

An Improved Gravitational Search Algorithm for Dynamic Neural Network Identification

Bao-Chang Xu Ying-Ying Zhang

Department of Automation, China University of Petroleum (Beijing), Beijing 102400, China

Abstract: Gravitational search algorithm (GSA) is a newly developed and promising algorithm based on the law of gravity and interaction between masses. This paper proposes an improved gravitational search algorithm (IGSA) to improve the performance of the GSA, and first applies it to the field of dynamic neural network identification. The IGSA uses trial-and-error method to update the optimal agent during the whole search process. And in the late period of the search, it changes the orbit of the poor agent and searches the optimal agent's position further using the coordinate descent method. For the experimental verification of the proposed algorithm, both GSA and IGSA are testified on a suite of four well-known benchmark functions and their complexities are compared. It is shown that IGSA has much better efficiency, optimization precision, convergence rate and robustness than GSA. Thereafter, the IGSA is applied to the nonlinear autoregressive exogenous (NARX) recurrent neural network identification for a magnetic levitation system. Compared with the system identification based on gravitational search algorithm neural network (GSANN) and other conventional methods like BPNN and GANN, the proposed algorithm shows the best performance.

Keywords: Gravitational search algorithm, orbital change, optimization, neural network, system identification.

1 Introduction

Gravitational search algorithm (GSA) is a novel meta-heuristic stochastic optimization algorithm inspired by the law of gravity and mass interactions^[1]. Experiments have proved that GSA demonstrates a strong optimizing ability compared with real genetic algorithm (RGA), particle swarm optimization (PSO) and central force optimization (CFO). So far this algorithm has been swiftly and widely applied to filter modeling^[2], forecasting future oil demand in Iran^[3], pipeline scheduling^[4], slope stability analysis^[5], dispatch problems^[6, 7] and many other research fields.

Nonlinear system identification based on dynamic neural network is always the difficulty and hotspot in the control theory research. The essence of system identification based on neural network is to choose the proper neural network parameters so as to approximate the actual system. Most of the network design efforts have been on algorithm selection for minimal iterations and better convergence in computation. Yang and Lee^[8] applied the back-propagation (BP) algorithm to three neural networks for system identification. Genetic algorithm (GA) has been applied to feedforward^[9] and radial basis function neural networks^[10]. Integration of GA with conjugate gradient, fuzzy logic and Newton-Raphson method has also been proposed^[11, 12]. But obtaining better convergence and avoiding trapping into the local minimum in the process of identifying non-linear system based on neural network has always been an open problem. As an effective global optimization algorithm, GSA has great potential to be used for training a neural network. Nevertheless, its application research in this aspect is still rare now.

The course of training neural network weights is also to

seek the minimum of a high-dimension multimodal function. In terms of the high-dimension multimodal function, it is important to carry out the precise local search in the late period of optimization for enhancing the optimization precision^[13]. However, GSA is not able to efficiently realize this for lacking an effective local search mechanism. Therefore, the performance of GSA still needs to be improved further and thus an improved gravitational search algorithm (IGSA) is proposed. In IGSA, trial-and-error method is adopted to update the optimal agent in the whole search process in order to facilitate the global exploration. And at the final stage of iterations, IGSA changes the orbit of the poor agents and searches the optimal agent's position further using the coordinate descent method to improve the quality of the solution. Numerical simulation results of optimization of four famous benchmark functions and a neural network identification problem demonstrate that the proposed strategies can significantly improve the GSA's convergence performance.

The rest of the paper is organized as follows. A brief review of GSA and the proposed IGSA algorithm are presented in Section 2. In Section 3, the test of the proposed IGSA through four benchmark functions is carried out and simulation results are compared with those obtained via GSA. The simulation results and analysis on neural network identification are presented in Section 4. Finally, the conclusion is presented in Section 5.

2 Improved algorithm of GSA

2.1 GSA

GSA is a heuristic optimization algorithm based on the law of gravity among objects. In GSA, the search agents are a collection of masses, and their interactions are based on the Newtonian laws of gravity and motion. The gravity

Regular paper
Manuscript received May 27, 2013; revised September 12, 2013
This work was supported by National Natural Science Foundation of China (No. 2011ZX05021-003) and Science Foundation of China University of Petroleum.

force is an acting force drawing objects closely. In the preliminary stage of the universe formation, various objects were disorderly distributed all around the universe. Due to the existence of the universal gravitation, the objects with higher gravitation gathered together and then evolved into the galaxy.

In GSA, each agent has four variables: position, inertial mass, active gravitational mass and passive gravitational mass. Now consider a system with N agents in the search scope. We define the position of the i -th agent (agent i) by $X_i = (x_i^1, \dots, x_i^d, \dots, x_i^N)$, $i = 1, 2, \dots, N$, where x_i^d is the d -th dimension value of agent i .

At time t , the force applied on agent i by agent j is

$$F_{ij}^d(t) = G(t) \frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon} (x_j^d(t) - x_i^d(t)) \quad (1)$$

where $M_{pi}(t)$ is the passive gravitational mass related to agent i , $M_{aj}(t)$ is the active gravitational mass related to agent j , ε is a small constant, $R_{ij}(t)$ is Euclidean distance between agent i and agent j , $G(t)$ is gravitational coefficient and is decreased over time so as to control the search accuracy.

$$R_{ij}(t) = \|X_i(t), X_j(t)\|_2 \quad (2)$$

$$G(t) = G_0 e^{-\tau \frac{t}{t_{\max}}} \quad (3)$$

The resultant force acting on agent i in the d -th dimension is

$$F_i^d(t) = \sum_{j \in K_{\text{best}}} \text{rand}_j F_{ij}^d(t). \quad (4)$$

According to Newton second law, the acceleration of agent i at time t in direction of the d -th dimension is

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \quad (5)$$

where $M_{ii}(t)$ is the inertial mass of agent i . $M_{ii}(t) = M_{pi}(t) = M_{ai}(t) = M_i$, $i = 1, 2, \dots, N$. The gravitational and inertia masses are updated by the following equations.

$$m_i(t) = \frac{\text{fit}_i(t) - \text{worst}(t)}{\text{best}(t) - \text{worst}(t)} \quad (6)$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^N m_j(t)}. \quad (7)$$

After the acceleration is calculated, the speed and position of agent can be updated as

$$v_i^d(t+1) = \text{rand}_i v_i^d(t) + a_i^d(t) \quad (8)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \quad (9)$$

where rand_i is a uniform random variable between 0 and 1. For the minimum problem, there are

$$\text{best}(t) = \min_{j \in \{1, \dots, N\}} \text{fit}_j(t) \quad (10)$$

$$\text{worst}(t) = \max_{j \in \{1, \dots, N\}} \text{fit}_j(t). \quad (11)$$

For the maximum problem, there are

$$\text{best}(t) = \max_{j \in \{1, \dots, N\}} \text{fit}_j(t) \quad (12)$$

$$\text{worst}(t) = \min_{j \in \{1, \dots, N\}} \text{fit}_j(t). \quad (13)$$

2.2 Main improved strategies of IGSA

1) Orbital change of poor agents' positions.

During GSA searching process, all agents gradually converge to a small local zone, which results in a low searching efficiency in the late period, so an effective mechanism should be established to help poor agents jump out of the local minimum. Influenced by the gravitational attraction, artificial satellites and space ships (including space station) fall at a speed of 100 m/d, which will hamper their normal operation. So during the flying, orbital change is always required. Based on the concept mentioned above, the paper performs an orbital change operation upon the poor agents (in the paper, the worst 10 agents are chosen according to the fitness) in the late search period of the algorithm in order to prevent them from falling into the local minimum and improve the algorithm performance.

Using (14), orbital change operation is able to enlarge or contract the positions of poor agents at a certain probability (named jump rate). The positions change adaptively along with its original value, and is called orbital change radius. That is, if agents' positions converge to a smaller value, the orbital change radius will be smaller, on the contrary, if agents' positions converge to a bigger value, the orbital change radius will be bigger, too. The orbital change operation is good for jumping out of the local minimum and improving the convergence speed, and yet not making a big disturbance upon the global.

$$xm_i = x_i + \text{rands } x_i, \quad i = 1, 2, \dots, N \quad (14)$$

where rands is a random number between -1 and 1 .

2) Further search of optimal agent position.

The GSA algorithm generally converges quickly in the early 70% iterations, and then the convergence speed becomes slow. In order to further intensify the optimal searching ability of the algorithm in the late period, the optimal agent is further optimized by coordinate descent method^[14] and it transforms the multi-variable optimization problem into some single-variable sub-problems. It helps optimize further the position of the optimal agent, establish an effective local search mechanism and thus improve the algorithm performance further. The detailed steps for coordinate descent method are as follows.

Step 1. The variable that needs further optimization is the optimal agent's position x_{best} . Define the initial unit orthogonal search direction, generally the coordinate axis direction, as the candidate, i.e., $d_1, \dots, d_{\text{dim}}$; the range of the variable x_{best} is $[\text{low}, \text{up}]^{\text{dim}}$, where dim is the dimension of x_{best} .

Step 2. Solving sub-problem

$$\text{For } (j = 1, j \leq \text{dim}, j++) \quad (15)$$

$$\min : f(x_{\text{best},j} + \lambda_j d_j) \tag{16}$$

where λ_j is the coordinate parameter in the direction of the j -th coordinate and is required to meet the feasible condition:

$$\text{low} - x_{\text{best},j} \leq \lambda_j \leq \text{up} - x_{\text{best},j}. \tag{17}$$

Step 3. By precise linear search, we can obtain the optimal solution and update the position of optimal agent by

$$x_{\text{best}} = x_{\text{best}} + \lambda_j d_j. \tag{18}$$

3) Update optimal agent using trial-and-error method.

In GSA, all current agents change at each step; if the optimal agent's fitness becomes bad, the next search will begin from a worse position. The optimal position of those historical search steps, L_{best} , and its fitness F_{best} , only play a role for comparison, rather than participate into each step of iterative search. In order to utilize the information of L_{best} , the optimal agent is updated using the trial-and-error method, i.e., after each iteration, the search will continue to the next step if the fitness of the optimal agent turns better. Otherwise, the position of optimal agent's position and fitness will be replaced by L_{best} and F_{best} .

2.3 Steps of IGSA algorithm

The steps of IGSA algorithm are as follows:

Step 1. Initialization of parameters.

Step 2. Fitness evaluation of agents.

Step 3. Update gravitational coefficient $G(t)$, best value $\text{best}(t)$ and worst value $\text{worst}(t)$.

Step 4. Update the optimal fitness $F_{\text{best}}(t)$ in the history record group and its corresponding position $L_{\text{best}}(t)$, and the trial-and-error method is adopted for the updating of optimal agent.

Step 5. Calculate the inertial mass, resultant force, acceleration and velocity of agents.

Step 6. Update the position of agents.

Step 7. If it has run 70% of the maximum iterative steps, the orbital change operation should be carried out

for those agents whose fitness values are bad.

Step 8. If it has run 70% of the maximum iterative steps, the coordinate descent method should be carried out for the optimal agent.

Step 9. Repeat Steps 2 to 8 until the stop criteria is reached.

3 Test and analysis for algorithm performance

3.1 Benchmark functions

In order to verify the improvement of IGSA algorithm for multimodal function optimization, four classic benchmark functions^[1,15] are chosen for comparison test shown in Table 1, where F_1 and F_2 are the famous Rastrigin and Griewank functions, respectively, and their dimensions are both 30 ($n=30$).

3.2 Optimization results and analysis

The parameter setting for IGSA is as follows: The agent scale N is 30, maximum number of iterations max_it is 500, and the orbital change probability (jump rate) J_r is 0.5. In order to decrease the influence of random factors used in the algorithm, fifty independent experiments are carried out for each function minimum optimization simulation and the average evolution curves for the fifty experiments are shown in Fig. 1.

The evaluation indexes of an algorithm's performance include the optimization precision, convergence speed and robustness. The robustness is evaluated by computing the ratio of the test times that the algorithm reaches the regulated threshold value to the total test times^[16], i.e., the success rate in Table 2. The thresholds of F_1, F_2, F_3 , and F_4 are set respectively as $e^{-5}, e^{-5}, 0.999$ and 10.1531 . From Table 2, we can clearly find that each index of IGSA is better than that of GSA, and its optimization precision is improved by over 60% compared with GSA, and the robustness is also increased by 100%.

Table 1 Test functions

Test function	Scale	f_{opt}	X_{opt}
$F_1(X) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[5.12, 5.12]^n$	0	$[0, \dots, 0]$
$F_2(X) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600, 600]^n$	0	$[0, \dots, 0]$
$F_3(X) = (\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^j (x_i - a_{ij})^6})^{-1}$	$[-65.53, 65.53]^2$	0.9980	$[-32, 32]$
where $(a_{ij}) = \begin{bmatrix} -32, -16, 0, 16, 32, 32, \dots, 0, 16, 32 \\ -32, -32, -32, -32, -16, \dots, 32, 32, 32 \end{bmatrix}$			
$F_4(X) = - \sum_{i=1}^5 [(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	-10.1532	$[4, 4, 4, 4]$
where if $i=1, a_i=[4,4,4,4], c_i=0.1;$			
if $i=2, a_i=[1,1,1,1], c_i=0.2;$			
if $i=3, a_i=[8,8,8,8], c_i=0.2;$			
if $i=4, a_i=[6,6,6,6], c_i=0.4;$			
if $i=5, a_i=[3,7,3,7], c_i=0.4.$			

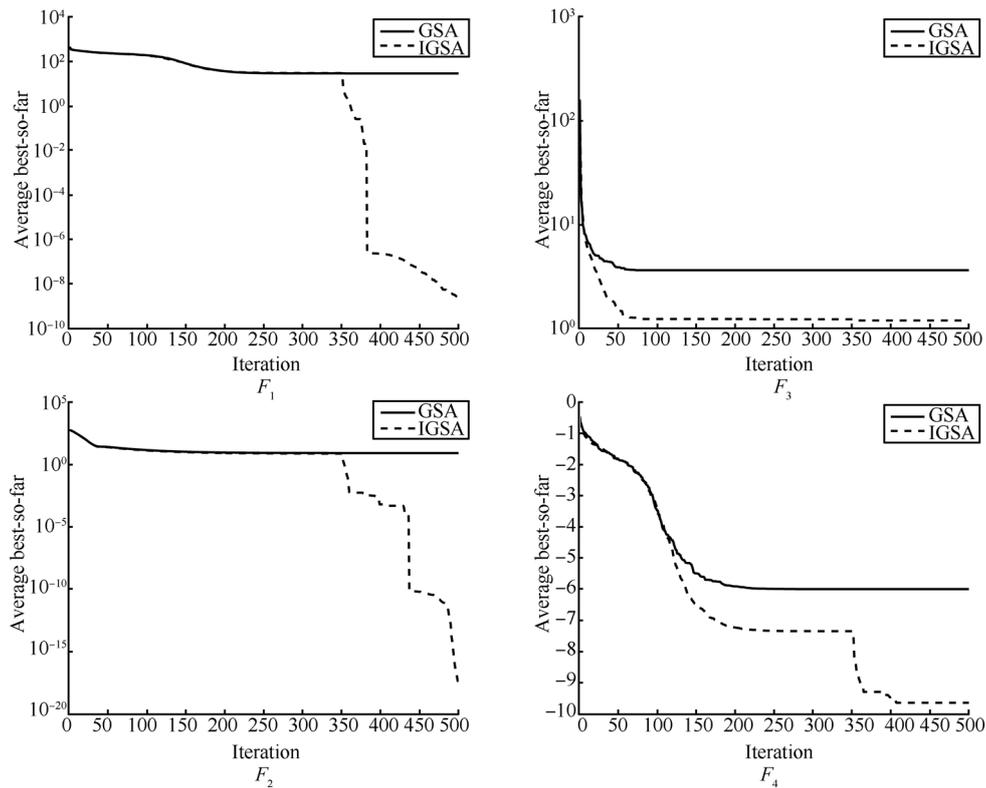


Fig. 1 Optimization evolution curves of test functions based on GSA and IGSA

From Fig. 1, we can find that the convergence speed of IGSA is similar to GSA in the preliminary stage of evolution. In the middle stage of evolution, both GSA and IGSA show a tendency to premature convergence and the evolution speed declines, but since IGSA has utilized the information of those historical search steps, the evolution space is exploited further in F_3 and F_4 , and the IGSA begins to exhibit a certain advantage. In the later stage of evolution, the difference between these two algorithms' performance becomes distinctive: The gravitational constant decreases with the increase of iteration, and the evolution is very difficult to be optimized further using GSA. However, based on the orbital change for the poor agents and the further search for the optimal agent, the convergence speed of IGSA is accelerated greatly and thus IGSA presents the "two-

section" evolution characteristic. The IGSA helps solve the premature convergence problem of GSA and enhances the exploitation capability.

Khajehzadeh et al.^[5] proposed the MGSA (modified GSA) algorithm, which adopts the self-adaptation maximum speed restriction strategy to control the global exploration ability of the GSA algorithm. However, the restriction parameter of the MGSA algorithm is not easy to regulate. The function F_4 in the reference paper and the function F_1 in this paper are both Rastrigin functions, and optimization result in the reference paper is 0.796 on average, the worst value is 2.985 and the best value is 0. Compared with the experimental result in Table 2, the IGSA algorithm proposed in the paper shows better optimization precision and performance.

Table 2 Optimization precision and robustness comparison of GSA and IGSA

Function	Method	Average	Best	Worst	Success rate (%)
F_1	GSA	29.0528	12.9345	47.7580	0
	IGSA	2.2768×10^{-9}	5.6843×10^{-13}	9.3698×10^{-8}	100
F_2	GSA	8.4050	2.6617	24.0624	0
	IGSA	4.4409×10^{-18}	0	2.2204×10^{-16}	100
F_3	GSA	3.6446	0.9980	12.9875	4
	IGSA	1.1964	0.9980	3.9683	88
F_4	GSA	-5.9981	-10.1532	-2.6829	38
	IGSA	-9.6461	-10.1532	2.2204×10^{-16}	90

3.3 Analysis of complexity

In the practical application of GSA, the calculation burden is mainly concentrated on the calculation of fitness function value, whose calculation complexity is $O(N \times \max_it)$. However, in the IGSA, further search for the optimal agent also adds computation burden to the external circulation complexity, so the complexity of IGSA turns into $O(N \times \max_it) + O(\dim \times \max_it \times 0.3)$. But with the improvement of the optimization performance, the scale of agents or iteration steps become lower, so the computation burden of IGSA should still be decreased and less than GSA.

CPU operation time is an important index to reflect the algorithm complexity. In [1], the group scale N for all test functions is 50, the maximum number of iterations \max_it for F_1 and F_2 is 1000, and \max_it for F_3 and F_4 is 500; while in IGSA, N and \max_it are set to be 30 and 500 for all functions, respectively. Other parameters are the same. Through the simulation for these four functions by using the GSA parameters in [1] and the IGSA parameters described above, the CPU operation time for the two algorithms can be obtained (see Table 3). According to the comparison, we can easily find that IGSA algorithm uses fewer group scales and iterations, its CPU operation time is less and the result is better than those in [1].

Table 3 Comparison of CPU operation time

	F_1 (s)	F_2 (s)	F_3 (s)	F_4 (s)
GSA	9.3	9.5	5.0	2.7
IGSA	6.3	5.8	4.1	1.9

4 Neural network identification based on IGSA

4.1 Neural network design

The common neural network types for nonlinear dynamic system identification mainly include nonlinear autoregressive models with exogenous inputs (NARX) regressive neural network^[17], proportional-integral-derivative (PID) neural network, complete feedback neural network and local feedback neural network. The NARX regressive neural network is also called the time delay neural network or the nonlinear auto-regressive filter, which comprises the time delay units plus multi-layer feedforward network. It is generally used for identifying dynamic system that can be described by the nonlinear auto-regressive moving average (ARMA) model. It has clear and simple structure and is easy to be analyzed.

The neural network structure for dynamic system identification is presented in Fig. 2. The expression of NARX model is shown in (19). The current output $y(k)$ only has relation to the current input and the previous input and output.

$$y(k) = f(y(k-1), \dots, y(k-n), u(k), u(k-1), \dots, u(k-m)). \quad (19)$$

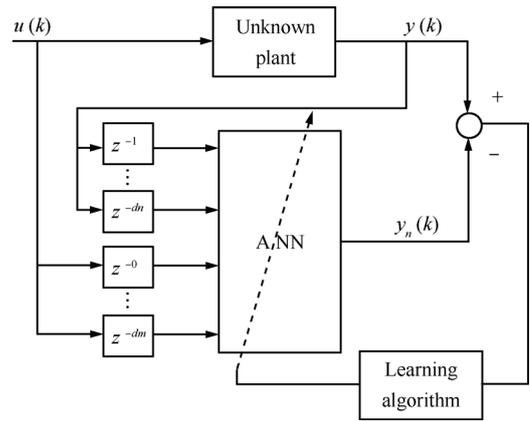


Fig. 2 NARX regressive neural network structure for dynamic system identification

Neural network training is an optimization problem of high-dimension multimodal function, which is to optimize the weights and biases by the learning algorithm and minimize an objective function of errors between the real and estimated values^[18]. In the paper, the root mean square error function (RMES) is chosen as the objective function described in (20). It is also the fitness function of the IGSA algorithm.

$$\min : J = \sqrt{\sum_{i=1}^N \frac{(y(i) - y_n(i))^2}{N}} \quad (20)$$

where $y(i)$ is the i -th real value of the sample, and $y_n(i)$ is the i -th network output value of the sample.

NARX regressive neural network structure is n - m - h , where n, m, h denotes the unit numbers of input layer, hidden layer and output layer, respectively. In order to make the variable connection between IGSA and neural network convenient, the bias information is included in the weight value, i.e., increase one dimension upon the input dimension number, and the constant input of this dimension is -1 , and then the bias will be included in the connection weight $w1$ between the input layer and hidden layer, so $w1$ turns into $(n+1) \times m$ dimensions. For the same reason, increase one more hidden unit, and the unit value is constant 1, independent of the input layer. Then incorporate the hidden unit into the connection weight $w2$ between the hidden layer and output layer, so $w2$ will turn into $h \times (m+1)$ dimensions.

By the encoder and decoder, information is transmitted between IGSA and the neural network, and the process of encoding and decoding is similar to those of chromosome in GA. As there are too many parameters in the neural network, the data will become too long if binary encoding is adopted, and it will also result in the decrease of calculation speed and precision. So the decimal encoding scheme is adopted.

4.2 Neural network design

In order to verify the effectiveness of the proposed algorithm for identification of a real process device, a magnetic levitation system is chosen as the identification object. The structure of the system is shown in Fig. 3. The equation of

motion for this system is

$$\frac{d^2y(t)}{dt^2} = -g + \frac{\alpha}{M} i^2(t) - \frac{\beta}{M} \frac{dy(t)}{dt} \tag{21}$$

where $y(t)$ is the distance of the magnet above the electromagnet, $i(t)$ is the current flowing in the electromagnet, M is the mass of the magnet, and g is the acceleration of gravity. The parameter β is a viscous friction coefficient that is determined by the material in which the magnet moves, and α is a field strength constant that is determined by the number of turns of wire on the electromagnet and the strength of the magnet. The system is a typical non-linear dynamic system, which is appropriate for modeling based on the NARX regressive neural network.

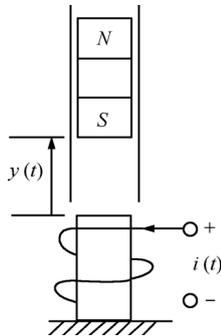


Fig. 3 NARX regressive neural network structure for dynamic system identification

In Fig. 4, samples collected for training the NARX regressive neural network are normalized into $[-1, 1]$, where the dotted line indicates the normalized voltage applied upon the electro magnet and the real line indicates the normalized position that the permanent magnet suspends above the electro magnet. The sampling period for the system is 0.3s, and 130 groups of data are sampled. The former 100 groups are used for training the network, and the latter 30 groups are used for checking the generalization ability of the network.

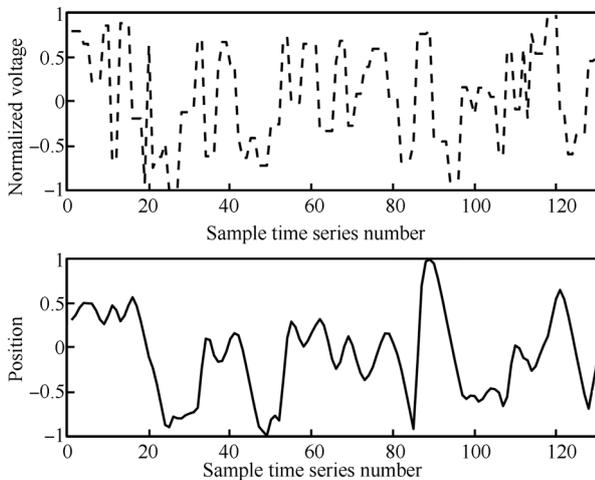


Fig. 4 NARX regressive neural network structure for dynamic system identification

The experimental parameters are set as follows.

1) NARX regressive neural network parameters: Network structure is 5-10-1, network input variable and output variable are voltage and position shown in Fig. 4, respectively; the time delay parameters are both 2.

2) IGSA algorithm parameters: The maximum number of iterations max_it is 200; group scale N is 50; dimension number dim is 71; search scope is $[-4, 6]^{dim}$; J_r is 0.5. This parameter setting is also used for the GSA algorithm.

3) BP algorithm parameters: Learning factor is 0.1; initial values of weights and biases are random numbers between 0 and 1; the maximum number of iterations is 1500.

4) GA algorithm parameters: The group number and maximum number of iterations are the same as IGSA; and the selection, crossover and mutation functions are subject to the roulette, scattered and uniform types, respectively; and probabilities of crossover and mutation are 0.8 and 0.1, respectively.

GSA, IGSA, BP and GA algorithms are used for training the NARX neural network, respectively, to fit with the real magnetic levitation system; and different convergence curves of each algorithm are shown in Fig. 5. The training errors and generalization errors for the four algorithms above are summarized in Table 4. From Fig. 5 and Table 4, we can find that the training effect for GA model is the worst. In the preliminary stage, BP model shows the fastest convergence speed, but it is liable to fall into the local optimization. Although its average training error is better than the GSA model, its generalization ability is a little weaker. Benefiting from the trial-and-error method for the optimal agent in the preliminary stage, the convergence curves of IGSA model are located under GSA. Benefiting from the further search optimization for the optimal agent and the orbital change operation of poor agents in the later stage, the optimization precision of IGSA model is raised quickly. As a whole, the performance of IGSA model is obviously better than GSA model, and it has the best performance among these four algorithms. The IGSA algorithm is very effective on the neural network identification for non-linear dynamic system.

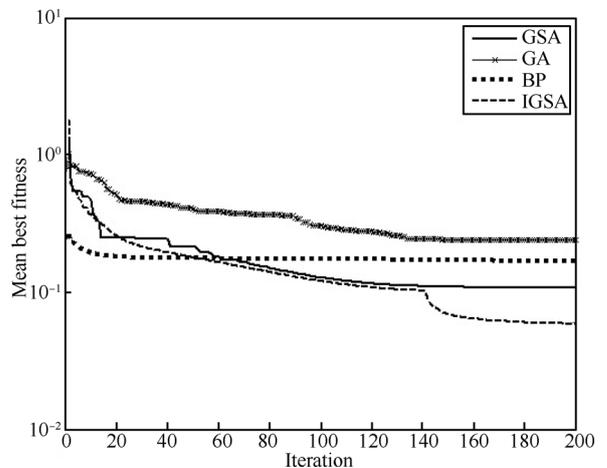


Fig. 5 Convergence curves of NARX regressive neural network based on GSA, IGSA, BP and GA

Table 4 Performance comparison for four algorithms

Name of neural network	Type	Training error	Generalization error
GSANN	Best	0.0578	0.0527
	Average	0.1083	0.0835
IGSANN	Best	0.0350	0.0418
	Average	0.0432	0.0735
BPNN	Best	0.0718	0.0838
	Average	0.0937	0.1084
GANN	Best	0.1061	0.1261
	Average	0.1806	0.1890

5 Conclusions

In this paper, an improved gravitational search algorithm (IGSA) is proposed and applied to the identification of dynamic neural network system. IGSA improves the original algorithm in three main aspects. First, inspired by the orbit change of satellites, we introduce an orbit change for poor agents to help them jump out of local minimum. Second, the coordinate descent method is introduced and applied to the optimal position search to establish an effective local search mechanism. Third, a trial-and-error method is used to update the optimal agent. The IGSA is easy to implement and can effectively reduce the iterative time. Compared with GSA on optimizing four well-known benchmark functions, our improved algorithm has been testified to possess excellent performance in terms of accuracy, convergence rate, stability and robustness. The IGSA together with BP, GA and GSA are applied to the neural network identification of a magnetic levitation dynamic system. Simulation results show that the IGSA algorithm has the lowest training error and generation error, which proves that it opens a new effective source for solving the non-linear dynamic system identification problems based on neural network.

References

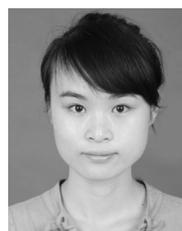
- [1] E. Rashedi, E. Nezamabadi-Pour, S. Saryazdi. GSA: A gravitational search algorithm. *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.
- [2] E. Rashedi, E. Nezamabadi-Pour, S. Saryazdi. Filter modeling using gravitational search algorithm. *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 117–122, 2011.
- [3] M. A. Behrang, E. Assareh, M. Ghalambaz, M. R. Assari, A. R. Noghrehabadi. Forecasting future oil demand in Iran using GSA (gravitational search algorithm). *Energy*, vol. 36, no. 9, pp. 5649–5654, 2011.
- [4] W. X. Gu, X. T. Li, L. Zhu, J. P. Zhou, Y. M. Hu. A gravitational search algorithm for flow shop scheduling. *CAAI Transactions on Intelligent Systems*, vol. 5, no. 5, pp. 411–418, 2010. (in Chinese)
- [5] M. Khajezadeh, M. R. Taha, A. El-Shafie, M. Eslami. A modified gravitational search algorithm for slope stability analysis. *Engineering Applications of Artificial Intelligence*, vol. 25, no. 8, pp. 1589–1597, 2012.
- [6] U. Güvenç, Y. Sönmez, S. Duman, N. Yörükeren. Combined economic and emission dispatch solution using gravitational search algorithm. *Scientia Iranica*, vol. 19, no. 6, pp. 1754–1762, 2012.
- [7] R. K. Swain, N. C. Sahu, P. K. Hota. Gravitational search algorithm for optimal economic dispatch. *Procedia Technology*, vol. 6, pp. 411–419, 2012.
- [8] S. M. Yang, G. S. Lee. Vibration control of smart structure by using neural networks. *Journal of Dynamic Systems, Measurement, and Control*, vol. 119, no. 1, pp. 34–39, 1997.
- [9] V. Maniezzo. Genetic evolution of the topology and weight distribution of neural networks. *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 39–53, 1994.
- [10] S. A. Billing, G. L. Zheng. Radial basis function network configuration using genetic algorithms. *Neural Networks*, vol. 8, no. 6, pp. 877–890, 1995.
- [11] B. D. S. L. P. De Lima, B. P. Jacob, N. F. F. Ebecken. A hybrid fuzzy/genetic algorithm for the design of offshore oil production risers. *International Journal for Numerical Methods in Engineering*, vol. 64, no. 11, pp. 1459–1482, 2005.
- [12] A. Rovira, M. Valdes, J. Casanova. A new methodology to solve non-linear equation systems using genetic algorithms. Application to combined cycle gas turbine simulation. *International Journal for Numerical Methods in Engineering*, vol. 63, no. 10, pp. 1424–1435, 2005.
- [13] J. Kennedy, R. C. Eberhart, Y. Shi. *Swarm Intelligence*, San Francisco, USA: Morgan Kaufman, pp. 249–375, 2001.
- [14] M. S. Bazaraa, H. D. Sherali, C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*, Hoboken, New Jersey: John Wiley & Sons, Inc, pp. 365–368, 2005.
- [15] Y. Zhang, D. W. Gong, W. Q. Zhang. A simplex method based improved particle swarm optimization and analysis on its global convergence. *Acta Automatica Sinica*, vol. 35, no. 3, pp. 289–297, 2009. (in Chinese)
- [16] S. J. Jia, B. Du. Hybrid optimized algorithms based on the Rosenbrock search method and dynamic inertia weight PSO. *Control and Decision*, vol. 26, no. 7, pp. 1060–1064, 2011. (in Chinese)
- [17] G. Lü, Y. Fan, G. G. Li. Hybrid nonlinear autoregressive neural networks for permanent-magnet linear synchronous motor identification. *Control Theory & Applications*, vol. 24, no. 1, pp. 99–102, 2007. (in Chinese)
- [18] P. Tahmasebi, A. Hezarkhani. A fast and independent architecture of artificial neural network for permeability prediction. *Journal of Petroleum Science and Engineering*, vol. 86–87, pp. 118–126, 2012.



Bao-Chang Xu graduated from Northeast Petroleum University (NPU), China in 2000. He received his M.Sc. degree from NPU in 2000, and his Ph.D. degree from Beijing University of Aeronautics and Astronautics, China in 2005. He is currently an associate professor of control theory and control engineering in China University of Petroleum (Beijing), China.

His research interests include system identification and advanced control, image processing, multisensor data fusion and soft sensor technology.

E-mail: xbcyl@163.com (Corresponding author)



Ying-Ying Zhang received her B.Eng. degree in automation from China University of Petroleum (Beijing), China in 2010. She is currently a Ph.D. candidate in control theory and control engineering at the College of Geophysics and Information Engineering, China University of Petroleum (Beijing).

Her research interests include neural network based system identification.

E-mail: kikia1b2@163.com