23 Discrete Optimal Control, and the Linear Quadratic Stochastic Regulator

There is no possibility of failure because you only control your actions and they only influence the probable evolution of your life over stochastic future paths. There is no failure, only feedback.
—Arthur De Vany

23.1 A version of Pontryagin for discrete optimal control

Up to this point we have only discussed optimal control for continuous variables, i.e. we have focused on state evolution equations of the form

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}, \mathbf{u}), \tag{23.1}$$

with some prescribed initial condition (or potentially endpoint condition). Although we would like to introduce stochastic optimal control in this setting, it would require a thorough discussion on Brownian motion, Ito Calculus, and the Kalman-Bucy filter none of which have been adequately covered in any of the material in these textbooks. For the sake of brevity, we will instead consider optimal control in a discrete setting instead which will *only* require knowledge of the discrete Kalman filter which you have already seen.

Our only motivation for introducing discrete systems isn't to include the stochastic terms however. In practice, all implementations of control are discrete, as CPU clock cycles are not really continuous (at least right now) and so the solution of any problem will ultimately rely on the solution of an underlying discretized system at some level. The question remains whether the discretization will take place at the beginning (such as is done here) at the level of the problem itself, or later on in the calculation of the solution.

All of this discussion is to say that throughout this Chapter we consider the evolution equation above under some time stepping mechanism, i.e. we will consider a discrete evolution equation of the form

$$\mathbf{x}_{k+1} = \mathbf{g}(\mathbf{x}_k, \mathbf{u}_k, k), \tag{23.2}$$

where we will use $\mathbf{g}$ for the discrete form of the evolution here, as it is certainly not the same as that appearing in (23.1). Note that we are not distinguishing between different time stepping mechanisms in this case, and may be ignoring the potential dependence of $\mathbf{g}$ on multiple previous time steps such as $\mathbf{x}_{n-1}$ etc. because this can be accounted for via a simple (yet deceiving) change of variables as shown in the homework.

Remark 23.1.1. Perhaps the most difficult aspect of treating discrete optimal control is the notation (sounds like a problem we have encountered before, right?). This is because we are interested in a set of temporal values for a vector valued $\mathbf{x}_k$, i.e. we want to find the value of $\mathbf{x}_k$ for all k in some finite set, but each of these can also be a vector valued quantity. This makes indexing a mess (in LaTex we refer to this as messy subscripting). For these reasons (and pure laziness) we will restrict a lot of the examples in this Chapter to the one-dimensional case, i.e. $x_k \in \mathbb{R}$, but it is important to recognize that this isn't necessary, and the ideas to multiple dimensions are immediate, if a little subscriptably messy.

Generically the discrete evolution equation is coupled with the goal of minimizing a cost function given by:

$$J[\mathbf{u}] = \sum_{k=0}^{T-1} L(\mathbf{x}_k, \mathbf{u}_k, k) + \phi(\mathbf{x}_T), \tag{23.3}$$

where the vector $\mathbf{u}$ is actually a concatenation of all of the vector valued control inputs $\mathbf{u}_k$ (a subscriptable nightmare remember). Note specifically that there is one more state variable $\mathbf{x}_k$ than there are control variables $\mathbf{u}_k$ (this is not referring to the dimension of each $\mathbf{x}_k$ and $\mathbf{u}_k$ specifically) because the first $\mathbf{x}_0$ is the initial condition for the state variable.

Remark 23.1.2. This is just to make certain there are enough remarks.

Just as we made a formal derivation for the Pontryagin maximum principle for the continuous case, we are going to similarly derive a set of necessary conditions for this discrete setting. Rather than keep track of all of the subscripting that comes in the multi-dimensional case, we are going to look at the one-dimensional state, and one-dimensional control version, i.e. we are interested in the discrete evolution equation

$$x_{k+1} = g(x_k, u_k, k), \tag{23.4}$$

and we seek to minimize the cost functional:

$$J[\mathbf{u}] = \sum_{k=1}^{T-1} L(x_k, u_k, k) + \phi(x_T). \tag{23.5}$$

To consider this, we will consider variations of the control $\mathbf{u} \to \mathbf{u}+\varepsilon\boldsymbol{\eta}$, where in this setting, $\boldsymbol{\eta} \in \mathbb{R}^T$ is a vector of variations corresponding to changes in each step in the control u_k. If we let $\tilde{\mathbf{u}}$ be the optimal choice of the control vector, then we are interested specifically in variations about this specific value, i.e. we want to understand how the system behaves under the perturbation $\tilde{\mathbf{u}}+\varepsilon\boldsymbol{\eta}$ about the optimal control where the corresponding optimal state is $\tilde{\mathbf{x}}$, i.e. $\tilde{\mathbf{x}}$ is the vector of discrete states that are induced by the optimal control $\tilde{\mathbf{u}}$. For $\tilde{\mathbf{u}}$ to truly be optimal then

$$\begin{aligned} 0 &= \frac{\partial J[\tilde{\mathbf{u}}+\varepsilon\boldsymbol{\eta}]}{\partial \varepsilon} \\ &= \sum_{k=0}^{T-1} \left[\frac{\partial L}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k) h_k + \frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k) \right] + \frac{\partial \phi}{\partial x_T}(\tilde{x}_T) h_T, \end{aligned} \tag{23.6}$$

where $h_k = \left.\frac{dx(\tilde{u}_k+\varepsilon\eta_k)}{d\varepsilon}\right|_{\varepsilon=0}$ or in other words, we have supposed (just as we did in the continuous case) that the prescribed perturbation in the control $\tilde{\mathbf{u}}+\varepsilon\boldsymbol{\eta}$ induces the change $\tilde{\mathbf{x}}+\varepsilon\mathbf{h}+O(\varepsilon^2)$ in the state. The state perturbation h_k will satisfy:

$$h_0 = 0, \tag{23.7}$$

$$h_{k+1} = \frac{\partial g}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k) h_k + \frac{\partial g}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k)\eta_k, \quad \text{for } k = 0, 1, \ldots T-1. \tag{23.8}$$

This discrete evolution of the perturbation (often referred to as the *sensitivity*) of the state comes from inserting $\tilde{x}_k + \varepsilon h_k + O(\varepsilon^2)$ and $\tilde{u}_k + \varepsilon \eta_k$ into (23.2) and matching $O(\varepsilon)$ terms.

Now, there is a form of summation by parts that closely imitates integration by parts in the continuous case which we can use here to rewrite $\frac{\partial L}{\partial \varepsilon}$. In addition, we may consider (23.2) as a constraint on the system, introducing a relevant Lagrange multiplier (which is actually the co-state) to do so. We will take a slightly different approach that is mildly more justified, and doesn't require us to muddle the waters between Lagrange multipliers and the co-state.

Hence, we will now select the co-state (of course with proper foresight this choice is obvious, but to the novice reader this should feel completely arbitrary and rather magical in fact) p_k to satisfy

$$\begin{aligned} p_k &= \frac{\partial g}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k)p_{k+1} - \frac{\partial L}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k), \qquad \text{for } k = 0, 1, \dots, T-1, \\ p_T &= -\frac{\partial \phi}{\partial x_T}. \end{aligned}$$

This allows us to rewrite

$$\frac{\partial L}{\partial x_k} = -p_k + \frac{\partial g}{\partial x_k} p_{k+1},$$

which we substitute back into (23.6) to reach

$$\begin{aligned} 0 &= \sum_{k=0}^{T-1} \left[h_k \left(-p_k + \frac{\partial g}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k)p_{k+1} \right) + \frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k)\eta_k \right] + \frac{\partial \phi}{\partial x_T} h_T \\ &= -\sum_{k=0}^{T-1} p_k h_k + \sum_{k=0}^{T-1} \left[\frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k)\eta_k + h_k \frac{\partial g}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k)p_{k+1} \right] + \frac{\partial \phi}{\partial x_T} h_T. \end{aligned}$$

Now, using the fact that $h_0 = 0$, and doing a shift in index on the first sum above, we arrive at

$$\begin{aligned} 0 &= -\sum_{k=0}^{T-2} h_{k+1}p_{k+1} + \sum_{k=0}^{T-1} \left[\frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k)\eta_k + h_k \frac{\partial g}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k)p_{k+1} \right] + \frac{\partial \phi}{\partial x_T} h_T, \\ &= \sum_{k=0}^{T-2} p_{k+1} \left[-h_{k+1} + \frac{\partial g}{\partial x_k}(\tilde{x}_k, \tilde{u}_k, k)h_k \right] + \sum_{k=0}^{T-1} \frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k)\eta_k \\ & p_T \frac{\partial g}{\partial x_{T-1}}(\tilde{x}_{T-1}, \tilde{u}_{T-1}, T-1)h_{T-1} + \frac{\partial \phi}{\partial x_T} h_T. \end{aligned}$$

Using (23.8) this can be rewritten as

$$\begin{aligned} 0 &= -\sum_{k=0}^{T-2} p_{k+1} \frac{\partial g}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k)\eta_k + \sum_{k=0}^{T-1} \frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k)\eta_k + p_T \left[-h_T + \frac{\partial g}{\partial x_{T-1}}(\tilde{x}_{T-1}, \tilde{u}_{T-1}, T-1)h_{T-1} \right] \\ &= \sum_{k=0}^{T-1} \eta_k \left[-p_{k+1} \frac{\partial g}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k) + \frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k) \right]. \end{aligned}$$

This final line needs to be true for any arbitrary variation η_k for all k and hence we find that

$$\frac{\partial L}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k) - p_{k+1} \frac{\partial g}{\partial u_k}(\tilde{x}_k, \tilde{u}_k, k) = 0, \qquad \text{for } k = 0, 1, \dots, T-1. \tag{23.9}$$

It turns out that if we define the discrete Hamiltonian

$$H_k = p_{k+1} g(x_k, u_k, k) - L(x_k, u_k, k), \qquad \text{for } k = 0, 1, \dots, T-1, \tag{23.10}$$

then the necessary conditions for a minimizer of $J[\mathbf{u}]$ are:

$$p_k = \frac{\partial H_k}{\partial x_k}, \quad p_T = -\frac{\partial \phi}{\partial x_T},$$

and $H_k(u_k)$ is maximized as a function of u_k for all k.

Remark 23.1.3. There are a couple of items to note about the differences between these necessary conditions and those for the continuous system. First, the Hamiltonian here is defined as a coupling between the co-state at step $k+1$ and the cost function and evolution of the state at step k. This could have been avoided using a different definition of the Hamiltonian, but the analogy to the continuous setting would have been less clear. Second, the evolution of the co-state is defined differently, i.e. there is a sign change from the continuous case.

One of the primary reasons for the differences between the discrete and continuous versions of Pontryagin's maximum principle is due to the difference between integration by parts and summation by parts (the discrete analogue). Although we have derived the discrete form here without explicitly resorting to summation by parts, we note that is precisely what we did when we shifted the summands, realized that $h_0 = 0$ (the boundary term vanishes), and controlled the term at $k = T-1$ using the endpoint cost which is rewritten in terms of the co-state.

A very similar statement can be made for vector valued, discrete state and control variables. Consider the optimization problem:

$$\min_{\mathbf{u}_k} \sum_{k=0}^{T-1} L(\mathbf{x}_k, \mathbf{u}_k, k) + \phi(\mathbf{x}_T), \tag{23.11}$$

subject to

$$\mathbf{x}_{k+1} = \mathbf{g}(\mathbf{x}_k, \mathbf{u}_k, k), \quad \mathbf{x}_0 \text{ given.} \tag{23.12}$$

Then the optimal control $\tilde{\mathbf{u}}_k$ will satisfy (for each $k = 0, 1, \dots, T-1$):

$$H_k = \mathbf{p}_{k+1} \cdot \mathbf{g}(\mathbf{x}_k, \mathbf{u}_k, k) - L(\mathbf{x}_k \mathbf{u}_k, k), \tag{23.13}$$

$$\mathbf{p}_k = \frac{DH_k}{D\mathbf{x}_k}, \quad \mathbf{p}_T = -\frac{D\phi}{D\mathbf{x}_T}, \tag{23.14}$$

where the Hamiltonian H_k is maximized in $\mathbf{u}_k$ for all k.

23.2 A few discrete examples

Example 23.2.1. Rather than consider a physically motivated example right off the bat, we will instead focus on a simple explicit example. Consider the discrete evolution equation

$$x_{n+1} = x_n + u_n, \quad \text{for } n = 1, 2, \quad \text{and } x_0 = 5$$

coupled with the minimization problem

$$\min_u J[u] = \min_u \frac{1}{2} \sum_{n=0}^{2} [x_n^2 + u_n^2].$$

This problem defines the Hamiltonian

$$H_n = p_{n+1}(x_n + u_n) - \frac{1}{2}[x_n^2 + u_n^2].$$

The maximum principle then tells us that

$$p_n = \frac{\partial H_n}{\partial x_n} = p_{n+1} - x_n, \quad p_3 = 0.$$

In addition we know that the Hamiltonian is maximized in u_n. As $H_n(u_n)$ is a quadratic function, this indicates that the optimal control $\tilde{u}_n$ is given by

$$u_n = p_{n+1}.$$

Putting all of this together, we are looking at the coupled set of discrete evolution equations:

$$\begin{aligned} x_{n+1} &= x_n + p_{n+1}, \quad x_0 = 5, \\ p_n &= p_{n+1} - x_n, \quad p_3 = 0. \end{aligned}$$

As we only have 3 possible states $(n = 0, 1, 2)$ then we can write all the cases out explicitly

$$\begin{aligned} &x_1 = 5 + p_1, \quad x_2 = x_1 + p_2, \quad x_3 = x_2, \\ &p_0 = p_1 - 5, \quad p_1 = p_2 - x_1, \quad p_2 = -x_2. \end{aligned}$$

The solution of this linear system leads to the optimal state, co-state, and control:

$$\begin{aligned} &x_1 = 2, \quad x_2 = 1, \quad x_3 = 1, \\ &p_0 = -8, \quad p_1 = -3, \quad p_2 = -1, \\ &u_0 = -3, \quad u_1 = -1, \quad u_2 = 0. \end{aligned}$$

The previous example was of course unmotivated and rather straightforward, mostly because it involved a scalar state and a scalar control variable. Before we get to anything motivating, we first consider a slightly more complicated example that is unmotivated, but at least multi-dimensional.

Example 23.2.2. Consider the following optimization problem:

$$\min_{u_k} \sum_{k=0}^{2} \left(\frac{1}{2}u_k^2 + x_{1,k}^2\right) - x_{2,3},$$

where $x_{j,k}$ refers to the jth component of the state at the kth step of the evolution. This system is subject to

$$\begin{aligned} x_{1,k+1} &= u_k + x_{1,k}, \quad x_{1,0} = 1, \\ x_{2,k+1} &= x_{1,k} + x_{2,k}, \quad x_{2,0} = 0. \end{aligned}$$

The Hamiltonian in this case is going to be given by

$$H_k = p_{1,k+1}(u_k + x_{1,k}) + p_{2,k+1}(x_{1,k} + x_{2,k}) - \frac{1}{2}u_k^2 - x_{1,k}^2,$$

so that

$$\begin{aligned} p_{1,k} &= p_{1,k+1} + p_{2,k+1} - 2x_{1,k}, \quad p_{1,3} = 0, \\ p_{2,k} &= p_{2,k+1}, \quad p_{2,3} = 1. \end{aligned}$$

In addition, maximizing the Hamiltonian H_k in u_k leads to an optimal control

$$\tilde{u}_k = p_{1,k+1}.$$

Hence the evolution equation for $x_{1,k}$ is actually coupled to the co-state by:

$$x_{1,k+1} = p_{1,k+1} + x_{1,k}, \quad x_{1,0} = 1.$$

Once again, because this is a rather small, simple system, we can afford to write out the entire solution one step at a time (this time it isn't quite as clean meaning interger-valued as the last example).

$$\begin{aligned} &x_{1,0} = 1, \quad x_{2,0} = 0, \quad p_{1,0} = p_{1,1} + p_{2,1} - 2, \quad p_{2,0} = p_{2,1}, \\ &x_{1,1} = p_{1,1} + 1, \quad x_{2,1} = 1, \quad p_{1,1} = p_{1,2} + p_{2,2} - 2x_{1,1}, \quad p_{2,1} = p_{2,2}, \\ &x_{1,2} = p_{1,2} + x_{1,1}, \quad x_{2,2} = x_{1,1} + 1, \quad p_{1,2} = 1 - 2x_{1,2}, \quad p_{2,2} = 1, \\ &x_{1,3} = p_{1,3} + x_{1,2}, \quad x_{2,3} = x_{1,2} + x_{2,2}, \quad p_{1,3} = 0, \quad p_{2,3} = 1. \end{aligned}$$

We first note that $p_{2,k} = 1$ for all k, and because $p_{1,3} = 0$ we can rewrite all of this as:

$$\begin{aligned} &x_{1,0} = 1, \quad x_{2,0} = 0, \quad p_{1,0} = p_{1,1} - 1, \quad p_{2,0} = 1, \\ &x_{1,1} = p_{1,1} + 1, \quad x_{2,1} = 1, \quad p_{1,1} = p_{1,2} + 1 - 2x_{1,1}, \quad p_{2,1} = 1, \\ &x_{1,2} = p_{1,2} + p_{1,1} + 1, \quad x_{2,2} = p_{1,1} + 2, \quad p_{1,2} = 1 - 2x_{1,2}, \quad p_{2,2} = 1, \\ &x_{1,3} = p_{1,2} + p_{1,1} + 1, \quad x_{2,3} = p_{1,2} + 2p_{1,1} + 3, \quad p_{1,3} = 0, \quad p_{2,3} = 1. \end{aligned}$$

Combining terms judiciously (basically choosing equations that give nice results) we see that:

$$\begin{aligned} &p_{1,2} = -2x_{1,2} + 1, \\ &x_{1,2} = x_{1,1} + p_{1,2} = x_{1,1} - 2x_{1,2} + 1, \Rightarrow x_{1,2} = \frac{1}{3}(x_{1,1} + 1), \\ &p_{1,1} = -2x_{1,1} + p_{1,2} + 1 = -2x_{1,1} - 2x1,2 + 2 = -\frac{8}{3}x_{1,1} + \frac{4}{3}, \\ &x_{1,1} = x_{1,0} + p_{1,1} = 1 - \frac{8}{3}x_{1,1} + \frac{4}{3} \Rightarrow x_{1,1} = \frac{7}{11}. \end{aligned}$$

Substituting this back into the difference equations, we find that the optimal state is $x_1 = (1, 7/11, 6/11, 6/11)$ and $x_2 = (0, 1, 18/11, 24, 11)$ with optimal control $u = (-4/11, -1/11, 0)$.

This was a decently tractable problem, but you can see how quickly these kind of situations can get out of hand when trying to find the solution by hand. Just as we saw with solutions of differential equations and continuous optimal control, exact solutions are rare and difficult to come by, not to mention, rarely of physical interest. On the contrary, we can consider some very interesting examples that are physically motivated, but are not nearly as tractable as the previous two examples.

Example 23.2.3. We will consider a discrete SIR model

$$\begin{aligned} S_{k+1} &= S_k(1-u_k) - \beta(S_k(1-u_k))I_k, \\ I_{k+1} &= I_k + \beta S_k(1-u_k)I_k - d_2 I_k, \\ R_{k+1} &= R_k + u_k S_k, \end{aligned}$$

where β is the transmission rate, and d_2 is the death rate due to the infection. The control parameter $0 \leq u_k \leq 1-d$ is the proportion of the population that is vaccinated at each discrete time step. This model will break down if we let $k \to \infty$ as the population percentages will eventually be non-positive. For short times, however this model is valid. With the supposition that we actually have control over the vaccination rate, we will pursue minimizing the following cost functional

$$J[\mathbf{u}] = \sum_{k=1}^{T-1} \left[I_k + Bu_k^2 + B_1 u_k \right] + I_T.$$

If we allow $p_{j,k}$ to denote the co-state at time k where $j = 1$ corresponds to S_k, $j = 2$ to I_k, and $j = 3$ to R_k, the Hamiltonian for this problem is then given by

$$\begin{aligned} H_k = p_{1,k+1}\left[S_k(1-u_k) - \beta(S_k(1-u_k))I_k\right] + p_{2,k+1}\left[I_k + \beta S_k(1-u_k)I_k - d_2 I_k\right] \\ + p_{3,k+1}\left[R_k + u_k S_k\right] - I_k - Bu_k^2 - B_1 u_k. \end{aligned}$$

It follows that the co-state also satisfies:

$$\begin{aligned} p_{1,k} &= p_{1,k+1}\left[1 - u_k - \beta(1-u_k)I_k\right] + \beta p_{2,k+1}(1-u_k)I_k + u_k p_{3,k+1}, \quad p_{1,T} = 0, \\ p_{2,k} &= -\beta p_{1,k+1}S_k(1-u_k) + p_{2,k+1}\left[1 + \beta(1-u_k) - d_2\right] - 1, \quad p_{2,T} = -1, \\ p_{3,k} &= p_{3,k+1}, \quad p_{3,T} = 0. \end{aligned}$$

Similarly because the Hamiltonian is naturally quadratic in the control u_k then we can identify the optimal control as the solution of $\frac{\partial H_k}{\partial u_k} = 0$, i.e.

$$\begin{aligned} \frac{\partial H_k}{\partial u_k} &= -p_{1,k+1}S_k + \beta p_{1,k+1}S_k I_k - \beta p_{2,k+1}S_k I_k + p_{3,k+1}S_k - 2Bu_k - B_1 = 0, \\ &\Rightarrow u_k = \frac{1}{2B}\left[-p_{1,k+1}S_k + \beta p_{1,k+1}S_k I_k - \beta p_{2,k+1}S_k I_k + p_{3,k+1}S_k - B_1\right]. \end{aligned}$$

With the additional constraint that u_k is bounded from above and recognizing that $p_{3,k} = 0$ for all k, in this discrete setting we can simply take the optimal control as

$$u_k = \min\left(1-d, \max\left(0, \frac{1}{2B}\left[-p_{1,k+1}S_k + \beta p_{1,k+1}S_k I_k - \beta p_{2,k+1}S_k I_k - B_1\right]\right)\right).$$

You may have noticed that all of the cost functions introduced in these examples, are quadratic in the control variable. This is of course intentional. Such a choice makes the optimal control computable, and the overall problem tractable. Not only does a quadratic cost make the problem more tractable in the discrete case as well as the continuous case, we have an analogue of LQR in the discrete setting as well.

Example 23.2.4. A very common occurrence is when we run into a linear quadratic regulator in discrete space, i.e.

$$\min_{\mathbf{u}} \frac{1}{2} \sum_{k=1}^{T-1} \left[\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k \right] + \mathbf{x}_T^T M \mathbf{x}_T,$$
$$\mathbf{x}_{k+1} = A_k \mathbf{x}_x + B_k \mathbf{u}_k,$$

where the Q_k and M are symmetric, positive semi-definite and all of the R_k are positive definite. Most of the time (and in everything that follows) each of these matrices do not depend on the time step k, but are constant in k. The solution in this case is given by $\mathbf{u}_k = K_k \mathbf{x}_k$ where the matrix K is given by

$$K_k = -\left(R + B^T P_{k+1} B\right)^{-1} B^T P_{k+1} A,$$

and P_k satisfies the discrete time algebraic Ricatti equation

$$P_{k-1} = Q + A^T P_k A - A^T P_k B (R + B^T P_k B)^{-1} B^T P_k A, \quad P_T = M.$$

Note that the discrete LQR solution here is far more convenient than it was for the continuous case. We can identify the transition matrix P_k iterating backward from the final cost matrix M, identifying the relevant 'gain' matrix K at each step k. There is no need to solve a quadratic matrix differential equation, but we do have to follow the iterative process from the end to the beginning. Derivation of this solution is actually mildly simpler than it was for the continuous case, using the iterative equation of state we can write out

$$\mathbf{x}_k = A\mathbf{x}_{k-1} + B\mathbf{u}_{k-1} = A(A\mathbf{x}_{k-2} + B\mathbf{u}_{k-2}) + B\mathbf{u}_{k-1} = \ldots$$

to eventually obtain a recursive formula for the state $\mathbf{x}_k$ in terms of the initial condition $\mathbf{x}_0$ and relevant control parameters $\mathbf{u}_k$ at each step. This can then be inserted into the cost functional and the optimal controls $\mathbf{u}_k$ can be selected as the solution of a corresponding least-squares problem (this is as messy to write out as it sounds).

23.3 Linear Quadratic Gaussian (LQG) Regulator

We first review the Kalman filter for linear systems. That is we are interested in a discrete time system that is described by

$$\mathbf{x}_k = A\mathbf{x}_{k-1} + B\mathbf{u}_{k-1} + \mathbf{w}_k, \tag{23.15}$$
$$\mathbf{z}_k = H\mathbf{x}_k + \mathbf{v}_k, \tag{23.16}$$

where $\mathbf{w}_k$ and $\mathbf{v}_k$ are Gaussian distributed random variables with mean zero, and covariance given by $Cov(\mathbf{w}_k) = W_k$ and $Cov(\mathbf{v}_k) = V_k$. Technically we could also allow A, B, and H to change with every time step, that is $A = A_k$ etc., but we ignore this possibility for the time being for the sake of simplicity. Recall that the interpretation of this setup is that the system is completely described by $\mathbf{x}_k$, but only the states in $H\mathbf{x}_k$ are observable. The two noise terms $\mathbf{w}_k$ and $\mathbf{v}_k$ arise from imperfect observations or inherent noise in the system.

The Kalman filter applied to this system implies a two-step update to both the expected value of the state $\mathbf{x}_k$, and its covariance matrix S_k (in Volume III this is referred to as P with a gain matrix K but this coincides a little to much with the LQR gain we will also make use of). This is given first as the update step which doesn't rely on the observables:

$$\hat{\mathbf{x}}_{k|k-1} = A\hat{\mathbf{x}}_{k-1} + B\mathbf{u}_k, \tag{23.17}$$

$$S_{k|k-1} = AS_{k-1}A^T + W_k. \tag{23.18}$$

This is followed by the adjustment step which incorporates the observed state $\mathbf{z}_k$ as well as the (known) distribution of the observational uncertainty, first by computing the Kalman Gain L_k:

$$L_k = S_{k|k-1}H^T(HS_{k|k-1}H^T + V_k)^{-1}, \tag{23.19}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + L_k(\mathbf{z}_k - H\hat{\mathbf{x}}_{k|k-1}), \tag{23.20}$$

$$S_k = (I - L_kH)S_{k|k-1}. \tag{23.21}$$

Remark 23.3.1. To add a little bit of side commentary to the definition of the Kalman filter above, we note that $H\hat{\mathbf{x}}_{k|k-1}$ is the predicted value of the observed state whereas $\mathbf{z}_k$ is the actual observed part of the state. Hence the updated state $\hat{\mathbf{x}}_k$ is the predicted state $\hat{\mathbf{x}}_{k|k-1}$ 'nudged' toward the observed value of the state via the Kalman gain matrix L_k. This should feel eerily familiar...

Now with the Kalman filter in mind, we can consider the LQG problem defined by the minimization of

$$\mathbb{E}\left[\sum_{k=0}^{T-1}\left(\mathbf{x}_k^TQ_k\mathbf{x}_k + \mathbf{u}_k^TR_k\mathbf{u}_k\right) + \mathbf{x}_T^TM\mathbf{x}_T\right], \tag{23.22}$$

where all of the Q_k and M are positive semi-definite, and all of the R_k are positive definite. This is coupled with the state space constraint given by (23.15) and (23.16). It turns out that the optimal solution of this problem is to first apply the Kalman filter directly to identify the expected value of the state update, and then to obtain an LQR solution via that estimated state variable. Although we don't prove this result here, it follows from the optimality of both the Kalman filter and LQR in their respective settings.

In summary, to identify the optimal control for this system, we first apply the Kalman filter as outlined above, and then compute the LQR gain via

$$P_T = M, \quad P_{k-1} = A^TP_kA + Q - A^TP_kB(R + B^TP_kB)^{-1}B^TP_kA,$$

and setting the gain matrix

$$K_k = -(R + B^TP_{k+1}B)^{-1}B^TP_{k+1}A.$$

Hence, we combine the Kalman filter and LQR to obtain the optimal control $\mathbf{u}_k = K_k\hat{\mathbf{x}}_k$.

Remark 23.3.2. Note that the LQR gain does NOT depend on H, W, or V. On the other hand the Kalman gain S does NOT depend on Q, R or B.

Essentially, to find the optimal solution for a quadratic cost functional where the state is partially observed via a linear evolution equation with additive Gaussian noise in both the state equation and observations, we apply the Kalman filter to first obtain an update on the state estimation, and then perform LQR on the estimated state.

Example 23.3.3. Now, in danger of creating an example for which there is no solution, consider the evolution equation

$$\begin{aligned} x_{k+1} &= x_k + y_k \\ y_{k+1} &= y_k + u_k, \end{aligned}$$

which is given initial conditions $(x_0, y_0) = (1, 0) + \eta$ where $\boldsymbol{\eta}$ is a two-dimensional standard normal random variable. Letting $\mathbf{x}_k = (x_k, y_k)^T$ this is coupled with the observable state

$$\mathbf{z}_k = H\mathbf{x}_k + \eta_h,$$

where $H = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, and once again we allow η_h to be a standard normal random variable representing the noise of the system.

If we desire to minimize the overall cost:

$$\mathbb{E}\left[\sum_{k=0}^{19} (x_k^2 + y_k^2 + \rho u_k^2) + x_{20}^2 + y_{20}^2\right],$$

then according to the derivation given above (note that $Q = M$ is the 2×2 identity matrix), the optimal solution will satisfy:

$$u_k = K_k \hat{x}_k,$$

as outlined above, where all of the relevant matrices are given by:

$$\begin{aligned} &A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad H = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad Q = M = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\ &V_k = W_k = 0.1 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad R = \rho. \end{aligned}$$

A very rough numerical solution is presented in Algorithm 23.1.

23.3.1 Kalman-Bucy filter

The Kalman filter has an analog in continuous variables. This is most commonly referred to as the Kalman-Bucy filter. The linear quadratic regulator on this type of setting also leads to an optimization that requires the application of the Kalman-Bucy filter first to estimate the state, and then an LQR step which yields the optimal control. The primary difficulty with working with this setting is that it requires an accurate understanding of stochastic differential equations, i.e. we would need to spend several weeks investigating how the Kalman-Bucy filter works before we could delve into LQG in this setting. Unfortunately this is relegated to future investigations...

Exercises

Note to the student: Each section of this chapter has several corresponding exercises, all collected here at the end of the chapter. The exercises between the first and second line are for Section 1, the exercises between the second and third lines are for Section 2, and so forth.

```python
import numpy as np
from matplotlib import pyplot as plt

def my_LQG(A,B,H,Q,M,V,W,R,x0,T):

    S = [None]*(T+1)
    P = [None]*(T+1)

    #setup the LQR gain here
    P[T] = M
    for k in range(T, -1, -1):
        var = R+np.dot(np.dot(B.T,P[k]),B)
        P[k-1] = np.dot(np.dot(A.T,P[k]),A) + Q - np.dot(np.dot(np.dot(np.dot(np.dot(np.dot(A.T,P[k]),B),np.linalg.pinv(var)),B.T),P[k]),A)

    u = [None]*T
    K = [None]*(T+1)
    L = [None]*T
    S = [None]*(T+1)
    S[0] = np.eye(2)

    #setup the Kalman gain
    for k in range(1,T):
        S[k] = A@(S[k-1]-S[k-1]@H.T@np.linalg.pinv(H@S[k-1]@H.T+V)@H@S[k-1])@A.T+W
    #now compute both gain matrices at each step
    #note that np.squeeze is only used because this term is actually a scalar...
    K[0] = -B.T@P[1]@A/(np.squeeze(B.T@P[1]@B)+R)
    for k in range(1,T):
        K[k] = -B.T@P[k+1]@A/(np.squeeze(B.T@P[k+1]@B)+R)
        L[k] = S[k-1]@H.T@np.linalg.pinv(H@S[k-1]@H.T+V)

    full_state = [None]*(T+1)
    full_state[0] = np.array([x0]).T
    observed_state = [None]*(T+1)

    #finally we are ready to compute the solution.
    u[0] = K[0]@full_state[0]

    for k in range(1,T):

        full_state[k] = A@full_state[k-1]+B@u[k-1] + np.random.normal(0,.1,size=(1,2))
        observed_state[k] = H@full_state[k] + np.random.normal(0,.1,size=(1,2))
        u[k] = -K[k]@observed_state[k]
    return u, observed_state, full_state
```

Algorithm 23.1: Algorithm for computing the solution of Example 23.3.3. Note that there are much more readable and simple approaches to coding this particular algorithm. This is by no means the best way to implement LQG...just the first one

You should **work every exercise** (your instructor may choose to let you skip some of the advanced exercises marked with *). We have carefully selected them, and each is important for your ability to understand subsequent material. Many of the examples and results proved in the exercises are used again later in the text. Exercises marked with ⚠ are especially important and are likely to be used later in this book and beyond. Those marked with † are harder than average, but should still be done.

Although they are gathered together at the end of the chapter, we strongly recommend you do the exercises for each section as soon as you have completed the section, rather than saving them until you have finished the entire chapter.

23.1. Using a forward Euler time-stepping discretization with time step Δt, find the discrete space state equation for a linear regulator, i.e.

$$\mathbf{x}' = A\mathbf{x} + B\mathbf{u}.$$

23.2. Derive (23.8) by looking at the discrete update equation $x_{k+1} = g(x_k, u_k, k)$ and inserting $x_k \to x_k + \varepsilon h_k + O(\varepsilon^2)$ and $u_k \to u_k + \varepsilon \eta_k$.

23.3. Solve the optimal control problem from Example 23.2.1 except now use initial condition $x_0 = 3$ and have the evolution equation be given by $x_{n+1} = 2x_n + u_n$ (you may need to use Python to help with the final linear algebra).

23.4. Calculate the optimal solution for the SIR example when $B_1 = 0,\ S_1 = 100,\ I_1 = 5,\ \beta = 0.002,\ d_2 = 0.1,\ d = 0.2,\ N = 4$ for each of $B = 1,\ 10$ and $B = 20$. You should generate these values numerically, and plot the evolution (up to time $T = 4$) for all three components of the total population. Does the model remain valid (do you get negative populations)?

23.5. Based on what you found for the first problem, is the forward Euler discretization of continuous LQR, also a discrete LQR problem? If so, what are the corresponding matrices in terms of the time-step Δt?

23.6. Adjust the code in Algorithm 23.1 so that it accounts for different specified covariance matrices for the noise.

23.7. Explore the numerical solutions of Example 23.3.3 when the noise on the observation, and evolution equation are not given by standard normal distributions, that is modify the code in Algorithm 23.1 to accommodate settings where V and W are not the identity (you can work with constant multiples of the identity).

23.8. *Modify the problem you studied for the Kalman Filter lab from Volume III so that you can implement LQG for the same type of problem, i.e. an object traveling in a 2D plane whose position is observed, but velocity is not. For this setup, you will ignore gravity (the object is in space I guess), but you want it to start with initial position $(x_0, y_0) = (-20, 30)$, with initial velocity $(v_x, v_y) = (-5, 45)$. The control variable for the problem is 2D corresponding to acceleration in x and y, i.e. $v_x' = u_x$ and $v_y' = u_y$ where (u_x, u_y) is the control vector.
Set up a cost function that will get your object close to the origin $(x, y) = (0, 0)$ with ending velocity near zero as well, but also does so with a reasonable amount of acceleration, by $T = 100$ time steps.

23.9. *Modify the code from the Kalman Filter Lab to implement a solution of the previous problem. Use the same values for the covariance matrices as you did in the lab.