MPC Control Using AR-Volterra Models

P. Taraba, Š. Kozák

***Abstract*: This paper deals with the model predictive control (MPC) with constraints. Each MPC application is based on a model – here, it is based on a Volterra model. Volterra model drawback is a lot of parameters, but making Volterra model of lower order means less parameters. Optimum is to find such model, which suits to process and has as low parameters as possible. This model predictive control is looking for optimal control value by using optimization techniques. It is examined on continuous mixed reactor.**

***Keywords*: MPC – model predictive control, optimization, Volterra models**

I. INTRODUCTION

Model predictive control is well studied in the linear theory. Problems arise, when we need to control nonlinear processes. There are several ways how to solve this problem, whereby each one starts with finding a model. One possible way is to design a Volterra model of a process. Then MPC problem can be be solved by a lot of approaches [1], one of them is to reduce problem to a simple optimization task. Problems with constraints were solved by adding some penalization function to cost function. This theory was examined on a model of a chemical continuous mixed reactor described in Section 2. Main theoretical results on control and identification of Volterra models are presented in Section 3. Optimization problems with constraints are solved in Section 4 and conclusions are given in Section 5.

II. CONTINUOUS MIXED REACTOR

This model can be described by three equations, the first two describe energy balance and the third describes material balance:

$$\rho_2 V_2 c_{p2}\frac{dT_2}{dt} = M_2 c_{p2} T_{2i} - M_2 c_{p2} T2 + \lambda S(T_1 - T_2) + V_2(-\Delta Hr)r \quad \text{... (1)}$$

$$\rho_1 V_1 c_{p1}\frac{dT_1}{dt} = M_1 c_{p1} T_{1i} - M_1 c_{p1} T_1 - \lambda S(T_1 - T_2) \quad \text{... (2)}$$

$$V_2\frac{dc_a}{dt} = \frac{M_2}{\rho_2}(c_{ai} - c_a) - V_2 r \quad \text{... (3)}$$

where

$$r = r_A = c_a k_0 \exp(-\frac{a}{T_2 + 273.15}) \quad \text{... (4)}$$

Department of Automatic Control, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Ilkovičova 3, 81219 Bratislava, Slovak Republic, Email: {taraba,kozak}@kasr.elf.stuba.sk

The system output c_a is controlled by input M_1. Input M_1 should be in the range $<0kg/s;0.3kg/s>$. Constants in equations (1), (2), (3), (4) are the following:

$k_0 = 1.01e12s^{-1}; a = 10067.36; \Delta H_r = -250800J/mol;$

$c_p = c_{p1} = c_{p2} = 4187J/kg/K; \rho = \rho_1 = \rho_2 = 1000kg/m^3;$

$V_2 = 0.05m^3; V_1 = 0.02m^3; \lambda = 800J/m^2/K/s; S = 1.25m^2;$

$M_2 = 0.1kg/s; T_{20} = 20^\circ C; T_{2i} = 20^\circ C; T_{1i} = 20^\circ C;$

$c_{ai} = 500mol/m^3; c_{ai0} = 0mol/m^3;$

III. MODEL PREDICTIVE CONTROL

A. *Modeling with AR-Volterra Models*

Volterra models are used to model nonlinear processes. The problem is that with increasing level of nonlinearity N (see equations (6), (7)) the number of parameters increases sharply. That's why we have used only Volterra model with N = 2. This kind of model can compensate processes with a quadratic transmission characteristic

$$y(k+j) = y_0 + \sum_{i=1}^{M1}\alpha_i y(k+j-i) + \sum_{i=1}^{M2}\beta_i u(k+j-i) + \sum_{i=1}^{N} v_{M2}^i(k+j) \quad \text{...(6)}$$

$$v_{M2}^i(k+j) = \sum_{j_1=1}^{M2}\ldots\sum_{j_i=1}^{j_{i-1}}\gamma(j_1,\ldots,j_i)u(k+j-j_1)..u(k+j-j_i) \quad \ldots (7)$$

M1 denotes the size of history dependence of new output on previous outputs, M2 is the size of history dependence of the new output on previous inputs. For N = 2 the number of parameters is as follows:

$$\begin{aligned} &y_0 - 1 \\ &\alpha_i - M1 \\ &\beta_i - M2 \\ &\gamma - \frac{M2(M2+1)}{2} \end{aligned} \quad \ldots (8)$$

$$\#\,parameters = 1 + M1 + M2 + \frac{M2(M2+1)}{2}$$

If we have steady-state input and output, we can denote input as constant U, and output as constant Y. Then for N = 2 , equations (6), (7) change to:

$$Y = y_0 + Y\sum_{i=1}^{M1}\alpha_i + U\sum_{i=1}^{M2}\beta_i + U^2\sum_{i=1}^{M2}\sum_{j=1}^{i}\gamma(i,j) \quad \ldots (9)$$

From equation (9) we can get:

$$Y = \varphi(U) = \varphi_0 + \varphi_1 U + \varphi_2 U^2$$

$$\varphi_0 = \frac{y_0}{1-\sum_{i=1}^{M1}\alpha_i}, \varphi_1 = \frac{\sum_{i=1}^{M2}\beta_i}{1-\sum_{i=1}^{M1}\alpha_i}, \varphi_2 = \frac{\sum_{i=1}^{M2}\sum_{j=1}^{i}\gamma(i,j)}{1-\sum_{i=1}^{M1}\alpha_i} \quad \text{... (10)}$$

This is the reason, why with this AR-Volterra model we can handle only processes with quadratic transmission characteristic.

B. *Identification of AR-Volterra Parameters*

Identification of AR-Volterra parameters has been performed using the least square method. Equation (1) can be transformed to

$$\begin{bmatrix} y_{k+1} \\ y_{k+2} \\ \dots \\ y_{k+M} \end{bmatrix} = H\theta \qquad \dots (11)$$

where

$$H = \begin{bmatrix} 1 & y_k & \dots & y_{k+1-M1} & u_k & \dots & u_{k+1-M2} & u_k u_k & \dots & u_{k+1-M2}u_{k+1-M2} \\ 1 & y_{k+1} & & y_{k+2-M1} & u_{k+1} & & u_{k+2-M2} & u_{k+1}u_{k+1} & & u_{k+2-M2}u_{k+2-M2} \\ \dots & \dots & & \dots & & & \dots & \dots & & \dots \\ 1 & y_{k+M-1} & & y_{k+M-M1} & u_{k+M-1} & & u_{k+M-M2} & u_{k+M-1}u_{k+M-1} & & u_{k+M-M2}u_{k+M-M2} \end{bmatrix}$$

and

$$\theta = \begin{bmatrix} y_0 \\ \alpha_1 \\ \dots \\ \alpha_{M1} \\ \beta_1 \\ \dots \\ \beta_{M2} \\ \gamma(1,1) \\ \dots \\ \gamma(M2, M2) \end{bmatrix}$$

Matrices Y and θ are known from observation of process output responses to input sequence u. Due to noise we can add the vector E to the Eq. (11)

$$Y = H\theta + E \qquad \dots (12)$$

The aim of the identification is to find optimal solution θ for the cost function J

$$J = \frac{1}{2}E^T E = \frac{1}{2}(Y - H\theta)^T (Y - H\theta) \qquad \dots (13)$$

The optimal solution θ^* has to satisfy

$$\frac{\partial J}{\partial \theta} = H^T Y - H^T H\theta^* = 0 \qquad \dots (14)$$

Thus optimal parameters in the vector θ are computed as follows

$$\theta^* = (H^T H)^{-1} H^T Y \qquad \dots (15)$$

1) *Identification of chemical reactor*

Volterra model have been checked in two ways. First, the transmission characteristic was checked, as the coefficients $\varphi_0, \varphi_1, \varphi_2$ in Eq. (10) are known, we are able to compare both transfer functions. We have found history dependence $M1 = 3, M2 = 3$ in Eqs. (6), (7) sufficient and we chose the time period $T = 200s$. The dash-dotted line in Fig. 3-1 displays transmission characteristic of the reactor and the solid one the transfer function of Volterra model.

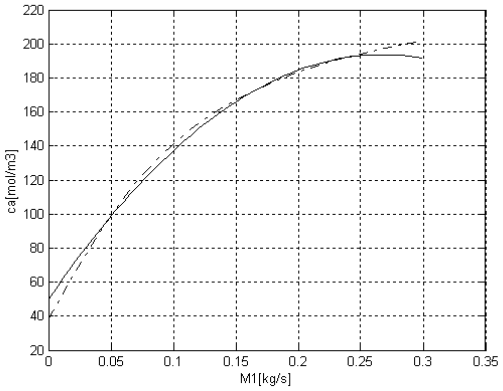

Figure III-1 Comparison of transmission characteristic of Volterra model and model of reactor

To check the dynamic behavior we compared responses of the reactor model and the Volterra model with same inputs. The dash-dotted line depicts the reactor response of and the solid one the Volterra model response to the like input.

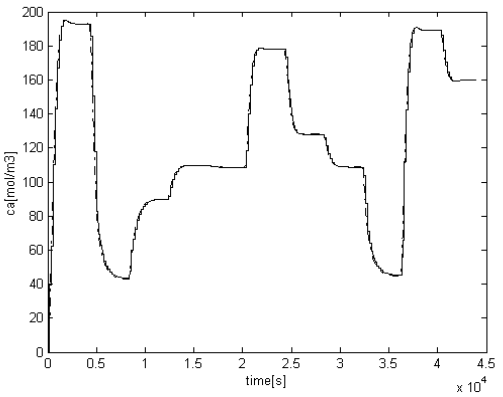

Figure III-2 Comparison of responses of the reactor and the Volterra model to the same input

As the dash-dotted lines in Figures 3-1 and 3-2 are almost matching the solid ones, we conclude that the Volterra model with the chosen number of parameters M1, M2 is adequate for the process of reactor.

C. *Model predictive control*

The model predictive control is an optimization problem of finding such an input vector U that minimizes the cost function J. P is the control horizon. (The control horizon and prediction horizon has been chosen of the same size).

$$J = \frac{1}{2}\sum_{i=1}^{P}(y(k+i) - w(k+i))^2 + \frac{1}{2}\gamma\sum_{i=1}^{P}(u(k-1+i) - u(k-2+i))^2 \qquad \dots (16)$$

where the vector U is

$$U = \begin{bmatrix} u(k) \\ \dots \\ u(k+p-1) \end{bmatrix} \qquad \dots (17)$$

Constant γ compensates for different magnitudes of the output y and input u(γ_1) and also indicates what is more

important for our controller, whether the speed of the output y reaching the setpoint w, or the size of control steps Δu (γ_2). This way we can factorize γ as

$$\gamma = \gamma_1 \gamma_2 = (\frac{\Delta y}{\Delta u})^2 \gamma_2 \quad \text{... (18)}$$

$\gamma_1 = const$ is square of fraction of adequate change of response of process to change of input. If $\gamma_2 = 1$, the weight of y reaching w and size of control steps are the same, if $\gamma_2 < 1$ then y reaches w earlier and if $\gamma_2 > 1$ then y reaches w slower, however this often helps to have smaller overshoot of the controlled variable y with respect to the setpoint w.

For finding the optimal vector U^* we will use the gradient method with a changeable step. Consider vector of setpoints (from Eq. (16))

$$W = \begin{bmatrix} w(k+1) \\ ... \\ w(k+p) \end{bmatrix} \quad \text{... (19)}$$

Then for every setpoint w, there exists such a control u, which is a solution to Eq. (10). Thus the starting vector U will be (there are two solutions, however just one u is from the applicable control interval):

$$U_0 = \begin{bmatrix} \varphi^{-1}(w(k+1)) \\ ... \\ \varphi^{-1}(w(k+p)) \end{bmatrix} \quad \text{... (20)}$$

As mentioned before, to find the optimal U we are using a gradient method. The following approximation by Taylor expansion has been used

$$J(U_{k+1}) \doteq f(U_k) + (U_{k+1} - U_k)^T g(U_k) + 0.5(U_{k+1} - U_k)^T H(U_k)(U_{k+1} - U_k) \quad \text{... (21)}$$

where g is the gradient:

$$g(U_k) = \left[\begin{matrix} \frac{\partial J}{\partial u(k)} \\ ... \\ \frac{\partial J}{\partial u(k-1+p)} \end{matrix} \right]_{U=U_k} \quad \text{... (22)}$$

and H is the Hesse matrix:

$$H(U_k) = \left[\begin{matrix} \frac{\partial^2 J}{\partial u(k)\partial u(k)} & ... & \frac{\partial^2 J}{\partial u(k)\partial u(k-1+p)} \\ ... & ... & \\ \frac{\partial^2 J}{\partial u(k-1+p)\partial u(k)} & & \frac{\partial^2 J}{\partial u(k-1+p)\partial u(k-1+p)} \end{matrix} \right]_{U=U_K} \quad \text{... (23)}$$

Due to using the gradient method, a new approximation will be:

$$U_{k+1} = U_k - \gamma_k g(U_k) \Rightarrow U_{k+1} - U_k = -\gamma_k g(U_k) \quad \text{... (24)}$$

Substituting (22) in (21) we obtain

$$J(U_{k+1}) \doteq f(U_k) - \gamma_k g(U_k)^T g(U_k) + 0.5\gamma_k^2 g(U_k)^T H(U_k) g(U_k) = \phi(\gamma_k) \quad \text{... (25)}$$

Partial derivative of (25) has to be zero, because we are looking for optimal step size, i.e.

$$\frac{\partial J}{\partial \gamma_k} = -g(U_k)^T g(U_k) + \gamma_k g(U_k)^T H(U_k) g(U_k) = 0 \Rightarrow \gamma_k = \frac{g(U_k)^T g(U_k)}{g(U_k)^T H(U_k) g(U_k)} \quad \text{... (26)}$$

A first problem is how to find the vector g:

$$g_i = \frac{\partial J}{\partial u(k-1+i)} = \sum_{j=1}^{P} (y(k+j) - w(k+j)) \frac{\partial y(k+j)}{\partial u(k-1+i)} + F_i \quad \text{... (27)}$$

where $i \in \{1,2,...,P\}$

$F_i = \gamma(u(k-1+i) - u(k-2+i)) - \gamma(u(k+i) - u(k-1+i))$ for $i \in \{1,2,...,P-1\}$ and $F_P = \gamma(u(k-1+P) - u(k-2+P))$.

To find all g_i we need to know $\frac{\partial y(k+j)}{\partial u(k-1+i)}$ for $\forall i, j \in \{1,2,...,P\}$.

For this we will construct a matrix J with $J_{j,i} = \frac{\partial y(k+j)}{\partial u(k-1+i)}$.

Now, dependence on previous y in (6) will be eliminated

$$y(k+j) = \psi_{k+j} = y_0 + \sum_{m=1}^{M2} \beta_m u(k+j-m) + \sum_{m=1}^{M2} \sum_{n=1}^{m} \gamma(m,n) u(k+j-m) u(k+j-n) \quad \text{... (28)}$$

Thus $J_{j,i} = \beta_{j-i+1} + \sum_{n=1}^{j-i+1} \gamma(j-i+1,n) u(k+j-n) + \sum_{m=j-i+1}^{M2} \gamma(m, j-i+1) u(k+j-m)$,

where of course if $i < j$ or $i \notin \{1,2,...,M2\}$ or $j \notin \{1,2,...,M2\}$ then $\gamma(i,j) = 0$.

Now we will add back dependence on previous outputs to (28)

$$y(k+j) = \sum_{m=1}^{M1} \alpha_m y(k+j-m) + \psi_{k+j} \quad \text{... (29)}$$

thus the j[th] line of J depends on the lines j-1,..., j-M1 (if they exist). So we start on the second line and continue up to last line. The algorithm works as follows

for line=2 to P
  for k=1 to M1
    if (line-k>=1)
      $J_{line,ALL} = J_{line,ALL} + \alpha_k J_{line-k,ALL}$
    endif
  next k
next line

Starting from the lowest line and coming up to the last one we find all partial derivatives. Due to knowing J we can find all g_i´s and thus we finally find the gradient vector.

Now, the Hesse matrix is to be found.

$$H_{j,i} = H_{i,j} = \frac{\partial^2 J}{\partial u(k-1+i)\partial u(k-1+j)} = \frac{\partial g_i}{\partial u(k-1+j)} \quad \text{... (30)}$$

After putting equation (27) to (30):

$$H_{j,i} = \sum_{m=1}^{P} (y(k+m) - w(k+m)) \frac{\partial^2 y(k+m)}{\partial u(k-1+i)\partial u(k-1+j)} + \frac{\partial F_i}{\partial u(k-1+j)} + \sum_{m=1}^{P} \frac{\partial y(k+m)}{\partial u(k-1+j)} \frac{\partial y(k+m)}{\partial u(k-1+i)} \quad \text{... (31)}$$

The second sum in (31) is known because we know all partial derivatives. The first sum consists of second partial derivatives. To get them we start with (28). The second partial derivative is

$$\frac{\partial^2 y(k+m)}{\partial u(k-1+i)\partial u(k-1+j)} = \gamma(m-i+1, m-j+1) + \gamma(m-j+1, m-i+1) \quad \text{... (32)}$$

Again, if $i < j$ or $i \notin \{1,2,...,M2\}$ or $j \notin \{1,2,...,M2\}$ then $\gamma(i,j) = 0$.

After adding back to (28) dependence on previous outputs, we obtain dependence on previous partial derivatives. Let the vector d be

$$d_{i,j} = \begin{bmatrix} \frac{\partial y(k+1)}{\partial u(k-1+i)\partial u(k-1+j)} \\ ... \\ \frac{\partial y(k+P)}{\partial u(k-1+i)\partial u(k-1+j)} \end{bmatrix} \quad ... (33)$$

Again, we can fill this vector d using (32) and afterwards we will apply the same algorithm as for the matrix J:

for line=2 to P

for k=1 to M1

if (line-k>=1)

$d_{i,j}(line) = d_{i,j}(line) + \alpha_k d_{i,j}(line-k)$

endif

next k

next line

Now there is only one unknown in (31) remaining

$$\frac{\partial F_i}{\partial u(k-1+j)} = -\gamma \text{ if } j = i-1 \text{ or } j = i+1 \quad ... (34)$$

$$\frac{\partial F_i}{\partial u(k-1+j)} = 2\gamma \text{ if } i = j \in \{1,2,...,P-1\} \quad ... (35)$$

$$\frac{\partial F_i}{\partial u(k-1+j)} = \gamma \text{ if } i = j = P \quad ... (36)$$

else partial derivative of F is zero. Now we are able to construct the whole Hesse matrix H and using to (24) we can iterate to the best solution. Iteration stops, if $\sqrt{(\gamma_k g(U_k))^T (\gamma_k g(U_k))} < \varepsilon$ (if last iteration step is very small). If control action u(k+i) steps outside interval $u(k+i) \notin < u_{\min}, u_{\max} >$ algorithm will set it to nearest bound. The final iteration U(k) can be used as part of the starting vector for the next period U(k+1): $U(k+1) = \begin{bmatrix} U(k)_2 \\ ... \\ U(k)_P \\ \varphi^{-1}(w(k+p+1)) \end{bmatrix}$. There is only one unknown position in the vector and this is filled using (20). Now there is only one thing to do. As you can see from Sections 3.2.1 and 3.2.2, the Volterra model doesn't match with process perfectly. So some corrections during computation are to be made. We need to change the setpoint as shown on zoom of Figure 3-1. If we want the controlled variable to reach *yr*, we need to have control *u0*, but because we are controlling the Volterra model, the optimal control is *u1*. Thus if we don't want to have differences between the setpoint and the output from process, we need to make correction to *w* (from *yr* to *yv*). We have to make same correction for measured process output. This problem was caused due to the fact that the model is of a higher order than the Volterra model (quadratic).

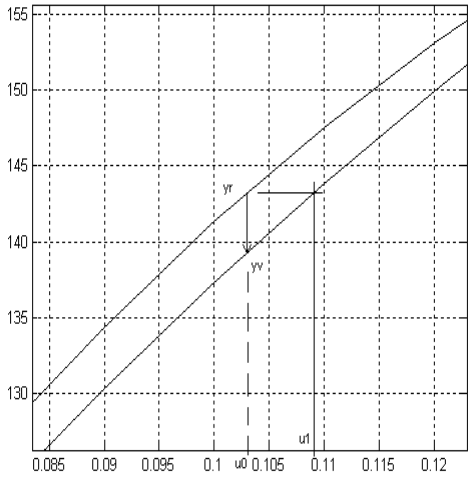

Figure III-3 Correction of the setpoint and the process output

1) Results for chemical reactor

Mostly after 10 periods output is almost constant, if we don't change input to this process with chosen period time 200s. That's why for control of such process we decided for prediction P=10.

Results with $\gamma_2 = 1$ (equation 18):

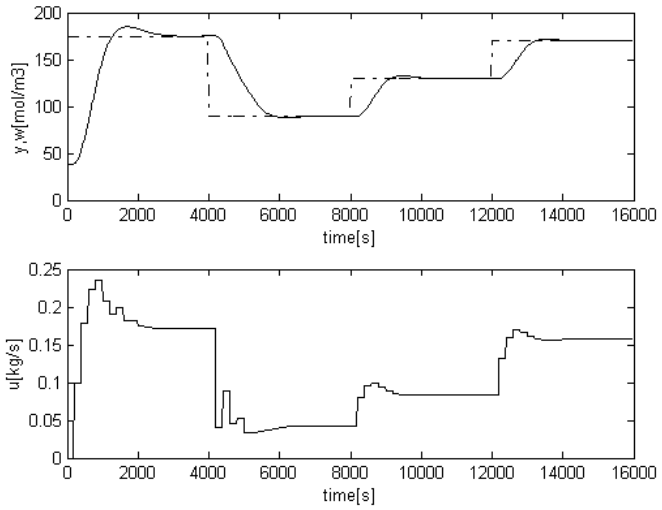

Figure III-4 Controlled variable and setpoint (higher graph), Control (lower graph)

On higher graph in figure 3-6 you can see dash-dot line – setpoint and solid one – controlled variable. On lower graph you can see control.

Results with $\gamma_2 = 0.1$:

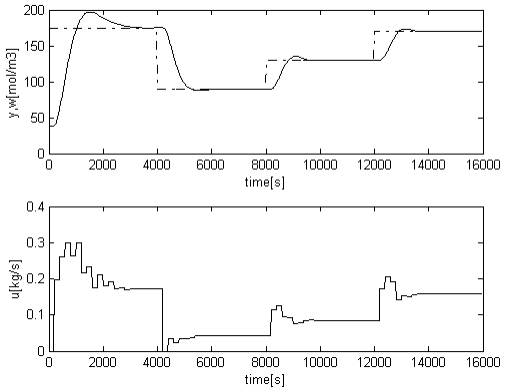

Figure III-5 Controlled variable and setpoint (upper plot), control (lower plot)

If you compare it to Figure 3-6, because change of control is less important, control is faster, but with higher overshoot.

IV. PROBLEMS WITH CONSTRAINTS

Problem arises, if $\gamma_2 = 0 \Rightarrow \gamma = 0$ and solution is near to bound. Then such optimization problem can have problems. You can see it on figure

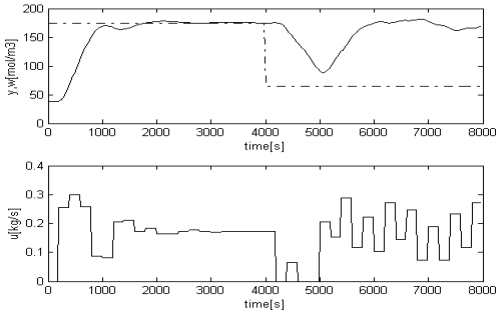

Figure IV-1 Problems of optimization problems with constraint

There is simple explanation of such problem. Prediction P doesn't last enough to come to wanted state (because of constrains, it's not possible to come there so soon) and that's why action oscillates. There are two ways how to solve this problem:

1. Enlarging P

We enlarged P from 10 to 20 and result is following:

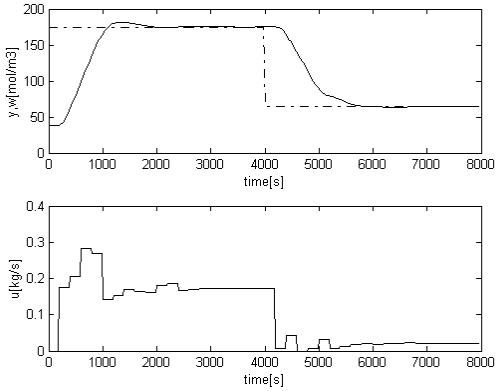

Figure IV-2 Problems of optimization problems with constraint

2. Putting something to equation, what will not let iteration diverge.

When we want to remove checking of possible values of control we can add to cost function something, what would worse much cost function in case of not possible control. Function should look this way:

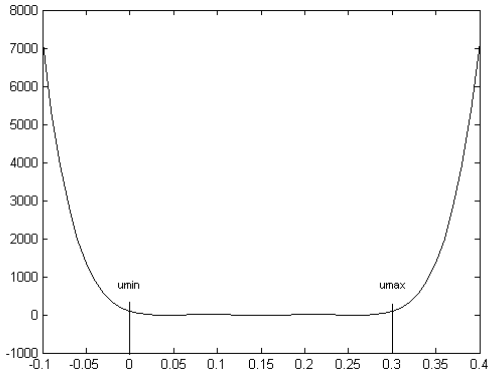

Figure IV-3 Penalization function

The problem is that we have to ensure, that in interval 0 – 0.3 it should not have big waves (or difference of local maximums and their neighbor minimums should be small enough, so cost function of controller will be more important than minimizing of penalization function. This means that iteration will converge to optimal control not optimal penalization function).

This function was approximated by polynomial function by least square method (for every predicted control action):

$$J_P = \sum_{i=1}^{P}\sum_{j=0}^{6}\sigma_j {u_i}^j \qquad \text{... (37)}$$

So final cost function will be $J' = J + J_P$. Now we have to change vector g and matrix H. We will just add what we added to cost function.

$$g_i' = g_i + \sum_{j=1}^{6}\sigma_j j {u_i}^{j-1} \qquad \text{... (38)}$$

And change of matrix H will be just on diagonal $H_{i,i}' = H_{i,i} + \sum_{j=2}^{6}\sigma_j j(j-1){u_i}^{j-2}$. After this modification still with $P = 10$ algorithm will converge:

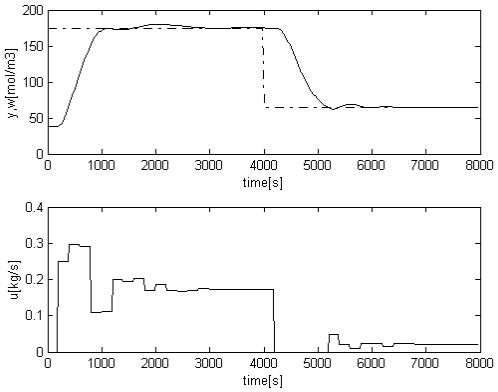

Figure IV-4 Converge of algorithm with modification for constrains

This new algorithm has advantage that we don't need to enlarge prediction horizon, because enlarging of horizon increases complexity of algorithm.

V. CONCLUSION

Joining three fields (model predictive control, Volterra models, optimization) leads to a successful predictive control. Volterra models of second order are suitable to model reactors. With a higher order, the model would be more precise, but control would become more complex. For our reactor Volterra model of degree two is satisfying, because it covered well transmission characteristic. Small differences lead to small differences between setpoint and controlled variable. This problem was solved by correction to setpoints. Simple optimization technique has been used to find the optimal control vector. Whenever this control was outside the allowed interval of control $< u_{\min}, u_{\max} >$, it was set to the nearest bound. At the end it has been shown, how to modify the algorithm if we don't want to have troubles in the vicinity of bounds and we don't want long prediction, because of complexity of algorithm. A simulation case study is provided to demonstrate the effectiveness of the predictive design procedure based on AR Volterra models.

ACKNOWLEDGEMENT

This paper has been supported by the Slovak Scientific Grant Agency, Grant No.1/0155/03.

REFERENCES

[1] F.J. Doyle III, R.K. Perason and B.A. Ogunnaike, Identification and Control Using Volterra Models, Springer-Verlag London Limited 2002

[2] Michael J Grimble, Industrial Control Systems Design, John Wiley & Sons Ltd, 2001

[3] P.Hudzovic, Optimalizacia. Published by Slovenska technicka univerzita v Bratislave, Vydavatelstvo STU, Bratislava, Vazovova 5, 2001.

[4] T.S.Shei, T.A.Johansen, Nonlinear model based/model predictive control with constraints and with/without nonlinear observer. SINTEF Electronics and Cybernetics, Automatic control.