J. Basic. Appl. Sci. Res., 3(4)893-900, 2013
© 2013, TextRoad Publication

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Nonlinear Model Predictive Control of a Continuous Stirred Tank Heater based on Multiple Neural Networks

***Ali Rabiee, Hooshang Jazayeri-Rad, Mohamad Ali Takassi**

Department of Instrumentation and Automation, Petroleum University of Technology, Ahwaz, Iran

ABSTRACT

In this paper nonlinear model predictive control (NMPC) based on multiple neural networks (MNNs) is considered. The control scheme includes a process model and an optimizer. MNNs are used to model the process and the model is then used in an NMPC scheme to control the process. Ensembles of artificial neural networks show improved generalization and noise-robustness capabilities in comparison to a single neural network (SNN). A genetic algorithm (GA) based optimizer is developed to determine a solution for the control trajectory. This optimizer is then combined with the sequential quadratic programming (SQP) technique. The resulting optimizer is named GASQP. As an application example of the developed algorithm, the real-time control of a continuous stirred tank heater (CSTH) is considered. Application results demonstrate that the developed method improves the control performance of the system.

KEYWORDS: Nonlinear Model Predictive Control, Multiple Neural Networks, Genetic Algorithm, Sequential Quadratic Programming , Continuous Stirred Tank Heater.

1. INTRODUCTION

NMPC has received substantial attention over the past years. NMPC expresses the model predictive control (MPC) arrangement that uses nonlinear models for prediction and permits a non-quadratic cost function and nonlinear constraints on the process variables [1]. MPC has made a considerable impact on control engineering. MPC is one of the most appropriate control techniques which denote a class of control techniques in which a dynamic process model is employed to predict and enhance process performance. Linear MPC which employs a linear model for prediction was effectively used for years in numerous advanced industrial applications. Because many processes are nonlinear and linear models are often insufficient to describe extremely nonlinear processes and reasonably nonlinear processes which have large operating changes, different NMPC methods have been established [2].

The MPC technique optimizes the process outputs over some predetermined future time interval called the prediction horizon N. At the each time step, the future outputs are predicted using a dynamic model of the process. This model is used to compute the current and future control actions (the control horizon) N_u ($N_u \leq N$), which minimize a user-specified performance index. After the N_uth time step, it is presumed that the control action is constant. Only the first of the resulting optimal inputs is applied to the process. This complete process is repeated at each time step [3].

One of the major limitations of the MPC techniques is the application of a linear model to the prediction of future process states. When the technique is applied to nonlinear processes, the model mismatch can degrade the control performance. As a solution to this problem, neural network (NN) based nonlinear models have already been proposed for the MPC.

Neural network ensemble is a learning example where a group of a finite number of neural networks is trained for the same task. Ensembles of artificial neural networks give enhanced generalization abilities that outperform those of SNNs [4]. There are numerous ensemble techniques, but the most prevalent include some elaboration of bagging [5], boosting [6] and stacking [7]. We concentrated on bagging in neural network ensembles. Bagging is anticipated by Breiman [5] founded on bootstrap sampling. It produces several training sets from the initial training set and then trains a component neural network from each of those training sets [8]. The resulting MNN can be used as a prediction model for control purposes.

A second problem is the selection of the optimization algorithm, since the lack of suitable optimization tools hinders the practical utility of NMPC. GA based optimization was employed as a possible optimization method.

GAs are search algorithms founded on mechanism of natural selection giving robust search abilities in complex space, and in this manner offering an effective method to problems necessitating efficient global searches [9]. This work employs GA to implement the optimizer in NMPC. In addition this optimizer is further modified by including

*** Corresponding Author:** Ali Rabiee, Department of Instrumentation and Automation, Petroleum University of Technology, Ahwaz, Iran. E-mail: alirabiee13@gmail.com Tel: +989131094568; Fax: +986115551021

the SQP technique [10] to improve the computation of manipulating variables. Unlike SQP, GA is a global search algorithm. It simultaneously searches for solutions in several regions, thus increasing the probability of finding the global optimum. Therefore, combining GA as a global search technique with SQP as a local search algorithm would increase the accuracy and speed of convergence. The optimizer executes GA until an approximate suboptimal solution is obtained. The outcome is further improved using the SQP technique.

The outline of this article is organized as follows. Section 2 briefly reviews NMPC. Process modeling is explained in Section 3. In Section 4, the optimization of the nonlinear control problem by the GA and SQP is explained. The CSTH case study is described in Section 5. Results and discussions are given in Section 6. Finally, Section 7 presents the conclusions of this work.

2. Nonlinear Model Predictive Control

A basic structure of NMPC is shown in Fig. 1. The components of the plan consist of a process, a model and a control block. The control block includes an optimizer, objective function and constraints.

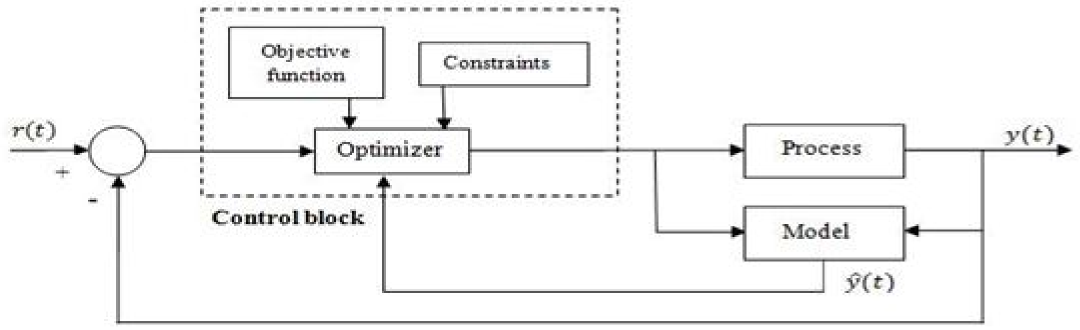

Fig. 1. Nonlinear model predictive control plan

Neural networks are used for experimental modeling of the process. The model will predict the process characteristics over a prediction horizon N, enabling the controller to incorporate future set point changes or disturbances.

The manipulating input applied to the process in NMPC is computed by the recurring solution of a finite horizon open loop optimal control problem subject to the system dynamics, input and state constraints. Based on measurements acquired at time t, the controller foresees the future dynamic behavior of the system over a prediction horizon N and computes (over a control horizon $N_u \leq N$) the inputs such that a scheduled open-loop performance objective functional is optimized. If there were no disturbances and no model discrepancy, and if the optimization problem could be solved for infinite horizons, then one could apply the input profile determined time $t = 0$ to the system for all times $t \geq 0$. However, this is not generally probable. Due to disturbances and model mismatch, the actual system performance is not the same as the predicted performance. To solve this problem feedback is considered in the system. In order to include some feedback mechanism, the open-loop manipulated input profile obtained will be applied to the plant only until the next measurement becomes obtainable. The time difference between the measurements can fluctuate, however often it is presumed to be fixed, i.e. the measurement will occur every δ sampling time units. Using the new measurement at time $t + \delta$, the entire procedure (prediction and optimization) is repeated to determine a new input profile with the control and prediction horizons moving ahead [11]. The overall principle of MPC is demonstrated in Fig 2.

Usually, the objective function, given below by Eqn. 1, includes two parts. The first part is known as the cost of predicted control errors, the differences between the set point y^{ref} and the predicted values of the output $\hat{y}$ over the prediction horizon N. The second part represents the penalties for the changes of the control value [12].

$$J(k) = \sum_{p=1}^{N} A(y^{ref}(k+p|k) - \hat{y}(k+p|k))^2 + \sum_{p=0}^{N_u-1} B(\Delta u(k+p|k))^2 \tag{1}$$

From the following optimization problem:

$$\min_{\Delta u(k|k),\dots,\Delta u(k+N_u-1|k)} \{J(k)\} \tag{2}$$

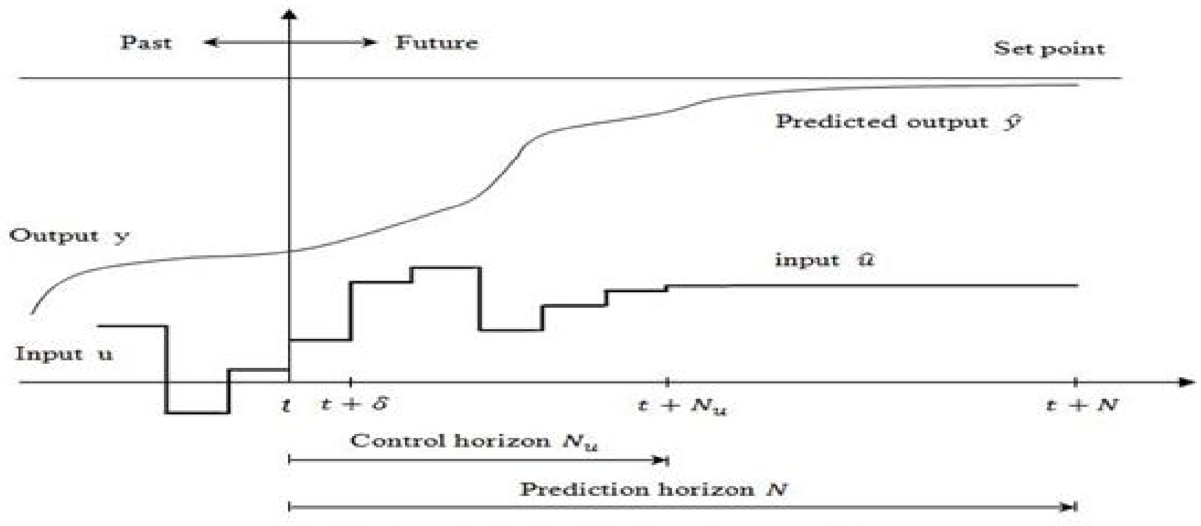

Fig. 2. Principle of model predictive control

Subject to:

$$u_{min} \le u(k+p|k) \le u_{max} \quad , \quad p = 0,\dots,N_u - 1 \tag{3}$$

$$\Delta u_{min} \le \Delta u(k+p|k) \le \Delta u_{max} \quad , \quad p = 0,\dots,N_u - 1 \tag{4}$$

$$y_{min} \le \hat{y}(k+p|k) \le y_{max} \quad , \quad p = 0,\dots,N \tag{5}$$

The future control increments, $\Delta u(k)$, are determined as:

$$\Delta u(k) = [\Delta u(k|k) \dots \Delta u(k+N_u-1|k)]^T \tag{6}$$

Only the first element of the determined sequence in Eqn. 6 is actually applied to the process, i.e.

$$u(k) = \Delta u(k|k) + u(k-1) \tag{7}$$

This whole procedure is then repeated for the next sampling time step.

3. Process Modeling Using Multiple Neural Networks

Recently, ensemble techniques have become very popular as methods for improving the generalization performance of unstable learners such as neural networks [13].

Bagging, an abbreviation for “bootstrap aggregation”, is extensively accepted as one of the most prevalent neural network ensemble methods. In this method, from a data set T comprising N examples one produces bootstrap re-samples $\{T_b^*\}$ by drawing with replacement N training patterns. The left over examples $V_b = T - T_b^*$ are usually used for validation purposes. Thus one generates B different members φ_b^* of the ensemble, the outputs of which on a test point x are finally averaged to create the aggregate prediction [14].

The bagging algorithm shown in Fig.3, can be summarized as:

i) Take a training set $T = \{(t_n, x_n)\}_{n=1}^N$

ii) For i = 1 to B:

a) Make a new training set $\{T^*\}_{b=1}^B$ by generating bootstrap re-samples of T. Each bootstrap re-sample T_b^* consists of N data pairs, sampled at random with replacement from T.

b) Train an estimator φ_b^* with the set T_b^* and add it to the ensemble.

iii) Using an appropriate weight w_b for each network aggregates these bootstrap versions by averaging to form a bagged prediction φ_{bag} as:

$$\varphi_{bag} = \sum_{b=1}^{B} w_b \varphi_b^*(x) \tag{8}$$

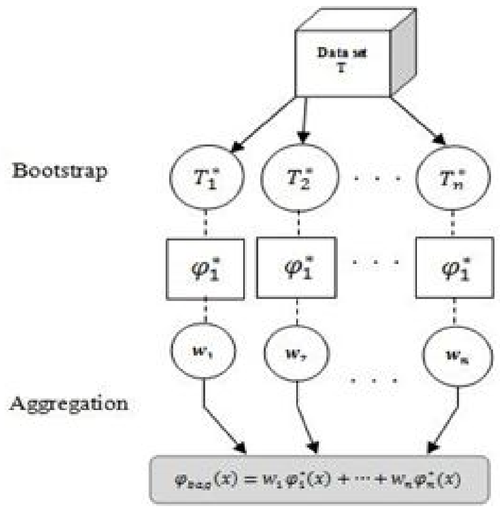

Fig. 3. Bagging

Recent advances in machine learning show that weighted ensembles of ANNs for regression can significantly improve prediction accuracy, generalization capability, and robustness against outliers when compared to SNNs. The weighting function employed in this work is given in [15] as:

$$w_i = \frac{\exp(-\alpha e_i)}{\sum_{j=1}^{B} \exp(-\alpha e_j)} \tag{9}$$

where e_i is the prediction error of the i-th model, B is the number of individual models in the ensemble, and α is a weighting coefficient. Setting $\alpha = 0$ results in the arithmetic mean for the weighting function.

Some of the advantages of the bagging algorithm are as follows:

i) It is easy to implement.
ii) It is very robust to noise. The noise-robustness property of the bagging algorithm is principally significant when the training data are contaminated with noise.
iii) Bagging decreases variance or model discrepancy over diverse data sets from a specified distribution, without increasing bias, which results in a reduced overall generalization error and enhanced stability.
iv) Bagging changes a collection of over-fitted neural networks into a better than perfectly-fitted network. Therefore, the conventional lengthy model selection is no longer needed. This could even counterbalance the computational overhead needed in bagging that includes training many neural networks.

4. Optimizer

GAs are flexible optimization algorithms with a probabilistic element that offers a method to search poorly understood, irregular spaces. Any GA begins with a population of randomly formed solutions, chromosomes, and progresses toward better solutions by applying genetic operators. GA is modeled on genetic processes taking place in nature [9].

GA works through function evaluation, not through differentiation or other such means. Thus GA is able to search for an optimum control sequence in a nonlinear control problem such as NMPC.

GA requires the problem of minimization to be stated in the form of an objective function. A set of variables for a given problem is encoded into a string (chromosome), and passed to an objective function to evaluate the fitness of that chromosome. GA selects parents from a pool of strings (population) according to the basic criteria of "survival of the fittest". Some members of the population undergo transformations by means of crossover and mutation operators to form new solutions. The crossover operator combines two parents to form a child. Mutation is a process by which GA is reinforced to reach the optimal solution through changing of a value with a probability P_m (the mutation probability) at a randomly selected point in the chromosome [16].

In this paper, the techniques proposed in [2] for GA is employed. The technique enables GA to be adopted in NMPC applications to calculate the control sequence. As shown in Fig. 4, the optimum string computed at time step k is modified to generate a new string which is always selected as one of a the strings at the (k+1)-th control step.

Furthermore, some of the best individuals at the k-th control interval are also included into the current initial population. This strategy improves the quality of current population and the stability of the NMPC algorithm.

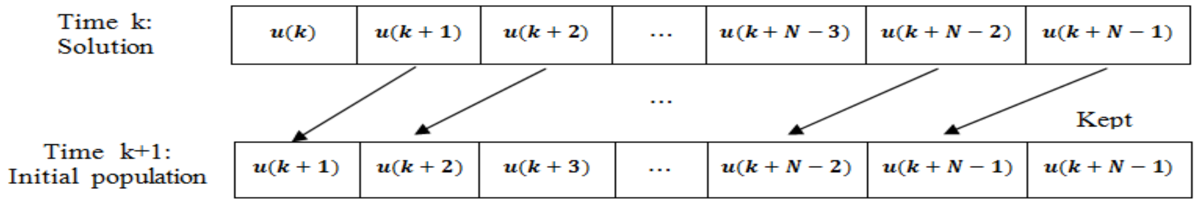

Fig. 4. The choice of initial population per iteration

Using the string shown in Fig. 4, the objective value J(k+1) at the (k+1)-th control interval is computed and compared with J(k). If J(k+1) is smaller than J(k) then this string is selected as the feasible solution, and its first element u(k+1|k+1) is applied to the plant. Otherwise, the optimization is performed as usual to compute the manipulating inputs [2].

In the GASQP developed in this work, GA determines approximate global estimates of the manipulating variables. These estimates are then used as the initial values for the SQP technique to improve the accuracy of the computed values.

5. Process Description

The CSTH [17] is used as the case study in this paper. Hot and cold water flow into the mixing tank, heated further using steam through a heating coil and drained from the tank through a long pipe. The configuration is shown in Fig. 5. The inputs are the cold water (CW) temperature, hot water (HW) temperature and steam valve. Outputs are level, cold and hot water flow and temperature. Manipulating CW will regulate the water tank temperature. The aim of simulation is to determine the dynamic responses of the temperature for specified variation in CW.

The CSTH is well mixed and therefore the temperature in the tank is assumed the same as the outflow temperature. The tank has a circular cross section with a volume of 0.81 m^3 and height of 50 cm. The initial values of temperatures of the hot and cold water feeds were set to 50 ℃ and 24 ℃, respectively. The maximum achievable temperature at the standard operating conditions is 65 ℃. For the temperature tests, the level was held constant at a set point of 20.48 cm. The sampling time used is 10 seconds. A normally distributed noise with zero mean and a standard deviation of 2 ℃ were added to the simulated tank temperature.

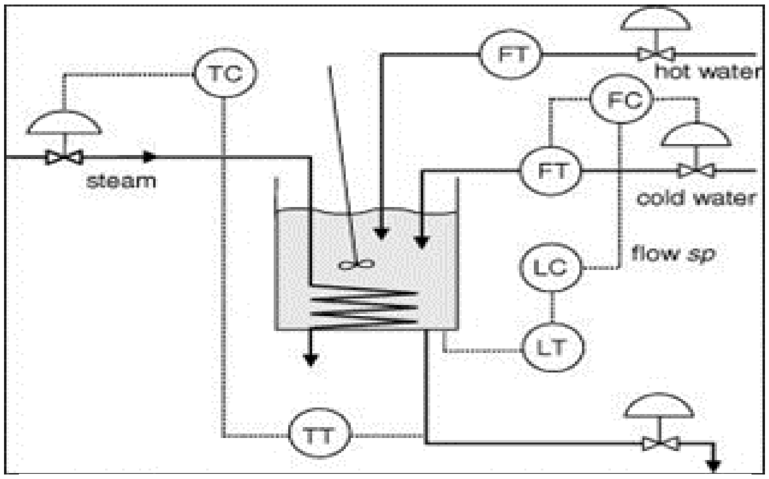

Fig. 5. The continuous stirred tank heater

6. RESULTS AND DISCUSSION

Numerous techniques have been used in bagging algorithm. These techniques include early stopping, epoch, NeuralBag, SECA and SimAnn [18]. In this work, the weighted forms of the early stopping (W-early stopping) and epoch (W-epoch) methods are implemented and then compared with each other. MNNs were combined using the weighted averaging approach. An MNN, consisting of 10 individual networks each of which having a single hidden layer with 10 hidden neurons, is developed to model the nonlinear dynamic relationship between the cold water temperature and the water temperature in the tank. An appropriate model structure for each component network was determined using cross validation. Table 1 shows the mean squared error of MNN resulting from various types of ensemble neural networks on the training and testing data averaged over 50 experiments. The best result in each case is highlighted in bold.

Table 1. Mean squared errors (in units of 10^{-4})

CSTH	single NN	early stopping	W-early stopping	epoch	W-epoch
Training data	8.43	4.86	4.72	2.14	1.77
Testing data	9.59	5.69	5.48	3.23	2.82

In this work, for construction of MNN the weighted epoch algorithm is employed. The NMPC algorithm is then developed using MNNs to control the tank temperature. The NMPC parameters are listed in Table 2.

The optimization method used is GA. For GA, the mutation probability is set to 0.1, population size is 10 and maximum number of generations is 20. For GASQP the maximum number of generations is 10. The tuning parameters of the PID controller are: $k_c = 6$, $\tau_I = 30$ and $\tau_D = 2$ as given by [17].

Fig. 6 compares the conventional PID with the NMPC based GA and the NMPC based GASQP for the given set point changes.

Table 2. Values of the NMPC parameters

Variables	Signification	Values
N	Prediction horizon	7
N_u	Control horizon	5
Y_{min}	Minimum temperature of the controlled variable	24 ℃
Y_{max}	Maximum temperature of the controlled variable	65 ℃
U_{min}	Minimum temperature of the manipulated variable	10 ℃
U_{max}	Maximum temperature of the manipulated variable	60 ℃
ΔU_{max}	Maximum change in manipulated variable	20
A	Output weight	1
B	Control weight	0.1

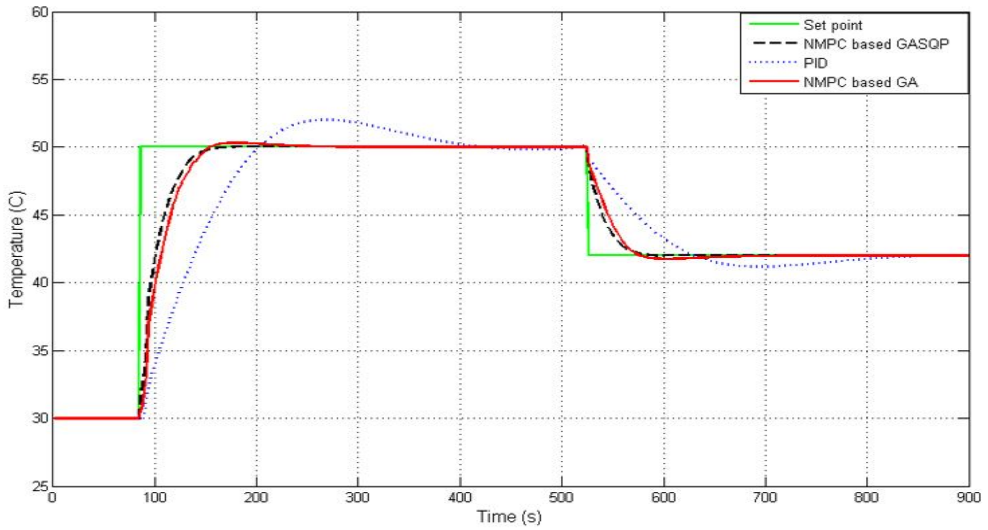

Fig. 6. Output responses for multiple set point changes in CSTH

Table 3 compares the performance of the three algorithms using the mean squared error (MSE) and percent of overshoot.

Table 3. Performance comparisons of simulation results

Method	Mean squared error	percent overshoot %
conventional PID	0.1412	10.13
NMPC based GA	0.0561	1.55
NMPC based GASQP	0.0409	0

Computational times at all sampling times of the NMPC based GA and the NMPC based GASQP are compared in Fig. 7. It is evident that the suboptimal NMPC algorithm based on GASQP has a considerably reduced demand on computational complexity.

These results show that NMPC based GA outperformed the conventional PID controller. In addition, the NMPC based GASQP method, developed in this work, provided improved control performance with lower computational load when compared against the NMPC based GA.

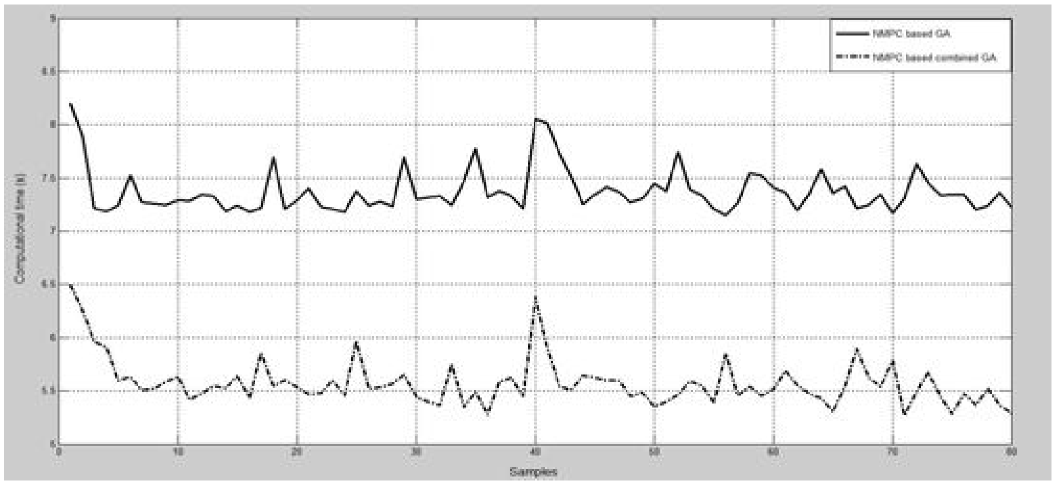

Fig. 7. The computational times of the NMPC based GA and NMPC base GASQP

In this paper, an implementation of the NMPC using an MNN model is proposed. Compared with a SNN model, in MNN the generalization, stability, smoothness and noise-robustness capabilities of neural network are improved. The component networks in an MNN were selected and combined in an attempt to obtain a better predictive model. In this work, several techniques were employed to construct MNNs. Among the discussed methods, the weighted epoch algorithm has shown better generalization performance compared to the other techniques. To determine the optimum control action an NMPC optimizer based on GA is used. In addition, GA was combined with SQP to develop the GASQP optimizer. The simulation results demonstrated that the NMPC based on GASQP technique provided enhanced control performance and reduced computational load when compared against the NMPC based on GA.

REFERENCES

1. Allgöwer, F., Findeisen, R., & Nagy, Z. K., 2004. Nonlinear Model Predictive Control : From Theory to Application, 35 (3): 299–315.

2. Chen, W., Zheng, T., Chen, M., & Li, X., 2011. Improved Nonlinear Model Predictive Control Based on Genetic Algorithm.

3. Jazayeri-Rad, H., 2004. The Nonlinear Model-Predictive Control of a Chemical Plant Using Multiple Neural Networks. *Neural Computing & Applications*, *13* (1): 2–15.

4. Granitto, P. M. M., Verdes, P. F. F., & Ceccatto, H. a. A., 2005. Neural Network Ensembles: Evaluation of Aggregation Algorithms. *Artificial Intelligence*, *163*(2), 139–162.

5. Breiman, L., 1996. Bagging Predictors. *Machine Learning*, *24* (2): 123–140.

6. Schapire, R. E., 1999. A Brief Introduction to Boosting. *In the Proceedings of International Joint Conference on Artificial Intelligence*, pp: 1401–1406.

7. Wolpert, D. H., 1992. Stacked Generalization. *Neural networks*, *5* (2): 241–259.

8. Zhou, Z.-H. H., Wu, J., & Tang, W., 2002. Ensembling Neural Networks: Many Could be Better than All. *Artificial Intelligence*, *137* (1-2): 239–263.

9. Goldberg, D. E., (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning* (Vol. 1).

10. Edgar, T. F., Himmelblau, D. M., & Lasdon, L. S., 2001. *Optimization of Chemical Processes* (Vol. 2).

11. Ower, F. A., Findeisen, R., Nagy, Z., Diehl, M., Bock, G., Oder, J. S., Allg, F., et al., 2007. Nonlinear Model Predictive Control for Large Scale Systems, pp: 43–54.

12. Stojanovski, G., & Stankovski, M., 2012. *Model Predictive Controller Employing Genetic Algorithm Optimization of Thermal Processes with Non-Convex Constraints.*

13. Carney, J. G., & Cunningham, P., 2000. Tuning Diversity in Bagged Ensembles. *International journal of neural systems*, *10* (4): 267–79.

14. Shafiei, E., & Jazayeri-Rad, H., 2012. Improving the Identification Performance of an Industrial Process Using Multiple Neural Networks. *American Journal of Intelligent Systems*, *2* (4): 40–44.

15. Romsdorfer, H., 2009. Weighted Neural Network Ensemble Models for Speech Prosody Control. *In the Proceedings of Tenth Annual Conference of the International Speech Communication Association*, pp: 492–495.

16. Bardsiri, V. K., Khatibi, A., and Khatibi, E., 2013. An Optimization-Based Method to Increase the Accuracy of Software. *Journal of Basic and Applied Scientific Research,* 3 (2): 159–166.

17. Thornhill, N. F., Patwardhan, S. C., & Shah, S. L., 2008. A Continuous Stirred Tank Heater Simulation Model With Applications. *Journal of Process Control, 18* (3-4): 347–360.

18. Granitto, P. M., Verdes, P. F., Navone, H. D., & Ceccatto, H. A., 2002. Aggregation Algorithms for Neural Network Ensemble Construction. *Neural Networks,*pp: 178–183.