Motion model identification of rescue robot based on optimized Jordan neural network

Guangbin Zhang\textsuperscript{1,a}, Runmei Zhang\textsuperscript{2,b}, Guangyin Wang\textsuperscript{1,c} and Yulu Wu\textsuperscript{1,d}

\textsuperscript{1}School of electronics and Information Engineering, Anhui Jianzhu University, Hefei, 230601 China
\textsuperscript{2}School of Mechanical and Electrical Engineering, Anhui Jianzhu University, Hefei, 230601 China

E-mail: \textsuperscript{a}16011513@qq.com, \textsuperscript{b}zhangrong@ahjzu.edu.cn, \textsuperscript{c}wangguangyin1123@163.com, \textsuperscript{d}2286734949@qq.com

Abstract. Considering the influence of various factors, such as speed, angle, depth of water, weight, and water flow, on the underwater rescue robot, a method based on neural network is proposed. According to the characteristics of Elman and Jordan neural network, a new dynamic neural network is constructed. The network can be used to remember the state of the hidden layer and increase the feedback of the output node. The improved Jordan network is optimized by chaos particle swarm optimization algorithm. The optimized neural network is applied to identify the dynamic model of the underwater rescue robot. The simulation results show that the neural network has good convergence speed and accuracy.

1 Introduction
Underwater rescue robot is a complex nonlinear system. It has a multi degree of freedom dynamics model, as well as unstructured uncertainty. It is difficult to establish accurate mathematical model\textsuperscript{[1]}. At present, the adaptive control algorithms and model identification of various underwater robots have been developed to some extent, the control problems caused by the uncertainty of the model are overcome\textsuperscript{[2]}. However, at present, most of the accurate identification needs all the state information of the underwater robot system, including the position and speed of each loop. For underwater rescue robots, some information such as velocity, flow and angle must be obtained indirectly by means of reconstruction. These factors greatly increase the difficulty of system identification\textsuperscript{[3]}.

Neural network has the ability to approximate any nonlinear mapping with arbitrary precision. Its learning ability reduces uncertainty and increases the generalization ability to adapt to environmental changes. Therefore, it is not necessary to establish an accurate mathematical model in the process of system identification. The research shows that using neural network to simulate the motion of underwater rescue robot has the characteristics of high speed and high precision\textsuperscript{[4]}.

In this paper, the Jordan neural network is improved and a hybrid neural network is established. The chaotic particle swarm optimization algorithm is used to optimize the network structure. The simulation results show that the proposed neural network model can be applied to the identification of the underwater vehicle motion model with high accuracy and fast speed.

2 Jordan neural network using particle swarm optimization
2.1 Basic Jordan network

Jordan neural network is a forward network based on the feedback output layer to the input layer. Similar to the Elman neural network, the Jordan network is not easy to identify high order systems. Therefore, the improved Jordan network absorbs the characteristics of Elman network and basic Jordan network. In addition to the input layer, the hidden layer and the output layer, there is a special receiving layer. The receiving layer is a delay unit, which can receive the feedback signal from the hidden layer, and remember the output value of the hidden layer unit at the previous moment. The output of the receiving layer is delayed and stored, and the self feedback factor is used to modify the input layer. In addition, the Jordan network in this paper contains feedback from the output layer. Specific Jordan network structure shown in figure 1.

Suppose that the input of the network is the k dimension vector x, the output of the hidden layer is m dimension vector p, the output of the q is the m vector, and the output layer is an n-dimensional vector y. The weights of the hidden layer and the input layer, the layer and the output layer are matrix U, V, W. The mathematical model of Jordan neural network:

\[
y(k) = g(Wp(k)) \quad \text{(1)}
\]

\[
p(k) = f(Vq(k) + Ux(k-1)) \quad \text{(2)}
\]

\[
q(k) = p(k-1) + aq(k-1) \quad \text{(3)}
\]

\[
x(k) = y(k) \quad \text{(4)}
\]

In formula (1), \(g(\cdot)\) is the activation function of the output layer unit, and is a linear combination of the output of the hidden layer. In formula (2), \(f(\cdot)\) is the activation function of the hidden layer unit, which is called Sigmoid function \(f(x) = 1/(1+e^{-x})\). In the formula (3), a is a self feedback gain factor of the receiving layer, the value range (0,1). The existence of a can better adjust the feedback signal of hidden layer\(^5\).

In this paper, the learning algorithm of Jordan network is based on ordered chain. Define the error function of K time system such as formula (5):

\[
E(k) = \frac{1}{2} \sum_{i=1}^{n} (y_i(k) - d_i(k))^2 \quad \text{(5)}
\]

In formula (5), \(y_i(k)\) is the actual output of the i node, \(d_i(k)\) is the expected output of the i node. The change of the weight of W between the hidden layer and the output of the network, such as formula (6):

\[
W(k + 1) = W(k) + \eta \left( - \frac{\partial E(k)}{\partial W} \right) \quad \text{(6)}
\]

In the formula (6), \(\eta\) is the learning step. Similarly for weights U and V. This modified gradient descent algorithm for network weights has a slow learning rate and is easy to produce local minimum value. The self feedback gain factor A, which is used to fix the network, is determined by the trial method, so the learning efficiency is low.
2.2 Chaos particle swarm optimization algorithm

The basic idea of particle swarm optimization is to find the optimal solution through the cooperation and information sharing among individuals[6]. The adaptive optimization algorithm is based on the concept of community and. The update of the particle state is based on the following rules: to maintain its own inertia; to change according to its optimal position; to change according to the best position of the swarm. Located in N dimensional search space. There are M particles to form a group. The position of each particle represents a possible solution to the optimization problem in space. Assume that the location of the i particle is: $x_i=(x_{i,1}, x_{i,2},...x_{i,n})$, speed: $v_i=(v_{i,1},v_{i,2},...v_{i,n})$. The particle is currently searching for the optimal location: $p_i=(p_{i,1},p_{i,2},...p_{i,n})$. The optimal locations currently searched for the population are: $P_e=(p_{e,1},p_{e,2},...p_{e,n},...p_{e,N})$, where i=1,2, ... M, n=1,2,...N. The particle updates its velocity and position according to formula (7) and (8), where $c_1$ and $c_2$ are nonnegative acceleration constants, which can reduce the local minimum value. $c_1$ adjust the step size of the particle moving to its optimal position, $c_2$ adjust the size of the particle moving to the global optimal position. $r_1$ and $r_2$ are random factors (0,1), which can increase the diversity of population.

$$v_{i,n}(k+1) = v_{i,n}(k) + c_1 r_1 (p_{i,n}(k) - x_{i,n}(k))$$ (7)

$$x_{i,n}(k+1) = x_{i,n}(k) + v_{i,n}(k+1)$$ (8)

Particle swarm optimization algorithm in the optimization process, all particles are moving in the direction of the optimal solution[7]. The phenomenon of particle aggregation, tend to the same, will gradually lose diversity, resulting in slow convergence rate. Therefore, chaotic particle swarm optimization is used to solve the problem of premature convergence. The core idea of the algorithm is to judge whether the current particle is in an early stage by means of early diagnosis. If so, the spatial location of the chaotic search is decided by the optimal position. Chaos search makes local search in the narrow space. At the end of the search, the optimal position is used as the optimal position of particle swarm, which leads to the local optimum. The algorithm solves the premature convergence problem of particle swarm optimization algorithm, accelerates the convergence speed and improves the convergence precision. The evolution of chaotic variables is based on the Logistic equation, as shown in formula (9).

$$cx_{i}^{k+1} = 4cx_{i}^{k} (1-cx_{i}^{k})$$ (9)

In formula (9), $k=1,2,...$, $K$. $cx_i$ are chaotic variables, which are expressed after K iterations. $K$ means the number of iterations of chaotic map.

2.3 Improved Jordan network learning algorithm

Chaotic particle swarm optimization algorithm for Jordan neural network optimization, including two aspects. First, in the network learning stage, optimize the connection weights between the layers in the network. Second, optimize the topology of the network[8,9]. This study is mainly used to optimize the weights of the network.

Step 1: the connection weights between different layers are encoded using matrix coding. The particle is encoded into a matrix, each particle represents a weight, so the structure in Figure 1 has three weight matrices.

Step 2: set the particle population size M, learning factor $c_1$ and $c_2$, the particle swarm optimization algorithm evolutionary maximum $T_{maxpso}$, iterative k chaotic map, the number of particle swarm energy function changes $T_{pso}$, the evolution of t, flag of energy function changes sign. The initial energy function value of each particle is evaluated, pbest, is the position of the current particle, and gbest, is the position of the best particle in the particle swarm.

Step 3: update the speed and position of all particles according to the formula (7) and (8). The particles which represent the new weights are generated, and the chaotic points of K points are generated by using the formula (9). Select the lowest point of the energy function as the new location of the particle.
Step 4: evaluate the fitness value of each new particle. If the energy function value of the i particles is lower than that of the previous update, update the location of the particle pbest. If the optimal pbest in the particle is better than the optimal value of the population gbest, then update the gbest, and set flag 0. Otherwise, the value of flag plus 1.

Step 5: if flag>T_pso, local search near gbest. If the search results are better than gbest, replace the pbest with this result and update the corresponding location. Otherwise, replace the worst performance of the pbest, with this result, the flag is 0, t value plus 1. If t>T_maxpso, go to step 6, otherwise go to step 3. If flag<T_pso, then t=t+1. Judge t>T_maxpso, go to step 6. Otherwise go to step 3.

Step 6: the end of the optimization step, the output information.

The result of this output is the best self feedback gain factor and the total weight of Jordan neural network.

3 Experiment and result analysis

The dynamic equation of the underwater rescue robot in the carrier coordinate system can be expressed as:

\[ M\ddot{v} + C(v)v + D(v)v + G(\eta) + F = u \]  \hspace{1cm} (10)

\[ \dot{\eta} = J(\eta)v \]  \hspace{1cm} (11)

There are six degrees of freedom in this paper. In the formula (10): M is symmetric and bounded positive inertia matrix, including additional quality. C(v) represents the centripetal moment, including the additional mass produce the centripetal force and Coriolis force. v is the velocity vector, including linear velocity and angular velocity. D (v) is a hydrodynamic lift and drag matrix. G(\eta) on behalf of the gravity and torque vector is one order differentiate bounded function. \eta is rescue robot position and posture vector. F is an unstructured dynamical uncertainty, including friction and other disturbances. u is the control vector applied to the rescue robot. J (\eta) for conversion matrix\(^\text{[2]}\).

According to the mathematical model and the actual situation of the underwater rescue robot, seven factors which are important to the underwater action of the rescue robot are selected as input. The longitudinal velocity, lateral velocity, yaw angle, pitch angle, flow momentum, depth of water, weight. Some of these factors are obtained directly, and some are obtained by indirect methods. Due to the limitations of the conditions and the reasons for the simplified model validation, other factors, such as vertical velocity, pitch angle, and distance from the bottom, are not taken into account. The number of neurons in the hidden layer is adjusted by trial method in the training process. When the number of neurons in the hidden layer is less than 10, it is difficult to achieve the required accuracy. When the number of neurons in the hidden layer was 10 to 16, the number of training was significantly reduced. Taking into account the complexity of the robot motion model and computing efficiency, the number of hidden layer nodes is 12. There are four neurons in the output layer, that is, longitudinal velocity, lateral velocity, yaw angle, pitching angle. The structure of the whole network is in the form of 7-12-12-4. It includes 7 input layer nodes, 12 hidden layer nodes, 12 nodes, and the network structure of the 4 output nodes. Some parameters in the network: particle population size M=30, learning factor c\(_1\)=2.46, c\(_2\)=1.18 the maximum number of iterations T\(_{\text{maxpso}}\)=300. the range of chaotic variables is [0,1.0]. r\(_1\), r\(_2\), diversity factor and random disturbance factor \eta are random numbers of(0,1). Undertake connection layer self feedback gain factor a is 0.5.
Through the use of depth meter, gyroscope, torque meter, Doppler velocimeter and other equipment, get the underwater rescue robot in some specific conditions of the original data, as a neural network model of the sample data. The actual measured value of the sensor is the expected output of the network. Using Matlab software to carry out the simulation experiment, the data are divided into two categories: training data and test data. The data should be normalized before training the original data. The results are obtained and the results are inverse normalized to normal data.

Figure 2 is the error curve of the basic Jordan network using training data to learn. Figure 3 is the error curve of the particle swarm optimization Jordan network using the training data to learn. Comparing Figure 2 and figure 3, we can see that both the evolutionary algebra and the error performance of the optimized Jordan network are better than the basic network.  

Figure 4 is the result of training data. It can be seen that the fitting degree of learning data obtained by training is ideal.
Figure 5 shows the results of the test data on a well trained network.

Figure 6 is the residual error between the test data and the expected results. The mean square error of MSE is 0.12. The results show that the network performance is good, and it can be used to identify the motion of the rescue robot.

4 Conclusion
According to the characteristics of Elman network and Jordan network, an improved Jordan neural network is constructed. In this network, the output of the hidden layer is fed back to the structure layer. Considering the influence of the feedback information of each layer on the signal processing ability of the network, the feedback of the output layer node is added to the input layer. A chaotic particle swarm optimization algorithm based on global search and local optimization is proposed, which is used to improve the weights of Jordan network. Using the optimized Jordan neural network to identify the underwater rescue robot. The simulation results show that the network has high
prediction accuracy and can be used to establish a multi factor dynamic model of underwater rescue robot.

Acknowledgements
This work was supported by Anhui quality engineering(2014zdjy091), 2015 year study abroad merit funding project, Anhui Jianzhu University doctoral start-up fund and Yi Hai talent project.

References
[1] Zhai Y., Liu L., Lu W.. The Application of Disturbance Observer to Propulsion Control of Sub-mini Underwater Robot[J]. Computational Science and Its Applications, 60, pp.590-593,(2010).
[2] Roland Dőczi, Bence Takács, Haidegger T.. Orientation and Depth Control in Rippling Water for an Autonomous Underwater Robot[J]. Advances in Robot Design and Intelligent Control, 540, pp.693-703,(2016).
[3] Steffen Weißmann, Pinkall U.. Underwater rigid body dynamics[J]. Acm Transactions on Graphics, 31(4), pp.1-7,(2012).
[4] Desell T., Clachar S., Higgins J., Wild B.. Evolving Neural Network Weights for Time-Series Prediction of General Aviation Flight Data[C]. Lecture Notes in Computer Science, 8672, pp.771-776,(2014).
[5] Bilski J., Smolał J.. Parallel Approach to Learning of the Recurrent Jordan Neural Network. Artificial Intelligence and Soft Computing[C]. Lecture Notes in Computer Science, 7894, pp. 32-36, (2013).
[6] Mao Chengying, Yu Xinxin , Xue Yunzh.Algorithm Design and Empirical Analysis for Particle Swarm Optimization-Based Test Data Generation[J]. Journal of Computer Research and Development, 50 (2), pp. 260-268, (2013).
[7] Liu J, Ma H, Ren X, et al. Optimal formation of robots by convex hull and particle swarm optimization[C]. IEEE,8034 pp. 104-111,(2013).
[8] Buyukada M.. Co-combustion of peanut hull and coal blends: Artificial neural networks modeling, particle swarm optimization and Monte Carlo simulation[J]. Bioresource Technology, 216, pp. 280-286,(2016).
[9] Sandeep Chakravorty,Amrita Chakrabarti. Hidden layer optimization of neural network using computational technique[C]. Proceedings of the International Conference on Advances in Computing, Communication and Control, ACM ,pp.559-561,(2009).
[10] Chen H., Zeng Z., Tang H.. Study on Landslide Deformation Prediction Based on Recurrent Neural Network under the Function of Rainfall[C]. Neural Information Processing, Lecture Notes in Computer Science, 7666, pp. 683-686,(2012).