Robust Neural NARX Controller Design using Evolution and Learning

P S Bhati
Department of InstrumentationEngineering,
Govt. Polytechnic College
Ajmer

Rajeev Gupta
Department of Electronic Instrumentation and Control Engineering.
University College of Engineering,
Rajasthan Technical University, Kota

ABSTRACT
A robust neural NARX controller design using evolution and learning is proposed in this paper. Parameters of neural NARX controller are encoded in individual member (chromosome) of GA population. Neural NARX controller learns stabilizing response at an operating point of plant as best fitness reaches large steady value during successive generations. Stable operating region of neural NARX controller is enlarged by incrementally allowing learn more operating points. Best neural NARX controller in last generation is saved as designed neural NARX controller. Proposed approach is validated on a cart pole nonlinear plant model. Optimal stabilizing response with designed neural NARX controller over wide range of operating points is obtained.

Keywords
Non-linear auto regressive with exogenous input (NARX), Reinforcement (RF) signal, Performance index (PI) Genetic algorithms (GAs).

1. INTRODUCTION
Most of industrial process control applications require a robust controller for satisfactory performance over wide operating conditions of the plant. The application domain includes magnetic levitation of objects, power system control, process control etc. Previous affords for design of robust controller are based on neural network [1]-[3], H infinity robust control [4], sliding mode controller [5], Adaptive Auto centering Control [6] nonlinear control [7, 8], fuzzy controller [9]-[10] and hybrid methods using GA [11-14]. Since for most plants mathematical model may not be known precisely during design process, slight deviation of the plant parameters or variation of operating point may result degradation of its stabilizing performance. Fuzzy logic based method for design of controller enables to include nonlinearity and uncertainty of plant model. Nevertheless neural network based design of controller incorporate learning component. Controller design based on neuro-fuzzy structure [15] includes both advantages. Controller design based on learning algorithms has been earlier reported [16-20]. A robust neural NARX controller design with evolution and learning is proposed in this paper. Neural NARX controller learns stabilizing behavior of plant at an operating condition as best fitness of GA population reaches large steady value during successive generations. More operating conditions of plant are allowed learn incrementally to enlarge stable operating region of NARX controller. A moderate order NARX learns control strategy to stabilize response at wide range of operating points. Only output state of plant needs to be measured in proposed controller.

2. NEURAL NARX CONTROLLER DESIGN
An artificial neural network has ability to learn arbitrary complex nonlinear input-output mapping through learning. Many neural network structures like multilayer feed forward, recurrent network are proposed as controller. A neural nonlinear autoregressive with exogenous input (NARX) controller design is proposed in this paper. NARX structure has ability to learn a dynamic input-output mapping. Knowledge about input-output mapping to be performed is stored in the form of link weights between layers of neurons. Parameters of neural NARX controller are designed using evolution and learning.

Evolutionary algorithms (EA), like Genetic algorithms (GA), Evolutionary programming (EP), Particle swarm optimization (PSO), and Ant colony optimization (ACO) are used in many optimization and machine learning tasks [36]-[41]. Generally EA work with population of members (individuals). Every EA encompasses mechanism of fitness assignment to each member, selection of better fit members and genotype variation to produce off-springs. Specific features of EA are inbuilt parallelism, ability to work with coding of parameters of real world optimization problem. It uses only fitness information of population members, can work with discontinuous fitness (objective) function and able to search global optimal solution in large search space. It starts with randomly initialized population of its members. Among EA, Genetic algorithms (GA) are most popular as it mimics the operations involved in natural genetics. Genetic algorithms are based on Darwin's theory of "Survival of fittest". Members of population containing better fitness to the environment have more probability of being copied to next generation. GA evolves new population members (off-springs) for next generation using its operators i.e. reproduction, crossover and mutation. GA searches the solutions for improved fitness value during successive generations [11].

Parameters (link weights) of neural NARX controller are encoded in a chromosome of genetic population. Output variable of NARX controller is a control signal. GA population of NARX controller learns stabilizing behavior of plant at an operating point as best fitness becomes large

steady value during successive generations. The stable operating region of neural NARX controller is enlarged by incrementally learning more operating points of the plant. Genetic evolution process propagates knowledge learned during successive generations about the plant behavior up to last generation. Best NARX controller in the last generation is saved as designed NARX controller. A quantitative analysis of incremental learning based on schema processing is given below.

2.1 Schema-Processing Theorem and Genetic Learning

A chromosome (string encoded with parameters of NARX controller) represents a point in the solution space that is mapped to an operating region in the operating space. A schema [11] is a similarity template of member chromosomes in GA population that represents group of points in solution space. Alternatively a schema represents group of operating regions in the operating space.

Schema-processing theorem is given as

$$m(H,n_0+1) \geq m(H,n_0).\frac{f(H)}{\bar{f}}.[1-p_c.\frac{\delta(H)}{(l-1)}-O(H)p_m] \quad \text{……..(1)}$$

Where,

m(H,n₀) is number of chromosomes representing schema *H,* in n_0^{th} generation.

f(H) is the average fitness of chromosomes representing schema *H*.

$\bar{f}$ is the average fitness of population.

P_c is the probability of crossover.

P_m is the probability of mutation.

l is the length of chromosome.

$\delta(H)$ is the defining length of schema.

$O(H)$ is the order of schema.

For short defining length and low order fit schema equation (1) reduces to

$$m(H,n_0+1) \geq K.m(H,n_0) \quad \text{……(2)}$$

$$K = \frac{f(H)}{\bar{f}}$$

Where, $\bar{f}$ is ratio of average fitness of chromosomes representing schema *H* to average fitness of population. Let *K* be nearly same for all generations. After n generations we have

$$m(H,n_0+n) \geq K^n.m(H,n_0) \quad \text{……..(3)}$$

It implies, short defining length, low order and fit schema ($K>1$) receives exponentially increasing number of chromosomes in successive generations. Alternatively, GA searches solutions for improved fitness value of population members during successive generations. Low fit schema ($K<1$) receives exponentially decreasing number of chromosomes in successive generations. Let n_1, n_2, n_3 …. n_r are number of generations allowed for learning of 1st, 2nd, 3rd…rth operating points respectively, then incrementally learning these operating points results

$$m(H,n_0+n_1+n_2....n_r) \geq K_1^{n_1}.K_2^{n_2}....K_r^{n_r}m(H,n_0). \quad \text{…..(4)}$$

An explanation of equation 4 follows. Consider two classes of chromosomes in a population having good fitness at initial operating point (K_1 >1), one covering narrow operating region and other covering broad operating region. When next operating point that may lie far away from current operating point is allowed learn, chromosomes covering narrow operating region will have low fitness value at new operating point (K_2 <1). Chromosomes covering broad operating region will have high fitness value at new operating point (K_2 >1). As more new operating points are successively allowed learn, chromosomes covering broad operating region would still have high fitness value (K_i > 1 for i=3, …r). Chromosomes covering narrow operating region would have low fitness value after switching to new operating points (K_i < 1 for i=3, …r) and would be successively discarded. Thus genetic learning would only propagate chromosomes fit at more operating points covering enlarged operating region to successive generations. Accordingly knowledge learned previously regarding operating points are retained during successive generations (not forgetting the acquired knowledge) as new operating points are learned. Consider a test operating point not seen during learning phase. It is more likely that test operating point lie within the operating region covered by genetically learned NARX controller. Genetically learned NARX controller would show robustness at other operating points.

3. ELEMENTS OF GENETIC LEARNING

Elements of genetic learning are discussed in the following subsections.

3.1 Neural NARX Controller Structure

A neural NARX is a recurrent network having feedback from output neurons with time delays as shown in fig. 1.0. An exogenous input signal with time delays is applied. One of input signal is a bias signal. NARX structure has ability to learn a dynamic input-output mapping. A neural NARX network of third order with seven input neurons, three hidden neurons and one output neuron is shown in figure. First layer consisting of input neurons accepts input signals and pass them to second layer through link weights. Second (hidden) layer has neurons with a nonlinear activation function (tanh(x)). Output of Hidden layer is given as

$$y_i = \tanh(\sum_{j=1}^{j=n} W_{ij}x_j) \quad \text{…….(5)}$$

Where, $\boldsymbol{W}$ is link weight matrix between inputs and hidden neurons. For n inputs and m neurons in hidden layer $\boldsymbol{W}$ is of [m×n] size and x_j is j^{th} input signal.

Signals after nonlinear processing from hidden neurons are passed to third layer of output neuron through another set of link weights. Output of output layer is given as

$$y_i' = \tanh(\sum_{j=1}^{j=m} W_{ij}'x_j) \quad \text{…….(6)}$$

Where, $\boldsymbol{W'}$ is link weight matrix between hidden neurons and output neurons. For m neurons in hidden layer and k neurons in output layer $\boldsymbol{W'}$ is of [k×m] size and x_j is j^{th} output form hidden neuron.

Knowledge about input-output mapping to be performed is stored in the form of link weights. Output neurons give the output at the next sampling instant that is used as control signal for plant to be controlled. NARX model is represented with the following equation.

u(n+1)=F(u(n), u(n-1), u(n-2), ….u(n-q+1), e(n), e(n-1), e(n-2), ….e(n-q+1)) ……..(7)

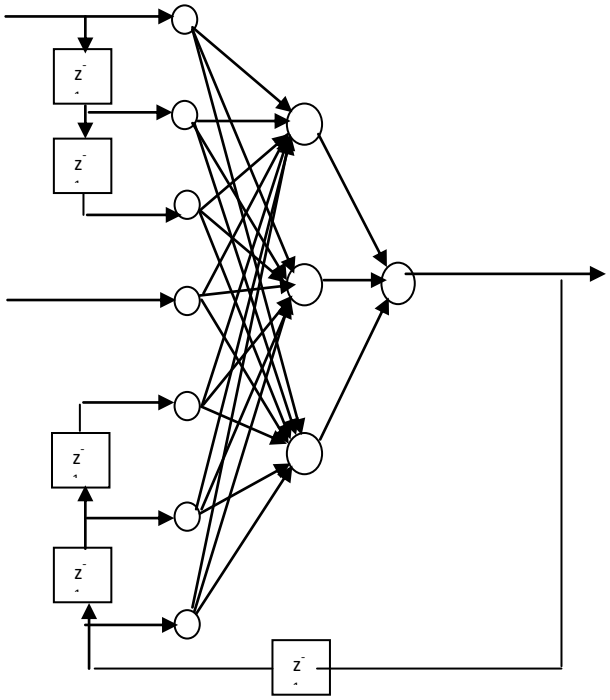

Fig. 1: neural NARX network with three hidden neurons

Where,

- ***u(n+1)*** is output of NARX controller at ***(n+1)th*** sampling instant.
- ***e(n)*** is exogenous input signal at nth sampling instant.
- ***q*** is order of NARX model.
- ***F*** is nonlinear activation function of neurons.

3.2 NARX Controller Encoding

Parameters of NARX controller are encoded in a chromosome of GA population as real values as shown in fig 2. W[m×k] and W′[1×m] are weight matrices of hidden layer and output layer for k number of inputs and m number of neurons in hidden layer. W_{ij} and W'_{ij} are elements of respective weight matrices. Each chromosome encoded represents a NARX controller.

W_{11}	W_{12}	…	W_{mk}	W'_{11}	W'_{12}	….	W'_{1m}

Fig 2: A chromosomes encoded with NARX controller parameters

3.3 Genetic Based Learning Procedure

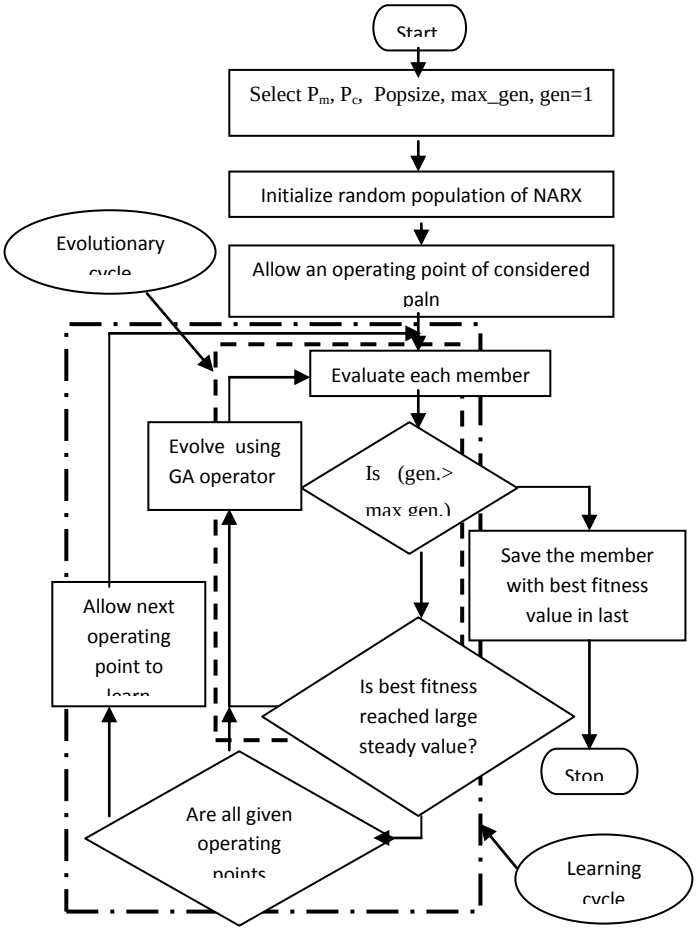

Fig 3: Flow chart of NARX controller learning with GA

Procedure for GA learning of NARX controller is shown in fig 3. GA parameters like crossover probability P_c, mutation probability P_m, population size and maximum generations are initialized. A generation counter is also initialized. Initially a random population of NARX controller is generated. Real number coded chromosome string is used instead of binary coding to achieve the computational efficiency in the proposed method. Real values in chromosome string represent the parameters of NARX controller. Fitness of each NARX controller is evaluated for a definite interval of time at an operating point of plant. Better fitness members are selected for reproduction. Selected member through crossover and mutation produces off-springs for next generation. GA operators evolve new population and generation counter is incremented. GA population of NARX controller learns stabilizing behavior of plant at an operating point with optimum performance as best fitness becomes large steady value during successive generations completing one evolutionary cycle. Incrementally GA population of NARX controller is allowed learn more operating points. GA is computed till the number of generations exceeds maximum number of allowed generations.

A learning cycle is completed when all operating point of plant are learned. NARX controller having best fitness value in last generation is saved as designed NARX controller.

3.4 Fitness Function Evolution

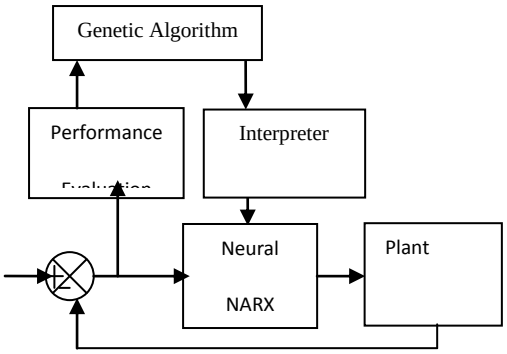

Fig.4: Fitness evaluation for neural NARX controller

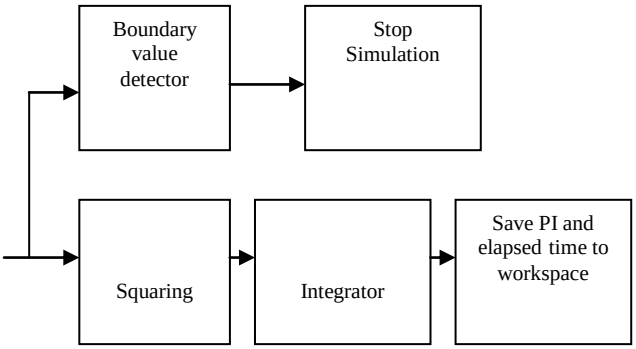

Fig. 5: Performance Index calculation

Each neural NARX controller represented in a chromosome string is to be evaluated for progress of GA. Fig. 4 shows scheme to calculate fitness value. An interpreter decodes a chromosome into a working neural NARX controller. Each neural NARX controller in GA population is evaluated by applying its output as control signal to plant for a trial. A trial completes if the number of time steps exceeds predefined value i.e. TRIAL_TIME or a failure of stabilizing control results. Control is considered unsuccessful if error between reference value and plant output exceeds certain bound i.e. MAX_ERROR. During a trial a performance evaluation block calculates performance index (PI) by resulting output response of the plant and also records elapsed time. Lastly fitness value is obtained with the defined fitness function. Only output state of the plant needs to be measured. Error in reference and plant output signal is used as exogenous input signal to neural NARX controller.

Procedure for GA learning of NARX controller is shown in fig 3. GA parameters like crossover probability P_c, mutation probability P_m, population size and maximum generations are initialized. A generation counter is also initialized. Initially a random population of NARX controller is generated. Real number coded chromosome string is used instead of binary coding to achieve the computational efficiency in the proposed method. Real values in chromosome string represent the parameters of NARX controller. Fitness of each NARX controller is evaluated for a definite interval of time at an operating point of plant. Better fitness members are selected for reproduction. Selected member through crossover and mutation produces off-springs for next generation. GA operators evolve new population and generation counter is incremented. GA population of NARX controller learns stabilizing behavior of plant at an operating point with optimum performance as best fitness becomes large steady value during successive generations completing one evolutionary cycle. Incrementally GA population of NARX controller is allowed learn more operating points. GA is computed till the number of generations exceeds maximum number of allowed generations.

To stabilize response of controller a performance index is defined as

$$PI = \int_0^{RF} |\varepsilon_p|^2 \, dt \qquad \ldots\ldots\ldots (8)$$

Where, ε_p is error between reference signal and plant output signal and ***RF*** is a reinforcement signal used to enhance better actions of NARX controller for stabilizing control for longer duration. Reinforcement signal is defined as follows

RF = TRIAL_TIME, if stable control results.

RF = Elapsed time in a trial, if unstable control results (error > MAX_ERROR). …… (09)

To maximize RF signal and minimize PI the fitness function is defined as

$$Fitness = \frac{RF}{PI} \qquad \ldots\ldots\ldots (10)$$

Fitness of neural NARX controller is maximized during successive generations. Fig. 5 shows the procedure to calculate the PI and RF values. Error signal is squared and integrated continuously with respect to elapsed time. Simultaneously the error signal is passed through a boundary value (MAX_ERROR) detector. If its value exceeds the boundary value then simulation is stopped and last value obtained by output of integrator block and elapsed time (RF signal) is saved to workspace. Otherwise these values are saved at the end of the TRIAL_TIME. Fitness is calculated by equation (10).

4. CASE STUDY

Proposed approach for robust design of controller is validated on a bench mark control problem.

4.1 Cart Pole System

A standard cart pole system is inherently unstable benchmark control problem. The aim is to find out sequence of force applied to cart so as to maintain pole position vertically up over different operating conditions. A nonlinear model of cart pole system is considered. States of plant are pole angle deviation from vertical, angular velocity, cart displacement from mean position, and linear velocity represented in a vector as [$\theta, \dot{\theta}, x, \dot{x}$]. Output of neural NARX controller is horizontal force *f* applied to cart in either direction to balance pole vertically up. Range of force are [-20N, 20N]. Universe of discourse for other states are θ [-1,1]rad, $\dot{\theta}$ [-1,1]rad/sec, *x* [(-5,5]m, $\dot{x}$ [-1,1] m/sec. Control is considered unsuccessful if value of θ exceeds interval [-1, 1]rad or the cart position exceeds beyond interval [-5, 5]m from center. For designing neural NARX controller over dynamic plant environment, bounds on pole position and cart position from the mean position is considered wide enough. Balancing of pole depends on states of the plant and force applied. Model of considered cart pole system is represented as follows:

$$\theta(t+1) = \theta(t) + \Delta \cdot \dot{\theta}(t) \quad \text{…….. (11)}$$

$$\dot{\theta}(t+1) = \dot{\theta}(t) + \Delta \cdot \frac{mgSin\theta(t) - Cos\theta(t)(f + m_p l \dot{\theta}(t)^2 Sin\theta(t))}{(4ml/3) - m_p l Cos^2\theta(t)} \quad \text{……. (12)}$$

$$x(t+1) = x(t) + \Delta \cdot \dot{x}(t) \quad \text{……. (13)}$$

$$\dot{x}(t+1) = \dot{x}(t) + \Delta \cdot \frac{f + m_p l(\dot{\theta}(t)^2 Sin\theta(t) + \dot{\theta}(t) Cos\theta(t))}{m} \quad \text{…….. (14)}$$

Where,

$\theta(t)$: Angular deviation from vertical axis.

$\dot{\theta}(t)$: Angular velocity of the pole.

$x(t)$: Position of cart from center.

$\dot{x}(t)$: Linear velocity of the cart.

f : Force applied to the cart

m_p : Mass of the pole.

m : Combined mass of the pole and cart.

l : Length of the pole.

Δ : Sampling period(0.001sec).

GA parameters used in present study are population size of 20, crossover probability of 0.6, mutation probability of 0.1, with real number encoding of neural NARX controller weights. TRIAL_TIME equal to 10sec, MAX_ERROR in angular position equal to ±1rad, output of neural NARX controller is control force *f* bounded to ±20N are considered. Neural NARX controller structure as shown in figure 1 is used. Error in angular position is used as an exogenous input signal to NARX controller. A third order neural NARX controller with 3 hidden neurons and one output neuron is used in present study.

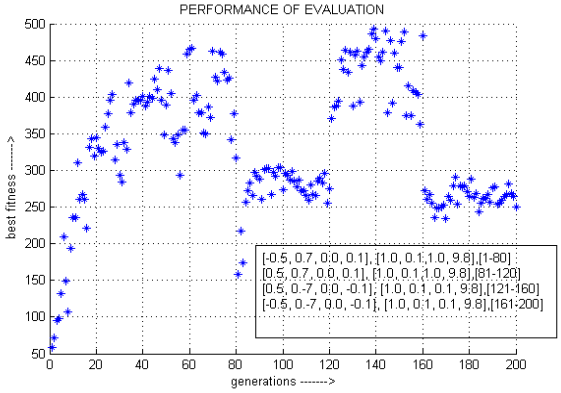

Fig. 6: Learning progress of neural NARX controller with GA

Fig. 6 shows learning progress of GA evaluation for 200 generations. First vector in legend entry represents initial conditions of state vector [$\theta, \dot{\theta}, x, \dot{x}$]. Second vector represents operating conditions (plant parameters) [*Pole length (l), Pole mass (m), Cart mass (M), Acceleration due to gravity (g)*]. Third vector represent generation interval [start - last] for an operating condition. Best fitness of GA population improves and reaches large steady value during 80 generations for first allowed operating conditions (operating point) of plant. Same operating point is allowed learn for other initial conditions during successive generations [81-120]. Best fitness change and again reaches steady large value. An evolutionary cycle is completed when current operating point is learned (best fitness reaches steady large value). Plant operating conditions are changed and next evolutionary cycle is started form generation number 121. Best fitness of GA population improves and reaches large steady value up to 200 generations for second allowed operating condition of plant. Incrementally more operating points of the plant are allowed in successive evolutionary cycles up to last generation. A learning cycle is completed when all considered operating point are learned. Best NARX controller in last generation is saved as designed NARX controller.

Fig. 7 to 10 shows position response of NARX controller for different initial conditions of the plant at operating points for which it was allowed learn. Fig. 7 shows position response for sinusoidal input reference signal. Stabilizing position response with optimum performance at various initial conditions shows that designed neural NARX controller has learned operating environments for which it was allowed learn.

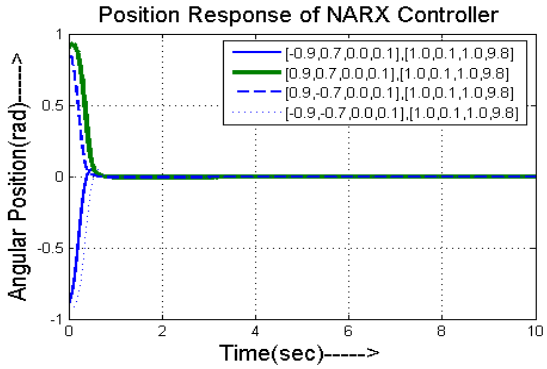

Fig. 7: Response of NARX controller for different initial conditions

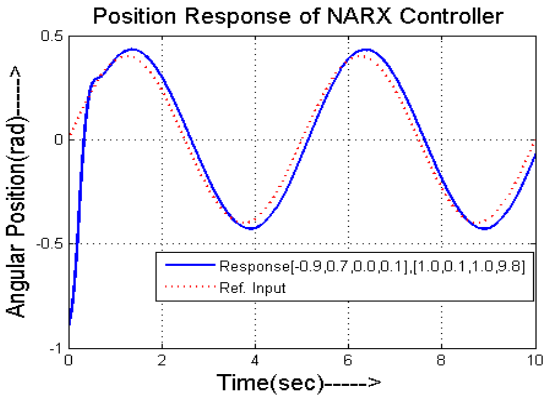

Fig. 8: Response of NARX controller for sinusoidal input signal

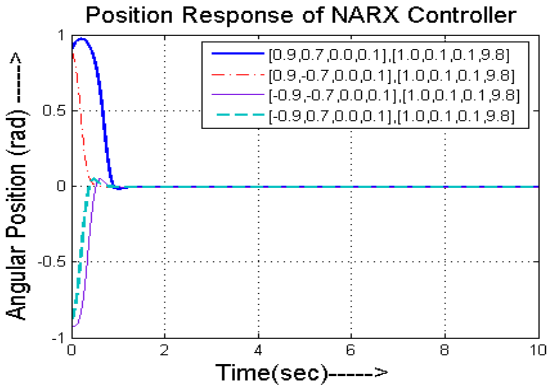

Fig. 9: Response of NARX controller for different initial conditions

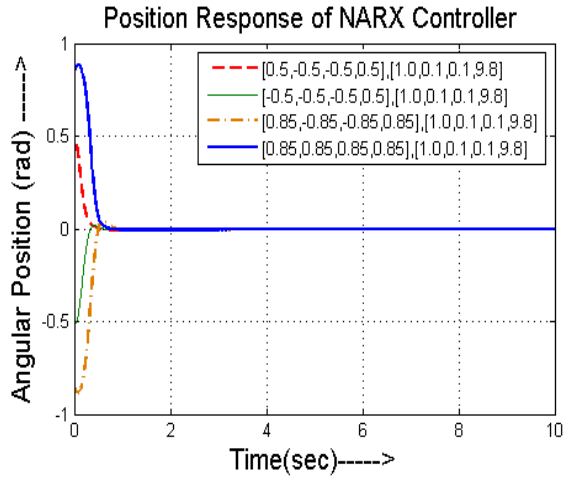

Fig. 10: Response of NARX controller for different initial conditions

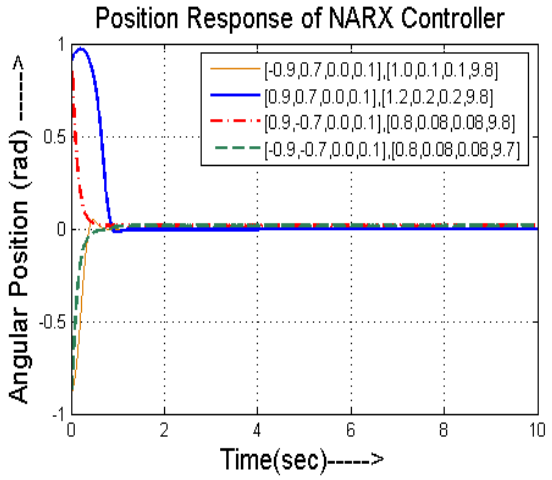

Fig. 11: Response of NARX controller for different initial conditions and operating points

Fig. 11 shows position response of NARX controller for different initial conditions of the plant at operating points for which it was not allowed learn. Stabilizing position response with optimum performance at various initial conditions and at different operating conditions shows that designed neural NARX controller has extended the stability region for wide range of operating conditions for which it was not allowed learn showing robust stabilizing performance.

5. CONCLUSIONS

A robust neural NARX controller design using evolution and learning is proposed. Neural NARX controller contributes stabilizing response at wide range of operating points with optimal performance. Proposed neural NARX controller efficiently extends stability region over wide range of operating conditions.

6. REFERENCES

[1] A. G. Barto, R. S. Sutton, C.W. Anderson, "Neuron like adaptive elements that can solve difficult learning control problems," IEEE Trans. Syst. Man, Cybern., vol. SMC-13, pp. 834-846, 1983.

[2] Takashi Hiyama and DEng, "Application of rule-based stabilizing controller to electrical power system", IEE Proc., part C, vol. 136, No.3, pp. 175-181, May. 1889.

[3] J. S Jang, "ANFIS: Adaptive-network-based fuzzy inference system" IEEE Trans. Syst. Man, Cybern., vol. 23, pp. 665-685, May 1993.

[4] Fumio Matsumura, Tom Namerikawa, Kazuhiko Hagiwara, and Masayuki Fujita, "Application of Gain Scheduled $H\infty$ Robust Controllers to a Magnetic Bearing," IEEE Trans. Control Syst. Technol., vol. 4, no. 5, pp. 484–493, Sept. 1996.

[5] Sannella, M. J. 1994 Constraint Satisfaction and Debugging for Interactive User Interfaces. Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.

[6] Kai-Yew Lum, Vincent T. Coppola, and Dennis S. Bernstein, "Adaptive Autocentering Control for an .Active Magnetic Bearing Supporting a Rotor with Unknown Mass Imbalance," IEEE Trans. Control Syst. Technol., vol. 4, no. 5, pp. 587–592, Sept. 1996.

[7] Marcio S. de Queiroz and Darren M. Dawson, "Nonlinear Control of Active Magnetic Bearings : A Backstepping Approach," IEEE Trans. Control Syst. Technol., vol. 4, no. 5, pp. 545–552, Sept. 1996.

[8] Ali Charara, Jerome De Miras, and Bernard Caron, "Nonlinear Control of a Magnetic Levitation System Without Premagnetization," IEEE Trans. Control Syst. Technol., vol. 4, no. 5, pp. 513–523, Sept. 1996.

[9] Hong-Chan Chang and Mang-Hui Wang, "Neural network based self-organizing fuzzy controller for transient stability of multimachine power system", IEEE Trans. on energy conversion, vol. 10, No.2, pp. 339-347, June 1995.

[10] Takashi Hiyama, Yoshiteru Ueki and Hiroaki Andou, "Integrated fuzzy logic generator controller for stability enhancement", IEEE Trans. on energy conversion, vol. 12, No.4, pp. 400-405, Dec. 1997.

[11] D. E. Goldberg, "Genetic Algorithms in search optimization and Machine learning", Addition Wesley 1989.

[12] A. Homaifar and E. McCormick, "Simultaneous design of membership functions and rule set for fuzzy controllers using genetic algorithms," IEEE Trans. Fuzzy Systems, vol. 3, pp. 129-139, April 1995.

[13] M. H. Lim, S Rahardja, and B. H. Gwee, "A GA paradigm for learning fuzzy rules," Fuzzy Sets and Systems, vol. 82, pp. 177-186, 1996.

[14] M. A. Lee and H. Takaji, "Integrating design stages of fuzzy systems using genetic algorithms," in Proc. IEEE int conf. Fuzzy Systems, pp. 612-617, New York, 1993.

[15] C. T. Lin, C. F. Juang, and C. P. Li, "Temperature control with a neural fuzzy inference network," IEEE Trans. Syst. Man, Cybern., vol. 29, pp. 440-459, 1999.

[16] H. R. Berenji and P. Khedkar, "Learning and tuning fuzzy logic controllers through reinforcement" IEEE Trans. Neural Networks, vol.3, pp.724-740, May1992.

[17] C. T. Lin and C. S. G. Lee, "Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems," IEEE Trans. Fuzzy Syst., vol. 2,pp. 46-63, Feb.1994.

[18] C. T. Lin and C. P. Jou, "GA- Based Fuzzy Reinforcement Learning for Control of a Magnetic Bearing System",. IEEE Trans. Systems, Man, and Cybernetics- Part B: Cybernetics, vol. 30, No. 2, pp. 276-289, April 2000.

[19] C. F. Juang, J. Y. Lin, C. T. Lin, "Genetic Reinforcement Learning Through Symbiotic Evolution for Fuzzy Controller Design", IEEE Trans. Systems, Man, and Cybernetics- Part B: Cybernetics, vol. 30, No. 2, pp. 290-301, April 2000.

[20] S. F. Su, and S. H. Hsieh, "Embedding Fuzzy Mechanisms and Knowledge in Box-Type Reinforcement Learning Controllers", IEEE Trans. Systems, Man, and Cybernetics- Part B: Cybernetics, vol. 32, No. 5, pp. 645-653, Oct. 2002.

[21] W. M. V. Buijtenen, G. Schram, R. Babuska, and H. B. Verbruggen, "Adaptive Fuzzy Control of Satellite Attitude by Reinforcement Learning", IEEE Trans. Fuzzy Systems, vol. 6, No. 2, pp. 185-194, 1998.

[22] P. Tsiotras and B. C. Wilson, "Zero and low-bias control designs for active magnetic bearings," IEEE Trans. Control Syst. Technol., vol. 11, no. 6, pp. 889–904, Nov. 2003.

[23] P. Tsiotras and M. Arcak, "Low-bias control of AMB subject to voltage saturation: State-feedback and observer designs," IEEE Trans. Control Syst. Technol., vol. 13, no. 2, pp. 262–273, Mar. 2005.

[24] M. S. de Queiroz and S. Pradhananga, "Control of Magnetic Levitation Systems with Reduced Steady-State Power Losses" IEEE Trans. On Control Systems Technology, vol. 15, No. 6, pp. 1096-1102, Nov. 2006.

[25] L. B. Antonio do Bomfim, G. N. Taranato, and D M Falcao, "Simultaneous tuning of power system damping controllers using genetic algorithms", IEEE Trans. on power system, vol. 15, No. 1, pp. 163-169, Feb. 2000.

[26] A. B. Chammas and C. T. Leondes, "Pole assignment by piecewise constant output feedback", Int. Journal on Control, Vol. 29, pp. 31-38, 1979.

[27] H. Werner and K. Furuta, "Simultaneous stabilization based on output measurement", Kybernetika, Vol. 31, pp. 395-411, 1995.

[28] R. Gupta, B. Bandyopadhyay and A. M. Kulkarani, "Design of power system stabilizer for single machine system using robust fast output sampling feedback technique" Elsevier Science, Electric Power System research, pp 247-257, 2003.

[29] R. Gupta, B. Bandyopadhyay and A. M. Kulkarani, "Robust decentralized fast output sampling technique based power system stabilizer for multi-machine power system", International Journal of System Science, Vol. 36., No.5, pp 297-314, April 2005.

[30] V. Bandal and B. Bandyopadhyay, "Robust decentralized output feedback sliding mode control technique-based power system stabilizer for multi-machine power system", IET Control Theory Applications, I(5), pp. 1512-1522, 2007.

[31] M. R. Gonzalez, and O. P. Malik, "Power system stabilizer design using an online adaptive neuro-fuzzy controller with adaptive input link weights", IEEE Trans. on energy conversion, vol. 23, No.3 pp. 914-922, Sept. 2008.

[32] R. Polikar, L. Udpa, S. S. Udpa, and V. Honavar, "Learn ++: An incremental learning algorithm for supervised neural networks," IEEE Trans. Systems, Man, and Cybernetics- Appl. Rev., Part C: vol. 31, No. 4, pp. 497-508, Nov 2001.

[33] K. Yamauchi, N. Yamauchi, and N. Ishii, "Incremental learning methods with retrieving of interfered patterns," IEEE Trans. Neural Networks, vol. 10, no. 6 pp.1351-1365, Nov. 1999.

[34] Jing Yang, Zhong -Wei Le, and Jian- Pei Zhang, "A training algorithm of incremental support vector machine with recombining method," Proc. of 4th int. conf. on Machine Learning and Cybernetics, Gaungzhou, pp. 4285-4288, 18-21 August 2005.

[35] Sheng-Uei Guan, and Fangming Zhu, "An Incremental Approach to Genetic Algorithms Based Classification," IEEE Trans. Systems, Man, and Cybernetics- Part B: Cybernetics, vol. 35, No. 2, pp. 227-239, April 2005.

[36] H. Y. Quek, K. C. Tan, C. K. Goh, H. A. Abbass, "Evolution and incremental learning in iterated prisoner's dilemma", IEEE Trans. on Evolutionary Computation, vol. 13, No. 2, pp. 303-320, April 2009.

[37] X. Chen, Y. S. Ong, M. H. Lim, and K. C. Tan, "A multi-facet survey on memetic computation", IEEE Trans. on Evolutionary Computation, vol. 15, No. 5, pp. 591-607, Oct 2011.

[38] G. Acampora, J. M. Cadenas, V. Loia, and E. M. Ballester, "Achieving memetic adaptability by means of agent-based machine learning", IEEE Trans. on Evolutionary Computation, vol. 07, No. 4, pp. 557-569, Nov. 2011.

[39] S. S. Choi and B. R. moon, "A graph based Lamarckian-Baldwinian hybrid for the sorting network problem", IEEE Trans. on Evolutionary Computation, vol. 09, No. 1, pp. 105-114, Feb. 2005.

[40] Y. S. Ong, and A. J. Keane, "Meta-Lamarckian learning in memetic algorithms", IEEE Trans. on Evolutionary Computation, vol. 08, No. 2, pp. 099-110, April 2004.

[41] R. P. Prado, S. G. Galan, J. E. M. Exposito and A. J. Yuste, "Knowledge acquisition in fuzzy-rule-based systems with particle swarm optimization", IEEE Trans. On fuzzy systems vol. 18, No. 6, pp. 1083-1097, Dec. 2010.

[42] A. Abbadi, L. Nezli, D. Boukhetala, "A nonlinear voltage controller based on interval type 2 fuzzy logic control system for multi-machine power systems", International Journal of Electrical Power & Energy Systems, Volume 45, Issue 1, Pages 456–467, February 2013.

[43] P. S. Bhati, Rajeev Gupta, "Robust fuzzy logic power system stabilizer based on evolution and learning" International Journal of Electrical Power & Energy Systems, Volume 53, Issue 1, Pages 357-366, May 2013.