Module 2-2: Introduction to Optimal Control

Linear Control Systems (2020)

Ding Zhao

Assistant Professor

College of Engineering

School of Computer Science

Carnegie Mellon University

Introduction to Optimal Control

- With pole placement, we learned how to place the eigenvalues at desired locations for MIMO systems, however
  - What eigenvalues should we choose? Not a very clear relationship between eigenvalues and performance.
  - Even if we do know what eigenvalues we want, the state feedback is not unique (for MIMO systems). Then what's the "best" choice of K?
- The solution: using Optimal Control!
- We will focus on one of the most useful optimal control schemes: Linear Quadratic Control
  - linear system dynamics
  - minimizing a quadratic (i.e. second order) function of the state and control

Table of Contents

Table of Contents

Finite Horizon (FH) Discrete-Time (DT) LQR

- Without loss of generality, the goal is set to **reach 0** from $x(0)$ in N steps.
- When the desired control target is a constant, we call the controller a regulator. If we need to follow a trajectory over time, we call it a "tracking" problem. In regulation problem, feedback is enough, while in tracking, it is desired to also have feedforward. We will work on the feedback controller first.
- The cost function (sometimes called "cost-to-go") is

$$J_{0,N} = \frac{1}{2} x(N)^T S_N x(N) + \frac{1}{2} \sum_{k=0}^{N-1} (x(k)^T Q x(k) + u(k)^T R u(k))$$

  with S_N, Q, R constant positive definite matrices.
- S_N, Q, R are weights to "penalize" state and control. Higher entries mean you care more about a particular state or control
- To solve for this problem, first we need to understand the Dynamic Programming.

Dynamic Programming

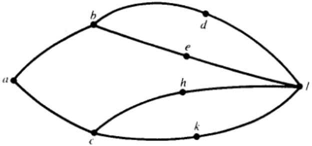
Bellman, 1920-1984

$J^*_{a,l}$ is the **min** cost from node a to the last node l.

$$J^*_{a,l} \triangleq \min\left[J_{a,b} + J^*_{b,l}, J_{a,c} + J^*_{c,l}\right] (*)$$

$$J^*_{b,l} = \min\left[J_{b,d} + J^*_{d,l}, J_{b,e} + J^*_{e,l}\right] = \min\left[J_{b,d} + J_{d,l}, J_{b,e} + J_{e,l}\right],$$

$$J^*_{c,l} = \min\left[J_{c,h} + J^*_{h,l}, J_{c,k} + J^*_{k,l}\right] = \min\left[J_{c,h} + J_{h,l}, J_{c,k} + J_{k,l}\right]$$



- recursive function
- Instantaneous cost: $J_{a,b}$ and $J_{a,c}$
- Minimum values of all future costs: $J^*_{a,l}, J^*_{b,l}, J^*_{c,l}$
- Start with the last optimal step and moving backward to the first
- We can also do this in time step (our focus in this lecture

Dynamic Programming - My Graduation Trip

Optimal Control for FH-DT-LQR (1)

- Start from the last step.

$$J_{N-1,N} = \frac{1}{2}x(N)^T S_N x(N) + \frac{1}{2}x(N-1)^T Q x(N-1) + \frac{1}{2}u(N-1)^T R u(N-1)$$

- We are also subject to the dynamics

$$x(N) = Ax(N-1) + Bu(N-1)$$

$$\begin{aligned} \Rightarrow J_{N-1,N} = &\frac{1}{2}((Ax(N-1) + Bu(N-1))^T S_N (Ax(N-1) + Bu(N-1)) \\ &+ \frac{1}{2}x(N-1)^T Q x(N-1) + \frac{1}{2}u(N-1)^T R u(N-1)) \end{aligned}$$

Matrix Calculus

Vector-by-scalar

$$\frac{\partial \mathbf{y}}{\partial x} = \begin{bmatrix} \frac{\partial y_1}{\partial x} \\ \frac{\partial y_2}{\partial x} \\ \vdots \\ \frac{\partial y_m}{\partial x} \end{bmatrix}$$

$$\dot{x} = \frac{\partial \mathbf{x}}{\partial t} = \begin{bmatrix} \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial t} \\ \vdots \\ \frac{\partial x_n}{\partial t} \end{bmatrix}$$

Scalar-by-vector

$$\frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \frac{\partial y}{\partial x_2} & \cdots & \frac{\partial y}{\partial x_n} \end{bmatrix}$$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} = \left(\frac{\partial f}{\partial \mathbf{x}}\right)^T = \frac{\partial^T f}{\partial \mathbf{x}}$$

Chain rule: $\dfrac{\partial g(u)}{\partial \mathbf{x}} = \dfrac{\partial g(u)}{\partial u} \dfrac{\partial u}{\partial \mathbf{x}}$

Vector-by-vector

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

Jacobian $\mathbf{A} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial x_1} & \cdots & \dfrac{\partial f_m}{\partial x_n} \end{bmatrix}$

$$\frac{\partial \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{A}$$

$\dfrac{\partial \mathbf{a}^\top \mathbf{x}}{\partial \mathbf{x}} = \dfrac{\partial \mathbf{x}^\top \mathbf{a}}{\partial \mathbf{x}} = \mathbf{a}^\top$, $\dfrac{\partial \mathbf{x}^\top \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{x}^\top \left(\mathbf{A} + \mathbf{A}^\top\right) = 2\mathbf{x}^\top \mathbf{A}$ (if A is symmetric)

$\nabla \mathbf{u}(\mathbf{x})^\top \mathbf{A}\mathbf{u}(\mathbf{x}) = (\dfrac{\partial \mathbf{u}(\mathbf{x})^\top \mathbf{A}\mathbf{u}(\mathbf{x})}{\partial \mathbf{x}})^T = 2(\dfrac{\partial \mathbf{u}(\mathbf{x})}{\partial \mathbf{x}})^T \mathbf{A}\mathbf{u}(\mathbf{x})$ (if A is symmetric)

Optimal Control for FH-DT-LQR (2)

- To find the optimal control, differentiate with respect to $u(N-1)$

$$
\begin{aligned}
\frac{\partial^T J_{N-1,N}}{\partial u(N-1)} &= 0 \\
&= B^T S_N [Ax(N-1) + Bu(N-1)] + Ru(N-1) \\
&= [R + B^T S_N B]u(N-1) + B^T S_N Ax(N-1) \\
\Rightarrow u^*(N-1) &= -[R + B^T S_N B]^{-1} B^T S_N Ax(N-1)
\end{aligned}
$$

- The second derivative $R + B^T S_N B > 0 \Rightarrow$ global minimum
  - The control is of the form

$$
u^*(N-1) = K_{N-1} x(N-1)
$$

$$
\text{with constant } K_{N-1} = -(R + B^T S_N B)^{-1} B^T S_N A
$$

Optimal Control for FH-DT-LQR (3)

With little algebra, we have

$$J^*_{N-1,N} = \frac{1}{2}x(N-1)^T[(A+BK_{N-1})^T S_N(A+BK_{N-1}) + Q + K^T_{N-1}RK_{N-1}]x(N-1)$$

Let $S_{N-1} = \frac{1}{2}[(A+BK_{N-1})^T S_N(A+BK_{N-1}) + Q + K^T_{N-1}RK_{N-1}]$. Just like S_N, we find S_{N-1} is also a constant!
We have

$$J^*_{N-1,N} = \frac{1}{2}x(N-1)^T S_{N-1}x(N-1)$$

We get a general form for $J^*_{N-1,N}$ given $x(N-1)$.

Optimal Control for FH-DT-LQR (4)

Step backwards again

$$J^*_{N-2,N} = \min_{u(N-2:N-1)} [J^{intermedia}_{N-2,N-1} + J_{N-1,N}]$$

because $J^*_{N-1,N} \leq J_{N-1,N}$

$$J^*_{N-2,N} = \min_{u(N-2:N-1)} [J^{intermedia}_{N-2,N-1} + J^*_{N-1,N}]$$

because $J^*_{N-1,N} = \frac{1}{2}x(N-1)^T S_{N-1} x(N-1)$, which is independent of $u(N-1)$

We will drop $u(N-1) \Rightarrow J^*_{N-2,N} = \min_{u(N-2)} [J^{intermedia}_{N-2,N-1} + J^*_{N-1,N}]$

$$= \min_{u(N-2)} \frac{1}{2}x(N-1)^T S_{N-1} x(N-1) + \frac{1}{2}x(N-2)^T Q x(N-2) + \frac{1}{2}u(N-2)^T R u(N-2)$$

Optimal Control for FH-DT-LQR (5)

- Since S_{N-1} is a constant, this is identical in form to $J_{N-1,N}$, and the solution is similar

$$u^*(N-2) = K_{N-2}x(N-2)$$
$$\text{with } K_{N-2} = -(R + B^T S_{N-1} B)^{-1} B^T S_{N-1} A$$

- We can continue, resulting in the general formula

$$K_k = -(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A$$
$$S_k = (A + BK_k)^T S_{k+1} (A + BK_k) + Q + K_k^T R K_k$$

- We can iteratively calculate $S_N \Rightarrow K_{N-1} \Rightarrow S_{N-1} \Rightarrow K_{N-2} \Rightarrow S_{N-2} \Rightarrow \cdots \Rightarrow S(0)$, then use $u^*(k) = K_k x(k)$

Optimal Control for FH-DT-LQR (6)

We could combine the two equations together by substitute K_k in the S_K equation:

$$
\begin{aligned}
K_k &= -(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A \\
S_k &= (A + BK_k)^T S_{k+1} (A + BK_k) + Q + K_k^T R K_k \\
&= A^T S_{k+1} A + A^T S_{k+1} B K_k + \underbrace{K_k^T B^T S_{k+1} A + K_k^T B^T S_{k+1} B K_k + K_k^T R K_k}_{\text{This parts become } 0 \text{ by substituting } K_k\text{!}} + Q \\
&= A^T S_{k+1} A - A^T S_{k+1} B (R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A + Q
\end{aligned}
$$

$$
S_k = A^T S_{k+1} A - A^T S_{k+1} B (R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A + Q
$$

is called "Discrete-time Difference Riccati Equation". Can be computed backwards with k from N to 0 given S_N. Then K_k can be directly computed by the first equation.

Recap: Finite Horizon Discrete-Time Linear Quadratic Regulator

Design control policy to minimize the cost function

$$J_{0,N} = \frac{1}{2}x(N)^T S_N x(N) + \frac{1}{2}\sum_{k=0}^{N-1}\left(x(k)^T Q x(k) + u(k)^T R u(k)\right)$$

where $S_N, Q, R \geq 0$, subject to the system dynamics

$$x(k+1) = Ax(k) + Bu(k)$$

Induction backwards in time to obtain the optimal control solution at each time. We found the minimal cost from state k to the end has an elegant solution

$$\min J_{k,N} = J^*_{k,N} = \frac{1}{2}x(k)^T S_k x(k)$$

where $S_k \geq 0$ is a constant only dependent on A, B, S_N, Q, R.

Recap: Solve FH-DT-LQR with Principle of Optimality

1. $J^*_{N,N} = \frac{1}{2}x(N)^T S_N x(N)$
2. $J^*_{N-1,N} = \min\{J^*_{N,N} + \frac{1}{2}x(N-1)^T Q x(N-1) + \frac{1}{2}u(N-1)^T R u(N-1)\}$
   $= \min\{\frac{1}{2}x(N)^T S_N x(N) + \frac{1}{2}x(N-1)^T Q x(N-1) + \frac{1}{2}u(N-1)^T R u(N-1)\}$
   $u^*(N-1) = -(R + B^T S_N B)^{-1} B^T S_N A x(N-1) = K_{N-1} x(N-1)$

$$
\begin{aligned}
J^*_{N-1,N} &= \frac{1}{2}x(N-1)^T[(A+BK_{N-1})^T S(A+BK_{N-1}) + Q + K^T_{N-1} R K_{N-1}]x(N-1) \\
&= \frac{1}{2}x(N-1)^T S_{N-1} x(N-1)
\end{aligned}
$$

3. $S_N \Rightarrow K_{N-1} \Rightarrow S_{N-1} \Rightarrow K_{N-2} \Rightarrow S_{N-2} \Rightarrow \cdots \Rightarrow S(0)$

$$
\begin{aligned}
K_k &= -(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A \\
S_k &= (A + BK_k)^T S_{k+1} (A + BK_k) + Q + K_k^T R K_k
\end{aligned}
$$

$\Rightarrow S_k = A^T S_{k+1} A - A^T S_{k+1} B (R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A + Q$ (DT Difference Riccati Equation)

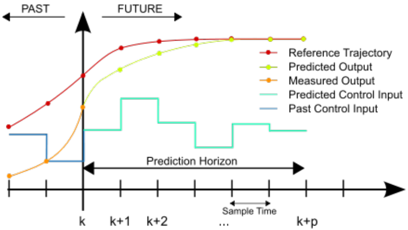
Recap: Matrix Equations

Discrete-time Difference Riccati Equation	Continuous-time Differential Riccati Equation
$S_k = A^T S_{k+1} A - A^T S_{k+1} B$ $(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A + Q$	$\dot{P}(t) = -Q + P(t) B R^{-1} B^T P(t) - P(t) A$ $-A^T P(t)$

Discrete-time Algebraic Riccati Equation (DARE)	Continuous-time Algebraic Riccati Equation (CARE)
$S = A^T S A - A^T S B$ $(R + B^T S B)^{-1} B^T S A + Q$	$A^T P + P A - P B R^{-1} B^T P + Q = 0$

Table of Contents

FH-DT-LQR, Constrained LQR, & MPC



Finite-Horizon Discrete-Time Linear Quadratic Regulator

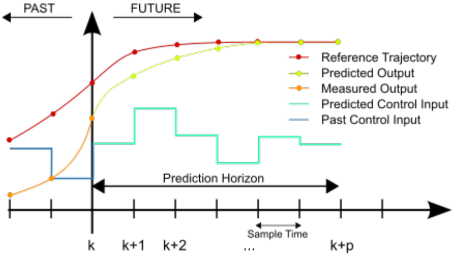
Design control policy to minimize the cost function

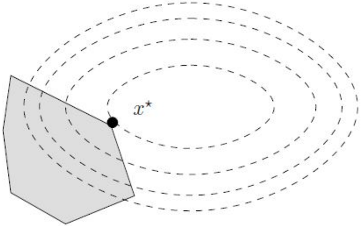
$$J_{0,N} = \frac{1}{2}x(N)^T S_N x(N) + \frac{1}{2}\sum_{k=0}^{N-1}\left(x(k)^T Q x(k) + u(k)^T R u(k)\right)$$

where $S_N, Q, R \geq 0$, subject to the system dynamics

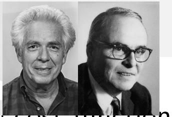
$$x(k+1) = Ax(k) + Bu(k)$$

FH-DT-LQR, Constrained LQR, & MPC





Karush 1917-1997
Kuhn 1925-2014
Tucker 1905-1995



Constrained LQR

Design control policy to minimize the cost function

$$J_{0,N} = \frac{1}{2} x(N)^T S_N x(N) + \frac{1}{2} \sum_{k=0}^{N-1} (x(k)^T Q x(k) + u(k)^T R u(k$$

where $S_N, Q, R \geq 0$, subject to the system dynamics

$$x(k+1) = Ax(k) + Bu(k)$$

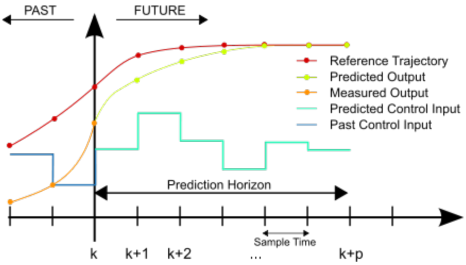
and constraints

$$Hx(k) \leq h$$

$$Fu(k) \leq f$$

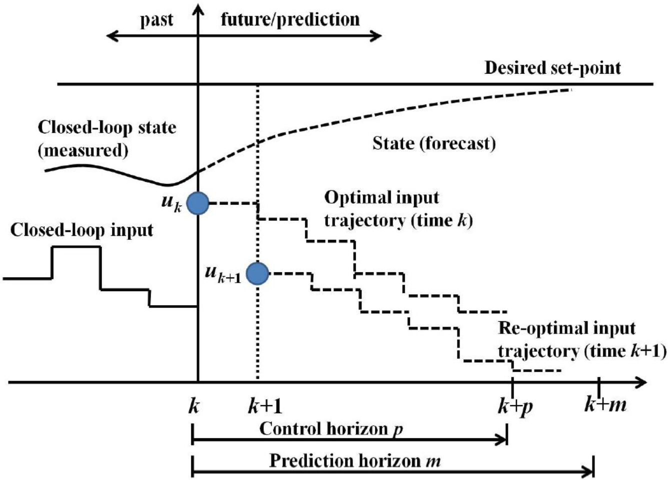
Quadratic programming
Many efficient and reliable algorithms available (KKT).

LQR, Constrained LQR, & MPC



Calculate $u^*(k:k+N)$, but only use $u^*(k)$ and recalculate $u^*(k+1:k+N+1)$ in the next step. Essentially, it is a closed loop version of Constrained LQR, therefore, could be more robust by increasing computation budget.

(Linear) Modal Predictive Control or "Receding Horizon Control"



Real-World Application of MPC

- More and more popular
- Used in all kinds of industry, economics, politics, etc
- Thanks to the fast development of compututational capability
- Many variants: Nonlinear MPC (nonlinear programming), Explicit MPC (offline, parametric programming for different control regions), etc

Table of Contents

Infinite Horizon Control

- In many mathematical cases, taking $k \to \infty$ may surprisingly simplify the analysis.
- It may cause little or no degradation in optimality because the optimal time-varying gains approach constant values in a few time constants after the control error diminishes.

Define

$$J_{0,\infty} = \sum_{k=0}^{\infty} x(k)^T Q x(k) + u(k)^T R u(k)$$

with

$$x(k+1) = Ax(k) + Bu(k)$$

Note that we do not have S_N penalty term in J because $\lim_{N \to \infty} x(N) = 0$

Solve Infinite Horizon Discrete-Time Linear Quadratic Regulator

$$
\begin{aligned}
K_k &= -(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A \\
S_k &= (A + BK_k)^T S_{k+1} (A + BK_k) + Q + K_k^T R K_k
\end{aligned}
$$

$\Rightarrow S_{k-1} = A^T S_k A - A^T S_k B (R + B^T S_k B)^{-1} B^T S_k A + Q$ (DT Difference Riccati Equation)

As $k \to \infty$, K_k, S_k converge. We can drop the index k

$$
\begin{aligned}
K^* &= -(R + B^T S B)^{-1} B^T S A \\
S &= A^T S A - A^T S B (R + B^T S B)^{-1} B^T S A + Q
\end{aligned}
$$

The form of this equation of S is called Discrete-time Algebraic Riccati Equation (DARE)!
And there are powerful numerical algorithms to solve it!

Python Code

```
from __future__ import division, print_function
import numpy as np
import scipy.linalg
def dlqr(A,B,Q,R):
"""Solve the discrete time lqr controller.
x[k+1] = A x[k] + B u[k]
cost = sum x[k].T*Q*x[k] + u[k].T*R*u[k]
"""
#ref Bertsekas, p.151
#first, try to solve the ricatti equation
S = np.matrix(scipy.linalg.solve_discrete_are(A, B, Q, R))
#compute the LQR gain
K = -np.matrix(scipy.linalg.inv(B.T@S@B+R)@(B.T@S@A))
eigVals, eigVecs = scipy.linalg.eig(A+B@K)
return K, X, eigVals
```

Recap: Matrix Equations

Discrete-time Difference Riccati Equation	Continuous-time Differential Riccati Equation
$S_k = A^T S_{k+1} A - A^T S_{k+1} B$ $(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A + Q$	$\dot{P}(t) = -Q + P(t) B R^{-1} B^T P(t) - P(t) A$ $-A^T P(t)$

Discrete-time Algebraic Riccati Equation (DARE)	Continuous-time Algebraic Riccati Equation (CARE)
$S = A^T S A - A^T S B$ $(R + B^T S B)^{-1} B^T S A + Q$	$A^T P + P A - P B R^{-1} B^T P + Q = 0$

Solving Discrete-time Algebraic Riccati Equation

Unlike the finite horizon problem, we have to worry about stability, existence and uniqueness.

Theorem

Let Q be factored into $Q = T^T T$. A unique and positive definite solution to the DARE exists $\Leftrightarrow$ (A, B) is stabilizable (ensures convergence of J and existence of K) and (A, T) (ensures uniqueness) is detectable.

- For most of the system, (A, B) is stabilizable and (A, C) is often detectable. Therefore, Q is often chosen to be $C^T C$ to make (A, T) detectable.

Summary of IH-DT-LQR

Three ingredients to make IH-DT-LQR much computationally lighter compared to FH-DT-LQR:

$N \to \infty$, constant K, Discrete-time Algebraic Riccati Equation

$$\min_K J = \sum_{k=0}^{\infty} x(k)^T Q x(k) + u(k)^T R u(k)$$

with

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ u(k) = Kx(k) \end{cases}$$

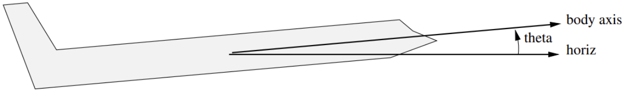
Optimal solution:

$$K^* = -(R + B^T S B)^{-1} B^T S A$$

Where S can be solved by a DARE

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

Example: Longitudinal Flight Control



(Courtesy: Jenny Hong, Nicholas Moehle, Stephen Boyd, EE103 Stanford University) Variables are (small) deviations from operating point or trim conditions;

State is $x_t = (w_t, v_t, \theta_t, q_t)$

- w_t: velocity of aircraft along body axis
- v_t: velocity of aircraft perpendicular to body axis (down is positive)
- θ_t: angle between body axis and horizontal (up is positive)
- $q_t = \dot{\theta}_t$: angular velocity of aircraft (pitch rate)

Control Input is $u_t = (e_t, f_t)$:

- e_t: elevator angle ($e_t > 0$ is down)
- f_t: thrust

Example: Longitudinal Flight Control

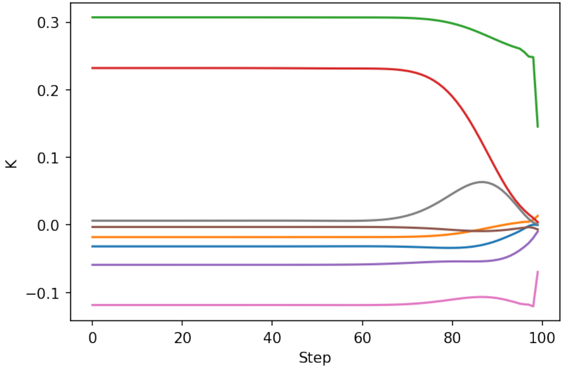
For Boeing 747, level flight, 40000 ft, 774 ft/sec, dynamics are $x_{k+1} = Ax_k + Bu_k$, where

$$A = \begin{bmatrix} .99 & .03 & -.02 & -.32 \\ .01 & .47 & 4.7 & .00 \\ .02 & -.06 & .40 & -.00 \\ .01 & -.04 & .72 & .99 \end{bmatrix}, \quad B = \begin{bmatrix} 0.01 & 0.99 \\ -3.44 & 1.66 \\ -0.83 & 0.44 \\ -0.47 & 0.25 \end{bmatrix}$$

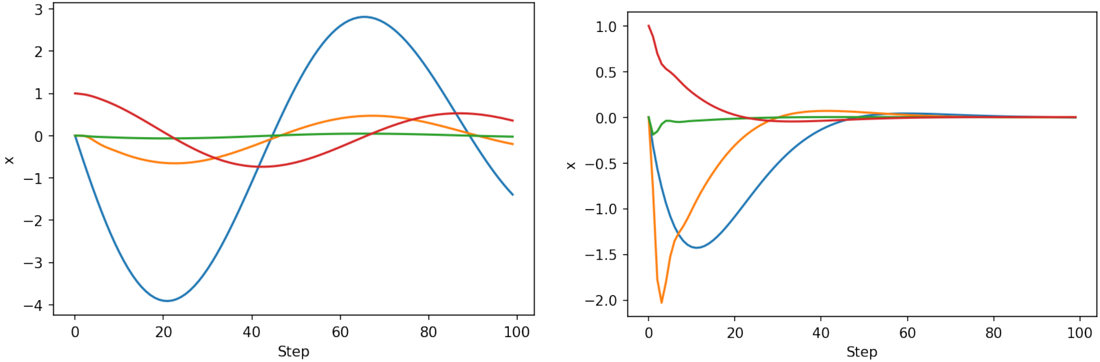
- units: ft, sec, crad($= 0.01$rad)
- discretization is 1 sec

Code: `https://colab.research.google.com/drive/1xxfcQYjAvyYs8KDMi3cx6VCi8QbbQbU1?usp=sharing`

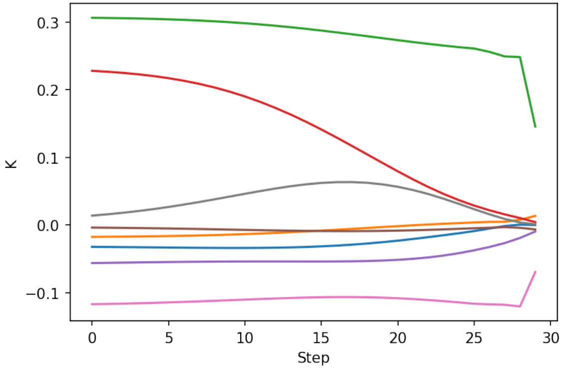
Example: T=100



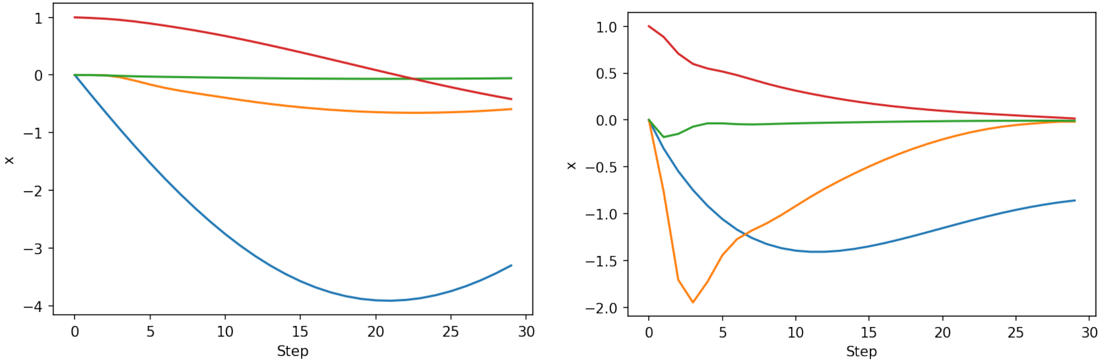
Example: T=100, Open-loop vs. Closed-loop



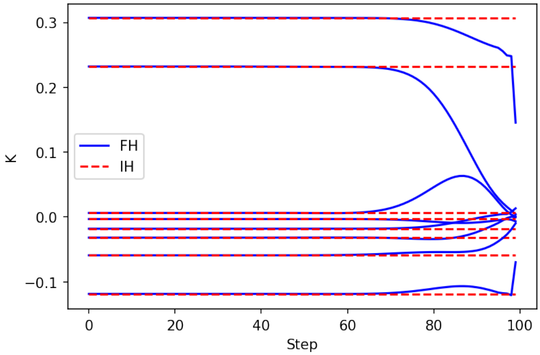
Example: T=30



Example: T=30, Open-loop vs. Closed-loop



Example: T=100 vs. T=30, Comparison between FH and IH



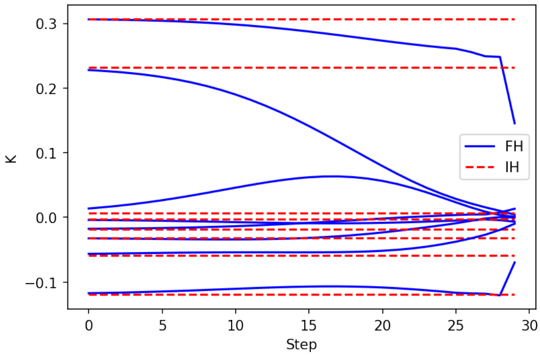


Table of Contents

Formation of the Tracking Problem

$$J_{0,N} = \frac{1}{2}\left[x(N) - x_d\right]^T S_N \left[x(N) - x_d\right] + \frac{1}{2}\sum_{k=0}^{N-1}\left\{[x(k) - \eta(k)]^T Q [x(k) - \eta(k)] + u(k)^T R u(k)\right\}$$

- This cost function attempts to make $x(k)$ follow the specified sequence $\eta(k)$
- Most of the solution details are the same as for the regulator problem. Expanding the quadratic terms in J shows that there are four additional terms to deal with, due to $x_d(k)$ and $\eta(k)$.

General Solution of Regulator Problem

With dynamic programing, the min cost at step k can be expressed as a difference equation

$$J_{k,N}^* = \min_{u(k)} \left\{ \frac{1}{2} x^T(k) Q x(k) + \frac{1}{2} u^T(k) R u(k) + J_{k+1,N}^* \right\} \tag{1}$$

with boundary condition: $g[x(N)] = \frac{1}{2} x(N)^T S_N x(N) - x(N)^T S_N x_d + \frac{1}{2} x_d^T S_N x_d$
For tracking problem, $J_{k,N}^*$ cannot be expressed as the simple form $J_{k,N}^* = \frac{1}{2} x(k)^T S_k x(k)$.
We solve by assuming a general solution quadratic matrices function of $x(k)$:

$$J_{k,N}^* = \frac{1}{2} x(k)^T S_k x(k) + x(k)^T V_k + Z_k \tag{2}$$

where $S_k \in \mathbb{R}^{n \times n}, V_k \in \mathbb{R}^{n \times 1}$ and scalar Z_k are all unknowns.

Derivation

Substitute Eq. (2) into Eq. (1)

$$J^*_{k,N} = \min_{u(k)} \{ \frac{1}{2} x^T(k) Q x(k) + \frac{1}{2} u^T(k) R u(k) + \\ \frac{1}{2} x(k+1)^T S_{k+1} x(k+1) + x(k+1)^T V_{k+1} + Z_{k+1} \}$$

Substitute $x(k+1) = Ax(k) + Bu(k)$

$$J^*_{k,N} = \min_{u(k)} \left\{ \frac{1}{2} x(k)^T \left[Q + A^T S_{k+1} A \right] x(k) + \frac{1}{2} u(k)^T \left[R + B^T S_{k+1} B \right] u(k) \right. \\ \left. + u(k)^T \left[B^T V_{k+1} + B^T S_{k+1} A x(k) \right] + x(k)^T A^T V_{k+1} + Z_{k+1} \right\} \tag{3}$$

Derivation Cont.

The minimizing $u(k)$ is found by setting $\partial\{\}/\partial u(k) = 0$.
(Assume $u(k)$ does not exceed its limits.)

$$\begin{aligned} u^*(k) &= -\left[R + B^T S_{k+1} B\right]^{-1} B^T [V_{k+1} + S_{k+1} A x(k)] \\ &= \underbrace{G_{ff}(k) V_{k+1}}_{\text{feedforward control}} - \underbrace{G_{fb}(k) x(k)}_{\text{feedback control}} \end{aligned}$$

Define $U_{k+1} = \left[R + B^T S_{k+1} B\right]^{-1}$. The goal of tracking controller design is to find the optimal:

- Feedforward control gain: $G_{ff}(k) = -U_{k+1} B^T$
- Feedback control gain: $G_{fb}(k) = U_{k+1} S_{k+1} A$
- Feedforward input: V_{k+1}

Derivation Cont.: Three Key Variables

Substitute $u(k)$ into Eq. (3)

$$
\begin{aligned}
J^*_{k,N} =& \frac{1}{2} x(k)^T \left[Q + A^T S_{k+1} A - A^T S_{k+1} B U_{k+1} B^T S_{k+1} A\right] x(k) \\
& + x(k)^T \left[A^T V_{k+1} - A^T S_{k+1} B U_{k+1} B^T V_{k+1}\right] \\
& + [Z_{k+1} - \frac{1}{2} V^T_{k+1} B U_{k+1} B^T V_{k+1}]
\end{aligned}
$$

Remember we assume $J^*_{k,N} = \frac{1}{2} x(k)^T S_k x(k) + x(k)^T V_k + Z_k$. Compare the quadratic terms, the linear terms, and the terms not involving x, we have the equations to solve S_k, V_k, and Z_k.

Derivation Cont.: Three Key Variables

$$S_k = Q + A^T S_{k+1} A - A^T S_{k+1} B U_{k+1} B^T S_{k+1} A \tag{4}$$

$$V_k = A^T V_{k+1} - A^T S_{k+1} B U_{k+1} B^T V_{k+1} - Q\eta(k) \tag{5}$$

$$Z_k = Z_{k+1} - \frac{1}{2} V_{k+1}^T B U_{k+1} B^T V_{k+1} + \frac{1}{2}\eta(k)^T Q \eta(k) \tag{6}$$

with the boundary conditions $W_N = S_N, V_N = -S_N x_d$, and $Z_N = \frac{1}{2} x_d^T S_N x_d$.

- Since $U_{k+1} = \left[R + B^T S_{k+1} B\right]^{-1}$, a computer, backward in time, easily solves S_k.
- Then solve V_k in Eq. (5) using S_k as a known coefficient matrix and we know $\eta(k)$.
- Now we could calculate Eq. (6) given S_k, V_k are available. But it actually never needs to be solved if the only interest is in finding the optimal control. Z_k is only needed if we want to know $J_{k,N}^*$.
- For regulator, $x_d = \eta(k) = 0$. Eq. (5) becomes a homogeneous equation with zero initial conditions, so V_k is zero for all stages. Similarly, Z_k is 0 too. Therefore, we get back to the quadratic term for $J_{k,N}^* = \frac{1}{2} x(k)^T S_k x(k)$.

Comparison of Feedback and Feedforward Control

- Feedback (FB) Control

Advantages:

- Corrective action occurs regardless of the source and type of disturbances.
- Requires little knowledge about the desired tracking trajectories
- Versatile and robust (Conditions change? May have to re-tune controller).

Disadvantages:

- FB control takes no corrective action until a deviation in the controlled variable occurs.
- FB control is incapable of correcting a deviation from set point at the time of its detection.
- Theoretically not capable of achieving "perfect control."
- For frequent and severe disturbances, process may not settle out.

Comparison of Feedback and Feedforward Control

- Feedforward (FF) Control

Advantages:

- Takes corrective action before the disturbance arrives
- Theoretically capable of "perfect control"
- Does not affect system stability.

Disadvantages:

- Disturbance must be measured (operating costs)

Feedforward Plus Feedback Control

- FF Control: Attempts to eliminate the effects of measurable (nonlinear) disturbances.
- FB Control: Corrects for unmeasurable disturbances, modeling errors, etc.

Table of Contents

Finite Horizon Continuous-Time Linear Quadratic Regulator (1)

- Derive on the blackboard again. The continuous version of the FH LQR is in the form

$$\text{minimize } J = \frac{1}{2}x^T(t_f)\,Sx(t_f) \;+\; \frac{1}{2}\int_{t_0}^{t} x^T(t)\,Qx(t) \;+\; u^T(t)\,Ru(t)\,dt$$

  subject to $\dot{x}(t) = Ax(t) + Bu(t)$

- Define the "cost-to-go" function $J(x(t), t), t \in [t_0, t_f]$ as the cost from t to t_f. As in DT, break the procedure into two steps: from t to $t + \delta t$ and from $t + \delta t$ to t_f

$$\begin{aligned} J(x(t), t) =& \frac{1}{2}\int_{t}^{t+\delta t} [x^T(t)\,Qx(t) + u^T(t)\,Ru(t)]dt \\ &+ \frac{1}{2}x^T(t_f)\,Sx(t_f) + \frac{1}{2}\int_{t+\delta t}^{t_f} [x^T(t)\,Qx(t) + u^T(t)\,Ru(t)]dt \end{aligned}$$

- Apply Dynamic Programming. The optimal "cost-to-go" function

$$J^*(x(t), t) = \min_{u(t:t_f)} \left\{ \frac{1}{2}\int_{t}^{t+\delta t} (x^T(t)Qx(t) + u^T(t)Ru(t))dt + J^*(x(t+\delta t), t+\delta t) \right\}$$

Finite Horizon Continuous-Time Linear Quadratic Regulator (2)

- When δt is small, $x(t+\delta t) - x(t) = \dot{x}(t)\delta t = (Ax(t) + Bu(t))\delta t$.
- Expand the last term with the Taylor series.

$$J^*(x(t+\delta t), t+\delta t) \approx J^*(x(t), t) + \frac{\partial J^*}{\partial t}|_{x(t),t}(t+\delta t - t) + \frac{\partial J^*}{\partial x}|_{x(t),t}(x(t+\delta t) - x(t))$$

$$= J^*(x(t), t) + \frac{\partial J^*}{\partial t}|_{x(t),t}\delta t + \frac{\partial J^*}{\partial x}|_{x(t),t}(Ax(t) + Bu(t))\delta t$$

Substitute this back to $J^*(x(t), t)$, $J^*(x(t), t)$

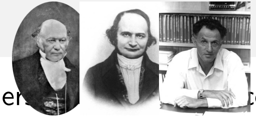
$$= \min_u \left\{ \frac{1}{2}\int_t^{t+\delta t}(x^TQx + u^TRu)dt + J^*(x(t), t) + \frac{\partial J^*}{\partial t}\delta t + \frac{\partial J^*}{\partial x}(Ax + Bu)\delta t \right\}$$

$J^*(x(t), t)$ cancelled $\Rightarrow$

$$\frac{\partial J^*}{\partial t}\delta t + \min_u \left\{ \frac{1}{2}\left(x^TQx + u^TRu\right)\delta t + \frac{\partial J^*}{\partial x}(Ax(t) + Bu(t))\delta t \right\} = 0$$

Hamilton-Jacobi-Bellman Equation

Hamilton 1805-1865
Jacobi 1804-1851
Bellman 1920-1984



- We get the Hamilton-Jacobi-Bellman Equation, one of the corner control

$$\frac{\partial J^*}{\partial t} + \min_u \left\{ \frac{1}{2} \left(x^T Q x + u^T R u \right) + \frac{\partial J^*}{\partial x} (Ax + Bu) \right\} = 0$$

- Define the Hamiltonian

$$H\left(x, u, J^*, t\right) \equiv \frac{1}{2} \left(x^T Q x + u^T R u \right) + \frac{\partial J^*}{\partial x} (Ax + Bu)$$

- To calculate $\min_u H$, take $\frac{\partial^T H}{\partial u} = 0$

$$\frac{\partial^T H}{\partial u} = Ru + B^T \left(\frac{\partial J^*}{\partial x} \right)^T = 0$$

$$\Rightarrow u^* = -R^{-1} B^T \left(\frac{\partial J^*}{\partial x} \right)^T$$

Finite Horizon Continuous-Time Linear Quadratic Regulator (3)

- We saw that $J^* = \frac{1}{2}x^T(k)S_k x(k)$ for DT... let's assume $J^* = \frac{1}{2}x^T(t)P(t)x(t)$. Substitute it to u^*

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

where $P \leq 0$

- How to find $P(t)$? Plug u^* back to the HJB equation!

$$\begin{aligned} 0 &= \frac{1}{2}x^T\dot{P}x + \frac{1}{2}x^T Q x + \frac{1}{2}\left(R^{-1}B^T P x\right)^T R R^{-1} B^T P x + x^T P (Ax - BR^{-1}B^T P x) \\ &= \frac{1}{2}x^T\dot{P}x + \frac{1}{2}x^T Q x - \frac{1}{2}x^T P B R^{-1} B^T P x + x^T P A x \end{aligned}$$

- $x^T P A x$ is a scalar. So

$$\begin{aligned} x^T P A x &= (x^T P A x)^T = x^T A^T P^T x = x^T A^T P x \\ &= \frac{1}{2}x^T\dot{P}x + \frac{1}{2}x^T Q x - \frac{1}{2}x^T P B R^{-1} B^T P x + \frac{1}{2}x^T P A x + \frac{1}{2}x^T A^T P x \\ &\Rightarrow x^T\left(\dot{P} + Q - PBR^{-1}B^T P + PA + A^T P\right)x = 0 \end{aligned}$$

Finite Horizon Continuous-Time Linear Quadratic Regulator (4)

- This must hold for all $x \Rightarrow$

$$\dot{P} = -Q + PBR^{-1}B^T P - PA - A^T P$$
$$u = -R^{-1}B^T Px$$

- Solve an ODE in P with boundary condition $P(t_f) = S$
- This is called differential Ricatti Equation
- Could certainly solve for $P(t)$ for a scalar system, but otherwise would resort to numerical solution $\Rrightarrow$ back to DT LQR

Recap: Finite Horizon Continuous-Time Linear Quadratic Regulator

$$\text{minimize } J = \frac{1}{2}x^T(t_f)\,Sx(t_f) \;+\; \frac{1}{2}\int_{t_0}^{t} x^T(t)\,Qx(t) \;+\; u^T(t)\,Ru(t)\,dt$$

subject to

$$\dot{x}(t) = Ax(t) + Bu(t)$$

The optimal control

$$u^*(t) = -R^{-1}B^TP(t)x(t)$$

where $P(t)$ is the solution of a Continuous-time Differential Riccati Equation

$$\dot{P}(t) = -Q + P(t)BR^{-1}B^TP(t) - P(t)A - A^TP(t)$$

with boundary condition $P(t_f) = S$.

Recap: Matrix Equations

Discrete-time Difference Riccati Equation	Continuous-time Differential Riccati Equation
$S_k = A^T S_{k+1} A - A^T S_{k+1} B$ $(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A + Q$	$\dot{P}(t) = -Q + P(t) B R^{-1} B^T P(t) - P(t) A$ $-A^T P(t)$

Discrete-time Algebraic Riccati Equation (DARE)	Continuous-time Algebraic Riccati Equation (CARE)
$S = A^T S A - A^T S B$ $(R + B^T S B)^{-1} B^T S A + Q$	$A^T P + P A - P B R^{-1} B^T P + Q = 0$

Table of Contents

Infinite Horizon Continuous-Time Linear Quadratic Regulator

Like in the discrete case, we make simplify the calculation of FH-CT-LQR as follows:

- extend the planning horizon to ∞
- remove the penalty for the final state
- use a constant control gain

$$J = \int_0^\infty x^T(t)Qx(t) + u^T(t)Ru(t)\ dt$$

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ u(t) &= Kx(t) \end{aligned}$$

Solving IH-CT-LQR

$$\dot{P}(t) = -Q + P(t)BR^{-1}B^T P(t) - P(t)A - A^T P(t)$$

- Because K is a constant, P must be a constant. $\dot{P} = 0$

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

- This is in the form of the solution to the Continuous-time Algebraic Riccati Equation (CARE). There are many reliable numerical solvers.
- The optimal control is then

$$K = -R^{-1}B^T P$$

Recap: Matrix Equations

Discrete-time Difference Riccati Equation	Continuous-time Differential Riccati Equation
$S_k = A^T S_{k+1} A - A^T S_{k+1} B$ $(R + B^T S_{k+1} B)^{-1} B^T S_{k+1} A + Q$	$\dot{P}(t) = -Q + P(t) B R^{-1} B^T P(t) - P(t) A$ $-A^T P(t)$

Discrete-time Algebraic Riccati Equation (DARE)	Continuous-time Algebraic Riccati Equation (CARE)
$S = A^T S A - A^T S B$ $(R + B^T S B)^{-1} B^T S A + Q$	$A^T P + PA - PBR^{-1} B^T P + Q = 0$