series, Kodansha, Tokyo, Japan, 2019.
- [17] K. Hirota, S. Wakitani and T. Yamamoto, Study on an optimal design method for control systems based on Bayesian optimization, *Proc. of Int. Conf. Artif. Life Robot*, vol.25, no.12, pp.133-136, 2021.
- [18] K. Sasaki and Y. Omae, Surrogate optimization with ensemble neural networks: The effect of hidden layer units on search performance, *IEICE General Conference*, 2025.
- [19] E. Schulz, M. Speekenbrink and A. Krause, A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions, *J. Math. Psychol.*, vol.85, pp.1-16, 2018.
- [20] R. Roussel et al., Bayesian optimization algorithms for accelerator physics, *arXiv Preprint*, arXiv: 2312.05667, 2023.
- [21] S. Surjanovic and D. Bingham, *Virtual Library of Simulation Experiments: Test Functions and Datasets*, <https://www.sfu.ca/~ssurjano/optimization.html>, Accessed on April 30, 2026.
- [22] M. Jamil and X. Yang, A literature survey of benchmark functions for global optimisation problems, *Int. J. Math. Model. Numer. Optim.*, vol.4, no.2, pp.150-194, 2013.
- [23] J. Snoek, H. Larochelle and R. P. Adams, Practical Bayesian optimization of machine learning algorithms, *Adv. Neural Inf. Process. Syst.*, vol.25, 2012.
- [24] R. G. Regis, Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points, *Eng. Optim.*, vol.46, no.2, pp.218-243, 2014.

Author Biography



Kyoya Sasaki received the bachelor's degree from the Department of Industrial Engineering and Management, College of Industrial Technology, Nihon University, Japan, in 2025. He is currently pursuing the master's degree at the College of Industrial Technology, Nihon University. He received a first-class scholarship from Nihon University in 2023. His research interests include machine learning-based optimization, and data-driven decision making.



Yuto Omae received the Ph.D. degree in Engineering from Nagaoka University of Technology, Japan, in 2016. He served as an Assistant Professor at the National Institute of Technology, Tokyo College, Japan, from April 2017 to March 2019, and as a Visiting Researcher at Metropolia University of Applied Sciences, Finland, from February 2024 to March 2025. He is currently a Senior Assistant Professor with the Department of Industrial Engineering and Management, College of Industrial Technology, Nihon University, Japan. He also serves as the Vice Director of the Artificial Intelligence Research Center, College of Industrial Technology, Nihon University. In addition, he is an Associate Editor for the IEICE Transactions on Information and Systems. His research interests include the theoretical aspects of machine learning. He is a member of the Institute of Electronics, Information and Communication Engineers (IEICE) and the Institute of Electrical and Electronics Engineers (IEEE).



Jun Toyotani received the Ph.D. degree from the Graduate School of Industrial Technology, Nihon University, Japan, in 1998. He is a Professor with the Department of Industrial Engineering and Management, College of Industrial Technology, Nihon University. His research interests include data science, digital marketing analysis, and research on streamlining operations using AI. He is a member of the Japan Society for Engineering Education, the Information Processing Society of Japan, and the Japan Society of Directories.