Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.025812

ARTICLE

Technique for Multi-Pass Turning Optimization Based on Gaussian Quantum-Behaved Bat Algorithm

Shutong Xie, Zongbao He and Xingwang Huang*

Computer Engineering College, Jimei University, Xiamen, 361021, China

*Corresponding Author: Xingwang Huang. Email: huangxw@jmu.edu.cn

Received: 31 July 2022 Accepted: 19 October 2022

ABSTRACT

The multi-pass turning operation is one of the most commonly used machining methods in manufacturing field. The main objective of this operation is to minimize the unit production cost. This paper proposes a Gaussian quantum-behaved bat algorithm (GQBA) to solve the problem of multi-pass turning operation. The proposed algorithm mainly includes the following two improvements. The first improvement is to incorporate the current optimal positions of quantum bats and the global best position into the stochastic attractor to facilitate population diversification. The second improvement is to use a Gaussian distribution instead of the uniform distribution to update the positions of the quantum-behaved bats, thus performing a more accurate search and avoiding premature convergence. The performance of the presented GQBA is demonstrated through numerical benchmark functions and a multi-pass turning operation problem. Thirteen classical benchmark functions are utilized in the comparison experiments, and the experimental results for accuracy and convergence speed demonstrate that, in most cases, the GQBA can provide a better search capability than other algorithms. Furthermore, GQBA is applied to an optimization problem for multi-pass turning, which is designed to minimize the production cost while considering many practical machining constraints in the machining process. The experimental results indicate that the GQBA outperforms other comparison algorithms in terms of cost reduction, which proves the effectiveness of the GQBA.

KEYWORDS

Bat algorithm; quantum behavior; gaussian distribution; numerical optimization; multi-pass turning

1 Introduction

In the manufacturing field, the technological challenge of machining operations is to produce products of the desired quality with high productivity and low cost. To reduce the machining cost for economical reasons, the optimization of the machining parameters is one of the most important issues, since these parameters have a strong impact on productivity, cost and quality [1]. Most turning processes require multi-pass turning, considering economic factors. The optimization problem for machining parameters in multi-pass turning becomes very complicated when a large number of actual machining constraints need to be considered [2]. Traditional optimization techniques, namely, dynamic programming [3] and the sequential unconstrained minimization technique [4], may be helpful in addressing some specific issues. However, these methods tend to find local optimal results. As a result,

heuristic algorithms and nature-inspired swarm intelligence (SI) techniques have been introduced to solve economic machining problems because they have power in global search and robustness [5–9]. Chen proposed a scatter search to address optimization problems [10]. The genetic algorithm (GA) has been used extensively by some researchers [11–13] to optimize the process of multi-pass turning. The researchers applied varying improvements of the genetic operators to improve the GA performance to obtain better results. To solve optimization problems, ant colony optimization (ACO) [13–15], particle swarm optimization (PSO) [16–18], cuckoo optimization algorithm (COA) [9], grasshopper optimization algorithm (GOA) [19], bird swarm algorithm (BSA) [20], and the hybrid immune algorithm [21] were also developed.

In this paper, the first application of an improved version of the prevailing bat algorithm (BA) in the optimization of turning operations is presented. The BA is a recently proposed swarm intelligence optimization technique [22]. By mimicking the foraging behaviours of bats in searching for prey, the BA has incorporated the advantages of many classical techniques in a reasonable manner, including PSO, the GA [23] and simulated annealing (SA) [24]. Not only does the BA retain simplicity, it has also been shown to be more effective than its predecessors, especially in cases of lower dimensions. In addition, it is convenient to implement using a variety of programming languages. Therefore, the BA has been used to solve various engineering optimization applications [25]. However, since the population diversity of the BA is not sufficiently high, the BA may fall into a local optimum when dealing with applications of high dimensions [26]. To address this defect, many versions of the BA have been proposed, such as the CLBA [27], IBA [28], DLBA [29], and HSBA [30]. The quantum bat algorithm (QBA) [31] is a new technique based on the idea of quantum computing, which can increase the population diversity of the BA. In addition, considering the self-adaptive compensation of the Doppler effect in sound, the echolocation mechanism of bats can be simulated in the QBA. The QBA has been proven to outperform the standard BA and some well-known algorithms [31].

However, because the position of the stochastic attractor determines the central position of the quantum-behaved bat search area in the population and some critical random numbers are produced in the quantum mutation phase utilizing a uniform distribution, there is still premature convergence in the QBA to a certain extent. Motivated by these disadvantages, this paper proposes a QBA approach based on a Gaussian distribution (GQBA) to address the problem of premature convergence. The main contributions of this paper are as follows:

1. We propose an improved quantum-behaved bat algorithm using Gaussian distribution and apply it to the multi-pass turning problem.
2. We introduce a new stochastic attractor updating strategy to promote the diversification of swarms and a new quantum-behaved bat updating strategy to perform a more accurate search and avoid premature convergence.
3. Experiments on the numerical and multi-pass turning optimization indicate the efficiency of the GQBA. Comparisons with other comparative algorithms validate that the GQBA is competitive and is a good alternative approach.

The remainder of this paper is structured as follows: Section 2 briefly introduces related works. Section 3 gives the details of the GQBA proposed in this article. Section 4 presents the numerical validation and comparison. Section 5 presents an application of the GQBA to the multi-pass turning operation problem as well as the experimental results. Finally, the concluding comments and some future research directions are presented in Section 6.

2 Related Works

2.1 The Original BA Algorithm

When bats are foraging for food, they use echolocation to find prey and avoid obstacles. Inspired by the foraging behaviour of real bats, Yang proposed the bat algorithm in 2010 [22]. The original BA uses a frequency-tuning method to promote the diversification of the population. Additionally, it utilizes the automatic scaling technique to maintain a balance between exploration and exploitation during the optimization process by imitating the changes in pulse loudness and emission frequency during foraging. The procedures of the original BA are given below.

Each virtual bat in the swarm moves towards the global optimal position, which is the reason that all bats fly towards prey when foraging. The frequency (f_i), velocity (v_i) and position value (x_i) of the virtual bats change during the search procedure according to the following Eqs. (1)–(3):

$$f_i = f_{min} + (f_{max} - f_{min})\,\beta \tag{1}$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - gb^{t-1})f_i \tag{2}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{3}$$

where β is a stochastic value that is produced uniformly in [0, 1], f_{min} denotes the lowest frequency, f_{max} denotes the highest frequency, and gb^t indicates the global optimal position. Utilizing the formulas above, the BA can carry out the exploration operation.

For the exploitation stage, to generate a new candidate position for every virtual bat when a position is selected from the current optimal positions, a technique called a local random walk is adopted, which is given as Eq. (4):

$$x_{new} = x_{old} + \varepsilon \overline{A}^{t-1}, \tag{4}$$

where ε indicates a uniform stochastic value in $[-1, 1]$ that determines the direction of the new candidate position. Note that $\overline{A}^{t-1}$ indicates the mean of the loudness values of all virtual bats at the $(t-1)$th iteration.

When foraging, every virtual bat gradually adjusts its loudness value and emission frequency to locate prey. The equations for updating the value of loudness A_i and the value of emission frequency r_i in every iteration can be given as Eqs. (5) and (6):

$$A_i^t = \alpha A_i^{t-1} \tag{5}$$

$$r_i^t = r_i^0[1 - \exp(-\gamma(t-1))], \tag{6}$$

where r_i^0 denotes the initial value of the emission rate of the ith virtual bat, α and γ indicate the loudness attenuation coefficient and the rate enhancement coefficient of pulse emission, respectively. The value of α is in the range [0, 1], and the value of γ is positive ($\gamma > 0$). Both α and γ are constants. In fact, similar to the cooling parameter in SA, the parameter α determines the convergence characteristic of the BA. $\alpha = \gamma$ is commonly used in the literature.

The pseudo-code of the original BA is shown in Fig. 1.

Initialize the bat swarm $x_i (i = 1, 2, ..., n)$ and v_i;
Define pulse frequency f_i, pulse rate r_i and the loudness A_i;
while $(t < t_{max})$ **do**
  Generate new solutions by adjusting frequency, updating velocities and positions using equations 1-3;
  if (rand $> r_i$) **then**
    Select a solution among the best solutions randomly;
    Generate a local solution around the selected best solution using equation 4;
  end if
  if (rand $< A_i$ && $f(x_i) < f(x_g)$) **then**
    Accept the new solutions;
    Increase r_i and reduce A_i using equations 5 and 6;
  end if
  Rank the bats and find the current best gb;
  $t = t + 1$;
end while

Figure 1: Pseudo-code of the original BA [22]

2.2 The QBA Algorithm

The original BA has been used in many applications. However, since the population diversity of the BA is not sufficiently high, the BA does not perform well in multimodal cases. By analyzing the flight path of virtual bats, Meng et al. [31] developed the QBA. In the QBA, both the quantum behaviour and Doppler effect are taken into account, as well as other characteristics of the original BA. The mutation operator of quantum behaviour contributes to promoting the diversity of the swarm, which is ultimately beneficial in avoiding premature convergence.

The QBA is generally proposed on the framework of the standard BA. The exploration and exploitation procedures in the QBA are controlled by the decreasing loudness value A and increasing emission rate value r, respectively. However, the approach to generating new candidate positions in the QBA is not the same as that in the standard BA. Two more idealized rules have been introduced [32]: (1) the virtual bats have two foraging habitats instead of one foraging habitat depending on a random choice, and (2) the virtual bats self-adaptively compensate for the Doppler effect in sound. The positions of the virtual bats determined by quantum behaviour in the QBA can be described as Eq. (7):

$$x_{id}^{t} = \begin{cases} p_d^{t-1} + \delta \times |mbest_d - x_{id}^{t-1}| \times \ln\left(\dfrac{1}{u}\right), & z < 0.5 \\ p_d^{t-1} - \delta \times |mbest_d - x_{id}^{t-1}| \times \ln\left(\dfrac{1}{u}\right), & z \geq 0.5 \end{cases} \tag{7}$$

where

$$p_d^{t-1} = gb_d^{t-1} \tag{8}$$

$$\delta = \delta_{max} - \frac{(\delta_{max} - \delta_{min})}{t_{max}} \times t \tag{9}$$

$$mbest_d = \frac{1}{N}\sum_{i=1}^{N} pbest_{id} \tag{10}$$

p_d^{t-1} denotes the value in the dth dimension of the best previous position gb at iteration t, called the stochastic attractor, x_{id}^t indicates the position of the ith virtual bat in the dth dimension at the tth iteration, and u and z are random values generated uniformly from [0, 1]. δ is a design parameter named the contraction-expansion parameter [32], which can be utilized to control the convergence rate of the techniques. δ_{max} and δ_{min} indicate the initial value and final value of δ, respectively. In the QBA, $\delta_{max} = 1$ and $\delta_{min} = 0.5$ are adopted. *mbest*, called the (Mean Best), represents the global point of the swarm, which is also the average of the present optimal positions *pbest* of all virtual bats. N is the population size, and $pbest_{i,d}$ indicates the value of the present optimal position of the ith bat in dimension d.

Considering the self-adaptive capability of virtual bats to compensate for the Doppler effect, the equations mentioned in Eqs. (1) and (2) can be changed as Eqs. (11)–(13):

$$f_{id} = \frac{(c + v_i^t)}{(c - v_g^{t-1})} \times f_{id} \times \left[1 + \phi_i \times \frac{(gb_d^t - x_{id}^t)}{|gb_d^t - x_{id}^t| + \varepsilon}\right] \tag{11}$$

$$v_{id}^t = (w \times v_{id}^{t-1}) + (gb_d^t - x_{id}^t) \times f_{id} \tag{12}$$

$$x_{id}^t = x_{id}^{t-1} + v_{id}^t, \tag{13}$$

where f_{id} is the frequency value of the ith virtual bat in the dth dimension, v_g^{t-1} indicates the velocity value of the global optimal position at the $(t-1)$th iteration, and ϕ_i indicates a positive value of the ith virtual bat in [0, 1]. The inertia weight factor w is used to update the velocity vector, which is similar to the inertia weight factor in PSO. C ($c = 340$ m/s) is the speed of sound in the air.

In the exploitation stage, in the QBA, the new candidate position of the virtual bat is produced as Eqs. (14) and (15):

$$x_{id}^t = gb_d^{t-1} \times \chi \tag{14}$$

$$\sigma^2 = |A_i^{t-1} - \overline{A}^{t-1}| + \varepsilon, \tag{15}$$

where $\chi = N(0, \sigma^2)$ indicates a Gaussian distribution, x_{id}^t denotes the position of the ith virtual bat at the tth iteration and gb_d^t indicates the current global optimal position searched by the virtual bats in the dth dimension. A_i^{t-1} indicates the loudness value of the ith virtual bat at the $(t-1)$th iteration. ϵ is utilized here to guarantee that σ^2 is positive.

During the search process, if the fitness of the objective function is not promoted in the Kth time step, then a simple strategy is employed to enhance the algorithm, which is to reassign the loudness value A_i and reset the temporary emission rate value r_i, which is a stochastic value generated uniformly from [0.85, 0.9].

The pseudo-code of the QBA is given in Fig. 2.

Define basic BA parameters: α, γ, f_{min}, f_{max}, A_0 and r_0;
Initialize the number of individuals (N) contained by the population, iteration (t_{max}), probability of habitat selection (P), inertia weight (w), compensation rates for Doppler Effect in echoes (C), contraction/expansion coefficient (δ);
Evaluate the value of fitness function for each;
while ($t < t_{max}$) **do**
  if (rand $> P$) **then**
    generate new solutions using equation in 7;
  else
    generate new solutions using equation in 1 and 11-13;
  end if
  if (rand $> r_i$) **then**
    generate a local solution around the selected best solution using equations 14 and 15;
  end if
  if (rand $< A_i$ && $f(x_i) < f(gb)$) **then**
    Accept the new solutions;
    Increase r_i and reduce A_i using equations 5 and 6;
  end if
  Rank the bats and find the current best gb;
  if gb does not improve in K time step **then**
    re-initialize the loudness A_i and set temporary pulse r_i [0.85-0.9]
  end if
  $t = t + 1$;
end while

Figure 2: Pseudo-code of the QBA [31]

3 The Proposed GQBA

As mentioned above, the QBA is an improved BA variant that has good performance. However, by analyzing Eq. (7), it can be seen that due to the convergence of quantum-behaved bats to the stochastic attractor p_d^{t-1}, when the stochastic attractor is very close to the global optimal individual, the bats will be concentrated near the global optimal solution, resulting in a mass aggregation phenomenon. However, if the stochastic attractor is located at the local optimal solution and is far from the global optimal solution, the bats will be distributed near the local optimal solution with a higher probability, which can easily lead to the premature convergence of the algorithm. Thus, the position of the stochastic attractor p_d^{t-1} determines the central position of the quantum-behaved bats' search area in the population.

To address the premature convergence problem of the QBA, this paper makes full use of the guiding role of the stochastic attractor and proposes a new strategy to update the position of the stochastic attractor as well as the bats in the population; this leads to the quantum-behaved bat algorithm based on a Gaussian distribution (the GQBA). In the implementation of the GQBA, some improved schemes have been substituted into the QBA, which are illustrated in the remainder of this section.

The first modification is to update the stochastic attractor with the guidance of the current optimal position value of the ith bat $pbest_{id}^{t-1}$ and the global best position gb_d^{t-1}. In fact, the introduction of $pbest_{id}^{t-1}$ can increase the diversity of the population and lead the QBA to perform a more thorough global search. Parameter p_d^{t-1} of Eq. (7) is changed according to the Eq. (16):

$$p_d^{t-1} = \frac{c_1 pbest_{id}^{t-1} + c_2 gb_d^{t-1}}{c_1 + c_2}, \tag{16}$$

where c_1 and c_2 are generated randomly within [0, 1].

The second change is to substitute coefficients u into Eq. (7) with the absolute value of the Gaussian distribution whose σ^2 is 1 and mean is 0, which can be defined as $G = \text{abs}(N(0, 1))$. Many studies [33–35] have shown that a Gaussian distribution with a long tail can perform a more accurate search near the last generation of individuals, improve the local search capability, provide a greater search step and random walking distance, expand the search space, and improve the algorithm's ability to jump out of local optimality.

The formula of probability density for G in one dimension can be stated as Eq. (17):

$$f(x) = \frac{2}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right), \quad x \geq 0. \tag{17}$$

Therefore, in the GQBA, the position of the quantum-behaved bats can be described as Eq. (18):

$$x_{id}^t = \begin{cases} p_d^t + \delta \times |mbest_d - x_{id}^t| \times \ln\left(\dfrac{1}{G}\right), & \text{rand} < 0.5 \\ p_d^t - \delta \times |mbest_d - x_{id}^t| \times \ln\left(\dfrac{1}{G}\right), & \text{rand} \geq 0.5, \end{cases} \tag{18}$$

where $G = \text{abs}(N(0, 1))$. Clearly, in line with Eq. (17), $f(0) = 0$; hence, G is within the logarithmic function's domain range (> 0).

In the GQBA, the stochastic attractor in terms of p_d instructs the exploration stage to ensure the convergence rate, while the habitat selection design of two different foraging habitats (a Gaussian quantum-behaved habitat and a Doppler effect-compensated mechanical habitat) contributes to the exploitation stage to help the algorithm jump out of the local best positions and avoid premature convergence.

The procedure of the GQBA is shown in Fig. 3.

Define basic BA parameters: α, γ, f_{min}, f_{max}, A_0 and r_0;
Initialize the number of individuals (N) contained by the population, iteration (t_{max}), probability of habitat selection (P), inertia weight (w), compensation rates for Doppler Effect in echoes (C), contraction/expansion coefficient (δ);
Evaluate the value of fitness function for each;
while ($t < t_{max}$) **do**
  if (rand $> P$) **then**
    generate new solutions using equation in 18;
  else
    generate new solutions using equation in 1 and 11-13;
  end if
  if (rand $> r_i$) **then**
    generate a local solution around the selected best solution using equations 14 and 15;
  end if
  if (rand $< A_i$ && $f(x_i) < f(gb)$) **then**
    Accept the new solutions;
    Increase r_i and reduce A_i using equations 5 and 6;
  end if
  Rank the bats and find the current best gb;
  if gb does not improve in K time step **then**
    re-initialize the loudness A_i and set temporary pulse r_i [0.85-0.9]
  end if
  $t = t + 1$;
end while

Figure 3: Pseudo-code of the GQBA

4 Numerical Validation and Comparison

In this section, thirteen classical benchmark functions are utilized to verify the efficiency of the GQBA, which are illustrated in Tables 1 and 2. These benchmark functions have been utilized in many numerical optimization studies [36–39]. In this paper, the thirteen classical test functions can be divided into two classes. The first class consists of 7 unimodal benchmark functions with only one global best solution and effectively validates the meta-heuristic algorithms in terms of the convergence rate as well as the local search ability. The second class consists of 6 multimodal functions, the quantity of whose local optima increases exponentially. Multimodal functions are usually utilized to examine the global search ability of algorithms. All the tests on each benchmark function are repeated 30 times independently. All the tests described in this article were executed using MATLAB 2014a on a computer equipped with an Intel(R) Core(TM) i5-6500 CPU at 3.20 GHz with 8.0 GB of RAM.

Table 1: Unimodal benchmark test cases F_1–F_7

Instance	D	Range	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	[−100, 100]	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[−10, 10]	0

(Continued)

Table 1 (continued)

Instance	D	Range	f_{min}
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	30	[−100, 100]	0
$F_4(x) = max_i\{\|x_i\|, 1 \le i \le n\}$	30	[−100, 100]	0
$F_5(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)\right]^2$	30	[−30, 30]	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	[−100, 100]	0
$F_7(x) = \sum_{i=1}^{n} ix_i^4 + \text{random}[0, 1]$	30	[−1.28, 1.28]	0

Table 2: Multimodal benchmark test cases F_8–F_{13}

Instance	D	Range	f_{min}
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\|x_i\|}\right)$	30	[−500, 500]	$-418.9829 \times D$
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30	[−5.12, 5.12]	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	[−32, 32]	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[−600, 600]	0
$F_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right\} + \sum_{i-1}^{n} u(x_i, 10, 100, 4) + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a)^m x_i < -a \end{cases}$	30	[−50, 50]	0
$F_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i + 1)\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right]\right\} + \sum_{i=1}^{n} u(x_x, 5, 100, 4)$	30	[−50, 50]	0

Since the heuristic algorithm is a random optimization method, at least 10 independent runs need to be carried out to generate meaningful statistical results. In addition to the average value and standard deviation, statistical tests, such as the Wilcoxon rank sum test, are required to test the significance of the consequences based on each separate run. In this research, nonparametric Wilcoxon rank sum tests were carried out to examine whether the consequences of the GQBA were conspicuously different from those of comparative techniques. $p < 0.05$ indicates that there exists a conspicuous difference between two techniques, while $p \ge 0.05$ indicates that there is no conspicuous difference.

To examine the efficiency of the GQBA and to obtain an exhaustive comparison, the GQBA and four other techniques, the BA [22], the QBA [31], PSO [40] and the GSA [41], are tested on the sets of benchmark instances above. In all experiments, the total number of iterations is 10000, and the number of individuals is 50. Table 3 shows the other specific parameter designs for each technique.

The simulation results are illustrated in Tables 4 and 5 and Figs. 4–16. The best consequences for each test function are denoted in bold.

Table 3: The parameter design for the BA, QBA, GQBA, PSO and GSA

Algorithms	Parameter design
BA [22]	$f_{min} = 0, f_{max} = 2, \alpha = 0.5, \gamma = 0.5, A = u(0, 1), r = 0.01$
QBA [31]	$\alpha = 0.5, \gamma = 0.5, A = u(0, 1), r = 0.01, \delta_{max} = 1.0, \delta_{min} = 0.5, P = u(0.6, 0.9)$
GQBA (Ours)	$A = u(0, 1), r = 0.01, \alpha = 0.5, \gamma = 0.5, \delta_{max} = 1.0, \delta_{min} = 0.5, P = u(0.6, 0.9)$
PSO [40]	Refer to the original paper
GSA [41]	Refer to the original paper

Note: *u(m, n)* represents a random value generated uniformly from [m, n].

As shown in Table 4, the GQBA outperforms the other techniques in most cases, followed by PSO, the GSA, the QBA and the BA. To a certain extent, this is evidence that when the dimensionality of the search scope is high, the local optimal avoidance effect of the BA is insufficient.

As seen from Table 4, concerning accuracy, the GQBA has a better mean than the other techniques in eight out of 13 instances (F_1, F_2, F_3, F_4, F_7, F_9, F_{10} and F_{11}). Meanwhile, in terms of stability, the GQBA performs better than the other techniques in 7 out of 13 instances (F_1, F_2, F_3, F_4, F_7, F_{10} and F_{11}). For local optimum avoidance, with regard to the p values in Table 5, the performance of the GQBA in three out of 13 test functions is not significantly different (F_8, F_9 and F_{11}). Taking into account the accuracy results above, it can be concluded that the GQBA developed in this paper is able to provide significant results in six out of 13 instances (F_1, F_2, F_3, F_4, F_7 and F_{10}), which means that the GQBA has better local optimum avoidance. For the other functions, the performance of the GQBA is ranked second and third two and three times, respectively. In general, this means that the GQBA performs well for both types of functions.

Table 4: Numerical comparison of different methods on the test instances

F	BA		QBA		GQBA (Ours)		PSO		GSA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
F_1	3.81e−05	5.72e−05	2.20e−36	1.12e−35	**0**	**0**	9.48e−84	5.19e−83	8.17e−18	1.68e−18
F_2	1.96e+04	7.70e+04	2.36e−21	5.49e−21	**0**	**0**	1.41e−40	6.08e−40	1.33e−08	1.57e−09
F_3	1.57e−04	6.44e−05	1.48e−19	5.41e−19	**0**	**0**	6.09e−08	7.02e−08	3.62e−17	9.34e−18
F_4	3.99e+01	6.35e+00	1.52e+01	5.63e+00	**0**	**0**	1.03e−04	1.07e−04	1.59e−09	1.74e−10
F_5	**8.61e+00**	3.00e+01	2.23e+01	1.97e+00	1.99e+01	1.12e+01	2.78e+01	2.58e+01	1.42e+01	**3.04e−01**
F_6	2.73e−05	7.57e−06	1.43e−01	3.00e−01	2.95e−15	2.25e−15	**5.34e−33**	**7.19e−33**	7.51e−18	2.24e−18
F_7	1.89e−03	5.94e−04	1.54e−03	9.24e−04	**1.29e−05**	**1.15e−05**	4.14e−03	1.28e−03	1.10e−02	2.72e−03
F_8	**−7.19e+03**	8.78e+02	−6.40e+03	8.72e+02	−6.90e+03	8.76e+02	−6.87e+03	8.52e+02	−2.71e+03	**5.20e+02**
F_9	1.89e+02	3.55e+01	1.47e+02	5.79e+01	**1.04e+01**	1.37e+01	2.25e+01	6.11e+00	1.25e+01	**3.22e+00**
F_{10}	20.0e+02	1.15e−03	6.28e−01	1.07e+00	**1.13e−15**	**9.01e−16**	9.53e−15	3.19e−15	2.24e−09	2.59e−10
F_{11}	1.74e+02	5.06e+01	7.51e−03	1.08e−02	**0**	**0**	1.54e−02	1.25e−02	9.04e−04	3.80e−03
F_{12}	2.46e+01	1.21e+01	4.72e+00	5.26e+00	2.97e−01	9.07e−01	**1.60e−32**	**4.51e−34**	4.97e−20	1.35e−20
F_{13}	7.28e+01	9.42e+00	9.75e+00	1.28e+01	2.76e−15	2.49e−15	3.66e−04	2.01e−03	**7.89e−19**	**1.75e−19**

Table 5: *p*-values over all tests

F	GQBA	BA	QBA	PSO	GSA
F_1	N/A	1.2118e − 12	1.2118e − 12	1.2118e − 12	1.2118e − 12
F_2	N/A	1.2118e − 12	1.2118e − 12	1.2118e − 12	1.2118e − 12
F_3	N/A	1.2118e − 12	1.2118e − 12	1.2118e − 12	1.2118e − 12
F_4	N/A	1.2118e − 12	1.2118e − 12	1.2118e − 12	1.2118e − 12
F_5	N/A	0.0042	0.0011	0.0011	3.9881e − 04
F_6	N/A	3.0199e − 11	3.0199e − 11	3.0199e − 11	3.0199e − 11
F_7	N/A	3.0199e − 11	3.0199e − 11	3.0199e − 11	3.0199e − 11
F_8	N/A	0.2838	0.0339	0.7958	3.0199e − 11
F_9	N/A	1.6179e − 11	4.4870e − 11	0.0045	0.0619
F_{10}	N/A	2.3638e − 12	2.3547e − 12	5.9197e − 13	2.3638e − 12
F_{11}	N/A	1.2118e − 12	2.9343e − 05	6.1501e − 10	0.1608
F_{12}	N/A	4.0772e − 11	9.7555e − 10	2.0535e − 11	3.0199e − 11
F_{13}	N/A	3.0199e − 11	3.0199e − 11	4.6152e − 10	3.0199e − 11

Note: N/A represents 'Not Applicable', and p values that are not less than 0.05 are underlined.

Figs. 4–16 show the convergence comparison of the five algorithms for the thirteen benchmark cases. The values shown in the convergence curves above them indicate the average of the objective values obtained from 30 separate runs. On the basis of the figures, the original BA and QBA converge at a faster speed with fewer iterations. However, in many cases, they are more likely to fall into local optima. It can also be seen from the figures that the convergence rate of the GQBA is similar to that of the QBA, while the GQBA can prevent premature convergence and can offer higher performance accuracy on most benchmark instances.

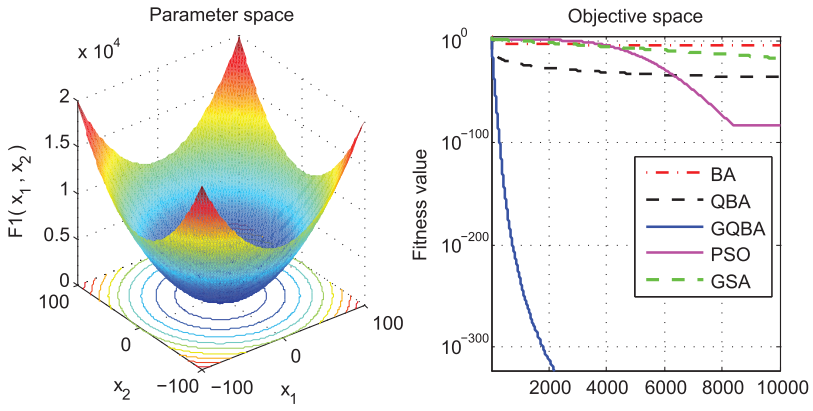

Figure 4: Convergence comparison of five techniques for F_1

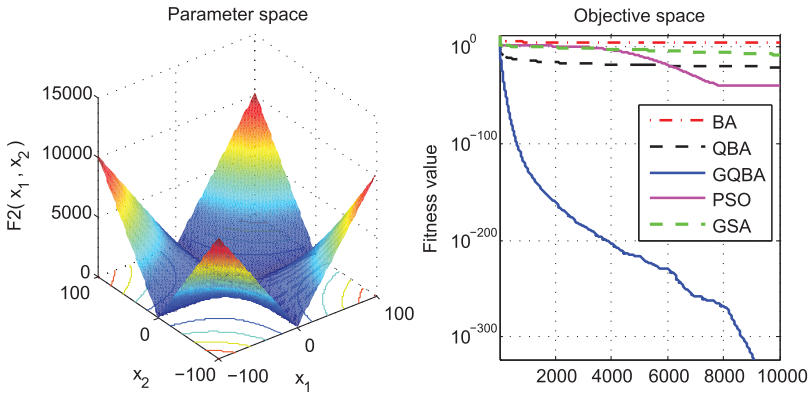

Figure 5: Convergence comparison of five techniques for F_2

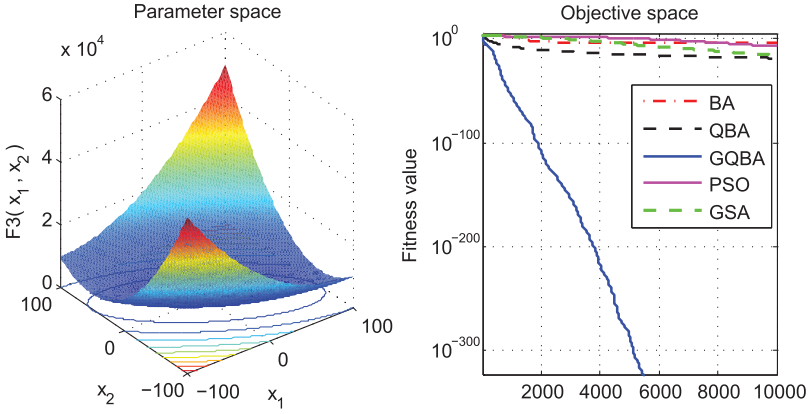

Figure 6: Convergence comparison of five techniques for F_3

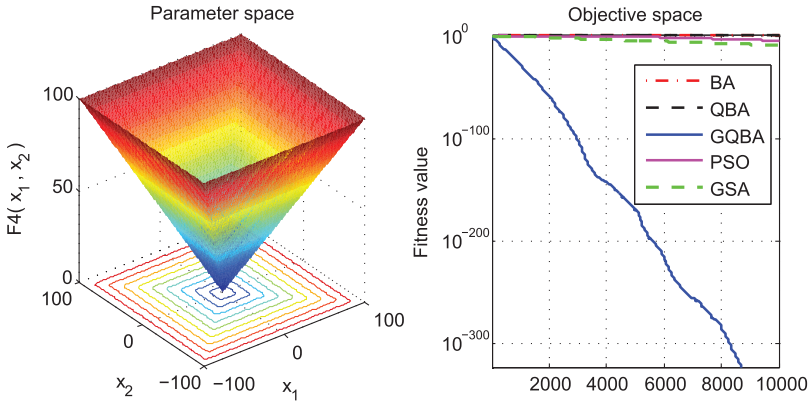

Figure 7: Convergence comparison of five techniques for F_4

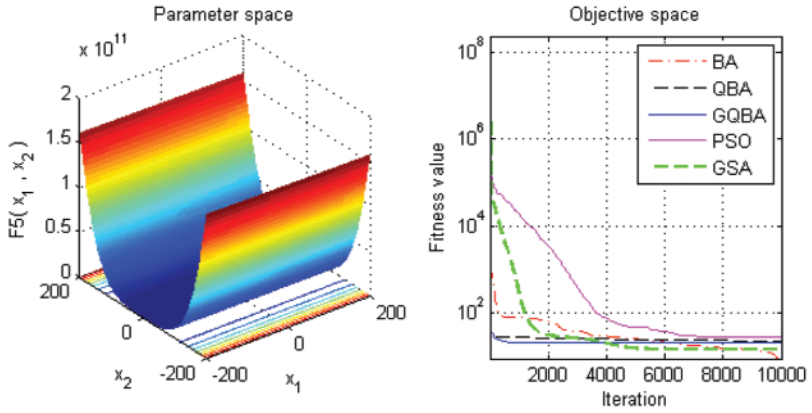

Figure 8: Convergence comparison of five techniques for F_5

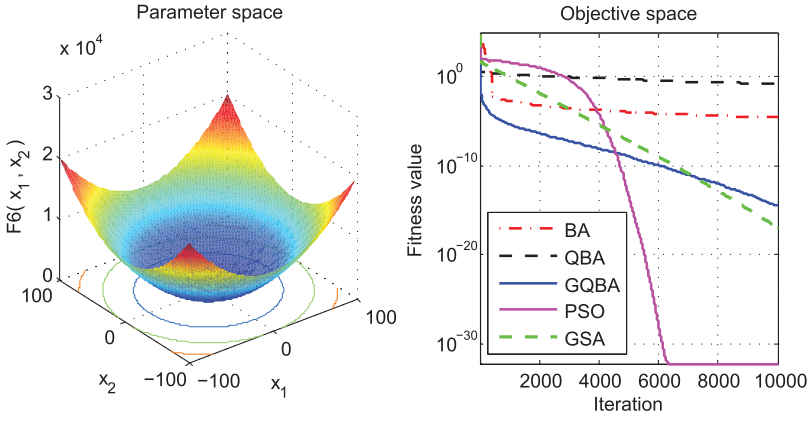

Figure 9: Convergence comparison of five techniques for F_6

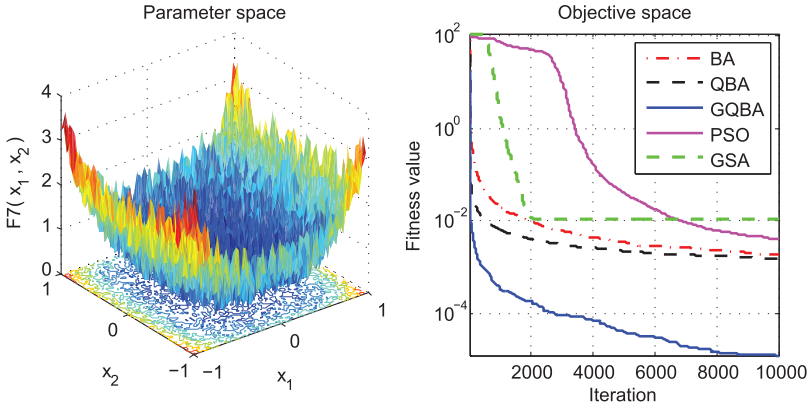

Figure 10: Convergence comparison of five techniques for F_7

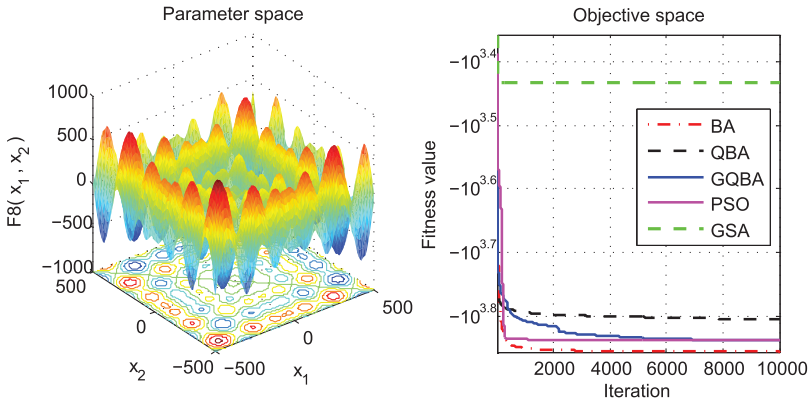

Figure 11: Convergence comparison of five techniques for F_8

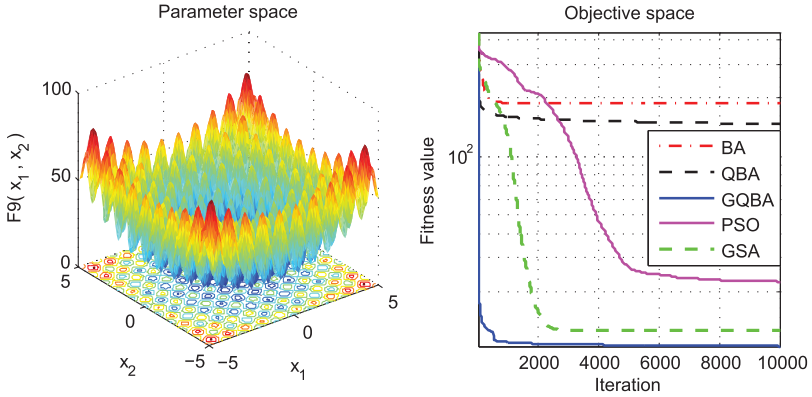

Figure 12: Convergence comparison of five techniques for F_9

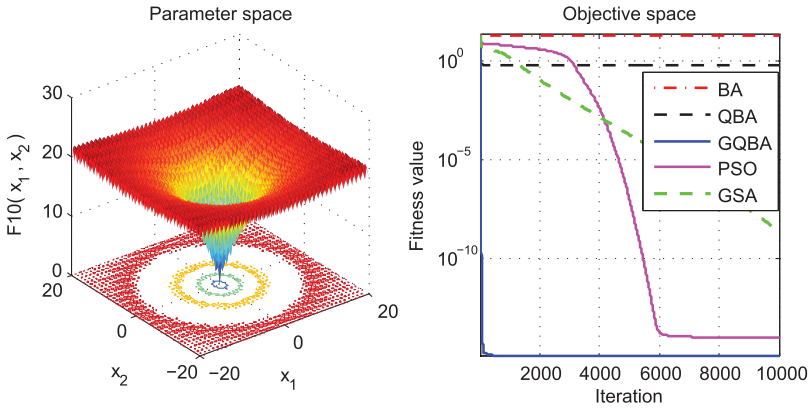

Figure 13: Convergence comparison of five techniques for F_{10}

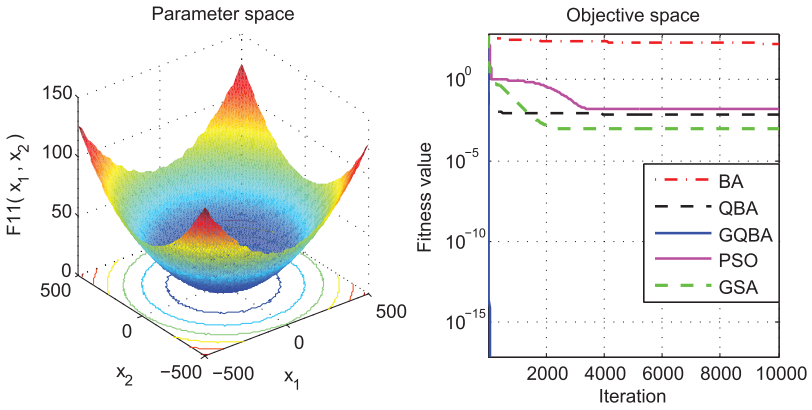

Figure 14: Convergence comparison of five techniques for F_{11}

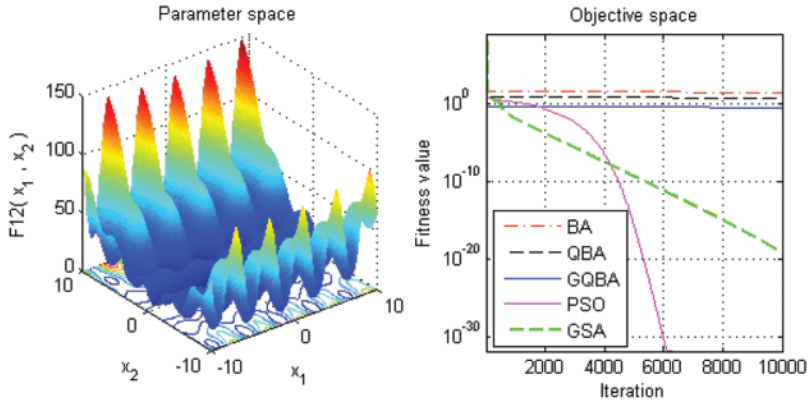

Figure 15: Convergence comparison of five techniques for F_{12}

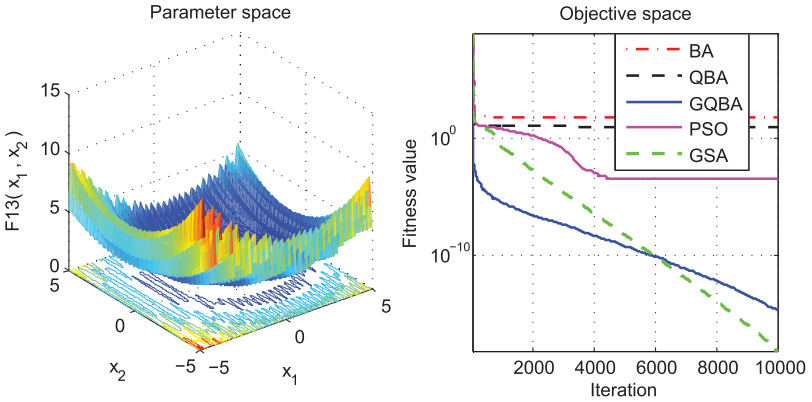

Figure 16: Convergence comparison of five techniques for F_{13}

5 GQBA for Optimization of the Multi-Pass Turning Process

In this section, the GQBA with the pruning strategy is developed to address the constrained manufacturing optimization problem of multi-pass turning optimization. The goal of this problem is to find the optimal cutting parameters to minimize the unit production cost (UC) [8,9,42]. The minimization process is subject to many machining constraints that describe the states of the machining procedure. Machining optimization has been investigated by different techniques, such as ACO [14], PSO [16,43], the GA [5], the artificial bee colony (ABC) [44–46], the COA [9], the firefly algorithm (FA) [47], the flower pollination algorithm (FPA) [8], and differential evolution (DE) [48]. It is very convenient to compare the proposed GQBA with the methods developed previously.

5.1 Mathematical Model of Multi-Pass Turning

In this article, we adopt the mathematical model presented in [8,9,42] for optimizing the cutting parameters. The goal of the optimization model is to find the optimal cutting parameters, i.e., the depth-of-cut, feed rate and cutting speed, for both finishing and rough machining to minimize the unit production cost. A schematic representation of a turning operation is given in Fig. 17.

5.1.1 The Objective Function: Unit Production Cost (UC)

The UC of the multi-pass turning process is usually indicated by a combination of 4 basic cost factors:

(1) The cost due to the real cutting time (C_M).
(2) The machine idle cost for setup operations and tool idling motion (C_I).
(3) The cost of tool replacement (C_R).
(4) The tool cost (C_T).

Thus, the UC can be expressed as following Eq. (19) [8,9,42]:

$$\begin{aligned} UC &= C_M + C_I + C_R + C_T \\ &= \left[\frac{\pi DL}{1000V_t f_r}\left(\frac{d_t - d_s}{d_r}\right) + \frac{\pi DL}{1000V_s f_s}\right]k_0 \\ &\quad + \left[t_c + (h_1 L + h_2)\left(\frac{d_t - d_s}{d_r} + 1\right)\right]k_0 \\ &\quad + \left[\frac{\pi DL}{1000V_t f_r}\left(\frac{d_t - d_s}{d_r}\right) + \frac{\pi DL}{1000V_s f_s}\right]\frac{t_e}{T_p}k_0 \\ &\quad + \left[\frac{\pi DL}{1000V_t f_r}\left(\frac{d_t - d_s}{d_r}\right) + \frac{\pi DL}{1000V_s f_s}\right]\frac{k_t}{T_p} \end{aligned} \tag{19}$$

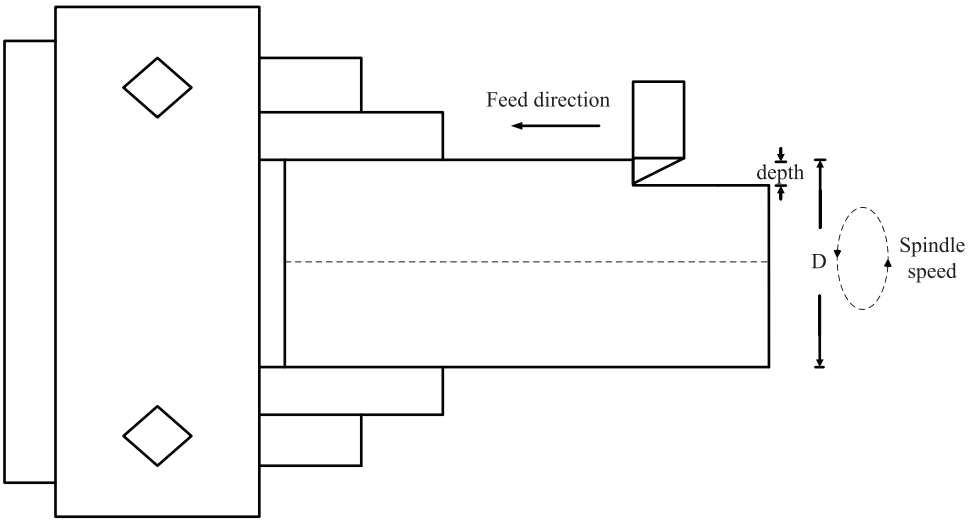

Figure 17: Schematic representation of a multi-pass turning operation

5.1.2 Machining Condition Constraints

To minimize the UC, practical machining constraints that describe the states of the machining procedure, i.e., finishing and rough machining, are taken into account. They are described in detail as follows [8,42]:

Rough Machining

The constraints of rough machining are listed as Eqs. (20)–(27):

$$\text{Range of cutting speed}: \ V_{rL} \le V_r \le V_{rU} \tag{20}$$

$$\text{Range of feed rate}: \ f_{rL} \le f_r \le f_{rU} \tag{21}$$

$$\text{Range of depth-of-cut}: \ d_{rL} \le d_r \le d_{rU} \tag{22}$$

$$\text{Tool life constraint}: \ T_L \le T_r \le T_U \tag{23}$$

$$\text{Maximum cutting force}: \ F_r = k_1 f_r^{\mu} d_r^{v} \le F_U \tag{24}$$

$$\text{Power constraint}: \ P_r = \frac{k_1 f_r^{\mu} d_r^{v} V_r}{6120\eta} \le P_U \tag{25}$$

The chip-tool interface temperature constraint is expressed as Eq. (26):

$$Q_r = k_2 V_r^{\tau} f_r^{\phi} d_r^{\delta} \le Q_U \tag{26}$$

$$\text{Stable cutting region constraint}: \ V_r^{\lambda} f_r d_r^{v} \ge SC \tag{27}$$

Finish Machining

The constraints of finish machining are listed as Eqs. (28)–(36):

$$\text{Range of cutting speed}: \ V_{sL} \le V_s \le V_{sU} \tag{28}$$

$$\text{Range of feed rate}: \ f_{sL} \le f_s \le f_{sU} \tag{29}$$

$$\text{Range of depth-of-cut}: \ d_{sL} \le d_s \le d_{sU} \tag{30}$$

$$\text{Tool life constraint}: \ T_L \le T_s \le T_U \tag{31}$$

$$\text{Maximum cutting force}: \ F_s = k_1 f_s^{\mu} d_s^{v} \le F_U \tag{32}$$

$$\text{Power constraint}: \ P_s = \frac{k_1 f_s^{\mu} d_s^{v} V_s}{6120\eta} \le P_U \tag{33}$$

The chip-tool interface temperature constraint is expressed as Eq. (26):

$$Q_s = k_2 V_s^{\tau} f_s^{\phi} d_s^{\delta} \le Q_U \tag{34}$$

$$\text{Stable cutting region constraint}: \ V_s^{\lambda} f_s d_s^{\upsilon} \ge SC \tag{35}$$

$$\text{Surface finishing constraint}: \ \frac{f_s^2}{8R} \le SR_U \tag{36}$$

Parameter Relations

The practical relationship between finish and rough machining can be given by Eqs. (37)–(40):

$$V_s \ge k_3 V_r \tag{37}$$

$$f_r \ge k_4 f_s \tag{38}$$

$$d_r \ge k_5 d_s \tag{39}$$

$$n = \frac{d_t - d_s}{d_r}, \text{and } n \in Z_+ \tag{40}$$

where the tool life can be formulated as Eq. (41).

$$T = \frac{C_0}{V^p f^q d^r} \tag{41}$$

It is supposed that the same tool is utilized during the whole machining operation procedure for both finishing and roughing. The wear rate of the cutter tools is commonly different between finishing and roughing due to changing machining conditions. Therefore, the life of the tool can be calculated as Eq. (42):

$$T_p = \theta T_r + (1 - \theta) T_s, \ \theta \in [0, 1] \tag{42}$$

where

$$T_r = \frac{C_0}{V_r^p f_r^q d_r^r} \tag{43}$$

$$T_s = \frac{C_0}{V_s^p f_s^q d_s^r} \tag{44}$$

In some previous research work, a simplified formula for T_p (Eq. (42)) was adopted by ignoring the weight factor θ, as given by Eq. (45):

$$T_p = T_r + T_s. \tag{45}$$

5.1.3 Pruning Strategies Using the Theoretical Lower Bound of the Subproblem

The machining parameters of multi-pass turning that need to be optimized include a single finish cut and multiple rough cuts. Therefore, the available quantity of rough cuts should be restricted to certain ranges by Eq. (46):

$$n_L \le n \le n_U, \tag{46}$$

where $n_L = \lceil (d_t - d_{sU})/d_{rU} \rceil$, $n_U = \lfloor (d_t - d_{sL})/d_{rL} \rfloor$ and n is a fixed integer.

Therefore, the total quantity of available values of n can be given as Eq. (47):

$$m = (n_U - n_L + 1), \tag{47}$$

where m is generally a small integer.

Hence, the issue of optimizing the cutting parameters of multi-pass turns is divided into m subproblems. Therefore, the search process for the entire optimization problem is equivalent to search processes of the m subproblems, and the solution corresponding to the optimal fitness value among the m subproblems is taken as the solution of the entire optimization problem. Inspired by pruning strategies, we have found that the theoretical lower bound on the unit production cost for each subproblem can be used to reduce the total running time during the enumerative process, as reported in our previous work [49]. The theoretical lower bound on the unit production cost can be calculated according to the different quantities of rough cuts for each subproblem [49], and the theoretical lower bound on UC for the j-th subproblem is represented by UC_{jL}. The flowchart of the GQBA for multi-pass turning optimization is given in Fig. 18.

The various notation used above is defined in Fig. 19.

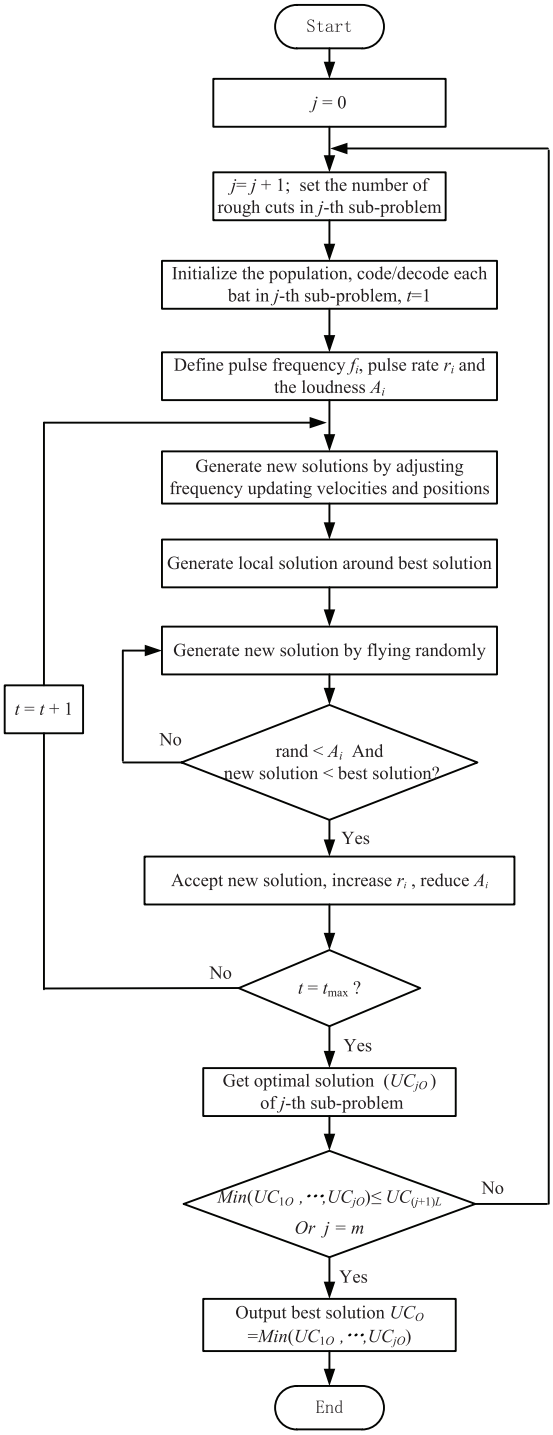

Figure 18: The flowchart of the GQBA for multi-pass turning optimization

UC	Unit production cost excluding material cost ($/piece)
k_0	Direct labor cost, including overhead ($/min)
k_t	Cutting edge cost ($/edge)
D, L	Diameter and length of work-piece (mm)
d_t	Depth of material to be removed (mm)
d_r, d_s	Depths of cut for each pass of rough and finish machining (mm)
d_{rL}, d_{rU}	Lower and upper bounds of depth of cut in rough machining (mm)
d_{sL}, d_{sU}	Lower and upper bounds of depth of cut in finish machining (mm)
n	Number of rough cuts, an integer
f_r, f_s	Feed rates in rough and finish machining (mm/rev)
f_{rL}, f_{rU}	Lower and upper bounds of feed rate in rough machining (mm/rev)
f_{sL}, f_{sU}	Lower and upper bounds of feed rate in finish machining (mm/rev)
V_r, V_s	Cutting speeds in rough and finish machining (m/min)
V_{rL}, V_{rU}	Lower and upper bounds of cutting speed in rough machining (m/min)
V_{sL}, V_{sU}	Lower and upper bounds of cutting speed in finish machining (m/min)
h_1, h_2	Constants relating to cutting tool travel and approach/departure time (min)
T, T_r, T_s	Tool life, expected tool life for rough and for finish machining (min)
T_P	Tool life of weighted combination of T_r and T_s (min)
θ	A weight for T_P, $0 \le \theta \le 1$
T_U, T_L	Upper and lower bounds for tool life (min)
p, q, r, C_0	Constants of tool-life equation
F_r, F_s	Cutting forces during rough and finish machining (kgf)
F_U	Maximum allowable cutting force (kgf)
k_1, μ, ν	Constants of cutting force equation
P_r, P_s	Cutting power during rough and finish machining (kW)
P_U	Maximum allowable cutting power (kW)
η	Power efficiency
λ, υ	Constants related to expression of stable cutting region
SC	Limit of stable cutting region constraint
Q_r, Q_s	Chip-tool interface temperatures during rough and finish machining (°C)
Q_U	Maximum allowable chip-tool interface temperature (°C)
k_2, τ, ϕ, δ	Constants related to chip-tool interface temperature equation
R	Nose radius of cutting tool (mm)
SR_U	Maximum allowable surface roughness (mm)
k_3, k_4, k_5	Constants for roughing and finishing parameter relations
t_c	Preparation time for loading and unloading time (min)
t_e	Time required to exchange a tool (min)
n_L, n_U	Lower and upper bounds of the number of rough cuts n
m	The total number of possible value of n, equals $n_U - n_L + 1$

Figure 19: List of symbols [48]

5.2 Experimental Verification and Comparisons

All the simulations were performed on a PC with the same characteristics as in Section 4. The numerical validation and comparisons using the machining model data [8,42] are shown in Table 6. We note that the tool life is usually defined by two different expressions (Eqs. (42) and (45)); thus, the proposed GQBA was tested using these two definitions. In each case, the GQBA was performed 50 times to obtain the average solution and optimized solution. The total number of iterations and the

total number of individuals in each test were set to 7500 and 100, respectively. The remaining parameter design of the GQBA is given in Table 3.

Table 6: Data for the machining model

Parameter	Value	Parameter	Value	Parameter	Value	Parameter	Value
D	50 mm	L	300 mm	d_t	6 mm	V_{rL}	50 m/min
f_{rL}	0.1 mm/rev	d_{rL}	1 mm	V_{rU}	500 m/min	f_{rU}	0.9 mm/rev
d_{rU}	3 mm	V_{sL}	50 m/min	f_{sL}	0.1 mm/rev	d_{sL}	1 mm
V_{sU}	500 m/min	f_{sU}	0.9 mm/rev	d_{sU}	3 mm	p	5
q	1.75	r	0.75	μ	0.75	ν	0.95
η	0.85	λ	2	υ	-1	τ	0.4
ϕ	0.2	δ	0.105	R	1.2 mm	C_0	6×10^{11}
T_L	25 min	T_U	45 min	F_U	200 Kgf	P_U	5 kW
SC	140	Q_U	1000°C	SR_U	10 μm	h_1	7×10^{-4}
h_2	0.3	t_e	1.5 min/edge	t_c	0.75 min/piece	k_t	2.5 $/edge
k_0	0.5 $/min	k_1	108	k_2	132	k_3	1
k_4	2.5	k_5	1				

Table 7 presents the results of the average unit production costs obtained by the GQBA using various mathematical models with different tool life expressions and cutting depths. We note that none of the solutions obtained in the experiment violate the practical machining constraints, suggesting that they are feasible. Moreover, it can be seen from this table that due to the low standard deviation, the solution obtained in each instance fluctuates within a small range, which indicates that the GQBA presented in this paper has good stability.

Table 7: Results obtained by the GQBA

Model	Depth-of-cut (mm)	Average UC *($/piece)*	Standard deviation	Function evaluations	Execution time (sec/run)
$T_p = T_r + T_s$	6	1.9602	0.00113	750, 000	95
$T_p = T_r + T_s$	8	2.4398	0.00153	750, 000	96
$T_p = \theta T_r + (1-\theta) T_s$	6	2.0284	0.00050	750, 000	98
$T_p = \theta T_r + (1-\theta) T_s$	8	2.5514	0.00172	750, 000	99

The optimization problem has been investigated by various approaches. To demonstrate the effectiveness of the proposed GQBA, we compare the simulation results with those reported in recent literature [5,8,9]. For a fair comparison, we select the results of other algorithms with consistent common parameters in these literatures. The detailed comparison results are summarized in Tables 8–11, where the best results are indicated by underlines.

Table 8: Comparison of the experimental results among different algorithms (when $T_p = T_r + T_s$, $d_t = 6$ mm)

Algorithm	V_r(m/min)	V_s(m/min)	f_r(mm/rev)	f_s(mm/rev)	d_r(mm)	d_s(mm)	UC($/piece)	Constraint violation
GQBA (Ours)	123.3238	169.9843	0.5654	0.2261	3	3	1.9592	0
FPA [8]	123.3431	169.9785	0.5655	0.2262	3	3	1.9591	0
COA [9]	123.1462	169.9876	0.5655	0.2262	3	3	1.959	0
HPSO [43]	123.3424	169.9783	0.5655	0.2262	3	3	1.959	0
GA [5]	122.42	161.08	0.56	0.21	3	3	2.038	0
PSO [16]	106.69	155.89	0.897	0.28	2	2	2.272	0
HRDE [48]	–	–	–	–	–	–	2.0461	–
SA-PS [42]	–	–	–	–	–	–	2.313	–
AIA [48]	–	–	–	–	–	–	2.12	–
DERE [44]	–	–	–	–	–	–	2.046	–
ABC [44]	–	–	–	–	–	–	2.118	–
DE [44]	–	–	–	–	–	–	2.136	–
HABC [45]	–	–	–	–	–	–	2.046	–
HRTLBO [46]	–	–	–	–	–	–	2.046	–
ACO [14]	103.05	162.02	0.9	0.24	–	–	1.626	not considering Eq. (40)
FA [47]	98.4102	162.2882	0.82	0.2582	3	3	1.824	Eq. (24)

Note: "–" denotes that the authors do not provide the specific value in their works.

Table 9: Comparison of the experimental results among different algorithms (when $T_p = T_r + T_s$, $d_t = 8$ mm)

Algorithm	V_r(m/min)	V_s(m/min)	f_r(mm/rev)	f_s(mm/rev)	d_r(mm)	d_s(mm)	UC($/piece)	Constraint violation
GQBA (Ours)	119.1607	164.2276	0.6562	0.2624	2.6673	2.6652	2.4385	0
HRDE [48]	–	–	–	–	–	–	2.4791	–
DERE [44]	–	–	–	–	–	–	2.4793	–
HABC [45]	–	–	–	–	–	–	2.4790	–
AIA [48]	–	–	–	–	–	–	2.51	–
ABC [44]	–	–	–	–	–	–	2.503	–
DE [44]	–	–	–	–	–	–	2.512	–
SS [10]	–	–	–	–	–	–	2.5417	–
SA-PS [42]	–	–	–	–	–	–	2.7411	–

Note: "–" denotes that the authors do not provide the specific value in their works.

From Table 8, it can be clearly seen that the GQBA presented in this paper outperforms many algorithms, and its performance is comparable with that of some previously best approaches such as the COA [9], hybrid particle swarm optimization (HPSO) [43] and the FPA [8]. All four methods can produce an optimal result of ∼**1.959**. In contrast, the performance of the remaining methods is much worse than that of the GQBA; the corresponding production costs are larger than **2.0** or the resulting solutions are infeasible. In Table 9, the GQBA can obtain the minimum production cost among all the algorithms when the depth-of-cut is 8 mm. The proposed GQBA can further reduce the production cost obtained by hybrid robust differential evolution (HRDE) [48], the differential evolution algorithm with receptor editing (DERE) [44] and the hybrid ABC (HABC) [45], and it can obtain much better

solutions than the other five comparative methods. In general, the GQBA can reduce costs by 2% to 12% compared to previously reported methods.

Table 10: Comparison of the experimental results among different algorithms (when $T_p = \theta T_r + (1 - \theta) T_s$, $d_t = 6$ mm)

Algorithm	V_r(m/min)	V_s(m/min)	f_r(mm/rev)	f_s(mm/rev)	d_r(mm)	d_s(mm)	UC($/piece)	Constraint violation
GQBA (Ours)	109.6672	169.9682	0.5655	0.2261	3	3	2.0279	0
FPA [8]	109.6631	169.9785	0.5655	0.2262	3	3	2.0351	0
HPSO [7]	109.6655	169.9796	0.5655	0.2262	3	3	2.0351	0
COA [9]	117.9322	123.1993	0.5655	0.2262	3	3	2.2390	0

Table 11: Optimal result obtained by the GQBA (when $T_p = \theta T_r + (1 - \theta) T_s$, $d_t = 8$ mm)

Algorithm	V_r(m/min)	V_s(m/min)	f_r(mm/rev)	f_s(mm/rev)	d_r(mm)	d_s(mm)	UC($/piece)	Constraint violation
GQBA (Ours)	106. 0599	164.2371	0.6563	0.2624	2.6671	2.6656	2.5494	0

As mentioned previously, different definitions have been proposed for the tool life expression in previous studies [5,8,9], i.e., $T_p = \theta T_r + (1 - \theta) T_s$. Because the formulation of tool life plays an important role in computing the production cost, the application effect of the GQBA in this case is also considered, as shown in Tables 10 and 11. From Table 10, it is obvious that the GQBA can obtain a minimum production cost of **2.0279**, which outperforms all other methods. The optimal results and machining parameters are summarized in Table 11 with a depth-of-cut of 8 mm, which, as far as we know, has not been stated in previously published literature. In addition, this is the first study to use a BA variant to reduce the unit production cost for multi-pass turning operations, and our results show that the proposed GQBA can address the optimization problem efficiently and achieve better results than other methods.

5.3 Discussion

The GQBA can find better results than the other algorithms to further cut down the unit production cost. The main reasons come from the following two aspects.

Firstly, we improve the original BA algorithm to form the novel GQBA. The first improvement is to incorporate the current optimal positions of quantum bats and the global best position into the stochastic attractor to facilitate population diversification. The second improvement is to use a Gaussian distribution instead of the uniform distribution to update the positions of the quantum-behaved bats, thus performing a more accurate search and avoiding premature convergence. All these are verified by the numerical simulation experiments in Section 4.

Secondly, to overcome the complicated optimization problems in various fields, we need to carefully consider the characteristics of the specific problem and use the specific domain knowledge to design the optimization algorithm. In this paper, for the optimization problem of multi-pass turning, because the machining process can be divided into different numbers of roughing cuts, we decompose the whole optimization problem into several simple sub-problems according to the different numbers of roughing cuts. Each sub-problem can be conquered individually, which greatly reduces the space of the problem solution.

Therefore, the performance of the combination of traditional divide-and-conquer strategy and swarm intelligence algorithm is better than other algorithms that only use traditional mathematical methods or swarm intelligence algorithms. However. the proposed GQBA may be suitable to solve the optimization problem of multi-pass turning, but it is not general as other algorithms, that is, they can also be applied to the optimization problems in other manufacturing fields. Therefore, the generality of our algorithm is insufficient, and we hope to improve it further in the future.

6 Conclusions

In this research, the GQBA is developed to promote the efficiency of the BA and QBA with respect to accuracy and stability. In the GQBA, the combination of the QBA and a Gaussian distribution can expand the search space and prevent premature convergence. The modification of the stochastic attractor can contribute to promoting swarm diversity. The GQBA also inherits the characteristics of the QBA, such as simplicity and feasibility. To conclude, the experimental results of numerical functions verify the effectiveness of the GQBA. In addition, the results of multi-pass turning operation optimization show that the GQBA is a good alternative method.

For future research directions, the proposed GQBA can be utilized with the angle modulation technique [50–52] to solve binary optimization problems. Moreover, the GQBA can be applied to other real-world problems, including artificial neural networks, task scheduling, feature selection and image segmentation.

Acknowledgement: The data that supports the findings of this study are authentic and reliable. And the authors appreciated the reviewers for their helpful suggestions which greatly improved the presentation of this paper.

Funding Statement: This research was supported by the the National Natural Science Foundation of Fujian Province of China (2020J01697, 2020J01699).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Chandrasekaran, M., Muralidhar, M., Krishna, C. M., Dixit, U. S. (2010). Application of soft computing techniques in machining performance prediction and optimization: A literature review. *International Journal of Advanced Manufacturing Technology, 46(5–8),* 445–464. DOI 10.1007/s00170-009-2104-x.
2. Sibalija, T. V. (2019). Particle swarm optimisation in designing parameters of manufacturing processes: A review (2008–2018). *Applied Soft Computing, 84,* 105743. DOI 10.1016/j.asoc.2019.105743.
3. Agapiou, J. S. (1992). The optimization of machining operations based on a combined criterion, Part 1: The use of combined objectives in single-pass operations. *Journal of Engineering for Industry, 114(4),* 500–507. DOI 10.1115/1.2900704.
4. Duffuaa, S. O., Shuaib, A. N., Alam, M. (1993). Evaluation of optimization methods for machining economics models. *Computers & Operations Research, 20(2),* 227–237. DOI 10.1016/0305-0548(93)90077-V.
5. Sofuoğlu, M. A., Çakır, F. H., Gürgen, S. (2019). An efficient approach by adjusting bounds for heuristic optimization algorithms. *Soft Computing, 23(13),* 5199–5212. DOI 10.1007/s00500-018-3327-2.
6. Miodragovi, G. R., Orevi, V., Bulatovi, R. R., Petrovi, A. (2019). Optimization of multi-pass turning and multi-pass face milling using subpopulation firefly algorithm. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 233(5),* 1520–1540.

7. Costa, A., Fichera, S. (2018). Response to cuckoo optimization algorithm for unit production cost in multi-pass turning operations [int j advmanuftechnol (2015) 76 (1): 647–656]. *The International Journal of Advanced Manufacturing Technology, 94(1),* 57–58. DOI 10.1007/s00170-017-0550-4.
8. Xu, S., Wang, Y., Huang, F. (2017). Optimization of multi-pass turning parameters through an improved flower pollination algorithm. *The International Journal of Advanced Manufacturing Technology, 89(1),* 503–514.
9. Mellal, M. A., Williams, E. J. (2015). Cuckoo optimization algorithm for unit production cost in multi-pass turning operations. *The International Journal of Advanced Manufacturing Technology, 76(1),* 647–656. DOI 10.1007/s00170-014-6309-2.
10. Chen, M. (2004). Optimizing machining economics models of turning operations using the scatter search approach. *International Journal of Production Research, 42(13),* 2611–2625. DOI 10.1080/00207540410001666251.
11. Onwubolu, G. C., Kumalo, T. (2001). Optimization of multipass turning operations with genetic algorithms. *International Journal of Production Research, 39(16),* 3727–3745. DOI 10.1080/00207540110056153.
12. Chen, M. C., Chen, K. Y. (2003). Optimization of multipass turning operations with genetic algorithms: A note. *International Journal of Production Research, 41(14),* 3385–3388. DOI 10.1080/0020754031000118143.
13. Sankar, R. S., Asokan, P., Saravanan, R., Kumanan, S., Prabhaharan, G. (2007). Selection of machining parameters for constrained machining problem using evolutionary computation. *The International Journal of Advanced Manufacturing Technology, 32(9–10),* 892–901. DOI 10.1007/s00170-006-0420-y.
14. Vijayakumar, K., Prabhaharan, G., Asokan, P., Saravanan, R. (2003). Optimization of multi-pass turning operations using ant colony system. *International Journal of Machine Tools & Manufacture, 43(15),* 1633–1639. DOI 10.1016/S0890-6955(03)00081-6.
15. Wang, Y. C. (2007). A note on 'optimization of multi-pass turning operations using ant colony system'. *International Journal of Machine Tools & Manufacture, 47(12–13),* 2057–2059. DOI 10.1016/j.ijmachtools.2007.03.001.
16. Srinivas, J., Giri, R., Yang, S. (2009). Optimization of multi-pass turning using particle swarm intelligence. *The International Journal of Advanced Manufacturing Technology, 40(1),* 56–66. DOI 10.1007/s00170-007-1320-5.
17. Yildiz, A. R. (2009). A novel particle swarm optimization approach for product design and manufacturing. *The International Journal of Advanced Manufacturing Technology, 40(5–6),* 617–628. DOI 10.1007/s00170-008-1453-1.
18. Raja, S. B., Baskar, N. (2010). Optimization techniques for machining operations: A retrospective research based on various mathematical models. *International Journal of Advanced Manufacturing Technology, 48(9–12),* 1075–1090. DOI 10.1007/s00170-009-2351-x.
19. Yan, X., Liu, Y., Xu, Y., Jia, M. (2020). Multistep forecasting for diurnal wind speed based on hybrid deep learning model with improved singular spectrum decomposition. *Energy Conversion and Management, 225,* 113456. DOI 10.1016/j.enconman.2020.113456.
20. Yan, X., She, D., Xu, Y., Jia, M. (2021). Deep regularized variational autoencoder for intelligent fault diagnosis of rotor-bearing system within entire life-cycle process. *Knowledge-Based Systems, 226,* 107142. DOI 10.1016/j.knosys.2021.107142.
21. Yildiz, A. R. (2009). Hybrid immune-simulated annealing algorithm for optimal design and manufacturing. *International Journal of Materials & Product Technology, 34(3),* 217–226. DOI 10.1504/IJMPT.2009.024655.
22. Yang, X. -S. (2010). A new metaheuristic bat-inspired algorithm. In: *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pp. 65–74. Berlin, Heidelberg: Springer.
23. Mirjalili, S. (2019). Genetic algorithm. In: *Evolutionary algorithms and neural networks*, pp. 43–55. Cham: Springer.

24. Van Laarhoven, P. J., Aarts, E. H. (1987). Simulated annealing. In: *Simulated annealing: Theory and applications*, pp. 7–15. Dordrecht: Springer.
25. Jayabarathi, T., Raghunathan, T., Gandomi, A. (2018). The bat algorithm, variants and some practical engineering applications: A review. *Nature-Inspired Algorithms and Applied Optimization*, pp. 313–330. Cham: Springer.
26. Jordehi, A. R. (2015). Chaotic bat swarm optimisation (CBSO). *Applied Soft Computing, 26,* 523–530. DOI 10.1016/j.asoc.2014.10.010.
27. Lin, J. H., Chou, C. W., Yang, C. H., Tsai, H. L. (2012). A chaotic levy flight bat algorithm for parameter estimation in nonlinear dynamic biological systems. *Computer and Information Technology, 2(2),* 56–63.
28. Yilmaz, S., Kucuksille, E. U. (2013). Improved bat algorithm (iba) on continuous optimization problems. *Lecture Notes on Software Engineering, 1(3),* 279. DOI 10.7763/LNSE.2013.V1.61.
29. Xie, J., Zhou, Y., Chen, H. (2013). A novel bat algorithm based on differential operator and lévy flights trajectory. *Computational Intelligence and Neuroscience, 2013*.
30. Wang, G., Guo, L. (2013). A novel hybrid bat algorithm with harmony search for global numerical optimization. *Journal of Applied Mathematics, 2013,* 696491. DOI 10.1155/2013/696491.
31. Meng, X. B., Gao, X. Z., Liu, Y., Zhang, H. (2015). A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization. *Expert Systems with Applications, 42(17–18),* 6350–6364. DOI 10.1016/j.eswa.2015.04.026.
32. Clerc, M., Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation, 6(1),* 58–73. DOI 10.1109/4235.985692.
33. Cai, X., Wang, L., Kang, Q., Wu, Q. (2014). Bat algorithm with Gaussian walk. *International Journal of Bio-Inspired Computation, 6(3),* 166–174. DOI 10.1504/IJBIC.2014.062637.
34. dos Santos Coelho, L. (2010). Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. *Expert Systems with Applications, 37(2),* 1676–1683. DOI 10.1016/j.eswa.2009.06.044.
35. Liu, J., Xu, W., Sun, J. (2005). Quantum-behaved particle swarm optimization with mutation operator. *17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)*. Hong Kong, China, IEEE.
36. Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems, 89,* 228–249. DOI 10.1016/j.knosys.2015.07.006.
37. Chen, K., Zhou, F., Liu, A. (2018). Chaotic dynamic weight particle swarm optimization for numerical function optimization. *Knowledge-Based Systems, 139,* 23–40. DOI 10.1016/j.knosys.2017.10.011.
38. Mirjalili, S., Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95,* 51–67. DOI 10.1016/j.advengsoft.2016.01.008.
39. Zhu, B., Zhu, W., Liu, Z., Duan, Q., Cao, L. (2016). A novel quantum-behaved bat algorithm with mean best position directed for numerical optimization. *Computational Intelligence and Neuroscience, 2016*. DOI 10.1155/2016/6097484.
40. James, K., Russell, E. (1995). Particle swarm optimization. *Proceedings of 1995 IEEE International Conference on Neural Networks*, Perth, WA, Australia.
41. Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S. (2009). Gsa: A gravitational search algorithm. *Information Sciences, 179(13),* 2232–2248. DOI 10.1016/j.ins.2009.03.004.
42. Chen, M. C., Tsai, D. (1996). A simulated annealing approach for optimization of multi-pass turning operations. *International Journal of Production Research, 34(10),* 2803–2825. DOI 10.1080/00207549608905060.
43. Costa, A., Celano, G., Fichera, S. (2011). Optimization of multi-pass turning economies through a hybrid particle swarm optimization technique. *The International Journal of Advanced Manufacturing Technology, 53(5),* 421–433. DOI 10.1007/s00170-010-2861-6.

44. Yildiz, A. R. (2012). A comparative study of population-based optimization algorithms for turning operations. *Information Sciences, 210,* 81–88. DOI 10.1016/j.ins.2012.03.005.
45. Yildiz, A. R. (2013). Optimization of cutting parameters in multi-pass turning using artificial bee colony-based approach. *Information Sciences, 220,* 399–407. DOI 10.1016/j.ins.2012.07.012.
46. Yildiz, A. R. (2013). Optimization of multi-pass turning operations using hybrid teaching learning-based approach. *The International Journal of Advanced Manufacturing Technology, 66(9),* 1319–1326. DOI 10.1007/s00170-012-4410-y.
47. Belloufi, A., Assas, M., Rezgui, I. (2014). Intelligent selection of machining parameters in multipass turnings using firefly algorithm. *Modelling and Simulation in Engineering, 2014,* 592627.
48. Yildiz, A. R. (2013). Hybrid taguchi-differential evolution algorithm for optimization of multi-pass turning operations. *Applied Soft Computing, 13(3),* 1433–1439. DOI 10.1016/j.asoc.2012.01.012.
49. Xie, S., Guo, Y. (2012). Optimisation of machining parameters in multi-pass turnings using ant colony optimisations. *International Journal of Machining and Machinability of Materials, 11(2),* 204–220. DOI 10.1504/IJMMM.2012.045983.
50. Leonard, B. J., Engelbrecht, A. P., Cleghorn, C. W. (2015). Critical considerations on angle modulated particle swarm optimisers. *Swarm Intelligence, 9(4),* 291–314. DOI 10.1007/s11721-015-0114-x.
51. Pampara, G. (2013). *Angle modulated population based algorithms to solve binary problems (Ph.D. Thesis)*. University of Pretoria.
52. Huang, X., Li, P., Pu, Y. (2019). Amplitude angle modulated bat algorithm with application to zero-one knapsack problem. *IEEE Access, 7,* 27957–27969. DOI 10.1109/ACCESS.2019.2901988.