5LMB0 - Model Predictive Control

Graded Assignment 3 - Inverted Pendulum Systems

Gregorius Rafael Widojoko (ID:1773828)
Ehsan Lakzaei (ID:1261797)

Department of Electrical Engineering

April 21, 2022

1. Non-linear MPC design using LPV approach

1.a. Linear Parameter-varying formulation

Discretizing the system (A) using the Forward-Euler method (with sample time T_s) will lead to:

$$\mathbf{x(k+1)} = \begin{bmatrix} x_1(k) + T_s x_2(k) \\ \frac{mgl}{J} T_s \mathbf{sin(x_1(k))} + (1 - \frac{b}{J} T_s) x_2(k) + \frac{k}{J} T_s x_3(k) \\ -\frac{k}{L} T_s x_2(k) + (1 - \frac{R}{L} T_s) x_3(k) + \frac{T_s}{L} u(k) \end{bmatrix} = \begin{bmatrix} x_1(k) + T_s x_2(k) \\ \frac{mgl}{J} T_s \mathbf{sinc(x_1(k))} \mathbf{x_1(k)} + (1 - \frac{b}{J} T_s) x_2(k) + \frac{k}{J} T_s x_3(k) \\ -\frac{k}{L} T_s x_2(k) + (1 - \frac{R}{L} T_s) x_3(k) + \frac{T_s}{L} u(k) \end{bmatrix}$$

Noticing the $\sin(x_1(k))$ part from the left matrix could be rewritten as $\frac{\sin(x_1(k))}{x_1(k)} x_1(k) := sinc(x_1(k))\, x_1(k)$, one can form the Linear Parameter Varying (LPV) formulation of $\boxed{x(k+1) = A(\rho(k))\, x(k) + B(\rho(k))\, u(k)}$ with :

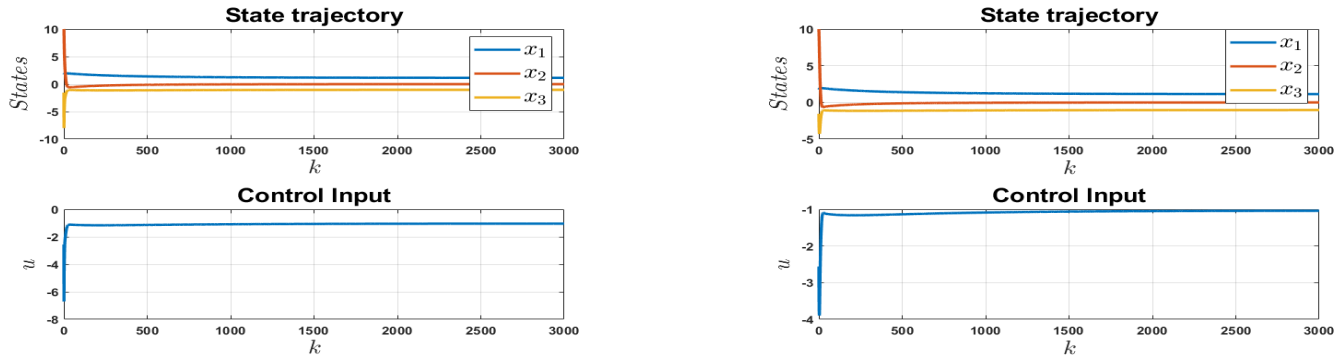
$$A(\rho(k)) = \begin{bmatrix} 1 & Ts & 0 \\ \frac{mgl}{J} T_s \rho(k) & (1 - \frac{b}{J} T_s) & \frac{k}{J} T_s \\ 0 & -\frac{k}{L} T_s & (1 - \frac{R}{L} T_s) \end{bmatrix} \quad B(\rho(k)) = \begin{bmatrix} 0 \\ 0 \\ \frac{T_s}{L} \end{bmatrix} \quad \rho(k) = sinc(x_1(k)) := \begin{cases} 1, \text{ for } x_1(k) = 0 \\ \frac{\sin(x_1(k))}{x_1(k)} \text{ otherwise} \end{cases}$$

1.b. Designing a MPC using an LPV approach

According to [1], the predicted sequence X_k in a compact form is defined as $\boxed{X_k = \Phi(P(k))x(k) + \Gamma(P(k))U_k}$, where $P(k) := \{\rho_{0|k}, ..., \rho_{N-1|k}\}$ and $\rho_{0|k} := \rho(k)$. To design a non-linear MPC controller using LPV approach, the following steps have been taken:

1. **Generating initial data ($P(k)$)** (Considering the given informations of $x(0)$, N, P, Q, R, A_d,B_d,C_d,D_d and x_{ss})
   Solving the constrained linear MPC quadratic program for a single run to obtain the first U_k for N steps prediction. Afterwards, based on the closed-loop system and predicted states for N steps in the future, the $P(k)$ vector is obtained.

2. **Main script**
   After computing the initial data (U_k and $P(k)$), a function has been made to obtain the Φ and Γ matrices based on [1] to make the predicted X_k sequence holds. The G, F (from the cost function), W, L and c (from constraints) matrices will be constructed similar to previous assignments (using getWLc() and QRPN2PsiOmega() functions). Afterwards, the closed loop will be constructed with the first elements in both $P(k)$ and $u_{mpc}(k)$. Updating the P vector with a for loop as mentioned in [2] will be the last step of the design procedure.

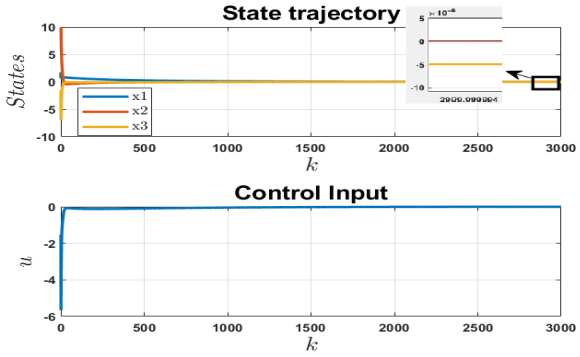


(a) Closed-loop (b) Simulink

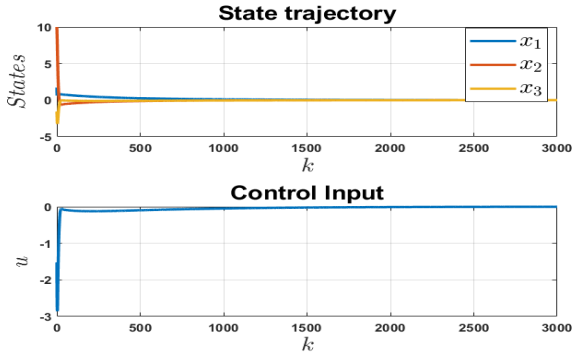
Figure 1: State trajectory and control inputs of LPV-MPC simulation

As it can be seen in figure 1a, the states and the input controller do not violate the constraints (given in the assignment description). Moreover, the states and the input controller do not converge to zero which is caused by the fact that the cost function is a non-convex problem. This means that answer from solving cost function is the local minima and not the global minima. One way to tackle this convergence problem is shifting the states by the steady state values. To do so, one can run the simulation and take the last value of the states as the steady state values. This leads to an error less than $|5e-04|$ which is reasonable and can be accepted as zero convergence (rule of thumb), see figure 2a. It should be mentioned that all these observations hold for the Simulink model as well.
The figures in the next page illustrate the closed-loop simulation and Simulink model simulation.

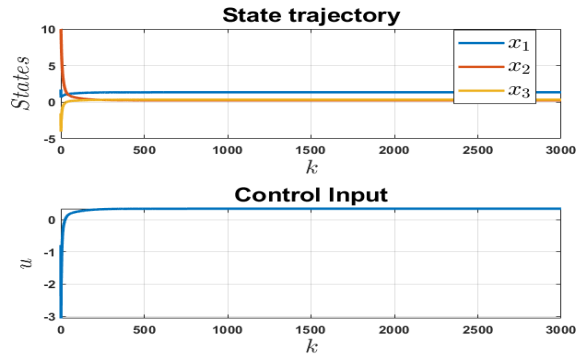
Bonus Question
Based on figures 2c and 2d, Although the states and input controller are less aggressive than the case when $N = 10$, it is clear that even by shifting the steady state values, the linear model as well as Simulink model will not converge to zero. Furthermore, the error in steady states in Simulink model is larger than the linear one.



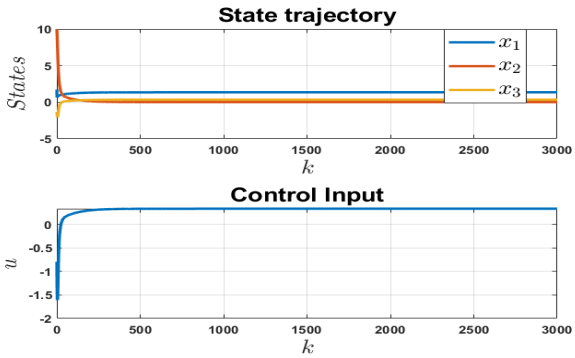
(a) Closed-loop



(b) Simulink



(c) Closed-loop (N=4)



(d) Simulink (N=4)

Figure 2: Corrected state trajectory and control inputs of LPV-MPC simulation

1.c. Computational Time comparison

In figure 3, it can be observed that the computation time for the LPV MPC simulation is way higher than the LPV MPC closed-loop simulation, with the average time of 56.83 and 1.84 milliseconds respectively.
However, the linear MPC's computational time (Simulink and closed-loop simulations) is no better than the LPV MPC ones. Both the average computational times are 5.85 and 73.81 milliseconds for the linearised closed-loop and linear MPC simulink simulations respectively.
This means that the LPV MPC performs faster in terms of handling the non-linear dynamics simulations compared to the linear MPC.

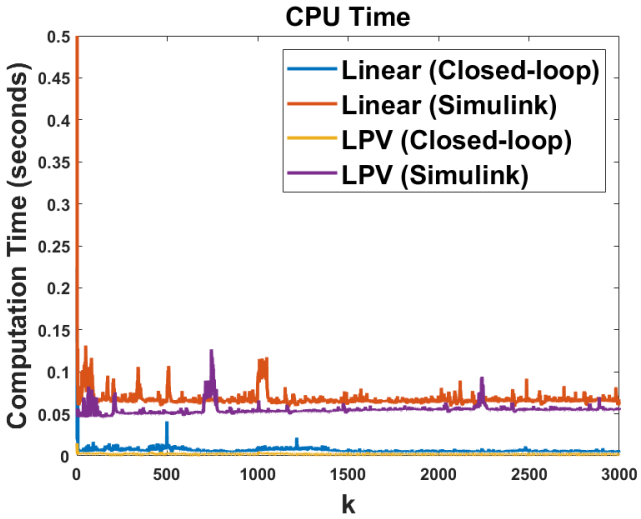


Figure 3: CPU time

APPENDIX

References

[1] Lecture 6, slide 24.

[2] Lecture 6, slide 27.

A. State space representation of the system-assignment 1

Based on previous assignments, the state space representation of the system is obtained as following:

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{x_2} \\ \frac{mgl}{J}\sin(\mathbf{x_1}) - \frac{b}{J}\mathbf{x_2} + \frac{k}{J}\mathbf{x_3} \\ -\frac{k}{L}\mathbf{x_2} - \frac{R}{L}\mathbf{x_3} + \frac{1}{L}\mathbf{u} \end{bmatrix}$$