Optimization-Based Control

Richard M. Murray
Control and Dynamical Systems
California Institute of Technology

DRAFT v2.1a, February 15, 2010

Preface

These notes serve as a supplement to *Feedback Systems* by Åström and Murray and expand on some of the topics introduced there. They are motivated by the increasing role of online optimization in feedback systems. This is a change from the traditional use of optimization in control theory for *offline* design of control laws and state estimators. Fueled by Moore's law and improvements in real-time algorithms, it is now possible to perform estimation and control design algorithms online, allowing the system to better account for nonlinearities and to adapt to changes in the underlying dynamics of the controlled process. This changes the way that we think about estimation and control since it allows much greater flexibility in the design process and more modularity and flexibility in the overall system.

Our goal in this supplement is to introduce the keys formalisms and tools required to design optimization-based controllers. Key topics include real-time trajectory generation using differential flatness, the maximum principle, dynamic programming, receding horizon optimal control, stochastic processes, Kalman filtering, moving horizon estimation and (distributed) sensor fusion. While these topics might normal consititute separate textbooks, in this set of notes we attempt to present them in a compact way that allows them to be used in engineering design. We also briefly survey additional advanced topics through the text, with pointers to further information for interested readers.

This supplement as been used in a second quarter controls course at Caltech, taken by a mixture of advanced undergraduates and beginning graduate students with interest in a variety of application areas. The first half of the 10 week course focuses on trajectory generation and optimal control, ending with receding horizon control. In the second half of the course, we introduce stochastic processes and derive the Kalman filter and its various extensions, including the information filter and sensor fusion. The prerequisites for the course are based on the material covered in *Feedback Systems*, including basic knowledge in Lyapunov stability theory and observers. If needed, these topics can be inserted at the appropriate point in covering the material in this supplement.

The notation and conventions in the book follow those used in the main text. Because the notes may not be used for a standalone class, we have attempted to write each as a standalone reference for advanced topics that are introduced in *Feedback Systems*. To this end, each chapter starts with a short description of the prerequisites for the chapter and citations to the relevant literature. Advanced sections, marked by the "dangerous bend" symbol shown in the margin, contain material that requires a slightly more technical background, of the sort that would be expected of graduate students in engineering. Additional information is available on the *Feedback Systems* web site:

Contents

Chapter 1
Trajectory Generation and Tracking

This chapter expands on Section 7.5 of *Feedback Systems* by Åström and Murray (ÅM08), which introduces the use of feedforward compensation in control system design. We begin with a review of the two degree of freedom design approach and then focus on the problem of generating feasible trajectories for a (nonlinear) control system. We make use of the concept of differential flatness as a tool for generating feasible trajectories.

Prerequisites. Readers should be familiar with modeling of input/output control systems using differential equations, linearization of a system around an equilibrium point and state space control of linear systems, including reachability and eigenvalue assignment. Although this material supplements concepts introduced in the context of output feedback and state estimation, no knowledge of observers is required.

1.1 Two Degree of Freedom Design

A large class of control problems consist of planning and following a trajectory in the presence of noise and uncertainty. Examples include autonomous vehicles maneuvering in city streets, mobile robots performing tasks on factor floors (or other planets), manufacturing systems that regulate the flow of parts and materials through a plant or factory, and supply chain management systems that balance orders and inventories across an enterprise. All of these systems are highly nonlinear and demand accurate performance.

To control such systems, we make use of the notion of *two degree of freedom* controller design. This is a standard technique in linear control theory that separates a controller into a feedforward compensator and a feedback compensator. The feedforward compensator generates the nominal input required to track a given reference trajectory. The feedback compensator corrects for errors between the desired and actual trajectories. This is shown schematically in Figure 1.1.

In a nonlinear setting, two degree of freedom controller design decouples the trajectory generation and asymptotic tracking problems. Given a desired output trajectory, we first construct a state space trajectory x_d and a nominal input u_d that satisfy the equations of motion. The error system can then be written as a time-varying control system in terms of the error, $e = x - x_d$. Under the assumption that that tracking error remains small, we can linearize this time-varying system about $e = 0$ and stabilize the $e = 0$ state. (Note: in ÅM08 the notation u_{ff} was used for the desired [feedforward] input. We use u_d here to match the desired state x_d.)

More formally, we assume that our process dynamics can be described by a

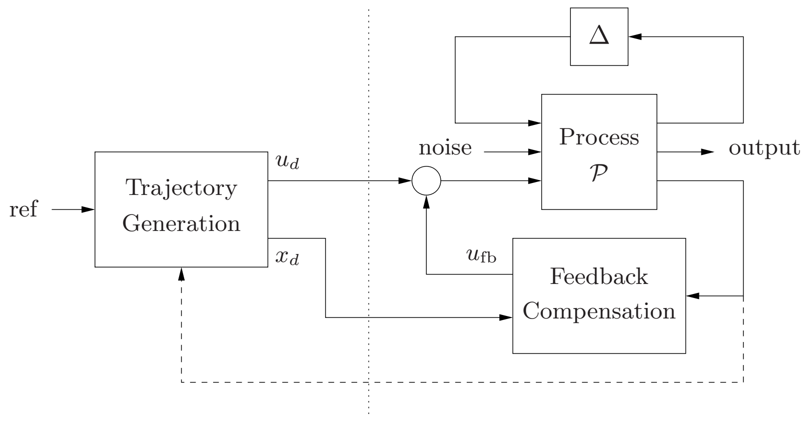


Figure 1.1: Two degree of freedom controller design for a process P with uncertainty Δ. The controller consists of a trajectory generator and feedback controller. The trajectory generation subsystem computes a feedforward command u_d along with the desired state x_d. The state feedback controller uses the measured (or estimated) state and desired state to compute a corrective input u_{fb}. Uncertainty is represented by the block Δ, representing unmodeled dynamics, as well as disturbances and noise.

nonlinear differential equation of the form

$$\begin{aligned} \dot x &= f(x,u), \qquad & x \in \mathbb{R}^n, u \in \mathbb{R}^m, \\ y &= h(x,u), \qquad & y \in \mathbb{R}^p, \end{aligned} \tag{1.1}$$

where x is the system state, u is a vector of inputs and f is a smooth function describing the dynamics of the process. The smooth function h describes the output y that we wish to control. We are particularly interested in the class of control problems in which we wish to track a time-varying reference trajectory $r(t)$, called the *trajectory tracking* problem. In particular, we wish to find a control law $u = \alpha(x, r(\cdot))$ such that

$$\lim_{t\to\infty} \bigl(y(t) - r(t)\bigr) = 0.$$

We use the notation $r(\cdot)$ to indicate that the control law can depend not only on the reference signal $r(t)$ but also derivatives of the reference signal.

A *feasible trajectory* for the system (1.1) is a pair $(x_d(t), u_d(t))$ that satisfies the differential equation and generates the desired trajectory:

$$\dot x_d(t) = f\bigl(x_d(t), u_d(t)\bigr) \qquad r(t) = h\bigl(x_d(t), u_d(t)\bigr).$$

The problem of finding a feasible trajectory for a system is called the *trajectory generation* problem, with x_d representing the desired state for the (nominal) system and u_d representing the desired input or the feedforward control. If we can find a feasible trajectory for the system, we can search for controllers of the form $u = \alpha(x, x_d, u_d)$ that track the desired reference trajectory.

In many applications, it is possible to attach a cost function to trajectories that describe how well they balance trajectory tracking with other factors, such as the magnitude of the inputs required. In such applications, it is natural to ask that we find the *optimal* controller with respect to some cost function. We can again use the two degree of freedom paradigm with an optimal control computation for generating the feasible trajectory. This subject is examined in more detail in Chapter 2. In addition, we can take the extra step of updating the generated trajectory based on the current state of the system. This additional feedback path is denoted by a dashed line in Figure 1.1 and allows the use of so-called *receding horizon control* techniques: a (optimal) feasible trajectory is computed from the current position to the desired position over a finite time T horizon, used for a short period of time $\delta < T$, and then recomputed based on the new system state. Receding horizon control is described in more detail in Chapter 3.

A key advantage of optimization-based approaches is that they allow the potential for customization of the controller based on changes in *mission*, *condition* and *environment*. Because the controller is solving the optimization problem online, updates can be made to the cost function, to change the desired operation of the system; to the model, to reflect changes in parameter values or damage to sensors and actuators; and to the constraints, to reflect new regions of the state space that must be avoided due to external influences. Thus, many of the challenges of designing controllers that are robust to a large set of possible uncertainties become embedded in the online optimization.

1.2 Trajectory Tracking and Gain Scheduling

We begin by considering the problem of tracking a feasible trajectory. Assume that a trajectory generator is able to generate a trajectory (x_d, u_d) that satisfies the dynamics (1.1) and satisfies $r(t) = h(x_d(t), u_d(t))$. To design the controller, we construct the *error system*. Let $e = x - x_d$ and $v = u - u_d$ and compute the dynamics for the error:

$$\begin{aligned}\dot e &= \dot x - \dot x_d = f(x,u) - f(x_d,u_d) \\ &= f(e + x_d, v + u_d) - f(x_d) =: F(e, v, x_d(t), u_d(t)).\end{aligned}$$

The function F represents the dynamics of the error, with control input v and external inputs x_d and u_d. In general, this system is time-varying through the desired state and input.

For trajectory tracking, we can assume that e is small (if our controller is doing a good job), and so we can linearize around $e = 0$:

$$\frac{de}{dt} \approx A(t)e + B(t)v, \qquad A(t) = \left.\frac{\partial F}{\partial e}\right|_{(x_d(t),u_d(t))}, \quad B(t) = \left.\frac{\partial F}{\partial v}\right|_{(x_d(t),u_d(t)}.$$

It is often the case that $A(t)$ and $B(t)$ depend only on x_d, in which case it is convenient to write $A(t) = A(x_d)$ and $B(t) = B(x_d)$.

We start by reviewing the case where $A(t)$ and $B(t)$ are constant, in which case our error dynamics become

$$\dot e = Ae + Bv.$$

This occurs, for example, if the original nonlinear system is linear. We can then search for a control system of the form

$$v = -Ke + k_r r.$$

In the case where r is constant, we can apply the results of Chapter 6 of ÅM08 and solve the problem by finding a gain matrix K that gives the desired closed loop dynamics (e.g., by eigenvalue assignment) and choosing k_r to give the desired output value at equilibrium. The equilibrium point is given by

$$x_e = -(A - BK)^{-1}Bk_r r \qquad \Longrightarrow \qquad y_e = -C(A - BK)^{-1}Bk_r r$$

and if we wish the output to be $y = r$ it follows that

$$k_r = -1/\bigl(C(A - BK)^{-1}B\bigr).$$

It can be shown that this formulation is equivalent to a two degree of freedom design where x_d and u_d are chosen to give the desired reference output (Exercise 1.1).

Returning to the full nonlinear system, assume now that x_d and u_d are either constant or slowly varying (with respect to the performance criterion). This allows us to consider just the (constant) linearized system given by $(A(x_d), B(x_d))$. If we design a state feedback controller $K(x_d)$ for each x_d, then we can regulate the system using the feedback

$$v = K(x_d)e.$$

Substituting back the definitions of e and v, our controller becomes

$$u = -K(x_d)(x - x_d) + u_d.$$

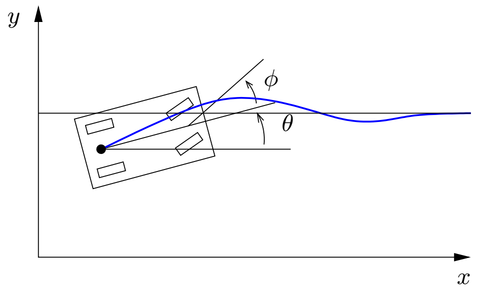
Note that the controller $u = \alpha(x, x_d, u_d)$ depends on (x_d, u_d), which themselves depend on the desired reference trajectory. This form of controller is called a *gain scheduled* linear controller with *feedforward* u_d.

More generally, the term gain scheduling is used to describe any controller that depends on a set of measured parameters in the system. So, for example, we might write

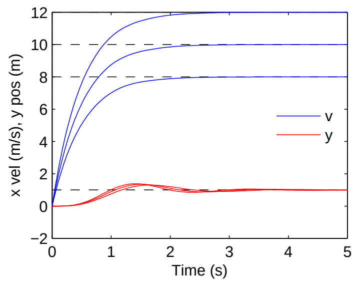
$$u = -K(x, \mu) \cdot (x - x_d) + u_d,$$

where $K(x, \mu)$ depends on the *current* system state (or some portion of it) and an external parameter μ. The dependence on the current state x (as opposed to the desired state x_d) allows us to modify the closed loop dynamics differently depending on our location in the state space. This is particularly useful when the dynamics of the process vary depending on some subset of the states (such as the altitude for an aircraft or the internal temperature for a chemical reaction). The dependence on μ can be used to capture the dependence on the reference trajectory, or they can reflect changes in the environment or performance specifications that are not modeled in the state of the controller.

Example 1.1 Steering control with velocity scheduling
Consider the problem of controlling the motion of a automobile so that it follows a given trajectory on the ground, as shown in Figure 1.2a. We use the model derived



(a)



(b)

Figure 1.2: Vehicle steering using gain scheduling.

in ÅM08, choosing the reference point to be the center of the rear wheels. This gives dynamics of the form

$$\dot{x} = \cos\theta\, v, \qquad \dot{y} \;= \sin\theta\, v, \qquad \dot{\theta} = \frac{v}{l}\tan\phi, \tag{1.2}$$

where (x, y, θ) is the position and orientation of the vehicle, v is the velocity and ϕ is the steering angle, both considered to be inputs, and l is the wheelbase.

A simple feasible trajectory for the system is to follow a straight line in the x direction at lateral position y_r and fixed velocity v_r. This corresponds to a desired state $x_d = (v_r t, y_r, 0)$ and nominal input $u_d = (v_r, 0)$. Note that (x_d, u_d) is not an equilibrium point for the system, but it does satisfy the equations of motion.

Linearizing the system about the desired trajectory, we obtain

$$\begin{aligned} A_d &= \left.\frac{\partial f}{\partial x}\right|_{(x_d,u_d)} = \left.\begin{bmatrix} 0 & 0 & -\sin\theta \\ 0 & 0 & \cos\theta \\ 0 & 0 & 0 \end{bmatrix}\right|_{(x_d,u_d)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \\ B_d &= \left.\frac{\partial f}{\partial u}\right|_{(x_d,u_d)} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & v_r/l \end{bmatrix}. \end{aligned}$$

We form the error dynamics by setting $e = x - x_d$ and $w = u - u_d$:

$$\dot{e}_x = w_1, \qquad \dot{e}_y = e_\theta, \qquad \dot{e}_\theta = \frac{v_r}{l} w_2.$$

We see that the first state is decoupled from the second two states and hence we can design a controller by treating these two subsystems separately. Suppose that we wish to place the closed loop eigenvalues of the longitudinal dynamics (e_x) at λ_1 and place the closed loop eigenvalues of the lateral dynamics (e_y, e_θ) at the roots of the polynomial equation $s^2 + a_1 s + a_2 = 0$. This can accomplished by setting

$$\begin{aligned} w_1 &= -\lambda_1 e_x \\ w_2 &= \frac{l}{v_r}(a_1 e_y + a_2 e_\theta). \end{aligned}$$

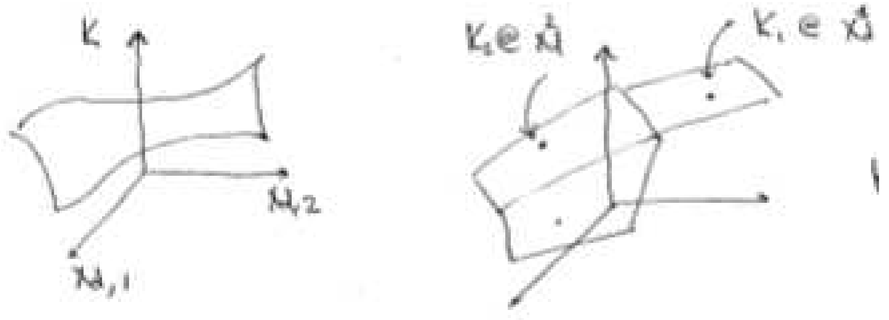


Figure 1.3: Gain scheduling. A general gain scheduling design involves finding a gain K at each desired operating point. This can be thought of as a gain surface, as shown on the left (for the case of a scalar gain). An approximation to this gain can be obtained by computing the gains at a fixed number of operating points and then interpolated between those gains. This gives an approximation of the continuous gain surface, as shown on the right.

Note that the gains depend on the velocity v_r (or equivalently on the nominal input u_d), giving us a gain scheduled controller.

In the original inputs and state coordinates, the controller has the form

$$\begin{bmatrix} v \\ \phi \end{bmatrix} = -\underbrace{\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \dfrac{a_1 l}{v_r} & \dfrac{a_2 l}{v_r} \end{bmatrix}}_{K_d} \underbrace{\begin{bmatrix} x - v_r t \\ y - y_r \\ \theta \end{bmatrix}}_{e} + \underbrace{\begin{bmatrix} v_r \\ 0 \end{bmatrix}}_{u_d}.$$

The form of the controller shows that at low speeds the gains in the steering angle will be high, meaning that we must turn the wheel harder to achieve the same effect. As the speed increases, the gains become smaller. This matches the usual experience that at high speed a very small amount of actuation is required to control the lateral position of a car. Note that the gains go to infinity when the vehicle is stopped ($v_r = 0$), corresponding to the fact that the system is not reachable at this point.

Figure 1.2b shows the response of the controller to a step change in lateral position at three different reference speeds. Notice that the rate of the response is constant, independent of the reference speed, reflecting the fact that the gain scheduled controllers each set the closed loop poles to the same values. ∇

One limitation of gain scheduling as we have described it is that a separate set of gains must be designed for each operating condition x_d. In practice, gain scheduled controllers are often implemented by designing controllers at a fixed number of operating points and then interpolating the gains between these points, as illustrated in Figure 1.3. Suppose that we have a set of operating points $x_{d,j}$, $j = 1, \ldots, N$. Then we can write our controller as

$$u = u_d - K(x)e \qquad K(x) = \sum_{j=1}^{N} \alpha_j(x) K_j,$$

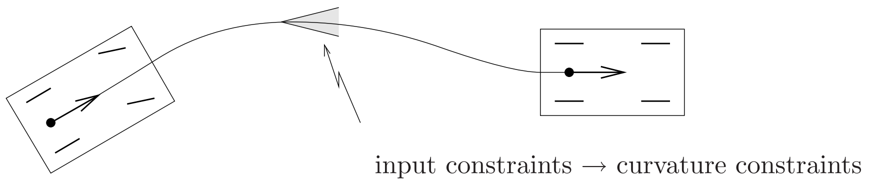


Figure 1.4: Simple model for an automobile. We wish to find a trajectory from an initial state to a final state that satisfies the dynamics of the system and constraints on the curvature (imposed by the limited travel of the front wheels).

where K_j is a set of gains designed around the operating point $x_{d,j}$ and $\alpha_j(x)$ is a weighting factor. For example, we might choose the weights $\alpha_j(x)$ such that we take the gains corresponding to the nearest two operating points and weight them according to the Euclidean distance of the current state from that operating point; if the distance is small then we use a weight very near to 1 and if the distance is far then we use a weight very near to 0.

While the intuition behind gain scheduled controllers is fairly clear, some caution in required in using them. In particular, a gain scheduled controller is not guaranteed to be stable even if $K(x, \mu)$ locally stabilizes the system around a given equilibrium point. Gain scheduling can be proven to work in the case when the gain varies sufficiently slowly (Exercise 1.3).

1.3 Trajectory Generation and Differential Flatness

We now return to the problem of generating a trajectory for a nonlinear system. Consider first the case of finding a trajectory $x_d(t)$ that steers the system from an initial condition x_0 to a final condition x_f. We seek a feasible solution $(x_d(t), u_d(t))$ that satisfies the dynamics of the process:

$$\dot{x}_d = f(x_d, u_d), \qquad x_d(0) = x_0,\, x_d(T) = x_f. \tag{1.3}$$

Formally, this problem corresponds to a two-point boundary value problem and can be quite difficult to solve in general.

In addition, we may wish to satisfy additional constraints on the dynamics. These can include input saturation constraints $|u(t)| < M$, state constraints $g(x) \leq 0$ and tracking constraints $h(x) = r(t)$, each of which gives an algebraic constraint on the states or inputs at each instant in time. We can also attempt to optimize a function by choosing $(x_d(t), u_d(t))$ to minimize

$$\int_0^T L(x, u)dt + V(x(T), u(T)).$$

As an example of the type of problem we would like to study, consider the problem of steering a car from an initial condition to a final condition, as show in Figure 1.4. To solve this problem, we must find a solution to the differential equations (1.2) that satisfies the endpoint conditions. Given the nonlinear nature of the dynamics, it seems unlikely that one could find explicit solutions that satisfy the dynamics except in very special cases (such as driving in a straight line).

A closer inspection of this system shows that it is possible to understand the trajectories of the system by exploiting the particular structure of the dynamics. Suppose that we are given a trajectory for the rear wheels of the system, $x_d(t)$ and $y_d(t)$. From equation (1.2), we see that we can use this solution to solve for the angle of the car by writing

$$\frac{\dot y}{\dot x} = \frac{\sin\theta}{\cos\theta} \qquad \Longrightarrow \qquad \theta_d = \tan^{-1}(\dot y_d/\dot x_d).$$

Furthermore, given θ we can solve for the velocity using the equation

$$\dot x = v\cos\theta \qquad \Longrightarrow \qquad v_d = \dot x_d/\cos\theta_d,$$

assuming $\cos\theta_d \neq 0$ (if it is, use $v = \dot y/\sin\theta$). And given θ, we can solve for ϕ using the relationship

$$\dot\theta = \frac{v}{l}\tan\phi \qquad \Longrightarrow \qquad \phi_d = \tan^{-1}(\frac{l\dot\theta_d}{v_d}).$$

Hence all of the state variables and the inputs can be determined by the trajectory of the rear wheels and its derivatives. This property of a system is known as *differential flatness.*

Definition 1.1 (Differential flatness)**.** A nonlinear system (1.1) is *differentially flat* if there exists a function α such that

$$z = \alpha(x, u, \dot u \ldots, u^{(p)})$$

and we can write the solutions of the nonlinear system as functions of z and a finite number of derivatives

$$\begin{aligned} x &= \beta(z, \dot z, \ldots, z^{(q)}), \\ u &= \gamma(z, \dot z, \ldots, z^{(q)}). \end{aligned} \tag{1.4}$$

For a differentially flat system, all of the feasible trajectories for the system can be written as functions of a flat output $z(\cdot)$ and its derivatives. The number of flat outputs is always equal to the number of system inputs. The kinematic car is differentially flat with the position of the rear wheels as the flat output. Differentially flat systems were originally studied by Fliess et al. [FLMR92].

Differentially flat systems are useful in situations where explicit trajectory generation is required. Since the behavior of a flat system is determined by the flat outputs, we can plan trajectories in output space, and then map these to appropriate inputs. Suppose we wish to generate a feasible trajectory for the the nonlinear system

$$\dot x = f(x, u), \qquad x(0) = x_0,\, x(T) = x_f.$$

If the system is differentially flat then

$$\begin{aligned} x(0) &= \beta\big(z(0), \dot z(0), \ldots, z^{(q)}(0)\big) = x_0, \\ x(T) &= \gamma\big(z(T), \dot z(T), \ldots, z^{(q)}(T)\big) = x_f, \end{aligned} \tag{1.5}$$

and we see that the initial and final condition in the full state space depends on just the output z and its derivatives at the initial and final times. Thus any trajectory for z that satisfies these boundary conditions will be a feasible trajectory for the system, using equation (1.4) to determine the full state space and input trajectories.

In particular, given initial and final conditions on z and its derivatives that satisfy equation (1.5), any curve $z(\cdot)$ satisfying those conditions will correspond to a feasible trajectory of the system. We can parameterize the flat output trajectory using a set of smooth basis functions $\psi_i(t)$:

$$z(t) = \sum_{i=1}^{N} \alpha_i \psi_i(t), \qquad \alpha_i \in \mathbb{R}.$$

We seek a set of coefficients α_i, $i = 1, \dots, N$ such that $z(t)$ satisfies the boundary conditions (1.5). The derivatives of the flat output can be computed in terms of the derivatives of the basis functions:

$$\begin{aligned} \dot{z}(t) &= \sum_{i=1}^{N} \alpha_i \dot{\psi}_i(t) \\ &\vdots \\ \dot{z}^{(q)}(t) &= \sum_{i=1}^{N} \alpha_i \psi_i^{(q)}(t). \end{aligned}$$

We can thus write the conditions on the flat outputs and their derivatives as

$$\begin{bmatrix} \psi_1(0) & \psi_2(0) & \dots & \psi_N(0) \\ \dot{\psi}_1(0) & \dot{\psi}_2(0) & \dots & \dot{\psi}_N(0) \\ \vdots & \vdots & & \vdots \\ \psi_1^{(q)}(0) & \psi_2^{(q)}(0) & \dots & \psi_N^{(q)}(0) \\ \psi_1(T) & \psi_2(T) & \dots & \psi_N(T) \\ \dot{\psi}_1(T) & \dot{\psi}_2(T) & \dots & \dot{\psi}_N(T) \\ \vdots & \vdots & & \vdots \\ \psi_1^{(q)}(T) & \psi_2^{(q)}(T) & \dots & \psi_N^{(q)}(T) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} = \begin{bmatrix} z(0) \\ \dot{z}(0) \\ \vdots \\ z^{(q)}(0) \\ z(T) \\ \dot{z}(T) \\ \vdots \\ z^{(q)}(T) \end{bmatrix}$$

This equation is a *linear* equation of the form $M\alpha = \bar{z}$. Assuming that M has a sufficient number of columns and that it is full column rank, we can solve for a (possibly non-unique) α that solves the trajectory generation problem.

Example 1.2 Nonholonomic integrator
A simple nonlinear system called a *nonholonomic integrator* [Bro81] is given by the differential equations

$$\dot{x}_1 = u_1, \qquad \dot{x}_2 = u_2, \qquad \dot{x}_3 = x_2 u_1.$$

This system is differentially flat with flat output $z = (x_1, x_3)$. The relationship between the flat variables and the states is given by

$$x_1 = z_1, \qquad x_2 = \dot{x}_3/\dot{x}_1 = \dot{z}_2/\dot{z}_1, \qquad x_3 = z_2. \tag{1.6}$$

Using simple polynomials as our basis functions,

$$\begin{aligned} &\psi_{1,1}(t) = 1, \quad \psi_{1,2}(t) = t, \quad \psi_{1,3}(t) = t^2, \quad \psi_{1,4}(t) = t^3, \\ &\psi_{2,1}(t) = 1 \quad \psi_{2,2}(t) = t, \quad \psi_{2,3}(t) = t^2, \quad \psi_{2,4}(t) = t^3, \end{aligned}$$

the equations for the feasible (flat) trajectory become

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & T & T^2 & T^3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2T & 3T^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T & T^2 & T^3 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2T & 3T^2 \end{bmatrix} \begin{bmatrix} \alpha_{11} \\ \alpha_{12} \\ \alpha_{13} \\ \alpha_{14} \\ \alpha_{21} \\ \alpha_{22} \\ \alpha_{23} \\ \alpha_{24} \end{bmatrix} = \begin{bmatrix} x_{1,0} \\ 1 \\ x_{3,0} \\ x_{2,0} \\ x_{1,f} \\ 1 \\ x_{3,f} \\ x_{2,f} \end{bmatrix}.$$

This is a set of 8 linear equations in 8 variables. It can be shown that the matrix M is full rank when $T \neq 0$ and the system can be solved numerically. ∇

Note that no ODEs need to be integrated in order to compute the feasible trajectories for a differentially flat system (unlike optimal control methods that we will consider in the next chapter, which involve parameterizing the *input* and then solving the ODEs). This is the defining feature of differentially flat systems. The practical implication is that nominal trajectories and inputs that satisfy the equations of motion for a differentially flat system can be computed in a computationally efficient way (solving a set of algebraic equations). Since the flat output functions do not have to obey a set of differential equations, the only constraints that must be satisfied are the initial and final conditions on the endpoints, their tangents, and higher order derivatives. Any other constraints on the system, such as bounds on the inputs, can be transformed into the flat output space and (typically) become limits on the curvature or higher order derivative properties of the curve.

If there is a performance index for the system, this index can be transformed and becomes a functional depending on the flat outputs and their derivatives up to some order. By approximating the performance index we can achieve paths for the system that are suboptimal but still feasible. This approach is often much more appealing than the traditional method of approximating the system (for example by its linearization) and then using the exact performance index, which yields optimal paths but for the wrong system.

In light of the techniques that are available for differentially flat systems, the characterization of flat systems becomes particularly important. Unfortunately, general conditions for flatness are not known, but many important class of nonlinear systems, including feedback linearizable systems, are differential flat. One large class of flat systems are those in "pure feedback form":

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2) \\ \dot{x}_2 &= f_2(x_1, x_2, x_3) \\ &\vdots \\ \dot{x}_n &= f_n(x_1, \dots, x_n, u). \end{aligned}$$

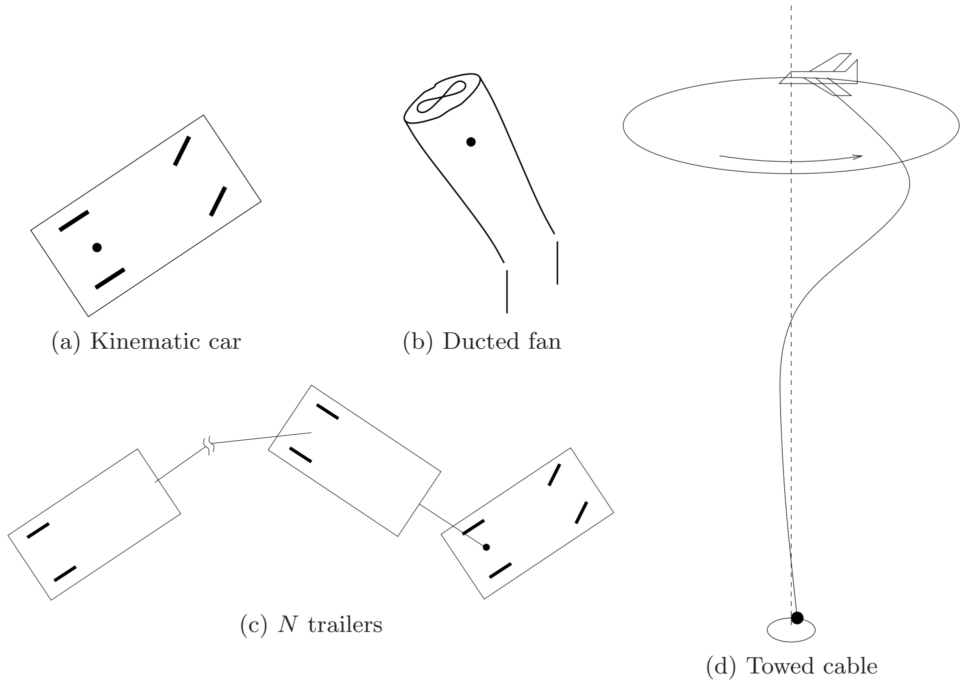


Figure 1.5: Examples of flat systems.

Under certain regularity conditions these systems are differentially flat with output $y = x_1$. These systems have been used for so-called "integrator backstepping" approaches to nonlinear control by Kokotovic et al. [KKM91] and constructive controllability techniques for nonholonomic systems in chained form [vNRM98]. Figure 1.5 shows some additional systems that are differentially flat.

Example 1.3 Vectored thrust aircraft
Consider the dynamics of a planar, vectored thrust flight control system as shown in Figure 1.6. This system consists of a rigid body with body fixed forces and is a simplified model for a vertical take-off and landing aircraft (see Example 2.9 in ÅM08). Let (x, y, θ) denote the position and orientation of the center of mass of the aircraft. We assume that the forces acting on the vehicle consist of a force F_1 perpendicular to the axis of the vehicle acting at a distance r from the center of mass, and a force F_2 parallel to the axis of the vehicle. Let m be the mass of the vehicle, J the moment of inertia, and g the gravitational constant. We ignore aerodynamic forces for the purpose of this example.

The dynamics for the system are

$$\begin{aligned} m\ddot{x} &= F_1 \cos\theta - F_2 \sin\theta, \\ m\ddot{y} &= F_1 \sin\theta + F_2 \cos\theta - mg, \\ J\ddot{\theta} &= rF_1. \end{aligned} \tag{1.7}$$

Martin et al. [MDP94] showed that this system is differentially flat and that one

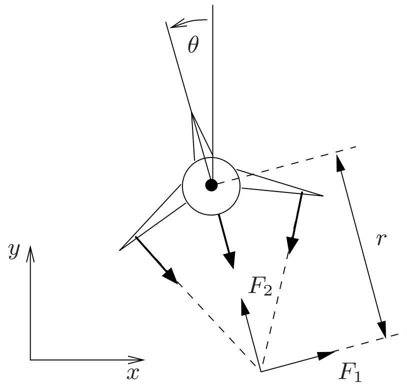


Figure 1.6: Vectored thrust aircraft (from ÅM08). The net thrust on the aircraft can be decomposed into a horizontal force F_1 and a vertical force F_2 acting at a distance r from the center of mass.

set of flat outputs is given by

$$\begin{aligned} z_1 &= x - (J/mr)\sin\theta, \\ z_2 &= y + (J/mr)\cos\theta. \end{aligned} \tag{1.8}$$

Using the system dynamics, it can be shown that

$$\ddot{z}_1 \cos\theta + (\ddot{z}_2 + g)\sin\theta = 0 \tag{1.9}$$

and thus given $z_1(t)$ and $z_2(t)$ we can find $\theta(t)$ except for an ambiguity of π and away from the singularity $\ddot{z}_1 = \ddot{z}_2 + g = 0$. The remaining states and the forces $F_1(t)$ and $F_2(t)$ can then be obtained from the dynamic equations, all in terms of z_1, z_2, and their higher order derivatives. ∇

1.4 Further Reading

The two degree of freedom controller structure introduced in this chapter is described in a bit more detail in ÅM08 (in the context of output feedback control) and a description of some of the origins of this structure are provided in the "Further Reading" section of Chapter 8. Gain scheduling is a classical technique that is often omitted from introductory control texts, but a good description can be found in the survey article by Rugh [Rug90] and the work of Shamma [Sha90]. Differential flatness was originally developed by Fliess, Levin, Martin and Rouchon [FLMR92]. See [Mur97] for a description of the role of flatness in control of mechanical systems and [vNM98, MFHM05] for more information on flatness applied to flight control systems.

Exercises

1.1 (Feasible trajectory for constant reference) Consider a linear input/output system of the form

$$\dot{x} = Ax + Bu, \qquad y = Cx \tag{1.10}$$

in which we wish to track a constant reference r. A feasible (steady state) trajectory for the system is given by solving the equation

$$\begin{bmatrix} 0 \\ r \end{bmatrix} = \begin{bmatrix} A & B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_d \\ u_d \end{bmatrix}$$

for x_d and u_d.

(a) Show that these equations always have a solution as long as the linear system (1.10) is reachable.

(b) In Section 6.2 of ÅM08 we showed that the reference tracking problem could be solved using a control law of the form $u = -Kx + k_r r$. Show that this is equivalent to a two degree of freedom control design using x_d and u_d and give a formula for k_r in terms of x_d and u_d. Show that this formula matches that given in ÅM08.

1.2 A simplified model of the steering control problem is described in Åström and Murray, Example 2.8. The lateral dynamics can be approximated by the linearized dynamics

$$\dot{z} = \begin{bmatrix} 0 & v \\ 0 & 0 \end{bmatrix} z + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \qquad y \quad = z_1,$$

where $z = (y, \theta) \in \mathbb{R}^2$ is the state of the system and v is the speed of the vehicle. Suppose that we wish to track a piecewise constant reference trajectory

$$r = \texttt{square}(2\pi t/20),$$

where `square` is the square wave function in MATLAB. Suppose further that the speed of the vehicle varies according to the formula

$$v = 5 + 3\sin(2\pi t/50).$$

Design and implement a gain-scheduled controller for this system by first designing a state space controller that places the closed loop poles of the system at the roots of $s^2 + 2\zeta\omega_0 s + \omega_0^2$, where $\zeta = 0.7$ and $\omega_0 = 1$. You should design controllers for three different parameter values: $v = 2, 5, 10$. Then use linear interpolation to compute the gain for values of v between these fixed values. Compare the performance of the gain scheduled controller to a simple controller that assumes $v = 5$ for the purpose of the control design (but leaving v time-varying in your simulation).

1.3 (Stability of gain scheduled controllers for slowly varying systems) Consider a nonlinear control system with gain scheduled feedback

$$\dot{e} = f(e, v) \qquad v = k(\mu)e,$$

where $\mu(t) \in \mathbb{R}$ is an externally specified parameter (e.g., the desired trajectory) and $k(\mu)$ is chosen such that the linearization of the closed loop system around the origin is stable for each fixed μ.

Show that if $|\dot\mu|$ is sufficiently small then the equilibrium point is locally asymptotically stable for the full nonlinear, time-varying system. (Hint: find a Lyapunov function of the form $V = x^T P(\mu)x$ based on the linearization of the system dynamics for fixed μ and then show this is a Lyapunov function for the full system.)

1.4 (Flatness of systems in reachable canonical form) Consider a single input system in reachable canonical form [ÅM08, Sec. 6.1]:

$$\begin{aligned}\frac{dx}{dt} &= \begin{bmatrix} -a_1 & -a_2 & -a_3 & \dots & -a_n \\ 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & 1 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} u, \\ y &= \begin{bmatrix} b_1 & b_2 & b_3 & \dots & b_n \end{bmatrix} x + du.\end{aligned} \tag{1.11}$$

Suppose that we wish to find an input u that moves the system from x_0 to x_f. This system is differentially flat with flat output given by $z = x_n$ and hence we can parameterize the solutions by a curve of the form

$$x_n(t) = \sum_{k=0}^{N} \alpha_k t^k, \tag{1.12}$$

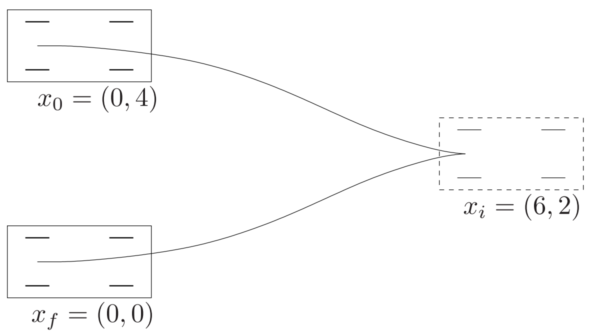
where N is a sufficiently large integer.

(a) Compute the state space trajectory $x(t)$ and input $u(t)$ corresponding to equation (1.12) and satisfying the differential equation (1.11). Your answer should be an equation similar to equation (1.6) for each state x_i and the input u.

(b) Find an explicit input that steers a double integrator system between any two equilibrium points $x_0 \in \mathbb{R}^2$ and $x_f \in \mathbb{R}^2$.

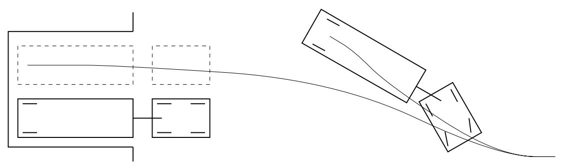
(c) Show that all reachable systems are differentially flat and give a formula for finding the flat output in terms of the dynamics matrix A and control matrix B.

1.5 Consider the lateral control problem for an autonomous ground vehicle as described in Example 1.1 and Section 1.3. Using the fact that the system is differentially flat, find an explicit trajectory that solves the following parallel parking maneuver:



Your solution should consist of two segments: a curve from x_0 to x_i with $v > 0$ and a curve from x_i to x_f with $v < 0$. For the trajectory that you determine, plot the trajectory in the plane (x versus y) and also the inputs v and ϕ as a function of time.

1.6 Consider first the problem of controlling a truck with trailer, as shown in the figure below:



$$\begin{aligned}
\dot{x} &= \cos\theta\, u_1 \\
\dot{y} &= \sin\theta\, u_1 \\
\dot{\phi} &= u_2 \\
\dot{\theta} &= \frac{1}{l}\tan\phi\, u_1 \\
\dot{\theta}_1 &= \frac{1}{d}\cos(\theta-\theta_1)\sin(\theta-\theta_1)u_1,
\end{aligned}$$

The dynamics are given above, where (x, y, θ) is the position and orientation of the truck, ϕ is the angle of the steering wheels, θ_1 is the angle of the trailer, and l and d are the length of the truck and trailer. We want to generate a trajectory for the truck to move it from a given initial position to the loading dock. We ignore the role of obstacles and concentrate on generation of feasible trajectories.

(a) Show that the system is differentially flat using the center of the rear wheels of the trailer as the flat output.

(b) Generate a trajectory for the system that steers the vehicle from an initial condition with the truck and trailer perpendicular to the loading dock into the loading dock.

(c) Write a simulation of the system stabilizes the desired trajectory and demonstrate your two-degree of freedom control system maneuvering from several different initial conditions into the parking space, with either disturbances or modeling errors included in the simulation.

Chapter 2
Optimal Control

This set of notes expands on Chapter 6 of *Feedback Systems* by Åström and Murray (ÅM08), which introduces the concepts of reachability and state feedback. We also expand on topics in Section 7.5 of ÅM08 in the area of feedforward compensation. Beginning with a review of optimization, we introduce the notion of Lagrange multipliers and provide a summary of the Pontryagin's maximum principle. Using these tools we derive the linear quadratic regulator for linear systems and describe its usage.

Prerequisites. Readers should be familiar with modeling of input/output control systems using differential equations, linearization of a system around an equilibrium point and state space control of linear systems, including reachability and eigenvalue assignment. Some familiarity with optimization of nonlinear functions is also assumed.

2.1 Review: Optimization

*O*ptimization refers to the problem of choosing a set of parameters that maximize or minimize a given function. In control systems, we are often faced with having to choose a set of parameters for a control law so that the some performance condition is satisfied. In this chapter we will seek to *optimize* a given specification, choosing the parameters that maximize the performance (or minimize the cost). In this section we review the conditions for optimization of a static function , and then extend this to optimization of trajectories and control laws in the remainder of the chapter. More information on basic techniques in optimization can be found in [Lue97] or the introductory chapter of [LS95].

Consider first the problem of finding the minimum of a smooth function $F : \mathbb{R}^n \to \mathbb{R}$. That is, we wish to find a point $x^* \in \mathbb{R}^n$ such that $F(x^*) \leq F(x)$ for all $x \in \mathbb{R}^n$. A necessary condition for x^* to be a minimum is that the gradient of the function be zero at x^*:

$$\frac{\partial F}{\partial x}(x^*) = 0.$$

The function $F(x)$ is often called a cost function and x^* is the optimal value for x. Figure 2.1 gives a graphical interpretation of the necessary condition for a minimum. Note that these are *not* sufficient conditions; the points x_1 and x_2 and x^* in the figure all satisfy the necessary condition but only one is the (global) minimum.

The situation is more complicated if constraints are present. Let $G_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, k$ be a set of smooth functions with $G_i(x) = 0$ representing the constraints. Suppose that we wish to find $x^* \in \mathbb{R}^n$ such that $G_i(x^*) = 0$ and $F(x^*) \leq F(x)$ for all $x \in \{x \in \mathbb{R}^n : G_i(x) = 0, i = 1, \dots, k\}$. This situation can be

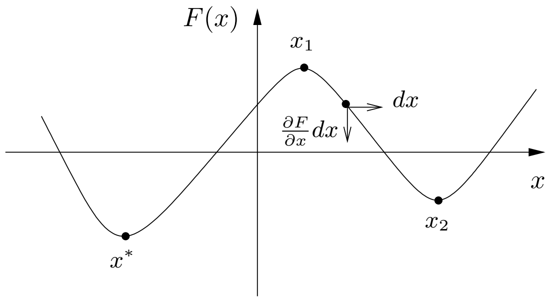
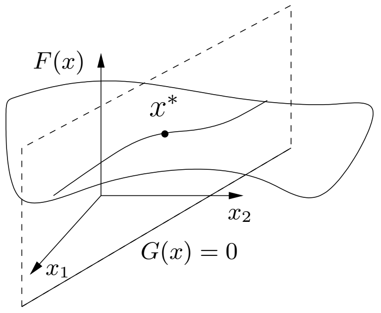
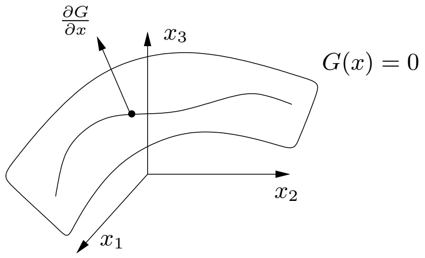


Figure 2.1: Optimization of functions. The minimum of a function occurs at a point where the gradient is zero.



(a) Constrained optimization



(b) Constraint normal vectors

Figure 2.2: Optimization with constraints. (a) We seek a point x^* that minimizes $F(x)$ while lying on the surface $G(x) = 0$ (a line in the x_1x_2 plane). (b) We can parameterize the constrained directions by computing the gradient of the constraint G. Note that $x \in \mathbb{R}^2$ in (a), with the third dimension showing $F(x)$, while $x \in \mathbb{R}^3$ in (b).

visualized as constraining the point to a surface (defined by the constraints) and searching for the minimum of the cost function along this surface, as illustrated in Figure 2.2a.

A necessary condition for being at a minimum is that there are no directions tangent to the constraints that also decrease the cost. Given a constraint function $G(x) = (G_1(x), \dots, G_k(x))$, $x \in \mathbb{R}^n$ we can represent the constraint as a $n - k$ dimensional surface in $\mathbb{R}^n$, as shown in Figure 2.2b. The tangent directions to the surface can be computed by considering small perturbations of the constraint that remain on the surface:

$$G_i(x + \delta x) \approx G_i(x) + \frac{\partial G_i}{\partial x}(x)\delta x = 0. \quad \Longrightarrow \quad \frac{\partial G_i}{\partial x}(x)\delta x = 0,$$

where $\delta x \in \mathbb{R}^n$ is a vanishingly small perturbation. It follows that the normal directions to the surface are spanned by $\partial G_i/\partial x$, since these are precisely the vectors that annihilate an admissible tangent vector δx.

Using this characterization of the tangent and normal vectors to the constraint, a necessary condition for optimization is that the gradient of F is spanned by vectors that are normal to the constraints, so that the only directions that increase the cost violate the constraints. We thus require that there exist scalars λ_i, $i = 1, \dots, k$ such that

$$\frac{\partial F}{\partial x}(x^*) + \sum_{i=1}^{k} \lambda_i \frac{\partial G_i}{\partial x}(x^*) = 0.$$

If we let $G = \begin{bmatrix} G_1 & G_2 & \dots & G_k \end{bmatrix}^T$, then we can write this condition as

$$\frac{\partial F}{\partial x} + \lambda^T \frac{\partial G}{\partial x} = 0 \tag{2.1}$$

the term $\partial F/\partial x$ is the usual (gradient) optimality condition while the term $\partial G/\partial x$ is used to "cancel" the gradient in the directions normal to the constraint.

An alternative condition can be derived by modifying the cost function to incorporate the constraints. Defining $\widetilde{F} = F + \sum \lambda_i G_i$, the necessary condition becomes

$$\frac{\partial \widetilde{F}}{\partial x}(x^*) = 0.$$

The scalars λ_i are called *Lagrange multipliers*. Minimizing $\widetilde{F}$ is equivalent to the optimization given by

$$\min_x \left(F(x) + \lambda^T G(x) \right). \tag{2.2}$$

The variables λ can be regarded as free variables, which implies that we need to choose x such that $G(x) = 0$ in order to insure the cost is minimized. Otherwise, we could choose λ to generate a large cost.

Example 2.1 Two free variables with a constraint
Consider the cost function given by

$$F(x) = F_0 + (x_1 - a)^2 + (x_2 - b)^2,$$

which has an unconstrained minimum at $x = (a, b)$. Suppose that we add a constraint $G(x) = 0$ given by

$$G(x) = x_1 - x_2.$$

With this constraint, we seek to optimize F subject to $x_1 = x_2$. Although in this case we could do this by simple substitution, we instead carry out the more general procedure using Lagrange multipliers.

The augmented cost function is given by

$$\tilde{F}(x) = F_0 + (x_1 - a)^2 + (x_2 - b)^2 + \lambda(x_1 - x_2),$$

where λ is the Lagrange multiplier for the constraint. Taking the derivative of $\tilde{F}$, we have

$$\frac{\partial \tilde{F}}{\partial x} = \begin{bmatrix} 2x_1 - 2a + \lambda & 2x_2 - 2b - \lambda \end{bmatrix}.$$

Setting each of these equations equal to zero, we have that at the minimum

$$x_1^* = a - \lambda/2, \qquad x_2^* = b + \lambda/2.$$

The remaining equation that we need is the constraint, which requires that $x_1^* = x_2^*$. Using these three equations, we see that $\lambda^* = a - b$ and we have

$$x_1^* = \frac{a+b}{2}, \qquad x_2^* = \frac{a+b}{2}.$$

To verify the geometric view described above, note that the gradients of F and G are given by

$$\frac{\partial F}{\partial x} = \begin{bmatrix} 2x_1 - 2a & 2x_2 - 2b \end{bmatrix}, \qquad \frac{\partial G}{\partial x} = \begin{bmatrix} 1 & -1 \end{bmatrix}.$$

At the optimal value of the (constrained) optimization, we have

$$\frac{\partial F}{\partial x} = \begin{bmatrix} b - a & a - b \end{bmatrix}, \qquad \frac{\partial G}{\partial x} = \begin{bmatrix} 1 & -1 \end{bmatrix}.$$

Although the derivative of F is not zero, it is pointed in a direction that is normal to the constraint, and hence we cannot decrease the cost while staying on the constraint surface. ∇

We have focused on finding the minimum of a function. We can switch back and forth between maximum and minimum by simply negating the cost function:

$$\max_x F(x) = \min_x \bigl(-F(x)\bigr)$$

We see that the conditions that we have derived are independent of the sign of F since they only depend on the gradient begin zero in approximate directions. Thus finding x^* that satisfies the conditions corresponds to finding an *extremum* for the function.

Very good software is available for solving optimization problems numerically of this sort. The NPSOL and SNOPT libraries are available in FORTRAN (and C). In MATLAB, the `fmin` function can be used to solve a constrained optimization problem.

2.2 Optimal Control of Systems

Consider now the *optimal control problem*:

$$\min_{u(\cdot)} \int_0^T L(x,u)\,dt + V\bigl(x(T)\bigr)$$

subject to the constraint

$$\dot{x} = f(x,u), \qquad x \in \mathbb{R}^n,\ u \in \mathbb{R}^m.$$

Abstractly, this is a constrained optimization problem where we seek a *feasible trajectory* $(x(t), u(t))$ that minimizes the cost function

$$J(x,u) = \int_0^T L(x,u)\,dt + V\bigl(x(T)\bigr).$$

More formally, this problem is equivalent to the "standard" problem of minimizing a cost function $J(x,u)$ where $(x,u) \in L_2[0,T]$ (the set of square integrable functions) and $h(z) = \dot{x}(t) - f(x(t),u(t)) = 0$ models the dynamics. The term $L(x,u)$ is referred to as the integral cost and $V(x(T))$ is the final (or terminal) cost.

There are many variations and special cases of the optimal control problem. We mention a few here:

Infinite horizon optimal control. If we let $T = \infty$ and set $V = 0$, then we seek to optimize a cost function over all time. This is called the *infinite horizon* optimal control problem, versus the *finite horizon* problem with $T < \infty$. Note that if an infinite horizon problem has a solution with finite cost, then the integral cost term $L(x,u)$ must approach zero as $t \to \infty$.

Linear quadratic (LQ) optimal control. If the dynamical system is linear and the cost function is quadratic, we obtain the *linear quadratic* optimal control problem:

$$\dot{x} = Ax + Bu, \qquad J = \int_0^T \left(x^T Q x + u^T R u\right)\, dt + x^T(T) P_1 x(T).$$

In this formulation, $Q \geq 0$ penalizes state error, $R > 0$ penalizes the input and $P_1 > 0$ penalizes terminal state. This problem can be modified to track a desired trajectory (x_d, u_d) by rewriting the cost function in terms of $(x - x_d)$ and $(u - u_d)$.

Terminal constraints. It is often convenient to ask that the final value of the trajectory, denoted x_f, be specified. We can do this by requiring that $x(T) = x_f$ or by using a more general form of constraint:

$$\psi_i(x(T)) = 0, \qquad i = 1, \dots, q.$$

The fully constrained case is obtained by setting $q = n$ and defining $\psi_i(x(T)) = x_i(T) - x_{i,f}$. For a control problem with a full set of terminal constraints, $V(x(T))$ can be omitted (since its value is fixed).

Time optimal control. If we constrain the terminal condition to $x(T) = x_f$, let the terminal time T be free (so that we can optimize over it) and choose $L(x,u) = 1$, we can find the *time-optimal* trajectory between an initial and final condition. This problem is usually only well-posed if we additionally constrain the inputs u to be bounded.

A very general set of conditions are available for the optimal control problem that captures most of these special cases in a unifying framework. Consider a nonlinear system

$$\begin{aligned} \dot{x} &= f(x,u), \quad & x &= \mathbb{R}^n, \\ x(0) &\text{ given}, \quad & u &\in \Omega \subset \mathbb{R}^m, \end{aligned}$$

where $f(x,u) = (f_1(x,u), \dots f_n(x,u)) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. We wish to minimize a cost function J with terminal constraints:

$$J = \int_0^T L(x,u)\, dt + V(x(T)), \qquad \psi(x(T)) = 0.$$

The function $\psi : \mathbb{R}^n \to \mathbb{R}^q$ gives a set of q terminal constraints. Analogous to the case of optimizing a function subject to constraints, we construct the *Hamiltonian*:

$$H = L + \lambda^T f = L + \sum \lambda_i f_i.$$

The variables λ are functions of time and are often referred to as the *costate variables.* A set of necessary conditions for a solution to be optimal was derived by Pontryagin [PBGM62].

Theorem 2.1 (Maximum Principle)**.** *If (x^*, u^*) is optimal, then there exists $\lambda^*(t) \in \mathbb{R}^n$ and $\nu^* \in \mathbb{R}^q$ such that*

$$\dot{x}_i = \frac{\partial H}{\partial \lambda_i} \qquad -\dot{\lambda}_i = \frac{\partial H}{\partial x_i} \qquad \begin{array}{l} x(0) \text{ given, } \psi(x(T)) = 0 \\ \lambda(T) = \dfrac{\partial V}{\partial x}(x(T)) + \nu^T \dfrac{\partial \psi}{\partial x} \end{array}$$

and

$$H(x^*(t), u^*(t), \lambda^*(t)) \leq H(x^*(t), u, \lambda^*(t)) \quad \text{for all} \quad u \in \Omega$$

The form of the optimal solution is given by the solution of a differential equation with boundary conditions. If $u = \arg\min H(x, u, \lambda)$ exists, we can use this to choose the control law u and solve for the resulting feasible trajectory that minimizes the cost. The boundary conditions are given by the n initial states $x(0)$, the q terminal constraints on the state $\psi(x(T)) = 0$ and the $n - q$ final values for the Lagrange multipliers

$$\lambda(T) = \frac{\partial V}{\partial x}(x(T)) + \nu^T \frac{\partial \psi}{\partial x}.$$

In this last equation, ν is a free variable and so there are n equations in $n + q$ free variables, leaving $n - q$ constraints on $\lambda(T)$. In total, we thus have $2n$ boundary values.

The maximum principle is a very general (and elegant) theorem. It allows the dynamics to be nonlinear and the input to be constrained to lie in a set Ω, allowing the possibility of bounded inputs. If $\Omega = \mathbb{R}^m$ (unconstrained input) and H is differentiable, then a necessary condition for the optimal input is

$$\frac{\partial H}{\partial u} = 0.$$

We note that even though we are *minimizing* the cost, this is still usually called the maximum principle (an artifact of history).

Sketch of proof. We follow the proof given by Lewis and Syrmos [LS95], omitting some of the details required for a fully rigorous proof. We use the method of Lagrange multipliers, augmenting our cost function by the dynamical constraints and the terminal constraints:

$$\begin{aligned} \tilde{J}(x(\cdot), u(\cdot), \lambda(\cdot), \nu) &= J(x, u) + \int_0^T -\lambda^T(t)\big(\dot{x}(t) - f(x, u)\big)\, dt + \nu^T \psi(x(T)) \\ &= \int_0^T \big(L(x, u) - \lambda^T(t)\big(\dot{x}(t) - f(x, u)\big)\, dt \\ &\qquad + V(x(T)) + \nu^T \psi(x(T)). \end{aligned}$$

Note that λ is a function of time, with each $\lambda(t)$ corresponding to the instantaneous constraint imposed by the dynamics. The integral over the interval $[0, T]$ plays the role of the sum of the finite constraints in the regular optimization.

Making use of the definition of the Hamiltonian, the augmented cost becomes

$$\tilde{J}(x(\cdot), u(\cdot), \lambda(\cdot), \nu) = \int_0^T \bigl(H(x,u) - \lambda^T(t)\dot{x}\bigr)\,dt + V(x(T)) + \nu^T \psi(x(T)).$$

We can now "linearize" the cost function around the optimal solution $x(t) = x^*(t) + \delta x(t)$, $u(t) = u^*(t) + \delta u(t)$, $\lambda(t) = \lambda^*(t) + \delta\lambda(t)$ and $\nu = \nu^* + \delta\nu$. Taking T as fixed for simplicity (see [LS95] for the more general case), the incremental cost can be written as

$$\begin{aligned}\delta\tilde{J} &= \tilde{J}(x^* + \delta x, u^* + \delta u, \lambda^* + \delta\lambda, \nu^* + \delta\nu) - \tilde{J}(x^*, u^*, \lambda^*, \nu^*)\\ &\approx \int_0^T \Bigl(\frac{\partial H}{\partial x}\delta x + \frac{\partial H}{\partial u}\delta u - \lambda^T \delta\dot{x} + \Bigl(\frac{\partial H}{\partial \lambda} - \dot{x}^T\Bigr)\delta\lambda\Bigr)\,dt\\ &\qquad + \frac{\partial V}{\partial x}\delta x(T) + \nu^T \frac{\partial \psi}{\partial x}\delta x(T) + \delta\nu^T \psi\bigl(x(T), u(T)\bigr),\end{aligned}$$

where we have omitted the time argument inside the integral and all derivatives are evaluated along the optimal solution.

We can eliminate the dependence on $\delta\dot{x}$ using integration by parts:

$$-\int_0^T \lambda^T \delta\dot{x}\,dt = -\lambda^T(T)\delta x(T) + \lambda^T(0)\delta x(0) + \int_0^T \dot{\lambda}^T \delta x\,dt.$$

Since we are requiring $x(0) = x_0$, the $\delta x(0)$ term vanishes and substituting this into $\delta\tilde{J}$ yields

$$\begin{aligned}\delta\tilde{J} \approx& \int_0^T \Bigl[\Bigl(\frac{\partial H}{\partial x} + \dot{\lambda}^T\Bigr)\delta x + \frac{\partial H}{\partial u}\delta u + \Bigl(\frac{\partial H}{\partial \lambda} - \dot{x}^T\Bigr)\delta\lambda\Bigr]\,dt\\ &+ \Bigl(\frac{\partial V}{\partial x} + \nu^T \frac{\partial \psi}{\partial x} - \lambda^T(T)\Bigr)\delta x(T) + \delta\nu^T \psi\bigl(x(T), u(T)\bigr).\end{aligned}$$

To be optimal, we require $\delta\tilde{J} = 0$ for all δx, δu, $\delta\lambda$ and $\delta\nu$, and we obtain the (local) conditions in the theorem. □

2.3 Examples

To illustrate the use of the maximum principle, we consider a number of analytical examples. Additional examples are given in the exercises.

Example 2.2 Scalar linear system
Consider the optimal control problem for the system

$$\dot{x} = ax + bu, \tag{2.3}$$

where $x = \mathbb{R}$ is a scalar state, $u \in \mathbb{R}$ is the input, the initial state $x(t_0)$ is given, and $a, b \in \mathbb{R}$ are positive constants. We wish to find a trajectory $(x(t), u(t))$ that minimizes the cost function

$$J = \tfrac{1}{2}\int_{t_0}^{t_f} u^2(t)\,dt + \tfrac{1}{2}cx^2(t_f),$$

where the terminal time t_f is given and $c > 0$ is a constant. This cost function balances the final value of the state with the input required to get to that state.

To solve the problem, we define the various elements used in the maximum principle. Our integral and terminal costs are given by

$$L = \tfrac{1}{2}u^2(t) \qquad V = \tfrac{1}{2}cx^2(t_f).$$

We write the Hamiltonian of this system and derive the following expressions for the costate λ:

$$\begin{aligned} H &= L + \lambda f = \tfrac{1}{2}u^2 + \lambda(ax + bu) \\ \dot{\lambda} &= -\frac{\partial H}{\partial x} = -a\lambda, \qquad \lambda(t_f) = \frac{\partial V}{\partial x} = cx(t_f). \end{aligned}$$

This is a final value problem for a linear differential equation in λ and the solution can be shown to be

$$\lambda(t) = cx(t_f)e^{a(t_f - t)}.$$

The optimal control is given by

$$\frac{\partial H}{\partial u} = u + b\lambda = 0 \quad \Rightarrow \quad u^*(t) = -b\lambda(t) = -bcx(t_f)e^{a(t_f-t)}.$$

Substituting this control into the dynamics given by equation (2.3) yields a first-order ODE in x:

$$\dot{x} = ax - b^2cx(t_f)e^{a(t_f-t)}.$$

This can be solved explicitly as

$$x^*(t) = x(t_o)e^{a(t-t_o)} + \frac{b^2c}{2a}x^*(t_f)\left[e^{a(t_f-t)} - e^{a(t+t_f-2t_o)}\right].$$

Setting $t = t_f$ and solving for $x(t_f)$ gives

$$x^*(t_f) = \frac{2a\, e^{a(t_f-t_o)}x(t_o)}{2a - b^2c\left(1 - e^{2a(t_f-t_o)}\right)}$$

and finally we can write

$$u^*(t) = -\frac{2abc\, e^{a(2t_f-t_o-t)}x(t_o)}{2a - b^2c\left(1 - e^{2a(t_f-t_o)}\right)} \tag{2.4}$$

$$x^*(t) = x(t_o)e^{a(t-t_o)} + \frac{b^2c\, e^{a(t_f-t_o)}x(t_o)}{2a - b^2c\left(1 - e^{2a(t_f-t_o)}\right)}\left[e^{a(t_f-t)} - e^{a(t+t_f-2t_o)}\right]. \tag{2.5}$$

We can use the form of this expression to explore how our cost function affects the optimal trajectory. For example, we can ask what happens to the terminal state $x^*(t_f)$ and $c \to \infty$. Setting $t = t_f$ in equation (2.5) and taking the limit we find that

$$\lim_{c\to\infty} x^*(t_f) = 0.$$

∇

Example 2.3 Bang-bang control
The time-optimal control program for a linear system has a particularly simple solution. Consider a linear system with bounded input

$$\dot{x} = Ax + Bu, \qquad |u| \leq 1,$$

and suppose we wish to minimize the time required to move from an initial state x_0 to a final state x_f. Without loss of generality we can take $x_f = 0$. We choose the cost functions and terminal constraints to satisfy

$$J = \int_0^T 1\,dt, \qquad \psi(x(T)) = x(T).$$

To find the optimal control, we form the Hamiltonian

$$H = 1 + \lambda^T(Ax + Bu) = 1 + (\lambda^T A)x + (\lambda^T B)u.$$

Now apply the conditions in the maximum principle:

$$\begin{aligned}\dot{x} &= \frac{\partial H}{\partial \lambda} = Ax + Bu\\ -\dot{\lambda} &= \frac{\partial H}{\partial x} = A^T\lambda\\ u &= \text{arg min } H = -\text{sgn}(\lambda^T B)\end{aligned}$$

The optimal solution always satisfies this equation (since the maximum principle gives a necessary condition) with $x(0) = x_0$ and $x(T) = 0$. It follows that the input is always either $+1$ or -1, depending on $\lambda^T B$. This type of control is called "bang-bang" control since the input is always on one of its limits. If $\lambda^T(t)B = 0$ then the control is not well defined, but if this is only true for a specific time instant (e.g., $\lambda^T(t)B$ crosses zero) then the analysis still holds. ∇

2.4 Linear Quadratic Regulators

In addition to its use for computing optimal, feasible trajectories for a system, we can also use optimal control theory to design a feedback law $u = \alpha(x)$ that stabilizes a given equilibrium point. Roughly speaking, we do this by continuously re-solving the optimal control problem from our current state $x(t)$ and applying the resulting input $u(t)$. Of course, this approach is impractical unless we can solve explicitly for the optimal control and somehow rewrite the optimal control as a function of the current state in a simple way. In this section we explore exactly this approach for the linear quadratic optimal control problem.

We begin with the the finite horizon, linear quadratic regulator (LQR) problem, given by

$$\begin{aligned}\dot{x} &= Ax + Bu, \qquad x \in \mathbb{R}^n, u \in \mathbb{R}^n, x_0 \text{ given},\\ \tilde{J} &= \frac{1}{2}\int_0^T \left(x^T Q_x x + u^T Q_u u\right)\,dt + \frac{1}{2}x^T(T)P_1 x(T),\end{aligned}$$

where $Q_x \geq 0$, $Q_u > 0$, $P_1 \geq 0$ are symmetric, positive (semi-) definite matrices. Note the factor of $\frac{1}{2}$ is usually left out, but we included it here to simplify the derivation. (The optimal control will be unchanged if we multiply the entire cost function by 2.)

To find the optimal control, we apply the maximum principle. We being by computing the Hamiltonian H:

$$H = \frac{1}{2}x^T Q_x x + \frac{1}{2}u^T Q_u u + \lambda^T(Ax + Bu).$$

Applying the results of Theorem 2.1, we obtain the necessary conditions

$$\begin{aligned} \dot{x} &= \left(\frac{\partial H}{\partial \lambda}\right)^T && = Ax + Bu && x(0) = x_0 \\ -\dot{\lambda} &= \left(\frac{\partial H}{\partial x}\right)^T && = Q_x x + A^T\lambda && \lambda(T) = P_1 x(T) \\ 0 &= \frac{\partial H}{\partial u} && = Q_u u + \lambda^T B. \end{aligned} \tag{2.6}$$

The last condition can be solved to obtain the optimal controller

$$u = -Q_u^{-1} B^T \lambda,$$

which can be substituted into the dynamic equation (2.6) To solve for the optimal control we must solve a *two point boundary value problem* using the initial condition $x(0)$ and the final condition $\lambda(T)$. Unfortunately, it is very hard to solve such problems in general.

Given the linear nature of the dynamics, we attempt to find a solution by setting $\lambda(t) = P(t)x(t)$ where $P(t) \in \mathbb{R}^{n \times n}$. Substituting this into the necessary condition, we obtain

$$\begin{aligned} \dot{\lambda} = \dot{P}x + P\dot{x} &= \dot{P}x + P(Ax - BQ_u^{-1}B^T P)x, \\ \implies \qquad -\dot{P}x - PAx + PBQ_u^{-1}BPx &= Q_x x + A^T Px. \end{aligned}$$

This equation is satisfied if we can find $P(t)$ such that

$$-\dot{P} = PA + A^T P - PBQ_u^{-1}B^T P + Q_x, \qquad P(T) = P_1. \tag{2.7}$$

This is a *matrix differential equation* that defines the elements of $P(t)$ from a final value $P(T)$. Solving it is conceptually no different than solving the initial value problem for vector-valued ordinary differential equations, except that we must solve for the individual elements of the matrix $P(t)$ backwards in time. Equation (2.7) is called the *Riccati ODE*.

An important property of the solution to the optimal control problem when written in this form is that $P(t)$ can be solved without knowing either $x(t)$ or $u(t)$. This allows the two point boundary value problem to be separated into first solving a final-value problem and then solving a time-varying initial value problem. More specifically, given $P(t)$ satisfying equation (2.7), we can apply the optimal input

$$u(t) = -Q_u^{-1} B^T P(t)x.$$

and then solve the original dynamics of the system forward in time from the initial condition $x(0) = x_0$. Note that this is a (time-varying) *feedback* control that describes how to move from *any* state to the origin.

An important variation of this problem is the case when we choose $T = \infty$ and eliminate the terminal cost (set $P_1 = 0$). This gives us the cost function

$$J = \int_0^\infty (x^T Q_x x + u^T Q_u u)\, dt. \tag{2.8}$$

Since we do not have a terminal cost, there is no constraint on the final value of λ or, equivalently, $P(t)$. We can thus seek to find a constant P satisfying equation (2.7). In other words, we seek to find P such that

$$PA + A^T P - PBQ_u^{-1}B^T P + Q_x = 0. \tag{2.9}$$

This equation is called the *algebraic Riccati equation*. Given a solution, we can choose our input as

$$u = -Q_u^{-1}B^T Px.$$

This represents a *constant gain* $K = Q_u^{-1}B^T P$ where P is the solution of the algebraic Riccati equation.

The implications of this result are interesting and important. First, we notice that if $Q_x > 0$ and the control law corresponds to a finite minimum of the cost, then we must have that $\lim_{t\to\infty} x(t) = 0$, otherwise the cost will be unbounded. This means that the optimal control for moving from any state x to the origin can be achieved by applying a feedback $u = -Kx$ for K chosen as described as above and letting the system evolve in closed loop. More amazingly, the gain matrix K can be written in terms of the solution to a (matrix) quadratic equation (2.9). This quadratic equation can be solved numerically: in MATLAB the command `K = lqr(A, B, Qx, Qu)` provides the optimal feedback compensator.

In deriving the optimal quadratic regulator, we have glossed over a number of important details. It is clear from the form of the solution that we must have $Q_u > 0$ since its inverse appears in the solution. We would typically also have $Q_x > 0$ so that the integral cost is only zero when $x = 0$, but in some instances we might only care about certain states, which would imply that $Q_x \geq 0$. For this case, if we let $Q_x = H^T H$ (always possible), our cost function becomes

$$J = \int_0^\infty x^T H^T Hx + u^T Q_u u\, dt = \int_0^\infty \|Hx\|^2 + u^T Q_u u\, dt.$$

A technical condition for the optimal solution to exist is that the pair (A, H) be *detectable* (implied by observability). This makes sense intuitively by considering $y = Hx$ as an output. If y is not observable then there may be non-zero initial conditions that produce no output and so the cost would be zero. This would lead to an ill-conditioned problem and hence we will require that $Q_x \geq 0$ satisfy an appropriate observability condition.

We summarize the main results as a theorem.

Theorem 2.2. *Consider a linear control system with quadratic cost:*

$$\dot{x} = Ax + Bu, \qquad J = \int_0^\infty x^T Q_x x + u^T Q_u u\, dt.$$

Assume that the system defined by (A, B) is reachable, $Q_x = Q_x^T \geq 0$ and $Q_u = Q_u^T > 0$. Further assume that $Q_u = H^T H$ and that the linear system with dynamics matrix A and output matrix H is observable. Then the optimal controller satisfies

$$u = -Q_u^{-1} B^T P x, \qquad PA + A^T P - PBQ_u^{-1} B^T P = -Q_x,$$

and the minimum cost from initial condition $x(0)$ is given by $J^* = x^T(0) P x(0)$.

The basic form of the solution follows from the necessary conditions, with the theorem asserting that a constant solution exists for $T = \infty$ when the additional conditions are satisfied. The full proof can be found in standard texts on optimal control, such as Lewis and Syrmos [LS95] or Athans and Falb [AF06]. A simplified version, in which we first assume the optimal control is linear, is left as an exercise.

Example 2.4 Optimal control of a double integrator
Consider a double integrator system

$$\frac{dx}{dt} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

with quadratic cost given by

$$Q_x = \begin{bmatrix} q^2 & 0 \\ 0 & 0 \end{bmatrix}, \qquad Q_u = 1.$$

The optimal control is given by the solution of matrix Riccati equation (2.9). Let P be a symmetric positive definite matrix of the form

$$P = \begin{bmatrix} a & b \\ b & c \end{bmatrix}.$$

Then the Riccati equation becomes

$$\begin{bmatrix} -b^2 + q^2 & a - bc \\ a - bc & 2b - c^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

which has solution

$$P = \begin{bmatrix} \sqrt{2q^3} & q \\ q & \sqrt{2q} \end{bmatrix}.$$

The controller is given by

$$K = Q_u^{-1} B^T P = [1/q \quad \sqrt{2/q}].$$

The feedback law minimizing the given cost function is then $u = -Kx$.

To better understand the structure of the optimal solution, we examine the eigenstructure of the closed loop system. The closed-loop dynamics matrix is given by

$$A_{cl} = A - BK = \begin{bmatrix} 0 & 1 \\ -1/q & -\sqrt{2/q} \end{bmatrix}.$$

The characteristic polynomial of this matrix is

$$\lambda^2 + \sqrt{\frac{2}{q}}\lambda + \frac{1}{q}.$$

Comparing this to $\lambda^2 + 2\zeta\omega_0\lambda + \omega_0^2$, we see that

$$\omega_0 = \sqrt{\frac{1}{q}}, \qquad \zeta = \frac{1}{\sqrt{2}}.$$

Thus the optimal controller gives a closed loop system with damping ratio $\zeta = 0.707$, giving a good tradeoff between rise time and overshoot (see ÅM08). ∇

2.5 Choosing LQR weights

One of the key questions in LQR design is how to choose the weights Q_x and Q_u. To choose specific values for the cost function weights Q_x and Q_u, we must use our knowledge of the system we are trying to control. A particularly simple choice is to use diagonal weights

$$Q_x = \begin{bmatrix} q_1 & & 0 \\ & \ddots & \\ 0 & & q_n \end{bmatrix}, \qquad Q_u = \begin{bmatrix} \rho_1 & & 0 \\ & \ddots & \\ 0 & & \rho_n \end{bmatrix}.$$

For this choice of Q_x and Q_u, the individual diagonal elements describe how much each state and input (squared) should contribute to the overall cost. Hence, we can take states that should remain small and attach higher weight values to them. Similarly, we can penalize an input versus the states and other inputs through choice of the corresponding input weight ρ_j.

Choosing the individual weights for the (diagonal) elements of the Q_x and Q_u matrix can be done by deciding on a weighting of the errors from the individual terms. Bryson and Ho [BH75] have suggested the following method for choosing the matrices Q_x and Q_u in equation (2.8): (1) choose q_i and ρ_j as the inverse of the square of the maximum value for the corresponding x_i or u_j; (2) modify the elements to obtain a compromise among response time, damping and control effort. This second step can be performed by trial and error.

It is also possible to choose the weights such that only a given subset of variable are considered in the cost function. Let $z = Hx$ be the output we want to keep small and verify that (A, H) is observable. Then we can use a cost function of the form

$$Q_x = H^T H \qquad Q_u = \rho I.$$

The constant ρ allows us to trade off $\|z\|^2$ versus $\rho\|u\|^2$.

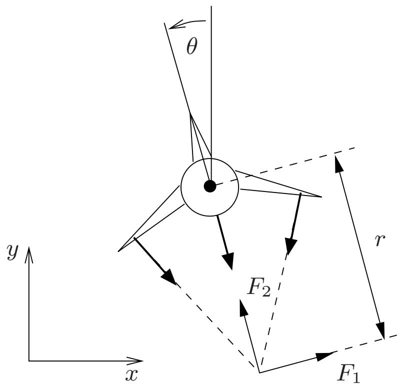
We illustrate the various choices through an example application.

Example 2.5 Thrust vectored aircraft

Consider the thrust vectored aircraft example introduced in ÅM08, Example 2.9. The system is shown in Figure 2.3, reproduced from ÅM08. The linear quadratic regulator problem was illustrated in Example 6.8, where the weights were chosen as $Q_x = I$ and $Q_u = \rho I$. Figure 2.4 reproduces the step response for this case.

A more physically motivated weighted can be computing by specifying the comparable errors in each of the states and adjusting the weights accordingly. Suppose, for example that we consider a 1 cm error in x, a 10 cm error in y and a 5° error in θ

(a) Harrier "jump jet"



(b) Simplified model

Figure 2.3: Vectored thrust aircraft. The Harrier AV-8B military aircraft (a) redirects its engine thrust downward so that it can "hover" above the ground. Some air from the engine is diverted to the wing tips to be used for maneuvering. As shown in (b), the net thrust on the aircraft can be decomposed into a horizontal force F_1 and a vertical force F_2 acting at a distance r from the center of mass.

to be equivalently bad. In addition, we wish to penalize the forces in the sidewards direction since these results in a loss in efficiency. This can be accounted for in the LQR weights be choosing

$$Q_x = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2\pi/9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad Q_u = 0.1 \times \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix}.$$

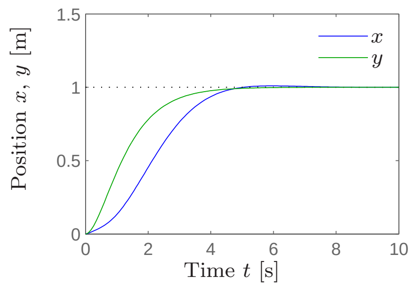
The results of this choice of weights are shown in Figure 2.5. ∇

2.6 Advanced Topics

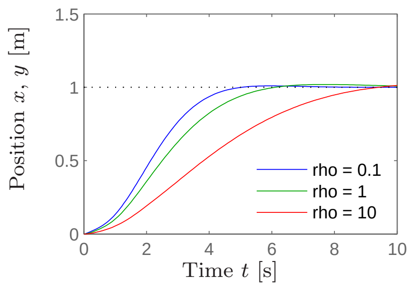
In this section we briefly touch on some related topics in optimal control, with reference to more detailed treatments where appropriate.

Singular extremals. The necessary conditions in the maximum principle enforce the constraints through the of the Lagrange multipliers $\lambda(t)$. In some instances, we can get an extremal curve that has one or more of the λ's identically equal to zero. This corresponds to a situation in which the constraint is satisfied strictly through the minimization of the cost function and does not need to be explicitly enforced. We illustrate this case through an example.

Example 2.6 Nonholonomic integrator
Consider the minimum time optimization problem for the nonholonomic integrator introduced in Example 1.2 with input constraints $|u_i| \leq 1$. The Hamiltonian for the

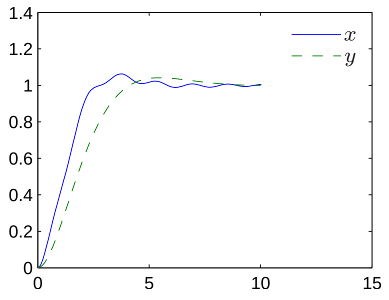


(a) Step response in x and y

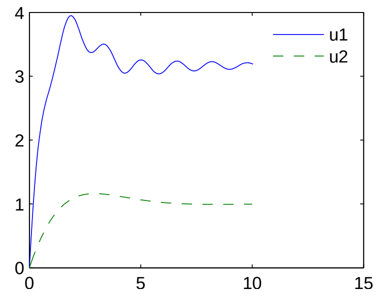


(b) Effect of control weight ρ

Figure 2.4: Step response for a vectored thrust aircraft. The plot in (a) shows the x and y positions of the aircraft when it is commanded to move 1 m in each direction. In (b) the x motion is shown for control weights $\rho = 1$, 10^2, 10^4. A higher weight of the input term in the cost function causes a more sluggish response.



(a) Step response in x and y



(b) Inputs for the step response

Figure 2.5: Step response for a vector thrust aircraft using physically motivated LQR weights (a). The rise time for x is much faster than in Figure 2.4a, but there is a small oscillation and the inputs required are quite large (b).

system is given by

$$H = 1 + \lambda_1 u_1 + \lambda_2 u_2 + \lambda_3 x_2 u_1$$

and the resulting equations for the Lagrange multipliers are

$$\dot{\lambda}_1 = 0, \qquad \dot{\lambda}_2 = \lambda_3 x_2, \qquad \dot{\lambda}_3 = 0. \tag{2.10}$$

It follows from these equations that λ_1 and λ_3 are constant. To find the input u corresponding to the extremal curves, we see from the Hamiltonian that

$$u_1 = -\text{sgn}(\lambda_1 + \lambda_3 x_2 u_1), \qquad u_2 = -\text{sgn}\lambda_2.$$

These equations are well-defined as long as the arguments of $\text{sgn}(\cdot)$ are non-zero and we get switching of the inputs when the arguments pass through 0.

An example of an abnormal extremal is the optimal trajectory between $x_0 = (0, 0, 0)$ to $x_f = (\rho, 0, 0)$ where $\rho > 0$. The minimum time trajectory is clearly given by moving on a straight line with $u_1 = 1$ and $u_2 = 0$. This extremal satisfies the necessary conditions but with $\lambda_2 \equiv 0$, so that the "constraint" that $\dot{x}_2 = u_2$ is not strictly enforced through the Lagrange multipliers. ∇

2.7 Further Reading

There are a number of excellent books on optimal control. One of the first (and best) is the book by Pontryagin et al. [PBGM62]. During the 1960s and 1970s a number of additional books were written that provided many examples and served as standard textbooks in optimal control classes. Athans and Falb [AF06] and Bryson and Ho [BH75] are two such texts. A very elegant treatment of optimal control from the point of view of optimization over general linear spaces is given by Luenberger [Lue97]. Finally, a modern engineering textbook that contains a very compact and concise derivation of the key results in optimal control is the book by Lewis and Syrmos [LS95].

Exercises

2.1 (a) Let $G_1, G_2, \dots, G_k$ be a set of row vectors on a $\mathbb{R}^n$. Let F be another row vector on $\mathbb{R}^n$ such that for every $x \in \mathbb{R}^n$ satisfying $G_i x = 0$, $i = 1, \dots, k$, we have $Fx = 0$. Show that there are constants $\lambda_1, \lambda_2, \dots, \lambda_k$ such that

$$F = \sum_{i=1}^{k} \lambda_k G_k.$$

(b) Let $x^* \in \mathbb{R}^n$ be an the extremal point (maximum or minimum) of a function f subject to the constraints $g_i(x) = 0$, $i = 1, \dots, k$. Assuming that the gradients $\partial g_i(x^*)/\partial x$ are linearly independent, show that there are k scalers λ_k, $i = 1, \dots, n$ such that the function

$$\tilde{f}(x) = f(x) + \sum_{i=1}^{n} \lambda_i g_i(x)$$

has an extremal point at x^*.

2.2 Consider the following control system

$$\begin{aligned}\dot{q} &= u \\ \dot{Y} &= qu^T - uq^T\end{aligned}$$

where $u \in \mathbb{R}^m$ and $Y \in \mathbb{R}^{m \times \ni}$ is a skew symmetric matrix, $Y^T = Y$.

(a) For the fixed end point problem, derive the form of the optimal controller minimizing the following integral

$$\frac{1}{2}\int_0^1 u^T u \, dt.$$

(b) For the boundary conditions $q(0) = q(1) = 0$, $Y(0) = 0$ and

$$Y(1) = \begin{bmatrix} 0 & -y_3 & y_2 \\ y_3 & 0 & -y_1 \\ -y_2 & y_1 & 0 \end{bmatrix}$$

for some $y \in \mathbb{R}^3$, give an explicit formula for the optimal inputs u.

(c) (Optional) Find the input u to steer the system from $(0, 0)$ to $(0, \tilde{Y}) \in \mathbb{R}^m \times \mathbb{R}^{m \times m}$ where $\tilde{Y}^T = -\tilde{Y}$.

(Hint: if you get stuck, there is a paper by Brockett on this problem.)

2.3 In this problem, you will use the maximum principle to show that the shortest path between two points is a straight line. We model the problem by constructing a control system

$$\dot{x} = u,$$

where $x \in \mathbb{R}^2$ is the position in the plane and $u \in \mathbb{R}^2$ is the velocity vector along the curve. Suppose we wish to find a curve of minimal length connecting $x(0) = x_0$ and $x(1) = x_f$. To minimize the length, we minimize the integral of the velocity along the curve,

$$J = \int_0^1 \|\dot{x}\| \, dt = \int_0^1 \sqrt{\dot{x}^T \dot{x}} \, dt,$$

subject to to the initial and final state constraints. Use the maximum principle to show that the minimal length path is indeed a straight line at maximum velocity. (Hint: try minimizing using the integral cost $\dot{x}^T \dot{x}$ first and then show this also optimizes the optimal control problem with integral cost $\|\dot{x}\|$.)

2.4 Consider the optimal control problem for the system

$$\dot{x} = -ax + bu,$$

where $x = \mathbb{R}$ is a scalar state, $u \in \mathbb{R}$ is the input, the initial state $x(t_0)$ is given, and $a, b \in \mathbb{R}$ are positive constants. (Note that this system is not quite the same as the one in Example 2.2.) The cost function is given by

$$J = \tfrac{1}{2} \int_{t_0}^{t_f} u^2(t) \, dt + \tfrac{1}{2} c x^2(t_f),$$

where the terminal time t_f is given and c is a constant.

(a) Solve explicitly for the optimal control $u^*(t)$ and the corresponding state $x^*(t)$ in terms of t_0, t_f, $x(t_0)$ and t and describe what happens to the terminal state $x^*(t_f)$ as $c \to \infty$.

(b) Show that the system is differentially flat with appropriate choice of output(s) and compute the state and input as a function of the flat output(s).

(c) Using the polynomial basis $\{t^k, k = 0, \dots, M-1\}$ with an appropriate choice of M, solve for the (non-optimal) trajectory between $x(t_0)$ and $x(t_f)$. Your answer should specify the explicit input $u_d(t)$ and state $x_d(t)$ in terms of t_0, t_f, $x(t_0)$, $x(t_f)$ and t.

(d) Let $a = 1$ and $c = 1$. Use your solution to the optimal control problem and the flatness-based trajectory generation to find a trajectory between $x(0) = 0$ and $x(1) = 1$. Plot the state and input trajectories for each solution and compare the costs of the two approaches.

(e) (Optional) Suppose that we choose more than the minimal number of basis functions for the differentially flat output. Show how to use the additional degrees of freedom to minimize the cost of the flat trajectory and demonstrate that you can obtain a cost that is closer to the optimal.

2.5 Repeat Exercise 2.4 using the system

$$\dot{x} = -ax^3 + bu.$$

For part (a) you need only write the conditions for the optimal cost.

2.6 Consider the problem of moving a two-wheeled mobile robot (e.g., a Segway) from one position and orientation to another. The dynamics for the system is given by the nonlinear differential equation

$$\dot{x} = \cos\theta\, v, \qquad \dot{y} \;= \sin\theta\, v, \qquad \dot{\theta} = \omega,$$

where (x, y) is the position of the rear wheels, θ is the angle of the robot with respect to the x axis, v is the forward velocity of the robot and ω is spinning rate. We wish to choose an input (v, ω) that minimizes the time that it takes to move between two configurations (x_0, y_0, θ_0) and (x_f, y_f, θ_f), subject to input constraints $|v| \leq L$ and $|\omega| \leq M$.

Use the maximum principle to show that any optimal trajectory consists of segments in which the robot is traveling at maximum velocity in either the forward or reverse direction, and going either straight, hard left ($\omega = -M$) or hard right ($\omega = +M$).

Note: one of the cases is a bit tricky and cannot be completely proven with the tools we have learned so far. However, you should be able to show the other cases and verify that the tricky case is possible.

2.7 Consider a linear system with input u and output y and suppose we wish to minimize the quadratic cost function

$$J = \int_0^\infty \left(y^T y + \rho u^T u\right)\, dt.$$

Show that if the corresponding linear system is observable, then the closed loop system obtained by using the optimal feedback $u = -Kx$ is guaranteed to be stable.

2.8 Consider the system transfer function

$$H(s) = \frac{s+b}{s(s+a)}, \qquad a, b > 0$$

with state space representation

$$\begin{aligned} \dot{x} &= \begin{bmatrix} 0 & 1 \\ 0 & -a \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \\ y &= \begin{bmatrix} b & 1 \end{bmatrix} x \end{aligned}$$

and performance criterion

$$V = \int_0^\infty (x_1^2 + u^2) dt.$$

(a) Let

$$P = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix},$$

with $p_{12} = p_{21}$ and $P > 0$ (positive definite). Write the steady state Riccati equation as a system of four explicit equations in terms of the elements of P and the constants a and b.

(b) Find the gains for the optimal controller assuming the full state is available for feedback.

(c) Find the closed loop natural frequency and damping ratio.

2.9 Consider the optimal control problem for the system

$$\dot{x} = ax + bu \qquad J = \tfrac{1}{2} \int_{t_0}^{t_f} u^2(t)\, dt + \tfrac{1}{2} c x^2(t_f),$$

where $x \in \mathbb{R}$ is a scalar state, $u \in \mathbb{R}$ is the input, the initial state $x(t_0)$ is given, and $a, b \in \mathbb{R}$ are positive constants. We take the terminal time t_f as given and let $c > 0$ be a constant that balances the final value of the state with the input required to get to that position. The optimal trajectory is derived in Example 2.2.

Now consider the infinite horizon cost

$$J = \tfrac{1}{2} \int_{t_0}^{\infty} u^2(t)\, dt$$

with $x(t)$ at $t = \infty$ constrained to be zero.

(a) Solve for $u^*(t) = -bPx^*(t)$ where P is the positive solution corresponding to the algebraic Riccati equation. Note that this gives an explicit feedback law ($u = -bPx$).

(b) Plot the state solution of the finite time optimal controller for the following parameter values

$$a = 2, \qquad b = 0.5, \qquad x(t_0) = 4,$$
$$c = 0.1,\ 10, \qquad t_f = 0.5,\ 1,\ 10.$$

(This should give you a total of 6 curves.) Compare these to the infinite time optimal control solution. Which finite time solution is closest to the infinite time solution? Why?

2.10 Consider the lateral control problem for an autonomous ground vehicle from Example 1.1. We assume that we are given a reference trajectory $r = (x_d, y_d)$ corresponding to the desired trajectory of the vehicle. For simplicity, we will assume that we wish to follow a straight line in the x direction at a constant velocity $v_d > 0$ and hence we focus on the y and θ dynamics:

$$\dot{y} = \sin\theta\, v_d, \qquad \dot{\theta} \;\; = \frac{1}{l} \tan\phi\, v_d.$$

We let $v_d = 10$ m/s and $l = 2$ m.

(a) Design an LQR controller that stabilizes the position y to $y_d = 0$. Plot the step and frequency response for your controller and determine the overshoot, rise time, bandwidth and phase margin for your design. (Hint: for the frequency domain specifications, break the loop just before the process dynamics and use the resulting SISO loop transfer function.)

(b) Suppose now that $y_d(t)$ is not identically zero, but is instead given by $y_d(t) = r(t)$. Modify your control law so that you track $r(t)$ and demonstrate the performance of your controller on a "slalom course" given by a sinusoidal trajectory with magnitude 1 meter and frequency 1 Hz.

Chapter 3
Receding Horizon Control

(with J. E. Hauser and A. Jadbabaie)

This set of notes builds on the previous two chapters and explores the use of online optimization as a tool for control of nonlinear control. We begin with a high-level discussion of optimization-based control, refining some of the concepts initially introduced in Chapter 1. We then describe the technique of receding horizon control (RHC), including a proof of stability for a particular form of receding horizon control that makes use of a control Lyapunov function as a terminal cost. We conclude the chapter with a detailed design example, in which we can explore some of the computational tradeoffs in optimization-based control.

Prerequisites. Readers should be familiar with the concepts of trajectory generation and optimal control as described in Chapters 1 and 2. For the proof of stability for the receding horizon controller that we present, familiarity with Lyapunov stability analysis at the level given in ÅM08, Chapter 4 (Dynamic Behavior) is assumed.

The material in this chapter is based on part on joint work with John Hauser and Ali Jadbabaie [MHJ$^+$03].

3.1 Optimization-Based Control

Optimization-based control refers to the use of online, optimal trajectory generation as a part of the feedback stabilization of a (typically nonlinear) system. The basic idea is to use a *receding horizon control* technique: a (optimal) feasible trajectory is computed from the current position to the desired position over a finite time T horizon, used for a short period of time $\delta < T$, and then recomputed based on the new system state starting at time $t + \delta$ until time $t + T + \delta$. Development and application of receding horizon control (also called model predictive control, or MPC) originated in process control industries where the processes being controlled are often sufficiently slow to permit its implementation. An overview of the evolution of commercially available MPC technology is given in [QB97] and a survey of the state of stability theory of MPC is given in [MRRS00].

Design approach

The basic philosophy that we propose is illustrated in Figure 3.1. We begin with a nonlinear system, including a description of the constraint set. We linearize this system about a representative equilibrium point and perform a linear control design using standard control design tools. Such a design can provide provably robust performance around the equilibrium point and, more importantly, allows the designer

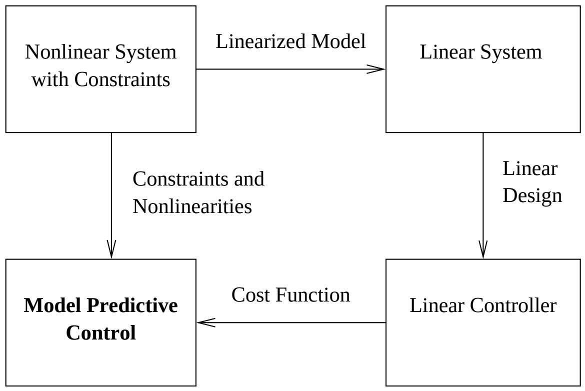


Figure 3.1: Optimization-based control approach.

to meet a wide variety of formal and informal performance specifications through experience and the use of sophisticated linear design tools.

The resulting linear control law then serves as a *specification* of the desired control performance for the entire nonlinear system. We convert the control law specification into a receding horizon control formulation, chosen such that for the linearized system, the receding horizon controller gives comparable performance. However, because of its use of optimization tools that can handle nonlinearities and constraints, the receding horizon controller is able to provide the desired performance over a much larger operating envelope than the controller design based just on the linearization. Furthermore, by choosing cost formulations that have certain properties, we can provide proofs of stability for the full nonlinear system and, in some cases, the constrained system.

The advantage of the proposed approach is that it exploits the power of humans in designing sophisticated control laws in the absence of constraints with the power of computers to rapidly compute trajectories that optimize a given cost function in the presence of constraints. New advances in online trajectory generation serve as an enabler for this approach and their demonstration on representative flight control experiments shows their viability [MFHM05]. This approach can be extended to existing nonlinear paradigms as well, as we describe in more detail below.

An advantage of optimization-based approaches is that they allow the potential for online customization of the controller. By updating the model that the optimization uses to reflect the current knowledge of the system characteristics, the controller can take into account changes in parameters values or damage to sensors or actuators. In addition, environmental models that include dynamic constraints can be included, allowing the controller to generate trajectories that satisfy complex operating conditions. These modifications allow for many state- and environment-dependent uncertainties to including the receding horizon feedback loop, providing potential robustness with respect to those uncertainties.

A number of approaches in receding horizon control employ the use of terminal state equality or inequality constraints, often together with a terminal cost, to ensure closed loop stability. In Primbs et al. [PND99], aspects of a stability-guaranteeing, global control Lyapunov function (CLF) were used, via state and control constraints, to develop a stabilizing receding horizon scheme. Many of the nice characteristics of the CLF controller together with better cost performance were realized. Unfortunately, a global control Lyapunov function is rarely available and often not possible.

Motivated by the difficulties in solving constrained optimal control problems, researchers have developed an alternative receding horizon control strategy for the stabilization of nonlinear systems [JYH01]. In this approach, closed loop stability is ensured through the use of a terminal cost consisting of a control Lyapunov function (defined later) that is an incremental upper bound on the optimal cost to go. This terminal cost eliminates the need for terminal constraints in the optimization and gives a dramatic speed-up in computation. Also, questions of existence and regularity of optimal solutions (very important for online optimization) can be dealt with in a rather straightforward manner.

Inverse Optimality

The philosophy presented here relies on the synthesis of an optimal control problem from specifications that are embedded in an externally generated controller design. This controller is typically designed by standard classical control techniques for a nominal process, absent constraints. In this framework, the controller's performance, stability and robustness specifications are translated into an equivalent optimal control problem and implemented in a receding horizon fashion.

One central question that must be addressed when considering the usefulness of this philosophy is: *Given a control law, how does one find an equivalent optimal control formulation?* The paper by Kalman [Kal64] lays a solid foundation for this class of problems, known as *inverse optimality.* In this paper, Kalman considers the class of linear time-invariant (LTI) processes with full-state feedback and a single input variable, with an associated cost function that is quadratic in the input and state variables. These assumptions set up the well-known linear quadratic regulator (LQR) problem, by now a staple of optimal control theory.

In Kalman's paper, the mathematical framework behind the LQR problem is laid out, and necessary and sufficient algebraic criteria for optimality are presented in terms of the algebraic Riccati equation, as well as in terms of a condition on the return difference of the feedback loop. In terms of the LQR problem, the task of synthesizing the optimal control problem comes down to finding the integrated cost weights Q_x and Q_u given only the dynamical description of the process represented by matrices A and B and of the feedback controller represented by K. Kalman delivers a particularly elegant frequency characterization of this map [Kal64], which we briefly summarize here.

We consider a linear system

$$\dot{x} = Ax + Bu \qquad x \in \mathbb{R}^n, u \in \mathbb{R}^m \tag{3.1}$$

with state x and input u. We consider only the single input, single output case for now ($m = 1$). Given a control law

$$u = Kx$$

we wish to find a cost functional of the form

$$J = \int_0^T x^T Q_x x + u^T Q_u u \, dt + x^T(T) P_T x(T) \tag{3.2}$$

where $Q_x \in \mathbb{R}^{n \times n}$ and $Q_u \in \mathbb{R}^{m \times m}$ define the integrated cost, $P_T \in \mathbb{R}^{n \times n}$ is the terminal cost, and T is the time horizon. Our goal is to find $P_T > 0$, $Q_x > 0$, $Q_u > 0$, and $T > 0$ such that the resulting optimal control law is equivalent to $u = Kx$.

The optimal control law for the quadratic cost function (3.2) is given by

$$u = -R^{-1} B^T P(t),$$

where $P(t)$ is the solution to the Riccati ordinary differential equation

$$-\dot{P} = A^T P + PA - PBR^{-1}B^T P + Q \tag{3.3}$$

with terminal condition $P(T) = P_T$. In order for this to give a control law of the form $u = Kx$ for a constant matrix K, we must find P_T, Q_x, and Q_u that give a constant solution to the Riccati equation (3.3) and satisfy $-R^{-1}B^T P = K$. It follows that P_T, Q_x and Q_u should satisfy

$$\begin{aligned} A^T P_T + P_T A - P_T B Q_u^{-1} B^T P_T + Q = 0 \\ -Q_u^{-1} B^T P_T = K. \end{aligned} \tag{3.4}$$

We note that the first equation is simply the normal algebraic Riccati equation of optimal control, but with P_T, Q, and R yet to be chosen. The second equation places additional constraints on R and P_T.

Equation (3.4) is exactly the same equation that one would obtain if we had considered an infinite time horizon problem, since the given control was constant and hence $P(t)$ was forced to be constant. This infinite horizon problem is precisely the one that Kalman considered in 1964, and hence his results apply directly. Namely, in the single-input single-output case, we can always find a solution to the coupled equations (3.4) under standard conditions on reachability and observability [Kal64]. The equations can be simplified by substituting the second relation into the first to obtain

$$A^T P_T + P_T A - K^T R K + Q = 0.$$

This equation is linear in the unknowns and can be solved directly (remembering that P_T, Q_x and Q_u are required to be positive definite).

The implication of these results is that any state feedback control law satisfying these assumptions can be realized as the solution to an appropriately defined receding horizon control law. Thus, we can implement the design framework summarized in Figure 3.1 for the case where our (linear) control design results in a state feedback controller.

The above results can be generalized to nonlinear systems, in which one takes a nonlinear control system and attempts to find a cost function such that the given controller is the optimal control with respect to that cost.

The history of inverse optimal control for nonlinear systems goes back to the early work of Moylan and Anderson [MA73]. More recently, Sepulchre et al. [SJK97] showed that a nonlinear state feedback obtained by Sontag's formula from a control Lyapunov function (CLF) is inverse optimal. The connections of this inverse optimality result to passivity and robustness properties of the optimal state feedback are discussed in Jankovic *et al.* [JSK99]. Most results on inverse optimality do not consider the constraints on control or state. However, the results on the unconstrained inverse optimality justify the use of a more general nonlinear loss function in the integrated cost of a finite horizon performance index combined with a real-time optimization-based control approach that takes the constraints into account.

Control Lyapunov Functions

For the optimal control problems that we introduce in the next section, we will make use of a terminal cost that is also a control Lyapunov function for the system. Control Lyapunov functions are an extension of standard Lyapunov functions and were originally introduced by Sontag [Son83]. They allow constructive design of nonlinear controllers and the Lyapunov function that proves their stability. A more complete treatment is given in [KKK95].

Consider a nonlinear control system

$$\dot{x} = f(x,u), \qquad x \in \mathbb{R}^n,\ u \in \mathbb{R}^m. \tag{3.5}$$

Definition 3.1 (Control Lyapunov Function)**.** A locally positive function $V : \mathbb{R}^n \to \mathbb{R}_+$ is called a *control Lyapunov function (CLF)* for a control system (3.5) if

$$\inf_{u \in \mathbb{R}^m} \left(\frac{\partial V}{\partial x} f(x,u) \right) < 0 \qquad \text{for all } x \neq 0.$$

In general, it is difficult to find a CLF for a given system. However, for many classes of systems, there are specialized methods that can be used. One of the simplest is to use the Jacobian linearization of the system around the desired equilibrium point and generate a CLF by solving an LQR problem.

As described in Chapter 2, the problem of minimizing the quadratic performance index,

$$J = \int_0^\infty (x^T(t) Q x(t) + u^T R u(t)) dt \qquad \text{subject to} \qquad \begin{aligned} \dot{x} &= Ax + Bu, \\ x(0) &= x_0, \end{aligned} \tag{3.6}$$

results in finding the positive definite solution of the following Riccati equation:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{3.7}$$

The optimal control action is given by

$$u = -R^{-1} B^T P x$$

and $V = x^T P x$ is a CLF for the system.

In the case of the nonlinear system $\dot{x} = f(x,u)$, A and B are taken as

$$A = \frac{\partial f(x,u)}{\partial x}|_{(0,0)} \qquad B = \frac{\partial f(x,u)}{\partial u}|_{(0,0)}$$

where the pairs (A, B) and $(Q^{\frac{1}{2}}, A)$ are assumed to be stabilizable and detectable respectively. The CLF $V(x) = x^T P x$ is valid in a region around the equilibrium $(0, 0)$, as shown in Exercise 3.1.

More complicated methods for finding control Lyapunov functions are often required and many techniques have been developed. An overview of some of these methods can be found in [Jad01].

Finite Horizon Optimal Control

We briefly review the problem of optimal control over a finite time horizon as presented in Chapter 2 to establish the notation for the chapter and set some more specific conditions required for receding horizon control. This material is based on [MHJ+03].

Given an initial state x_0 and a control trajectory $u(\cdot)$ for a nonlinear control system $\dot{x} = f(x, u)$, let $x^u(\cdot; x_0)$ represent the state trajectory. We can write this solution as a continuous curve

$$x^u(t; x_0) = x_0 + \int_0^t f(x^u(\tau; x_0), u(\tau))\, d\tau$$

for $t \geq 0$. We require that the trajectories of the system satisfy an *a priori* bound

$$\|x(t)\| \leq \beta(x, T, \|u(\cdot)\|_1) < \infty, \qquad t \in [0, T],$$

where β is continuous in all variables and monotone increasing in T and $\|u(\cdot)\|_1 = \|u(\cdot)\|_{L_1(0,T)}$. Most models of physical systems will satisfy a bound of this type.

The performance of the system will be measured by an integral cost $L : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$. We require that L be twice differentiable (C^2) and fully penalize both state and control according to

$$L(x, u) \geq c_q(\|x\|^2 + \|u\|^2), \qquad x \in \mathbb{R}^n, u \in \mathbb{R}^m$$

for some $c_q > 0$ and $L(0, 0) = 0$. It follows that the quadratic approximation of L at the origin is positive definite,

$$\left.\frac{\partial L}{\partial x}\right|_{(0,0)} \geq c_q I > 0.$$

To ensure that the solutions of the optimization problems of interest are well behaved, we impose some convexity conditions. We require the set $f(x, \mathbb{R}^m) \subset \mathbb{R}^n$ to be convex for each $x \in \mathbb{R}^n$. Letting $\lambda \in \mathbb{R}^n$ represent the co-state, we also require that the pre-Hamiltonian function $\lambda^T f(x, u) + L(x, u) =: K(x, u, \lambda)$ be strictly convex for each $(x, \lambda) \in \mathbb{R}^n \times \mathbb{R}^n$ and that there is a C^2 function $\bar{u}^* : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m$ providing the global minimum of $K(x, u, \lambda)$. The Hamiltonian $H(x, \lambda) := K(x, \bar{u}^*(x, \lambda), \lambda)$ is then C^2, ensuring that extremal state, co-state, *and* control trajectories will all be sufficiently smooth (C^1 or better). Note that these conditions are automatically satisfied for control affine f and quadratic L.

The cost of applying a control $u(\cdot)$ from an initial state x over the infinite time interval $[0, \infty)$ is given by

$$J_\infty(x, u(\cdot)) = \int_0^\infty L(x^u(\tau; x), u(\tau))\, d\tau\ .$$

The optimal cost (from x) is given by

$$J^*_\infty(x) = \inf_{u(\cdot)} J_\infty(x, u(\cdot))$$

where the control function $u(\cdot)$ belongs to some reasonable class of admissible controls (e.g., piecewise continuous). The function $J^*_\infty(x)$ is often called the *optimal value function* for the infinite horizon optimal control problem. For the class of f and L considered, it can be verified that $J^*_\infty(\cdot)$ is a positive definite C^2 function in a neighborhood of the origin [HO01].

For practical purposes, we are interested in finite horizon approximations of the infinite horizon optimization problem. In particular, let $V(\cdot)$ be a nonnegative C^2 function with $V(0) = 0$ and define the finite horizon cost (from x using $u(\cdot)$) to be

$$J_T(x, u(\cdot)) = \int_0^T L(x^u(\tau; x), u(\tau))\, d\tau + V(x^u(T; x)) \tag{3.8}$$

and denote the optimal cost (from x) as

$$J^*_T(x) = \inf_{u(\cdot)} J_T(x, u(\cdot)) \ .$$

As in the infinite horizon case, one can show, by geometric means, that $J^*_T(\cdot)$ is locally smooth (C^2). Other properties will depend on the choice of V and T.

Let Γ^∞ denote the domain of $J^*_\infty(\cdot)$ (the subset of $\mathbb{R}^n$ on which J^*_∞ is finite). It is not too difficult to show that the cost functions $J^*_\infty(\cdot)$ and $J^*_T(\cdot)$, $T \geq 0$, are continuous functions on Γ_∞ [Jad01]. For simplicity, we will allow $J^*_\infty(\cdot)$ to take values in the extended real line so that, for instance, $J^*_\infty(x) = +\infty$ means that there is no control taking x to the origin.

We will assume that f and L are such that the minimum value of the cost functions $J^*_\infty(x)$, $J^*_T(x)$, $T \geq 0$, is attained for each (suitable) x. That is, given x and $T > 0$ (including $T = \infty$ when $x \in \Gamma^\infty$), there is a (C^1 in t) optimal trajectory $(x^*_T(t; x), u^*_T(t; x))$, $t \in [0, T]$, such that $J_T(x, u^*_T(\cdot; x)) = J^*_T(x)$. For instance, if f is such that its trajectories can be bounded on finite intervals as a function of its input size, e.g., there is a continuous function β such that $\|x^u(t; x_0)\| \leq \beta(\|x_0\|, \|u(\cdot)\|_{L_1[0,t]})$, then (together with the conditions above) there will be a minimizing control (cf. [LM67]). Many such conditions may be used to good effect; see [Jad01] for a more complete discussion.

It is easy to see that $J^*_\infty(\cdot)$ is proper on its domain so that the sub-level sets

$$\Gamma^\infty_r := \{x \in \Gamma^\infty : J^*_\infty(x) \leq r^2\}$$

are compact and path connected and moreover $\Gamma^\infty = \bigcup_{r \geq 0} \Gamma^\infty_r$. Note also that Γ^∞ may be a proper subset of $\mathbb{R}^n$ since there may be states that cannot be driven to the origin. We use r^2 (rather than r) here to reflect the fact that our integral cost is quadratically bounded from below. We refer to sub-level sets of $J^*_T(\cdot)$ and $V(\cdot)$ using

$$\Gamma^T_r := \text{path connected component of } \{x \in \Gamma^\infty : J^*_T(x) \leq r^2\} \text{ containing } 0,$$

and

$$\Omega_r := \text{path connected component of } \{x \in \mathbb{R}^n : V(x) \leq r^2\} \text{ containing } 0.$$

These results provide the technical framework needed for receding horizon control.

3.2 Receding Horizon Control with CLF Terminal Cost

In receding horizon control, a finite horizon optimal control problem is solved, generating open-loop state and control trajectories. The resulting control trajectory is applied to the system for a fraction of the horizon length. This process is then repeated, resulting in a sampled data feedback law. Although receding horizon control has been successfully used in the process control industry for many years, its application to fast, stability-critical nonlinear systems has been more difficult. This is mainly due to two issues. The first is that the finite horizon optimizations must be solved in a relatively short period of time. Second, it can be demonstrated using linear examples that a naive application of the receding horizon strategy can have undesirable effects, often rendering a system unstable. Various approaches have been proposed to tackle this second problem; see [MRRS00] for a comprehensive review of this literature. The theoretical framework presented here also addresses the stability issue directly, but is motivated by the need to relax the computational demands of existing stabilizing RHC formulations.

Receding horizon control provides a practical strategy for the use of information from a model through on-line optimization. Every δ seconds, an optimal control problem is solved over a T second horizon, starting from the current state. The first δ seconds of the optimal control $u_T^*(\cdot; x(t))$ is then applied to the system, driving the system from $x(t)$ at current time t to $x_T^*(\delta, x(t))$ at the next sample time $t+\delta$ (assuming no model uncertainty). We denote this receding horizon scheme as $\mathcal{RH}(T, \delta)$.

In defining (unconstrained) finite horizon approximations to the infinite horizon problem, the key design parameters are the terminal cost function $V(\cdot)$ and the horizon length T (and, perhaps also, the increment δ). We wish to characterize the sets of choices that provide successful controllers.

It is well known (and easily demonstrated with linear examples), that simple truncation of the integral (i.e., $V(x) \equiv 0$) may have disastrous effects if $T > 0$ is too small. Indeed, although the resulting value function may be nicely behaved, the "optimal" receding horizon closed loop system can be unstable.

A more sophisticated approach is to make good use of a suitable terminal cost $V(\cdot)$. Evidently, the best choice for the terminal cost is $V(x) = J_\infty^*(x)$ since then the optimal finite and infinite horizon costs are the same. Of course, if the optimal value function were available there would be no need to solve a trajectory optimization problem. What properties of the optimal value function should be retained in the terminal cost? To be effective, the terminal cost should account for the discarded tail by ensuring that the origin can be reached from the terminal state $x^u(T; x)$ in an efficient manner (as measured by L). One way to do this is to use an appropriate control Lyapunov function, which is also an upper bound on the cost-to-go.

The following theorem shows that the use of a particular type of CLF is in fact effective, providing rather strong and specific guarantees.

Theorem 3.1. [JYH01] *Suppose that the terminal cost $V(\cdot)$ is a control Lyapunov function such that*

$$\min_{u \in \mathbb{R}^m} (\dot{V} + L)(x, u) \leq 0 \tag{3.9}$$

for each $x \in \Omega_{r_v}$ for some $r_v > 0$. Then, for every *$T > 0$ and $\delta \in (0, T]$, the resulting receding horizon trajectories go to zero exponentially fast. For each $T > 0$, there is an $\bar{r}(T) \geq r_v$ such that $\Gamma^T_{\bar{r}(T)}$ is contained in the region of attraction of $\mathcal{RH}(T, \delta)$. Moreover, given any compact subset Λ of Γ^∞, there is a T^* such that $\Lambda \subset \Gamma^T_{\bar{r}(T)}$ for all $T \geq T^*$.*

Theorem 3.1 shows that for *any* horizon length $T > 0$ and *any* sampling time $\delta \in (0, T]$, the receding horizon scheme is exponentially stabilizing over the set $\Gamma^T_{r_v}$. For a given T, the region of attraction estimate is enlarged by increasing r beyond r_v to $\bar{r}(T)$ according to the requirement that $V(x^*_T(T; x)) \leq r_v^2$ on that set. An important feature of the above result is that, for operations with the set $\Gamma^T_{\bar{r}(T)}$, there is no need to impose stability ensuring constraints which would likely make the online optimizations more difficult and time consuming to solve.

Sketch of proof. Let $x^u(\tau; x)$ represent the state trajectory at time τ starting from initial state x and applying a control trajectory $u(\cdot)$, and let $(x^*_T, u^*_T)(\cdot, x)$ represent the optimal trajectory of the finite horizon, optimal control problem with horizon T. Assume that $x^*_T(T; x) \in \Omega_r$ for some $r > 0$. Then for any $\delta \in [0, T]$ we want to show that the optimal cost $x^*_T(\delta; x)$ satisfies

$$J^*_T\big(x^*_T(\delta; x)\big) \leq J^*_T(x) - \int_0^\delta q\big(L(x^*_T(\tau; x), u^*_T(\tau; x)\big)\, d\tau. \tag{3.10}$$

This expression says that solution to the finite-horizon, optimal control problem starting at time $t = \delta$ has cost that is less than the cost of the solution from time $t = 0$, with the initial portion of the cost subtracted off.. In other words, we are closer to our solution by a finite amount at each iteration of the algorithm. It follows using Lyapunov analysis that we must converge to the zero cost solution and hence our trajectory converges to the desired terminal state (given by the minimum of the cost function).

To show equation (3.10) holds, consider a trajectory in which we apply the optimal control for the first T seconds and then apply a closed loop controller using a stabilizing feedback $u = -k(x)$ for another T seconds. (The stabilizing compensator is guaranteed to exist since V is a control Lyapunov function.) Let $(x^*_T, u^*_T)(t; x)$, $t \in [0, T]$ represent the optimal control and $(x^k, u^k)(t - T; x^*_T(T; x))$, $t \in [T, 2T]$ represent the control with $u = -k(x)$ applied where k satisfies $(\dot{V} + L)(x, -k(x)) \leq 0$. Finally, let $(\tilde{x}(t), \tilde{u}(t))$, $t \in [0, 2T]$ represent the trajectory obtained by concatenating the optimal trajectory (x^*_T, u^*_T) with the CLF trajectory (x^k, u^k).

We now proceed to show that the inequality (3.10) holds. The cost of using $\tilde{u}(\cdot)$ for the first T seconds starting from the initial state $x^*_T(\delta; x))$, $\delta \in [0, , T]$ is given by

$$\begin{aligned} J_T(x_T^*(\delta;x),\tilde u(\cdot)) &= \int_\delta^{T+\delta} L(\tilde x(\tau),\tilde u(\tau))\,d\tau + V(\tilde x(T+\delta)) \\ &= J_T^*(x) - \int_0^\delta L(x_T^*(\tau;x),u_T^*(\tau;x))\,d\tau - V(x_T^*(T;x)) \\ &\qquad + \int_T^{T+\delta} L(\tilde x(\tau),\tilde u(\tau))\,d\tau + V(\tilde x(T+\delta)). \end{aligned}$$

Note that the second line is simply a rewriting of the integral in terms of the optimal cost J_T^* with the necessary additions and subtractions of the additional portions of the cost for the interval $[\delta, T+\delta]$. We can how use the bound

$$L(\tilde x(\tau),\tilde u(\tau)) \leq \dot V(\tilde x(\tau),\tilde u(\tau), \qquad \tau\in[T,2T],$$

which follows from the definition of the CLF V and stabilizing controller $k(x)$. This allows us to write

$$\begin{aligned} J_T(x_T^*(\delta;x),\tilde u(\cdot)) &\leq J_T^*(x) - \int_0^\delta L(x_T^*(\tau;x),u_T^*(\tau;x))\,d\tau - V(x_T^*(T;x)) \\ &\qquad - \int_T^{T+\delta} \dot V(\tilde x(\tau),\tilde u(\tau))\,d\tau + V(\tilde x(T+\delta)) \\ &= J_T^*(x) - \int_0^\delta L(x_T^*(\tau;x),u_T^*(\tau;x))\,d\tau - V(x_T^*(T;x)) \\ &\qquad - V(\tilde x(\tau))\Big|_T^{T+\delta} + V(\tilde x(T+\delta)) \\ &= J_T^*(x) - \int_0^\delta L(x_T^*(\tau;x),u_T^*(\tau;x)). \end{aligned}$$

Finally, using the optimality of u_T^* we have that $J_T^*(x_T^*(\delta;x)) \leq J_T(x_T^*(\delta;x),\tilde u(\cdot))$ and we obtain equation (3.10). □

An important benefit of receding horizon control is its ability to handle state and control constraints. While the above theorem provides stability guarantees when there are no constraints present, it can be modified to include constraints on states and controls as well. In order to ensure stability when state and control constraints are present, the terminal cost $V(\cdot)$ should be a local CLF satisfying $\min_{u\in\mathcal{U}} \dot V + L(x,u) \leq 0$ where $\mathcal{U}$ is the set of controls where the control constraints are satisfied. Moreover, one should also require that the resulting state trajectory $x^{CLF}(\cdot)\in\mathcal{X}$, where $\mathcal{X}$ is the set of states where the constraints are satisfied. (Both $\mathcal{X}$ and $\mathcal{U}$ are assumed to be compact with origin in their interior). Of course, the set Ω_{r_v} will end up being smaller than before, resulting in a decrease in the size of the guaranteed region of operation (see [MRRS00] for more details).

3.3 Receding Horizon Control Using Differential Flatness

In this section we demonstrate how to use differential flatness to find fast numerical algorithms for solving the optimal control problems required for the receding horizon control results of the previous section. We consider the affine nonlinear control system

$$\dot{x} = f(x) + g(x)u, \tag{3.11}$$

where all vector fields and functions are smooth. For simplicity, we focus on the single input case, $u \in \mathbb{R}$. We wish to find a trajectory of equation (3.11) that minimizes the performance index (3.8), subject to a vector of initial, final, and trajectory constraints

$$\begin{aligned} lb_0 &\leq \psi_0(x(t_0), u(t_0)) \leq ub_0, \\ lb_f &\leq \psi_f(x(t_f), u(t_f)) \leq ub_f, \\ lb_t &\leq S(x, u) \leq ub_t, \end{aligned} \tag{3.12}$$

respectively. For conciseness, we will refer to this optimal control problem as

$$\min_{(x,u)} J(x, u) \qquad \text{subject to} \qquad \begin{cases} \dot{x} = f(x) + g(x)u, \\ lb \leq c(x, u) \leq ub. \end{cases} \tag{3.13}$$

Numerical Solution Using Collocation

A numerical approach to solving this optimal control problem is to use the direct collocation method outlined in Hargraves and Paris [HP87]. The idea behind this approach is to transform the optimal control problem into a nonlinear programming problem. This is accomplished by discretizing time into a grid of $N - 1$ intervals

$$t_0 = t_1 < t_2 < \ldots < t_N = t_f \tag{3.14}$$

and approximating the state x and the control input u as piecewise polynomials $\tilde{x}$ and $\tilde{u}$, respectively. Typically a cubic polynomial is chosen for the states and a linear polynomial for the control on each interval. Collocation is then used at the midpoint of each interval to satisfy equation (3.11). Let $\tilde{x}(x(t_1), ..., x(t_N))$ and $\tilde{u}(u(t_1), ..., u(t_N))$ denote the approximations to x and u, respectively, depending on $(x(t_1), ..., x(t_N)) \in \mathbb{R}^{nN}$ and $(u(t_1), ..., u(t_N)) \in \mathbb{R}^N$ corresponding to the value of x and u at the grid points. Then one solves the following finite dimension approximation of the original control problem (3.13):

$$\min_{y \in \mathbb{R}^M} F(y) = J(\tilde{x}(y), \tilde{u}(y)) \qquad \text{subject to} \qquad \begin{cases} \dot{\tilde{x}} - f(\tilde{x}(y)) + g(\tilde{x}(y))\tilde{u}(y) = 0, \\ lb \leq c(\tilde{x}(y), \tilde{u}(y)) \leq ub, \\ \qquad \forall t = \dfrac{t_j + t_{j+1}}{2} \quad j = 1, \ldots, N-1 \end{cases} \tag{3.15}$$

where $y = (x(t_1), u(t_1), \ldots, x(t_N), u(t_N))$, and $M = \dim y = (n+1)N$.

Seywald [Sey94] suggested an improvement to the previous method (see also [Bry99, p. 362]). Following this work, one first solves a subset of system dynamics in equation (3.13) for the the control in terms of combinations of the state and its time derivative. Then one substitutes for the control in the remaining system dynamics and constraints. Next all the time derivatives $\dot{x}_i$ are approximated by the finite difference approximations

$$\dot{\bar{x}}(t_i) = \frac{x(t_{i+1}) - x(t_i)}{t_{i+1} - t_i}$$

to get

$$\left.\begin{array}{l} p(\dot{x}(t_i), x(t_i)) = 0 \\ q(\dot{x}(t_i), x(t_i)) \leq 0 \end{array}\right\} \quad i = 0, ..., N-1.$$

The optimal control problem is turned into

$$\min_{y\in\mathbb{R}^M} F(y) \qquad \text{subject to} \qquad \begin{cases} p(\dot{x}(t_i), x(t_i)) = 0 \\ q(\dot{x}(t_i), x(t_i)) \leq 0 \end{cases} \tag{3.16}$$

where $y = (x(t_1), \dots, x(t_N))$, and $M = \dim y = nN$. As with the Hargraves and Paris method, this parameterization of the optimal control problem (3.13) can be solved using nonlinear programming.

The dimensionality of this discretized problem is lower than the dimensionality of the Hargraves and Paris method, where both the states and the input are the unknowns. This induces substantial improvement in numerical implementation.

Differential Flatness Based Approach

The results of Seywald give a constrained optimization problem in which we wish to minimize a cost functional subject to $n-1$ equality constraints, corresponding to the system dynamics, at each time instant. In fact, it is usually possible to reduce the dimension of the problem further. Given an output, it is generally possible to parameterize the control and a part of the state in terms of this output and its time derivatives. In contrast to the previous approach, one must use more than one derivative of this output for this purpose.

When the whole state and the input can be parameterized with one output, the system is differentially flat, as described in Section 1.3. When the parameterization is only partial, the dimension of the subspace spanned by the output and its derivatives is given by r the *relative degree* of this output [Isi89]. In this case, it is possible to write the system dynamics as

$$\begin{aligned} x &= \alpha(z, \dot{z}, \dots, z^{(q)}) \\ u &= \beta(z, \dot{z}, \dots, z^{(q)}) \\ \Phi(z, \dot{z}, &\dots, z^{n-r}) = 0 \end{aligned} \tag{3.17}$$

where $z \in \mathbb{R}^p$, $p > m$ represents a set of outputs that parameterize the trajectory and $\Phi : \mathbb{R}^n \times \mathbb{R}^m$ represents $n-r$ remaining differential constraints on the output. In the case that the system is flat, $r = n$ and we eliminate these differential constraints.

Unlike the approach of Seywald, it is not realistic to use finite difference approximations as soon as $r > 2$. In this context, it is convenient to represent z using B-splines. B-splines are chosen as basis functions because of their ease of enforcing continuity across knot points and ease of computing their derivatives. A pictorial representation of such an approximation is given in Figure 3.2. Doing so we get

$$z_j = \sum_{i=1}^{p_j} B_{i,k_j}(t) C_i^j, \qquad p_j = l_j(k_j - m_j) + m_j$$

where $B_{i,k_j}(t)$ is the B-spline basis function defined in [dB78] for the output z_j with order k_j, C_i^j are the coefficients of the B-spline, l_j is the number of knot intervals,

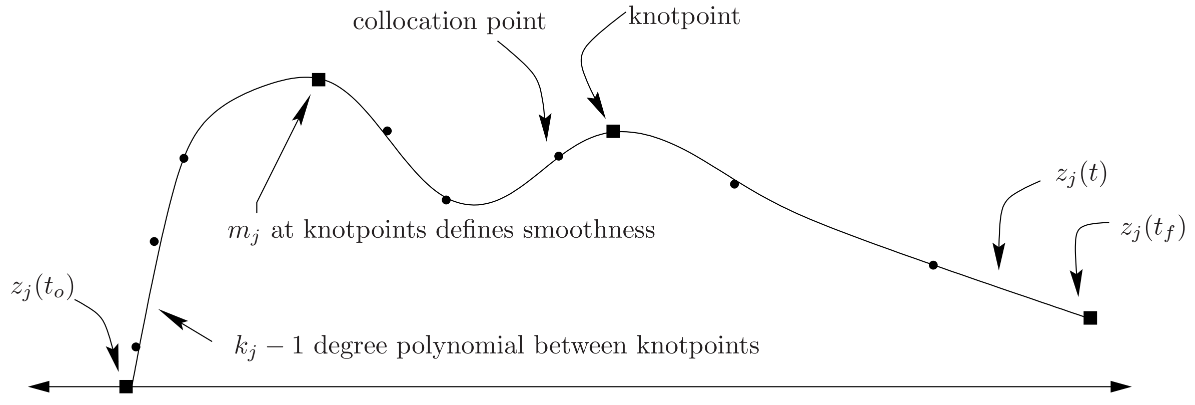

collocation point
knotpoint
$z_j(t)$
$z_j(t_f)$
m_j at knotpoints defines smoothness
$z_j(t_o)$
$k_j - 1$ degree polynomial between knotpoints

Figure 3.2: Spline representation of a variable.

and m_j is number of smoothness conditions at the knots. The set $(z_1, z_2, \ldots, z_{n-r})$ is thus represented by $M = \sum_{j\in\{1,r+1,\ldots,n\}} p_j$ coefficients.

In general, w collocation points are chosen uniformly over the time interval $[t_o, t_f]$ (though optimal knots placements or Gaussian points may also be considered). Both dynamics and constraints will be enforced at the collocation points. The problem can be stated as the following nonlinear programming form:

$$\min_{y\in\mathbb{R}^M} F(y) \qquad \text{subject to} \qquad \begin{cases} \Phi(z(y), \dot{z}(y), \ldots, z^{(n-r)}(y)) = 0 \\ \quad lb \le c(y) \le ub \end{cases} \tag{3.18}$$

where

$$y = (C_1^1, \ldots, C_{p_1}^1, C_1^{r+1}, \ldots, C_{p_{r+1}}^{r+1}, \ldots, C_1^n, \ldots, C_{p_n}^n).$$

The coefficients of the B-spline basis functions can be found using nonlinear programming.

A software package called Nonlinear Trajectory Generation (NTG) has been written to solve optimal control problems in the manner described above (see [MMM00] for details). The sequential quadratic programming package NPSOL by [GMSW] is used as the nonlinear programming solver in NTG. When specifying a problem to NTG, the user is required to state the problem in terms of some choice of outputs and its derivatives. The user is also required to specify the regularity of the variables, the placement of the knot points, the order and regularity of the B-splines, and the collocation points for each output.

3.4 Implementation on the Caltech Ducted Fan

To demonstrate the use of the techniques described in the previous section, we present an implementation of optimization-based control on the Caltech Ducted Fan, a real-time, flight control experiment that mimics the longitudinal dynamics of an aircraft. The experiment is show in Figure 3.3.

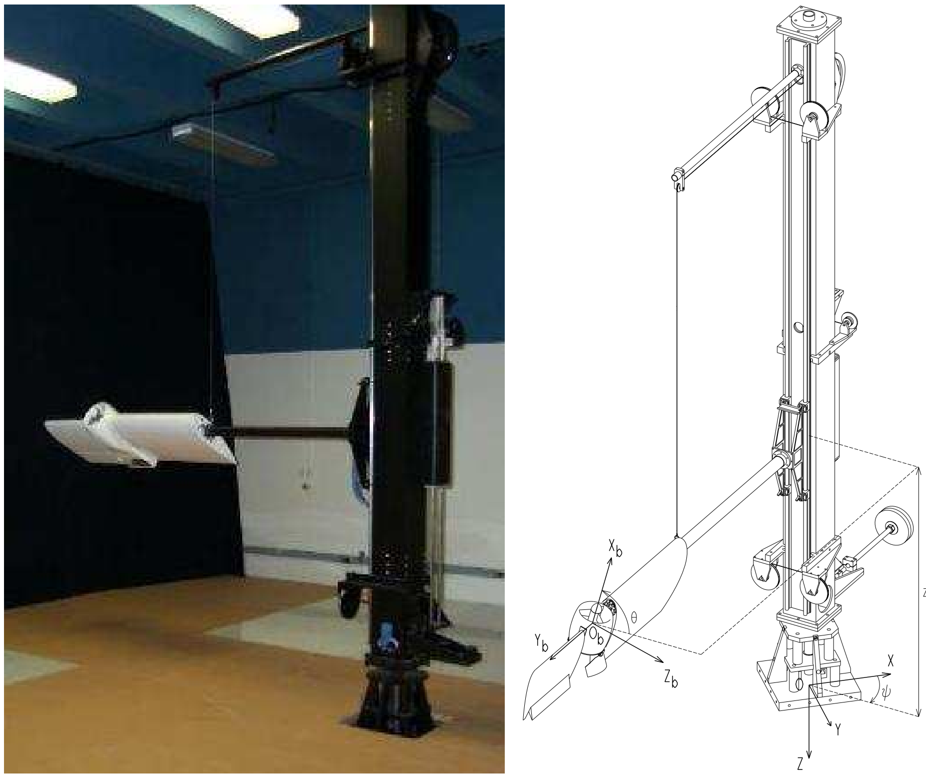


Figure 3.3: Caltech ducted fan.

Description of the Caltech Ducted Fan Experiment

The Caltech ducted fan is an experimental testbed designed for research and development of nonlinear flight guidance and control techniques for Uninhabited Combat Aerial Vehicles (UCAVs). The fan is a scaled model of the longitudinal axis of a flight vehicle and flight test results validate that the dynamics replicate qualities of actual flight vehicles [MM99].

The ducted fan has three degrees of freedom: the boom holding the ducted fan is allowed to operate on a cylinder, 2 m high and 4.7 m in diameter, permitting horizontal and vertical displacements. A counterweight is connected to the vertical axis of the stand, allowing the effective mass of the fan to be adjusted. Also, the wing/fan assembly at the end of the boom is allowed to rotate about its center of mass. Optical encoders mounted on the ducted fan, counterweight pulley, and the base of the stand measure the three degrees of freedom. The fan is controlled by commanding a current to the electric motor for fan thrust and by commanding RC servos to control the thrust vectoring mechanism.

The sensors are read and the commands sent by a DSP-based multi-processor system, comprised of a D/A card, a digital I/O card, two Texas Instruments C40 signal processors, two Compaq Alpha processors, and a high-speed host PC interface. A real-time interface provides access to the processors and I/O hardware. The

NTG software resides on both of the Alpha processors, each capable of running real-time optimization.

The ducted fan is modeled in terms of the position and orientation of the fan, and their velocities. Letting x represent the horizontal translation, z the vertical translation and θ the rotation about the boom axis, the equations of motion are given by

$$\begin{aligned}
&m\ddot{x} + F_{X_a} - F_{X_b}\cos\theta - F_{Z_b}\sin\theta = 0,\\
&m\ddot{z} + F_{Z_a} + F_{X_b}\sin\theta - F_{Z_b}\cos\theta = mg_{\text{eff}},\\
&J\ddot{\theta} - M_a + \frac{1}{r_s}I_p\Omega\dot{x}\cos\theta - F_{Z_b}r_f = 0,
\end{aligned} \tag{3.19}$$

where $F_{X_a} = D\cos\gamma + L\sin\gamma$ and $F_{Z_a} = -D\sin\gamma + L\cos\gamma$ are the aerodynamic forces and F_{X_b} and F_{Z_b} are thrust vectoring body forces in terms of the lift (L), drag (D), and flight path angle (γ). I_p and Ω are the moment of inertia and angular velocity of the ducted fan propeller, respectively. J is the moment of ducted fan and r_f is the distance from center of mass along the X_b axis to the effective application point of the thrust vectoring force. The angle of attack α can be derived from the pitch angle θ and the flight path angle γ by

$$\alpha = \theta - \gamma.$$

The flight path angle can be derived from the spatial velocities by

$$\gamma = \arctan\frac{-\dot{z}}{\dot{x}}.$$

The lift (L) ,drag (D), and moment (M) are given by

$$L = qSC_L(\alpha) \qquad D = qSC_D(\alpha) \qquad M = \bar{c}SC_M(\alpha),$$

respectively. The dynamic pressure is given by $q = \frac{1}{2}\rho V^2$. The norm of the velocity is denoted by V, S the surface area of the wings, and ρ is the atmospheric density. The coefficients of lift ($C_L(\alpha)$), drag ($C_D(\alpha)$) and the moment coefficient ($C_M(\alpha)$) are determined from a combination of wind tunnel and flight testing and are described in more detail in [MM99], along with the values of the other parameters.

Real-Time Trajectory Generation

In this section we describe the implementation of the trajectory generation algorithms by using NTG to generate minimum time trajectories in real time. An LQR-based regulator is used to stabilize the system. We focus in this section on aggressive, forward flight trajectories. The next section extends the controller to use a receding horizon controller, but on a simpler class of trajectories.

Stabilization Around Reference Trajectory

The results in this section rely on the traditional two degree of freedom design paradigm described in Chapter 1. In this approach, a local control law (inner loop) is used to stabilize the system around the trajectory computed based on a nominal model. This compensates for uncertainties in the model, which are predominantly due to aerodynamics and friction. Elements such as the ducted fan flying through its own wake, ground effects and velocity- and angle-of-attack dependent thrust contribute to the aerodynamic uncertainty. Actuation models are not used when generating the reference trajectory, resulting in another source of uncertainty.

Since only the position of the fan is measured, we must estimate the velocities. We use an extended Kalman filter (described in later chapters) with the optimal gain matrix is gain scheduled on the (estimated) forward velocity.

The stabilizing LQR controllers were gain scheduled on pitch angle, θ, and the forward velocity, $\dot{x}$. The pitch angle was allowed to vary from $-\pi/2$ to $\pi/2$ and the velocity ranged from 0 to 6 m/s. The weights were chosen differently for the hover-to-hover and forward flight modes. For the forward flight mode, a smaller weight was placed on the horizontal (x) position of the fan compared to the hover-to-hover mode. Furthermore, the z weight was scheduled as a function of forward velocity in the forward flight mode. There was no scheduling on the weights for hover-to-hover. The elements of the gain matrices for each of the controller and observer are linearly interpolated over 51 operating points.

Nonlinear Trajectory Generation Parameters

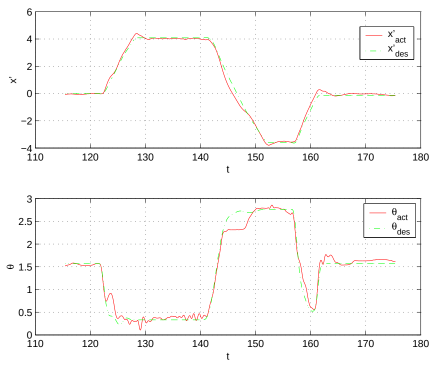
We solve a minimum time optimal control problem to generate a feasible trajectory for the system. The system is modeled using the nonlinear equations described above and computed the open loop forces and state trajectories for the nominal system. This system is not known to be differentially flat (due to the aerodynamic forces) and hence we cannot completely eliminate the differential constraints.

We choose three outputs, $z_1 = x$, $z_2 = z$, and $z_3 = \theta$, which results in a system with one remaining differential constraint. Each output is parameterized with four, sixth order C^4 piecewise polynomials over the time interval scaled by the minimum time. A fourth output, $z_4 = T$, is used to represent the time horizon to be minimized and is parameterized by a scalar. There are a total of 37 variables in this optimization problem. The trajectory constraints are enforced at 21 equidistant breakpoints over the scaled time interval.

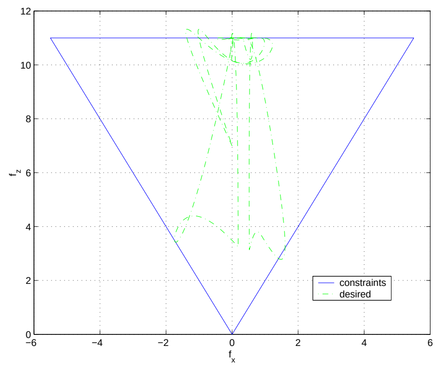
There are many considerations in the choice of the parameterization of the outputs. Clearly there is a trade between the parameters (variables, initial values of the variables, and breakpoints) and measures of performance (convergence, run-time, and conservative constraints). Extensive simulations were run to determine the right combination of parameters to meet the performance goals of our system.

Forward Flight

To obtain the forward flight test data, an operator commanded a desired forward velocity and vertical position with joysticks. We set the trajectory update time δ to 2 seconds. By rapidly changing the joysticks, NTG produces high angle of attack maneuvers. Figure 3.4aa depicts the reference trajectories and the actual θ and $\dot{x}$ over 60 s. Figure 3.4b shows the commanded forces for the same time interval. The sequence of maneuvers corresponds to the ducted fan transitioning from near hover to forward flight, then following a command from a large forward velocity to a large negative velocity, and finally returning to hover.



(a) System state



(b) Input forces

Figure 3.4: Forward flight test case: (a) θ and $\dot{x}$ desired and actual, (b) desired F_{X_b} and F_{Z_b} with bounds.

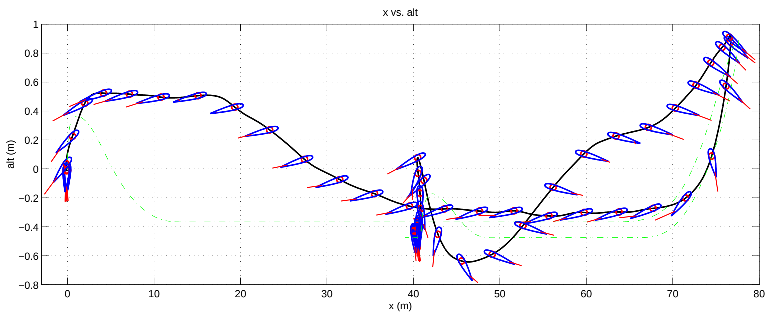


Figure 3.5: Forward flight test case: altitude and x position (actual (solid) and desired (dashed)). Airfoil represents actual pitch angle (θ) of the ducted fan.

Figure 3.5 is an illustration of the ducted fan altitude and x position for these maneuvers. The air-foil in the figure depicts the pitch angle (θ). It is apparent from this figure that the stabilizing controller is not tracking well in the z direction. This is due to the fact that unmodeled frictional effects are significant in the vertical direction. This could be corrected with an integrator in the stabilizing controller.

An analysis of the run times was performed for 30 trajectories; the average computation time was less than one second. Each of the 30 trajectories converged to an optimal solution and was approximately between 4 and 12 seconds in length. A random initial guess was used for the first NTG trajectory computation. Subsequent NTG computations used the previous solution as an initial guess. Much improvement can be made in determining a “good” initial guess. Improvement in the initial guess will improve not only convergence but also computation times.

Receding Horizon Control

The results of the previous section demonstrate the ability to compute optimal trajectories in real time, although the computation time was not sufficiently fast for closing the loop around the optimization. In this section, we make use of a shorter update time δ, a fixed horizon time T with a quadratic integral cost, and a CLF terminal cost to implement the receding horizon controller described in Section 3.2. We also limit the operation of the system to near hover, so that we can use the local linearization to find the terminal CLF.

We have implemented the receding horizon controller on the ducted fan experiment where the control objective is to stabilize the hover equilibrium point. The quadratic cost is given by

$$\begin{aligned} L(x,u) &= \frac{1}{2}\hat{x}^T Q \hat{x} + \frac{1}{2}\hat{u}^T R \hat{u} \\ V(x) &= \gamma \hat{x}^T P \hat{x} \end{aligned} \tag{3.20}$$

where

$$\begin{aligned} \hat{x} &= x - x_{eq} = (x, z, \theta - \pi/2, \dot{x}, \dot{z}, \dot{\theta}) \\ \hat{u} &= u - u_{eq} = (F_{X_b} - mg, F_{Z_b}) \\ Q &= \operatorname{diag}\{4, 15, 4, 1, 3, 0.3\} \\ R &= \operatorname{diag}\{0.5, 0.5\}, \end{aligned}$$

For the terminal cost, we choose $\gamma = 0.075$ and P is the unique stable solution to the algebraic Riccati equation corresponding to the linearized dynamics of equation (3.19) at hover and the weights Q and R. Note that if $\gamma = 1/2$, then $V(\cdot)$ is the CLF for the system corresponding to the LQR problem. Instead V is a relaxed (in magnitude) CLF, which achieved better performance in the experiment. In either case, V is valid as a CLF only in a neighborhood around hover since it is based on the linearized dynamics. We do not try to compute off-line a region of attraction for this CLF. Experimental tests omitting the terminal cost and/or the input constraints leads to instability. The results in this section show the success of this choice for V for stabilization. An inner-loop PD controller on $\theta, \dot{\theta}$ is implemented to stabilize to the receding horizon states $\theta^*_T, \dot{\theta}^*_T$. The θ dynamics are the fastest for this system and although most receding horizon controllers were found to be nominally stable without this inner-loop controller, small disturbances could lead to instability.

The optimal control problem is set-up in NTG code by parameterizing the three position states (x, z, θ), each with 8 B-spline coefficients. Over the receding horizon time intervals, 11 and 16 breakpoints were used with horizon lengths of 1, 1.5, 2, 3, 4 and 6 seconds. Breakpoints specify the locations in time where the differential equations and any constraints must be satisfied, up to some tolerance. The value of $F^{\max}_{X_b}$ for the input constraints is made conservative to avoid prolonged input saturation on the real hardware. The logic for this is that if the inputs are saturated on the real hardware, no actuation is left for the inner-loop θ controller and the system can go unstable. The value used in the optimization is $F^{\max}_{X_b} = 9$ N.

Computation time is non-negligible and must be considered when implementing the optimal trajectories. The computation time varies with each optimization as

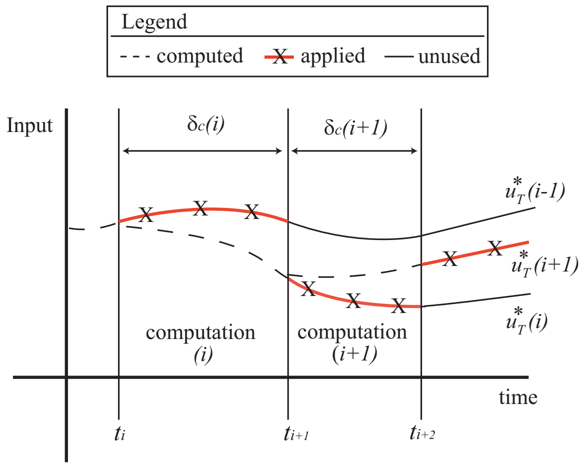
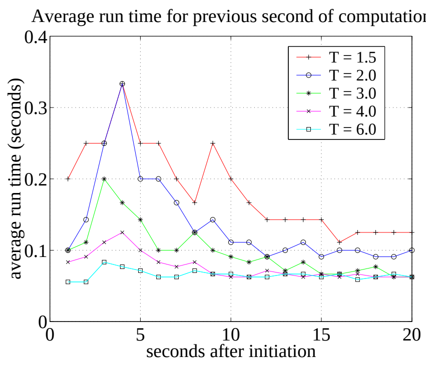


Figure 3.6: Receding horizon input trajectories.

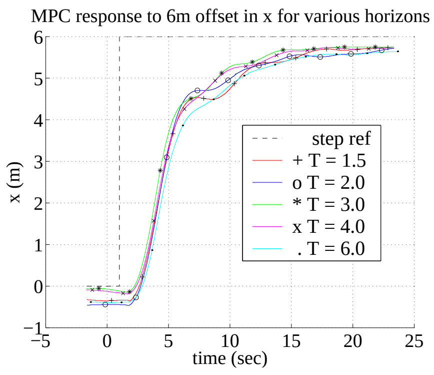
the current state of the ducted fan changes. The following notational definitions will facilitate the description of how the timing is set-up:

i	Integer counter of RHC computations
t_i	Value of current time when RHC computation i started
$\delta_c(i)$	Computation time for computation i
$u_T^*(i)(t)$	Optimal output trajectory corresponding to computation i, with time interval $t \in [t_i, t_i + T]$

A natural choice for updating the optimal trajectories for stabilization is to do so as fast as possible. This is achieved here by constantly resolving the optimization. When computation i is done, computation $i+1$ is immediately started, so $t_{i+1} = t_i + \delta_c(i)$. Figure 3.6 gives a graphical picture of the timing set-up as the optimal input trajectories $u_T^*(\cdot)$ are updated. As shown in the figure, any computation i for $u_T^*(i)(\cdot)$ occurs for $t \in [t_i, t_{i+1}]$ and the resulting trajectory is applied for $t \in [t_{i+1}, t_{i+2}]$. At $t = t_{i+1}$ computation $i+1$ is started for trajectory $u_T^*(i+1)(\cdot)$, which is applied as soon as it is available ($t = t_{i+2}$). For the experimental runs detailed in the results, $\delta_c(i)$ is typically in the range of $[0.05, 0.25]$ seconds, meaning 4 to 20 optimal control computations per second. Each optimization i requires the current measured state of the ducted fan and the value of the previous optimal input trajectories $u_T^*(i-1)$ at time $t = t_i$. This corresponds to, respectively, 6 initial conditions for state vector x and 2 initial constraints on the input vector u. Figure 3.6 shows that the optimal trajectories are advanced by their computation time prior to application to the system. A dashed line corresponds to the initial portion of an optimal trajectory and is not applied since it is not available until that computation is complete. The figure also reveals the possible discontinuity between successive applied optimal input trajectories, with a larger discontinuity more likely for longer computation times. The initial input constraint is an effort to reduce such discontinuities, although some discontinuity is unavoidable by this method. Also note that the same discontinuity is present for the 6 open-loop optimal state trajectories generated, again with a likelihood for greater discontinuity for longer



(a) Average run time



(b) Step responses

Figure 3.7: Receding horizon control: (a) moving one second average of computation time for RHC implementation with varying horizon time, (b) response of RHC controllers to 6 meter offset in x for different horizon lengths.

computation times. In this description, initialization is not an issue because we assume the receding horizon computations are already running prior to any test runs. This is true of the experimental runs detailed in the results.

The experimental results show the response of the fan with each controller to a 6 meter horizontal offset, which is effectively engaging a step-response to a change in the initial condition for x. The following details the effects of different receding horizon control parameterizations, namely as the horizon changes, and the responses with the different controllers to the induced offset.

The first comparison is between different receding horizon controllers, where time horizon is varied to be 1.5, 2.0, 3.0, 4.0 or 6.0 seconds. Each controller uses 16 breakpoints. Figure 3.7a shows a comparison of the average computation time as time proceeds. For each second after the offset was initiated, the data correspond to the average run time over the previous second of computation. Note that these computation times are substantially smaller than those reported for real-time trajectory generation, due to the use of the CLF terminal cost versus the terminal constraints in the minimum-time, real-time trajectory generation experiments.

There is a clear trend toward shorter average computation times as the time horizon is made longer. There is also an initial transient increase in average computation time that is greater for shorter horizon times. In fact, the 6 second horizon controller exhibits a relatively constant average computation time. One explanation for this trend is that, for this particular test, a 6 second horizon is closer to what the system can actually do. After 1.5 seconds, the fan is still far from the desired hover position and the terminal cost CLF is large, likely far from its region of attraction. Figure 3.7b shows the measured x response for these different controllers, exhibiting a rise time of 8–9 seconds independent of the controller. So a horizon time closer to the rise time results in a more feasible optimization in this case.

3.5 Further Reading

Exercises

3.1 Consider a nonlinear control system

$$\dot{x} = f(x, u)$$

with linearization

$$\dot{x} = Ax + Bu.$$

Show that if the linearized system is reachable, then there exists a (local) control Lyapunov function for the nonlinear system. (Hint: start by proving the result for a stable system.)

3.2 In this problem we will explore the effect of constraints on control of the linear unstable system given by

$$\begin{aligned}\dot{x}_1 &= 0.8x_1 - 0.5x_2 + 0.5u\\ \dot{x}_2 &= x_1 + 0.5u\end{aligned}$$

subject to the constraint that $|u| \leq a$ where a is a postive constant.

(a) Ignore the constraint ($a = \infty$) and design an LQR controller to stabilize the system. Plot the response of the closed system from the initial condition given by $x = (1, 0)$.

(b) Use SIMULINK or `ode45` to simulate the system for some finite value of a with an initial condition $x(0) = (1, 0)$. Numerically (trial and error) determine the smallest value of a for which the system goes unstable.

(c) Let $a_{\min}(\rho)$ be the smallest value of a for which the system is unstable from $x(0) = (\rho, 0)$. Plot $a_{\min}(\rho)$ for $\rho = 1$, 4, 16, 64, 256.

(d) *Optional*: Given $a > 0$, design and implement a receding horizon control law for this system. Show that this controller has larger region of attraction than the controller designed in part (b). (Hint: solve the finite horizon LQ problem analytically, using the bang-bang example as a guide to handle the input constraint.)

3.3 Consider the optimal control problem given in Example 2.2:

$$\dot{x} = ax + bu, \qquad J = \tfrac{1}{2}\int_{t_0}^{t_f} u^2(t)\,dt + \tfrac{1}{2}cx^2(t_f),$$

where $x \in \mathbb{R}$ is a scalar state, $u \in \mathbb{R}$ is the input, the initial state $x(t_0)$ is given, and $a, b \in \mathbb{R}$ are positive constants. We take the terminal time t_f as given and let $c > 0$ be a constant that balances the final value of the state with the input required to get to that position. The optimal control for a finite time $T > 0$ is derived in Example 2.2. Now consider the infinite horizon cost

$$J = \tfrac{1}{2}\int_{t_0}^{\infty} u^2(t)\,dt$$

with $x(t)$ at $t = \infty$ constrained to be zero.

(a) Solve for $u^*(t) = -bPx^*(t)$ where P is the positive solution corresponding to the algebraic Riccati equation. Note that this gives an explicit feedback law ($u = -bPx$).

(b) Plot the state solution of the finite time optimal controller for the following parameter values

$$a = 2 \qquad b = 0.5 \qquad x(t_0) = 4$$
$$c = 0.1,\ 10 \qquad t_f = 0.5,\ 1,\ 10$$

(This should give you a total of 6 curves.) Compare these to the infinite time optimal control solution. Which finite time solution is closest to the infinite time solution? Why?

Using the solution given in equation (2.5), implement the finite-time optimal controller in a receding horizon fashion with an update time of $\delta = 0.5$. Using the parameter values in part (b), compare the responses of the receding horizon controllers to the LQR controller you designed for problem 1, from the same initial condition. What do you observe as c and t_f increase?

(Hint: you can write a MATLAB script to do this by performing the following steps:

(i) set $t_0 = 0$

(ii) using the closed form solution for x^* from problem 1, plot $x(t)$, $t \in [t_0, t_f]$ and save $x_\delta = x(t_0 + \delta)$

(iii) set $x(t_0) = x_\delta$ and repeat step (ii) until x is small.)

3.4 In this problem we will explore the effect of constraints on control of the linear unstable system given by

$$\dot{x}_1 = 0.8x_1 - 0.5x_2 + 0.5u, \qquad \dot{x}_2 = x_1 + 0.5u,$$

subject to the constraint that $|u| \leq a$ where a is a postive constant.

(a) Ignore the constraint ($a = \infty$) and design an LQR controller to stabilize the system. Plot the response of the closed system from the initial condition given by $x = (1, 0)$.

(b) Use SIMULINK or `ode45` to simulate the system for some finite value of a with an initial condition $x(0) = (1, 0)$. Numerically (trial and error) determine the smallest value of a for which the system goes unstable.

(c) Let $a_{\min}(\rho)$ be the smallest value of a for which the system is unstable from $x(0) = (\rho, 0)$. Plot $a_{\min}(\rho)$ for $\rho = 1,\ 4,\ 16,\ 64,\ 256$.

(d) *Optional*: Given $a > 0$, design and implement a receding horizon control law for this system. Show that this controller has larger region of attraction than the controller designed in part (b). (Hint: solve the finite horizon LQ problem analytically, using the bang-bang example as a guide to handle the input constraint.)

Chapter 4
Stochastic Systems

In this chapter we present a focused overview of stochastic systems, oriented toward the material that is required in Chapters 5 and 6. After a brief review of random variables, we define discrete-time and continuous-time random processes, including the expectation, (co-)variance and correlation functions for a random process. These definitions are used to describe linear stochastic systems (in continuous time) and the stochastic response of a linear system to a random process (e.g., noise). We initially derive the relevant quantities in the state space, followed by a presentation of the equivalent frequency domain concepts.

Prerequisites. Readers should be familiar with basic concepts in probability, including random variables and standard distributions. We do not assume any prior familiarity with random processes.

Caveats. This chapter is written to provide a brief introduction to stochastic processes that can be used to derive the results in the following chapters. In order to keep the presentation compact, we gloss over several mathematical details that are required for rigorous presentation of the results. A more detailed (and mathematically precise) derivation of this material is available in the book by Åström [Åst06].

4.1 Brief Review of Random Variables

To help fix the notation that we will use, we briefly review the key concepts of random variables. A more complete exposition is available in standard books on probability, such as Hoel, Port and Stone [HPS71].

Random variables and processes are defined in terms of an underlying *probability space* that captures the nature of the stochastic system we wish to study. A probability space has three elements:

- a *sample space* Ω that represents the set of all possible outcomes;
- a set of *events* $\mathcal{F}$ the captures combinations of elementary outcomes that are of interest; and
- a *probability measure* $\mathcal{P}$ that describes the likelihood of a given event occurring.

Ω can be any set, either with a finite, countable or infinite number of elements. The event space $\mathcal{F}$ consists of subsets of Ω. There are some mathematical limits on the properties of the sets in $\mathcal{F}$, but these are not critical for our purposes here. The probability measure $\mathcal{P}$ is a mapping from $\mathcal{P} : \mathcal{F} \to [0, 1]$ that assigns a probability to each event. It must satisfy the property that given any two disjoint set $A, B \subset \mathcal{F}$,

$P(A \cup B) = P(A) + P(B)$. The term *probability distribution* is also to describe a probability measure.

With these definitions, we can model many different stochastic phenomena. Given a probability space, we can choose samples $\omega \in \Omega$ and identify each sample with a collection of events chosen from $\mathcal{F}$. These events should correspond to phenomena of interest and the probability measure $\mathcal{P}$ should capture the likelihood of that even occurring in the system that we are modeling. This definition of a probability space is very general and allows us to consider a number of situations as special cases.

A *random variable* X is a function $X : \Omega \to S$ that gives a value in S, called the state space, for any sample $\omega \in \Omega$. Given a subset $A \subset S$, we can write the probability that $X \in A$ as

$$P(X \in A) = P(\omega \in \Omega : X(\omega) \in A).$$

We will often find it convenient to omit ω when working random variables and hence we write $X \in S$ rather than the more correct $X(\omega) \in S$.

A *continuous (real-valued) random variable* X is a variable that can take on any value in the set of real numbers $\mathbb{R}$. We can model the random variable X according to its *probability distribution* P:

$$P(x_l \leq X \leq x_u) = \text{probability that } x \text{ takes on a value in the range } x_l,\, x_u.$$

More generally, we write $P(A)$ as the probability that an event A will occur (e.g., $A = \{x_l \leq X \leq x_u\}$). It follows from the definition that if X is a random variable in the range $[L, U]$ then $P(L \leq X \leq U) = 1$. Similarly, if $Y \in [L, U]$ then $P(L \leq X \leq Y) = 1 - P(Y \leq X \leq U)$.

We characterize a random variable in terms of the *probability density function* (pdf) $p(x)$. The density function is defined so that its integral over an interval gives the probability that the random variable takes its value in that interval:

$$P(x_l \leq X \leq x_u) = \int_{x_l}^{x_u} p(x)dx. \tag{4.1}$$

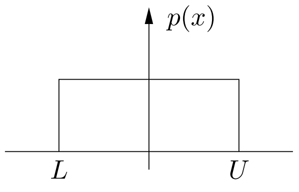
It is also possible to compute $p(x)$ given the distribution P as long as the distribution is suitably smooth:

$$p(x) = \left.\frac{\partial P(x_l \leq x \leq x_u)}{\partial x_u}\right|_{\substack{x_l \text{ fixed}, \\ x_u = x,}} \qquad x > x_l.$$

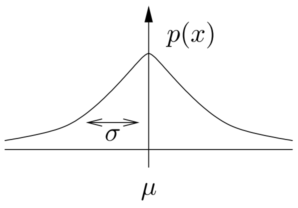
We will sometimes write $p_X(x)$ when we wish to make explicit that the pdf is associated with the random variable X. Note that we use capital letters to refer to a random variable and lower case letters to refer to a specific value.

Probability distributions provide a general way to describe stochastic phenomena. Some standard probability distributions include a *uniform distribution*,

$$p(x) = \frac{1}{U - L}, \tag{4.2}$$



(a) Uniform distribution



(b) Gaussian distribution

Figure 4.1: Probability density function (pdf) for uniform and Gaussian distributions.

and a *Gaussian distribution* (also called a *normal distribution*),

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}. \tag{4.3}$$

In the Gaussian distribution, the parameter μ is called the *mean* of the distribution and σ is called the *standard deviation* of the distribution. Figure 4.1 gives a graphical representation of uniform and Gaussian pdfs. There many other distributions that arise in applications, but for the purpose of these notes we focus on uniform distributions and Gaussian distributions.

If two random variables are related, we can talk about their *joint probability distribution*: $P_{X,Y}(A, B)$ is the probability that both event A occurs for X and B occurs for Y. This is sometimes written as $P(A \cap B)$, where we abuse notation by implicitly assuming that A is associated with X and B with Y. For continuous random variables, the joint probability distribution can be characterized in terms of a *joint probability density function*

$$P(x_l \le X \le x_u,\, y_l \le Y \le y_u) = \int_{y_l}^{y_u} \int_{x_l}^{x_u} p(x, y)\, dx dy. \tag{4.4}$$

The joint pdf thus describes the relationship between X and Y, and for sufficiently smooth distributions we have

$$p(x, y) = \left.\frac{\partial^2 P(x_l \le X \le x_u,\, y_l \le Y \le y_u)}{\partial x_u \partial y_u}\right|_{\substack{x_l, y_l \text{ fixed},\\ x_u = x,\; y_u = y,}} \qquad \begin{array}{l} x > x_l, \\ y > y_l. \end{array}$$

We say that X and Y are *independent* if $p(x, y) = p(x)p(y)$, which implies that $P_{X,Y}(A, B) = P_X(A)P_Y(B)$ for events A associated with X and B associated with Y. Equivalently, $P(A \cap B) = P(A)P(B)$ if A and B are independent.

The *conditional probability* for an event A given that an event B has occurred, written as $P(A|B)$, is given by

$$P(A|B) = \frac{P(A \cap B)}{P(B)}. \tag{4.5}$$

If the events A and B are independent, then $P(A|B) = P(A)$. Note that the individual, joint and conditional probability distributions are all different, so we should really write $P_{X,Y}(A \cap B)$, $P_{X|Y}(A|B)$ and $P_Y(B)$.

If X is dependent on Y then Y is also dependent on X. *Bayes' theorem* relates the conditional and individual probabilities:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \qquad P(B) \neq 0. \tag{4.6}$$

Bayes' theorem gives the conditional probability of event A on event B given the inverse relationship (B given A). It can be used in situations in which we wish to evaluate a hypothesis H given data D when we have some model for how likely the data is given the hypothesis, along with the unconditioned probabilities for both the hypothesis and the data.

The analog of the probability density function for conditional probability is the *conditional probability density function* $p(x|y)$

$$p(x|y) = \begin{cases} \dfrac{p(x,y)}{p(y)} & 0 < p(y) < \infty \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

It follows that

$$p(x,y) = p(x|y)p(y) \tag{4.8}$$

and

$$\begin{aligned} P(x_l \leq X \leq x_u|y) &:= P(x_l \leq X \leq x_u|Y = y) \\ &= \int_{x_l}^{x_u} p(x|y)dx = \frac{\int_{x_l}^{x_u} p(x,y)dx}{p(y)}. \end{aligned} \tag{4.9}$$

If X and Y are independent than $p(x|y) = p(x)$ and $p(y|x) = p(y)$. Note that $p(x,y)$ and $p(x|y)$ are different density functions, though they are related through equation (4.8). If X and Y are related with joint probability density function $p(x,y)$ and conditional probability density function $p(x|y)$ then

$$p(x) = \int_{-\infty}^{\infty} p(x,y)dy = \int_{-\infty}^{\infty} p(x|y)p(y)dy.$$

Example 4.1 Conditional probability for sum
Consider three random variables X, Y and Z related by the expression

$$Z = X + Y.$$

In other words, the value of the random variable Z is given by choosing values from two random variables X and Y and adding them. We assume that X and Y are independent Gaussian random variables with mean μ_1 and μ_2 and standard deviation $\sigma = 1$ (the same for both variables).

Clearly the random variable Z is not independent of X (or Y) since if we know the values of X then it provides information about the likely value of Z. To see this, we compute the joint probability between Z and X. Let

$$A = \{x_l \leq x \leq x_u\}, \qquad B = \{z_l \leq z \leq z_u\}.$$

The joint probability of both events A and B occurring is given by

$$\begin{aligned} P_{X,Z}(A \cap B) &= P(x_l \leq x \leq x_u,\, z_l \leq x + y \leq z_u) \\ &= P(x_l \leq x \leq x_u,\, z_l - x \leq y \leq z_u - x). \end{aligned}$$

We can compute this probability by using the probability density functions for X and Y:

$$P(A \cap B) = \int_{x_l}^{x_u} \Big(\int_{z_l - x}^{z_u - x} p_Y(y) dy \Big) p_X(x) dx$$
$$= \int_{x_l}^{x_u} \int_{z_l}^{z_u} p_Y(z - x) p_X(x) dz dx =: \int_{z_l}^{z_u} \int_{x_l}^{x_u} p_{Z,X}(z, x) dx dz.$$

Using Gaussians for X and Y we have

$$p_{Z,X}(z, x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(z - x - \mu_Y)^2} \cdot \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x - \mu_X)^2}$$
$$= \frac{1}{2\pi} e^{-\frac{1}{2}\big((z - x - \mu_Y)^2 + (x - \mu_X)^2\big)}.$$

A similar expression holds for $p_{Z,Y}$. ∇

Given a random variable X, we can define various standard measures of the distribution. The *expectation* or *mean* of a random variable is defined as

$$\mathbb{E}[X] = \langle X \rangle = \int_{-\infty}^{\infty} x \, p(x) \, dx,$$

and the *mean square* of a random variable is

$$\mathbb{E}[X^2] = \langle X^2 \rangle = \int_{-\infty}^{\infty} x^2 \, p(x) \, dx.$$

If we let μ represent the expectation (or mean) of X then we define the *variance* of X as

$$\mathbb{E}[(X - \mu)^2] = \langle (X - \langle X \rangle)^2 \rangle = \int_{-\infty}^{\infty} (x - \mu)^2 \, p(x) \, dx.$$

We will often write the variance as σ^2. As the notation indicates, if we have a Gaussian random variable with mean μ and (stationary) standard deviation σ, then the expectation and variance as computed above return μ and σ^2.

Several useful properties follow from the definitions.

Proposition 4.1 (Properties of random variables)**.**

1. *If X is a random variable with mean μ and variance σ^2, then αX is random variable with mean αX and variance $\alpha^2 \sigma^2$.*
2. *If X and Y are two random variables, then $\mathbb{E}[\alpha X + \beta Y] = \alpha \mathbb{E}[X] + \beta \mathbb{E}[Y]$.*
3. *If X and Y are Gaussian random variables with means μ_X, μ_Y and variances σ_X^2, σ_Y^2,*

$$p(x) = \frac{1}{\sqrt{2\pi\sigma_X^2}} e^{-\frac{1}{2}\left(\frac{x - \mu_X}{\sigma_X}\right)^2}, \qquad p(y) = \frac{1}{\sqrt{2\pi\sigma_Y^2}} e^{-\frac{1}{2}\left(\frac{y - \mu_Y}{\sigma_Y}\right)^2},$$

then $X + Y$ is a Gaussian random variable with mean $\mu_Z = \mu_X + \mu_Y$ and variance $\sigma_Z^2 = \sigma_X^2 + \sigma_Y^2$,

$$p(x + y) = \frac{1}{\sqrt{2\pi\sigma_Z^2}} e^{-\frac{1}{2}\left(\frac{x + y - \mu_Z}{\sigma_Z}\right)^2}.$$

Proof. The first property follows from the definition of mean and variance:

$$\begin{aligned}
\mathbb{E}[\alpha X] &= \int_{-\infty}^{\infty} \alpha x\, p(x)\, dx = \alpha \int_{-\infty}^{\infty} \alpha x\, p(x)\, dx = \alpha \mathbb{E}[X] \\
\mathbb{E}[(\alpha X)^2] &= \int_{-\infty}^{\infty} (\alpha x)^2\, p(x)\, dx = \alpha^2 \int_{-\infty}^{\infty} x^2\, p(x)\, dx = \alpha^2 \mathbb{E}[X^2].
\end{aligned}$$

The second property follows similarly, remembering that we must take the expectation using the joint distribution (since we are evaluating a function of two random variables):

$$\begin{aligned}
\mathbb{E}[\alpha X + \beta Y] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (\alpha x + \beta y)\, p_{X,Y}(x,y)\, dxdy \\
&= \alpha \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x\, p_{X,Y}(x,y)\, dxdy + \beta \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} y\, p_{X,Y}(x,y)\, dxdy \\
&= \alpha \int_{-\infty}^{\infty} x\, p_X(x)\, dx + \beta \int_{-\infty}^{\infty} y\, p_Y(y)\, dy = \alpha \mathbb{E}[X] + \beta \mathbb{E}[Y].
\end{aligned}$$

The third item is left as an exercise. □

4.2 Introduction to Random Processes

A *random process* is a collection of time-indexed random variables. Formally, we consider a random process X to be a joint mapping of sample and a time to a state: $X : \Omega \times \mathcal{T} \to S$, where $\mathcal{T}$ is an appropriate time set. We view this mapping as a generalized random variable: a sample corresponds to choosing an entire function of time. Of course, we can always fix the time and interpret $X(\omega, t)$ as a regular random variable, with $X(\omega, t')$ representing a different random variable if $t \neq t'$. Our description of random processes will consist of describing how the random variable at a time t relates to the value of the random variable at an earlier time s. To build up some intuition about random processes, we will begin with the discrete time case, where the calculations are a bit more straightforward, and then proceed to the continuous time case.

A *discrete-time random process* is a stochastic system characterized by the *evolution* of a sequence of random variables $X[k]$, where k is an integer. As an example, consider a discrete-time linear system with dynamics

$$x[k+1] = Ax[k] + Bu[k] + Fw[k], \qquad y[k] \quad = Cx[k] + v[k]. \tag{4.10}$$

As in ÅM08, $x \in \mathbb{R}^n$ represents the state of the system, $u \in \mathbb{R}^m$ is the vector of inputs and $y \in \mathbb{R}^p$ is the vector of outputs. The (possibly vector-valued) signal w represents disturbances to the process dynamics and v represents noise in the measurements. To try to fix the basic ideas, we will take $u = 0$, $n = 1$ (single state) and $F = 1$ for now.

We wish to describe the evolution of the dynamics when the disturbances and noise are not given as deterministic signals, but rather are chosen from some probability distribution. Thus we will let $W[k]$ be a collection of random variables where the values at each instant k are chosen from the probability distribution $P[W, k]$. As the notation indicates, the distributions might depend on the time instant k, although the most common case is to have a *stationary* distribution in which the distributions are independent of k (defined more formally below).

In addition to stationarity, we will often also assume that distribution of values of W at time k is independent of the values of W at time l if $k \neq l$. In other words, $W[k]$ and $W[l]$ are two separate random variables that are independent of each other. We say that the corresponding random process is *uncorrelated* (also defined more formally below). As a consequence of our independence assumption, we have that

$$\mathbb{E}[W[k]W[l]] = \mathbb{E}[W^2[k]]\delta(k-l) = \begin{cases} \mathbb{E}[W^2[k]] & k = l \\ 0 & k \neq l. \end{cases}$$

In the case that $W[k]$ is a Gaussian with mean zero and (stationary) standard deviation σ, then $\mathbb{E}[W[k]W[l]] = \sigma^2\,\delta(k-l)$.

We next wish to describe the evolution of the state x in equation (4.10) in the case when W is a random variable. In order to do this, we describe the state x as a sequence of random variables $X[k]$, $k = 1, \cdots, N$. Looking back at equation (4.10), we see that even if $W[k]$ is an uncorrelated sequence of random variables, then the states $X[k]$ are not uncorrelated since

$$X[k+1] = AX[k] + FW[k],$$

and hence the probability distribution for X at time $k+1$ depends on the value of X at time k (as well as the value of W at time k), similar to the situation in Example 4.1.

Since each $X[k]$ is a random variable, we can define the mean and variance as $\mu[k]$ and $\sigma^2[k]$ using the previous definitions at each time k:

$$\mu[k] := \mathbb{E}[X[k]] = \int_{-\infty}^{\infty} x\, p(x,k)\, dx,$$

$$\sigma^2[k] := \mathbb{E}[(X[k]-\mu[k])^2] = \int_{-\infty}^{\infty} (x-\mu[k])^2\, p(x,k)\, dx.$$

To capture the relationship between the current state and the future state, we define the *correlation function* for a random process as

$$\rho(k_1,k_2) := \mathbb{E}[X[k_1]X[k_2]] = \int_{-\infty}^{\infty} x_1 x_2\, p(x_1,x_2;k_1,k_2)\, dx_1 dx_2$$

The function $p(x_i, x_j; k_1, k_2)$ is the *joint probability density function*, which depends on the times k_1 and k_2. A process is *stationary* if $p(x, k+d) = p(x, d)$ for all k, $p(x_i, x_j; k_1+d, k_2+d) = p(x_i, x_j; k_1, k_2)$, etc. In this case we can write $p(x_i, x_j; d)$ for the joint probability distribution. We will almost always restrict to this case. Similarly, we will write $p(k_1, k_2)$ as $p(d) = p(k, k+d)$.

We can compute the correlation function by explicitly computing the joint pdf (see Example 4.1) or by directly computing the expectation. Suppose that we take a random process of the form (4.10) with $x[0] = 0$ and W having zero mean and standard deviation σ. The correlation function is given by

$$\begin{aligned}\mathbb{E}[X[k_1]X[k_2]] &= E\Big\{\big(\sum_{i=0}^{k_1-1} A^{k_1-i}BW[i]\big)\big(\sum_{j=0}^{k_2-1} A^{k_2-j}BW[j]\big)\Big\}\\ &= E\Big\{\sum_{i=0}^{k_1-1}\sum_{j=0}^{k_2-1} A^{k_1-i}BW[i]W[j]BA^{k_2-j}\Big\}.\end{aligned}$$

We can now use the linearity of the expectation operator to pull this inside the summations:

$$\begin{aligned}\mathbb{E}[X[k_1]X[k_2]] &= \sum_{i=0}^{k_1-1}\sum_{j=0}^{k_2-1} A^{k_1-i}B\mathbb{E}[W[i]W[j]]BA^{k_2-j}\\ &= \sum_{i=0}^{k_1-1}\sum_{j=0}^{k_2-1} A^{k_1-i}B\sigma^2\delta(i-j)BA^{k_2-j}\\ &= \sum_{i=0}^{k_1-1} A^{k_1-i}B\sigma^2BA^{k_2-i}.\end{aligned}$$

Note that the correlation function depends on k_1 and k_2.

We can see the dependence of the correlation function on the time more clearly by letting $d = k_2 - k_1$ and writing

$$\begin{aligned}\rho(k, k+d) = \mathbb{E}[X[k]X[k+d]] &= \sum_{i=0}^{k_1-1} A^{k-i}B\sigma^2BA^{d+k-i}\\ &= \sum_{j=1}^{k} A^{j}B\sigma^2BA^{j+d} = \Big(\sum_{j=1}^{k} A^{j}B\sigma^2BA^{j}\Big)A^{d}.\end{aligned}$$

In particular, if the discrete time system is stable then $|A| < 1$ and the correlation function decays as we take points that are further departed in time (d large). Furthermore, if we let $k \to \infty$ (i.e., look at the steady state solution) then the correlation function only depends on d (assuming the sum converges) and hence the steady state random process is stationary.

In our derivation so far, we have assumed that $X[k+1]$ only depends on the value of the state at time k (this was implicit in our use of equation (4.10) and the assumption that $W[k]$ is independent of X). This particular assumption is known as the *Markov property* for a random process: a Markovian process is one in which the distribution of possible values of the state at time k depends only on the values of the state at the prior time and not earlier. Written more formally, we say that a discrete random process is Markovian if

$$p_{X,k}(x|X[k-1], X[k-2], \ldots, X[0]) = p_{X,k}(x|X[k-1]). \quad (4.11)$$

Markov processes are roughly equivalent to state space dynamical systems, where the future evolution of the system can be completely characterized in terms of the current value of the state (and not it history of values prior to that).

4.3 Continuous-Time, Vector-Valued Random Processes

We now consider the case where our time index is no longer discrete, but instead varies continuously. A fully rigorous derivation requires careful use of measure theory and is beyond the scope of this text, so we focus here on the concepts that will be useful for modeling and analysis of important physical properties.

A *continuous-time random process* is a stochastic system characterized by the evolution of a random variable $X(t)$, $t \in [0, T]$. We are interested in understanding how the (random) state of the system is related at separate times. The process is defined in terms of the "correlation" of $X(t_1)$ with $X(t_2)$.

We call $X(t) \in \mathbb{R}^n$ the *state* of the random process at time t. For the case $n > 1$, we have a vector of random processes:

$$X(t) = \begin{bmatrix} X_1(t) \\ \vdots \\ X_n(t) \end{bmatrix}$$

We can characterize the state in terms of a (vector-valued) time-varying pdf,

$$P(x_l \le X_i(t) \le x_u) = \int_{x_l}^{x_u} p_{X_i}(x; t) dx.$$

Note that the state of a random process is not enough to determine the next state (otherwise it would be a deterministic process). We typically omit indexing of the individual states unless the meaning is not clear from context.

We can characterize the dynamics of a random process by its statistical characteristics, written in terms of *joint probability* density functions:

$$\begin{aligned} &P(x_{1l} \le X_i(t_1) \le x_{1u}, x_{2l} \le X_j(t_2) \le x_{2u}) \\ &\qquad = \int_{x_{2l}}^{x_{2u}} \int_{x_{1l}}^{x_{1u}} p_{X_i,Y_i}(x_1, x_2; t_1, t_2)\, dx_1 dx_2 \end{aligned}$$

The function $p(x_i, x_j; t_1, t_2)$ is called a *joint probability density function* and depends both on the individual states that are being compared and the time instants over which they are compared. Note that if $i = j$, then p_{X_i,X_i} describes how X_i at time t_1 is related to X_i at time t_2.

In general, the distributions used to describe a random process depend on the specific time or times that we evaluate the random variables. However, in some cases the relationship only depends on the difference in time and not the absolute times (similar to the notion of time invariance in deterministic systems, as described in ÅM08). A process is *stationary* if $p(x, t + \tau) = p(x, t)$ for all τ, $p(x_i, x_j; t_1 + \tau, t_2 + \tau) = p(x_i, x_j; t_1, t_2)$, etc. In this case we can write $p(x_i, x_j; \tau)$ for the joint probability distribution. Stationary distributions roughly correspond to the steady state properties of a random process and we will often restrict our attention to this case.

In looking at biomolecular systems, we are going to be interested in random processes in which the changes in the state occur when a random event occurs (such as a molecular reaction or binding event). In this case, it is natural to describe the state of the system in terms of a set of times $t_0 < t_1 < t_2 < \cdots < t_n$ and $X(t_i)$ is the random variable that corresponds to the possible states of the system at time t_i. Note that time time instants do not have to be uniformly spaced and most often (for biomolecular systems) they will not be. All of the definitions above carry through, and the process can now be described by a probability distribution of the form

$$P\Big(X(t_i) \in [x_i, x_i + dx_i], i = 1, \ldots, n\Big) = \int \ldots \int p(x_n, x_{n-1}, \ldots, x_0; t_n, t_{n-1}, \ldots, t_0) dx_n\, dx_{n-1}\, dx_1,$$

where dx_i are taken as infinitesimal quantities.

An important class of stochastic systems is those for which the next state of the system depends only on the current state of the system and not the history of the process. Suppose that

$$P\Big(X(t_n) \in [x_n, x_n + dx_n] | X(t_i) \in [x_i, x_i + dx_i]|, i = 1, \ldots, n-1\Big) \\ = P(X(t_n) \in [x_n, x_n + dx_n] | X(t_{n-1}) \in [x_{n-1}, x_{n-1} + dx_{n-1}]). \quad (4.12)$$

That is, the probability of being in a given state at time t_n depends *only* on the state that we were in at the previous time instant t_{n-1} and not the entire history of states prior to t_{n-1}. A stochastic process that satisfies this property is called a *Markov process*.

In practice we do not usually specify random processes via the joint probability distribution $p(x_i, x_j; t_1, t_2)$ but instead describe them in terms of a *propogater function*. Let $X(t)$ be a Markov process and define the Markov propogater as

$$\Xi(dt; x, t) = X(t + dt) - X(t), \text{ given } X(t) = x.$$

The propogater function describes how the random variable at time t is related to the random variable at time $t + dt$. Since both $X(t + dt)$ and $X(t)$ are random variables, $\Xi(dt; x, t)$ is also a random variable and hence it can be described by its density function, which we denote as $\Pi(\xi, x; dt, t)$:

$$P\big(x \le X(t + dt) \le x + \xi\big) = \int_x^{x+\xi} \Pi(dx, x; dt, t)\, dx.$$

The previous definitions for mean, variance and correlation can be extended to the continuous time, vector-valued case by indexing the individual states:

$$E\{X(t)\} = \begin{bmatrix} E\{X_1(t)\} \\ \vdots \\ E\{X_n(t)\} \end{bmatrix} =: \mu(t)$$

$$E\{(X(t) - \mu(t))(X(t) - \mu(t))^T\} = \begin{bmatrix} E\{X_1(t)X_1(t)\} & \ldots & E\{X_1(t)X_n(t)\} \\ & \ddots & \vdots \\ & & E\{X_n(t)X_n(t)\} \end{bmatrix} =: \Sigma(t)$$

$$E\{X(t)X^T(s)\} = \begin{bmatrix} E\{X_1(t)X_1(s)\} & \ldots & E\{X_1(t)X_n(s)\} \\ & \ddots & \vdots \\ & & E\{X_n(t)X_n(s)\} \end{bmatrix} =: R(t, s)$$

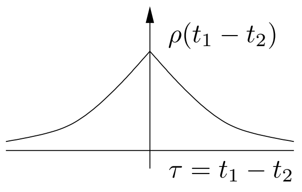


Figure 4.2: Correlation function for a first-order Markov process.

Note that the random variables and their statistical properties are all indexed by the time t (and s). The matrix $R(t,s)$ is called the *correlation matrix* for $X(t) \in \mathbb{R}^n$. If $t = s$ then $R(t,t)$ describes how the elements of x are correlated at time t (with each other) and in the case that the processes have zero mean, $R(t,t) = \Sigma(t)$. The elements on the diagonal of $\Sigma(t)$ are the variances of the corresponding scalar variables. A random process is uncorrelated if $R(t,s) = 0$ for all $t \neq s$. This implies that $X(t)$ and $X(s)$ are independent random events and is equivalent to $p_{X,Y}(x,y) = p_X(x)p_Y(y)$.

If a random process is stationary, then it can be shown that $R(t+\tau, s+\tau) = R(t,s)$ and it follows that the correlation matrix depends only on $t-s$. In this case we will often write $R(t,s) = R(s-t)$ or simple $R(\tau)$ where τ is the correlation time. The correlation matrix in this case is simply $R(0)$.

In the case where X is also scalar random process, the correlation matrix is also a scalar and we will write $\rho(\tau)$, which we refer to as the (scalar) correlation function. Furthermore, for stationary scalar random processes, the correlation function depends only on the absolute value of the correlation function, so $\rho(\tau) = \rho(-\tau) = \rho(|\tau|)$. This property also holds for the diagonal entries of the correlation matrix since $R_{ii}(s,t) = R_{ii}(t,s)$ from the definition.

Example 4.2 Ornstein-Uhlenbeck process
Consider a scalar random process defined by a Gaussian pdf with $\mu = 0$,

$$p(x,t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{x^2}{\sigma^2}},$$

and a correlation function given by

$$\rho(t_1,t_2) = \frac{Q}{2\omega_0} e^{-\omega_0 |t_2 - t_1|}.$$

The correlation function is illustrated in Figure 4.2. This process is also known as an *Ornstein-Uhlenbeck process*, a term that is commonly used in the scientific literature. This is a stationary process. ∇

The terminology and notation for covariance and correlation varies between disciplines. The term covariance is often used to refer to both the relationship between different variables X and Y and the relationship between a single variable at different times, $X(t)$ and $X(s)$. The term "cross-covariance" is used to refer to the covariance between two random vectors X and Y, to distinguish this from the covariance of the elements of X with each other. The term "cross-correlation" is sometimes also used. Finally, the term "correlation coefficient" refers to the normalized correlation $\bar{\rho}(t,s) = \mathbb{E}[X(t)X(s)]/\mathbb{E}[X(t)X(t)]$..

`MATLAB` has a number of functions to implement covariance and correlation, which mostly match the terminology here:

- `cov(X)` - this returns the variance of the vector `X` that represents samples of a given random variable or the covariance of the columns of a matrix X where the rows represent observations.
- `cov(X, Y)` - equivalent to `cov([X(:), Y(:)])`. Computes the covariance between the columns of X and Y, where the rows are observations.
- `xcorr(X, Y)` - the "cross-correlation" between two random sequences. If these sequences came from a random process, this is correlation function $\rho(t)$.
- `xcov(X, Y)` - this returns the "cross-covariance", which `MATLAB` defines as the "mean-removed cross-correlation".

The `MATLAB` help pages give the exact formulas used for each, so the main point here is to be careful to make sure you know what you really want.

We will also make use of a special type of random process referred to as "white noise". A *white noise process* $X(t)$ satisfies $E\{X(t)\} = 0$ and $R(t,s) = W\delta(s-t)$, where $\delta(\tau)$ is the impulse function and W is called the *noise intensity*. White noise is an idealized process, similar to the impulse function or Heaviside (step) function in deterministic systems. In particular, we note that $\rho(0) = E\{X^2(t)\} = \infty$, so the covariance is infinite and we never see this signal in practice. However, like the step function, it is very useful for characterizing the responds of a linear system, as described in the following proposition. It can be shown that the integral of a white noise process is a Wiener process, and so often white noise is described as the derivative of a Wiener process.

4.4 Linear Stochastic Systems with Gaussian Noise

We now consider the problem of how to compute the response of a linear system to a random process. We assume we have a linear system described in state space as

$$\dot{X} = AX + FW, \qquad Y \quad = CX \tag{4.13}$$

Given an "input" W, which is itself a random process with mean $\mu(t)$, variance $\sigma^2(t)$ and correlation $\rho(t, t+\tau)$, what is the description of the random process Y?

Let W be a white noise process, with zero mean and noise intensity Q:

$$\rho(\tau) = Q\delta(\tau).$$

We can write the output of the system in terms of the convolution integral

$$Y(t) = \int_0^t h(t-\tau)W(\tau)\,d\tau,$$

where $h(t-\tau)$ is the impulse response for the system

$$h(t-\tau) = Ce^{A(t-\tau)}B + D\delta(t-\tau).$$

We now compute the statistics of the output, starting with the mean:

$$\begin{aligned}\mathbb{E}[Y(t)] &= E\{\int_0^t h(t-\eta)W(\eta)\,d\eta\} \\ &= \int_0^t h(t-\eta)E\{W(\eta)\}\,d\eta = 0.\end{aligned}$$

Note here that we have relied on the linearity of the convolution integral to pull the expectation inside the integral.

We can compute the covariance of the output by computing the correlation $\rho(\tau)$ and setting $\sigma^2 = \rho(0)$. The correlation function for y is

$$\begin{aligned}\rho_Y(t,s) = E\{Y(t)Y(s)\} &= E\{\int_0^t h(t-\eta)W(\eta)\,d\eta \cdot \int_0^s h(s-\xi)W(\xi)\,d\xi\} \\ &= E\{\int_0^t\int_0^s h(t-\eta)W(\eta)W(\xi)h(s-\xi)\,d\eta d\xi\}\end{aligned}$$

Once again linearity allows us to exchange expectation and integration

$$\begin{aligned}\rho_Y(t,s) &= \int_0^t\int_0^s h(t-\eta)E\{W(\eta)W(\xi)\}h(s-\xi)\,d\eta d\xi \\ &= \int_0^t\int_0^s h(t-\eta)Q\delta(\eta-\xi)h(s-\xi)\,d\eta d\xi \\ &= \int_0^t h(t-\eta)Qh(s-\eta)\,d\eta\end{aligned}$$

Now let $\tau = s-t$ and write

$$\begin{aligned}\rho_Y(\tau) = \rho_Y(t,t+\tau) &= \int_0^t h(t-\eta)Qh(t+\tau-\eta)\,d\eta \\ &= \int_0^t h(\xi)Qh(\xi+\tau)\,d\xi \qquad \text{(setting } \xi = t-\eta)\end{aligned}$$

Finally, we let $t\to\infty$ (steady state)

$$\lim_{t\to\infty}\rho_Y(t,t+\tau) = \bar\rho_Y(\tau) = \int_0^\infty h(\xi)Qh(\xi+\tau)d\xi \tag{4.14}$$

If this integral exists, then we can compute the second order statistics for the output Y.

We can provide a more explicit formula for the correlation function ρ in terms of the matrices A, F and C by expanding equation (4.14). We will consider the general case where $W\in\mathbb{R}^m$ and $Y\in\mathbb{R}^p$ and use the correlation matrix $R(t,s)$ instead of the correlation function $\rho(t,s)$. Define the *state transition matrix* $\Phi(t,t_0) = e^{A(t-t_0)}$ so that the solution of system (4.13) is given by

$$x(t) = \Phi(t,t_0)x(t_0) + \int_{t_0}^t \Phi(t,\lambda)Fw(\lambda)d\lambda$$

Proposition 4.2 (Stochastic response to white noise)**.** *Let* $E\{X(t_0)X^T(t_0)\} = P(t_0)$ *and* W *be white noise with* $E\{W(\lambda)W^T(\xi)\} = R_W\delta(\lambda-\xi)$*. Then the correlation matrix for* X *is given by*

$$R_X(t,s) = P(t)\Phi^T(s,t)$$

where $P(t)$ *satisfies the linear matrix differential equation*

$$\dot{P}(t) = AP + PA^T + FR_WF, \qquad P(0) \quad = P_0.$$

Proof. Using the definition of the correlation matrix, we have

$$\begin{aligned} E\{X(t)X^T(s)\} &= E\Big\{\Phi(t,0)X(0)X^T(0)\Phi^T(t,0) + \text{cross terms} \\ &\qquad + \int_0^t \Phi(t,\xi)FW(\xi)\,d\xi \int_0^s W^t(\lambda)F^T\Phi(s,\lambda)\,d\lambda\Big\} \\ &= \Phi(t,0)E\{X(0)X^T(0)\}\Phi(s,0) \\ &\qquad + \int_0^t\int_0^s \Phi(t,\xi)FE\{W(\xi)W^T(\lambda)\}F^T\Phi(s,\lambda)\,d\xi\,d\lambda \\ &= \Phi(t,0)P(0)\phi^T(s,0) + \int_0^t \Phi(t,\lambda)FR_W(\lambda)F^T\Phi(s,\lambda)\,d\lambda. \end{aligned}$$

Now use the fact that $\Phi(s,0) = \Phi(s,t)\Phi(t,0)$ (and similar relations) to obtain

$$R_X(t,s) = P(t)\Phi^T(s,t)$$

where

$$P(t) = \Phi(t,0)P(0)\Phi^T(t,0) + \int_0^T \Phi(t,\lambda)FR_WF^T(\lambda)\Phi^T(t,\lambda)d\lambda$$

Finally, differentiate to obtain

$$\dot{P}(t) = AP + PA^T + FR_WF, \qquad P(0) \quad = P_0$$

(see Friedland for details). □

The correlation matrix for the output Y can be computing using the fact that $Y = CX$ and hence $R_Y = C^TR_XC$. We will often be interested in the steady state properties of the output, which given by the following proposition.

Proposition 4.3 (Steady state response to white noise)**.** *For a time-invariant linear system driven by white noise, the correlation matrices for the state and output converge in steady state to*

$$R_X(\tau) = R_X(t,t+\tau) = Pe^{A^T\tau}, \qquad R_Y(\tau) = CR_X(\tau)C^T$$

where P *satisfies the algebraic equation*

$$AP + PA^T + FR_WF^T = 0 \qquad P > 0. \tag{4.15}$$

Equation (4.15) is called the *Lyapunov equation* and can be solved in MATLAB using the function `lyap`.

Example 4.3 First-order system
Consider a scalar linear process

$$\dot{X} = -aX + W, \qquad Y = cX,$$

where W is a white, Gaussian random process with noise intensity σ^2. Using the results of Proposition 4.2, the correlation function for X is given by

$$R_X(t, t+\tau) = p(t)e^{-a\tau}$$

where $p(t) > 0$ satisfies

$$p(t) = -2ap + \sigma^2.$$

We can solve explicitly for $p(t)$ since it is a (non-homogeneous) linear differential equation:

$$p(t) = e^{-2at}p(0) + (1 - e^{-2at})\frac{\sigma^2}{2a}.$$

Finally, making use of the fact that $Y = cX$ we have

$$\rho(t, t+\tau) = c^2(e^{-2at}p(0) + (1 - e^{-2at})\frac{\sigma^2}{2a})e^{-a\tau}.$$

In steady state, the correlation function for the output becomes

$$\rho(\tau) = \frac{c^2\sigma^2}{2a}e^{-a\tau}.$$

Note correlation function has the same form as the Ornstein-Uhlenbeck process in Example 4.2 (with $Q = c^2\sigma^2$). ∇

4.5 Random Processes in the Frequency Domain

As in the case of deterministic linear systems, we can analyze a stochastic linear system either in the state space or the frequency domain. The frequency domain approach provides a very rich set of tools for modeling and analysis of interconnected systems, relying on the frequency response and transfer functions to represent the flow of signals around the system.

Given a random process $X(t)$, we can look at the frequency content of the properties of the response. In particular, if we let $\rho(\tau)$ be the correlation function for a (scalar) random process, then we define the *power spectral density function* as the Fourier transform of ρ:

$$S(\omega) = \int_{-\infty}^{\infty} \rho(\tau)e^{-j\omega\tau}\, d\tau, \qquad \rho(\tau) \quad = \frac{1}{2\pi}\int_{-\infty}^{\infty} S(\omega)e^{j\omega\tau}\, d\tau.$$

The power spectral density provides an indication of how quickly the values of a random process can change through the frequency content: if there is high frequency content in the power spectral density, the values of the random variable can change quickly in time.

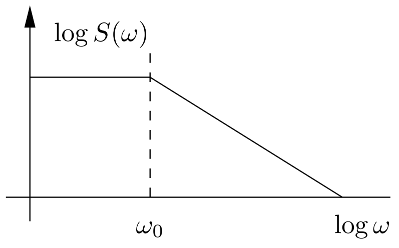


Figure 4.3: Spectral power density for a first-order Markov process.

.

Example 4.4 First-order Markov process
To illustrate the use of these measures, consider a first-order Markov process as defined in Example 4.2. The correlation function is

$$\rho(\tau) = \frac{Q}{2\omega_0} e^{-\omega_0(\tau)}.$$

The power spectral density becomes

$$\begin{aligned} S(\omega) &= \int_{-\infty}^{\infty} \frac{Q}{2\omega_0} e^{-\omega|\tau|} e^{-j\omega\tau}\, d\tau \\ &= \int_{-\infty}^{0} \frac{Q}{2\omega_0} e^{(\omega - j\omega)\tau}\, d\tau + \int_{0}^{\infty} \frac{Q}{2\omega_0} e^{(-\omega - j\omega)\tau}\, d\tau = \frac{Q}{\omega^2 + \omega_0^2}. \end{aligned}$$

We see that the power spectral density is similar to a transfer function and we can plot $S(\omega)$ as a function of ω in a manner similar to a Bode plot, as shown in Figure 4.3. Note that although $S(\omega)$ has a form similar to a transfer function, it is a real-valued function and is not defined for complex s. ∇

Using the power spectral density, we can more formally define "white noise": a *white noise process* is a zero-mean, random process with power spectral density $S(\omega) = W = \text{constant}$ for all ω. If $X(t) \in \mathbb{R}^n$ (a random vector), then $W \in \mathbb{R}^{n\times n}$. We see that a random process is white if all frequencies are equally represented in its power spectral density; this spectral property is the reason for the terminology "white". The following proposition verifies that this formal definition agrees with our previous (time domain) definition.

Proposition 4.4. *For a white noise process,*

$$\rho(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{j\omega\tau}\, d\tau = W\delta(\tau),$$

where $\delta(\tau)$ is the unit impulse function.

Proof. If $\tau \neq 0$ then

$$\rho(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} W(\cos(\omega\tau) + j\sin(\omega\tau)\, d\tau = 0$$

If $\tau = 0$ then $\rho(\tau) = \infty$. Can show that

$$\rho(0) = \lim_{\epsilon \to 0} \int_{-\epsilon}^{\epsilon} \int_{-\infty}^{\infty} (\cdots)\, d\omega d\tau = W\delta(0)$$

$\square$

Given a linear system

$$\dot{X} = AX + FW, \qquad Y = CX,$$

with W given by white noise, we can compute the spectral density function corresponding to the output Y. We start by computing the Fourier transform of the steady state correlation function (4.14):

$$\begin{aligned}
S_Y(\omega) &= \int_{-\infty}^{\infty} \left[\int_0^{\infty} h(\xi) Q h(\xi + \tau) d\xi \right] e^{-j\omega\tau}\, d\tau \\
&= \int_0^{\infty} h(\xi) Q \left[\int_{-\infty}^{\infty} h(\xi + \tau) e^{-j\omega\tau}\, d\tau \right] d\xi \\
&= \int_0^{\infty} h(\xi) Q \left[\int_0^{\infty} h(\lambda) e^{-j\omega(\lambda - \xi)}\, d\lambda \right] d\xi \\
&= \int_0^{\infty} h(\xi) e^{j\omega\xi}\, d\xi \cdot Q H(j\omega) = H(-j\omega) Q_u H(j\omega)
\end{aligned}$$

This is then the (steady state) response of a linear system to white noise.

As with transfer functions, one of the advantages of computations in the frequency domain is that the composition of two linear systems can be represented by multiplication. In the case of the power spectral density, if we pass white noise through a system with transfer function $H_1(s)$ followed by transfer function $H_2(s)$, the resulting power spectral density of the output is given by

$$S_Y(\omega) = H_1(-j\omega) H_2(-j\omega) Q_u H_2(j\omega) H_1(j\omega).$$

As stated earlier, white noise is an idealized signal that is not seen in practice. One of the ways to produced more realistic models of noise and disturbances it to apply a filter to white noise that matches a measured power spectral density function. Thus, we wish to find a covariance W and filter $H(s)$ such that we match the statistics $S(\omega)$ of a measured noise or disturbance signal. In other words, given $S(\omega)$, find $W > 0$ and $H(s)$ such that $S(\omega) = H(-j\omega) W H(j\omega)$. This problem is know as the *spectral factorization problem.*

Figure 4.4 summarizes the relationship between the time and frequency domains.

4.6 Further Reading

There are several excellent books on stochastic systems that cover the results in this chapter in much more detail. For discrete-time systems, the textbook by Kumar and Varaiya [KV86] provides an derivation of the key results. Results for continuous-time systems can be found in the textbook by Friedland [Fri04]. Åström [Åst06] gives a very elegant derivation in a unified framework that integrates discrete-time and continuous-time systems.

$$p(v) = \frac{1}{\sqrt{2\pi R_V}} e^{-\frac{v^2}{2R_V}} \qquad V \longrightarrow \boxed{H} \longrightarrow Y \qquad p(y) = \frac{1}{\sqrt{2\pi R_Y}} e^{-\frac{y^2}{2R_Y}}$$

$$S_V(\omega) = R_V \qquad\qquad S_Y(\omega) = H(-j\omega) R_V H(j\omega)$$

$$\rho_V(\tau) = R_V \delta(\tau) \qquad \begin{aligned} \dot X &= AX + FV \\ Y &= CX \end{aligned} \qquad \begin{gathered} \rho_Y(\tau) = R_Y(\tau) = CPe^{-A|\tau|}C^T \\ AP + PA^T + FR_V F^T = 0 \end{gathered}$$

Figure 4.4: Summary of steady state stochastic response.

Exercises

4.1 Let Z be a random random variable that is the sum of two independent normally (Gaussian) distributed random variables X_1 and X_2 having means m_1, m_2 and variances σ_1^2, σ_2^2 respectively. Show that the probability density function for Z is

$$p(z) = \frac{1}{2\pi\sigma_1\sigma_2} \int_{-\infty}^{\infty} \exp\left\{ -\frac{(z-x-m_1)^2}{2\sigma_1^2} - \frac{(x-m_2)^2}{2\sigma_2^2} \right\} dx$$

and confirm that this is normal (Gaussian) with mean m_1+m_2 and variance $\sigma_1^2+\sigma_2^2$. (Hint: Use the fact that $p(z|x_2) = p_{X_1}(x_1) = p_{X_1}(z-x_2)$.)

4.2 (ÅM08, Exercise 7.13) Consider the motion of a particle that is undergoing a random walk in one dimension (i.e., along a line). We model the position of the particle as

$$x[k+1] = x[k] + u[k],$$

where x is the position of the particle and u is a white noise processes with $E\{u[i]\} = 0$ and $E\{u[i]\,u[j]\} R_u \delta(i-j)$. We assume that we can measure x subject to additive, zero-mean, Gaussian white noise with covariance 1. Show that the expected value of the particle as a function of k is given by

$$E\{x[k]\} = E\{x[0]\} + \sum_{i=0}^{k-1} E\{u[i]\} = E\{x[0]\} =: \mu_x$$

and the covariance $E\{(x[k]-\mu_x)^2\}$ is given by

$$E\{(x[k]-\mu_x)^2\} = \sum_{i=0}^{k-1} E\{u^2[i]\} = kR_u$$

4.3 Consider a second order system with dynamics

$$\begin{bmatrix} \dot X_1 \\ \dot X_2 \end{bmatrix} = \begin{bmatrix} -a & 0 \\ 0 & -b \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} v, \qquad Y \;= \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$$

that is forced by Gaussian white noise with zero mean and variance σ^2. Assume $a, b > 0$.

(a) Compute the correlation function $\rho(\tau)$ for the output of the system. Your answer should be an explicit formula in terms of a, b and σ.

(b) Assuming that the input transients have died out, compute the mean and variance of the output.

4.4 Find a constant matrix A and vectors F and C such that for

$$\dot{X} = AX + FW, \qquad Y = CX$$

the power spectrum of Y is given by

$$S(\omega) = \frac{1+\omega^2}{(1-7\omega^2)^2+1}$$

Describe the sense in which your answer is unique.

Chapter 5
Kalman Filtering

In this chapter we derive the optimal estimator for a linear system in continuous time (also referred to as the Kalman-Bucy filter). This estimator minimizes the covariance and can be implemented as a recursive filter.

Prerequisites. Readers should have basic familiarity with continuous-time stochastic systems at the level presented in Chapter 4.

5.1 Linear Quadratic Estimators

Consider a stochastic system

$$\dot{X} = AX + Bu + FW, \qquad Y \quad = CX + V,$$

where X represents that state, u is the (deterministic) input, W represents disturbances that affect the dynamics of the system and V represents measurement noise. Assume that the disturbance W and noise V are zero-mean, Gaussian white noise (but not necessarily stationary):

$$\begin{aligned} p(w) &= \frac{1}{\sqrt[n]{2\pi}\sqrt{\det R_W}} e^{-\frac{1}{2} w^T R_W^{-1} w} & \qquad E\{W(s)W^T(t)\} &= R_W(t)\delta(t-s) \\ p(v) &= \frac{1}{\sqrt[n]{2\pi}\sqrt{\det R_v}} e^{-\frac{1}{2} v^T R_v^{-1} v} & \qquad E\{V(s)V^T(t)\} &= R_v(t)\delta(t-s) \end{aligned}$$

We also assume that the cross correlation between W and V is zero, so that the disturbances are not correlated with the noise. Note that we use multi-variable Gaussians here, with noise intensities $R_W \in \mathbb{R}^{m\times m}$ and $R_V \in \mathbb{R}^{p\times p}$. In the scalar case, $R_W = \sigma_W^2$ and $R_V = \sigma_V^2$.

We formulate the optimal estimation problem as finding the estimate $\hat{X}(t)$ that minimizes the mean square error $E\{(X(t) - \hat{X}(t))(X(t) - \hat{X}(t))^T\}$ given $\{Y(\tau) : 0 \le \tau \le t\}$. It can be shown that this is equivalent to finding the expected value of X subject to the "constraint" given by all of the previous measurements, so that $\hat{X}(t) = E\{X(t)|Y(\tau), \tau \le t\}$. This was the way that Kalman originally formulated the problem and it can be viewed as solving a *least squares* problem: given all previous $Y(t)$, find the estimate $\hat{X}$ that satisfies the dynamics and minimizes the square error with the measured data. We omit the proof since we will work directly with the error formulation.

Theorem 5.1 (Kalman-Bucy, 1961)**.** *The optimal estimator has the form of a linear observer*

$$\dot{\hat{X}} = A\hat{X} + BU + L(Y - C\hat{X})$$

where $L(t) = P(t)C^T R_v^{-1}$ *and* $P(t) = E\{(X(t) - \hat{X}(t))(X(t) - \hat{X}(t))^T\}$ *satisfies*

$$\begin{aligned}\dot{P} &= AP + PA^T - PC^T R_v^{-1}(t)CP + FR_W(t)F^T,\\ P(0) &= E\{X(0)X^T(0)\}.\end{aligned}$$

Sketch of proof. The error dynamics are given by

$$\dot{E} = (A - LC)E + \xi, \qquad \xi = FW - LV, \qquad R_\xi = FR_W F^T + LR_v L^T$$

The covariance matrix $P_E = P$ for this process satisfies

$$\begin{aligned}\dot{P} &= (A - LC)P + P(A - LC)^T + FR_W F^T + LR_v L^T\\ &= AP + PA^T + FR_W F^T - LCP - PC^T L^T + LR_v L^T\\ &= AP + PA^T + FR_W F^T + (LR_v - PC^T)R_v^{-1}(LR_v + PC^T)^T\\ &\qquad - PC^T R_v CP,\end{aligned}$$

where the last line follows by completing the square. We need to find L such that $P(t)$ is as small as possible, which can be done by choosing L so that $\dot{P}$ decreases by the maximum amount possible at each instant in time. This is accomplished by setting

$$LR_v = PC^T \qquad \Longrightarrow \qquad L = PC^T R_v^{-1}.$$

□

Note that the Kalman filter has the form of a *recursive* filter: given $P(t) = E\{E(t)E^T(t)\}$ at time t, can compute how the estimate and covariance *change*. Thus we do not need to keep track of old values of the output. Furthermore, the Kalman filter gives the estimate $\hat{X}(t)$ *and* the covariance $P_E(t)$, so you can see how well the error is converging.

If the noise is stationary (R_W, R_V constant) and *if* the dynamics for $P(t)$ are stable, then the observer gain converges to a constant and satisfies the *algebraic Riccati equation*:

$$L = PC^T R_v^{-1} \qquad AP + PA^T - PC^T R_v^{-1} CP + FR_W F^T.$$

This is the most commonly used form of the controller since it gives an explicit formula for the estimator gains that minimize the error covariance. The gain matrix for this case can solved use the `lqe` command in MATLAB.

Another property of the Kalman filter is that it extracts the maximum possible information about output data. To see this, consider the *residual random process*

$$R = Y - C\hat{X}$$

(this process is also called the *innovations process*). It can be shown for the Kalman filter that the correlation matrix of R is given by

$$R_R(t, s) = W(t)\delta(t - s).$$

This implies that the residuals are a white noise process and so the output error has *no* remaining dynamic information content.

5.2 Extensions of the Kalman Filter

Correlated disturbances and noise

The derivation of the Kalman filter assumes that the disturbances and noise are independent and white. Removing the assumption of independence is straightforward and simply results in a cross term ($E\{W(t)V(s)\} = R_{WV}\delta(s-t)$) being carried through all calculations.

To remove the assumption of white noise disturbances, we can construct a filter that takes white noise as an input and produces a disturbance source with the appropriate correlation function (or equivalently, spectral power density function). The intuition behind this approach is that we must have an internal model of the noise and/or disturbances in order to capture the correlation between different times.

Eliminating correlated sensor noise is more difficult.

Extended Kalman filters

Consider a *nonlinear* system

$$\begin{aligned} \dot X &= f(X,U,W), & X &\in \mathbb{R}^n, u \in \mathbb{R}^m, \\ Y &= CX + V, & Y &\in \mathbb{R}^p, \end{aligned}$$

where W and V are Gaussian white noise processes with covariance matrices R_W and R_V. A nonlinear observer for the system can be constructed by using the process

$$\dot{\hat X} = f(\hat X, U, 0) + L(Y - C\hat X).$$

If we define the error as $E = X - \hat X$, the error dynamics are given by

$$\begin{aligned} \dot E &= f(X,U,W) - f(\hat X,U,0) - LC(X-\hat X) \\ &= F(E,\hat X,U,W) - LCe \end{aligned}$$

where

$$F(E,\hat X,U,W) = f(E+\hat X,U,W) - f(\hat X,U,0)$$

We can now linearize around *current* estimate $\hat X$:

$$\begin{aligned} \hat E &= \frac{\partial F}{\partial E}E + \underbrace{F(0,\hat X,U,0)}_{=0} + \underbrace{\frac{\partial F}{\partial W}W}_{\text{noise}} - \underbrace{LCe}_{\text{observer gain}} + \text{h.o.t} \\ &\approx \tilde A E + \tilde F W - LCE, \end{aligned}$$

where the matrices

$$\begin{aligned} \tilde A &= \left.\frac{\partial F}{\partial e}\right|_{(0,\hat X,U,0)} &= \left.\frac{\partial f}{\partial X}\right|_{(\hat X,U,0)} \\ \tilde F &= \left.\frac{\partial F}{\partial W}\right|_{(0,\hat X,U,0)} &= \left.\frac{\partial f}{\partial W}\right|_{(\hat X,U,0)} \end{aligned}$$

depend on current estimate $\hat{X}$. We can now design an observer for the linearized system around the *current* estimate:

$$\begin{aligned} \dot{\hat{X}} &= f(\hat{X}, U, 0) + L(Y - C\hat{X}), \qquad L = PC^T R_v^{-1}, \\ \dot{P} &= (\tilde{A} - LC)P + P(\tilde{A} - LC)^T + \tilde{F} R_W \tilde{F}^T + L R_v L^T, \\ P(t_0) &= E\{X(t_0)X^T(t_0)\} \end{aligned}$$

This is called the (Schmidt) *extended Kalman filter* (EKF).

The intuition in the Kalman filter is that we replace the prediction portion of the filter with the nonlinear modeling while using the instantaneous linearization to compute the observer gain. Although we lose optimality, in applications the extended Kalman filter works very well and it is very versatile, as illustrated in the following example.

Example 5.1 Online parameter estimation
Consider a linear system with unknown parameters ξ

$$\begin{aligned} \dot{X} &= A(\xi)X + B(\xi)U + FW, \qquad \xi \in \mathbb{R}^p, \\ Y &= C(\xi)X + V. \end{aligned}$$

We wish to solve the parameter identification problem: given $U(t)$ and $Y(t)$, estimate the value of the parameters ξ.

One approach to this online parameter estimation problem is to treat ξ as an unknown *state* that has zero derivative:

$$\dot{X} = A(\xi)X + B(\xi)U + FW, \qquad \dot{\xi} = 0.$$

We can now write the dynamics in terms of the extended state $Z = (X, \xi)$:

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} X \\ \xi \end{bmatrix} &= \overbrace{\begin{bmatrix} A(\xi) & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} X \\ \xi \end{bmatrix} + \begin{bmatrix} B(\xi) \\ 0 \end{bmatrix} U}^{f(\left[\begin{smallmatrix} X \\ \xi \end{smallmatrix}\right], U, W)} + \begin{bmatrix} F \\ 0 \end{bmatrix} W, \\ Y &= \underbrace{C(\xi)X + V}_{h(\left[\begin{smallmatrix} X \\ \xi \end{smallmatrix}\right], W)}. \end{aligned}$$

This system is nonlinear in the extended state Z, but we can use the extended Kalman filter to *estimate* Z. If this filter converges, then we obtain both an estimate of the original state X and an estimate of the unknown parameter $\xi \in \mathbb{R}^p$.

Remark: need various observability conditions on augmented system in order for this to work. ∇

5.3 LQG Control

Return to the original "H_2" control problem

Figure	$\dot{X} = AX + BU + FW$ $Y = CX + V$	W, V Gaussian white noise with covariance R_W, R_V

Stochastic control problem: find $C(s)$ to minimize

$$J = E\left\{\int_0^\infty \left[(Y-r)^T R_W (Y-r)^T + U^T R_v U\right] dt\right\}$$

Assume for simplicity that the reference input $r = 0$ (otherwise, translate the state accordingly).

Theorem 5.2 (Separation principle)**.** *The optimal controller has the form*

$$\begin{aligned}\dot{\hat{X}} &= A\hat{X} + BU + L(Y - C\hat{X}) \\ U &= K(\hat{X} - X_d)\end{aligned}$$

where L is the optimal observer gain ignoring the controller and K is the optimal controller gain ignoring the noise.

This is called the *separation principle* (for H_2 control).

5.4 Application to a Thrust Vectored Aircraft

To illustrate the use of the Kalman filter, we consider the problem of estimating the state for the Caltech ducted fan, described already in Section 3.4.

The following code implements an extended Kalman filter in MATLAB, by constructing a state vector that consists of the actual state, the estimated state and the elements of the covariance matrix $P(t)$:

```
% pvtol.m - nonlinear PVTOL model, with LQR and EKF
% RMM, 5 Feb 06
%
% This function has the dynamics for a nonlinear planar vertical takeoff
% and landing aircraft, with an LQR compensator and EKF estimator.
%
% state(1) x position, in meters
% state(2) y position, in meters
% state(3) theta angle, in radians
% state(4-6) velocities
% state(7-12) estimated states
% state(13-48) covariance matrix (ordered rowise)

function deriv = pvtol(t, state, flags)
global pvtol_K; % LQR gain
global pvtol_L; % LQE gain (temporary)
global pvtol_Rv; % Disturbance covariance
global pvtol_Rw; % Noise covariance
global pvtol_C; % outputs to use
global pvtol_F; % disturbance input

% System parameters
J = 0.0475; % inertia around pitch axis
m = 1.5; % mass of fan
```

```
r = 0.25; % distance to flaps
g = 10; % gravitational constant
d = 0.2;   % damping factor (estimated)

% Initialize the derivative so that it is correct size and shape
deriv = zeros(size(state));

% Extract the current state estimate
x = state(1:6);
xhat = state(7:12);

% Get the current output, with noise
y = pvtol_C*x + pvtol_C * ...
  [0.1*sin(2.1*t); 0.1*sin(3.2*t); 0; 0; 0; 0];

% Compute the disturbance forces
fd = [
  0.01*sin(0.1*t); 0.01*cos(0.027*t); 0
];

% Compute the control law
F = -pvtol_K * xhat + [0; m*g];

% A matrix at current estimated state
A = [  0     0     0     1     0     0;
       0     0     0     0     1     0;
       0     0     0     0     0     1;
       0, 0, (-F(1)*sin(xhat(3)) - F(2)*cos(xhat(3)))/m, -d, 0, 0;
       0, 0, (F(1)*cos(xhat(3)) - F(2)*sin(xhat(3)))/m, 0, -d, 0;
       0     0     0     0     0     0 ];

% Estimator dynamics (prediction)
deriv(7) = xhat(4); deriv(8) = xhat(5); deriv(9) = xhat(6);
deriv(10) = (F(1) * cos(xhat(3)) - F(2) * sin(xhat(3)) - d*xhat(4)) / m;
deriv(11) = (F(1) * sin(xhat(3)) + F(2) * cos(xhat(3)) - m*g - d*xhat(5)) / m;
deriv(12) = (F(1) * r) / J;

% Compute the covariance
P = reshape(state(13:48), 6, 6);
dP = A * P + P * A' - P * pvtol_C' * inv(pvtol_Rw) * pvtol_C * P + ...
  pvtol_F * pvtol_Rv * pvtol_F';
L = P * pvtol_C' * inv(pvtol_Rw);

% Now compute correction
xcor = L * (y - pvtol_C*xhat);
for i = 1:6, deriv(6+i) = deriv(6+i) + xcor(i); end;

% PVTOL dynamics
deriv(1) = x(4); deriv(2) = x(5); deriv(3) = x(6);
deriv(4) = (F(1)*cos(x(3)) - F(2)*sin(x(3)) - d*x(4) + fd(1)) / m;
deriv(5) = (F(1)*sin(x(3)) + F(2)*cos(x(3)) - m*g - d*x(5) + fd(2)) / m;
deriv(6) = (F(1) * r + fd(3)) / J;
```

```
% Copy in the covariance updates
for i = 1:6,
  for j = 1:6,
    deriv(6+6*i+j) = dP(i, j);
  end;
end;

% All done
return;
```

To show how this estimator can be used, consider the problem of stabilizing the system to the origin with an LQR controller that uses the estimated state. The following MATLAB code implements the controller, using the previous simulation:

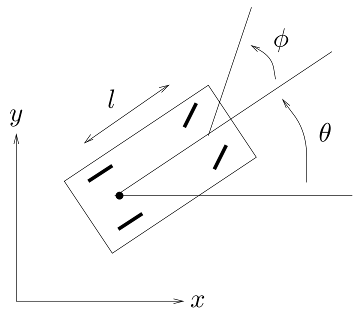
```
% kf_dfan.m - Kalman filter for the ducted fan
% RMM, 5 Feb 06

% Global variables to talk to simulation modle
global pvtol_K pvtol_L pvtol_C pvtol_Rv pvtol_Rw pvtol_F;

%%
%% Ducted fan dynamics
%%
%% These are the dynamics for the ducted fan, written in state space
%% form.
%%

% System parameters
J = 0.0475; % inertia around pitch axis
m = 1.5; % mass of fan
r = 0.25; % distance to flaps
g = 10; % gravitational constant
d = 0.2;   % damping factor (estimated)

% System matrices (entire plant: 2 input, 2 output)
A = [  0     0     0     1     0     0;
       0     0     0     0     1     0;
       0     0     0     0     0     1;
       0     0    -g   -d/m    0     0;
       0     0     0     0   -d/m    0;
       0     0     0     0     0     0 ];

B = [  0     0;
       0     0;
       0     0;
      1/m    0;
       0    1/m;
      r/J    0    ];

C = [  1     0    0      0     0     0;
       0     1    0      0     0     0 ];
```

```
D = [  0     0;   0      0];

dfsys = ss(A, B, C, D);

%%
%% State space control design
%%
%% We use an LQR design to choose the state feedback gains
%%

K = lqr(A, B, eye(size(A)), 0.01*eye(size(B'*B)));
pvtol_K = K;

%%
%% Estimator #1
%%

% Set the disturbances and initial condition
pvtol_F = eye(6);
pvtol_Rv = diag([0.0001, 0.0001, 0.0001, 0.01, 0.04, 0.0001]);

x0 = [0.1 0.2 0 0 0 0];
R11 = 0.1; R22 = 0.1; R33 = 0.001;

% Set the weighting matrices (L is computed but not used)
pvtol_C = [1 0 0 0 0 0; 0 1 0 0 0 0];
pvtol_Rw = diag([R11 R22]);
pvtol_L = lqe(A, pvtol_F, pvtol_C, pvtol_Rv, pvtol_Rw);

[t1, y1] = ode45(@pvtol, [0, 15], ...
  [x0 0*x0 reshape(x0'*x0, 1, 36)]);

subplot(321);
  plot(t1, y1(:,1), 'b-', t1, y1(:,2), 'g--');
  legend x y;
  xlabel('time');
  ylabel('States (no \theta)');
  axis([0 15 -0.3 0.3]);

subplot(323);
  plot(t1, y1(:,7) - y1(:,1), 'b-', ...
       t1, y1(:,8) - y1(:,2), 'g--', ...
       t1, y1(:,9) - y1(:,3), 'r-');
  legend xerr yerr terr;
  xlabel('time');
  ylabel('Errors (no \theta)');
  axis([0 15 -0.2 0.2]);

subplot(325);
  plot(t1, y1(:,13), 'b-', t1, y1(:,19), 'g--', t1, y1(:,25), 'r-');
  legend P11 P22 P33
  xlabel('time');
  ylabel('Covariance (w/ \theta)');
  axis([0 15 -0.2 0.2]);

%%
%% Estimator #2
%%

% Now change the output and see what happens (L computed but not used)
pvtol_C = [1 0 0 0 0 0; 0 1 0 0 0 0; 0 0 1 0 0 0];
pvtol_Rw = diag([R11 R22 R33]);
pvtol_L = lqe(A, pvtol_F, pvtol_C, pvtol_Rv, pvtol_Rw);

[t2, y2] = ode45(@pvtol, [0, 15], ...
  [x0 0*x0 reshape(x0'*x0, 1, 36)]);
subplot(322);
  plot(t2, y2(:,1), 'b-', t2, y2(:,2), 'g--');
  legend x y;
  xlabel('time');
  ylabel('States (w/ \theta)');
  axis([0 15 -0.3 0.3]);

subplot(324);
  plot(t2, y2(:,7) - y2(:,1), 'b-', ...
       t2, y2(:,8) - y2(:,2), 'g--', ...
       t2, y2(:,9) - y2(:,3), 'r-');
  legend xerr yerr terr;
  xlabel('time');
  ylabel('Errors (w/ \theta)');
  axis([0 15 -0.2 0.2]);

subplot(326);
  plot(t2, y2(:,13), 'b-', t2, y2(:,19), 'g--', t2, y2(:,25), 'r-');
  legend P11 P22 P33
  xlabel('time');
  ylabel('Covariance (w/ \theta)');
  axis([0 15 -0.2 0.2]);

print -dpdf dfan_kf.pdf
```

5.5 Further Reading

Exercises

5.1 Consider the problem of estimating the position of an autonomous mobile vehicle using a GPS receiver and an IMU (inertial measurement unit). The dynamics of the vehicle are given by



$$\dot{x} = \cos\theta\, v$$
$$\dot{y} = \sin\theta\, v$$
$$\dot{\theta} = \frac{1}{\ell}\tan\phi\, v,$$

We assume that the vehicle is disturbance free, but that we have noisy measurements from the GPS receiver and IMU and an initial condition error.

In this problem we will utilize the full form of the Kalman filter (including the $\dot{P}$ equation).

(a) Suppose first that we only have the GPS measurements for the xy position of the vehicle. These measurements give the position of the vehicle with approximately 1 meter accuracy. Model the GPS error as Gaussian white noise with $\sigma = 1.2$ meter in each direction and design an optimal estimator for the system. Plot the estimated states and the covariances for each state starting with an initial condition of 5 degree heading error at 10 meters/sec forward speed (i.e., choose $x(0) = (0, 0, 5\pi/180)$ and $\hat{x} = (0, 0, 0)$).

(b) An IMU can be used to measure angular rates and linear acceleration. Assume that we use a Northrop Grumman LN200 to measure the angular rate $\dot{\theta}$. Use the datasheet on the course web page to determine a model for the noise process and design a Kalman filter that fuses the GPS and IMU to determine the position of the vehicle. Plot the estimated states and the covariances for each state starting with an initial condition of 5 degree heading error at 10 meters/sec forward speed.

Note: be careful with units on this problem!

Chapter 6
Sensor Fusion

In this chapter we consider the problem of combining the data from different sensors to obtain an estimate of a (common) dynamical system. Unlike the previous chapters, we focus here on discrete-time processes, leaving the continuous-time case to the exercises. We begin with a summary of the input/output properties of discrete-time systems with stochastic inputs, then present the discrete-time Kalman filter, and use that formalism to formulate and present solutions for the sensor fusion problem. Some advanced methods of estimation and fusion are also summarized at the end of the chapter that demonstrate how to move beyond the linear, Gaussian process assumptions.

Prerequisites. The material in this chapter is designed to be reasonably self-contained, so that it can be used without covering Sections **??**–4.4 or Chapter 5 of this supplement. We assume rudimentary familiarity with discrete-time linear systems,, at the level of the brief descriptions in Chapters 2 and 5 of ÅM08, and discrete-time random processes as described in Section 4.2 of these notes.

6.1 Discrete-Time Stochastic Systems

We begin with a concise overview of stochastic system in discrete time, echoing our development of continuous-time random systems described in Chapter 4. We consider systems of the form

$$X[k+1] = AX[k] + Bu[k] + FW[k], \qquad Y[k] \quad = CX[k] + V[k], \tag{6.1}$$

where $X \in \mathbb{R}^n$ represents the state, $u \in \mathbb{R}^m$ represents the (deterministic) input, $W \in \mathbb{R}^q$ represents process disturbances, $Y \in \mathbb{R}^p$ represents the system output and $W \in \mathbb{R}^p$ represents measurement noise.

As in the case of continuous-time systems, we are interested in the response of the system to the random input $W[k]$. We will assume that W is a Gaussian process with zero mean and correlation function $\rho_W(k, k+d)$ (or correlation matrix $R_W(k, k+d)$ if W is vector valued). As in the continuous case, we say that a random process is white noise if $R_W(k, k+d) = R_W \delta(d)$ with $\delta(d) = 1$ if $d = 0$ and 0 otherwise. (Note that in the discrete-time case, white noise has finite covariance.)

To compute the response $Y[k]$ of the system, we look at the properties of the state vector $X[k]$. For simplicity, we take $u = 0$ (since the system is linear, we can always add it back in by superposition). Note first that the state at time $k + l$ can be written as

$$\begin{aligned} X[k+l] &= AX[k+l-1] + FW[x+l-1] \\ &= A(AX[k+l-2] + FW[x+l-2]) + FW[x+l-1] \\ &= A^l X[k] + \sum_{j=1}^{l} A^{j-1} FW[k+l-j]. \end{aligned}$$

The mean of the state at time k is given by

$$\mathbb{E}[X[k]] = A^k \mathbb{E}[E[0]] + \sum_{j=1}^{k} A^{j-1} F \mathbb{E}[W[k-j]] = A^k \mathbb{E}[X[0]].$$

To compute the covariance $R_X(k,k+d)$, we start by computing $R_X(k,k+1)$:

$$\begin{aligned} R_X(k,k+1) &= E\{X[k]X^T[k+1]\} \\ &= E\{(A^k x[0] + A^{k-1}Fw[0] + \cdots + ABw[k-2] + B[k-1])\cdot \\ &\qquad (A^{k+1}x[0] + A^k Bw[0] + \cdots + Bw[k])^T\} \end{aligned}$$

Performing a similar calculation for $R_X(k,k+l)$, it can be shown that

$$\begin{aligned} R_X(k,k+l) = \big(A^k P[0](A^T)^k &+ A^{k-1} F R_W[0] F^T (A^T)^{k-1} + \dots \\ &+ F R_W[k] F^T\big)(A^T)^l =: P[k](A^T)^l, \end{aligned} \tag{6.2}$$

where

$$P[k+1] = AP[k]A^T + FR_W[k]F^T. \tag{6.3}$$

The matrix $P[k]$ is the covariance of the state matrix and we see that its value can be computed *recursively* starting with $P[0] = \mathbb{E}[X[0]X^T[0]]$ and then applying equation (6.3). Equations (6.2) and (6.3) are the equivalent of Proposition 4.2 for continuous-time processes. If we additionally assume that W is stationary and focus on the steady state response, we obtain the following.

Proposition 6.1 (Steady state response to white noise)**.** *For a discrete-time, time-invariant, linear system driven by white noise, the correlation matrices for the state and output converge in steady state to*

$$R_X(d) = R_X(k,k+d) = PA^d, \qquad R_Y(d) = CR_X(d)C^T,$$

where P satisfies the algebraic equation

$$APA^T + FR_WF^T = 0 \qquad P > 0. \tag{6.4}$$

6.2 Kalman Filters in Discrete Time (AM08)

We now consider the optimal estimator in discrete time. This material is presented in ÅM08 in slightly simplified (but consistent) form.

Consider a discrete time, linear system with input, having dynamics

$$\begin{aligned} X[k+1] &= AX[k] + Bu[k] + FW[k], \\ Y[k] &= CX[k] + V[k], \end{aligned} \tag{6.5}$$

where $W[k]$ and $V[k]$ are Gaussian, white noise processes satisfying

$$\begin{aligned} E\{W[k]\} &= 0 & E\{V[k]\} &= 0 \\ E\{W[k]W^T[j]\} &= \begin{cases} 0 & k \neq j \\ R_W & k = j \end{cases} & E\{V[k]V^T[j]\} &= \begin{cases} 0 & k \neq j \\ R_V & k = j \end{cases} \\ && E\{W[k]V^T[j]\} = 0. \end{aligned} \tag{6.6}$$

We assume that the initial condition is also modeled as a Gaussian random variable with

$$E\{X[0]\} = x_0 \qquad E\{X[0]X^T[0]\} = P[0]. \tag{6.7}$$

We wish to find an estimate $\hat{X}[k]$ that gives the minimum mean square error (MMSE) for $E\{(X[k] - \hat{X}[k])(X[k] - \hat{X}[k])^T\}$ given the measurements $\{Y[l] : 0 \leq l \leq k\}$. We consider an observer of the form

$$\hat{X}[k+1] = A\hat{X}[k] + Bu[k] + L[k](Y[k] - C\hat{X}[k]). \tag{6.8}$$

The following theorem summarizes the main result.

Theorem 6.2. *Consider a random process $X[k]$ with dynamics* (6.5) *and noise processes and initial conditions described by equations* (6.6) *and* (6.7)*. The observer gain L that minimizes the mean square error is given by*

$$L[k] = AP[k]C^T(R_V + CP[k]C^T)^{-1},$$

where

$$\begin{aligned} P[k+1] &= (A - LC)P[k](A - LC)^T + FR_WF^T + LR_VL^T \\ P[0] &= E\{X[0]X^T[0]\}. \end{aligned} \tag{6.9}$$

Proof. We wish to minimize the mean square of the error, $E\{(X[k] - \hat{X}[k])(X[k] - \hat{X}[k])^T\}$. We will define this quantity as $P[k]$ and then show that it satisfies the recursion given in equation (6.9). Let $E[k] = Y[k] - C\hat{X}[k]$ be the residual between the measured output and the estimated output. By definition,

$$\begin{aligned} P[k+1] &= E\{E[k+1]E^T[k+1]\} \\ &= (A - LC)P[k](A - LC)^T + FR_WF^T + LR_VL^T \\ &= AP[k]A^T - AP[k]C^TL^T - LCP[k]A^T + \\ &\qquad L(R_V + CP[k]C^T)L^T + FR_WF^T. \end{aligned}$$

Letting $R_\epsilon = (R_V + CP[k]C^T)$, we have

$$\begin{aligned} P[k+1] &= AP[k]A^T - AP[k]C^TL^T - LCP[k]A^T + LR_\epsilon L^T + FR_WF^T \\ &= AP[k]A^T + \big(L - AP[k]C^TR_\epsilon^{-1}\big)R_\epsilon\big(L - AP[k]C^TR_\epsilon^{-1}\big)^T \\ &\qquad - AP[k]C^TR_\epsilon^{-1}CP[k]^TA^T + FR_WF^T. \end{aligned}$$

In order to minimize this expression, we choose $L = AP[k]C^TR_\epsilon^{-1}$ and the theorem is proven. □

Note that the Kalman filter has the form of a *recursive* filter: given $P[k] = E\{E[k]E[k]^T\}$ at time k, can compute how the estimate and covariance *change*. Thus we do not need to keep track of old values of the output. Furthermore, the Kalman filter gives the estimate $\hat{X}[k]$ *and* the covariance $P[k]$, so we can see how reliable the estimate is. It can also be shown that the Kalman filter extracts the maximum possible information about output data. It can be shown that for the Kalman filter the correlation matrix for the error is

$$R_E[j,k] = R\delta_{jk}.$$

In other words, the error is a white noise process, so there is no remaining dynamic information content in the error.

In the special case when the noise is stationary (R_W, R_V constant) and *if* $P[k]$ converges, then the observer gain is constant:

$$L = APC^T(R_V + CPC^T),$$

where P satisfies

$$P = APA^T + FR_WF^T - APC^T\bigl(R_V + CPC^T\bigr)^{-1}CPA^T.$$

We see that the optimal gain depends on both the process noise and the measurement noise, but in a nontrivial way. Like the use of LQR to choose state feedback gains, the Kalman filter permits a systematic derivation of the observer gains given a description of the noise processes. The solution for the constant gain case is solved by the `dlqe` command in MATLAB.

6.3 Predictor-Corrector Form

The Kalman filter can be written in a two step form by separating the correction step (where we make use of new measurements of the output) and the prediction step (where we compute the expected state and covariance at the next time instant).

We make use of the notation $\hat{X}[k|j]$ to represent the estimated state at time instant k given the information up to time j (where typically $j = k-1$). Using this notation, the filter can be solved using the following algorithm:

Step 0: Initialization.

$$\begin{aligned} k &= 1 \\ \hat{X}[0|0] &= E\{X[0]\} \\ P[0|0] &= E\{X[0]X^T[0]\} \end{aligned}$$

Step 1: Prediction. Update the estimates and covariance matrix to account for all data taken up to time $k-1$:

$$\begin{aligned} \hat{X}[k|k{-}1] &= A\hat{X}[k{-}1|k{-}1] + Bu[k-1] \\ P[k|k{-}1] &= AP[k{-}1|k{-}1]A^T + FR_W[k-1]F^T \end{aligned}$$

Step 2: Correction. Correct the estimates and covariance matrix to account for the data taken at time step k:

$$\begin{aligned} L[k] &= P[k|k-1]C^T(R_V + CP[k|k-1]C^T)^{-1}, \\ \hat{X}[k|k] &= \hat{X}[k|k-1] + L[k](Y[k] - C\hat{X}[k|k-1]), \\ P[k|k] &= P[k|k-1] - L[k]CP[k|k-1]. \end{aligned}$$

Step 3: Iterate. Set k to $k+1$ and repeat steps 1 and 2.

Note that the correction step reduces the covariance by an amount related to the relative accuracy of the measurement, while the prediction step increases the covariance by an amount related to the process disturbance.

This form of the discrete-time Kalman filter is convenient because we can reason about the estimate in the case when we do not obtain a measurement on every iteration of the algorithm. In this case, we simply update the prediction step (increasing the covariance) until we receive new sensor data, at which point we call the correction step (decreasing the covariance).

The following lemma will be useful in the sequel:

Lemma 6.3. *The optimal gain $L[k]$ satisfies*

$$L[k] = P[k|k]C^T R_V^{-1}$$

Proof. $L[k]$ is defined as

$$L[k] = P[k|k-1]C^T(R_V + CP[k|k-1]C^T)^{-1}.$$

Multiplying through by the inverse term on the right and expanding, we have

$$\begin{aligned} L[k](R_V + CP[k|k-1]C^T) &= P[k|k-1]C^T, \\ L[k]R_V + L[k]CP[k|k-1]C^T &= P[k|k-1]C^T, \end{aligned}$$

and hence

$$\begin{aligned} L[k]R_V &= P[k|k-1]C^T - L[k]CP[k|k-1]C^T, \\ &= (I - L[k]C)P[k|k-1]C^T = P[k|k]C^T. \end{aligned}$$

The desired results follows by multiplying on the right by ${R_V}^{-1}$. □

6.4 Sensor Fusion

We now return to the main topic of the chapter: sensor fusion. Consider the situation described in Figure 6.1, where we have an input/output dynamical system with multiple sensors capable of taking measurements. The problem of sensor fusion involves deciding how to best combine the measurements from the individual sensors in order to accurately estimate the process state X. Since different sensors may have different noise characteristics, evidently we should combine the sensors in a way that places more weight on sensors with lower noise. In addition, in some situations we may have different sensors available at different times, so that not all information is available on each measurement update.

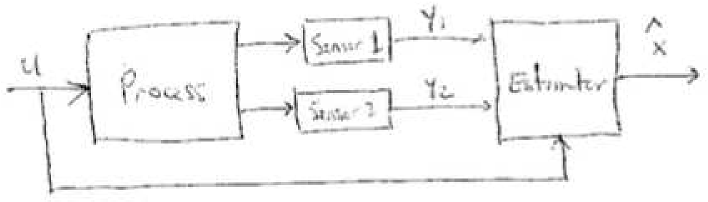


Figure 6.1: Sensor fusion

To gain more insight into how the sensor data are combined, we investigate the functional form of $L[k]$. Suppose that each sensor takes a measurement of the form

$$Y^i = C^i X + V^i, \qquad i = 1, \dots, p,$$

where the superscript i corresponds to the specific sensor. Let V^i be a zero mean, white noise process with covariance $\sigma_i^2 = R_{V^i}(0)$. It follows from Lemma 6.3 that

$$L[k] = P[k|k] C^T R_W^{-1}.$$

First note that if $P[k|k]$ is small, indicating that our estimate of X is close to the actual value (in the MMSE sense), then $L[k]$ will be small due to the leading $P[k|k]$ term. Furthermore, the characteristics of the individual sensors are contained in the different σ_i^2 terms, which only appears in R_W. Expanding the gain matrix, we have

$$L[k] = P[k|k] C^T R_W^{-1}, \qquad R_W^{-1} = \begin{bmatrix} 1/\sigma_1^2 & & \\ & \ddots & \\ & & 1/\sigma_p^2 \end{bmatrix}.$$

We see from the form of R_W^{-1} that each sensor is inversely weighted by its covariance. Thus noisy sensors ($\sigma_i^2 \gg 1$) will have a small weight and require averaging over many iterations before their data can affect the state estimate. Conversely, if $\sigma_i^2 \ll 1$, the data is "trusted" and is used with higher weight in each iteration.

6.5 Information Filters

An alternative formulation of the Kalman filter is to make use of the inverse of the covariance matrix, called the *information matrix*, to represent the error of the estimate. It turns out that writing the state estimator in this form has several advantages both conceptually and when implementing distributed computations. This form of the Kalman filter is known as the *information filter*.

We begin by defining the information matrix I and the weighted state estimate $\hat{Z}$:

$$I[k|k] = P^{-1}[k|k], \qquad \hat{Z}[k|k] = P^{-1}[k|k]\hat{X}[k|k].$$

We also make use of the following quantities, which appear in the Kalman filter equations:

$$\Omega_i[k|k] = (C^i)^T R_{W^i}^{-1}[k|k] C^i, \qquad \Psi_i[k|k] = (C^i)^T R_{W^i}^{-1}[k|k] C^i \hat{X}[k|k].$$

Using these quantities, we can rewrite the Kalman filter equations as

$$\begin{array}{ll} \text{Prediction} & \text{Correction} \\ I[k|k-1] = \left(AI^{-1}[k-1|k-1]A^T + R_W\right)^{-1}, & I[k|k] = I[k|k-1] + \sum_{i=1}^{p} \Omega_i[k|k], \\ \hat{Z}[k|k-1] = I[k|k-1]AI^{-1}[k-1|k-1]\hat{Z}[k-1|k-1] + Bu[k-1], & \hat{Z}[k|k] = \hat{Z}[k|k-1] + \sum_{i=1}^{p} \Psi_i[k|k]. \end{array}$$

Note that these equations are in a particularly simple form, with the information matrix being updated by each sensor's Ω_i and similarly the state estimate being updated by each sensors Ψ_i.

Remarks:

1. Information form allows simple addition for correction step. Intuition: add information through additional data.
2. Sensor fusion: information content = inverse covariance (for each sensor)
3. Variable rate: incorporate new information whenever it arrives. No data $\implies$ no information $\implies$ prediction update only.

6.6 Additional topics

Converting continuous time stochastic systems to discrete time

$$\dot{X} = AX + Bu + Fw$$

$$\begin{aligned} x(t+h) &\approx x(t) + h\dot{x}(t) \\ &= x(t) + hAx(t) + hBu(t) + hFW(t) \\ &= (I + hA)X(t) + (hB)u(t) + (hF)W(t) \end{aligned}$$

$$X[k+1] = \underbrace{(I + hA)}_{\tilde{A}} X[k] + \underbrace{(bB)}_{\tilde{B}} u[k] + \underbrace{(hF)}_{\tilde{F}} W[k].$$

Correlated disturbances and noise

As in the case of continuous-time Kalman filters, in the discrete time we can include noise or disturbances that are non-white by using a filter to generate noise with the appropriate correlation function.

On practical method to do this is to collect samples $W[1], W[2], \dots, W[N]$ and then numerically compute the correlation function

$$R_W(l) = E\{W[i]W[i+l]\} = \frac{1}{N-l} \sum_{j=1}^{N-l} W[j]W[j+l].$$

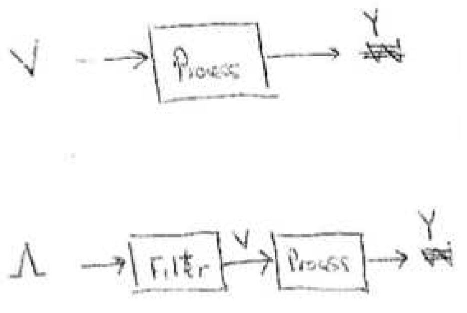
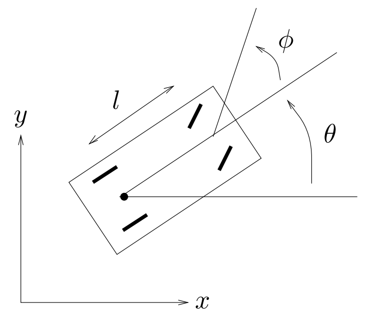


Figure 6.2: Sensor fusion with correlated measurement noise

6.7 Further Reading

Exercises

6.1 Consider the problem of estimating the position of an autonomous mobile vehicle using a GPS receiver and an IMU (inertial measurement unit). The continuous time dynamics of the vehicle are given by



$$\dot{x} = \cos\theta\, v$$
$$\dot{y} = \sin\theta\, v$$
$$\dot{\theta} = \frac{1}{\ell}\tan\phi\, v,$$

We assume that the vehicle is disturbance free, but that we have noisy measurements from the GPS receiver and IMU and an initial condition error.

(a) Rewrite the equations of motion in discrete time, assuming that we update the dynamics at a sample time of $h = 0.005$ sec and that we can take $\dot{x}$ to be roughly constant over that period. Run a simulation of your discrete time model from initial condition $(0, 0, 0)$ with constant input $\phi = \pi/8$, $v = 5$ and compare your results with the continuous time model.

(b) Suppose that we have a GPS measurement that is taken every 0.1 seconds and an IMU measurement that is taken every 0.01 seconds. Write a MATLAB program that that computes the discrete time Kalman filter for this system, using the same disturbance, noise and initial conditions as Exercise 5.1.

6.2 Consider a continuous time dynamical system with multiple measurements,

$$\dot{X} = AX + Bu + FV, \qquad Y^i = C^i x + W^i, \quad i = 1, \dots, q.$$

Assume that the measurement noises W^i are indendendent for each sensor and have zero mean and variance σ_i^2. Show that the optimal estimator for X weights the measurements by the inverse of their covariances.

6.3 Show that if we formulate the optimal estimate using an estimator of the form

$$\hat{X}[k+1] = A\hat{X}[k] + L[k](Y[k+1] - CA\hat{X}[k])$$

that we recover the update law in the predictor-corrector form.

Bibliography

[AF06] M. Athans and P. L. Falb. *Optimal Control: An Introduction to the Theory and Its Applications*. Dover, 2006. Originally published in 1963.

[ÅM08] K. J. Åström and R. M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2008. Available at `http://www.cds.caltech.edu/~murray/amwiki`.

[Åst06] K. J. Åström. *Introduction to Stochastic Control Theory*. Dover, New York, 2006. Originally published by Academic Press, New York, 1970.

[BH75] A. E. Bryson, Jr. and Y.-C. Ho. *Applied Optimal Control: Optimization, Estimation, and Control*. Wiley, New York, 1975.

[Bro81] R. W. Brockett. Control theory and singular Riemannian geometry. In *New Directions in Applied Mathematics*, pages 11–27. Springer-Verlag, New York, 1981.

[Bry99] A. E. Bryson. *Dynamic optimization*. Addison Wesley, 1999.

[dB78] C. de Boor. *A Practical Guide to Splines*. Springer-Verlag, 1978.

[FLMR92] M. Fliess, J. Levine, P. Martin, and P. Rouchon. On differentially flat nonlinear systems. *Comptes Rendus des Séances de l'Académie des Sciences*, 315:619–624, 1992. Serie I.

[Fri04] B. Friedland. *Control System Design: An Introduction to State Space Methods*. Dover, New York, 2004.

[GMSW] P. E. Gill, W. Murray, M. A. Saunders, and M. Wright. *User's Guide for NPSOL 5.0: A Fortran Package for Nonlinear Programming*. Systems Optimization Laboratory, Stanford University, Stanford, CA 94305.

[HO01] J. Hauser and H. Osinga. On the geometry of optimal control: The inverted pendulum example. In *American Control Conference*, 2001.

[HP87] C. Hargraves and S. Paris. Direct trajectory optimization using nonlinear programming and collocation. *AIAA J. Guidance and Control*, 10:338–342, 1987.

[HPS71] P. G. Hoel, S. C. Port, and C. J. Stone. *Introduction to Probability Theory*. Houghton Mifflin Company, 1971.

[Isi89] A. Isidori. *Nonlinear Control Systems*. Springer-Verlag, 2nd edition, 1989.

[Jad01] A. Jadbabaie. *Nonlinear Receding Horizon Control: A Control Lyapunov Function Approach*. PhD thesis, California Institute of Technology, Control and Dynamical Systems, 2001.

[JSK99] M. Jankovic, R. Sepulchre, and P. V. Kokotović. CLF based designs with robustness to dynamic input uncertainties. *Systems Control Letters*, 37:45–54, 1999.

[JYH01] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776–783, 2001.

[Kal64] R. E. Kalman. When is a linear control system optimal? *J. Basic Engrg. Trans. ASME Ser. D*, 86:51–60, 1964.

[KKK95] M. Krstić, I. Kanellakopoulos, and P. Kokotović. *Nonlinear and Adaptive Control Design*. Wiley, 1995.

[KKM91] I. Kanellakopoulos, P. V. Kokotovic, and A. S. Morse. Systematic design of adaptive controllers for feedback linearizable systems. *IEEE Transactions on Automatic Control*, 36(11):1241–1253, 1991.

[KV86] P. R. Kumar and P. Varaiya. *Stochastic Systems: Estimation, Identification, and Adaptive Control*. Prentice Hall, Inc., 1986.

[LM67] E. B. Lee and L. Markus. *Foundations of Optimal Control Theory*. Robert E. Krieger Publishing Company, 1967.

[LS95] F. L. Lewis and V. L. Syrmos. *Optimal Control*. Wiley, second edition, 1995.

[Lue97] David G. Luenberger. *Optimization by Vector Space Methods*. Wiley, New York, 1997.

[MA73] P. J. Moylan and B. D. O. Anderson. Nonlinear regulator theory and an inverse optimal control problem. *IEEE Trans. on Automatic Control*, 18(5):460–454, 1973.

[MDP94] P. Martin, S. Devasia, and B. Paden. A different look at output tracking—Control of a VTOL aircraft. *Automatica*, 32(1):101–107, 1994.

[MFHM05] M. B. Milam, R. Franz, J. E. Hauser, and R. M. Murray. Receding horizon control of a vectored thrust flight experiment. *IEE Proceedings on Control Theory and Applications*, 152(3):340–348, 2005.

[MHJ$^+$03] R. M. Murray, J. Hauser, A. Jadbabaie, M. B. Milam, N. Petit, W. B. Dunbar, and R. Franz. Online control customization via optimization-based control. In T. Samad and G. Balas, editors, *Software-Enabled Control: Information Technology for Dynamical Systems*. IEEE Press, 2003.

[MM99] M. B. Milam and R. M. Murray. A testbed for nonlinear flight control techniques: The Caltech ducted fan. In *Proc. IEEE International Conference on Control and Applications*, 1999.

[MMM00] M. B. Milam, K. Mushambi, and R. M. Murray. A computational approach to real-time trajectory generation for constrained mechanical systems. In *Proc. IEEE Control and Decision Conference*, 2000.

[MRRS00] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[Mur97] R. M. Murray. Nonlinear control of mechanical systems: A Lagrangian perspective. *Annual Reviews in Control*, 21:31–45, 1997.

[PBGM62] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. *The Mathematical Theory of Optimal Processes*. Wiley-Interscience, 1962. (translated from Russian).

[PND99] J. A. Primbs, V. Nevistić, and J. C. Doyle. Nonlinear optimal control: A control Lyapunov function and receding horizon perspective. *Asian Journal of Control*, 1(1):1–11, 1999.

[QB97] S. J. Qin and T. A. Badgwell. An overview of industrial model predictive control technology. In J.C. Kantor, C.E. Garcia, and B. Carnahan, editors, *Fifth International Conference on Chemical Process Control*, pages 232–256, 1997.

[Rug90] W. J. Rugh. Analytical framework for gain scheduling. In *Proc. American Control Conference*, pages 1688–1694, 1990.

[Sey94] H. Seywald. Trajectory optimization based on differential inclusion. *J. Guidance, Control and Dynamics*, 17(3):480–487, 1994.

[Sha90] J. S. Shamma. Analysis of gain scheduled control for nonlinear plants. *IEEE Transactions on Automatic Control*, 35(12):898–907, 1990.

[SJK97] R. Sepulchre, M. Jankovic, and P. V. Kokotović. *Constructive Nonlinear Control*. Springer, London, 1997.

[Son83] E. D. Sontag. A Lyapunov-like characterization of asymptotic controllability. *SIAM Journal of Control and Optimization*, 21:462–471, 1983.

[vNM98] M. J. van Nieuwstadt and R. M. Murray. Rapid hover to forward flight transitions for a thrust vectored aircraft. *Journal of Guidance, Control, and Dynamics*, 21(1):93–100, 1998.

[vNRM98] M. van Nieuwstadt, M. Rathinam, and R. M. Murray. Differential flatness and absolute equivalence. *SIAM Journal of Control and Optimization*, 36(4):1225–1239, 1998.

Index

Note: Under construction! The indexing for the text has not yet been done and so this index contains an incomplete and unedited set of terms.