Conditioning optimization of extreme learning machine by multitask beetle antennae swarm algorithm

Xixian Zhang1 · Zhijing Yang1 · Faxian Cao1 · Jiangzhong Cao1 · Meilin Wang1 · Nian Cai1

Received: 26 January 2019 / Accepted: 30 April 2020 / Published online: 16 May 2020
© Springer-Verlag GmbH Germany, part of Springer Nature 2020

Abstract
Extreme learning machine (ELM) as a simple and rapid neural network has been shown its good performance in various areas. Different from the general single hidden layer feedforward neural network (SLFN), the input weights and biases in hidden layer of ELM are generated randomly, so that it only takes a little computational overhead to train the model. However, the strategy of selecting input weights and biases at random may result in ill-conditioned problems. Aiming to optimize the conditioning of ELM, we propose an effective particle swarm heuristic algorithm called Multitask Beetle Antennae Swarm Algorithm (MBAS), which is inspired by the structures of artificial bee colony (ABC) algorithm and Beetle Antennae Search (BAS) algorithm. Then, the proposed MBAS is applied for optimizing the input weights and biases of ELM to solve its ill-conditioned problems. Experiment results show that the proposed method is capable of simultaneously reducing the condition number and regression error, and achieving good generalization performance.

Keywords Extreme learning machine (ELM) · Conditioning optimization · Beetle antennae search (BAS) · Heuristic algorithm

1 Introduction
Extreme learning machine (ELM) proposed by Huang et al. [1], is a feasible single hidden layer feedforward network (SLFN). It is composed of three core components: input layer, hidden layer and output layer. It has been successfully applied for many research and engineering problems. As a flexible and fast SLFN, the input weights and biases in hidden layer of ELM are assigned randomly, making the training speed increase greatly and process huge size of data in short time. There is no doubt that ELM is a good choice to cope with the tasks requiring instantaneity. Wong et al. utilizes ELM to detect the real-time fault signal of gas turbine generator system [2]. Xu et al. [3, 4] proposed a predictor based on ELM model for immediate assessment of electrical power system. Meanwhile, it has been proved that the performance of ELM and its variants are superior to most classical machine learning methods in the field of image processing [5–7], speech recognition [8–10], biomedical sciences [11–13], and so on.

There are plenty of efforts have been laid emphasis on enhancing the accuracy of ELM by means of adjusting neural network architecture and changing the numbers of hidden layer by certain rules. An incremental constructive ELM, adding its hidden nodes on the basic of convex optimization means and neural network theory, was proposed by Huang et al. [14]. While Rong et al. [15] dropped the irrelevant hidden nodes by calculating the correlation of each node with statistical approach. Zhu et al. proposed Evolution extreme learning machine [16], which achieved better performance by employing the differential evolutionary algorithm to tune parameters in hidden layer. It is inappropriate to evaluate the performance of a model by just considering the testing accuracy. That is because the stability is also one of the most significant criterions to evaluate the machine learning models. Choosing the coefficient randomly in hidden nodes is unable to ensure the numerical stability of ELM. In contrast, it will increase the risk of ill-conditioned problem, which implies that the output of network may tremendously change even though the input values appear slight fluctuation.

1 School of Information Engineering, Guangdong University of Technology, Guangzhou 510006, China
To obtain well-conditioned ELM network, heuristic algorithms are generally adopted to optimize the parameters in hidden nodes, such as Simulate Anneal (SA), Harmony Search (HS) [17], etc. To a certain extent, SA can prevent the occurrence of a local optimal solution with simple computing process. However, it is difficult for SA to search the globally optimal solution because of the small searching range. HS enlarges the searching range by adding a harmony memory library, which contains many optional solutions. There are two ways to generate a new solution in each iteration. One is tuning the solution from harmony memory library, and the other is generating a new solution randomly. If the new solution is better than the worst one in the harmony memory library, it will replace the worst one. Both SA and HS generate new solutions with random direction. The alterations of solution in SA and HS are irregular which will affect their performance.

Recently, a promising heuristic algorithm called Beetle antennae search algorithm (BAS) was proposed [18], which has two antennas in different directions to detect the new solution. The Beetle model always moves in the direction of antennae with the better result. Because of its good mechanism of velocity alteration, it shows very good performance in many applications. Nevertheless, it is hard for BAS to find an acceptable solution in the case of non-ideal scene, since the searching ability is sensitive to the initial length of step. If the length is too large, it may miss the globally optimal solution. The small size of step will lead to ‘false convergence’ cause of the decreasing step.

The Beetle in BAS always moves toward the better direction. Since improving the conditioning of ELM network is not a unimodal problem, it will be easy to fall into a local optima solution by single beetle particle. In order to enhance the searching ability of BAS, we attempt to increase more beetle particles. Different from the Particle Swarm Optimization (PSO) [19], we propose a novel particle swarm algorithm called Multitask Beetle Antennae Swarm Algorithm (MBAS) based on the framework of the Artificial Bee Colony (ABC) algorithm [20], where different particles have different update rules. We add some follower particles and explore particles to enlarge the searching range and prevent falling in local optimal solutions. Some particle swarm based algorithms [21–23] have adopted to increase the accuracy of ELM. Analogously, we can put the condition number of ELM into the fitness function of MBAS to optimize input weights and biases in the hidden layer.

The main contributions of this paper can be summarized as follows: (1) a novel beetle swarm optimization algorithm as the extending version of BAS is proposed to enhance the ability of searching the optimal solution. A beetle group with fixed population is defined, where each beetle has different function to enlarge the searching route around the solution. (2) An improved ELM named Multitask Beetle Antennae Swarm Algorithm Extreme Learning Machine (MBAS-ELM) is then proposed, by combining the beetle swarm algorithm with ELM to optimize the parameters of hidden nodes. Experiment results show that MBAS-ELM is available to lower the condition number as well as testing error for regression. More details of MBAS and MBAS-ELM will be discussed in Sect. 3.

The remainder of this paper is organized as follows. In Sect. 2, we briefly introduce the outline of ELM and BAS. In Sect. 3, the proposed MBAS and MBAS-ELM are given in details. Section 4 shows the results and discussion of the experiments. Finally, conclusion of our works is summarized in Sect. 5.

2 Reviews of ELM and BAS algorithms

2.1 Extreme learning machine

The extreme learning machine (ELM) contains three main layers (input layer, hidden layer, output layer). The input weights \( W \) between input layer and hidden layer are randomly generated. And the output weights \( \beta \) between hidden layer and output layer can be train without iteration. The network structure of ELM is given in Fig. 1.

Suppose the size of the training data is \( N \times d \), where \( N \) denotes the number of samples, and \( d \) denotes the dimension of features. Let \( x^i = [x_1^i, x_2^i, \ldots, x_d^i] \) and \( y^i = [y_1^i, y_2^i, \ldots, y_m^i] \) denote the \( i \)-th sample and target respectively, where \( m \) is the size of the target vector. Then the output of the \( j \)-th node in the hidden layer is given as follows:

\[
h_j = g \left( W_j x_i^T + b_j \right)
\]

where \( g(\cdot) \) denotes the activation function in the hidden layer. Our activation function in this paper is sigmoid.
function. Certainly, it can be formulated as common sigmoid function, such as tanh, ReLU. \( W_j = [\omega_{1j}, \omega_{2j}, \ldots, \omega_{dj}] \) is an \( d \)-dimensional input weight vector pointing to the \( j \)-th hidden node, and \( b_j \) is the hidden bias belonging to the \( j \)-th hidden node. The input weights and biases are assigned randomly. The \( k \)-th node in the output layer can be described as follows:

\[
y_k = \sum_{j=1}^{L} h_j \beta_{j,k}^T
\]  \hfill (2)

where \( \beta_{j,k} = [\beta_{1,k}, \beta_{2,k}, \ldots, \beta_{L,k}]^T \) is the output weights vector connecting to the \( k \)-th output-node, and \( L \) is the number of hidden nodes. The above equations can be rewritten into a matrix form as follows:

\[
Y = H \beta
\]  \hfill (3)

where 

\[
H = \begin{bmatrix}
g(W_1x_1^T + b_1) & \cdots & g(W_Lx_1^T + b_L) \\
g(W_1x_2^T + b_1) & \cdots & g(W_Lx_2^T + b_L) \\
\vdots & \ddots & \vdots \\
g(W_1x_N^T + b_1) & \cdots & g(W_Lx_N^T + b_L)
\end{bmatrix}_{N \times L},
\]

\[
\beta = \begin{bmatrix}
\beta^1 \\
\vdots \\
\beta^L
\end{bmatrix}, \quad Y = \begin{bmatrix}
y^1 \\
\vdots \\
y^m
\end{bmatrix}_{N \times m}.
\]

Finally, the output weight \( \beta \) can be calculated analytically by

\[
\beta = \text{pinv}(H)Y
\]  \hfill (4)

where \( \text{pinv}(\cdot) \) denotes the Moore–Penrose (MP) inverse function. It should be rewritten when matrix \( H \) is nonsingular as following equation:

\[
\beta = (H^T H)^{-1} H^T Y
\]  \hfill (5)

The essence of training ELM network is to estimate the output weight \( \beta \) by the analytical method.

### 2.2 Beetle antennae search algorithm

The BAS algorithm is the simulation of beetle foraging. Suppose that a beetle is in the \( k \)-dimension space, and its position is denoted as \( P_{cen} = [P_1, P_2, \ldots, P_k] \), where \( P_{cen} \in \mathbb{R}^k \). Two antennae endpoints of beetle are on the right and left sides, denoted as \( P_{rig}, P_{lef} \), where \( P_{rig}, P_{lef} \in \mathbb{R}^k \). Three points \( P_{cen}, P_{rig}, P_{lef} \) are on the same line. Figure 2 shows the process of searching minimum with BAS in two-dimensional plane. Two directions are assigned at random. Beetle \( P_{cen} \) moves to next position in the direction of its antennae endpoint \( P_{rig} \) or \( P_{lef} \) with the smaller value. The step size can be variable. After a few times of iteration, beetle will be very close to the solution. More details of BAS will be given in follows.
where $\eta$ denotes the decay factor, $0 < \eta < 1$, $d_{a}^{\text{min}}$ and $\delta_{\text{min}}$ denote the minimum sizes of the antennae and step, respectively.

The BAS algorithm for searching the minimum port is summarize in Algorithm 1.

**Algorithm 1: BAS algorithm for searching minimum**

**Initialization:**
- Input $\eta$, $d_{a}^{\text{min}}$, $\delta_{\text{min}}$, $d_{a}^{0}$, $\delta^{0}$, $t = 0$, $T_{\text{max}}$
- Initial position $P_{\text{cen}}(p_{1}, p_{2}, ..., p_{k})$

**Iteration:**
- While ($t < T_{\text{max}}$) or (stop criterion)
  - $\text{Dir}^{t} = \frac{\text{rand}(k)}{|\text{rand}(k)|}$
  - $P_{\text{rig.}}^{t} = P_{\text{cen}}^{t} + d_{a} \times \text{Dir}^{t}$
  - $P_{\text{lef.}}^{t} = P_{\text{cen}}^{t} - d_{a} \times \text{Dir}^{t}$
  - $P_{\text{cen}}^{t+1} = P_{\text{cen}}^{t} - \delta^{t} \times \text{Dir}^{t} \times \text{sign}(f(P_{\text{rig.}}^{t}) - f(P_{\text{lef.}}^{t}))$
  - $d_{a}^{t+1} = \eta \times d_{a}^{t}$, if $d_{a}^{t} > d_{a}^{\text{min}}$
  - $\delta^{t+1} = \eta \times \delta^{t}$, if $\delta^{t} > \delta_{\text{min}}$
- If $f(P_{\text{cen}}^{t+1}) < f(P_{\text{cen}}^{t})$
  - $P_{\text{best}} = P_{\text{cen}}^{t+1}$
  - $F_{\text{best}} = f(P_{\text{cen}}^{t+1})$
- End if
- $t = t + 1$
- End While

**Return:** $P_{\text{best}}$, $F_{\text{best}}$

### 3 The proposed method

#### 3.1 The proposed beetle swarm optimization algorithm

The seeking capability of single beetle particle is limited, especially for complex tasks since BAS tends to sink into a local optimal solution or incapable of obtaining adequate solutions. The idea of the particle swarm optimization increases the number of searching in each iteration, hence there is higher possibility to find the better solution. Inspired by animal population, Cui et al. [24] proposed a novel pigeon-inspired optimization algorithm for many-objective optimization problems. Wang et al. [25] proposed an Evolutionary multi-objective Optimization algorithm for Cyber Physical Social Systems. Cai et al. [26] proposed an improved version of bat algorithm via optimal forage strategy and random disturbance strategy. To combine particle swarm strategy with BAS, Wang et al. [27] presented a type of beetle swarm algorithm for multi-objective optimization which had shown its competitive performance. It was shown that the hybrid of BAS and swarm optimization is worth of study.

In human society, everyone has his own specific vocation, working in cooperation with each other. In order to complete a complex mission, people on different occupation make full use of their respective advantages to maximize the efficiency. Correspondingly, the division of labor widely exists in animal kingdom, especially the gregarious animals. There are several works following the rule of division of labor, such as Artificial Bee Colony algorithm (ABC) and Chicken Swarm Optimization algorithm (CSO) [28]. In general, this type of swarm algorithms creates a population with fixed number of particle. Each particle in this group is appointed to play the specified role, updating their position with different rule, which enriches the diversity of particle moving methods. The above process simulates how individual exerts its function and interdependent collaboration.

Inspired by ABC algorithm, we propose an individual cooperation method based on beetle swarm optimization...
algorithm namely Multitask Beetle Antennae Swarm Algorithm (MBAS). Details are given as follows.

Firstly, we define a beetle particle swarm with the population number $N$ in the $k$-dimension space. The main vocations of the beetle particles in this group are searchers, follower and explorer, respectively. For searchers, the most important mission is to search the solution in the feasible set and renew the position with Algorithm 1. The alteration of the position is given by the following equation:

$$P_{t+1}^b = BAS\_move (P_t^b, L_{t}^{min}, d_0^{min}, \eta)$$

(12)

where $BAS\_move(\cdot)$ represents the movement of certain iteration in Algorithm 1. After selecting the parameters for Algorithm 1, two endpoint positions $P_t^{rig}$ and $P_t^{lef}$ are calculated by $P_t^b(\omega, b), L_t^b$ and $L_t^{min}$ respectively are initial step and minimum step. Here, $P_t^b(\omega, b) \in \mathbb{R}^k$ represents the initial position of each particle, and $P_{t+1}^b(\omega, b)$ corresponds to $P_t^{cen*}$ calculated by Eq. (9). For convenience, let $d_{a}^{min} = 0$.

For followers, they pursue certain searcher beetle step by step to find the latent optimal solutions around the current global optimal solution. For convenience, all the follower particles follow the searcher particle with the best fitness values in our work. The corresponding movements are as follows:

$$D_F^i = P_S^i(\omega, b) - P_S^i(\omega, b)$$

(13)

$$P_{Ft+1}^i(\omega, b) = P_F^i + \frac{D_F^i}{|D_F^i|} \cdot L_F$$

(14)

where $L_F$ is the length of the step of the follower, and $P_F^i(\omega, b)$ is the position of the best searcher particle.

For explorers, they move randomly with fixed step length, preventing the whole system to fall into a local optimal solution. The equations are as follows:

$$|D_E^i| = \frac{rand(k)}{[rand(k)]}$$

(15)

$$P_{Ft+1}^i = P_E^i + |D_E^i| \cdot L_E$$

(16)

where $|D_E^i| \in \mathbb{R}^k$ is an unit vector, representing unit direction of explorers randomly generated in the $t$th iteration. $L_E$ denotes the step size of explores and $P_E^i$ denotes the position of explorer particle in the $t$th iteration.

Secondly, determine the numbers of each beetle with three vocations respectively as $N_S$, $N_F$ and $N_E$ with $N_S + N_F + N_E = N$. To generate the position $P_i(\omega, b)$ of each beetle particle randomly, for $i = 1, 2, \ldots, N$, calculate the fitness value $F_i(\omega, b)$ of each particle, and sort $F_i(\omega, b)$ from smallest to largest. Then appoint the top $N_S$ particle as the searchers, the top $N_S + 1$ to $N_S + N_F$ as the followers. The remaining particles are explorers.

Thirdly, the follower beetles approach the searcher beetles with the best fitness value. All the searchers beetles and explorer beetles move to the next position in their own ways. Calculate new fitness values with new positions for all particles. Redistribute the vocation to every beetle according to the new fitness values at the present iteration. If the best fitness value in current iteration is better than the previous best one, the current best fitness value replaces the previous best value.

Finally, the above process is repeated until $N_f$ epochs are completed. Return the best position and best fitness value. The MBSA algorithm is summarized in Algorithm 2.
In this section, the details of the proposed MBAS have been given. It can be seen that different from the Ref. [18], we do not use the frame of PSO to modify the BAS. In the frame of PSO, the update strategy of all particles is the same. If all the particles gather into a small space where the global solution is not inside, then no particle can jump out from this space. However, in ABC and MBAS, some particles move randomly to prevent the particles gather together. Compared with ABC algorithm, MBAS uses BAS to simplify the foraging process of employed bees. We add several particles with different functions to improve the searching ability of the BAS.

### 3.2 Beetle swarm optimization extreme learning machine

The original ELM does not need to train the hidden layer biases and input weights which are generated randomly. Although this mechanism can greatly reduce the
In this section, we introduce a new type ELM namely Multitask Beetle Antennae Swarm Algorithm Extreme learning machine (MBAS-ELM), and the flowchart of training MBAS-ELM is given in Fig. 3. The key of MBAS-ELM is to choose a suitable fitness function to satisfy the criterion. In order to build a reliable network, four criterions (denoted as $RMSE$, $R^2$, $K_2(H), \beta_2$) are considered as our optimal targets.

computation time, it restricts the stability and performance. The optimization of parameters in hidden layer of ELM is a complex problem. The searching ranges of SA, HS and BAS are limited by the particle number. In MBAS, follower particles and explore particles are applied to assist the searcher particles to find more potential solution around the best solution. Position interaction between different particles become stronger, thus MBAS is suitable for complex optimization problems.
Root mean squared error (RMSE) and coefficient of determination \(R^2\) are two most classical criterions for regression. RMSE is used to measure distance among predicted value and ground-truth. The smaller the RMSE value is, the better the performance is. RMSE is defined as follows.

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

where \(\hat{y}_i\) is the \(i\)th prediction in ELM, \(y_i\) is the \(i\)th ground-truth, and \(N\) is the total number of prediction.

\(R^2\) is applied to measure the fitting degree between the predicted value and ground-truth. The closer to 1 the \(R^2\) is, the better the fitting degree is.

\[
R^2 = 1 - \frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}
\]  

where \(\hat{y}_i\) is the \(i\)th prediction, \(y_i\) is the \(i\)th ground truth, and \(\bar{y}\) is the mean of all ground truth.

Furthermore, the norm of output weight \(\beta_2\) and condition number \(K_2(H)\) are also the significance criterion for ELM regression. The generalization ability is closely connected to the norm of output weight \(\beta_2\) \([29, 30]\). The smaller the norm value is, the better the generalization performance is. Zhao et al. \([31]\) employed the condition number \(K_2(H)\) to measure the stability of ELM. The global optimum solution of condition number is 1. A matrix with high condition

### Table 1 Information of real-world regression dataset

| Names     | Features | Training data | Test data |
|-----------|----------|---------------|-----------|
| bodyfat   | 14       | 100           | 152       |
| housing   | 13       | 250           | 256       |
| abalone   | 8        | 2000          | 2177      |
| mg        | 6        | 600           | 785       |
| eunite    | 16       | 150           | 186       |
| cpusmall  | 12       | 2000          | 6192      |
| wine quality (Red) | 11 | 700 | 899 |
| real estate | 6 | 200 | 214 |
| energy efficiency | 8 | 300 | 468 |
| Yacht     | 6        | 150           | 158       |
| mgp       | 7        | 150           | 242       |

### Table 2 Parameters setting of BAS-ELM and MBAS-ELM

| Names     | \(\eta\)  | \(d^0\)  | \(\delta\) | \(\delta_{\min}\) | \(N_S\) | \(N_F\) | \(N_E\) |
|-----------|-----------|----------|------------|-----------------|--------|--------|--------|
| BAS-ELM   | 0.999     | 0.05     | 10         | 0.01            | 1      |        |        |
| MBAS-ELM  | 0.999     | 0.05     | 10         | 0.01            | 3      | 6      | 1      |

### Table 3 RMSE of six ELM algorithms on all datasets (the best result is marked in bold type, and standard deviation is in parentheses)

| Names     | ELM    | SA-ELM | HS-ELM | BA-ELM | BAS-ELM | MBAS-ELM |
|-----------|--------|--------|--------|--------|---------|----------|
| bodyfat   | 0.0655 (0.0165) | 0.0540 (0.0119) | 0.0478 (0.0103) | 0.0508 (0.0103) | 0.0513 (0.0104) | 0.0513 (0.0074) |
| housing   | 0.1351 (0.0159) | 0.1274 (0.0131) | 0.1250 (0.0103) | 0.1251 (0.0111) | 0.1265 (0.0145) | 0.1208 (0.0107) |
| abalone   | 0.0816 (0.0026) | 0.0803 (0.0017) | 0.0800 (0.0019) | 0.0809 (0.0021) | 0.0799 (0.0017) | 0.0791 (0.0006) |
| mg        | 0.1593 (0.0039) | 0.1609 (0.0032) | 0.1598 (0.0036) | 0.1605 (0.0034) | 0.1567 (0.0033) | 0.1598 (0.0033) |
| eunite    | 0.1126 (0.0088) | 0.1001 (0.0144) | 0.0906 (0.0088) | 0.0987 (0.0135) | 0.0988 (0.0106) | 0.0883 (0.0088) |
| cpusmall  | 0.1059 (0.0117) | 0.1043 (0.0056) | 0.0982 (0.0064) | 0.1056 (0.0112) | 0.0939 (0.0056) | 0.0919 (0.0048) |
| Wine quality (Red) | 0.1353 (0.0046) | 0.1333 (0.0034) | 0.1325 (0.0034) | 0.1335 (0.0029) | 0.1325 (0.0031) | 0.1320 (0.0032) |
| real estate | 0.0807 (0.0075) | 0.0784 (0.0079) | 0.0778 (0.0071) | 0.0785 (0.0080) | 0.0781 (0.0078) | 0.0776 (0.0068) |
| Energy efficiency | 0.0852 (0.0047) | 0.0838 (0.0049) | 0.0822 (0.0049) | 0.0830 (0.0049) | 0.0852 (0.0039) | 0.0820 (0.0038) |
| Yacht     | 0.1496 (0.0093) | 0.1492 (0.0099) | 0.1476 (0.0081) | 0.1467 (0.0095) | 0.1453 (0.0112) | 0.1451 (0.0092) |
| mgp       | 0.0910 (0.0080) | 0.0854 (0.0070) | 0.0856 (0.0057) | 0.0866 (0.0070) | 0.0860 (0.0052) | 0.0843 (0.0066) |
number implies that it’s ill-conditioned. Its definition is given as follows:

$$K_2(H) = \sqrt{\frac{\lambda_{max}(H^TH)}{\lambda_{min}(H^TH)}}$$

(19)

where $\lambda_{max}(H^TH)$ and $\lambda_{min}(H^TH)$ are the largest and the smallest eigenvalue of matrix $H^TH$, respectively. The fitness function $F()$ is constituted of above four criterions, which is given by the following equation:

$$F() = \|\beta_2\| * K_2(H) + \gamma * (1 - R^2)$$

(20)
where $\gamma$ is a scaling factor. The reason why we design such fitness function will be explained in Sect. 4.3.

The training procedure of MBAS-ELM can be summarized as follows:

1. Given the training data $X_{\text{train}}$, training label $Y_{\text{train}}$, testing data $X_{\text{test}}$, and testing label $Y_{\text{test}}$. We divide $X_{\text{train}}$ and $Y_{\text{train}}$ into three non-intersect subset $(X_r', Y_r')$ for $r$-fold cross-validation, where $r = 1, 2, 3$.

2. Fix the size of population. Randomly initialize position of each particle position, which made up of input weights and biases in the $r^{th}$ subset, described as,

Fig. 4 The condition value comparison of six algorithms on six datasets in 30 epochs, a bodyfat, b housing, c abalone, d mg, e eunite, f cupsmall

Fig. 5 The condition value comparison of six algorithms on five datasets in 30 epochs, g wine quality (Red), h real estate, i energy efficiency, j Yacht, k mgp
where \( n \) is the size of input layer, \( L \) denotes the number of hidden layer, \( P_{r}(\omega, b) \in [-1, 1]^{(n+1)\times L} \) represents the parameters of the \( r \)-th hidden layer, \( \omega_{n1}, \omega_{n2}, \ldots, \omega_{nL}, b_{1}, b_{2}, \ldots, b_{L} \) are the weights and biases of the \( r \)-th hidden layer, respectively.

\[
P_{r}(\omega, b) = [\omega_{11}, \omega_{12}, \ldots, \omega_{1L}, \omega_{21}, \omega_{22}, \ldots, \omega_{2L}, \omega_{n1}, \omega_{n2}, \ldots, \omega_{nL}, b_{1}, b_{2}, \ldots, b_{L}]^{(n+1)\times L}
\]

\( r = 1, 2, 3 \).

**Step 3** Calculate fitness function and RMSE for all particles in each iteration. Select the position corresponding to the smallest fitness value in current iteration as \( P_{\text{current}} \). The best position \( P_{\text{best}} \) renew as following equation,
\[ P_{\text{best}} = \begin{cases} P_{\text{cur}}, & F(P_{\text{cur}}) < F(P_{\text{best}}) \text{ and } \text{RMSE}_{\text{cur}} < \text{RMSE}_{\text{best}} \\ P_{\text{best}}, & \text{else} \end{cases} \]  

(22)

Step 4: Repeat Step 3 for each subset, select the best position \( P^r_{\text{best}}, r = 1, 2, 3 \).

Step 5: Employ \( \hat{P}_{\text{best}}(\hat{\omega}, \hat{b}) \) obtained from Step 4 to estimate the trained network.

4 Experiment verification

4.1 Experiment setup

In this section, to test the performance of the proposed MBAS-ELM, several well-known methods are used for comparisons, including the ELM, BAS-ELM, HS-ELM (Harmony Search), SA-ELM (Simulate Anneal) and BA-ELM (Bat Algorithm) on 11 real-world regression benchmark datasets. Details of the datasets are given in Table 1. All the datasets are available from LIBSVM dataset [32] and UCI dataset [33]. Four criterions (RMSE, \( R^2 \), \( K_2(H)\beta_2 \)) as discussed in Sect. 3.2 are adopted to estimate the performance of all the algorithms. The influence of hidden node number is not the emphases in this paper, thus we empirically assign 10 nodes for each ELM network.

BAS-ELM can be seen as a particular case of MBAS-ELM, which only contains one searcher. The parameters of the searchers in BAS-ELM and MBAS-ELM stay the same, which are given in Table 2. We assign \( N_3, N_F\) and \( N_E\) with the values of 3, 6, and 1 empirically based on lots of experiments. For HS-ELM and SA-ELM, we use the harmony search algorithm and simulate anneal algorithm to optimize the parameter in ELM hidden layer. Here, in the case of HS-ELM, the harmony memory size (HMS) is 10, harmony memory considering rate (HMCR) and pitch adjusting rate (PAR) are 0.7, band-width is 0.1, and the maximum iteration is 30. As for SA-ELM, the maximum temperature is 1.6701, the minimum temperature is 1, temperature attenuation is 0.95, and the maximum iteration is 30. For BA-ELM, the maximum iteration is 30, initial loudness \( A^0 = 0.5\), and initial pulse rate is 0.5.

4.2 Experiment results

Six aforementioned ELM algorithms (original ELM, SA-ELM, HS-ELM, BA-ELM, BAS-ELM, MBAS-ELM) are independently tested 30 times on 11 datasets with the same training and testing data in each epoch. The training data and testing data are chosen randomly at each epoch. The outputs of six algorithms are normalized to the range from 0 to 1, and the input data are normalized from -1 to 1. After 30 times testing, we calculate the mean of RMSE, \( R^2 \), \( K_2(H)\), \( \beta_2 \), and the standard deviation of RMSE. We use the cross validation to decide the parameter \( \gamma \) in Eq. (20) from the candidate numbers \{100, 300, 600, 900\}. The experiment results of four criterions are given in Tables 3, 4, 5 and 6, and the line chart of \( K_2(H) \) and \( \beta_2 \) in 30 times are given in Figs. 4, 5, 6 and 7.

As can be seen from Table 3, the RMSE of all ELM algorithms on all datasets are presented. All the five optimized ELM algorithms (SA-ELM, HS-ELM, BA-ELM, BAS-ELM, MBAS-ELM) can reduce the RMSE in 30 epochs. In most cases, the proposed MBAS-ELM gets the best performance in RMSE, especially in bodyfat and eunite datasets. In contrast to the original ELM, the RMSEs on the above two datasets decrease from 0.0655 to 0.0430 and from 0.1126 to 0.0883 respectively. Although the performance on mg is not the best, MBAS-ELM gets very close results to the best. Moreover, MBAS-ELM achieves the best RMSE and standard deviation in most cases simultaneously.

As shown in Table 4, the mean value of \( R^2 \) of all ELM algorithms on all dataset are presented. The five most significant improvements are in the four datasets, including bodyfat, housing, eunite, cupsmall. The improvement of MBAS-ELM compared to the original ELM are 0.0978, 0.0837, 0.1592, 0.0833 respectively for \( R^2 \). All the five modified ELM algorithms improve the performance in the rest datasets with different degree. From Tables 3 and 4, the best RMSE and \( R^2 \) of each dataset are from the same algorithm. Because \( R^2 \) is closely relative to RMSE. As can be seen that, MBAS-ELM achieves the best performance in \( R^2 \) and RMSE simultaneously.

In [34], a standard related to condition number is given to describe the degree of collinearity. If the condition number is less than 100, the collinearity of ELM is weak. If the condition number is larger than 100 and less than 1000, the collinearity of ELM is moderate. If condition number is larger than 1000, the collinearity of ELM is severe. It is inevitable to suffer collinearity and ill-condition for ELM, due to the random parameter in hidden layer. In Table 5, the optimized ELM algorithms tune the random parameters in training stage. Hence, the condition numbers greatly decrease. For ELM, only 2 datasets (housing, eunite) are defined as weak collinearity (condition number < 100). In contrast, the numbers of weak collinearity of SA-ELM, HS-ELM, BA-ELM, BAS-ELM are 5, 5, 5, 9 respectively. The proposed MBAS-ELM has the weakest collinearity (9 times) in all the ELM algorithms. In addition, the smaller the condition number is, the better numerical stability is. Obviously, the mean condition number of MBAS-ELM is the smallest in all datasets. Thus, MBAS-ELM achieves the best performance of \( K_2(H) \) in all the datasets.

As shown in Table 6, the mean value of \( \beta_2 \) of all the ELM algorithms on all dataset are presented. The four most significant improvements are in the four datasets, including
mg, cpusmall, Yacht, mgp. The decrements of MBAS-ELM compared to the original ELM are 3.1881, 2.9671, 1.7384, 2.1264 respectively for $\beta_2$. All the five modified ELM algorithms also improve the performance in the rest datasets with different degree. In addition, the smaller the $\beta_2$ is, the better generalization ability is. All the smallest performances are from MBAS-ELM. Thus, MBAS-ELM achieves the best performance of $\beta_2$ in all the datasets.

The condition numbers of the six methods in the 11 datasets are given in Figs. 4 and 5, respectively. Not surprisingly, the original ELM appears the maximum fluctuation on all datasets, due to the random parameter in the hidden layer. All the five optimized ELM algorithms can reduce the risk of ill-conditioned in different degrees. The results from Figs. 4d and 5h–k obviously show that the fluctuation of condition number of SA-ELM, HS-ELM and BA-ELM are smaller than ELM. In contrast to BAS-ELM, MBAS-ELM contains less deviation point in Figs. 4b, f, 5i. In general, the proposed MBAS-ELM can achieve the best performance in system stability.

The norm values of the six methods in the 11 datasets are given in Figs. 6 and 7, respectively. The variation curve of the original ELM appears great fluctuation. All the five optimized ELM algorithms can reduce the fluctuation and the mean of norm value with different degrees. From Figs. 6d, 7j, k and it is obviously to find that the norm value of MBAS-ELM achieves the smallest change. Moreover, the proposed MBAS-ELM achieves the smallest mean on all datasets.

4.3 Experiment discussion

The fitness function of MBAS-ELM is given as Eq. (20), constituted by three criteria ($R^2, K_2(H), \beta_2$). The above three criteria seem unrelated to collaborate. Hence, we will discuss the details with respect to designing of fitness function in this section. In order to fully improve the conditioning and generalization ability, we consider the $K_2(H)$ and $\beta_2$ as variables in our fitness function. Meanwhile, we add $R^2$ to ensure the regression precision. From the definition equation of RMSE and $R^2$, it can be easily seen that they contain the same component $\sum_{i=1}^{n} (\hat{y}_i - y_i)^2$, so we can tune $R^2$ instead of RMSE to lower the error. If a few of predicting points with huge error appear in regression result, the mean error may not give rise to great changes, but the fitting degree may probably decrease. For regression task, we hope the fitting degree can be as close to 1 as possible. Moreover, the scaling factor $\gamma$ of Eq. (20) can be seen as a weight of accuracy and conditioning. If $\gamma$ is small, the MBAS algorithm tends to search the solution with low condition number and norm value. On the contrary, large $\gamma$ probably leads to ignore the condition number and norm value. In general, the range of $R^2$ is uncertain. Although we normalize the output from 0 to 1, the range of RMSE is still uncertain if the prediction number is too large. It’s hard to determine $\gamma$ when RMSE is in a large range. Different form RMSE, the range of the norm value is usually form 1 to 6, and the range of condition number of ELM is usually from 40 to 400. Thus, the product of condition number and norm value is approximately from 40 to 2400. Obviously, $R^2$ is from 0 to 1, so we can easily determine $\gamma$ to change the proportion of precision and stability according to actual needs. Hence, $R^2$ should be considered in the fitness function.

The assignment of particle numbers ($N_S$, $N_F$ and $N_E$) is related to robustness and computational cost. In general, searcher particle $N_S$ is positively correlated to searching ability and computational time. Numbers of follower particle and explore particle are positively correlated to the robustness. In this paper, we assign particle for all experiments as shown in Table 2, which is tradeoff between performance and computation time. As can be shown in Tables 3, 4, 5 and 6, performance of BAS-ELM and MBAS-ELM is very close on dataset Yacht and mgp. Although the result on Yacht and mgp is similar, the computational time of BAS-ELM is much less than MBAS-ELM. Therefore, we can reduce $N_S$ when requiring less computational time.

From above experiment results, the proposed ELM algorithm shows its great performance in most datasets. MBAS-ELM can ensure the regression accuracy within acceptable range and greatly enhance stability and generalization ability.

5 Conclusion

In this paper, a swarm optimization algorithm called MBAS algorithm is proposed. Inspired by the existing swarm optimization algorithm, the particles play different roles in MBAS to enlarge the searching area. Furthermore, we apply MBAS algorithm to optimize the input weights and biases in hidden layer of ELM to improve conditioning and generalization ability. Experiments are established to show the performance of the proposed MBAS-ELM algorithm by comparing with the original ELM, SA-ELM, HS-ELM, BA-ELM and BAS-ELM. Experiments show that MBAS-ELM can achieve the best performance on almost all datasets, and effectively improve the conditioning and generalization ability. Since MBAS-ELM contains more beetle particles than BAS-ELM, it needs relatively more time for computation. Fortunately, BS is a fast and simple heuristic algorithm. Therefore, the computational cost can stay in an acceptable level by choosing reasonable particle population at the beginning of MBAS-ELM.

Moreover, the assignment of particle number is also the vital problem for MBAS-ELM. General speaking, it is
unnecessary to assign too much searcher particle and follower particle for easy function. In the future, we will focus on reducing computational cost of MBAS-ELM, and studying how to quantified different particles in MBAS.

Acknowledgements This work is supported in part by the Guangdong Graduate Education Innovation Project (No. 2020XSLT16), Science and Technology Project of Guangdong Province (No. 2019A050513011), National Nature Science Foundation of China (No. U1701266), and Guangdong Provincial Key Laboratory of Intellectual Property and Big Data under Grant (No. 2018B030322016).

References

1. Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1–3):489–501
2. Wong PK, Yang Z, Vong CM, Zhong J (2014) Real-time fault diagnosis for gas turbine generator systems using extreme learning machine. Neurocomputing 128:249–257
3. Zhang L, Hu X, Li P, Shi F, Yu Z (2017) ELM model for power system transient stability assessment. In: Chinese automation congress (CAC), pp 5740–5744
4. Xu Y, Dai Y, Dong ZY, Zhang R, Meng K (2013) Extreme learning machine-based predictor for real-time frequency stability assessment of electric power systems. Neural Comput Appl 22(3–4):501–508
5. Cao J, Zhang K, Luo M, Yin C, Lai X (2016) Extreme learning machine and adaptive sparse representation for image classification. Neural Netw 81:91–102
6. Cao F, Liu B, Park DS (2013) Image classification based on effective extreme learning machine. Neurocomputing 102:90–97
7. Marques I, Graña M (2012) Face recognition with lattice independent component analysis and extreme learning machines. Soft Comput 16(9):1525–1537
8. Muthusamy H, Polat K (2015) Yaacob S (2015) Improved emotion recognition using gaussian mixture model and extreme learning machine in speech and glottal signals. Math Probl Eng 52:96
9. Lan Y, Hu Z, Soh YC, Huang GB (2013) An extreme learning machine approach for speaker recognition. Neural Comput Appl 22(3–4):417–425
10. Kaya H, Salah AA (2016) Combining modality-specific extreme learning machines for emotion recognition in the wild. J Multimodal User Interfaces 10(2):139–149
11. Song Y, Crowcroft J, Zhang J (2012) Automatic epileptic seizure detection in EEGs based on optimized sample entropy and extreme learning machine. J Neurosci Methods 210(2):132–146
12. Kim J, Shin HS, Shin K, Lee M (2009) Robust algorithm for arrhythmia classification in ECG using extreme learning machine. Biomed Eng Online 8(1):31
13. Wang G, Zhao Y, Wang D (2008) A protein secondary structure prediction framework based on the extreme learning machine. Neurocomputing 70(16–18):3056–3062
14. Huang GB, Chen L (2007) Convex incremental extreme learning machine. Neurocomputing 70(1–3):262–268
15. Rong HJ, Huang GB, Sundararajan N, Saratchandran P (2009) Online sequential fuzzy extreme learning machine for function approximation and classification problems. IEEE Trans Syst Man Cybern B 39(4):1067–1072
16. Zhu QY, Qin AK, Suganthan PN, Huang GB (2005) Evolutionary extreme learning machine. Pattern Recognit 38(10):1759–1763
17. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68
18. Zhu Z, Zhang Z, Man W, Tong X, Qiu J, Li F (2018) A new beetle antennae search algorithm for multi-objective energy management in microgrid. In: 2018 13th IEEE conference on industrial electronics and applications (ICIEA), pp 1599–1603
19. Shi Y (2001) Particle swarm optimization: developments, applications and resources. In: Proceedings of the 2001 congress on evolutionary computation, pp 81–86
20. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471
21. Li H, Du B, Liu J, Xia H, Yeap WK (2017) A kernel extreme learning machine algorithm based on improved particle swarm optimization. Memet Comput 9(2):121–128
22. Han F, Yao HF, Ling QH (2013) An improved evolutionary extreme learning machine based on particle swarm optimization. Neurocomputing 116:87–93
23. Li G, Niu P, Ma Y, Wang H, Zhang W (2014) Tuning extreme learning machine by an improved artificial bee colony to model and optimize the boiler efficiency. Knowledge-Based Syst 67:278–289
24. Cui ZH, Zhang JJ, Wang YC, Cao Y, Cai XJ, Zhang WS, Chen JJ (2019) A pigeon-inspired optimization algorithm for many-objective optimization problems. Sci China Inf Sci 62(7):569
25. Wang G, Cai XJ, Cui ZH, Min GY, Chen JJ (2017) High performance computing for cyber physical social systems by using evolutionary multi-objective optimization algorithm. IEEE Trans Emerg Top Comput 97:1–1
26. Cai XJ, Gao XZ, Xue Y (2016) Improved bat algorithm with optimal forage strategy and random disturbance strategy. Int J Bio-Inspired Comput 8(4):205–214
27. Wang T, Yang L, Liu Q (2018) Beetle swarm optimization algorithm: theory and application. arXiv preprint arXiv:1808.00206
28. Meng X, Liu Y, Gao X, Zhang H (2014) A new bio-inspired algorithm: chicken swarm optimization. In: International conference in swarm intelligence, pp 86–94
29. Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: IEEE Proceedings of 2004 international joint conference on neural networks, pp 985–990
30. Huang GB, Ding X, Zhou H (2010) Optimization method based extreme learning machine for classification. Neurocomputing 74(1–3):155–163
31. Zhao G, Shen Z, Mao C, Man Z (2009) On improving the conditioning of extreme learning machine: a linear case. In: IEEE ICICS 2009. 7th international conference on information, communications and signal processing, pp 1–5
32. LIBSVM Data Homepage, https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets. (Accessed on 08 July 2019)
33. UCI Homepage, http://archive.ics.uci.edu/ml/datasets. (Accessed on 08 July 2019)
34. Kassani PH, Teoh ABJ, Kim E (2018) Sparse pseudoinverse incremental extreme learning machine. Neurocomputing 287:128–142

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.