Lecture 1

Linear quadratic regulator: Discrete-time finite horizon

- LQR cost function
- multi-objective interpretation
- LQR via least-squares
- dynamic programming solution
- steady-state LQR control
- extensions: time-varying systems, tracking problems

LQR problem: background

discrete-time system $x_{t+1} = Ax_t + Bu_t$, $x_0 = x^{\mathrm{init}}$

problem: choose $u_0, u_1, \ldots$ so that

- $x_0, x_1, \ldots$ is 'small', *i.e.*, we get good *regulation* or *control*
- $u_0, u_1, \ldots$ is 'small', *i.e.*, using small *input effort* or *actuator authority*

- we'll define 'small' soon
- these are usually competing objectives, *e.g.*, a large u can drive x to zero fast

linear quadratic regulator (LQR) theory addresses this question

LQR cost function

we define *quadratic cost function*

$$J(U) = \sum_{\tau=0}^{N-1} \left(x_\tau^T Q x_\tau + u_\tau^T R u_\tau \right) + x_N^T Q_f x_N$$

where $U = (u_0, \ldots, u_{N-1})$ and

$$Q = Q^T \geq 0, \qquad Q_f = Q_f^T \geq 0, \qquad R = R^T > 0$$

are given *state cost*, *final state cost*, and *input cost* matrices

- N is called *time horizon* (we'll consider $N = \infty$ later)
- first term measures *state deviation*
- second term measures *input size* or *actuator authority*
- last term measures *final state deviation*
- Q, R set relative weights of state deviation and input usage
- $R > 0$ means any (nonzero) input adds to cost J

LQR problem: find $u_0^{\mathrm{lqr}}, \ldots, u_{N-1}^{\mathrm{lqr}}$ that minimizes $J(U)$

Comparison to least-norm input

c.f. least-norm input that steers x to $x_N = 0$:

- no cost attached to $x_0, \ldots, x_{N-1}$
- x_N must be exactly zero

we can approximate the least-norm input by taking

$$R = I, \qquad Q = 0, \qquad Q_f \text{ large, } e.g., \; Q_f = 10^8 I$$

Multi-objective interpretation

common form for Q and R:

$$R = \rho I, \qquad Q = Q_f = C^T C$$

where $C \in \mathbf{R}^{p \times n}$ and $\rho \in \mathbf{R}$, $\rho > 0$

cost is then

$$J(U) = \sum_{\tau=0}^{N} \|y_\tau\|^2 + \rho \sum_{\tau=0}^{N-1} \|u_\tau\|^2$$

where $y = Cx$

here $\sqrt{\rho}$ gives relative weighting of output norm and input norm

Input and output objectives

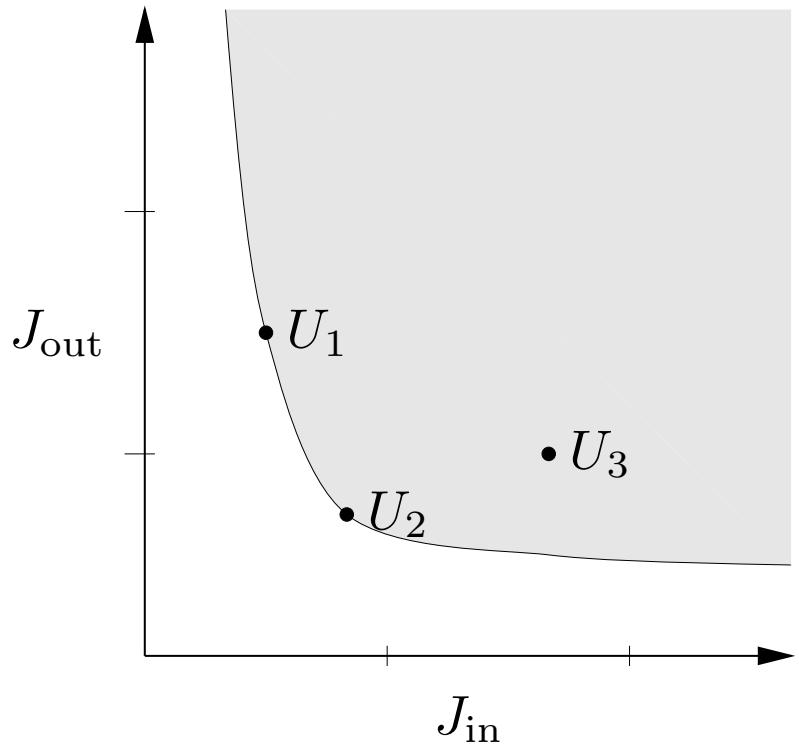
fix $x_0 = x^{\mathrm{init}}$ and horizon N; for any input $U = (u_0, \ldots, u_{N-1})$ define

- input cost $J_{\mathrm{in}}(U) = \sum_{\tau=0}^{N-1} \|u_\tau\|^2$
- output cost $J_{\mathrm{out}}(U) = \sum_{\tau=0}^{N} \|y_\tau\|^2$

these are (competing) objectives; we want *both* small

LQR quadratic cost is $J_{\mathrm{out}} + \rho J_{\mathrm{in}}$

plot $(J_{\mathrm{in}}, J_{\mathrm{out}})$ for all possible U:



- shaded area shows $(J_{\mathrm{in}}, J_{\mathrm{out}})$ achieved by some U
- clear area shows $(J_{\mathrm{in}}, J_{\mathrm{out}})$ not achieved by any U

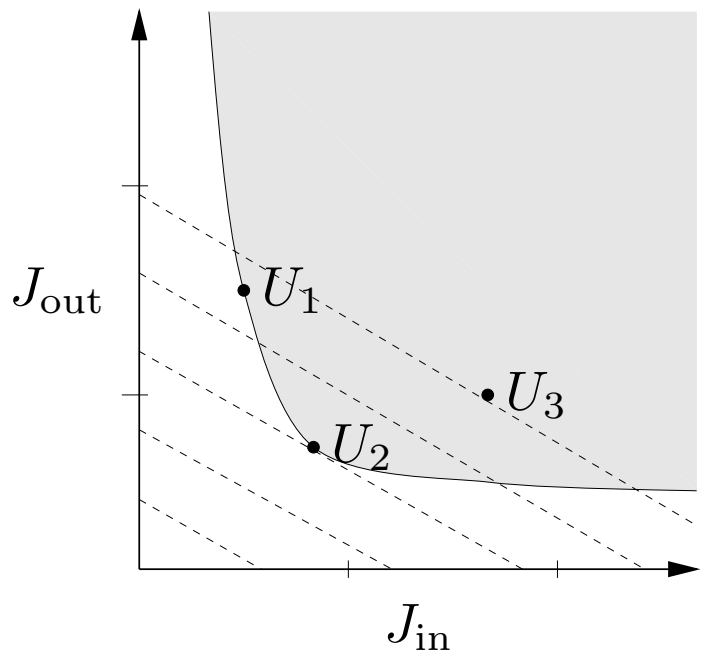
three sample inputs U_1, U_2, and U_3 are shown

- U_3 is worse than U_2 on both counts ($J_{\rm in}$ and $J_{\rm out}$)
- U_1 is better than U_2 in $J_{\rm in}$, but worse in $J_{\rm out}$

interpretation of LQR quadratic cost:

$$J = J_{\rm out} + \rho J_{\rm in} = \text{ constant}$$

corresponds to a line with slope $-\rho$ on $(J_{\rm in}, J_{\rm out})$ plot



- LQR optimal input is at boundary of shaded region, just touching line of smallest possible J
- u_2 is LQR optimal for ρ shown
- by varying ρ from 0 to $+\infty$, can sweep out *optimal tradeoff curve*

LQR via least-squares

LQR can be formulated (and solved) as a least-squares problem

$X = (x_0, \ldots x_N)$ is a *linear function* of x_0 and $U = (u_0, \ldots, u_{N-1})$:

$$\begin{bmatrix} x_0 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} 0 & \cdots & & \\ B & 0 & \cdots & \\ AB & B & 0 & \cdots \\ \vdots & \vdots & & \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix} \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix} + \begin{bmatrix} I \\ A \\ \vdots \\ A^N \end{bmatrix} x_0$$

express as $X = GU + Hx_0$, where $G \in \mathbf{R}^{Nn \times Nm}$, $H \in \mathbf{R}^{Nn \times n}$

express LQR cost as

$$\begin{aligned} J(U) &= \left\| \mathbf{diag}(Q^{1/2}, \ldots, Q^{1/2}, Q_f^{1/2})(GU + Hx_0) \right\|^2 \\ &+ \left\| \mathbf{diag}(R^{1/2}, \ldots, R^{1/2})U \right\|^2 \end{aligned}$$

this is just a (big) least-squares problem

this solution method requires forming and solving a least-squares problem with size $N(n+m) \times Nm$

using a naive method ($e.g.$, QR factorization), cost is $O(N^3nm^2)$

Dynamic programming solution

- gives an efficient, recursive method to solve LQR least-squares problem; cost is $O(Nn^3)$
- (but in fact, a less naive approach to solve the LQR least-squares problem will have the same complexity)
- useful and important idea on its own
- same ideas can be used for many other problems

Value function

for $t = 0, \ldots, N$ define the **value function** $V_t : \mathbf{R}^n \to \mathbf{R}$ by

$$V_t(z) = \min_{u_t,\ldots,u_{N-1}} \sum_{\tau=t}^{N-1} \left(x_\tau^T Q x_\tau + u_\tau^T R u_\tau\right) + x_N^T Q_f x_N$$

subject to $x_t = z$, $x_{\tau+1} = Ax_\tau + Bu_\tau$, $\tau = t, \ldots, T$

- $V_t(z)$ gives the minimum LQR cost-to-go, starting from state z at time t
- $V_0(x_0)$ is min LQR cost (from state x_0 at time 0)

we will find that

- V_t is quadratic, *i.e.*, $V_t(z) = z^T P_t z$, where $P_t = P_t^T \geq 0$
- P_t can be found recursively, working backward from $t = N$
- the LQR optimal u is easily expressed in terms of P_t

cost-to-go with no time left is just final state cost:

$$V_N(z) = z^T Q_f z$$

thus we have $P_N = Q_f$

Dynamic programming principle

- now suppose we know $V_{t+1}(z)$
- what is the optimal choice for u_t?
- choice of u_t affects
  - current cost incurred (through $u_t^T R u_t$)
  - where we land, x_{t+1} (hence, the min-cost-to-go from x_{t+1})
- **dynamic programming (DP) principle:**

$$V_t(z) = \min_w \left(z^T Q z + w^T R w + V_{t+1}(Az + Bw) \right)$$

  - $z^T Q z + w^T R w$ is cost incurred at time t if $u_t = w$
  - $V_{t+1}(Az + Bw)$ is min cost-to-go from where you land at $t + 1$

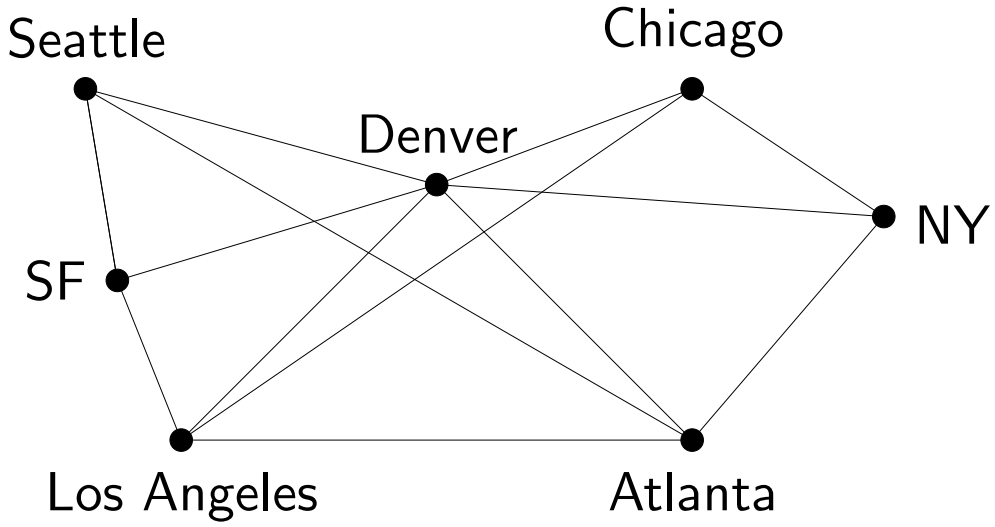
- follows from fact that we can minimize in any order:

$$\min_{w_1,\ldots,w_k} f(w_1,\ldots,w_k) = \min_{w_1} \underbrace{\left(\min_{w_2,\ldots,w_k} f(w_1,\ldots,w_k) \right)}_{\text{a fct of } w_1}$$

in words:
min cost-to-go from where you are $=$ min over
(current cost incurred $+$ min cost-to-go from where you land)

Example: path optimization

- edges show possible flights; each has some cost
- want to find min cost route or path from SF to NY



dynamic programming (DP):

- $V(i)$ is min cost from airport i to NY, over all possible paths
- to find min cost from city i to NY: minimize sum of flight cost plus min cost to NY from where you land, over all flights out of city i
  (gives optimal flight out of city i on way to NY)
- if we can find $V(i)$ for each i, we can find min cost path from any city to NY
- DP principle: $V(i) = \min_j(c_{ji} + V(j))$, where c_{ji} is cost of flight from i to j, and minimum is over all possible flights out of i

HJ equation for LQR

$$V_t(z) = z^T Q z + \min_w \left(w^T R w + V_{t+1}(Az + Bw) \right)$$

- called DP, Bellman, or Hamilton-Jacobi equation
- gives V_t recursively, in terms of V_{t+1}
- any minimizing w gives optimal u_t:

$$u_t^{\mathrm{lqr}} = \operatorname*{argmin}_w \left(w^T R w + V_{t+1}(Az + Bw) \right)$$

- let's assume that $V_{t+1}(z) = z^T P_{t+1} z$, with $P_{t+1} = P_{t+1}^T \geq 0$
- we'll show that V_t has the same form
- by DP,

$$V_t(z) = z^T Q z + \min_w \left(w^T R w + (Az + Bw)^T P_{t+1} (Az + Bw) \right)$$

- can solve by setting derivative w.r.t. w to zero:

$$2w^T R + 2(Az + Bw)^T P_{t+1} B = 0$$

- hence optimal input is

$$w^* = -(R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A z$$

- and so (after some ugly algebra)

$$\begin{aligned} V_t(z) &= z^T Q z + w^{*T} R w^* + (Az + Bw^*)^T P_{t+1}(Az + Bw^*) \\ &= z^T \left(Q + A^T P_{t+1} A - A^T P_{t+1} B (R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A \right) z \\ &= z^T P_t z \end{aligned}$$

  where

$$P_t = Q + A^T P_{t+1} A - A^T P_{t+1} B (R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A$$

- easy to show $P_t = P_t^T \geq 0$

Summary of LQR solution via DP

1. set $P_N := Q_f$

2. for $t = N, \ldots, 1$,

$$P_{t-1} := Q + A^T P_t A - A^T P_t B (R + B^T P_t B)^{-1} B^T P_t A$$

3. for $t = 0, \ldots, N-1$, define $K_t := -(R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A$

4. for $t = 0, \ldots, N-1$, optimal u is given by $u_t^{\mathrm{lqr}} = K_t x_t$

- optimal u is a linear function of the state (called *linear state feedback*)
- recursion for min cost-to-go runs backward in time

LQR example

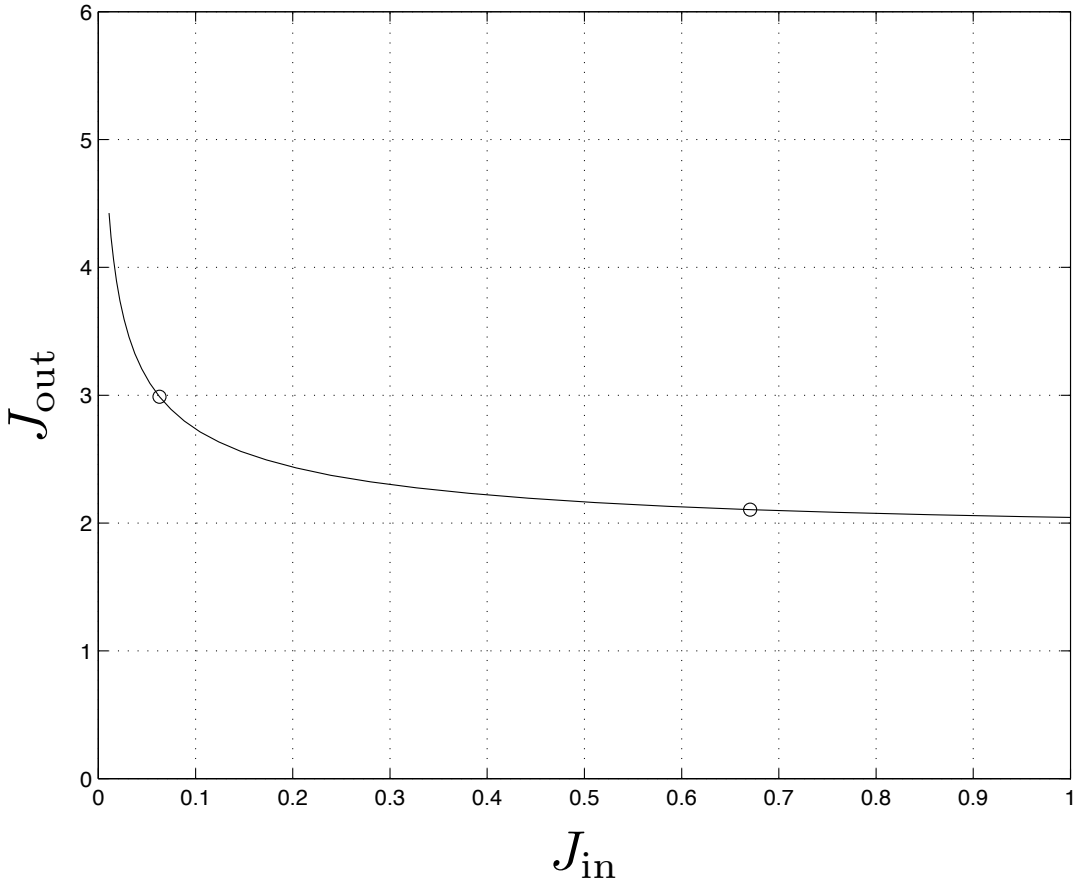
2-state, single-input, single-output system

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

with initial state $x_0 = (1, 0)$, horizon $N = 20$, and weight matrices

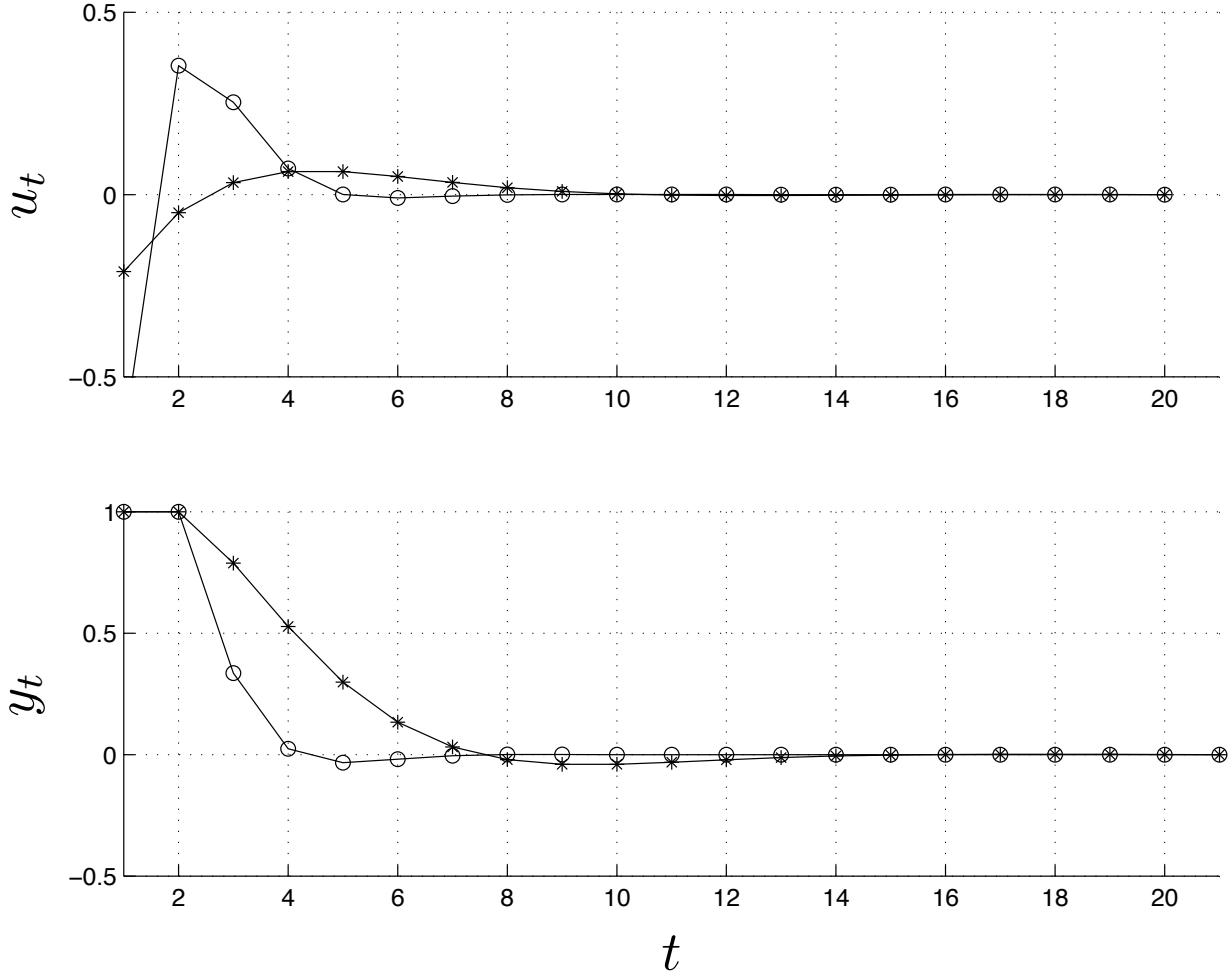
$$Q = Q_f = C^T C, \qquad R = \rho I$$

optimal trade-off curve of $J_{\rm in}$ vs. $J_{\rm out}$:



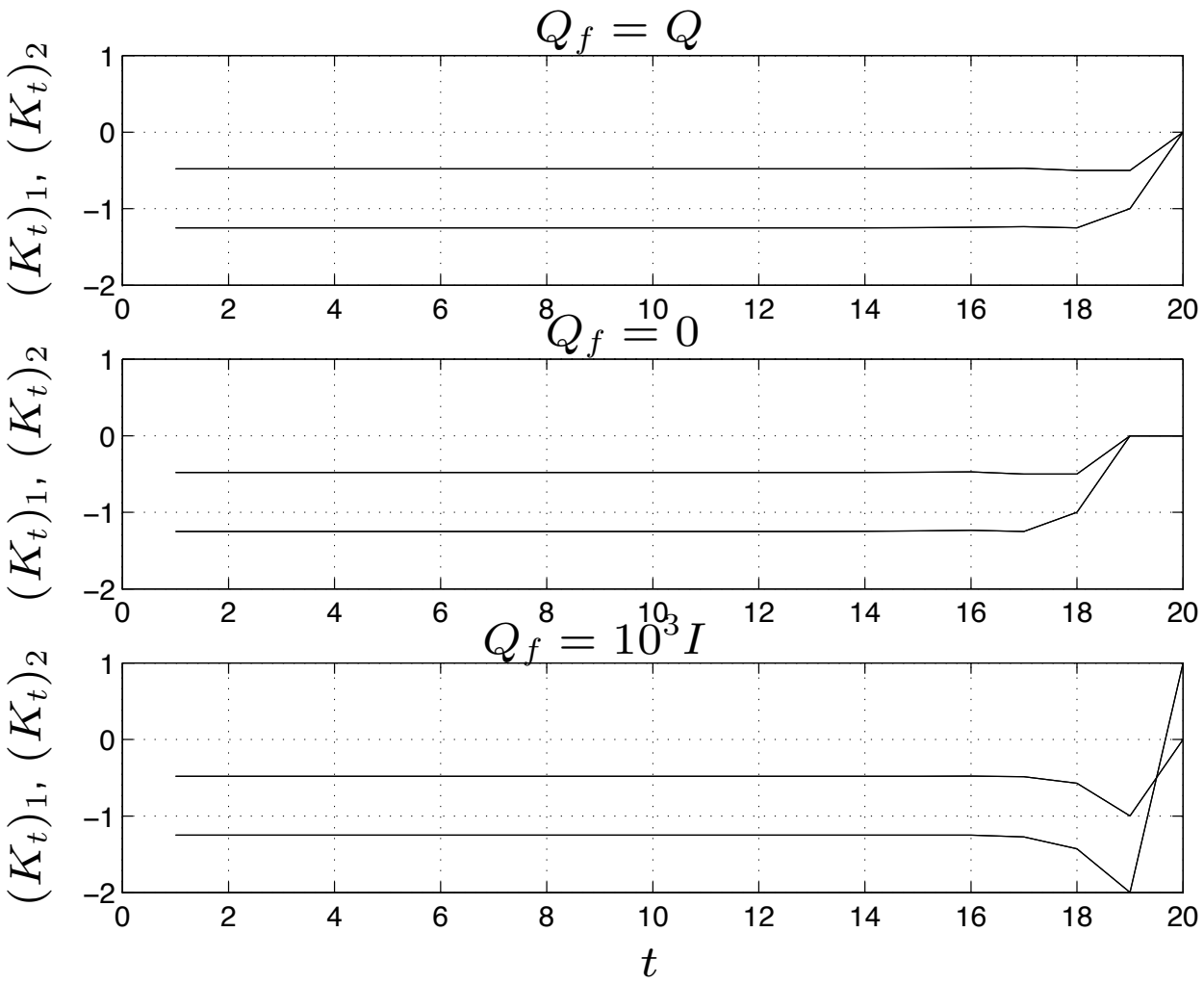
circles show LQR solutions with $\rho = 0.3$, $\rho = 10$

u & y for $\rho = 0.3$, $\rho = 10$:



optimal input has form $u_t = K_t x_t$, where $K_t \in \mathbf{R}^{1\times 2}$

state feedback gains vs. t for various values of Q_f (note convergence):



Steady-state regulator

usually P_t rapidly converges as t decreases below N

limit or steady-state value $P_{\rm ss}$ satisfies

$$P_{\rm ss} = Q + A^T P_{\rm ss} A - A^T P_{\rm ss} B (R + B^T P_{\rm ss} B)^{-1} B^T P_{\rm ss} A$$

which is called the (DT) algebraic Riccati equation (ARE)

- $P_{\rm ss}$ can be found by iterating the Riccati recursion, or by direct methods
- for t not close to horizon N, LQR optimal input is approximately a linear, constant state feedback

$$u_t = K_{\rm ss} x_t, \qquad K_{\rm ss} = -(R + B^T P_{\rm ss} B)^{-1} B^T P_{\rm ss} A$$

(very widely used in practice; more on this later)

Time-varying systems

LQR is readily extended to handle time-varying systems

$$x_{t+1} = A_t x_t + B_t u_t$$

and time-varying cost matrices

$$J = \sum_{\tau=0}^{N-1} \left(x_\tau^T Q_\tau x_\tau + u_\tau^T R_\tau u_\tau \right) + x_N^T Q_f x_N$$

(so Q_f is really just Q_N)

DP solution is readily extended, but (of course) there need not be a steady-state solution

Tracking problems

we consider LQR cost with state and input offsets:

$$\begin{aligned} J &= \sum_{\tau=0}^{N-1} (x_\tau - \bar{x}_\tau)^T Q (x_\tau - \bar{x}_\tau) \\ &+ \sum_{\tau=0}^{N-1} (u_\tau - \bar{u}_\tau)^T R (u_\tau - \bar{u}_\tau) \end{aligned}$$

(we drop the final state term for simplicity)

here, $\bar{x}_\tau$ and $\bar{u}_\tau$ are given desired state and input trajectories

DP solution is readily extended, even to time-varying tracking problems

Gauss-Newton LQR

nonlinear dynamical system: $x_{t+1} = f(x_t, u_t)$, $x_0 = x^{\mathrm{init}}$

objective is

$$J(U) = \sum_{\tau=0}^{N-1} \left(x_\tau^T Q x_\tau + u_\tau^T R u_\tau \right) + x_N^T Q_f x_N$$

where $Q = Q^T \geq 0$, $Q_f = Q_f^T \geq 0$, $R = R^T > 0$

start with a guess for U, and alternate between:

- linearize around current trajectory
- solve associated LQR (tracking) problem

sometimes converges, sometimes to the globally optimal U

some more detail:

- let u denote current iterate or guess
- simulate system to find x, using $x_{t+1} = f(x_t, u_t)$
- linearize around this trajectory: $\delta x_{t+1} = A_t \delta x_t + B_t \delta u_t$

$$A_t = D_x f(x_t, u_t) \qquad B_t = D_u f(x_t, u_t)$$

- solve time-varying LQR tracking problem with cost

$$\begin{aligned} J &= \sum_{\tau=0}^{N-1} (x_\tau + \delta x_\tau)^T Q (x_\tau + \delta x_\tau) \\ &+ \sum_{\tau=0}^{N-1} (u_\tau + \delta u_\tau)^T R (u_\tau + \delta u_\tau) \end{aligned}$$

- for next iteration, set $u_t := u_t + \delta u_t$

Lecture 2
LQR via Lagrange multipliers

- useful matrix identities
- linearly constrained optimization
- LQR via constrained optimization

Some useful matrix identities

let's start with a simple one:

$$Z(I+Z)^{-1} = I - (I+Z)^{-1}$$

(provided $I+Z$ is invertible)

to verify this identity, we start with

$$I = (I+Z)(I+Z)^{-1} = (I+Z)^{-1} + Z(I+Z)^{-1}$$

re-arrange terms to get identity

an identity that's a bit more complicated:

$$(I + XY)^{-1} = I - X(I + YX)^{-1}Y$$

(if either inverse exists, then the other does; in fact $\det(I + XY) = \det(I + YX)$)

to verify:

$$\begin{aligned}
\left(I - X(I + YX)^{-1}Y\right)(I + XY) &= I + XY - X(I + YX)^{-1}Y(I + XY) \\
&= I + XY - X(I + YX)^{-1}(I + YX)Y \\
&= I + XY - XY = I
\end{aligned}$$

another identity:

$$Y(I + XY)^{-1} = (I + YX)^{-1}Y$$

to verify this one, start with $Y(I + XY) = (I + YX)Y$

then multiply on left by $(I + YX)^{-1}$, on right by $(I + XY)^{-1}$

- note dimensions of inverses not necessarily the same
- mnemonic: lefthand Y moves into inverse, pushes righthand Y out . . .

and one more:

$$(I + XZ^{-1}Y)^{-1} = I - X(Z + YX)^{-1}Y$$

let's check:

$$\begin{aligned}
\left(I + X(Z^{-1}Y)\right)^{-1} &= I - X\left(I + Z^{-1}YX\right)^{-1} Z^{-1}Y \\
&= I - X(Z(I + Z^{-1}YX))^{-1}Y \\
&= I - X(Z + YX)^{-1}Y
\end{aligned}$$

Example: rank one update

- suppose we've already calculated or know A^{-1}, where $A \in \mathbf{R}^{n \times n}$
- we need to calculate $(A + bc^T)^{-1}$, where $b,\ c \in \mathbf{R}^n$
  ($A + bc^T$ is called a *rank one update* of A)

we'll use another identity, called *matrix inversion lemma*:

$$(A + bc^T)^{-1} = A^{-1} - \frac{1}{1 + c^T A^{-1} b}(A^{-1}b)(c^T A^{-1})$$

note that RHS is easy to calculate since we know A^{-1}

more general form of matrix inversion lemma:

$$(A + BC)^{-1} = A^{-1} - A^{-1}B\left(I + CA^{-1}B\right)^{-1}CA^{-1}$$

let's verify it:

$$\begin{aligned}
(A + BC)^{-1} &= \left(A(I + A^{-1}BC)\right)^{-1} \\
&= (I + (A^{-1}B)C)^{-1}A^{-1} \\
&= \left(I - (A^{-1}B)(I + C(A^{-1}B))^{-1}C\right)A^{-1} \\
&= A^{-1} - A^{-1}B(I + CA^{-1}B)^{-1}CA^{-1}
\end{aligned}$$

Another formula for the Riccati recursion

$$
\begin{aligned}
P_{t-1} &= Q + A^T P_t A - A^T P_t B (R + B^T P_t B)^{-1} B^T P_t A \\
&= Q + A^T P_t \left(I - B (R + B^T P_t B)^{-1} B^T P_t \right) A \\
&= Q + A^T P_t \left(I - B ((I + B^T P_t B R^{-1}) R)^{-1} B^T P_t \right) A \\
&= Q + A^T P_t \left(I - B R^{-1} (I + B^T P_t B R^{-1})^{-1} B^T P_t \right) A \\
&= Q + A^T P_t \left(I + B R^{-1} B^T P_t \right)^{-1} A \\
&= Q + A^T \left(I + P_t B R^{-1} B^T \right)^{-1} P_t A
\end{aligned}
$$

or, in pretty, symmetric form:

$$
P_{t-1} = Q + A^T P_t^{1/2} \left(I + P_t^{1/2} B R^{-1} B^T P_t^{1/2} \right)^{-1} P_t^{1/2} A
$$

Linearly constrained optimization

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Fx = g \end{array}$$

- $f : \mathbf{R}^n \to \mathbf{R}$ is smooth *objective function*
- $F \in \mathbf{R}^{m \times n}$ is fat

form *Lagrangian* $L(x, \lambda) = f(x) + \lambda^T(g - Fx)$ (λ is *Lagrange multiplier*)

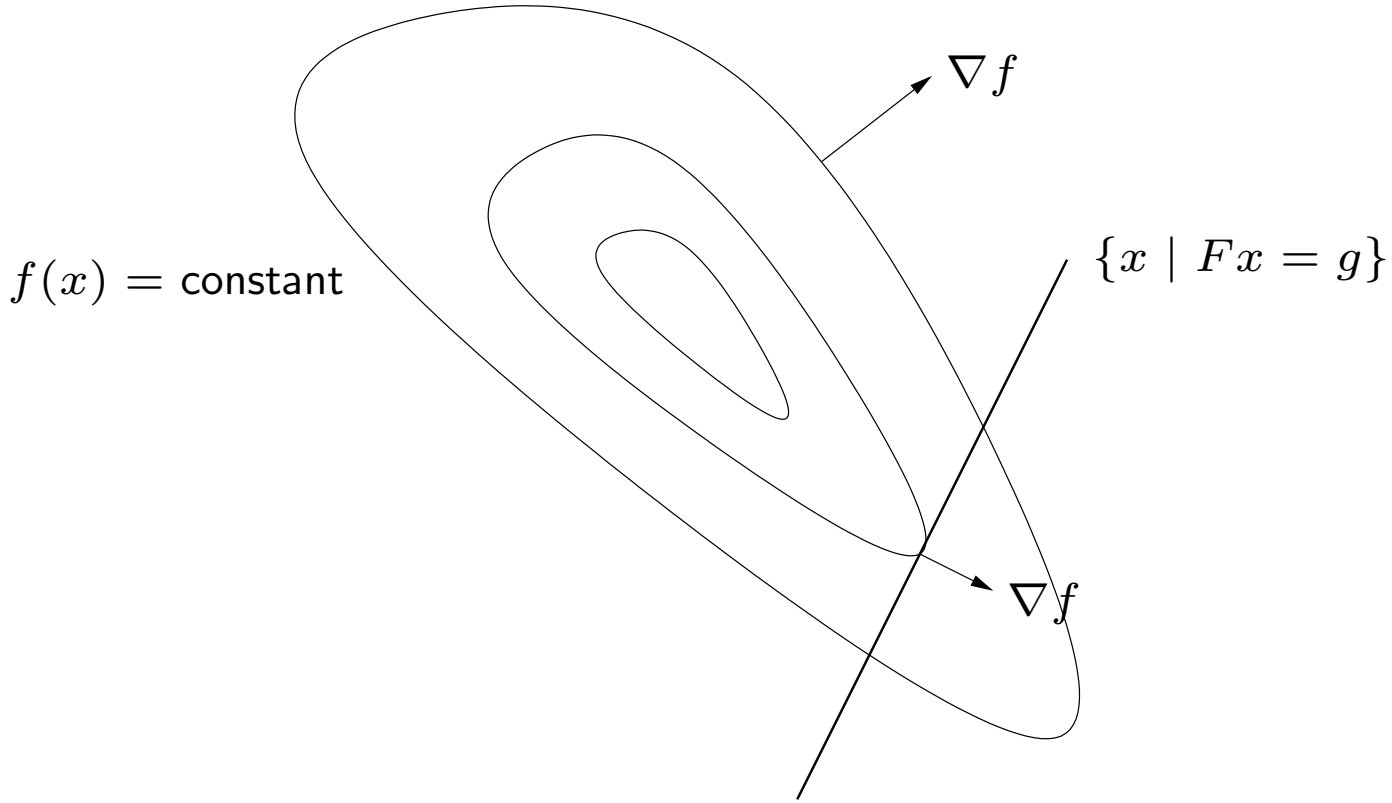
if x is optimal, then

$$\nabla_x L = \nabla f(x) - F^T\lambda = 0, \qquad \nabla_\lambda L = g - Fx = 0$$

i.e., $\nabla f(x) = F^T\lambda$ for some $\lambda \in \mathbf{R}^m$

(generalizes optimality condition $\nabla f(x) = 0$ for unconstrained minimization problem)

Picture



$\nabla f(x) = F^T\lambda$ for some $\lambda \iff \nabla f(x) \in \mathcal{R}(F^T) \iff \nabla f(x) \perp \mathcal{N}(F)$

Feasible descent direction

suppose x is current, feasible point ($i.e.$, $Fx = g$)

consider a small step in direction v, to $x + hv$ (h small, positive)

when is $x + hv$ better than x?

need $x + hv$ feasible: $F(x + hv) = g + hFv = g$, so $Fv = 0$

$v \in \mathcal{N}(F)$ is called a *feasible direction*

we need $x + hv$ to have smaller objective than x:

$$f(x + hv) \approx f(x) + h\nabla f(x)^T v < f(x)$$

so we need $\nabla f(x)^T v < 0$ (called a *descent direction*)

(if $\nabla f(x)^T v > 0$, $-v$ is a descent direction, so we need only $\nabla f(x)^T v \neq 0$)

x is not optimal if there exists a feasible descent direction

if x is optimal, every feasible direction satisfies $\nabla f(x)^T v = 0$

$$
\begin{aligned}
Fv = 0 \;\Rightarrow\; \nabla f(x)^T v = 0 \quad &\Longleftrightarrow \quad \mathcal{N}(F) \subseteq \mathcal{N}(\nabla f(x)^T) \\
&\Longleftrightarrow \quad \mathcal{R}(F^T) \supseteq \mathcal{R}(\nabla f(x)) \\
&\Longleftrightarrow \quad \nabla f(x) \in \mathcal{R}(F^T) \\
&\Longleftrightarrow \quad \nabla f(x) = F^T \lambda \;\text{ for some }\; \lambda \in \mathbf{R}^m \\
&\Longleftrightarrow \quad \nabla f(x) \perp \mathcal{N}(F)
\end{aligned}
$$

LQR as constrained minimization problem

$$\begin{array}{ll} \text{minimize} & J = \frac{1}{2}\sum_{t=0}^{N-1}\left(x_t^T Q x_t + u_t^T R u_t\right) + \frac{1}{2}x_N^T Q_f x_N \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t, \quad t = 0, \ldots, N-1 \end{array}$$

- variables are $u_0, \ldots, u_{N-1}$ and $x_1, \ldots, x_N$
  ($x_0 = x^{\text{init}}$ is given)

- objective is (convex) quadratic
  (factor $1/2$ in objective is for convenience)

introduce Lagrange multipliers $\lambda_1, \ldots, \lambda_N \in \mathbf{R}^n$ and form Lagrangian

$$L = J + \sum_{t=0}^{N-1} \lambda_{t+1}^T \left(Ax_t + Bu_t - x_{t+1}\right)$$

Optimality conditions

we have $x_{t+1} = Ax_t + Bu_t$ for $t = 0, \ldots, N-1$, $x_0 = x^{\text{init}}$

for $t = 0, \ldots, N-1$, $\nabla_{u_t} L = Ru_t + B^T \lambda_{t+1} = 0$

hence, $u_t = -R^{-1} B^T \lambda_{t+1}$

for $t = 1, \ldots, N-1$, $\nabla_{x_t} L = Qx_t + A^T \lambda_{t+1} - \lambda_t = 0$

hence, $\lambda_t = A^T \lambda_{t+1} + Qx_t$

$\nabla_{x_N} L = Q_f x_N - \lambda_N = 0$, so $\lambda_N = Q_f x_N$

these are a set of linear equations in the variables

$$u_0, \ldots, u_{N-1}, \quad x_1, \ldots, x_N, \quad \lambda_1, \ldots, \lambda_N$$

Co-state equations

optimality conditions break into two parts:

$$x_{t+1} = Ax_t + Bu_t, \qquad x_0 = x^{\text{init}}$$

this recursion for state x runs forward in time, with initial condition

$$\lambda_t = A^T\lambda_{t+1} + Qx_t, \qquad \lambda_N = Q_f x_N$$

this recursion for λ runs backward in time, with final condition

- λ is called *co-state*
- recursion for λ sometimes called *adjoint system*

Solution via Riccati recursion

we will see that $\lambda_t = P_t x_t$, where P_t is the min-cost-to-go matrix defined by the Riccati recursion

thus, Riccati recursion gives clever way to solve this set of linear equations

it holds for $t = N$, since $P_N = Q_f$ and $\lambda_N = Q_f x_N$

now suppose it holds for $t+1$, *i.e.*, $\lambda_{t+1} = P_{t+1} x_{t+1}$

let's show it holds for t, *i.e.*, $\lambda_t = P_t x_t$

using $x_{t+1} = Ax_t + Bu_t$ and $u_t = -R^{-1}B^T\lambda_{t+1}$,

$$\lambda_{t+1} = P_{t+1}(Ax_t + Bu_t) = P_{t+1}(Ax_t - BR^{-1}B^T\lambda_{t+1})$$

so

$$\lambda_{t+1} = (I + P_{t+1}BR^{-1}B^T)^{-1}P_{t+1}Ax_t$$

using $\lambda_t = A^T\lambda_{t+1} + Qx_t$, we get

$$\lambda_t = A^T(I + P_{t+1}BR^{-1}B^T)^{-1}P_{t+1}Ax_t + Qx_t = P_tx_t$$

since by the Riccati recursion

$$P_t = Q + A^T(I + P_{t+1}BR^{-1}B^T)^{-1}P_{t+1}A$$

this proves $\lambda_t = P_tx_t$

let's check that our two formulas for u_t are consistent:

$$\begin{aligned} u_t &= -R^{-1}B^T\lambda_{t+1} \\ &= -R^{-1}B^T(I + P_{t+1}BR^{-1}B^T)^{-1}P_{t+1}Ax_t \\ &= -R^{-1}(I + B^TP_{t+1}BR^{-1})^{-1}B^TP_{t+1}Ax_t \\ &= -((I + B^TP_{t+1}BR^{-1})R)^{-1}B^TP_{t+1}Ax_t \\ &= -(R + B^TP_{t+1}B)^{-1}B^TP_{t+1}Ax_t \end{aligned}$$

which is what we had before

Lecture 3
Infinite horizon linear quadratic regulator

- infinite horizon LQR problem
- dynamic programming solution
- receding horizon LQR control
- closed-loop system

Infinite horizon LQR problem

discrete-time system $x_{t+1} = Ax_t + Bu_t$, $x_0 = x^{\text{init}}$

problem: choose $u_0, u_1, \ldots$ to minimize

$$J = \sum_{\tau=0}^{\infty} \left(x_\tau^T Q x_\tau + u_\tau^T R u_\tau \right)$$

with given constant state and input weight matrices

$$Q = Q^T \geq 0, \qquad R = R^T > 0$$

. . . an *infinite dimensional problem*

problem: it's possible that $J = \infty$ for all input sequences $u_0, \ldots$

$$x_{t+1} = 2x_t + 0u_t, \qquad x^{\mathrm{init}} = 1$$

let's assume (A, B) is controllable

then for any x^{init} there's an input sequence

$$u_0, \ldots, u_{n-1},\ 0,\ 0, \ldots$$

that steers x to zero at $t = n$, and keeps it there

for this u, $J < \infty$

and therefore, $\min_u J < \infty$ for any x^{init}

Dynamic programming solution

define **value function** $V : \mathbf{R}^n \to \mathbf{R}$

$$V(z) = \min_{u_0,\ldots} \sum_{\tau=0}^{\infty} \left(x_\tau^T Q x_\tau + u_\tau^T R u_\tau \right)$$

subject to $x_0 = z$, $x_{\tau+1} = Ax_\tau + Bu_\tau$

- $V(z)$ is the minimum LQR cost-to-go, starting from state z
- doesn't depend on time-to-go, which is always ∞; infinite horizon problem is *shift invariant*

Hamilton-Jacobi equation

fact: V is quadratic, *i.e.*, $V(z) = z^T P z$, where $P = P^T \geq 0$
(can be argued directly from first principles)

HJ equation:

$$V(z) = \min_w \left(z^T Q z + w^T R w + V(Az + Bw) \right)$$

or

$$z^T P z = \min_w \left(z^T Q z + w^T R w + (Az + Bw)^T P (Az + Bw) \right)$$

minimizing w is $w^* = -(R + B^T P B)^{-1} B^T P A z$

so HJ equation is

$$\begin{aligned} z^T P z &= z^T Q z + w^{*T} R w^* + (Az + Bw^*)^T P (Az + Bw^*) \\ &= z^T \left(Q + A^T P A - A^T P B (R + B^T P B)^{-1} B^T P A \right) z \end{aligned}$$

this must hold for all z, so we conclude that P satisfies the ARE

$$P = Q + A^T PA - A^T PB(R + B^T PB)^{-1} B^T PA$$

and the optimal input is constant state feedback $u_t = Kx_t$,

$$K = -(R + B^T PB)^{-1} B^T PA$$

compared to finite-horizon LQR problem,

- value function and optimal state feedback gains are time-invariant
- we don't have a recursion to compute P; we only have the ARE

fact: the ARE has only one positive semidefinite solution P

i.e., ARE plus $P = P^T \geq 0$ uniquely characterizes value function

consequence: the Riccati recursion

$$P_{k+1} = Q + A^T P_k A - A^T P_k B(R + B^T P_k B)^{-1} B^T P_k A, \qquad P_1 = Q$$

converges to the unique PSD solution of the ARE
(when (A, B) controllable)

(later we'll see direct methods to solve ARE)

thus, infinite-horizon LQR optimal control is same as steady-state finite horizon optimal control

Receding-horizon LQR control

consider cost function

$$J_t(u_t, \ldots, u_{t+T-1}) = \sum_{\tau=t}^{\tau=t+T} \left(x_\tau^T Q x_\tau + u_\tau^T R u_\tau\right)$$

- T is called *horizon*
- same as infinite horizon LQR cost, truncated after T steps into future

if $(u_t^*, \ldots, u_{t+T-1}^*)$ minimizes J_t, u_t^* is called (T-step ahead) *optimal receding horizon control*

in words:

- at time t, find input sequence that minimizes T-step-ahead LQR cost, starting at current time
- then use only the first input

example: 1-step ahead receding horizon control

find $u_t,\ u_{t+1}$ that minimize

$$J_t = x_t^T Q x_t + x_{t+1}^T Q x_{t+1} + u_t^T R u_t + u_{t+1}^T R u_{t+1}$$

first term doesn't matter; optimal choice for u_{t+1} is 0; optimal u_t minimizes

$$x_{t+1}^T Q x_{t+1} + u_t^T R u_t = (Ax_t + Bu_t)^T Q (Ax_t + Bu_t) + u_t^T R u_t$$

thus, 1-step ahead receding horizon optimal input is

$$u_t = -(R + B^T Q B)^{-1} B^T Q A x_t$$

. . . a constant state feedback

in general, optimal T-step ahead LQR control is

$$u_t = K_T x_t, \qquad K_T = -(R + B^T P_T B)^{-1} B^T P_T A$$

where

$$P_1 = Q, \qquad P_{i+1} = Q + A^T P_i A - A^T P_i B (R + B^T P_i B)^{-1} B^T P_i A$$

$i.e.$: same as the optimal finite horizon LQR control, $T - 1$ steps before the horizon N

- a constant state feedback
- state feedback gain converges to infinite horizon optimal as horizon becomes long (assuming controllability)

Closed-loop system

suppose K is LQR-optimal state feedback gain

$$x_{t+1} = Ax_t + Bu_t = (A + BK)x_t$$

is called *closed-loop system*

($x_{t+1} = Ax_t$ is called *open-loop system*)

is closed-loop system stable? consider

$$x_{t+1} = 2x_t + u_t, \qquad Q = 0, \qquad R = 1$$

optimal control is $u_t = 0x_t$, *i.e.*, closed-loop system is unstable

fact: if (Q, A) observable and (A, B) controllable, then closed-loop system is stable

Lecture 10
Linear Quadratic Stochastic Control with Partial State Observation

- partially observed linear-quadratic stochastic control problem
- estimation-control separation principle
- solution via dynamic programming

Linear stochastic system

- linear dynamical system, over finite time horizon:

$$x_{t+1} = Ax_t + Bu_t + w_t, \qquad t = 0, \ldots, N-1$$

  with state x_t, input u_t, and process noise w_t

- linear noise corrupted observations:

$$y_t = Cx_t + v_t, \qquad t = 0, \ldots, N$$

  y_t is output, v_t is measurement noise

- $x_0 \sim \mathcal{N}(0, X)$, $w_t \sim \mathcal{N}(0, W)$, $v_t \sim \mathcal{N}(0, V)$, all independent

Causal output feedback control policies

- causal feedback policies:
  - input must be function of past and present outputs
  - roughly speaking: current state x_t is *not known*

- $u_t = \phi_t(Y_t), \quad t = 0, \ldots, N-1$
  - $Y_t = (y_0, \ldots, y_t)$ is output history at time t
  - $\phi_t : \mathbf{R}^{p(t+1)} \to \mathbf{R}^m$ called the control policy at time t

- closed-loop system is

$$x_{t+1} = Ax_t + B\phi_t(Y_t) + w_t, \qquad y_t = Cx_t + v_t$$

- $x_0, \ldots, x_N$, $y_0, \ldots, y_N$, $u_0, \ldots, u_{N-1}$ are all random

Stochastic control with partial observations

- objective:

$$J = \mathbf{E}\left(\sum_{t=0}^{N-1}\left(x_t^T Q x_t + u_t^T R u_t\right) + x_N^T Q x_N\right)$$

  with $Q \geq 0$, $R > 0$

- partially observed linear quadratic stochastic control problem (a.k.a. LQG problem):

  choose output feedback policies $\phi_0, \ldots, \phi_{N-1}$ to minimize J

Solution

- optimal policies are $\phi_t(Y_t) = K_t \, \mathbf{E}(x_t|Y_t)$
  - K_t is optimal feedback gain matrix for associated LQR problem
  - $\mathbf{E}(x_t|Y_t)$ is the MMSE estimate of x_t given measurements Y_t (can be computed using Kalman filter)

- called *separation principle*: optimal policy consists of
  - estimating state via MMSE (ignoring the control problem)
  - using estimated state as if it were the actual state, for purposes of control

LQR control gain computation

- define $P_N = Q$, and for $t = N, \ldots, 1$,

$$P_{t-1} = A^T P_t A + Q - A^T P_t B (R + B^T P_t B)^{-1} B^T P_t A$$

- set $K_t = -(R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A, \quad t = 0, \ldots, N-1$

- K_t does not depend on data C, X, W, V

Kalman filter current state estimate

- define
  - $\hat{x}_t = \mathbf{E}(x_t|Y_t)$ (current state estimate)
  - $\Sigma_t = \mathbf{E}(x_t - \hat{x}_t)(x_t - \hat{x}_t)^T$ (current state estimate covariance)
  - $\Sigma_{t+1|t} = A\Sigma_t A^T + W$ (next state estimate covariance)

- start with $\Sigma_{0|-1} = X$; for $t = 0, \ldots, N$,

$$\begin{aligned} \Sigma_t &= \Sigma_{t|t-1} - \Sigma_{t|t-1}C^T(C\Sigma_{t|t-1}C^T + V)^{-1}C\Sigma_{t|t-1}, \\ \Sigma_{t+1|t} &= A\Sigma_t A^T + W \end{aligned}$$

- define $L_t = \Sigma_{t|t-1}C^T(C\Sigma_{t|t-1}C^T + V)^{-1}$, $t = 0, \ldots, N$

- set $\hat{x}_0 = L_0 y_0$; for $t = 0, \ldots, N-1$,

$$\hat{x}_{t+1} = A\hat{x}_t + Bu_t + L_{t+1}e_{t+1}, \quad e_{t+1} = y_{t+1} - C(A\hat{x}_t + Bu_t)$$

  - e_{t+1} is next output prediction error
  - $e_{t+1} \sim \mathcal{N}(0, C\Sigma_{t+1|t}C^T + V)$, independent of Y_t

- Kalman filter gains L_t do not depend on data B, Q, R

Solution via dynamic programming

- let $V_t(Y_t)$ be optimal value of LQG problem, from t on, conditioned on the output history Y_t:

$$V_t(Y_t) = \min_{\phi_t,\ldots,\phi_{N-1}} \mathbf{E}\left(\sum_{\tau=t}^{N-1}(x_\tau^T Q x_\tau + u_\tau^T R u_\tau) + x_N^T Q x_N \;\middle|\; Y_t\right)$$

- we'll show that V_t is a quadratic function plus a constant, in fact,

$$V_t(Y_t) = \hat{x}_t^T P_t \hat{x}_t + q_t, \quad t = 0, \ldots, N,$$

  where P_t is the LQR cost-to-go matrix ($\hat{x}_t$ is a linear function of Y_t)

- we have

$$V_N(Y_N) = \mathbf{E}(x_N^T Q x_N | Y_N) = \hat{x}_N^T Q \hat{x}_N + \mathbf{Tr}(Q\Sigma_N)$$

  (using $x_N|Y_N \sim \mathcal{N}(\hat{x}_N, \Sigma_N)$) so $P_N = Q$, $q_N = \mathbf{Tr}(Q\Sigma_N)$

- dynamic programming (DP) equation is

$$V_t(Y_t) = \min_{u_t} \mathbf{E}\left(x_t^T Q x_t + u_t^T R u_t + V_{t+1}(Y_{t+1}) | Y_t\right)$$

  (and argmin, which is a function of Y_t, is optimal input)

- with $V_{t+1}(Y_{t+1}) = \hat{x}_{t+1}^T P_{t+1} \hat{x}_{t+1} + q_{t+1}$, DP equation becomes

$$\begin{aligned} V_t(Y_t) &= \min_{u_t} \mathbf{E}\left(x_t^T Q x_t + u_t^T R u_t + \hat{x}_{t+1}^T P_{t+1} \hat{x}_{t+1} + q_{t+1} | Y_t\right) \\ &= \mathbf{E}(x_t^T Q x_t | Y_t) + q_{t+1} + \min_{u_t} \left(u_t^T R u_t + \mathbf{E}(\hat{x}_{t+1}^T P_{t+1} \hat{x}_{t+1} | Y_t)\right) \end{aligned}$$

- using $x_t|Y_t \sim \mathcal{N}(\hat{x}_t, \Sigma_t)$, the first term is

$$\mathbf{E}(x_t^T Q x_t|Y_t) = \hat{x}_t^T Q \hat{x}_t + \mathbf{Tr}(Q\Sigma_t)$$

- using

$$\hat{x}_{t+1} = A\hat{x}_t + Bu_t + L_{t+1}e_{t+1},$$

with $e_{t+1} \sim \mathcal{N}(0, C\Sigma_{t+1|t}C^T + V)$, independent of Y_t, we get

$$\mathbf{E}(\hat{x}_{t+1}^T P_{t+1} \hat{x}_{t+1}|Y_t) = \hat{x}_t^T A^T P_{t+1} A\hat{x}_t + u_t^T B^T P_{t+1} B u_t + 2\hat{x}_t^T A^T P_{t+1} B u_t + \mathbf{Tr}\left((L_{t+1}^T P_{t+1} L_{t+1})(C\Sigma_{t+1|t}C^T + V)\right)$$

- using $L_{t+1} = \Sigma_{t+1|t}C^T(C\Sigma_{t+1|t}C^T + V)^{-1}$, last term becomes

$$\mathbf{Tr}(P_{t+1}\Sigma_{t+1|t}C^T(C\Sigma_{t+1|t}C^T+V)^{-1}C\Sigma_{t+1|t}) = \mathbf{Tr}\, P_{t+1}(\Sigma_{t+1|t}-\Sigma_{t+1})$$

- combining all terms we get

$$\begin{aligned} V_t(Y_t) &= \hat{x}_t^T(Q + A^T P_{t+1} A)\hat{x}_t + q_{t+1} + \mathbf{Tr}(Q\Sigma_t) \\ &\quad + \mathbf{Tr}\, P_{t+1}(\Sigma_{t+1|t} - \Sigma_{t+1}) \\ &\quad + \min_{u_t}(u_t^T(R + B^T P_{t+1} B)u_t + 2\hat{x}_t^T A^T P_{t+1} B u_t) \end{aligned}$$

- minimization same as in deterministic LQR problem
- thus optimal policy is $\phi_t^\star(Y_t) = K_t\hat{x}_t$, with

$$K_t = -(R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A$$

- plugging in optimal u_t we get $V_t(Y_t) = \hat{x}_t^T P_t \hat{x}_t + q_t$, where

$$\begin{aligned} P_t &= A^T P_{t+1} A + Q - A^T P_{t+1} B(R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A \\ q_t &= q_{t+1} + \mathbf{Tr}(Q\Sigma_t) + \mathbf{Tr}\, P_{t+1}(\Sigma_{t+1|t} - \Sigma_{t+1}) \end{aligned}$$

- recursion for P_t is exactly the same as for deterministic LQR

Optimal objective

- optimal LQG cost is

$$J^\star = \mathbf{E}\, V_0(y_0) = q_0 + \mathbf{E}\, \hat{x}_0^T P_0 \hat{x}_0 = q_0 + \mathbf{Tr}\, P_0(X - \Sigma_0)$$

  using $\hat{x}_0 \sim \mathcal{N}(0, X - \Sigma_0)$

- using $q_N = \mathbf{Tr}\, Q\Sigma_N$ and

$$q_t = q_{t+1} + \mathbf{Tr}(Q\Sigma_t) + \mathbf{Tr}\, P_{t+1}(\Sigma_{t+1|t} - \Sigma_{t+1})$$

  we get

$$J^\star = \sum_{t=0}^{N} \mathbf{Tr}(Q\Sigma_t) + \sum_{t=0}^{N} \mathbf{Tr}\, P_t(\Sigma_{t|t-1} - \Sigma_t)$$

  using $\Sigma_{0|-1} = X$

- we can write this as

$$J^\star = \sum_{t=0}^{N} \mathbf{Tr}(Q\Sigma_t) + \sum_{t=1}^{N} \mathbf{Tr}\, P_t(A\Sigma_{t-1}A^T + W - \Sigma_t) + \mathbf{Tr}(P_0(X - \Sigma_0))$$

  which simplifies to

$$J^\star = J_{\rm lqr} + J_{\rm est}$$

  where

$$\begin{aligned} J_{\rm lqr} &= \mathbf{Tr}(P_0 X) + \sum_{t=1}^{N} \mathbf{Tr}(P_t W), \\ J_{\rm est} &= \mathbf{Tr}((Q - P_0)\Sigma_0) + \sum_{t=1}^{N} \mathbf{Tr}((Q - P_t)\Sigma_t) + \mathbf{Tr}(P_t A\Sigma_{t-1}A^T) \end{aligned}$$

  - $J_{\rm lqr}$ is the stochastic LQR cost, *i.e.*, the optimal objective if you knew the state
  - $J_{\rm est}$ is the cost of not knowing (*i.e.*, estimating) the state

- when state measurements are exact ($C = I$, $V = 0$), we have $\Sigma_t = 0$, so we get

$$J^\star = J_{\mathrm{lqr}} = \mathbf{Tr}(P_0 X) + \sum_{t=1}^{N} \mathbf{Tr}(P_t W)$$

Infinite horizon LQG

- choose policies to minimize infinite horizon average stage cost

$$J = \lim_{N\to\infty} \frac{1}{N} \mathbf{E} \sum_{t=0}^{N-1} \left(x_t^T Q x_t + u_t^T R u_t \right)$$

- optimal average stage cost is

$$J^\star = \mathbf{Tr}(Q\Sigma) + \mathbf{Tr}(P(\tilde{\Sigma} - \Sigma))$$

  where P and $\tilde{\Sigma}$ are PSD solutions of AREs

$$\begin{aligned} P &= Q + A^T P A - A^T P B (R + B^T P B)^{-1} B^T P A, \\ \tilde{\Sigma} &= A \tilde{\Sigma} A^T + W - A \tilde{\Sigma} C^T (C \tilde{\Sigma} C^T + V)^{-1} C \tilde{\Sigma} A^T \end{aligned}$$

  and $\Sigma = \tilde{\Sigma} - \tilde{\Sigma} C^T (C \tilde{\Sigma} C^T + V)^{-1} C \tilde{\Sigma}$

- optimal average stage cost doesn't depend on X
- (an) optimal policy is

$$u_t = K\hat{x}_t, \quad \hat{x}_{t+1} = A\hat{x}_t + Bu_t + L(y_{t+1} - C(A\hat{x}_t + Bu_t))$$

  where

$$K = -(R + B^T P B)^{-1} B^T P A, \qquad L = \tilde{\Sigma} C^T (C\tilde{\Sigma} C^T + V)^{-1}$$

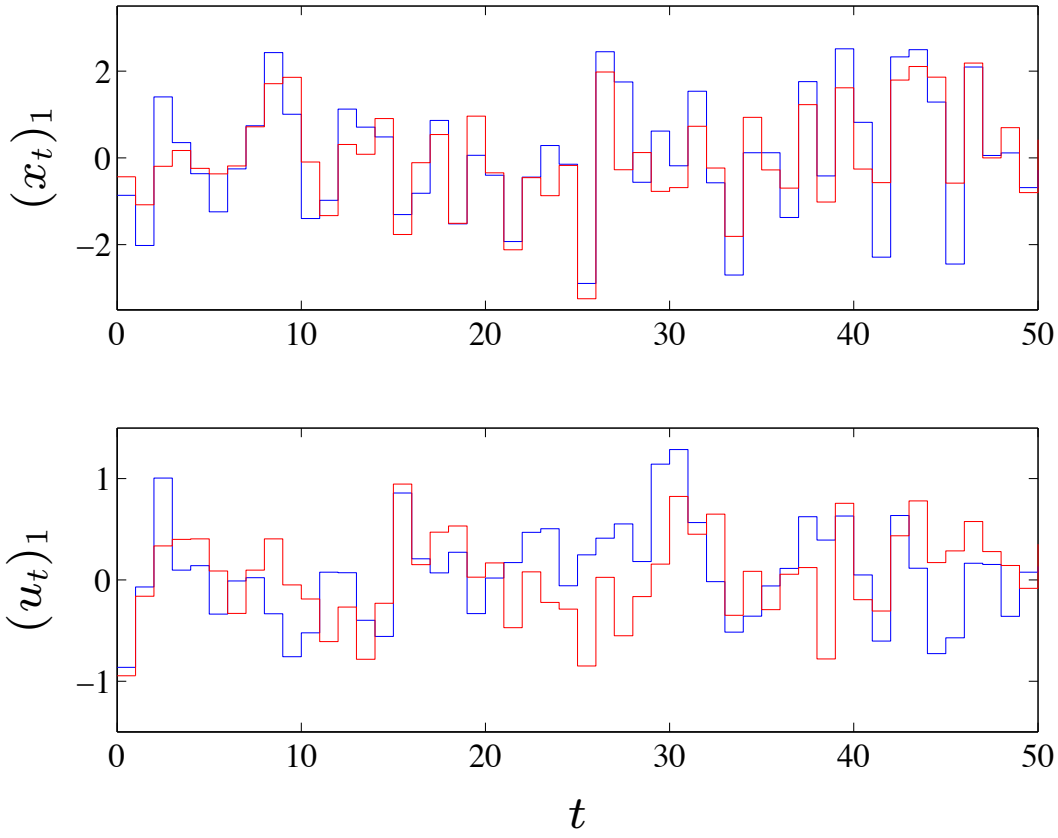
- K is steady-state LQR feedback gain
- L is steady-state Kalman filter gain

Example

- system with $n = 5$ states, $m = 2$ inputs, $p = 3$ outputs; infinite horizon
- A, B, C chosen randomly; A scaled so $\max_i |\lambda_i(A)| = 1$
- $Q = I$, $R = I$, $X = I$, $W = 0.5I$, $V = 0.5I$
- we compare LQG with the case where state is known (stochastic LQR)

Sample trajectories

sample trace of $(x_t)_1$ and $(u_t)_1$ in steady state



blue: LQG, red: stochastic LQR

Cost histogram

histogram of stage costs for 5000 steps in steady state