Optimization-Based Control

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

Version v2.3d (14 Jan 2023)

Chapter 3

Optimal Control

This chapter expands on Chapter 7 of FBS2e, which introduces the concepts of reachability and state feedback. We also expand on topics in Section 8.5 of FBS2e 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.

3.1 Review: Optimization

Optimization 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 $F\colon \mathbb{R}^n \to \mathbb{R}$. 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 3.1 gives a graphical interpretation of the necessary condition for a minimum. Note that these are *not* sufficient conditions; the points x_1, x_2, and x^* in the figure

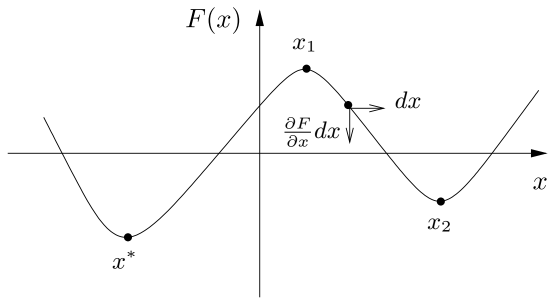


Figure 3.1: Optimization of functions. The minimum of a function occurs at a point where the gradient is zero.

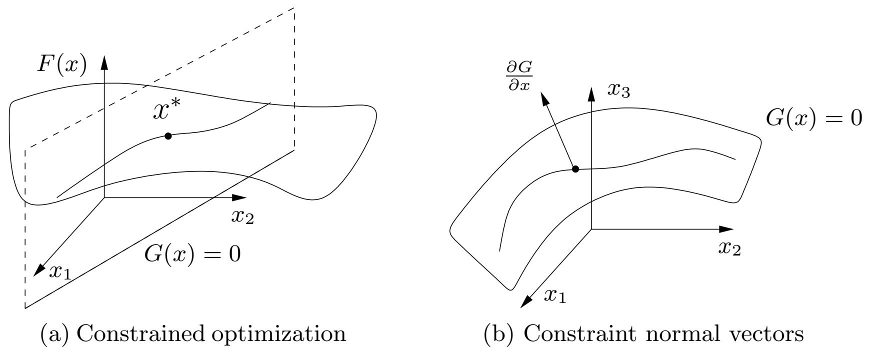

$F(x)$
x^*
x_2
x_1
$G(x) = 0$
(a) Constrained optimization
$\frac{\partial G}{\partial x}$
x_3
$G(x) = 0$
x_2
x_1
(b) Constraint normal vectors

Figure 3.2: 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).

all satisfy the necessary condition but only one is the (global) minimum.

The situation is more complicated if constraints are present. Let $G_i \colon \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, k$ 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 $x \in \{x \in \mathbb{R}^n \colon G_i(x) = 0, i = 1, \dots, k\}$. This situation can be 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 3.2a.

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 3.2b. The tangent directions to the surface can be computed by considering small perturbations of the constraint that remain on the surface:

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

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

$$\frac{\partial F}{\partial x} + \lambda^\mathsf{T} \frac{\partial G}{\partial x} = 0 \tag{3.1}$$

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

$$\min_x \left(F(x) + \lambda^\mathsf{T} G(x) \right). \tag{3.2}$$

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 3.1 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 numerically solving optimization problems of this sort. The NPSOL, SNOPT, and IPOPT libraries are available in FORTRAN and C. In Python, the scipy.optimize module of SciPy can be used to solve constrained optimization problems.

3.2 Optimal Control of Systems

We now return to the problem of finding a feasible trajectory for a system that satisfies some performance condition. The basic idea is to try to optimize a given cost function over all trajectories of a system that satisfy the possible dynamics of the system. The input to the system is used to "parameterize" the possible trajectories of the system.

More concretely, we consider the *optimal control problem*:

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

subject to the constraint

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

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

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

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 $g(z) = \dot{x}(t) - f(x(t),u(t)) = 0$ 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:

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

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 $x(T) = x_\mathrm{f}$ 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 $x(T) = x_\mathrm{f}$, 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 is available for the optimal control problem that captures most of these special cases in a unifying framework. Consider a nonlinear system

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

where $f(x,u) = (f_1(x,u), \dots f_n(x,u))\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. 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 $\psi\colon \mathbb{R}^n \to \mathbb{R}^q$ gives a set of q terminal constraints. Analogous to the case of optimizing a function subject to constraints, we construct the *Hamiltonian*:

$$H = L + \lambda^{\mathsf{T}} f = L + \sum_{i=1}^{n} \lambda_i f_i.$$

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

$$\begin{aligned}
\dot{x}_i^* &= \frac{\partial H}{\partial \lambda_i}(x^*, \lambda^*), && x(0) \text{ given, } \psi(x(T)) = 0,\\
-\dot{\lambda}_i^* &= \frac{\partial H}{\partial x_i}(x^*, \lambda^*), && \lambda_i^*(T) = \frac{\partial V}{\partial x_i}\big(x^*(T)\big) + \sum_{j=1}^{q} \nu_j^* \frac{\partial \psi_j}{\partial x_i}\big(x^*(T)\big),
\end{aligned}$$

and

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

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

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

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

$$\frac{\partial H}{\partial u_i} = 0, \qquad i = 1, \dots, m.$$

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:

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

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

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

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

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

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:

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

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

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

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. □

3.3 Examples

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

Example 3.2 Scalar linear system
Consider the optimal control problem for the system

$$\dot{x} = ax + bu, \tag{3.3}$$

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

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

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

$$L = \tfrac{1}{2}u^2(t), \qquad V = \tfrac{1}{2}cx^2(t_\text{f}).$$

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

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

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

$$\lambda(t) = cx(t_\text{f})e^{a(t_\text{f}-t)}.$$

The optimal control is given by

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

Substituting this control into the dynamics given by equation (3.3) yields a first-order ODE in x:

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

This can be solved explicitly as

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

Setting $t = t_\text{f}$ and solving for $x(t_\text{f})$ gives

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

and finally we can write

$$u^*(t) = -\frac{2abc\, e^{a(2t_\mathrm{f}-t_o-t)}x(t_o)}{2a - b^2c\left(1 - e^{2a(t_\mathrm{f}-t_o)}\right)}, \tag{3.4}$$

$$x^*(t) = x(t_o)e^{a(t-t_o)} + \frac{b^2c\, e^{a(t_\mathrm{f}-t_o)}x(t_o)}{2a - b^2c\left(1 - e^{2a(t_\mathrm{f}-t_o)}\right)}\left[e^{a(t_\mathrm{f}-t)} - e^{a(t+t_\mathrm{f}-2t_o)}\right]. \tag{3.5}$$

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 $x^*(t_\mathrm{f})$ as $c \to \infty$. Setting $t = t_\mathrm{f}$ in equation (3.5) and taking the limit we find that

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

∇

Example 3.3 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 $x_\mathrm{f} = 0$. We choose the cost functions and terminal constraints to satisfy

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

In this case, the time T is not fixed and so it turns out that one additional condition is required for the optimal controller. For the case considered in Theorem 3.1, where the cost functions and constraints do not depend explictly on time, the additional condition is

$$H(x(T), u(T)) = 0$$

(see [LS95]).

To find the optimal control, we form the Hamiltonian

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

Now apply the conditions in the maximum principle:

$$\begin{aligned}
\dot{x} &= \left(\frac{\partial H}{\partial \lambda}\right)^\mathsf{T} = Ax + Bu,\\
-\dot{\lambda} &= \left(\frac{\partial H}{\partial x}\right)^\mathsf{T} = A^\mathsf{T}\lambda,\\
u &= \arg\min\, H = -\operatorname{sgn}(\lambda^\mathsf{T}B),\\
& 1 + \lambda^\mathsf{T}(T)(Ax(T) + Bu(T)) = 0.
\end{aligned}$$

The optimal solution always satisfies this set of equations (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 $\lambda^\mathsf{T} B$. This type of control is called "bang-bang" control since the input is always on one of its limits. If $\lambda^\mathsf{T}(t)B = 0$ then the control is not well defined, but if this is only true for a specific time instant (e.g., $\lambda^\mathsf{T}(t)B$ crosses zero) then the analysis still holds. ∇

3.4 Implementation in Python[1]

The `optimal` module of the Python Control Systems Library (python-control) provides support for optimization-based controllers for nonlinear systems with state and input constraints.

The optimal control module provides a means of computing optimal trajectories for nonlinear systems and implementing optimization-based controllers, including model predictive control (described in Chapter 4). The basic optimal control problem is to solve the optimization

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

subject to the constraint

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

Constraints on the states and inputs along the trajectory and at the end of the trajectory can also be specified:

$$\begin{aligned} \mathrm{lb}_i \leq g_i(x,u) \leq \mathrm{ub}_i, &\qquad i = 1,\dots,k \\ \psi_i(x(T)) = 0, &\qquad i = 1,\dots,q. \end{aligned}$$

The python-control implementation of optimal control follows the basic problem setup described here, but carries out all computations in discrete time (so that integrals become sums) and over a finite horizon.

To describe an optimal control problem we need an input/output system, a time horizon, a cost function, and (optionally) a set of constraints on the state and/or input, either along the trajectory and at the terminal time. The optimal control module operates by converting the optimal control problem into a standard optimization problem that can be solved by `scipy.optimize.minimize()`. The optimal control problem can be solved by using the `solve_ocp()` function:

```
res = obc.solve_ocp(sys, timepts, X0, cost, constraints)
```

The `sys` parameter should be an `InputOutputSystem` and the `timepts` parameter should represent a time vector that gives the list of times at which the cost and constraints should be evaluated.

The cost function has call signature `cost(x, u)` and should return the (incremental) cost at the given time, state, and input. It will be evaluated at each point in the time points vector. The `terminal_cost` parameter can be used to specify a cost function for the final point in the trajectory.

The constraints parameter is a list of constraints similar to that used by the `scipy.optimize.minimize()` function. Each constraint is in one of the following forms:

[1]The material in this section is drawn from [FGM+21].

```
LinearConstraint(A, lb, ub)
NonlinearConstraint(f, lb, ub)
```

For a linear constraint, the 2D array `A` is multiplied by a vector consisting of the current state `x` and current input `u` stacked vertically, then compared with the upper and lower bound. This constraint is satisfied if

```
lb <= A @ np.hstack([x, u]) <= ub
```

A nonlinear constraint is satisfied if

```
lb <= f(x, u) <= ub
```

By default, constraints are taken to be trajectory constraints holding at all points on the trajectory. The `terminal_constraint` parameter can be used to specify a constraint that only holds at the final point of the trajectory.

The return value for `solve_ocp()` is a bundle object that has the following elements:

`res.success`: `True` if solved successfully
`res.inputs`: optimal input
`res.states`: state trajectory (if `return_x == True`)
`res.time`: copy of the time points vector

In addition, the results from `scipy.optimize.minimize()` are also available.

Providing a reasonable initial guess is often needed in order for the optimizer to find a good answer. The `initial_guess` parameter provides trajectories for the initial states and/or inputs to use as a guess for the optimal trajectory. For collocation methods, the initial guess is either the input vector or a 2-tuple consisting guesses for the state and the input. For shooting methods, an array of inputs for each time point should be specified. For an optimal control problem that uses a terminal condition or a large terminal cost to get to a neighborhood of a final point, a straight line in the state space often works well.

3.5 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

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

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 begin by computing the Hamiltonian H:

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

Applying the results of Theorem 3.1, we obtain the necessary conditions

$$\begin{aligned} \dot{x} &= \left(\frac{\partial H}{\partial \lambda}\right)^{\mathsf{T}} = Ax + Bu, & x(0) &= x_0, \\ -\dot{\lambda} &= \left(\frac{\partial H}{\partial x}\right)^{\mathsf{T}} = Q_x x + A^{\mathsf{T}}\lambda, & \lambda(T) &= P_1 x(T), \\ 0 &= \left(\frac{\partial H}{\partial u}\right)^{\mathsf{T}} = Q_u u + B^{\mathsf{T}}\lambda. \end{aligned} \tag{3.6}$$

The last condition can be solved to obtain the optimal controller

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

which can be substituted into the dynamic equation (3.6) 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

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

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

$$-\dot{P} = PA + A^{\mathsf{T}}P - PBQ_u^{-1}B^{\mathsf{T}}P + Q_x, \qquad P(T) = P_1. \tag{3.7}$$

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 (3.7) 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 (3.7), we can apply the optimal input

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

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 optimally move from *any* state toward the origin in time T.

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

$$J = \int_0^\infty (x^\mathsf{T} Q_x x + u^\mathsf{T} Q_u u)\, dt. \tag{3.8}$$

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 (3.7). In other words, we seek to find P such that

$$PA + A^\mathsf{T} P - PBQ_u^{-1} B^\mathsf{T} P + Q_x = 0. \tag{3.9}$$

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

$$u = -Q_u^{-1} B^\mathsf{T} P x.$$

This represents a *constant gain* $K = Q_u^{-1} B^\mathsf{T} P$ 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 toward 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 (3.9). This quadratic equation can be solved numerically: in python-control the command `K, P, E = ct.lqr(A, B, Qx, Qu)` provides the optimal feedback compensator K, the solution to the algebraic Riccati equation P, and the closed loop eigenvalues for the system E.

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 $Q_x = H^\mathsf{T} H$ (always possible), our cost function becomes

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

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 3.2. *Consider a linear control system with quadratic cost:*

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

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

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

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

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 3.4 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 (3.9). 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

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

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

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

The characteristic polynomial of this matrix is

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

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

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

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 FBS2e). ∇

3.6 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 (3.8): (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

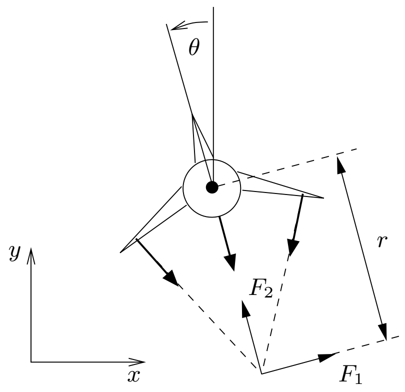
$$Q_x = H^{\mathsf{T}} H \qquad Q_u = \rho I.$$

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

We illustrate the various choices through an example application.

Example 3.5 Thrust vectored aircraft
Consider the thrust vectored aircraft example introduced in FBS2e, Example 3.12. The system is shown in Figure 3.3, reproduced from FBS2e. The linear quadratic regulator problem was illustrated in FBS2e 2e, Example 7.9, where the weights were chosen as $Q_x = I$ and $Q_u = \rho I$. Figure 3.4 reproduces the step response

(a) Harrier "jump jet"



(b) Simplified model

Figure 3.3: 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.

for this case, but using the full nonlinear, multi-input/multi-output model for the system.

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 to be equivalently bad. In addition, we wish to penalize the forces in the sidewards direction (F_1) since these result in a loss in efficiency. This can be accounted for in the LQR weights be choosing

$$Q_x = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 36/\pi & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad Q_u = \begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}.$$

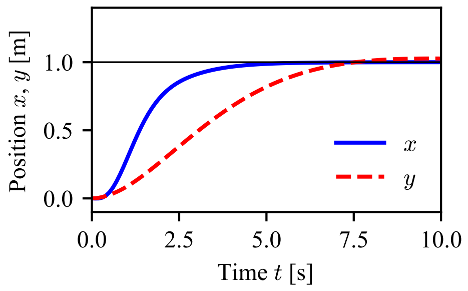
The results of this choice of weights are shown in Figure 3.5. ∇

3.7 Advanced Topics

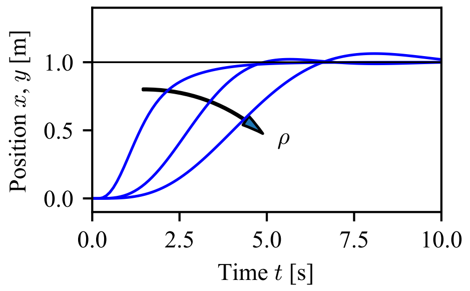
In this section we briefly touch on some related topics in optimal control, with reference to more detailed treatments where appropriate.

Dynamic programming

An alternative formulation to the optimal control problem is to make use of the "principle of optimality", which states (roughly) that if we are given an optimal

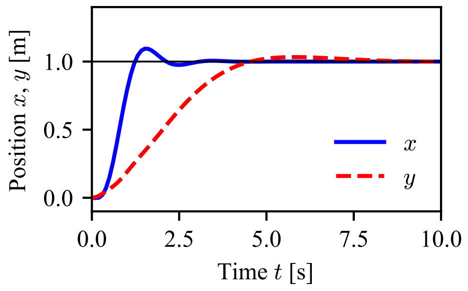


(a) Step response in x and y

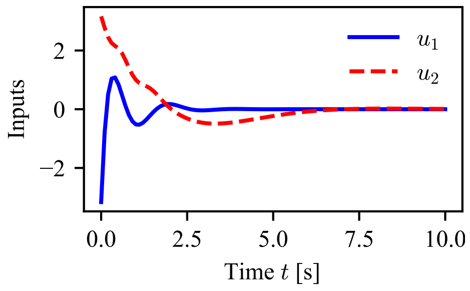


(b) Effect of control weight ρ

Figure 3.4: 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 3.5: Step response for a vector thrust aircraft using physically motivated LQR weights (a). The rise time for x is much faster than in Figure 3.4a, but there is a small oscillation and the inputs required are quite large (b).

policy, then the optimal policy from any point along the implementation of that policy must also be optimal. In the context of optimal trajectory generation, we can interpret this as saying that if we solve an optimal control problem from any state along an optimal trajectory, we will get the remainder of the optimal trajectory (how could it be otherwise!?).

The implication of this statement for trajectory generation is that we can work from the final time of our optimal control problem and compute the cost by moving backwards in time until we reach the initial time. Toward this end, we define the "cost to go" from a given state x at time t as

$$J(x,t) = \int_t^T L(x(\tau), u(\tau))\, d\tau + V(x(T)). \tag{3.10}$$

Given a state $x(t)$, We see that the cost at time T is given by $J(T,x) = V(x(T))$ and the cost at other times includes the integral of the cost from time t to T plus the terminal cost.

It can be shown that a necessary condition for a trajectory $x(\cdot)$, $u(\cdot)$ to be optimal is that the *Hamilton-Jacobi-Bellman equation* (HJB equation) be satisfied:

$$\frac{\partial J^*}{\partial t}(x,t) = -H(x, u^*, \frac{\partial J^{*\mathsf{T}}}{\partial x}(x,t)), \qquad J(x,T) = V(x), \tag{3.11}$$

where H is the Hamiltonian function, u^* is the optimal input, and $V : \mathbb{R}^n \to \mathbb{R}$ is the terminal cost. As in the case of the maximum principle, we choose u^* to minimize the Hamiltonian:

$$u^* = \arg\min_u H\big(x^*, u, \frac{\partial J^{*\mathsf{T}}}{\partial x}(x^*, u)\big).$$

Equation (3.11) is a partial differential equation for $J(x,t)$ with boundary condition $J(x,T) = V(x)$.

From the form of the Hamilton-Jacobi-Bellman equation, we see that we can interpret the costate variables λ as

$$\lambda^T = \frac{\partial J^*}{\partial x}(x,t).$$

Thus the costate variables can be thought of as the sensitivity of the cost to go at a given point along the optimal trajectory. This interpretation allows some alternative formulations of the optimal control problem, as well as additional insights.

While solving the Hamilton-Jacobi-Bellman equation is not particularly easy in the continuous case, it turns out that discrete version of the problem can make good use of the principle of optimality. In particular, for problems with variables that take on a sequence of values from a finite set (known as discrete-decision making problems), we can compute the optimal set of decisions by starting with the cost at the end of the sequence and then computing the values of the optimized cost stepping backwards in time. This technique is known as *dynamic programming* and arises in a number of different applications in computer science, economics, and other areas.

Detailed explanations of dynamic programming formulations of optimal control are available in a wide variety of textbooks. The online notes from Daniel Liberzon are a good open source resource for this material:

`http://liberzon.csl.illinois.edu/teaching/cvoc/cvoc.html`

Chapter 5 in those notes provides a more detailed explanation of the material briefly summarized here.

Integral action

Controllers based on state feedback achieve the correct steady-state response to command signals by having a good model of the system that can generate the proper feedforward signal u_d. However, if the model is incorrect or disturbances are present, there may be steady state errors. Integral feeback is a classic technique for achieving zero steady state output error in the presence of constant disturbances or errors in the feedforward signal.

The basic approach in integral feedback is to create a state within the controller that computes the integral of the error signal, which is then used as a feedback term. We do this by augmenting the description of the system with a new state z, which is the integral of the difference between the the actual output $y = h(x)$ and desired output $y_\text{d} = h(x_\text{d})$. The augmented state equations become

$$\frac{d\xi}{dt} = \frac{d}{dt}\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} f(x,u) \\ h(x) - h(x_d) \end{bmatrix} =: F(\xi, u, x_\text{d}). \tag{3.12}$$

We can now design a feedback compensator (such as an LQR controller) that stabilizes the system to the desired trajectory $\xi_\text{d} = (x_\text{d}, 0)$, which will cause y to approach $h(x_\text{d})$ in steady state.

Given the augmented system, we design a state space controller in the usual fashion, with a control law of the form

$$u = u_\text{d} - K(\xi - \xi_\text{d}) = u_\text{d} - K\begin{bmatrix} x - x_\text{d} \\ z \end{bmatrix}, \tag{3.13}$$

where K is state feedback term for the augmented system (3.12).

Integral action can be included in the python-control `create_statefbk_iosystem` function using the `integral_action` keyword. The value of this keyword can either be an matrix (2D array) or a function. If a matrix C is specified, the difference between the desired state and system state will be multiplied by this matrix and integrated. The controller gain should then consist of a set of proportional gains K_p and integral gains K_i with

$$K = \begin{bmatrix} K_\text{p} \\ K_\text{i} \end{bmatrix}.$$

If `integral_action` is a function h, that function will be called with the signature `h(t, x, u, params)` to obtain the outputs whose error should be integrated. The number of output errors that are to be integrated must match the number of additional columns in the K matrix. If an estimator is specified, $\hat{x}$ will be used in place of x. Gain scheduling can also be used, as described in Section 2.4.

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 3.6 Nonholonomic integrator
Consider the minimum time optimization problem for the nonholonomic integrator introduced in Example 2.2 with input constraints $|u_i| \leq 1$. The Hamiltonian for the 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

$$\dot{\lambda}_1 = 0, \qquad \dot{\lambda}_2 = \lambda_3 x_2, \qquad \dot{\lambda}_3 = 0. \tag{3.14}$$

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

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

These equations are well-defined as long as the arguments of $\operatorname{sgn}(\cdot)$ 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 $x_\mathrm{f} = (\rho, 0, 0)$ 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. ∇

3.8 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]. A modern engineering textbook that contains a very compact and concise derivation of the key results in optimal control is the book by Lewis, Vrabie, and Syrmos [LVS12]. Finally, the online (and open access) notes by Daniel Liberzon [Lib10] provide detailed coverage of all of the topics in this chapter.

Exercises

3.1. (a) Let $G_1, G_2, \dots, G_k$ be a set of row vectors on $\mathbb{R}^n$. 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

$$F = \sum_{i=1}^{k} \lambda_i G_i.$$

(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_i, $i = 1, \dots, k$ such that

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

3.2. Consider the following control system

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

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

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

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

(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 $\tilde{Y}^{\mathsf{T}} = -\tilde{Y}$.

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

3.3. 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,

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

subject to to the initial and final state constraints. Use the maximum principle to show that the minimal length path is a straight line.

3.4. 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 3.2.) 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) Let $a = 1$, $b = 1$, $c = 1$, and $t_\text{f} - t_0 = 1$. Solve the optimal control problem numerically using python-control (or MATLAB) and compare it to your analytical solution by plotting the state and input trajectories for each solution. Take the initial conditions as $x(t_0) = 1$, 5, and 10.

(c) Suppose that we wish to have the final state to be exactly zero. Change the optimization problem to impose a final constraint instead of a final cost. Plot the state and input trajectories, and compare the computation times for the results in 0b to the computation time using a terminal constraint.

(d) 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).

(e) 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.

(f) 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.

(g) (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.

3.5. Repeat Exercise 3.4 using the system

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

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

3.6. 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.

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

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

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.

3.8. 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.

3.9. 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 3.2.

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?

3.10. Consider a linear system of the form

$$\frac{dx}{dt} = \begin{bmatrix} 0 & 2 \\ -1 & -0.1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (u + d)$$

where d is a disturbance input.

(a) Design an LQR controller for the system that regulates the system to a desired equilibrium point $x_\text{d} = (1, 0)$, assuming no disturbance ($d = 0$). Plot the response of the system starting from an initial condition $x(0) = (0, 0)$ and show that the system response converges to the desired equilibrium point.

(b) Assume now that $d = 0.1$. Show that the system response with your LQR controller from part (a) no longer converges to the desired equilibrium point and construct a controller using integral action that recovers the ability to drive the system state to that point.

(c) Suppose that there is uncertainty in the input matrix, so that the B matrix becomes

$$B = \begin{bmatrix} 0 \\ \gamma \end{bmatrix}, \qquad 0.5 \le \gamma \le 2.$$

Show that the uncertain system response with your original LQR controller no longer converges to the desired equilibrium (even with $d = 0$), but that the controller with integral action still causes the system to converge to the desired equilibrium. (OK to just show this for $\gamma = 0.5$ and 2.)

3.11. Consider the lateral control problem for an autonomous ground vehicle from Example 2.1. 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:

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

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.

3.12. Consider the dynamics of the vectored thrust aircraft described in Examples 2.4 and 3.5. The equations of motion are given by

$$\begin{aligned} m\ddot{x} &= F_1\cos\theta - F_2\sin\theta - c\dot{x}, \\ m\ddot{y} &= F_1\sin\theta + F_2\cos\theta - c\dot{y} - mg, \\ J\ddot{\theta} &= rF_1. \end{aligned} \tag{3.15}$$

with parameter values

$$m = 4 \text{ kg}, \quad J = 0.0475 \text{ kg m}^2, \quad r = 0.25 \text{ m}, \quad g = 9.8 \text{ m/s}^2, \quad c = 0.05 \text{ Ns/m},$$

which corresponds roughly to the values for the Caltech ducted fan flight control testbed.

We wish to generate an optimal trajectory for the system that corresponds to moving the system for an initial hovering position to a hovering position one meter to the right ($x_\text{f} = x_0 + 1$).

For each of the parts below, you should solve for the optimal input, simulate the (open loop) system, and plot the xy trajectory of the system, along with the angle θ and inputs F_1 and F_2 over the time interval. In addition, create a time and record the following information for each approach:

- the computation time required;
- the final position (for the open loop system);
- the weighted integrated cost of the input along the trajectory:

$$\int_0^T \big(10F_1^2(\tau) + (F_2 - mg)^2(\tau)\big)\, d\tau. \tag{3.16}$$

(a) Solve for an optimal trajectory using a quadratic cost from the final point with weights

$$Q_x = \text{diag}([1, 1, 10, 0, 0, 0]), \qquad Q_u = \text{diag}([10, 1]).$$

This cost function attempts to minimize the angular deviation θ and the sideways force F_1.

(b) Re-solve the problem using Bezier curves as the basis functions for the inputs. This should give you smoother inputs and a nicer response.

(c) Re-solve the problem using a terminal cost $V(x(T)) = x(T)^{\mathsf{T}} P_1 x(T)$ to try to get the system closer to the final value. You should try adjusting the cost along the trajectory Q_x versus the terminal cost P_1 to minimize the weighted integrated cost (3.16).

(d) Re-solve the problem using a terminal *constraint* to try to get the system closer to the final value. Adjust the cost along the trajectory to try to minimize the cost in equation (3.16). (Hint: you may have to use an initial guess to get the optimization to converge.)

(e) If $c = 0$, it can be shown that this system is differentially flat (see Example 2.4). Setting $c = 0$, re-solve the optimization problem using differential flatness. (The flatness mappings can be found in the file `pvtol.py`, available on the companion website.)