University of Pennsylvania, ESE 6190

Model Predictive Control

Chapter 5: Optimal Control Introduction and Unconstrained Linear Quadratic Control

Prof. Manfred Morari

Spring 2023

Coauthors: Prof. Colin Jones, EPFL
Prof. Francesco Borrelli, UC Berkeley

F. Borrelli, A. Bemporad, and M. Morari, Predictive Control for Linear and Hybrid Systems, Cambridge University Press, 2017. [Ch. 7-9].

Outline

Outline

Optimal Control Introduction (1/2)

- Discrete-time **optimal control** is concerned with choosing an optimal input sequence $U_{0\to N-1} := [u_0^\top, u_1^\top, \dots]^\top$ (as measured by some objective function), over a finite or infinite time horizon, in order to apply it to a system with a given initial state $x(0)$.
- The objective, or cost, function is often defined as a sum of **stage costs** $q(x_k, u_k)$ and, when the horizon has finite length N, a **terminal cost** $p(x_N)$:

$$J_{0\to N}(x_0, U_{0\to N-1}) := p(x_N) + \sum_{k=0}^{N-1} q(x_k, u_k)$$

- The states $\{x_k\}_{k=0}^N$ must satisfy the system dynamics

$$\begin{aligned} x_{k+1} &= g(x_k, u_k),\ k = 0, \dots, N-1 \\ x_0 &= x(0) \end{aligned}$$

and there may be state and/or input constraints

$$h(x_k, u_k) \le 0,\ k = 0, \dots, N-1.$$

Optimal Control Introduction (2/2)

- In the finite horizon case, there may also be a constraint that the final state x_N lies in a set $\mathcal{X}_f$
$$x_N \in \mathcal{X}_f$$
- A general finite horizon optimal control formulation for discrete-time systems is therefore

$$\begin{aligned} J^\star_{0\to N}(x(0)) := \min_{U_{0\to N-1}} \ & J_{0\to N}(x(0), U_{0\to N-1}) \\ \text{subj. to } & x_{k+1} = g(x_k, u_k),\ k = 0, \dots, N-1 \\ & h(x_k, u_k) \le 0,\ k = 0, \dots, N-1 \\ & x_N \in \mathcal{X}_f \\ & x_0 = x(0) \end{aligned}$$

General Problem Formulation (1/2)

Consider the nonlinear time-invariant system

$$x(t+1) = g(x(t), u(t))$$

subject to the constraints

$$h(x(t), u(t)) \leq 0, \forall t \geq 0$$

with $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$ the state and input vectors. Assume that $g(0,0) = 0, h(0,0) \leq 0$.
Consider the following *objective or cost function*

$$J_{0\to N}(x_0, U_{0\to N-1}) := p(x_N) + \sum_{k=0}^{N-1} q(x_k, u_k)$$

where

- N is the time *horizon*,
- $x_{k+1} = g(x_k, u_k), k = 0, \ldots, N-1$ and $x_0 = x(0)$,
- $U_{0\to N-1} := \left[u_0^\top, \ldots, u_{N-1}^\top\right]^\top \in \mathbb{R}^s, s = mN$,
- $q(x_k, u_k)$ and $p(x_N)$ are the *stage cost* and *terminal cost*, respectively.

General Problem Formulation (2/2)

Consider the **C**onstrained **F**inite **T**ime **O**ptimal **C**ontrol (CFTOC) problem.

$$
\begin{aligned}
J^\star_{0\to N}(x(0)) := \min_{U_{0\to N-1}} \ & J_{0\to N}(x(0), U_{0\to N-1}) \\
\text{subj. to } & x_{k+1} = g(x_k, u_k),\ k = 0, \dots, N-1 \\
& h(x_k, u_k) \le 0,\ k = 0, \dots, N-1 \\
& x_N \in \mathcal{X}_f \\
& x_0 = x(0)
\end{aligned}
$$

- $\mathcal{X}_f \subset \mathbb{R}^n$ is a *terminal* region.
- $\mathcal{X}_{0\to N} \subset \mathbb{R}^n$ is the set of feasible initial conditions $x(0)$.
- The optimal cost $J^\star_{0\to N}(x_0)$ is called *value function*.
- Assume that there exists a minimum.
- denote by $U^\star_{0\to N}$ one of the minima.

Objectives

- **Finite Time Solution**
  - a general nonlinear programming problem (*batch approach*)
  - recursively by invoking Bellman's Principle of Optimality (*recursive approach*)
  - discuss in details the linear system case

- **Infinite Time Solution**. We will investigate
  - if a solution exists as $N \to \infty$
  - the properties of this solution
  - approximate of the solution by using a *receding horizon technique*

- **Uncertainty**. We will discuss how to extend the problem description and consider uncertiainty.

Outline

Solution via Batch Approach (1/2)

Write the equality constraints from system constraints as

$$\begin{aligned}
x_1 &= g(x(0), u_0)\\
x_2 &= g(x_1, u_1)\\
&\vdots\\
x_N &= g(x_{N-1}, u_{N-1})
\end{aligned}$$

then the optimal control problem

$$\begin{aligned}
J^\star_{0\to N}(x(0)) := \min_{\substack{U_{0\to N-1}\\ x_1,\dots,x_N}} \ & p(x_N) + \sum_{k=0}^{N-1} q(x_k, u_k)\\
\text{subj. to } & x_1 = g(x(0), u_0)\\
& x_2 = g(x_1, u_1)\\
& \vdots\\
& x_N = g(x_{N-1}, u_{N-1})\\
& h(x_k, u_k) \le 0, k = 0, \dots, N-1\\
& x_N \in \mathcal{X}_f\\
& x_0 = x(0)
\end{aligned}$$

is a general Non-linear Programming (NLP) problem with variables $u_0, \dots, u_{N-1}$ and $x_1, \dots, x_N$.

Solution via Batch Approach (2/2)

- In this class we will “trust” a general-purpose nonlinear solver
- One could design more efficient solvers for tailored problem (will not do this)
- Sometimes still some work to do in writing the constraints “$h(x_k, u_k) \leq 0$” (think of an obstacle avoidance example)

Outline

Solution via Recursive Approach (1/6)

Theorem: Principle of optimality

For a trajectory $x_0, x_1^\star, \dots, x_N^\star$ to be optimal, the trajectory starting from any intermediate point $x_j^\star$, i.e. $x_j^\star, x_{j+1}^\star, \dots, x_N^\star, 0 \le j \le N-1$, must be optimal.

Suppose that the fastest route from Los Angeles to Boston passes through Chicago. The principle of optimality formalizes the obvious fact that the Chicago to Boston portion of the route is also the fastest route for a trip that starts from Chicago and ends in Boston.

Solution via Recursive Approach (2/6)

Theorem: Principle of optimality

For a trajectory $x_0, x_1^\star, \dots, x_N^\star$ to be optimal, the trajectory starting from any intermediate point $x_j^\star$, i.e. $x_j^\star, x_{j+1}^\star, \dots, x_N^\star, 0 \le j \le N-1$, must be optimal.

Define the cost from j to N

$$J_{j\to N}(x_j, u_j, u_{j+1}, \dots, u_{N-1}) := p(x_N) + \sum_{k=j}^{N-1} q(x_k, u_k)$$

also called the j-step cost-to-go. Then the optimal cost-to-go $J_{j\to N}^\star$ is

$$\begin{aligned} J_{j\to N}^\star(x_j) = \min_{u_j, u_{j+1}, \dots, u_{N-1}} \quad & J_{j\to N}(x_j, u_j, u_{j+1}, \dots, u_{N-1}) \\ \text{subj. to} \quad & x_{k+1} = g(x_k, u_k) \\ & h(x_k, u_k) \le 0, k = 0, \dots, N-1 \\ & x_N \in \mathcal{X}_f \end{aligned}$$

Note that $J_{j\to N}^\star(x_j)$ depends only on the initial state x_j.

Solution via Recursive Approach (3/6)

By the **principle of optimality** the cost $J^\star_{j-1\to N}$ can be found by solving

$$\begin{aligned} J^\star_{j-1\to N}(x_{j-1}) = \min_{u_{j-1}} \ & \overbrace{q(x_{j-1}, u_{j-1})}^{\text{stage cost}} + \overbrace{J^\star_{j\to N}(x_j)}^{\text{optimal cost-to-go}} \\ \text{subj. to } & x_j = g(x_{j-1}, u_{j-1}) \\ & h(x_{j-1}, u_{j-1}) \le 0 \\ & x_j \in \mathcal{X}_{j\to N} \end{aligned} \qquad (\star)$$

Note that

- the only decision variable is u_{j-1}
- the inputs $u^\star_j, \ldots, u^\star_{N-1}$ have already been selected optimally to yield the optimal cost-to-go $J^\star_{j\to N}(x_j)$
- in $J^\star_{j\to N}(x_j)$, the states x_j can be replaced by $g(x_{j-1}, u_{j-1})$
- the set $\mathcal{X}_{j\to N}$ is the set of states x_j for which $(\star)$ is feasible. We will study these sets later in the class.

Solution via Recursive Approach (4/6)

The following (recursive) **dynamic programming** algorithm can be used to compute the optimal control law.

$$\begin{aligned} J^\star_{N\to N} &= p(x_N) \\ \mathcal{X}_{N\to N} &= \mathcal{X}_f \end{aligned}$$

$$\begin{aligned} J^\star_{N-1\to N}(X_{N-1}) \quad &= \min_{u_{N-1}} && q(x_{N-1}, u_{N-1}) + J^\star_{N\to N}(g(x_{N-1}, u_{N-1})) \\ & \text{Subj. to} && h(x_{N-1}, u_{N-1}) \le 0 \\ & && g(x_{N-1}, u_{N-1}) \in \mathcal{X}_{N\to N} \\ &\vdots \\ J^\star_{0\to N}(x_0) \quad &= \min_{u_0} && q(x_0, u_0) + J^\star_{1\to N}(g(x_0, u_0)) \\ & \text{Subj. to} && h(x_0, u_0) \le 0 \\ & && g(x_0, u_0) \in \mathcal{X}_{1\to N} \\ & && x_0 = x(0) \end{aligned}$$

Solution via Recursive Approach (5/6)

Comments

- DP algorithm is appealing because at each step j only u_j is computed.
- Need to construct the optimal cost-to-go $J^\star_{N-j}(x_j)$, a function defined over $\mathcal{X}_{j\to N}$.
- In a few special cases we know the type of function and we can find it efficiently.
- "brute force" approach. Construct $J_{j-1\to N}$ by griding the set $\mathcal{X}_{j-1\to N}$ and computing the optimal cost-to-go function on each grid point.
- A nonlinear feedback (time varying) control law is implicitly defined:

$$\begin{aligned} u_j^\star(x_j) = \underset{u_j}{\text{argmin}} \; & q(x_j, u_j) + J^\star_{j+1\to N}(g(x_j, u_j)) \\ \text{subj. to } & h(x_j, u_j) \leq 0 \\ & g(x_j, u_j) \in \mathcal{X}_{j+1\to N} \end{aligned}$$

Solution via Recursive Approach (6/6)

For the sake of simplicity we will use the following shorter notation

$$\begin{aligned} J_j^\star(x_j) &:= J_{j\to N}^\star(x_j), j = 0, \dots, N \\ J_\infty^\star(x_0) &:= J_{0\to\infty}^\star(x_j) \\ \mathcal{X}_j &:= \mathcal{X}_{j\to N}, j = 0, \dots, N \\ \mathcal{X}_\infty &:= \mathcal{X}_{0\to\infty} \\ U_0 &:= U_{0\to N-1} \end{aligned}$$

Assume that a solution exists for $N \to \infty$, then we obtain the Bellman Equation:

$$\begin{aligned} J_\infty^\star(x) = \min_u \; & q(x,u) + J_\infty^\star(g(x,u)) \\ \text{subj. to } & h(x,u) \le 0 \\ & g(x,u) \in \mathcal{X}_\infty \end{aligned}$$

Outline

Linear Quadratic Optimal Control

- In this section, only **linear** discrete-time time-invariant systems

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

  and **quadratic** cost functions

$$J_0(x_0, U) := x_N^\top P x_N + \sum_{k=0}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k) \quad (1)$$

  are considered, and we consider only the problem of regulating the state to the origin, **without state or input constraints**.
- The two most common solution approaches will be described here
  1. **Batch Approach**, which yields a series of **numerical values** for the input
  2. **Recursive Approach**, which uses Dynamic Programming to compute control **policies** or **laws**, i.e. functions that describe how the control decisions depend on the system states.

Unconstrained Finite Horizon Control Problem

- **Goal:** Find a sequence of inputs $U_{0\to N-1} := [u_0^\top, \dots, u_{N-1}^\top]^\top$ that minimizes the objective function

$$
\begin{aligned}
J_0^\star(x(0)) := \min_{U_{0\to N-1}} \; & x_N^\top P x_N + \sum_{k=0}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k) \\
\text{subj. to } & x_{k+1} = Ax_k + Bu_k,\ k = 0, \dots, N-1 \\
& x_0 = x(0)
\end{aligned}
$$

- $P \succeq 0$, with $P = P^\top$, is the **terminal** weight
- $Q \succeq 0$, with $Q = Q^\top$, is the **state** weight
- $R \succ 0$, with $R = R^\top$, is the **input** weight
- N is the horizon length
- Note that $x(0)$ is the current state, whereas $x_0, \dots, x_N$ and $u_0, \dots, u_{N-1}$ are **optimization variables** that are constrained to obey the system dynamics and the initial condition.

Outline

2. Finite Horizon

    Batch Approach

    Recursive Approach

Batch Approach

Final Result

- The problem is unconstrained.
- Setting the gradient to zero:

$$U_0^\star(x(0)) = \mathbf{K}x(0)$$

- which implies

$$u^\star(0)(x(0)) = K_0 x(0), \dots, u^\star(N-1)(x(0)) = K_{N-1}x(0)$$

  which is a linear, open-loop controller function of the initial state $x(0)$.
- The optimal cost is
$$J_0^\star(x(0)) = x^\top(0) P_0 x(0)$$
  which is a positive definite quadratic function of the initial state $x(0)$.

Solution approach 1: Batch Approach (1/4)

- The batch solution explicitly represents all future states x_k in terms of initial condition x_0 and inputs $u_0, \dots, u_{N-1}$.
- Starting with $x_0 = x(0)$, we have $x_1 = Ax(0) + Bu_0$, and $x_2 = Ax_1 + Bu_1 = A^2x(0) + ABu_0 + Bu_1$, by substitution for x_1, and so on. Continuing up to x_N we obtain:

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} I \\ A \\ \vdots \\ \vdots \\ A^N \end{bmatrix} x(0) + \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \ddots & \ddots & 0 \\ A^{N-1}B & \cdots & AB & B \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ \vdots \\ u_{N-1} \end{bmatrix}$$

- The equation above can be represented as

$$\mathcal{X} := \mathcal{S}^x x(0) + \mathcal{S}^u U\,. \tag{2}$$

Solution approach 1: Batch Approach (2/4)

- Define

$$\overline{Q} := \text{blockdiag}(Q, \dots, Q, P) \quad \text{and} \quad \overline{R} := \text{blockdiag}(R, \dots, R)$$

  Then the finite horizon cost function (1) can be written as

$$J_0(x(0), U) = \mathcal{X}^\top \overline{Q} \mathcal{X} + U^\top \overline{R} U\,. \tag{3}$$

- Eliminating $\mathcal{X}$ by substituting from (2), equation (3) can be expressed as:

$$\begin{aligned} J_0(x(0), U) &= (\mathcal{S}^x x(0) + \mathcal{S}^u U)^\top \overline{Q} (\mathcal{S}^x x(0) + \mathcal{S}^u U) + U^\top \overline{R} U \\ &= U^\top H U + 2x(0)^\top F U + x(0)^\top \mathcal{S}^{x^\top} \overline{Q} \mathcal{S}^x x(0) \end{aligned}$$

  where $H := (\mathcal{S}^u)^\top \overline{Q} \mathcal{S}^u + \overline{R}$ and $F := (\mathcal{S}^x)^\top \overline{Q} \mathcal{S}^u$.
- Note that $H \succ 0$, since $R \succ 0$ and $(\mathcal{S}^u)^\top \overline{Q} \mathcal{S}^u \succeq 0$.

Solution approach 1: Batch Approach (3/4)

- Since the problem is unconstrained and $J_0(x(0), U)$ is a positive definite quadratic function of U we can solve for the optimal input $U^\star$ by setting the gradient with respect to U to zero:

$$\begin{aligned}\nabla_U J_0(x(0), U) &= 2HU + 2F^\top x(0) = 0 \\ \Rightarrow U^\star(x(0)) &= -H^{-1}F^\top x(0) \\ &= -\left((\mathcal{S}^u)^\top \overline{Q}\mathcal{S}^u + \overline{R}\right)^{-1} (\mathcal{S}^u)^\top \overline{Q}\mathcal{S}^x x(0)\,,\end{aligned}$$

  which is a linear function of the initial state $x(0)$.
  Note H^{-1} always exists, since $H \succ 0$ and therefore has full rank.
- The optimal cost can be shown (by back-substitution) to be

$$\begin{aligned}J_0^\star(x(0)) &= -x(0)^\top FHF^\top x(0) + x(0)^\top (\mathcal{S}^x)^\top \overline{Q}\mathcal{S}^x x(0) \\ &= x(0)^\top \left((\mathcal{S}^x)^\top \overline{Q}\mathcal{S}^x - (\mathcal{S}^x)^\top \overline{Q}\mathcal{S}^u((\mathcal{S}^u)^\top \overline{Q}\mathcal{S}^u + \overline{R})^{-1}(\mathcal{S}^u)^\top \overline{Q}\mathcal{S}^x\right) x(0)\end{aligned}$$

Solution approach 1: Batch Approach (4/4)

Summary

- The Batch Approach expresses the cost function in terms of the initial state $x(0)$ and input sequence U by eliminating the states x_k.
- Because the cost $J_0(x(0), U)$ is a strictly convex quadratic function of U, its minimizer $U^\star$ is unique and can be found by setting $\nabla_U J_0(x(0), U) = 0$. This gives the optimal input sequence $U^\star$ as a linear function of the intial state $x(0)$:

$$U^\star(x(0)) = -\left((\mathcal{S}^u)^\top \overline{Q} \mathcal{S}^u + \overline{R}\right)^{-1} (\mathcal{S}^u)^\top \overline{Q} \mathcal{S}^x x(0)$$

- The optimal cost is a quadratic function of the initial state $x(0)$

$$J_0^\star(x(0)) = x(0)^\top \left((\mathcal{S}^x)^\top \overline{Q} \mathcal{S}^x - (\mathcal{S}^x)^\top \overline{Q} \mathcal{S}^u \left((\mathcal{S}^u)^\top \overline{Q} \mathcal{S}^u + \overline{R}\right)^{-1} (\mathcal{S}^u)^\top \overline{Q} \mathcal{S}^x \right) x(0)$$

- If there are state or input constraints, solving this problem by matrix inversion is not guaranteed to result in a feasible input sequence

Outline

2. Finite Horizon

    Batch Approach

    Recursive Approach

Recursive Approach

Final Result

- The problem is unconstrained
- Using the Dynamic Programming Algorithm we have,

$$u^\star(k) = F_k x(k)$$

  which is a linear, time-varying state-feedback controller.
- the optimal cost-to-go $k \to N$ is

$$J_k^\star(x(k)) = x^\top(k) P_k x(k)$$

  which is a positive definite quadratic function of the state at time k.
- F_k is computed by using P_{k+1}
- Each P_k is related to P_{k+1} by a recursive equation (Riccati Difference Equation)

Solution approach 2: Recursive Approach (1/8)

- Alternatively, we can use dynamic programming to solve the same problem in a recursive manner.
- Define the "j-step optimal cost-to-go" as the **optimal** cost attainable for the step j problem:

$$
\begin{aligned}
J_j^\star(x(j)) := \min_{u_{j\to N-1}} \; & x_N^\top P x_N + \sum_{k=j}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k) \\
\text{subj. to } & x_{k+1} = A x_k + B u_k,\; k = j, \dots, N-1 \\
& x_j = x(j)
\end{aligned}
$$

This is the minimum cost attainable for the remainder of the horizon after step j.

Solution approach 2: Recursive Approach (2/8)

- Consider the 1-step problem (solved at time $N-1$)

$$J^\star_{N-1}(x_{N-1}) = \min_{u_{N-1}} x_{N-1}^\top Q x_{N-1} + u_{N-1}^\top R u_{N-1} + x_N^\top P_N x_N \quad (4)$$

$$\text{s.t. } x_N = A x_{N-1} + B u_{N-1} \quad (5)$$

$$P_N = P$$

  where we introduced the notation P_j to express the optimal cost-to-go $x_j^\top P_j x_j$. In particular, $P_N = P$.

- Substituting (5) into (4)

$$\begin{aligned} J^\star_{N-1}(x_{N-1}) = \min_{u_{N-1}} \{ & x_{N-1}^\top (A^\top P_N A + Q) x_{N-1} \\ & + u_{N-1}^\top (B^\top P_N B + R) u_{N-1} \\ & + 2 x_{N-1}^\top A^\top P_N B u_{N-1} \} \end{aligned}$$

Solution approach 2: Recursive Approach (3/8)

- Solving again by setting the gradient to zero leads to the following optimality condition for u_{N-1}

$$2(B^\top P_N B + R)u_{N-1} + 2B^\top P_N A x_{N-1} = 0$$

Optimal 1-step input:

$$\begin{aligned} u^\star_{N-1} &= -(B^\top P_N B + R)^{-1} B^\top P_N A x_{N-1} \\ &:= F_{N-1} x_{N-1} \end{aligned}$$

1-step cost-to-go:

$$J^\star_{N-1}(x_{N-1}) = x_{N-1}^\top P_{N-1} x_{N-1}\,,$$

where

$$P_{N-1} = A^\top P_N A + Q - A^\top P_N B (B^\top P_N B + R)^{-1} B^\top P_N A\,.$$

Solution approach 2: Recursive Approach (4/8)

- The recursive solution method used from here relies on Bellman's **Principle of Optimality**
- For any solution for steps 0 to N to be optimal, any solution for steps j to N with $j \geq 0$, taken from the 0 to N solution, must itself be optimal for the j-to-N problem
- Therefore we have, for any $j = 0, \dots, N$

$$J_j^\star(x_j) = \min_{u_j} \; J_{j+1}^\star(x_{j+1}) + x_j^\top Q x_j + u_j^\top R u_j$$
$$\text{subj. to } x_{j+1} = Ax_j + Bu_j$$

- Interpretation:
  Suppose that the fastest route from Los Angeles to Boston passes through Chicago. Then the principle of optimality formalizes the obvious fact that the Chicago to Boston portion of the route is also the fastest route for a trip that starts from Chicago and ends in Boston.

Solution approach 2: Recursive Approach (5/8)

- Now consider the 2-step problem, posed at time $N-2$

$$J^\star_{N-2}(x_{N-2}) = \min_{u_{N-1},u_{N-2}} \sum_{k=N-2}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + x_N^\top P x_N$$
$$\text{subj. to } x_{k+1} = Ax_k + Bu_k, \quad k = N-2, N-1$$

- From the Principle of Optimality, the cost function is equivalent to

$$\begin{aligned} J^\star_{N-2}(x_{N-2}) &= \min_{u_{N-2}} \; x_{N-2}^\top Q x_{N-2} + u_{N-2}^\top R u_{N-2} + J^\star_{N-1}(x_{N-1}) \\ &= \min_{u_{N-2}} \; x_{N-2}^\top Q x_{N-2} + u_{N-2}^\top R u_{N-2} + x_{N-1}^\top P_{N-1} x_{N-1} \end{aligned}$$

Solution approach 2: Recursive Approach (6/8)

- As with 1-step solution, solve by setting the gradient with respect to u_{N-2} to zero

  Optimal 2-step input

  $$u^\star_{N-2} = -(B^\top P_{N-1} B + R)^{-1} B^\top P_{N-1} A x_{N-2} \; := F_{N-2} x_{N-2}$$

  2-step cost-to-go

  $$J^\star_{N-2}(x_{N-2}) = x_{N-2}^\top P_{N-2} x_{N-2}\,,$$

  where

  $$P_{N-2} = A^\top P_{N-1} A + Q - A^\top P_{N-1} B (B^\top P_{N-1} B + R)^{-1} B^\top P_{N-1} A$$

- We now recognize the recursion for P_j and $u^\star_j$, $j = N-1, \cdots, 0$.

Solution approach 2: Recursive Approach (7/8)

- We can obtain the solution for any given time step k in the horizon

$$\begin{aligned} u_k^\star &= -(B^\top P_{k+1} B + R)^{-1} B^\top P_{k+1} A x(k) \\ &:= F_k x_k \quad \text{for } k = 1, \dots, N \end{aligned}$$

  where we can find any P_k by recursive evaluation from $P_N = P$, using

$$P_k = A^\top P_{k+1} A + Q - A^\top P_{k+1} B (B^\top P_{k+1} B + R)^{-1} B^\top P_{k+1} A \tag{6}$$

  This is called the **Discrete Time Riccati equation** or **Riccati Difference equation (RDE)**.

- Evaluating down to P_0, we obtain the N-step cost-to-go

$$J_0^\star(x(0)) = x(0)^\top P_0 x(0)$$

Solution approach 2: Recursive Approach (8/8)

Summary

- From the Principle of Optimality, the optimal control policy for any step k is then given by

$$u_k^\star = -(B^\top P_{k+1} B + R)^{-1} B^\top P_{k+1} A x_k = F_k x_k$$

  and the optimal cost-to-go is

$$J_k^\star(x_k) = x_k^\top P_k x_k$$

- Each P_k is related to P_{k+1} by the Riccati Difference equation

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

  which can be initialized with $P_N = P$, the given terminal weight

Comparison of Batch and Recursive Approaches (1/2)

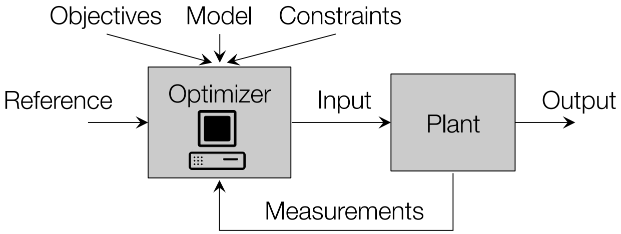
- Fundamental difference: Batch optimization returns a sequence $U^\star(x(0))$ of **numeric values** depending only on the initial state $x(0)$, while dynamic programming yields **feedback policies** $u_k^\star = F_k x_k$, $k = 0, \dots, N-1$ depending on each x_k.
- If the state evolves exactly as modelled, then the sequences of control actions obtained from the two approaches are identical.
- The recursive solution should be more robust to disturbances and model errors, because if the future states later deviate from their predicted values, the exact optimal input can still be computed.
- The Recursive Approach is computationally more attractive because it breaks the problem down into single-step problems. For large horizon length, the Hessian H in the Batch Approach, which must be inverted, becomes very large.

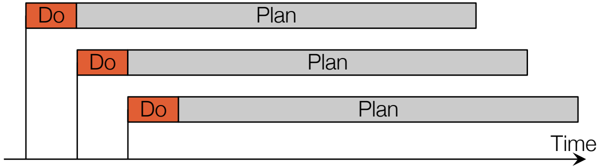
Comparison of Batch and Recursive Approaches (2/2)

- Without any modification, both solution methods will break down when inequality constraints on x_k or u_k are added.
- The Batch Approach is far easier to adapt than the Recursive Approach when constraints are present: just perform a constrained minimization for the current state.
- Doing this at **every** time step within the time available, and then using only the first input from the resulting sequence, amounts to **receding horizon control**.

Outline

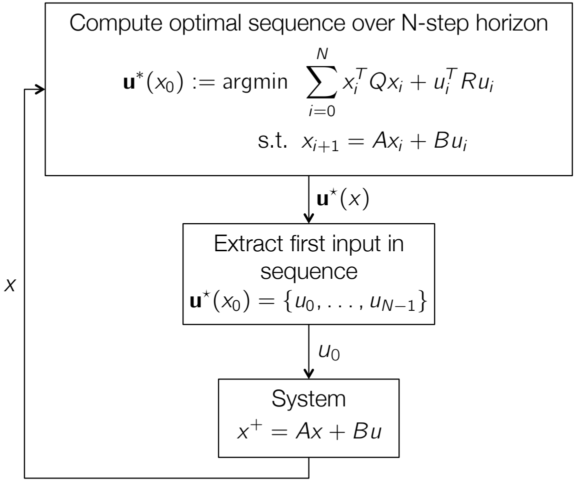
Receding horizon control





Receding horizon strategy introduces feedback.

Receding Horizon Control


Compute optimal sequence over N-step horizon
$\mathbf{u}^*(x_0) := \text{argmin} \sum_{i=0}^{N} x_i^T Q x_i + u_i^T R u_i$
s.t. $x_{i+1} = Ax_i + Bu_i$
$\mathbf{u}^\star(x)$
Extract first input in sequence
$\mathbf{u}^\star(x_0) = \{u_0, \ldots, u_{N-1}\}$
u_0
System
$x^+ = Ax + Bu$
x

For unconstrained systems, this is a **constant linear controller**
However, can extend this concept to much more complex systems (MPC)

Example - Impact of Horizon Length

Consider the lightly damped, stable system

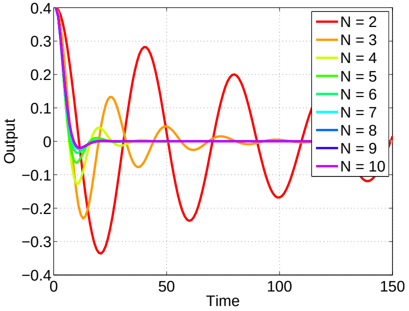
$$G(s) := \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2}$$

where $\omega = 1$, $\zeta = 0.01$. We sample at 10Hz and set $P = Q = I$, $R = 1$.

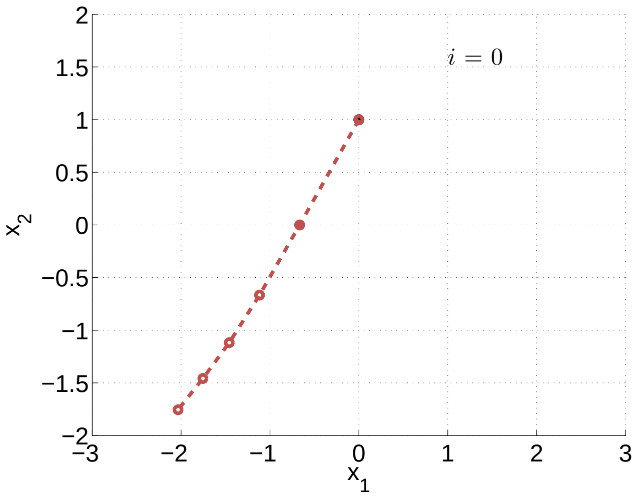
Discrete-time state-space model:

$$x^+ = \begin{bmatrix} 1.988 & -0.998 \\ 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0.125 \\ 0 \end{bmatrix} u$$

Closed-loop response

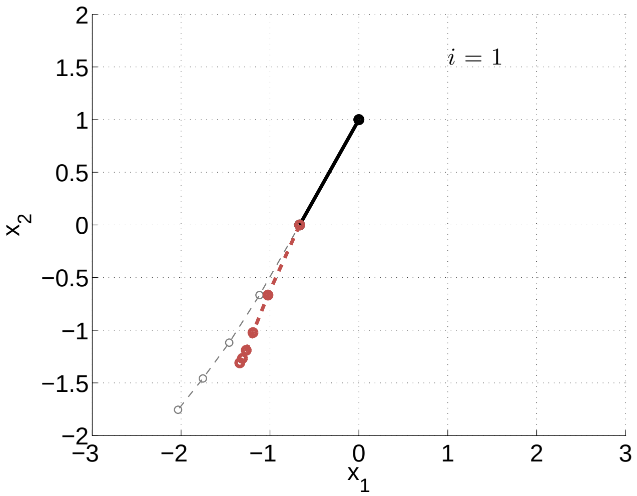


Example: Short horizon $N = 5$



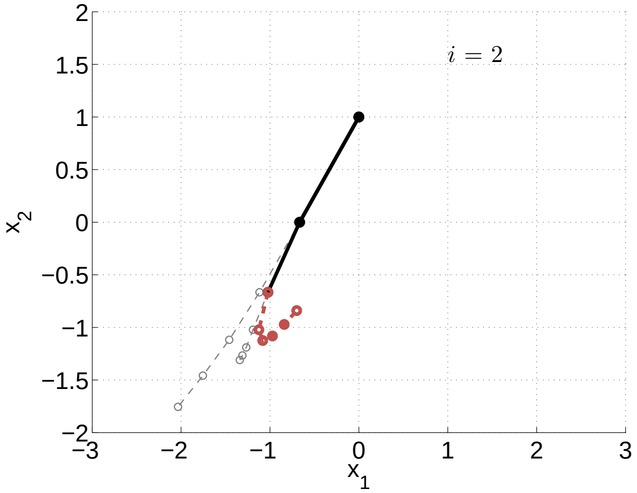
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



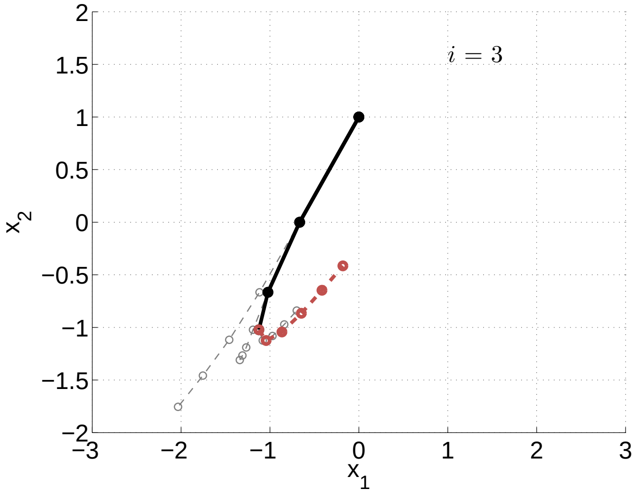
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



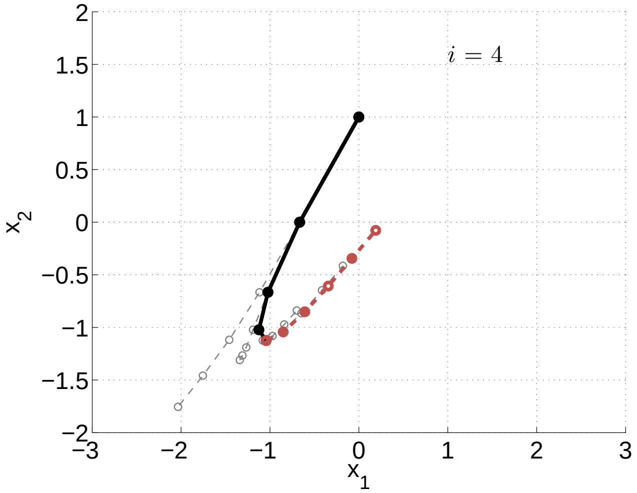
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



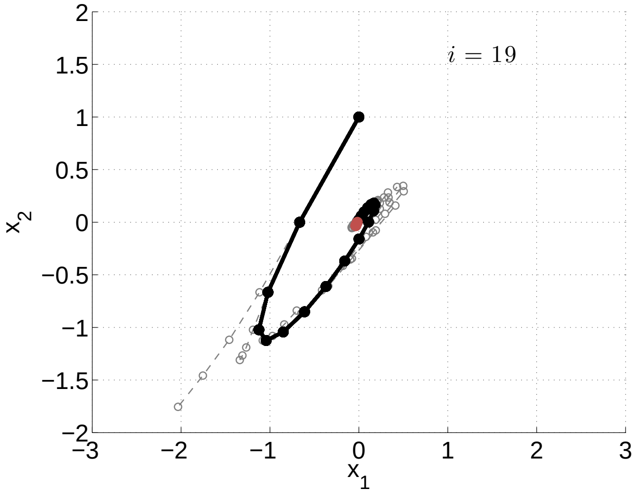
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



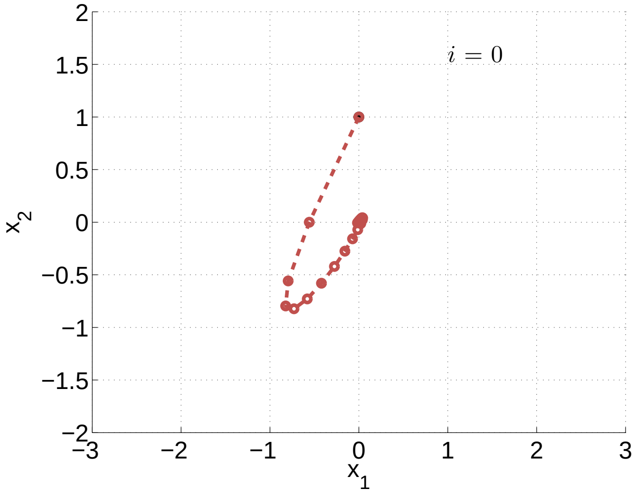
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



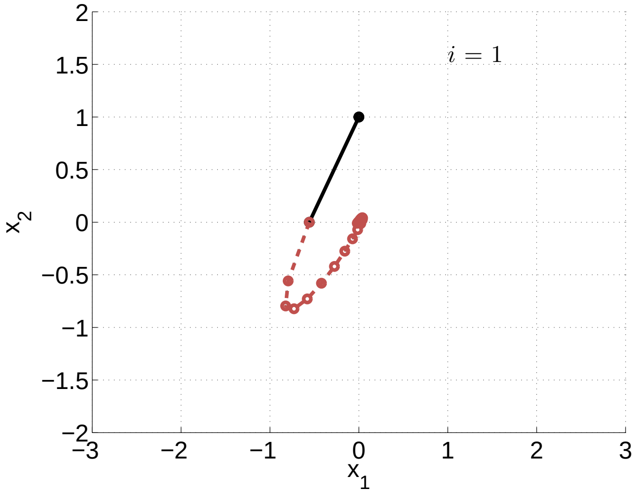
Short horizon: Prediction and closed-loop response differ.

Example: Long horizon $N = 20$



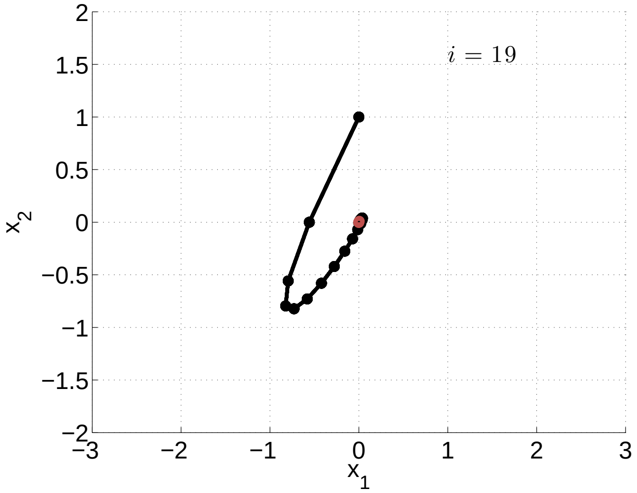
Long horizon: Prediction and closed-loop match.

Example: Long horizon $N = 20$



Long horizon: Prediction and closed-loop match.

Example: Long horizon $N = 20$



Long horizon: Prediction and closed-loop match.

Stability of Finite-Horizon Optimal Control Laws

Consider the system

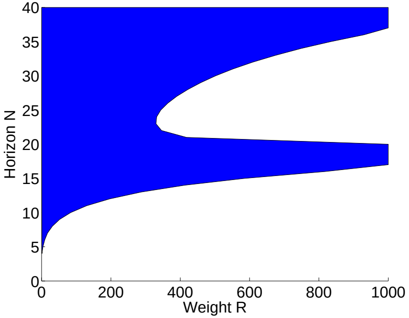
$$G(s) = \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2}$$

where $\omega = 0.1$ and $\zeta = -1$, which has been discretized at $1r/s$.
(Note that this system is unstable)

Is the system $x^+ = (A + BK_{R,N})x$ stable?

Where $K_{R,N}$ is the finite horizon LQR controller with horizon N and weight R (Q taken to be the identity)

Blue = stable, white = unstable



Outline

Infinite Horizon Control Problem: Optimal Solution (1/2)

- In some cases we may want to solve the same problem with an infinite horizon:

$$
\begin{aligned}
J_\infty(x(0)) = \min_{u(\cdot)} \sum_{k=0}^{\infty} &\left(x_k^\top Q x_k + u_k^\top R u_k\right) \\
\text{subj. to } & x_{k+1} = Ax_k + Bu_k, \quad k = 0, 1, 2, \dots, \infty\,, \\
& x_0 = x(0)
\end{aligned}
$$

- As with the Dynamic Programming approach, the optimal input is of the form

$$u^\star(k) = -(B^\top P_\infty B + R)^{-1} B^\top P_\infty A x(k) := F_\infty x(k)$$

and the infinite-horizon cost-to-go is

$$J_\infty(x(k)) = x(k)^\top P_\infty x(k)\,.$$

Infinite Horizon Control Problem: Optimal Solution (2/2)

- The matrix P_∞ comes from an infinite recursion of the RDE, from a notional point infinitely far into the future.
- Assuming the RDE does converge to some constant matrix P_∞, it must satisfy the following (from (6), with $P_k = P_{k+1} = P_\infty$)

$$P_\infty = A^\top P_\infty A + Q - A^\top P_\infty B(B^\top P_\infty B + R)^{-1} B^\top P_\infty A\,,$$

  which is called the **Algebraic Riccati equation (ARE)**.
- The constant feedback matrix F_∞ is referred to as the asymptotic form of the **Linear Quadratic Regulator (LQR)**.
- In fact, if (A, B) is stabilizable and $(Q^{1/2}, A)$ is detectable, then the RDE (initialized with Q at $k = \infty$ and solved for $k \searrow 0$) converges to the unique positive definite solution P_∞ of the ARE.

Stability of Infinite-Horizon LQR

- In addition, the closed-loop system with $u(k) = F_\infty x(k)$ is guaranteed to be asymptotically stable, under the stabilizability and detectability assumptions of the previous slide.
- The latter statement can be proven by substituting the control law $u(k) = F_\infty x(k)$ into $x(k+1) = Ax(k) + Bu(k)$, and then examining the properties of the system

$$x(k+1) = (A + BF_\infty)x(k)\,. \tag{7}$$

- The asymptotic stability of (7) can be proven by showing that the infinite horizon cost $J_\infty^\star(x(k)) = x(k)^\top P_\infty x(k)$ is actually a Lyapunov function for the system, i.e. $J_\infty^\star(x(k)) > 0$, $\forall k \neq 0$, $J_\infty^\star(0) = 0$, and $J_\infty^\star(x(k+1)) < J_\infty^\star(x(k))$, for any $x(k)$. This implies that

$$\lim_{k\to\infty} x(k) = 0\,.$$

Choices of Terminal Weight P in Finite Horizon Control (1/2)

1. The terminal cost P of the finite horizon problem can in fact trivially be chosen so that its solution matches the infinite horizon solution
   - To do this, make P equal to the optimal cost from N to ∞ (i.e. the cost with the optimal controller choice). This can be computed from the ARE:

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

   - This approach rests on the assumption that no constraints will be active after the end of the horizon.

Choices of Terminal Weight P in Finite Horizon Control (2/2)

2. Choose P assuming no control action after the end of the horizon, so that

$$x(k+1) = Ax(k),\ k = N, \dots, \infty$$

   - This P can be determined from solving the Lyapunov equation

$$A^\top PA + Q = P\,.$$

   - This approach only makes sense if the system is asymptotically stable (or no positive definite solution P will exist).

3. Assume we want the state and input both to be zero after the end of the finite horizon. In this case no P but an extra constraint is needed

$$x_N = 0$$