ACS6116 Advanced Control (Model Predictive Control)

Laboratory Exercises 2019–20

Dr Paul Trodden
p.trodden@shef.ac.uk
Room C10, AJB

This document describes the laboratory exercises for ACS6116 Advanced Control. The sections in this document align with different aspects/topics of the module content. Each section contains a number of structured exercises. Exercises are indicated by a box around the text. Read all instructions carefully, and attempt all exercises.

In general, each set of exercises begins with smaller problems of a technical or factual nature, and ends with more challenging for which deeper understanding and some thought or discussion is required.

While these exercises are not assessed, and no report is to be submitted, you will find that completing these exercises is necessary in order to attempt the assignment for this part of the module.

You have 12 hours of supervised lab time to support these exercises, but please note that it may take you less time or more time than this to complete everything.

1 Using state-space models to make predictions

Consider a linear discrete-time system

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned}$$

where

- the state x is an $n \times 1$ vector;
- the input u is an $m \times 1$ vector;
- the output y is a $p \times 1$ vector;
- A is an $n \times n$ matrix, B is $n \times m$ and C is $p \times n$.

Starting from an initial state $x(k)$ at time k, this model may be applied recursively to obtain a prediction of what the state will be at time $k+i$ when the sequence of inputs applied between times k and $k+i$ is

$$\{u(k|k), u(k+1|k), \ldots, u(k+i-1|k)\}$$

In particular, the i-step ahead state prediction is

$$x(k+i|k) = A^i x(k) + A^{(i-1)}Bu(k|k) + A^{(i-2)}Bu(k+1|k) + \cdots + Bu(k+i-1|k)$$

Collecting predictions over all steps $i = 1, \ldots, N$ ahead from k, we obtain the prediction equation

$$\mathbf{x}(k) = Fx(k) + G\mathbf{u}(k). \tag{1}$$

In this equation, the boldface variables $\mathbf{x}(k)$ and $\mathbf{u}(k)$ are, respectively, the stacked vectors of state predictions and future inputs; the normalface $x(k)$ is the initial state from which the predictions start. F and G are larger matrices formed from the matrices A and B. In particular:

$$\mathbf{x}(k) \triangleq \begin{bmatrix} x(k+1|k) \\ x(k+2|k) \\ \vdots \\ x(k+N|k) \end{bmatrix}, \quad \mathbf{u}(k) \triangleq \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+N-1|k) \end{bmatrix}, \quad F = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \quad G = \begin{bmatrix} B & 0 & \ldots & 0 \\ AB & B & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \ldots & B \end{bmatrix}.$$

Exercises

1.1: Download the MATLAB function `predict_mats.m`, which computes the prediction matrices, F and G, given system matrices (A, B) and prediction horizon length N.

Use the function to obtain F and G for the following system.

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 1 & 1.5 \\ 0 & 0.5 \end{bmatrix} x(k) + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u(k), \\ y(k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(k) \end{aligned} \tag{2}$$

with a horizon of $N = 6$.

1.2: Given an initial state $x(0) = [1, 0]^\top$, obtain the state predictions for $N = 6$ when the control sequence $\mathbf{u}(0) = [-0.25, 0, 0, 0, 0, 0]^\top$ is applied to the system (2).

Plot the state predictions as

a) a **time series** plot of both $x_1(0+i|0)$ versus $i = 0 \dots N$, and $x_2(0+i|0)$ versus $i = 0 \dots N$, on the same axes.

b) a **phase** plot of $x_2(0+i|0)$ versus $x_1(0+i|0)$ for $i = 0 \dots N$.

Hint: The predicted states are obtained from $\mathbf{x}(0) = Fx(0) + G\mathbf{u}(0)$, and are provided in the form of an $(Nn) \times 1$ vector, containing $x(0+1|0), x(0+2|0), \dots, x(0+N|k)$. Therefore, the predicted state $x(0+i|0)$, which is an $n \times 1$ vector, is contained in elements $((i-1)n+1)$ to $((i-1)n+n)$. You need to extract and plot each of the $n = 2$ elements of $x(0+j|0)$ for steps $i = 1, \dots, N$; the odd elements of x are the x_1 predictions and the even elements the x_2 predictions. Finally, don't forget to augment $\mathbf{x}(0)$ with the initial state before plotting!

1.3: (Harder) Find the sequence $\mathbf{u}(0) = \{u(0|0), u(1|0)\}$ that steers the state of (2) from $x(0) = [3, -1]^\top$ to $x_f = [-1, 4]^\top$ in $N = n = 2$ prediction steps.

2 Setting up the MPC optimization problem

The MPC objective function is

$$V_N\big(x(k), \mathbf{u}(k)\big) = \sum_{i=0}^{N-1} \big(x^\top(k+i|k)Qx(k+i|k) + u^\top(k+i|k)Ru(k+i|k)\big) + x^\top(k+N|k)Px(k+N|k) \tag{3}$$

where N is the prediction horizon length, and Q, R and P are, respectively, $n \times n$, $m \times m$ and $n \times n$ matrices. The unconstrained MPC optimization problem at a state $x(k)$ at time k is then to minimize this objective function, assuming the linear prediction model holds:

$$V_N^0\big(x(k)\big) = \min_{\mathbf{u}(k)} V_N\big(x(k), \mathbf{u}(k)\big)$$

subject to, for $i = 0, \dots, N-1$,

$$\begin{aligned} x(k+i+1|k) &= Ax(k+i|k) + Bu(k+i|k), \\ x(k|k) &= x(k). \end{aligned}$$

By stacking variables into vectors, and using the prediction equation (1) to eliminate dependent variables, this optimization problem may be rewritten in the following standard QP form:

$$\min_{\mathbf{u}(k)} \frac{1}{2}\mathbf{u}^\top(k)H\mathbf{u}(k) + x^\top(k)L^\top\mathbf{u}(k) + x^\top(k)Mx(k), \tag{4}$$

where

$$H = 2(G^\top \tilde{Q} G + \tilde{R}), \qquad L = 2G^\top \tilde{Q} F, \qquad M = F^\top \tilde{Q} F + Q,$$

and

$$\tilde{Q} = \begin{bmatrix} Q & 0 & \dots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & Q & 0 \\ 0 & \dots & 0 & P \end{bmatrix}, \qquad \tilde{R} = \begin{bmatrix} R & 0 & \dots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & R & 0 \\ 0 & \dots & 0 & R \end{bmatrix}. \tag{5}$$

Exercises

2.1: Download the MATLAB function `cost_mats.m`, which calculates the cost matrices H, L and M given matrices Q and R, and the prediction matrices F, G.

Use the function to obtain the cost matrices for the system (2), using $N = 6$, $Q = C^\top C$, $R = 1$ and $P = Q$.

2.2: Use `quadprog` in MATLAB to solve the unconstrained MPC problem for the system (2) with $N = 6$, $Q = C^\top C$, $R = 1$, $P = Q$ and an initial state $x(0) = [1, 0]^\top$. Hence, obtain the optimal sequence $\mathbf{u}^0(0)$ and the value function at $x(0)$, $V_N^0\big(x(0)\big)$, for this system.

Hint: The command `[zopt,cost] = quadprog(H,f)` solves the problem

$$\min_{\mathbf{z}} \frac{1}{2}\mathbf{z}^\top H \mathbf{z} + f^\top \mathbf{z}$$

and returns the minimizer $\mathbf{z}^0$ and minimum cost $0.5(\mathbf{z}^0)^\top H \mathbf{z}^0 + f^\top \mathbf{z}^0$. Note that this minimum cost is different to $V_N^0\big(x(0)\big)$. (Why?)

2.3: Vary the weighting matrix R, keeping $Q = I$, and compare the resulting optimized control sequences and associated state predictions. (You may find it helpful to construct two plots—one of the control sequence and one of the state predictions—and use the `hold` function to compare solutions for different R). The aim of this exercise is for you to see the effect of relative weights of the matrices Q and R on the optimized predictions.

3 The unconstrained LQ regulator

It can be shown that, in the absence of constraints, the optimal solution to the unconstrained MPC problem is given by

$$\mathbf{u}^0(k) = -H^{-1} L x(k), \tag{6}$$

where H and L are the cost matrices in (4). Since H and L are constant, this optimal solution defines a state feedback policy that is independent of time and the initial state; i.e., the optimal control sequence is always obtained as $-H^{-1}L$ times the initial state, regardless of what the initial state is. The corresponding state predictions are then

$$\begin{aligned} \mathbf{x}^0(k) &= Fx(k) + G\big(-H^{-1}Lx(k)\big) \\ &= \big(F - GH^{-1}L\big)x(k). \end{aligned} \tag{7}$$

In the receding-horizon framework of MPC, the first control in the optimized sequence is applied to the system, and the optimization problem is subsequently re-solved. However, since the optimal solution is always the same linear function of the state, (6), the unconstrained MPC control law is the time-invariant state feedback control law

$$u(k) = K_N x(k)$$

where K_N is obtained by taking the first m rows of the matrix $-H^{-1}L$, where m is the dimension of the input vector u. The resulting closed-loop system is

$$x(k+1) = (A + BK_N)x(k). \tag{8}$$

The most important consequence of this is that we can implement unconstrained MPC without solving a single optimization problem: given the system matrices and objective matrices, we can compute (offline) cost matrices H and L, determine the optimal MPC feedback matrix K_N as the first m rows of $-H^{-1}L$, and apply the optimal controller as $u(k) = K_N x(k)$ at every time step.

However, as this exercise aims to show, an interesting consequence of employing a finite, rather than infinite, prediction horizon is that the state trajectory of the closed-loop system (8) does not necessarily match the predictions (7), even when the model of the system used is perfect.

Exercises The system

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(k) \end{aligned} \tag{9}$$

is to be controlled via unconstrained model predictive control. The objective function is

$$V_N\big(x(k), \mathbf{u}(k)\big) = \sum_{i=0}^{N-1} y^2(k+i|k) + u^2(k+i|k)$$

3.1: For an initial state $x(0) = [3, 0]^\top$, use (6) to compute the optimal control sequence $\mathbf{u}^0(0)$ for a horizon of $N = 10$. Verify that the answer is the same as that given by solving the MPC problem via `quadprog.m`.

3.2: Using $\mathbf{u}^0(0)$ in (1) or the expression (7), obtain the **open-loop** state predictions $\mathbf{x}^*(0)$.

Produce two plots: one of the optimal predictions $u(0+i|0)$ against prediction step i, and the other of the associated $x(0+i|0)$ against prediction step i. For the plot of $u(0+i|0)$, use the `stairs` command (rather than `plot`) to reflect the fact that the control input is adjusted by the controller in a stepwise manner.

3.3: Using (8), simulate the **closed-loop** system under the MPC control law $u(k) = K_{10}x(k)$, starting from $x(0) = [3, 0]^\top$, to obtain the actual controls $u(k)$ and states $x(k)$ for $k = 0$ to 10. Add these to your plots of open-loop predictions. Does the closed-loop system follow the predictions? If not, why not — can you give an explanation?

Hint: Use a for loop to simulate the system and collect the states and inputs, e.g.

```
% get MPC control law
K = ?;   % see (7) and (8) in the preamble

% set number of simulation steps
nk = 10; % same as horizon for now, but does not have to be

% initialize
x = x0;
Acl = (A+B*K);
xs = zeros(n,nk+1);
us = zeros(m,nk+1);
```

```
% loop
for k = 1:nk+1
    % store
    xs(:,k) = x;
    us(:,k) = K*x;

    % move
    x = Acl*x;
end
```

3.4: Repeat for $N = 5$ and $N = 15$, comparing closed-loop trajectories with predictions. What do you notice?

3.5: Compute the infinite-horizon LQR controller, K_∞, via

```
K_inf = -dlqr(A,B,Q,R);
```

Compare K_∞ to K_5, K_{10} and K_{15}. What conclusions can be made about choosing the prediction horizon in unconstrained MPC?

3.6: Keeping Q the same and $P = 0$, investigate how the remaining controller parameters—R and N—affect the settling time of the closed-loop system. Your aim is to determine whether it is N or R that should be used to tune the closed-loop response.

4 Stability and terminal costs

As seen in the previous section, a consequence of using a finite prediction horizon is that the closed-loop system may not behave as predicted. If the horizon is very short, the closed-loop system may even lose stability.

The problem goes away if one uses an infinite horizon, minimizing the objective function

$$\sum_{i=0}^{\infty} x^\top(k+i|k)Qx(k+i|k) + u^\top(k+i|k)Ru(k+i|k),$$

However, the obvious drawback is complexity: the optimization problem would have, in theory, an infinite number of decision variables and constraints.

To avoid untractable complexity yet retain the advantage of an infinite prediction horizon, a useful and common approach is to employ a *stabilizing* terminal cost in the finite-horizon objective function. Recall the MPC objective function

$$V_N\big(x(k), \mathbf{u}(k)\big) = \sum_{i=0}^{N-1} \big(x^\top(k+i|k)Qx(k+i|k) + u^\top(k+i|k)Ru(k+i|k)\big) + \underbrace{x^\top(k+N|k)Px(k+N|k)}_{\text{terminal cost}}.$$

The idea is to set the matrix P so that the terminal cost approximates the cost of what will happen beyond the horizon. That is,

$$x^\top(k+N|k)Px(k+N|k) = \underbrace{\sum_{i=N}^{\infty} x^\top(k+i|k)Qx(k+i|k) + u^\top(k+i|k)Ru(k+i|k)}_{\text{cost-to-go from } N \text{ to } \infty}. \tag{10}$$

The surprising fact is that we can find a P that satisfies (10) very easily. First, we assume that, beyond the horizon, our system will be controlled by a known stabilizing control law $u = Kx$. (We can find such a K using any suitable method[1]—we just need to ensure that all eigenvalues

[1] Assuming (A, B) is stabilizable.

of $(A + BK)$ are within the unit circle.) Then, given K, the matrix P is the unique solution[2] to the Lyapunov equation

$$(A + BK)^\top P(A + BK) - P + (Q + K^\top RK) = 0. \tag{11}$$

The resulting unconstrained MPC controller, using cost matrices Q, R and P, is then (under mild conditions on (A, B, Q, R)) guaranteed to be stabilizing for an horizon length N.

Exercises

4.1: Show that the closed-loop system (9) controlled by unconstrained MPC with $N = 3$ and terminal cost matrix $P = 0$ is unstable. (Use the same set of parameters—*i.e.* Q and R—that you used in Exercise 3, but make sure to set P to a zero matrix.)

Hint: Analyse the closed-loop eigenvalues.

4.2: Still using $P = 0$, simulate the closed-loop system under the controller K_3, from an an initial state $x(0) = [3, 0]^\top$, and plot the value function $V_N^0\big(x(k)\big)$ against k for $k = 0 \ldots 10$. What do you notice?

Hint: The value function $V_N^0\big(x(k)\big)$ is the objective value of the optimal solution to the MPC problem (the minimum cost) at state $x(k)$; i.e.,

$$\begin{aligned} V_N^0\big(x(k)\big) &= \min_{\mathbf{u}(k)} \sum_{i=0}^{N-1} \big(x^\top(k+i|k)Qx(k+i|k) + u^\top(k+i|k)Ru(k+i|k)\big) \\ &\qquad + x^\top(k+N|k)Px(k+N|k) \\ &= \min_{\mathbf{u}(k)} \frac{1}{2}\mathbf{u}^\top(k)H\mathbf{u}(k) + x^\top(k)L^\top\mathbf{u}(k) + x^\top(k)Mx(k) \end{aligned}$$

4.3: Show that the feedback matrix $K = [-2 \ -6]$ is stabilizing for (A, B). Therefore, using this K, solve the Lyapunov equation (11) to obtain the terminal cost matrix P.

Hint: The MATLAB function `P = dlyap(Phi',S)` solves the Lyapunov equation

$$\Phi^\top P\Phi - P + S = 0$$

By comparing this with (11), you can define suitable Φ and S—in terms of A, B, K—and compute P.

4.4: Show that using a stabilizing terminal cost in the MPC objective now results in a stable closed-loop system, even with $N = 3$.

Hint: Re-compute K_3 with your new P, and analyse the closed-loop eigenvalues.

4.5: Simulate the closed-loop system with the new K_3, and plot the value function against k for $k = 0 \ldots 10$. What do you notice now?

4.6: Using the same values of Q and R, compute K_∞ (using `dlqr.m`) and now use this as the mode-2 control law K in order to compute a new terminal matrix P. What is K_N, the mode-1 MPC control law, using this new P?

4.7: Using this new P, and the same Q as before, investigate the effect of R and N on settling time. As before, your aim is to determine whether R or N should be used to tune the closed-loop response.

[2] Unique iff the eigenvalues of $(A + BK)$ are strictly within the unit circle.

5 Constrained MPC

The most useful and important feature of MPC is its ability to handle constraints. Constraints exist on almost all systems, often in the form of actuator saturation or rate limits, but also sometimes state or output constraints are imposed for reasons of safety, performance or economy.

We will consider upper and lower bounds on states and inputs,

$$\begin{aligned} x_{\min} &\leqslant x(k) \leqslant x_{\max}, \\ u_{\min} &\leqslant u(k) \leqslant u_{\min}, \end{aligned}$$

for all k. Imposing these constraints over the horizon, the constrained MPC optimization problem is then

$$\min_{\mathbf{u}(k)} \sum_{k=0}^{N-1} \left(x^\top(k+k|k) Q x(k+i|k) + u^\top(k+i|k) R u(k+i|k) \right) + x^\top(k+N|k) P x(k+N|k)$$

subject to, for $i = 0, \ldots, N-1$,

$$\begin{aligned} x(k+i+1|k) &= Ax(k+i|k) + Bu(k+i|k) \\ x(k|k) &= x(k) \\ u_{\min} &\leqslant u(k+i|k) \leqslant u_{\min}, \\ x_{\min} &\leqslant x(k+i|k) \leqslant x_{\max}, \\ x_{\min,N} &\leqslant x(k+N|k) \leqslant x_{\max,N}, \end{aligned}$$

where the latter inequalities represent terminal state constraints, which may (or may not) differ from the normal state constraints.

By using the prediction equation (1), this may be rewritten as a constrained QP:

$$\min_{\mathbf{u}(k)} \frac{1}{2} \mathbf{u}^\top(k) H \mathbf{u}(k) + x^\top(k) L^\top \mathbf{u}(k) + x^\top(k) M x(k).$$

subject to

$$P_c \mathbf{u}(k) \leqslant q_c + S_c x(k),$$

where H, L, M are the same matrices as in the unconstrained case, and P_c, q_c, S_c apply the state and input limits.

Exercises

5.1: Download the MATLAB function `constraint_mats.m`, which computes the matrices P_c, q_c and S_c given the input, state and terminal constraints in inequality form, the prediction matrices F and G, and the horizon length N.

Use the function to obtain the matrices P_c, q_c and S_c for the system (9) with $N = 3$ and subject to constraints

$$\begin{bmatrix} -10 \\ -5 \end{bmatrix} \leqslant x \leqslant \begin{bmatrix} +10 \\ +5 \end{bmatrix}$$
$$-1.5 \leqslant u \leqslant +1.5$$

5.2: In this exercise, you are to obtain the constrained MPC control law $u(k) = \kappa_N\big(x(k)\big)$, simulate the closed-loop system, and investigate its stability and feasibility.

Consider system (9) again, subject to the constraints in the previous part, and assume the same parameters that you used in Exercise 4. That is,

- Cost matrices $Q = C^\top C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ and $R = 1$.
- Mode-2 controller $K = [-2 \ -6]$ with associated stabilizing terminal cost matrix $P = \begin{bmatrix} 22 & 29 \\ 29 & 53 \end{bmatrix}$.
- A prediction horizon of $N = 3$.
- An initial state of $x(0) = [3, 0]^\top$.

a) Simulate the first 15 steps of the closed-loop system under the constrained control law $u(k) = \kappa_3\big(x(k)\big)$, and plot the controls $u(k)$ and states $x(k)$ for $k = 0$ to 15. Compare your plots with the same ones obtained for the unconstrained controller $u = K_3 x$ (in Exercise 4). What happens? Are constraints satisfied? Is stability maintained?

b) Show that, if the simulation is allowed to continue for a few more steps, the MPC optimization becomes infeasible. Which constraint do you think is violated?
You may wish to make use of the following code snippet:

```
% solve the constrained QP
[Uopt,fval,flag] = quadprog(H,L*x,Pc,qc+Sc*x);

% check feasibility
if flag < 1
    disp(['Optimization infeasible at k = ' num2str(k)])
    break
end
```

c) Show that if the state constraints are removed, the system no longer loses feasibility but the closed-loop system is unstable.
Hint: It is sufficient to simulate, say, 50 steps of the closed-loop system and observe what happens to the states.

d) Repeat the previous exercise (with or without state constraints) with an initial state of $x(0) = [2, 0]^\top$. Is the system stable? Hence, carefully describe the kind of stability obtained for this system.

e) Show that, with the state constraints reinstated, the system can be stabilized from $x(0) = [3, 0]^\top$, while remaining feasible, by increasing the prediction horizon. What is the smallest value of N that achieves stability while all other parameters remain the same?

f) Show that, with the state constraints present, if the input constraints are relaxed to

$$-1.75 \leqslant u \leqslant +1.75$$

then then system can be stabilized from $x(0) = [3, 0]^\top$, while remaining feasible, with a horizon of $N = 3$.

g) Can you stabilize the system, starting again from an initial state $x(0) = [3, 0]^\top$, if the input constraint is changed to

$$-1 \leqslant u \leqslant 1$$

If so, how? Can you find a stabilizing controller with $N = 5$?

h) For this kind of MPC (LQ-MPC with a stabilizing terminal cost) it is claimed that *feasibility implies stability*. Is this true or false?

i) Add the constraint $x(k+N|k) = 0$ to the MPC formulation, and investigate the feasibility and stability of the system from a range of initial states. Does feasibility imply stability? Can you estimate the region of attraction for the closed-loop system?

Hint: Run `help constraint_mats`—this will tell you how to impose separate constraints on the terminal state.