KTH Electrical Engineering

EL2700 - Model Predictive Control

Exercise Compendium

LINNEA PERSSON, MARTIN BIEL, JEZDIMIR MILOSEVIC, PEDRO LIMA, VALERIO TURRI
MIKAEL JOHANSSON

Stockholm August 30, 2018

Automatic Control
School of Electrical Engineering
Kungliga Tekniska Högskolan

Contents

Part I
Exercises

1 Discrete-time linear systems

Exercise 1.1 Discrete-time responses

Consider the following discrete-time system

$$x_{t+1} = \begin{bmatrix} 0.75 & 0.1 \\ 0 & 0.5 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 2 \end{bmatrix} u_t$$
$$y_t = \begin{bmatrix} 1 & 2 \end{bmatrix} x_t.$$

(a) Let $u_t = 0$ for all t, and $x_0 = \begin{pmatrix} 1 & 1 \end{pmatrix}^T$. Compute y_2.

(b) Let $u_t = 5$ for all t, and $x_0 = \begin{pmatrix} 1 & 1 \end{pmatrix}^T$. Compute y_2.

(c) Does the system output y_t converge to some finite value as t tends to infinity? If so, find this value.

Exercise 1.2 Sampling linear systems

Discretize the following systems using the sample-and-hold method.

(a) The integrator

$$\dot{x}(t) = u(t)$$

(b) A first-order system with pole in $s = a$ and stationary gain K

$$\dot{x}(t) = ax(t) + aKu(t)$$

(c) The simple harmonic oscillator

$$\ddot{x}(t) = -\omega^2 x(t) + u(t)$$

(d) The double integrator

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

Exercise 1.3 Discretization of a low-pass filter

Consider the continuous single-input single-output system described by

$$\dot{x}(t) = ax(t) + u(t)$$

where $a < 0$ is a system parameter. Let the discretization time be h.

(a) Discretize the system using explicit Euler method, implicit Euler method, and zero-order-hold method.
Hint: The explicit Euler method consist of approximating $\dot{x}(t)$ with

$$\dot{x}(t) \approx \frac{x(t+h) - x(t)}{h}$$

while in the implicit Euler method, $\dot{x}(t)$ is approximated with

$$\dot{x}(t) \approx \frac{x(t) - x(t-h)}{h}.$$

(b) How does the position of the pole depend on the discretization time in the two discretized systems?

(c) When are the systems stable?

Exercise 1.4 Pole locations

Consider the transfer function

$$G(s) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \tag{1}$$

where ω_0 is the natural frequency that characterizes the response time, and ζ is the damping that quantifies how oscillatory the response is.

(a) Verify that the system $Y(s) = G(s)U(s)$ with $G(s)$ given by (1) admits a state-space representation

$$\begin{aligned} \dot{x}(t) &= \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \omega_0^2 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \end{aligned}$$

(b) Write a short program in Matlab that takes ω_0, ζ and h as input and computes the matrices A, B and C of the corresponding discrete-time system.

(c) Use the program to draw how the poles location in the continuous system maps in the poles location in the discrete system.

(i) Consider first the particular system given by $\omega_0 = 1$ and $\zeta = 0$. Plot the discrete-time system poles when h varies. What happens when $h = \pi$? Can you explain?

(ii) Set $\zeta = 0$ and $h = 0.1$, draw the continuous and discrete-time pole locations (in different plots) as ω_0 varies. Interpret the result.

(iii) Fix $\omega_0 = 1$. For each value of the sampling time h in $\{0.1, 0.2, 0.5, 1\}$ plot the continuous and discrete-time pole locations as ζ varies from -1 to 1. Interpret the result.

Exercise 1.5 Discrete-time stability

Consider the systems

$$
\begin{aligned}
(i) \qquad x_{t+1} &= \begin{bmatrix} 1 & 0.1 \\ 0 & 2 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_t \\
y_t &= \begin{bmatrix} 1 & 0 \end{bmatrix} x_t \\
(ii) \qquad x_{t+1} &= \begin{bmatrix} -0.5 & -0.2 \\ 0.2 & -0.8 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_t \\
y_t &= \begin{bmatrix} 1 & 0 \end{bmatrix} x_t \\
(iii) \qquad x_{t+1} &= \begin{bmatrix} 0.5 & 0.1 \\ 0.2 & 0.2 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_t \\
y_t &= \begin{bmatrix} 1 & 0 \end{bmatrix} x_t.
\end{aligned}
$$

(a) Determine if the systems (i)–(iii) are stable.

(b) Pair the step responses shown in Figure 1 with the systems (i)–(iii).

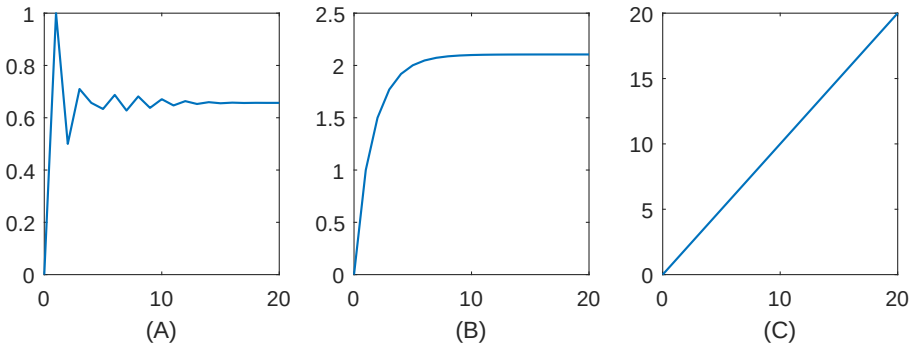


Figure 1: Step responses for the systems given in Exercise 1.5.

Exercise 1.6 Discrete-time reachability

In this exercise, we derive Popov-Belevitch-Hautus (PBH) test for reachability and investigate some of its implications.

(a) Prove the following statement: (A, B) is unreachable if and only if there exists $w \neq 0$ such that $w^T A = \lambda w^T$, $w^T B = 0$ through the following steps:

(i) First, assume that there is a $w \neq 0$, and show that this implies that the standard rank condition for reachability fails.

(ii) Next, suppose that (A, B) is not reachable. Then we can perform a coordinate transformation $z = Px$ with P invertible which reveals the unreachable subspace, *i.e.*, such that

$$
z(t+1) = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ 0 & \tilde{A}_{22} \end{bmatrix} z(t) + \begin{bmatrix} \tilde{B}_1 \\ 0 \end{bmatrix} u(t)
$$

Let $\tilde{w}$ be a right eigenvector $\tilde{A}_{22}$ and show that

$$w = P^T \begin{bmatrix} 0 \\ \tilde{w} \end{bmatrix}$$

satisfies the PBH conditions.

(b) Prove that state feedback does not alter reachability. Specifically, show that $(A - BL, B)$ is unreachable if and only if (A, B) is.

Exercise 1.7 Understanding reachability [5]

Consider the system

$$x_{t+1} = \begin{bmatrix} a & b \\ 0 & c \end{bmatrix} x_t + \begin{bmatrix} d \\ e \end{bmatrix} u_t,$$

where a, b, c, d and e are scalars.

(a) Deduce precisely what condition these coefficients satisfy when the system is not reachable. Use controllability matrix for this purpose.

(b) Let $a = 1, b = 2, c = 3, d = 0$, and $e = 1$. Investigate reachability of this system using PBH test.

(c) Draw a block diagram corresponding to the above system and use it to interpret the following special cases in which reachability is lost:
   (i) $e = 0$;
   (ii) $b = 0$ and $d = 0$;
   (iii) $b = 0$ and $c = a$.

Exercise 1.8 Observe the state [1]

From the system

$$x_{t+1} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_t,$$
$$y_t = \begin{bmatrix} 0 & 1 \end{bmatrix} x_t,$$

the following values are obtained $y_1 = 0, u_1 = 1, y_2 = 1, u_2 = -1$.

(a) Determine the value of the state at $t = 3$.

(b) Use PBH test to determine if the system is observable.

(c) Confirm the result obtained in (b) by using the observability matrix.

Exercise 1.9 Understanding observability [5]

(a) Given the *observable* discrete-time *n–th* order system

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

show that we can uniquely determine the initial condition $x(0)$ from output measurements alone, i.e. without knowledge of the inputs $u(i)$, if

$$D = 0 \qquad CB = 0 \qquad CAB = 0 \qquad \dots \qquad CA^{n-2}B = 0.$$

(b) Prove that the sufficient condition in (a) is also necessary in the case where the output y is scalar.

Exercise 1.10 Stabilization with state-feedback [5]

Let $(A, B, C, 0)$ be a reachable and observable linear-time invariant state-space description of a discrete-time system. Let its input u be related to its output y by the following output feedback law:

$$u(k) = Fy(k) + r(k)$$

for some constant matrix F , where r is a new external input to the closed-loop system that results from the output feedback.

(a) Write down the state-space description of the system mapping r to y.

(b) Is the new system reachable?

(c) Is the new system observable?

Exercise 1.11 Poles of state-feedback system

Consider the system

$$\begin{aligned} x_{t+1} &= \begin{bmatrix} 0.5 & 1 \\ 0.8 & 0.2 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t \\ y_t &= \begin{bmatrix} 0 & 1 \end{bmatrix} x_t \end{aligned}$$

(a) Calculate a state feedback $u_t = -Lx_t$ such that both of the system poles are located at 0.2.

(b) Calculate a state feedback $u_t = -Lx_t + l_r r_t$ such that one pole equals to 0.1, the other one to 0.3, and the static gain from r to y equals to 1.

Exercise 1.12 Poles of observer

Consider the system

$$\begin{aligned} x_{t+1} &= \begin{bmatrix} 0.2 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 2 \end{bmatrix} u_t \\ y_t &= \begin{bmatrix} 0 & 1 \end{bmatrix} x_t \end{aligned}$$

(a) Is it possible to place the poles of the observer at arbitrary locations? Motivate.

(b) Is it possible to place the poles of the observer at locations $\{0.2, 0.3\}$? In case it is, find the observer gain that achieves this.

(c) Is it possible to place the poles of the observer at locations $\{0.4, 0.5\}$? In case it is, find the observer gain that achieves this.

Exercise 1.13 Pole placement design Exam: 2016-10-22

In this problem, we will perform a simple pole-placement design of an observer-based controller for a system described by the continuous-time dynamics

$$\dot{x}(t) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x(t) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u(t)$$
$$y(t) = \begin{pmatrix} 1 & 0 \end{pmatrix} x(t)$$

(a) Sample the system with sampling time h to obtain an equivalent discrete-time state-space model (equivalent here means that the discrete-time model should exactly predict the state of the continuous system at sampling intervals, provided that the input is held constant between samples).

(b) You decide to sample your system with $h = 2$, which gives the discrete-time model

$$x(t+1) = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} x(t) + \begin{pmatrix} 2 \\ 2 \end{pmatrix} u(t)$$
$$y(t) = \begin{pmatrix} 1 & 0 \end{pmatrix} x(t)$$

Verify that the system is reachable, and determine a discrete-time state feedback control law

$$u(t) = -Lx(t)$$

that places the closed-loop poles in $z = 1/2$.

(c) Your controller cannot measure the full state vector, but only the output $y(t)$. You will therefore have to construct an observer which estimates the full state vector based on knowledge of $u(t)$ and $y(t)$. Compute the observer gains which place the poles of the observer error dynamics in $z = 1/4$.

Exercise 1.14 Pole placement design 2 Exam: 2016-12-19

Consider the linear system

$$x_{t+1} = Ax_t + Bu_t$$
$$y_t = Cx_t$$

with matrices

$$A = \begin{bmatrix} 0 & -1/2 \\ 3/2 & -2 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ b \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 2 \end{bmatrix}.$$

(a) For which values of the parameter b is the system open-loop stable?

(b) For which values of the parameter b is the system controllable?

(c) Let $b = 0$. Compute a state-feedback law $u_t = -Lx_t$ which places all system poles at the origin. After how many steps does the state, at most, arrive at the origin with this controller?

Exercise 1.15 Pole placement design 3 Exam: 2017-12-18

A process is described by the discrete-time linear system

$$x_{t+1} = \begin{bmatrix} 0.4 & 0.8 \\ 0.4 & 0 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t$$
$$y_t = \begin{bmatrix} 1 & 1 \end{bmatrix} x_t$$

(a) Prove that the open-loop system is asymptotically stable.

(b) Is the system controllable?

(c) Design a state feedback controller

$$u_t = -Lx_t$$

which places both closed-loop poles in $z = 0.5$.

(d) Is the system observable?

(e) Design an observer

$$\hat{x}_{t+1} = A\hat{x}_t + Bu_t + K(y_t - \hat{y}_t)$$
$$\hat{y}_t = C\hat{x}_t$$

which places one observer pole at $z = 0$ and the other at $z = -0.4$.

(f) Can you find an observer gain which places both observer poles at the origin? Explain!

Exercise 1.16 Stirred tank Exam: 2017-10-23

Consider the stirred tank shown in Figure 2. The tank is fed with two inflows with flow rates $F_1(t)$ and $F_2(t)$. Both feeds contain material with constant concentrations c_1 and c_2. It is assumed that the tank is stirred well, so that the concentration of the outgoing flow equals the concentration $c(t)$ in the tank.

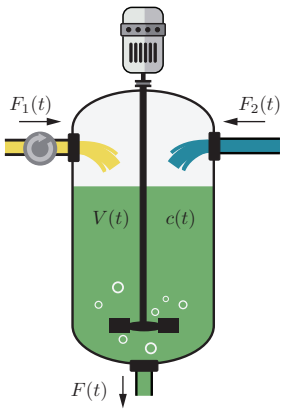


Figure 2: A stirred tank with controlled inflow F_1 and disturbance inflow F_2.

The tank dynamics can be described by the following linear system

$$\frac{dx(t)}{dt} = \begin{bmatrix} -\frac{1}{2\theta} & 0 \\ 0 & -\frac{1}{\theta} \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ \frac{c_1 - c_0}{V_0} \end{bmatrix} F_1(t) + \begin{bmatrix} 1 \\ \frac{c_2 - c_0}{V_0} \end{bmatrix} F_2(t)$$

As usual, all variables and inputs describe deviations from the equilibrium values V_0, c_0, F_{10} and F_{20}. The first state is the deviation in product volume present in the tank and the second state describes the difference between the actual material concentration in the tank and its equilibrium value. We will assume that $F_1(t)$ is under our control but consider $F_2(t)$ as a disturbance.

(a) Derive an exact discrete-time equivalent system under the assumption that the signals are sampled every T seconds and that the two inflows are held constant between sample instances.

(b) For a certain sampling interval and system parameters, the discrete-time equivalent is

$$x_{t+1} = \begin{bmatrix} 0.95 & 0 \\ 0 & 0.90 \end{bmatrix} x_t + \begin{bmatrix} 5 \\ -1 \end{bmatrix} F_t^{(1)} + \begin{bmatrix} 5 \\ 4 \end{bmatrix} F_t^{(2)}$$

Recall that $F_t^{(1)}$ is our control variable, while $F_t^{(2)}$ is a disturbance. Is the discrete-time system with $F^{(1)}$ as input controllable?

(c) Determine a linear state feedback

$$F_t^{(1)} = -Lx_t$$

which places the closed-loop poles in $z = -0.5$ and $z = -0.6$.

(d) Your main target is to keep the concentration

$$z_t = \begin{bmatrix} 0 & 1 \end{bmatrix} x_t$$

at its target value 0, despite a constant input disturbance $F_2(t) = 1$. However, when you try your controller from (c), you notice a stationary error.

(e) Show that by including an integral state in the controller, *i.e.* using

$$F_t^{(1)} = -Lx_t - l_i i_t$$

where the integral state satisfies

$$i_{t+1} = i_t + (r_t - y_t) = i_t - y_t$$

guarantees that $\lim_{t\to\infty} z_t = 0$ in steady-state.
You may assume that the feedback gains L and l_i are chosen so that the closed-loop system is asymptotically stable and that the steady-state is attained.

2 Stability and Invariance

2.1 Stability

Exercise 2.1 Lyapunov stability 1

Consider the nonlinear system

$$x_{k+1} = \frac{x_k}{1 + x_k^2}$$

(a) Verify that $x = 0$ is an equilibrium point for the system.

(b) Use the Lyapunov function $V(x) = x^2$ to show that the origin is globally asymptotically stable.

Exercise 2.2 Lyapunov stability 2

Consider the nonlinear system

$$x_{k+1} = \frac{1}{2}x_k + x_k^2$$

(a) Verify that $x = 0$ is an equilibrium point for the system.

(b) Use the Lyapunov function $V(x) = x^2$ to determine stability of the origin. Is the origin globally asymptotically stable? If not, determine a region of attraction for the origin.

Exercise 2.3 Lyapunov stability 3

Consider the nonlinear system

$$x_{k+1} = \frac{2x_k}{1 + x_k^2}$$

(a) Verify that $x = 0$ is an `unstable` equilibrium point for the system.

(b) Use the Lyapunov function $V(x) = x^2$ to show that all trajectories remain bounded.

Exercise 2.4 Lyapunov stability 4

Consider the nonlinear system

$$x_{k+1} = \frac{1}{2}\sin x_k$$

Use the Lyapunov function $V(x) = x^2$ to show that the origin is globally asymptotically stable.

Exercise 2.5 Lyapunov stability 5

Consider the nonlinear system

$$x_{k+1} = g(x_k)$$

where g has the property that

$$|g(x)| < |x|, \quad \forall x \neq 0$$

and $g(0) = 0$. Show that the origin is globally asymptotically stable.

Exercise 2.6 Discrete-time Lyapunov stability

Prove that

$$x_{t+1} = Ax_t$$

is asymptotically stable if and only if for every positive definite matrix Q, there exists a unique positive definite matrix P that satisfies

$$A^T P A - P + Q = 0$$

Exercise 2.7 Discrete-time Lyapunov instability

Prove that

$$x_{t+1} = Ax_t$$

is unstable if there exists a positive definite matrix Q such that the Lyapunov equation

$$A^T P A - P + Q = 0$$

has an indefinite solution P.

Exercise 2.8 Discrete-time stability

Consider the linear system

$$x_{t+1} = Ax_t$$

where

$$A = \begin{pmatrix} 1/4 & 1 \\ 1/2 & 0 \end{pmatrix}$$

(a) Compute the eigenvalues of the system matrix A. Is the system asymptotically stable?

(b) Solve the discrete-time Lyapunov equation

$$A^T P A - P + Q = 0$$

with $Q = I$. What does the matrix solution P say about system stability?

(c) Repeat (a) and (b) for the matrix

$$\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

2.2 Invariant sets

Exercise 2.9 Predecessor sets

Consider the following linear system

$$x_{t+1} = \begin{pmatrix} 0.5 & 0 \\ 1 & -0.5 \end{pmatrix} x_t$$

For each of the following polyhedral sets $\mathcal{X}$, determine $\mathrm{Pre}\,(\mathcal{X})$ under the given dynamics.

Hint: $\mathrm{Pre}\,(\mathcal{X})$ *under* $x_{t+1} = Ax_t$ *is defined as*

$$\mathrm{Pre}\,(\mathcal{X}) = \{x \in \mathbb{R}^n \mid Ax \in \mathcal{X}\}$$

(a)

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid 2x_1 + 3x_2 \leq 5\}$$

(b)

$$\mathcal{X} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} -1 \\ -2 \end{bmatrix} \leq x \leq \begin{bmatrix} 2 \\ 3 \end{bmatrix} \right\}$$

(c)

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid x \geq 0\}$$

Exercise 2.10 Geometric condition for invariance

Show that a set $\mathcal{O} \subseteq \mathcal{X}$ is positive invariant under $x_{t+1} = Ax_t$ if and only if

$$\mathcal{O} \subseteq \mathrm{Pre}\,(\mathcal{O})$$

Hint: $\mathcal{O}$ is positive invariant if

$$x_t \in \mathcal{O} \Rightarrow x_k \in \mathcal{O} \quad \forall k \geq t$$

Exercise 2.11 Invariant sets

For each of the sets $\mathcal{X}$ in Exercise 2.9, determine if $\mathcal{X}$ is positive invariant under

$$x_{t+1} = \begin{pmatrix} 0.5 & 0 \\ 1 & -0.5 \end{pmatrix} x_t$$

Hint: Use the result from Exercise 2.10.

Exercise 2.12 Invariant sets 2

Repeat exercise 2.11 for the system

$$x_{t+1} = \begin{pmatrix} 0.5 & 0 \\ 0.5 & 0.5 \end{pmatrix} x_t$$

In other words, determine if $\mathcal{X}$ is positive invariant under these dynamics for each of the sets $\mathcal{X}$ in Exercise 2.9.

Exercise 2.13 One-step controllable sets

Consider the double integrator

$$x_{t+1} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x_t + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_t$$

under the state constraints

$$u_t \in \mathcal{U} = \{u \in \mathbb{R} \mid -1 \leq u \leq 1\}$$

For each of the following polyhedral sets $\mathcal{X}$, determine $\mathrm{C}\,(\mathcal{X};\mathcal{U})$ under the given dynamics.

Hint: $\mathrm{C}(\mathcal{X};\mathcal{U})$ *is defined as*

$$\mathrm{C}(\mathcal{X};\mathcal{U}) = \{x \in \mathbb{R}^n \mid \exists u \in \mathcal{U} \text{ s.t. } Ax + Bu \in \mathcal{X}\}$$

(a)

$$\mathcal{X} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} -5 \\ -5 \end{bmatrix} \leq x \leq \begin{bmatrix} 5 \\ 5 \end{bmatrix} \right\}$$

(b)

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid x_1 \geq -3, x_1 + 2.5x_2 \geq -3, x_1 \leq 3, x_1 + 2.5x_2 \leq 3\}$$

Exercise 2.14 Geometric condition for controlled invariance

Show that a set $\mathcal{C} \subseteq \mathcal{X}$ is controlled invariant under $x_{t+1} = Ax_t + Bu_t,\ u_t \in \mathcal{U}$ if and only if

$$\mathcal{C} \subseteq \mathrm{C}(\mathcal{C};\mathcal{U})$$

Hint: $\mathcal{C}$ *is controlled invariant if*

$$x_t \in \mathcal{C} \Rightarrow \exists \{u_k\}_{k=t}^{\infty} \in \mathcal{U} \text{ s.t. } x_k \in \mathcal{C},\ \forall k \geq t$$

Exercise 2.15 Controlled invariant sets

For each of the sets $\mathcal{X}$ in Exercise 2.13, determine if $\mathcal{X}$ is controlled invariant under

$$x_{t+1} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x_t + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_t, \quad u_t \in \mathcal{U} = \{u \in \mathbb{R} \mid -1 \leq u \leq 1\}$$

Hint: Use the result from Exercise 2.14.

Exercise 2.16 Feedback invariance

Assume that there are no input constraints ($\mathcal{U} = \mathbb{R}^m$) and show that a set $\mathcal{C} \subseteq \mathcal{X}$ is controlled invariant under $x_{t+1} = Ax_t + Bu_t$ if and only if there exists an $m \times n$ matrix F such that $\mathcal{C}$ is positive invariant under $x_{t+1} = (A + BF)x_t$

Exercise 2.17 Another geometric condition

Assume that there are no input constraints ($\mathcal{U} = \mathbb{R}^m$) and show that a set $\mathcal{C} \subseteq \mathcal{X}$ is controlled invariant under $x_{t+1} = Ax_t + Bu_t$ if and only if

$$\mathrm{Reach}(\mathcal{C}) \subseteq \mathcal{C} + \mathrm{Im}\ B$$

where $\mathrm{Reach}(\mathcal{C})$ is under $x_{t+1} = Ax_t$. For simplicity, assume that $\mathcal{C}$ is a bounded polytope.

Hint: $\mathrm{Reach}(\mathcal{X})$ *under* $x_{t+1} = Ax_t$ *is defined as*

$$\mathrm{Reach}(\mathcal{X}) = \{x \in \mathbb{R}^n \mid \exists x_0 \in \mathcal{X} \text{ s.t. } x = Ax_0\}$$

Exercise 2.18 Constrained linear control Exam: 18-12-18

Consider the linear system

$$x_{t+1} = \begin{pmatrix} 0.8 & -0.1 \\ 1 & 1.6 \end{pmatrix} x_t + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_t := Ax_t + Bu_t \tag{2}$$

subject to the input and state constraints

$$\begin{aligned} x_t \in \mathcal{X} &= \left\{ x \in \mathbb{R}^2 \text{ such that } \begin{bmatrix} -5 \\ -5 \end{bmatrix} \leq x \leq \begin{bmatrix} 5 \\ 5 \end{bmatrix} \right\}, \quad \forall t \geq 0 \\ u_t \in \mathcal{U} &= \{ u \in \mathbb{R} \text{ such that } -3 \leq u \leq 3 \}, \quad \forall t \geq 0 \end{aligned}$$

We are interested in solving the following infinite-horizon optimal control problem

$$\begin{aligned} \underset{\{u_k\}_{k=0}^{\infty}}{\text{minimize}} \quad & \textstyle\sum_{k=0}^{\infty} x_k^T Q_1 x_k + u_k^T Q_2 u_k \\ \text{such that} \quad & x_{k+1} = \begin{pmatrix} 0.8 & -0.1 \\ 1 & 1.6 \end{pmatrix} x_k + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_k, \quad k = 0, \ldots, \infty \\ & x_k \in \mathcal{X}, \quad k = 0, \ldots, \infty \\ & u_k \in \mathcal{U}, \quad k = 0, \ldots, \infty \\ & x_0 = x \end{aligned} \tag{3}$$

where the penalty matrices are given by

$$Q_1 = \begin{pmatrix} 0.8 & 0.18 \\ 0.18 & 1.05 \end{pmatrix}, \quad Q_2 = 1.$$

(a) Consider the stabilizing control law

$$u_t = -Lx_t = -\begin{pmatrix} 1 & 1 \end{pmatrix} x \tag{4}$$

Estimate the infinite-horizon cost of applying this control law to (2) by solving the Lyapunov equation

$$(A - BL)^T P (A - BL) - P + (Q_1 + L^T Q_2 L) = 0$$

and then verify that the feedback law is indeed stabilizing

(b) It is sometimes convenient to verify set invariance using the concept of one-step reachable sets. Recall the following definitions:

Definition 1. The one-step reachable set $\text{Reach}(\mathcal{X})$ from $\mathcal{X}$ is given by

$$\text{Reach}(\mathcal{X}) = \{ y \in \mathbb{R}^n \mid \exists x \in \mathcal{X} \text{ s.t } y = Ax \}$$

Definition 2. A set $\mathcal{O} \subseteq \mathcal{X}$ is positive invariant under $x_{t+1} = Ax_t$ if

$$x_0 \in \mathcal{O} \Rightarrow x_t \in \mathcal{O}, \ \forall t \geq 0$$

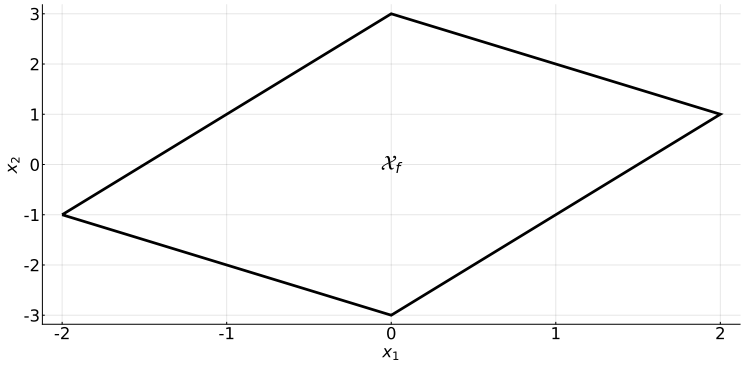
Show, using these definitions, that $\mathcal{O} \subseteq \mathcal{X}$ is positive invariant if and only if

$$\text{Reach}(\mathcal{O}) \subseteq \mathcal{O}$$

(c) Show that the set

$$\mathcal{X}_f = \left\{ x \in \mathbb{R}^2 \text{ such that } \begin{matrix} x_1 + x_2 \leq 3 \\ x_1 + x_2 \geq -3 \\ 2x_2 - x_1 \leq 3 \\ 2x_2 - x_1 \geq -3 \end{matrix} \right\}$$

is control invariant under $x_{t+1} = Ax_t + Bu_t$ by verifying that it is positive invariant under $x_{t+1} = (A - BL)x_t$ for L as defined in (4). Moreover, show that the feedback law (4) is admissible in $\mathcal{X}_f$.



Hint: The one-step reachable set from a set $\mathcal{X}$ under $x_{t+1} = Ax_t$ can be characterized by observing the effect of A on the vertex points of $\mathcal{X}$.

(d) Consider the following finite-dimensional optimization problem

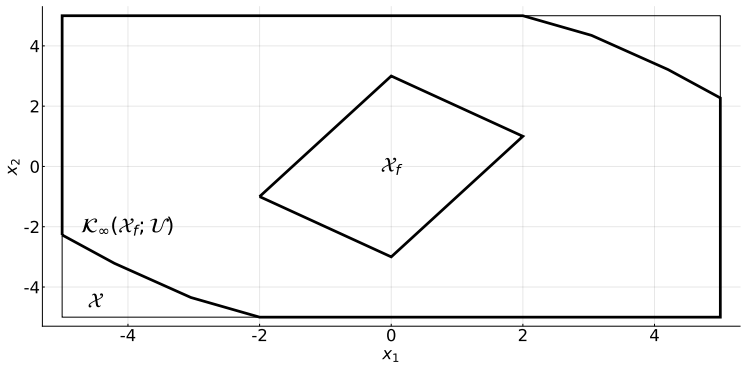
$$\begin{aligned} \underset{u_0,\ldots,u_{N-1}}{\text{minimize}} \quad & \textstyle\sum_{k=0}^{N-1} x_k^T Q_1 x_k + u_k^T Q_2 u_k + x_N^T P x_N \\ \text{such that} \quad & x_{k+1} = \begin{pmatrix} 0.8 & -0.1 \\ 1 & 1.6 \end{pmatrix} x_k + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_k, \quad k = 0, \ldots, N-1 \\ & x_k \in \mathcal{X}, \quad k = 0, \ldots, N \\ & u_k \in \mathcal{U}, \quad k = 0, \ldots, N-1 \\ & x_N \in \mathcal{X}_f \\ & x_0 = x \end{aligned} \tag{5}$$

where P is the solution in (a) and $\mathcal{X}_f$ is the set defined in (c). Explain, with reference to theory presented in the course, why a feasible solution to (5) defines an input-admissible control sequence $\{u_0, \ldots, u_{N-1}\}$ that drives the system to a state from which the feedback law (4) can stabilize the system. Comment on the optimality of solutions to (5) in relation to (3).

(e) The largest control invariant set $\mathcal{K}_\infty(\mathcal{X}_f; \mathcal{U}) \subseteq \mathcal{X}$ under $x_{t+1} = Ax_t + Bu_t$ that contains $\mathcal{X}_f$ is shown below
Comment on the feasibility of (5) for

- $x_0 = \begin{pmatrix} -4 & -4 \end{pmatrix}^T$
- $x_0 = \begin{pmatrix} -4 & 2 \end{pmatrix}^T$

for any given horizon length $N > 0$.



3 Finite-time optimal control

3.1 Convex optimization

Exercise 3.1 Convex functions

Show that the following functions are convex.

(a) any norm $f(x) = \|x\|$ on $\mathbb{R}^n$.
Hint: Recall that f is a norm if it satisfies the following three conditions: (i) $\|x\| = 0$ if and only if $x = 0$, (ii) $\|ax\| \leq |a|\|x\|$, (iii) $\|x + y\| \leq \|x\| + \|y\|$..

(b) $f(x) = \max\{f_1(x), f_2(x)\}$ where f_1 and f_2 are convex functions, with $\operatorname{dom} f = \operatorname{dom} f_1 \cap \operatorname{dom} f_2$.
Hint: Recall that $\max\{a + b, c + d\} \leq \max\{a, c\} + \max\{b, d\}$.

Exercise 3.2 Convex sets

In the course, constraints are often described by linear inequalities or (convex) quadratic inequalities.

(a) The solution set to a linear inequality

$$a^T x \leq b$$

with $a, b \in \mathbb{R}^n$ and $b \in \mathbb{R}$ defines a halfspace

$$\mathcal{H} = \left\{x \,|\, a^T x \leq b\right\} \subseteq \mathbb{R}^n \tag{6}$$

The set of solution to multiple linear inequalities is therefore described by the intersection of several halfspaces, that is, by a polyhedron. Determine a minimal set of linear inequalities that describe the polyhedron shown in Figure 3.

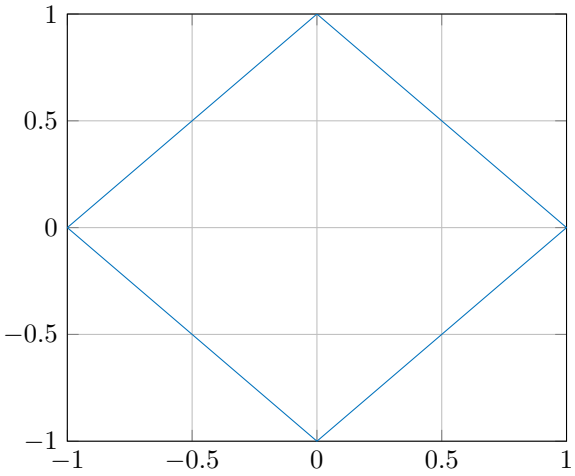


Figure 3: Polyhedron.

(b) If P is positive semidefinite, then the solution set to a quadratic inequality

$$x^T P x + 2q^T x + r \leq 0$$

defines an ellipsoid

$$\mathcal{E} = \left\{ x \,|\, x^T P x + 2q^T x + r \leq 0 \right\}. \tag{7}$$

It is often more convenient to represent the ellipsoid in the standard form.

$$\mathcal{E} = \left\{ x \,|\, (x - x_c)^T Q^{-1} (x - x_c) \leq 1 \right\} \tag{8}$$

Determine the parameters Q and x_c such that (7) and (8) are equal.

(c) Another standard representation for ellipsoids is to describe them as an affine transformation of the unit ball, i.e.

$$\mathcal{E} = \{ x = Mu + m \,|\, \|u\|_2 \leq 1 \} \tag{9}$$

Derive an explicit relationship for M and m in terms of the original problem data P, q and r.

(d) Use your result from (b) and (c) to plot the ellipsoid given by

$$P = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}, \quad q = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad r = -2$$

Exercise 3.3 Convex duality

In class, we used Lagrange duality to derive an explicit solution to the problem

$$\begin{array}{ll} \text{minimize} & x^T x \\ \text{subject to} & Cx = d \end{array}$$

In this exercise, we will consider the related problem

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & x^T x \leq 1 \end{array} \tag{10}$$

(a) Show that you can solve (10) by solving a related problem on the standard form for convex optimization problems used in class, *i.e.*

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_1(x) \leq 0 \end{array}$$

for some convex functions $f_0(x)$ and $f_1(x)$. How are the optimal solutions and optimal values of the two problems related?

(b) Use the method of Lagrange multipliers to find the optimal solution to the related problem derived in (a).

(c) Use your result in (b) to derive an explicit condition for when the ellipsoid

$$\mathcal{E} = \{x = Mu + m \,|\, \|u\| \leq 1\}$$

is contained in the half space

$$\mathcal{H} = \left\{x \,|\, a^T x \leq b\right\}$$

Hint: The containment is equivalent to saying that $a^T x \leq b$ for all $x \in \mathcal{E}$.

Exercise 3.4 Linear programming

(a) Solve the following linear programming problems using the Matlab command `linprog`.

(i)

$$\begin{array}{ll} \underset{x_1,\ldots,x_4}{\text{minimize}} & -5x_1 - 7x_2 - 12x_3 + x_4 \\ \text{subject to} & 2x_1 + 3x_2 + 2x_3 + x_4 \leq 38 \\ & 3x_1 + 2x_2 + 4x_3 - x_4 \leq 55 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{array}$$

(ii)

$$\begin{array}{ll} \underset{x_1,\ldots,x_4}{\text{maximize}} & 5x_1 + 3x_2 + 2x_3 \\ \text{subject to} & 4x_1 + 5x_2 + 2x_3 + x_4 \leq 20 \\ & 3x_1 + 4x_2 - x_3 + x_4 \leq 30 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{array}$$

(iii)

$$\begin{array}{ll} \underset{x_1,x_2}{\text{minimize}} & 2x_1 + 3x_2 \\ \text{subject to} & 3x_1 + 2x_2 = 14 \\ & 2x_1 - 4x_2 \geq 2 \\ & 4x_1 + 3x_2 \leq 19 \\ & x_1, x_2 \geq 0 \end{array}$$

(b) Solve problems (i)-(iii) using YALMIP

Exercise 3.5 Transportation problem

A company owns three factories that supply $s_i,\ i = 1, \ldots, 4$ amounts of goods respectively each week. Moreover, the company owns three warehouses that must be filled by $d_j,\ j = 1, \ldots, 3$ amounts of goods by the end of each week. The cost of moving $x_{i,j}$ amounts of goods from factory i to warehouse j is given by $c_{i,j}$. The company wants to meet each warehouse demand with as low cost as possible. To that end, the following linear program is formulated.

$$\begin{aligned}
\underset{x_{i,j}}{\text{minimize}} \quad & \sum_{i=1}^{4}\sum_{j=1}^{3} c_{i,j}x_{i,j} \\
\text{subject to} \quad & \sum_{j=1}^{3} x_{i,j} \leq s_i, \quad i = 1, \ldots, 4 \\
& \sum_{i=1}^{4} x_{i,j} \geq d_i, \quad j = 1, \ldots, 3 \\
& x_{i,j} \geq 0, \quad i = 1, \ldots, 4,\ j = 1, \ldots, 3
\end{aligned}$$

Solve the transportation problem for the following sets of parameters

$$c_{i,j} = 1 + (i-j)^2, \quad s_i = i^2,\ i = 1, \ldots, 4, \quad d_j = 10,\ j = 1, \ldots, 3$$

in MATLAB, using either `linprog` or YALMIP.

Exercise 3.6 Quadratic programming

(a) Solve the following quadratic programming problems using the Matlab command `quadprog`.

(i)

$$\begin{aligned}
\underset{x \in \mathbb{R}^3}{\text{minimize}} \quad & \frac{1}{2}x^T Q x \\
\text{subject to} \quad & Ax \leq b
\end{aligned}$$

where

$$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad A = \begin{pmatrix} -1 & -1 & -1 \\ -2 & 1 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} -3 \\ -1 \end{pmatrix}$$

(ii)

$$\begin{aligned}
\underset{x_1, x_2}{\text{minimize}} \quad & 2x_1 + 3x_2 + 4x_1^2 + 2x_1x_2 + x_2^2 \\
\text{subject to} \quad & x_1 - x_2 \geq 0 \\
& x_1 + x_2 \leq 4 \\
& x_1 \leq 3
\end{aligned}$$

(iii)

$$\begin{aligned} \underset{x_1,x_2,u_0,u_1}{\text{minimize}} \quad & \frac{1}{2}(x_1^2 + x_2^2 + u_0^2 + u_1^2) \\ \text{subject to} \quad & x_1 = 0.5x_0 + u_0 \\ & x_2 = 0.5x_1 + u_1 \\ & 2 \leq x_1 \leq 5 \\ & -2 \leq x_2 \leq 2 \\ & -1 \leq u_0 \leq 1 \\ & -1 \leq u_1 \leq 1 \end{aligned}$$

with $x_0 = 3$.

(b) Solve problems (i)-(iii) using YALMIP

Exercise 3.7 Curve fitting

In this problem, we will study how to solve some basic parameter estimation problems using linear and quadratic programming. In particular, given measurements of $x \in \mathbb{R}^n$ and $y \in \mathbb{R}$, we are interested in finding optimal parameters for the linear model

$$y = a^T x + b.$$

To this end, we are given m data pairs (x_i, y_i) for $i = 1, 2, \ldots, m$. These data points are inconsistent with the model, in the sense that there is no single (a, b) that fits perfectly to all measured data. Instead, the measured data satisfy

$$y_i - a^T x_i - b = r_i$$

for some scalar values r_i. We call these r_i the *residuals*, since they represent the error between the true measurements and the output predicted by our model. It is natural to try to find the parameters (a, b) that make the residuals as small as possible, i.e. to try to find a and b that solve

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_{i=1}^m f(r_i) \\ \text{subject to} \quad & y_i - a^T x_i - b = r_i \qquad i = 1, \ldots, m \end{aligned} \tag{11}$$

However, depending on how we decide to quantify the size of the residuals, through our choice of the loss functions f_i, we get quite different solutions.

(a) We start exploring quadratic losses,

$$f(r) = r^2, \quad i = 1, \ldots, m$$

Show that the optimization problem (11) can be written as a quadratic program on the standard form

$$\begin{aligned} \text{minimize} \quad & z^T H z + 2f^T z + d \\ \text{subject to} \quad & Gz = h \end{aligned}$$

where the decision vector z contains the parameter estimates a and b. Note that, if z only contains the parameter estimates a and b, the problem can be formulated as an unconstrained optimization problem.

(b) As an alternative, we consider the absolute value loss

$$f(r) = |r|, \quad i = 1, \ldots, m$$

which corresponds to minimizing the l_1 norm of the residuals (recall the definition of the l_1 norm, $\|x\|_1 = \sum_i |x_i|$). Introduce a new variable t_i such that $t_i = f_i(r) = |r|$ for $i = 1, \ldots, m$. Then, show that the condition

$$t_i \geq |r_i|$$

can be expressed as two linear inequalities. Use this to formulate the l_1 norm minimization problem as a linear program

$$\begin{array}{ll} \text{minimize} & c^T z \\ \text{subject to} & Az \leq b \end{array}$$

where the decision vector now contains the parameter a and b, and the additional variables t_i discussed above.

(c) Finally, we consider the maximum of the absolute value loss, i.e.,

$$f(r) = \|r\|_\infty = \max_i |r_i| \qquad\qquad i = 1, \ldots, m$$

Use the same strategy as in the last question to formulate the l_∞ norm minimization problem as a linear program

$$\begin{array}{ll} \text{minimize} & c^T z \\ \text{subject to} & Az \leq b \end{array}$$

(d) Unzip the `dataset.zip` available on the course homepage, and implement the three estimation procedures in YALMIP and try them on real data (for the linear fitting case, i.e., when a is scalar). The estimation procedures should be tested on the `noisy_dataset.mat` and `outliers_dataset.mat`. You can confirm and compare your results using the `ideal_dataset.mat`. What is the influence of outliers in the different estimation procedures?

3.2 Dynamic programming

Exercise 3.8 LQ problems

Solve the following linear-quadratic problems using dynamic programming. Specify the optimal strategy $\hat{u}_0, \ldots, \hat{u}_3$, and the resulting objective.

(a)

$$\begin{array}{ll} \underset{u_0,\ldots,u_2}{\text{minimize}} & \displaystyle\sum_{k=0}^{2} u_k^2 + x_3^2 \\ \text{subject to} & x_{k+1} = x_k + u_k \\ & x_0 = 4 \end{array}$$

(b)

$$\begin{array}{ll} \underset{u_0,\ldots,u_2}{\text{minimize}} & \sum_{k=0}^{2} u_k^2 \\ \text{subject to} & x_{k+1} = 2x_k - u_k \\ & x_0 = 21 \\ & x_3 = 0 \end{array}$$

(c)

$$\begin{array}{ll} \underset{u_0,\ldots,u_2}{\text{minimize}} & \sum_{k=0}^{2} x_k^2 + u_k^2 + x_3^2 \\ \text{subject to} & x_{k+1} = 2x_k + u_k \\ & x_0 = 5 \end{array}$$

Exercise 3.9 Dynamical systems

Solve the following nonlinear problems using dynamic programming. Specify the optimal strategy $\hat{u}_0, \ldots, \hat{u}_3$, and the resulting objective.

(a)

$$\begin{array}{ll} \underset{u_0,\ldots,u_2}{\text{maximize}} & \sum_{k=0}^{2} x_k - x_3 \\ \text{subject to} & x_{k+1} = x_k u_k \\ & 0 \leq u_k \leq 1, \quad k = 0, \ldots, 2 \\ & x_0 = 1 \end{array}$$

(b)

$$\begin{array}{ll} \underset{u_0,\ldots,u_2}{\text{maximize}} & \sum_{k=0}^{2} \log u_k x_k + \log x_3 \\ \text{subject to} & x_{k+1} = x_k(1 - u_k) \\ & 0 < u_k < 1, \quad k = 0, \ldots, 2 \\ & x_0 = 4 \\ & (x_k > 0, \quad k = 0, \ldots, 3) \end{array}$$

(c)

$$\begin{array}{ll} \underset{u_0,\ldots,u_2}{\text{minimize}} & \sum_{k=0}^{2} u_k^4 + x_3^4 \\ \text{subject to} & x_{k+1} = x_k + u_k \\ & x_0 = 4 \end{array}$$

Exercise 3.10 State-constrained problems

Solve the following state-constrained problems using dynamic programming. Specify the optimal strategy $\hat{u}_0, \ldots, \hat{u}_3$, and the resulting objective.

(a)

$$\begin{aligned} \underset{u_0,\dots,u_2}{\text{minimize}} \quad & \sum_{k=0}^{2} u_k^2 + x_3^2 \\ \text{subject to} \quad & x_{k+1} = x_k + u_k \\ & x_k \geq 2, \quad k = 0, \dots, 3 \\ & x_0 = 4 \end{aligned}$$

(b)

$$\begin{aligned} \underset{u_0,\dots,u_2}{\text{maximize}} \quad & \sum_{k=0}^{2} \log u_k x_k + \log x_3 \\ \text{subject to} \quad & x_{k+1} = x_k(1 - u_k) \\ & 0 < u_k < 1, \quad k = 0, \dots, 2 \\ & x_k \geq 2, \quad k = 0, \dots, 3 \\ & x_0 = 4 \\ & (x_k > 0, \quad k = 0, \dots, 3) \end{aligned}$$

(c) For the above problems, qualitatively describe the resulting optimal input sequences for arbitrary starting points $x_0 \geq 2$.

Exercise 3.11 Investment planning Exam: 2017-10-23

You currently own $x_c > N$ units of company shares, of some company C. You have reason to believe that company C will go bankrupt in N years, at which your shares would be worth nothing. However, up to that time, you receive dividends equal to $\theta\times$ your current share holdings, where $0 < \theta < 1$. In addition, every year you are allowed to buy or sell at most *one* unit of shares without any extra cost. Your intuition tells you that it should be possible to make a profit if you act in a clever way during these N years (assuming your bankruptcy assumption is true). Therefore, you formulate the following discrete optimal control problem:

$$\begin{aligned} x_c + \underset{u_k}{\text{maximize}} \quad & \sum_{k=0}^{N-1} [\theta x_k] - 2x_N \\ \text{subject to} \quad & x_{k+1} = x_k + u_k \\ & |u_k| \leq 1, \quad k = 0, \dots, N-1 \\ & x_0 = x_c \end{aligned}$$

The intuition of the objective is that you each year receive dividends θx_k, and at the end of the period you lose whatever amount x_N you have left as well as gain/lose $x_c - x_N$ depending on how many shares you sold/bought. Note, that x_c is constant and can be removed from the maximization. Furthermore, $x_c > N$ so it is not possible to sell of all shares and you can assume $x_k > 0$ throughout the period. (It is assumed that unit shares have unit cost, and that you have infinite capital.)

(a) For the specific case $x_c = 3$, $N = 2$, and $\theta = 0.4$, solve the above problem using dynamic programming. Specify the optimal strategy $\hat{u}_0, \hat{u}_1$, and the resulting profit.

(b) Formulate a value function $V_{k+1}(x_{k+1})$ and show by induction that it is indeed a valid value function for the dynamic programming recursion:

$$V_k(x_k) = \max_{|u_k| \leq 1} [\theta x_k + V_{k+1}(x_{k+1})]$$

Hint: Think carefully about what structure the value function had in (a).

(c) Use $V_{N-1}(x_{N-1})$ as a starting point of the recursion obtained in (b), and determine an optimal control law $\hat{u}_k$ as a function of θ and N.

(d) Explain qualitatively how the optimal strategy depends on θ and N.

Exercise 3.12 Investment planning revisited Exam: 2018-10-20

You currently own x_c units of shares, of some company C. You have reason to believe that company C will bankrupt in N years, at which point your shares would be worth nothing. However, up to that time, you receive dividends equal to $\theta\times$ your current share holdings, where $0 < \theta < 1$. In addition, every year you are allowed to buy or sell at most *one* unit of shares without any extra cost. Your intuition tells you that it should be possible to make a profit if you act in a clever way during these N years (assuming your bankruptcy asssumption is true). Therefore, you formulate the following discrete optimal control problem:

$$\begin{aligned} x_c + \underset{u_k}{\text{maximize}} \quad & \sum_{k=0}^{N-1} \{\theta x_k\} - 2x_N \\ \text{subject to} \quad & x_{k+1} = x_k + u_k, && k = 0, \ldots, N-1 \\ & u_k \in \{-1, 0, 1\}, && k = 0, \ldots, N-1 \\ & x_k \geq 0, && k = 0, \ldots, N \\ & x_0 = x_c \end{aligned} \tag{12}$$

The intuition of the objective is that you each year receive dividends θx_k, and at the end of the period you lose whatever amount x_N you have left as well as gain/lose $x_c - x_N$ depending on how many shares you sold/bought. Note, that x_c is constant and can be removed from the maximization. Furthermore, you cannot own a negative amount of shares, so $x_k \geq 0$. It is assumed that unit shares have unit cost, and that you have infinite capital.

(a) For the specific case $x_c = 3$, $N = 2$, and $\theta = 0.4$, solve (12) using dynamic programming. Specify the optimal strategy $\hat{u}_0, \hat{u}_1$, and the resulting profit.

Hint: Since $x_c > N$, the constraint $x_k \geq 0$ can be ignored.

(b) Consider now the case $x_c = 1$, $N = 3$, and $\theta = 0.8$. Solve (12) using dynamic programming. Specify the optimal strategy $\hat{u}_0, \hat{u}_1, \hat{u}_2$, and the resulting profit. *Note: Solution by enumeration is not an acceptable solution.*

Exercise 3.13 Capital distribution Exam: 2017-12-18

A public company has profit x_k at year k. This profit is distributed partly to the shareholders as dividends and partly as reinvestment in the company itself. Reinvesting increases the company profit by $\theta\times$ the invested capital. To boost

reputation, the company decides to maximize the amount distributed to the shareholdes over an N year period. Therefore, the following discrete optimal control problem is formulated:

$$\begin{aligned} \underset{u_k}{\text{maximize}} \quad & \sum_{k=0}^{N-1} (1-u_k)x_k \\ \text{subject to} \quad & x_{k+1} = x_k + \theta u_k x_k \\ & 0 \leq u_k \leq 1, \quad k = 0, \ldots, N-1 \\ & x_0 = x_c \end{aligned}$$

The intuition of the objective is that each year the profit x_k is divided into $u_k x_k$ which is reinvested and $(1-u_k)x_k$ which is given to the shareholders. $x_c > 0$ is the current profit which can be distributed the first year.

(a) For the specific case $x_c = 3$, $N = 2$, and $\theta = 1.5$, solve the above problem using dynamic programming. Specify the optimal strategy $\hat{u}_0, \hat{u}_1$, and the total distributed profit.

(b) Formulate a value function $V_{k+1}(x_{k+1})$ and show by induction that it is indeed a valid value function for the dynamic programming recursion:

$$V_k(x_k) = \max_{0 \leq u_k \leq 1} [(1-u_k)x_k + V_{k+1}(x_{k+1})]$$

Next, determine an optimal control law $\hat{u}_k$ as a function of θ and N.
Hint: Think carefully about what structure the value function had in (a).

(c) Qualitatively describe the optimal strategy.

Exercise 3.14 Capital distribution revisited

The shareholders were notified about the optimal strategy from the previous problem, and they were not happy about receiving zero dividends while the company builds capital. Therefore, company management decided to reformulate the underlying planning problem into:

$$\begin{aligned} \underset{u_k}{\text{maximize}} \quad & \prod_{k=0}^{N-1} (1-u_k)x_k \\ \text{subject to} \quad & x_{k+1} = x_k + \theta u_k x_k \\ & 0 \leq u_k \leq 1, \quad k = 0, \ldots, N-1 \\ & x_0 = x_c \end{aligned}$$

The effect of this reformulation will now be analyzed. Note, that solving the above problem is equivalent to solving:

$$\begin{aligned} \underset{u_k}{\text{maximize}} \quad & \sum_{k=0}^{N-1} \log (1-u_k)x_k \\ \text{subject to} \quad & x_{k+1} = x_k + \theta u_k x_k \\ & 0 \leq u_k < 1, \quad k = 0, \ldots, N-1 \\ & x_0 = x_c \\ & ((1-u_k)x_k > 0, \quad k = 0, \ldots, N-1) \end{aligned}$$

(a) For the specific case $x_c = 3$, $N = 2$, and $\theta = 1.5$, solve the above problem using dynamic programming. Specify the optimal strategy $\hat{u}_0, \hat{u}_1$, and the total distributed profit. How does this differ from the result in Exercise 3.13 (a)?

(b) Formulate a value function $V_{k+1}(x_{k+1})$ and show by induction that it is indeed a valid value function for the dynamic programming recursion:

$$V_k(x_k) = \max_{0 \le u_k \le 1} [\log(1 - u_k)x_k + V_{k+1}(x_{k+1})]$$

Next, determine an optimal control law $\hat{u}_k$ as a function of θ and N.
Hint: Think carefully about what structure the value function had in (a).

(c) Qualitatively describe how the resulting optimal strategy differs from that in Exercise 3.13.

Exercise 3.15 Knapsack problems

(a) Solve the following knapsack problem using dynamic programming

$$\begin{aligned} \underset{x_1,\ldots,x_3}{\text{maximize}} \quad & 6x_1 + 10x_2 + 12x_3 \\ \text{subject to} \quad & x_1 + 2x_2 + 3x_3 = 5 \\ & x_i \in \{0, 1\}, \quad i = 1, \ldots, 3 \end{aligned}$$

Hint: Let each variable choice represent a stage and introduce the remaining constraint gap as a state variable R_i. In other words, $R_0 = 5$ and the recursion is as follows:

$$V_i(R_{i-1}) = \max_{x_i \in \{0,1\}, w_i x_i \le R_{i-1}} [v_i x_i + V_{i+1}(R_{i-1} - w_i x_i)]$$

(b) Implement a MATLAB function `knapsack(v,w,W)` that solves the general knapsack problem

$$\begin{aligned} \underset{x_1,\ldots,x_n}{\text{maximize}} \quad & \sum_{i=1}^{N} v_i x_i \\ \text{subject to} \quad & \sum_{i=1}^{N} w_i x_i \le W \\ & x_i \in \{0, 1\}, \quad i = 1, \ldots, N \end{aligned}$$

given value coefficients v_i, weight coefficients w_i and W. Use the function to solve the following knapsack problems.
(i)

$$\begin{aligned} \underset{x_1,\ldots,x_5}{\text{maximize}} \quad & x_1 + 2x_2 + 3x_3 + 2x_4 + x_5 \\ \text{subject to} \quad & x_1 + x_2 + x_3 + x_4 + x_5 \le 3 \\ & x_1, \ldots, x_5 \in \{0, 1\} \end{aligned}$$

(ii)

$$\begin{aligned} \underset{x_1,\ldots,x_5}{\text{maximize}} \quad & 2x_1 + 2x_2 + 3x_3 + 5x_4 + 3x_5 \\ \text{subject to} \quad & x_1 + 2x_2 + x_3 + 4x_4 + 2x_5 \leq 10 \\ & x_1, \ldots, x_5 \in \{0, 1\} \end{aligned}$$

(iii)

$$\begin{aligned} \underset{x_1,\ldots,x_7}{\text{maximize}} \quad & 3x_1 + 4x_2 + 7x_3 + 2x_4 + 5x_5 + x_6 + 3x_7 \\ \text{subject to} \quad & x_1 + 2x_2 + 3x_3 + x_4 + 4x_5 + x_6 + 2x_7 \leq 15 \\ & x_1, \ldots, x_7 \in \{0, 1\} \end{aligned}$$

Exercise 3.16 Cargo loading

A company has N types of goods that are to be exported through cargo freight. A single unit of the ith good weighs w_i tonnes, and each cargo has a capacity of W tonnes. The value of exporting a single unit of the ith good is given by v_i. The aim is to load the vessels with the distribution of goods that gives the most value possible. Hence, if a_i is the number of units of the ith good, the goal is captured by the following optimization problem:

$$\begin{aligned} \underset{a_1,\ldots,a_N}{\text{maximize}} \quad & \sum_{i=1}^{N} v_i a_i \\ \text{subject to} \quad & \sum_{i=1}^{N} w_i a_i \leq W \\ & a_i \in \mathbb{Z}_+, \quad i = 1, \ldots, N \end{aligned}$$

Implement a MATLAB function `cargo_load(v,w,W)` that solves the cargo loading problem given values, weights and cargo capacity. Use the function to solve the following cargo loading problems.
Hint: Consider the following recursion:

$$V_i(x_i) = \max_{a_j = 0,1,\ldots,\lfloor \frac{W}{a_i} \rfloor} [v_i a_j + V_{i+1}(x_i - w_i a_j)]$$

(a)

$$\begin{aligned} \underset{a_1,\ldots,a_3}{\text{maximize}} \quad & 31a_1 + 47a_2 + 14a_3 \\ \text{subject to} \quad & 2a_1 + 3a_2 + a_3 \leq 4 \\ & a_1, \ldots, a_3 \in \mathbb{Z}_+ \end{aligned}$$

(b)

$$\begin{aligned} \underset{a_1,\ldots,a_4}{\text{maximize}} \quad & 5a_1 + 2a_2 + 7a_3 + a_4 \\ \text{subject to} \quad & 4a_1 + 6a_2 + 2a_3 + a_4 \leq 10 \\ & a_1, \ldots, a_4 \in \mathbb{Z}_+ \end{aligned}$$

(c)

$$\begin{aligned} \underset{a_1,\ldots,a_5}{\text{maximize}} \quad & 2a_1 + 4a_2 + 6a_3 + 3a_4 + 20a_5 \\ \text{subject to} \quad & 4a_1 + 2a_2 + 3a_3 + 2a_4 + 8a_5 \leq 15 \\ & a_1, \ldots, a_5 \in \mathbb{Z}_+ \end{aligned}$$

Exercise 3.17 Inverted Pendulum

In this exercise, a numerical dynamic programming strategy is used to compute a finite-time optimal control strategy for an inverted pendulum system.

System description Consider a simple inverted pendulum mounted on a movable cart, as shown in Fig 4.

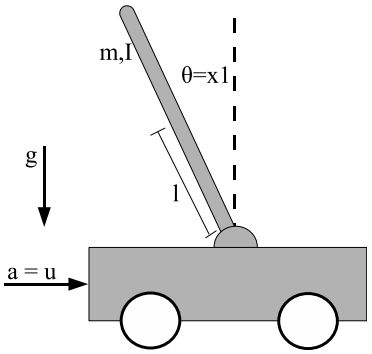


Figure 4: Inverted pendulum schematic.

The goal is to stabilize the pendulum by moving the cart. The inverted pendulum system is governed by the nonlinear dynamics

$$(I + ml^2)\ddot{\theta}_t + b\dot{\theta}_t - mgl \sin\theta_t = mlu_t \cos\theta_t$$

where u is the cart acceleration, θ is the pendulum angle and m, l, and I are physical parameters. By introducing $x_1 = \theta$ and $x_2 = \dot{\theta}$, the system can be desribed by a first order ODE system

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -a_0 \sin x_1 - a_1 x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ b_0 u \cos x_1 \end{bmatrix}$$

where $a_0 = -\frac{mgl}{I+ml^2}$, $a_1 = \frac{b}{I+ml^2}$ and $b_0 = \frac{ml}{I+ml^2}$. Throughout, let $a_0 = -2.5$, $a_1 = 0.05$ and $b_0 = 0.25$. Linearizing the system around $x_1 = 0$, $x_2 = 0$ gives

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ b_0 \end{bmatrix} u := Ax + Bu$$

Further, the system can be discretized to obtain

$$x_{k+1} = \Phi x_k + \Gamma u_k$$

Optimal control formulation The aim of this exercise is to compute a finite-time optimal control strategy for this system. The following linear-quadratic formulation will be used:

$$\begin{aligned} \underset{u_k}{\text{minimize}} \quad & \sum_{k=1}^{N} x_k^T Q x_k + u_k^T R u_k \\ \text{subject to} \quad & x_{k+1} = \Phi x_k + \Gamma u_k \end{aligned} \tag{13}$$

As N grows large, the above formulation becomes a good approximation of the discrete linear-quadratic regulator. Q and R can be tuned to balance the level of stabilization with the size of the control input. Here, the problem (13) will be solved numerically using a dynamic programming approach.

Dynamic programming through gridding The problem (13) can be solved using a dynamic programming recursion:

$$J_k(x_k) = \min_{u_k} \{x_k^T Q_k x_k + u_k^T R u_k + J_{k+1}(\Phi x_k + \Gamma u_k)\}$$

The difficulty in running the recursion numerically is that the cost-to-go function $J_{k+1}(x_{k+1})$ is not readily available. It is possible to approximate the cost-to-go function by gridding the state space as follows:

```
X1 = linspace(-pi/4,pi/4,10);
X2 = linspace(-pi/2,pi/2,5);
```

and then calculate the cost-to-go at the grid points. The MATLAB function `interp2` can then be used to obtain cost-to-go values for states between grid points. This allows the use of MATLAB's built in optimization functions, such as `fminunc`. Hence, the optimal control input and optimal cost will be computable in each grid point. For an example, consider

```
U = zeros(length(X1),length(X2));
J = zeros(length(X1),length(X2));
% Stage cost
Jstage = @(x,u) x'*Q*x + u'*R*u;
% Initial cost-to-go
Jtogo = @(x) x'*Q*x;
% Calculate optimal solution at a single grid point
options = optimoptions('fminunc','Display','off');
x = [X1(1),X2(1)];
[U(1,1),J(1,1)] = fminunc( @(u) Jstage(x,u) + Jtogo(Phi*x+Gamma*u), 0 );
```

If this is performed at every gridpoint, an improved representation of the cost-to-go can be formed as follows:

```
% new cost-to-go
Jtogo = @(x) interp2(X1,X2,J',x(1),x(2),'spline');
```

Simulation If the optimal control input is available at every defined grid point and time instance the system can be simulated as follows:

```
x0 = [pi/6; 0];
ts = [];
```

```
us = [];
xs = [];
Js = [0];
x = x0;
for k = 1:N
    Jtogo = @(x) interp2(theta, thetadot, J(:,:,N+1-k)', x(1), x(2),'spline');
    u = fminunc( @(u) Jstage(x,u) + Jtogo(Ad*x+Bd*u), 0, options);
    f = @(t,x) [x(2); -a0*sin(x(1))-a1*x(2) + b0*u*cos(x(1))];
    [t,x] = ode45(f,[(k-1)*Ts,k*Ts], x);
    ts = [ts t'];
    xs = [xs x'];
    for i = 1:size(x,1)
        Js = [Js Js(end) + x(i,:)*Q*x(i,:)' + u'*R*u];
    end
    us = [us u*ones(1,length(t))];
    x = xs(:,end);
end
Js = Js(2:end)
figure(2)
subplot(3,1,1)
plot(ts,xs)
subplot(3,1,2)
plot(ts,us)
subplot(3,1,3)
plot(ts,Js)
```

(a) Discretize the system at a sampling time of 0.1 seconds using `c2d` and implement a dynamic programming scheme using the gridding technique introduced above.

(b) Compute the optimal cost as $N \to \infty$ as follows:

```
X1_lqr = linspace(-pi/4,pi/4,100);
X2_lqr = linspace(-pi/2,pi/2,50);
J_lqr = zeros(length(X1_lqr),length(X2_lqr));
[K,P] = dlqr(Phi,Gamma,Q,R);
for i = 1:length(X1_lqr)
  for j = 1:length(X2_lqr)
    x = [X1_lqr(i); X2_lqr(j)];
    J_lqr(i,j) = x'*P*x;
  end
end
surf(X1_lqr,X2_lqr,J_lqr')
hold on
```

and compare with the approximated cost-to-go from running the DP recursion.

```
    surf(X1, X2, J(:,:,N)')
```

Investigate the impact of the following parameters:

(i) The value of N (Try $N = 1$ and $N = 10$ for example)

(ii) The coarseness of the discretization

(iii) The values of Q and R

(c) Compute an optimal controller using your implementation, with the following parameters:

```
X1 = linspace(-pi/4,pi/4,10);
X2 = linspace(-pi/2,pi/2,5);
N = 10;
Q = diag([1 0]);
R = 1;
```

Simulate the system from the initial point $x_0 = \begin{pmatrix} \frac{\pi}{6} & 0 \end{pmatrix}$ with your computed optimal controller for 5 seconds. Is the system stabilized? What if $x_0 = \begin{pmatrix} \frac{\pi}{3} & 0 \end{pmatrix}$?

(d) Repeat the above procedure for other values of Q and R and investigate the effect on the behaviour of the closed-loop system (Some may require a larger N or a longer simulation time):

(i) `Q = diag([1 2]), R = 1`

(ii) `Q = diag([1 0]), R = 1/100`

(iii) `Q = diag([1 2]), R = 1/100`

(iv) `Q = diag([1 0]), R = 20`

4 Linear quadratic regulator

Exercise 4.1 Scalar LQR

Consider the general scalar linear-quadratic problem

$$\begin{aligned} \underset{u_0,\ldots,u_{N-1}}{\text{minimize}} \quad & \sum_{n=0}^{N-1} qx_n^2 + ru_n^2 + x_N^2 \\ \text{subject to} \quad & x_{n+1} = ax_n + bu_n \\ & x_0 \text{ given} \end{aligned}$$

where $x_{n+1} = ax_n + bu_n$ is controllable and $q > 0$, $r > 0$, $r \neq 0$, $r \neq 0$.

(a) Using dynamic programming, derive an optimal control law as a closed form recursion.

(b) Under the given assumptions, there exists an optimal linear state feedback to the infinite-horizon problem

$$\begin{aligned} \underset{u_n}{\text{minimize}} \quad & \sum_{n=0}^{\infty} qx_n^2 + ru_n^2 \\ \text{subject to} \quad & x_{n+1} = ax_n + bu_n \\ & x_0 \text{ given} \end{aligned}$$

Derive the optimal feedback law from the optimal input sequence in the finite-horizon problem.

Exercise 4.2 Optimal control law [7]

Consider the discrete-time LTI system defined by

$$x_{i+1} = Ax_i + Bu_i, \quad y_i = Cx_i$$

with

$$A = \begin{bmatrix} 4/3 & -2/3 \\ 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} -2/3 & 1 \end{bmatrix}$$

(a) Compute the optimal control law that minimizes the following cost

$$V = \sum_{i=0}^{N-1} (x_i^T Q x_i + u_i^T R u_i) + x_N^T Q_N x_N$$

with

$$Q = C^T C + 0.0001 I_{2\times 2}, \quad R = 0.001, \quad Q_N = Q$$

using discrete-time Riccati recursion.

(b) Compute the closed-loop state trajectory in a receding horizon fashion from state $x = [10\ 10]^T$. Find the minimum horizon length $N^\star$ that stabilizes the system.

(c) Plotting the prediction, motivate why increasing the horizon stabilizes the closed loop system.

Exercise 4.3 Linear quadratic regulator [7]

Consider again the system presented in Exercise 3.2.

(a) Implement the infinite horizon LQR controller $u_t = K_\infty x_t$.
Hint: Use `dlqr` Matlab function to compute the LQR controller.

(b) Compute the infinite horizon cost for the system in closed loop with $u_t = K_\infty x_t$ and compare it with the infinite horizon cost for the system in closed loop with $u_t = K_{N^\star} x_t$, where $K_{N^\star}$ is the control law computed in the Exercise 4.2. Which one gives a lower cost?
Hint: You can approximate the infinite horizon cost for the closed loop system numerically using a long state and input trajectory:

$$V_\infty = \sum_{i=0}^{\infty} (x_i^T Q x_i + x_i^T K^T R K x_i) \approx \sum_{i=0}^{1000} (x_i^T Q x_i + x_i^T K^T R K x_i).$$

Exercise 4.4 Solving the Riccati equation

We consider the system

$$x_{t+1} = \begin{bmatrix} 0.8 & 0 \\ 1 & 0.4 \end{bmatrix} x_t + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u_t.$$

Our goal is to design the optimal LQR controller by solving Riccati equation. Let the performance weights be given by

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix} \qquad\qquad R = \begin{bmatrix} 20 & 0 \\ 0 & 20 \end{bmatrix}$$

(a) Write down the Riccatti equation based on which we calculate the optimal feedback gain.

(b) Write down the short Matlab code that solves the previous equation, and calculates the optimal feedback gain using the obtained solution.

(c) Calculate the optimal feedback gain directly using function `dlqr`, and compare the obtained solutions.

Exercise 4.5 Cheap and expensive control

We consider the system

$$x_{t+1} = 0.9 x_t + u_t.$$

To design the optimal feedback controller, the performance weights

$$Q = 1 \qquad\qquad R = \rho$$

are chosen. Coefficient ρ can take three different values

$$\text{(i) } \rho = 2 \qquad\qquad \text{(ii) } \rho = 10 \qquad\qquad \text{(iii) } \rho = 100$$

Based on these values of ρ, three optimal controllers were designed. These controllers were then implemented on the process. The responses of the three feedback systems on the initial condition $x(0) = 10$ are shown in Figure 5, and control efforts are shown in Figure 6. Pair the responses and control efforts with different values of ρ.

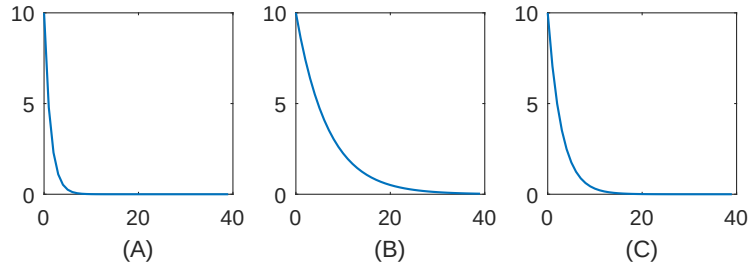


Figure 5: Evolution of the state x for three feedback systems (Exercise 4.5).

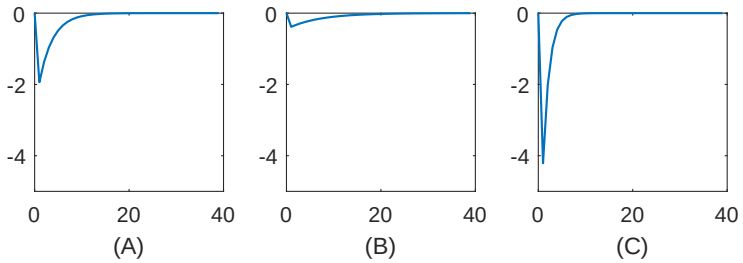


Figure 6: Evolution of the control input u for three feedback systems (Exercise 4.5).

Exercise 4.6 Designing performance weights

The dynamics of a DVD servo can be modeled as a discrete linear system

$$x_{t+1} = \begin{bmatrix} 1.0001 & -0.0061 \\ 0.0016 & 0.9978 \end{bmatrix} x_t + \begin{bmatrix} -0.0001 \\ -0.0104 \end{bmatrix} u_t$$
$$y_t = \begin{bmatrix} 3354 & 5.40 \end{bmatrix} x_t.$$

The sampling time is $h = 10^{-5}$[s]. The goal is to design the LQR controller $u_t = -Lx_t + l_r r_t$ that satisfies the following specifications:

- Rise time of the system should be below 0.004 s.

- Control input should remain within the interval $[-0.003, 0.003]$
- Static gain from the input to the output of the system should equal 1.

Find the weights Q, ρ, and l_r such that the aforementioned specifications are satisfied.

Exercise 4.7 LQR for a triple accumulator. [2]

We consider the system

$$x_{t+1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u_t$$
$$y_t = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} x_t$$

This system is called a triple accumulator, since it consists of a cascade of three accumulators. (An accumulator is the discrete- time analog of an integrator: its output is the running sum of its input.) We use the LQR cost function

$$J = \sum_{t=0}^{N-1} u_t^2 + \sum_{t=0}^{N} y_t^2 \qquad\qquad N = 50.$$

Find P_t (numerically), and verify that the Riccati recursion converges to a steady-state value in fewer than about 10 steps. Find the optimal time-varying state feedback gain K_t, and plot its components $(K_t)_{11}$, $(K_t)_{12}$, and $(K_t)_{13}$, versus t.

Exercise 4.8 Integral action

We consider the system

$$x_{t+1} = \begin{bmatrix} 1 & 0 \\ 2 & 3 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_t$$
$$y_t = \begin{bmatrix} 0 & 1 \end{bmatrix} x_t$$

(a) Design LQR controller of the form $u_t = -Lx_t - l_r r_t$ using Matlab. The performance weights

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad\qquad R = 1$$

should be used, and static gain from r to y should equal 1.

(b) Assume now that the there exists disturbance acting on the input, so $u_t' = u_t + d$. Assume $r_t = 1$ and $d_t = 1$. Do you expect the static error $\lim_{t\to\infty} e_t = \lim_{t\to\infty} y_t - r_t$ to be present? Simulate the response of y_t to verify or disprove intuition.

(c) Introduce additional integral state x_t' in the system and design LQR controller $u_t = -Lx_t - l_r x_t'$ using Matlab. The performance weights

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad\qquad R = 1$$

should be used.

(d) Assume again $d_t = 1$ and $r_t = 1$. Do you expect the static error to be present now? Simulate the response of y_t to verify or disprove intuition.

Exercise 4.9 Disturbance rejection

We consider the system

$$x_{t+1} = \begin{bmatrix} 0.1 & 0 \\ 0.2 & 0.3 \end{bmatrix} x_t + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u_t + \begin{bmatrix} 1 \\ 0 \end{bmatrix} (d_t + r_t)$$
$$y_t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_t$$

where d_t is a sinusoidal disturbance with natural frequency $\omega_0 = 2$ rad/s. Assume sampling time to be $h = 0.001$. Design observer of the system and LQR controller of the form $u_t = -L_x\hat{x}_t - L_\chi\chi_t + l_r r_t$ using Matlab. The goals are:

- to achieve disturbance rejection;
- gain from reference r_t to first input should be equal to 1.

The performance weights for the filter and the observer can be chosen arbitrarily and the result should be verified through the simulations.

Exercise 4.10 Time-invariant LQR gain [2]

Consider a discrete-time LQR problem with horizon $t = N$, with optimal input $u(t) = K_t x(t)$. Is there a choice of the terminal cost Q_f (symmetric and positive semidefinite, of course) for which K_t is constant, i.e., $K_0 = ... = K_{N-1}$?

Exercise 4.11 Kalman Filter

We consider a scalar system

$$x_{t+1} = ax_t + v_t$$
$$y_t = x_t + w_t$$

where $x_t \in \mathbb{R}$ is the system state, $y_t \in \mathbb{R}$ is the measurement, and $v_t \in \mathbb{R}$ and $w_t \in \mathbb{R}$ are the process and measurement noises respectively. Given a sequence of observations $\{y_0, \ldots, y_{N-1}\}$, our goal is to determine the optimal sequence of states that minimizes the size of disturbance sequences required to explain the observations. That is, the sequence of state estimates should minimize the cost

$$J_N(x_0, \ldots, x_N) = R_0(x_0 - \mu_0)^2 + \sum_{k=0}^{N-1} R_1 w_k^2 + R_2 v_k^2 \tag{14}$$

To derive the optimal estimator, we use the dynamic programing approach. We want to show by induction that arrival costs for any t have the form

$$W_t(x_t) = S_t(x_t - \hat{x}_{t|t-1})^2 + c_t$$

where P_t, c_t, and $\hat{x}_{t|t-1}$ depend only on problem data, P_{t-1}, c_{t-1}, and $\hat{x}_{t-1|t-2}$.

(a) *Initial Step.* Show that W_0 can be written in the form

$$W_0(x_0) = S_0(x_0 - \hat{x}_{0|-1})^2 + c_0.$$

Inductive Step. Assume now an arbitrary t. Bellman equation can be written in the form

$$W_{t+1}(x_{t+1}) = \min_{x_t}\{W_t(x_t) + q_1(x_t, y_t) + q_2(x_{t+1}, x_t)\}$$

(b) Find $q_1(x_t, y_t)$ and $q_2(x_{t+1}, x_t)$.

(c) Assume $W_t(x_t) = S_t(x_t - \hat{x}_{t|t-1})^2 + c_t$. Find minimizer $x_{t|t}$ of

$$W_t^+(x_t) = W_t(x_t) + q_1(x_t, y_t).$$

(d) Show that $W_t^+(x_t)$ can be written as

$$W_t^+(x_t) = S_t^+(x_t - \hat{x}_{t|t})^2 + c_t^+$$

(e) Now consider again

$$W_{t+1}(x_{t+1}) = \min_{x_t}\{W_t^+(x_t) + q_2(x_{t+1}, x_t)\}$$

find minimizer x_t^* of this cost.

(f) Denote by $\hat{x}_{t+1|t} = Ax_t^*$. Finalize the proof by showing that

$$W_{t+1}(x_{t+1}) = S_{t+1}(x_{t+1} - \hat{x}_{t+1|t})^2 + c_{t+1}$$

(g) Write down recursive procedure for obtaining a sequence of state estimates.

(h) Can you explain the meaning of $\hat{x}_{t|t}$ and $\hat{x}_{t+1|t}$?

Exercise 4.12 Closed-loop stability for receding horizon LQR. [2]

We consider the system

$$x_{t+1} = \begin{bmatrix} 1 & 0.4 & 0 & 0 \\ -0.6 & 1 & 0.4 & 0 \\ 0 & 0.4 & 1 & -0.6 \\ 0 & 0 & 0.4 & 1 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u_t$$
$$y_t = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} x_t.$$

Consider the receding horizon LQR control problem with horizon T, using state cost matrix $Q = C^T C$ and control cost $R = 1$. For each T, the receding horizon control has linear state feedback form

$$u_t = K_T x_t.$$

The associated closed-loop system has the form

$$x_{t+1} = (A + BK_T)x_t.$$

What is the smallest horizon T for which the closed-loop system is stable?

Interpretation. As you increase T, receding horizon control becomes less myopic and greedy; it takes into account the effects of current actions on the long-term system behavior. If the horizon is too short, the actions taken can result in an unstable closed-loop system.

Exercise 4.13 LQR with exponential weighting. [2]

A common variation on the LQR problem includes explicit time-varying weighting factors on the state and input costs,

$$J = \sum_{\tau=0}^{N-1} \gamma^\tau (x_\tau^T Q x_\tau + u_\tau^T R u_\tau) + \gamma^N x_N^T Q_N x_N,$$

where $x_{t+1} = Ax_t + Bu_t$, $y_t = Cx_t$, x_0 is given, and, as usual, we assume $Q = Q^T \geq 0$, $Q_N = Q_N^T \geq 0$, and $R = R^T > 0$ are constant. The parameter γ, called the exponential weighting factor, is positive. For $\gamma = 1$, this reduces to the standard LQR cost function. For $\gamma < 1$, the penalty for future state and input deviations is smaller than in the present; in this case we call γ the *discount factor* or *forgetting factor*. When $\gamma > 1$, future costs are accentuated compared to present costs. This gives added incentive for the input to steer the state towards zero quickly.

(a) Note that we can find the input sequence $u_0^\star, ..., u_{N-1}^\star$ that minimizes J using standard LQR methods, by considering the state and input costs as time varying, with $Q_t = \gamma^t Q$, $R_t = \gamma^t R$, and final cost given by $\gamma^N Q_N$. Thus, we know at least one way to solve the exponentially weighted LQR problem. Use this method to find the recursive equations that give $u^\star$.

(b) Exponential weights can also be incorporated directly into a dynamic programming formulation. We define

$$W_t(z) = \min \sum_{\tau=t}^{N-1} \gamma^{\tau-t} (x_\tau^T Q x_\tau + u_\tau^T R u_\tau) + \gamma^{N-t} x_N^T Q_N x_N,$$

where $x_t = z$, $x_{\tau+1} = Ax_\tau + Bu_\tau$, and the minimum is over $u_t, ..., u_{N-1}$. This is the minimum cost-to-go, if we started in state z at time t, with the time weighting also starting at t. Argue that we have

$$W_N(z) = x_N^T Q_N x_N$$

$$W_t(z) = \min_w \big(z^T Q z + w^T R w + \gamma W_{t+1}(Az + Bw)\big),$$

and that the minimizing w is in fact $u_t^\star$. In other words, work out a backwards recursion for W_t, and give an expression for $u_t^\star$ in terms of W_t. Show that this method yields the same $u^\star$ as the first method.

(c) Yet another method can be used to find $u^\star$. Define a new system as

$$y_{t+1} = \gamma^{1/2} A y_t + \gamma^{1/2} B z_t$$

Argue that we have $y_t = \gamma^{t/2} x_t$, provided $z_t = \gamma^{t/2} u_t$, for $t = 0, ..., N-1$. With this choice of z, the exponentially weighted LQR cost J for the original system is given by

$$J = \sum_{\tau=0}^{N-1} (y_\tau^T Q y_\tau + z_\tau^T R z_\tau) + y_N^T Q_N y_N,$$

i.e., the unweighted LQR cost for the modified system. We can use the standard formulas to obtain the optimal input for the modified system

$z^\star$, and from this, we can get $u^\star$. Do this, and verify that once again, you get the same $u^\star$.

Exercise 4.14 Estimating the cost of a control law Exam: 2016-10-22

In class, we have learned how to compute the optimal control law for the criterion

$$J_{N,t}(x_t) = \sum_{k=t}^{N} x_k^T Q_1 x_k + u_k^T Q_2 u_k$$

when the underlying system dynamics is linear and on the form

$$x_{t+1} = Ax_t + Bu_t$$

When N tends to infinity, the optimal control law is a linear state-feedback.

In this problem, we are interested in estimating the quadratic cost for an arbitrary *given* (not necessarily optimal) linear control law

$$u_t = -Lx_t$$

Of course, if N is small and we are only interested in one particular initial value x_0, we could simply simulate the control law and evaluate the cost. But as N grows large, this is increasingly inefficient. We will therefore derive an alternative approach.

(a) Use (backward) induction to show that $J_{N,0}(x_0)$ is a quadratic function of the initial state, *i.e.* $J_{N,0}(x_0) = x_0^T V_{N,0} x_0$ for some matrix $V_{N,0}$. Derive a recursion for how $V_{N,k}$ depends on $V_{N,k+1}$. Make sure to specify how the recursion should be initialized (how $V_{N,N}$ should be selected).

(b) Assume that the iteration converges as $N \to \infty$, *i.e.* $V_{N,k} = V_{N,k+1} = V_\infty$ for all finite values of k. Derive a matrix equation that determines V_∞ in terms of A, B and L. Explain how you can solve the equation that you derived using a numerical routine for solving Lyapunov equations (i.e. a numerical routine for computing P such that $\Phi^T P \Phi - P + Q = 0$ for given Φ and Q).

Exercise 4.15 Observability characterization Exam: 2016-12-19

Observability characterizes our ability to estimate the initial state from measurements of the system output. A disadvantage of the observability tests that we have discussed in class is that they are binary: systems are either observable or not. In practice, however, systems can be observable to varying degree, and different initial states can be more or less hard to estimate. In this problem, we will investigate one observability measure that can make these distinctions.

Consider an autonomous discrete-time system

$$\begin{aligned} x_{t+1} &= Ax_t \\ y_t &= Cx_t \end{aligned}$$

If the system is not observable, then there will be initial states x_0 which do not generate any output energy, i.e. for which

$$V_t = \sum_{k=0}^{t} y_k^T y_k \tag{15}$$

is zero. Initial states x_0 which generate little output energy can be assumed hard to estimate, while initial states which generate a lot of output energy are easier to estimate.

Of course, for a given initial value, one can estimate the quantity (15) using simulation. But such a simulation can take long time if t is large, and one may need to redo the simulation for many different initial values to gain an understanding of the system.

(a) Use forward induction to show that V_t is a quadratic function of x_0, i.e.

$$V_t = x_0^T S_t x_0$$

for some positive (semi-)definite matrix S_t. Derive a recursion for how S_{t+1} depends on the system matrices A and C, and S_t.

(b) Assume that the iteration converges, in the sense that $S_t \to S$ as $t \to \infty$. Then

$$V_\infty = \sum_{k=0}^{\infty} y_k^T y_k = x_0^T S x_0$$

so the infinite sum is readily evaluated, and we can also see which initial values generate a lot (and a little) output energy. Derive a matrix equation for computing S.

(c) Consider the system given by the matrices

$$A = \begin{bmatrix} 0.8 & a \\ 0 & 0.1 \end{bmatrix}, \quad C = \begin{bmatrix} 0.6 & 0 \end{bmatrix}.$$

Without carrying out any numerical computations, what structure would you expect S to have when $a = 0$? Justify your answer!

(d) Let instead $a = 0.115$. Compute the matrix S and determine the ratio between the output energy generated from initial value $x_0 = \begin{pmatrix} 1 & 0 \end{pmatrix}$ and initial value $x_0 = \begin{pmatrix} 0 & 1 \end{pmatrix}$.

Exercise 4.16 Input-constrained LQR Exam: 2017-12-18

Consider the constrained optimal control problem

$$\begin{aligned} \underset{u_k}{\text{minimize}} \quad & \sum_{k=0}^{\infty} x_k^2 + \sum_{k=0}^{\infty} u_k^2 \\ \text{subject to} \quad & x_{k+1} = x_k + \frac{3}{2} u_k \\ & |u_k| \leq 1 \\ & x_0 \text{ given} \end{aligned}$$

over an infinite horizon.

(a) Determine an optimal feedback law to the corresponding unconstrained problem

$$\begin{aligned} \underset{u_k}{\text{minimize}} \quad & \sum_{k=0}^{\infty} x_k^2 + \sum_{k=0}^{\infty} u_k^2 \\ \text{subject to} \quad & x_{k+1} = x_k + \frac{3}{2} u_k \\ & x_0 \text{ given} \end{aligned}$$

(b) Determine an approximate optimal feedback law to the infinite-horizon problem by solving

$$\begin{aligned} \underset{u_k}{\text{minimize}} \quad & \sum_{k=0}^{N} x_k^2 + \sum_{k=0}^{N} u_k^2 \\ \text{subject to} \quad & x_{k+1} = x_k + \frac{3}{2} u_k \\ & |u_k| \leq 1 \\ & x_0 \text{ given} \end{aligned}$$

using dynamic programming, with a quadratic value function

$$V_{k+1}(x_{k+1}) = p_{k+1} x_{k+1}^2$$

as a starting guess. Next, determine the optimal feedback law to the infinite-horizon problem by letting $N \to \infty$.
Hint: The optimal feedback $\hat{u}$ for the unconstrained problem is still optimal if $|\hat{u}| \leq 1$.

(c) Show that the controller you computed in (b) is stabilizing, regardless of the initial state x_0. Note that, if necessary, you can use LQR stability without proof.

Exercise 4.17 LQR cost-to-go Exam: 2018-10-20

In class, we claimed that the cost-to-go function for the infinite-horizon LQR problem is quadratic, but we never actually showed it. This problem gives you the opportunity to formally show this fact. Specifically, we will show that for the infinite-horizon LQR problem, the minimum cost-to-go starting in state x_0 is a quadratic form in x_0.

Consider a reachable discrete-time linear system

$$x_{t+1} = Ax_t + Bu_t,$$

let $Q_1 \succeq 0$, $Q_2 \succ 0$ and define the cost

$$J(u, x_0) = \sum_{i=0}^{\infty} (x_t^T Q_1 x_t + u_t^T Q_2 u_t)$$

where $x_t \in \mathbb{R}^n$ and $u_t \in \mathbb{R}^m$. Let u denote an infinite sequence $\{u_0, u_1, \ldots\}$ and

$$V(x_0) = \min_u J(u, x_0)$$

be the associated value function for initial value x_0.

(a) Use the fact that

$$x_t = A^t x_0 + \sum_{i=1}^{t} A^{i-1} B u_{t-i}.$$

to show that for all $\lambda \in \mathbb{R},\ J(\lambda u, \lambda x_0) = \lambda^2 J(u, x_0)$, and conclude that

$$V(\lambda x_0) = \lambda^2 V(x_0). \tag{16}$$

(b) Let u and $\tilde{u}$ be two input sequences, and let x_0 and $\tilde{x}_0$ be two initial states. Show that

$$J(u + \tilde{u}, x_0 + \tilde{x}_0) + J(u - \tilde{u}, x_0 - \tilde{x}_0) = 2J(u, x_0) + 2J(\tilde{u}, \tilde{x}_0)$$

Demonstrate how minimizing the right-hand-side with respect to u and $\tilde{u}$ allows to conclude that

$$V(x_0 + \tilde{x}_0) + V(x_0 - \tilde{x}_0) \leq 2V(x_0) + 2V(\tilde{x}_0)$$

(c) Let $x_0 = (x_0' + \tilde{x}_0')/2$ and $\tilde{x}_0 = (x_0' - \tilde{x}_0')/2$. Show how the above inequality implies that

$$V(x_0 + \tilde{x}_0) + V(x_0 - \tilde{x}_0) = 2V(x_0) + 2V(\tilde{x}_0) \tag{17}$$

(d) Recall that $V : \mathbb{R}^n \mapsto \mathbb{R}$ and that $\nabla V(z) = \frac{d}{dz} V(z)$. Show how (17) can be used to derive the following relation

$$\nabla V(x_0 + \tilde{x}_0) = \nabla V(x_0) + \nabla V(\tilde{x}_0) \tag{18}$$

Hint: Consider adding the gradients of (17) *with respect to* x_0 *and* $\tilde{x}_0$.

(e) Demonstrate how (16) implies that

$$\nabla V(\lambda x_0) = \lambda \nabla V(x_0) \tag{19}$$

(f) The results (18) and (19) show that $\nabla V(x_0)$ is linear in x_0. Therefore, $\exists M \in \mathbb{R}^{n \times n}$ such that $\nabla V(x_0) = M x_0$.
Next, to conclude the proof and show that the cost-to-go is quadratic in x_0, show that $V(x_0) = x_0^T P x_0$, where $P = \frac{1}{4}(M + M^T)$. Show that $P \succ 0$.

Hint:

$$V(x_0) = V(0) + \int_0^1 \nabla V(\theta x_0)^T x_0 \, d\theta$$

Exercise 4.18 LQR with reference tracking Exam: 2018-12-18

This problem considers an extension of linear-quadratic regulation to tracking of a reference trajectory. To that end, consider a discrete-time linear system

$$x_{t+1} = A x_t + B u_t$$

with performance criterion

$$J = \sum_{k=0}^{N} (x_k - x_k^{\text{ref}})^T Q_1 (x_k - x_k^{\text{ref}}) + \sum_{k=0}^{N-1} u_k^T Q_2 u_k$$

where $Q_1 \succ 0$ and $Q_2 \succ 0$. The reference trajectory $\{x_k^{\text{ref}}\}$ is assumed to be known. We want to use dynamic programming to derive the optimal control law and show that it consists of a feedback term from the system state x_k and feed-forward term from the reference state x_k^{ref}.

(a) Show that the cost-to-go function at time N, $V_N(x)$ has the form

$$V_N(x) = x^T P_N x + 2q_N^T x + r_N$$

for some $P_N \succ 0$, q_N and r_N. (1p)

(b) Assume that

$$V_{t+1}(x) = x^T P_{t+1} x + 2q_{t+1}^T x + r_{t+1}$$

with $P_{t+1} \succ 0$. Show that the optimal control at time t is on the form

$$u_t^\star = -L_t x_t + m_t$$

where

$$L_t = (Q_2 + B^T P_{t+1} B)^{-1} B^T P_{t+1} A, \qquad m_t = -(Q_2 + B^T P_{t+1} B)^{-1} B^T q_{t+1}$$

(c) Use dynamic programming (backward induction) to show that $V_N, V_{N-1}, \ldots V_0$ all have the given form. Verify that

$$\begin{aligned}
P_t &= Q_1 + A^T P_{t+1} A - A^T P_{t+1} B (Q_2 + B^T P_{t+1} B)^{-1} B^T P_{t+1} A, \\
q_t &= (A - BL_t)^T q_{t+1} - Q_1 x_t^{\text{ref}} \\
r_t &= r_{t+1} + (x_t^{\text{ref}})^T Q_1 x_t^{\text{ref}} + q_{t+1}^T B m_t
\end{aligned}$$

5 Model predictive control

Exercise 5.1 Pair the weights with responses

Consider an arbitrary discrete system:

$$x_{t+1} = \begin{bmatrix} 1.0015 & 0.1465 \\ 0.0183 & 0.8367 \end{bmatrix} x_t + \begin{bmatrix} 0.1027 \\ 0.2755 \end{bmatrix} u_t$$

The model predictive controller has a horizon $N = 10$ and a sampling time $dt = 0.2$. If the weights in the cost function are

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1,$$

then the response to the closed loop MPC system looks like:

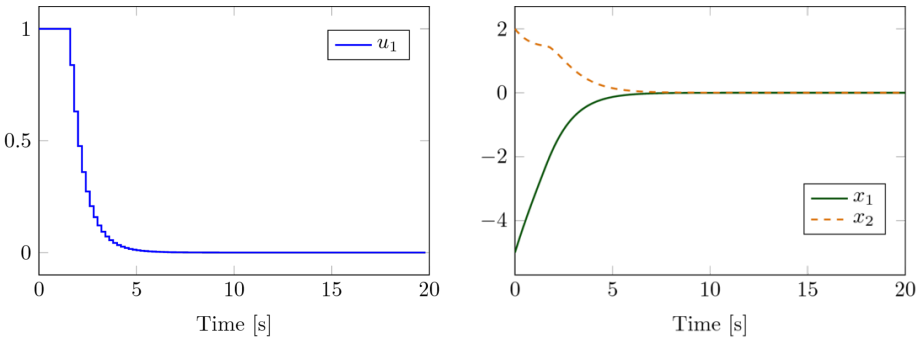


Figure 7: The response to the system in Exercise 1.

(a) Pair the following cost-function matrices with responses A, B and C shown in Figure 8.

(i)

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix}, \quad R = 1,$$

(ii)

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 10,$$

(iii)

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1,$$

Exercise 5.2 MPC as a quadratic program

Consider an MPC problem:

$$\begin{aligned} \text{minimize} \quad & x_{t+T|t}^T Q_f x_{t+T|t} + \textstyle\sum_{k=0}^{T-1} x_{t+k|t}^T Q x_{t+k|t} + u_{t+k|t}^T R u_{t+k|t} \\ \text{subject to} \quad & x_{t+1} = Ax_t + Bu_t \\ & x_{min} \leq x_{t+i} \leq x_{max} \qquad \text{for } i = 1, \ldots, T \\ & u_{min} \leq u_{t+i} \leq u_{max} \qquad \text{for } i = 0, \ldots, T-1 \end{aligned}$$

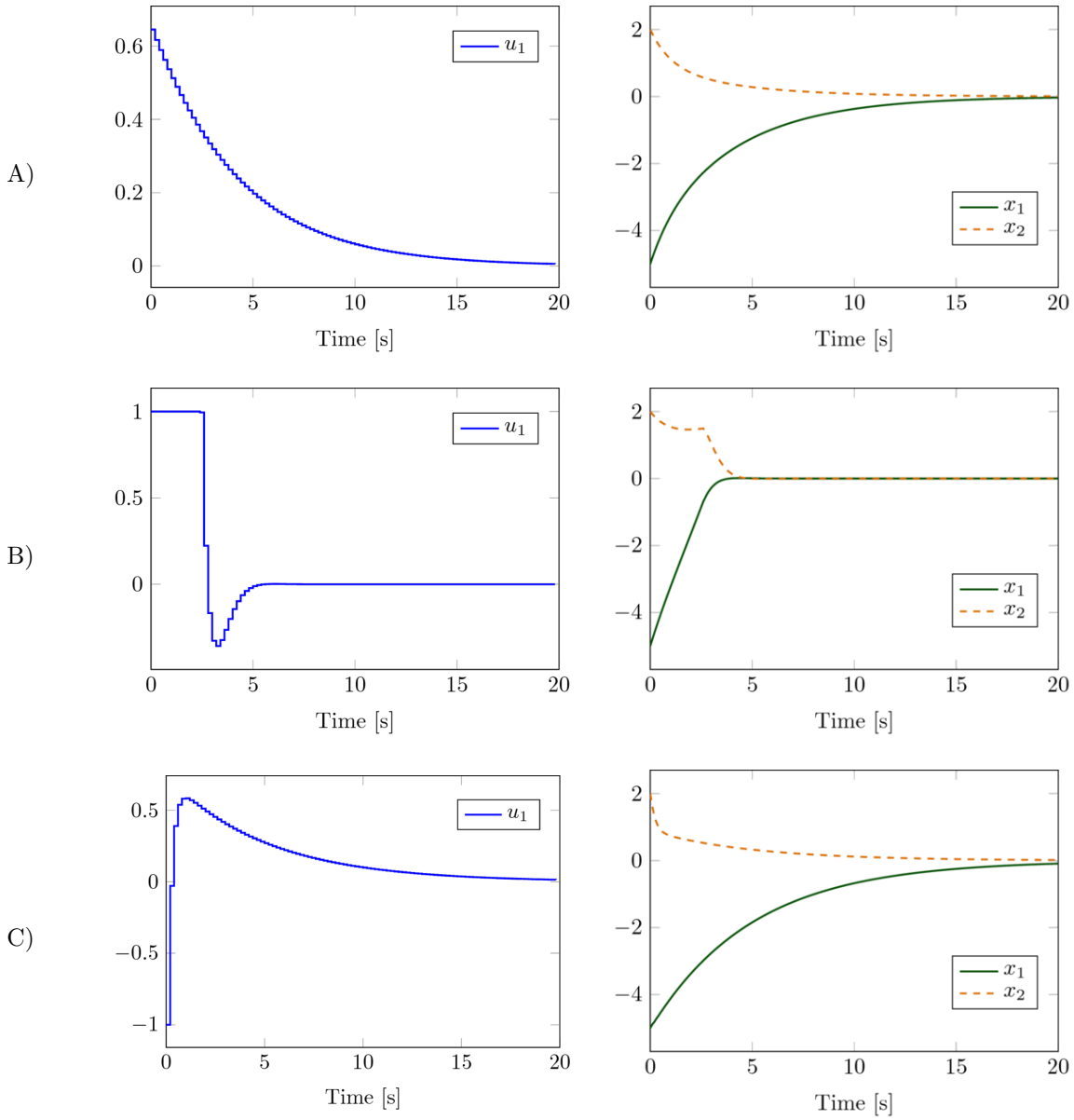


Figure 8: Input and state response pairs for the different MPC configurations in Exercise 1.

Assume that you have a system with $n = 2$, $m = 1$.

(a) Formulate this MPC as a quadratic program

$$\begin{aligned} \text{minimize} \quad & z^T H z \\ \text{subject to} \quad & Pz \leq h \\ & Cz = b. \end{aligned}$$

(b) What are the dimensions of the matrices if $T = 10$?

Exercise 5.3 System with actuator rate limitations

You are given an MPC with horizon N for a system with one state and one input, $m = 1$, $n = 1$. The state and input constraints are in the optimization problem given as:

$$Pz \leq h,$$

where

$$z = \begin{bmatrix} u_t & x_{t+1} & \dots & u_{t+N-1} & x_{t+N} \end{bmatrix}^T.$$

$$P = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ -1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ 0 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & -1 & 0 \\ 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 0 & -1 \end{bmatrix}, \quad h = \begin{bmatrix} u_{lim} \\ u_{lim} \\ x_{lim} \\ x_{lim} \\ \vdots \\ u_{lim} \\ u_{lim} \\ x_{lim} \\ x_{lim} \end{bmatrix}.$$

Your goal is to add a rate limitation on the input.

(a) Add rows in the P and h matrices so that the input rate is limited to $\pm r_{lim}$

Exercise 5.4 From infinite to finite horizon [3]

A predictive controller minimizes the cost function $J(t) = \sum_{i=0}^{\infty} (y_{t+i|t}^2 + u_{t+i|t}^2)$ at each time-step t subject to input constraints. The system output y is related to the control input u via $x_{t+1} = Ax_t + Bu_t$, $y_t = Cx_t$, where

$$A = \begin{bmatrix} -2 & 1 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

(a) Assume that the optimization problem has a finite control horizon N over which the input signals are optimized, so that the input in the optimization problem is given by

$$u_{i|t} = \begin{cases} \text{optimization variables} & i = 0, \dots, N-1 \qquad \text{(mode 1)} \\ Kx_{i|t} & i = N, \dots \qquad \text{(mode 2)} \end{cases}$$

If the mode 2 stabilizing controller is $u_t = Kx_t = \begin{bmatrix} 2 & -1 \end{bmatrix} x_t$, show that

$$J(t) = \sum_{i=0}^{N-1} \left[y_{t+i|t}^2 + u_{t+i|t}^2 \right] + x_{t+N|t}^T \begin{bmatrix} 13 & -1 \\ -1 & 2 \end{bmatrix} x_{t+N|t}.$$

(b) Show that the constraints

$$-1 \leq u_{t+i|t} \leq 2, \quad i = 0, 1, \ldots, N+1$$

ensure that the predictions satisfy $-1 \leq u_{t+i|t} \leq 2 \quad \forall i \geq 0$.

(c) Derive a bound on $J^\star(t+1) - J^\star(t)$, where $J^\star(t)$ is the optimal value of $J(t)$. Hence show that $\sum_{t=0}^{\infty} \left[y_t^2 + u_t^2\right] \leq J^\star(0)$ along trajectories of the closed loop system.

(d) Is the closed loop system stable? Explain your answer.

Exercise 5.5 Pair the weights with responses

Consider the following combinations of state dynamics and cost function matrices:

(i)	(ii)	(iii)	(iv)
$A = \begin{bmatrix} 1 & -0.1 \\ 0 & 1 \end{bmatrix}$	$A = \begin{bmatrix} 1 & -0.1 \\ 0 & 1 \end{bmatrix}$	$A = \begin{bmatrix} 1.02 & 0.5 \\ 0 & 1 \end{bmatrix}$	$A = \begin{bmatrix} 1.02 & 0.5 \\ 0 & 1 \end{bmatrix}$
$B = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$	$B = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$	$B = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$	$B = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$
$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$Q = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$	$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$Q = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$
$R = 1$	$R = 1$	$R = 1$	$R = 1$

(a) Pair the combinations with the responses in Figure 9

Exercise 5.6 The terminal constraint [3]

(a) Explain the function of terminal constraints in a model predictive control strategy for a system with input or state constraints. Define two principal properties that must be satisfied by a terminal constraint set.

(b) A discrete time system has the state space model

$$x(k+1) = Ax(k) + Bu(k), \text{ where } A = \begin{bmatrix} 0.3 & -0.9 \\ -0.4 & -2.1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

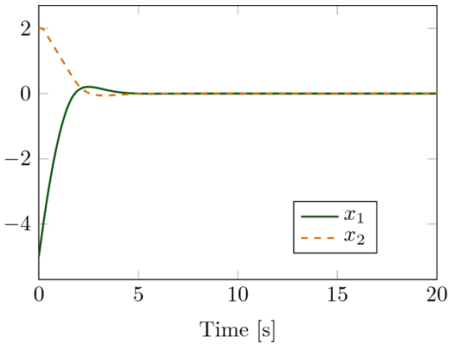
and constraints

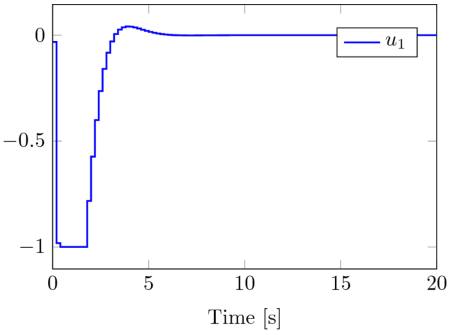
$$|x_1 + x_2| \leq 1, \quad |x_1 - x_2| \leq 1, \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

(i) If the terminal feedback law is $u(k) = Kx(k)$, $K = [0.4 \quad 1.8]$, show that the following set is a valid terminal constraint set

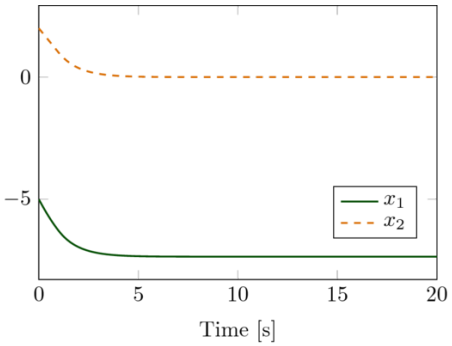
$$\{x : |x_1 + x_2| \leq 1, |x_1 - x_2| \leq 1\}.$$

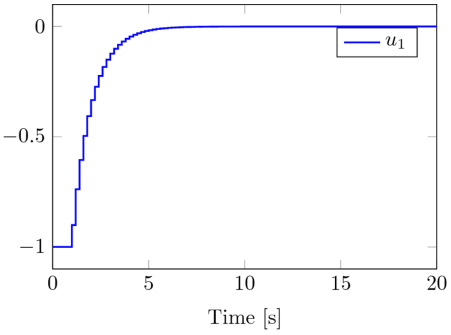
A)



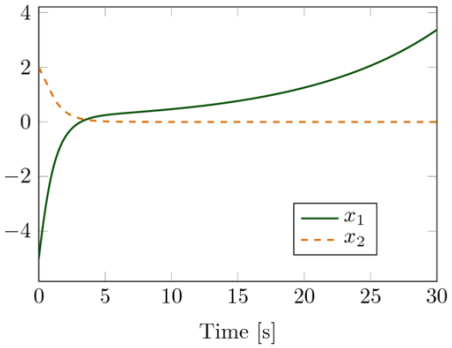


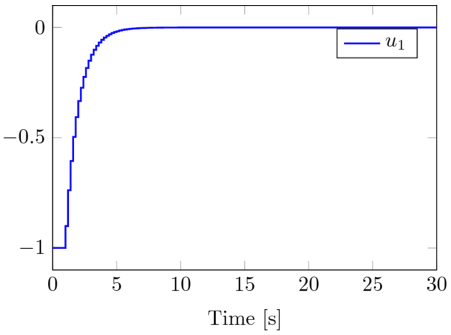
B)



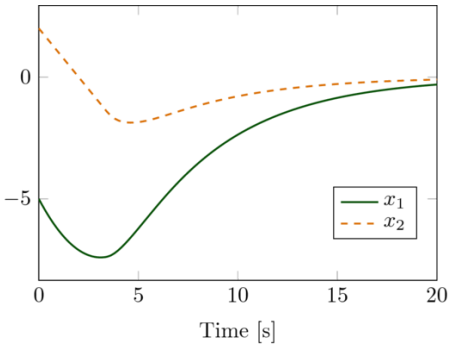


C)





D)



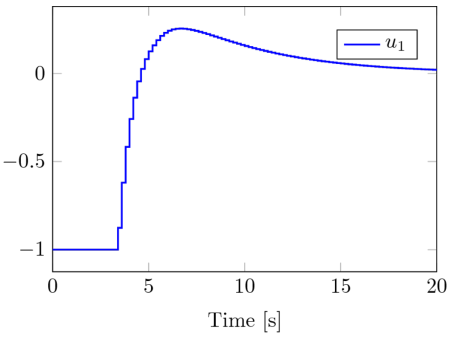


Figure 9: Input and state response pairs for the different MPC configurations in Exercise 5.

(ii) Describe a procedure for determining the largest terminal constraint set for the case of a general feedback gain K.

(c) What are the main considerations that govern the choice of the prediction horizon N?

Exercise 5.7 From output to input constraint [6]

Consider the state-space system:

$$\begin{aligned} x(k+1) &= 2x(k) + u(k), \\ y(k) &= 3x(k), \end{aligned} \tag{20}$$

with the output constraint:

$$-1 \leq y(k) \leq 2, \quad \forall k$$

If the current state estimate $x(k) = 3$ and the previous control input $u(k-1) = -1$, show that the corresponding constraint on the control signal change $\Delta u(k)$ is given by:

$$-\frac{16}{3} \leq \Delta u(k) \leq -\frac{13}{3}$$

Exercise 5.8 Predicted output as function of the input [6]

The state-space system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k \end{aligned}$$

is to be controlled using a model predictive controller. The equations describing the predicted outputs can be written on vector form as:

$$\mathcal{Y} = S_x x_k + S_{u-1} u_{k-1} + S_u \Delta\mathcal{U}$$

where:

$$\mathcal{Y} = \begin{bmatrix} \hat{y}_{k+1} \\ \hat{y}_{k+1} \\ \vdots \\ \hat{y}_{k+N} \end{bmatrix}, \quad \Delta\mathcal{U} = \begin{bmatrix} \Delta u_k \\ \Delta u_{k+1} \\ \vdots \\ \Delta u_{k+N-1} \end{bmatrix}$$

are the vectors of predicted outputs and control changes respectively.

(a) What are the structures of the matrices S_x, S_{u-1} and S_u?

(b) Give some interpretations of S_x, S_{u-1} and S_u in terms of linear systems theory.

Exercise 5.9 Unconstrained MPC stability [8]

Let the cost function be defined as

$$V(x, U) := x_N^T P x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k$$

where $x_0 = x = x(t)$ is the state at time t and

$$x_{k+1} = Ax_k + Bu_k, \quad k = 0, 1, \ldots, N-1.$$

Let the value function be

$$V^\star(x) := \min_U V(x, U)$$

and the optimal input sequence be

$$\begin{aligned} U^\star(x) &:= (u_0^\star \quad u_1^\star \quad \cdots \quad u_{N-1}^\star) \\ &:= \operatorname*{argmin}_U V(x, U) \end{aligned} \tag{21}$$

Consider also the input sequence that is constructed by shifting the optimal input sequence and appending it as follows:

$$\tilde{U}(x) := (u_1^\star \quad \cdots \quad u_{N-1}^\star \quad Kx_N), \tag{22}$$

where the terminal control law $u = Kx$ is chosen such that $A + BK$ is stable and the predicted optimal state trajectory is defined as

$$\begin{aligned} x_0^\star &= x, \\ x_{k+1}^\star &= Ax_k^\star + Bu_k^\star, \qquad k = 0,\, 1,\, \ldots,\, N-1. \end{aligned} \tag{23}$$

The optimal input sequence is implemented in a receding horizon fashion, i.e., the closed-loop system is given by

$$x(t+1) = Ax(t) + Bu_0^\star.$$

(a) Why is $V(Ax + Bu_0^\star, \tilde{U}(x))$ an upper bound for $V^\star(Ax + Bu_0^\star)$?

(b) By noting that $V^\star(x) = V(x, U^\star)$, show that

$$V(Ax + Bu_0^\star, \tilde{U}(x)) = V^\star(x) + l(x)$$

where

$$\begin{aligned} l(x) := &- x^T Q x - {u_0^\star}^T R u_0^\star - {x_N^\star}^T P x_N^\star \\ &+ {x_N^\star}^T Q x_N^\star + {x_N^\star}^T K^T R K x_N^\star \\ &+ (Ax_N^\star + BKx_N^\star)^T P (Ax_N^\star + BKx_N^\star). \end{aligned} \tag{24}$$

(c) Show that $l(x) < 0$ for all $x \neq 0$ if Q and R are positive definite and P and K are chosen to satisfy

$$(A + BK)^T P (A + BK) - P \leq -Q - K^T R K.$$

Hence, show that $V^\star(\cdot)$ is a Lyapunov function for the closed-loop system if P is also positive definite.

Exercise 5.10 Filing the patent [9]

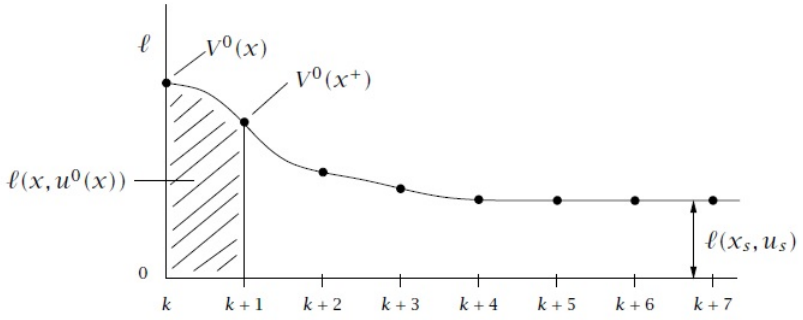


Figure 10: Stage cost versus time for the case of unreachable set-point. The cost $V^0(x(k))$ is the area under the curve to the right of time k.

An excited graduate student shows up at your office. He begins, "Look, I have discovered a great money-making scheme using MPC." You ask him to tell you about it. "Well" he says, "you told us in class that the optimal steady state is asymptotically stable even if you use the stage cost measuring distance from the unreachable set-point, right?". You reply, "Yes, that's what I said." He continues, "OK, well look at this little sketch I drew," and he shows you a picture like Figure 10. "So imagine I use the infinite horizon cost function so the open-loop and closed-loop trajectories are identical. If the best steady state is asymptotically stable, then the stage cost asymptotically approaches $l(x_s, u_s)$ right?" You reply, "I guess that looks right." He then says, "OK, well if I look at the optimal cost using state $x(k)$ at time k and state $x(k+1)$ at time $k+1$, by the principle of optimality I get the usual cost decrease"

$$V^0(x(k+1)) \leq V^0(x(k)) - l(x(k),\, u^0(x(k))) \tag{25}$$

You interrupt, "Wait, these $V^0(\cdot)$ costs are not bounded in this case!" Unfazed, the student replies, "Yeah, I realize that, but this sketch is basically correct regardless. Say we just make the horizon really long; then the costs are all finite and this equation becomes closer and closer to being true as we make the horizon longer and longer." You start to feel a little queasy at this point. The student continues, "OK, so if this inequality basically holds, $V^0(x(k))$ is decreasing with k along the closed-loop trajectory, it is bounded below for all k, it converges, and, therefore, $l(x(k),\, u^0(x(k)))$ goes to zero as k goes to ∞." You definitely don't like where this is heading, and the student finishes with, "But $l(x,\, u) = 0$ implies $x = x_{sp}$ and $u = u_{sp}$, and the set-point is supposed to be unreachable. But I have proven that infinite horizon MPC can reach an unreachable set-point. We should patent this!"

How do you respond to this student? Here are some issues to consider.

(a) Does the principle of optimality break down in the unreachable set-point case?

(b) Are the open-loop and closed-loop trajectories identical in the limit of an infinite horizon controller with an unreachable set-point?

(c) Does inequality (25) hold as $N \to \infty$? If so, how can you put it on solid footing? If not, why not, and with what do you replace it?

(d) Do you file for patent?

Exercise 5.11 From 1-norm to QP [8]

Consider the following finite horizon unconstrained optimal control problem:

$$\underset{u_0,\, u_1,\, \ldots, u_{N-1}}{\text{minimize}} \quad \sum_{i=0}^{N-1} (x_i^{\mathrm{T}} Q x_i + ||u_i||_1 + x_N^T P x_N \tag{26a}$$

$$\text{subject to} \quad x_{i+1} = Ax_i + Bu_i, \quad \forall i = 0,\, 1,\, \ldots,\, N-1, \tag{26b}$$

$$x_0 = 0. \tag{26c}$$

Show that this optimization problem can be posed as a quadratic program. (Recall that the 1-norm $||\cdot||_1$ of a vector $v \in R^n$ is defined as the sum of the absolute values of its elements, i.e.

$$||v||_1 = \sum_{j=0}^{n} |v_{(j)}|,$$

where $v_{(j)}$ denotes the j-th element of the vector v.)

Exercise 5.12 Stability and recursive feasibility [8]

Consider the problem of designing a state feedback receding horizon controller for the system $x_{t+1} = 2x_t + 3u_t$. Therefore, we want to solve the following problem

$$\underset{u_k}{\text{minimize}} \quad \sum_{k=0}^{N-1} (x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k) + x_N^T P x_N \tag{27a}$$

$$\text{subject to} \quad x_{k+1} = 2x_k + 3u_k, \quad k = 0,\, 1,\, \ldots,\, N-1, \tag{27b}$$

(a) Which range of values for the weight P in the terminal cost $x_N^{\mathrm{T}} P x_N$ and gain K in the terminal control law $u = Kx$ are sufficient to ensure that the closed-loop system is stabilized by a receding horizon controller, where the state and input weights are $Q = 10$ and $R = 2$, respectively?
Now, include the following constraints on the state and input

$$\begin{aligned} -3 &\leq u_k \leq 2, \qquad k = 0, 1, \ldots, N-1, \\ -8 &\leq x_k \leq 10, \qquad k = 1, 2, \ldots, N-1, \\ -c &\leq x_N \leq c, \end{aligned} \tag{28}$$

where c is a strictly positive scalar.

(b) For which values of c is the terminal constraint state-admissible?

(c) Given a terminal control law $u = Kx$, for which values of c is the terminal constraint input-admissible?

(d) For which values of K is the terminal constraint invariant for the closed-loop system $x_{t+1} = (A + BK)x_t$?

(e) Based on your answers to parts (a)–(d), for which values of c can one choose the terminal weight P such that the closed-loop system is stabilized by a constrained receding horizon controller, while also guaranteeing that the constraints are satisfied for all time?

Exercise 5.13 Exam 161022, question 4

We are interested in designing an MPC controller for the system

$$x(t+1) = 3x(t) + u(t),$$

where the input and the state are constrained according to

$$\begin{aligned} -20 \leq & u(t) \leq 25, \\ & |x(t)| \leq 8. \end{aligned}$$

(a) Let us first disregard the input and state constraints and formulate the standard (unconstrained) LQR problem. The cost function we wish to minimize is

$$\sum_{k=0}^{\infty} 17x_k^2 + 2u_k^2.$$

Compute the optimal feedback control law $u(k) = -Lx(k)$ for this problem.

(b) Second, we convert the LQR problem to an equivalent *unconstrained* MPC problem with control hoizon N_u and prediction horizon N. Write down the complete formulation of the equivalent *unconstrained* MPC (in the form of "minimize ..., subject to ..."). Justify why the two formulations are equivalent. Finally, determine the control law $u(t) = -Kx(t)$ for the MPC mode 2 (i.e, the control law to use from the end of the control horizon until the end of the prediction horizon) such that the predicted trajectories in the LQR and this *unconstrained* MPC are equivalent.

(c) We are now ready to introduce the input and state constraints in our MPC formulation. Compute the final state constraints necessary to guarantee the stability of the MPC formulation. Assume that the MPC mode 2 responds to the control law $u(t) = -Kx(t)$, where K has been defined in (b). Write down the resulting MPC formulation.
Note. The terminal set should be as large as possible.

Exercise 5.14 Exam 171218, question 3

Consider the discrete time system

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t, \\ y_t &= Cx_t \end{aligned}$$

with

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad B = \frac{1}{2}\begin{bmatrix} -1 \\ 1 \end{bmatrix} \qquad C = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \end{bmatrix}.$$

We want to design a model predictive controller that minimizes the cost function

$$J = \sum_{k=0}^{\infty} \frac{1}{2}(y_k^2 + u_k^2).$$

(a) Show that if $u_k = \frac{1}{\sqrt{2}} y_k$, then

$$\sum_{k=0}^{\infty} \frac{1}{2}(y_k^2 + u_k^2) = \|x_0\|^2.$$

(b) A predictive control law is defined at each time step t by

$$u_t = u_t^\star$$

where $\{u_t^\star, u_{t+1}^\star, \ldots, u_{t+N-1}^\star\}$ is the minimizing argument of

$$\sum_{k=0}^{N-1} \frac{1}{2}(y_{t+k|t}^2 + u_{t+k|t}^2) + \|x_{t+N|t}\|^2$$

Show that the control law yields a stable closed-loop system.

(c) The system is now subject to the constraint $-1 \leq y_t \leq 1$, for all t. Show that if $u_t = \frac{1}{\sqrt{2}} y_t$, then if $|y_t| \leq 1$ and $|y_{t+1}| \leq 1$, it will hold that

$$-1 \leq y_{t+k} \leq 1 \qquad \forall k \geq 0$$

(d) Now, consider the model predictive control law $u_t = u_t^\star$ where $\{u_t^\star, , \ldots, u_{t+N}^\star\}$ is the minimizing argument of

$$\begin{array}{lll} \text{minimize} & \sum_{k=0}^{N-1} \frac{1}{2}(y_{t+k|t}^2 + u_{t+k|t}^2) + \|x_{t+N|t}\|^2 & \\ \text{subject to} & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k|t} & k = 0, 1, \ldots, N \\ & Y_{t+k|t} = Cx_{t+k|t} & k = 0, 1, \ldots, N+1 \\ & -1 \leq y_{t+k|t} \leq 1, & k = 0, 1, \ldots, N+1 \end{array}$$

Will this MPC controller guarantee a stable closed-loop system?

Exercise 5.15 Implement MPC [4]

Consider the discrete-time linear time-invariant system defined by

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} 1.1 & 2 \\ 0 & 0.95 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 0.0787 \end{bmatrix} u_k \\ y_k &= \begin{bmatrix} -1 & 1 \end{bmatrix} x_k \end{aligned}$$

with constraints

$$|u_k| \leq 1.$$

(a) Implement an MPC controller for this system with a horizon of $N = 10$ and a with an objective function given by

$$J(x, u) = \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k + x_N^T Q_f x_N.$$

Use $Q = Q_f = C^T C$, $R = 1$, and $x_0 = \begin{bmatrix} 0.5 & -0.5 \end{bmatrix}^T$

(b) Choose Q_f such that the $J(x, u)$ is equivalent to the infinite horizon cost. Can you guarantee closed-loop stability of the MPC?

(c) Compute a terminal set constraint that ensures recursive feasibility and stability of the closed-loop system. Is the point $x_0 = \begin{bmatrix} 0.5 & -0.5 \end{bmatrix}$ inside the region of attraction of the MPC law, i.e., can you reach the terminal set with N steps respecting the input constraints?

Exercise 5.16 Minimum time optimal control in MATLAB

In this problem, we consider time-optimal control of a linear system with bounded controls. In other words, we are interested in finding the optimal control that takes the system from a given initial value $x(0)$ to a desired target state x_{tgt} in *minimal* time.

While this problem is not easy to solve analytically, we will see that it is rather easy to write a computer code that finds the minimal time and the associated open-loop optimal control. We proceed in the following steps:

(a) Consider a linear system with bounded controls

$$x(t+1) = Ax(t) + Bu(t), \qquad |u(t)| \leq u_{\text{lim}}$$

where $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$. Assume that the initial state x_0 is given, along with time horizon T and the target state x_{tgt}.
Write a program

```
[u, exitflag]=findCtrl(A,B,ulim,x0,xtgt,T)
```

that takes A, B, u_{lim}, x_0, x_{tgt} and a target time T as input, and returns the value 1 if there is a control sequence that drives the system state from $x(0) = x_0$ to $x(T) = x_{\text{tgt}}$, and that returns -1 if no such solution was found. If the problem is feasible, then the optimal control sequence should be returned as an $m \times T$ vector; if no solution can be found, then you can return the zero vector of the same dimensions.

Hint. Usually, optimization solvers return a status code which reflects whether an associated LP was solved to optimality or determined to be unfeasible. If you use `linprog` in MATLAB, then you can read the returned value of `exitflag`. If you use `YALMIP` you can use the `diagnostic` output of the `optimize` command.

(b) Write a command

```
[u, exitflag] =  timeOptimalCtrl(A,B,ulim,x0, xtgt, Tguess)
```

that finds the minimum time required to perform the state transfer from $x(0) = x_0$ to $x(T_{\text{min}}) = x_{\text{tgt}}$.
Hint. It can be useful to proceed in two steps.
First, find an upper bound T_{upper} on the target time. Initialize $T_{\text{upper}} = T_{\text{guess}}$ and run `findCtrl` to see if it is possible to perform the state transfer in T_{upper} steps. If not, then double the value of T_{upper} and try again. If you want, you can add a maximum limit to how many times you perform the step, so as not to get stuck if it is impossible to execute the state transfer.
Once found a feasible target time T_{upper} (that allows `findCtrl` to compute a feasible control sequence u), find the smallest time by bisection. Initialize two variables $T_{\text{lo}} = 0$ and $T_{\text{hi}} = T_{\text{guess}}$. Then, in each iteration, find $T_{\text{mid}} = (T_{\text{lo}} + T_{\text{hi}})/2$ and execute `findCtrl` with target time T_{mid}. If a feasible solution exists, let $T_{\text{hi}} = T_{\text{mid}}$, otherwise let $T_{\text{lo}} = T_{\text{mid}}$. This

procedure is continued until $T_{\text{lo}} = T_{\text{hi}}$. The value of these variables are then the minimum time $T_{\min}$.

Exercise 5.17 Conceptual questions

(a) Why is the MPC optimization performed repeatedly at every sampling instant?

(b) Explain the "receding horizon" principle.

(c) What is meant by the terms the terms *prediction horizon* and *control horizon*?

Part II
Solutions

1 Discrete-time linear systems

Solution to Exercise 1.1

(a) The value of x at $t = 1$ is given by

$$x_1 = Ax_0 + Bu_0 = \begin{bmatrix} 0.75 & 0.1 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} 0$$
$$= \begin{bmatrix} 0.75 \cdot 1 + 0.1 \cdot 1 \\ 0 \cdot 1 + 0.5 \cdot 1 \end{bmatrix} = \begin{bmatrix} 0.85 \\ 0.5 \end{bmatrix}.$$

Similarly, the value of x at $t = 2$ is

$$x_2 = Ax_1 + Bu_1 = \begin{bmatrix} 0.75 & 0.1 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 0.85 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} 0$$
$$= \begin{bmatrix} 0.75 \cdot 0.85 + 0.1 \cdot 0.5 \\ 0 \cdot 0.85 + 0.5 \cdot 0.5 \end{bmatrix} = \begin{bmatrix} 0.6875 \\ 0.25 \end{bmatrix}.$$

The value of y at $t = 2$ is then

$$y_2 = Cx_2 = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 0.6875 \\ 0.25 \end{bmatrix} = \begin{bmatrix} 1 \cdot 0.6875 + 2 \cdot 0.25 \end{bmatrix} = 1.1875.$$

(b) By following the same procedure as in (a), we obtain

$$x_1 = Ax_0 + Bu_0 = \begin{bmatrix} 0.75 & 0.1 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} 5 = \begin{bmatrix} 5.85 \\ 10.5 \end{bmatrix}$$
$$x_2 = Ax_1 + Bu_1 = \begin{bmatrix} 0.75 & 0.1 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 5.85 \\ 10.5 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} 5 = \begin{bmatrix} 10.4375 \\ 15.25 \end{bmatrix}$$
$$y_2 = Cx_2 = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 10.4375 \\ 15.25 \end{bmatrix} = 40.9375.$$

(c) The matrix A is stable, since all the eigenvalues are less then 1 (A is diagonal, so eigenvalues are equal to elements on diagonal). Therefore, the state x converges to the steady state value x_{ss}, which has to satisfy the steady state equation

$$x_{ss} = Ax_{ss} + Bu \Rightarrow (I - A)x_{ss} = Bu \Rightarrow x_{ss} = (I - A)^{-1}Bu.$$

In (a), we have $u = 0$, thus $x_{ss} = 0$ and $y_{ss} = 0$.
In (b), we have $u = 5$. From

$$I - A = \begin{bmatrix} 0.25 & -0.1 \\ 0 & 0.5 \end{bmatrix} \Rightarrow (I - A)^{-1} = \begin{bmatrix} 4 & 0.8 \\ 0 & 2 \end{bmatrix}$$

it follows

$$x_{ss} = \begin{bmatrix} 4 & 0.8 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} 5 = \begin{bmatrix} 28 \\ 20 \end{bmatrix}.$$

Thus,

$$y_{ss} = Cx_{ss} = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 28 \\ 20 \end{bmatrix} = 68.$$

Solution to Exercise 1.2

In the case of the sample-and-hold-method with sampling time h, Matrices A and B of the discrete system can be computed using formulas

$$A = e^{A_c h} \qquad\qquad B = \int_0^h e^{A_c s} B_c ds.$$

(a) $A_c = 0, \quad B_c = 1$

$$A = e^{0h} = 1, \quad B = \int_0^h e^{0s} 1 ds = [s]_0^h = h$$

(b) $A_c = a, \quad B_c = aK$

$$A = e^{ah}, \quad B = \int_0^h e^{as} aK ds = aK \int_0^h e^{as} ds = aK \left[\frac{e^{as}}{a} \right]_0^h = K(e^{ah} - 1)$$

(c) The harmonic oscillator

$$\ddot{x}(t) = -\omega^2 x(t) + u(t)$$

admits a state-space description

$$\dot{x}(t) = \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) := A_c x(t) + B_c u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t).$$

We begin by computing

$$\begin{aligned} A = e^{A_c h} &= I + A_c h + A_c^2 \frac{h^2}{2} + \cdots = \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + h \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix} + \frac{h^2}{2!} \begin{bmatrix} -\omega^2 & 0 \\ 0 & -\omega \end{bmatrix} + \cdots = \\ &= \begin{bmatrix} 1 - \omega^2 h^2/2! + \omega^4 h^4/4! - \ldots & \omega h - \omega^3 h^3/3! + \ldots \\ -\omega h + \omega^3 h^3/3! & 1 - \omega^2 h^2/2! + \omega^4 h^4/4! + \ldots \end{bmatrix} = \\ &= \begin{bmatrix} \cos(\omega h) & \sin(\omega h) \\ -\sin(\omega h) & \cos(\omega h) \end{bmatrix} \end{aligned}$$

where the last step follows by identifying the Taylor expansions.
We can now proceed with

$$B = \int_{s=0}^h e^{As} B \, ds = \int_{s=0}^h \begin{bmatrix} \sin(\omega s) \\ \cos(\omega s) \end{bmatrix} ds = \frac{1}{\omega} \begin{bmatrix} 1 - \cos(\omega h) \\ \sin(\omega h) \end{bmatrix}$$

As usual, C does not change with sampling. The solution is thus complete.

(d) The discrete state matrix A can be computed as $A = e^{A_c h}$. In order to compute the matrix exponential we use the following formula that can be obtained using the Taylor theorem,

$$e^{Mh} = I + Mh + \frac{M^2h^2}{2!} + \frac{M^3h^3}{3!} + \dots.$$

We can now compute A as

$$A = e^{A_c h} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} h + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} h^2 = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix}.$$

The discrete input matrix B can be computed as

$$B = \int_0^h e^{A_c s} B ds = \int_0^h \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} ds = \int_0^h \begin{bmatrix} s \\ 1 \end{bmatrix} ds = \begin{bmatrix} \frac{h^2}{2} \\ h \end{bmatrix}$$

Note that, in constrast to the continuous double integrator, the first state is affected immediately by the input since the first element of B is non-zero.

Solution to Exercise 1.3

Let's call the state of the discrete system as $x_t = x(ht)$, where h is the discretization time.

(a) Explicit Euler:

$$\frac{x_{t+1} - x_t}{h} = ax_t + u_t \quad \longrightarrow \quad x_{t+1} = (1 + ha)x_t + hu_t.$$

The pole of the discrete system is located in $p = 1 + ha$. The system is stable if the pole is inside the unit circle, i.e., $|1 + ha| < 1$. Since $a < 0$ this is true if $h < -2/a$. That means that, to maintain the stability property of the system, h should be small enough.

(b) Implicit Euler:

$$\frac{x_t - x_{t-1}}{h} = ax_t + u_t \longrightarrow \quad x_t = \frac{1}{1 - ha} x_{t-1} + h/(1 - ha)u_t.$$

The pole of discrete system is located in $p = 1/(1 - ha)$. The system is stable if the pole is inside the unit circle, i.e., $|1/(1 - ha)| < 1$. Since $a < 0$ and $h > 0$ this is always satisfied.

(c) Zero-order hold discretization. The A and B matrix of the discrete system can be computed as:

$$A = e^{ah}, \quad B = \int_0^h e^{as} ds = \left[\frac{1}{a} e^{as} \right]_0^h = \frac{1}{a}(e^{ah} - 1)$$

The resulting discrete system is therefore described by

$$x_{t+1} = e^{ah} x_t + \frac{1}{a}(e^{ah} - 1)u_t$$

and it is stable if $|e^{ah}| < 1$, that is always true since $a < 0$ and $h > 0$.

Solution to Exercise 1.4

(a) From the transfer function, it follows

$$s^2Y(s) + 2\xi\omega_0 sY(s) + \omega_0^2 Y(s) = \omega_0^2 U(s).$$

By taking inverse Laplace transform of the both sides, we obtain

$$\ddot{y}(t) + 2\xi\omega_0\dot{y}(t) + \omega_0^2 y(t) = \omega_0^2 u(t).$$

By choosing the states

$$z_1(t) = y(t) \qquad z_2(t) = \dot{y}(t)$$

we get

$$\begin{aligned}
\dot{z}_1(t) &= \dot{y}(t) = z_2(t) \\
\dot{z}_2(t) &= \ddot{y}(t) = -2\xi\omega_0\dot{y}(t) - \omega_0^2 y(t) + \omega_0^2 u(t) = -\omega_0^2 z_1(t) - 2\xi\omega_0 z_2(t) + \omega_0^2 u(t) \\
y(t) &= z(t)
\end{aligned}$$

which correspond to the given state space realization.

(b) Matlab code is provided below

```
function [Ad,Bd,Cd]=FormMatrices(omega0,eps,h)
    % omega0 - the natural frequency
    % eps - damping
    % h - sampling time
    % Ad,Bd,Cd - matrices of discrete system

    % STEP 1. Form the matrices A,B,C of continuous system
    A=[0  1; -power(omega0,2)  -2*eps*omega0];
    B=[0 ; power(omega0,2)];
    C=[1 0];
    D=0;

    % STEP 2. Convert matrices A,B,C,D into the continuous system
    ContinuousSystem = ss(A,B,C,D);

    % STEP 3. Discretize the continuous system using zero order hold method
    DiscreteSystem=c2d(ContinuousSystem,h,'zoh');

    % STEP 4. Convert discrete system into matrices Ad,Bd,Cd
    Ad=DiscreteSystem.A;
    Bd=DiscreteSystem.B;
    Cd=DiscreteSystem.C;
end
```

(c) Matlab code is provided below

```
%% Part (i)
clear all; close all; clc;

omega0 = 1; % Natural frequency
eps = 0; % Damping
h = 0.001:0.1:pi; % Sampling time (varies in this part)
pole1 = zeros(1,length(h)); % Locations of pole 1
pole2 = zeros(1,length(h)); % Locations of pole 2

% For all sampling times
for i=1:1:length(h)
```

```
    % Find matrices of discrete system
    [Ad,Bd,Cd]=FormMatrices(omega0,eps,h(i));
    % Calculate poles
    poles=eig(Ad);
    % Store value of poles
    pole1(i)=poles(1);
    pole2(i)=poles(2);
end

% Plot pole locations
figure,
hold on;
ylabel('imag')
xlabel('real')
title('Pole locations')
scatter(real(pole1),imag(pole1),'full')
scatter(real(pole2),imag(pole2),'full')

%% Part (ii)
clear all; close all;

omega0 = 0.1:0.1:20*pi;  % Natural frequency (varies in this part)
eps = 0; % Damping
h = 0.1; % Sampling time
pole1 = zeros(1,length(omega0)); % Locations of pole 1
pole2 = zeros(1,length(omega0)); % Locations of pole 2

% For all natural frequencies
for i=1:1:length(omega0)
    % Find matrices of discrete system
    [Ad,Bd,Cd]=FormMatrices(omega0(i),eps,h);
    % Calculate poles
    poles=eig(Ad);
    % Store value of poles
    pole1(i)=poles(1);
    pole2(i)=poles(2);
end
% Plote pole locations
figure,
hold on;
ylabel('imag')
xlabel('real')
title(Pole locations')
scatter(real(pole1),imag(pole1),'full')
scatter(real(pole2),imag(pole2),'full')

%% Part (iii)
clear all; close all;

omega0 = 1;      % Natural frequency
eps = -1:0.01:1; % Damping (varies in this part)
h = 1; % Sampling time

pole1discrete = zeros(1,length(omega0)); % Locations of pole 1 - discrete system
pole2discrete = zeros(1,length(omega0)); % Locations of pole 2 - discrete system

pole1continuous = zeros(1,length(omega0)); % Locations of pole 1 - continuous system
pole1continuous = zeros(1,length(omega0)); % Locations of pole 2 - continuous system

% For all damping coeficients
for i=1:1:length(eps)
    % Find matrices of discrete system
    [Ad,Bd,Cd]=FormMatrices(omega0,eps(i),h);
    % Find poles of discrete system
    poles=eig(Ad);
    % Store poles of discrete system
    pole1discrete(i)=poles(1);
    pole2discrete(i)=poles(2);
    % Find matrix A for continuous system
    Ac = [0 1; -power(omega0,2) -2*eps(i)*omega0];
    % Find poles of continuous system
    poles=eig(Ac);
    % Store poles of continuous system
    pole1continious(i)=poles(1);
```

```
    pole2continious(i)=poles(2);
end

% Plote pole locations
figure,
hold on;
ylabel('imag')
xlabel('real')
title('Pole locations - Discrete system')
scatter(real(pole1discrete),imag(pole1discrete),'full')
scatter(real(pole2discrete),imag(pole2discrete),'full')

figure,
hold on;
ylabel('Imag')
xlabel('Real')
title('Pole locations - Continuous system')
scatter(real(pole1continious),imag(pole1continious),'full')
scatter(real(pole2continious),imag(pole2continious),'full')
```

Solution to Exercise 1.5

(a) We first analyze the stability for A_1. To determine if A_1 is stable, we find the characteristic polynomial of this matrix

$$\begin{aligned}\det(A_1 - \lambda I) &= \det \begin{bmatrix} 1-\lambda & 0.1 \\ 0 & 2-\lambda \end{bmatrix} \\ &= (1-\lambda)(2-\lambda) - 0 \cdot 0.1 = \lambda^2 - 3\lambda + 2.\end{aligned}$$

By solving the previous quadratic equation, we obtain the poles of the system to be $\lambda_1 = 1$ and $\lambda_2 = 2$. Since λ_1 and λ_2 are not inside the unit circle ($|\lambda_1| \geq 1$ and $|\lambda_2| \geq 1$), the system A_1 is unstable. By repeating the same procedure, we obtain $\lambda_{1,2} = -0.65 \pm 0.1323i$ for A_2, and $\lambda_1 = 0.5562$ and $\lambda_2 = 0.1438$ for A_3. These systems are stable, since eigenvalues are inside the unit circle ($|\lambda_1| < 1$ and $|\lambda_2| < 1$ for both the systems).

(b) Only the system A_1 is unstable, so we expect the step response of this system to diverge. Thus, A_1 corresponds to (C). Only the system A_2 has the poles with imaginary parts, so we expect its step response to oscillate before it converges to the steady state. Thus, A_2 corresponds to (A). Thus, the only possible choice for A_3 is (B). This is according to intuition, since both poles of A_3 are real, so we expect the smooth step response.

Solution to Exercise 1.6

(a) Since $w \neq 0$,

$$w^T[B \ AB \ ... \ A^{n-1}B] = [w^T B \ w^T \lambda B \ ... \ w^T \lambda^{n-1} B] = 0.$$

Therefore, the controllability matrix $[B \ AB \ ... \ A^{n-1}B]$ has a null space, so we conclude $\text{rank}([B \ AB \ ... \ A^{n-1}B]) < n$.

(b) $\tilde{A} = PAP^{-1}$, $\tilde{B} = PB$ We know that $\tilde{w}^T \tilde{A}_{22} = \lambda \tilde{w}^T$, so the first condition

of the PBH test is

$$\begin{aligned}
[0\ \tilde{w}^T]\tilde{A} &= [0\ \tilde{w}^T\tilde{A}_{22}] \\
[0\ \tilde{w}^T]\tilde{A} &= \lambda[0\ \tilde{w}^T] \\
[0\ \tilde{w}^T]PAP^{-1} &= \lambda[0\ \tilde{w}^T] \\
[0\ \tilde{w}^T]PA &= \lambda[0\ \tilde{w}^T]P \\
w^TA &= \lambda w^T,
\end{aligned}$$

where $w = P^T[0\ \tilde{w}^T]^T$. The second condition is

$$w^TB = [0\ \tilde{w}^T][B_1^T\ 0]^T = 0$$

(c) We have to prove two things:
i) (A, B) unreachable $\Rightarrow$ $(A - BL, B)$ unreachable;
ii) $(A - BL, B)$ unreachable $\Rightarrow$ (A, B) unreachable.
i) (A, B) is unreachable iif $\exists w \neq 0$ $w^TA = \lambda w^T$, $w^TB = 0$ Therefore, $w^T(A - BL) = w^TA - w^TBL = w^TA = \lambda w^T$. So $(A - BL, B)$ is unreachable.
ii) $(A - BL, B)$ is unreachable iif $\exists v \neq 0$ $v^T(A - BL) = \lambda v^T$, $v^TB = 0$ Therefore, $v^TA - v^TBL = \lambda v^T \Leftrightarrow v^TA = \lambda v^T$. So (A, B) is unreachable.

Solution to Exercise 1.7

(a) Let's compute the reachability matrix.

$$\mathcal{C} = \begin{bmatrix} d & ad + be \\ e & ce \end{bmatrix}. \tag{29}$$

The system is not reachable when the determinant of $\mathcal{C}$ is 0, i.e., when the relation $e(dc - ad - eb) = 0$ is satisfied.

(b) Let $w^T = \begin{bmatrix} w_1 & w_2 \end{bmatrix}$. According to PBH test, the system is unreachable if and only if: (1) $w \neq 0$; (2) $w^TA = \lambda w^T$; (3) $w^TB = 0$. From (3), we have

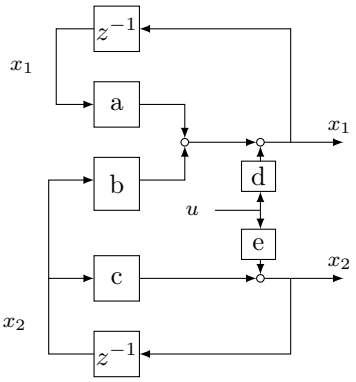
$$w^TB = \begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0 \Rightarrow w_2 = 0.$$

From (2), it follows

$$w^TA = \lambda w^T \Rightarrow \begin{bmatrix} w_1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & 3 \end{bmatrix} = \begin{bmatrix} w_1 & 2w_1 \end{bmatrix} = \begin{bmatrix} \lambda w_1 & 0 \end{bmatrix} \Rightarrow w_1 = 0.$$

Therefore, $w = 0$. Thus, the system is reachable based on PBH test.

(c) Let's now draw the block diagram of the system



(i) when $e = 0$, u cannot effect the second state x_2.
(ii) when $b = 0$ and $d = 0$, u cannot effect the first state x_1.
(iii) when $b = 0$ and $c = a$, the states x_1 and x_2 have the same dynamics. If the states start from the origin, their values are always equal. Thus, the input cannot drive apart the two states.

Solution to Exercise 1.8

(a) From the output equation:

$$\begin{aligned} x_2(1) &= y(1) = 0; \\ x_2(2) &= y(2) = 1; \end{aligned} \tag{30}$$

From the output equation:

$$\begin{aligned} &x_2(2) = 1 = x_1(1) + x_2(1) = x_1(1) + 0 \rightarrow x_1(1) = 1; \\ &x_1(2) = x_1(1) + u(1) = 1 + 1 = 2; \end{aligned} \tag{31}$$

We can now compute $x(3)$ as

$$x(3) = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} x(2) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(2) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

(b) Let $w = \begin{bmatrix} w_1 & w_2 \end{bmatrix}^T$. According to PBH test, the system is unobservable if and only if: (1) $w \neq 0$; (2) $Aw = \lambda w$; (3) $Cw = 0$. From (3), we have

$$Cw = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = w_2 \Rightarrow w_2 = 0.$$

From (2), it follows

$$Aw = \lambda w \Rightarrow \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ 0 \end{bmatrix} = \begin{bmatrix} w_1 \\ w_1 \end{bmatrix} = \begin{bmatrix} \lambda w_1 \\ 0 \end{bmatrix} \Rightarrow w_1 = 0.$$

Therefore, $w = 0$. Thus, the system is observable based on PBH test.

(c) The observability matrix of the system is

$$\mathcal{O} = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

Since rank $\mathcal{O} = 2$, the system is observable.

Solution to Exercise 1.9

(a) The state x at time k can be expressed as

$$x_t = A^t x_0 + \sum_{i=0}^{t-1} A^{t-i-1} B u_i,$$

and consequently the output y as

$$y_t = CA^t x_0 + C \sum_{i=0}^{t-1} A^{t-i-1} B u_i + D u_t. \tag{32}$$

If $D = 0, CB = 0, CAB = 0, \ldots, CA^{n-2}B = 0$, this relation can be rewritten as $y_t = CA^k x_0$. By rewriting the previous equation for $t = 0, \ldots, n-1$, we obtain the following system of equations:

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix} = \mathcal{O} x_0, \text{ where } \mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{33}$$

corresponds to the observability matrix. Since the system is observable, $\mathcal{O}$ is full-rank and therefore from the system of equations we can compute the initial state x_0.

(b) We now prove that the sufficient condition in (a) is also necessary if y_k is scalar. Let's rewrite equation (32) for $t = 0, \ldots, n-1$:

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix} = \mathcal{O} x_0 + \mathcal{M} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \end{bmatrix} \tag{34}$$

where

$$\mathcal{M} = \begin{bmatrix} D & 0 & \cdots & 0 \\ CB & D & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{n-2}B & CA^{n-3}B & \cdots & D \end{bmatrix}.$$

Since $y(k)$ is scalar, $\mathcal{O}$ and $\mathcal{M}$ are square matrices. Hence, the system contains n equation in $n + n$ unknown variables x_0, $u_0, \ldots, u_{n-1}$. To compute x_0, from the system we require that $M = 0$, that is,

$$D = 0, CB = 0, CAB = 0, \ldots, CA^{n-2}B = 0.$$

Solution to Exercise 1.10

(a) Let's substitute the feedback law in the system dynamics:

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t = Ax_t - BLy_t + Br_t \\ &= (A - BLC)x_t + Br_t, \\ y_t &= Cx_t. \end{aligned} \tag{35}$$

The closed loop system is described by the state-space description $(\tilde{A}, B, C, 0)$, where $\tilde{A} = A - BLC$.

(b) To solve this point we will use the PBH test for reachability. Since the system $(A, B, C, 0)$ is reachable then

$$\nexists w \neq 0 \text{ such that } w^{\mathrm{T}}A = \lambda w^{\mathrm{T}} \text{ and } w^{\mathrm{T}}B = 0$$

Therefore

$$\begin{aligned} &\nexists w \neq 0 \text{ such that } w^{\mathrm{T}}A - w^{\mathrm{T}}BLC = \lambda w^{\mathrm{T}} \text{ and } w^{\mathrm{T}}B = 0 \\ &\nexists w \neq 0 \text{ such that } w^{\mathrm{T}}(A - BLC) = \lambda w^{\mathrm{T}} \text{ and } w^{\mathrm{T}}B = 0 \\ &\nexists w \neq 0 \text{ such that } w^{\mathrm{T}}\tilde{A} = \lambda w^{\mathrm{T}} \text{ and } w^{\mathrm{T}}B = 0 \end{aligned}$$

Hence, the closed-loop system described by $(\tilde{A}, B, C, 0)$ is also reachable.

(c) The observability of the closed-loop system can be proved using the PBH test for observability, similarly to the solution of point (b).

Solution to Exercise 1.11

(a) Let $L = \begin{bmatrix} l_1 & l_2 \end{bmatrix}$ The closed-loop system matrix $A - BL$ is given by

$$A - BL = \begin{bmatrix} 0.5 & 1 \\ 0.8 - l_1 & 0.2 - l_2 \end{bmatrix}.$$

Thus, the characteristic polynomial

$$\begin{aligned} p(z) = \det zI - (A + BL) = \det \begin{bmatrix} z - 0.5 & -1 \\ -0.8 + l_1 & z - 0.2 + l_2 \end{bmatrix} = \\ = z^2 - (0.7 - l_2)z + l_1 - 0.5l_2 - 0.7 \end{aligned}$$

The desired closed-loop polynomial is

$$p_{\mathrm{des}}(z) = (z - 0.2)^2 = z^2 - 0.4z + 0.04$$

Identifying the coefficients, we get

$$\begin{aligned} -l_2 + 0.7 = 0.4 \qquad &\Rightarrow l_2 = 0.3 \\ +l_1 - 0.5l_2 - 0.7 = 0.04 \qquad &\Rightarrow l_1 = 0.89 \end{aligned}$$

Thus, $L = \begin{bmatrix} 0.89 & 0.3 \end{bmatrix}$.

(b) The characteristic polynomial is the same as in (a), and the desired closed-loop polynomial is this time

$$p_{\mathrm{des}}(z) = (z - 0.1)(z - 0.3) = z^2 - 0.4z + 0.03$$

Identifying the coefficients, we get

$$\begin{aligned} -l_2 + 0.7 = 0.4 \qquad & \Rightarrow l_2 = 0.3 \\ l_1 - 0.5l_2 - 0.7 = 0.03 \qquad & \Rightarrow l_1 = 0.88 \end{aligned}$$

The closed loop system is then

$$x_{t+1} = (A - BL)x_t + Bl_r r_t$$

where

$$(A - BL) = \begin{bmatrix} 0.5 & 1 \\ -0.08 & -0.1 \end{bmatrix}$$

Since the system is stable, it converges to a steady state x_{ss} when $r_t = 1$. The steady state x_{ss} can be obtained from the equation

$$x_{ss} = (A - BL)x_{ss} + Bl_r 1 \Rightarrow x_{ss} = (I - A + BL)^{-1} Bl_r$$

The steady state value for measurements y_{ss}

$$y_{ss} = Cx_{ss} = C(I - A + BL)^{-1} Bl_r$$

By matrix computations, it follows

$$\begin{aligned} C(I - A + BL)^{-1}B &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 & -1 \\ 0.08 & 1.1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1.17460 & 1.5873 \\ -0.1270 & 0.7937 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0.7937 \end{aligned}$$

Thus, to obtain $y_{ss} = 1$, we need to have

$$l_r = \frac{y_{ss}}{C(I - A + BL)^{-1}B} = \frac{1}{0.7937} = 1.2599.$$

Solution to Exercise 1.12

(a) The observability matrix of the system is

$$\mathcal{O} = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

Thus, since rank $\mathcal{O} = 1$, the system is not observable, so the poles of the system cannot be placed at arbitrary locations.

(b) Let $K = \begin{bmatrix} k_1 & k_2 \end{bmatrix}^T$. The error dynamics of the observer is governed by the matrix

$$A - KC = \begin{bmatrix} 0.2 & 1 - k_1 \\ 0 & 1 - k_2 \end{bmatrix}$$

The characteristic polynomial of the observer is

$$p(z) = \det\,(zI - (A - KC)) = (z - 0.2)(z - 1 + k_2).$$

Thus, we can place the poles of the observer at positions $\{0.2, 0.3\}$ by choosing $k_2 = 0.7$ and arbitrary k_1.

(c) Note from the previous problem that we can only choose one pole of the system, while the second one is always 0.2. Therefore, it is not possible to place of the observer at $\{0.4, 0.5\}$.

Solution to Exercise 1.13

(a) The discrete state matrix A can be computed as $A = e^{A_c h}$. In order to compute the matrix exponential we use the following formula that can be obtained using the Taylor theorem,

$$e^{Mh} = I + Mh + \frac{M^2h^2}{2!} + \frac{M^3h^3}{3!} + \dots.$$

We can now compute A as

$$A = e^{A_c h} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} h + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} h^2 = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix}.$$

The discrete input matrix B can be computed as

$$B = \int_0^h e^{A_c s} B ds = \int_0^h \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} ds = \int_0^h \begin{bmatrix} s \\ 1 \end{bmatrix} ds = \begin{bmatrix} \frac{h^2}{2} \\ h \end{bmatrix}$$

(b) The controllability matrix is given by

$$W_c = \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} 2 & 6 \\ 2 & 2 \end{bmatrix}$$

which has full rank. Hence, the system is reachable.
The closed-loop matrix is $A - BL$, which evaluates to

$$\begin{bmatrix} 1 - 2l_1 & 2 - 2l_2 \\ -2l_1 & 1 - 2l_2 \end{bmatrix}$$

Its characteristic polynomial is

$$\begin{aligned} \det(zI - (A - BL)) &= (z - 1 + 2l_1)(z - 1 + 2l_2) + 4l_1 - 4l_1 l_2 = \\ &= z^2 + z(-2 + 2l_1 + 2l_2) + (1 - 2l_1)(1 - 2l_2) + 4l_1 - 4l_1 l_2 = \\ &= z^2 + z(-2 + 2l_1 + 2l_2) + 1 - 2l_1 - 2l_2 + 4l_1 \end{aligned}$$

The desired characteristic polynomial is

$$p(z) = (z - \frac{1}{2})^2 = z^2 - z + \frac{1}{4}$$

which implies that

$$\begin{aligned} -2 + 2l_1 + 2l_2 &= -1 \\ 1 + 2l_1 - 2l_2 &= 1/4 \end{aligned}$$

Adding the two equations yields $4l_1 = 1/4$, *i.e.* $l_1 = 1/16$; The first equation then gives that $l_2 = 7/16$.

(c) The error dynamics is governed by the matrix $A - KC$

$$A - KC = \begin{bmatrix} 1 - k_1 & 2 \\ -k_2 & 1 \end{bmatrix}$$

whose characteristic polynomial is

$$\det(zI - (A - KC)) = (z - 1 + k_1)(z - 1) + 2k_2 = z^2 + z(k_1 - 2) + 1 - k_1 + 2k_2$$

The desired characteristic polynomial is

$$p(z) = \left(z - \frac{1}{4}\right)^2 = z^2 - \frac{1}{2}z + \frac{1}{16}$$

which implies that we must have

$$k_1 - 2 = -\frac{1}{2}$$
$$1 - k_1 + 2k_2 = \frac{1}{16}$$

In other words,

$$k_1 = \frac{3}{2}, \qquad k_2 = \frac{9}{32}$$

Solution to Exercise 1.14

(a) *Open loop* stability is not affected by the B matrix and hence does not depend on the parameter b. To determine if the system is stable, we compute the eigenvalues of A as the roots of its characteristic polynomial:

$$p(\lambda) = \det(\lambda I - A) = \det \begin{bmatrix} \lambda & 1/2 \\ -3/2 & \lambda + 2 \end{bmatrix} =$$
$$= \lambda^2 + 2\lambda + 3/4 = (\lambda + 1/2)(\lambda + 3/2)$$

Hence, A has eigenvalues $\lambda = -1/2$ and $\lambda = -3/2$. Since one of the eigenvalues have magnitude greater than one, the system is open-loop unstable. We thus conclude that the system is not open-loop stable for any values of the parameter b.

(b) The controllability matrix is

$$W_c = \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} 1 & -b/2 \\ b & 3/2 - 2b \end{bmatrix}$$

which looses rank when $\det W_c = 0$, *i.e.* when

$$\det W_c = b^2/2 - 2b + 3/2 = \frac{1}{2}(b - 3)(b - 1) = 0$$

Hence, the system is controllable, unless $b = 1$ or $b = 3$.

(c) With $b = 0$, the closed-loop dynamics is

$$x_{t+1} = A_c x_t$$

where

$$A_c = A - BL = \begin{bmatrix} 0 & -1/2 \\ 3/2 & -2 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} l_1 & l_2 \end{bmatrix} = \begin{bmatrix} -l_1 & -1/2 - l_2 \\ 3/2 & -2 \end{bmatrix}$$

The matrix A_c has characteristic polynomial

$$p(\lambda) = (\lambda + l_1)(\lambda + 2) + 3/2(1/2 + l_2) = \lambda^2 + (2 + l_1)\lambda + 2l_1 + 3/4 + 3/2l_2$$

To place both eigenvalues at $\lambda = 0$, we need $p(\lambda) = \lambda^2$, *i.e.*

$$\begin{aligned} l_1 &= -2 \\ l_2 &= 13/6 \end{aligned}$$

With this feedback, the closed-loop system matrix A_c will become nilpotent. In particular,

$$A_c^2 = 0$$

so the system state will converge to zero in no more that two steps.

Solution to Exercise 1.15

(a) The open-loop system is asymptotically stable if all the eigenvalues of the system matrix A has magnitude strictly less than one. Since

$$p(z) = \det zI - A = \det \begin{bmatrix} z - 0.4 & -0.8 \\ -0.4 & z \end{bmatrix} = z^2 - 0.4z - 0.32 = (z - 0.8)(z + 0.4)$$

The open loop system has poles in $z = 0.8$ and $z = -0.4$, and is thus stable.

(b) The controllability matrix

$$\begin{bmatrix} B \\ AB \end{bmatrix} = \begin{bmatrix} 0 & 0.8 \\ 1 & 0 \end{bmatrix}$$

has full rank. Hence, the system is reachable, and therefore also controllable.

(c) The closed-loop system matrix $A - BL$ has characteristic polynomial

$$\begin{aligned} p(z) = \det zI - (A - BL) = \det \begin{bmatrix} z - 0.4 & -0.8 \\ -0.4 + l_1 & z + l_2 \end{bmatrix} = \\ = z^2 + (l_2 - 0.4)z - 0.32 + 0.8l_1 - 0.4l_2 \end{aligned}$$

The desired closed-loop polynomial is

$$p_{\text{des}}(z) = (z - 0.5)^2 = z^2 - z + 0.25$$

Identifying the coefficients, we get

$$\begin{aligned} l_2 - 0.4 &= -1 && \Rightarrow l_2 = -0.6 \\ -0.32 - 0.4l_2 + 0.8l_1 &= 0.25 && \Rightarrow l_1 = 33/80 \end{aligned}$$

(d) The observability matrix

$$\begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0.8 & 0.8 \end{bmatrix}$$

does not have full rank (its rows are linearly dependent). Hence, the system is not observable.

(e) The error dynamics of the observer is governed by the matrix

$$A - KC = \begin{bmatrix} 0.4 - k_1 & 0.8 - k_1 \\ 0.4 - k_2 & -k_2 \end{bmatrix}$$

whose characteristic polynomial is

$$p(z) = \det zI - (A - KC) = z^2 + (k_1 + k_2 - 0.4)z - 0.4(k_1 + k_2 - 0.8).$$

With desired poles in $z = 0$ and $z = -0.4$ we have

$$p_{\text{des}}(z) = z(z + 0.4) = z^2 + 0.4z$$

which is satisfied for any gains for which $k_1 + k_2 = 0.8$.

(f) There are no values for k_1 and k_2 which gives the desired characteristic polynomial is $p_{\text{des}}(z) = z^2$. This is to be expected. Since the system is not observable, we are not guaranteed to be able to place the poles in arbitrary locations.

Solution to Exercise 1.16

(a) We introduce the following notation for the continuous-time system

$$\dot{x}(t) = A_c x(t) + B_{1c} F_1(t) + B_{2c} F_2(t)$$

and

$$x_{t+1} = Ax_t + Bu_t + B_w w_t$$

where $x_t = x(kh)$, $u_t = F_1(kh)$ and $w_t = F_2(kh)$ for $k = 0, 1, \ldots$.
The formulas for zero-order hold sampling yield

$$A = e^{A_c h} = \begin{bmatrix} e^{-\frac{h}{2\theta}} & 0 \\ 0 & e^{-\frac{h}{\theta}} \end{bmatrix}$$

while

$$B = \int_{s=0}^{h} e^{A_c s} B_{1c}\, ds = \int_{s=0}^{h} \begin{bmatrix} e^{-\frac{s}{2\theta}} \\ \frac{c_1 - c_0}{V_0} e^{-\frac{s}{\theta}} \end{bmatrix} ds = \begin{bmatrix} 2\theta(1 - e^{-\frac{h}{2\theta}}) \\ \theta \frac{c_1 - c_0}{V_0}(1 - e^{-\frac{h}{\theta}}) \end{bmatrix}$$

and

$$B_w = \int_{s=0}^{h} e^{A_c s} B_{2c}\, ds = \int_{s=0}^{h} \begin{bmatrix} e^{-\frac{s}{2\theta}} \\ \frac{c_2 - c_0}{V_0} e^{-\frac{s}{\theta}} \end{bmatrix} ds = \begin{bmatrix} 2\theta(1 - e^{-\frac{h}{2\theta}}) \\ \theta \frac{c_2 - c_0}{V_0}(1 - e^{-\frac{h}{\theta}}) \end{bmatrix}$$

(b) To check controllability, we form the controllability matrix

$$\mathcal{C}_2 = \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} 5 & 4.75 \\ -1 & -0.9 \end{bmatrix}$$

Since the matrix is square, we can show that it has full rank by verifying that its determinant is non-zero. In this case

$$\det \mathcal{C}_2 = 0.25$$

so the matrix has full rank and the system is controllable.

(c) The closed loop system matrix is

$$\begin{aligned} A - BL &= \begin{bmatrix} 0.95 & 0 \\ 0 & 0.90 \end{bmatrix} - \begin{bmatrix} 5 \\ -1 \end{bmatrix} \begin{bmatrix} l_1 & l_2 \end{bmatrix} = \\ &= \begin{bmatrix} 0.95 - 5l_1 & -5l_2 \\ l_1 & 0.9 + l_2 \end{bmatrix} \end{aligned}$$

and its eigenvalues are the roots of its characteristic polynomial

$$\begin{aligned} p(\lambda) &= \det\left(\lambda I - A + BL\right) = \\ &= \lambda^2 + (5l_1 - l_2 - 1.85)\lambda + 0.8550 - 4.5l_1 + 0.95l_2 \end{aligned}$$

The desired characteristic polynomial is

$$p_{\text{des}}(\lambda) = (\lambda + 0.5)(\lambda + 0.6) = \lambda^2 + 1.1\lambda + 0.3$$

The two coincide if we choose l_1 and l_2 such that

$$\begin{aligned} 5l_1 - l_2 - 1.85 &= 1.1 \\ 0.855 - 4.5l_1 + 0.95l_2 &= 0.3 \end{aligned}$$

This system of equations has solution

$$l_1 = 8.99 \qquad\qquad l_2 = 42$$

(d) With the added integrator state, the closed-loop dynamics is given by

$$\begin{aligned} x_{t+1} &= (A - BL)x_t + Bl_i i_t + B_w w \\ i_{t+1} &= -Cx_t + i_t \end{aligned}$$

At steady-state, $i_{t+1} = i_t$, which implies that

$$\lim_{t\to\infty} y_t = \lim_{t\to\infty} Cx_t = 0$$

2 Stability and invariance

2.1 Stability

Solution to Exercise 2.1

(a) It holds that

$$0 = \frac{0}{1+0^2}$$

(b)

$$\begin{aligned}\Delta V(x) &= (\frac{x}{1+x^2})^2 - x^2 \\ &= -\frac{2x^4+x^6}{(1+x^2)^2} \\ &< 0, \quad \forall x \neq 0\end{aligned}$$

Hence, the origin is globally asymptotically stable.

Solution to Exercise 2.2

(a) It holds that

$$0 = \frac{1}{2} \cdot 0 + 0^2$$

(b)

$$\begin{aligned}\Delta V(x) &= (\frac{1}{2}x + x^2)^2 - x^2 \\ &= -x^2(\frac{3}{2}+x)(\frac{1}{2}-x) \\ &< 0, \quad \forall |x| \leq \frac{1}{2}, x \neq 0\end{aligned}$$

Hence, the origin is asymptotically stable, with region of attraction $|x| \leq \frac{1}{2}$.

Solution to Exercise 2.3

(a) It holds that

$$0 = \frac{2 \cdot 0}{1+0^2}$$

For any x_k it holds that $|x_{k+1}| > |x_k|$, so the origin is an unstable equilibrium point.

(b)

$$\begin{aligned}\Delta V(x) &= (\frac{2x}{1+x^2})^2 - x^2 \\ &= -\frac{x^2(x^2+3)(x^2-1)}{(x^2+1)^2} \\ &\leq 0, \quad \forall x \geq 1\end{aligned}$$

Hence, all trajectories will remain bounded.

Solution to Exercise 2.4

$$\begin{aligned}
\Delta V(x) &= (\frac{1}{2}\sin(x))^2 - x^2 \\
&= \frac{1}{4}|\sin(x)|^2 - x^2 \\
&\leq \frac{1}{4}|x|^2 - x^2 \quad (\forall x \neq 0) \\
&= -\frac{3}{4}x^2 \\
&< 0, \quad \forall x \neq 0
\end{aligned}$$

Hence, the origin is globally asymptotically stable.

Solution to Exercise 2.5

Consider the Lyapunov function $V(x) = x^2$ and note that

$$\begin{aligned}
\Delta V(x) &= g(x)^2 - x^2 \\
&= |g(x)|^2 - x^2 \\
&< |x|^2 - x^2 \quad (\forall x \neq 0) \\
&= 0
\end{aligned}$$

Hence, the origin is globally asymptotically stable.

Solution to Exercise 2.6

Assume that P is a positive definite solution to

$$A^T P A - P + Q = 0$$

for any positive definite Q. Introduce the Lyapunov function

$$V(x) = x^T P x$$

and note that

$$\begin{aligned}
\Delta V(x) &= V(x_{t+1}) - V(x_t) \\
&= x_{t+1}^T P x_{t+1} - x_t^T P x_t \\
&= x_t^T (A^T P A - P) x_t \\
&= -x_t^T Q x_t \\
&\leq 0
\end{aligned}$$

since Q is positive definite. Hence, the system is asymptotically stable. This proves sufficiency. Assume instead that the system is asymptotically stable, so that $|\lambda_i(A)| < 1$ for all i. It follows that

$$P = \sum_{k=1}^{\infty} (A^T)^k Q A^k$$

exists. It is easily verified that P is symmetric and positive definite since for every k it holds that

$$((A^T)^k Q A^k)^T = (A^T)^k Q^T A^k = (A^T)^k Q^T A^k$$

and

$$x^T (A^T)^k Q A^k x = y^T Q y \geq 0, \quad \forall x$$

Moreover, it holds that

$$\begin{aligned} A^T P A - P + Q &= \sum_{k=1}^{\infty} (A^T)^{k+1} Q A^{k+1} - (A^T)^k Q A^k + Q \\ &= Q - Q + \sum_{k=1}^{\infty} (A^T)^k Q A^k - (A^T)^k Q A^k \\ &= 0 \end{aligned}$$

so that P satisfies the Lyapunov equation. If any other such P' exists then

$$A^T (P - P') A - (P - P') = 0 \Rightarrow P - P' = \lim_{k \to \infty} (A^T)^k (P - P') A^k = 0$$

since A is a Schur matrix. Hence, if the system is asymptotically stable, then P as defined above is the uniqe positive definite solution to the Lyapunov equation. This proves necessity.

□

Solution to Exercise 2.7

Consider $V(x) = x^T P x$ and note that

$$\begin{aligned} \Delta V(x) &= V(x_{t+1}) - V(x_t) \\ &= x_{t+1}^T P x_{t+1} - x_t^T P x_t \\ &= x_t^T (A^T P A - P) x_t \\ &= -x_t^T Q x_t \\ &\leq 0 \end{aligned}$$

since Q is positive definite. Now, since $P \not\succ 0$ there exists some x_0 such that $V(x_0) = x_0^T P x_0 < 0$. It holds that

$$V(x_k) \leq V(x_0), \quad \forall k \geq 1$$

Moreover, $V(x_0) < 0$, and it can be concluded that

$$V(x_k) < 0, \quad \forall k \geq 1 \tag{36}$$

If the system would have been asymptotically stable it would hold that $x_k \to 0$ as $k \to \infty$, which implies that $V(x_k) \to 0$ as $k \to \infty$. This contradicts (36), so the system must be unstable.

□

Solution to Exercise 2.8

(a) The characteristic polynomial is given by

$$\begin{bmatrix} 1/4-\lambda & 1 \\ 1/2 & -\lambda \end{bmatrix} = \lambda^2 - \frac{\lambda}{4} - \frac{1}{2}$$

so that

$$\lambda_1 = \frac{1}{8}(1-\sqrt{33}) \approx -0.59$$

and

$$\lambda_2 = \frac{1}{8}(1+\sqrt{33}) \approx 0.84$$

Since $|\lambda_1| < 1$ and $|\lambda_2| < 1$, the system is asymptotically stable.

(b)

$$\begin{pmatrix} 1/4 & 1 \\ 1/2 & 0 \end{pmatrix}^T \begin{pmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{pmatrix} \begin{pmatrix} 1/4 & 1 \\ 1/2 & 0 \end{pmatrix} - \begin{pmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

so that

$$\begin{pmatrix} -\frac{15p_{11}}{16} + \frac{p_{12}}{4} + \frac{p_{22}}{4} + 1 & \frac{p_{11}}{4} - \frac{p_{12}}{2} \\ \frac{p_{11}}{4} - \frac{p_{12}}{2} & p_{11} - p_{22} + 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

In other words,

$$\begin{aligned} -\frac{15p_{11}}{16} + \frac{p_{12}}{4} + \frac{p_{22}}{4} + 1 &= 0 \\ \frac{p_{11}}{4} - \frac{p_{12}}{2} &= 0 \\ p_{11} - p_{22} + 1 &= 0 \end{aligned}$$

with unique solution

$$\begin{aligned} p_{11} &= 2.2 \\ p_{12} &= 1.1 \\ p_{22} &= 3.2 \end{aligned}$$

so that

$$P = \begin{pmatrix} 2.2 & 1.1 \\ 1.1 & 3.2 \end{pmatrix}$$

is a positive definite solution to the Lyapunov equation. Therefore, the system is asymptotically stable.

(c) The characteristic polynomial is given by

$$\begin{bmatrix} 1-\lambda & -1 \\ -1 & 1-\lambda \end{bmatrix} = \lambda^2 - 2\lambda$$

so that

$$\lambda_1 = 0$$

and

$$\lambda_2 = 2$$

Since $|\lambda_2| > 1$ the system is unstable.

$$\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}^T \begin{pmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{pmatrix} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} - \begin{pmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

so that

$$\begin{pmatrix} -2p_{12} + p_{22} + 1 & -p_{11} + p_{12} - p_{22} \\ -p_{11} + p_{12} - p_{22} & p_{11} - 2p_{12} + 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

In other words,

$$\begin{aligned} -2p_{12} + p_{22} + 1 &= 0 \\ -p_{11} + p_{12} - p_{22} &= 0 \\ p_{11} - 2p_{12} + 1 &= 0 \end{aligned}$$

with unique solution

$$\begin{aligned} p_{11} &= \frac{1}{3} \\ p_{12} &= \frac{2}{3} \\ p_{22} &= \frac{1}{3} \end{aligned}$$

so that

$$P = \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & \frac{1}{3} \end{pmatrix}$$

which is an indefinite matrix. Thus, the system is unstable.

2.2 Invariant sets

Solution to Exercise 2.9

It holds that

$$\mathrm{Pre}\,(\mathcal{X}) = \{x \in \mathbb{R}^2 \mid HAx \leq h\}$$

if $\mathcal{X}$ is represented as

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid Hx \leq h\}$$

(a)

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid \begin{pmatrix} 2 & 3 \end{pmatrix} x \leq 5\}$$

Hence,

$$\mathrm{Pre}\,(\mathcal{X}) = \{x \in \mathbb{R}^2 \mid \begin{pmatrix} 4 & -1.5 \end{pmatrix} x \leq 5\}$$

(b)

$$\mathcal{X} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} I \\ -I \end{pmatrix} x \leq \begin{pmatrix} 2 \\ 3 \\ 1 \\ 2 \end{pmatrix} \right\}$$

Hence,

$$\text{Pre}\,(\mathcal{X}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} 0.5 & 0 \\ 1 & -0.5 \\ -0.5 & 0 \\ -1 & 0.5 \end{pmatrix} x \leq \begin{pmatrix} 2 \\ 3 \\ 1 \\ 2 \end{pmatrix} \right\}$$

(c)

$$\mathcal{X} = \{ x \in \mathbb{R}^2 \mid -Ix \leq 0 \}$$

Hence,

$$\text{Pre}\,(\mathcal{X}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} -0.5 & 0 \\ -1 & 0.5 \end{pmatrix} x \leq 0 \right\}$$

Solution to Exercise 2.10

Proof. First, assume that $\mathcal{O} \not\subseteq \text{Pre}\,(\mathcal{O})$. It follows that $\exists \bar{x} \in \mathcal{O}$ such that $\bar{x} \notin \text{Pre}\,(\mathcal{O})$. Hence, it holds that $A\bar{x} \notin \mathcal{O}$ so that $\mathcal{O}$ is not positive invariant. This proves necessity. Next, assume that $\mathcal{O}$ is not positive invariant, so that $\exists \bar{x} \in \mathcal{O}$ such that $A\bar{x} \notin \mathcal{O}$. It follows that $\bar{x} \notin \text{Pre}\,(\mathcal{O})$. Hence, $\mathcal{O} \not\subseteq \text{Pre}\,(\mathcal{O})$. This proves sufficiency. ■

Solution to Exercise 2.11

(a)

$$\text{Pre}\,(\mathcal{X}) = \{ x \in \mathbb{R}^2 \mid \begin{pmatrix} 4 & -1.5 \end{pmatrix} x \leq 5 \}$$

It does not hold that $\mathcal{X} \subseteq \text{Pre}\,(\mathcal{X})$. For instance $x = \begin{pmatrix} 2 & 0 \end{pmatrix}$ is in $\mathcal{X}$ but not $\text{Pre}\,(\mathcal{X})$. Hence, $\mathcal{X}$ is not positive invariant under $x_{t+1} = Ax_t$.

(b)

$$\text{Pre}\,(\mathcal{X}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} 0.5 & 0 \\ 1 & -0.5 \\ -0.5 & 0 \\ -1 & 0.5 \end{pmatrix} x \leq \begin{pmatrix} 2 \\ 3 \\ 1 \\ 2 \end{pmatrix} \right\}$$

Again, it does not hold that $\mathcal{X} \subseteq \text{Pre}\,(\mathcal{X})$. For instance $x = \begin{pmatrix} -1 & 3 \end{pmatrix}$ is in $\mathcal{X}$ but not $\text{Pre}\,(\mathcal{X})$. Hence, $\mathcal{X}$ is not positive invariant under $x_{t+1} = Ax_t$.

(c)

$$\text{Pre}\,(\mathcal{X}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} -0.5 & 0 \\ -1 & 0.5 \end{pmatrix} x \leq 0 \right\}$$

Again, it does not hold that $\mathcal{X} \subseteq \text{Pre}\,(\mathcal{X})$. For instance $x = \begin{pmatrix} 1 & 3 \end{pmatrix}$ is in $\mathcal{X}$ but not $\text{Pre}\,(\mathcal{X})$. Hence, $\mathcal{X}$ is not positive invariant under $x_{t+1} = Ax_t$.

Solution to Exercise 2.12

(a)

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid \begin{pmatrix} 2 & 3 \end{pmatrix} x \leq 5\}$$

Hence,

$$\mathrm{Pre}\,(\mathcal{X}) = \{x \in \mathbb{R}^2 \mid \begin{pmatrix} 2.5 & 1.5 \end{pmatrix} x \leq 5\}$$

It does not hold that $\mathcal{X} \subseteq \mathrm{Pre}\,(\mathcal{X})$. For instance $x = \begin{pmatrix} 5 & -2 \end{pmatrix}$ is in $\mathcal{X}$ but not $\mathrm{Pre}\,(\mathcal{X})$. Hence, $\mathcal{X}$ is not positive invariant under $x_{t+1} = Ax_t$.

(b)

$$\mathcal{X} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} I \\ -I \end{pmatrix} x \leq \begin{pmatrix} 2 \\ 3 \\ 1 \\ 2 \end{pmatrix} \right\}$$

Hence,

$$\mathrm{Pre}\,(\mathcal{X}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} 0.5 & 0 \\ 0.5 & 0.5 \\ -0.5 & 0 \\ -0.5 & -0.5 \end{pmatrix} x \leq \begin{pmatrix} 2 \\ 3 \\ 1 \\ 2 \end{pmatrix} \right\}$$

For any $x \in \mathcal{X}$ it holds that

$$\begin{aligned} x_1 \leq 2 &\Rightarrow 0.5x_1 \leq 1 < 2 \\ x_1 \leq 2, x_2 \leq 3 &\Rightarrow 0.5x_1 + 0.5x_2 \leq 2.5 < 3 \\ -x_1 \leq 1 &\Rightarrow -0.5x_1 \leq 0.5 < 1 \\ -x_1 \leq 1, -x_2 \leq 2 &\Rightarrow -0.5x_1 - 0.5x_2 \leq 1.5 < 2 \end{aligned}$$

Hence, $x \in \mathrm{Pre}\,(\mathcal{X})$, so that $\mathcal{X} \subseteq \mathrm{Pre}\,(\mathcal{X})$. Thus, $\mathcal{X}$ is positive invariant under $x_{t+1} = Ax_t$.

(c)

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid -Ix \leq 0\}$$

Hence,

$$\mathrm{Pre}\,(\mathcal{X}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} -0.5 & 0 \\ -0.5 & -0.5 \end{pmatrix} x \leq 0 \right\}$$

For any $x \in \mathcal{X}$ it holds that

$$\begin{aligned} -x_1 \leq 0 &\Rightarrow -0.5x_1 \leq 0 \\ -x_1 \leq 0, -x_2 \leq 0 &\Rightarrow -0.5x_1 - 0.5x_2 \leq 0 \end{aligned}$$

Hence, $x \in \mathrm{Pre}\,(\mathcal{X})$, so that $\mathcal{X} \subseteq \mathrm{Pre}\,(\mathcal{X})$. Thus, $\mathcal{X}$ is positive invariant under $x_{t+1} = Ax_t$.

Solution to Exercise 2.13

It holds that

$$\mathrm{C}\,(\mathcal{X};\mathcal{U}) = \left\{ x \in \mathbb{R}^2, u \in \mathbb{R} \;\middle|\; \begin{pmatrix} HA & HB \\ 0 & H_u \end{pmatrix} \begin{pmatrix} x \\ u \end{pmatrix} \leq \begin{pmatrix} h \\ h_u \end{pmatrix} \right\}$$

if $\mathcal{X}$ is represented as

$$\mathcal{X} = \{x \in \mathbb{R}^2 \mid Hx \leq h\}$$

and $\mathcal{U}$ is represented as

$$\mathcal{U} = \{u \in \mathbb{R} \mid H_u u \leq h_u\}$$

Throughout, it holds that

$$H_u = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad h_u = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

(a)

$$\mathcal{X} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} I \\ -I \end{pmatrix} x \leq \begin{pmatrix} 5 \\ 5 \\ 5 \\ 5 \end{pmatrix} \right\}$$

Hence,

$$\mathrm{C}\left(\mathcal{X};\mathcal{U}\right) = \left\{ x \in \mathbb{R}^2, u \in \mathbb{R} \;\middle|\; \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ -1 & -1 & 0 \\ 0 & -1 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ u \end{pmatrix} \leq \begin{pmatrix} 5 \\ 5 \\ 5 \\ 5 \\ 1 \\ 1 \end{pmatrix} \right\}$$

Now, apply Fourier-Motzkin elimination:

$$\mathcal{U} = \{u \leq 5 - x_2, u \leq 1\}$$
$$\mathcal{L} = \{u \geq -5 - x_2, u \geq -1\}$$

Now, it holds that

$$\mathrm{proj}_x(\mathrm{C}\left(\mathcal{X};\mathcal{U}\right)) = \left\{x \in \mathbb{R}^2 \;\middle|\; x_1 + x_2 \leq 5, -x_1 - x_2 \leq 5, \{L_i(x) \leq U_j(x), L_i \in \mathcal{L}, U_j \in \mathcal{U}\}\right\}$$

so that

$$\mathrm{C}\left(\mathcal{X};\mathcal{U}\right) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} 1 & 1 \\ -1 & -1 \\ 0 & -1 \\ 0 & 1 \end{pmatrix} x \leq \begin{pmatrix} 5 \\ 5 \\ 6 \\ 6 \end{pmatrix} \right\}$$

(b)

$$\mathcal{X} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{pmatrix} -1 & 0 \\ -1 & -2.5 \\ 1 & 0 \\ 1 & 2.5 \end{pmatrix} x \leq \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \end{pmatrix} \right\}$$

Hence,

$$\mathrm{C}\left(\mathcal{X};\mathcal{U}\right) = \left\{ x \in \mathbb{R}^2, u \in \mathbb{R} \;\middle|\; \begin{pmatrix} -1 & -1 & 0 \\ -1 & -3.5 & -2.5 \\ 1 & 1 & 0 \\ 1 & 3.5 & 2.5 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ u \end{pmatrix} \leq \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 1 \\ 1 \end{pmatrix} \right\}$$

Now, apply Fourier-Motzkin elimination:

$$\mathcal{U} = \{u \leq 1.2 - 0.4x_1 - 1.4x_2, u \leq 1\}$$
$$\mathcal{L} = \{u \geq -1.2 - 0.4x_1 - 1.4x_2, u \geq -1\}$$

Now, it holds that

$$\text{proj}_x(\text{C}(\mathcal{X};\mathcal{U})) = \left\{x \in \mathbb{R}^2 \,\middle|\, -x_1 - x_2 \leq 3, x_1 + x_2 \leq 3, \{L_i(x) \leq U_j(x), L_i \in \mathcal{L}, U_j \in \mathcal{U}\}\right\}$$

so that

$$\text{C}(\mathcal{X};\mathcal{U}) = \left\{ x \in \mathbb{R}^2 \,\middle|\, \begin{pmatrix} -1 & -1 \\ 1 & 1 \\ -1 & -3.5 \\ 1 & 3.5 \end{pmatrix} x \leq \begin{pmatrix} 3 \\ 3 \\ 5.5 \\ 5.5 \end{pmatrix} \right\}$$

Solution to Exercise 2.14

Proof. First, assume that $\mathcal{C} \not\subseteq \text{C}(\mathcal{C};\mathcal{U})$. It follows that $\exists \bar{x} \in \mathcal{C}$ such that $\bar{x} \notin \text{C}(\mathcal{C};\mathcal{U})$. Therefore, there is no $u \in \mathcal{U}$ such that $A\bar{x} + Bu \in \mathcal{C}$. Hence, $\mathcal{C}$ is not controlled invariant. This proves necessity. Next, assume that $\mathcal{C}$ is not controlled invariant, so that there exists some state trajectory $\{x_t\}_{t=0}^{\infty}$ where for some t it holds that $x_t \in \mathcal{C}$ but $x_{t+1} \notin \mathcal{C}$. In other words, $Ax_t + Bu \notin \mathcal{C}$ for any $u \in \mathcal{U}$. It follows that $x_t \notin \text{C}(\mathcal{C};\mathcal{U})$. Hence, $\mathcal{C} \not\subseteq \text{C}(\mathcal{C};\mathcal{U})$. This proves sufficiency. ■

Solution to Exercise 2.15

(a)

$$\text{C}(\mathcal{X};\mathcal{U}) = \left\{ x \in \mathbb{R}^2 \,\middle|\, \begin{pmatrix} 1 & 1 \\ -1 & -1 \\ 0 & -1 \\ 0 & 1 \end{pmatrix} x \leq \begin{pmatrix} 5 \\ 5 \\ 6 \\ 6 \end{pmatrix} \right\}$$

It does not hold that $\mathcal{X} \subseteq \text{C}(\mathcal{X};\mathcal{U})$. For instance $x = \begin{pmatrix} 5 & 5 \end{pmatrix}$ is in $\mathcal{X}$ but not $\text{C}(\mathcal{X};\mathcal{U})$. Hence, $\mathcal{X}$ is not controlled invariant under $x_{t+1} = Ax_t + Bu_t,\ u_t \in \mathcal{U}$.

(b)

$$\text{C}(\mathcal{X};\mathcal{U}) = \left\{ x \in \mathbb{R}^2 \,\middle|\, \begin{pmatrix} -1 & -1 \\ 1 & 1 \\ -1 & -3.5 \\ 1 & 3.5 \end{pmatrix} x \leq \begin{pmatrix} 3 \\ 3 \\ 5.5 \\ 5.5 \end{pmatrix} \right\}$$

For any $x \in \mathcal{X}$ it holds that

$$-x_1 \leq 3, -x_1 - 2.5x_2 \leq 3 \Rightarrow -x_1 - x_2 \leq 3$$
$$x_1 \leq 3, x_1 + 2.5x_2 \leq 3 \Rightarrow x_1 + x_2 \leq 3$$
$$-x_1 - 2.5x_2 \leq 3, x_1 \leq 3 \Rightarrow -x_1 - 3.5x_2 \leq 5.4 < 5.5$$
$$-x_1 \leq 3, x_1 + 2.5x_2 \leq 3 \Rightarrow x_1 + 3.5x_2 \leq 5.4 < 5.5$$

Hence, $x \in \mathrm{C}(\mathcal{X};\mathcal{U})$, so that $\mathcal{X} \subseteq \mathrm{C}(\mathcal{X};\mathcal{U})$. Thus, $\mathcal{X}$ is controlled invariant under $x_{t+1} = Ax_t + Bu_t,\ u_t \in \mathcal{U}$.

Solution to Exercise 2.16

Proof. Assume that no such matrix F exists. It follows that

$$\exists \bar{x} \in \mathcal{C} : A\bar{x} + B(F\bar{x}) \notin \mathcal{C}$$

for any F. Consequently, there can not exist any u such that $A\bar{x} + Bu \in \mathcal{C}$ as one could otherwise define F such that

$$F\bar{x} = u$$

and get a contradiction. Hence, $\mathcal{C}$ is not controlled invariant. This proves necessity. Next, assume that there exists an $m \times n$ matrix F such that $\mathcal{C}$ is positive invariant under $x_{t+1} = (A + BF)x_t$. It follows that for any $x \in \mathcal{C}$ it holds that

$$Ax + BFx \in \mathcal{C}$$

Hence, by observing that $u := Fx \in \mathcal{U}$ it follows that $x \in \mathrm{C}(\mathcal{C};\mathcal{U})$ under $x_{t+1} = Ax_t + Bu_t$, so that $\mathcal{C} \subseteq \mathrm{C}(\mathcal{C};\mathcal{U})$. By the result shown in Exercise 2.14, $\mathcal{C}$ is controlled invariant under $x_{t+1} = Ax_t + Bu_t$. This proves sufficiency. ■

Solution to Exercise 2.17

Proof. Assume $\mathcal{C}$ is controlled invariant. Take any $x \in \mathcal{C}$ and consider $y = Ax + Bu$. As $\mathcal{C}$ is controlled invariant there exists some $u \in \mathbb{R}^m$ so that $y \in \mathcal{C}$. It follows that

$$Ax = y - Bu$$

Now, $Ax \in \mathrm{Reach}(\mathcal{C})$ under $x_{t+1} = Ax_t$, $y \in \mathcal{C}$ and $Bu \in \mathrm{Im}\ B$. As x was chosen arbitrary from $\mathcal{C}$, it follows that $\mathrm{Reach}(\mathcal{C}) \subseteq \mathcal{C} + \mathrm{Im}\ B$. This proves necessity. Next, assume that $\mathrm{Reach}(\mathcal{C}) \subseteq \mathcal{C} + \mathrm{Im}\ B$ and let $c_1, \ldots, c_r$ be the vertex points for $\mathcal{C}$. It follows that

$$Ac_i = \tilde{c}_i + Bu_i, \quad i = 1, \ldots, r$$

for some $\tilde{c}_i \in \mathcal{C}$ and $u_i \in \mathbb{R}^m$. Now, introduce an $m \times n$ matrix F for which

$$Fc_i = -u_i, \quad i = 1, \ldots, r$$

(Note, that F is not unique if $r < n$.) It follows that

$$\begin{aligned} Ac_i &= \tilde{c}_i - BFc_i, \quad i = 1, \ldots, r \\ \Leftrightarrow (A + BF)c_i &= \tilde{c}_i \in \mathcal{C}, \quad i = 1, \ldots, r \end{aligned}$$

Now, for some $\bar{c} \in \mathcal{C}$ it holds that

$$\begin{aligned}(A+BF)\bar{c} &= (A+BF)\sum_{i=1}^{r}\alpha_i c_i \quad \left(\sum_{i=1}^{r}\alpha_i = 1\right) \\ &= \sum_{i=1}^{r}\alpha_i(A+BF)c_i \in \mathcal{C}\end{aligned}$$

since $(A+BF)c_i \in \mathcal{C},\ i = 1,\ldots,r$ and polytopes are closed under convex combinations. Since $\bar{c}$ was chosen arbitrarily from $\mathcal{C}$, it follows that $\mathcal{C} \subseteq \text{Pre}\,(C)$ under $x_{t+1} = (A+BF)x_t$. By the result shown in Exercise 2.16, $\mathcal{C}$ is controlled invariant. This proves sufficiency. ■

Solution to Exercise 2.18

(a) The optimal control problem is given by

$$\begin{gathered}(A-BL)^T P(A-BL) - P + (Q_1 + L^T Q_2 L) = 0 \\ \begin{pmatrix}0.8 & 0\\ -0.1 & 0.6\end{pmatrix}\begin{pmatrix}p_{11} & p_{12}\\ p_{12} & p_{22}\end{pmatrix}\begin{pmatrix}0.8 & -0.1\\ 0 & 0.6\end{pmatrix} - \begin{pmatrix}p_{11} & p_{12}\\ p_{12} & p_{22}\end{pmatrix} + \begin{pmatrix}1.8 & 1.18\\ 1.18 & 2.05\end{pmatrix} = 0 \\ \begin{pmatrix}-0.36p_{11} & -0.08p_{11} - 0.52p_{12}\\ -0.08p_{11} - 0.52p_{12} & 0.01p_{11} - 0.12p_{12} - 0.64p_{22}\end{pmatrix} = \begin{pmatrix}-1.8 & -1.18\\ -1.18 & -2.05\end{pmatrix}\end{gathered}$$

In other words,

$$\begin{aligned}-0.36p_{11} &= -1.8 && & p_{11} &= 5\\ -0.08p_{11} - 0.52p_{12} &= -1.18 && \Rightarrow & p_{12} &= 1.5\\ 0.01p_{11} - 0.12p_{12} - 0.64p_{22} &= -2.05 && & p_{22} &= 3\end{aligned}$$

The infinite-horizon cost is approximately given by $x_0^T P x_0$ where

$$P = \begin{pmatrix}5 & 1.5\\ 1.5 & 3\end{pmatrix}$$

Since P is positive definite, the feedback law is stabilizing.

(b) *Proof.* Assume $\text{Reach}(\mathcal{O}) \not\subseteq \mathcal{O}$, so that there is some $\bar{y} \in \text{Reach}(\mathcal{O})$ and $\bar{y} \notin \mathcal{O}$. As $\bar{y} \in \text{Reach}(\mathcal{O})$, it holds that $\exists \bar{x} \in \mathcal{O}$ such that $\bar{y} = A\bar{x}$. Hence, there is some $\bar{x} \in \mathcal{O}$ such that $A\bar{x} \notin \mathcal{O}$, implying that $\mathcal{O}$ is not positive invariant. This proves necessity. Next, assume that $\mathcal{O}$ is not positive invariant, so that for some $\bar{x}_0 \in \mathcal{O}$ there is some t such that $\bar{x}_t \notin \mathcal{O}$. It follows that $\bar{x}_{t-1} \in \mathcal{O}$ and $A\bar{x}_{t-1} \notin \mathcal{O}$. However, it then also holds that $\bar{y} = A\bar{x}_{t-1} \in \text{Reach}(\mathcal{O})$ since $\bar{x}_{t-1} \in \mathcal{O}$, and also $\bar{y} \notin \mathcal{O}$. Therefore, there is some $\bar{y}$ in $\text{Reach}(\mathcal{O})$ but not in $\mathcal{O}$, implying that $\text{Reach}(\mathcal{O}) \not\subseteq \mathcal{O}$. This proves sufficiency. ■

(c) Applying

$$A - BL = \begin{pmatrix}0.8 & -0.1\\ 0 & 0.6\end{pmatrix}$$

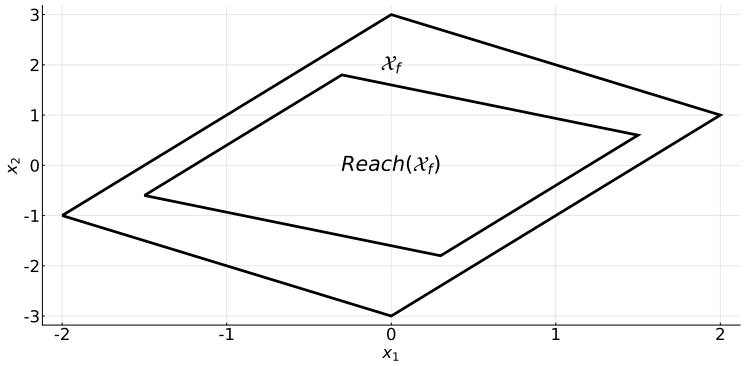
on the vertices of $\mathcal{X}_f$

$$v_1 = \begin{pmatrix} -2 \\ -1 \end{pmatrix}, \quad v_2 = \begin{pmatrix} 0 \\ 3 \end{pmatrix}, \quad v_3 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \quad v_4 = \begin{pmatrix} 0 \\ -3 \end{pmatrix}$$

yields

$$\tilde{v}_1 = \begin{pmatrix} -1.5 \\ -0.6 \end{pmatrix}, \quad \tilde{v}_2 = \begin{pmatrix} -0.3 \\ 1.8 \end{pmatrix}, \quad \tilde{v}_3 = \begin{pmatrix} 1.5 \\ 0.6 \end{pmatrix}, \quad \tilde{v}_4 = \begin{pmatrix} 0.3 \\ -1.8 \end{pmatrix}$$

These vertices span a polytope contained completely in $\mathcal{X}_f$, as shown in the following figure so that $\text{Reach}(\mathcal{X}_f) \subseteq \mathcal{X}_f$. It follows from the result



shown in (b) that $\mathcal{X}_f$ is positive invariant under $x_{t+1} = (A - BL)x_t$, so that $\mathcal{X}_f$ is control invariant under $x_{t+1} = Ax_t + Bu_t$. The feedback law

$$u_t = -Lx_t = -\begin{pmatrix} 1 & 1 \end{pmatrix} x_t$$

is admissible if

$$x_t \in \left\{ x \in \mathbb{R}^2 \mid -3 \leq -Lx \leq 3 \right\} = \left\{ x \in \mathbb{R}^2 \mid \begin{matrix} x_1 + x_2 \leq 3 \\ x_1 + x_2 \geq -3 \end{matrix} \right\} \subseteq \mathcal{X}_f$$

In other words, the given feedback law is admissible in $\mathcal{X}_f$

(d) If the finite-horizon problem has a feasible solution, then the system state can be driven to $x_N \in \mathcal{X}_f$ in N steps using admissible control inputs. It was shown in (c) that the given feedback law is admissible in $\mathcal{X}_f$. Moreoever, $\mathcal{X}_f$ was also shown to be invariant under the given feedback law. Hence, the system will eventually be stabilized under the given feedback law from any $x_N \in \mathcal{X}_f$. It holds that

$$\begin{aligned} x_N^T P x_N = \quad & \min_{\{u_k\}_{k=N}^{\infty}} \quad \sum_{k=N}^{\infty} x_k^T (Q_1 + L^T Q_2 L) x_k \\ & \text{such that} \quad x_{k+1} = (A - BL)x_k, \quad k = N, \ldots, \infty \end{aligned}$$

Hence, optimal solutions to the finite-horizon problem will be equivalent

to the cost of

$$\begin{aligned}
\underset{\{u_k\}_{k=0}^{\infty}}{\text{minimize}} \quad & \sum_{k=0}^{\infty} x_k^T Q_1 x_k + u_k^T Q_2 u_k \\
\text{such that} \quad & x_{k+1} = Ax_k + Bu_k, \quad k = 0, \ldots, N-1 \\
& x_{k+1} = (A - BL)x_k, \quad k = N, \ldots, \infty \\
& x_k \in \mathcal{X}, \quad k = 0, \ldots, \infty \\
& u_k \in \mathcal{U}, \quad k = 0, \ldots, \infty \\
& x_0 = x
\end{aligned}$$

This value coincides with the value of the initial infinite-horizon formulation only if L is the infinite-horizon LQR solution.

(e) $x_0 = \begin{pmatrix} -4 & -4 \end{pmatrix}^T \notin \mathcal{K}_\infty(\mathcal{X}_f; \mathcal{U})$. Hence, there is no admissible input sequence that can steer the system from x_0 into $\mathcal{K}_\infty(\mathcal{X}_f; \mathcal{U})$ and thereby to $\mathcal{X}_f$. Therefore, the finite-horizon problem is not feasible for any $N > 0$. $x_0 = \begin{pmatrix} -4 & 2 \end{pmatrix}^T \in \mathcal{K}_\infty(\mathcal{X}_f; \mathcal{U})$. Hence, there exists an admissible input sequence that steers the system from x_0 into $\mathcal{X}_f$. However, it is not certain how many steps is required. Therefore, there exist $N > 0$ for which the finite-horizon problem is feasible, but it is most probably not feasible for all $N > 0$.

3 Finite-time optimal control

3.1 Convex optimization

Solution to Exercise 3.1

(a) Let $x, y \in \mathbb{R}^n$. For θ with $0 \leq \theta \leq 1$, it holds that

$$\begin{aligned}
f(\theta x + (1-\theta)y) &= \|\theta x + (1-\theta)y\| \\
&\leq \|\theta x\| + \|(1-\theta)y\| \\
&\leq \theta\|x\| + (1-\theta)\|y\|
\end{aligned}$$

(b) Let $x, y \in \operatorname{dom} f$. For θ with $0 \leq \theta \leq 1$, it holds that

$$\begin{aligned}
f(\theta x + (1-\theta)y) &= \max\{f_1(\theta x + (1-\theta)y), f_2(\theta x + (1-\theta)y)\} \\
&\leq \max\{\theta f_1(x) + (1-\theta)f_1(y), \theta f_2(x) + (1-\theta)f_2(y)\} \\
&\leq \theta \max\{f_1(x), f_2(x)\} + (1-\theta)\max\{f_1(y), f_2(y)\} \\
&= \theta f(x) + (1-\theta)f(y)
\end{aligned}$$

Thus, $f(x) = \max\{f_1(x), f_2(x)\}$ is convex.

Solution to Exercise 3.2

(a)

$$\begin{aligned} x_1 + x_2 &\geq -1 \\ x_1 + x_2 &\leq 1 \\ x_1 - x_2 &\leq 1 \\ x_1 - x_2 &\geq -1 \end{aligned}$$

(b) Note that the quadratic constraint can be scaled by any scalar $\alpha > 0$ and still define the same solution set. We thus consider the set

$$\left\{x \,|\, \alpha \left(x^T P x + 2q^T x + r\right) \leq 0\right\}$$

and write (8) as

$$\left\{x \,|\, x^T Q^{-1} x - 2x_c^T Q^{-1} x + x_c^T Q^{-1} x_c - 1 \leq 0\right\}$$

For the two sets to be equal, we need

$$\alpha P = Q^{-1}, \quad \alpha q = -Q^{-1} x_c, \quad \alpha r = x_c^T Q^{-1} x_c - 1$$

which leads to

$$Q^{-1} = \alpha P, \quad x_c = -\alpha Q q = -P^{-1} q, \quad \alpha r = \alpha q^T P^{-1} q - 1$$

which allow us to conclude

$$Q^{-1} = \frac{1}{q^T P^{-1} q - r} P, \qquad x_c = -P^{-1} q$$

(c) We have

$$\begin{aligned} \mathcal{E} &= \{x = Mu + m \,|\, \|u\|_2 \leq 1\} \\ &= \left\{x \,|\, (x-m)^T M^{-T} M^{-1} (x-m) \leq 1\right\} \end{aligned}$$

From which we deduce that

$$M = Q^{1/2}, \quad m = x_c,$$

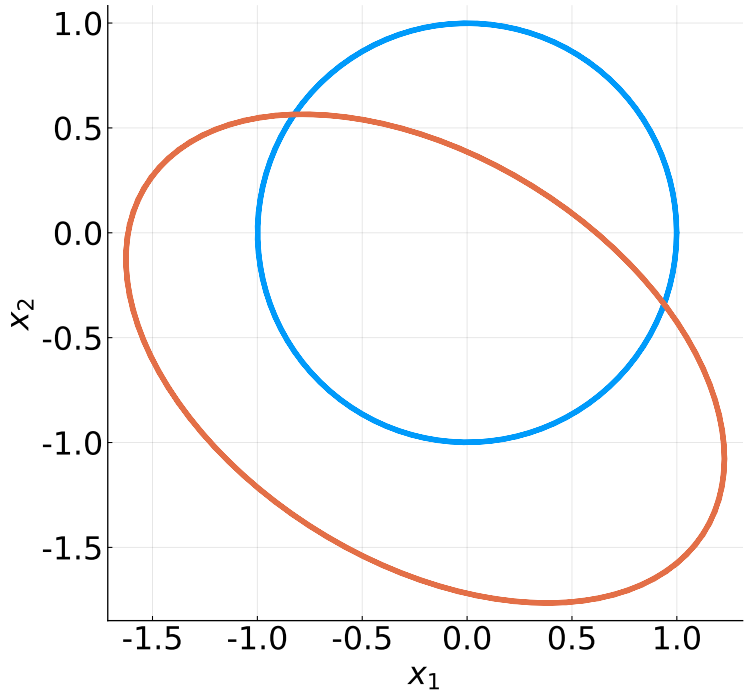
(d) Using the formulas from (b) yields

$$Q = \begin{pmatrix} 2.04 & -0.68 \\ -0.68 & 1.36 \end{pmatrix}, \quad x_c = \begin{pmatrix} -0.2 \\ -0.6 \end{pmatrix}$$

so that

$$M \approx \begin{pmatrix} 1.40 & -0.27 \\ -0.27 & 1.13 \end{pmatrix}, \quad m = \begin{pmatrix} -0.2 \\ -0.6 \end{pmatrix}$$

The ellopsoid is shown below along with the unit ball from which it is transformed.



Solution to Exercise 3.3

(a) First note that $\max f(x) = -\min(-f(x))$. This fact follows from the the following observations. If x maximizes f, then $f(y) \leq f(x)$ for all y. Equivalently, $-f(x) \leq -f(y)$ for all y, *i.e.* x minimizes $-f$. So, the original problem can be solved through the related formulation

$$\begin{array}{ll} \text{minimize} & -c^T x \\ \text{subject to} & x^T x \leq 1 \end{array}$$

The two problem have the same minimizer. If the optimal value to the related formulation is $f^\star$, then the optimal value of the original formulation is $-f^\star$.

(b) Form the Lagrangian

$$L(x, \lambda) = -c^T x + \lambda(x^T x - 1)$$

The first-order condition $\nabla L(x^\star, \lambda) = 0$ gives

$$-c + 2\lambda x^\star = 0 \Rightarrow x^\star = \frac{1}{2\lambda} c$$

We can now form the dual function

$$g(\lambda) = L(x^\star, \lambda) = -\frac{1}{4\lambda} c^T c - \lambda$$

The dual function is maximized for $\lambda = \frac{1}{2}\|c\|$. Hence,

$$x^\star = \frac{1}{\|c\|} c$$

and the optimal value of the original problem equals $\|c\|$.

(c) The containment holds if

$$a^T x = a^T(Mu + m) = a^T Mu + a^T m \leq b \text{ for all } u \text{ with } \|u\| \leq 1$$

From (b), we know that

$$\max a^T Mu \text{ such that } \|u\| \leq 1$$

equals $\|Ma\|$. So the containment holds if

$$\|Ma\| + a^T m \leq b$$

Solution to Exercise 3.4

(a) Solutions using `linprog` are given below.

(i)

```
f = [-5 -7 -12 1]';
A = [2 3 2 1
     3 2 4 -1];
b = [38 55]';
lb = zeros(4,1);
ub = inf(4,1);
[x,fval] = linprog(f,A,b,[],[],lb,ub)

Optimal solution found.


x =

         0
         0
   15.5000
    7.0000


fval =

  -179
```

(ii)

```
f = [5 3 2 0]';
A = [4 5 2 1
     3 4 -1 1];
b = [20 30]';
lb = zeros(4,1);
ub = inf(4,1);
[x,fval] = linprog(f,A,b,[],[],lb,ub)

Optimal solution found.


x =

     0
     0
     0
     0


fval =

     0
```

(iii)

```
f = [2 3]';
A = [-2 4
      4 3];
b = [-2 19]';
Aeq = [3 2]
beq = 14
lb = zeros(2,1);
ub = inf(2,1);
[x,fval] = linprog(f,A,b,Aeq,beq,lb,ub)

Optimal solution found.


x =

    4.6667
         0


fval =

    9.3333
```

(b) Solutions using YALMIP are given below. The solutions differ slightly to exemplify the versatility of YALMIP syntax.

(i)

```
f = [-5 -7 -12 1]';
A = [2 3 2 1
     3 2 4 -1];
b = [38 55]';
lb = zeros(4,1);
% YALMIP
x = sdpvar(4,1);
cost = f'*x;
constraints = [A*x <= b, x >= lb];
options = sdpsettings('verbose',1,'solver','linprog');
diagnostic = optimize(constraints,cost,options);
optimalcost = double(cost)

optimalcost =

  -179

xopt = double(x)

  xopt =

         0
         0
   15.5000
    7.0000
```

(ii)

```
x = sdpvar(4,1);
cost = 5*x(1)+3*x(2)+2*x(3);
constraints = [4*x(1)+5*x(2)+2*x(3)+x(4) <= 20
               3*x(1)+4*x(2)-x(3)+x(4) <= 30
               x(1) >= 0
               x(2) >= 0
               x(3) >= 0
               x(4) >= 0];
options = sdpsettings('verbose',1,'solver','linprog');
diagnostic = optimize(constraints,cost,options);
optimalcost = double(cost)

optimalcost =

  0

xopt = double(x)

  xopt =
```

```
         0
         0
         0
         0
```

(iii)

```
x = sdpvar(2,1);
cost = 2*x(1)+3*x(2);
constraints = [[3 2]*x == 14
               [2 -4]*x >= 2
               [4 3]*x <= 19
               x(1) >= 0
               x(2) >= 0];
options = sdpsettings('verbose',1,'solver','linprog');
diagnostic = optimize(constraints,cost,options);
optimalcost = double(cost)

optimalcost =

  9.3333

xopt = double(x)

  xopt =

         4.6667
         0
```

Solution to Exercise 3.5

```
c = @(i,j) 1 + (i-j)^2
s = @(i) i^2
d = 10
% YALMIP
x = sdpvar(4,3);
cost = 0
constraints = []
for i = 1:4
  % Supply
  constraints = [constraints; sum(x(i,:)) <= s(i)]
  for j = 1:3
    % Cost of sending from i to j
    cost = cost + c(i,j)*x(i,j)
    % Non-zero flow
    constraints = [constraints; x(i,j) >= 0]
  end
end
for j = 1:3
  % Demand
  constraints = [constraints; sum(x(:,j)) >= d]
end
options = sdpsettings('verbose',1,'solver','linprog');
diagnostic = optimize(constraints,cost,options);
optimalcost = double(cost)

optimalcost =

92

xopt = double(x)

xopt =

 1     0     0
 4     0     0
 5     4     0
 0     6    10
```

Solution to Exercise 3.6

(a) Solutions using `quadprog` are given below. Note, that `quadprog` solves problems that are formulated as follows:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}x^T H x + f^T x \\ \text{subject to} \quad & Ax \leq b \\ & (A_{eq}x = b_{eq}) \\ & (lb \leq x \leq ub) \end{aligned}$$

and reformulations are sometimes necessary.

(i)
```
Q = eye(3)
A = [-1 -1 -1;
     -2 1 0];
b = [-3 -1]';
[x,fval] = quadprog(Q,[],A,b)

x =

    1.0000
    1.0000
    1.0000


fval =

    1.5000
```

(ii)
```
Q = [8 2
     2 2];
f = [2 3]';
A = [-1 1
      1 1
      1 0];
b = [0 4 3]';
[x,fval] = quadprog(Q,f,A,b)

x =

    0.1667
   -1.6667


fval =

   -2.3333
```

(iii)
```
Q = eye(4);
A = [eye(4)
     -eye(4)];
      1 0];
b = [5 2 1 1 -2 2 1 1]';
Aeq = [1 0 -1 0
       -0.5 1 0 -1];
beq = [0.5*3 0]';
[x,fval] = quadprog(Q,[],A,b,Aeq,beq)

x =

    2.0000
    0.5000
    0.5000
   -0.5000
```

```
fval =

    2.3750
```

Note, that this defines an optimal input sequence $\hat{u_0} = 0.5, \hat{u_1} = -0.5$ to a simple MPC problem of prediction horizon 1, for the discrete linear system $x_{t+1} = 0.5x_t + u_t$.

(b) Solutions using YALMIP are given below. The solutions differ slightly to exemplify the versatility of YALMIP syntax.

(i)

```
Q = eye(3)
A = [-1 -1 -1;
     -2 1 0];
b = [-3 -1]';
% YALMIP
x = sdpvar(3,1);
cost = 0.5*x'*Q*x;
constraints = [A*x <= b];
options = sdpsettings('verbose',1,'solver','quadprog');
diagnostic = optimize(constraints,cost,options);
optimalcost = double(cost)

optimalcost =

  1.5

xopt = double(x)

  xopt =

    1.0001
    0.9999
    1.0000
```

(ii)

```
x = sdpvar(2,1);
cost = 2*x(1)+3*x(2)+4*x(1)^2+2*x(1)*x(2)+x(2)^2
constraints = [x(1)-x(2) >= 0
               x(1)+x(2) <= 4
               x(1) <= 3];
options = sdpsettings('verbose',1,'solver','quadprog');
diagnostic = optimize(constraints,cost,options);
optimalcost = double(cost)

optimalcost =

  -2.3333

xopt = double(x)

  xopt =

        0.1667
       -1.6667
```

(iii)

```
xlb = [2 -2]';
xub = [5 2]';
ulb = [-1 -1]';
uub = [1 1]';
%YALMIP
x = sdpvar(3,1);
u = sdpvar(2,1)
cost = 0
constraints = []
for i = 2:3
  cost = cost + 0.5*x(i)^2+0.5*u(i-1)^2
  constraints = [constraints
                 x(i) == 0.5*x(i-1)+u(i-1)
                 xlb(i-1) <= x(i) <= xub(i-1)
                 ulb(i-1) <= u(i-1) <= uub(i-1)];
```

```
end
constraints = [constraints; x(1) == 3]
options = sdpsettings('verbose',1,'solver','quadprog');
diagnostic = optimize(constraints,cost,options);
optimalcost = double(cost)

optimalcost =

  2.3750

xopt = double(x)

  xopt =

        3.0000
        2.0000
        0.5000

uopt = double(u)

uopt =

    0.5000
   -0.5000
```

Note, that the YALMIP syntax used here is perhaps a bit convoluted for this small problem. However, this type of formulation will more suitable when full MPC problems are solved later.

Solution to Exercise 3.7

(a) First, we rewrite the residuals $r_i = y_i - a^T x_i - b$ in vector form,

$$\begin{bmatrix} r_1 \\ \vdots \\ r_m \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} - \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_m & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

So, $R = Y - \Phi z$. Our cost function is now, $(Y - \Phi z)^T (Y - \Phi z) = Y^T Y - 2\Phi^T Y z + z^T \Phi^T \Phi z = d + 2f^T z + z^T H z$.

```
load noisy_dataset;
M = length(x);
yalmip('clear') % clear environment
Y = y';
Phi = [ones(M,1), x'];
z = sdpvar(2,1);
cost = 0; %total cost
constraints = []; %constraints
H = Phi'*Phi;
f = -Phi'*Y;
d = Y'*Y;
% YALMIP
cost = z'*H*z + 2*f'*z + d;
options = sdpsettings('verbose',1,
                      'solver','quadprog',
                      'savedebug',1,
                      'savesolveroutput',1);
diagnostic = optimize(constraints,cost,options);
cost = double(cost)
z_star = double(z)
figure(1); hold on;
plot(x,y,'.r');
plot(x, Phi*z_star,'r');
```

(b) Lets create a new variable t_i such that $t_i = f_i(r) = |y_i - a^T x_i - b|$, then we can rewrite the optimization problem as:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{m} t_i \\ \text{subject to} & t_i = |y_i - a^T x_i - b|,\ i = 1, ..., m \end{array}$$

The constraints of the problem above are nonlinear. So, we need to transform them into linear constraints. First note that, since we are minimizing t_i we can transform $t_i = |y_i - a^T x_i - b|$ into $t_i \geq |y_i - a^T x_i - b|$. Then, we can transform this inequality into two inequalities

$$t_i \geq y_i - a^T x_i - b$$
$$t_i \geq -y_i + a^T x_i + b$$

Thus, we can formulate the problem as

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{m} t_i \\ \text{subject to} & t_i \geq y_i - a^T x_i - b \\ & t_i \geq -y_i + a^T x_i + b \end{array}$$

In order to formulate the optimization problem as a linear program we define $z' = [t_1\ ...\ t_m\ z]^T$ and rewrite

$$\begin{array}{ll} \text{minimize} & [\mathbf{1}^T\ 0\ 0]z' \\ \text{subject to} & [\mathbf{I}\ \Phi]z' \geq Y \\ & [\mathbf{I}\ -\Phi]z' \geq -Y \end{array}$$

```
load noisy_dataset;
M = length(x);
yalmip('clear') % clear environment
Y = y';
Phi = [ones(M,1), x'];
t = sdpvar(M,1);
z = sdpvar(2,1);
cost = 0; %total cost
constraints = []; %constraints
H = Phi'*Phi;
f = -Phi'*Y;
d = Y'*Y;
% YALMIP
cost = [ones(1,M) 0 0]*[t; z];
constraints = [constraints,
               Y <= [eye(M) Phi]*[t; z],
              -Y <= [eye(M) -Phi]*[t; z]];
options = sdpsettings('verbose',1,
                      'solver','quadprog',
                      'savedebug',1,
                      'savesolveroutput',1);
diagnostic = optimize(constraints,cost,options);
cost = double(cost)
z_star = double(z)
t = double(t);
figure(1); hold on;
plot(x,y,'.r');
plot(x, Phi*z_star,'r');
```

(c) The procedure is the same as before. However, now $t = \max_i(|r_i|)$. So, t is scalar. Defining $z' = [t\ z]^T$ we can formulate the optimization problem as

$$\begin{array}{ll} \text{minimize} & [1\ 0\ 0]z' \\ \text{subject to} & [\mathbf{1}\ \Phi]z' \geq Y \\ & [\mathbf{1}\ -\Phi]z' \geq -Y \end{array}$$

```
load noisy_dataset;
M = length(x);
yalmip('clear') % clear environment
Y = y';
Phi = [ones(M,1), x'];
t = sdpvar(1,1);
z = sdpvar(2,1);
cost = 0; %total cost
constraints = []; %constraints
H = Phi'*Phi;
f = -Phi'*Y;
d = Y'*Y;
% YALMIP
cost = [1 0 0]*[t; z];
constraints = [constraints,
               Y <= [ones(M,1) Phi]*[t;z],
              -Y <= [ones(M,1) -Phi]*[t;z]];
options = sdpsettings('verbose',1,
                      'solver','quadprog',
                      'savedebug',1,
                      'savesolveroutput',1);
diagnostic = optimize(constraints,cost,options);
cost = double(cost)
z_star = double(z)
t = double(t);
figure(1); hold on;
plot(x,y,'.r');
plot(x, Phi*z_star,'r');
```

(d) Implementing in YALMIP we can see that the absolute residuals minimization is less sensitive to outliers that the least squares and the least absolute strategies. See code above.

3.2 Dynamic programming

Solution to Exercise 3.8

(a) It holds that

$$V_3(x) = x^2$$

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \min_{u_2} \{u_2^2 + (x_2 + u_2)^2\} \\ &= \frac{1}{2} x_2^2 \end{aligned}$$

with $\hat{u}_2 = -\frac{1}{2} x_2$. The next step is given by

$$\begin{aligned} V_1(x_1) &= \min_{u_1} \left\{ u_1^2 + \frac{1}{2}(x_1 + u_1)^2 \right\} \\ &= \frac{1}{3} x_1^2 \end{aligned}$$

with $\hat{u}_1 = -\frac{1}{3} x_1$. The final step is given by

$$\begin{aligned} V_0(x_0) &= \min_{u_0} \left\{ u_0^2 + \frac{1}{3}(x_0 + u_0)^2 \right\} \\ &= \frac{1}{4} x_0^2 \end{aligned}$$

with $\hat{u}_0 = -\frac{1}{4}x_0$. Hence, the optimal objective is given by

$$V_0(4) = 4$$

and the optimal input sequence is as follows

$$u_0 = -1, \quad u_1 = -1, \quad u_2 = -1$$

(b) The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \min_{u_2} \{u_2^2\} \\ &= 4x_2^2 \end{aligned}$$

since $\hat{u}_2 = 2x_2$ is required to achieve $x_3 = 0$. The next step is given by

$$\begin{aligned} V_1(x_1) &= \min_{u_1} \{u_1^2 + 4(2x_1 - u_1)^2\} \\ &= \frac{16}{5}x_1^2 \end{aligned}$$

with $\hat{u}_1 = \frac{8}{5}x_1$. The final step is given by

$$\begin{aligned} V_0(x_0) &= \min_{u_0} \left\{ u_0^2 + \frac{16}{5}(2x_0 - u_0)^2 \right\} \\ &= \frac{64}{21}x_0^2 \end{aligned}$$

with $\hat{u}_0 = \frac{32}{21}x_0$. Hence, the optimal objective is given by

$$V_0(21) = 1344$$

and the optimal input sequence is as follows

$$u_0 = 32, \quad u_1 = 16, \quad u_2 = 8$$

(c) It holds that

$$V_3(x) = x^2$$

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \min_{u_2} \{x_2^2 + u_2^2 + (2x_2 + u_2)^2\} \\ &= 3x_2^2 \end{aligned}$$

with $\hat{u}_2 = -x_2$. The next step is given by

$$\begin{aligned} V_1(x_1) &= \min_{u_1} \{x_1^2 + u_1^2 + 3(2x_1 + u_1)^2\} \\ &= 4x_1^2 \end{aligned}$$

with $\hat{u}_1 = -\frac{3}{2}x_1$. The final step is given by

$$\begin{aligned} V_0(x_0) &= \min_{u_0}\{x_0^2 + u_0^2 + 4(2x_0 + u_0)^2\} \\ &= \frac{21}{5}x_0^2 \end{aligned}$$

with $\hat{u}_0 = -\frac{8}{5}x_0$. Hence, the optimal objective is given by

$$V_0(5) = 210$$

and the optimal input sequence is as follows

$$u_0 = -8, \quad u_1 = -3, \quad u_2 = -1$$

Solution to Exercise 3.9

(a) It holds that

$$V_3(x) = -x$$

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \max_{0 \le u_2 \le 1}\{x_2 - x_2 u_2\} \\ &= x_2 \end{aligned}$$

with $\hat{u}_2 = 0$. This only holds if $x_2 \ge 0$, which will be true since $x_0 > 1$ and $0 \le u_k \le 1$ for $k = 0, \ldots, 2$. The next step is given by

$$\begin{aligned} V_1(x_1) &= \max_{0 \le u_1 \le 1}\{x_1 + x_1 u_2\} \\ &= 2x_1 \end{aligned}$$

with $\hat{u}_1 = 1$ (again, $x_1 \ge 0$ will hold). The final step is given by

$$\begin{aligned} V_0(x_0) &= \max_{0 \le u_0 \le 1}\{x_0 + 2x_1 u_1\} \\ &= 3x_0 \end{aligned}$$

with $\hat{u}_0 = 1$ ($x_0 \ge 0$). Hence, the optimal objective is given by

$$V_0(1) = 3$$

and the optimal input sequence is as follows

$$u_0 = 1, \quad u_1 = 1, \quad u_2 = 0$$

(b) It holds that

$$V_3(x) = \log x$$

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \max_{0<u_2<1} \{\log u_2 x_2 + \log x_2(1-u_2)\} \\ &= \max_{0<u_2<1} \left\{\log x_2^2 u_2(1-u_2)\right\} \\ &= \log \frac{x_2^2}{4} \end{aligned}$$

with $\hat{u}_2 = \frac{1}{2}$. The next step is given by

$$\begin{aligned} V_1(x_1) &= \max_{0<u_1<1} \left\{\log u_1 x_1 + \log \frac{x_1^2(1-u_1)^2}{4}\right\} \\ &= \max_{0<u_1<1} \left\{\log \frac{x_1^3 u_1(1-u_1)^2}{4}\right\} \\ &= \log \frac{1}{27} x_1^3 \end{aligned}$$

with $\hat{u}_1 = \frac{1}{3}$. The final step is given by

$$\begin{aligned} V_0(x_0) &= \max_{0<u_0<1} \left\{\log u_0 x_0 + \log \frac{x_0^3(1-u_0)^3}{27}\right\} \\ &= \max_{0<u_0<1} \left\{\log \frac{x_0^4 u_0(1-u_0)^3}{27}\right\} \\ &= \log \frac{1}{256} x_0^4 \end{aligned}$$

with $\hat{u}_0 = \frac{1}{4}$. Hence, the optimal objective is given by

$$V_0(4) = 0$$

and the optimal input sequence is as follows

$$u_0 = \frac{1}{4}, \quad u_1 = \frac{1}{3}, \quad u_2 = \frac{1}{2}$$

(c) It holds that

$$V_3(x) = x^4$$

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \min_{u_2} \left\{u_2^4 + (x_2+u_2)^4\right\} \\ &= \frac{1}{8} x^4 \end{aligned}$$

with $\hat{u}_2 = -\frac{1}{2} x_2$. The next step is given by

$$\begin{aligned} V_1(x_1) &= \min_{u_1} \left\{(u_1^4 + \frac{1}{8}(x_1+u_1)^4)\right\} \\ &= \frac{1}{27} x^4 \end{aligned}$$

with $\hat{u}_1 = -\frac{1}{3}x$. The final step is given by

$$\begin{aligned} V_0(x_0) &= \min_{u_1}\left\{(u_0^4 + \frac{1}{27}(x_0+u_0)^4)\right\} \\ &= \frac{1}{64}x_0^4 \end{aligned}$$

with $\hat{u}_1 = -\frac{1}{4}x$. Hence, the optimal objective is given by

$$V_0(4) = 4$$

and the optimal input sequence is as follows

$$u_0 = -1, \quad u_1 = -1, \quad u_2 = -1$$

Solution to Exercise 3.10

(a) It holds that

$$V_3(x) = x^2$$

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \min_{u_2, x_3 \geq 2}\left\{u_2^2 + (x_2+u_2)^2\right\} \\ &= \begin{cases} \frac{1}{2}x_2^2, & x_2 \geq 4 \\ (2-x_2)^2 + 4, & x_2 < 4 \end{cases} \end{aligned}$$

with

$$\hat{u}_2 = \begin{cases} -\frac{1}{2}x_2, & x_2 \geq 4 \\ 2 - x_2, & x_2 < 4 \end{cases}$$

The next step is given by

$$\begin{aligned} V_1(x_1) &= \min_{u_1, x_2 \geq 2}\left\{u_1^2 + V_2(x_1+u_1)\right\} \\ &= \min_{u_1, x_2 \geq 2}\left\{u_1^2 + \begin{cases} \frac{1}{2}(x_1+u_1)^2, & x_1+u_1 \geq 4 \\ (2-x_1-u_1)^2 + 4, & x_1+u_1 < 4 \end{cases}\right\} \end{aligned}$$

Assume first that $x_1 + u_1 \geq 4$. It follows that

$$\begin{aligned} V_1(x_1) &= \min_{u_1, x_2 \geq 2}\left\{u_1^2 + \frac{1}{2}(x_1+u_1)^2\right\} \\ &= \frac{1}{3}x_1^2 \end{aligned}$$

with $\hat{u}_1 = -\frac{1}{3}x_1$. The control is necessarily admissible since $x_1 + u_1 \geq 4 > 2$ was assumed. Next, assume instead that $x_1 + u_1 < 4$. It holds that

$$\begin{aligned} V_1(x_1) &= \min_{u_1, x_2 \geq 2}\left\{u_1^2 + (2-x_1-u_1)^2 + 4\right\} \\ &= 2(1 - \frac{1}{2}x_1)^2 + 4 \end{aligned}$$

with

$$\hat{u}_1 = 1 - \frac{1}{2}x_1$$

which is admissible for $x_1 \geq 2$. Therefore, it can be concluded that

$$V_1(x_1) = \begin{cases} \frac{1}{3}x_1^2, & x_1 \geq 6 \\ 2(1 - \frac{1}{2}x_1)^2 + 4, & 2 \geq x_1 < 6 \end{cases}$$

with

$$\hat{u}_1 = \begin{cases} -\frac{1}{3}x_1, & x_1 \geq 6 \\ 1 - \frac{1}{2}x_1, & 2 \geq x_1 < 6 \end{cases}$$

The final step is given by

$$\begin{aligned} V_0(x_0) &= \min_{u_0, x_1 \geq 2} \left\{ u_0^2 + V_1(x_0 + u_0) \right\} \\ &= \min_{u_0, x_1 \geq 2} \left\{ u_0^2 + \begin{cases} \frac{1}{3}(x_0 + u_0)^2, & x_1 + u_1 \geq 6 \\ 2(1 - \frac{1}{2}(x_0 + u_0))^2 + 4, & x_1 + u_1 < 6 \end{cases} \right\} \end{aligned}$$

Assume first that $x_0 + u_0 \geq 6$. It follows that

$$\begin{aligned} V_0(x_0) &= \min_{u_0, x_1 \geq 2} \left\{ u_0^2 + \frac{1}{3}(x_0 + u_0)^2 \right\} \\ &= \frac{1}{4}x_0^2 \end{aligned}$$

with $\hat{u}_0 = -\frac{1}{4}x_0$. The control is necessarily admissible since $x_0 + u_0 \geq 6 > 4$ was assumed. Next, assume instead that $x_0 + u_0 < 6$. It holds that

$$\begin{aligned} V_0(x_0) &= \min_{u_0, x_1 \geq 2} \left\{ u_0^2 + 2(1 - \frac{1}{2}(x_0 + u_0))^2 + 4 \right\} \\ &= 3\left(\frac{2 - x_0}{3} \right)^2 + 4 \end{aligned}$$

with

$$\hat{u}_0 = \frac{2 - x_0}{3}$$

which is admissible for $x_0 \geq 2$. Therefore, it can be concluded that

$$V_0(x_0) = \begin{cases} \frac{1}{4}x_0^2, & x_0 \geq 8 \\ 3\left(\frac{2-x_0}{3}\right)^2 + 4, & 2 \geq x_0 < 8 \end{cases}$$

with

$$\hat{u}_0 = \begin{cases} -\frac{1}{4}x_0, & x_0 \geq 8 \\ \frac{2-x_0}{3}, & 2 \geq x_0 < 8 \end{cases}$$

Hence, the optimal objective is given by

$$V_0(4) = \frac{16}{3} (\approx 5.3 > 4)$$

and the optimal input sequence is as follows

$$u_0 = -\frac{2}{3}, \quad u_1 = -\frac{2}{3}, \quad u_2 = -\frac{2}{3}$$

(b) It holds that

$$V_3(x) = \log x$$

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \max_{0<u_2<1, x_3 \geq 2} \{\log u_2 x_2 + \log x_2(1-u_2)\} \\ &= \max_{0<u_2<1, x_3 \geq 2} \log x_2^2 u_2(1-u_2) \\ &= \begin{cases} \log \frac{x_2^2}{4}, & x_2 \geq 4 \\ \log[2x_2 - 4], & 2 < x_2 < 4 \end{cases} \end{aligned}$$

with

$$\hat{u}_2 = \begin{cases} \frac{1}{2}, & x_2 \geq 4 \\ 1 - \frac{2}{x_2}, & 2 < x_2 < 4 \end{cases}$$

Note, that there no admissible control defined for $x_2 = 2$. The next step is given by

$$\begin{aligned} V_1(x_1) &= \max_{0<u_1<1, x_2 \geq 2} \{\log u_1 x_1 + V_2(x_1(1-u_1))\} \\ &= \max_{0<u_1<1, x_2 \geq 2} \left\{ \log u_1 x_1 + \begin{cases} \log \frac{(x_1(1-u_1))^2}{4}, & x_1(1-u_1) \geq 4 \\ \log[2x_1(1-u_1) - 4], & 2 < x_1(1-u_1) < 4 \end{cases} \right\} \end{aligned}$$

Assume first that $x_1(1-u_1) \geq 4$. It follows that

$$\begin{aligned} V_1(x_1) &= \max_{0<u_1<1, x_2 \geq 2} \left\{ \log u_1 x_1 + \log \frac{x_1^2(1-u_1)^2}{4} \right\} \\ &= \max_{0<u_1<1, x_2 \geq 2} \left\{ \log \frac{x_1^3 u_1 (1-u_1)^2}{4} \right\} \\ &= \log \frac{1}{27} x_1^3 \end{aligned}$$

with $\hat{u}_1 = \frac{1}{3}$. The control is necessarily admissible since $x_1(1-u_1) \geq 4 > 2$ was assumed. Next, assume instead that $x_1(1-u_1) < 4$. It holds that

$$\begin{aligned} V_1(x_1) &= \max_{0<u_1<1, x_2 \geq 2} \{\log u_1 x_1 + \log[2x_1(1-u_1) - 4]\} \\ &= \max_{0<u_1<1, x_2 \geq 2} \{\log[2x_1^2 u_1(1-u_1) - 4u_1 x_1]\} \\ &= \log\left[\frac{1}{2}(x_1 - 2)^2\right] \end{aligned}$$

with

$$\hat{u}_1 = \frac{x_1 - 2}{2x_1}$$

which is admissible if $x_1 > 2$. Again, there is no defined control for $x_1 = 2$. Therefore, it can be concluded that

$$V_1(x_1) = \begin{cases} \log \frac{1}{27} x_1^3, & x_1 \geq 6 \\ \log\left[\frac{1}{2}(x_1 - 2)^2\right], & 2 < x_1 < 6 \end{cases}$$

with

$$\hat{u}_1 = \begin{cases} \frac{1}{3}, & x_1 \geq 6 \\ \frac{x_1-2}{2x_1}, & 2 < x_1 < 6 \end{cases}$$

The final step is given by

$$\begin{aligned} V_0(x_0) &= \max_{0<u_0<1, x_1 \geq 2} \{\log u_0 x_0 + V_1(x_0(1-u_0))\} \\ &= \max_{0<u_0<1, x_1 \geq 2} \left\{ \log u_0 x_0 + \begin{cases} \log \frac{1}{27}(x_0(1-u_0))^3, & x_0(1-u_0) \geq 6 \\ \log\left[\frac{1}{2}(x_0(1-u_0)-2)^2\right], & x_0(1-u_0) < 6 \end{cases} \right\} \end{aligned}$$

Assume first that $x_0(1-u_0) \geq 6$. It follows that

$$\begin{aligned} V_0(x_0) &= \max_{0<u_0<1, x_1 \geq 2} \left\{ \log u_0 x_0 + \log \frac{1}{27}(x_0(1-u_0))^2 \right\} \\ &= \max_{0<u_0<1, x_1 \geq 2} \left\{ \log \frac{x_0^4 u_0 (1-u_0)^3}{27} \right\} \\ &= \log \frac{1}{256} x_0^4 \end{aligned}$$

with $\hat{u}_0 = \frac{1}{4}$. The control is necessarily admissible since $x_0(1-u_0) \geq 6 > 4$ was assumed. Next, assume instead that $x_0(1-u_0) < 6$. It holds that

$$\begin{aligned} V_0(x_0) &= \max_{0<u_0<1, x_1 \geq 2} \left\{ \log u_0 x_0 + \log\left[\frac{1}{2}(x_0(1-u_0)-2)^2\right] \right\} \\ &= \max_{0<u_0<1, x_1 \geq 2} \left\{ \log\left[\frac{1}{2} x_0 u_0 (x_0(1-u_0)-2)^2\right] \right\} \\ &= \log\left[\frac{2}{27}(x_0-2)^3\right] \end{aligned}$$

with

$$\hat{u}_0 = \frac{x_0 - 2}{3x_0}$$

which is again admissible for $x_0 > 2$ and there is no defined control for $x_0 = 2$. Therefore, it can be concluded that

$$V_0(x_0) = \begin{cases} \log \frac{1}{256} x_0^4, & x_0 \geq 8 \\ \log\left[\frac{2}{27}(x_0-2)^3\right], & x_0 < 8 \end{cases}$$

with

$$\hat{u}_0 = \begin{cases} \frac{1}{4}, & x_0 \geq 8 \\ \frac{x_0-2}{3x_0}, & 2 < x_0 < 8 \end{cases}$$

Hence, the optimal objective is given by

$$V_0(4) = \log \frac{16}{27} (\approx -0.52 < 0)$$

and the optimal input sequence is as follows

$$u_0 = \frac{1}{6}, \quad u_1 = \frac{1}{5}, \quad u_2 = \frac{1}{4}$$

Note, that the optimal input sequence is only defined for $x_0 > 2$.

(c) For $x_0 = 2$, the optimal solution is to do nothing in problem (a), so that the state rests at the constraint bound throughout the period. For problem (b), there is no feasible input sequence for $x_0 = 2$. For $2 < x_0 < 8$, the optimal solution in both problems is to drive the system to $x_3 = 2$ in the most cost-effective way possible. For $x_0 > 8$, the unconstrained optimal input sequences from Exercise 3.8 and Exercise 3.9 become feasible in both problems.

Solution to Exercise 3.11

(a) The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_1(x_1) &= \max_{|u_1| \leq 1} \{0.4x_1 - 2(x_1 + u_1)\} \\ &= \max_{|u_1| \leq 1} \{-1.6x_1 - 2u_2\} \\ &= -1.6x_1 + 2 \end{aligned}$$

with $\hat{u}_1 = -1$ The next step, which is the last step, is given by

$$\begin{aligned} V_0(x_0) &= \max_{|u_0| \leq 1} \{0.4x_0 + V_1(x_0 + u_0)\} \\ &= \max_{|u_0| \leq 1} \{0.4x_0 - 1.6(x_0 + u_0) + 2\} \\ &= \max_{|u_0| \leq 1} \{-1.2x_0 - 1.6u_0 + 2\} \\ &= -1.2x_0 + 3.6 \end{aligned}$$

with $\hat{u}_0 - 1$. In brief, the optimal strategy is $\hat{u}_0 = -1$, $\hat{u}_1 = -1$ with profit

$$3 + V_0(3) = 3$$

In other words, sell of as much as possible each year.

(b) From the results in (a), a reasonable value function candidate would be

$$V_{k+1}(x_{k+1}) = a_{k+1}x_{k+1} + b_{k+1}$$

Inserting the guess in the dynamic programming recursion gives

$$\begin{aligned} V_k(x_k) &= \max_{|u_k| \leq 1} \{\theta x_k + a_{k+1}(x_k + u_k) + b_{k+1}\} \\ &= \max_{|u_k| \leq 1} \{(a_{k+1} + \theta)x_k + a_{k+1}u_k + b_{k+1}\} \end{aligned}$$

The optimal choice of u_k is given by

$$\hat{u}_k = \begin{cases} 1, & a_{k+1} \geq 0 \\ -1, & a_{k+1} < 0 \end{cases} = \operatorname{sgn}(a_{k+1})$$

so that

$$V_k(x_k) = \begin{cases} (a_{k+1} + \theta)x_k + b_{k+1} + a_{k+1}, & a_{k+1} \geq 0 \\ (a_{k+1} + \theta)x_k + b_{k+1} - a_{k+1}, & a_{k+1} < 0 \end{cases}$$

Hence, $V_k(x_k)$ is on the form $V_k(x_k) = a_k x_k + b_k$, where

$$a_k = a_{k+1} + \theta$$

and

$$b_k = \begin{cases} b_{k+1} + a_{k+1}, & a_{k+1} \geq 0 \\ b_{k+1} - a_{k+1}, & a_{k+1} < 0 \end{cases} = b_{k+1} + |a_{k+1}|$$

By induction, it follows that the above value function is valid.

(c) At $N-1$, the value function is given by

$$\begin{aligned} V_{N-1}(x_{N-1}) &= \max_{|u_{N-1}| \leq 1} \{\theta x_{N-1} - 2(x_{N-1} + u_{N-1})\} \\ &= (\theta - 2)x_{N-1} + 2 \end{aligned}$$

with $\hat{u}_{N-1} = -1$. Therefore, $a_{N-1} = \theta - 2$ and $b_{N-1} = 2$. Using the recurrence relation obtained in (b), a closed form expression for a_k is given by

$$a_k = (N-k)\theta - 2 \Rightarrow a_{k+1} = (N-k-1)\theta - 2$$

Thus, the optimal input is given by

$$\hat{u}_k(\theta, N) = \begin{cases} 1, & \theta \geq \frac{2}{N-k-1} \\ -1, & \theta < \frac{2}{N-k-1} \end{cases} \quad k = 0, 1, \ldots, N-2$$

and $\hat{u}_{N-1}(\theta, N) = -1$.

(d) θ defines a cut-off point, which specifies how many years it is profitable to increase the share holdings. Specifically, it is optimal to buy a share each year up to the cut-off point

$$k = \left\lfloor \frac{\theta(N-1) - 2}{\theta} \right\rfloor$$

and afterwards sell off a share each year. The horizon, N, has to be large enough for it to even be profitable to increase the share holdings. Specifically, for a given θ, if

$$N < \frac{2}{\theta} + \theta$$

then

$$\theta < \frac{2}{N-k-1}, \quad k = 0, 1, \ldots, N-2$$

and it is optimal to sell off a share each year (as in (a)). In brief, large θ and N implies a longer period spent on increasing share holdings, whereas lower values on θ and N implies that most of the time is spent on selling of shares.

Solution to Exercise 3.12

The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_1(x_1) &= \max_{u_1 \in \{-1,0,1\}} \{0.4x_1 - 2(x_1 + u_1)\} \\ &= \max_{u_1 \in \{-1,0,1\}} \{-1.6x_1 - 2u_2\} \\ &= -1.6x_1 + 2 \end{aligned}$$

with $\hat{u}_1 = -1$. The next step, which is the last step, is given by

$$\begin{aligned} V_0(x_0) &= \max_{u_0 \in \{-1,0,1\}} \{0.4x_0 + V_1(x_0 + u_0)\} \\ &= \max_{u_0 \in \{-1,0,1\}} \{0.4x_0 - 1.6(x_0 + u_0) + 2\} \\ &= \max_{u_0 \in \{-1,0,1\}} \{-1.2x_0 - 1.6u_0 + 2\} \\ &= -1.2x_0 + 3.6 \end{aligned}$$

with $\hat{u}_0 - 1$. In brief, the optimal strategy is $\hat{u}_0 = -1$, $\hat{u}_1 = -1$ with profit

$$3 + V_0(3) = 3$$

i.e. sell of as much as possible each year.

(a) The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_2(x_2) &= \max_{u_2 \in \{-1,0,1\}, x_3 \geq 0} \{0.8x_2 - 2(x_2 + u_2)\} \\ &= \max_{u_2 \in \{-1,0,1\}, x_3 \geq 0} \{-1.2x_2 - 2u_2\} \\ &= \begin{cases} -1.2x_2 + 2, & x_2 \geq 1 \\ 0, & x_2 = 0 \end{cases} \end{aligned}$$

with

$$\hat{u}_2 = \begin{cases} -1, & x_2 \geq 1 \\ 0, & x_2 = 0 \end{cases}$$

The next step is given by

$$\begin{aligned} V_1(x_1) &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \{0.8x_1 + V_2(x_1 + u_1)\} \\ &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \left\{ 0.8x_1 + \begin{cases} -1.2(x_1 + u_1) + 2, & x_1 + u_1 \geq 1 \\ 0, & x_1 + u_1 = 0 \end{cases} \right\} \end{aligned}$$

Assume first that $x_1 \geq 2$. It follows that $x_1 + u_1 \geq 1$ and

$$\begin{aligned} V_1(x_1) &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \{0.8x_1 - 1.2(x_1 + u_1) + 2\} \\ &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \{-0.4x_1 - 1.2u_1 + 2\} \\ &= -0.4x_1 + 3.2 \end{aligned}$$

with $\hat{u}_1 = -1$. The control is necessarily admissible since $x_1 + u_1 \geq 1 > 0$ was assumed. Next, assume instead that $x_1 = 1$. It follows that

$$\begin{aligned} V_1(x_1) &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \{0.8 + V_2(1 + u_1)\} \\ &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \left\{ 0.8 + \begin{cases} -0.4, & u_1 = 1 \\ 0.8, & u_1 = 0 \\ 0, & u_1 = -1 \end{cases} \right\} \\ &= 1.6 \end{aligned}$$

with

$$\hat{u}_1 = 0$$

Finally, assume that $x_1 = 0$. It follows that

$$\begin{aligned} V_1(x_1) &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \{V_2(u_1)\} \\ &= \max_{u_1 \in \{-1,0,1\}, x_2 \geq 0} \left\{ \begin{cases} -0.4, & u_1 = 1 \\ 0, & u_1 = 0 \end{cases} \right\} \\ &= 0 \end{aligned}$$

with

$$\hat{u}_1 = 0$$

Therefore, it can be concluded that

$$V_1(x_1) = \begin{cases} -0.4x_1 + 3.2, & x_1 \geq 2 \\ 1.6, & x_1 = 1 \\ 0, & x_1 = 0 \end{cases}$$

with

$$\hat{u}_1 = \begin{cases} -1, & x_1 \geq 2 \\ 0, & x_1 < 2 \end{cases}$$

The final step is given by

$$\begin{aligned} V_0(x_0) &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \{0.8x_0 + V_1(x_0 + u_0)\} \\ &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \left\{ 0.8x_0 + \begin{cases} -0.4(x_0 + u_0) + 3.2, & x_0 + u_0 \geq 2 \\ 1.6, & x_0 + u_0 = 1 \\ 0, & x_0 + u_0 = 0 \end{cases} \right\} \end{aligned}$$

Assume first that $x_0 \geq 3$. It follows that $x_0 + u_0 \geq 2$ and

$$\begin{aligned} V_0(x_0) &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \{0.8x_0 - 0.4(x_0 + u_0) + 3.2\} \\ &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \{0.4x_0 - 0.4u_0 + 3.2\} \\ &= 0.4x_0 + 3.6 \end{aligned}$$

with $\hat{u_0} = -1$. The control is necessarily admissible since $x_0 + u_0 \geq 2 > 0$ was assumed. Next, assume instead that $x_0 = 2$. It follows that

$$\begin{aligned} V_0(x_0) &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \{1.6 + V_1(2 + u_0)\} \\ &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \left\{ 1.6 + \begin{cases} 2, & u_0 = 1 \\ 2.4, & u_0 = 0 \\ 1.6, & u_0 = -1 \end{cases} \right\} \\ &= 4 \end{aligned}$$

with

$$\hat{u}_0 = 0$$

Next, assume that $x_0 = 1$. It follows that

$$\begin{aligned} V_0(x_0) &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \{0.8 + V_1(1 + u_0)\} \\ &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \left\{ 0.8 + \begin{cases} 2.4, & u_0 = 1 \\ 1.6, & u_0 = 0 \\ 0, & u_0 = -1 \end{cases} \right\} \\ &= 3.2 \end{aligned}$$

with

$$\hat{u}_0 = 1$$

Finally, assume that $x_0 = 0$. It follows that

$$\begin{aligned} V_0(x_0) &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \{V_1(u_0)\} \\ &= \max_{u_0 \in \{-1,0,1\}, x_1 \geq 0} \left\{ \begin{cases} 1.6, & u_0 = 1 \\ 0, & u_0 = 0 \end{cases} \right\} \\ &= 1.6 \end{aligned}$$

with

$$\hat{u}_0 = 1$$

Therefore, it can be concluded that

$$V_0(x_0) = \begin{cases} 0.4x_0 + 3.6, & x_0 \geq 3 \\ 4, & x_0 = 2 \\ 3.2, & x_0 = 1 \\ 1.6, & x_0 = 0 \end{cases}$$

with

$$\hat{u}_0 = \begin{cases} -1, & x_0 \geq 3 \\ 0, & x_0 = 2 \\ 1, & x_0 < 2 \end{cases}$$

In brief, the optimal strategy is $\hat{u}_0 = 1$, $\hat{u}_1 = -1$, $\hat{u}_2 = -1$ with profit

$$1 + V_0(1) = 4.2$$

Solution to Exercise 3.13

(a) The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_1(x_1) &= \max_{0 \leq u_1 \leq 1} \{(1 - u_1)x_1\} \\ &= x_1 \end{aligned}$$

with $\hat{u}_1 = 0$ The next step, which is the last step, is given by

$$\begin{aligned} V_0(x_0) &= \max_{0\leq u_0\leq 1}\{(1-u_0)x_0 + V_1(x_0 + \frac{3}{2}u_0x_0)\} \\ &= \max_{0\leq u_0\leq 1}\{(1-u_0)x_0 + x_0 + \frac{3}{2}u_0x_0\} \\ &= \max_{0\leq u_0\leq 1}\{(2+\frac{1}{2}u_0)x_0\} \\ &= \frac{5}{2}x_0 \end{aligned}$$

with $\hat{u}_0 = 1$. In brief, the optimal strategy is $\hat{u}_0 = 1$, $\hat{u}_1 = 0$ with profit

$$V_0(3) = 7.5$$

being distributed to the shareholders.

(b) From the results in (a), a reasonable value function candidate would be

$$V_{k+1}(x_{k+1}) = c_{k+1}x_{k+1}$$

Inserting the guess in the dynamic programming recursion gives

$$\begin{aligned} V_k(x_k) &= \max_{0\leq u_k\leq 1}\{(1-u_k)x_k + c_{k+1}(x_k + \theta u_k x_k)\} \\ &= \max_{0\leq u_k\leq 1}\{((1+c_{k+1}) + (c_{k+1}\theta - 1)u_k)x_k\} \\ &= \max\ [1+c_{k+1}, (1+\theta)c_{k+1}]x_k \end{aligned}$$

Now, introduce the recursion

$$c_k = c_{k+1} + \max[1, \theta c_{k+1}]$$

so that

$$V_k(x_k) = c_k x_k$$

By induction, it follows that the above value function is valid. The optimal choice of u_k is determined by the largest expression in the max term above. In brief, $u_k = 0$ when

$$\theta \leq \frac{1}{c_{k+1}}$$

and 1 otherwise. Since $V_N(x_N) = 0$, it holds that

$$c_k = c_{k+1} + 1 = N - k$$

until the cut-off point $\tilde{k}$ when

$$\theta > \frac{1}{c_{\tilde{k}+1}} = \frac{1}{N-k-1}$$

Therefore, the optimal control law is given by

$$\hat{u}_k = \begin{cases} 1, & \theta > \frac{1}{N-k-1} \\ 0, & \theta \leq \frac{1}{N-k-1} \end{cases}$$

(c) In brief, the optimal strategy is to invest all profits in the early years to build capital, and distribute everything during the last years of the period. The cut-off point is given by

$$k = \left\lfloor \frac{\theta(N-1)-1}{\theta} \right\rfloor$$

As in Exercise 3.11, if the horizon N is not large enough, all profits are distributed each year in the period.

Solution to Exercise 3.14

(a) The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_1(x_1) &= \max_{0\le u_1\le 1}\{\log(1-u_1)x_1\} \\ &= \log x_1 \end{aligned}$$

with $\hat{u}_1 = 0$. The next step, which is the last step, is given by

$$\begin{aligned} V_0(x_0) &= \max_{0\le u_0\le 1}\{\log(1-u_0)x_0 + V_1(x_0 + \frac{3}{2}u_0x_0)\} \\ &= \max_{0\le u_0\le 1}\{\log(1-u_0)x_0 + \log[x_0 + \frac{3}{2}u_0x_0]\} \\ &= \max_{0\le u_0\le 1}\{\log[(1-u_0)x_0^2 + \frac{3}{2}x_0^2u_0(1-u_0)]\} \\ &= 2\log x_0 + \log\frac{25}{24} \end{aligned}$$

with $\hat{u}_0 = \frac{1}{6}$. In brief, the optimal strategy is $\hat{u}_0 = \frac{1}{6}$, $\hat{u}_1 = 0$ with profit

$$\frac{5}{6}\cdot 3 + (3 + \frac{3}{2}\cdot\frac{1}{6}3) = 6.25$$

being distributed to the shareholders. The total distributed amount if less than in Exercise 3.12, but the shareholders received dividends every year.

(b) From the results in (a), a reasonable value function candidate would be

$$V_{k+1}(x_{k+1}) = a_{k+1}\log x_{k+1} + b_{k+1}$$

Inserting the guess in the dynamic programming recursion gives

$$\begin{aligned} V_k(x_k) &= \max_{0\le u_k<1}\{\log(1-u_k)x_k + a_{k+1}\log[x_k + \theta u_k x_k] + b_{k+1}\} \\ &= \begin{cases} (a_{k+1}+1)\log x_k + b_{k+1} + (a_{k+1}+1)\log\frac{a_{k+1}(\theta+1)}{\theta(1+a_{k+1})}, & \theta > \frac{1}{a_{k+1}} \\ (a_{k+1}+1)\log x_k + b_{k+1}, & \theta \le \frac{1}{a_{k+1}} \end{cases} \end{aligned}$$

with

$$\hat{u}_k = \begin{cases} \frac{1}{1+a_{k+1}}\left(a_{k+1} - \frac{1}{\theta}\right), & \theta > \frac{1}{a_{k+1}} \\ 0, & \theta \le \frac{1}{a_{k+1}} \end{cases}$$

Hence, $V_k(x_k)$ is on the form $V_k(x_k) = a_k \log x_k + b_k$, where

$$a_k = a_{k+1} + 1$$

and

$$b_k = \begin{cases} b_{k+1} + (a_{k+1}+1)\log \frac{a_{k+1}(\theta+1)}{\theta(1+a_{k+1})}, & \theta > \frac{1}{a_{k+1}} \\ b_{k+1}, & \theta \leq \frac{1}{a_{k+1}} \end{cases}$$

By induction, it follows that the above value function is valid. At $N-1$, the value function is given by

$$\begin{aligned} V_{N-1}(x_{N-1}) &= \max_{0 \leq u_{N-1} \leq 1} \{\log(1-u_{N-1})x_{N-1}\} \\ &= \log x_{N-1} \end{aligned}$$

with $\hat{u}_{N-1} = 0$. Therefore, $a_{N-1} = 1$ and $b_{N-1} = 0$. Consequently,

$$a_k = (N-k) \Rightarrow a_{k+1} = (N-k-1)$$

Thus, the optimal input is given by

$$\hat{u}_k(\theta, N) = \begin{cases} \frac{1}{N-k}\left(N-k-1-\frac{1}{\theta}\right), & \theta > \frac{1}{N-k-1} \\ 0, & \theta \leq \frac{1}{N-k-1} \end{cases} \quad k = 0, 1, \ldots, N-2$$

(c) In brief, the optimal strategy is to invest a portion of the profits in the early years to build capital, and distribute everything during the last years of the period. The cut-off point is given by

$$k = \left\lfloor \frac{\theta(N-1)-1}{\theta} \right\rfloor$$

In other words, the cut-off point is the same as in Exercise 3.12. The difference is that some capital is always being distributed to the shareholders during the capital building period. As before, if the horizon N is not large enough, all profits are distributed each year in the period.

Solution to Exercise 3.15

(a) The first step of the dynamic programming recursion is given by

$$\begin{aligned} V_3(R_2) &= \max_{x_3 \in \{0,1\}, 3x_3 \leq R_2} \{12x_3\} \\ &= \begin{cases} 12, & \frac{R_2}{3} \geq 1 \\ 0, & \frac{R_2}{3} < 1 \end{cases} \end{aligned}$$

with

$$\hat{x}_3 = \begin{cases} 1, & R_2 \geq 3 \\ 0, & R_2 < 3 \end{cases}$$

The next step is given by

$$\begin{aligned} V_2(R_1) &= \max_{x_2 \in \{0,1\}, 2x_2 \leq R_1} \{10x_2 + V_2(R_1 - 2x_2)\} \\ &= \max_{x_2 \in \{0,1\}, 2x_2 \leq R_1} \left\{ 10x_2 + \begin{cases} 12, & R_1 - 2x_2 \geq 3 \\ 0, & R_1 - 2x_2 < 3 \end{cases} \right\} \end{aligned}$$

Assume first that $R_1 - 2x_2 \geq 3$. It follows that

$$\begin{aligned} V_2(R_1) &= \max_{x_2 \in \{0,1\}, 2x_2 \leq R_1} \{10x_2 + 12\} \\ &= 22 \end{aligned}$$

with $\hat{x}_2 = 1$, which is feasible since $2x_2 \leq R_1 - 3 \Rightarrow 2x_2 \leq R_1$ was assumed. Next, assume instead that $R_1 - 2x_2 < 3$, so that

$$\begin{aligned} V_2(R_1) &= \max_{x_2 \in \{0,1\}, 2x_2 \leq R_1} \{10x_2\} \\ &= \begin{cases} 10, & \frac{R_1}{2} \geq 1 \\ 0, & \frac{R_1}{2} < 1 \end{cases} \end{aligned}$$

with

$$\hat{x}_2 = \begin{cases} 1, & R_1 \geq 2 \\ 0, & R_1 < 2 \end{cases}$$

Therefore, it can be concluded that

$$V_2(R_1) = \begin{cases} 22, & R_1 \geq 5 \\ 10, & 2 \leq R_1 < 5 \\ 0, & R_1 < 2 \end{cases}$$

with

$$\hat{x}_1 = \begin{cases} 1, & R_1 \geq 2 \\ 0, & R_1 < 2 \end{cases}$$

The final step is given by

$$\begin{aligned} V_1(R_0) &= \max_{x_1 \in \{0,1\}, x_1 \leq R_0} \{6x_1 + V_1(R_0 - x_1)\} \\ &= \max_{x_1 \in \{0,1\}, x_1 \leq R_0} \left\{ 6x_1 + \begin{cases} 22, & R_0 - x_1 \geq 5 \\ 10, & 2 \leq R_0 - x_1 < 5 \\ 0, & R_0 - x_1 < 2 \end{cases} \right\} \end{aligned}$$

Now, insert $R_0 = 5$ to obtain

$$\begin{aligned} V_1(5) &= \max_{x_1 \in \{0,1\}, x_1 \leq R_0} \left\{ 6x_1 + \begin{cases} 16, & x_1 = 1 \\ 22, & x_1 = 0 \end{cases} \right\} \\ &= 22 \end{aligned}$$

with $\hat{x}_1 = 0$. Hence, the optimal objective is given by

$$V_1(5) = 22$$

and the optimal choice of variables is as follows

$$x_1 = 0, \quad x_2 = 1, \quad x_2 = 1$$

(b) A suggested implementation is shown below.

```
function [xopt,optval] =  knapsack(vs,ws,W)
    N = length(vs);         % Number of stages
    xgrid = zeros(N,W+1);   % Optimal amounts for each remaining
                            % amount of resource in each stage
    V = zeros(N+1,W+1);     % Optimal value as a function of stage and resource

    s = N; % Recurse backwards from N
    while s > 0
        for w = 2:W+1
            if ws(s) <= w-1
                % Maximize over skipping/taking resource
                [v,i] = max([V(s+1,w), vs(s) + V(s+1,w-ws(s))]);
                xgrid(s,w) = i-1; % Remember if resource was put in
                                  % knapsack
                V(s,w) = v;       % Store cost-to-go
            else
                V(s,w) = NaN; % Not possible to add this resource
            end
        end
        s = s - 1;
    end
    % Forward iteration of xgrid to collect optimal amounts
    xopt = zeros(N,1);
    xopt(1) = xgrid(1,end);
    r = W-ws(1)*xopt(1);
    for s = 2:N
        xopt(s) = xgrid(s,r+1);
        r = r - ws(s)*xopt(s);
    end
    % The optimal value corresponds to the full basket size in the
    % first stage
    optval = V(1,end);
end
```

The above function is used to solve the given problems, with the results shown below

```
% a)
v = [1 2 3 2 1]';
w = [1 1 1 1 1]';
W = 3;

[xopt,optval] = knapsack(v,w,W)

xopt =

     0
     1
     1
     1
     0


optval =

     7

% b)
v = [2 2 3 5 3]';
w = [1 2 1 4 2]';
W = 6;

[xopt,optval] = knapsack(v,w,W)
```

```
xopt =

     1
     0
     1
     1
     0


optval =

    10

% c)
v = [3 4 7 2 5 1 3]';
w = [1 2 3 1 4 1 2]';
W = 10;

[xopt,optval] = knapsack(v,w,W)

xopt =

     1
     1
     1
     0
     1
     0
     0


optval =

    19
```

Solution to Exercise 3.16

A suggested implementation is shown below.

```
function [aopt,optval] = cargo_load(v,w,W)
    N = length(v); % Number of stages
    agrid = zeros(N,W+1); % Optimal amounts for each remaining capacity
                          % in each stage
    V = zeros(N+1,W+1);   % Optimal value as a function of stage
                          % and capacity

    s = N; % Recurse backwards from N
    while s > 0
        a_i = 1:(floor(W/w(s))+1); % All possible amounts for quantity a_i
        x_i = 1:(W+1);             % All possible capacities
        if s == 1
            x_i = W+1; % At the first stage, all capacity is available
        end
        K = zeros(length(x_i),length(a_i)); % Grid of optimal value for
                                            % each given capacity and amount
        for x = x_i
            for a = a_i
                if w(s)*(a-1) <= x-1
                    % Value of current amount + the maximum utilization
                    % of the remaining capacity
                    K(x,a) = v(s)*(a-1) + V(s+1,x-w(s)*(a-1));
                else
                    % There is not enough remaining capacity for this amount
                    K(x,a) = NaN;
                end
            end
            [k,i] = max(K(x,:)); % Maximize the cargo value for this capacity
            agrid(s,x) = i-1;    % Remember the optimal amount at this capacity
            f(s,x) = k;          % Store the optimal cargo value for
                                 % this stage and capacity level
        end
```

```
        s = s-1;
    end

    % Forward iteration of agrid to collect optimal amounts
    aopt = zeros(N,1);
    aopt(1) = agrid(1,end);
    r = W-w(1)*aopt(1);
    for s = 2:N
        aopt(s) = agrid(s,r+1);
        r = r - w(s)*aopt(s);
    end
    % The optimal value is given by using all capacity in the first stage
    optval = V(1,end);
end
```

The above function is used to solve the given problems, with the results shown below

```
% a)
v = [31 47 14]';
w = [2 3 1]';
W = 4;

[aopt,optval] = cargo_load(v,w,W)

aopt =

     2
     0
     0


optval =

    62

% b)
v = [5 2 7 1]';
w = [4 6 2 1]';
W = 10;

[aopt,optval] = cargo_load(v,w,W)

aopt =

     0
     0
     5
     0


optval =

    35

% c)
v = [2 4 6 3 20]';
w = [4 2 3 2 8]';
W = 15;

[aopt,optval] = cargo_load(v,w,W)

aopt =

     0
     2
     1
     0
     1


optval =
```

Solution to Exercise 3.17

(a) A suggested implementation is shown below

```
clear all
close all
clc

% inverted pendulum parameters
a0 = -2.5;
a1 = 0.05;
b0 = 2.5;
dt = 0.1;

% Controller parameters
Q = diag([1 0]);
R = 1;

% Continuous-time linearized state-space
Ac = [0 1;-a0 -a1];
Bc = [0; b0];

% Discrete-time linearized state-space
[Ad Bd] = c2d(Ac,Bc,dt);

% Horizon
N = 10;
% Stage cost
Jstage = @(x,u) x'*Q*x + u'*R*u;
% Initial cost-to-go
Jtogo = @(x) x'*Q*x;

% Grid of the state-space
X1 = linspace(-pi/4,pi/4,10);
X2 = linspace(-pi/2,pi/2,5);
U = zeros(length(X1),length(X2), N);
J = zeros(length(X1),length(X2), N);

% calculate optimal control using dynamic programming and gridding
options = optimoptions('fminunc','Display','off');
for n = 1:N
    % calculate cost-to-go at each grid point
    for i = 1:length(X1)
        for j = 1:length(X2)
            % current state
            x = [X1(i); X2(j)];
            % calculate optimal control and cost-to-go
            [U(i,j,n) J(i,j,n)] = fminunc( @(u) Jstage(x,u) + Jtogo(Ad*x+Bd*u), 0, options);
        end
    end
    % new cost-to-go
    Jtogo = @(x) interp2(X1,X2,J(:,:,n)',x(1),x(2),'spline');
end

% simulate inverted pendulum
ts = [];
us = [];
xs = [];
Js = [0];
x = x0;
for k = 1:N
    Jtogo = @(x) interp2(X1,X2,J(:,:,N+1-k)',x(1),x(2), 'spline');
    u = fminunc( @(u) Jstage(x,u) + Jtogo(Ad*x+Bd*u), 0);
    f = @(t,x) [x(2); -a0*sin(x(1))-a1*x(2) + b0*u*cos(x(1))];
    [t,x] = ode45(f,[(k-1)*dt,k*dt],x);
    ts = [ts t'];
    xs = [xs x'];
```

```
    Js = [0];
    for i = 1:size(x,1)
        Js = [Js Js(end) + x(i,:)*Q*x(i,:)' + u'*R*u];
    end
    us = [us u*ones(1,length(t))];
    x = xs(:,end);
end
Js = Js(2:end)
figure(1)
subplot(3,1,1)
plot(ts,xs)
subplot(3,1,2)
plot(ts,us)
subplot(3,1,3)
plot(ts,Js)
```

(b) (i) The value of N has a large impact on the quality of the approximation. At $N = 20$ the shape is nearly identical to the exact cost, at the rather coarse suggested discretization level.

(ii) The discretization does not necessarily have to be fine to get a good approximation of the exact cost. However, too few grid points will make it impossible to match the shape of the exact cost, regardless of the value of N.

(iii) The values of Q and R govern the shape of the cost function, but can also have an impact on the quality of the approximation. Specifically, if the input is penalized more (large R) then a larger value of N is required to achieve a good approximation.

(c) The system is stabilized by the controller.

(d) (i) The system is eventually stabilized after around 10 seconds.

(ii) The system is stabilized more aggressively.

(iii) The system again requires longer time to stabilize

(iv) With $N = 30$, the system can be stabilized in 5 seconds.

4 Linear quadratic regulator

Solution to Exercise 4.1

(a) First, with $V_N(x) = x^2$, consider

$$\begin{aligned} V_{N-1} &= \min_u \left\{ q x_{N-1}^2 + r u_{N-1}^2 + (a x_{N-1} + b u_{N-1})^2 \right\} \\ &= \left[(q+a)^2 - \frac{(ab)^2}{r+b^2} \right] x_{N-1}^2 \end{aligned}$$

with $\hat{u}_{N-1} = -\frac{b}{r+b^2}$. Hence, it is natural to consider the ansatz

$$V_{n+1}(x_{n+1}) = p_{n+1} x_{n+1}^2$$

Next, consider

$$\begin{aligned}
V_n &= \min_u \{ q x_n^2 + r u_n^2 + p_{n+1}(a x_n + b u_n)^2 \} \\
&= \left\{ \hat{u}_n = -\frac{abp_{n+1}}{r + b^2 p_{n+1}} x_n \right\} = \\
&= \left[q + \frac{r a^2 b^2 p_{n+1}^2}{(r + b^2 p_{n+1})^2} + \frac{r^2 a^2 p_{n+1}}{(r + b^2 p_{n+1})^2} \right] x_n^2 \\
&= \left[q + a^2 p_{n+1} - \frac{(abp_{n+1})^2}{r + b^2 p_{n+1}} \right] x_n^2 \\
&:= p_n x_n^2
\end{aligned}$$

By induction, it follows that the value function $V_{n+1}(x_{n+1}) = p_{n+1} x_{n+1}$ is valid for the problem. In brief, the optimal control law is given by

$$\hat{u}_n = -\frac{abp_{n+1}}{r + b^2 p_{n+1}} x_n$$

where $p_N = 1$ and

$$p_n = q + a^2 p_{n+1} - \frac{(abp_{n+1})^2}{r + b^2 p_{n+1}}$$

(b) As the previously derived optimal input sequence is valid for all values of N, and there exists a control law for the infinite-horizon problem, it must hold that the sequence $\{p_n\}_{n=1}^{\infty}$ converges to a steady state value p_{ss}. Hence, p_{ss} must satisfy

$$p_{ss} = q + a^2 p_{ss} - \frac{(abp_{ss})^2}{r + b^2 p_{ss}}$$

the scalar discrete algebraic Riccati equation. The optimal feedback law is given by

$$\hat{u}_n = -\frac{abp_{ss}}{r + b^2 p_{ss}} x_n$$

Solution to Exercise 4.2

See code. Created by Andrea Alessandretti (29-01-2012).

```
function OptimalControlLaw
A = [4/3 -2/3;1 0];
B = [1;0];
n = 2;
C = [-2/3 1];
Q = C'*C+0.001*eye(2);
R = 0.0001;
N     = 8; % Prediction horizon
SOLVE_TYPE = 'DP';
% SOLVE_TYPE = 'LEAST-SQUARES';
x0 = [10  ; 10]; % Initial state
fprintf('Prediction horizon : %i\n', N);
if strncmp(SOLVE_TYPE, 'DP', 1)
        fprintf('Computing controller via dynamic programming\n');
        % Compute via DP
        H = Q;
```

```
        for i = N:-1:1
                K = -inv(R+B'*H*B)*B'*H*A;
                H = Q + K'*R*K + (A+B*K)'*H*(A+B*K);
                % Store the future feedback laws so that we can later plot the prediction
                Kpred{i} = K;
        end
else
        fprintf('Computing controller via least-squares\n');
        % Compute via optimization
        QQ = kron(eye(N),Q);
        RR = kron(eye(N),R);
        AA = [[kron(zeros(1,N-1),zeros(n));kron(eye(N-1),A)] kron(ones(N,1),zeros(n))];
        AA = AA + -eye(size(AA,1));
        BB = kron(eye(N),B);
        CC = [-A;kron(ones(N-1,1),zeros(n))];
        FF = -inv(AA)*BB;
        GG =  inv(AA)*CC;
        KK = -inv(RR+FF'*QQ*FF)*FF'*QQ*GG;
        Kpred = KK;
        K = KK(1,:);
end
%% Simulate closed-loop dynamic
figure(1)
x(:,1) = x0;
i = 1;
while 1
        if norm(x(:,i)) > 3*norm(x0), fprintf('===> System unstable\n'); break; end
        if norm(x(:,i)) < 5e-2,       fprintf('===> System stable\n');   break; end
        u(:,i)   = K*x(:,i);
        x(:,i+1) = A*x(:,i) + B*u(:,i);
        %% Plot prediction
        x_pred   = compute_prediction(x(:,i),Kpred,A,B);
        clf;hold on;grid on;
        plot(x_pred(1,:), x_pred(2,:), '-o');
        plot(x(1,1:i) , x(2,1:i) , 'LineWidth',1.5);
        legend('Prediction','Closed Loop System');
        title('State Space');
        pause(0.1)
        i = i + 1;
end
y = C*x;
figure(2)
%% Final Plot
subplot(2,2,1);
plot(u,'-*');title('u');grid on;
subplot(2,2,3);
plot(y,'-*');title('y');grid on;
subplot(2,2,[2,4]);
plot(x(1,:),x(2,:),'-o');
title('x');xlabel('x1');ylabel('x2');grid on;
% Compare cost
[Klqr,~,~] = dlqr(A,B,Q,R);
costlqr = compute_cost(x0, A-B*Klqr, Klqr , Q, R, 1000);
costPn  = compute_cost(x0, A+B*K,    K    , Q, R, 1000);
fprintf('Cost of the optimal LQR controller : %.2f\n', costlqr);
fprintf('Cost of the N-step controller      : %.2f\n', costPn);
end
%% Simulate the closed-loop system and compute the cost
function cost = compute_cost(x0,Ak,K,Q,R,steps)
cost = 0;
x = x0;
for i=1:steps
        cost = cost + x'*(Q+K'*R*K)*x;
        x = Ak*x;
end
end
%% Compute the prediction
function x = compute_prediction(x0,K,A,B)
x(:,1) = x0;
if iscell(K) % We used Riccati
        for i = 1:length(K)
                x(:,i+1) = (A + B*K{i})*x(:,i);
        end
else % We used least-squares
```

```
        u = K*x0;
        for i = 1:length(u)
                x(:,i+1) = A*x(:,i) + B*u(i);
        end
end
end
```

(a) $N^\star = 7$, see code in the previous question.

(b) For short horizons, the balance between the cost of the state trajectory and the input trajectory comes down in favour of the input, since the state doesnt move too much over the short term if no input action is taken. Over a longer horizon, the cost of the state trajectory will dominate, causing the input to take action to decrease it.

Solution to Exercise 4.3

The LQR controller has lower cost. See code in the previous question.

Solution to Exercise 4.4

```
clear all; close all;clc;

A = [0.8 0;1 0.4];
B = eye(2);

Q = [1 0;0 10];
R = 20*eye(2);

%% METHOD 1. Calculate the gain by solving the Riccati equation

% STEP 1. Solve the Riccati equation
X = dare(A,B,Q,R);

% STEP 2. Calculate the optimal gain from the solution of Riccati equation
K1 = inv(B'*X*B+R)*B'*X*A

%% METHOD 2. Calculate the optimal gain directly using dlqr
K2 = dlqr(A,B,Q,R)
```

Solution to Exercise 4.5

Large values of ρ imply that we want to reduce the control effort. This have the following consequences: (1) State converges slower to the steady state; (2) The magnitude of the control signal is expected to be lower. Thus, solutions are the following:
States: ρ=2 – Figure 5 (A); ρ=10 – Figure 5 (C); ρ=100 – Figure 5 (B)
Control: ρ=2 – Figure 6 (C); ρ=10 – Figure 6 (A); ρ=100 – Figure 6 (B).

Solution to Exercise 4.6

```
clear all; close all;clc;

A = [1.0001 -0.0061; 0.0016 0.9978];
B = [-0.0001;-0.0104];
C = [3354 5.40];

% Define the weight matrices such as to achieve the desired rise time
```

```
Q = [1 0;0 1];
r  = 0.01;

% Design the LQR feedback gain
[L,S,e] = dlqr(A,B,Q,r)

% Design reference weight that ensures steady state gain 1
lr = inv(C*inv(eye(2,2)-A+B*L)*B);

%% Simulate the response
N=2000;

x=zeros(2,N); % Predefine states
y=zeros(1,N); % Predefine output
u=zeros(1,N); % Predefine control

for i=2:1:N
    % Calculate input u
    u(:,i-1) = -L*x(:,i-1)+lr*1;
    % Calculate output y
    y(i-1) = C*x(:,i-1);
    % Calculate state x
    x(:,i) = A*x(:,i-1)+B*u(:,i-1);
end

% Calculate input and output at time step N
u(:,N) = -L*x(:,N)+lr*1;
y(N)   = C*x(:,N);

%% Plot the output and the control to verify that the controller performs satisfactory

% Time scale needs to be multiplied by sampling time in order to get scale
% in seconds
sampling_time=10^(-5);
t=sampling_time*(0:1:N-1);

figure,
plot(t,y,'linewidth',1)
ylabel('Output')
xlabel('Time')

figure,
plot(t,u,'linewidth',1)
ylabel('Control')
xlabel('Time')
```

Solution to Exercise 4.7

(a) See code. After about 9 iterations we see that the matrix $P_{t+1} - P_t$ has elements on the order of 10^{-3}, compared to typical P_t values of around 5.

(b) The optimal cost, when started in state x_0, is $x_0^T P_0 x_0$. To maximize this we choose x_0 as an eigenvector of P_0 associated with its maximum eigenvalue. The eigenvector associated with the maximum eigenvalue is $x_0^{\max} = [0.5428, 0.7633, 0.3504]$.

```
% LQR for a triple accumulator
% data
A = [1 0 0; 1 1 0; 0 1 1];
B = [1 0 0]';
C = [0 0 1];
Q = C'*C; R = 1;
m=1; n=3; N=50;
% Riccati recursion
P = zeros(n,n,N+1);
P(:,:,N+1) = Q;
K = zeros(m,n,N);
```

```
%Nrm = zeros(N,1);
for i = N:-1:1
P(:,:,i) = Q+A'*P(:,:,i+1)*A-...
A'*P(:,:,i+1)*B*pinv(R+B'*P(:,:,i+1)*B)*B'*P(:,:,i+1)*A;
K(:,:,i) = -pinv(R+B'*P(:,:,i+1)*B)*B'*P(:,:,i+1)*A;
% Nrm(N+1-i) =norm(P(:,:,i+1)-P(:,:,i))/norm(P(:,:,i+1));
end
% worst initial condition (max_x(0) min_u J)
[V,D] = eig(P(:,:,1));
x0 = -V(:,1);
% optimal u and resulting x
x = zeros(3,N); x(:,1) = x0;
u = zeros(1,N); u(1) = K(:,:,1)*x(:,1);
for t = 1:N-1
x(:,t+1) = A*x(:,t) + B*u(t);
u(t+1) = K(:,:,t+1)*x(:,t+1);
end
% plots
figure(1);
t = 0:49; K = shiftdim(K);
subplot(3,1,1); plot(t,K(1,:)); ylabel('K1(t)');
subplot(3,1,2); plot(t,K(2,:)); ylabel('K2(t)');
subplot(3,1,3); plot(t,K(3,:)); ylabel('K3(t)'); xlabel('t');
%print -depsc tripleaccKt
figure(2);
subplot(4,1,1); plot(t,x(1,:)); ylabel('x1(t)');
subplot(4,1,2); plot(t,x(2,:)); ylabel('x2(t)');
subplot(4,1,3); plot(t,x(3,:)); ylabel('x3(t)');
subplot(4,1,4); plot(t,u); ylabel('u(t)'); xlabel('t');
%print -depsc tripleaccXU
%figure(3);
%semilogy(Nrm); title('Riccati convergence');
```

Solution to Exercise 4.8

(a)

```
% Define system matrices
A=[1 0; 2 3];
B=[1 0]';
C=[0 1];

% Define performance weights
Q=[1 0;0 1];
R=1;

% Design feedback gain
L = dlqr(A,B,Q,R);

% Adjust the gain of the reference such as
% to have gain from r to y equal to 1
lr=1/(C*inv(eye(2,2)-A+B*L)*B);
```

(b)

```
% Simulate the system with disturbance
N = 30;          % Set the running time of the simulation
x = zeros(2,N); % Predefine state vectors
u = zeros(1,N); % Predefine control inputs
y = zeros(1,N); % Predefine output signal
r=1;             % Set the reference signal to be equal to 1
d=1;             % Set the disturbance signal to be 1

%Simulate the response
for i=2:1:N
   x(:,i) = (A-B*L)*x(:,i-1) + B*lr*r+B*d;
   y(i)  = C*x(:,i);
   end

%Plot the response
figure,
plot(0:1:N-1,y)
```

(c)
```
%% With integral state

% Define system matrices with integral state
A2=[1 0 0; 2 3 0; -C 1];
B2=[1 0 0]';
C2=[0 1 0];
Br=[0 0 1]';

% Define performance weights
Q2=[1 0 0;0 1 0; 0 0 1];
R2=1;

% Design feedback gain
L2 = dlqr(A2,B2,Q2,R2);
```

(d)
```
% Simulate the system with disturbance
N = 30;          % Set the running time of the simulation
x = zeros(2,N); % Predefine state vectors
u = zeros(1,N); % Predefine control inputs
y = zeros(1,N); % Predefine output signal
r=1;             % Set the reference signal to be equal to 1
d=1;             % Set the disturbance signal to be 1

% Simulate the response
for i=2:1:N
   x(:,i) = (A2-B2*L2)*x(:,i-1) + Br*r+B2*d;
   y(i)  = C2*x(:,i);
end

% Plot the response
figure,
plot(0:1:N-1,y)
```

Solution to Exercise 4.9

(a)
```
clear all; close all; clc;

A  = [0.1 0; 0.2 0.3];
B  = [1 0;0 1];
Br = [1;0];
C  = [1 0;0 1];

% Disturbance model - sinusoidal disturbance
omega_0   = 2;
h         = 0.001;
AX        = [cos(2*pi*omega_0*h) sin(2*pi*omega_0*h);-sin(2*pi*omega_0*h) cos(2*pi*omega_0*h)];
CX        = [1 0];
Bd        = [1;0];
Dd        = 0;

% Extended system - the state is [x_t; X_t]

Ae  = [A Bd*CX; zeros(2,2) AX];
Be  = [B;0 0;0 0];
Bre = [Br;0;0];
Ce  = [C zeros(2,2)];

% Verify observability
rank([Ce;Ce*Ae;Ce*Ae*Ae])

%% Controller design

% STEP 1. Observer design

% Define the performance weights
Q_obs = 0.1*eye(4);
R_obs = eye(2,2);
```

```
% Remark - you can try some other weights

% Calculate the observer gain
K = dlqr(Ae',Ce',Q_obs,R_obs)';

%%
% STEP 2. LQR controller design

% Define performance weights
Q_cont = Ce'*Ce;
R_cont = 0.1*eye(2,2);

% Remark - you can try some other weights
Le = dlqr(Ae,Be,Q_cont,R_cont);

% Adjust the gain of the reference such as
% to have gain from r to y equal to 1
lr=1/(Ce(1,:)*inv(eye(4,4)-Ae+Be*Le)*Bre);

%% Simulate the system with disturbance
N = 1000;              % Set the running time of the simulation
xe = zeros(4,N);       % Predefine state vectors
xe_obs = zeros(4,N);   % Predefine state vectors
u = zeros(2,N);        % Predefine control inputs
ye = zeros(2,N);       % Predefine output signal
r=1;                   % Set the reference signal to be equal to 1
xe(:,1)=[0;0;0.1;0];   % Initial state of the system

%Simulate the response
for i=2:1:N
   % Measurement collected from the process
   ye(:,i-1)  = Ce*xe(:,i-1);
   % Control signal
   u(:,i-1)= -Le*xe_obs(:,i-1);
   % State observer
   xe_obs(:,i) = (Ae-K*Ce)*xe_obs(:,i-1) + K*ye(:,i-1)+Be*u(:,i-1)+Bre*lr*r;
   % State
   xe(:,i) = Ae*xe(:,i-1)+Be*u(:,i-1)+Bre*lr*r;
end

% Update the measurement from time step N
ye(:,i)  = Ce*xe(:,i);

%% Plot the first output to verify that controller is functioning properly
figure,
hold on;
ylim([0 1.5])
plot(0:1:N-1,ye(1,:),'linewidth',1)
plot(0:1:N-1,ones(1,N),'linewidth',1)
xlabel('Time step k')
ylabel('Output y_1')
legend('y_1','Reference')
```

Solution to Exercise 4.10

K_t is defined by $K_t = -(R+B^T P_{t+1} B)^{-1} B^T P_{t+1} A$. Therefore, if $P_1 = ... = P_N$, we have $K_0 = ... = K_{N-1}$. Since $Q_f = P_N$, and for t far from N we have that P_t converges to the steady-state solution, we see that to have constant P_t, we must have Q_f equal to the steady-state solution P_{ss}. With this choice, we have $P_t = P_{ss}$ for all t, and therefore $K_t = K_{ss}$ for all t.

Solution to Exercise 4.11

(a) From (14), it follows that

$$W_0(x_0) = R_0(x_0 - \mu_0)^2$$

Thus, by setting $S_0 = R_0$, $\hat{x}_{0|-1} = \mu_0$, $c_0 = 0$, we obtain the desired result.

(b) From (14), it follows that

$$\begin{aligned} W_{t+1}(x_{t+1}) &= \min_{x_t}\{W_t(x_t) + R_1 w_k^2 + R_2 v_k^2\} \\ &= \min_{x_t}\{W_t(x_t) + R_1(x_{t+1} - Ax_t)^2 + R_2(y_t - Cx_t)^2\}. \end{aligned}$$

We conclude

$$\begin{aligned} q_1(x_t, y_t) &= R_2(y_t - Cx_t)^2 \\ q_2(x_{t+1}, x_t) &= R_1(x_{t+1} - Ax_t)^2. \end{aligned}$$

(c) We have

$$W_t^+(x_t) = S_t(x_t - \hat{x}_{t|t-1})^2 + c_t + R_2(y_t - Cx_t)^2 \tag{37}$$

$$= S_t(x_t^2 - 2x_t\hat{x}_{t|t-1} + \hat{x}_{t|t-1}^2) + R_2(C^2x_t^2 - 2y_tCx_t + y_t^2) + c_t \tag{38}$$

$$= (S_t + R_2C^2)x_t^2 - 2(S_t\hat{x}_{t|t-1} + R_2y_tC)x_t + S_t\hat{x}_{t|t-1}^2 + R_2y_t^2 + c_t \tag{39}$$

$$= (S_t + R_2C^2)(x_t^2 - 2\frac{S_t\hat{x}_{t|t-1} + R_2Cy_t}{S_t + R_2C^2}x_t) + S_t\hat{x}_{t|t-1}^2 + R_2y_t^2 + c_t \tag{40}$$

Therefore, since $W_t^+(x_t)$ is quadratic in x_t, we can find the minimizer $\hat{x}_{t|t}$ by setting the first derivative to 0. The first derivative of $W_t^+(x_t)$ is

$$\begin{aligned} (W_t^+)'(x_t) &= 2(S_t + R_2C^2)x_t - 2(S_t + R_2C^2)\frac{S_t\hat{x}_{t|t-1} + R_2Cy_t}{S_t + R_2C^2} \\ &= 2(S_t + R_2C^2)x_t - 2(S_t\hat{x}_{t|t-1} + R_2Cy_t). \end{aligned}$$

Hence, from $(W_t^+)'(\hat{x}_{t|t}) = 0$ it follows that the minimizer is

$$\hat{x}_{t|t} = \frac{S_t\hat{x}_{t|t-1} + R_2Cy_t}{S_t + R_2C^2}.$$

(d) Let $S_t^+ = S_t + R_2C^2$. We can then rewrite (37) as

$$W_t^+(x_t) = S_t^+(x_t^2 - 2\hat{x}_{t|t}x_t) + S_t\hat{x}_{t|t-1}^2 + R_2y_t^2 + c_t$$

By adding and subtracting $S_t^+\hat{x}_{t|t}^2$ from the previous equation, we obtain

$$\begin{aligned} W_t^+(x_t) &= S_t^+(x_t^2 - 2\hat{x}_{t|t}x_t + \hat{x}_{t|t}^2) + S_t\hat{x}_{t|t-1}^2 + R_2y_t^2 + c_t - S_t^+\hat{x}_{t|t}^2 \\ &= S_t^+(x_t - \hat{x}_{t|t})^2 + c_t^+ \end{aligned}$$

where $c_t^+ = c_t + S_t\hat{x}_{t|t-1}^2 + R_2y_t^2 - S_t^+\hat{x}_{t|t}^2$.

(e) By plunging the expression for $W_t^+(x_t)$ into $W_{t+1}(x_{t+1})$, we obtain

$$\begin{aligned}
&W_{t+1}(x_{t+1}) = \min\{W_t^+(x_t) + R_2 v_k^2 + R_1(x_{t+1} - Ax_t)^2\} \\
&= \min\{S_t^+(x_t - \hat{x}_{t|t})^2 + c_t^+ + R_1(x_{t+1} - Ax_t)^2\} \\
&= \min\{(S_t^+ + R_1A^2)x_t^2 - 2(S_t^+\hat{x}_{t|t} + R_1Ax_{t+1})x_t + S_t^+\hat{x}_{t|t}^2 + R_1x_{t+1}^2 + c_t^+\}
\end{aligned}$$

Since the last expression is quadratic in x_t, we can find the minimizer $\hat{x}_t^*$ by setting the first derivative to 0, that is,

$$\begin{aligned}
&2(S_t^+ + R_1A^2)x_t^* - 2(S_t^+\hat{x}_{t|t} + R_1Ax_{t+1}) = 0 \\
&\Rightarrow x_t^* = \frac{S_t^+\hat{x}_{t|t} + R_1Ax_{t+1}}{S_t^+ + R_1A^2}.
\end{aligned}$$

(f) By plugging the expression for x_t^* into the previous equation, it follows

$$\begin{aligned}
&W_{t+1}(x_{t+1}) = (S_t^+ + R_1A^2)(x_t^*)^2 - 2(S_t^+\hat{x}_{t|t} + R_1Ax_{t+1})x_t^* \\
&\ldots + S_t^+\hat{x}_{t|t}^2 + R_1x_{t+1}^2 + c_t^+ \\
&= -\frac{(S_t^+\hat{x}_{t|t} + R_1Ax_{t+1})^2}{S_t^+ + R_1A^2} + S_t^+\hat{x}_{t|t}^2 + R_1x_{t+1}^2 + c_t^+ \\
&= (S_t^+ - \frac{(S_t^+)^2}{S_t^+ + R_1A^2})\hat{x}_{t|t}^2 - 2\frac{S_t^+R_1A}{S_t^+ + R_1A^2}\hat{x}_{t|t}x_{t+1} \\
&\ldots + (R_1 - \frac{(R_1A)^2}{S_t^+ + R_1A^2})x_{t+1}^2 + c_t^+ \\
&= \frac{S_t^+R_1A^2}{S_t^+ + R_1A^2}\hat{x}_{t|t}^2 - 2\frac{S_t^+R_1A}{S_t^+ + R_1A^2}\hat{x}_{t|t}x_{t+1} + \frac{R_1S_t^+}{S_t^+ + R_1A^2}x_{t+1}^2 + c_t^+ \\
&= \frac{S_t^+R_1}{S_t^+ + R_1A^2}(A^2\hat{x}_{t|t}^2 - 2A\hat{x}_{t|t}x_{t+1} + x_{t+1}^2) + c_t^+
\end{aligned}$$

By introducing $\hat{x}_{t+1|t} = A\hat{x}_{t|t}$, $S_{t+1} = \frac{S_t^+R_1}{S_t^+ + R_1A^2}$, and $c_{t+1} = c_t^+$ we obtain

$$W_{t+1}(x_{t+1}) = S_{t+1}(x_{t+1} - \hat{x}_{t+1|t})^2 + c_{t+1}$$

which is the desired form of the arrival cost. This concludes the proof.

(g) The algorithm can be summarized as follows

- *Initialization.* Set $S_0 = R_0$, $S_0 = 0$, $\hat{x}_{0|-1} = \mu_0$, $c_0 = 0$
- Repeat from $t = 0$ to $t = N$
- *Measurement update step.* Find $\hat{x}_{t|t} = \frac{S_t\hat{x}_{t|t-1} + R_2Cy_t}{S_t + R_2C^2}$
- Update $S_t^+ = S_t + R_2C^2$, $c_t^+ = c_t + S_t\hat{x}_{t|t-1}^2 + R_2y_t^2 - S_t^+\hat{x}_{t|t}^2$
- *Prediction step.*Find $\hat{x}_{t+1|t} = A\hat{x}_{t|t}$
- Update $S_{t+1} = \frac{S_t^+R_1}{S_t^+ + R_1A^2}$, $c_{t+1} = c_t^+$

(h) $\hat{x}_{t|t}$ represents the estimate of the state x_t based on the measurements $y_0, \ldots, y_t$. $\hat{x}_{t+1|t}$ represents the estimate of the state x_{t+1} based on the measurements $y_0, \ldots, y_t$, that is, the best prediction of the state x_{t+1} based on information up to time t.

Solution to Exercise 4.12

The closed-loop system $x_{t+1} = (A + BK_T)x_t$ is stable if the spectral radius of $A + BK_T$ is strictly less than 1. (The spectral radius of a square matrix P is $\max_{i=1,\ldots,n} |\lambda_i|$, where $\lambda_1, ..., \lambda_n$ are its eigenvalues.) In order to determine the smallest horizon T that gives a stable closed-loop system, we will find K_T for $T = 1, 2...$ and check whether the spectral radius of $A + BK_T$ is less than 1. We have $K_T = -(R + B^T P_T B)^{-1} B^T P_T A$, where $P_1 = Q, P_{i+1} = Q + A^T P_i - A^T P_i B(R + B_T PB)^{-1} B^T P_i A$. See the code for more detail.

```
A = [ 1 .4 0 0; -.6 1 .4 0;0 .4 1 -.6;0 0 .4 1];
B = [1 0 0 0]';
C = [0 0 0 1];
Q = C'*C;R = 1;
m=1;n=4;T=25;
% recursion
P=zeros(n,n,T+1);
K=zeros(m,n,T);
P(:,:,T+1)=Q;
spec_rad = zeros(1,T);
for i = T:-1:1
    % LQR-optimal state feedback
    K(:,:,i) = -inv(R + B'*P(:,:,i+1)*B)*B'*P(:,:,i+1)*A;
    % spectral radius
    spec_rad(T-i+1) = max(abs(eig(A+B*K(:,:,i))));
    P(:,:,i)=Q + A'*P(:,:,i+1)*A + A'*P(:,:,i+1)*B*K(:,:,i);
end
% plots
figure(1);
t = 0:T-1; K = shiftdim(K);
subplot(4,1,1); plot(t,K(1,:)); ylabel('K1(t)');
subplot(4,1,2); plot(t,K(2,:)); ylabel('K2(t)');
subplot(4,1,3); plot(t,K(3,:)); ylabel('K3(t)');
subplot(4,1,4); plot(t,K(4,:)); ylabel('K4(t)'); xlabel('t');
% print -depsc receding_horizon_gains.eps
figure(2)
plot(1:T,spec_rad); hold on; plot(1:T,spec_rad,'.')
xlabel('T'); ylabel('\rho(A+BK)')
```

Solution to Exercise 4.13

(a) The value function is

$$J = \min \Big(\sum_{\tau=0}^{N-1} \gamma^\tau (x_\tau^T Q x_\tau + u_\tau^T R u_\tau) + \gamma^N x_N^T Q_N x_N \Big),$$

where the minimum is over all input sequences $u(0), ..., u(N-1)$, and $x_t = z$, $x_{\tau+1} = Ax_\tau + Bu_\tau$ for $\tau = t, ..., N-1$.We can express this as

$$J = \min \Big(\sum_{\tau=0}^{N} (x_\tau^T Q_\tau x_\tau + u_\tau^T R_\tau u_\tau) + \gamma^N x_N^T Q_N x_N \Big),$$

where $Q_\tau = \gamma^\tau Q$ and $R_\tau = \gamma^\tau R$. This is just a standard LQR problem with time-varying Q and R. Thus from the lecture notes we have

$$V_t(z) = z^T P_t z, u_t^\star = K_t x_t,$$

where P_t and K_t are given by the following recursion

$$\begin{aligned}
P_N &= \gamma^N Q_N, \\
P_{t-1} &= Q_{t-1} + A^T P_t A - A^T P_t B(R_{t-1} + B^T P_t B)^{-1} B^T P_t A \\
&= \gamma^{t-1} Q + A^T P_t A - A^T P_t B(\gamma^{t-1} R + B^T P_t B)^{-1} B^T P_t A, \\
K_t &= -(R_t + B^T P_{t+1} B)^{-1} B^T P_{t+1} A \\
&= -(\gamma^t R_t + B^T P_{t+1} B)^{-1} B^T P_{t+1} A.
\end{aligned}$$

(b) Clearly we have $W_N(z) = z^T Q_N z$, since at the last step the cost is not a function of the input. Now let $u_t = w$. We can apply the dynamic programming principle to the equation for $W_t(z)$ by first minimizing with respect to w and then with respect to the other inputs. If we do that, it is easy to see that

$$W_t(z) = \min_w \big(z^T Q z + w^T R w + \gamma W_{t+1}(Az + Bw)\big).$$

Note that the minimizing w is equal to the optimal input $u_t^\star$. This is because $W_0(z) = V_0(z)$ for all z and therefore a set of minimizing inputs for the above HJ equation will also minimize J. We will now show by induction that $W_t(z) = z^T S_t z$, i.e. that $W_t(z)$ is a quadratic form in z. Let us assume that this is true for $t = \tau + 1$. Then using the above HJ equation we have

$$W_\tau(z) = \min_w \big(z^T Q z + w^T R w + \gamma(Az + Bw)^T S_{\tau+1}(Az + Bw)\big)$$

To find $w^\star$, we take the derivatives with respect to w of the argument within the minimum above, and set it equal to zero. We get

$$w^\star = -\gamma(R + \gamma B^T S_{\tau+1} B)^{-1} B^T S_{\tau+1} A z.$$

By substituting the optimal w in the expression for $W_t(z)$ we obtain

$$W_t(z) = z^T(Q + \gamma A^T S_{t+1} A - \gamma 2 A^T S_{t+1} B(R + \gamma B^T S_{t+1} B)^{-1} B^T S_{t+1} A) z,$$

which is still a quadratic form. This combined with the fact that $W_N(z) = z^T Q z$ is sufficient to see that $W_t(z)$ is indeed a quadratic form. Also, the optimum input $u_t^\star$ is given by

$$u_t^\star = -\gamma(R + \gamma B^T S_{\tau+1} B)^{-1} B^T S_{\tau+1} A x_t.$$

Actually S_t and P_t are closely related, since we have

$$\begin{aligned}
W_t(z) &= \min \big(\sum_{\tau=t}^{N-1} \gamma^{\tau-t}(x_\tau^T Q x_\tau + u_\tau^T R u_\tau) + x_N^T Q_N x_N \big). \\
&= \gamma^{-t} \min \big(\sum_{\tau=t}^{N-1} \gamma^{\tau}(x_\tau^T Q x_\tau + u_\tau^T R u_\tau) + x_N^T Q_N x_N \big). \\
&= \gamma^{-t} V_t(z)
\end{aligned}$$

Thus

$$z^T S_t z = \gamma^{-t} z^T P_t z, \quad \forall z,$$

or in other words $St = \gamma^{-t} P_t$. Substituting this into the formula for $u_t^\star$ we get

$$\begin{aligned} u_t^\star &= -(R + B^T(\gamma^{-t}P_{t+1})B)^{-1}B^T A(\gamma^{-t}P_{t+1})x_t \\ &= -(\gamma^t R + B^T P_{t+1}B)^{-1}B^T A(P_{t+1})x_t \end{aligned}$$

Therefore this method gives the same formula for $u_t^\star$ as the one of part 1..

(c) We want to show that $y_t = \gamma^{t/2}x_t$, provided that $z_t = \gamma^{t/2}u_t$. This can be shown by induction. The argument holds for $t = 0$, since $y_0 = x_0$. Now suppose that $y_t = \gamma^{t/2}x_t$. We have

$$\begin{aligned} y_{t+1} &= \gamma^{1/2}Ay_t + \gamma^{1/2}Bz_t \\ &= \gamma^{(t+1)/2}Ax_t + \gamma^{(t+1)/2}Bu_t \\ &= \gamma^{(t+1)/2}(Ax_t + Bu_t) \\ &= \gamma^{(t+1)/2}x_{t+1} \end{aligned}$$

Thus the argument also holds for y_{t+1}. Now, let's express the cost function in terms of y_t and z_t. We have

$$\begin{aligned} &\sum_{\tau=0}^{N-1} \gamma^\tau (x_\tau^T Q x_\tau + u_\tau^T R u_\tau) + \gamma^N x_N^T Q_N x_N \\ &= \sum_{\tau=0}^{N-1} ((\gamma^{\tau/2}x_\tau^T)Q(\gamma^{\tau/2}x_\tau) + (\gamma^{\tau/2}u_\tau^T)R(\gamma^{\tau/2}u_\tau^T)) + (\gamma^{N/2}x_N^T)Q_N(\gamma^{N/2}x_N) \\ &= \sum_{\tau=0}^{N} (y_\tau^T Q y_\tau + z_\tau^T R z_\tau) + y_N^T Q_N y_N. \end{aligned}$$

Using the formulas from the lecture notes we get that the minimizing z_t for the above expression is given by

$$z_t^\star = -\gamma(R + \gamma B^T \tilde{P}_t + 1B)^{-1}B^T \tilde{P}_{t+1} A y_t,$$

where $\tilde{P}_t$ is found by the following recursion

$$\begin{aligned} \tilde{P}_N &= Q_N, \\ \tilde{P}_{t-1} &= Q + \gamma A^T \tilde{P}_t A - \gamma 2 A^T \tilde{P}_t B(R + \gamma B^T \tilde{P}_t B)^{-1} B^T \tilde{P}_t A. \end{aligned}$$

Note that since the original problem and this modified problem have the same cost function, the optimum $z_t^\star$ will give us the optimum $u_t^\star$ of the original problem, using the relation $z_t^\star = \gamma^{t/2}u_t^\star$. We can also show by an inductive argument that $P_t = \gamma^t \tilde{P}_t$. Combining these two relations in the formula for $z_t^\star$, we get

$$\begin{aligned} \gamma^{t/2}u_t^\star &= -\gamma^{-t}(R + \gamma^{-t}B^T P_{t+1}B)^{-1}B^T P_{t+1}Ay_t \\ &= -\gamma^{t/2}(\gamma^t R + B^T P_{t+1}B)^{-1}B^T P_{t+1}Ax_t \end{aligned}$$

We therefore get

$$u_t^\star = -(\gamma^t R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A x_t$$

which is the same expression as the one obtained in parts 1. and 2..

Solution to Exercise 4.14

(a) First, we note that

$$J_{N,N}(N) = x_N^T (Q_1 + L^T Q_2 L) x_N := x_N^T V_{N,N} x_N$$

Now, assume that $J_{N,t}(x_t) = x_t^T V_{N,t} x_t$. Then,

$$\begin{aligned} J_{N,t-1}(x_{t-1}) &= \sum_{k=t-1}^{N} x_k^T Q_1 x_k + u_k^T Q_2 u_k = x_{t-1}^T Q_1 x_{t-1} + u_{t-1}^T Q_2 u_{t-1} + x_t^T V_{N,t} x_t = \\ &= x_{t-1}^T \left(Q_1 + L^T Q_2 L + (A - BL)^T V_{N,t} (A - BL) \right) x_{t-1} := x_{t-1}^T V_{N,t-1} x_{t-1} \end{aligned}$$

We have thus derived the recursion

$$V_{N,t-1} = Q_1 + L^T Q_2 L + (A - BL)^T V_{N,t} (A - BL)$$

(b) When the iterations converge, we get the equality

$$V_\infty = Q_1 + L^T Q_2 L + (A - BL)^T V_\infty (A - BL)$$

which we can re-write as

$$(A - BL)^T V_\infty (A - BL) - V_\infty + (Q_1 + L^T Q_2 L) = 0$$

We notice that this equation has the structure of a Lyapunov equation

$$\Phi^T P \Phi - P + Q = 0 \tag{41}$$

with $\Phi = A - BL$ and $Q = Q_1 + L^T Q_2 L$. So, we can compute V_∞ by forming Φ and Q and calling the routine for solving (41).

Solution to Exercise 4.15

(a) We first notice that $V_0(x_0)$ has the desired form with $S_0 = C^T C$. In fact, it is also straight forward to verify that S_t has the desired form for any $t \geq 0$ since

$$\begin{aligned} V_t(x_0) &= \sum_{k=0}^{t} y_k^T y_k = \sum_{k=0}^{t} x_k^T C^T C x_k = \\ &= x_0^T \left(\sum_{k=0}^{t} (A^k)^T C^T C A^k \right) x_0 \end{aligned}$$

Thus,

$$S_t = \sum_{k=0}^{t} (A^k)^T C^T C A^k$$

Therefore,

$$\begin{aligned} S_{t+1} &= \sum_{k=0}^{t+1} (A^k)^T C^T C A^k = C^T C + \sum_{k=1}^{t+1} (A^k)^T C^T C A^k = \\ &= C^T C + A^T \left(\sum_{k=0}^{t} (A^k)^T C^T C A^k \right) A = \\ &= C^T C + A^T S_t A \end{aligned}$$

(b) If the iteration converges, we have $S_t = S$ for all t. Hence,

$$S = A^T S A + C^T C$$

We recognize that this expression has the form of a Lyapunov equation $A^T P A - P = -Q$, for $P = S$ and $Q = C^T C$.

(c) When $a = 0$, the two system states are decoupled and do not affect each other. At the same time, we can only observe the first state in the output, so we should expect S to have the form

$$S = \begin{bmatrix} s_{11} & 0 \\ 0 & 0 \end{bmatrix}$$

(d) When $a = 0.115$, we parameterize the unknown S as

$$S = \begin{bmatrix} s_{11} & s_{12} \\ s_{12} & s_{22} \end{bmatrix}$$

and attempt to solve the matrix equation $S - A^T S A = C^T C$:

$$\begin{bmatrix} 0.36 s_{11} & 0.92(s_{12} - 0.1 s_{11}) \\ 0.92(s_{12} - 0.1 s_{11}) & 0.99 s_{22} - 0.115(0.115 s_{11} + 0.2 s_{12}) \end{bmatrix} = \begin{bmatrix} 0.36 & 0 \\ 0 & 0 \end{bmatrix}$$

From which we find

$$\begin{aligned} s_{11} &= 1 \\ s_{12} &= 0.1 \\ s_{22} &= 0.015525/0.99 \approx 0.0157 \end{aligned}$$

Hence,

$$V_t(x_0) = x_0^T \begin{bmatrix} 1 & 0.1 \\ 0.1 & 0.157 \end{bmatrix} x_0$$

from which we can see that the second state still generates quite limited output energy, and hence should intuitively be harder to estimate than the first state (at least in the presence of disturbances and measurement noise).

Solution to Exercise 4.16

(a) This is a scalar LQR problem, where the optimal feedback is given by

$$\hat{u}_k = -\frac{3/2p}{1+(3/2)^2 p}x_k = -\frac{6p}{4+9p}x_k$$

where p solves the scalar riccati equation

$$p = 1 + p - \frac{(3/2)^2 p^2}{1+(3/2)^2 p} = 1 + p - \frac{9p^2}{4+9p}$$

$$\Leftrightarrow 9p^2 - 9p - 4 = 0$$

with solutions

$$p = \frac{1}{2} \overset{(+)}{(-)} \frac{5}{6} = \frac{4}{3}$$

so that

$$\hat{u}_k = -\frac{1}{2}x_k$$

(b) The dynamic programming recursion is given by

$$\begin{aligned} V_k(x_k) &= \min_{|u_k|\leq 1} \left\{ x_k^2 + u_k^2 + p_{k+1}x_{k+1}^2 \right\} \\ &= \min_{|u_k|\leq 1} \left\{ x_k^2 + u_k^2 + p_{k+1}\left(x_k + \frac{3}{2}u_k\right)^2 \right\} \end{aligned}$$

Without constraints, the optimal solution is given by

$$\hat{u}_k = -\frac{6p_k}{4+9p_k}x_k$$

where p_k satisfies the recursion

$$p_k = 1 + p_{k+1} - \frac{9p_{k+1}^2}{4+9p_{k+1}}$$

If the optimal feedback is attainable, i.e. within the constraints on u_k, then

$$\left| \frac{6p_k}{4+9p_k}x_k \right| \leq 1$$

so that

$$|x_k| \leq \frac{4+9p_k}{6p_k} = \frac{3}{2} + \frac{2}{3p_k}$$

When x_k is outside this region, the optimum is achieved at either constraint bound. Due to the quadratic expressions it is straightforward to see that

$$x_k + \frac{3}{2}u_k$$

governs the optimal choice of u_k. If $x_k > \frac{3}{2} + \frac{2}{3p_k}$ then the optimal decision will be to counteract x_k by choosing $u_k = -1$. Likewise, if $x_k < -\frac{3}{2} - \frac{2}{3p_k}$

then $u_k = 1$ is the best choice. Therefore, the optimal feedback law at stage k is given by

$$\hat{u}_k = \begin{cases} 1, & x_k < -\frac{3}{2} - \frac{2}{3p_k} \\ -\frac{6p_k}{4+9p_k} x_k, & -\frac{3}{2} - \frac{2}{3p_k} \leq x_k \leq \frac{3}{2} + \frac{2}{3p_k} \\ -1, & x_k > \frac{3}{2} + \frac{2}{3p_k} \end{cases}$$

where p_k satisfies

$$p_k = 1 + p_{k+1} - \frac{9p_{k+1}^2}{4 + 9p_{k+1}}$$

and $p_N = 1$. Moreover, when $N \to \infty$, p_k converges to the stationary solution of the riccati recursion. This solution was found in (a) as $p_k = \frac{4}{3}$. In conclusion, the optimal feedback law to the constrained LQR problem is

$$\hat{u}_k = \begin{cases} 1, & x_k < -2 \\ -\frac{1}{2} x_k, & -2 \leq x_k \leq 2 \\ -1, & x_k > 2 \end{cases}$$

(c) Three cases have to be covered:

i) $-2 \leq x_0 \leq 2$

ii) $x_0 < -2$

iii) $x_0 > 2$

i) First, if $-2 \leq x_0 \leq 2$, then the dynamics under the optimal feedback law from (b) are given by

$$x_{k+1} = x_k + \frac{3}{2}\left(-\frac{1}{2} x_k\right) = \frac{1}{4} x_k$$

which is a stable closed loop system (alternatively, argue that LQR feedback laws yield a stable system under the given conditions). Note, that the subset $\mathcal{X} = \{x_k \mid -2 \leq x_k \leq 2\}$ is invariant under the feedback law, so that $x_k \in \mathcal{X} \Rightarrow x_{k+1} \in \mathcal{X}$, and hence any starting point $x_0 \in \mathcal{X}$ will evolve to zero without ever leaving $\mathcal{X}$.

ii) The dynamics are now given by

$$x_{k+1} = x_k + \frac{3}{2}$$

so that x_k is increasing. Note, that $x_{k+1} - x_k = \frac{3}{2} < 4$, so that if $x_k < -2$ then $x_{k+1} < 2$. Hence, for any $x_0 < -2$ there is some k such that $-2 \leq x_k \leq 2$, which is the situation of case i). Thus, any $x_0 < -2$ will also evolve to zero.

iii) Likewise, if $x_0 > 2$ then

$$x_{k+1} = x_k - \frac{3}{2}$$

so that x_k is always decreasing and there is some k such that $-2 \leq x_k \leq 2$, which again yields a stable system. In other words, any $x_0 > 2$ will evolve to zero.

In conclusion, the optimal feedback law determined in (b) is stabilizing. □

Solution to Exercise 4.17

(a) Let x_t be the state that corresponds to an input u_t and an initial condition x_0.
The state at time t x_t has a linear dependence on $\begin{bmatrix} x_0, u_0, u_1, \ldots, u_{t-1} \end{bmatrix}^T$.

$$x_t = A^t x_0 + \sum_{i=1}^{t} A^{i-1} B u_{t-i}.$$

If we now scale the input u_t and initial condition x_0 with an constant λ, then the state at time t will be scaled accordingly

$$A^t \lambda x_0 + \sum_{i=1}^{t} A^{i-1} B(\lambda u_{t-i}) = \lambda x_t.$$

This leads to

$$J(\lambda u, \lambda x_0) = \sum_{t=0}^{\infty} (\lambda^2 x_t^T Q_1 x_t + \lambda^2 u_t^T Q_2 u_t) = \lambda^2 J(u, x_0).$$

(b) Suppose

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t \\ \tilde{x}_{t+1} &= A\tilde{x}_t + B\tilde{u}_t \end{aligned}$$

Adding or subtracting the above equations yields

$$x_{t+1} \pm \tilde{x}_{t+1} = A(x_t \pm \tilde{x}_t) + B(u_t \pm \tilde{u}_t)$$

Therefore, $x_t \pm \tilde{x}_t$ is the state that corresponds to an input $u_t \pm \tilde{u}_t$ and initial condition $x_t \pm \tilde{x}_t$ respectively. Now

$$\begin{aligned} &J(u+\tilde{u}, x_0 + \tilde{x}_0) + J(u - \tilde{u}, x_0 - \tilde{x}_0) \\ &\quad = \sum_{i=0}^{\infty} ((x_t + \tilde{x}_t)^T Q_1 (x_t + \tilde{x}_t) + (x_t - \tilde{x}_t)^T Q_1 (x_t - \tilde{x}_t) \\ &\qquad + (u_t + \tilde{u}_t)^T Q_2 (u_t + \tilde{u}_t) + (u_t - \tilde{u}_t)^T Q_2 (u_t - \tilde{u}_t)) \\ &\quad = \sum_{t=0}^{\infty} 2x_t^T Q_1 x_t + 2\tilde{x}_t^T Q_1 \tilde{x}_t + 2u_t^T Q_2 u_t + 2\tilde{u}_t^T Q_2 \tilde{u}_t \\ &\quad = 2J(u, x_0) + 2J(\tilde{u}, \tilde{x}_0) \end{aligned}$$

By minimizing both sides with respect to u and $\tilde{u}$ we obtain

$$\min_{u,\tilde{u}} \{J(u+\tilde{u}, x_0+\tilde{x}_0) + J(u-\tilde{u}, x_0-\tilde{x}_0)\} = \min_{u} 2J(u, x_0) + \min_{\tilde{u}} 2J(\tilde{u}, \tilde{x}_0)$$

The right-hand side of this equation is obviously $2V(x_0) + 2V(\tilde{x_0})$. Examining the left-hand side,

$$\min_{u,\tilde{u}}\{J(u+\tilde{u}, x_0+\tilde{x}_0)+J(u-\tilde{u}, x_0-\tilde{x}_0)\} \geq \min_{u,\tilde{u}} J(u+\tilde{u}, x_0+\tilde{x}_0)+\min_{u,\tilde{u}} J(u-\tilde{u}, x_0-\tilde{x}_0)$$

Leads to

$$V(x_0 + \tilde{x_0}) + V(x_0 - \tilde{x_0}) \leq 2V(x_0) + 2V(\tilde{x_0})$$

(c) Performing this substitution yields

$$V(x_0)+V(\tilde{x}_0) \leq 2V(\frac{x_0 + \tilde{x}_0}{2})+2V(\frac{x_0 - \tilde{x}_0}{2}) = \frac{1}{2}V(x_0 + \tilde{x}_0)+\frac{1}{2}V(x_0 - \tilde{x}_0)$$

where the result from (a) was used in the last step. The conclusion must be that

$$2V(x_0) + 2V(\tilde{x}_0) = V(x_0 + \tilde{x}_0) + V(x_0 - \tilde{x}_0)$$

(d) Taking derivatives w.r.t. the inner variables result in

$$\begin{aligned}
\nabla_{x_0} V(x_0 + \tilde{x}_0) &= \nabla V(x_0 + \tilde{x}_0) \\
\nabla_{x_0} V(x_0 - \tilde{x}_0) &= \nabla V(x_0 - \tilde{x}_0) \\
\nabla_{\tilde{x}_0} V(x_0 + \tilde{x}_0) &= \nabla V(x_0 + \tilde{x}_0) \\
\nabla_{\tilde{x}_0} V(x_0 - \tilde{x}_0) &= -\nabla V(x_0 - \tilde{x}_0)
\end{aligned}$$

Adding the derivatives as suggested in the hint now results in

$$\begin{aligned}
&\nabla_{x_0} V(x_0 + \tilde{x}_0) + \nabla_{x_0} V(x_0 - \tilde{x}_0) + \nabla_{\tilde{x}_0} V(x_0 + \tilde{x}_0) + \nabla_{\tilde{x}_0} V(x_0 - \tilde{x}_0) \\
&\qquad = 2\nabla V(x_0 + \tilde{x}_0)
\end{aligned}$$

(e) Taking gradients of both sides of equation (16), we obtain

$$\lambda \nabla V(\lambda x_0) = \lambda^2 \nabla V(x_0) \implies \nabla V(\lambda x_0) = \lambda \nabla V(x_0)$$

therefore, $\nabla V(x_0)$ is linear in x_0, and as such $\exists M \in \mathbb{R}^{n \times n}$ such that $\nabla V(x_0) = M x_0$

(f) Substituting λ in (a), we get

$$V(0) = V(0 \cdot x_0) = 0V(x_0) = 0$$

And thus

$$V(x_0) = \int_0^1 \nabla V(\theta x_0)^T x_0 \, d(\theta) = \int \nabla V(x_0)^T x_0 \theta \, d\theta = \frac{1}{2} x_0^T M^T x_0$$

Therefore we can write $V(x_0)$ as

$$V(x_0) = \frac{x_0^T M^T x_0}{2} = \frac{x_0^T M x_0}{2} = \frac{1}{4}\left(x_0^T M x_0 + x_0^T M^T x_0\right) = \frac{1}{4} x_0^T \left(M + M^T\right) x_0$$

Therefore P must be symmetric. Also,

$$\min_u J(u, x_0) \geq 0 \implies V(x_0) = x_0^T P x_0 \geq 0 \implies P \succ 0$$

means that P is also positive definite. This concludes our proof.

Solution to Exercise 4.18

(a) From the definition of the loss function,

$$V_N = (x_N - x_N^{\text{ref}})^T Q_1 (x_N - x_N^{\text{ref}}) = x_N^T Q_1 x_N - 2(x_N^{\text{ref}})^T Q_1 x + (x_N^{\text{ref}})^T Q_1 (x_N - x_N^{\text{ref}})$$

which has the desired form with $P_N = Q_1$, $q_N = Q_1 x_N^{\text{ref}}$ and $r_N = (x_N^{\text{ref}})^T Q_1 x_N^{\text{ref}}$.

(b) For an arbitrary stage t, we have

$$V_t = \min_{u_t} \left\{ (x_t - x_t^{\text{ref}})^T Q_1 (x_t - x_t^{\text{ref}}) + u_t^T Q_2 u_t + V_{t+1}(Ax_t + Bu_t) \right\}.$$

Using the induction hypotehsis,

$$\begin{aligned} V_t = \min_{u_t} \big\{ & (x_t - x_t^{\text{ref}})^T Q_1 (x_t - x_t^{\text{ref}}) + u_t^T Q_2 u_t + \\ & + (Ax_t + Bu_t)^T P_{t+1} (Ax_t + Bu_t) + 2q_{t+1}^T (Ax_t + Bu_t) + r_{t+1} \big\} = \\ = \min_{u_t} \big\{ & u_t^T (Q_2 + B^T P_{t+1} B) u_t + 2(B^T q_{t+1} + B^T P_{t+1} A x_t)^T u_t + \\ & + x_t^T A^T P_{t+1} A x_t + 2q_{t+1}^T A x_t + (x_t - x_t^{\text{ref}})^T Q_1 (x_t - x_t^{\text{ref}}) + r_{t+1} \big\} \end{aligned}$$

Now, we can use the completion-of-squares lemma in the notes to deduce that

$$u_t^\star = -(Q_2 + B^T P_{t+1} B)^{-1} (B^T P_{t+1} A x_t + B^T q_{t+1}) := -L_t x_t + m_t$$

and

$$\begin{aligned} V_t = & \, x_t^T A^T P_{t+1} A x_t + 2q_{t+1}^T A x_t + (x_t - x_t^{\text{ref}})^T Q_1 (x_t - x_t^{\text{ref}}) + r_{t+1} \\ & - (B^T P_{t+1} A x_t + B^T q_{t+1})^T (Q_2 + A^T P_{t+1} A)^{-1} (B^T P_{t+1} A x_t + B^T q_{t+1}) \end{aligned}$$

Since V_t is assumed to be on the form

$$V_t = x_t^T P_t x_t + 2q_t^T x_t + r_t$$

we can immediately identify the relationships

$$\begin{aligned} P_t &= Q_1 + A^T P_{t+1} A - A^T P_{t+1} B (A^T P_{t+1} A + Q_2)^{-1} B^T P_{t+1} A \\ q_t^T &= q_{t+1}^T (A + B(Q_2 + A^T P A)^{-1} B^T P_{t+1} A) - (x_t^{\text{ref}})^T Q_1 \\ r_t &= r_{t+1} + (x_t^{\text{ref}})^T Q_1 x_t^{\text{ref}} - q_{t+1}^T B (Q_2 + A^T P_{t+1} A)^{-1} B^T q_{t+1} \end{aligned}$$

(c) Combining these expressions with the derived values of L_t and m_t above yields the desired result.

5 Model predictive control

Solution to Exercise 5.1

The cost-function matrices in ii) has a higher weight on the input signal. This corresponds to the responses in A), which has a smaller input signal and slower convergence. The cost-function matrices in i) and iii) have higher weights on the second and first state respectively. The response in C) clearly has a slower x_1 and a faster x_2 than B). The pairings should be: A:ii, B:iii, C:i.

Solution to Exercise 5.2

(a) Stack all the decision variables

$$z = \begin{bmatrix} u(t) & x(t+1) & u(t+1) & \dots & x(T) \end{bmatrix}^T \in \mathbb{R}^{T(n+m)}$$

We then have that

$$H = \begin{bmatrix} R & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & Q & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & R & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & Q & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & R & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & Q_f \end{bmatrix} \in \mathbb{R}^{T(n+m) \times T(n+m)}$$

$$P = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ -1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & -1 \end{bmatrix} \in \mathbb{R}^{2T(m+n) \times T(m+n)} \quad h = \begin{bmatrix} u_{1,max} \\ -u_{1,min} \\ \vdots \\ -x_{n,min} \\ u_{1,max} \\ \vdots \\ -x_{n,min} \end{bmatrix} \in \mathbb{R}^{2T(m+n)}$$

$$C = \begin{bmatrix} -B & I & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & -A & -B & I & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & -A & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & I & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & -A & -B & I \end{bmatrix} \in \mathbb{R}^{Tn \times T(n+m)} \quad b = \begin{bmatrix} Ax(t) \\ \bar{0} \\ \vdots \\ \bar{0} \\ \bar{0} \end{bmatrix} \in \mathbb{R}^{Tn}$$

(b) When $T = 10$, the dimensions are

$$z \in \mathbb{R}^{30}, \quad H \in \mathbb{R}^{30 \times 30}, \quad P \in \mathbb{R}^{60 \times 30}, \quad C \in \mathbb{R}^{20 \times 30}$$

Solution to Exercise 5.3

(a) We limit the rate by limiting the change of the input between two time instances

$$-r_{lim} \leq u(t+i+1) - u(t+i) \leq r_{lim}, \quad i = 0, 1, \ldots, N-1.$$

This can be expressed as

$$\begin{array}{rcll} u(t+i+1) - u(t+i) & \leq & r_{lim} & \\ -u(t+i+1) + u(t+i) & \leq & r_{lim} & \end{array} \qquad i = 0, 1, \ldots, N-1.$$

Which means that we need to add the following lines to the inequality constraint of the optimization problem

$$P' = \begin{bmatrix} -1 & 0 & 1 & 0 & 0 & \ldots & 0 \\ 1 & 0 & -1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 0 & -1 & \ldots & 0 \\ & & & & & \ddots & \vdots \\ 0 & 0 & -1 & 0 & 1 & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ldots & \vdots \end{bmatrix}, \quad h' = \begin{bmatrix} r_{lim} \\ \vdots \\ r_{lim} \end{bmatrix}$$

Solution to Exercise 5.4

(a) Given the cost function in the question we identify that $Q = C^T C,\ R = 1$. The cost over mode 2 is

$$\sum_{i=N}^{\infty} [y_{t+i|t}^2 + u_{t+i|t}^2] = \sum_{i=N}^{\infty} [x_{t+i|t}^T (C^T C + K^T K) x_{t+i|t}] = x_{t+N|t}^T Q_f x_{t+N|t}$$

where matrix Q_f satisfies the discrete time Lyapunov equation

$$Q_f - (A+BK)^T Q_f (A+BK) = C^T C + K^T K.$$

For the given (A, B, C), K, and Q_f we get

$$A + BK = \begin{bmatrix} 0 & 0 \\ 2 & 0 \end{bmatrix}, \quad C^T C + K^T K = \begin{bmatrix} 5 & -1 \\ -1 & 2 \end{bmatrix},$$
$$(A+BK)^T Q_f (A+BK) = \begin{bmatrix} 8 & 0 \\ 0 & 0 \end{bmatrix}$$

and hence $Q_f = C^T C + K^T K + (A+BK)^T Q_f (A+BK)$, as required.

(b) Over the mode 2 prediction horizon, the inputs can be expressed as

$$u_{t+N+i|t} = K(A+BK)^i x_{t+N|t}, \quad i = 0, 1, \ldots$$

For this system feedback gain K we get $(A+BK)^i = 0$ for all $i \geq 2$, and hence $K(A+BK)^i x = 0$ for all $i \geq 2$. Therefore, the constraints need only be invoked over the mode 1 and at prediction time N and $N+1$ in order to ensure that they are satisfied at every instant on an infinite prediction horizon.

(c) If the optimization is feasible at time k, then, since predictions at k satisfy constraints at all future times, the input sequence defined by $u(k+1+i|k+1) = u^*(k+1+i|k)$ for $i = 0, 1, \ldots$ must satisfy the constraints in the optimization at time $k+1$. This sequence gives a (suboptimal) cost of $J^*(k) - (y^2(k) + u^2(k))$, and after optimization at time $k+1$ we therefore get $J^*(k+1) \leq J^*(k) - (y^2(k) + u^2(k))$.
Summing this inequality for $k = 0, 1, \ldots$ gives

$$\sum_{k=0}^{\infty} \left(y_k^2 + u_k^2\right) \leq J^*(0) - \lim_{k\to\infty} J^*(k) \leq J^*(0).$$

(d) The bound on $J^*(k+1) - J^*(k)$ in part (d) implies that the closed loop system is stable (i.e. $x = 0$ is locally asymptotically stable) if the pair (A, C) is observable, which is the case here since $CA = [-2 \quad 2]$ is not parallel with $C = [1 \quad 1]$. Thus, the observability matrix is full rank.

Solution to Exercise 5.5

By inspecting the matrices, we see that system dynamics of (i) and (ii) are stable (eigenvalues 1 and 1), while systems (iii) and (iv) are unstable (eigenvalues 1 and 1.02). We also note that system (i) and (iii) are observable ($\text{rank}(\mathcal{O}(A, Q^{1/2})) = 2$) whilst the two remaining systems are unobservable ($\text{rank}(\mathcal{O}(A, Q^{1/2})) = 1$).

(i)	Stable	Observable
(ii)	Stable	Unbservable
(iii)	Unstable	Observable
(iv)	Unstable	Unobservable

Looking at the responses, only one system is closed-loop unstable (C). This must belong to system (iv) since the controller could not stabilize the system. One of the responses is stable but not asymptotically (B). This corresponds to system (ii).

The remaining systems are similar. One difference is that in (i), state 2 has a negative effect on state 1, whereas in (iii) state 2 has a positive effect on state 1. This means that, from the given initial values, we would expect (iii) to be able to converge faster. Thus (iii) belongs to (A) and (i) to (D).

Solution to Exercise 5.6

(a) Terminal constraints ensure that predictions satisfy the system input and state constraints over the infinite horizon at prediction times $k = N, N+1, \ldots)$. This provides a recursive guarantee of feasibility by ensuring that, at each sampling instant k, the tail of the predicted input and state sequences that solve the constrained MPC optimization at time k will satisfy the constraints of the MPC optimization at time $k+1$. This makes it possible to derive a guarantee that the optimal predicted cost decreases with k, and hence a guarantee of closed loop stability.
A terminal constraint set S must be:

- invariant under the feedback law $u(k) = Kx(k)$ from N to ∞, i.e., $x_k \in S$ implies $x_{k+1} \in S$, for all $k \geq N$.

- feasible, i.e., state constraints are instantaneously satisfied and input constraints are satisfied by the feedback law $u(k) = Kx(k)$ from N to ∞, for all $x \in S$.

(b) (i) It is easy to verify invariance here because $A + BK$ is diagonal. Specifically, $S = x : |x_1 + x_2| \leq 1, |x_1 - x_2| \leq 1$ is invariant under $x(k+1) = (A + BK)x(k)$ because $A + BK = \text{diag}(0.5, -0.3)$ so the vertices of $S : v = [\pm 1 \ 0]^T, [0 \ \pm 1]^T$ satisfy $(A + BK)v = [\pm 0.5 \ 0]^T$, $[0 \ \mp 0.3]^T \in S$. Furthermore, it is obvious that S is feasible with respect to the state constraints.

(ii) The maximal terminal set is obtained by checking the system constraints over a sufficiently long constraint checking horizon N_c, i.e. for some finite N_c:

$$S_{\max} = x : (A + BK)^i x \in S, \ i = 0, \ 1, \ ..., \ N_c - 1, \tag{42}$$

where $S = x : |x_1 + x_2| \leq 1, |x_1 - x_2| \leq 1$ is the constraint set for the system state. The minimum allowable value for N_c can be found by checking whether $(A + BK)^{M+1} x \in S$ for all x such that $(A + BK)^i x \in S$ for $i = 0, \ 1, \ ..., \ M$. If this is true, then $N_c = M$, otherwise $N_c > M$. For given M this condition can be checked by solving a pair of linear programs:

$$z = \max \left\{ \max_x [1 \ 1](A + BK)^{M+1} x, \max_x [1 \ -1](A + BK)^{M+1} x \right\} \tag{43}$$

where each of the two maximizations over x is performed subject to $(A + BK)^i x \in S$, $i = 0, \ ..., \ M$, and then checking whether $z < 1$.

(c) Increasing the MPC prediction horizon N results in:

- improved performance, since the optimal value of predicted cost is non-increasing with increasing N (but note that, for each initial condition $x(0)$, there exists a finite value, N_∞, such that no improvement in closed loop performance can be obtained with $N > N_\infty$);
- larger operating region, since the set of feasible initial conditions is non-decreasing with increasing N;
- higher computational load, since the number of optimization variables increases.

Hence choosing N involves a trade-off between performance and computation.

Solution to Exercise 5.7

The constraint must be fulfilled at all time instances k, so we have:

$$-1 \leq y(k+1) \leq 2$$

but:

$$y(k+1) = 3x(k+1) = 6x(k) + 3u(k)$$

Rewriting $u(k)$ as $u(k-1) + \Delta u(k)$ gives:

$$y(k+1) = 6x(k) + 3u(k-1) + 3\Delta u(k)$$

Substituting this into the constraint gives:

$$-1 \leq 6x(k) + 3u(k-1) + 3\Delta u(k) \leq 2$$

Using the fact that $x(k) = 3$ and $u(k-1) = -1$ gives:

$$-\frac{16}{3} \leq \Delta u(k) \leq -\frac{13}{3}$$

as required.

Solution to Exercise 5.8

(a)

$$S_x = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix}, \quad S_{u-1} = \begin{bmatrix} CB \\ CAB + CB \\ \vdots \\ \sum_{j=0}^{N-1} CA^j B \end{bmatrix},$$

$$S_u = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB + CB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{j=0}^{N-1} CA^j B & \sum_{j=0}^{N-2} CA^j B & \cdots & CB \end{bmatrix}.$$

where p is the prediction horizon.

(b) From above, we see that S_x has the structure of an observability matrix. $S_x x(k)$ represents the component of the predicted future outputs which are obtained from the current state x_k.
S_{u-1} represents the propagation of the previous sample's control signal $u(k-1)$ through the prediction horizon. Note that if the control changes over the horizon are zero, then the predicted outputs are given by $Y = S_x x(k) + S_{u-1} u(k-1)$.
S_u is a lower triangular Toeplitz matrix which describes the effects of the sequence of control changes ΔU on the predicted outputs.

Solution to Exercise 5.9

(a) $V^*(Ax + Bu_0^*)$ is the optimal cost if $Ax + Bu_0^*$ is the current state. $\tilde{U}(x)$ is only a candidate input sequence and not necessarily the optimal input sequence if the current state is $Ax + Bu_0^*$. An arbitrary input sequence will result in a cost that is greater or equal to the optimal cost, hence

$$V^*(Ax + Bu_0^*) \leq V(Ax + Bu_0^*, \tilde{U}(x)).$$

(b) Since $x = x_0^*$, it follows that

$$V^*(x) = x^T Q x + {u_0^*}^T R u_0^* + {x_N^*}^T P x_N^* + \sum_{k=1}^{N-1} {x_k^*}^T Q x_k^* + {u_k^*}^T R u_k^*.$$

Since $Ax + Bu_0^*$ is the optimal state at time $k+1$, it follows that

$$\begin{aligned} V(Ax + Bu_0^*, \tilde{U}(x)) =& ((A+BK)x_N^*)^T P(A+BK)x_N^* + {x_N^*}^T Q x_N^* \\ &+ {x_N^*}^T K^T R K x_N^* + \sum_{k=1}^{N-1} {x_k^*}^T Q x_k^* + {u_k^*}^T R u_k^* \end{aligned} \tag{44}$$

By subtracting the expressions of $V(Ax + Bu_0^*, \tilde{U}(x))$ and $V^*(x)$, the expression of $l(x)$ follows by inspection.

(c) If Q and R are positive definite, then $x^T Q x > 0 \ \forall x \neq 0$ and ${u_0^*}^T R u_0^* \geq 0 \ \forall x$. By noting that the expression of $l(x)$ can be rewritten as

$$\begin{aligned} l(x) =& - x^T Q x - {u_0^*}^T R u_0^* \\ &+ {x_N^*}^T [(A+BK)^T P(A+BK) - P + Q + K^T R K] x_N^*. \end{aligned} \tag{45}$$

and, since

$$(A+BK)^T P(A+BK) - P + Q + K^T R K \leq 0$$

then

$$l(x) < 0 \quad \forall x \neq 0.$$

We can now proceed to show that $V^*(\cdot)$ is a Lyapunov function for the closed-loop system. Combining part (a) with part (b) we get

$$V^*(Ax + Bu_0^*) \leq V(Ax + Bu_0^*, \tilde{U}(x)) = V^*(x) + l(x)$$

Since $l(x) < 0 \ \forall x \neq 0$, it follows that $V(Ax + Bu_0^*, \tilde{U}(x)) < V^*(x)$, hence

$$V^*(Ax + Bu_0^*) < V^*(x) \quad \forall x \neq 0.$$

Since P, Q and R are all positive definite, it follows that $V^*(0) = 0$ (since $U^*(0) = 0$) and

$$V^*(x) \geq x^T Q x > 0 \quad \forall x \neq 0.$$

(Note that it is also important to show that $V^*(\cdot)$ satisfies certain continuity conditions. However, the proof of this is not straightforward. For this course, it is sufficient to just assume that $V^*(\cdot)$ has the required continuity properties.) $V^*(\cdot)$ satisfies all the conditions to be a Lyapunov function for the closed-loop system

$$x(t+1) = Ax(t) + Bu_0^*.$$

Solution to Exercise 5.10

(a) The principle of optimality still holds. However, since the stage-cost converges to $l(x_s, u_s) > 0$, the cost-to-go $V^0(x)$ is infinite for all feasible x.

(b) Since the principle of optimality holds, yes.

(c) Inequality (5) is still valid. However, since optimal cost $V^0(x)$ is infinite for all feasible x, the inequality does not imply that $V^0(x)$ converges to zero. Note that $V^0(x)$ is not a valid Lyapunov function since is not positive definite. In fact, $V^0(0)$ does not exist since it is the solution of an unfeasible problem.

(d) No.

Solution to Exercise 5.11

We start by rewriting the model in vectorial form as a function of x_0, i.e.,

$$X = Fx_0 + GU.$$

Then, we can substitute summation over x in the cost function by the equivalent vectorial expression, i.e.,

$$\min_U x_0^T F^T Q F x_0 + 2x_0^T F^T Q G U + U^T G Q G U + \sum_{i=0}^{N-1} ||u_i||_1.$$

Since the 1-norm of a vector is defined as the sum of the absolute values of its elements, we can rewrite the above as

$$\min_U x_0^T F^T Q F x_0 + 2x_0^T F^T Q G U + U^T G Q G U + \mathbf{1}^T |U|,$$

where the vector $\mathbf{1}$ is an appropriately sized vector of 1s. The above unconstrained optimization function is not quadratic, due to the appearance of the absolute value term $|U|$. We can rewrite the above as a constrained quadratic problem by introducing an additional vector-valued variable V, such that

$$-V \leq U \leq V,$$

where the inequalities are to be interpreted element-wise. The problem can then be rewritten as

$$\min_U x_0^T F^T Q F x_0 + 2x_0^T F^T Q G U + U^T G Q G U + \mathbf{1}^T V,$$

$$\text{subject to} - V \leq U \leq V,$$

which is a quadratic program (QP) in standard form.

Solution to Exercise 5.12

Note that there is only one state and one input in this example, hence K, P, etc., are all scalars.

(a) First, the control gain K has to be chosen such that the closed-loop system $x(t+1) = (A + BK)x(t)$ is stable, i.e.

$$\begin{aligned} \rho(A + BK) < 1 &\Leftrightarrow \rho(2 + 3K) < 1 \\ &\Leftrightarrow |2 + 3K| < 1 \\ &\Leftrightarrow -1 < 2 + 3K < 1 \\ &\Leftrightarrow -1 < K < -\tfrac{1}{3}. \end{aligned} \tag{46}$$

Next, recall that the terminal cost has to be a Lyapunov function associated with the terminal controller, i.e. P has to be strictly positive if it is a scalar (recall that P has to be positive definite if P is a matrix) and

$$\begin{aligned} &(A+BK)^T P(A+BK) - P \leq -Q - K^T RK \\ \Leftrightarrow &P(2+3K)^2 - P \leq -10 - 2K^2 \\ \Leftrightarrow &P(9K^2+12K+3) \leq -10-2K^2. \end{aligned} \tag{47}$$

In order to proceed, note that the function $f(K) = 9K^2 + 12K + 3$ has roots at -1 and $-1/3$ and that $f(K)$ is strictly less than zero over the range $-1 < K < -1/3$, hence

$$P(9K^2+12K+3) \leq -10-2K^2 \Leftrightarrow P \geq \frac{-10-2K^2}{9K^2+12K+3}.$$

Next, note that $-(10+2K^2)/(9K^2+12K+3)$ is also strictly greater than zero over the range $-1 < K < -1/3$. Combining this with the fact that P has to be strictly positive, it follows that

$$P \geq -\frac{10+2K^2}{9K^2+12K+3}$$

is sufficient for the above inequality to hold if $-1 < K < -1/3$.

(b) The lower constraint on the state implies that $-8 \leq -c$ and the upper constraint implies that $c \leq 10$. Combining this with the assumption that $c > 0$, it follows that $0 < c \leq 8$.

(c) We require that $-3 \leq Kx_N \leq 2$ for all x_N that satisfy $-c \leq x_N \leq c$. Since K and x_N are scalars, we only need to consider the upper and lower admissible values of the terminal state; if the input constraints are satisfied at the extreme values of the set of admissible terminal states, then the input constraints are also satisfied at intermediate values for the terminal state. Taking the upper admissible value for the terminal state, it follows that $-3 \leq Kc \leq 2$ and the lower admissible value for the terminal state, it follows that $-3 \leq -Kc \leq 2$ or, equivalently, $3 \geq Kc \geq -2$. Combining all of the above constraints, it follows that

$$\max\{-2,-3\} \leq Kc \leq \min\{2,3\} \Leftrightarrow -2 \leq Kc \leq 2 \Leftrightarrow |Kc| \leq 2.$$

Under the assumption that $K \neq 0$ (if $K = 0$ then any positive value for c is admissible) , it follows that

$$0 < c \leq 2/|K|.$$

(d) We require that $-c \leq (A+BK)x_N \leq c$ for all x_N that satisfy $-c \leq x_N \leq c$. As with with part (c), since A, B and K are scalars, we need only check the extreme values of the terminal state for the system $x(t+1) = (2+3K)x(t)$. Taking the upper admissible value for the terminal state, we require that $-c \leq (2+3K)c \leq c$ and the lower admissible value for the terminal state, we require that $-c \leq -(2+3K)c \leq c$ or,

equivalently, $c \geq (2+3K)c \geq -c$. Combining all of the above constraints, it follows that

$$-c \leq (2+3K)c \leq c \Leftrightarrow -1 \leq 2+3K \leq 1 \Leftrightarrow -1 \leq K \leq -\frac{1}{3}.$$

(e) Combining parts (a) and (d), it is clear that only values for K in the range $-1 < K < -1/3$ are allowed.
From part (c) it follows that, for a given c, we have to ensure that $|K| \leq 2/c$. If $c \geq 6$, then this implies that K also has to satisfy $|K| < 1/3$. This is in conflict with the constraint that $-1 < K < -1/3$, which implies that $1/3 < |K| < 1$. Hence, only values of c less than 6 are allowed, i.e. $0 < c < 6$.

Solution to Exercise 5.13

(a) In order to compute L, we solve the algebraic Riccati equation

$$P = 17 + 3^2 P - \frac{3^2 P^2}{2+P}$$

and the corresponding optimal control law $L = \frac{3P}{2+P}$, from which we obtain $P = 34$ and $L = 51/18$. The LQR feedback control law is therefore $u(t) = -(51/18)x(t)$.

(b) To move from the infinite horizon to finite horizon, we add the final cost $x_N^{\mathrm{T}} Q_f x_N$, where $Q_f = P = 34$. The resulting MPC formulation is

$$\begin{aligned} \underset{\{u_0, \ldots, u_{N-1}\}}{\text{minimize}} \quad & \sum_{k=0}^{N-1} \left(17x_k^2 + 2u_k^2\right) + 34x_N^2, \\ \text{subject to} \quad & x_{k+1} = 3x_k + u_k, \\ & x_0 = x(t), \end{aligned}$$

In order to make the LQR and the unconstrained MPC formulations equivalent, the MPC mode 2 control law should be chosen according the LQR feedback law, i.e., $u_k = -(51/18)x_k$, for $k = N, N+1, \ldots$.

(c) In order to guarantee the stability of the MPC, the final state constraint should be state-feasible (i.e., satisfy the state constraint), input-feasible (i.e., satisfy the input constraint), and invariant while the mode 2 control law is used. These conditions are needed for proving that the cost function is a valid Lyapunov function for the MPC controller.
Let us define the final state constraint as $x_l \leq x_N \leq x_u$.

- We examine the invariance by looking at the closed-loop dynamics. We have that $(A - BK) = 1/6$, which means that we only have to require that $x_l \leq 0$ and $x_u \geq 0$.
- To be state-admissible, $x_l \geq -8$ and $x_u \leq 8$.
- To be input-admissible, $-20 \leq -Kx_N \leq 25$, for all $x_l \leq x_N \leq x_u$. Since the constraint is linear we only need to check that it is satisfied

for the extreme values: (i) $-25 \leq -Kx_l \leq 20$ and (ii) $-25 \leq -Kx_u \leq 20$, where $K = (51/18)$. We can rewrite the constraints as

$$-\frac{450}{51} \leq x_l \leq \frac{360}{51} \quad \text{and} \quad -\frac{450}{51} \leq x_u \leq \frac{360}{51}$$

Combining the three conditions and looking for the less restrictive constraint on x_N, we obtain $x_l = -8$ and $x_u = 360/51$.

Solution to Exercise 5.14

(a) The infinite cost is given by

$$\sum_{k=0}^{\infty} \frac{1}{2}(y_k^2 + u_k^2) = x_0^T P x_0,$$

where P respects the ARE

$$(A + BK)^T P(A + BK) - P = -Q - K^T RK,$$

where $Q = \frac{1}{2}C^T C$, $R = \frac{1}{2}$, $K = \frac{1}{\sqrt{2}}C$. Therefore, if we note that $P = I$ and substitute it in the ARE, then we get

$$(A + BK)^T P(A + BK) - P = -Q - K^T RK = \begin{bmatrix} -\frac{3}{8} & -\frac{3}{8} \\ -\frac{3}{8} & -\frac{3}{8} \end{bmatrix}$$

Hence the solution of the ARE is $P = I$, which implies $\sum_{k=0}^{\infty} \frac{1}{2}(y_k^2 + u_k^2) = ||x_0||^2$.

(b) From part (a), the finite-horizon cost function is equal to infinite-horizon cost function and the control input is $u_k = \frac{1}{\sqrt{2}}y_k$ for all $k \geq N$. Let $J^\star(x(t))$ be the minimum value of this cost over $\{u^\star_{0|t}, ..., u^\star_{N-1|t}\}$. Then, at time $t + 1$, the predicted input sequence is $\{u^\star_{1|t}, ..., u^\star_{N-1|t}, Ky_N\}$, which gives

$$J(x(t+1)) = \sum_{k=1}^{\infty} \frac{1}{2}(y_{k|t}^2 + u_{k|t}^2) = J^\star(x(t)) - \frac{1}{2}(y_k^2 + u_k^2)$$

and since the optimal cost at time $t + 1$ satisfies

$$J^\star(x(t+1)) \leq J(x(t+1)),$$

we can conclude that

$$J^\star(x(t+1)) \leq J^\star(x(t)) - \frac{1}{2}(y_k^2 + u_k^2).$$

This implies closed loop stability because $J^\star(x(t))$ is positive definite in $x(t)$ since (A, C) is observable (the observability matrix is full rank) and (A, B) is reachable (the controllability matrix is full rank).

(c) At time t, if the constraint is

$$-1 \leq y_t = Cx_t = \frac{1}{\sqrt{2}}[1\ 1]x_t \leq -1$$

and at time $t+1$ if the constraint is

$$-1 \leq y_{t+1} = C(A+BCK)x_t = \frac{1}{\sqrt{2}}[-1\ 1]x_t \leq -1$$

that implies that at $t+k$

$$-1 \leq y_{t+k} = C(A+BCK)^{t+k}x_t \leq -1.$$

(d) The closed loop system will be stable if the predicted trajectories satisfy $-1 \leq y_t \leq 1$ for all $t \geq 0$. Imposing constraints in the MPC formulation ensures constraint satisfaction $-1 \leq y_t \leq 1$ for $t = 1, 2, ..., N$. Also, by imposing the constraint on $N+1$ we ensure constraint satisfaction to all $t > 0$, as proven in (c). Moreover, we have proven in (a) and (b) that the unconstrained controller is stable and that the finite cost is equivalent to the infinite cost, under the control law $u_t^\star$. Therefore, the closed loop system will be necessarily stable.

Solution to Exercise 5.15

```
A = [1.1  2; 0  0.95];
B = [0;0.0787];
C = [-1  1];
D = 0;

Q = C'*C;
R = 1;

z0 = [0.5, -0.5];

nstates = 2;
ninputs = 1;

[K_lqr,Qf,~] = dlqr(A,B,Q,R);
K_lqr = -K_lqr;
M = 1;
while 1
yalmip('clear');

% maximize
x = sdpvar(2,1);

constr = [];

cost = -K_lqr*(A+B*K_lqr)^(M+1)*x;
for i = 0:M
constr = [constr, K_lqr*(A+B*K_lqr)^i*x<=1, K_lqr*(A+B*K_lqr)^i
    *x>=-1]
end

options = sdpsettings('solver','linprog','savedebug',0,'
    savesolveroutput',0);
```

```
diagnostic = optimize(constr,cost,options);

u_max = -double(cost);

yalmip('clear');

% minimize
x = sdpvar(2,1);

constr = [];

cost = K_lqr*(A+B*K_lqr)^(M+1)*x;
for i = 0:M
constr = [constr, K_lqr*(A+B*K_lqr)^i*x<=1, K_lqr*(A+B*K_lqr)^i
    *x>=-1]
end

options = sdpsettings('solver','linprog','savedebug',0,'
    savesolveroutput',0);
diagnostic = optimize(constr,cost,options);

u_min = double(cost);

if u_min >=-1 && u_max <= 1
Mc = M;
break;
else

M = M+1;
end
end

A_ineq = [];
b_ineq = [];
for i = 0:Mc

A_ineq(2*(i+1)-1,:) = (K_lqr*(A+B*K_lqr)^(i));
A_ineq(2*(i+1),:) = -(K_lqr*(A+B*K_lqr)^(i));
b_ineq(2*(i+1)-1,:) = 1;
b_ineq(2*(i+1),:) = 1;

end
P = polytope(A_ineq,b_ineq); % you need MPT toolbox installed (
    http://people.ee.ethz.ch/~mpt/3/)

%% MPC (a)
N = 50;
H = 10;

z = zeros(nstates,N);
y = zeros(1,N);
u = zeros(ninputs,N);

z(:,1) = z0; % set initial state
y(1) = C*z0';
z_old = z(:,1);

for i = 1:N % simulate for N steps

yalmip('clear') % clear environment
z_mpc = sdpvar(nstates,H+1);
u_mpc = sdpvar(ninputs,H);
```

```
% y_mpc = sdpvar(1,H);

cost = 0; %total cost of MPC
constr = []; %constr of

% note that the final state is H+1 since 1 is the initial (old)
    state
for k = 1:H
constr = [constr,z_mpc(:,k+1) == A*z_mpc(:,k)+B*u_mpc(:,k)];
cost = cost + z_mpc(:,k)'*Q*z_mpc(:,k) + u_mpc(k)'*R*u_mpc(k);
constr = [constr, u_mpc(:,k)>=-1, u_mpc(:,k)<=1];

end

cost = cost + z_mpc(:,H+1)'*Qf*z_mpc(:,H+1);
constr = [constr, A_ineq*z_mpc(:,H+1) <= b_ineq];

constr = [constr,z_mpc(:,1) == z_old];

options = sdpsettings('solver','quadprog','savesolveroutput',0,
    'savedebug',0);
diagnostic = optimize(constr,cost,options);

z_horizon = double(z_mpc); % predicted states by MPC
u_horizon = double(u_mpc); % predicted inputs by MPC

for k = 1:H
y_horizon(k) = C*z_horizon(:,k);
end

% save optimal input and simulate (and save) the new state
u(i) = u_horizon(:,1);
[z(:,i+1),y(i+1)] = sim_model(z(:,i),u(:,i),A,B,C); % x_new = A
    *x+B*u;y = C*x_new;
z_old = z(:,i+1);
end
```

Solution to Exercise 5.16

(a)

```
function [u, exitflag]=findCtrl(A,B,ulim, x0, xtgt, T)
[n,m]=size(B);
HT=[];
for k = T-1:-1:0,
HT = [HT, A^k*B];
end;
ht=xtgt-(A^T)*x0;
c = ones(m*T,1);
Aeq = HT;
beq = ht;
umin = -ulim*ones(m*T,1);
umax = ulim*ones(m*T,1);
[xstar, fstar, lpexitflag]=linprog(c, [], [], Aeq, beq,
    umin, umax);
if lpexitflag==1,
exitflag=1;
u = reshape(xstar, m, T);
else,
exitflag=-1;
```

```
u = zeros(m,T);
end;
```

(b)

```
function [u, Tstar, exitflag]= timeOptimalCtrl(A,B,ulim,
    x0, xtgt, Tguess)
[n,m]=size(B);
maxIter=5; feasible=-1;
for i=1:maxIter,
[u, exitflag]=findCtrl(A,B,ulim, x0, xtgt, Tguess)
if exitflag==1,
feasible=1;
break;
else
Tguess=Tguess*2;
end;
end;
if feasible,
Thi=Tguess;
Tlo=0;
done=0;
while ~done,
Tmid=floor((Tlo+Thi)/2);
[u, exitflag]=findCtrl(A,B,ulim,x0,xtgt, Tmid);
if exitflag==1,
Thi=Tmid;
else
Tlo=Tmid;
end;
if Thi-Tlo<=1,
break;
end;
end;
[u, lpexitflag]=findCtrl(A,B,ulim,x0,xtgt,Tlo);
if lpexitflag==1,
Tstar=Tlo;
exitflag=1;
else
[u, lpexitflag]=findCtrl(A,B,ulim,x0,xtgt,Thi);
Tstar=Thi;
exitflag=1;
end;
else
exitflag=-1;
u=zeros(m,T);
end;
```

Solution to Exercise 5.17

(a) Repeating the MPC optimization introduces feedback into the control law (which provides some robustness to model and measurement uncertainty), since the optimal predicted input sequence at time k depends on $x(k)$. It also removes some of the suboptimality that results from optimizing performance over a finite number of free variables.

(b) The receding horizon principle involves finding an open-loop control sequence of inputs that minimizes a certain cost function over a finite horizon. This procedure is performed each time new measurements are

available. The first step in the computed control sequence is used as the control signal. Although the computed control sequences are open-loop sequences, the calculation of a new sequence at each sample can be thought of as providing feedback.

(c) The *prediction horizon* is the finite horizon over which the cost function is evaluated. The *control horizon* is smaller than or equal to the prediction horizon. The decision variables are used until the control horizon is reached, thereafter the mode 2 controller is used.

References

[1] K. J. Åström and B. Wittenmark. *Computer-controlled systems: theory and design*. Courier Corporation, 2013.

[2] Stephen Boyd. *Linear Dynamical Systems Exercises EE363*. Stanford University, USA.

[3] M. Cannon. C21 Model predictive control. Lecture notes, 2011. University of Oxford, Oxford, GB.

[4] J. Colin. ME-425 Model Predictive Control. Lecture notes, 2012. EPFL École polytechnique fédérale de Lausanne, Lausanne, Switzerland.

[5] E. Frazzoli. 6.241J Dynamic Systems and Control. Lecture notes, 2011. MIT Massachusetts Institute of Technology, Cambridge, MA, USA.

[6] R. Johansson. FRTN15 Predictive Control. Lecture notes, 2016. Lund University, Lund, Sweden.

[7] Colin Jones. *Model Predictive Control Course Exercises ME-425*. Laboratoire d'Automatique, EPFL.

[8] Jan Maciejowski. Material for 4f3: Nonlinear control, 2015.

[9] Rawlings, Blake, and Mayne. *Model predictive control: Theory and design*. Nob Hill Pub., 2009.