# Model Predictive Control

# Chapter 2: Unconstrained Linear Quadratic Optimal Control

Prof. Melanie Zeilinger

ETH Zurich

Spring 2020

Coauthors: Prof. Manfred Morari, University of Pennsylvania
Prof. Colin Jones, EPFL

# Learning Objectives – Lecture 2

- Learn to compute finite horizon unconstrained linear quadratic optimal controller in two ways
- Understand principle of optimality
- Learn to compute infinite horizon unconstrained linear quadratic optimal controller
- Understand impact of horizon length
- Prove stability of infinite horizon unconstrained linear quadratic optimal control
- Learn how to 'simulate' quasi-infinite horizon

# Outline

1. Introduction

2. Receding Horizon

3. Infinite Horizon LQR

# Outline

# Optimal Control Introduction (1/2)

- Discrete-time **optimal control** is concerned with choosing an optimal input sequence $U := [u_0^\top, u_1^\top, \ldots]^\top$ (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** $l(x_i, u_i)$ and, when the horizon has finite length $N$, a **terminal cost** $l_f(x_N)$:

$$J(x_0, U) := l_f(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i)$$

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

$$x_{i+1} = g(x_i, u_i),\ i = 0, \ldots, N-1$$
$$x_0 = x(0)$$

and there may be state and/or input constraints

$$h(x_i, u_i) \leq 0,\ i = 0, \ldots, N-1.$$

# 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

$$\begin{aligned} J^\star(x(0)) := \min_U \; & J(x(0), U) \\ \text{subj. to } & x_{i+1} = g(x_i, u_i),\ i = 0, \ldots, N-1 \\ & h(x_i, u_i) \leq 0,\ i = 0, \ldots, N-1 \\ & x_N \in \mathcal{X}_f \\ & x_0 = x(0) \end{aligned}$$

# 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

$$J(x_0, U) := x_N^\top P x_N + \sum_{i=0}^{N-1} (x_i^\top Q x_i + u_i^\top R u_i) \tag{1}$$

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 $U := [u_0^\top, \ldots, u_{N-1}^\top]^\top$ that minimizes the objective function

$$J^\star(x(0)) := \min_U \; x_N^\top P x_N + \sum_{i=0}^{N-1} (x_i^\top Q x_i + u_i^\top R u_i)$$

$$\text{subj. to } x_{i+1} = Ax_i + Bu_i,\; i = 0, \ldots, N-1$$

$$x_0 = x(0)$$

- $P \succeq 0$, 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 $x_0, \ldots, x_N$ and $u_0, \ldots, u_{N-1}$ are **optimization variables** that are constrained to obey the system dynamics and the initial condition.

# Outline

# Batch Approach (1/4)

- The batch solution explicitly represents all future states $x_i$ in terms of initial condition $x_0$ and inputs $u_0, \ldots, u_{N-1}$.
- 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

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

# Batch Approach (2/4)

- Define

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

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

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

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

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

  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 $(\mathcal{S}^u)^\top \overline{Q} \mathcal{S}^u \succeq 0$.

# Batch Approach (3/4)

- Since the problem is unconstrained and $J(x(0), U)$ 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:

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

  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

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

# 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 $x_i$.
- Because the cost $J(x(0), U)$ is a positive definite quadratic function of $U$, its minimizer $U^\star$ is unique and can be found by setting $\nabla_U J(x(0), U) = 0$. This gives the optimal input sequence $U^\star$ as a linear function of the initial 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 $x(0)$

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

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

# Outline

# 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:

$$J_j^\star(x(j)) := \min_{U_{j\to N}} x_N^\top P x_N + \sum_{i=j}^{N-1} (x_i^\top Q x_i + u_i^\top R u_i)$$

$$\text{subj. to } x_{i+1} = Ax_i + Bu_i,\ i = j, \ldots, N-1$$

$$x_j = x(j)$$

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

# Recursive Approach (2/8)

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

$$J^{\star}_{N-1}(x_{N-1}) = \min_{u_{N-1}} x^{\top}_{N-1} Q x_{N-1} + u^{\top}_{N-1} R u_{N-1} + x^{\top}_{N} P_N x_N \tag{4}$$

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

$$P_N = P$$

  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)

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

# 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:**

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

where

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

# Recursive Approach (4/8)

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

$$J^{\star}_{N-2}(x_{N-2}) = \min_{u_{N-1},u_{N-2}} \sum_{i=N-2}^{N-1} \left(x_i^{\top} Q x_i + u_i^{\top} R u_i\right) + x_N^{\top} P x_N$$
$$\text{subj. to } x_{i+1} = Ax_i + Bu_i, \quad i = N-2, N-1$$

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

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

# Recursive Approach (5/8)

- 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**

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

  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 $u^\star_j$, $j = N-1, \cdots, 0$.

# Recursive Approach (6/8)

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

$$\begin{aligned} u_i^\star &= -(B^\top P_{i+1} B + R)^{-1} B^\top P_{i+1} A x(i) \\ &:= F_i x_i \quad \text{for } i = 1, \ldots, N \end{aligned}$$

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

$$P_i = A^\top P_{i+1} A + Q - A^\top P_{i+1} B (B^\top P_{i+1} B + R)^{-1} B^\top P_{i+1} A \qquad (6)$$

  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

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

# Recursive Approach (7/8)

- 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 $j = 0, \ldots, N$

$$J_j^\star(x_j) = \min_{u_j} \; l(x_i, u_i) + J_{j+1}^\star(x_{j+1})$$

$$\text{subj. to } x_{j+1} = Ax_j + Bu_j$$

- 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.*

# Recursive Approach (8/8)

## Summary

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

$$u_i^\star = -(B^\top P_{i+1} B + R)^{-1} B^\top P_{i+1} A x_i = F_i x_i$$

  and the optimal cost-to-go is

$$J_i^\star(x_i) = x_i^\top P_i x_i$$

- Each $P_i$ is related to $P_{i+1}$ by the Riccati Difference equation

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

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

# Comparison of Batch and Recursive Approaches (1/2)

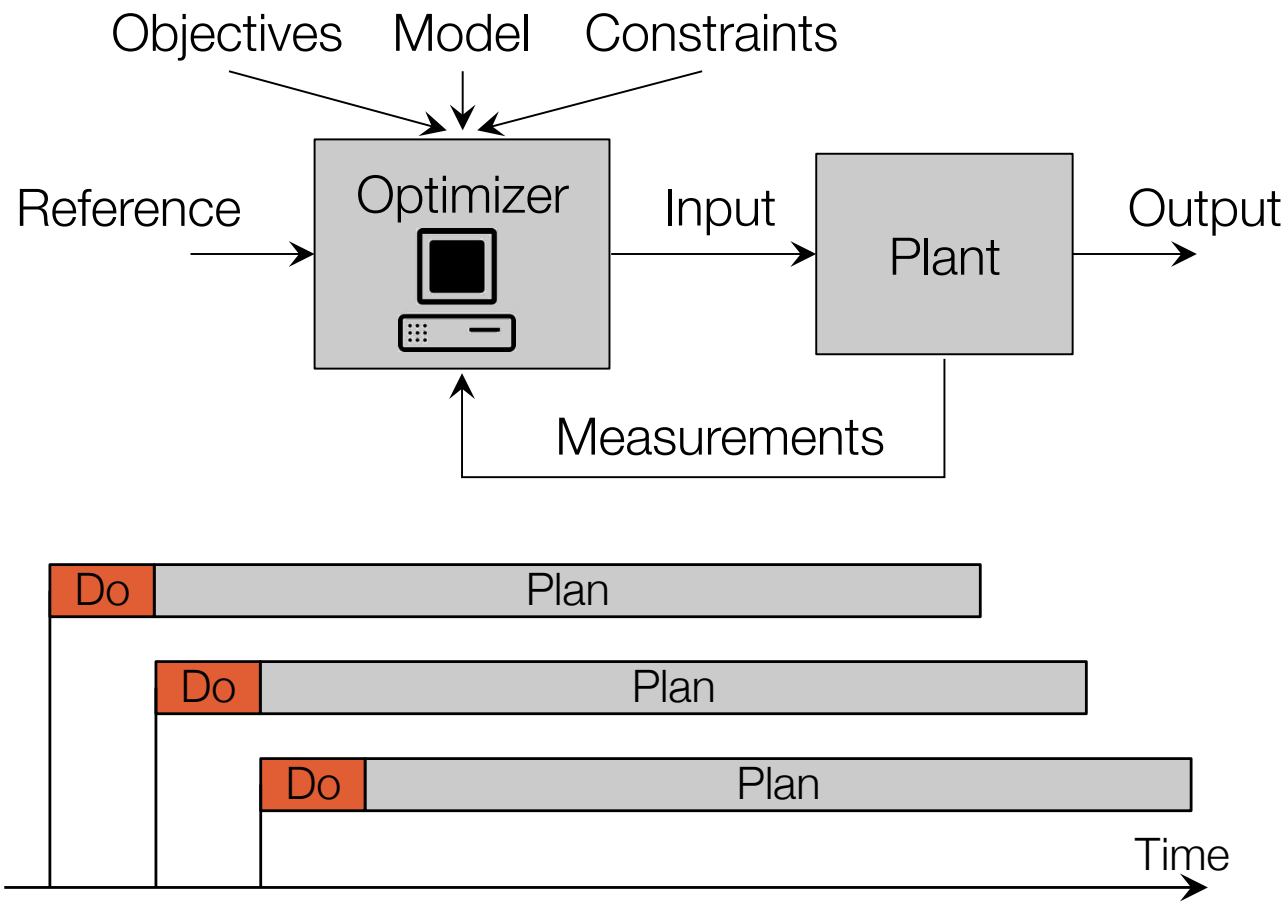
- Fundamental difference: Batch optimization returns a sequence $U^\star(x(0))$ of **numeric values** depending only on the initial state $x(0)$, while dynamic programming yields **feedback policies** $u_i^\star = F_i x_i$, $i = 0, \ldots, N-1$ depending on each $x_i$.
- If the state evolves exactly 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 $x_i$ or $u_i$ 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.

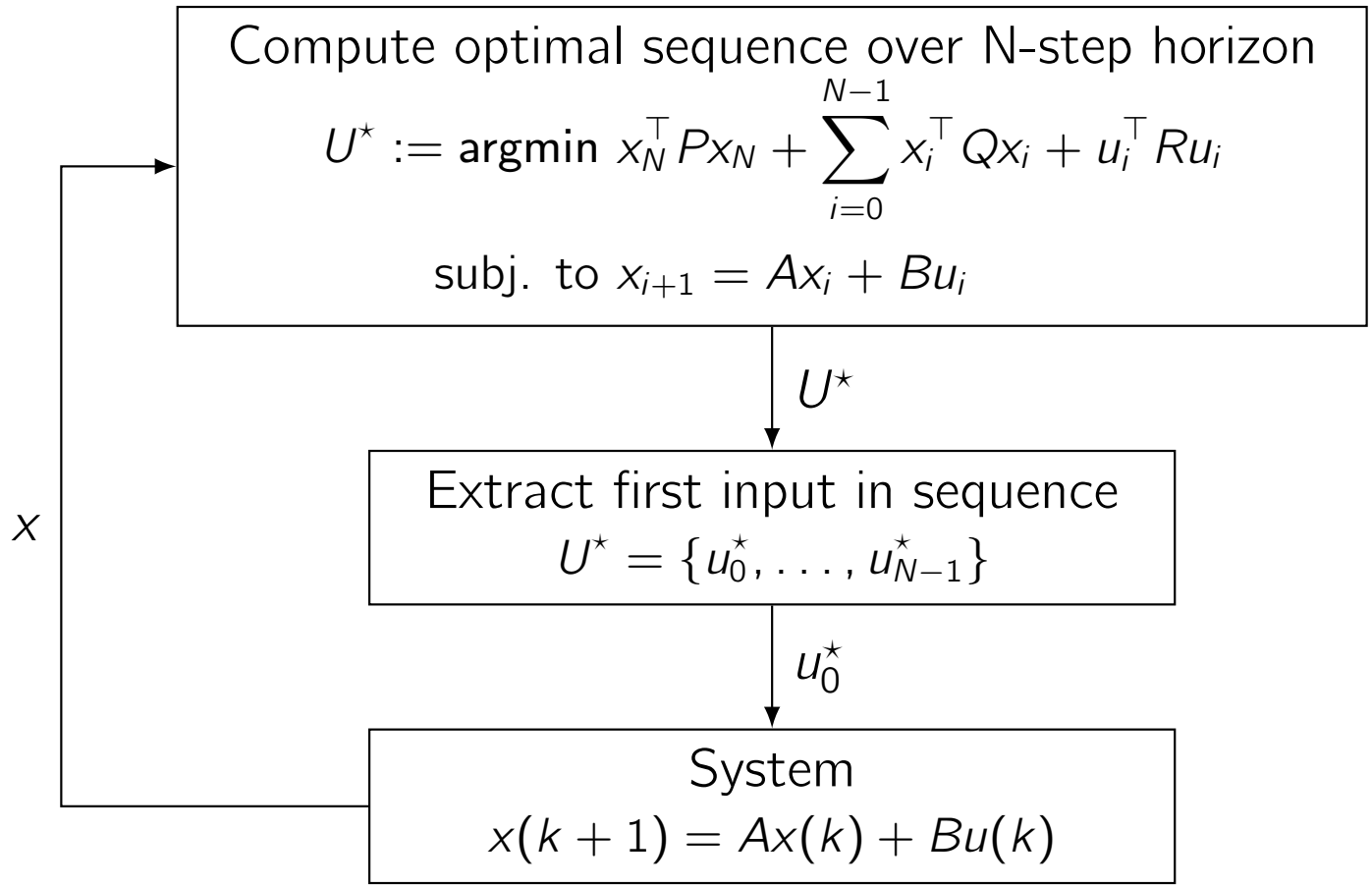
# Outline

# Receding horizon control




Receding horizon strategy introduces feedback.

# Receding Horizon Control




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

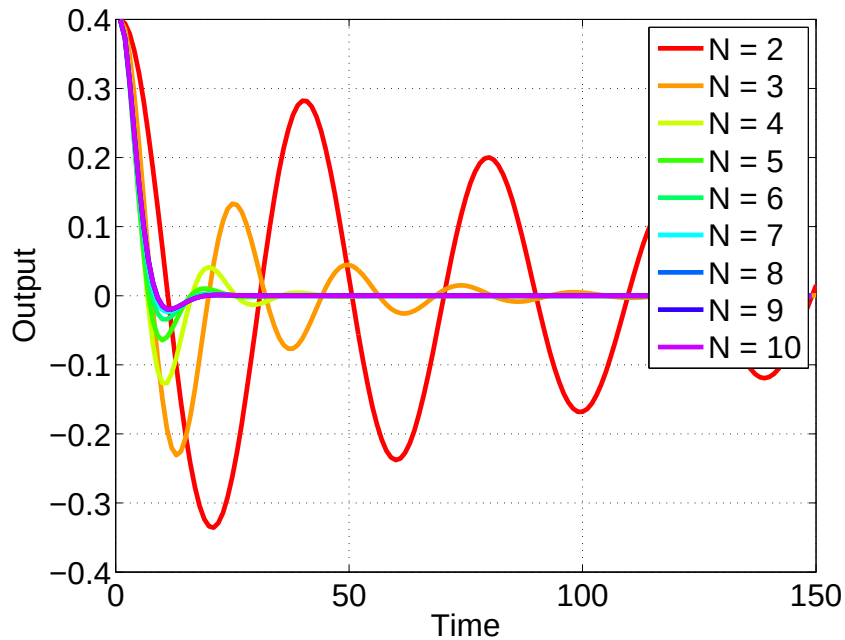
$$G(s) := \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2}$$

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

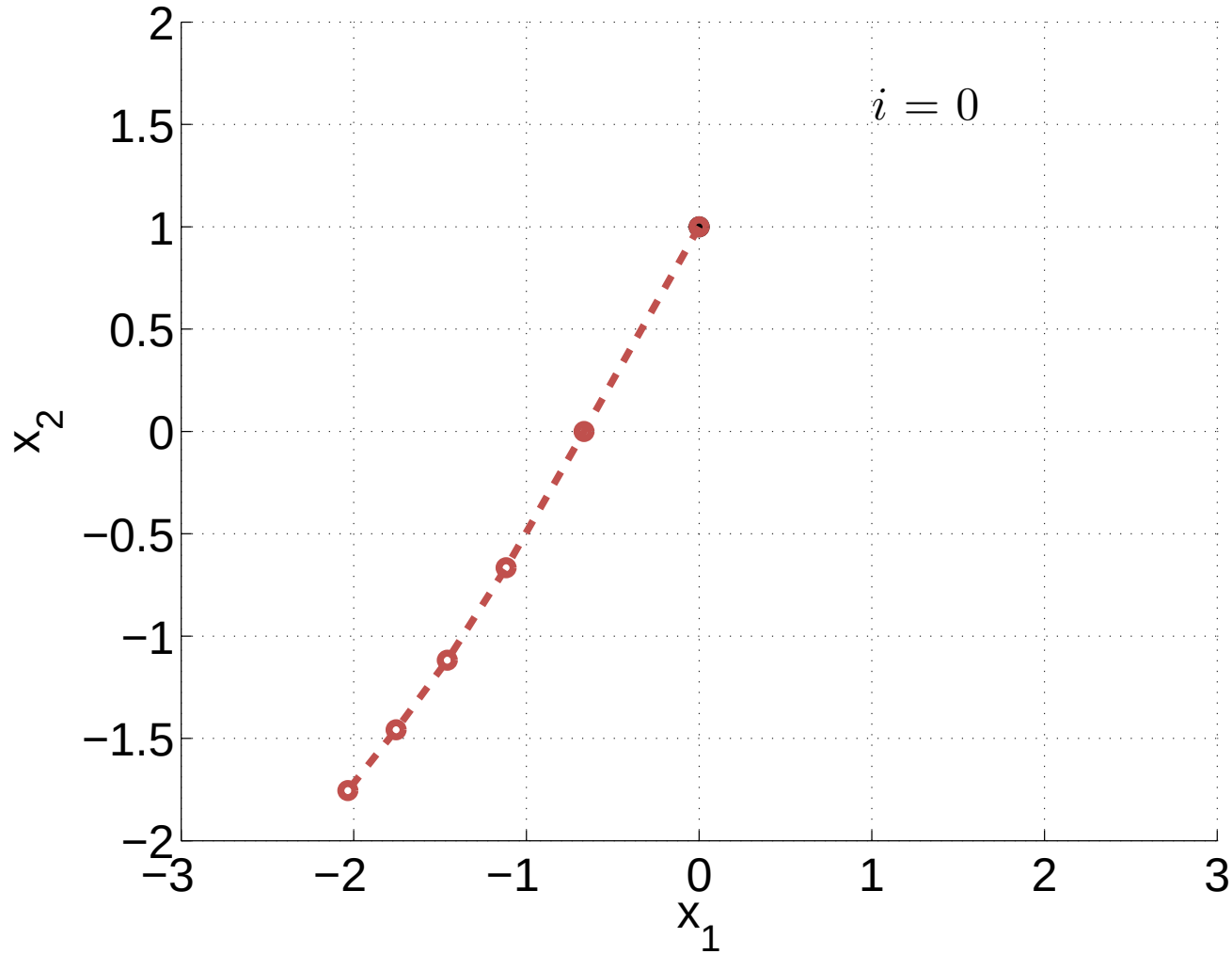
Discrete-time state-space model:

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

Closed-loop response

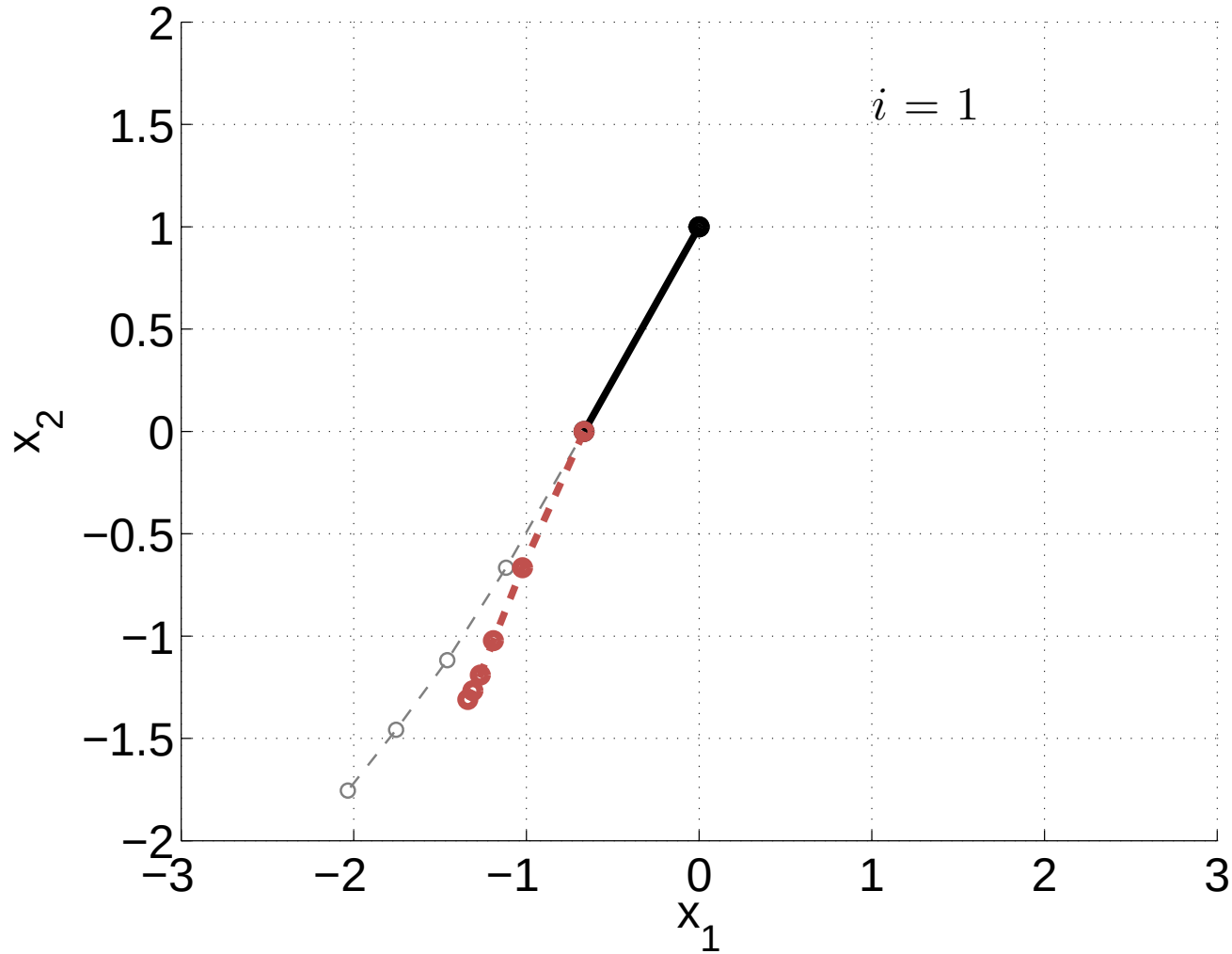


# Example: Short horizon $N = 5$



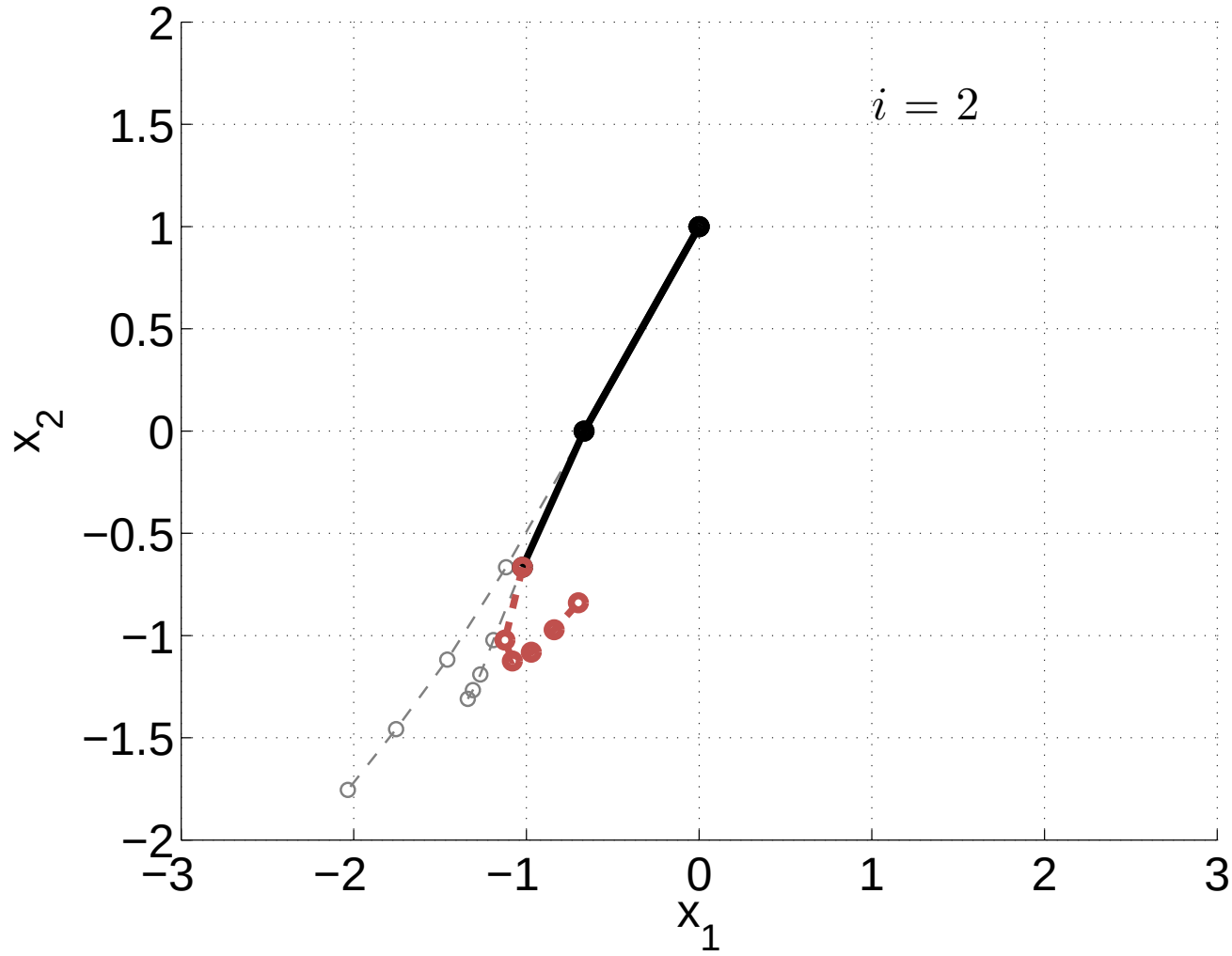

Short horizon: Prediction and closed-loop response differ.

# Example: Short horizon $N = 5$



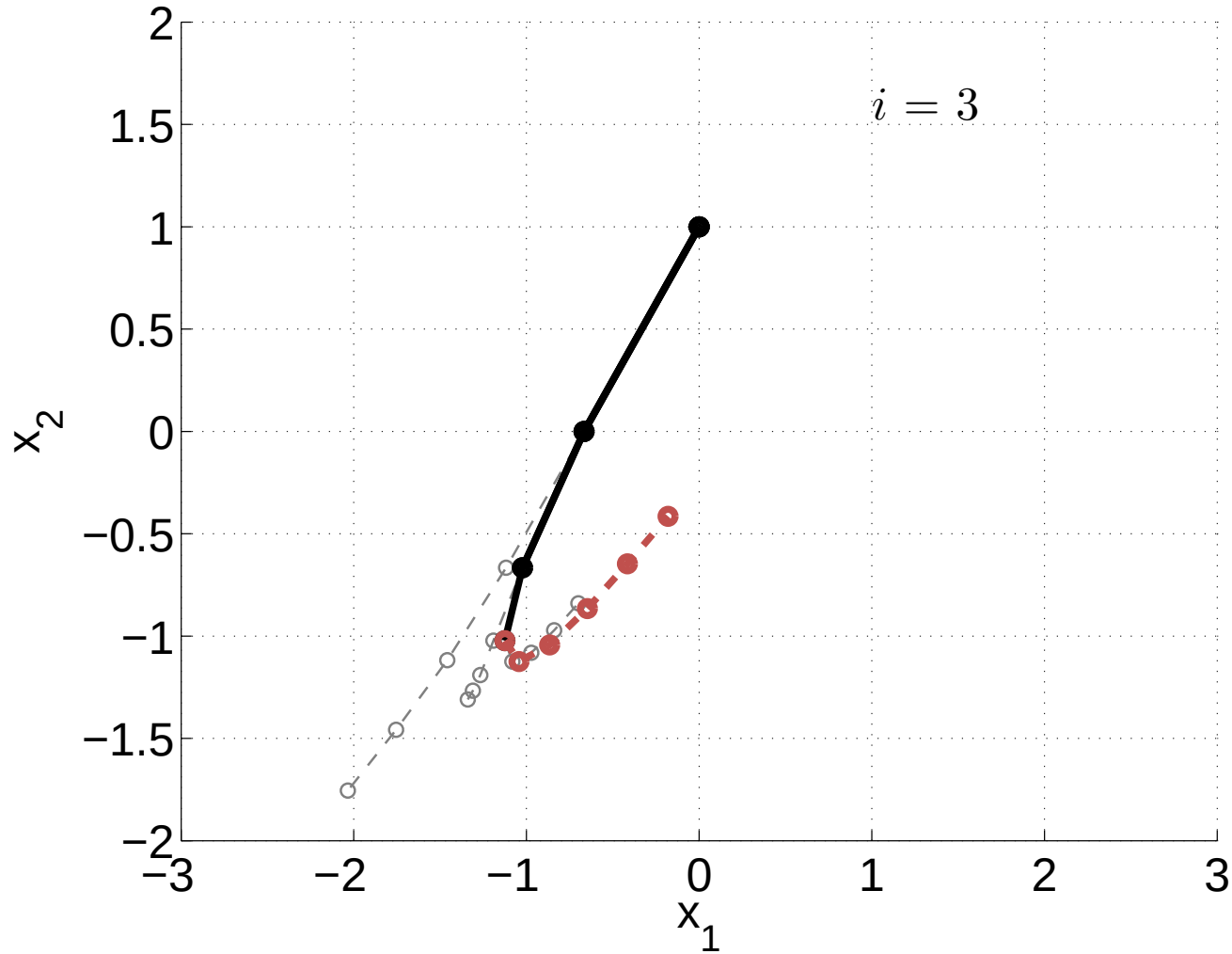

Short horizon: Prediction and closed-loop response differ.

# Example: Short horizon $N = 5$



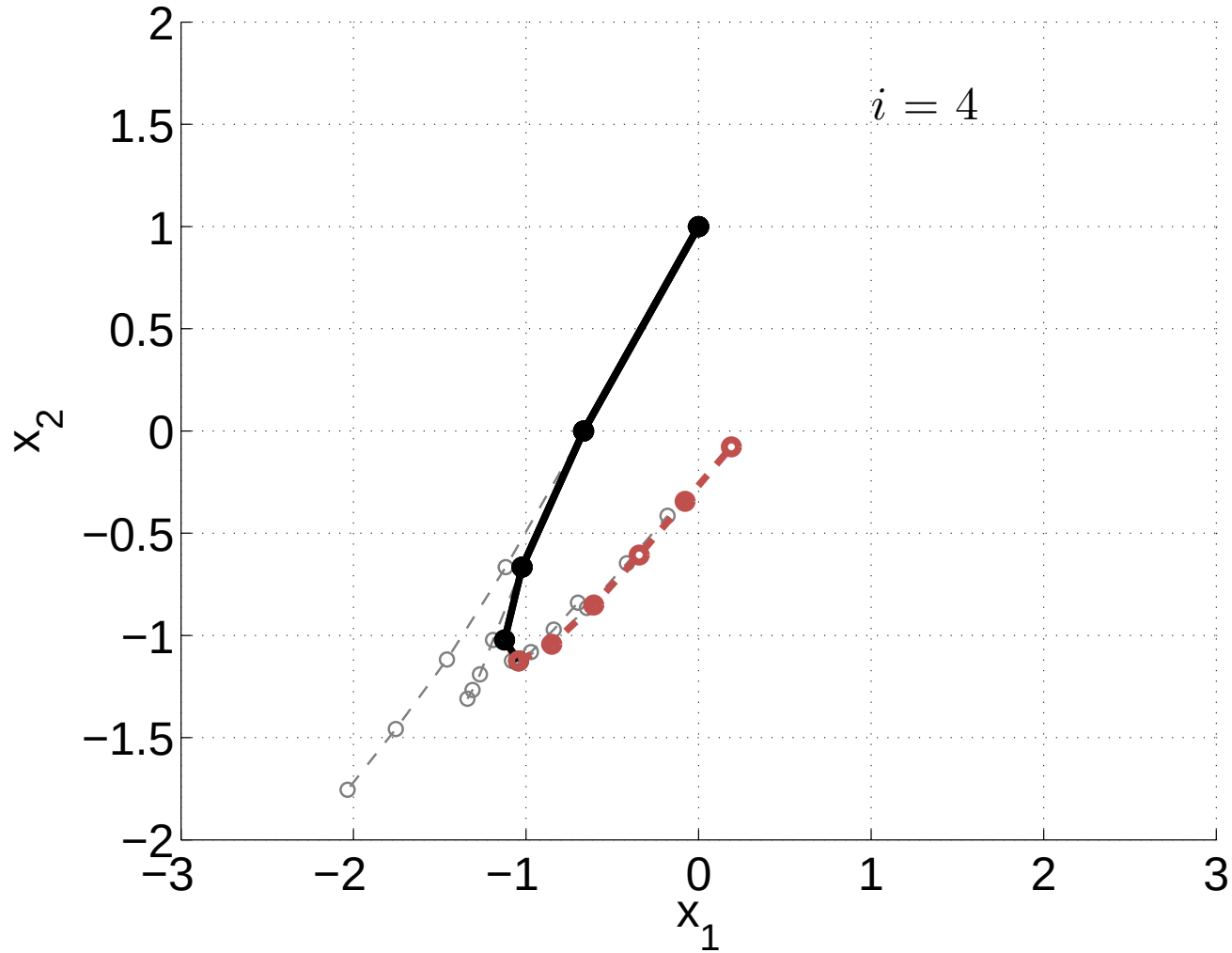

Short horizon: Prediction and closed-loop response differ.

# Example: Short horizon $N = 5$



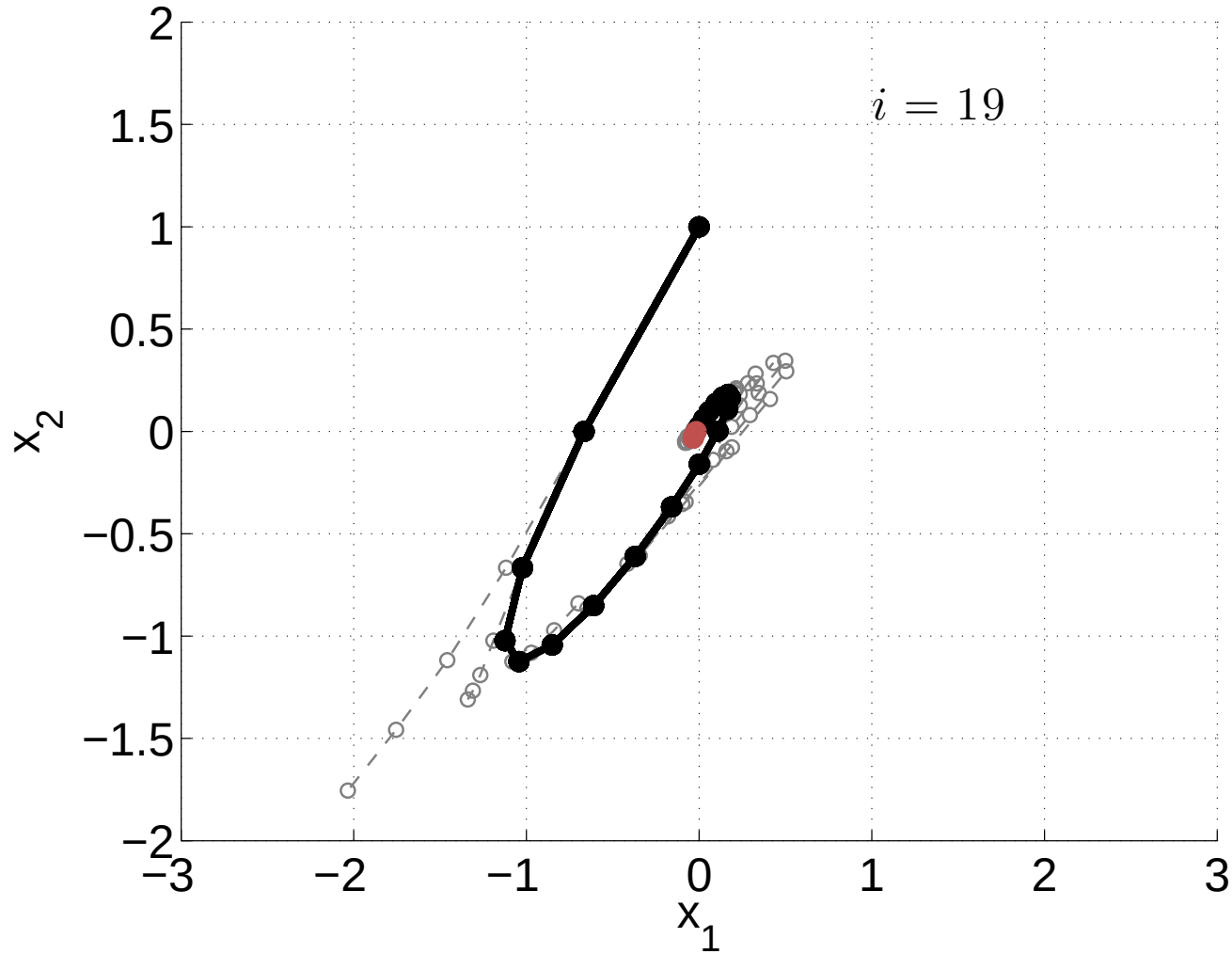

Short horizon: Prediction and closed-loop response differ.

# Example: Short horizon $N = 5$



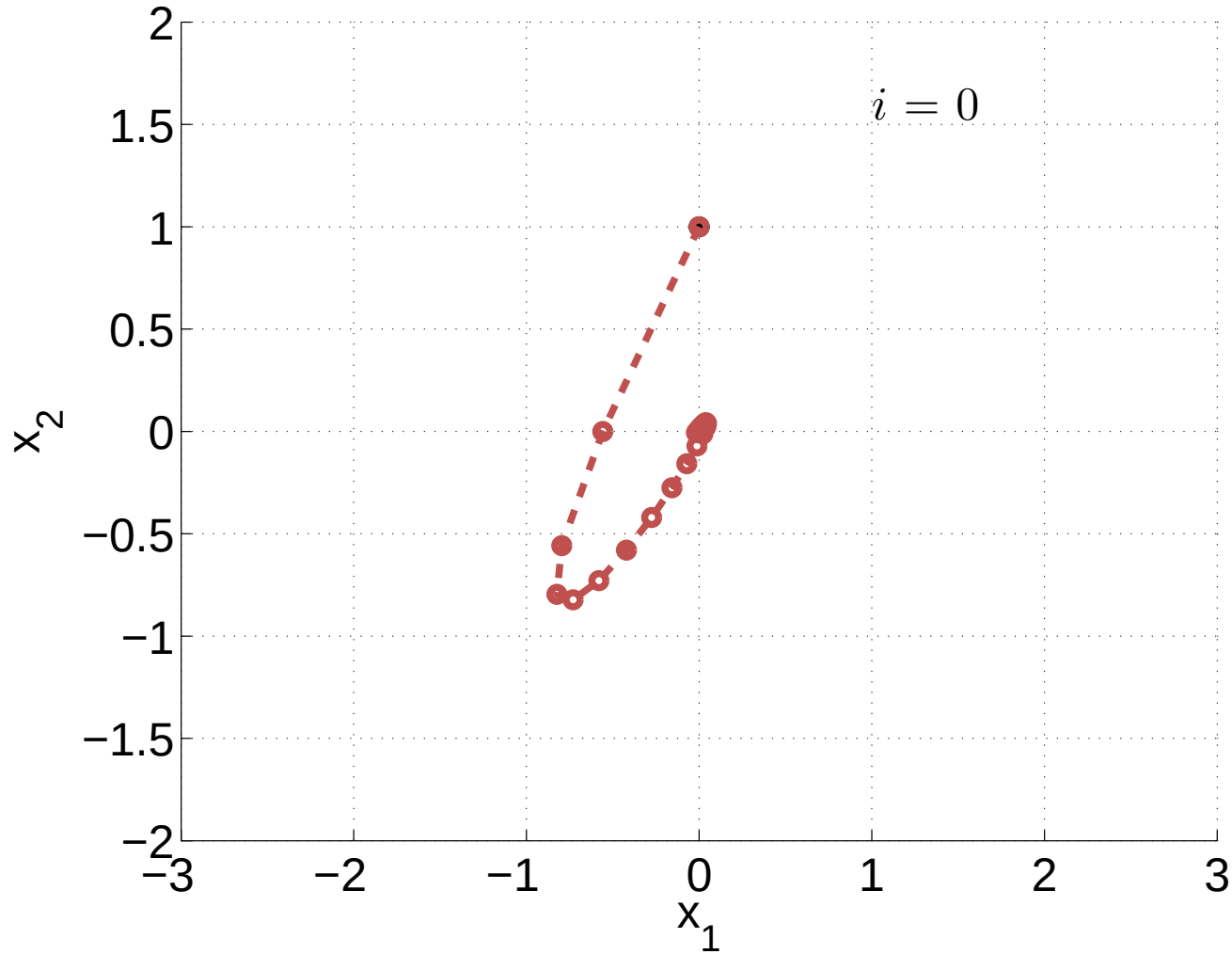

Short horizon: Prediction and closed-loop response differ.

# Example: Short horizon $N = 5$



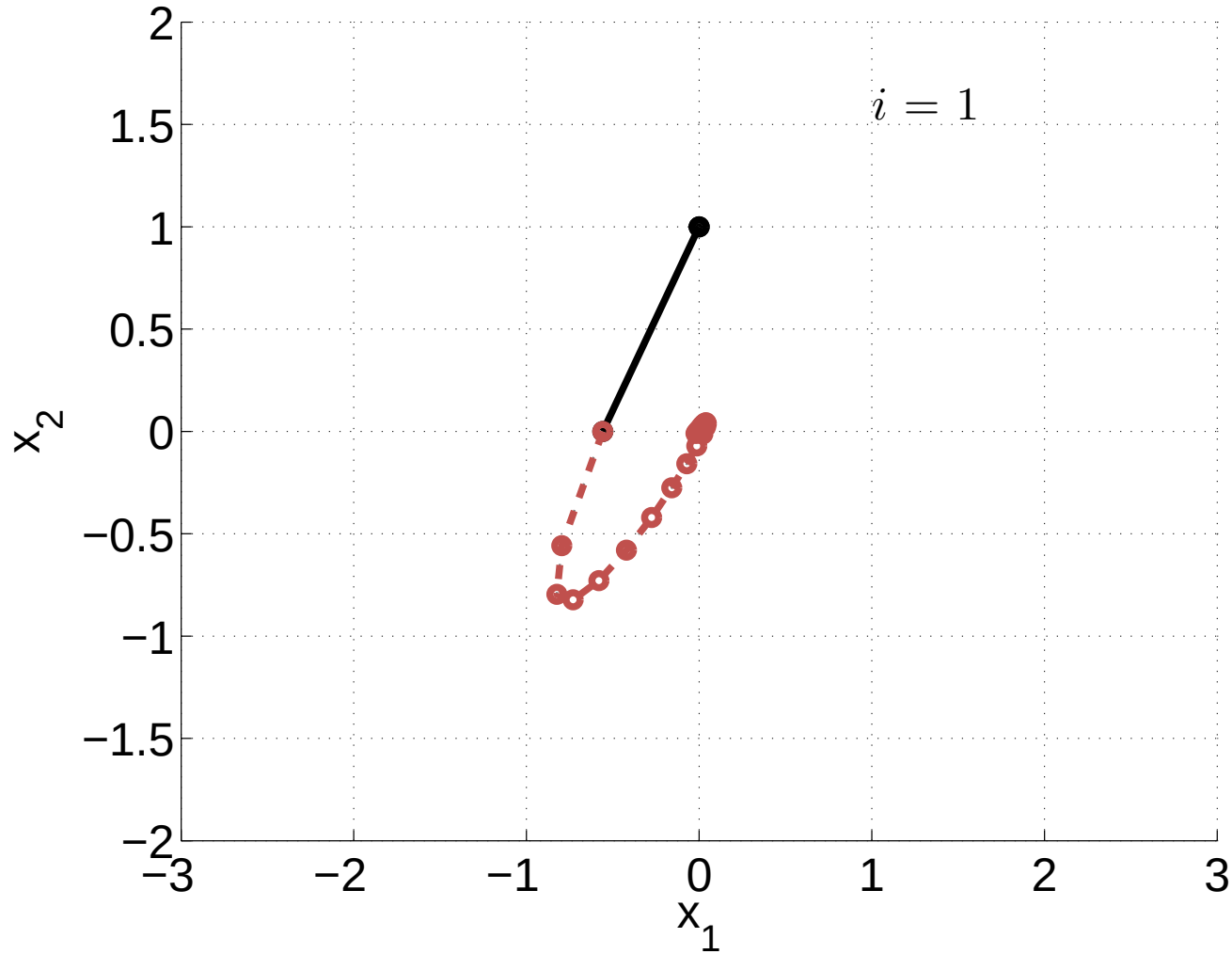

Short horizon: Prediction and closed-loop response differ.

# Example: Long horizon $N = 20$



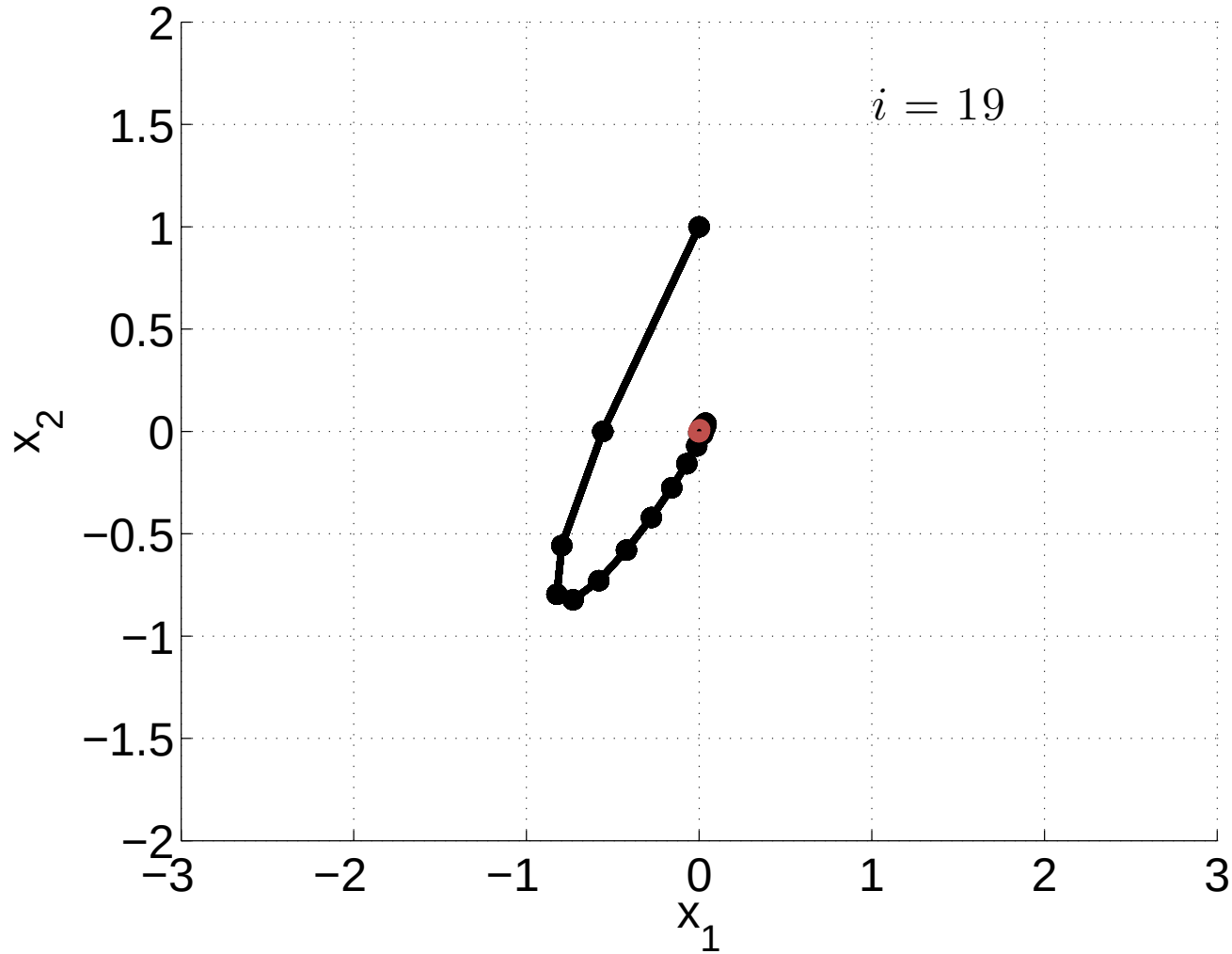

Long horizon: Prediction and closed-loop match.

# 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

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

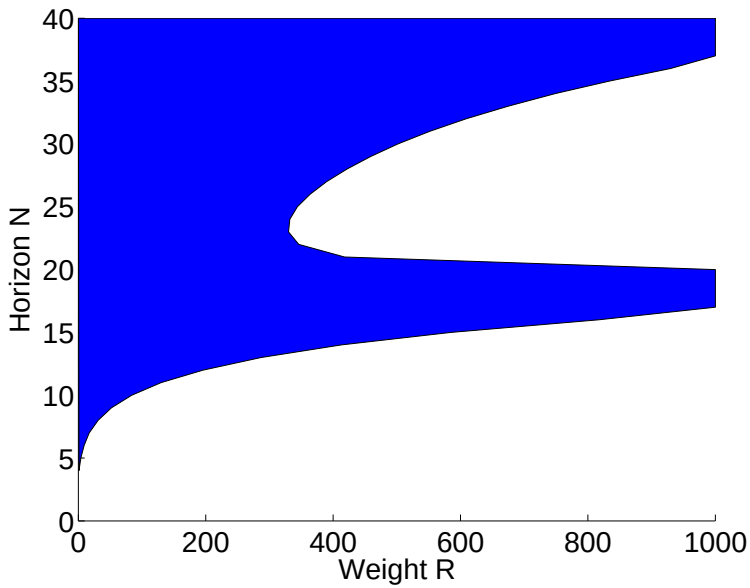
where $\omega = 0.1$ and $\zeta = -1$, which has been discretized at $1r/s$.
(Note that this system is unstable)

Is the system

$$x(k+1) = (A + BK_{R,N})x(k)$$

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:

$$J_\infty(x(0)) = \min_{u(\cdot)} \sum_{i=0}^{\infty} \left(x_i^\top Q x_i + u_i^\top R u_i\right)$$
$$\text{subj. to } x_{i+1} = Ax_i + Bu_i, \quad i = 0, 1, 2, \ldots, \infty,$$
$$x_0 = x(0)$$

- 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 $P_\infty$ 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 $P_\infty$, it must satisfy the following (from (6), with $P_i = P_{i+1} = P_\infty$)

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

  which is called the **Algebraic Riccati equation (ARE)**.
- The constant feedback matrix $F_\infty$ 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 $i = \infty$ and solved for $i \searrow 0$) converges to the unique positive definite solution $P_\infty$ of the ARE.

# Stability of Infinite-Horizon LQR (1/2)

- 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.
- 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 $J^\star(x(k)) = x(k)^\top P_\infty x(k)$ is actually a Lyapunov function for the system, i.e. 1) $J^\star(x) > 0\ \forall x \neq 0$, $J^\star(0) = 0$, 2) $J^\star(x) \to \infty$ for $\|x\| \to \infty$ and 3) $J^\star(x(k+1)) < J^\star(x(k))$, for any $x(k) \neq 0$.
  This implies that

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

# Stability of Infinite-Horizon LQR (2/2)

**Lemma: Lyapunov function for LQR**

If the system is stabilizable and detectable, the optimal value function $J^\star(x) = x^T P_\infty x$ is a Lyapunov function for the system $x^+ = (A + BF_\infty)x$ where $F_\infty = -(R + B^T P_\infty B)^{-1} B^T P_\infty A$ and $P_\infty$ solves the Algebraic Riccati equation for some $Q \succeq 0$, $R \succ 0$.

$P_\infty \succ 0$ gives the first two requirements.

$$J^\star(x(k)) = x^T(k) P_\infty x(k) = \sum_{i=k}^{\infty} x^T(i)(Q + F_\infty^T R F_\infty)x(i)$$

Consider the value of $J^\star(x(k+1))$

$$\begin{aligned} J^\star(x(k+1)) = J^\star((A + BF_\infty)x(k)) &= \sum_{i=k+1}^{\infty} x^T(i)(Q + F_\infty^T R F_\infty)x(i) \\ &= J^\star(x(k)) - x^T(k)(Q + F_\infty^T R F_\infty)x(k) < J^\star(x(k)) \end{aligned}$$

# 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 $\infty$ (i.e. the cost with the optimal controller choice). This can be computed from the ARE:

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

# 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

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

   - 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

$$x_{i+N} = 0$$

# Linear Quadratic Optimal Control: Recap

**Goal:** Control law to minimize relative 'energy' of input and state/output

**Why?**

- Easy to describe objective / tune controller
- Simple to compute and implement
- Proven and effective

**Why infinite-horizon?**

- Stable
- Optimal solution (doesn't usually matter)

In MPC we normally cannot have an infinite horizon because it results in an infinite number of optimization variables.
Use 'tricks' to 'simulate' quasi-infinite horizon.