Model Predictive Control
Prof. Manfred Morari

ESE 6190, Spring 2023
Due: Mar 15

Exercise sheet 4
MPC Theory I

Instructions:

1. Apart from your solutions, you are also required to upload your code to Gradescope. Make sure the code is well commented and add a `readme.m` file describing how to run your code. You could use MATLAB publisher to print out your solutions. This allows the comments and the plots to appear inline as they are in the MATLAB script.
2. You will need the Multi-Parametric Toolbox in Exercise 1. Follow the installation instructions here: `https://www.mpt3.org/Main/Installation`

Exercise 1 **Programming Exercise**

Consider the second-order system with sampling time $T = 0.1s$:

$$\begin{aligned} x(k+1) &= \begin{pmatrix} 0.7115 & -0.4345 \\ 0.4345 & 0.8853 \end{pmatrix} x(k) + \begin{pmatrix} 0.2173 \\ 0.0573 \end{pmatrix} u(k) \\ y(k) &= \begin{pmatrix} 0 & 1 \end{pmatrix} x(k) \end{aligned} \tag{1}$$

Your task is to develop an MPC controller that regulates the system to the origin while fulfilling the input constraint

$$-5 \leq u(k) \leq 5, \quad \forall k,$$

and in a second step you will also need to make sure that your controller fulfils the additional state constraint:

$$0 \leq x_2(k), \quad \forall k.$$

Let us first refresh the definition of a quadratic programming based MPC controller for linear systems. The optimization problem used in an MPC controller is given by

$$\min_{\mathbf{u}} \sum_{k=0}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k) + x_N^\top P x_N \tag{2a}$$

$$\text{s.t.} \, u_{min} \leq u_k \leq u_{max} \tag{2b}$$

$$x_{k+1} = A x_k + B u_k \tag{2c}$$

$$x_{min} \leq x_k \leq x_{max} \tag{2d}$$

where $x_0 = x(k)$, and the matrices A and B are as defined in (1). In this exercise, the matrices Q, R and P are treated as design variables. To simplify the implementation, it helps to introduce a vectorized notation.

$$\begin{aligned} \mathbf{x} &= \begin{pmatrix} x_0^\top & x_1^\top & x_2^\top & \dots & x_N^\top \end{pmatrix}^\top \\ \mathbf{u} &= \begin{pmatrix} u_0^\top & u_1^\top & \dots & u_{N-1}^\top \end{pmatrix}^\top \end{aligned}$$

With these definitions, we can write the state sequence as [1]

$$\mathbf{x} = \mathcal{A}x_0 + \mathcal{B}\mathbf{u}$$

where

$$\mathcal{A} = \begin{pmatrix} I \\ A \\ A^2 \\ \vdots \\ A^N \end{pmatrix}, \quad \mathcal{B} = \begin{pmatrix} 0 & \dots & \dots & 0 \\ B & 0 & \dots & 0 \\ AB & B & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1}B & \dots & AB & B \end{pmatrix}$$

.

By introducing block-diagonal matrices $\mathcal{Q}$ and $\mathcal{R}$, the quadratic cost can be written as $\mathbf{x}^\top \mathcal{Q}\mathbf{x} + \mathbf{u}^\top \mathcal{R}\mathbf{u}$.

1. **[2 pt]** What do $\mathcal{Q}$ and $\mathcal{R}$ look like?
   Hint: Do not forget the terminal state weight.

2. **[4 pt]** The first goal in this exercise is to write the whole MPC problem in a format that can be sent to a quadratic programming solver (such as `quadprog`). To do this, expand the expression for the optimization cost $\mathbf{x}\mathcal{Q}\mathbf{x} + \mathbf{u}^\top \mathcal{R}\mathbf{u}$ into the format $\frac{1}{2}\mathbf{u}^\top H\mathbf{u} + x_0^\top F\mathbf{u} + \frac{1}{2}x_0^\top Y x_0$. Derive the matrices H, F and Y in terms of Q, $\mathcal{A}$, $\mathcal{B}$, $\mathcal{Q}$ and $\mathcal{R}$. Neglect the state constraints x_{min} and x_{max} for now.
   Hint: See slide 26 of MPC chapter 6.

3. **[2 pt]** Is the term $\frac{1}{2}x_0^\top Y x_0$ needed to solve the optimization problem? Why or why not?

4. **[12 pt]** The next exercise requires some advanced MATLAB programming. Write a MATLAB function `u = mympc(A,B,Q,R,P,N,umin,umax,xmin,xmax,x)`, i.e. a function which takes an arbitrary system (you may assume single input-single output), cost matrices of suitable dimensions, a desired prediction horizon, control constraints, state constraints (box-constraints), the current state, and returns the optimal control input. The MATLAB hints at the end should be enough to get you through the tricky parts. For your and the assistant's sanity, try to structure your code by using sub-functions or similar concepts. For the next couple of questions we will neglect the state constraints (you can set them to xmin=$[-\infty, -\infty]^T$ and xmax=$[+\infty, +\infty]^T$).

5. **[2 pt]** Calculate a terminal state weight-matrix P_L by solving the discrete Lyapunov equation (`help dlyap`) and an alternative weight-matrix P_R by solving the associated LQR problem (`help dlqr`) (please read the associated pages in the script now). What does the choice of P_L or P_R for the terminal weight P imply for u_k, $k \geq N$? Which of the two matrices has the larger 2−norm (i.e. largest maximum singular value)? Why?

6. **[6 pt]** You are now supposed to actually use your MPC controller. Start with, e.g. $Q = I$, $R = 1$ and $N = 5$ and your favourite terminal state weight. Write a simple loop to simulate the (discrete-time) closed-loop behavior from the initial condition $[0\ 10]^\top$. Plot both the states and the optimal input on the same figure for the first 5 s of the simulation. Are the results satisfactory? Is the prediction horizon reasonable? Try to tune the MPC controller to reduce the oscillations and the over-shoot.

[1] Note that we are using a formulation where we only keep $\mathbf{u}$ as optimization variables. An alternative formulation that also keeps $\mathbf{x}$ as optimization variables can be found in the lecture notes. Feel free to implement the alternative formulation as a separate exercise at home.

7. **[4 pt]** Reduce your prediction horizon to $N = 1$ and check if there is any major degradation in performance. Now keep the short prediction horizon but change the terminal state weight to Q, i.e. $P = Q$. How does the MPC controller perform with a (too) short horizon and this choice of the terminal state weight?

8. **[4 pt]** Compute the optimal input for the state $[0\ 10]^\top$ using the LQR based terminal state weight, $Q = 100I$, $R = 1$ and $N = 2$ (make sure to use the same weights in the MPC controller and in the computation of P_R and the associated LQR controller). Compare the optimal control input to the control input that an LQR controller would give at the same state. Do the same thing when you use the terminal state weight based on the zero input assumption (i.e. the terminal state weight computed using the Lyapunov equation). Now change the state to $[0.1\ 0.1]^\top$ and repeat the procedure. Observations?

9. **[4 pt]** For the last task of the programming exercise you will need to implement state constraints. For `quadprog` you need to bring the state constraints into the form $A_{ineq}\mathbf{u} \leq \mathbf{b}$. *Hint:* See slide 28 of MPC chapter 6.

10. **[4 pt]** Take the parameters from task 6. Consider as an additional constraint, that there should be no undershoot in x_2 and start from an initial condition $[0\ 6]^\top$. What happens if you start from an initial condition $[0\ 10]^\top$?

11. **[8 pt]** We want to prove asymptotic stability of the MPC control law generated by (2). For simplicity we assume there exists a terminal set $\mathcal{X}_f$ and constraint $x_N \in \mathcal{X}_f$. Additionally, assume there exist a linear state feedback controller K, such that $v(x_N) = Kx_N$ (see slide 37 of MPC chapter 7).

    i) State the implicit conditions on $\mathcal{X}_f$, P, Q, R and K, so that we can prove asymptotic stability in this general setting.

    ii) Show that under these assumptions, $J_0^*(x_0)$ is a Lyapunov function decreasing along the closed loop trajectories

    Hint: See slides 34-38 of MPC chapter 7.

12. **[8 pt]** For the last task, we consider the system,

$$x(k+1) = \begin{pmatrix} \cos(0.5) & -\sin(0.5) \\ \sin(0.5) & \cos(0.5) \end{pmatrix} x(k) + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} u(k)\,, \tag{3}$$

    with inputs constraints of the from

$$\mathcal{U} = \{u = (u^{(1)}, u^{(2)}) | -5 \leq u^{(i)} \leq 5, \quad i = 1, 2\}\,.$$

    The goal of this exercise is to design a state feedback controller $u = Kx$, such that the set,

$$\mathcal{X}_f = \{x = (x^{(1)}, x^{(2)}) | -10 \leq x^{(i)} \leq 10, \quad i = 1, 2\}\,,$$

    is an invariant set of the closed loop system under the given actuator constraints. Please compute all values of K, which satisfy the given constraints.

    Hint 1: You may restrict your attention to diagonal feedback matrix of the from,

$$K = \begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix}\,.$$

    Hint 2: Argue why it is sufficient to only check the corner points of the set $\mathcal{X}_f$.

MATLAB hints

To create a block-diagonal matrix with blocks X and Y, use `Z = blkdiag(X,Y)`. If you want to create a matrix with several blocks, use the command repeatedly. A more general and very useful command is the function `kron`. To create a matrix with Q repeated N times on the diagonal, use `kron(eye(N),Q)`. To stack the matrices over each other instead, use `kron(ones(N,1),Q)`.

The matrices $\mathcal{A}$ and $\mathcal{B}$ also appeared in the previous exercise. Try to reuse your code !

Additionally, the commands `ones`, `zeros`, `size` and `repmat` might turn out to be useful. Finally, do not forget the command `help`.

Exercise 2 **Batch Approach**

Consider the double integrator

$$y(t) = \frac{1}{s^2} u(t) \tag{4}$$

and its equivalent discrete-time state-space representation

$$\begin{cases} x(t+1) &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \end{cases} \tag{5}$$

obtained by setting $\ddot{y}(t) = \frac{\dot{y}(t+T)-\dot{y}(t)}{T}$, $\dot{y}(t) = \frac{y(t+T)-y(t)}{T}$, $T = 1s$.
Define the following cost function

$$J(U_0, x(0)) \triangleq x_N' P x_N + \sum_{k=0}^{N-1} x_k' Q x_k + u_k' T u_k \tag{6}$$

$N = 5$, $Q = eye(2)$, $R = 0.1$, $P = eye(2)$.
The constraints are

$$-1 \leq u(k) \leq 1 \quad k = 0, 1, \cdots, 5; \tag{7a}$$

$$\begin{bmatrix} -15 \\ -15 \end{bmatrix} \leq x(k) \leq \begin{bmatrix} 15 \\ 15 \end{bmatrix} \quad k = 0, 1, \cdots, 5; \tag{7b}$$

1. **[14 pt]** Select 100 initial conditions x_0 (by gridding the space of initial conditions with 10×10 points) and compute the solution to the finite time optimization control problem for each x_0 ($\in 8b$) using `quadprog` and the batch approach. Plot $u_0^\star(x_0)$ at the grid points.
2. **[6 pt]** Report the solution $[u_0^\star; u_1^\star; u_2^\star; u_3^\star; u_4^\star]$ for $x_0 = [1; 1]$ and $x_0 = [-1; -1]$.