

## DESIGN OF AN COMPUTATIONALLY PROXY OPTIMAL CONTROL SCHEME FOR A NON-LINEAR SYSTEM

<sup>1</sup>Narendra Ilaya Pallavan.P, <sup>2</sup>Srinivasan, S.

<sup>1</sup>Assistant Professor, Department of Electronics and Instrumentation Engineering,  
Bannari Amman Institute of Technology Campus,  
Sathyamangalam, Tamilnadu, India,

<sup>2</sup>Associate Professor, Department of Instrumentation, Engineering,  
Madras Institute of Technology Campus,  
Chennai, Tamilnadu, India,

<sup>1</sup>narendrapallavan@gmail., <sup>2</sup>comsrini@mitindia.edu

**Abstract—** In this work Model Predictive Static Programming (MPSP) scheme proposed by R. Padhi and Kothari (2009) has been implemented on the simulated model of the single-tank level control system. In addition, state space model based MPC is also designed and implemented on the above simulated model. In this work we assume all state variables are available for measurement. From the simulation studies the servo performance of MPSP and NMPC has been compared and results are obtained.

**Keywords—** Model Predictive Control; Model Predictive Static Programming;

### 1. INTRODUCTION

Model Predictive Controller (MPC) is widely used in the supervisory level in process industries in order to control the key variables without violating the input, output and state constraints [1]. It should be noted that the MPC scheme is found to be computationally prohibitive and there are some attempts in recent literature to formulate fast MPC [4].

Recently Padhi and Kothari (2009) developed a MPSP algorithm for the class of finite-horizon nonlinear problems with terminal constraints [3].

Design and implementation of MPSP for a grinding mill circuit is reported in Roux, et al. (2014) [2]. The organization of the paper is as follows.

The MPSP algorithm is described in section 2.

Simulation results are reported in section 3.

The conclusion drawn from simulation studies is presented in section 4.

### 2. MODEL PREDICTIVE STATIC PROGRAMMING

Model Predictive Static Programming (MPSP) is a sub optimal control scheme proposed in Padhi and Kothari [2]. This scheme combines the philosophies of MPC and approximate dynamic programming to modify the dynamic optimization problem into a static optimization problem. It is a class of finite horizon optimal control problem which is formulated in the framework of parametric programming. The algorithm reduces the optimization time by calculating the sensitivity matrices recursively and uses closed form expression to update the control moves. In MPSP scheme control and prediction horizons are assumed to be identical. The detailed MPSP algorithm is reported in Roux, et al. (2014) [2].

#### 2.1 Model predictive static programming algorithm [2]

Let the nonlinear system be described using the following state and output equations.

$$X_{k+1}^i = F_k(X_k^i, U_k^i) \quad (1)$$

$$Y_k^i = H_k(X_k^i, U_k^i) \quad (2)$$

Where,  $k$  is the discrete time index and  $i$  is the iteration index.

The main objective of MPSP is to find an input projection  $U_k$ ,  $k = 1, 2, \dots, N_{so}$  that the output  $Y_k$  goes to the desired output value  $Y_k^*$  for time steps 1 to  $N$ . It's necessary to start with an initial guess for  $U_k \forall k = 1, 2, \dots, N$  with which we can find  $X_k, Y_k \forall k = 1, 2, \dots, N$ . The convergence is checked with the below condition.

$$\frac{\|Y_k^i - Y_k^*\|}{\|Y_k^*\|} < \epsilon_k, \forall k = 1, 2, \dots, N \quad (3)$$

The output  $Y_k^{i+1}$  at time step  $k$  and iteration  $(i + 1)$  can be expanded using Taylor series expansion, retaining only first order terms the equation (2) becomes

$$Y_k^{i+1} = Y_k^i + \left[ \frac{\partial H_k}{\partial X_k} \right] \Delta X_k^i + \left[ \frac{\partial H_k}{\partial U_k} \right] \Delta U_k^i \quad (4)$$

$$Y_k^{i+1} - Y_k^i = \Delta Y_k^i = \left[ \frac{\partial H_k}{\partial X_k} \right] \Delta X_k^i + \left[ \frac{\partial H_k}{\partial U_k} \right] \Delta U_k^i \quad (5)$$

Similarly, equation (1) becomes

$$\Delta X_{k+1}^i = \left[ \frac{\partial F_k}{\partial X_k} \right] \Delta X_k^i + \left[ \frac{\partial F_k}{\partial U_k} \right] \Delta U_k^i \quad (6)$$

Where  $\Delta X_k^i$  is the error in the state and  $\Delta U_k^i$  is the error in the input solution at time step  $k$  and iteration  $i$ . The output error  $\Delta Y_k^i$  can be written in terms of the state and input error. For small deviation  $\Delta Y_k^i = dY_k^i$ ,  $\Delta X_k^i = dX_k^i$  and  $\Delta U_k^i = dU_k^i$ .

$$dY_k^i = [A^k]^i dX_1^i + [B_1^k]^i dU_1^i + \dots + [B_{k-1}^k]^i dU_{k-1}^i + [B_k^k]^i dU_k^i \quad (7)$$

$$dY_k^i = \sum_{j=1}^k [B_j^k]^i dU_j^i \quad (8)$$

where

$$[A^k]^i = \left[ \frac{\partial H_k}{\partial X_k} \right] \left[ \frac{\partial F_{k-1}}{\partial X_{k-1}} \right] \left[ \frac{\partial F_{k-2}}{\partial X_{k-2}} \right] \dots \left[ \frac{\partial F_1}{\partial X_1} \right] \quad (9)$$

$$[B_j^k]^i = \left[ \frac{\partial H_k}{\partial X_k} \right] \left[ \frac{\partial F_{k-1}}{\partial X_{k-1}} \right] \dots \left[ \frac{\partial F_{j+1}}{\partial X_{j+1}} \right] \left[ \frac{\partial F_j}{\partial U_j} \right] \quad \forall j < k \quad (10)$$

$$[B_j^k]^i = \left[ \frac{\partial H_k}{\partial U_k} \right] \quad \forall j = k$$

$$[B_1^k]^i = 0 \quad \forall j > k$$

Objective function is given below

$$J^i = \frac{1}{2} \sum_{k=1}^N (Y_k^{i+1} - Y_k^*)^T Q_k (Y_k^{i+1} - Y_k^*) + \frac{1}{2} \sum_{k=1}^N (U_k^{i+1} - U_k^i)^T R_k (U_k^{i+1} - U_k^i) \quad (11)$$

$$J^i = \frac{1}{2} \sum_{k=1}^N (Y_k^i + dY_k^i - Y_k^*)^T Q_k (Y_k^i + dY_k^i - Y_k^*) + \frac{1}{2} \sum_{k=1}^N (dU_k^i)^T R_k (dU_k^i) \quad (12)$$

Applying the Taylor's series expansion for the cost function and differentiating it with respect to control inputs  $U_z$  where  $z=1$  to

$N$ ,

$$\frac{\partial J}{\partial (dU_z)} = \sum_{k=1}^N \left( B_z^{kT} Q_k \sum_{j=1}^k B_j^k dU_j \right) - \sum_{k=1}^N B_z^{kT} Q_k dY_k^* + R_z dU_z \quad (13)$$

$$\begin{aligned} & \sum_{k=1}^N \left( B_z^{kT} Q_k \sum_{j=1}^k B_j^k dU_j \right) \\ &= \sum_{k=1}^N \left( (B_z^{1T} Q_1 B_1^1 dU_1) \right. \\ &+ (B_z^{2T} Q_1 B_1^2 dU_1 + B_z^{2T} Q_2 B_2^2 dU_2) \\ &+ \dots (B_z^{NT} Q_N B_1^N dU_1 + B_z^{NT} Q_N B_2^N dU_2 \\ &+ \dots + B_z^{NT} Q_N B_N^N dU_N) \left. \right) \quad (14) \end{aligned}$$

$$\begin{aligned} \sum_{k=1}^N \left( B_z^{kT} Q_k \sum_{j=1}^k B_j^k dU_j \right) &= \sum_{l=1}^N (B_z^{lT} Q_1 B_1^l dU_1) \\ &+ \sum_{l=1}^N (B_z^{lT} Q_2 B_2^l dU_2) + \dots \\ &+ (B_z^{NT} Q_N B_{N-1}^N dU_N) \quad (15) \end{aligned}$$

$$\begin{aligned} \frac{\partial J}{\partial (dU_z)} &= C_{z1} dU_1 + C_{z2} dU_2 + \dots + C_{zN} dU_N \\ &- \sum_{k=1}^N B_z^{kT} Q_k dY_k^* + dU_z \quad (16) \end{aligned}$$

$C_{ej}$  can be defined as

$$C_{ej} = \sum_{l=1}^N B_e^{lT} Q_l B_j^l \quad (17)$$

Equating equation (16) to zero

$$\begin{aligned} \frac{\partial J}{\partial (dU_z)} &= 0 \\ \sum_{k=1}^N B_z^{kT} Q_k dY_k^* + dU_z \\ &= C_{z1} dU_1 + C_{z2} dU_2 + \dots \\ &\quad + C_{zN} dU_N \end{aligned} \quad (18)$$

Compiling all equations for all times steps, the system of equations can be written as

$$[dU_e] = [C_{ej} + \delta_{ej} R_e]^{-1} [b_e] \quad (19)$$

Where

$$[b_e] = \sum_{k=1}^N B_z^{kT} Q_k dY_k^* \quad (20)$$

It should be noted that the change in control output for 1 to N time instants are function of  $[B_j^k]^i$  and  $[A^k]^i$  which are nothing but the sensitivity matrices. Since optimization is solved as a function of sensitivity matrices, this type of optimization is called **parametric optimization**. At the same time we obtain optimized control update as a **closed form solution** equation (19) which reduces the optimization time.

The steps involved in the computation of controller output by MPSP are as follows:

**Given:** Initial state  $X_1$  and plant model.

**Step-1:** Initialize the control input trajectory  $U_{k+j}^0, k = 1, 2, \dots, N$ . Initialize  $i$  to 0, and  $j$  to 0.

**Step-2:** With known initial condition and input trajectory  $U_{k+j}^i, k = 1, 2, \dots, N$ . Obtain the state trajectory ( $X_{k+1+j}^i, k = 1, 2, \dots, N$ ) using equation (1). Find the output trajectory using equation (2). And check for convergence using equation (3).

**Step-3:** If convergence is satisfied then use  $U_{k+j} = U_{k+j}^i, k = 1, 2, \dots, N$  as optimal trajectory. Apply the first control move to the process and move forward in time ( $j$  to  $j+1$ ) and continue with step 1. Initial state  $X_{j+1}$  can be found by giving the optimal first move  $U_j^i$  to the plant model (1), (2).

**Step-4:** If convergence is not satisfied, increase the iteration index  $i$  to  $i+1$  and using equation (19) find the change in control moves. Find new control input trajectory with the relation  $U_k^i = U_k^{i-1} + \Delta U_k^i \forall k = 1, 2, \dots, N$ , and go to step 2.

### 3. RESULTS AND DISCUSSIONS

In this section we report the design and implementation of MPSP scheme on the simulated model of a single-tank system.

### 3.1 Implementation of algorithm

The governing equation of single-tank system is as follows.

$$\frac{dh}{dt} = F_{in} - 0.5\sqrt{h} \quad (21)$$

The discretized model is given below.

$$h(k+1) = h(k) + (F_{in}(k) - 0.5\sqrt{h(k)})T \quad (22)$$

Where, T is the sampling time.

We need first derivative matrixes for finding the sensitivity matrixes in the algorithm.

$$\frac{\partial X(k+1)}{\partial X(k)} = \frac{\partial h(k+1)}{\partial h(k)} = 1 + \left( -0.25 * \frac{T}{\sqrt{h(k)}} \right) \quad (23)$$

$$\frac{\partial X(k+1)}{\partial U(k)} = \frac{\partial h(k+1)}{\partial F_{in}(k)} = T \quad (24)$$

The process is simulated in MATLAB with a simulation time of 50 seconds. The servo response with MPSP is shown in Figure 6.

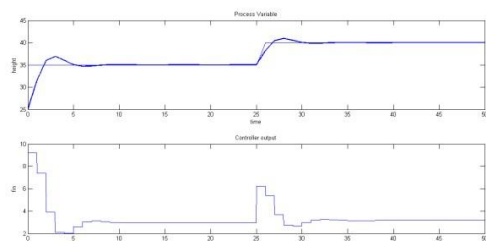
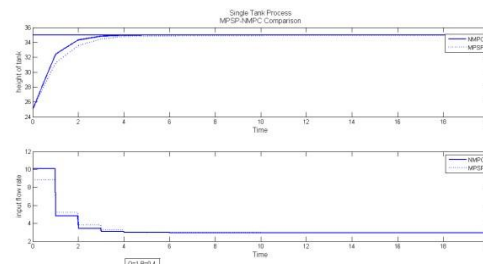


Figure 6: Servo response of the singletank system with MPSP algorithm



### 3.2 IMC type offset elimination

In the presence of plant-model mismatch, MPSP will exhibit an offset between SP and PV. In order to eliminate the offset, IMC type compensation has been carried out. In this method, the difference between the measured  $y(k)$  and model output  $y_m(k)$  is added to the future predicted outputs at all instants from  $k$  to  $k+N_p$ .

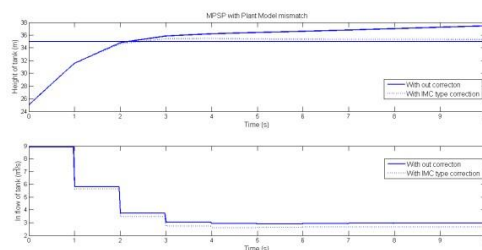


Figure 7: Servo response of single-tank system with MPSP with IMC type compensation

### 3.3 MPSP-NMPC comparison for the single-tank system

The cost function of the NMPC is as follows:

$$J = \frac{1}{2} \sum_{k=1}^{N_p} (Y_k - Y_k^*)^T Q_k (Y_k - Y_k^*) + \frac{1}{2} \sum_{k=1}^{N_c} (U_{k+1} - U_k)^T R_k (U_{k+1} - U_k)$$

The first term in the cost function is identical to the MPSP scheme, however the second term differs. In NMPC the variance of control input between consecutive time steps is minimized. Whereas in MPSP the variance of control input between consecutive iteration is minimized. So, while comparing both we have to make the effect of the second term minimum by minimizing the R matrix value. The results are obtained by keeping the prediction horizons to be 5s for both the cases and the control horizon is 5s in case of NMPC.

Table 1: Performance comparison of MPSP & NMPC

| Performance measure | MPSP     | NMPC   |
|---------------------|----------|--------|
| ISE                 | 113.48   | 106.74 |
| Control effort      | 15.74    | 15.38  |
| Computational time  | 0.4690 s | 2.4 s  |

Figure 7: Comparison of performance of MPSP and NMPC schemes

From Table.1, it can be inferred that computational time is less in case of MPSP.

## 4. CONCLUSION

In this work MPSP and NMPC schemes have been implemented on the simulated model of the single-tank system. From the extensive simulation studies it can be observed that the performance of MPSP scheme is found to be matching with NMPC scheme. However the computational time of MPSP is found to be less compared to the NMPC.

## 5. REFERENCES

- [1] **Wayne Bequette B**, "Process Control Modelling, Design and Simulation", Prentice Hall International Series, (2002).
- [2] **Johan D. le Roux, Radhakant Padhi and Ian K. Craig**, "Optimal control of grinding mill circuit using model predictive static programming: A new nonlinear MPC paradigm", Journal of Process Control, vol. 20, pp. 29–40, (2014).
- [3] **R. Padhi and M. Kothari**, "Model Predictive Static Programming: A Computationally Efficient Technique for Suboptimal Control Design", International Journal of Innovative Computing Information and Control, IJICIC, vol. 5, pp. 399-411, (2009).
- [4] **Yang Wang and Stephen Boyd**, "Fast Model PredictiveControl Using Online Optimization" IEEE Transactions on Control Systems Technology vol. 15, no. 2, (2010).



