A modified whale optimization algorithm for parameter estimation of software reliability growth models

Kezhong Lu¹,² and Zongmin Ma¹

Abstract
Software reliability growth models are nonlinear in nature, so it is difficult to estimate the proper parameters. An estimation method based on a modified whale optimization algorithm in which parameters are estimated is discussed in this paper. The whale optimization algorithm is a new swarm intelligence optimization algorithm. This algorithm is not perfect enough. Based on the analysis of whale optimization algorithm, we point out the disadvantages of whale optimization algorithm, and propose a modified whale optimization algorithm algorithm from four aspects: choice regarding the dimension, exploration control, encircling prey modified, and candidate solution selection. The experimental results based on 34 benchmark functions demonstrate that the proposed modified whale optimization algorithm has better accuracy. The modified whale optimization algorithm is used to predict software reliability by predicting the faults during the software testing process using software faults' historical data. The proposed modified whale optimization algorithm shows significant advantages in handling a variety of modeling problems such as the exponential model, power model, delayed s-shaped model, and modified sigmoid model. Experimental results show that the fitting accuracy of the modified sigmoid model is minimal on three data sets. The modified whale optimization algorithm with the modified sigmoid model can provide a better estimate of the software faults.

Keywords
Software reliability growth models, parameter estimation, whale optimization algorithm, modified sigmoid model

Introduction
With the development of computer technology, all kinds of software systems have permeated our modern society, involving wristwatches, telephones, home appliances, buildings, automobiles, aircraft, and other key areas, and play an increasingly important role. An important issue in developing such software systems is to produce a high-quality software system that satisfies user requirements. Therefore, software reliability is becoming more and more important to researchers. The reliability of a software system can be defined as: “The probability of failure-free software operation for a specified period of time in a specified environment.” Software reliability modeling is one of the important fields of theoretical research and engineering practice for software reliability. Various software reliability growth models (SRGMs) have been formulated over the years but according to an observation made by Lyu, “There is no universally acceptable model that can be trusted to give accurate results in all circumstances; users should not trust claims to the contrary.” Every model has some advantages and some disadvantages; so, the choice regarding which model to follow for reliability prediction depends upon the requirements of the software.

The fault prediction process in SRGMs depends on representing the relationship between the time span of software testing and cumulative number of errors detected. Most SRGMs, known in history, have two or three parameters to be estimated. They include the expected number of failures in the software by the end of the testing process and the initial failure intensity. A number of unknown parameters such as the expected number of failures and the failure intensity are estimated using either the least square or maximum likelihood estimation techniques. Model parameters are normally in nonlinear relationships. This means that traditional

¹College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
²College of Computer Science, Chizhou University

Corresponding author:
Zongmin Ma, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106.
Email: zongminma@nuaa.edu.cn

Creative Commons Non Commercial CC BY-NC: This article is distributed under the terms of the Creative Commons Attribution-NonCommercial 4.0 License (https://creativecommons.org/licenses/by-nc/4.0/) which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access page (https://us.sagepub.com/en-us/nam/open-access-at-sage).
parameter estimation techniques suffer from many problems in finding the best set of parameters to tune the model for better prediction.6 The whale optimization algorithm (WOA) is a relatively new meta-heuristic optimization technique proposed by Mirjalili and Lewis,7 which is inspired by the bubble-net hunting of humpback whales. WOA is easy to implement and has few adjustment parameters, which make it superior to particle swarm optimization (PSO), grey wolf optimizer (GWO), gravitational search algorithm, and so on.8 As a result, WOA has attracted much attention and has been applied to handle many practical engineering application problems, such as training multilayer perceptrons in neural networks,9 tracking the maximum power point (MPP) of photovoltaic systems,10 optimizing the active and reactive power dispatch problem,11 feature selection,12 parameter estimation,13 resource allocation,14 and so on. Fundamentally, WOA is free from restrictive assumptions on the continuity, the existence of derivatives, the unimodality, and so on. It looks very attractive to use WOA for estimating the parameters of SRGMs.

In this paper, we analyze WOA and propose a modified WOA (MWOA). MWOA is used to predict software reliability by predicting the faults during the software testing process using software fault historical data. Moreover, an modified sigmoid model (MSM) is developed for a better estimate of the software faults. The rest of the paper is organized as follows: the next section presents a brief review of the works carried out in the area of software reliability prediction in research. The section “software reliability growth models” provides an overview of various SRGMs. The section “whale optimization algorithm” proposes the WOA briefly, and an MWOA algorithm is proposed in the following section. The section “performance test for MWOA” tests MWOA on 34 benchmark functions. In the section “Experimental evaluation,” detailed experiments results are provided. The final section concludes this paper.

Related work

As most of the SRGM are nonlinear in nature, the basic two methods (maximum likelihood and least square) are not suitable.15 Neural networks and Bayesian learning are introduced for SRGMs’ parameter estimations.16,17 Interest in using evolutionary computation to solve prediction and software engineering problems expanded in the recent years.

Aljahdali and El-Telbany18 explored genetic algorithms (GAs) as an alternative approach to derive these SRGMs. Souza and Vergilio19 and Su and Huang20 proposed an artificial neural-network-based approach for software reliability estimation and modeling. However, neural networks-based models were not easily interpreted; so, some other techniques were needed to be explored. Another technique inspired by GAs and named genetic programming was proposed to obtain software reliability model (SRM) for forecasting the reliability by Oliveira et al.21 Zhang et al.22 applied a PSO algorithm, for the estimation, but it was observed that the searching range is too large, the convergence speed is slow and accuracy is not high. The use of fuzzy logic to build an SRGM has also been explored by Aljahdali and Sheta,23 who proposed a fuzzy model, which consists of a collection of linear sub-models joined together smoothly using fuzzy membership functions to represent the fuzzy model. Sharma and Pant24 proposed a modified version of the artificial bee colony (ABC) to estimate the parameters of SRGMs. Alaa and Abdel-Raouf25 explored the advantages of the GWO algorithm in estimating the SRGM’s parameters. Choudhary and Baghel26 used cuckoo search optimization to find the effectiveness of the reliability model.

An approach based on an ant colony algorithm was proposed by Zheng et al.27 to estimate parameters of SRMs. Zhou et al.28 proposed a harmony search algorithm to estimate parameters of the SRM. Afterward Xiao and Dohi29 studied the stochastic behavior of software fault detection process. It was shown that stochastic behavior-based SRMs outperformed the existing SRMs. After this, various stochastic techniques were used for parameter estimation of various SRMs, which gave better results than that of other traditional or evolutionary techniques. WOA is also a kind of swarm intelligence algorithm-based stochastic technique. In this paper, we explore the use of WOA and the proposed MWOA to predict the faults during the software testing process using software fault historical data. Detailed results are provided to explore the advantages of using MWOA in solving this problem.

Software reliability growth models

An SRGM can be considered as one of the fundamental techniques to assess the reliability of software quantitatively. Any SRM presents a mathematical function which illustrates defect detection rates. These models are classified in two categories: concave and S-shaped models.3 Concave models are so-called because they bend downward. S-shaped models, however, are first convex and then concave. The two models have the same asymptotic behavior: The defect-detection rate decreases as the number of defects detected increases, and the total number of defects detected asymptotically approaches a finite value. Numerous researchers proposed various SRGMs to predict the quality of software components; three well-known SRGMs used in our study are discussed as follows.

Exponential model (EXPM)

This is a continuous time-independent and identical error behavior model. This model was first introduced by Goel and Okumoto30 and is also known as the GO model.

\[ \mu(t; \beta) = \beta_0(1 - e^{-\beta_1t}) \] (1)
The equations which govern the relationship between the time $t$ and both $\mu(t; \beta)$ and $\lambda(t; \beta)$ are:

$$\mu(t; \beta) = \beta_0 e^{-\beta_1 t}$$

$$\lambda(t; \beta) = \beta_0 \beta_1^t e^{\beta_1 t - 1}$$

**Power model (POWM)**

The POWM is provided by Crow.\(^{31}\) It is a graphical approach to perform analysis of reliability growth data and is simple and straightforward to understand. The model is based on the nonhomogeneous Poisson process model. The equations which govern the relationship between the time $t$ and both $\mu(t; \beta)$ and $\lambda(t; \beta)$ are:

$$\mu(t; \beta) = \beta_0^t$$

$$\lambda(t; \beta) = \beta_0^t e^{\beta_1 t - 1}$$

**Delayed S-shaped model (DSSM)**

This model describes the software reliability process as a DSSM.\(^{32}\) Here, the per fault failure distribution is gamma. The number of failures per time period, however, is a Poisson type model using the classification scheme of Musa and Okumoto rather than considered as binomial. This model is also a finite failure model. The system equations for $\mu(t; \beta)$ and $\lambda(t; \beta)$ are:

$$\mu(t; \beta) = \beta_0^t (1 - (1 + \beta_1 t e^{-\beta_1 t}) \right)$$

$$\lambda(t; \beta) = \beta_0 \beta_1^t e^{\beta_1 t - 1}$$

**Modified sigmoid model**

There is no universally acceptable model that can be trusted to give accurate results in all circumstances; users should not trust claims to the contrary. Every model has some pros and cons; so, the choice regarding which model to follow for reliability prediction depends upon the requirements of the software.\(^{3}\) Generally speaking, SRMs always go through three stages of generation, development and maturity, and the speed of each stage is different. For S-shaped models, usually at the stage of generation, the speed of change is slow; in the stage of development, the speed of change is quicker; in the mature stage, the speed of change is also slow. There is no universally acceptable single model that can be trusted to give accurate results in most or all applications. In view of the different growth characteristics at each stage of the model, a MSM is proposed in this paper to better fit the data and improve the accuracy of the model.

A sigmoid function is a mathematical function having a characteristic S-shaped or sigmoid curve. Sigmoid function shown in Figure 1 and defined by the formula (8) is often used in neural networks and logistic regression. For the SRMs, the sigmoid function is modified as formula (9).

$$S(t) = \frac{1}{1 - e^{-t}}$$

$$\mu(t; \beta) = \frac{\beta_0}{\beta_2 + e^{-\beta_3 t} + \beta_3}$$

where $\beta_2$ is associated with the amplitude of the curve $\mu$, $\beta_1$ is associated with the steepness of the curve, $\beta_2$ is associated with the amplitude, and the sigmoid’s midpoint, $\beta_3$ is associated with the $Y$-axis intercept.

**Whale optimization algorithm**

The WOA is a nature-inspired meta-heuristic optimization algorithm proposed by Mirjalili and Lewis in 2016.\(^{7}\) WOA mimics the bubble-net feeding in the foraging behavior of the humpback whales. The humpback whales hunt close to the surface with trapping the prey in distinctive bubbles along a circle or 9-shaped path. The bubble-net hunting model is shown in Figure 2, and the mathematical model of WOA is described in three sections: encircling prey, bubble net hunting method, and search the prey.

**Encircling prey**

Humpback whales can recognize the location of prey and encircle them. Since the position of the optimal design in the search space is not known a priori, WOA expects that the present best candidate solution is the objective prey, and the other search agents will hence try to update their
positions toward the best search agent. This behavior is mathematically formulated as:

\[ \overrightarrow{D} = \mid \overrightarrow{C} \cdot \overrightarrow{X'}(t) - \overrightarrow{X}(t) \mid \] (10)

\[ \overrightarrow{X}(t + 1) = \overrightarrow{X'}(t) - \overrightarrow{A} \cdot \overrightarrow{D} \] (11)

where \( t \) indicates the current iteration, \( \overrightarrow{X'} \) is the position vector of the best solution obtained so far, \( \overrightarrow{X} \) is the position vector, \( \mid \) is the absolute value, and \( \cdot \) is an element-by-element multiplication. \( \overrightarrow{A} \) and \( \overrightarrow{C} \) are coefficient vectors that are calculated as follows:

\[ \overrightarrow{A} = 2\overrightarrow{a} \cdot \overrightarrow{r} - \overrightarrow{a} \] (12)

\[ \overrightarrow{C} = 2\overrightarrow{r} \] (13)

where \( \overrightarrow{a} \) is linearly decreased from 2 to 0 over the course of iterations (in both exploration and exploitation phases, and can be calculated by equation (14) and \( \overrightarrow{r} \) is a random vector in \([0,1]\).

\[ \overrightarrow{a} = 2(1 - t / MaxIter) \] (14)

where \( t \) is the iteration number and \( MaxIter \) is the maximum number of allowed iterations.

**Bubble-net attacking method (exploitation phase)**

Two strategies are utilized to figure the feeding conduct of humpback whales as follows.

**Shrinking circling system.** This behavior is achieved by decreasing the value of \( a \) from 2 to 0 in equation (14) over the course of iterations. The new position of a search agent can be defined anywhere in between the original position of the agent and the position of the current best agent by setting random values for \( \overrightarrow{A} \) in \([-1, 1]\).

**Spiral redesigning position.** A spiral equation is then created between the position of whale and prey to mimic the helix-shaped movement of humpback whales as follows:

\[ \overrightarrow{X}(t + 1) = \overrightarrow{D} \cdot e^{bl} \cdot \cos 2\pi l + \overrightarrow{X'}(t) \] (15)

where \( \overrightarrow{D} = \mid \overrightarrow{X'}(t) - \overrightarrow{X}(t) \mid \) and indicates the distance of the \( i \)th whale to the prey, \( b \) is a constant for defining the shape of the logarithmic spiral, \( l \) is a random number in \([-1,1]\), and \( \cdot \) is an element-by-element multiplication.

In hunting whales swim around the prey in above two paths simultaneously. To update the whales’ positions, 50% probability is taken for the above two methods.

\[ \overrightarrow{X}(t + 1) = \begin{cases} \overrightarrow{X'}(t) - \overrightarrow{A} \cdot \overrightarrow{D} & \text{if } p < 0.5 \\ \overrightarrow{D} \cdot e^{bl} \cdot \cos (2\pi l) + \overrightarrow{X'}(t) & \text{if } p \geq 0.5 \end{cases} \] (16)

where \( p \) is a random number in \([0,1]\).

**Search for prey (exploration phase)**

To have a global optimizer, a randomly chosen solution is used to update the position instead of the best search agent found so far. So, a vector \( \overrightarrow{A} \) with the random value \( >1 \) or \( < -1 \) is used to force a solution to move far away from the best-known search agent. This mechanism can be mathematically modeled as in equations (17) and (18).

\[ \overrightarrow{D} = \mid \overrightarrow{C} \cdot \overrightarrow{X}_{rand}(t) - \overrightarrow{X}(t) \mid \] (17)

\[ \overrightarrow{X}(t + 1) = \overrightarrow{X}_{rand}(t) - \overrightarrow{A} \cdot \overrightarrow{D} \] (18)

where \( \overrightarrow{X}_{rand}(t) \) is a random position vector (a random whale) chosen from the current population. Figure 3 shows the flow-chart of the proposed algorithm. For further details, the reader may refer to the work by Mirjalili and Lewis.\(^7\)

**A modified whale optimization algorithm**

Although WOA has several characteristics, there still exist its limitation for some complex problems, especially the phenomenon of premature convergence leading to local optimization and the deficiency of global search ability, which affect the performance of WOA. Through analyzing the basic WOA, four improvements have been made to improve the method for enhancing search performance and search quality, and an MWOA is proposed in this paper. The details are as follows.

**Choosing dimensions**

There is a probability \( p \) to choose between the shrinking encircling mechanism or the spiral model to update the position of whales in the basic WOA. The choice probability \( p \) between the above two operations is the same for a whale during optimization. That is to say between the
shrinking encircling update and spiral update, only one is selected for a whale. For some complex problems, this choice is difficult to well balance between exploitation and exploration. The choice probability \( p \) changes in each dimension of a whale in this paper. The probability \( p \) varies uniformly and randomly between 0 and 1, and the probability in each dimension is independent. So there is a probability \( p \) to choose shrinking encircling update or spiral update in each dimension of a whale for enhancing the search performance of WOA (see step 10 in Figure 4).

**Exploration control**

The exploration control by the random number \( A \) in WOA, when \(|A| \geq 1\) WOA emphasizes exploration and allow performing a global search. \( A \) is a random value in the interval \([-a,a]\) where \( a \) is decreased from 2 to 0 over the course of iterations according to equation (14). According to equation (12), when \( t > \text{MaxIter}/2 \), so \( t/\text{MaxIter} > 1/2 \), \( 1 - t/\text{MaxIter} < 1/2 \), \( 2(1 - t/\text{MaxIter}) < 1 \), according to equation (14), \( a < 1 \), and \(|A|\leq a < 1\). The range of \( A \) is shown in Figure 5(a). So, when \( t > \text{MaxIter}/2 \) there is only encircling prey without search for prey. However, search for prey is an exploration phase; it helps to avoid premature convergence. To improve exploration ability of the later iteration, we define a new control parameter \( B \). The parameter \( B \) is calculated as follows:

\[
B = (a + 2) \cdot r - 2 \tag{19}
\]

where \( a \) is linearly decreased from 2 to 0 by equation (14) and \( r \) is a random number in \([0,1]\).

So \( B \) is a random value in the interval \([-2, a]\). \( B \) can take positive or negative values by the random number \( r \) during the whole search process. The range of \( B \) is shown in Figure 5(b). A random search agent is chosen when \( B \geq 0 \) (exploration phase), while the best solution is selected when \( B < 0 \) for updating the position of the search agents (exploitation phase). According to Figure 5(b), the search agents still have the opportunity to explore in the later iteration. The exploration decreases linearly with iterative \( t \), which is advantageous to algorithm convergence (see steps 8 and 12 in Figure 4).

**Modified encircling prey**

In the encircling prey section of WOA, the search agents update their positions toward the best search agent. It is easy to make the algorithm trap into the local solution and lead to premature convergence. Hence, to enable all agents to get rid of the stagnation easily, a modified encircling prey is proposed as follows:

\[
X_i(t+1) = X_{r_1} + 0.5(X_{r_2} - X_{r_3}) \tag{20}
\]

where \( r_1, r_2, r_3 \) are randomly chosen agents, and \( r_1 \neq r_2 \neq r_3 \neq i \). This is similar to the differential evolution (DE) algorithm.
Candidate solution selection

In WOA, the newly generated candidate solutions are accepted without selection. This may cause the rebound of new search agents, which will reduce the convergence speed and the convergence accuracy. To decide whether or not it should become a member of generation \( t + 1 \), a trial vector \( \vec{X}'(t + 1) \) is introduced and equations (15), (18), (20) are altered to equations (21)–(23). Then, \( \vec{X}'(t + 1) \) is compared to the target vector \( \vec{X}(t) \) using the greedy criterion. If vector \( \vec{X}'(t + 1) \) yields a smaller cost function value than \( \vec{X}(t) \), then \( \vec{X}(t + 1) \) is set to \( \vec{X}'(t + 1) \); otherwise, the old value \( \vec{X}(t) \) is retained (see steps 23–27 in Figure 4).

\[
\vec{X}'(t + 1) = X_{r1} + 0.5(X_{r2} - X_{r1}) \tag{21}
\]

\[
\vec{X}'(t + 1) = D' \cdot e^{bl} \cdot \cos 2\pi l + \vec{X}'(t) \tag{22}
\]

\[
\vec{X}'(t + 1) = \vec{X}_{rand}(t) - \vec{A} \cdot \vec{D} \tag{23}
\]

Performance test for MWOA

In this section, we compare the proposed MWOA with WOA based on the following 34 benchmark functions.

Benchmark functions

In this paper, MWOA is used to estimate the parameters of the SRGM. The parameters to be estimated are less than 10 in this paper. Therefore, the benchmark functions are chosen whose dimensions are not more than 10. A total of 34 benchmark functions are chosen from Doğan and Ölmez\(^\text{33}\) and they are only briefly described in Table 1. A more detailed description of these functions can be found in Doğan and Ölmez\(^\text{33}\) where the functions include many different types of problems, such as unimodal, multimodal, regular, irregular, separable, non-separable, and multidimensional.

| No. | Function            | C | Range          | d |
|-----|---------------------|---|----------------|---|
| 1   | Beale               | UN | [−4.5, 4.5]   | 5 |
| 2   | Easom               | UN | [−100, 100]   | 2 |
| 3   | Matyas              | UN | [−10, 10]     | 2 |
| 4   | Colville            | UN | [−10, 10]     | 4 |
| 5   | Trid6               | UN | [−d^2, d^2]   | 6 |
| 6   | Trid10              | UN | [−d^2, d^2]   | 10|
| 7   | Zakharov            | UN | [−5, 10]      | 10|
| 8   | Foxholes            | MS | [−65.536, 65.536] | 2 |
| 9   | Branin              | MS | [−5, 10]×[0, 15]| 2 |
| 10  | Bohachevsky1        | MS | [−100, 100]   | 2 |
| 11  | Booth               | MS | [−10, 10]     | 2 |
| 12  | Michalewicz2        | MS | [0, π]        | 2 |
| 13  | Michalewicz5        | MS | [0, π]        | 5 |
| 14  | Michalewicz10       | MS | [0, π]        | 10|
| 15  | Schaffer            | MN | [−100, 100]   | 2 |
| 16  | Six Hump Camel Back | MN | [−5, 5]       | 2 |
| 17  | Bohachevsky2        | MN | [−100, 100]   | 2 |
| 18  | Bohachevsky3        | MN | [−100, 100]   | 2 |
| 19  | Shubert             | MN | [−10, 10]     | 2 |
| 20  | Goldstein-Price     | MN | [−2, 2]       | 2 |
| 21  | Kowalik             | MN | [−5, 5]       | 4 |
| 22  | Shekel5             | MN | [0, 10]       | 4 |
| 23  | Shekel7             | MN | [0, 10]       | 4 |
| 24  | Shekel10            | MN | [0, 10]       | 4 |
| 25  | Perm                | MN | [−d, d]       | 4 |
| 26  | PowerSum            | MN | [0, d]        | 4 |
| 27  | Hartman3            | MN | [0, 1]        | 3 |
| 28  | Hartman6            | MN | [0, 1]        | 6 |
| 29  | Langerman2          | MN | [0, 10]       | 2 |
| 30  | Langerman5          | MN | [0, 10]       | 5 |
| 31  | Langerman10         | MN | [0, 10]       | 10|
| 32  | Fletcher Powell2    | MN | [−π, π]       | 2 |
| 33  | Fletcher Powell5    | MN | [−π, π]       | 5 |
| 34  | Fletcher Powell10   | MN | [−π, π]       | 10|

D: dimension; C: characteristics; U: unimodal; M: multimodal; S: separable; N: non-separable.

Algorithm settings

As mentioned before in the previous section, WOA and MWOA are parameter-free algorithms. There are no additional parameters except some common parameters of swarm intelligence algorithm, such as population size and maximum iteration number. For all the algorithms, a population size and maximum iteration number equal to 30 and 500 have been utilized. All results reported are obtained based on 30 independent runs. For clarity, the results of the best algorithms are marked in boldface.

Test for four improved WOA algorithms

For the basic WOA, there are four improvements to enhance the search performance and search quality. The first is “choice on dimension,”
named WOA1. The second is “exploration control,” named WOA2. The third is “modified encircling prey,” named WOA3. The last is “candidate solution selection,” named WOA4. We tested the four methods with \( p = 0.5 \), \( n = 30 \), and \( Maxiter = 500 \) on 10 functions. To compare mean WOA1, WOA2, WOA3, WOA4 and WOA, Wilcoxon rank-sum tests (5% significance level) are performed. The signs \( - \), \(+\), and \( =\), respectively, indicate that the comparison algorithm is inferior, superior, and equivalent to the basic WOA. The experiment results are listed in Table 2. From Table 2, the performance of the basic WOA algorithm is improved obviously by the four methods.

**Effect of different \( p \) value**

A probability \( p \) was used to switch between the shrinking encircling mechanism and the spiral model to update the position of whales in WOA and MWOA. The different probability \( p \) will have an impact on the search performance of MWOA. In this section, we tested MWOA with \( p = \{0.1, 0.3, 0.5, 0.7, 0.9\} \), \( n = 30 \), and \( Maxiter = 500 \) on 10 functions. The experiment results are listed in Table 3. From Table 3, it is found that MWOA can obtain better values when \( p = 0.9 \). Therefore, in this paper, the probability \( p \) is set to 0.9.

**Test results**

The overall performances of the algorithms are studied for a constant number of iterations. After a certain number of iterations, the algorithms are evaluated according to the mean, standard deviation and best fitness values found for each benchmark function. The statistical result is reported in Table 4. The \( h \) values are the results of Wilcoxon rank-sum tests (5% significance level). From Table 4, it is found that the best values of MWOA can achieve or approach the optimal solutions of all 34 benchmark functions. The best mean values of MWOA are less than those of WOA for functions \( f_1, f_2, f_4, f_6, f_7, f_{11}, f_{13}, f_{14}, f_{18}, f_{21}, f_{22}, f_{23}, f_{24}, f_{25}, f_{26}, f_{28}, f_{31}, f_{32}, f_{33}, f_{34}, \) and equal or equivalent to those of WOA for the rest of the functions except \( f_3 \). The mean values of MWOA are less than or equal to those of WOA for 33 benchmark functions, and greater than those of WOA only for functions \( f_3 \). General speaking, it can be concluded that as a whole the proposed MWOA significantly improves the basic WOA.

In addition, we also compared the performance of our algorithm with that of PSO, \(^{34}\) ABC, \(^{35}\) and DE. \(^{36}\) The parameter values were set as follows: \( c_1 = c_2 = 2 \) for PSO, \( F = 0.5 \), and \( CR = 0.9 \) for DE, \( limit = 20 \) for ABC. Experiments were executed for the same population sizes and switch probabilities established in previous tests. All experiments were run 30 times, independently. The mean error and its deviation were under investigation. A summary of obtained results is provided in Table 5.
Table 3. Comparison between different values on 10 problems.

| No. | Min. | Mean ± SD | Mean ± SD | Mean ± SD |
|-----|------|-----------|-----------|-----------|
|     |      |           |           |           |
| p=0.9 | 0     | 1.16 ± 1.18 | 1.16 ± 1.25 | 1.15 ± 0.95 |
| p=0.7 | 0.13 | 0.93 ± 0.97 | 0.93 ± 0.98 | 0.92 ± 0.95 |
| p=0.5 | 0.15 | 0.79 ± 0.77 | 0.79 ± 0.79 | 0.78 ± 0.75 |
| p=0.3 | 0.19 | 0.61 ± 0.59 | 0.61 ± 0.61 | 0.60 ± 0.57 |
| p=0.1 | 0.23 | 0.43 ± 0.42 | 0.43 ± 0.43 | 0.42 ± 0.40 |

To compare the performance of different algorithms, the results are labeled by the “−,” “+”, and “≈.” The “−” denotes our algorithm is better than other algorithms. And the “+” and “≈” denote our algorithm is inferior to, equal to other algorithms. This comparison is between PSO, ABC, and DE with our algorithm MWOA. In Table 5, we can find that MWOA can perform better than other algorithms in most of the cases. Among these algorithms used for comparison, PSO is weaker in performance. It cannot provide better solution compared with MWOA algorithm. For the unimodal function f1–f7, MWOA can obtain the six better solutions than other algorithms except f3. For f3, DE can give the better solution than other algorithms. For multimodal functions f8–f34 with many local minima, the final results are more important because of these functions can reflect the algorithm’s ability to escape from the poor local optimal and obtain the near-global optimum. MWOA provides better solutions for some functions except f3, f14, f22, f24, f25, f28, f32, and f34. For these eight functions, the optimal solution can be obtained by ABC and DE. All in all, it can be observed that MWOA is inferior to, equal to, superior to compared algorithm in 11, 31, and 60 cases, respectively, out of the 102 cases. In general, the performance of MWOA is highly competitive with other algorithms in solving these problems.

**Experimental evaluation**

To develop our new technique for solving the parameter estimation problem of SRGM, we used WOA and the proposed MWOA. Our objective is to estimate the parameters such that the error difference, between the actual fault and the estimated faults based on the parameter estimated using WOA or MWOA, is minimal. RMSE criterion is used to measure the performance. It is just the square root of the mean square error as shown in equation (24) given below:

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2}
\]

where \( N \) represents the number of measurements used for estimating the model parameters, \( y_i \) represents the \( i \)th value of the effort, and \( \hat{y}_i \) is the estimated effort. The performances of WOA and MWOA are evaluated by computing the objective evaluation criterion in terms of the fitness.

**Test/debug data sets**

Three real measured test/debug data sets are applied in our proposed approaches. The first data set of 109 measurements was presented in the work by Jeet et al. and Alaa and Abdel-Raouf.\(^{17,25}\) The test/debug data was measured day by day, and includes the reading of the measured
| No. | Min. | WOA        | MWOA       |
|-----|------|------------|------------|
| f₁  | 0    | 0.076207   | 0.07318    |
|     | f₂  | -1         | -0.96666   |
|     | f₃  | 2.9737 × 10⁻¹⁹ | 2.9737 × 10⁻²⁹ |
|     | f₄  | 0.96666    | 0.96666    |
|     | f₅  | -50        | -50        |
|     | f₆  | -210       | -210       |
|     | f₇  | 0.998      | 168.35     |
|     | f₈  | 0.398      | 0.47767    |
|     | f₉  | 0.998      | 168.35     |
|     | f₁₀ | 0.001519   | 0.00039965 |
|     | f₁₁ | -1.8013    | -1.8013    |
|     | f₁₂ | -4.6877    | -4.6877    |
|     | f₁₃ | -9.6602    | -9.6602    |
|     | f₁₄ | 0.0045341  | 0.0045341  |
|     | f₁₅ | -1.0316    | -1.0316    |
|     | f₁₆ | 0.001      | 0.001      |
|     | f₁₇ | 10.4       | 10.4       |
|     | f₁₈ | -186.73    | -186.73    |
|     | f₁₉ | 3          | 3          |
|     | f₂₀ | 0.0031     | 0.00039965 |
|     | f₂₁ | -10.15     | -10.15     |
|     | f₂₂ | -10.4      | -10.4      |
|     | f₂₃ | -10.53     | -10.53     |
|     | f₂₄ | 0         | 0.001923   |
|     | f₂₅ | 0          | 0          |
|     | f₂₆ | 0          | 0.001923   |
|     | f₂₇ | -3.32      | -3.32      |
|     | f₂₈ | -3.0812    | -3.0812    |
|     | f₂₉ | -0.89238   | -0.89238   |
|     | f₃₀ | -1.008     | -1.008     |
|     | f₃₁ | -1.5       | -1.5       |
|     | f₃₂ | 0          | 0.015103   |
|     | f₃₃ | 0          | 2.1518 × 10⁻⁰⁷ |
|     | f₃₄ | 0          | 12.803     |

WOA: whale optimization algorithm; MWOA: modified whale optimization algorithm.

Table 4. Comparison between WOA and MWOA.
Table 5. Comparison between MWOA and other methods.

| No. | Min. | PSO Mean ± SD | ABC Mean ± SD | DE Mean ± SD | MWOA Mean ± SD |
|-----|------|---------------|--------------|-------------|----------------|
| f₁  | 0    | 1.72 x 10⁻²⁴ ± 4.13 x 10⁻²⁴ | 0.0021 ± 0.0031 | 0 ± 0 | 0 ± 0 |
| f₂  | -1   | 1 ± 0⁺⁻ | 0.093 ± 0.092 | 1.08 x 10⁻¹²⁸ ± 4.52 x 10⁻¹²⁸ | 1.25 x 10⁻⁴⁵ ± 6.41 x 10⁻⁵⁵ |
| f₃  | 0    | 1.45 x 10⁻²⁶ ± 3.09 x 10⁻²⁶ | 4.01 x 10⁻⁵ ± 8.66 x 10⁻⁵ | 1.13 ± 2.03 | 4.69 ± 0.047 |
| f₄  | 0    | 0.11 ± 0.082 | 1.38 ± 0.88 | 50.1 ± 1.12 x 10⁻⁷ | 50.2 ± 5.7 x 10⁻¹⁴ |
| f₅  | -50  | -50 ± 1.14 x 10⁻⁷ | -49.85 ± 0.10 | -209.5 ± 1.0149 | -210 ± 2.66 x 10⁻⁷ |
| f₆  | -210 | -108.08 ± 372 | -182.23 ± 20.263 | 0.00041 ± 0.0020 | 0.998 ± 0 |
| f₇  | 0    | 8.99 ± 22.61 | 31.34 ± 12.47 | 1.09 ± 0.54 | 1.65 x 10⁻¹⁰ ± 6.79 x 10⁻¹⁰ |
| f₈  | 0.998 | 0.998 ± 2.23 x 10⁻¹⁶ | 461.05 ± 118.49 | 0.398 ± 0 | 0.998 ± 0 |
| f₉  | 0.398 | 0.398 ± 0 | 0.398 ± 2.27 x 10⁻¹⁵ | 0 ± 0 | 0.398 ± 0 |
| f₁₀ | 0    | 0 ± 0 | 0 ± 0 | 0 ± 0 | 0 ± 0 |
| f₁₁ | 0    | 1.19 x 10⁻²⁷ ± 3.45 x 10⁻²⁷ | 2.28 x 10⁻⁵ ± 6.33 x 10⁻⁷ | 0 ± 0 | 0 ± 0 |
| f₁₂ | 1.8013 | -1.305 ± 0.25 | -1.8013 ± 1.02 x 10⁻¹⁵ | -1.8013 ± 9.03 x 10⁻¹⁶ | -1.8013 ± 9.03 x 10⁻¹⁶ |
| f₁₃ | 4.6877 | -2.1594 ± 0.30 | -4.6877 ± 1.28 x 10⁻⁵ | -4.6504 ± 0.047 | -4.6662 ± 0.07 |
| f₁₄ | -9.6602 | -3.201 ± 0.34 | -9.4625 ± 0.11 | -8.8998 ± 0.57 | -9.1973 ± 0.36 |
| f₁₅ | 0    | 0.0032 ± 0.0046 | 0.0060 ± 0.0038 | 0.0016 ± 0.0036 | 0.00064 ± 0.0024 |
| f₁₆ | -1.0316 | -1.0316 ± 4.51 x 10⁻¹⁶ | -1.0316 ± 5.48 x 10⁻¹⁶ | 0 ± 0 | -1.0316 ± 4.51 x 10⁻¹⁶ |
| f₁₇ | 0    | 0 ± 0 | 2.14 x 10⁻⁹ ± 1.17 x 10⁻⁸ | 0 ± 0 | 0 ± 0 |
| f₁₈ | 0    | 7.71 ± 8.65 x 10⁻⁵ | 7.71 ± 8.65 x 10⁻⁵ | 0 ± 0 | 0 ± 0 |
| f₁₉ | -186.73 | -186.73 ± 1.66 x 10⁻¹⁴ | -186.73 ± 0.00084 | -186.73 ± 3.94 x 10⁻¹⁴ | -186.73 ± 3.87 x 10⁻¹⁴ |
| f₂₀ | 3    | 3 ± 2.22 x 10⁻¹⁵ | 3 ± 1.9 x 10⁻¹⁵ | 3 ± 1.9 x 10⁻¹⁵ | 3 ± 2.02 x 10⁻¹⁵ |
| f₂₁ | 0.00031 | 0.0094 ± 0.0093 | 0.0010 ± 0.0026 | 0.0038 ± 0.0075 | 0.00055 ± 0.00041 |
| f₂₂ | -10.15 | -1.08 ± 0.67 | -10.11 ± 0.71 | -9.48 ± 2.07 | -9.30 ± 1.93 |
| f₂₃ | -10.4 | -1.05 ± 0.67 | -10.37 ± 0.79 | -10.22 ± 0.96 | -10.40 ± 1.04 x 10⁻⁶ |
| f₂₄ | -10.53 | -1.15 ± 0.36 | -10.47 ± 0.19 | -9.77 ± 2.34 | -9.72 ± 2.10 |
| f₂₅ | 0    | 29.88 ± 163.49 | 0.38 ± 0.37 | 0.057 ± 0.12 | 0.070 ± 0.14 |
| f₂₆ | 0    | 31.13 ± 27.80 | 0.037 ± 0.026 | 0.0015 ± 0.0027 | 0.00076 ± 0.0028 |
| f₂₇ | -3.86 | -3.55 ± 0.23 | -3.86 ± 2.20 x 10⁻¹⁵ | -3.86 ± 2.71 x 10⁻¹⁵ | -3.86 ± 2.71 x 10⁻¹⁵ |
| f₂₈ | -3.32 | -1.65 ± 0.49 | -3.32 ± 3.86 x 10⁻⁹ | -3.24 ± 0.058 | -3.26 ± 0.063 |
| f₂₉ | -1.08 | -0.61 ± 0.34 | -1.08 ± 3.69 x 10⁻⁵ | -1.08 ± 4.51 x 10⁻⁹ | -1.08 ± 4.51 x 10⁻¹⁶ |
| f₃₀ | -1.5 | -0.20 ± 0.21 | -1.25 ± 0.28 | -1.32 ± 0.28 | -1.46 ± 0.14 |
| f₃₁ | NA  | -0.00019 ± 0.00087 | -0.44 ± 0.18 | -0.53 ± 0.27 | -0.63 ± 0.32 |
| f₃₂ | 0    | 70.52 ± 215.21 | 5.08 x 10⁻⁶ ± 2.78 x 10⁻⁶ | 0 ± 0 | 2.69 x 10⁻²⁵ ± 1.47 x 10⁻⁲⁸ |
| f₃₃ | 0    | 532.87 ± 879.21 | 16.10 ± 13.23 | 224.93 ± 552.01 | 102.91 ± 33.51 |
| f₃₄ | 0    | 5907 ± 3862 | 825 ± 436 | 2013 ± 2571 | 828 ± 1226 |

PSO: particle swarm optimization; ABC: artificial bee colony; DE: differential evolution; MWOA: modified whale optimization algorithm.
Table 6. The results of WOA and MWOA.

| Data | Model | RMSE of WOA |  | RMSE of MWOA |
|------|-------|-------------|----------------|----------------|
|      |       | Mean        | SD         | Best           | Mean           | SD           | Best           |
| 1    | EXPM  | 28.749      | 0.0575     | 28.702         | 28.701         | 1.08 × 10⁻¹⁴ | 28.701         |
| 1    | POWM  | 41.670      | 0.6719     | 41.334         | 41.334         | 7.92 × 10⁻¹⁵ | 41.334         |
| 1    | DSSM  | 14.935      | 0.0306     | 14.903         | 14.902         | 8.88 × 10⁻¹⁵ | 14.902         |
| 1    | MSM   | 9.819       | 0.682      | 9.208          | 9.208          | 2.86 × 10⁻¹⁵ | 9.208          |
| 2    | EXPM  | 28.124      | 0.0345     | 28.102         | 28.102         | 6.33 × 10⁻¹⁵ | 28.102         |
| 2    | POWM  | 47.943      | 0.2760     | 47.686         | 47.686         | 1.32 × 10⁻¹⁵ | 47.686         |
| 2    | DSSM  | 18.055      | 0.0034     | 18.052         | 18.052         | 1.62 × 10⁻¹⁵ | 18.052         |
| 2    | MSM   | 39.58       | 49.93      | 17.032         | 17.030         | 9.94 × 10⁻¹⁵ | 17.030         |
| 3    | EXPM  | 12.281      | 0.3949     | 12.161         | 12.161         | 3.75 × 10⁻¹⁵ | 12.161         |
| 3    | POWM  | 12.451      | 0.2762     | 12.277         | 12.277         | 3.57 × 10⁻¹⁵ | 12.277         |
| 3    | DSSM  | 18.604      | 0.0339     | 18.582         | 18.582         | 2.95 × 10⁻¹⁵ | 18.582         |
| 3    | MSM   | 18.850      | 20.263     | 12.080         | 12.080         | 4.90 × 10⁻⁵  | 12.080         |

WOA: whale optimization algorithm; MWOA: modified whale optimization algorithm; POWM: power model; DSSM: delayed S-shaped model; MSM: modified sigmoid model; EXPM: exponential model; RMSE: square root of the mean square error; SRGM: software reliability growth model.

Table 7. SRGMs with parameter estimated using MWOA—109 measurements.

| Model                     | Parameter                          |
|---------------------------|------------------------------------|
| Exponential model (EXPM)  | $\mu(t; \beta) = 713.8678(1 - e^{-0.015055t})$ |
| Power model (POWM)       | $\mu(t; \beta) = 24.52470(0.48022t)$ |
| Delayed S-shaped model (DSSM) | $\mu(t; \beta) = 563.7610(1 - (1 + 0.049256t))e^{-0.049256t}$ |
| Modified sigmoid model (MSM) | $\mu(t; \beta) = 78.492/(0.13253 + e^{-0.064557t}) - 53.449$ |

POWM: power model; DSSM: delayed S-shaped model; MSM: modified sigmoid model; EXPM: exponential model; SRGM: software reliability growth model; MWOA: modified whale optimization algorithm.

Table 8. SRGMs with parameter estimated using MWOA—111 measurements.

| Model                     | Parameter                          |
|---------------------------|------------------------------------|
| Exponential model (EXPM)  | $\mu(t; \beta) = 538.0712(1 - e^{-0.025751t})$ |
| Power model (POWM)       | $\mu(t; \beta) = 45.0802(0.53169t)$ |
| Delayed S-shaped model (DSSM) | $\mu(t; \beta) = 488.1190(1 - (1 + 0.066293t))e^{-0.066293t}$ |
| Modified sigmoid model (MSM) | $\mu(t; \beta) = 201.31/(0.31167 + e^{-0.064861t}) - 160.67$ |

POWM: power model; DSSM: delayed S-shaped model; MSM: modified sigmoid model; EXPM: exponential model; SRGM: software reliability growth model; MWOA: modified whale optimization algorithm.

faults ($u_i$), the accumulated faults ($y_i$), and the number of test workers ($w_i$) involved in the testing process. The second data set of 111 measurements presented in the work by Alaa and Abdel-Raouf, is used for our experiments. The third data set has 46 measurements and was presented in the work by Zheng et al., . The number of measurements collected during the testing process is small. This represents a difficulty for traditional parameter estimation techniques.

**Model parameter estimation**

Our experiments explore the use of WOA and MWOA methods to estimate the parameters of three software projects using three known SRGMs (EXPM, POWM, and DSSM) and an MSM in this section. In our case, the range of the parameters for EXPM, POWM, and DSSM models are set as

![Figure 6. Best so far convergence curve for the modified whale optimization algorithm (MWOA) using single models on 109 measurements.](Image)
follows: $\beta_0 \in [0, 2000]$, $\beta_1 \in [0, 1]$. The range of the parameters for 
MSM: $\beta_1 \in [0, 1]$ for all data sets, $\beta_0$, $\beta_2$, $\beta_3 \in [-100, 100]$ for 
data1, $\beta_0$, $\beta_2$, $\beta_3 \in [-300, 300]$ for data2, and $\beta_0$, $\beta_3 \in 
[500, 1000]$, $\beta_2 \in [-10, 10]$ for data3. For both WOA and 
MWOA, a population size and maximum iteration number 
equal to 40 and 1000 have been utilized. The result reported 
in Table 6 is obtained based on 30 independent runs. The 
best estimated parameters for the SRGM are given with the 
model equations in Tables 6–8. The convergence process 
was recorded by keeping an eye on the best individual after 
each generation.

From Table 6, it is found that the best RMSEs of 
MWOA are slightly less than those of WOA on data1 + 
EXPM, data1 + POWM, data2 + POWM, and data2 + 
MSM. They are same in other instances. The average
Table 9. SRGMs with parameter estimated using MWOA—46 measurements.

| Exponential model (EXPM) | Power model (POWM) | Delayed S-shaped model (DSSM) | Modified sigmoid model (MSM) |
|--------------------------|--------------------|-------------------------------|----------------------------|
| μ(t; β) = 423.6390(1 − e^{−0.02318t}) | μ(t; β) = 16.619(0.74306)^t | μ(t; β) = 280.3165((1 − (1 + 0.097213t))e^{−0.097213t}) | μ(t; β) = 698.89((−1.7014 + e^{−0.0668049t}) − 1000 |

SRGM: software reliability growth models; MWOA: modified whale optimization algorithm; EXPM: exponential model; POWM: power model; DSSM: delayed S-shaped model; MSM: modified sigmoid model.

Table 10. The best RMSEs of MWOA and other algorithms.

| Data | Model   | MWOA | GWO | PSO | DABC |
|------|---------|------|-----|-----|------|
| 1    | EXPM    | 28.701 | 28.706 | 67.976 | – |
| 1    | POWM    | 41.334 | 41.334 | 86.281 | – |
| 1    | DSSM    | 14.902 | 14.912 | 19.721 | – |
| 2    | EXPM    | 28.102 | 28.102 | 49.544 | 134.89 |
| 2    | POWM    | 47.686 | 51.194 | 87.272 | 332.33 |
| 2    | DSSM    | 18.052 | 18.085 | 19.760 | 98.8429 |
| 3    | EXPM    | 12.161 | 12.161 | 13.121 | – |
| 3    | POWM    | 12.277 | 12.281 | 12.627 | – |
| 3    | DSSM    | 18.582 | 18.583 | 28.962 | – |

MWOA: modified whale optimization algorithm; RMSE: square root of the mean square error; EXPM: exponential model; POWM: power model; DSSM: delayed S-shaped model.

RMSEs of MWOA are less than those of WOA in all instances. That shows the average optimization ability of MWOA is better than that of WOA. The standard deviations of MWOA are small in all instances. That shows MWOA is robust, and MWOA can overcome WOA’s shortcoming that is easy to relapse into local best and premature convergence.

In Figures 6, 7, and 8, we show the best so far convergence curves of the evolutionary process obtained by MWOA for the four developed models EXPM, PowM, DSSM, and MSM on three data sets. The actual and accumulated faults (failures) curves for the four developed models are shown in Figures 9, 10, and 11.

For three data sets, we find that the MSM model is able to provide the best results using MWOA tuned parameters. The model error is the minimum compared to other proposed models. That shows MSM is a good SRGM and can fit data well.

The results between MWOA and PSO, dichotomous artificial bee colony (DABC), and GWO are shown in Tables 9 and 10. The RMSEs of MWOA and GWO are obviously less than those of PSO and DABC. Compared to other algorithms, MWOA obtains the best results.

Conclusion and future work

In this paper, we explored the use of WOA to estimate the parameters of SRGMs. WOA is a new swarm intelligence optimization algorithm. This algorithm is not perfect enough. Based on the analysis of WOA, we point out the disadvantages of WOA, and propose an MWOA algorithm. Several benchmark functions are tested and the experimental results show that the proposed MWOA significantly improves the basic WOA. We also compared the performance of our algorithm with that of PSO, ABC, and DE; the performance of MWOA is also highly competitive with other algorithms.

We use MWOA to estimate the parameters of three known SRGMs (i.e. EXPM, PowM, and S-shaped model). Experimental results show that MWOA outperforms WOA, GWO, PSO, and DABC. MWOA gets minimum RMSEs.

An SRGM (MSM) is proposed in this paper. MWOA is used to estimate MSMs’ parameters on three data sets. Experimental results show that the fitting accuracy of the MSM model is better than that of three known models. The MSM model can provide a better estimate of the software faults.

For future work, we plan to explore using MWOA in developing a multi-stage hybrid structure growth model which can provide an advanced relationship between μ(t; β) and λ(t;β) to better model the software reliability prediction process. MWOA can build a multi-stage hybrid structure with a set of parameters which can provide an accurate estimate of the software faults.

Declaration of conflicting interests

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The authors disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Natural Science Foundation of China (Grant no. 61772269), and the key project of Anhui natural science research (Grant no. KJ2016A514).

ORCID iD

Kezhong Lu https://orcid.org/0000-0001-7803-4639

References

1. Lyu MR Handbook of software reliability engineering. McGraw-Hill: IEEE Computer Society Press, 1996.
2. ANSI/IEEE. IEEE standard glossary of data management terminology. ANSI/IEEE, 1991.
3. Bidhan K and Awasthi A. A review on parameter estimation techniques of software reliability growth models. *Int J Comput Appl Technol Res* 2014; 3: 267–272.

4. ALRahameleon Z, Reyat M, Sheta AF, et al. A New software reliability growth model: genetic-programming-based approach. *Int J Softw Eng Appl* 2011; 4: 476–481.

5. Alaa S. Reliability growth modeling for software fault detection using particle swarm optimization. In: Proceedings of the IEEE international conference on evolutionary computation, 2006, pp.3071–3078.

6. Alweshah M, Ahmed W and Aldabbas H. Evolution of software reliability growth models: a comparison of auto-regression and genetic programming models. *Int J Comput Appl* 2015; 125: 20–25.

7. Mirjallili S and Lewis A. The Whale Optimization Algorithm. *Adv Eng Softw* 2016; 95: 51–67.

8. Zhong M and Long W. Whale Optimization Algorithm with nonlinear control parameter. *MATEC Web Conf* 2017; 139: 1–5.

9. Bhesaditya RH, Jangir P, Jangir N, et al. Training multi-layer perceptron in neural network using Whale Optimization Algorithm. *Indian J Sci Technol* 2016; 7: 1–15.

10. Kumar CHS and Rao RS. A novel global MPP tracking of photovoltaic system based on Whale Optimization Algorithm. *Int J Renew Energy Dev* 2016; 5: 225–232.

11. Medani KBO, Sayah S, Bekrar A, et al. Whale Optimization Algorithm based optimal reactive power dispatch: a case study of the Algerian Power System. *Electr Power Syst Res* 2018; 163: 696–705.

12. Agrawal R, Kaur B and Sharma S. Quantum based Whale Optimization Algorithm for wrapper feature selection. *Appl Soft Comput* 2020; 89: 106692.

13. Cao Y, Li Y, Zhang G, et al. An efficient terminal voltage control for PEMFC based on an improved version of whale optimization algorithm. *Energy Rep* 2020; 6: 530–542.

14. Pham Q, Mirjallili S, Kumar N, et al. Whale optimization algorithm with applications to resource allocation in wireless networks. *IEEE Trans Veh Technol* 2020; 69: 4285–4297.

15. Shanmugam L and Florence L. A comparison of parameter best estimation method for software reliability models. *Int J Softw Eng Appl* 2012; 3: 91–102.

16. Zeng H and Rine D. Estimation of software defects fix effort using neural networks. In: *Proceedings of the 28th international computer software and applications*, Hong Kong, 2004, pp.20–21.

17. Jeet K, Dhir R and Verma H. A comparative study of Bayesian and fuzzy approach to assess and predict maintainability of the software using activity-based quality model. *SIGSOFT Softw Eng Notes* 2012; 37: 1–9.

18. Aljahdali SH and El-Telbany ME. Genetic Algorithms for optimizing ensemble of models in software reliability prediction. *ICGST-AIML J* 2008; 8: 5–12.

19. Souza GA and Vergilio SR. Modeling software reliability growth with artificial neural networks. In: *IEEE Latin American test workshop*, Buenos Aires, Argentina, 2006, pp.165–170.

20. Su YS and Huang CY. Neural-network-based approaches for software reliability estimation using dynamic weighted combinatorial models. *J Syst Softw*. 2007; 80: 606–615.

21. Oliveira E, Silia C, Pozo A, et al. Modeling software reliability growth with genetic programming. In: Proceedings of the 16th IEEE international symposium on software reliability engineering, Chicago, 2005, pp.171–180.

22. Zhang K, Li A and Song B. Estimating parameters of software reliability models using PSO. *Comput Eng Appl* 2008; 44: 47–49. (in Chinese).

23. Aljahdali S and Sheta AF. Predicting the reliability of software systems using fuzzy logic. In: *Proceedings of 2011 eighth international conference on information technology: New generations*, Las Vegas, 2011, pp.36–40.

24. Sharma TK, Pant M and Abraham A. Dichotomous search in ABC and its application in parameter estimation of software reliability growth models. In: *Proceedings of nature and biologically inspired computing*, Salamanka, 2011, pp.207–212.

25. Alaa F and Abdel-Raouf A. Estimating the parameters of software reliability growth models using the grey wolf optimization algorithm. *Int J Adv Comput Sci Appl* 2016; 7: 499–505.

26. Choudhary A and Baghel AS. Software reliability prediction using cuckoo search optimization, empirical mode decomposition, and ARIMA model: CS-EEMD-ARIMA based SRGM. *Int J Open Source Softw Process* 2016; 7: 39–54.

27. Zheng C, Liu X and Huang S. Estimating parameters of software reliability models by ant colony algorithm. *J Comput Appl* 2012; 32: 1147–1151.

28. Zhou Y, Qian L and Li J. Estimating parameters of software reliability model by harmony search algorithm. *J Shandong Univ Technol* 2017; 331: 44–48.

29. Xiao X and Dohi T. Estimating software reliability using extreme value distribution. In: *Proceedings of software engineering, business continuity, and education*, Jeju Island, 2011, pp.399–406.

30. Geol A and Okumoto K. Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE Trans Reliab* 1979; 28: 206–211.

31. Crow LH. Reliability for complex repairable systems. Reliability and biometry. Edited by Proshan F and Serfling RJ, SIAM, Philadelphia, PA, 1974, pp.379–410.

32. Yamada S, Ohba M and Osaki S. S-Shaped reliability growth modeling for software error detection. *IEEE Trans Reliability* 2009; 32: 475–484.

33. Doğan B and Ölmez T. A new metaheuristic for numerical function optimization: Vortex Search algorithm. *Inf Sci* 2015; 293: 125–145.

34. Kennedy J and Eberhart RC. Particle swarm optimization. In: Proceedings of the 1995 IEEE international conference on neural networks, Perth, 1995, pp.1942–1948.

35. Karaboga D. An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005, pp.1–10.

36. Storn R and Price KV. Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces. *J Global Optim* 1997; 11: 341–359.