

Symbiotic Neuron Evolution of a Neural-Network-Aided Grey Model for Time Series Prediction

Shih-Hung Yang

Institute of Electrical and Control Engineering
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.
pshong.ece93g@nctu.edu.tw

Yon-Ping Chen

Institute of Electrical and Control Engineering
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.
ypchen@cc.nctu.edu.tw

Abstract—This paper introduces a symbiotic neuron evolution algorithm (SNEA) to determine the topology of a neural-network-aided grey model (NNAGM) for time series prediction problem. The SNEA uses an evolutionary approach to evolve partially connected neural networks (NNs) and determine the number of hidden neurons. To achieve symbiotic evolution, SNEA first establishes a neuron population where each neuron is randomly created, and evaluates the neurons by constructing NNs with different numbers of neurons. Each neuron shares fitness from participating NNs. This algorithm then performs evolution on the neuron population by crossover and mutation based on neuron fitness. An NNAGM designed by SNEA is applied to the prediction problems and compared with other methods. The experimental results show that SNEA can produce an NNAGM with appropriate topology and higher prediction performance than other methods.

Keywords—grey model; symbiotic evolution; neural network; prediction.

I. INTRODUCTION

In time series prediction, a small subset of recent time series information is usually used for prediction. The prediction assumes that the trends of the previous time series will be maintained in the future [1]. This approach is called lagged variables, shift registers, or tapped delay line. This means that accurate predictions can be obtained by using only a finite segment of previous data up to the prediction point [2]-[4]. Many researchers have developed variant systems to address the forecast problem in time series prediction, such as grey theory [5], [6] and neural networks (NNs) [7], [8].

Grey theory, a numerical method, is used to cope with systems with partial information or system dynamic model. The first-order single variable grey model (GM(1,1)) was developed for prediction [9]-[11] based on the gray theory. The GM(1,1) first transforms an amount of historical data into a more exponential-like and smooth signal. The grey differential equation, modified from the first-order ordinary differential equation, is then adopted to calculate the predictive value. Therefore, GM(1,1) can achieve high accuracy in prediction of exponential-type signals, offer the advantages of simplicity, and reduce computation time. However, the precision of this method of prediction is not sufficient for other types of signals.

In addition to GM(1,1), have been developed as a forecasting method based on soft-computing [12], [13]. Cybenko [14] and Hornik et al. [15] suggested that a single hidden layer is sufficient for NNs to approximate any complex nonlinear function and predict time series. Therefore, most researchers use only one hidden layer for prediction purposes. In general, NNs with fewer hidden neurons usually have better generalization ability and present a less severe overfitting problem. However, NNs with too few hidden neurons may not be able to accurately model and learn the data due to limited information processing capability [16], [17]. Thus, determining the number of hidden neurons can be difficult.

Based on the complementary advantages of GM(1,1) and NN, our previous work [18], [19] integrated GM(1,1) and NN into a fusion scheme, neural-network-aided grey model (NNAGM). The NNAGM employed GM(1,1) to predict signals and adopted NN to learn the prediction error of GM(1,1). The NN used in NNAGM has a fully connected topology and the number of hidden neurons is determined by trial-and-error. However, it is possible to have networks that are not fully connected in prediction problems [20]. Thus, automatic design of the topology of NNs is necessary to enable efficient solution of various prediction problems.

Symbiotic evolution is an implicit fitness-sharing algorithm used in an immune system model [21], [22]. In symbiotic evolution, each individual in the population represents a partial solution to a given problem, and complete solutions are constructed from several individuals. The partial solutions can be considered specializations that ensure diversity and prevent a population from converging into suboptimal solutions. The work of [22] proposed symbiotic adaptive neuron evolution to develop NNs according to a population of neurons. The performance of a neuron is determined by how well it cooperates with the other neurons which it combines. This process shows that a neuron that cooperates well with one set of neurons may cooperate poorly with other sets of neurons. However, the number of hidden neurons must be assigned prior to the evolution process. Previous studies have applied symbiotic evolution to design fuzzy controller and neuro-fuzzy systems [23]-[27]. The results of these studies have demonstrated the efficiency and feasibility of symbiotic evolution in reinforcement learning.

This paper presents a symbiotic neuron evolution algorithm (SNEA) to design an NNAGM. This algorithm determines the topology of NNAGM, including the number of hidden neurons and connected topology. Section II introduces the concept of NNAGM. Section III indicates that the proposed SNEA does not develop NNs with a fixed number of hidden neurons but rather constructs NNs of different size by selecting the neurons from the neuron population. The neurons are evolved through the crossover and mutation operations, according to the shared fitness from the participating NNs. Section IV demonstrates the ability of the proposed SNEA to create an NNAGM for two prediction problems. Finally, Section V presents the conclusion of this study.

II. NEURAL-NETWORK-AIDED GREY MODEL

An NNAGM combines a GM(1,1) and an NN where GM(1,1) mainly predicts signals and NN learns the prediction error of GM(1,1) [18]. The structure of an NNAGM is depicted in Fig. 1.

Consider a discrete data sequence of length $n \geq 4$ formed as the following column vector

$$\mathbf{x}^{(0)} = [x^{(0)}[1] \ x^{(0)}[2] \ \dots \ x^{(0)}[n]]^T \quad (1)$$

where each element has the same numeric sign. In general, the GM(1,1) adopts three fundamental operations, given as

Accumulate Generating Operation (AGO):

$$x^{(1)}[k] = \sum_{l=1}^k x^{(0)}[l], \quad k = 1, 2, \dots, n \quad (2)$$

Mean Generating Operation (MGO):

$$z^{(1)}[k] = \alpha x^{(1)}[k] + (1 - \alpha)x^{(1)}[k-1], \quad k = 2, 3, \dots, n \quad (3)$$

Inverse Accumulate Generating Operation (IAGO):

$$x^{(0)}[k] = x^{(1)}[k] - x^{(1)}[k-1], \quad k = 2, 3, \dots, n \quad (4)$$

where α is often set as 0.5. According to the GM(1,1) [10], its grey differential equation is presented as

$$x^{(0)}[k] + az^{(1)}[k] = b, \quad k = 1, 2, \dots, n \quad (5)$$

where a is the development coefficient and b is the grey input. Both a and b are unknown and have to be determined first by rearranging (5) into the following matrix form

$$\mathbf{x}^{(0)} = \mathbf{X}^{(1)} \begin{bmatrix} a \\ b \end{bmatrix} \quad (6)$$

where

$$\mathbf{X}^{(1)} = \begin{bmatrix} -\frac{x^{(1)}[1]}{2} & -\frac{x^{(1)}[1] - x^{(1)}[1]}{2} \\ 1 & 1 \\ \dots & \dots \\ \dots & -\frac{x^{(1)}[n] - x^{(1)}[n-1]}{2} \\ & 1 \end{bmatrix}^T \quad (7)$$

Then a and b could be solved by the least square method as below

$$\begin{bmatrix} a \\ b \end{bmatrix} = (\mathbf{X}^{(1)T} \mathbf{X}^{(1)})^{-1} \mathbf{X}^{(1)T} \mathbf{x}^{(0)} \quad (8)$$

Based on the GM(1,1), the solution of the grey first-order differential equation (5) is estimated as [10]

$$\hat{x}^{(1)}[n+j] = (x^{(0)}[1] - b/a)e^{-a(n+j-1)} + b/a, \quad j \geq 1 \quad (9)$$

Further applying the IAGO in (4) to (9) yields

$$\hat{x}^{(0)}[n+j] = (x^{(0)}[1] - b/a)e^{-a(n+j-1)}(1 - e^a) \quad (10)$$

which is the so-called predictive data. The one-step-ahead predictive value is then calculated by setting $j = 1$, which is the first predictive value after the original data sequence.

Since the GM(1,1) performs the prediction via the data sequence with the same numeric sign, the preprocess is employed to transform the raw data in (1) into

$$x'^{(0)}[l] = x^{(0)}[l] - \min(x^{(0)}) + \gamma, \quad l = 1, 2, \dots, n \quad (11)$$

where γ is a constant bias to avoid the output to be zero. Then the transformed value is used to estimate $\hat{x}'^{(0)}[n+j]$ from (2)–(11). As a result, the one-step-ahead predictive value is determined as

$$\hat{x}^{(0)}[n+j] = \hat{x}'^{(0)}[n+j] + \min(x^{(0)}) - \gamma. \quad (12)$$

Therefore, the GM(1,1) can process the data sequence with different numeric signs.

The NNAGM receives the input data $\mathbf{x}^{(0)}[k] = [x^{(0)}[k-n+1] \ x^{(0)}[k-n+2] \ \dots \ x^{(0)}[k]]^T$ at time step k corresponding to $\mathbf{x}^{(0)} = [x^{(0)}[1] \ x^{(0)}[2] \ \dots \ x^{(0)}[n]]^T$ in (1). Based on (12), the one-step-ahead predictive value is $\hat{x}^{(0)}[k+1]$ whose corresponding prediction error is $e_{\text{GM}}[k]$. To improve the prediction accuracy, NNAGM adopts NN to estimate the prediction error $\hat{e}_{\text{GM}}[k+1]$ by $[e_{\text{GM}}[k-m] \ \dots \ e_{\text{GM}}[k-1] \ e_{\text{GM}}[k]]$. The weight vector of NN is adjusted by the backpropagation algorithm.

With the fusion scheme, NNAGM integrates the outputs of both GM(1,1) and NN. The NNAGM possesses not only the capacity to predict data using GM(1,1), but also the intelligence to learn to estimate the prediction error using NN. The

remaining problem involves determining the topology of NN, i.e., the number of neurons and connections. The next section introduces the proposed SNEA employed to design the topology of an NN.

III. SYMBIOTIC NEURON EVOLUTION ALGORITHM

This section presents the proposed SNEA in detail. In its current implementation, SNEA is used to determine the structure of an NNAGM, i.e., the number of neurons and connections. The prior work [18] provided a constraint such that the topology of NN used in an NNAGM should be fully connected. Furthermore, the number of neurons is determined through a trial-and-error method. This paper uses SNEA to design NN in an evolutionary way.

The structure of an NN consists of input neurons, hidden neurons, output neurons, and connections between each layer. The number of neurons in the input and output layer is problem-dependent. For the prediction problem in Section IV, four input neurons and one output neuron are employed. Therefore, SNEA mainly determines the number of hidden neurons and the connected topology between input and hidden neurons.

To evolve the neurons, each neuron is encoded as bj , where b indicates the connection of bias and j indicates the topology related to the input neurons. The value of j is a bitwise representation and determines which input neuron is connected to the neuron bj . For a four-input-one-output prediction problem, each neuron has 2^4-1 types of topology, i.e., $j = 1, 2, \dots, 9, A, \dots, F$. Note that one input neuron should at least be connected to the hidden neuron. Fig. 2 shows two examples of the coding of two neurons and the topology of an NN constructed from the two neurons.

The main steps of SNEA are summarized in Fig. 3 and further explained as follows.

- Step 1) Create a neuron population. The neuron population consists of randomly generated N_{pop} neurons.
- Step 2) Randomly select p neurons to construct an NN in the p th group, $p = 1, 2, \dots, P$. As shown in Fig. 4, each group consists of d NNs where each NN of the p th group includes p neurons.
- Step 3) Train each NN using the backpropagation algorithm and then calculate the fitness of each NN where the fitness f_{NN} is the inverse of the root mean square error (RMSE).
- Step 4) Share the fitness of NN with each participating neuron. For NN in the p th group, each participating neuron can share fitness as f_{NN}/p .
- Step 5) Determine the participating time and fitness of each neuron. The neuron participating T NNs has its fitness f_n as

$$f_n = \frac{1}{T} \sum_{t=1}^T f_n^t \quad (13)$$

where f_n^t is the shared fitness in t th NN.

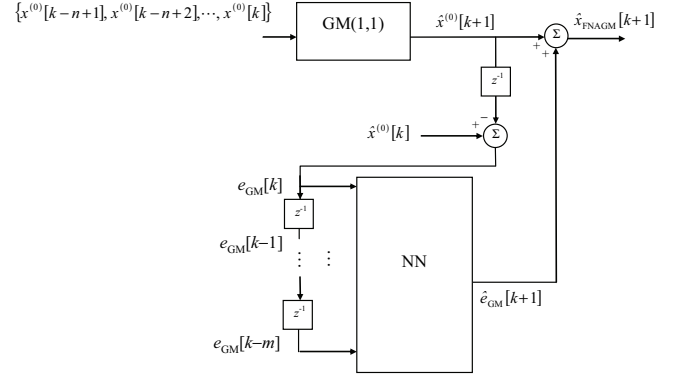


Fig. 1. Neural-network-aided grey model (NNAGM).

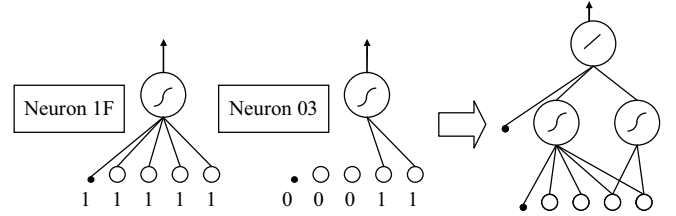


Fig. 2. Coding of two neurons and an NN constructed via two neurons.

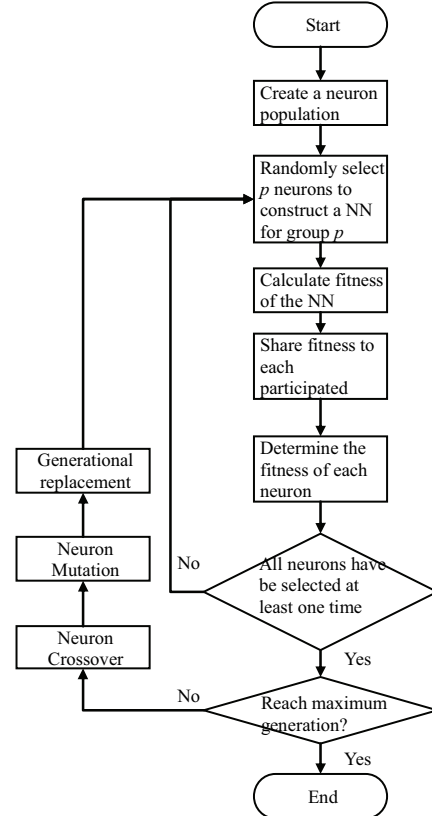


Fig. 3. Flowchart of SNEA.

- Step 6) If one of the neurons in the neuron population has not been selected at least one time, then go to Step 2); otherwise, go to Step 7).

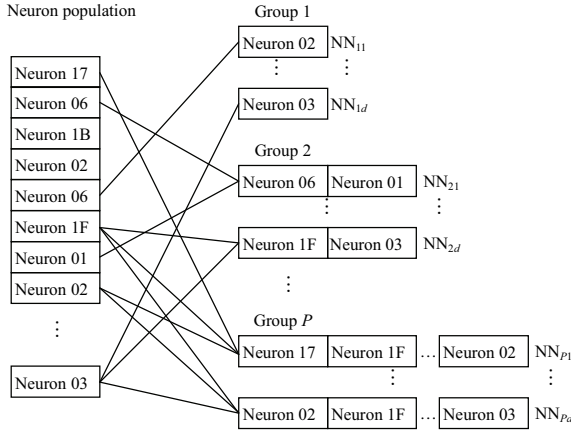


Fig. 4. Construct of NNs of each group from the neuron population.

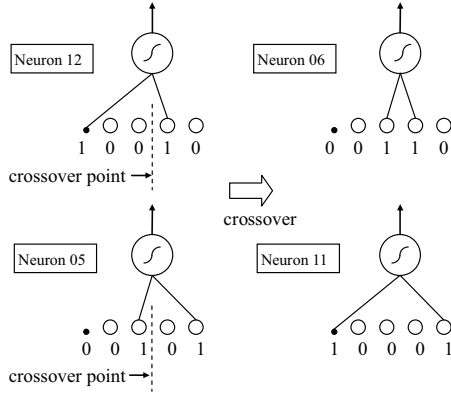


Fig. 5. Neuron crossover.

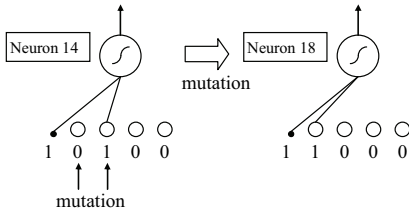


Fig. 6. Neuron mutation.

- Step 7) Check the stopping criterion. If the maximum generation is achieved, then stop the algorithm; otherwise, go to Step 8).
- Step 8) Perform the neuron crossover to the neuron population. First, the neuron crossover adopts the tournament selection to choose two parents according to their fitness f_n . The crossover point is then randomly assigned, as shown in Fig. 5. Finally, the two parents produce two children by exchanging the connections.
- Step 9) Perform neuron mutation on the children. Each connection to either the input or bias might change according to the probability of mutation p_m . For the example shown in Fig. 6, two positions of Neuron 14 are selected to be mutated and then the corresponding connections are disconnected

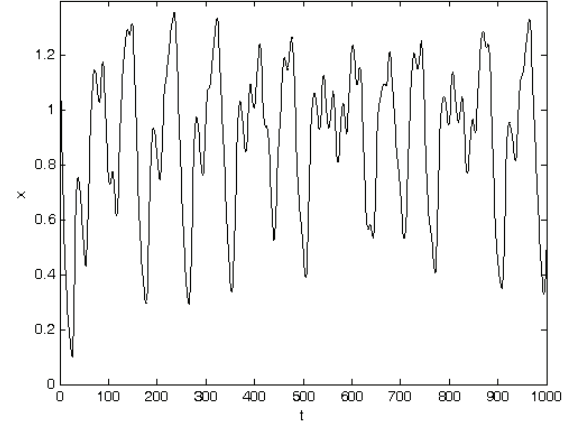


Fig. 7. Mackey-Glass time series.

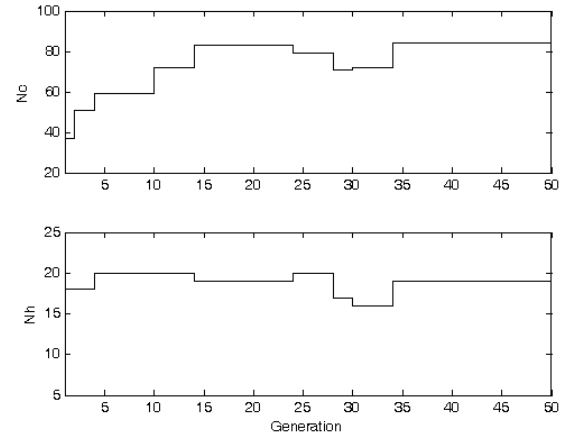


Fig. 8. Evolution progress of neurons and connections for example 1.

($1 \rightarrow 0$) or connected ($0 \rightarrow 1$). As a result, Neuron 18 is generated.

Step 10) Perform generational replacement, meaning that all children replace the parents and become parents in the next generation.

IV. EXPERIMENTAL RESULTS

This section provides two examples to verify the performance of the proposed SNEA-NNAGM. The numerical simulations were executed by MATLAB software in an Intel Pentium CPU at 1.5 GHz with 512 MBytes RAM.

Example 1: Chaotic Time Series Prediction

The Mackey-Glass time series shown in Fig. 7 is generated from the following delay differential equation

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \quad (14)$$

where $\tau = 25$ and $x(0) = 1.2$ in the simulation. The objective involves using $[x(t-3) \ x(t-2) \ x(t-1) \ x(t)]$ to predict $x(t+1)$.

TABLE I. COMPARISON OF PREDICTION RESULTS FOR MACKEY-GLASS TIME SERIES.

	GM(1,1)	Advanced GM(1,1)	NN	NNAGM	SNEA-NNAGM
Training RMSE	---	---	3.27×10^{-4}	2.47×10^{-4}	2.52×10^{-4}
Testing RMSE	1.09×10^{-2}	1.10×10^{-3}	3.58×10^{-4}	2.93×10^{-4}	2.66×10^{-4}
N_h	---	---	34	27	19
N_c	---	---	205	163	84

TABLE II. COMPARISON OF THE PERFORMANCE IN EXAMPLE 2 SUNSPOTS PREDICTION.

	GM(1,1)	Advanced GM(1,1)	NN	NNAGM	SNEA-NNAGM
Training RMSE	---	---	8.24	11.74	12.81
Testing RMSE	30.49	137.52	34.32	65.85	22.01
N_h	---	---	16	14	9
N_c	---	---	97	85	37

Thus, the input of NN of NNAGM is $[e_{GM}(t-3) \ e_{GM}(t-2) \ e_{GM}(t-1) \ e_{GM}(t)]$ and the output is $e_{GM}(t+1)$. The first 500 pairs are used as the training data and the next 500 pairs are the testing data. The parameter of NNAGM is set as $\gamma = 2$ in (12). The parameters of SNEA are set as follows: probability of crossover $p_c = 0.8$, probability of mutation $p_m = 0.5$, neuron population size $N_{pop} = 40$, number of groups $N_g = 20$, number of group members $N_m = 5$, and maximum number of generations $G_{max} = 50$.

The top side of Fig. 8 shows the number of connections of the evolved NNAGM with the best fitness, N_c , in each generation. The NN has few connections in the first generations and N_c does not always increase. The N_c drops from 83 to 79 in the 24th generation and from 79 to 71 in the 28th generation, implying that more connections do not necessarily result in superior fitness. In other words, appropriate connected topology is the key to improving fitness, not the number of connections. The evolution of NNs almost converges and no further improvement of NN is observed after the 35th generation. The highest quality structure with well-trained weights of NNAGM is then determined.

The bottom side of Fig. 8 shows the number of the hidden neurons, N_h , in the evolved NNAGM. It can be observed that N_h increases to 20 in the 4th generation and decreases to 16 in the 30th generation. As a result, the highest quality structure with well-trained weights of NNAGM has 19 hidden neurons and 84 connections. Clearly, NN of NNAGM is a partially connected network.

Table I shows the prediction results of SNEA-NNAGM compared with other methods including GM(1,1) [9], Advanced GM(1,1) [28], NN [7], and NNAGM [18]. The number of neurons of NNAGM is chosen according to the experimental tests where several NNAGMs with different numbers of neurons, from 1 to 50, were constructed. As a result, a NNAGM with 27 neurons demonstrates the highest performance. The number of neurons of NN is chosen by the same procedure. Consequently, an NN with 34 neurons has the best performance. The neurons of both NN and NNAGM are

fully connected. It can be seen that NNAGM has a better RMSE 2.93×10^{-4} than GM(1,1), Advanced GM(1,1), or NN, while SNEA-NNAGM improves the RMSE in advanced 2.66×10^{-4} using the evolutionary design of neural networks. It is clear that the evolved NNAGM via SNEA consists of a partially connected NN with 19 neurons and 84 connections. The SNEA-NNAGM obtained a compact structure with fewer neurons and connections, and is superior to NN and NNAGM according to the prediction error.

Example 2: Forecasting the Number of Sunspots

In this example, the sunspots varying nonlinearly from 1709 to 2008 were adopted. The input-output data pair is given as follows:

$$[x[k-3], x[k-2], \dots, x[k]; x[k+1]] \quad (15)$$

where the four most recent data points are the input data and one step ahead value is the output data. The sunspots of the first 150 years, from 1709 to 1858, were used as the training data and those of the next 150 years, from 1859 to 2008, were the testing data. The input of NN of NNAGM is $[e_{GM}[k-3] \ e_{GM}[k-2] \ e_{GM}[k-1] \ e_{GM}[k]]$ and the output is $e_{GM}[k+1]$. Furthermore, the training data were normalized to the range $[0, 1]$.

Fig. 9 shows the prediction results of the five methods including GM(1,1) [9], Advanced GM(1,1) [28], NNAGM [18], NN [7], and SNEA-NNAGM. The number of neurons of NNAGM is determined by the experimental tests where numerous NNAGMs with different numbers of neurons, from 1 to 50, were constructed. As a result, a NNAGM with 14 neurons has the best performance. The number of neurons of an NN was determined by the same procedure. Consequently, an NN with 16 neurons has the best performance. Note that all neurons in both NNAGM and NN were fully connected. It can be observed that the performance of GM(1,1) was better than the Advanced GM(1,1) in this example, while the performance of NNAGM was better than GM(1,1) due to the use of error prediction via NN. The SNEA-NNAGM further improved the performance by selecting the appropriate structure of NN through SNEA. Table II presents the performance comparison of the five methods. Although SNEA-NNAGM obtained a larger training RMSE than both NN and NNAGM, the testing RMSE is smaller than both of them. Furthermore, the evolved structure of NNAGM via SNEA has 9 neurons and 37 connections. Clearly, the evolved NNAGM is partially connected and is more compact than both NN and NNAGM.

To summarize, through the use of SNEA that designs the structure of NNAGM via symbiotic evolution, SNEA-NNAGM can obtain an appropriate NN structure and outperform NN and NNAGM with a smaller prediction error.

V. CONCLUSION

This paper proposes SNEA to construct automatically NN of an NNAGM. The SNEA determines not only the number of hidden neurons, but also the connected topology between input and hidden layers. The idea behind SNEA is to develop

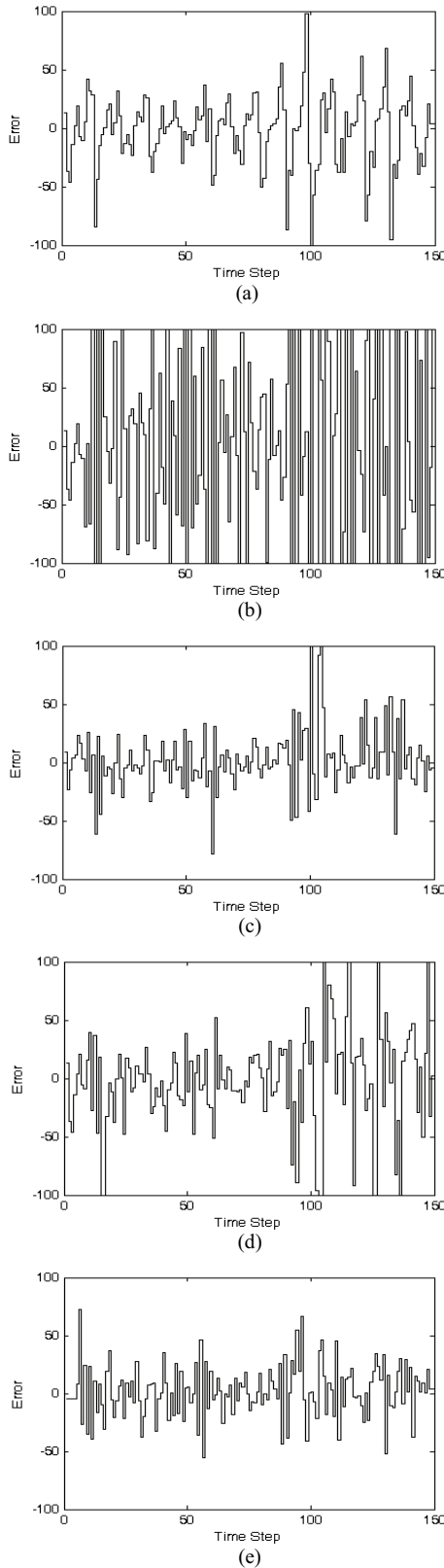


Fig. 9. Prediction errors of (a) GM(1,1) [9], (b) Advanced GM(1,1) [28], (c) NNAGM [18], (d) NN [7], and (e) SNEA-NNAGM.

neurons through symbiotic evolution. This approach facilitates

structure learning of NNAGM by evolving neuron population. Each neuron can participate with NNs of different groups and is evaluated through sharing the fitness of the participating NNs. Through symbiotic evolution, NN with appropriate neurons is constructed and the corresponding topology is concurrently established. The experiments in this paper were conducted to evaluate how well SNEA-NNAGM performed in prediction problems. The SNEA can evolve a partially connected NN of an NNAGM with less neurons and connections than a fully connected NN and NNAGM. Furthermore, the prediction performance of SNEA-NNAGM is better than other methods and demonstrates feasibility. Further work should be done to evaluate the performance of this methodology operating in different problems. Thus, future works attempt to analyze the characteristics of this method regarding to the effect of the parameters and apply this method to different numerical experiment and real-world problem.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their constructive advices and suggestions. This work was supported in part by a grant provided by National Science Council, Taiwan, R.O.C. (NSC 99-2221-E-009 -107) and in part by a grant provided by Industrial Technology Research Institute (Development of Embedded Vision and Intelligent Interactive Behaviors for Home Robots, FY100).

REFERENCES

- [1] A. S. Weigend and N. A. Gershenfeld, Time series prediction: Forecasting the future and understanding the past. Addison Wesley, 1994.
- [2] J. B. Franch, D. C. Bayarri, "Recurrence plots in nonlinear time series analysis: free software," Journal of Statistical Software, vol. 7, no. 9, 2002.
- [3] F. Takens, "Detecting strange attractors in turbulence," Dynamical Systems and Turbulence, vol. 898, pp. 366 1981.
- [4] D. Ruta and B. Gabrys, "Neural Network Ensembles for Time Series Prediction," Proc. Int'l Joint Conf. Neural Networks (IJCNN '07), pp. 1204-1209, 2007.
- [5] C. C. Hsu and C. Y. Chen, "Applications of improved grey prediction model for power demand forecasting," Energy Conversion and Management, vol. 44, pp. 2241-2249, 2003.
- [6] W. L. Yao, S. C. Chi, and J. H. Chen, "An improved grey-based approach for electricity demand forecasting," Electric Power Systems Research, vol. 67, no. 3, pp. 217-224, Dec. 2003.
- [7] Adya, M., Collopy, F., "How effective are neural networks at forecasting and prediction? A review and evaluation," Journal of Forecasting, vol. 17, pp. 481-495, 1998.
- [8] C. M. Zealand, D. H. Burn, and S. P. Simonovic, "Short term streamflow forecasting using artificial neural networks," Journal of Hydrology, vol. 214, pp. 32-48, 1999.
- [9] J. L. Deng, "Control problems of grey systems," System & Control Letters, vol. 1, pp. 288-294, 1982.
- [10] J. L. Deng, "Introduction of grey system theory," The J. of Grey System, vol. 1, no. 1, pp. 1-24, 1989.
- [11] Kayacan, et. al. Grey system theory-based models in time series prediction. Expert Systems with Applications, vol. 37, pp. 1784-1789, 2010.
- [12] V. Viswanathan, V. Krishnan, and L. H. Tsoukalas, "Novel AI approaches in power systems," Int. Conf. Information Intelligence and Systems, pp. 275-280, 1999.

- [13] S. Crone, "Stepwise selection of artificial neural network models for time series prediction," *Journal of Intelligent Systems*, vol. 14, no. 2-3, pp 99-122, 2005.
- [14] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematical Control Signals Systems*, vol. 2, pp. 303-314, 1989.
- [15] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359-366, 1989.
- [16] G. Zhang, B. E. Patuwo, M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *International Journal of Forecasting*, vol. 14, no. 1, pp. 35-62, 1998.
- [17] M. M. Islam, M. A. Sattar, M. F. Amin, X. Yao, and K. Murase, "A new adaptive merging and growing algorithm for designing artificial neural networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 3, pp. 705-722, 2009.
- [18] S.-H. Yang and Y.-P. Chen, "Intelligent Forecasting System Based on Grey Model and Neural Network," *IEEE/ASME International Conf. on Advanced Intelligent Mechatronics*, pp. 699-704, 2009.
- [19] S. H. Yang and Y. P. Chen, "Intelligent Forecasting System Using Grey Model Combined with Neural Network," *International Journal of Fuzzy Systems*, vol. 13, no. 1, pp. 8-15, 2011.
- [20] S.-T. Chen, D. C. Yu, and A. R. Moghaddamjo, "Weather sensitive short-term load forecasting using nonfully connected artificial neural network," *IEEE Trans. Power Systems*, vol. 7, no. 3, pp.1098-1105, 1992.
- [21] R.E. Smith, S. Forrest and A.S. Perelson, "Searching for diverse, cooperative populations with genetic algorithms," *Evol. Comput.*, vol. 1, no. 2, pp. 127-149, 1993.
- [22] D. E. Moriarty and R. Miikkulainen, "Efficient reinforcement learning through symbiotic evolution," *Mach. Learn.*, vol. 22, pp. 11-32, 1996.
- [23] C. F. Juang, J. Y. Lin, and C. T. Lin, "Genetic reinforcement learning through symbiotic evolution for fuzzy controller design," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 2, pp. 290-302, 2000.
- [24] C. J. Lin and Y. J. Xu, "A self-adaptive neural fuzzy network with groupbased symbiotic evolution and its prediction applications," *Fuzzy Sets Syst.*, vol. 157, no. 8, pp. 1036-1056, 2006.
- [25] C. J. Lin, C. H. Chen, and C. T. Lin, "Efficient self-evolving evolutionary learning for neurofuzzy inference systems," *IEEE Trans. Fuzzy Syst.*, vol. 16, no. 6, pp. 1476-1490, 2008.
- [26] Y.-C. Hsu and S.-F. Lin, "Reinforcement group cooperation-based symbiotic evolution for recurrent wavelet-based neuro-fuzzy systems," *Neurocomputing*, vol. 72, no. 10-12, pp. 2418-2432, 2009.
- [27] Y.-C. Hsu, S.-F. Lin, and Y.-C. Cheng, "Multi groups cooperation based symbiotic evolution for TSK-type neuro-fuzzy systems design," *Expert Systems with Applications*, vol. 37, no. 7, pp. 5320-5330, 2010.
- [28] H. C. Ting, J. L. Chang, C. H. Yeh et al., "Discrete time sliding-mode control design with grey predictor," *International Journal of Fuzzy Systems*, vol. 3, pp. 179-185, 2007.