CH5120 (Jul'17- Nov'17) Course Project - 02

MODEL PREDICTIVE CONTROL

RIDHI PUPPALA - ME15B133 AKASH ANILKUMAR - ME15B005

Model predictive controllers rely on dynamic models of the process, most often linear empirical models obtained by system identification.
In the given project System (Plant) is a non linear model defined in **fcc_parametres.m and fcc_fn_to_solve_odemodel.m** files.
The function ode15s from Matlab was used to integrate the Yrates from ODE file and compute Plant measurement data generation. A function handler (fun) was defined for Fcc Model file.
The model equations for implementing Model Predictive Control (MPC) is given as:

$$\mathbf{x}_{k+1} = A\,\mathbf{x}_k + B\,\mathbf{u}_k$$
$$\mathbf{y}_k = C\mathbf{x}_k$$

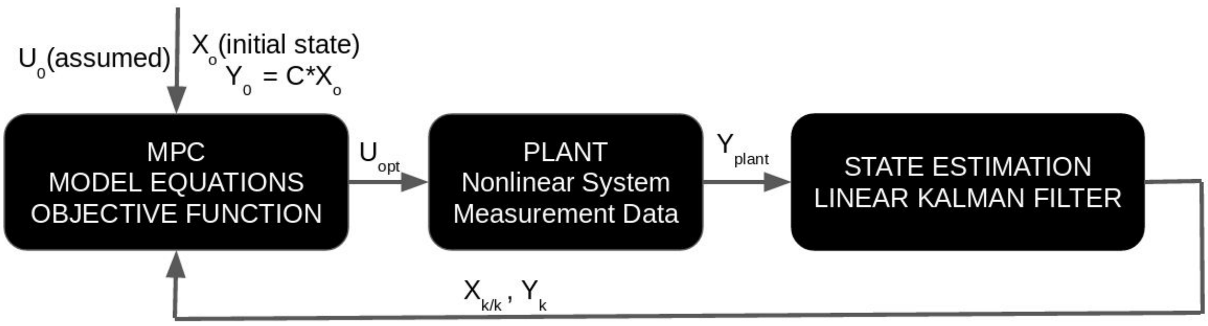
The model equations are also used for State Estimation through Linear Kalman Filter implementation.

General Settings for implementation of MPC are :

Number of iteration **(Nsteps)** = **2000** (So as to observe the dynamics of model over large time)
Control Horizon (**M**) = **3** Prediction Horizon (**P**) = **3**
$\mathbf{U_0}$ = **[0;0]**; (assumed) $Y_0 = C*X_o$ (initial value for first iteration of MPC)
Objective function file has been defined in **objFmincon.m.** It is called in **gen_mpc.m** file
Fmincon function from Matlab was used for objective function minimization.
Selection of the weights Q_u and Q_y : The selection of the weights were based on the ranges of operation. Since the first two measured parameters were concentrations there ranges of operation were between 0 and 1.Whereas the third measurement being temperature. The range of operation was between 0 and 1000. Q_y was chosen to be diagonal to attain ***decoupling***. Since both our inputs were defined as flow rates their weights were chosen to be the same. Therefore the Q_u was chosen to be diag ([1 1)].
Kalman parmameters: The measurement noise covariance R is estimated from knowledge of predicted observation errors. We chose it to be a diagonal matrix to achieve ***decoupling*** of error contributions. The entries were fixed on a trial and error basis was chosen to have a higher value compared to Q because we depend more on the predictions.

Part - (a) and (b)

Flow diagram for general MPC implementation



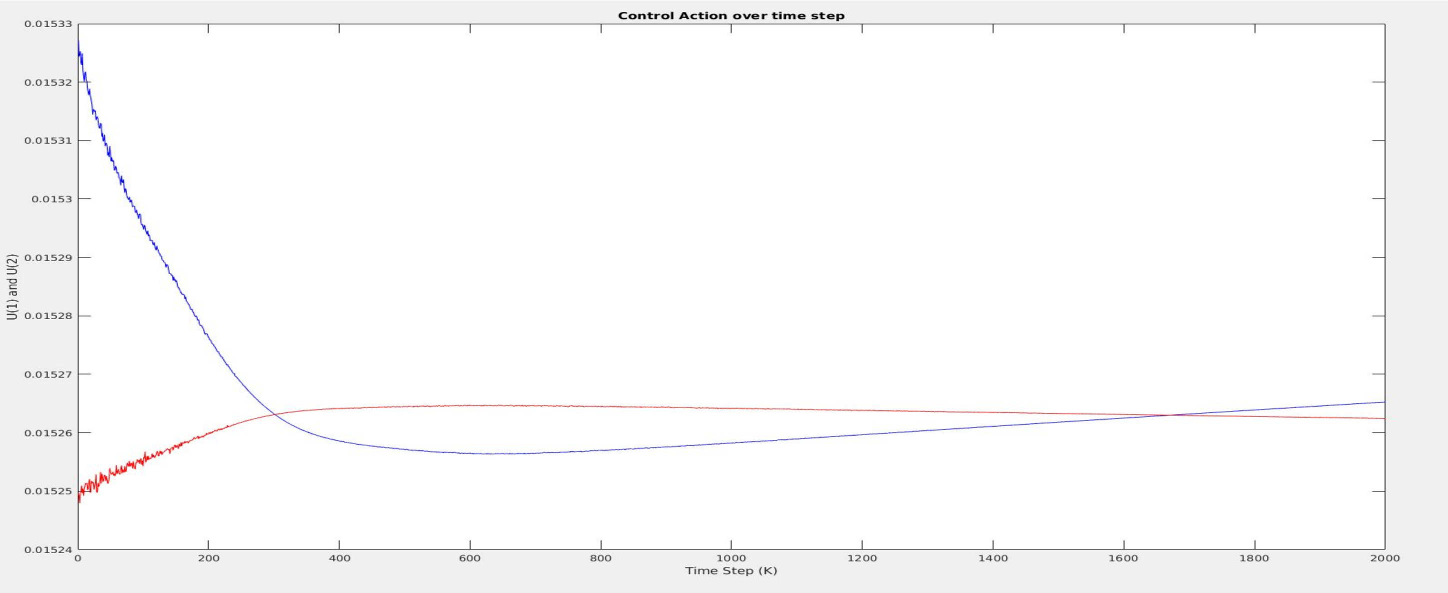
With a M = 3, P = 3, time for each iteration ~ 0.3s Total Run time for 2000 iterations ~10minutes
Fmincon with a lower bound (lb) = [0;0] for Uhat was used.
Tuning parameters (MPC) :weightX (Q_x) , weightU (Q_u), P, M
Tuning parameters (Kalman Filter - KF) : $P_{o/o}$ (assumed a small value of order 0.001), R, Q

Procedure:

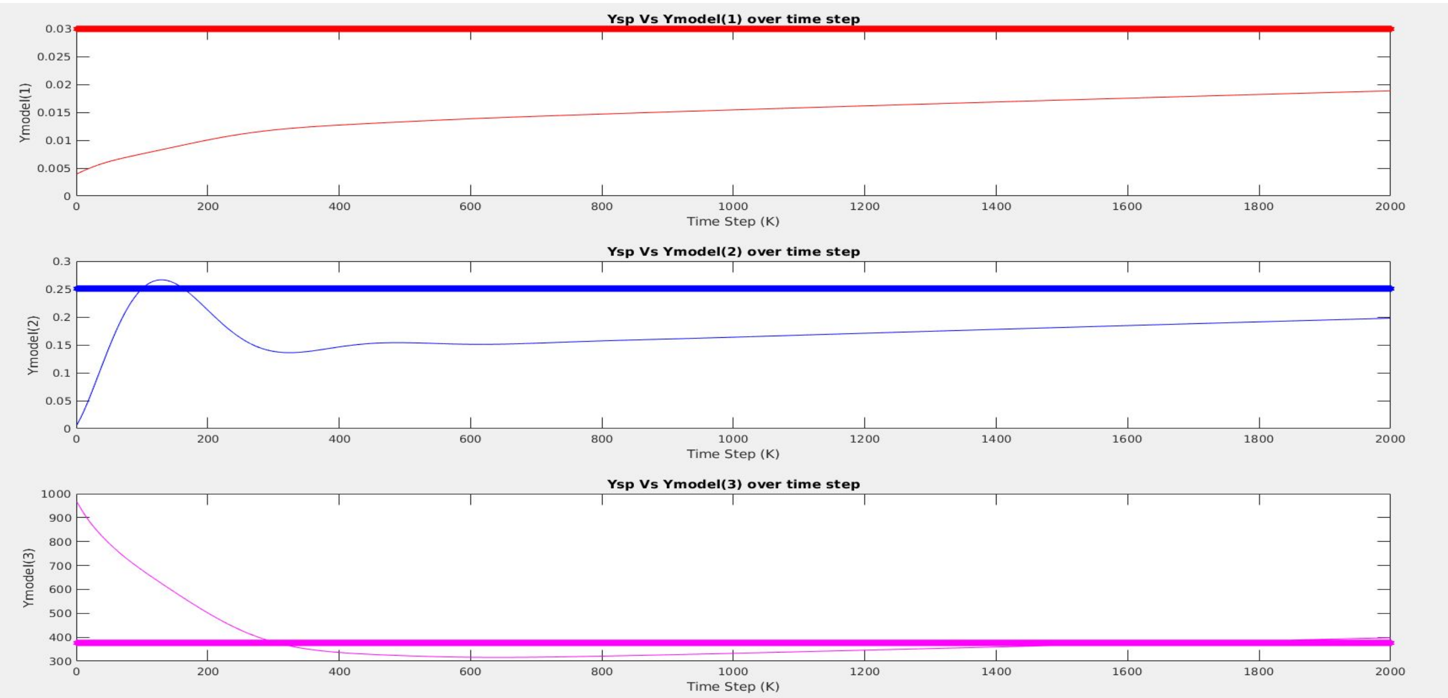
1. The parametres of weightX, weightU, P, M and $P_{o/o}$ were adjusted to get optimal results. Tuning was done till the Y predicted values converged or tried to converge to setpoint(Ysp).
2. With a high values of P and M, the computation time was very high.So a low value was selected. Also a lower value of control horizon(M) lead to aggressive control action.
3. Q was computed from variance of 3 elements of X_o, while R was assumed suitably.
4. $Y_0 = C*X_o$ was used as initial value for plant measurement generation and MPC
5. Lower bound lb = [0;0] was introduced in fmincon as practically, U<0 is not possible for plant.

6. No upper bound or other constraints have been used

Results:



Ymodel (predicted Y) versus Ysp over 2000 timesteps

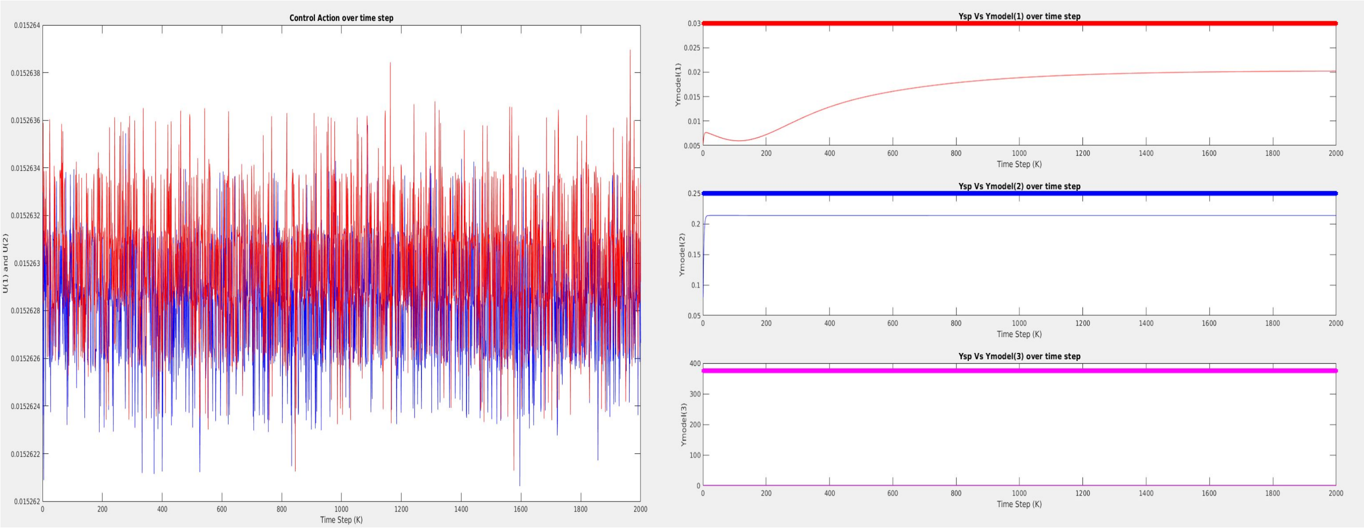


Control action (U(1) in blue and U(2) in red) over 2000 timesteps

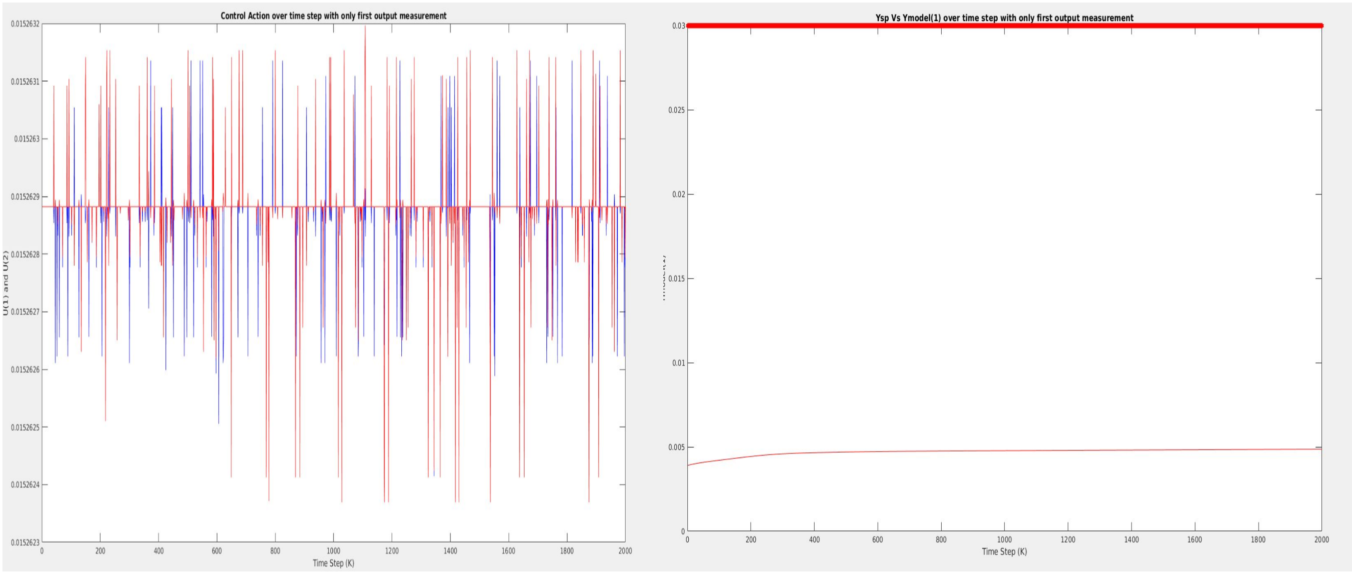
Observation:

1. While tuning Ymodel values predicted were very sensitive to $P_{0/0}$.
2. There was a setpoint error in Y(2), which could have been corrected using **equality constraints** in fmincon function. Ysp = Yk = $C^*(AX_k + BU_k)$ => A_{eq} = CB and b_{eq} =(Yk - CAX_k) could be used to eliminate deviation of Ymodel from Ysp.
3. 2000 iterations was chosen to observe the all parametres of Y to converge to setpoint.Y(2) converges quickly while Y(1) takes huge amount of time to converge. For large values of P and M, computation time for 2000 iterations was very high, so a lower value of M and P = 3 was chosen.
4. Since the inputs U determines the flow in an FCC Model. They cannot be negative. As a result while optimization a lower bound of 0 was given for the results. This ensured that the optimized U always had a non zero value.

Part - (c)
To have a control over number of outputs, a matrix **Cbar = diag[1 1 1]** was defined.
C = Cbar*C_new , where C_new was given in **linssmodel.mat**
By changing the elements in Cbar, we had a control on what outputs will be there.
Case-1: Cbar = [1 1 0] ⇒ only first two outputs
Case-2: Cbar = [1 0 0] ⇒ only first output Case-2: Cbar = [1 1 1] ⇒ all three outputs
Figures :
Y_case2_b.fig and U_case2_b for only first two output measurement case
Y_case3_b.fig and U_case3_b for only first output measurement case



Input and Output plot respectively in the case of Only first 2 outputs measured



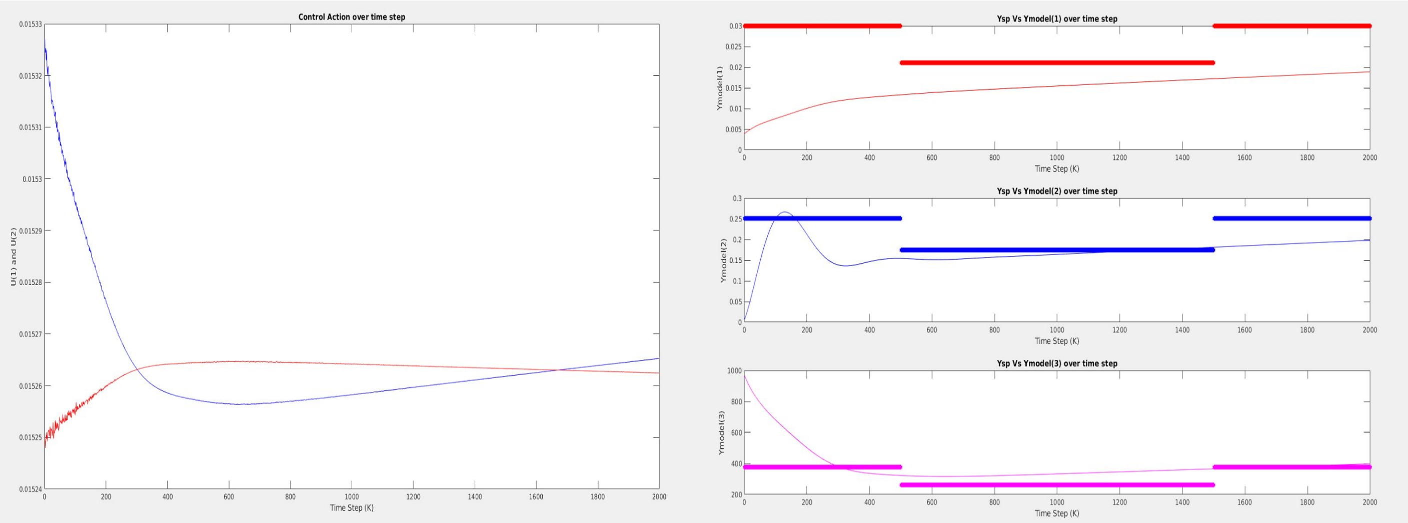
Input and Output plots respectively in the case of Only single output

Observation :

1. This plots clearly show that good control can't be achieved with only one or two output.
2. In both cases the input profile or control action is very noisy and sharp.The input changes drastically for consecutive time step. Practically this is not possible for plant model given.
3. We could introduce an upper bound ub so as to ensure deltaU for each consecutive time step is within practical limits of actual plant.

Part - (d)
Using the same parameters, but with a desired profile, plots were generated. Similar results were obtained, in all three cases of outputs as observed in plots attached.
Figures :
Y_case1_d.fig and U_case1_d.fig for only first two output measurement case
Y_case2_d.fig and U_case2_d.fig for only first two output measurement case

Y_case3_d.fig and U_case3_d.fig for only first output measurement case



Input and Output plots respectively with desired profile and all three output

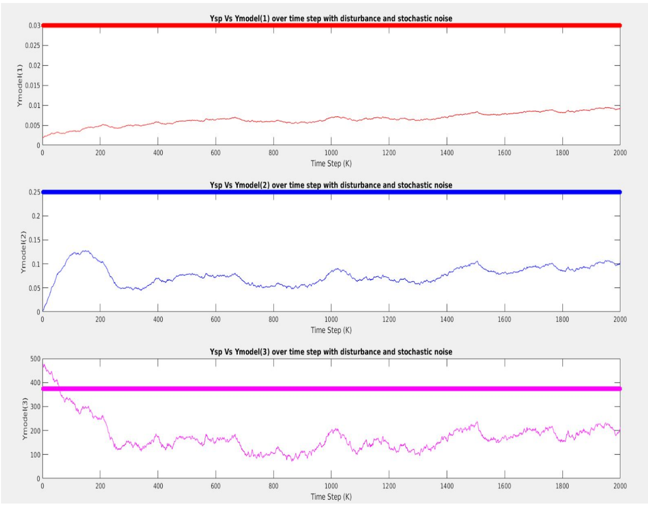
Part - (e)

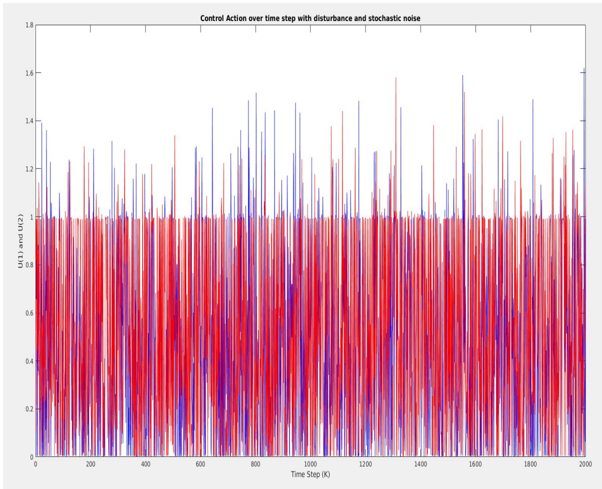
Disturbance (dk) was introduced with **rand()** function appropriately according to magnitude of measurements in the Y equation of model.

Normal stochastic error(w_k) with mean = 0 and sigma was introduced using normrnd() function in Plant measurement generation in Case1.

Bias given in problem statement and was introduced in Case2 in Plant measurement.

Case1 - Disturbance in Model and Stochastic noise in plant measurement

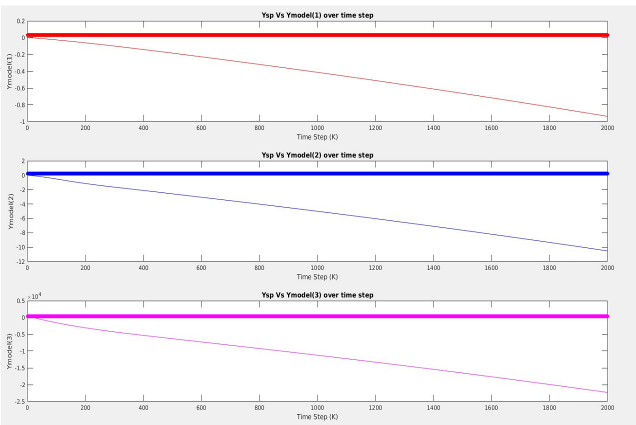


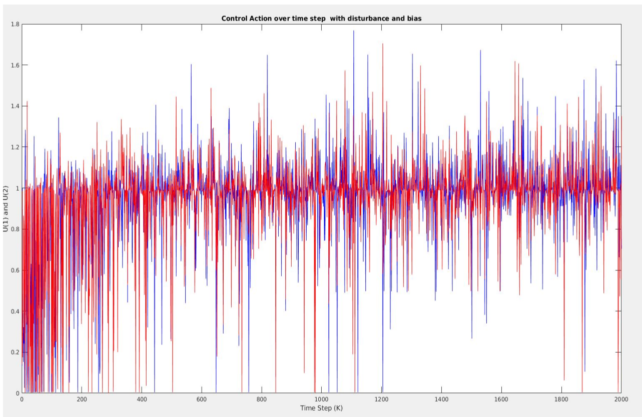


From the above plots it can be inferred that Ymodel vs time is more prone to noise and less controllable. Input profile is very sharp and changing drastically at each time step.

Solution to this can be introducing an upper bound constraint on the magnitude of deltaU and fitting a curve through the the noisy U to attain a smooth input profile. Tuning KF and MPC parameters will help

Case2:





In case of bias and disturbance the system is not controllable even after tuning several times.

Figures :

Y_dk_wk.fig and U_dk_wk.fig are for Case 1

Y_dk_bias.fig and U_dk_bias.fig are for Case 2

Part - (f)

Effect of X0 in MPC performance:

Two different cases were chosen for X0,

Case1 - X0 = [0;0;0] : Y(2)(O_d) is converging with an offset error. Characteristics of Y(1) and Y(2) are similar. Input is more smoother in the beginning as compared to original X0 given. Convergence of inputs has shifted accordingly.

Case2 - X0 = 0.5*X0 (given) : Y(1) and Y(2) converge while Y(2) has crossed setpoint and oscillating over 2000 timesteps. Input profile for both elements of U have an increased variation.

Figures :

Y_zero_X0.fig and U_zero_X0.fig are for Case 1

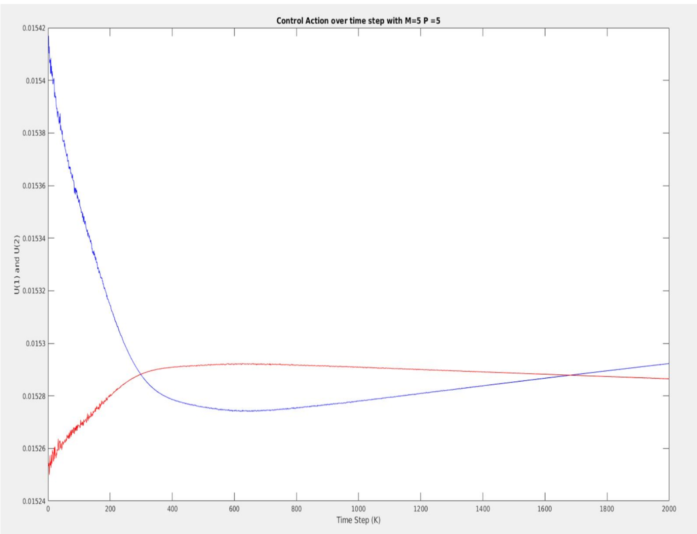
Y_half_X0.fig and U_half_X0.fig are for Case 2

Effect of M and P in MPC performance:

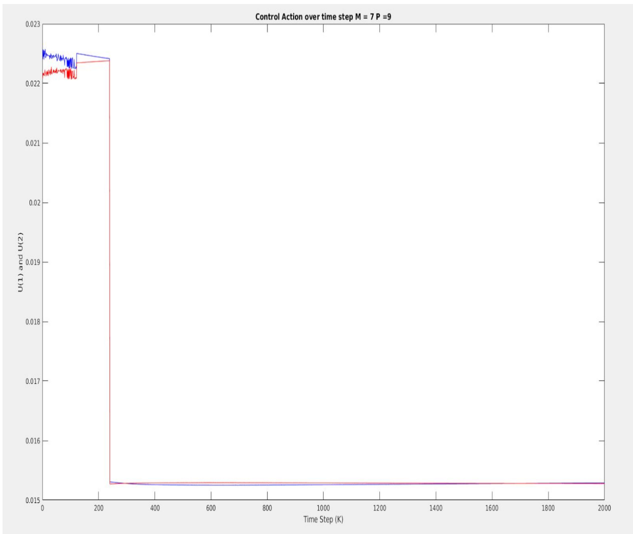
Case 1 : M = 7 and P =9

Case 2 : M = 5 and P = 9

- For P = 3 and M <3, the control action was very aggressive and led to huge setpoint error, while computation time for each iteration was in the order 0.05seconds.
- For M = 9 and P = 7 it was observed that input profile had a huge profile around 220 timestep due to unknown reasons, which could be probably due to difference in M and P.
- The profiles for Ymodel were similar in all cases of different M and P
- Small *M* means fewer variables to compute in the QP solved at each control interval, which promotes faster computations.
- If the plant includes delays, *M*<*P* is essential. Otherwise, some MV moves might not affect any of the plant outputs before the end of the prediction horizon, leading to a singular QP Hessian matrix.
- Small *M* promotes an internally stable controller.



Input Profile for M = P =5



Input profile for M=7 and P =9 has a step at ~230 timestep

Note: Check attached figures and codes in the zip file