GENERALIZED PREDICTIVE CONTROL WITH NON-LINEAR MODEL

Vojtěch Mikšánek
Doctoral Degree Programme (1), FEEC BUT
E-mail: xmiksa00@stud.feec.vutbr.cz

Supervised by: Petr Pivoňka
E-mail: pivonka@feec.vutbr.cz

ABSTRACT

This paper deals with Model Predictive Control (MPC) – a control algorithm which optimizes control action and output error within a time horizon. A nonlinear model is obtained by using Neural Network and has to be linearized in each operating point. A model is used to predict the future behaviour. First, the control algorithm was implemented in MATLAB/Simulink and tested on mathematical models. After that it was implemented in the Programmable Logical Controller (PLC) B&R and tested on a physical model.

1. INTRODUCTION

The beginnings of Model Predictive Control (MPC) date back to the 1970s. Model Predictive Control integrates optimal control, dead time processes control, multivariable control and future references when available. The MPC is not a specific control strategy but an ample range of control methods where the control signal is obtained by minimizing an objective function. The model is the cornerstone of the MPC wherefore it is necessary to obtain the best possible model, and that can be done by using Neural Network.

Predictive control algorithm was written in ANSI C and tested in MATLAB/Simulink and implemented in Programmable Logical Controller B&R.

2. GENERALIZED PREDICTIVE CONTROL

Generalized Predictive Control (GPC) is one of the most popular methods of predictive control. It was proposed in 1987 in [2] and has become one of the most popular MPC methods [1] in both industry and academia. The Generalized Predictive Control algorithm consists in applying a control sequence that minimizes a cost function (1).

$$J(p,r,\lambda) = \sum_{j=1+d}^{p+d} \left[\hat{y}(t+j|t) - w(t+j)\right]^2 + \lambda \sum_{j=1}^{r} \left[\Delta u(t+j-1)\right]^2 \qquad (1)$$

where $\hat{y}(t+j|t)$ is the predicted system output in *j*-th prediction step in discrete time *t*, $w(t+j)$ is reference trajectory, $\Delta(t+j)$ is *j*-th increment of control action, *p* is predicted horizon, *r* is control horizon, *λ* is cost constant and *d* is delay. The first term considers the predicted error and the second term considers penalized future control increments.

The criterion (1) can be rewritten to a matrix form [1]:

$$J = (\mathbf{G}\mathbf{u} + \hat{\mathbf{y}} - \mathbf{w})^{\mathrm{T}} (\mathbf{G}\mathbf{u} + \hat{\mathbf{y}} - \mathbf{w}) + \lambda\, \mathbf{u}^{\mathrm{T}}\mathbf{u} \tag{2}$$

where $\hat{\mathbf{y}}$ is vector of predicted system outputs for prediction horizon, $\mathbf{w}$ is vector of future references.

$$\mathbf{u} = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+r-1) \end{bmatrix} \quad \hat{\mathbf{y}} = \begin{bmatrix} \hat{y}(t+d+1\,|\,t) \\ \hat{y}(t+d+2\,|\,t) \\ \vdots \\ \hat{y}(t+d+p\,|\,t) \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w(t+d+1\,|\,t) \\ w(t+d+2\,|\,t) \\ \vdots \\ w(t+d+p\,|\,t) \end{bmatrix}$$

$\mathbf{G}$ is matrix of dynamics

$$\mathbf{G} = \begin{bmatrix} \left.\frac{\partial f}{\partial u_0}\right|_{k=1} & \left.\frac{\partial f}{\partial u_1}\right|_{k=1} & \cdots & \left.\frac{\partial f}{\partial u_{r-1}}\right|_{k=1} \\ \left.\frac{\partial f}{\partial u_0}\right|_{k=2} & \left.\frac{\partial f}{\partial u_1}\right|_{k=2} & \cdots & \left.\frac{\partial f}{\partial u_{r-1}}\right|_{k=2} \\ \vdots & \vdots & \ddots & \vdots \\ \left.\frac{\partial f}{\partial u_0}\right|_{k=r-1} & \left.\frac{\partial f}{\partial u_1}\right|_{k=r-1} & \cdots & \left.\frac{\partial f}{\partial u_{r-1}}\right|_{k=r-1} \\ \vdots & \vdots & \ddots & \vdots \\ \left.\frac{\partial f}{\partial u_0}\right|_{k=p} & \left.\frac{\partial f}{\partial u_1}\right|_{k=p} & \cdots & \left.\frac{\partial f}{\partial u_{r-1}}\right|_{k=p} \end{bmatrix}, \qquad \text{where } y = f(x_0, x_1, \ldots, x_{r-1}) \qquad \text{and}$$

$k = 1, 2, \ldots, p$, as shown in [5].

For linear causal systems:

$$\mathbf{G} = \begin{bmatrix} g_0 & 0 & \cdots & 0 \\ g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{r-1} & g_{r-2} & \cdots & g_0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{p-1} & g_{p-2} & \cdots & g_{p-r} \end{bmatrix}$$

where element g_j is j-th coefficient of model step response (3).

$$g_j = -\sum_{i=1}^{j} a_i g_{j-i} + \sum_{i=0}^{j} b_i \tag{3}$$

Cost function minimum (2) is obtained by making the gradient of J equal zero [1]. The result is equation (4), which is used for computation of the future control action increments vector.

$$\mathbf{u} = (\mathbf{G}^{\mathrm{T}}\mathbf{G} + \lambda \mathbf{I})^{-1} \mathbf{G}^{\mathrm{T}} (\mathbf{w} - \hat{\mathbf{y}}) \tag{4}$$

where **k** is the first row of the matrix $\left(\mathbf{G}^{\mathrm{T}}\mathbf{G}+\lambda\mathbf{I}\right)^{-1}\mathbf{G}^{\mathrm{T}}$.

$$\Delta u(k)=\mathbf{k}\left(\mathbf{w}-\mathbf{y}\right) \tag{5}$$

Only the first increment of control action is used for control (5).

3. IDENTIFICATION

The model is the cornerstone of MPC. Neural Networks (NN) are used for realizing the model of an analog physical model. The analog model, which contains operational amplifiers, resistors and capacitors, represents the third order process. The analog model is not completely linear and for this reason the network contains neurons with non-linear transfer functions. The transfer function of neurons in the hidden layer is sigmoid and the transfer function of output neuron is linear. Inputs of the Neural Network are delayed last three process inputs and outputs (**Fig. 1**).

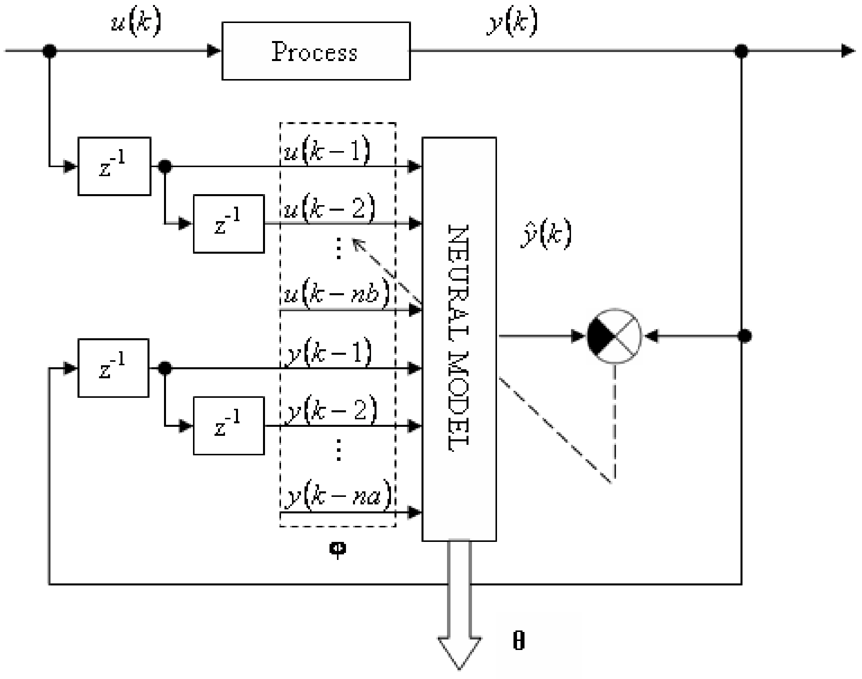

Fig. 1: Neural model of process.

To train the Neural Network we use the Levenberg-Marquardt (LM) algorithm [3]

$$\boldsymbol{\theta}(k+1)=\boldsymbol{\theta}(k)-\left(\mathbf{J}^{\mathrm{T}}\mathbf{J}+\gamma\mathbf{I}\right)^{-1}\mathbf{J}^{\mathrm{T}}\boldsymbol{\varepsilon}(k) \tag{6}$$

where **J** is Jacobian $\mathbf{J}(\boldsymbol{\theta})=\frac{\partial\boldsymbol{\varepsilon}(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}$, **θ** is weight vector a **ε** is vector of neural network's output errors.

4. APPLICATION OF GPC

The predictive control algorithm was written in ANSI C and tested in MATLAB/Simulink on mathematical models. After that the GPC was implemented in the Programmable Logical Controller B&R.

The nonlinear neural model obtained by Levenberg-Marquardt training algorithm has to be linearized because of using in GPC algorithm. The third order ARMA model was obtained by linearization in each operating point according to [4]. In **Fig. 2** is shown the comparison of GPC and PSD. The GPC parameters were $p = 10$, $r = 5$ and $\lambda = 0.1$.

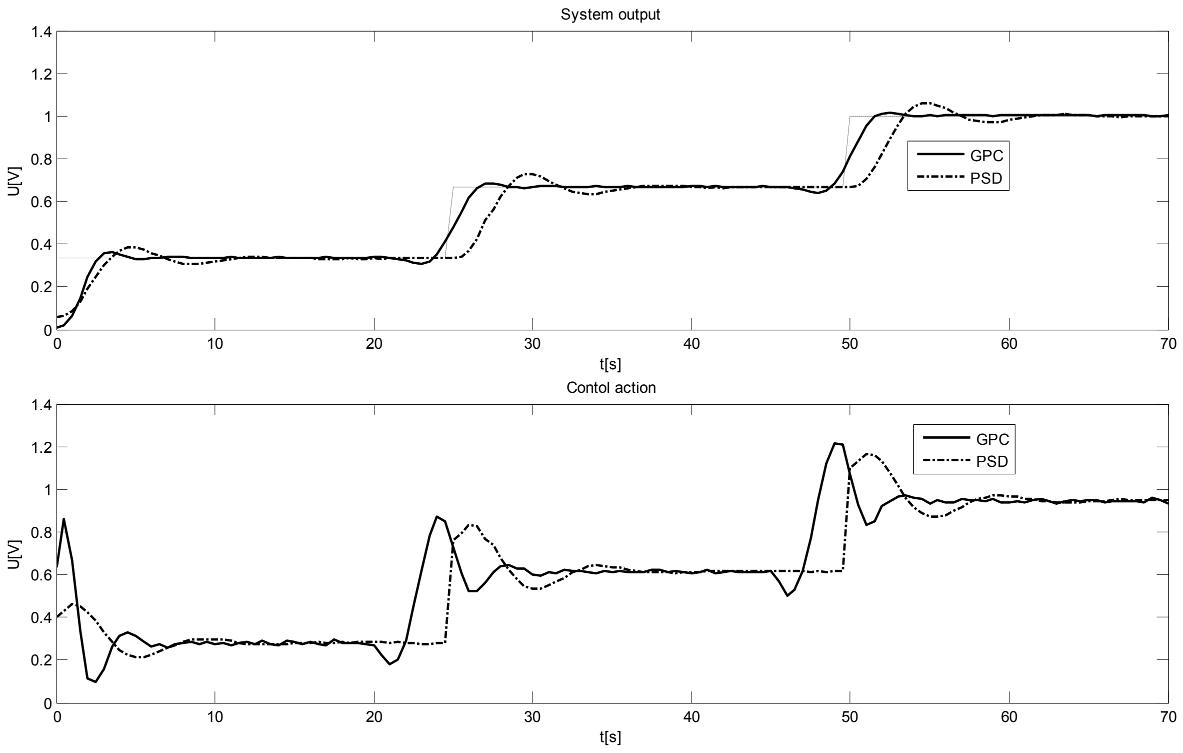

Fig. 2: Comparison of GPC and PSD.

Criterion \ controller	GPC	PSD
$c_{\mathrm{Q}} = \sum \left[y(k) - w(k) \right]^2$	0,4	1,1
$c_{\mathrm{E}} = \sum \left[u(k) \right]^2$	58	65
$c_{\mathrm{D}} = \sum \left[\Delta u(k) \right]^2$	0,7	1

Tab. 1: Quality of regulation GPC a PSD.

5. CONCLUSION

This paper shows one of the most popular methods of Model Predictive Control. The General Predictive Control was implemented in PLC B&R and tested on physical models and compared with PSD. Parameters of the PSD were obtained by minimizing an objective function in MATLAB by function *fminsearch*. To obtain the PSD parameters was used linear third order ARMA model and the objective function is similar to the function (1).

The cornerstone of Generalized Predictive Control is the model used to predict the future behavior of the system output. Therefore it is necessary to obtain the best possible model. The model is obtained by using a Neural Network with the Levenberg-Marquardt training algorithm. This nonlinear neural model has to be linearized in each operating point according to [4].

The GPC deals with future behavior of process and reference trajectory therefore it can react to change of the trajectory before it happened as shown in Fig. 2.

Acknowledgement: The paper has been prepared as a part of the solution of Czech Science Foundation GAČR project No. 102/06/1132 Soft Computing in Control and by the Czech Ministry of Education in the frame of MSM MSM0021630529 Intelligent Systems in Automation.

REFERENCES

[1] Camacho, E. F. – Bordons, C.: Model Predictive Control, London, Springer 1999, ISBN 3-540-76241-8

[2] Clarke, D. W. – Mohtadi, C. – Tuffs, P. S.: Generalized Predictive Control – Part I. The Basic Algorithm, Automatica, 23, s. 137-148 (1987)

[3] Hristev, R. M.: The ANN Book, GNU Public License, 1998

[4] Sunan, H. – Kiong, T. K. – Heng, L. T.: Applied Predictive Control, London, Springer 2002, ISBN 1-85233-338-3

[5] Vychodil, H.: Generalized Predictive Control with a Non-linear Autoregressive Model. Automatic Control Modeling and Simulation ACMOS'05. Praha: WSEAS, 2005, pp. 85 - 89, ISBN 960-8457-12-2

[6] Pivoňka, P. - Nepevný, P.: Generalized Predictive Control with Adaptive Model Based on Neural Networks. In Proceedings of the WSEAS Conferences NN'05, FS'05, EC'05. Lisabon: WSEAS, 2005, pp. 1 - 5, ISBN 960-8457-24-6

[7] Mikšánek, V.: Predictive controllers in MATLAB- PLC environment (in Czech). Diploma thesis, FEEC BUT, Brno 2006