Computation of Predictions in Multivariable Predictive Control

MAREK KUBALČÍK, VLADIMÍR BOBÁL
Tomas Bata University in Zlín
Department of Process Control
Centre of Polymer Systems
Nad Stráněmi 4511, 76005, Zlín
CZECH REPUBLIC
kubalcik@fai.utb.cz, bobal@fai.utb.cz
http://web.fai.utb.cz/

Abstract: - Model Based Predictive Control (MBPC) or only Predictive Control is one of the control methods which have developed considerably over a few past years. It is mostly based on discrete models of controlled systems. Model of a controlled system is used for computation of predictions of the systems output on the basis of past inputs, outputs and states and designed sequence of future control increments. This paper is focused in comparison of various approaches to computation of multi – step ahead predictions using a multivariable input – output model. Computational aspects of derivation of predictions can be limitting especially in adaptive predictive control.

Key-Words: - Predictive control, multivariable systems, multi-step-ahead prediction, Diophantine equations, CARIMA model

1 Introduction

Model Based Predictive Control (MBPC) or only Predictive Control [1], [2], [3] is one of the control methods which have developed considerably over a few past years. Predictive control is essentially based on discrete or sampled models of processes. Computation of appropriate control algorithms is then realized especially in the discrete domain.

The basic idea of MPC [4], [5] is to use a model of a controlled process to predict N future outputs of the process. A trajectory of future manipulated variables is given by solving an optimization problem incorporating a suitable cost function and constraints. Only the first element of the obtained control sequence is applied. The whole procedure is repeated in following sampling period. This principle is known as the receding horizon strategy. The computation of a control law of MPC is based on minimization of the following criterion

$$J(k)=\sum_{j=1}^{N}\boldsymbol{e}(k+j)^2+\lambda\sum_{j=1}^{N_u}\Delta\boldsymbol{u}(k+j)^2 \qquad (1)$$

where $\boldsymbol{e}(k+j)$ is a vector of predicted control errors, $\Delta\boldsymbol{u}(k+j)$ is a vector of future increments of manipulated variables (for the system with two inputs and two outputs each vector has two elements), N is length of the prediction horizon, N_u is length of the control horizon and λ is a weighting factor of control increments.

A predictor in a vector form is given by

$$\hat{\boldsymbol{y}}=\boldsymbol{G}\Delta\boldsymbol{u}+\boldsymbol{y}_0 \qquad (2)$$

where $\hat{\boldsymbol{y}}$ is a vector of system predictions along the horizon of the length N. The first element in the equation (2) represents the forced response of the system. $\Delta\boldsymbol{u}$ is a vector of control increments and $\boldsymbol{G}$ is a matrix of the dynamics, $\boldsymbol{y}_0$ is the free response vector.

The first task is computation of predictions for an arbitrary prediction horizon. Dynamics of most of processes require horizons of length where it is not possible to compute predictions in a simple straightforward way. For a particular model, it is necessary to compute prediction equations. The most often used models in applications and academic papers are state – space and input output models. For state-space models computation of predictions is rather obvious. For input – output models there are several approaches how to compute prediction equations. All the approaches result to the same prediction equations. But computational demands for particular methods are different. Of course, the main computational problem in predictive control is solving the optimization problem. But in adaptive control [6], [7] when it is necessary to compute prediction equations in each sampling period the computational time consumption can be important. It can also be important while using the prediction for other purposes than for the predictive control. Some of the methods also are not algorithmically understandable and clear.

One of the main advantages of predictive control is its simple applicability for control of multi – input multi – output (MIMO) systems. It is one of the most effective approaches to control of multivariable systems since multivariable systems can be handled in a straightforward manner. The aim of the paper is then to compare various approaches to computation of multi –

step ahead predictions using a multivariable input – output model.

2 Model of the System

Let us consider a two input – two output system. The two – input/two – output (TITO) processes are the most often encountered multivariable processes in practice and many processes with inputs/outputs beyond two can be treated as several TITO subsystems [8].

A general transfer matrix of a two-input–two-output system with significant cross-coupling between the control loops is expressed as:

$$\boldsymbol{G}(z)=\begin{bmatrix} G_{11}(z) & G_{12}(z) \\ G_{21}(z) & G_{22}(z) \end{bmatrix} \tag{3}$$

$$\boldsymbol{Y}(z)=\boldsymbol{G}(z)\boldsymbol{U}(z) \tag{4}$$

where $\boldsymbol{U}(z)$ and $\boldsymbol{Y}(z)$ are vectors of the manipulated variables and the controlled variables, respectively.

$$\boldsymbol{U}(z)=[u_1(z),u_2(z)]^T \qquad \boldsymbol{Y}(z)=[y_1(z),y_2(z)]^T \tag{5}$$

It may be assumed that the transfer matrix can be transcribed to the following form of the matrix fraction:

$$\boldsymbol{G}(z)=\boldsymbol{A}^{-1}(z^{-1})\boldsymbol{B}(z^{-1})=\boldsymbol{B}_1(z^{-1})\boldsymbol{A}_1^{-1}(z^{-1}) \tag{6}$$

where the polynomial matrices $\boldsymbol{A}\in R_{22}[z^{-1}]$, $\boldsymbol{B}\in R_{22}[z^{-1}]$ are the left coprime factorizations of matrix $\boldsymbol{G}(z)$ and the matrices $\boldsymbol{A}_1\in R_{22}[z^{-1}]$, $\boldsymbol{B}_1\in R_{22}[z^{-1}]$ are the right coprime factorizations of $\boldsymbol{G}(z)$. The model can be also written in the form

$$\boldsymbol{A}(z^{-1})\boldsymbol{Y}(z)=\boldsymbol{B}(z^{-1})\boldsymbol{U}(z) \tag{7}$$

As an example a model with polynomials of second order was chosen. This model proved to be effective for control of several TITO laboratory processes [9], where controllers based on a model with polynomials of the first order failed. The model has sixteen parameters. The matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are defined as follows

$$\boldsymbol{A}(z^{-1})=\begin{bmatrix} 1+a_1z^{-1}+a_2z^{-2} & a_3z^{-1}+a_4z^{-2} \\ a_5z^{-1}+a_6z^{-2} & 1+a_7z^{-1}+a_8z^{-2} \end{bmatrix} \tag{8}$$

$$\boldsymbol{B}(z^{-1})=\begin{bmatrix} b_1z^{-1}+b_2z^{-2} & b_3z^{-1}+b_4z^{-2} \\ b_5z^{-1}+b_6z^{-2} & b_7z^{-1}+b_8z^{-2} \end{bmatrix} \tag{9}$$

A widely used model in general model predictive control is the CARIMA model which we can obtain from the nominal model (7) by adding a disturbance model

$$\boldsymbol{A}(z^{-1})\boldsymbol{y}(k)=\boldsymbol{B}(z^{-1})\boldsymbol{u}(k)+\frac{\boldsymbol{C}(z^{-1})}{\Delta}\boldsymbol{e}_s(k) \tag{10}$$

where $\boldsymbol{e}_s(k)$ is a non-measurable random disturbance that is assumed to have zero mean value and constant covariance and the operator delta is an integrator. The matrix $\boldsymbol{C}(z^{-1})$ will be further considered as 2x2 identity matrix. Application of this model enables to achieve integral action.

Difference equations of the incremental form without the unknown term are as follows

$$\begin{aligned} y_1(k+1)&=(1-a_1)y_1(k)+(a_1-a_2)y_1(k-1)+a_2y_1(k-2)-\\ &-a_3y_2(k)+(a_3-a_4)y_2(k-1)+a_4y_2(k-2)+b_1\Delta u_1(k)+\\ &+b_2\Delta u_1(k-1)+b_3\Delta u_2(k)+b_4\Delta u_2(k-1)\\ y_2(k+1)&=(1-a_7)y_2(k)+(a_7-a_8)y_2(k-1)+a_8y_2(k-2)-\\ &-a_5y_1(k)+(a_5-a_6)y_1(k-1)+a_6y_1(k-2)+b_5\Delta u_1(k)+\\ &+b_6\Delta u_1(k-1)+b_7\Delta u_2(k)+b_8\Delta u_2(k-1) \end{aligned} \tag{11}$$

3 Computation of Predictions

In academic literature there are described several methods for computation of prediction equations for models based on transfer function. This paper will be focused in most often used approaches: methods based on Diophantine equations [1], methods based on matrix operations [10] and straightforward computation on the basis of the CARIMA model [11]. Particular methods will be described in the following subsections.

3.1 Method Based on Matrix Operations

A general difference equation for one – step ahead prediction can be written as follows

$$\begin{aligned} \boldsymbol{y}(k+1)&=\boldsymbol{A}_1\boldsymbol{y}(k)+\cdots+\boldsymbol{A}_{n_a+1}\boldsymbol{y}(k-n_a)+\boldsymbol{B}_1\Delta\boldsymbol{u}(k)+\\ &+\boldsymbol{B}_2\Delta\boldsymbol{u}(k-1)+\cdots+\boldsymbol{B}_{n_b}\Delta\boldsymbol{u}(k-n_b+1) \end{aligned} \tag{12}$$

where n_a is the order of the matrix $\boldsymbol{A}$ and n_b is the order of the matrix $\boldsymbol{B}$.

Now we can formulate equation for i-step ahead prediction.

$$\hat{\boldsymbol{y}}(k+i)=\boldsymbol{H}^{[i]}\underset{\rightarrow k-1}{\Delta\boldsymbol{u}}+\boldsymbol{P}^{[i]}\underset{\leftarrow k-1}{\Delta\boldsymbol{u}}+\boldsymbol{Q}^{[i]}\underset{\leftarrow k}{\boldsymbol{y}} \tag{13}$$

It consists of three terms. As for the notation, the arrow pointing right is used for strictly future – not including current value and arrow pointing left is used for past including current value. The particular terms are then past output values, past control increments and future control increments. The matrices $\boldsymbol{H}$, $\boldsymbol{P}$ and $\boldsymbol{Q}$ are matrices of coefficients. Initialization of the matrices $\boldsymbol{H}$, $\boldsymbol{P}$ and $\boldsymbol{Q}$ for i equal to one is as follows

$$\begin{aligned} &\boldsymbol{H}^{[1]}=[\boldsymbol{B}_1,0,0,0,\cdots]; \quad \boldsymbol{P}^{[1]}=[\boldsymbol{B}_2,\boldsymbol{B}_3,\cdots,\boldsymbol{B}_{n_b}];\\ &\boldsymbol{Q}^{[1]}=[\boldsymbol{A}_1,\boldsymbol{A}_2,\cdots,\boldsymbol{A}_{n_a+1}] \end{aligned} \tag{14}$$

Further a recursive substitution will be used to find prediction (15)

$$\hat{\boldsymbol{y}}(k+i+1)=\boldsymbol{H}^{[i]}\underset{\rightarrow k}{\Delta\boldsymbol{u}}+\boldsymbol{P}^{[i]}\underset{\leftarrow k}{\Delta\boldsymbol{u}}+\boldsymbol{Q}^{[i]}\underset{\leftarrow k+1}{\boldsymbol{y}} \tag{15}$$

Following expressions which can be substituted into prediction equation (15) can be derived

$$\begin{aligned} \boldsymbol{Q}^{[i]}\underset{\leftarrow k+1}{\boldsymbol{y}}&=[\boldsymbol{Q}^{[i]},0]\begin{bmatrix}\boldsymbol{y}(k+1)\\ \underset{\leftarrow k}{\boldsymbol{y}}\end{bmatrix}=\\ &=\boldsymbol{Q}_1^{[i]}\boldsymbol{y}(k+1)+[\boldsymbol{Q}_2^{[i]},\cdots,\boldsymbol{Q}_{n_a}^{[i]},0]\underset{\leftarrow k}{\boldsymbol{y}} \end{aligned} \tag{16}$$

$$\boldsymbol{P}^{[i]} \underset{\leftarrow k}{\Delta \boldsymbol{u}} = \left[\boldsymbol{P}^{[i]}, 0\right] \begin{bmatrix} \underset{\leftarrow k}{\Delta \boldsymbol{u}} \\ \underset{\leftarrow k-1}{\Delta \boldsymbol{u}} \end{bmatrix} = \\ = \boldsymbol{P}_1^{[i]} \Delta \boldsymbol{u}(k) + \left[\boldsymbol{P}_2^{[i]}, \cdots, \boldsymbol{P}_{n_b-1}^{[i]}, 0\right] \underset{\leftarrow k-1}{\Delta \boldsymbol{u}} \tag{17}$$

Prediction equation (15) then takes this form

$$\hat{\boldsymbol{y}}(k+i+1) = \left[\boldsymbol{P}_1^{[i]}, \boldsymbol{H}^{[i]}\right] \underset{\rightarrow k-1}{\Delta \boldsymbol{u}} + \left[\boldsymbol{P}_2^{[i]}, \cdots \boldsymbol{P}_{n_b-1}^{[i]}, 0\right] \underset{\leftarrow k-1}{\Delta \boldsymbol{u}} + \\ + \left[\boldsymbol{Q}_2^{[i]}, \cdots \boldsymbol{Q}_{n_a}^{[i]}, 0\right] \underset{\leftarrow k}{\boldsymbol{y}} + \boldsymbol{Q}_1^{[i]} \boldsymbol{y}(k+1) \tag{18}$$

After substitution of one step ahead prediction we obtain

$$\hat{\boldsymbol{y}}(k+i+1) = \left[\boldsymbol{P}_1^{[i]}, \boldsymbol{H}^{[i]}\right] \underset{\rightarrow k-1}{\Delta \boldsymbol{u}} + \left[\boldsymbol{P}_2^{[i]}, \cdots \boldsymbol{P}_{n_b-1}^{[i]}, 0\right] \underset{\leftarrow k-1}{\Delta \boldsymbol{u}} + \\ + \left[\boldsymbol{Q}_2^{[i]}, \cdots \boldsymbol{Q}_{n_a}^{[i]}, 0\right] \underset{\leftarrow k}{\boldsymbol{y}} + \boldsymbol{Q}_1^{[i]} \left[\boldsymbol{H}^{[1]} \underset{\rightarrow k-1}{\Delta \boldsymbol{u}} + \boldsymbol{P}^{[1]} \underset{\leftarrow k-1}{\Delta \boldsymbol{u}} + \boldsymbol{Q}^{[1]} \underset{\leftarrow k}{\boldsymbol{y}}\right] \tag{19}$$

Common terms can be grouped together

$$\hat{\boldsymbol{y}}(k+i+1) = \underbrace{\left[\boldsymbol{P}_1^{[i]} + \boldsymbol{Q}_1^{[i]} \boldsymbol{H}^{[1]}, \boldsymbol{H}^{[i]}\right]}_{\boldsymbol{H}^{[i+1]}} \underset{\rightarrow k-1}{\Delta \boldsymbol{u}} + \\ + \underbrace{\left\{\left[\boldsymbol{P}_2^{[i]}, \cdots \boldsymbol{P}_{n_b-1}^{[i]}, 0\right] + \boldsymbol{Q}_1^{[i]} \boldsymbol{P}^{[1]}\right\}}_{\boldsymbol{P}^{[i+1]}} \underset{\leftarrow k-1}{\Delta \boldsymbol{u}} + \\ + \underbrace{\left\{\left[\boldsymbol{Q}_2^{[i]}, \cdots \boldsymbol{Q}_{n_a}^{[i]}, 0\right] + \boldsymbol{Q}_1^{[i]} \boldsymbol{Q}^{[1]}\right\}}_{\boldsymbol{Q}^{[i+1]}} \underset{\leftarrow k}{\boldsymbol{y}} \tag{20}$$

The recursion is then performed according to the following expressions

$$\boldsymbol{H}^{[i+1]} = \left[\boldsymbol{P}_1^{[i]} + \boldsymbol{Q}_1^{[i]} \boldsymbol{H}^{[1]}, \boldsymbol{H}^{[i]}\right] \\ \boldsymbol{P}^{[i+1]} = \left[\boldsymbol{P}_2^{[i]}, \cdots \boldsymbol{P}_{n_b-1}^{[i]}, 0\right] + \boldsymbol{Q}_1^{[i]} \boldsymbol{P}^{[1]} \\ \boldsymbol{Q}^{[i+1]} = \left[\boldsymbol{Q}_2^{[i]}, \cdots \boldsymbol{Q}_{n_a}^{[i]}, 0\right] + \boldsymbol{Q}_1^{[i]} \boldsymbol{Q}^{[1]} \tag{21}$$

Initialization of the matrices $\boldsymbol{H}$, $\boldsymbol{P}$, $\boldsymbol{Q}$ for our TITO example is then following

$$\boldsymbol{H} = \begin{bmatrix} b_1 & b_3 \\ b_5 & b_7 \end{bmatrix} \quad \boldsymbol{P} = \begin{bmatrix} b_2 & b_4 \\ b_6 & b_8 \end{bmatrix}$$

$$\boldsymbol{Q} = \begin{bmatrix} 1-a_1 & -a_3 & a_1-a_2 & a_3-a_4 & a_2 & a_4 \\ -a_5 & 1-a_7 & a_5-a_6 & a_7-a_8 & a_6 & a_8 \end{bmatrix} \tag{22}$$

The recursion then proceeds according to the expressions (21).

3.2 Method Based on Diophantine Equations

It is possible to compute j-step ahead prediction from the model (10)

$$\hat{\boldsymbol{y}}(k+j) = \frac{\boldsymbol{B}}{\boldsymbol{A}} \boldsymbol{u}(k+j) + \frac{\boldsymbol{C}}{\Delta \boldsymbol{A}} \boldsymbol{e}_s(k+j) \tag{23}$$

From the last term of this expression can be separated terms with positive powers of z where $\boldsymbol{E}$ is a polynomial matrix of the order j minus one and $\boldsymbol{F}$ is a polynomial matrix of the same order as the polynomial matrix $\boldsymbol{A}$.

$$\frac{\boldsymbol{C}(z^{-1})}{\Delta \boldsymbol{A}(z^{-1})} = \boldsymbol{E}_j(z^{-1}) + z^{-j} \frac{\boldsymbol{F}_j(z^{-1})}{\Delta \boldsymbol{A}(z^{-1})} \tag{24}$$

After substitution to equation (23) we can obtain the predictor in the form

$$\hat{\boldsymbol{y}}(k+j) = \frac{\boldsymbol{B}}{\boldsymbol{A}} \boldsymbol{u}(k+j) + \boldsymbol{E}_j \boldsymbol{e}_s(k+j) + \frac{\boldsymbol{F}_j}{\Delta \boldsymbol{A}} \boldsymbol{e}_s(k) \tag{25}$$

From the original equation (23) we can compute the disturbance and substitute to equation (25)

$$\hat{\boldsymbol{y}}(k+j) = \frac{\boldsymbol{B}}{\boldsymbol{C}} \left[\frac{\boldsymbol{C}}{\Delta \boldsymbol{A}} - z^{-j} \frac{\boldsymbol{F}_j}{\Delta \boldsymbol{A}}\right] \Delta \boldsymbol{u}(k+j) + \frac{\boldsymbol{F}_j}{\boldsymbol{C}} \boldsymbol{y}(k) + \\ + \boldsymbol{E}_j \boldsymbol{e}_s(k+j) \tag{26}$$

After substitution we obtain

$$\hat{\boldsymbol{y}}(k+j) = \frac{\boldsymbol{B}\boldsymbol{E}_j}{\boldsymbol{C}} \Delta \boldsymbol{u}(k+j) + \frac{\boldsymbol{F}_j}{\boldsymbol{C}} \boldsymbol{y}(k) + \boldsymbol{E}_j \boldsymbol{e}_s(k+j) \tag{27}$$

Now let us make two simplifications: a white noise case will be considered and future noise values will be further omitted.

$$\hat{\boldsymbol{y}}(k+j) = \boldsymbol{B}\boldsymbol{E}_j \Delta \boldsymbol{u}(k+j) + \boldsymbol{F}_j \boldsymbol{y}(k) \tag{28}$$

The matrix $\boldsymbol{G}$ is defined as follows

$$\boldsymbol{G}_j = \boldsymbol{B}\boldsymbol{E}_j \quad \hat{\boldsymbol{y}}(k+j) = \boldsymbol{G}_j \Delta \boldsymbol{u}(k+j) + \boldsymbol{F}_j \boldsymbol{y}(k) \tag{29}$$

For the design of the j – step ahead predictor the following Diophantine equation is solved

$$\boldsymbol{I} = \boldsymbol{E}_j \Delta \boldsymbol{A} + z^{-j} \boldsymbol{F}_j \tag{30}$$

Further it is necessary to solve a recursion of the Diophantine equation (30). Particular matrices in the Diophantine equation can be expanded as follows

$$\tilde{\boldsymbol{A}}(z^{-1}) = \Delta \boldsymbol{A}(z^{-1}) = \boldsymbol{I} + \boldsymbol{A}_1 z^{-1} + \boldsymbol{A}_2 z^{-2} + \cdots + \\ + \boldsymbol{A}_{n_a} z^{-n_a} + \boldsymbol{A}_{n_a+1} z^{-(n_a+1)} \tag{31}$$

$$\boldsymbol{E}_j(z^{-1}) = E_{j,0} + E_{j,1} z^{-1} + E_{j,2} z^{-2} + \cdots + E_{j,j-1} z^{j-1} \tag{32}$$

$$\boldsymbol{F}_j(z^{-1}) = F_{j,0} + F_{j,1} z^{-1} + F_{j,2} z^{-2} + \cdots + F_{j,j-1} z^{j-1} \tag{33}$$

Let us consider the Diophantine equation corresponding to prediction $\hat{\boldsymbol{y}}(k+j+1)$

$$\boldsymbol{I} = \boldsymbol{E}_{j+1}(z^{-1}) \tilde{\boldsymbol{A}}(z^{-1}) + z^{-(j+1)} \boldsymbol{F}_{j+1}(z^{-1}) \tag{34}$$

It is possible to subtract the Diophantine equation (30) from the Diophantine equation (34) and obtain the following expression

$$0 = \left(\boldsymbol{E}_{j+1}(z^{-1}) - \boldsymbol{E}_j(z^{-1})\right) \tilde{\boldsymbol{A}}(z^{-1}) + \\ + z^{-j}\left(z^{-1} \boldsymbol{F}_{j+1}(z^{-1}) - \boldsymbol{F}_j(z^{-1})\right) \tag{35}$$

Now it is possible to define the following term

$$\boldsymbol{E}_{j+1}(z^{-1}) - \boldsymbol{E}_j(z^{-1}) = \tilde{\boldsymbol{R}}(z^{-1}) + R_j z^{-1} \tag{36}$$

After substitution

$$0 = \tilde{\boldsymbol{R}}(z^{-1}) \tilde{\boldsymbol{A}}(z^{-1}) + z^{-j}\left(R_j \tilde{\boldsymbol{A}}(z^{-1}) + z^{-1} \boldsymbol{F}_{j+1}(z^{-1}) - \boldsymbol{F}_j(z^{-1})\right) \tag{37}$$

it is obvious that $\tilde{R}(z^{-1}) = 0$ in order to obtain the zero matrix on the left side of the equation (37). The matrix $\boldsymbol{E}$ can be then computed recursively according to the following expression

$$\boldsymbol{E}_{j+1}(z^{-1}) = \boldsymbol{E}_j(z^{-1}) + R_j z^{-j} \tag{38}$$

Following expressions can be obtained from the equation (37)

$$R_j = F_{j,0} \\ F_{j+1,i} = F_{j,i+1} - R_j A_{i+1} \quad \text{for} \quad \mathrm{i} = 0 \cdots \delta(\boldsymbol{F}_{j+1}) \tag{39}$$

Initial conditions for the recursion are as follows

$$\begin{aligned} \boldsymbol{E}_1 &= \boldsymbol{I} \\ \boldsymbol{F}_1 &= z\left(I - \widetilde{\boldsymbol{A}}\right) \end{aligned} \tag{40}$$

The final prediction equation has the following form

$$\hat{\boldsymbol{y}}(k+j) = \boldsymbol{G}_j \Delta \boldsymbol{u}(k+j) + \boldsymbol{F}_j \boldsymbol{y}(k) \tag{41}$$

In our TITO example, the matrix $\widetilde{\boldsymbol{A}}$ has the following form

$$\begin{aligned} &\widetilde{\boldsymbol{A}}(z^{-1}) = \boldsymbol{\Delta A}(z^{-1}) = \\ &= \begin{bmatrix} 1 + (a_1 - 1)z^{-1} + (a_2 - a_1)z^{-2} - a_2 z^{-3} & a_3 z^{-1} + (a_4 - a_3)z^{-2} - a_4 z^{-3} \\ a_5 z^{-1} + (a_6 - a_5)z^{-2} - a_6 z^{-3} & 1 + (a_7 - 1)z^{-1} + (a_8 - a_7)z^{-2} - a_8 z^{-3} \end{bmatrix} \end{aligned} \tag{42}$$

Initial conditions of the recursion are

$$\boldsymbol{E}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{43}$$

$$\boldsymbol{F}_1 = z\left(I - \widetilde{\boldsymbol{A}}\right) = \begin{bmatrix} \boldsymbol{f}_0 & \boldsymbol{f}_1 & \boldsymbol{f}_2 \end{bmatrix} \tag{44}$$

$$\boldsymbol{f}_0 = \begin{bmatrix} 1 - a_1 & -a_3 \\ -a_5 & 1 - a_7 \end{bmatrix} \tag{45}$$

$$\boldsymbol{f}_1 = \begin{bmatrix} a_1 - a_2 & a_3 - a_4 \\ a_5 - a_6 & a_7 - a_8 \end{bmatrix} \tag{46}$$

$$\boldsymbol{f}_2 = \begin{bmatrix} a_2 & a_4 \\ a_6 & a_8 \end{bmatrix} \tag{47}$$

Initialization of the matrix of the free response and the matrix of the dynamics are following

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{f}_0 & \boldsymbol{f}_1 & \boldsymbol{f}_2 & \boldsymbol{B}_2 \end{bmatrix} \tag{48}$$

$$\boldsymbol{G} = \boldsymbol{B}_1 \tag{49}$$

where

$$\boldsymbol{B}_1 = \begin{bmatrix} b_1 & b_3 \\ b_5 & b_7 \end{bmatrix} \qquad \boldsymbol{B}_2 = \begin{bmatrix} b_2 & b_4 \\ b_6 & b_8 \end{bmatrix} \tag{50}$$

The recursion then proceeds according to previously introduced steps.

$$\boldsymbol{R} = \boldsymbol{f}_0 \tag{51}$$

$$\boldsymbol{f}_0 = \boldsymbol{f}_1 - \boldsymbol{R}(a_1 - 1) \tag{52}$$

$$\boldsymbol{f}_1 = \boldsymbol{f}_2 - \boldsymbol{R}(a_2 - a_1) \qquad \boldsymbol{f}_2 = \boldsymbol{R} a_2 \tag{53}$$

$$\boldsymbol{E} = \begin{bmatrix} \boldsymbol{E} & \boldsymbol{R} \end{bmatrix} \tag{54}$$

Extension of the matrices of the dynamics and the free response is as follows:

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G} \\ \boldsymbol{B}_1 \boldsymbol{E}(i+1) + \boldsymbol{B}_2 \boldsymbol{E}(i) \end{bmatrix} \tag{55}$$

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{f}_0 \quad \boldsymbol{f}_1 \quad \boldsymbol{f}_2 \quad \boldsymbol{B}_2 \boldsymbol{E}(i+1) \end{bmatrix} \tag{56}$$

3.3 Method Based on Direct Computation from CARIMA Model

This method is based on an analytical derivation of certain predictions and subsequent recursive derivation of later predictions. The number of predictions which are necessary to compute directly depends on the order of the system. The a priori analytical computation, which is required, enables to reduce number of matrix operations which are necessary to perform during the matrix methods.

The differential equations (11) can be rewritten into the matrix form (57)

$$\begin{aligned} \boldsymbol{y}(k+1) &= \boldsymbol{A}_1 \boldsymbol{y}(k) + \boldsymbol{A}_2 \boldsymbol{y}(k-1) + \boldsymbol{A}_3 \boldsymbol{y}(k-2) + \\ &+ \boldsymbol{B}_1 \Delta \boldsymbol{u}(k) + \boldsymbol{B}_2 \Delta \boldsymbol{u}(k-1) \end{aligned} \tag{57}$$

where

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 - a_1 & -a_3 \\ -a_5 & 1 - a_7 \end{bmatrix} \boldsymbol{A}_2 = \begin{bmatrix} a_1 - a_2 & a_3 - a_4 \\ a_5 - a_6 & a_7 - a_8 \end{bmatrix}$$

$$\boldsymbol{A}_3 = \begin{bmatrix} a_2 & a_4 \\ a_6 & a_8 \end{bmatrix} \tag{58}$$

$$\boldsymbol{B}_1 = \begin{bmatrix} b_1 & b_3 \\ b_5 & b_7 \end{bmatrix} \boldsymbol{B}_2 = \begin{bmatrix} b_2 & b_4 \\ b_6 & b_8 \end{bmatrix}$$

It was necessary to directly compute three step ahead predictions in a straightforward way by establishing of previous predictions to later predictions. The model order defines that computation of one step ahead prediction is based on three past values of the system output.

$$\begin{aligned} \hat{\boldsymbol{y}}(k+1) &= \boldsymbol{A}_1 \boldsymbol{y}(k) + \boldsymbol{A}_2 \boldsymbol{y}(k-1) + \boldsymbol{A}_3 \boldsymbol{y}(k-2) + \\ &+ \boldsymbol{B}_1 \Delta \boldsymbol{u}(k) + \boldsymbol{B}_2 \Delta \boldsymbol{u}(k-1) \\ \hat{\boldsymbol{y}}(k+2) &= \boldsymbol{A}_1 \boldsymbol{y}(k+1) + \boldsymbol{A}_2 \boldsymbol{y}(k) + \boldsymbol{A}_3 \boldsymbol{y}(k-1) + \\ &+ \boldsymbol{B}_1 \Delta \boldsymbol{u}(k+1) + \boldsymbol{B}_2 \Delta \boldsymbol{u}(k) \\ \hat{\boldsymbol{y}}(k+3) &= \boldsymbol{A}_1 \boldsymbol{y}(k+2) + \boldsymbol{A}_2 \boldsymbol{y}(k+1) + \boldsymbol{A}_3 \boldsymbol{y}(k) + \\ &+ \boldsymbol{B}_1 \Delta \boldsymbol{u}(k+2) + \boldsymbol{B}_2 \Delta \boldsymbol{u}(k+1) \end{aligned} \tag{59}$$

It is possible to divide computation of the predictions to recursion of the free response and recursion of the matrix of the dynamics. The free response vector can be expressed as:

$$\boldsymbol{y}_0 = \left[\begin{array}{cc|cc|cc|cc} p(1,1) & p(1,2) & p(1,3) & p(1,4) & p(1,5) & p(1,6) & p(1,7) & p(1,8) \\ p(2,1) & p(2,2) & p(2,3) & p(2,4) & p(2,5) & p(2,6) & p(2,7) & p(2,8) \\ \hline p(3,1) & p(3,2) & p(3,3) & p(3,4) & p(3,5) & p(3,6) & p(3,7) & p(3,8) \\ p(4,1) & p(4,2) & p(4,3) & p(4,4) & p(4,5) & p(4,6) & p(4,7) & p(4,8) \\ \hline p(5,1) & p(5,2) & p(5,3) & p(5,4) & p(5,5) & p(5,6) & p(5,7) & p(5,8) \\ p(6,1) & p(6,2) & p(6,3) & p(6,4) & p(6,5) & p(6,6) & p(6,7) & p(6,8) \end{array}\right] \left[\begin{array}{c} y_1(k) \\ y_2(k) \\ \hline y_1(k-1) \\ y_2(k-1) \\ \hline y_1(k-2) \\ y_2(k-2) \\ \hline \Delta u_1(k-1) \\ \Delta u_2(k-1) \end{array}\right] =$$

$$= \begin{bmatrix} \boldsymbol{P}(1,1) & \boldsymbol{P}(1,2) & \boldsymbol{P}(1,3) & \boldsymbol{P}(1,4) \\ \boldsymbol{P}(2,1) & \boldsymbol{P}(2,2) & \boldsymbol{P}(2,3) & \boldsymbol{P}(2,4) \\ \boldsymbol{P}(3,1) & \boldsymbol{P}(3,2) & \boldsymbol{P}(3,3) & \boldsymbol{P}(3,4) \end{bmatrix} \begin{bmatrix} \boldsymbol{y}(k) \\ \boldsymbol{y}(k-1) \\ \boldsymbol{y}(k-2) \\ \Delta \boldsymbol{u}(k-1) \end{bmatrix} = \boldsymbol{P} \begin{bmatrix} \boldsymbol{y}(k) \\ \boldsymbol{y}(k-1) \\ \boldsymbol{y}(k-2) \\ \Delta \boldsymbol{u}(k-1) \end{bmatrix} \tag{60}$$

All the elements $P(i,j)$ i=1…3, j=1…4 have to be directly computed to initialize the recursion. The next row of the matrix $\boldsymbol{P}$ is repeatedly computed on the basis of the three previous predictions until the prediction horizon is achieved. As an illustrative example it is given the computation of the next element of the first column:

$$\boldsymbol{P}(4,1) = \begin{bmatrix} p(7,1) & p(7,2) \\ p(8,1) & p(8,2) \end{bmatrix} = \boldsymbol{A}_1 \boldsymbol{P}(3,1) + \boldsymbol{A}_2 \boldsymbol{P}(2,1) + \boldsymbol{A}_3 \boldsymbol{P}(1,1) \tag{61}$$

The recursion of the matrix $\boldsymbol{G}$ is similar. The next element of the first column is repeatedly computed and

the remaining columns are shifted. This procedure is performed repeatedly until the prediction horizon is achieved. If the control horizon is lower than the prediction horizon a number of columns in the matrix is reduced. The technique is apparent from the equations (62) and (63).

$$\boldsymbol{G}\Delta\boldsymbol{u} = \left[\begin{array}{cc|cc} g(1,1) & g(1,2) & 0 & 0 \\ g(2,1) & g(2,2) & 0 & 0 \\ \hline g(3,1) & g(3,2) & g(1,1) & g(1,2) \\ g(4,1) & g(4,2) & g(2,1) & g(2,2) \\ \hline g(5,1) & g(5,2) & g(3,1) & g(3,2) \\ g(6,1) & g(6,2) & g(4,1) & g(4,2) \end{array}\right] \left[\begin{array}{c} \Delta u_1(k) \\ \Delta u_2(k) \\ \hline \Delta u_1(k+1) \\ \Delta u_2(k+1) \end{array}\right] = \tag{62}$$

$$= \begin{bmatrix} G(1,1) & 0 \\ G(2,1) & G(1,1) \\ G(3,1) & G(2,1) \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{u}(k) \\ \Delta\boldsymbol{u}(k+1) \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{G}(4,1) &= \begin{bmatrix} g(7,1) & g(7,2) \\ g(8,1) & g(8,2) \end{bmatrix} = \\ &= \boldsymbol{A}_1\boldsymbol{G}(3,1) + \boldsymbol{A}_2\boldsymbol{G}(2,1) + \boldsymbol{A}_3\boldsymbol{G}(1,1) = \\ &= \begin{bmatrix} 1-a_1 & -a_3 \\ -a_5 & 1-a_7 \end{bmatrix} \begin{bmatrix} g(5,1) & g(5,2) \\ g(6,1) & g(6,2) \end{bmatrix} + \\ &+ \begin{bmatrix} a_1-a_2 & a_3-a_4 \\ a_5-a_6 & a_7-a_8 \end{bmatrix} \begin{bmatrix} g(3,1) & g(3,2) \\ g(4,1) & g(4,2) \end{bmatrix} + \\ &+ \begin{bmatrix} 1-a_1 & -a_3 \\ -a_5 & 1-a_7 \end{bmatrix} \begin{bmatrix} g(1,1) & g(1,2) \\ g(2,1) & g(2,2) \end{bmatrix} \end{aligned} \tag{63}$$

4 Simulation Results

Predictions of the following systems behaviour is given here as an example.

$$\boldsymbol{A}(z^{-1}) = \begin{bmatrix} 1-0.5827z^{-1}+0.1745z^{-2} & -0.0220z^{-1}+0.1797z^{-2} \\ 0.0167z^{-1}-0.0886z^{-2} & 1-0.4564z^{-1}-0.0830z^{-2} \end{bmatrix}$$

$$\boldsymbol{B}(z^{-1}) = \begin{bmatrix} -0.0035z^{-1}+0.0955z^{-2} & 0.1484z^{-1}+0.2197z^{-2} \\ 0.2783z^{-1}+0.3107z^{-2} & -0.0371z^{-1}-0.3489z^{-2} \end{bmatrix} \tag{64}$$

Fig. 1 shows the plant‘s step response

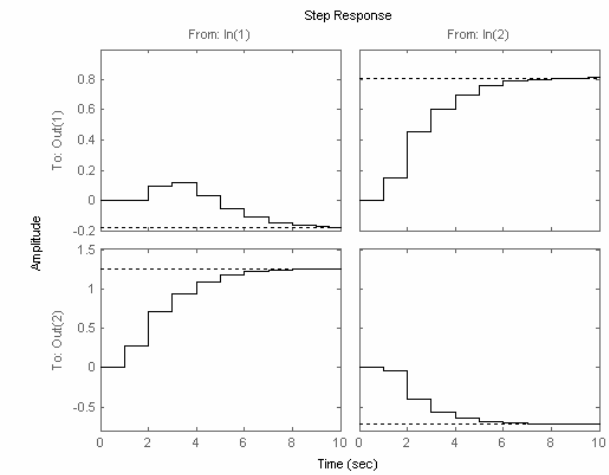

Fig. 1 Step response of the plant

As the input of the system it was chosen a random signal with zero mean value. Values of the signal were generated a priori. Predictions are computed so that such a number of outputs is predicted which corresponds to the value of the prediction horizon. Only the first predicted value is taken and the procedure is repeated.

Results obtained for particular methods were compared each other. In all cases were obtained identical courses of systems output predictions. It means that each different method makes the same final prediction equations. The results were also compared to results of simulation, which means simple input – output response. Courses of the outputs were also identical and correctness of all methods was proved. In the following figures are results of simulation and prediction (only one figure for prediction is presented because as it was mentioned the results were identical). Both prediction and control horizons were set to 10.

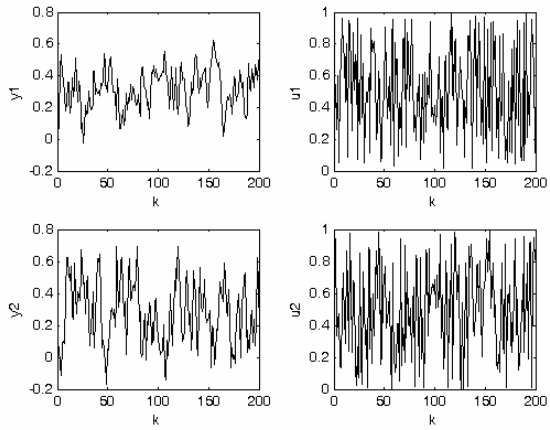

Fig. 2 Prediction

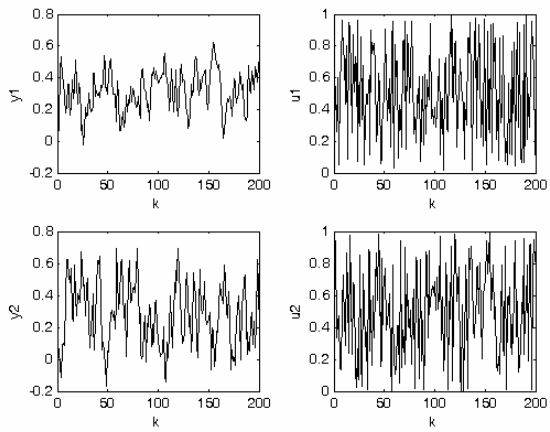

Fig. 3 Simulation

All three methods resulted to the same prediction equations. But computational demands for particular methods are different. Computational time demands on CPU of personal computer for particular methods were evaluated. Algorithms for recursive computation of matrices of coefficients, where particular methods differ,

were considered. The sequences of commands were tested using Matlab. During these experiments only the simulation programs for particular methods run on the computer. Measurement for each method was performed thirty times and the results were averaged. Individual time periods of computation for particular methods did not differ significantly so that the results can be considered as authentic. Results of the time measurements are in the following table

Table. 1 Time periods of computation

Method based on matrix operations	Method based on Diophantine equations	Direct computation from CARIMA model
0,0109 s	0,0154 s	0,0007 s

5 Conclusion

Three methods of computation of multivariable systems output prediction were algorithmically realized. Correctness of all methods was verified by simulation. Simulation results proved applicability of the methods. The methods were evaluated from the point of view of computational time demands. From this evaluation results that the method based on direct computation of predictions from CARIMA model is significantly faster than the remaining two methods, where comparable results were achieved. If the methods are applied for predictive controllers based on models with fixed parameters then the computation of predictions is necessary to perform only once. In this case saving of computational time does not have significant importance. But in case that the predictive controller is realized as an adaptive predictive controller the computation of predictions is necessary to perform in each sampling period. Saving of computational time then has particular significance. It can also be important while using the prediction for other purposes than for the predictive control.

From the algorithmic point of view the method based on Diophantine equations seems to be less understandable and clear. The method of computation of predictions is not as straightforward as the remaining two methods, where computation is quite clear. A disadvantage of direct computation from CARIMA model is necessity of direct analytical computation of a certain number of predictions.

Acknowledgement

The authors wish to thank to the Ministry of Education, Youth and Sports of the Czech Republic (MSM7088352101) for financial support.

References:

[1] E. F. Camacho, C. Bordons, *Model Predictive Control,* Springer-Verlag, London, 2004.

[2] M. Morari, J. H. Lee, Model predictive control: past, present and future. *Computers and Chemical Engineering,* 23, 1999, 667-682.

[3] R. R. Bitmead, M. Gevers, V. Hertz, *Adaptive Optimal Control. The Thinking Man's GPC,* Prentice Hall, Englewood Cliffs, New Jersey, 1990.

[4] D. W. Clarke, C. Mohtadi, P. S. Tuffs, Generalized predictive control, part I: the basic algorithm. *Automatica*, 23, 1987, 137-148.

[5] D. W. Clarke, C. Mohtadi, P. S. Tuffs, Generalized predictive control, part II: extensions and interpretations. *Automatica*, 23, 1987, 149-160.

[6] I. D. Landau, R. Lozano, M. M'Saad, *Adaptive Control,* Springer - Verlag, Berlin, 1998.

[7] V. Bobal, J., Bőhm , J., Fessl, J., Machacek, *Digital Self-Tuning Controllers*, Springer - Verlag, London, 2005.

[8] F. G. Shinskey, *Process Control System,* 4th ed.; McGraw-Hill. New York, 1996.

[9] M. Kubalčík, V. Bobál, Adaptive Control of Coupled Drives Apparatus Based on Polynomial Theory, *IMechE Part I: J. Systems and Control Engineering*, 220(I7), 2006, 641-654.

[10] J. A. Rossiter, *Model Based Predictive Control: a Practical Approach,* CRC Press, 2003.

[11] M. Kubalčík, V. Bobál, Adaptive Predictive Control Applied to Coupled Drives Process. *Proc. 28th IASTED International Conference on Modeling, Identification and Control MIC 2009*, Innsbruck, Austria, 2009, 331-336.