Impact of exploration-exploitation trade-off on UCB-based Bayesian Optimization

Chao Bian, Xiaofang Wang, Changjun Liu, Xinyu Xie, Haitao Liu*
School of Energy and Power Engineering, Dalian University of Technology, Dalian 116024, China

*Corresponding Author, Email: htliu@dlut.edu.cn

Abstract. As an efficient global optimization method to deal with expensive black-box functions, Bayesian optimization (BO) has great potential in the field of turbomachinery component design. It can relieve the complexity, long cycle, and excessive dependence on experience of the design process. A core of Bayesian optimization is acquisition functions which aim to guide the search to efficiently find the optimum. An excellent acquisition function can greatly improve the performance of the algorithm. The well-known acquisition function, upper confidence bound (UCB), in Bayesian optimization achieves the balance between local exploitation and global exploration through an explicit trade-off coefficient. The trade-off coefficient is a key to the quality of UCB-based Bayesian optimization. Currently, this coefficient is either manually specified according to some criterion or sampled from a specific distribution. Due to the lack of a comparative study on existing trade-off strategies, this paper attempts to comprehensively analyze and highlight the impact of various trade-off strategies on the performance of UCB-based BO algorithm through eleven numerical examples and two turbomachinery design cases. The comparative results indicate that the proper trade-off is key for the optimization quality, and the strategy of looping through a trade-off set shows superiority in our numerical experiments.

1. Introduction
In the field of turbomachinery design, such as the design of compressor, steam turbine, and turbine, there is often no explicit expression between design variables and output results. That is, the simulation model is a black-box function with only input and output. Moreover, the cost of single simulation calculation may be very expensive. In view of these problems, traditional optimization methods, such as gradient based optimization method, cannot meet the needs of efficient design optimization.

To this end, surrogate based optimization has been proposed and studied for simulation-based optimization. For example, based on the combination of artificial neural network and genetic algorithm, the optimization of compressor return channel blades and low flowrate coefficient centrifugal compressor impellers have significantly improved the aerodynamic performance of centrifugal compressors [1]-[2]. The impeller disk is optimized based on the kriging model [3]. After optimization, the mass of the impeller is reduced by 15%, and the axial deformation of the friction end surface is reduced by 29%. Based on the CORS-BRF constrained optimization algorithm [4], the impeller geometry is optimized and the stress on the impeller is reduced. Using response surface methodology to optimize the centrifugal fan volute, the fan efficiency increased by 4.6% [5].

Particularly, this paper focuses on Bayesian optimization for simulation-based optimization. Bayesian optimization is an efficient surrogate-based global optimization algorithm, which is composed of two core parts: probabilistic surrogate model and acquisition function [6]. The probabilistic
surrogate model can be used to approximate the complex problems, and the acquisition function plays the role of selecting the next most potential evaluation point. When the objective function of optimization is expensive black-box function or lack of gradient information, Bayesian optimization has obvious advantages in comparison to other optimization algorithms. It can use a small amount of data from previous samples to generate a statistical model of the system. This model is then used to suggest the next input through an acquisition function. By sequentially selecting new sample points to update the model, the optimum of the complex problem is finally obtained. Bayesian optimization has been successfully applied in machine learning [7] and design optimization [8]. It also has been applied and studied in the field of turbomachinery. For example, the parallel constrained Bayesian optimization [9] has been proposed to optimize the geometric parameters of the centrifugal slurry pump impeller and effectively reduce the wear rate of the impeller. Besides, to obtain maximum efficiency of the pin fin array in the developing region of the flow, Bayesian optimization is used to change the shape of the cooling channel in the trailing edge of the blade [10].

As a core component of Bayesian optimization, the acquisition function usually determines the optimization performance of the algorithm. The acquisition function suggests the next input point to update the surrogate model. A reasonable acquisition function needs to fully consider the trade-off of local exploitation and global exploration, in order to ensure that the algorithm has a fast convergence rate and the ability to jump out of the local optimum. Along this line, there are many species of acquisition functions, which consider the balance between exploitation and exploration from different perspectives. This paper only focuses on Bayesian optimization based on upper confidence bound (UCB) method. The UCB usually consists of posterior mean and posterior standard deviation provided by the probabilistic surrogate model. The exploitation and exploration can be balanced by multiplying the posterior standard deviation by a trade-off coefficient.

The trade-off coefficient, which often has a great impact on the performance of the Bayesian optimization algorithm, could be specified by the user. In practical application, people usually set this parameter as a fixed constant for convenience [11]. Subsequently, the Gaussian process upper confidence bound (GP-UCB) was proposed, which provides strong theoretical guarantees on the overall convergence rate while balancing local and global [12]. As an improvement of GP-UCB, the randomized Gaussian Process Upper Confidence Bound (RGP-UCB) is proposed to retain the theoretical guarantee of convergence speed and improve the local and global trade-off effect through sampling from gamma distribution [13].

Due to the lack of a comparative study on existing trade-off strategies, this paper attempts to comprehensively analyse and highlight the impact of various trade-off strategies on the performance of UCB-based BO algorithm through eleven numerical examples and two turbomachinery design cases.

The rest of the paper is organized as follow. Section 2 first gives a brief introduction of the UCB-based Bayesian optimization as well as Gaussian process. Thereafter, section 3 reviews existing trade-off strategies for Bayesian optimization, followed by the numerical experiments on benchmarks and turbomachinery cases in sections 4 and 5, respectively. Finally, section 6 offers the conclusions.

2. Bayesian optimization revisited

In this section, we provide a brief review of Bayesian optimization and Gaussian process, with an emphasis on the selection of reasonable trade-off coefficients among various UCB-based Bayesian optimization methods.

2.1. Bayesian Optimization

Bayesian optimization is an efficient global optimization algorithm for solving expensive black-box functions. The goal of the algorithm is to find the global optimal solution of (1) in a bounded search space.

$$x^* = \arg \max_{x \in \mathcal{X}} f(x) \quad (1)$$

where $\mathcal{X}$ is often a compact subset of $\mathbb{R}^d$ and represents the design space of interest, $x$ is a $d$-dimensional decision vector, and $f$ is the objective function, which is usually an unknown black box function without
explicit expression, but the corresponding response value can be obtained at input \( x \) in the domain. This corresponding response value is a noise-corrupted output \( y = f(x) + \epsilon_{\text{noise}} \), where \( \epsilon_{\text{noise}} \sim \mathcal{N}(0, \sigma^2) \).

This type of problem is difficult to solve using the classic gradient ascent methods. Generally speaking, Bayesian optimization is a sequence model-based method to solve such problems. We specify a prior belief for the possible objective function, and then combine the data we already have to build a probabilistic surrogate model. Thereafter, we use the Bayesian posterior information of the model, which represents our updated beliefs about the possible objective function being optimized, to establish an appropriate acquisition function, \( \alpha(x) \). The acquisition function actively selects the most “potential” evaluation points for evaluation, avoiding unnecessary sampling. The most commonly used probabilistic surrogate model in Bayesian optimization is Gaussian process (GP), which is the model used in our work. It is notable that other models such as random forest and deep neural network can be used, once they could offer probabilistic predictions. The most popular acquisition functions in BO are improved probability (PI) [14], expected improvement (EI) [15], and upper bound confidence bound (UCB) [11].

2.2. Gaussian Process

Gaussian process (GP) is a nonparametric model which is the most common statistical model used in Bayesian optimization. It consists of a mean function, \( m: \mathcal{X} \rightarrow \mathbb{R} \) and a semidefinite covariance function, \( k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R} \) (an effective covariance function must be semi positive definite [16]).

\[
f(x) \sim \text{GP}(m(x), k(x, x'))
\]

where the mean function \( m(x) = \mathbb{E}[f(x)] \), for simplicity, is usually set to \( m(x) = 0 \), the covariance function \( k(x, x') = \mathbb{E}[f(x) - m(x)(f(x') - m(x'))] \) quantifies the correlation between two data points. In practical application, the most popular covariance function is the squared exponential kernel [17].

\[
k_{SE}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2} \sum_{t=1}^{T} \left(\frac{x_{it} - x_{jt}}{l_t^2}\right)^2\right)
\]

where \( \sigma_f^2 \) is an output scale, and \( l_t \) is an input scale along the \( t \)th dimension.

Gaussian process is a set of random variables and any finite random variables satisfy a joint Gaussian distribution. Suppose that the latent function values \( f = \{f(x_1), f(x_2), \ldots, f(x_t)\} \) follow a zero-mean GP prior:

\[
p(f|X, \theta) = \mathcal{N}(0, \Sigma)
\]

where \( X = \{x_1, x_2, \ldots, x_t\} \) is the training set, \( \Sigma \) is the covariance matrix composed of \( k(x, x') \) (\( \Sigma_{ij} = k(x_i, x_j) \)), and finally \( \theta \) comprises the hyperparameters.

When there is observation noise, i.e., \( y = f(x) + \epsilon_{\text{noise}} \), where \( \epsilon_{\text{noise}} \sim \mathcal{N}(0, \sigma^2) \), the likelihood distribution is obtained:

\[
p(y|f) = \mathcal{N}(f, \sigma^2 I)
\]

The corresponding marginal likelihood distribution of observations \( y = \{y_1, y_2, \ldots, y_t\} \) is:

\[
p(y|X, \theta) = \int p(y|f)p(f|X, \theta)df = \mathcal{N}(0, \Sigma + \sigma^2 I)
\]

Usually, we optimize the hyperparameters \( \theta \) by maximizing the marginal likelihood distribution. According to the properties of Gaussian process, the following joint distribution can be obtained:

\[
[Y, f] \sim \mathcal{N}\left(0, \left[\begin{array}{cc}
\Sigma + \sigma^2 I & K_* \\
K_* & K_*^T
\end{array}\right]\right)
\]
where \( f_* \) is the prediction function values, \( x_* \) is the prediction input, \( k_* = \{ k(x_1, x_*), k(x_2, x_*), \ldots, k(x_t, x_*) \} \), \( k_{**} = k(x_*, x_*) \). From this joint distribution, we can get the following prediction distribution:

\[
p(f_* | X, y, \theta) = \mathcal{N}(\bar{f}_*, \text{cov}(f_*))
\]

\[
\bar{f}_* = k_* (\Sigma + \sigma^2 I)^{-1} y
\]

\[
\text{cov}(f_* ) = k_{**} - k_* (\Sigma + \sigma^2 I)^{-1} k_*^T
\]

where \( \bar{f}_* \) is the prediction mean, \( \text{cov}(f_* ) \) is the prediction covariance.

### 2.3. Acquisition Function

In the framework of BO, we first use Gaussian process to construct the probability surrogate model; next, we combine the existing data set \( D \) and the surrogate model to create an acquisition function, \( \alpha(x) \); finally, we select the next evaluation point \( x_{t+1} \) by maximizing the acquisition function.

\[
x_{t+1} = \arg\max_{x \in \mathcal{X}} \alpha(x; D)
\]

In the design of acquisition function, we need to balance local exploitation and global exploration reasonably. Local exploitation means that the sampling process pays attention to the improved locations, which have high probability of containing the global optimum. On the contrary, global exploration means that the sampling process tends to make the samples far away from each other in order to explore the unknown region which may contain the optimum. Excessive exploitation will cause the acquisition function to fall into the local optimum, while excessive exploration will slow the convergence speed. Hence, it is important to make a reasonable trade-off between exploitation and exploration in order to improve the optimization quality. Our work only focuses on Bayesian Optimization based on UCB. In the UCB method, a trade-off coefficient is used to balance the local and global. We are concerned about the influence of the trade-off coefficient set by different methods on Bayesian Optimization. Next, we will introduce different trade-off strategies for UCB.

### 3. Trade-off strategies for UCB-based BO

#### 3.1. Bayesian Optimization

In the Sequential Design for Optimization (SDO) algorithm proposed by Cox and John [11], they select the prediction site which has the smallest value of lower confidence bound (LCB) as the next evaluation point:

\[
\alpha_{\text{LCB}}(x; D) = -(\mu_t(x) - k \sigma_t(x))
\]

where \( k \geq 0 \). When we are concerned with the maximum value of the objective function, we usually use the upper confidence bound (UCB) strategy:

\[
\alpha_{\text{UCB}}(x; D) = \mu_t(x) + k \sigma_t(x)
\]

which is the acquisition function commonly used in Bayesian optimization algorithm. For any input \( x \), it has a mean \( \mu_t(x) \) and a standard deviation \( \sigma_t(x) \) (indicating uncertainty). We believe that \( k \sigma_t(x) \) is a relatively reliable bound. Hence, \( f(x) \) has a relatively high probability to reach \( \mu_t(x) + k \sigma_t(x) \). The process of maximizing the acquisition function is to maximize our belief. In the process of maximizing the acquisition function, \( \mu_t(x) \) makes the collection function tend to exploit the improved area, which belongs to exploitation item. \( \sigma_t(x) \) makes the collection function choose the new point that is that is usually indicated with large standard deviation, which belongs to the exploration item. As a local and global trade-off coefficient, \( k \) is usually set as a constant. In [11], they set \( k = 1.96 \) to indicate the meaning of the “confidence factor”. In our work, we set \( k = 1.96 \). As a comparison, we also set \( k = 10 \), which is a trade-off coefficient value favorable for exploration.
3.2. Gaussian process upper confidence bound

Srinivas et al. [12] proposed the Gaussian process upper confidence bound (GP-UCB) method based on cumulative regret boundary. The convergence of the algorithm is proved by strict theory.

\[ a_{\text{GP-UCB}}(x; D) = \mu_t(x) + \sqrt{\beta_t \sigma_t(x)} \quad (14) \]

GP-UCB regards the Bayesian optimization as a multi-armed bandit and obtains instantaneous regret function:

\[ r(x) = f(x_*) - f(x) \quad (15) \]

The goal of optimization in the framework is equivalent to:

\[ \min \sum_{t=1}^{T} r(x_t) = \max \sum_{t=1}^{T} f(x_t) \quad (16) \]

where \( T \) is the number of optimization iterations. It is proved that if the kernel satisfies the following formula:

\[ P\{\sup_{x \in \mathcal{X}}|\partial f / \partial x| > L\} \leq a e^{-(L/b)^2} \quad (17) \]

where \( t = 1, \ldots, T \), the constants \( a, b > 0, \mathcal{X} \subseteq [0, m]^d, d \in \mathbb{N}, \) and \( m > 0 \). Given \( \delta \in (0,1) \) and

\[ \beta_t = 2 \log \left( \frac{t^2 \pi^2}{3 \delta} \right) + 2d \log (t^2 \delta^{1/2} m \sqrt{\log(4da/\delta)}) \quad (18) \]

Then with high probability of \( 1 - \delta \), the algorithm is no regret, i.e. \( \lim_{T \to \infty} R_T / T = 0 \), where \( R_T \) is the cumulative regret:

\[ R_T = \sum_{t=1}^{T} f(x_*) - f(x_t) \quad (19) \]

This implies a lower-bound on the convergence rate for the optimization problem, which guarantees the convergence of the algorithm.

3.3. Randomized Gaussian process upper confidence bound

Although the selection of \( \beta_t \) in GP-UCB provides a regret bound, it is considered that the value of \( \beta_t \) is often greater than the required value, which will lead to the algorithm taking too much consideration of global exploration [13]. Hence, a method called Randomized Gaussian Process Upper Confidence Bound (RGP-UCB) has been proposed. It can keep the regret bound and select a smaller trade-off coefficient. Consequently, the algorithm can better balance local exploitation and global exploration as:

\[ a_{\text{RGP-UCB}}(x; D) = \mu_t(x) + \sqrt{\tilde{\beta}_t \sigma_t(x)} \quad (20) \]

where \( \tilde{\beta}_t \) is a random number following the gamma distribution \( \Gamma(\gamma_t, \theta) \), with \( \theta > 0 \) being a constant used to adjust the size of \( \tilde{\beta}_t \), but setting a proper \( \theta \) requires certain prior knowledge. In order to ensure that the algorithm maintain a regret bound, \( \gamma_t \) has been proven to follow:

\[ \gamma_t = \frac{\log \left( \frac{1}{\sqrt{2\pi}} (t^2 + 1) \right)}{\log(1+\theta/2)} \quad (21) \]

Through the use of \( \Gamma(\gamma_t, \theta) \), the trade-off coefficient \( \tilde{\beta}_t \) selected in RGP-UCB is smaller than \( \beta_t \) in the original GP-UCB while still maintaining convergence guarantees.
4. Numerical experiments

In this section, numerical examples and engineering cases are used to investigate the performance of UCB-based Bayesian optimization algorithms using various trade-off strategies.

In each case, the experiment was repeated 10 times with \(10d\) initial points [18] generated by the TPLHD (translational propagation Latin hypercube design) sample scheme [19]. The number of iterations required for each algorithm to reach a relative error of 0.01% (the absolute error is used when the optimal value is 0) is recorded. And the allowed maximal number of function evaluations is set to 400 in our experiments. The methods involved in the comparison are:

- Upper Confidence Bound with \(k = 1.96\).
- Upper Confidence Bound with \(k = 10\).
- Upper Confidence Bound with \(k \in [0, 1, 100]\).
- Standard Gaussian process upper confidence bound (GP-UCB).
- Standard randomized Gaussian process upper confidence bound (RGP-UCB).

In the above competitors, the UCB using a small, fixed trade-off coefficient \(k = 1.96\) favors local exploitation; while the UCB using a large \(k = 10\) prefers global exploration; the UCB using a trade-off set \(k \in [0, 1, 100]\) performs a search loop using various \(k\) values, thus resulting in a comprehensive consideration of local exploitation and global exploration; finally, the last two methods adopt an adaptive \(k\) during the optimization.

4.1. Test functions

In this section, the performance of the algorithms is verified numerically on 11 benchmarks with different characteristics. The problem dimensions, the abbreviation used, the bounds of design space and the known global optimum of each test function are summarized in Table 1. Additionally, Figure 1 depicts the convergence histories of these algorithms on eleven benchmarks, respectively. The mathematical formulas of these benchmark functions can be found on the web page\(^1\).

| Test functions      | Abbreviation | Dimension | Domain             | Global minimum |
|---------------------|--------------|-----------|--------------------|----------------|
| Branin              | BA           | 2         | [-5,10] x [0,15]   | 0.3978         |
| Bohachevsky         | BO           | 2         | [-2,2]²           | 0.0000         |
| easom               | EA           | 2         | [-10,10]²          | -1.0000        |
| Schwef              | SCH          | 2         | [-400,500]²       | 0.0000         |
| Shubert             | SHU          | 2         | [-2,2]²           | -186.7309      |
| Six-hump camel      | SC6          | 2         | [-3,2]²           | -1.0316        |
| Hartman3            | H3           | 3         | [0,1]³            | -3.8627        |
| Rosenbrock          | RO           | 3         | [-2,2]³           | 0.0000         |
| Levy                | LE           | 3         | [-8,8]³           | 0.0000         |
| Shekel 5            | SH5          | 4         | [0,5]⁴           | -10.1532       |
| Sphere              | SP           | 5         | [-3,5]⁵           | 0.0000         |

4.2. Results and discussions

Table 2 showcases the number of functional evaluations [mean and standard deviation (SD)] required to achieve the specified relative error (0.01%) for the UCB-based BO using different trade-off strategies, with the best results marked in bold.

\(^1\) http://www.sfu.ca/~ssurjano/optimization.html
Figure 1. Performance comparison of UCB-based Bayesian optimization using different trade-off strategies on eleven benchmarks.
Table 2. Number of function evaluations (mean and SD) required to achieve a specified relative error (0.01%).

| Test functions | UCB \((k = 1.96)\) | UCB \((k = 10)\) | UCB \((k ∈ [0,1,10])\) | GP-UCB | RGP-UCB |
|----------------|-------------------|-------------------|--------------------------|--------|--------|
| BA             | 38.3(1.3)         | 44.2(1.4)         | 35.8(2.7)                | 41.4(0.8) | 39.6(1.5) |
| BO             | 71.3(9.8)         | 318(19.1)         | 51.9(1.5)                | 292.4(11.2) | 190.3(70.4) |
| EA             | 125.1(57.3) \(^a\) | 227.0(46.7) \(^b\) | 113.2(73.6)               | 179.6(47.1) \(^c\) | 142.1(78.7) \(^d\) |
| SCH            | >400              | >400              | 172.8(13.4)              | 258.6(25.4) \(^c\) | 178.9(4.5) |
| SHU            | >400              | 210.5(1.2)        | 195.5(52.2)              | 154.1(12.4) | 94.2(6.2) |
| SC6            | 44.0(0.8)         | 63.7(1.9)         | 49.0(2.3)                | 58.9(1.0) | 55(2.1) |
| H3             | 42.6(1.6)         | 81.5(3.4)         | 43(0.0)                  | 64.6(4.0) | 54.5(3.2) |
| RO             | 86.9(6.1)         | 124.9(14.2)       | 98.9(9.8)                | 120.2(13.3) | 109.3(16.5) |
| LE             | 367.0(41.0) \(^a\) | >400              | 144.4(34.3) \(^d\)       | >400   | >400   |
| SH5            | 86.8(8.0)         | >400              | 103.7(5.4)               | >400   | 183.1(21.3) \(^d\) |
| SP             | 70.8(0.6)         | 141.9(1.1)        | 69.9(2.5)                | 115.4(1.6) | 89.9(1.4) |

\(^a\) The algorithm only succeeded to achieve the required relative error within the limited computing resource in eight out of ten runs for EA, and in two out of ten runs for LE. Hence, the data is the average of successful runs.
\(^b\) The algorithm only succeeded in two out of ten runs for EA.
\(^c\) The algorithm only succeeded in five out of ten runs for EA, and in three out of ten runs for SCH.
\(^d\) The algorithm only succeeded in seven out of ten runs for EA, and in nine out of ten runs for SH5.

The comparative results in Table 2 imply that for the UCB-based BO using a trade-off set \(k ∈ [0,1,10]\), the global optimum can be quickly located in the most test functions with multiple local optimal solution or flat interested area, see for example, the BA, BO, EA, SC6, LE, and SP cases. Taking a set of trade-off coefficient values which are biased towards local exploitation can speed up the convergence while having the ability to jump out of the local optimal solution.

When the trade-off coefficient value \(k = 1.96\) is employed, the performance of BO show superiority on the SC6, H3, RO, and SH5 functions. This is because these test functions are relatively simple, and pure local exploitation is conducive to rapid convergence to global optimum. Contrarily, for the SHU comprising multiple global optimal solutions with large gradients, the small trade-off coefficient \(k = 1.96\) makes the algorithm fall into the local optimum. And the set \([0,1,10]\) tends to local exploitation, which wastes too much resources on local solutions. RGP-UCB can just jump out of the local optimal solution without being too local.

Differently, we can intuitively observe that when \(k = 10\), the acquisition function is biased towards global exploration, and excessive resources are placed on the exploration of the design space. Even after locating the interested area, it cannot quickly converge to the required accuracy. Hence, the UCB-based BO using \(k = 10\) yields the poorest results in general.

For the GP-UCB algorithm, although it provides a regret bound to ensure convergence, is the algorithm prefers a large trade-off coefficient, which is biased towards global exploration. On the contrary, RGP-UCB uses the gamma distribution to retain the regret bound while generate relatively small trade-off coefficient, which hence improves the performance of the algorithm. However, as shown in Figure 1, the performance of RGP-UCB is poor on the simple functions (for example BO, EA, SH5, and SP), because the value of the trade-off coefficient is still large.

5. Engineering applications
This section investigates the characteristics and performance of the UCB-based BOs using different trade-off strategies on two turbomachinery design cases.

5.1. Weight optimization of aircraft wing
The first engineering case is the task of reducing the wing weight of the Cessna C172 Skyhawk aircraft. According to [20], we use the following analytical expression to approximate the weight of the aircraft wing:

\[ \text{Weight} = \text{const} \times \left( \frac{S}{	ext{aspect ratio}} \right) \]
\[ f(x) = 0.036 S_w^{0.758} W_{fw}^{0.0035} \left( \frac{A}{\cos^2(\Lambda)} \right)^{0.6} q^{0.006} \lambda^{0.04} \left( \frac{100 t_c}{\cos(\Lambda)} \right)^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p \] (22)

Table 3 contains the value range and physical meaning of each variable corresponding to the analytical expression, which roughly represents the design variable and corresponding value range of the Cessna C172 Skyhawk aircraft. The algorithm configurations keep the same as the benchmarks in section 2.

| Symbol and Domain | Parameter |
|-------------------|-----------|
| \( S_w \in [150, 200] \) | Wing area (ft\(^2\)) |
| \( W_{fw} \in [220, 300] \) | Weight of fuel in the wing (lb) |
| \( A \in [6, 10] \) | Aspect ratio |
| \( \Lambda \in [-10, 10] \) | Quarter-chord sweep (degrees) |
| \( q \in [16, 45] \) | Dynamic pressure at cruise (lb/ft\(^2\)) |
| \( \lambda \in [0.5, 1] \) | Taper ratio |
| \( t_c \in [0.08, 0.18] \) | Arofoil thickness to chord ratio |
| \( N_z \in [2.5, 6] \) | Ultimate load factor |
| \( W_{dg} \in [1700, 2500] \) | Flight design gross weight (lb) |
| \( W_p \in [0.025, 0.08] \) | Paint weight (lb/ft\(^2\)) |

Figure 2 shows the performance of different UCB-based BOs for optimizing the weight of aircraft wings. It can be observed that for this complex engineering case, in the initial stage of optimization, fully exploring the design space will bring benefits. As a result, \( k = 10 \) and GP-UCB have a better convergence rate in the initial iteration. However, as the number of iteration steps increases, the algorithms that are biased towards local exploitation can finally achieve a better convergence accuracy. Therefore, after enough iterations, the acquisition function with a trade-off set \( k \in [0, 1, 100] \), which fully considers the trade-off between local exploitation and global exploration, has a better performance over the others.
5.2. Drag optimization of turbofan engine

The second engineering case focuses on optimizing the ram drag of the turbofan engine, which can be solved by the EngineSim² simulator developed by NASA Glenn Research Center. This simulator can model the design and testing of jet engines and explore the influence of different design variables on engine performance. Figure 3 shows the geometry of the turbofan engine. Table 4 shows the selected design variables and the corresponding value ranges in our optimization case.

![Figure 3. Geometry of turbofan engine².](https://www.grc.nasa.gov/WWW/K-12/airplane/ngnsim.html)

Table 4. The input variables of a light aircraft wing and their input ranges.

| Symbol and Domain | Parameter                                      |
|-------------------|------------------------------------------------|
| FanBPR ∈ [3.0, 3.6] | Fan bypass pressure ratio                      |
| FanPR ∈ [1.7, 1.8]  | Fan pressure ratio                              |
| CompPR ∈ [18, 24]   | Compressor pressure ratio                       |
| CombTMax ∈ [2200, 2800] | Combustion temperature maximum (K)            |
| Mach ∈ [0.6, 0.9]   | Flight Mach number                             |
| Altitude ∈ [5000, 10000] | Flight altitude (m)                           |

Figure 4 shows the performance of different algorithms to optimize the ram resistance of the turbofan engine. For this extreme problem that is biased towards global exploration, it can be observed that the larger trade-off coefficient yields better performance., see for example the results of $k = 10$ and GP-UCB. As for the BO using a set of trade-off coefficient values, since there is a case of $k = 100$, it converges quickly at the beginning of the iteration. However, with the optimization proceed, the cases of $k = 0.1$ makes the BO using a set of trade-off coefficient values more concerned about local exploitation, and the final result is poor. Therefore, it can be concluded that the proper trade-off set is problem-dependent, and how to achieve an adaptive and dynamic trade-off set is left for our future work.

² https://www.grc.nasa.gov/WWW/K-12/airplane/ngnsim.html.
6. Conclusion
Through numerical benchmarks and engineering applications, the impact of trade-off coefficients of local exploitation and global exploration on the performance of UCB-based Bayesian optimization is compared and studied. The results show that the performance of the algorithm varies greatly when different trade-off strategies are used. When a fixed value $k = 1.96$ is used, the algorithm favors local exploitation and has a faster convergence speed on simple benchmarks, which however is not the case for complex benchmarks. When $k = 10$, the algorithm is biased towards global exploration. Consequently, most of the resources might be wasted in exploring the design space, hence slowing the convergence speed. When GP-UCB is used, although the algorithm provides a regret bound and ensures convergence, it usually provides a relatively large value for the trade-off coefficient, which cannot well balance local and global well, thus resulting in poor performance. Besides, the RGP-UCB uses a gamma distribution to sample small trade-off coefficient with high probability while ensuring convergence; but it requires a certain prior knowledge to set a reasonable $\theta$, which will adjust the trade-off coefficient. This is difficult for complex problems. The trade-off set $k \in [0, 1, 100]$ is generally biased towards local exploitation, which improves the convergence speed while ensuring the ability to jump out of the local optimum, and has a better performance in general. But this specific trade-off set has also been found to converge slowly for several cases. Therefore, it can be concluded that the proper trade-off set is problem-dependent, and the issue of how to achieve an adaptive and dynamic trade-off set is left for our future work.

7. References
[1] Chen H, Wen S, Xu Z and Xue P 2013 Chinese Journal of Turbomachinery 43-46
[2] Yin X, Wang J, Fan Y, Wen S and Zhang P 2018 Chinese Journal of Turbomachinery 60 16-22
[3] Ji C, Xu H, Sun Q and Sun M 2017 Chinese Journal of Turbomachinery 59 44-48
[4] Xu S, Zhong S, Guo Z, Yang S and Meng J 2018 Chinese Journal of Turbomachinery 60 64-69
[5] Jiang W, Zhang P, Yang Z and Pei Q 2019 Chinese Journal of Turbomachinery 61 23-28
[6] Cui J and Yang B 2018 Journal of Software 29 3068-3090
[7] Rasmussen C E 2003 Machine Learning Summer School Conf. (Tubingen) vol 3176 (Berlin: Springer-verlag Berlin) pp 63-71
Acknowledgments

This work was supported by the National Key Research and Development Project (2016YFB0600104), the National Natural Science Foundation of China (52005074), the research and Innovation in Science and Technology Major Project of Liaoning Province (2019JH1-10100024), and the Fundamental Research Funds for the Central Universities (DUT19RC(3)070) at Dalian University of Technology.