Model Predictive Control

Lecture: Optimal Control of Unconstrained Systems

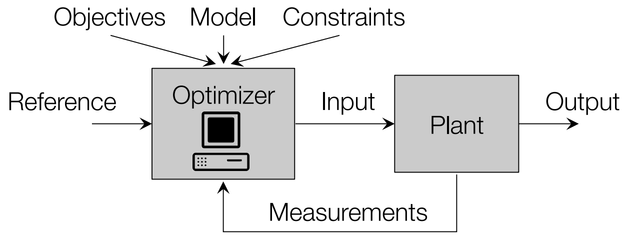
Colin Jones

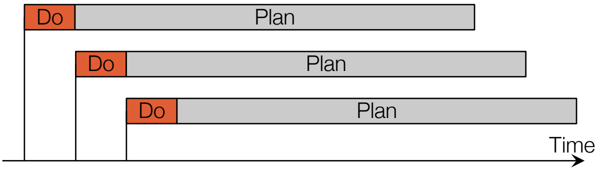
Laboratoire d'Automatique, EPFL

Outline

1. Recap
    - Receding Horizon Control
    - Modeling for MPC
    - Lyapunov Functions
2. Linear quadratic regulator
    - Computation of LQR Controllers
    - Stability of LQR Controllers
3. Summary of Exercise Session

Receding horizon control





Receding horizon strategy introduces feedback.

Why is This a Good Idea?

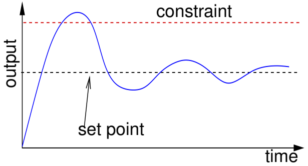
All physical systems have **constraints**.

- Physical constraints, e.g. actuator limits
- Performance constraints, e.g. overshoot
- Safety constraints, e.g. temperature/pressure limits

Optimal operating points are often near constraints.

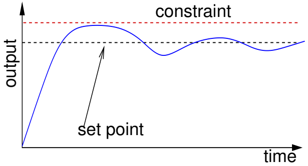
Classical control methods:

- No knowledge of constraints
- Set point sufficiently far from constraints
- Suboptimal plant operation



Predictive control:

- Constraints included in the design
- Set point optimal
- Efficient plant operation



MPC: Mathematical formulation

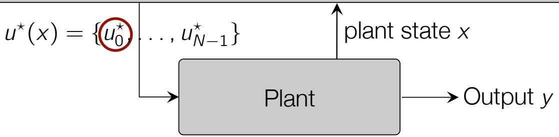
$$
\begin{aligned}
u^\star(x) := \text{argmin} \quad & x_N^T Q_f x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i & \\
\text{s.t.} \quad & x_0 = x & \text{measurement} \\
& x_{i+1} = A x_i + B u_i & \text{system model} \\
& C x_i + D u_i \leq b & \text{constraints} \\
& R \succ 0, Q \succ 0 & \text{performance weights}
\end{aligned}
$$

Problem is defined by

- **Objective** that is minimized,
  e.g., distance from origin, sum of squared/absolute errors, economic,...
- Internal **system model** to predict system behavior
  e.g., linear, nonlinear, single-/multi-variable, ...
- **Constraints** that have to be satisfied
  e.g., on inputs, outputs, states, linear, quadratic,...

MPC: Mathematical formulation

$$
\begin{aligned}
u^\star(x) := \operatorname{argmin} \quad & x_N^T Q_f x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i \\
\text{s.t.} \quad & x_0 = x && \text{measurement} \\
& x_{i+1} = A x_i + B u_i && \text{system model} \\
& C x_i + D u_i \leq b && \text{constraints} \\
& R \succ 0, Q \succ 0 && \text{performance weights}
\end{aligned}
$$



At each sample time:

- Measure /estimate current state
- Find the optimal input sequence for the entire planning window N
- Implement only the **first** control action

Summary

- Optimize over future possible trajectories of the system to:
  1. Satisfy constraints (now and always)
  2. Stabilize the system
  3. Optimize "performance"

  In that order!

- Re-optimizing when new measurements are obtained **introduces feedback**
  - The model is wrong
  - Unknown disturbances will act in the future

Modeling for MPC: Review

Models in MPC are (usually): Discrete-time, time invariant, state-space and

Nonlinear	$x^+ = f(x, u)$	$y = h(x, u)$
Linear	$x^+ = Ax + Bu$	$y = Cx + Du$

Notes:

- Assume state-measurement $\Rightarrow$ often drop the $y = h(x, u)$.
- Old MPC approaches were based on step response models. Still common in industry, but theoretically a very bad idea.
- Frequency concepts (Bode, Nyquist, Laplace, etc) and controllers based on these ($\mathcal{H}_\infty$, lead/lag filters, etc) are not used in MPC because **constraints make all systems nonlinear**.
- Throughout the course, we will assume a discrete-time, state-space model provided

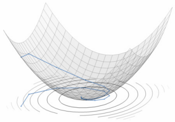
Lyapunov Functions

Idea: System is stable, if total ‘energy’ is decreasing over time. Lyapunov function is a system theoretic generalization of ‘energy’.

Lyapunov function

A continuous[1] function $V : \mathbb{R}^n \to \mathbb{R}_+$ is called a (asymptotic) **Lyapunov function** for the system $x^+ = f(x)$, if

- $\|x\| \to \infty \Rightarrow V(x) \to \infty$
- $V(0) = 0$ and $V(x) > 0 \; \forall x \in \mathbb{R}^n \setminus \{0\}$
- $V(f(x)) < V(x) \; \forall x \in \mathbb{R}^n \setminus \{0\}$



We will often speak of a local Lyapunov function, in which these conditions need only be satisfied in some region $x \in \mathcal{X}$.

[1] This assumption can be relaxed by requiring an additional state dependent upper bound on $V(x)$ [Rawlings & Mayne, 2009].

Lyapunov Functions for Stability

Theorem: Global Lyapunov Stability

If a system admits a (asymptotic) Lyapunov function, then the equilibrium point at the origin is **asymptotically stable**.

Rough sketch of proof.

Consider a system $x^+ = f(x)$ with Lyapunov function V and initial state x_0.

The resulting state sequence $\{x_0, x_1, x_2, \dots\}$ will have an associated sequence $\{V(x_0), V(x_1), V(x_2), \dots\}$ which is:

- positive
- monotonically decreasing

Since the only point where $V(x) = 0$ is $x = 0$, we have that in the limit $V(x_i)$ tends to zero, and therefore x_i tends to the origin. $\square$

Remarks on Lyapunov functions

- Finding a Lyapunov function (and proving that it is one!) is the challenge
- Find Lyapunov function for optimization-based controller??? No idea?!
- MPC: setup the problem so that the **optimal value of the cost function is always a Lyapunov function** by design.
  - Will see a simple version of this today with LQR
- Stable linear systems: $V(x) = x^T Px$ is always a Lyapunov function
  Find P by solving the Lyapunov equation for some $Q > 0$

$$A^T PA - P = -Q$$

  Matlab: `P = dlyap(A,Q);` Solves discrete-time Lyapunov equation

Outline

1. Recap
   - Receding Horizon Control
   - Modeling for MPC
   - Lyapunov Functions

2. Linear quadratic regulator
   - Computation of LQR Controllers
   - Stability of LQR Controllers

3. Summary of Exercise Session

Linear Quadratic Regulator

$$x^+ = Ax + Bu$$

Goal: Move from state x to the origin. (i.e., keep x 'small')

Consider N inputs into the future

$$\mathbf{u} := \{u_0, \ldots, u_{N-1}\}$$

Express the 'cost' of being in state x and applying input u with the function

$$l(x, u) := x^T Q x + u^T R u$$

Cost of following a trajectory:

$$V(x_0, \mathbf{u}) = \sum_{i=0}^{N} x_i^T Q x_i + u_i^T R u_i$$

Assume: $R \succ 0$, $Q \succeq 0$. Real, symmetric and positive (semi)definite.

Motivation for LQR

Consider the system:

$$x^+ = Ax + Bu \qquad\qquad y = Cx$$

and set $Q = C^T C$, $R = \rho I$. Minimize the cost

$$\sum_{i=0}^{N} \|y_i\|_2^2 + \rho \|u_i\|_2^2$$

We're minimizing the **energy** in the input and output signals.

Large $\rho \Rightarrow$ small input energy, output weakly controlled
Small $\rho \Rightarrow$ large input energy, output strongly controlled

Motivation for LQR

Consider the system:

$$x^+ = Ax + Bu \qquad\qquad y = Cx$$

and set $Q = C^T C$, $R = \rho I$. Minimize the cost

$$\sum_{i=0}^{N} \|y_i\|_2^2 + \rho \|u_i\|_2^2$$

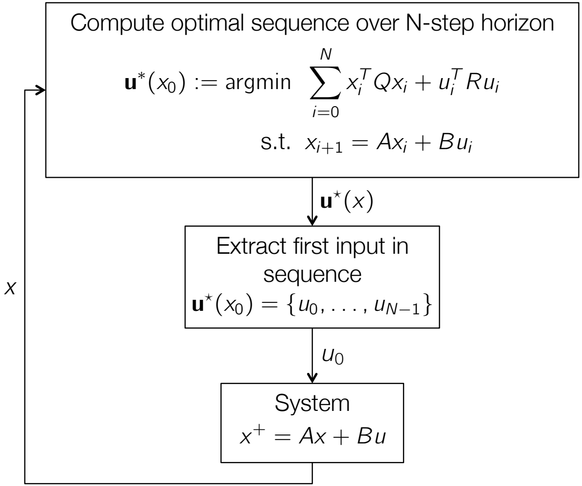
We're minimizing the **energy** in the input and output signals.

Large $\rho \Rightarrow$ small input energy, output weakly controlled
Small $\rho \Rightarrow$ large input energy, output strongly controlled

Real motivation

- Works well in practice
- We can solve it (very common motivation in control!)
- Solution is simple, and easy to implement in embedded controller

Receding Horizon Control



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

LQR Solution Methods

Two equivalent solution procedures:

Dynamic programming

Pros:

- Leads to elegant closed-form solution for LQR
- Provides a solution when $N \to \infty$

Cons:

- Virtually no problems have simple, closed-form solutions (except LQR)

Optimization / Least-squares

Pros:

- Can extend to nonlinear, constrained systems with complex cost-functions

Cons:

- Finite-horizon only
- More computationally intense

Principle of Optimality/Dynamic Programming

$$V^\star(x_0) := \min_{\mathbf{u}} \sum_{k=0}^{N} l(x_k, u_k) \quad \text{s.t. } x_{k+1} = Ax_k + Bu_k$$

Consider problem with $N = 2$:

$$\begin{aligned} V^\star(x_0) = \min_{u_0, u_1, u_2} \; & l(x_0, u_0) + l(x_1, u_1) + l(x_2, u_2) \\ \text{s.t. } & x_1 = Ax_0 + Bu_0 \\ & x_2 = Ax_1 + Bu_1 \end{aligned}$$

Principle of Optimality/Dynamic Programming

$$V^\star(x_0) := \min_{\mathbf{u}} \sum_{k=0}^{N} l(x_k, u_k) \quad \text{s.t. } x_{k+1} = Ax_k + Bu_k$$

Consider problem with $N = 2$:

$$\begin{aligned} V^\star(x_0) = \min_{u_0, u_1, u_2} \ & l(x_0, u_0) + l(x_1, u_1) + \overbrace{l(x_2, u_2)}^{\text{Fix } x_2 \text{ and this is a function only of } u_2} \\ \text{s.t. } \ & x_1 = Ax_0 + Bu_0 \\ & x_2 = Ax_1 + Bu_1 \end{aligned}$$

Principle of Optimality/Dynamic Programming

$$V^\star(x_0) := \min_{\mathbf{u}} \sum_{k=0}^{N} l(x_k, u_k) \quad \text{s.t. } x_{k+1} = Ax_k + Bu_k$$

Consider problem with $N = 2$:

$$\begin{aligned} V^\star(x_0) = \min_{u_0,u_1,u_2} \; & l(x_0,u_0) + l(x_1,u_1) + \overbrace{l(x_2,u_2)}^{\text{Fix } x_2 \text{ and this is a function only of } u_2} \\ \text{s.t. } \; & x_1 = Ax_0 + Bu_0 \\ & x_2 = Ax_1 + Bu_1 \end{aligned}$$

$$\begin{aligned} = \min_{u_0,u_1} \; & l(x_0,u_0) + l(x_1,u_1) + V_2^\star(Ax_1 + Bu_1) \\ \text{s.t. } \; & x_1 = Ax_0 + Bu_0 \end{aligned}$$

where:

$$V_2^\star(x_2) := \min_{u_2} \; l(x_2,u_2)$$

Principle of Optimality/Dynamic Programming

$$
\begin{aligned}
V^\star(x_0) = \min_{u_0, u_1} \; & l(x_0, u_0) + \; l(x_1, u_1) + V_2^\star(Ax_1 + Bu_1) \\
\text{s.t. } \; & x_1 = Ax_0 + Bu_0
\end{aligned}
$$

Principle of Optimality/Dynamic Programming

$$V^\star(x_0) = \min_{u_0, u_1} l(x_0, u_0) + \overbrace{l(x_1, u_1) + V_2^\star(Ax_1 + Bu_1)}^{\substack{\text{Fix } x_1 \text{ and this is} \\ \text{a function only of } u_1}}$$
$$\text{s.t. } x_1 = Ax_0 + Bu_0$$

Principle of Optimality/Dynamic Programming

$$
\begin{aligned}
V^\star(x_0) = \min_{u_0, u_1}\ & l(x_0, u_0) + \overbrace{l(x_1, u_1) + V_2^\star(Ax_1 + Bu_1)}^{\substack{\text{Fix } x_1 \text{ and this is} \\ \text{a function only of } u_1}} \\
\text{s.t. }\ & x_1 = Ax_0 + Bu_0 \\
= \min_{u_0}\ & l(x_0, u_0) + V_1^\star(Ax_0 + Bu_0)
\end{aligned}
$$

where:

$$
V_1^\star(x_1) := \min_{u_1}\ l(x_1, u_1) + V_2^\star(Ax_1 + Bu_1)
$$

Principle of Optimality/Dynamic Programming

$$
\begin{aligned}
V^\star(x_0) = \min_{u_0, u_1} \; & l(x_0, u_0) + \overbrace{l(x_1, u_1) + V_2^\star(Ax_1 + Bu_1)}^{\substack{\text{Fix } x_1 \text{ and this is} \\ \text{a function only of } u_1}} \\
\text{s.t.} \; & x_1 = Ax_0 + Bu_0 \\
= \min_{u_0} \; & l(x_0, u_0) + V_1^\star(Ax_0 + Bu_0)
\end{aligned}
$$

where:

$$
V_1^\star(x_1) := \min_{u_1} \; l(x_1, u_1) + V_2^\star(Ax_1 + Bu_1)
$$

Finally only u_0 to minimize:

$$
V^\star(x_0) = \min_{u_0} \; l(x_0, u_0) + V_1^\star(Ax_0 + Bu_0)
$$

The value that minimizes this function $u_0^\star(x_0)$ is our control input.

Dynamic Programming

Procedure:

1. Start at step N and compute

$$V_N^\star(x_N) := \min_{u_N} \; l(x_N, u_N)$$

2. Iterate *backwards* for $i = N-1 \dots 0$ (*DP iteration*)

$$V_i^\star(x_i) := \min_{u_i} \; l(x_i, u_i) + V_{i+1}^\star(Ax_i + Bu_i)$$

3. $V^\star(x_0) := V_0^\star(x_0)$ and the optimal controller is the optimizer $u_0^\star(x_0)$

Requirements:

- Closed-form representation of the function $V_i^\star(x)$
- Ability to compute a DP iteration

Normally impossible. Some special cases (e.g., LQR).

DP Solution of LQR

$$V^\star(x_0) := \min_{\mathbf{u}} \sum_{i=0}^{N} {x_i}^T Q x_i + {u_i}^T R u_i \quad \text{s.t.} \quad x_{i+1} = Ax_i + Bu_i$$

DP iteration:

$$V_i^\star(x_i) = \min_{u_i} \; {x_i}^T Q x_i + {u_i}^T R u_i + V_{i+1}^\star(Ax_i + Bu_i)$$

for $i = N-1, \ldots, 0$.

We will show:

- $V_i^\star(x)$ is quadratic (and therefore $V^\star(x)$ is)
- $V_i^\star(x)$ is positive definite (and therefore $V^\star(x)$ is)
- Optimizer $u_0^\star(x)$ is linear

Bellman Recursion

Assume $V_{i+1}(x_{i+1}) = x_{i+1}^T H_{i+1} x_{i+1}$ is PSD.

DP iteration:

$$\begin{aligned} V_i(x_i) &= \min_{u_i} \quad {x_i}^T Q x_i + {u_i}^T R u_i + V_{i+1}(Ax_i + Bu_i) \\ &= \min_{u_i} \; \left(x_i^T Q x_i + u_i^T R u_i + (Ax_i + Bu_i)^T H_{i+1} (Ax_i + Bu_i) \right) \end{aligned}$$

Bellman Recursion

Assume $V_{i+1}(x_{i+1}) = x_{i+1}^T H_{i+1} x_{i+1}$ is PSD.

DP iteration:

$$\begin{aligned} V_i(x_i) &= \min_{u_i} \quad {x_i}^T Q x_i + {u_i}^T R u_i + V_{i+1}(A x_i + B u_i) \\ &= \min_{u_i} \ \left(x_i^T Q x_i + u_i^T R u_i + (A x_i + B u_i)^T H_{i+1} (A x_i + B u_i) \right) \end{aligned}$$

Setting derivative to zero

$$\begin{gathered} 2 u_i^T R + 2(A x_i + B u_i)^T H_{i+1} B = 0 \\ u_i^T (R + B^T H_{i+1} B) = -x_i^T A^T H_{i+1} B \end{gathered}$$

Bellman Recursion

Assume $V_{i+1}(x_{i+1}) = x_{i+1}^T H_{i+1} x_{i+1}$ is PSD.

DP iteration:

$$\begin{aligned} V_i(x_i) &= \min_{u_i} \quad {x_i}^T Q x_i + {u_i}^T R u_i + V_{i+1}(Ax_i + Bu_i) \\ &= \min_{u_i} \; \left(x_i^T Q x_i + u_i^T R u_i + (Ax_i + Bu_i)^T H_{i+1} (Ax_i + Bu_i) \right) \end{aligned}$$

Setting derivative to zero

$$2u_i^T R + 2(Ax_i + Bu_i)^T H_{i+1} B = 0$$
$$u_i^T (R + B^T H_{i+1} B) = -x_i^T A^T H_{i+1} B$$

gives the optimal input as

$$u_i^\star = K_i x_i \qquad\qquad K_i = -(R + B^T H_{i+1} B)^{-1} B^T H_{i+1} A$$

and the optimal cost

$$\begin{aligned} V_i^\star(x_i) &= {x_i}^T (Q + {K_i}^T R K_i + (A + BK_i)^T H_{i+1} (A + BK_i)) x_i \\ &= {x_i}^T H_i x_i \end{aligned}$$

Dynamic Programming

1. Start at step N and compute

$$\begin{aligned} V_N^\star(x_N) &:= \min_{u_N}\ {x_N}^T Q x_N + {u_N}^T R u_N \\ &= {x_N}^T Q x_N \end{aligned}$$

$$H_N := Q$$

2. Iterate *backwards* for $i = N-1 \dots 0$ (*DP iteration*)

$$V_i^\star(x_i) := \min_{u_i}\ {x_i}^T Q x_i + {u_i}^T R u_i + V_{i+1}^\star(Ax_i + Bu_i)$$

$$\begin{aligned} u_i^\star(x_i) &= K_i x_i & K_i &= -(R + B^T H_{i+1} B)^{-1} B^T H_{i+1} A \\ V_i^\star(x_i) &= x_i^T H_i x_i & H_i &:= Q + K_i^T R K_i + (A + BK_i)^T H_{i+1}(A + BK_i) \end{aligned}$$

3. $V^\star(x_0) := V_0^\star(x_0)$ and the optimal controller is the optimizer $u_0^\star(x_0)$

Finite-Horizon LQR Solution

Defines the optimal control law:

$$u_0^\star(x) = K_0 x \qquad\qquad V_0^\star(x) = x^T H_0 x$$

- We only ever apply the controller $u = K_0 x$ in a **receding-horizon fashion**.
- K_i's are for **planning** and are not used
- This is a simple, unconstrained, linear quadratic MPC problem

To make this work, we required:

- $V_i^\star(x)$ to have a **very** nice form (quadratic)
- Ability to solve the DP iteration in closed form

This cannot be done for almost any other problem...

LQR Solution Methods

Two equivalent solution procedures:

Dynamic programming

Pros:

- Leads to elegant closed-form solution for LQR
- Provides a solution when $N \to \infty$

Cons:

- Virtually no problems have simple, closed-form solutions (except LQR)

Optimization / Least-squares

Pros:

- Can extend to nonlinear, constrained systems with complex cost-functions

Cons:

- Finite-horizon only
- More computationally intense

Parametric Solution of Finite-Horizon LQR

$$V^\star(x_0) := \min_{\mathbf{u}} \sum_{i=0}^{N} {x_i}^T Q x_i + {u_i}^T R u_i \quad \text{s.t. } x_{i+1} = Ax_i + Bu_i$$

Writing it out in full gives:

$$\min_{\mathbf{u}} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}^T \begin{bmatrix} Q & & & \\ & Q & & \\ & & \ddots & \\ & & & Q \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} + \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_N \end{pmatrix}^T \begin{bmatrix} R & & & \\ & R & & \\ & & \ddots & \\ & & & R \end{bmatrix} \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_N \end{pmatrix}$$

$$\begin{bmatrix} -I & 0 & \cdots & \cdots & \cdots & 0 \\ A & -I & 0 & \cdots & \cdots & 0 \\ 0 & A & -I & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & A & -I \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} + \begin{bmatrix} B & 0 & \cdots & 0 \\ 0 & B & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & B \end{bmatrix} \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_N \end{pmatrix} = \begin{bmatrix} -A \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_0$$

Parametric Solution of Finite-Horizon LQR

Simple formulation of the **parametric least-squares problem**:

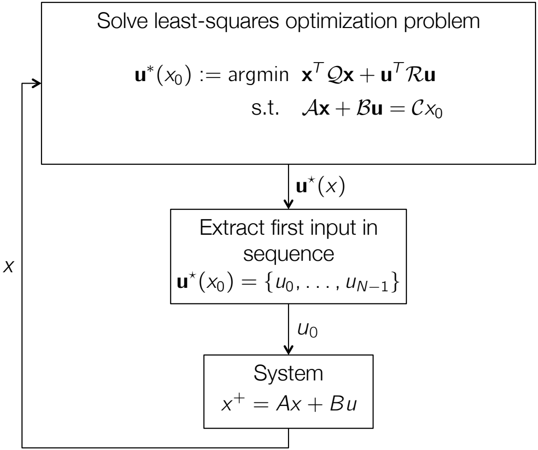
$$V^\star(x_0) := \min_{\mathbf{u}} \quad \mathbf{x}^T \mathcal{Q} \mathbf{x} + \mathbf{u}^T \mathcal{R} \mathbf{u} \quad \text{s.t.} \quad \mathcal{A}\mathbf{x} + \mathcal{B}\mathbf{u} = \mathcal{C} x_0$$

where $\mathbf{x} = \begin{bmatrix} x_1^T & \cdots & x_N^T \end{bmatrix}^T$, $\mathbf{u} = \begin{bmatrix} u_0^T & \cdots & u_{N-1}^T \end{bmatrix}^T$,

$$\mathcal{A} := \begin{bmatrix} -I & 0 & \cdots & \cdots & \cdots & 0 \\ A & -I & 0 & \cdots & \cdots & 0 \\ 0 & A & -I & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & A & -I \end{bmatrix} \quad \mathcal{B} := \begin{bmatrix} B & 0 & \cdots & 0 \\ 0 & B & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & B \end{bmatrix} \quad \mathcal{C} := \begin{bmatrix} -A \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\mathcal{Q} := \operatorname{diag}(Q, \dots, Q) \qquad \mathcal{R} := \operatorname{diag}(R, \dots, R)$$

LQR via Optimization



Implicitly defines a controller $\kappa(x) := \mathbf{u}_0^\star$, and for each fixed x_0, we can use a standard constrained least-squares solver to compute it.

Parametric Solution of Finite-Horizon LQR

Can re-write as a **parametric optimization problem** in the parameter x_0:

$$V^\star(x_0) := \min_{\mathbf{u}} \quad \mathbf{x}^T \mathcal{Q} \mathbf{x} + \mathbf{u}^T \mathcal{R} \mathbf{u} \quad \text{s.t.} \quad \mathcal{A}\mathbf{x} + \mathcal{B}\mathbf{u} = \mathcal{C} x_0$$

$\mathcal{A}$ is always invertible, so: $\mathbf{x} = -\mathcal{A}^{-1}\mathcal{B}\mathbf{u} + \mathcal{A}^{-1}\mathcal{C}x_0 = F\mathbf{u} + Gx_0$

$$= \min_{\mathbf{u}} \;\; (F\mathbf{u} + Gx_0)^T \mathcal{Q} (F\mathbf{u} + Gx_0) + \mathbf{u}^T \mathcal{R} \mathbf{u}$$

Take derivative and set to zero:

$$2\mathbf{u}^T \mathcal{R} + 2(F\mathbf{u} + Gx_0)^T \mathcal{Q} F = 0$$

Solving gives:

$$\mathbf{u} = \mathcal{K} x_0 = \begin{bmatrix} K_0 \\ \vdots \\ K_{N-1} \end{bmatrix} x_0 \qquad\qquad \mathcal{K} = -(\mathcal{R} + F^T \mathcal{Q} F)^{-1} F^T \mathcal{Q} G$$

This is a special kind of MPC, where we can write the solution in **closed-form**.

Explicit MPC lectures will show how to solve for some more general systems

Comparison of Solution Methods

Dyanmic Programming

- Can compute the infinite-horizon solution
  - Infinite-horizon guaranteed to be stabilizing

Optimization

- Can only compute finite-horizon
  - May not be stable
- Solution complexity is quadratic in horizon length vs linear for DP
- Concept extends to nonlinear, constrained systems with non-quadratic cost functions (i.e., MPC)

Both methods compute the same controller! (For a given horizon $N < \infty$)

Next : Impact of horizon length and infinite-horizon solutions.

Example - Impact of Horizon Length

Consider the lightly damped, stable system

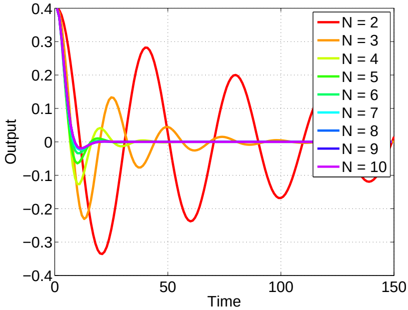
$$G(s) := \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2}$$

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

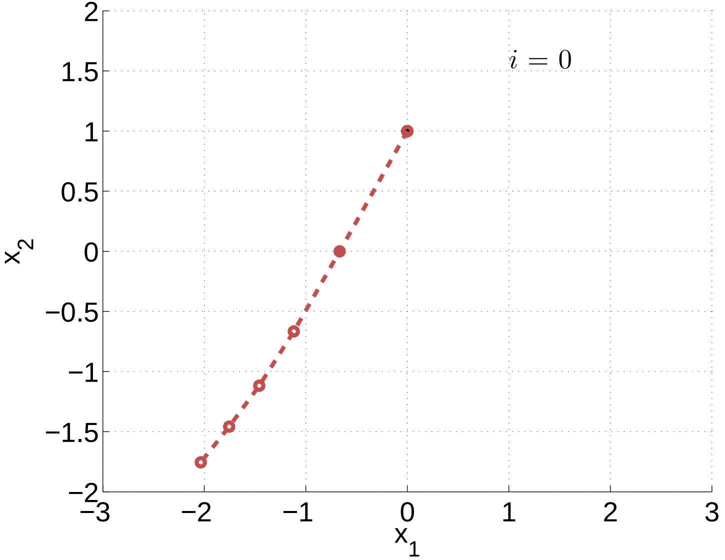
Discrete-time state-space model:

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

Closed-loop response

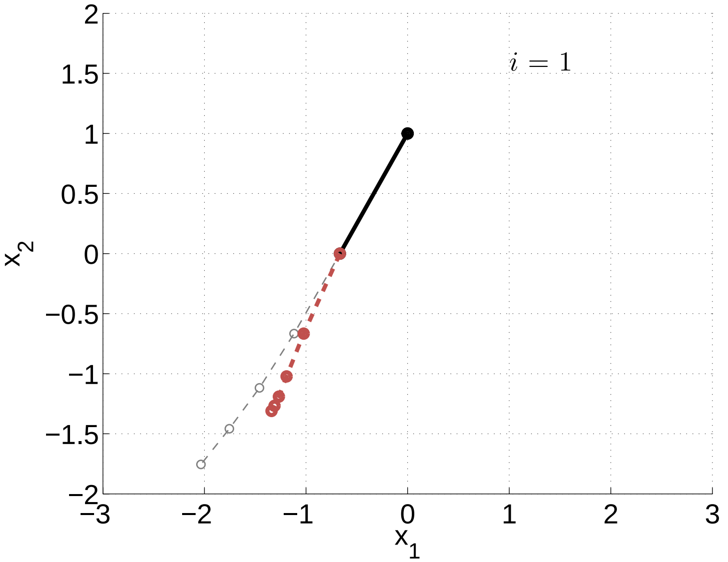


Example: Short horizon $N = 5$



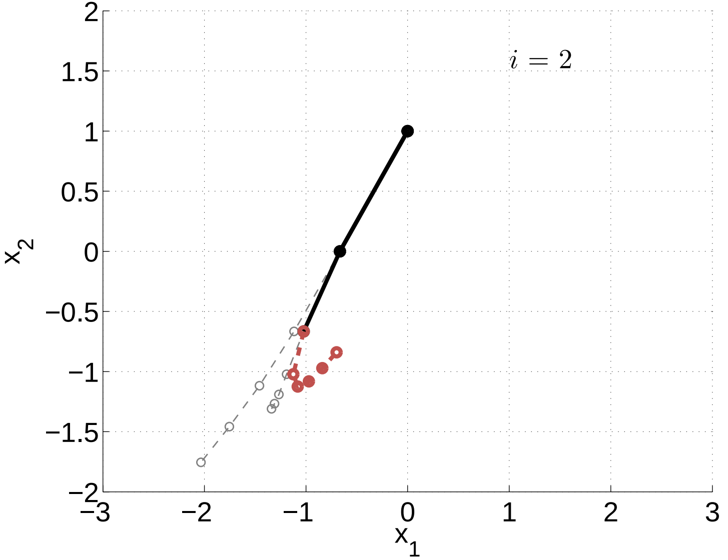
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



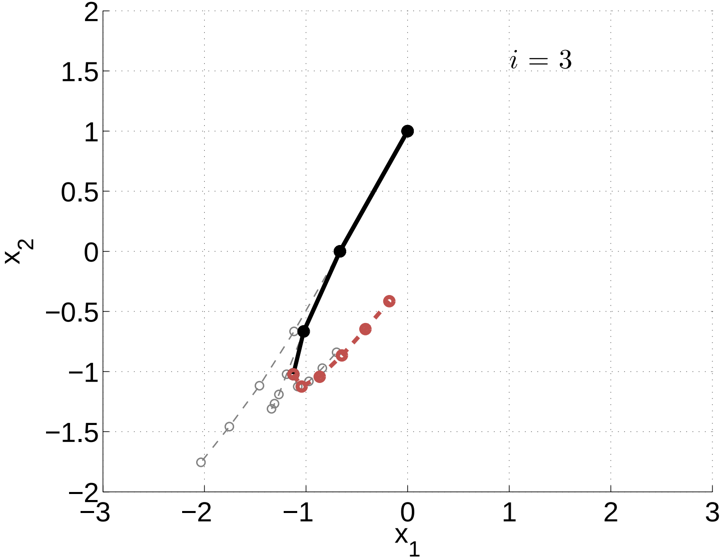
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



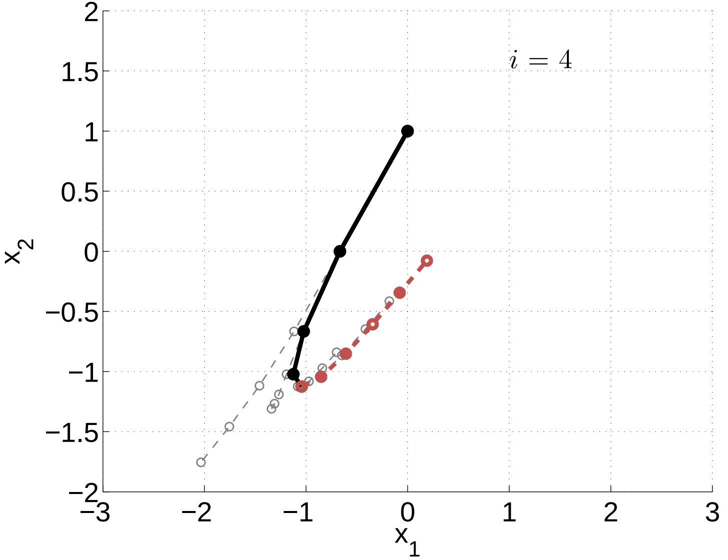
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



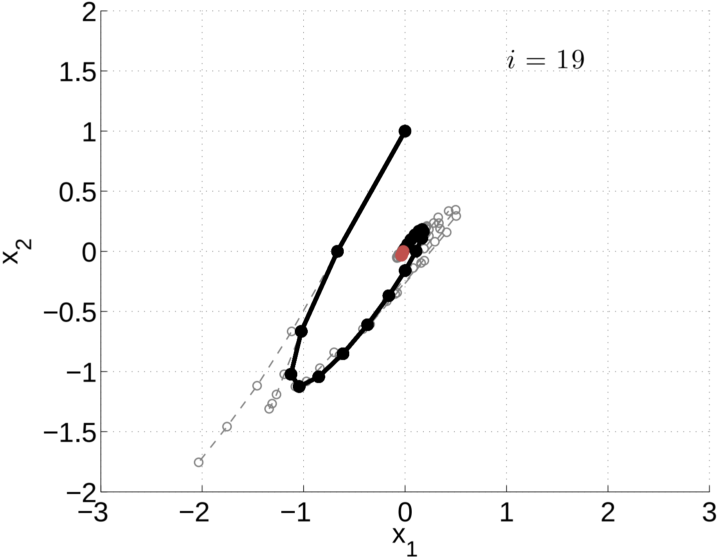
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



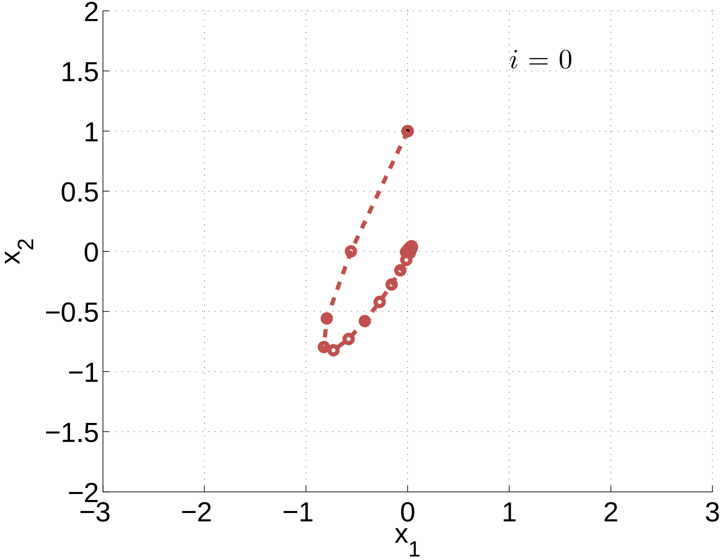
Short horizon: Prediction and closed-loop response differ.

Example: Short horizon $N = 5$



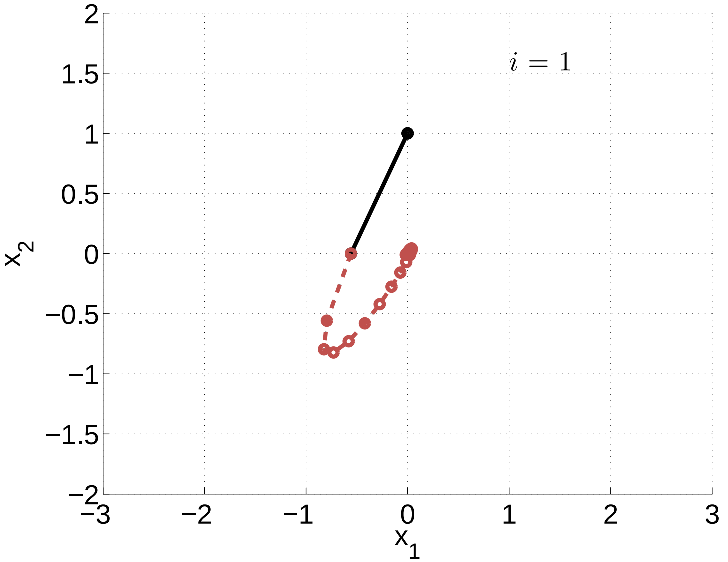
Short horizon: Prediction and closed-loop response differ.

Example: Long horizon $N = 20$



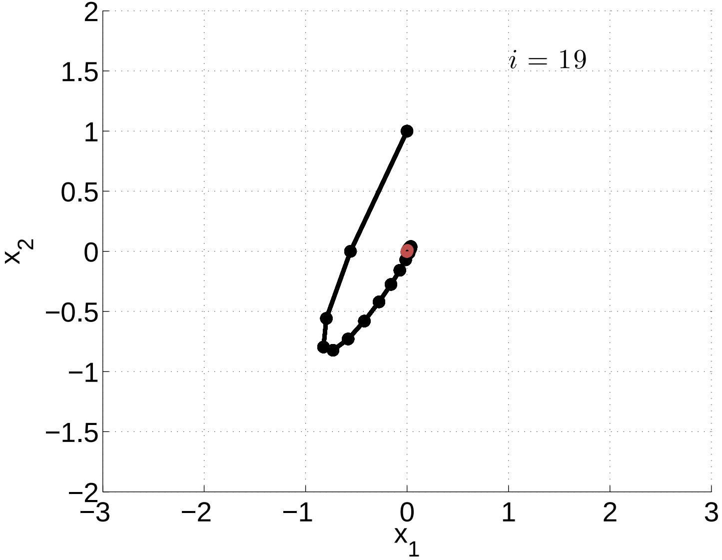
Long horizon: Prediction and closed-loop match.

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

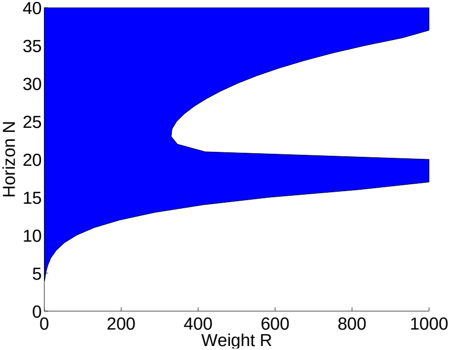
$$G(s) = \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2}$$

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



Infinite-Horizon LQR

Show that the infinite-horizon controller is nominally stable:

$$V^\star(x) := \min_{\mathbf{u}} \sum_{i=0}^{\infty} x_i^T Q x_i + u_i^T R u_i$$
$$\text{s.t. } x_{i+1} = A x_i + B u_i$$

1. System must be **controllable**
   - Have input sequence that generates a **bounded** cost
2. Finite horizon LQR converges to static solution as $N \to \infty$
3. Infinite-horizon LQR is nominally stabilizing

Solving Infinite-Horizon LQR

Consider the DP iteration:

$$V_i^\star(x_i) := \min_{u_i} \; l(x_i, u_i) + V_{i+1}^\star(Ax_i + Bu_i)$$

If $V_i^\star(\cdot) = V_{i+1}^\star(\cdot)$, then $V_j^\star(\cdot) = V_{i+1}^\star(\cdot)$ for all $j \le i$.

Therefore, if we can find a function V such that

$$V^\star(x) := \min_{u} \; l(x, u) + V^\star(Ax + Bu)$$

then $V^\star(\cdot) = V_\infty^\star(\cdot)$.

This is called the Bellman equation
(The Hamilton-Jacobi-Bellman equation is the continuous time version)

Solving Infinite-Horizon LQR

Fact: $V^\star(x)$ is quadratic, $V^\star(x) = x^T P x$ for $P \succ 0$[2]

Bellman equation:

$$\begin{aligned} V(x) &= \min_u x^T Q x + u^T R u + V(Ax + Bu) \\ x^T P x &= \min_u x^T Q x + u^T R u + (Ax + Bu)^T P (Ax + Bu) \end{aligned}$$

minimizing gives $u^\star = -(R + B^T P B)^{-1} B^T P A x$, giving

$$\begin{aligned} x^T P x &= x^T Q x + {u^\star}^T R u^\star + (Ax + Bu^\star)^T P (Ax + Bu^\star) \\ x^T P x &= x^T (Q + A^T P A - A^T P B (R + B^T P B)^{-1} B^T P A) x \end{aligned}$$

[2] Reference here

Infinite-Horizon LQR

This must hold for all x, so P must satisfy the discrete-time algebraic Riccati equation (DARE)

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

The optimal input is the constant state feedback

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

Lyapunov Function for LQR-Controlled System

Lemma: Lyapunov function for LQR

The optimal value function $V^\star(x) = x^T P x$ is a Lyapunov function for the system $x^+ = (A + BK)x$ where $K = -(R + B^T P B)^{-1} B^T P A$ and P solves

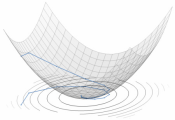
$$P = Q + A^T P A - A^T P B (R + B^T P B)^{-1} B^T P A$$

for some $Q \succeq 0$, $R \succ 0$.

Lyapunov function

A continuous function $V : \mathbb{R}^n \to \mathbb{R}_+$ is called a (asymptotic) **Lyapunov function** for the system $x^+ = f(x)$, if

- $\|x\| \to \infty \Rightarrow V(x) \to \infty$
- $V(0) = 0$ and $V(x) > 0 \ \forall x \in \mathbb{R}^n \setminus \{0\}$
- $V(f(x)) < V(x) \ \forall x \in \mathbb{R}^n \setminus \{0\}$



Lyapunov Function for LQR-Controlled System

Lemma: Lyapunov function for LQR

The optimal value function $V^\star(x) = x^T P x$ is a Lyapunov function for the system $x^+ = (A + BK)x$ where $K = -(R + B^T P B)^{-1} B^T P A$ and P solves

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

for some $Q \succeq 0$, $R \succ 0$.

$P \succ 0$ gives the first two requirements.

$$V^\star(x_0) = {x_0}^T P x_0 = \sum_{i=0}^{\infty} {x_i}^T (Q + K^T R K) x_i$$

Consider the value of $V^\star(x_1)$

$$\begin{aligned} V^\star(x_1) = V^\star((A + BK)x_0) &= \sum_{i=1}^{\infty} {x_i}^T (Q + K^T R K) x_i \\ &= V^\star(x_0) - {x_0}^T (Q + K^T R K) x_0 < V^\star(x_0) \end{aligned}$$

Optimal Control: Recap

Goal: Control law to minimize relative 'energy' of input and output signals

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.

Outline

1. Recap
    - Receding Horizon Control
    - Modeling for MPC
    - Lyapunov Functions

2. Linear quadratic regulator
    - Computation of LQR Controllers
    - Stability of LQR Controllers

3. Summary of Exercise Session

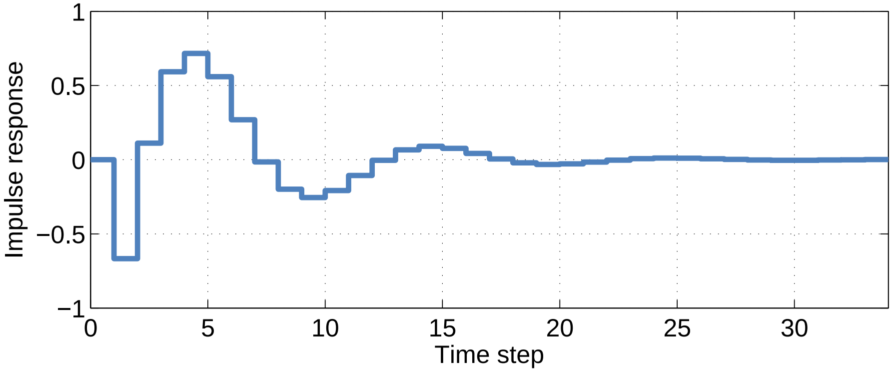
Exercise Session #1

Consider the discrete-time LTI system defined by

$$x_{i+1} = Ax_i + Bu_i \qquad\qquad y_i = Cx_i$$

with

$$A = \begin{pmatrix} 4/3 & -2/3 \\ 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad C = \begin{pmatrix} -2/3 \\ 1 \end{pmatrix}$$



Exercise Session #1

Exercises:

1. Computation of finite-horizon LQR control laws.
   (Use either dynamic programming, or least-squares optimization)

2. Investigate relationship between stability and horizon length.
   (Plot the predictions, and compare to the closed-loop trajectories.)

3. Compare your finite-horizon controller to Matlab's infinite-horizon one.

You may find slides 2-30, 2-33, 2-34 and 2-36 useful.

The matlab command `kron` is useful if you choose the least-squares optimization formulation.