AA203
Optimal and Learning-based Control

Course overview; intro to nonlinear optimization

Stanford University

Autonomous Systems Lab

Course mechanics

Teaching team:

- Instructors: Marco Pavone (OH: Tue 1pm - 2pm) and Daniele Gammelli (OH: Thur 1:30pm – 2:30pm)
- CAs: Matt Foutter and Daniel Morton (OH: TBD)

Logistics:

- Lecture slides, homework assignments: http://asl.stanford.edu/aa203/
- Lecture recordings, announcements: https://canvas.stanford.edu/courses/188274
- Discussion forum: https://edstem.org/us/courses/57727/
- Homework submission: https://www.gradescope.com/courses/760531
- For urgent questions: aa203-spr2324-staff@lists.stanford.edu

Course requirements

- Homework: there will be a total of four graded problem sets
  - Mixture of theory and implementation (Python)
- Final project: details on the course website
  - Open-ended, groups of (up to) 3 people
- Grading:
  - Homework: 60% (15% per HW)
  - Final project: 40% (5% proposal, 10% midterm report, 25% final report)
  - Ed Discussion: bonus up to 5%, 0.5% per endorsed post
- Late day policy: 6 total, maximum of 3 on any given assignment

Course material

- Course notes: an evolving set of partial course notes is available at https://github.com/StanfordASL/AA203-Notes
- Recitations: Friday recitations (F 9:00am-11:30am, weeks 1-4) led by the CAs covering relevant tools (computational and mathematical)
- Textbooks that may be valuable for context or further reference are listed in the syllabus

Prerequisites

- Familiarity with a standard undergraduate engineering mathematics curriculum (e.g., CME100-106; vector calculus, ordinary differential equations, introductory probability theory)
- Strong familiarity with linear algebra (e.g., EE263 or CME200)
- Nice-to-have: a course in optimization (e.g., EE364A, CME307, CS269O, AA222)
- To get the most out of this class, at least one of:
  - A course in machine learning (e.g., CS229, CS230, CS231N)
    or
  - A course in control (e.g., ENGR105, ENGR205, AA212)

Homework 0 (ungraded) is out now to help you gauge your preparedness

Caveats

- Arguably, this class aims for “breadth over depth”
  - Past students have found self-study of the details necessary
- This class is quite challenging/demanding
  - Past students have noted that project progress is difficult to pace with HWs
- Projects focused on learning-based control may require some self-study before the relevant lectures (talk to the teaching staff for pointers)

Today's Outline

1. Context and course goals
2. Problem formulation for optimal control
3. Introduction to non-linear optimization

Today's Outline

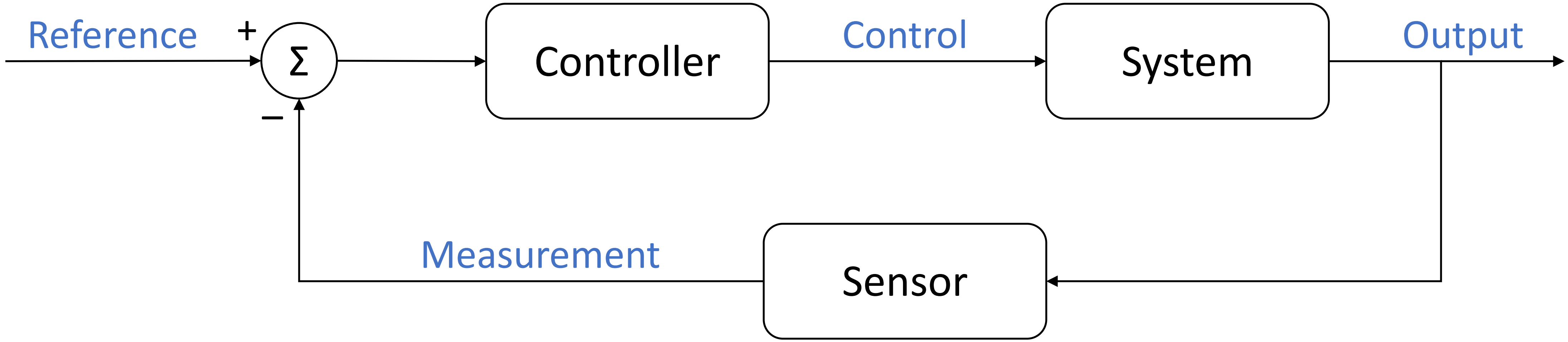
1. Context and course goals

2. Problem formulation for optimal control

3. Introduction to non-linear optimization

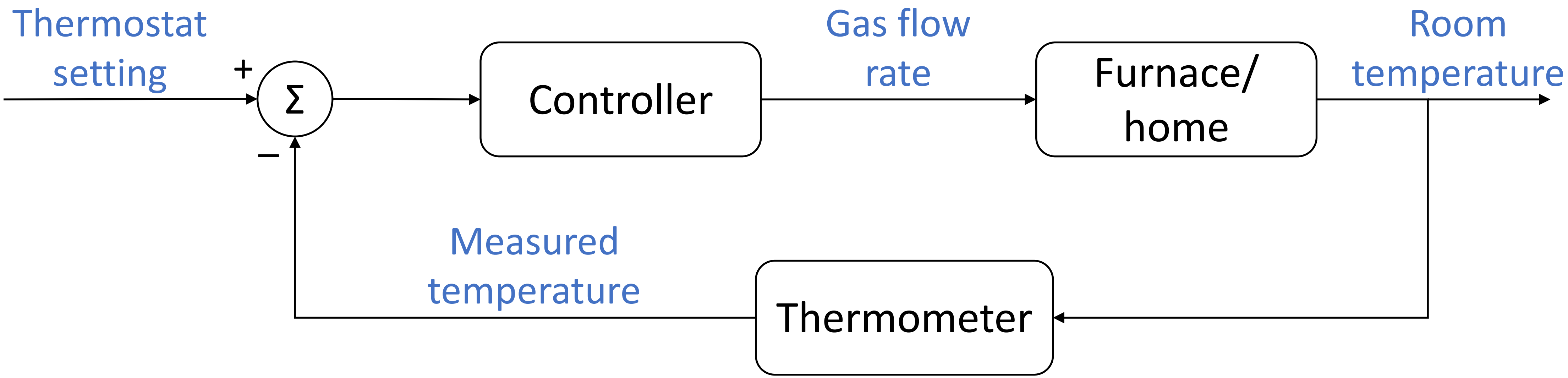
Feedback control

- Tracking a reference signal



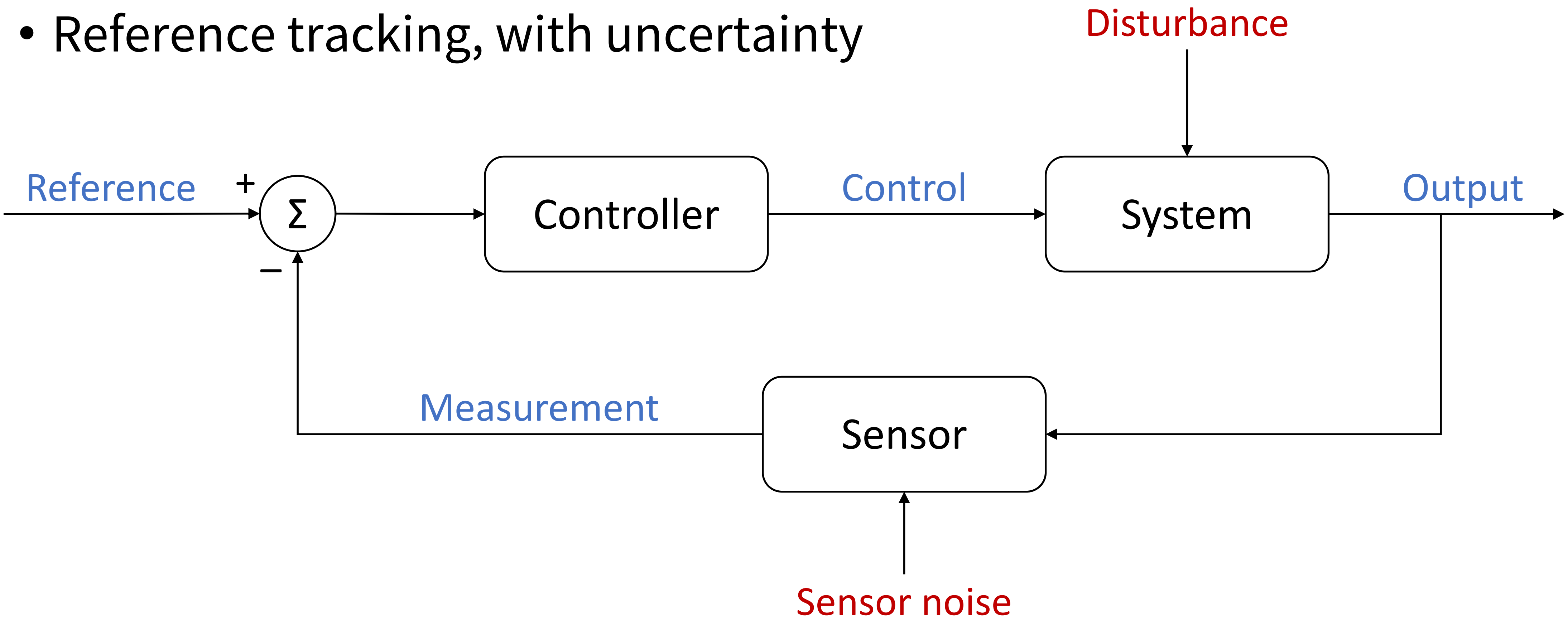
Feedback control

- Tracking a reference signal



Feedback control

- Reference tracking, with uncertainty



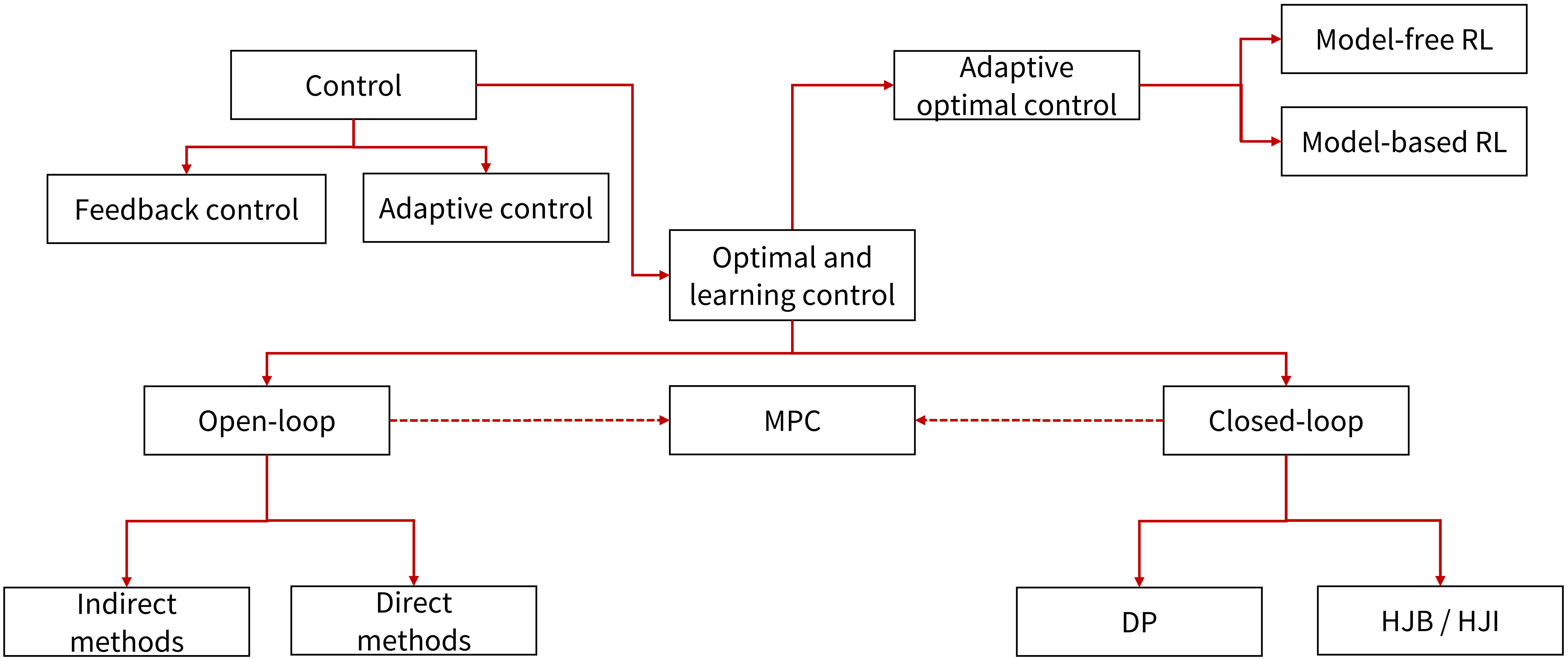
Feedback control desiderata

- Stability: multiple notions; loosely system output is “under control”
- Tracking: the output should track the reference “as closely as possible”
- Disturbance rejection: the output should be “as insensitive as possible” to disturbances/noise
- Robustness: controller should still perform well up to “some degree of” model misspecification

What’s missing?

- Performance: mathematical quantification of the above desiderata, and providing a control that best realizes the tradeoffs between them
- Planning: providing an appropriate reference trajectory for the controller to track (particularly nontrivial, e.g., when controlling mobile robots)
- Learning: a controller that adapts to an initially unknown, or possibly time-varying system

Course overview



Course goals

To learn the *theoretical* and *implementation* aspects of main techniques in optimal and learning-based control

Course goals

To learn the *theoretical* and *implementation* aspects of main techniques in optimal and learning-based control

To provide a *unified framework and context* for understanding and relating these techniques to each other

Today’s Outline

1. Context and course goals

2. Problem formulation for optimal control

3. Introduction to non-linear optimization

Problem formulation

- Mathematical description of the system to be controlled
- Statement of the constraints
- Specification of a performance criterion

Mathematical model

$$\dot{x}_1(t) = f_1(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), u_2(t), \ldots, u_m(t), t)$$
$$\dot{x}_2(t) = f_2(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), u_2(t), \ldots, u_m(t), t)$$
$$\vdots \qquad\qquad \vdots$$
$$\dot{x}_n(t) = f_n(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), u_2(t), \ldots, u_m(t), t)$$

Where

- $x_1(t), x_2(t), \ldots, x_n(t)$ are the state variables
- $u_1(t), u_2(t), \ldots, u_m(t)$ are the control inputs

Mathematical model

In compact form

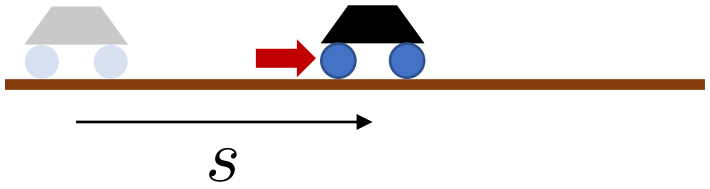
$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)$$

- a history of control input values during the interval $[t_0, t_f]$ is called a *control history*
- a history of state values during the interval $[t_0, t_f]$ is called a *state trajectory*

Illustrative example: double integrator

- Double integrator: point mass under controlled acceleration

$$\ddot{s}(t) = a(t)$$

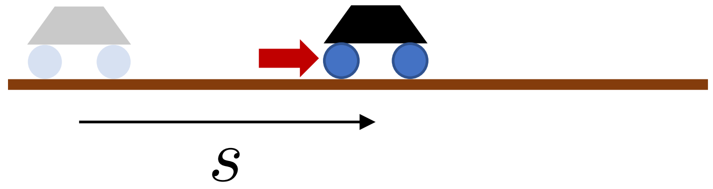


Illustrative example: double integrator

- Double integrator: point mass under controlled acceleration

$$\ddot{s}(t) = a(t)$$

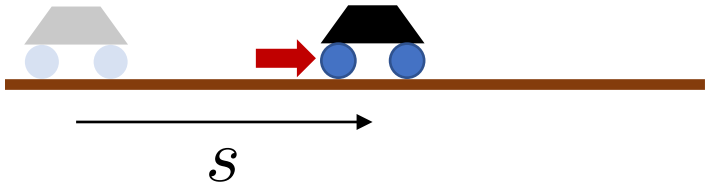
$$\begin{bmatrix} \dot{s} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v \\ a \end{bmatrix}$$



Illustrative example: double integrator

- Double integrator: point mass under controlled acceleration

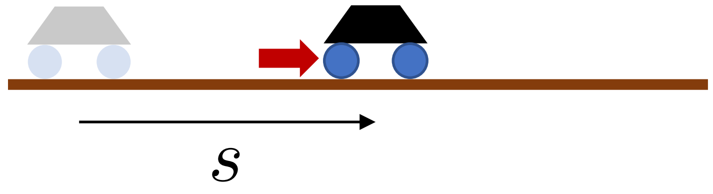
$$\begin{bmatrix} \dot{s} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} s \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} [a]$$



Illustrative example: double integrator

- Double integrator: point mass under controlled acceleration

$$\begin{bmatrix} \dot{s} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} s \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} [a]$$



$$\dot{\mathbf{x}}(t) = A \quad \mathbf{x}(t) + B \quad \mathbf{u}(t)$$

LTI system

Constraints

- initial and final conditions (boundary conditions)

$$\mathbf{x}(t_0) = \mathbf{x}_0, \qquad \mathbf{x}(t_f) = \mathbf{x}_f$$

- constraints on state trajectories

$$\underline{X} \leq \mathbf{x}(t) \leq \overline{X}$$

- control authority

$$\underline{U} \leq \mathbf{u}(t) \leq \overline{U}$$

- and many more…

Constraints

- A control history which satisfies the control constraints during the entire time interval $[t_0, t_f]$ is called an admissible control
- A state trajectory which satisfies the state variable constraints during the entire time interval $[t_0, t_f]$ is called an admissible trajectory

Performance measure

$$J = h(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t)\, dt$$

- h (terminal cost) and g (stagewise/running cost) are scalar functions
- t_f may be specified or free

Optimal control problem

Find an *admissible control* $\mathbf{u}^*$ which causes the system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)$$

to follow an *admissible trajectory* $\mathbf{x}^*$ that minimizes the performance measure

$$J = h(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t)\, dt$$

Very general problem formulation!

Optimal control problem

Comments:

- minimizer $(\mathbf{x}^*, \mathbf{u}^*)$ called optimal trajectory-control pair
- existence: in general, not guaranteed
- uniqueness: optimal control may not be unique
- minimality: we are seeking a global minimum
- for maximization, we rewrite the problem as $\min_{\mathbf{u}} -J$

Forms of optimal control

1. if $\mathbf{u}^* = \pi(\mathbf{x}(t), t)$, then π is called optimal control law or optimal policy (*closed-loop*)
   - important example: $\pi(\mathbf{x}(t), t) = F\,\mathbf{x}(t)$

2. if $\mathbf{u}^* = e(\mathbf{x}(t_0), t)$, then the optimal control is *open-loop*
   - optimal *only* for a particular initial state value

Discrete-time formulation

- System: $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, k), \quad k = 0, \dots, N-1$
- Control constraints: $\mathbf{u}_k \in U$
- Cost:

$$J(\mathbf{x}_0; \boldsymbol{u}_0, \dots, \boldsymbol{u}_{N-1}) = h_N(\mathbf{x}_N) + \sum_{k=0}^{N-1} g_k(\mathbf{x}_k, \mathbf{u}_k, k)$$

- Decision-making problem:

$$J^*(\mathbf{x}_0) = \min_{\mathbf{u}_k \in U,\, k=0,\dots,N-1} J(\mathbf{x}_0; \boldsymbol{u}_0, \dots, \boldsymbol{u}_{N-1})$$

Discrete-time formulation

- System: $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, k), \quad k = 0, \dots, N-1$
- Control constraints: $\mathbf{u}_k \in U$
- Cost:

$$J(\mathbf{x}_0; \boldsymbol{u}_0, \dots, \boldsymbol{u}_{N-1}) = h_N(\mathbf{x}_N) + \sum_{k=0}^{N-1} g_k(\mathbf{x}_k, \mathbf{u}_k, k)$$

- Decision-making problem:

$$J^*(\mathbf{x}_0) = \min_{\mathbf{u}_k \in U, k=0,\dots,N-1} J(\mathbf{x}_0; \boldsymbol{u}_0, \dots, \boldsymbol{u}_{N-1})$$

<u>Extension to stochastic setting will be covered later in the course</u>

Today’s Outline

1. Context and course goals

2. Problem formulation for optimal control

3. Introduction to non-linear optimization

Non-linear optimization

Unconstrained non-linear program

$$\min_{\mathbf{x} \in \mathbb{R}^n} \; f(\mathbf{x})$$

- f usually assumed continuously differentiable (and often twice continuously differentiable)

Local and global minima

- A vector $\mathbf{x}^*$ is said an unconstrained *local* minimum if $\exists \epsilon > 0$ such that

$$f(\mathbf{x}^*) \leq f(\mathbf{x}), \qquad \forall \mathbf{x} \,\Big|\, \|\mathbf{x} - \mathbf{x}^*\| < \epsilon$$

- A vector $\mathbf{x}^*$ is said an unconstrained *global* minimum if

$$f(\mathbf{x}^*) \leq f(\mathbf{x}), \qquad \forall \mathbf{x} \in \mathbb{R}^n$$

- $\mathbf{x}^*$ is a strict local/global minimum if the inequality is strict

Necessary conditions for optimality

Key idea: compare cost of a vector with cost of its close neighbors

- Assume $f \in C^1$, by using Taylor series expansion

$$f(\mathbf{x}^* + \Delta\mathbf{x}) - f(\mathbf{x}^*) \approx \nabla f(\mathbf{x}^*)' \Delta\mathbf{x}$$

- If $f \in C^2$

$$f(\mathbf{x}^* + \Delta\mathbf{x}) - f(\mathbf{x}^*) \approx \nabla f(\mathbf{x}^*)' \Delta\mathbf{x} + \frac{1}{2} \Delta\mathbf{x}' \nabla^2 f(\mathbf{x}^*) \Delta\mathbf{x}$$

Necessary conditions for optimality

- We expect that if $\mathbf{x}^*$ is an unconstrained local minimum, the first order cost variation due to a small variation $\Delta\mathbf{x}$ is nonnegative, i.e.,

$$\nabla f(\mathbf{x}^*)'\Delta\mathbf{x} = \sum_{i=1}^{n} \frac{\partial f(\mathbf{x}^*)}{\partial x_i}\Delta x_i \geq 0$$

- By taking $\Delta\mathbf{x}$ to be positive and negative multiples of the unit coordinate vectors, we obtain conditions of the type

$$\frac{\partial f(\mathbf{x}^*)}{\partial x_i} \geq 0, \quad \text{and} \quad \frac{\partial f(\mathbf{x}^*)}{\partial x_i} \leq 0$$

- Equivalently we have the necessary condition

$$\boxed{\nabla f(\mathbf{x}^*) = 0} \qquad (\mathbf{x}^* \text{ is said a stationary point})$$

Necessary conditions for optimality

- Of course, also the second order cost variation due to a small variation $\Delta\mathbf{x}$ must be non-negative

$$\nabla f(\mathbf{x}^*)'\Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}'\nabla^2 f(\mathbf{x}^*)\Delta\mathbf{x} \geq 0$$

- Since $\nabla f(\mathbf{x}^*)'\Delta\mathbf{x}=0$, we obtain $\Delta\mathbf{x}'\nabla^2 f(\mathbf{x}^*)\Delta\mathbf{x} \geq 0$. Hence

$$\boxed{\nabla^2 f(\mathbf{x}^*) \text{ has to be positive semidefinite}}$$

NOC – formal

Theorem: NOC

Let $\mathbf{x}^*$ be an unconstrained local minimum of $f: \mathbb{R}^n \mapsto \mathbb{R}$ and assume that f is C^1 in an open set S containing $\mathbf{x}^*$. Then

$$\nabla f(\mathbf{x}^*) = 0 \qquad \text{(first order NOC)}$$

If in addition $f \in C^2$ within S,

$$\nabla^2 f(\mathbf{x}^*) \text{ positive semidefinite} \qquad \text{(second order NOC)}$$

SOC

- Assume that $\mathbf{x}^*$ satisfies the first order NOC

$$\nabla f(\mathbf{x}^*) = 0$$

- and also assume that the second order NOC is strengthened to

$$\nabla^2 f(\mathbf{x}^*) \quad \text{positive } \textit{definite}$$

- Then, for all $\Delta\mathbf{x} \neq 0$, $\Delta\mathbf{x}'\nabla^2 f(\mathbf{x}^*)\Delta\mathbf{x} > 0$. Hence, f tends to increase *strictly* with small excursions from $\mathbf{x}^*$, suggesting SOC…

SOC

Theorem: SOC

Let $f: \mathbb{R}^n \mapsto \mathbb{R}$ be C^2 in an open set S. Suppose that a vector $\mathbf{x}^* \in S$ satisfies the conditions

$$\nabla f(\mathbf{x}^*) = 0 \qquad \text{and} \qquad \nabla^2 f(\mathbf{x}^*) \text{ positive definite}$$

Then $\mathbf{x}^*$ is a strict unconstrained local minimum of f

Special case: convex optimization

A subset C of $\mathbb{R}^n$ is called convex if

$$\alpha \mathbf{x} + (1-\alpha)\mathbf{y} \in C, \quad \forall \mathbf{x}, \mathbf{y} \in C, \forall \alpha \in [0,\ 1]$$

Let C be convex. A function $f: C \to \mathbb{R}$ is called convex if

$$f(\alpha \mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y})$$

Let $f: C \to \mathbb{R}$ be a convex function over a convex set C

- A local minimum of f over C is also a global minimum over C. If in addition f is strictly convex, then there exists at most one global minimum of f
- If f is in C^1 and convex, and the set C is open, $\nabla f(\mathbf{x}^*) = 0$ is a necessary and sufficient condition for a vector $\mathbf{x}^* \in C$ to be a global minimum over C

Discussion

- Optimality conditions are important to filter candidates for global minima
- They often provide the basis for the design and analysis of optimization algorithms
- They can be used for sensitivity analysis

Next lecture

Computational methods for non-linear optimization;
constrained optimization