Slovak University of Technology in Bratislava
Institute of Information Engineering, Automation, and Mathematics

PROCEEDINGS

17th International Conference on Process Control 2009
Hotel Baník, Štrbské Pleso, Slovakia, June 9 – 12, 2009
ISBN 978-80-227-3081-5
http://www.kirp.chtf.stuba.sk/pc09

Editors: M. Fikar and M. Kvasnica

Doležel, P., Taufer, I.: PSD Controller Tuning Using Artificial Intelligence Techniques, Editors: Fikar, M., Kvasnica, M., In *Proceedings of the 17th International Conference on Process Control '09*, Štrbské Pleso, Slovakia, 120–124, 2009.

Full paper online: http://www.kirp.chtf.stuba.sk/pc09/data/abstracts/007.html

PSD CONTROLLER TUNING USING ARTIFICIAL INTELLIGENCE TECHNIQUES

P. Doležel*, I. Taufer**

* *University of Pardubice, Faculty of Electrical Engineering and Informatics, Department of Process Control, Nám. Čs. legií 565, 532 10 Pardubice*
e-mail : petr.dolezel@upce.cz

** *University of Pardubice, Faculty of Electrical Engineering and Informatics, Department of Process Control, Nám. Čs. legií 565, 532 10 Pardubice*
e-mail : ivan.taufer@upce.cz

Abstract: There is described new method of PSD controller tuning in this paper. This method tunes PSD controller parameters online through the use of genetic algorithm and neural model of controlled system in order to control successfully even highly nonlinear systems. After method description and some discussion, there is performed comparison to one chosen conventional control technique.

Keywords: Artificial Neural Networks, Genetic Algorithm, PSD Controller

1 INTRODUCTION

Artificial neural networks represent effective tool for even highly nonlinear systems modeling. However, possibilities of neural model usage in process control are limited because control techniques in use (mostly based on PSD controllers applying) cannot employ neural models.

There are many well-known techniques of PSD controllers tuning. However, all of them suppose linear controlled system. The method explained here aims to tune PSD controller online. It expects knowledge of controlled system neural model and process of reference variable over known future finite horizon. The method amplifies the basic feedback control loop connection illustrated in Fig. 1. Its structure is illustrated in Fig. 2.

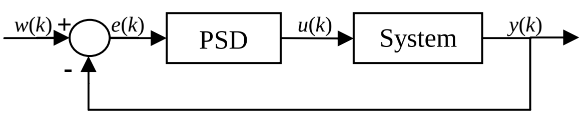

Fig. 1. Feedback control loop

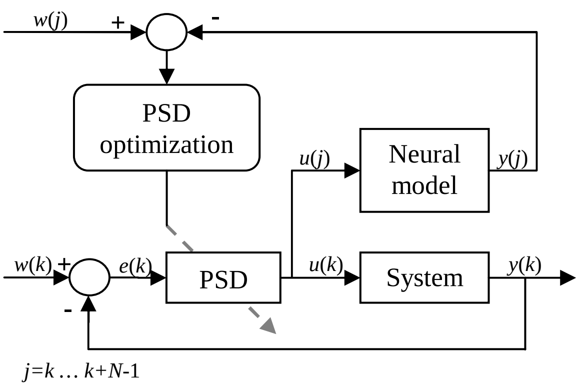

Fig. 2. Feedback control loop with PSD Controller Tuning Using Artificial Intelligence Techniques

So the premise is an availability of controlled system neural model and knowledge of reference variable process over future horizon *N*. Then there are chosen the parameters of PSD controller repeatedly every discrete time instant so that the control response computed via the neural model over future horizon is optimal (according to chosen performance criterion).

2 METHOD DESCRIPTION

It is clear that the crucial problem is to choose an optimization algorithm. The optimization of PSD controller parameters has to run repeatedly in every single step of sampling interval, which lays great demands on computing time of optimization algorithm. Naturally, there is suggested usage of some iterative optimization algorithm with only one iteration realization every time instant. Gradient descent techniques seem inconvenient because of neural model usage. Neural model is black-box-like model so it is not possible to determine gradient descent analytically. On the other hand, genetic algorithm (GA - see (Hynek 2008)) appears to be suitable because it does not require any particular information about optimization problem except of input variables ranges. The other indisputable advantage is its operating principle. In each iteration, GA explores not only one value of input variables but whole set of variables (one generation of individual solutions), which lowers significantly troubles with initial parameters random choice.

The control method described here does not require any special form of PSD controller. Most widely known form of PSD controller

$$\begin{aligned} u(k) &= q_0 \cdot e(k) + q_1 \cdot e(k-1) + \\ &+ q_2 \cdot e(k-2) + u(k-1) \end{aligned} \tag{1}$$

where $u(k)$ - manipulated variable

$e(k)$ - control error

q_0, q_1, q_2 - PSD controller parameters

suits quite well. However, controller behaviour dependence on variation of parameters q_0, q_1, q_2 is not completely clear and some parameters can get both positive and negative value. In term of GA using, it seems more convenient to use that form of PSD controller whose values of parameters are at least unilaterally bounded. It is realized in the PSD controller of form (Bobál *et al.* 1999)

$$u(k) = u_P(k) + u_I(k) + u_D(k) \tag{2}$$

where $u_P(k) = q'_0 \cdot e(k)$

$$u_I(k) = u_I(k-1) + q'_1 \cdot e(k)$$

$$u_D(k) = q'_2 \cdot \left[e(k) - e(k-1)\right]$$

It is obvious that the form of PSD controller described by Eq. (2) is formally similar to continuous-action PID controller hence all the parameters q'_0, q'_1, q'_2 will be positive for controlled systems with positive gain. This information will improve accuracy of GA results.

3 ALGORITHM RESUMPTION

Whole algorithm of described control method is compiled in following points:

1. Create dynamical neural model of controlled system
2. Choose future horizon length N
3. Choose GA parameters (number of individual solutions in one generation, length of chromosome, conversion between phenotype and PSD controller parameters definition) and their initial values
4. Measure controlled variable $y(k)$
5. Perform one iteration of GA (based on the knowledge of controlled variable $y(k)$, process of its reference $w(k)$ till $w(k+N-1)$ and neural model of controlled system)
   a) perform control simulation with PSD controller and the neural model over future horizon N and evaluate cost function (fitness function in GA nomenclature) for all the individual solutions from current generation
   b) Determine and save best solution (elitism)
   c) Select individual solutions for next generation breeding through their fitness function values (tournament selection, roulette wheel selection, …)
   d) Apply cross-over (e.g. one point cross-over with random point of cross-over)
   e) Apply mutation with dynamically changing value of probability (mutation probability should rise with lowering selection pressure)
   f) Evaluate fitness functions of offspring (see step a)) and replace the poorest offspring solution by the best solution obtained from step b)
   g) Choose the best individual solution from next generation
6. Evaluate manipulated variable $u(k)$ with PSD controller determined by best individual solution obtained in step 5g)
7. $k = k + 1$, go to step 4

There will be described few remarks in next sentences.

Future horizon length N is important parameter of the algorithm. There are no exact rules how to choose it. Too short horizon does not provide sufficient data to GA. However, too long one brings data so distant from the current state that this data should not influence next controller output value. It has to be mentioned that long future horizon length causes long computing time (computing time is one of key troubles).

There is similar situation in choice of number of individual solutions in each generation and in choice of length of chromosome. Their rising leads to better control performance but it extends the computing time immoderately.

Mutation is key part of GA in this case. The only mutation can ensure sufficient diversity of individual solutions in population. Optimization works online so fitness function parameters are changed in each iteration step. Thus, solutions, which seem acceptable in one iteration step, can lead up to unstable control response in another iteration step. Mutation has to ensure sufficient diversity of individual solutions so that each generation contains solution leading at least to stable control performance.

Suitable definition of cost function (fitness function) is

$$J = \frac{1}{N} \cdot \sum_{i=k}^{k+N-1} |e(i)| + \frac{h_1}{N-1} \cdot \sum_{i=k+1}^{k+N-1} |\Delta u(i)| + \\ + h_2 \cdot |e(k+N-1)| \tag{3}$$

where $\Delta u(i) = u(i) - u(i-1)$

$e(i)$ - control error $w(i)$ - $y(i)$

h_1 - function parameter influencing manipulated variable differences

h_2 - function parameter influencing the state on the end of future horizon

N - length of future horizon

$w(i)$ - reference variable

Eventually, Most of real controlled systems have constrained inputs. It is useful to include that limitation to control simulation (step 5a)) in order to influence PSD controller parameters optimization.

4 EXAMPLE OF NONLINEAR SYSTEM CONTROL

Demonstrative nonlinear controlled system is described by the function

$$\begin{aligned} & y(k) - 1{,}236 \cdot y(k-1) + 0{,}3772 \cdot y(k-2) + \\ & + 0.1000 \cdot [y(k-1)]^2 = 0{,}08191 \cdot u(k-1) + \\ & + 0{,}05918 \cdot u(k-2) + 0{,}05000 \cdot u(k-1) \cdot \\ & \cdot u(k-2) + 0{,}2000 \cdot [u(k-1)]^2 \end{aligned} \tag{4}$$

For apprehension, there is shown response of system (4) to sum of delayed step functions in Fig. 3.

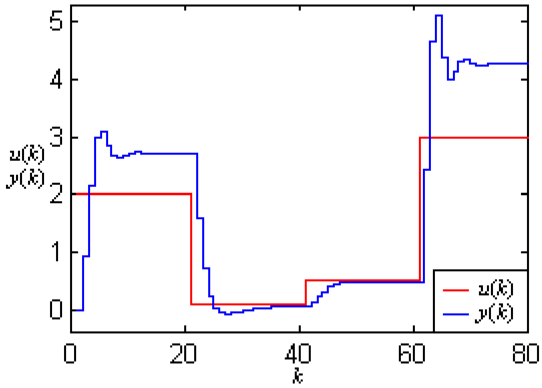

Fig. 3. System response to sum of delayed step functions

Control design was made according to paragraph 3.

First, there was designed dynamical neural model of controlled system (see (Taufer *et al.* 2008)) in form of equation

$$\hat{y}(k) = \mathrm{NET}[\hat{y}(k-1), \hat{y}(k-2), u(k-1), u(k-2)] \tag{5}$$

Then, there were chosen following parameters based on compromise between control performance and computing time:

Future horizon length N	50
Number of individual solutions	14
Chromosome length	36 binary values
Mutation probability	10^{-4} for high selection pressure
	0.3 for low selection pressure

Low selection pressure was defined for cases when the fitness function value of best individual solution was at the most five percent more favourable than average of all fitness function values in current generation.

As there were optimized three parameters of PSD controller (2), there had to be defined conversion formula between phenotype of each solution and mentioned three parameters. Several simulations proved following formula to be sufficient:

$$q_0' = \frac{\sum_{i=1}^{12} ch(i) \cdot 2^{12-i}}{4000}, \quad q_1' = \frac{\sum_{i=13}^{24} ch(i) \cdot 2^{24-i}}{4000}, \\ q_2' = \frac{\sum_{i=25}^{36} ch(i) \cdot 2^{36-i}}{4000} \tag{6}$$

where ch - vector of values included in each solution chromosome

Cost function was defined by Eq. (3) whereas $h_1 = 0.4$ and $h_2 = 0.2$.

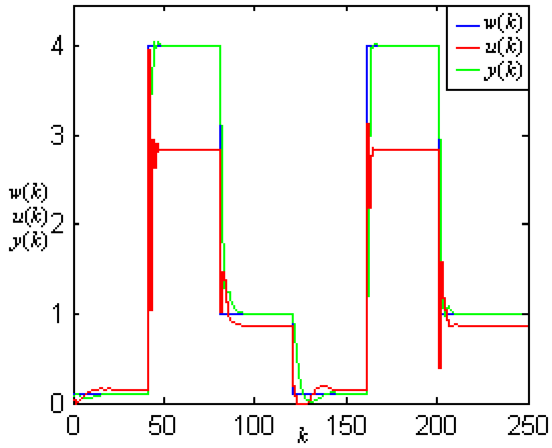

Fig. 4. Control response

From Eqs. (6), it is obvious that PSD controller parameters can get values from interval (0; 1.02375) with uncertainty of about $2.5 \cdot 10^{-4}$.

It was simulated control response (Fig. 4.) for mentioned values, random initial generation of individual solutions and chosen process of reference variable *w*. Manipulated variable $u(k)$ was constrained on interval <0; 5>.

Retrieved control response was compared to response gained by common control technique. It was chosen LQ control technique derived from Algebraic Control Theory which is described in (Drábek *et al.* 1987). This technique tuned controller with two degrees of freedom and integral element according to the criterion

$$J' = \sum_{k=0}^{\infty} \left\{ [w(k) - y(k)]^2 + h \cdot [u(k)]^2 \right\} \tag{7}$$

The technique required linear ARX model of controlled system. Second order ARX model was obtained by Least Mean Square Identification Technique with the same data which was used to neural model design. That ARX model was updated online with Recurrent Least Mean Square Identification Technique with forgetting factor α = 0.9 (Drábek *et al.* 1987). Through that ARX model, there was built controller with two degrees of freedom adaptively each time instant *k* according to criterion (7) with parameter h = 10. Final control response on equal terms like previous one is figured in Fig. 5. Comparison of Fig. 4 and Fig. 5 tells that in this case (and many others) PSD Controller Tuning Using Artificial Intelligence Techniques provides much better performance than certain conventional method.

5 CONCLUSIONS

There is described control method in this paper, which employs artificial intelligence techniques.

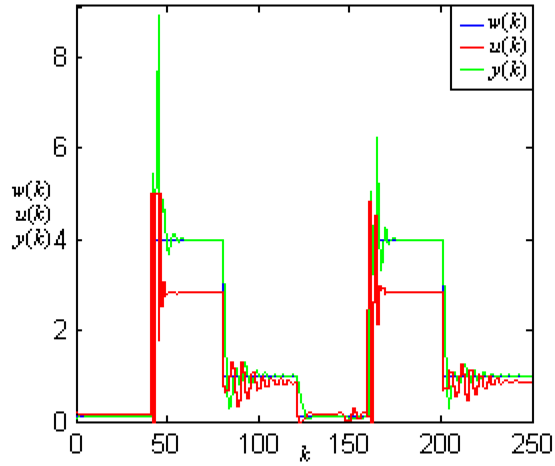

Fig. 5. Control response with LQ controller

The method is suitable especially for highly nonlinear time-invariant systems control.

It can utilize manipulated variable limitations in a certain manner, which is not quite common feature. On the other hand, it requires precise neural model of controlled system, which can be difficult to obtain. The method is computationally demanding so it is rather suitable for systems with longer sample time (decimals of seconds and longer according to applied computer). There is included significant stochastic element in this method due to GA so every other control response is different from previous one.

Algorithm itself could be slightly improved before being applied. For instance, it seems convenient to reduce violent changes of manipulated variable (e.g. by low-pass filter). Method can be modified for other types of controllers, too.

In fine, described control technique has abilities to control highly nonlinear time-invariant systems which had to be controlled by adaptive control techniques till this time. However, it is not proper for time-variant systems control without modifications needed to be made.

ACKNOWLEDGMENTS

The work has been supported by program of Czech Republic MSM 0021627505. This support is very gratefully acknowledged.

6 REFERENCES

Bobál, V., Böhm, J., Prokop, R. (1999). *Praktické aspekty samočinně se nastavujících regulátorů: algoritmy a implementace.* VITIUM, Brno. ISBN 80-214-1299-2

Drábek, O., Macháček, J. (1987). *Experimentální identifikace*. VŠChT Pardubice, Pardubice.

Hynek, J. (2008). *Genetické algoritmy a genetické programování*. Grada Publishing, Prague. ISBN 978-80-247-2695-3

Taufer, I., Drábek, O., Seidl, P. (2008). Umělé neuronové sítě – základy teorie a aplikace (10), *CHEMagazín*, vol. XVII, issue 1, 35-37. ISSN 1210-7409