Trajectory Design for Space Systems

EN.530.626 (Fall 2025)

Lecture 8

Instructor: Prof. Abhishek Cauligi
Course Assistant 1: Arnab Chatterjee
Course Assistant 2: Mark Gonzales

Class review

- So far: smooth constrained optimization

$$\min_x f(x) \qquad\qquad f \in \mathscr{C}^2$$

$$\texttt{subject to:} \quad h_i(x) = 0, \quad i = 1,\ldots,m \qquad h_i \in \mathscr{C}^2$$

$$g_i(x) = 0, \quad i = 0,\ldots,p \qquad g_i \in \mathscr{C}^2$$

- Allows us to leverage Newton method-style approaches for solving problem

Class review

Quadratic programs

- One particular formulation we have returned to has been *convex* quadratic programs

$$\min_x \frac{1}{2}x^T P x + p^T x \qquad P \in \mathbb{S}^n_+$$

$$\texttt{subject to:} \quad Ax = 0 \qquad A \in \mathbb{R}^{m \times n}$$

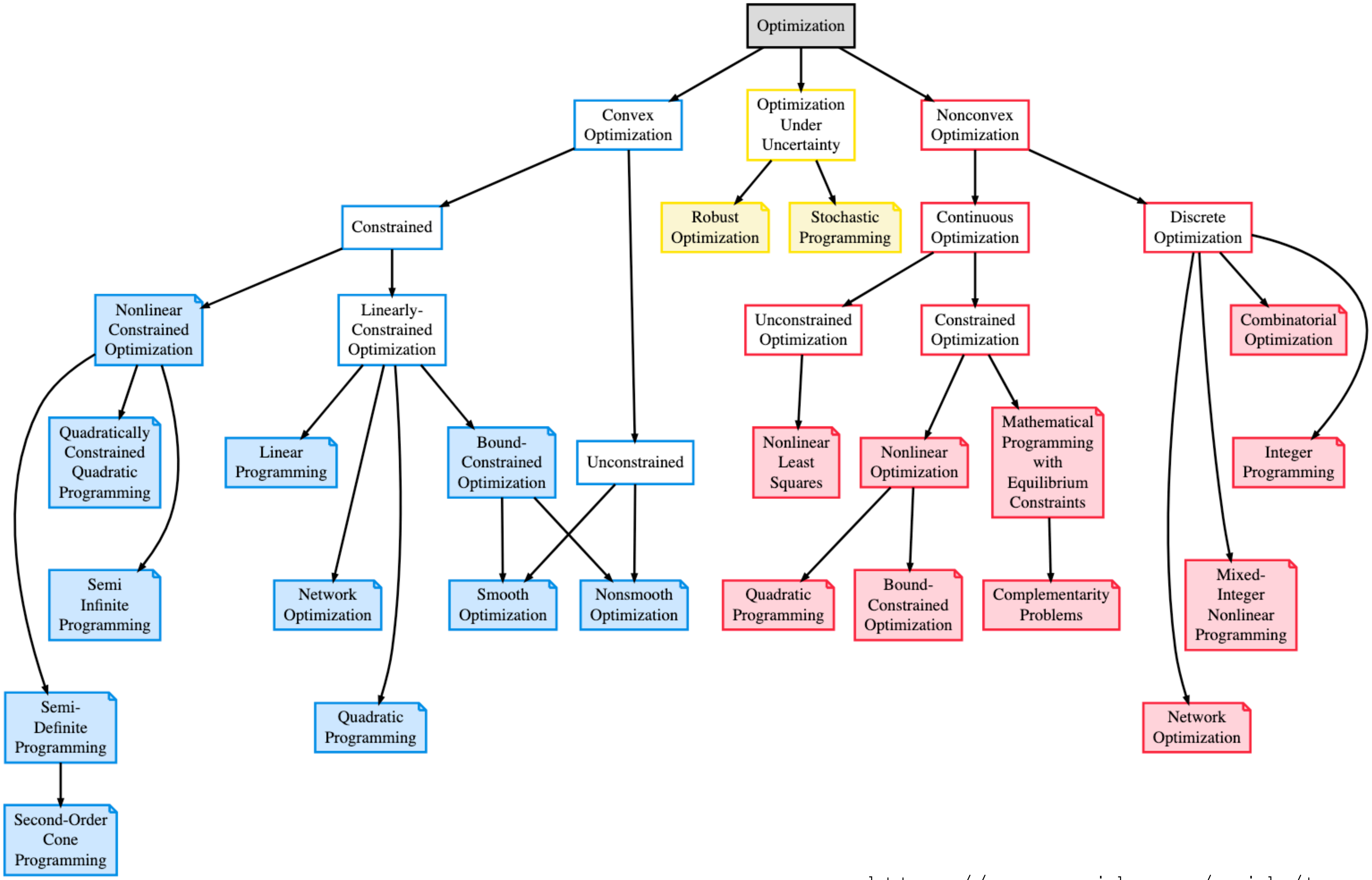
$$Gx \leq h \qquad G \in \mathbb{R}^{p \times n}$$

- Applied primal-dual interior point methods to find solutions for these

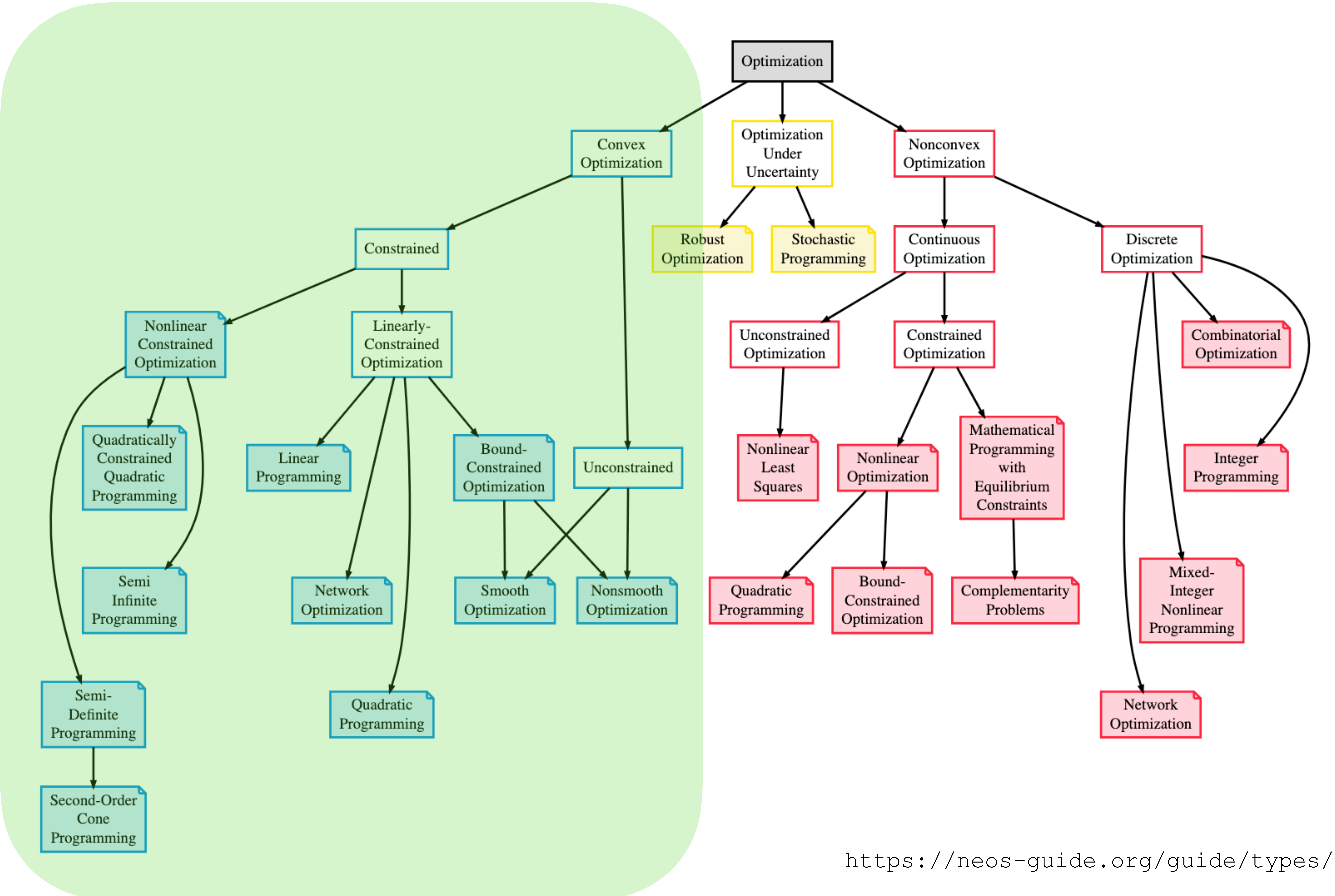
What’s next?

Recall optimization classes:

- Unconstrained vs. discrete
- Smooth vs. non smooth
- Convex vs. non-convex
- Continuous vs. discrete



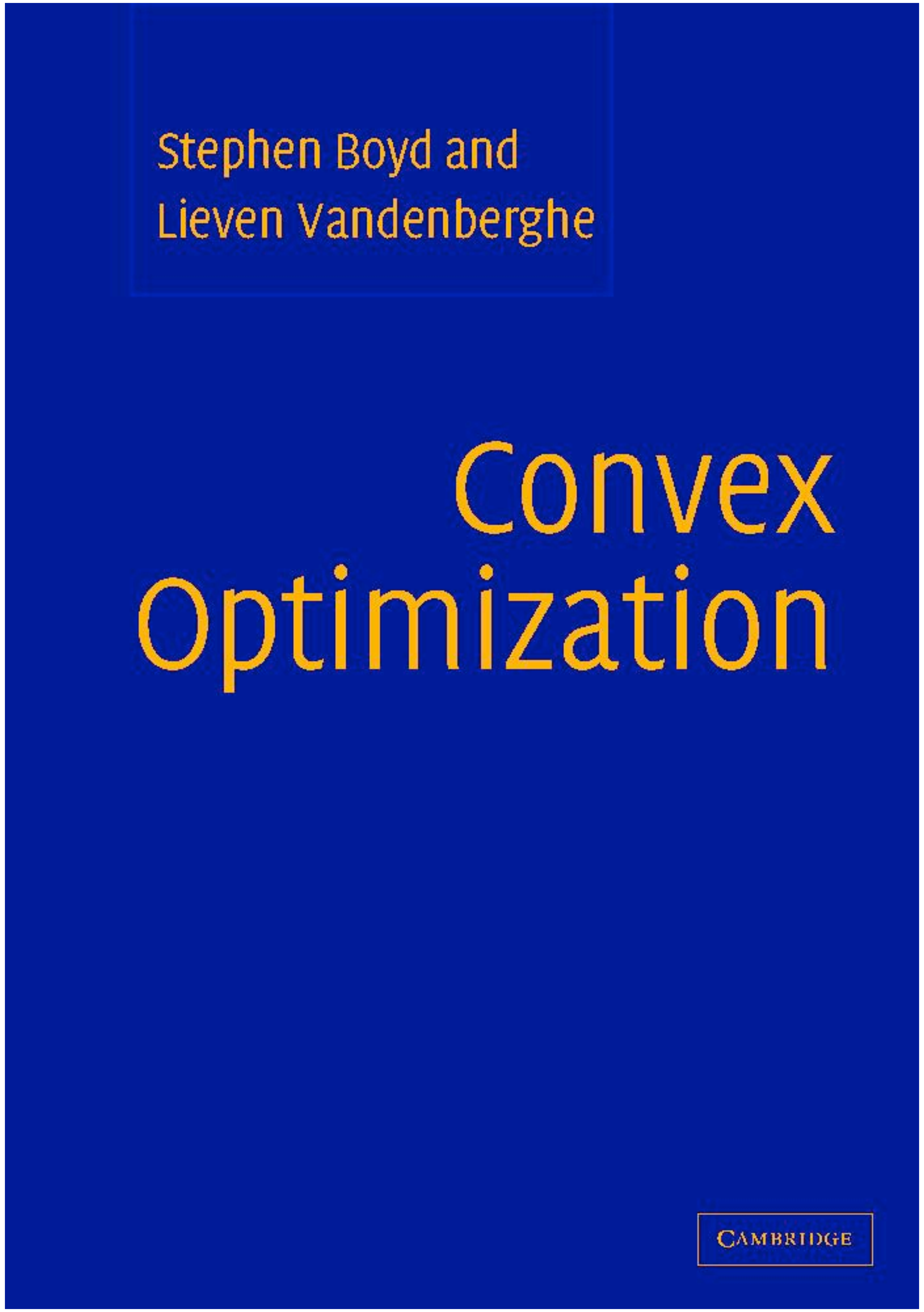
https://neos-guide.org/guide/types/



https://neos-guide.org/guide/types/

Convexity

- A whole field of optimization on its own
- Convex $\neq$ easy
- Advantages
  - (1) Enjoys strong convergence guarantees
  - (2) Local optimizer is global one



Convexity

Convex sets

- A set $\mathcal{X}$ is convex if $\forall x_1, x_2 \in \mathcal{X}$ and $\theta \in [0,1]$

$$\theta x_1 + (1 - \theta) x_2 \in \mathcal{X}$$

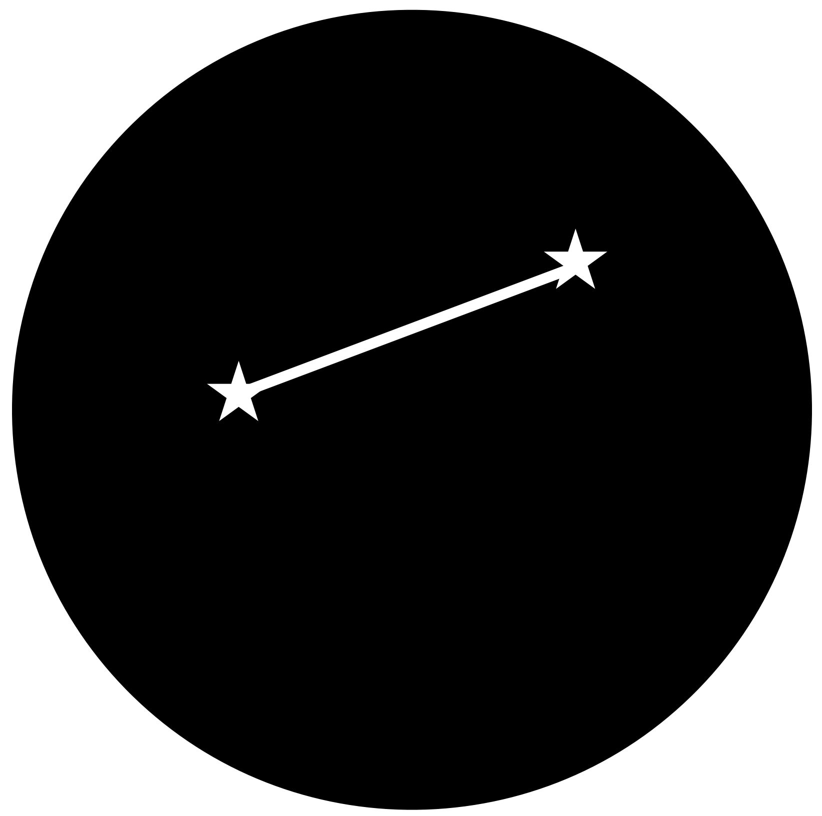
Convexity

Convex sets

- A set $\mathcal{X}$ is convex if $\forall x_1, x_2 \in \mathcal{X}$ and $\theta \in [0,1]$

$$\theta x_1 + (1-\theta)x_2 \in \mathcal{X}$$

$\|x\|_2 \leq r_{\text{dist}}$



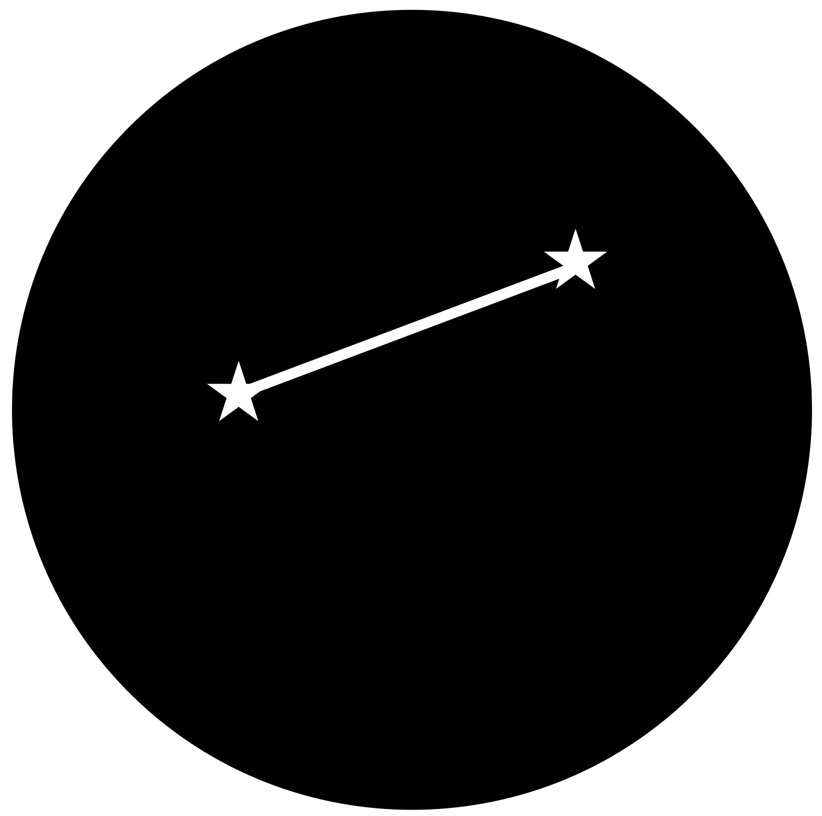
Convexity

Convex sets

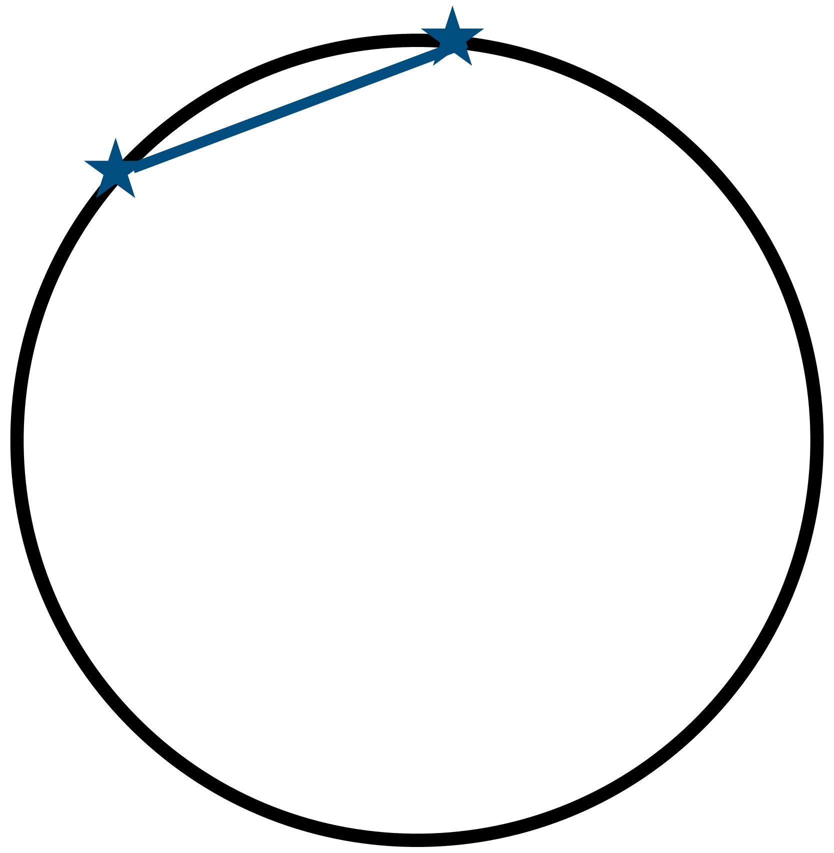
- A set $\mathcal{X}$ is convex if $\forall x_1, x_2 \in \mathcal{X}$ and $\theta \in [0,1]$

$$\theta x_1 + (1 - \theta) x_2 \in \mathcal{X}$$

$\|x\|_2 \leq r_{\text{dist}}$



$\|x\|_2 = r_{\text{dist}}$



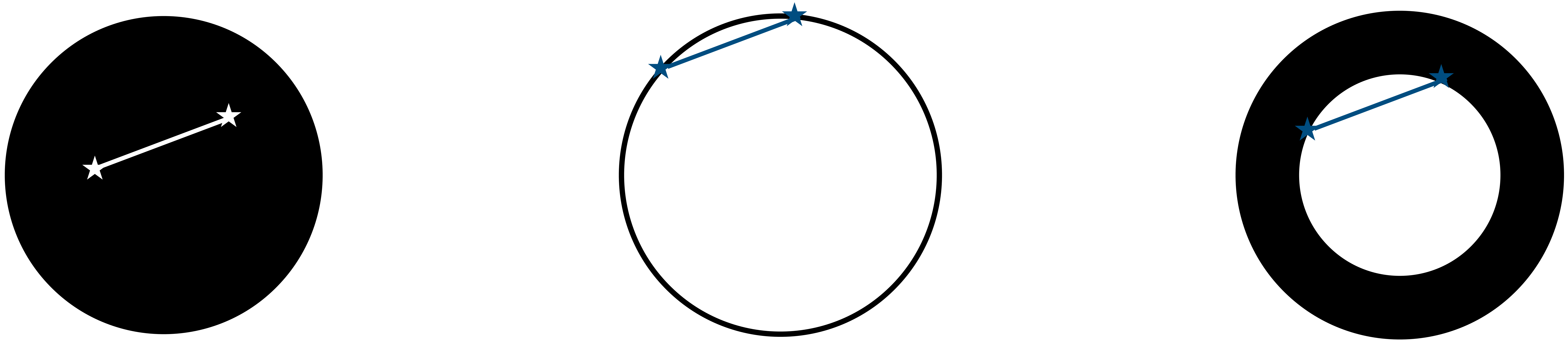
Convexity

Convex sets

- A set $\mathcal{X}$ is convex if $\forall x_1, x_2 \in \mathcal{X}$ and $\theta \in [0,1]$

$$\theta x_1 + (1-\theta)x_2 \in \mathcal{X}$$

$\|x\|_2 \leq r_{\text{dist}}$ $\quad$ $\|x\|_2 = r_{\text{dist}}$ $\quad$ $r_{\text{min}} \leq \|x\|_2 \leq r_{\text{max}}$

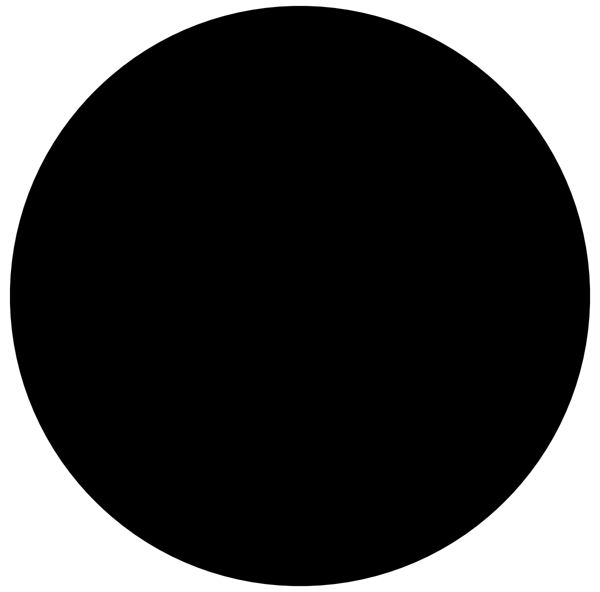


Convexity

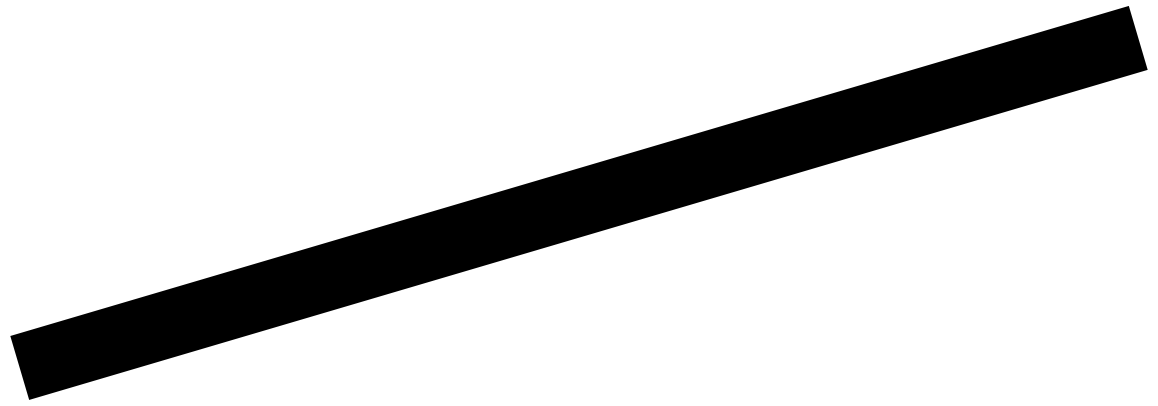
Convex sets

- The intersection of convex sets is convex

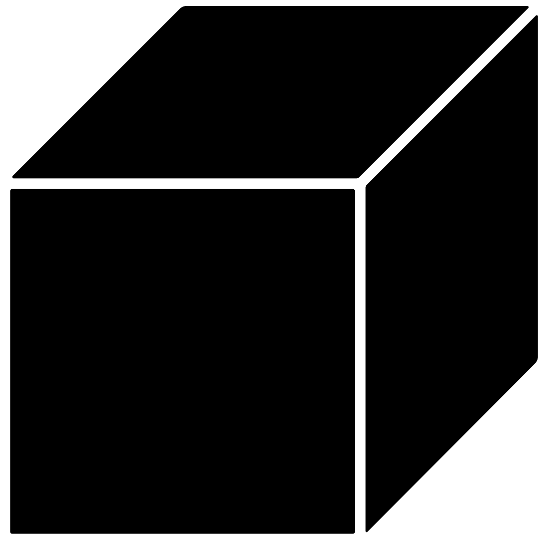
$\|x\|_2 \leq r_{\text{dist}}$



$y = c^T x + b$



$\|u\|_1 \leq u_{\text{max}}$



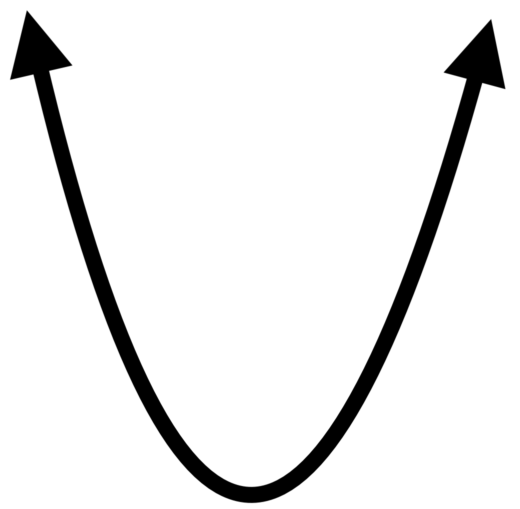
Convexity

Convex functions

- A function $f(x)$ is convex if $\forall x_1, x_2 \in \text{dom}\, f$ and $\theta \in [0,1]$

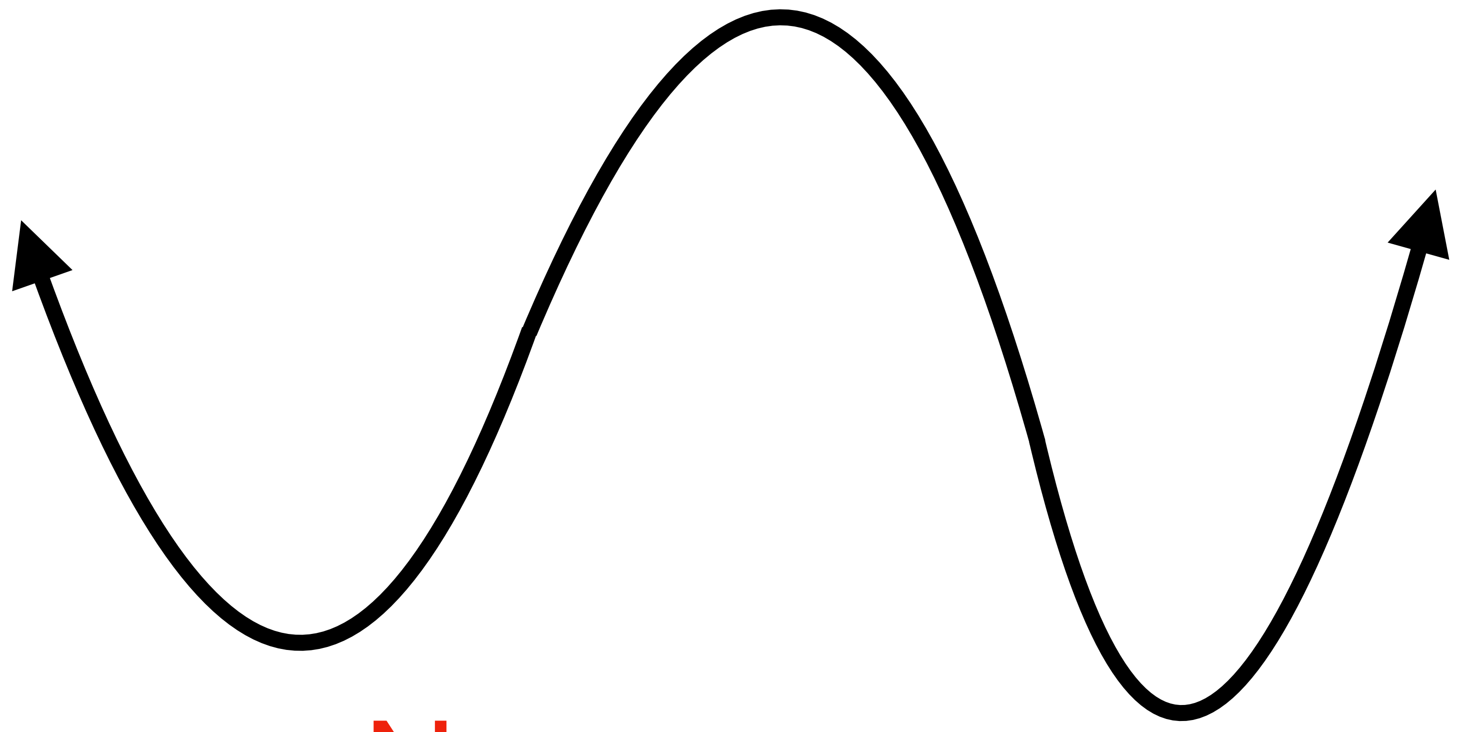
$$f(\theta x_1 + (1-\theta)x_2) \leq \theta f(x_1) + (1-\theta)f(x_2)$$

$f(x) = x^2$



Convex

$f(x) = x^4 - 2 * x^3 - 10x^2 + 7x - 17$



Non-convex

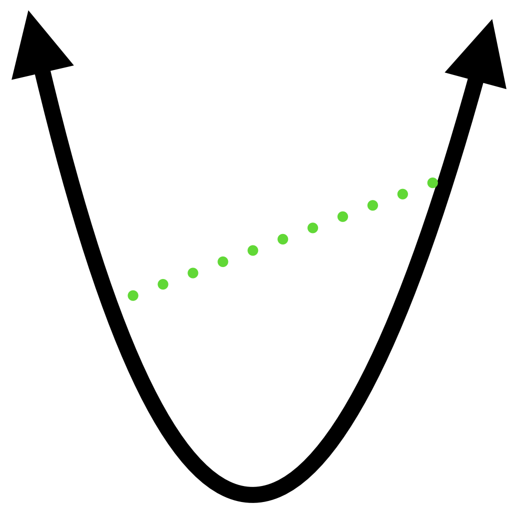
Convexity

Convex functions

- A function $f(x)$ is convex if $\forall x_1, x_2 \in \text{dom} f$ and $\theta \in [0,1]$

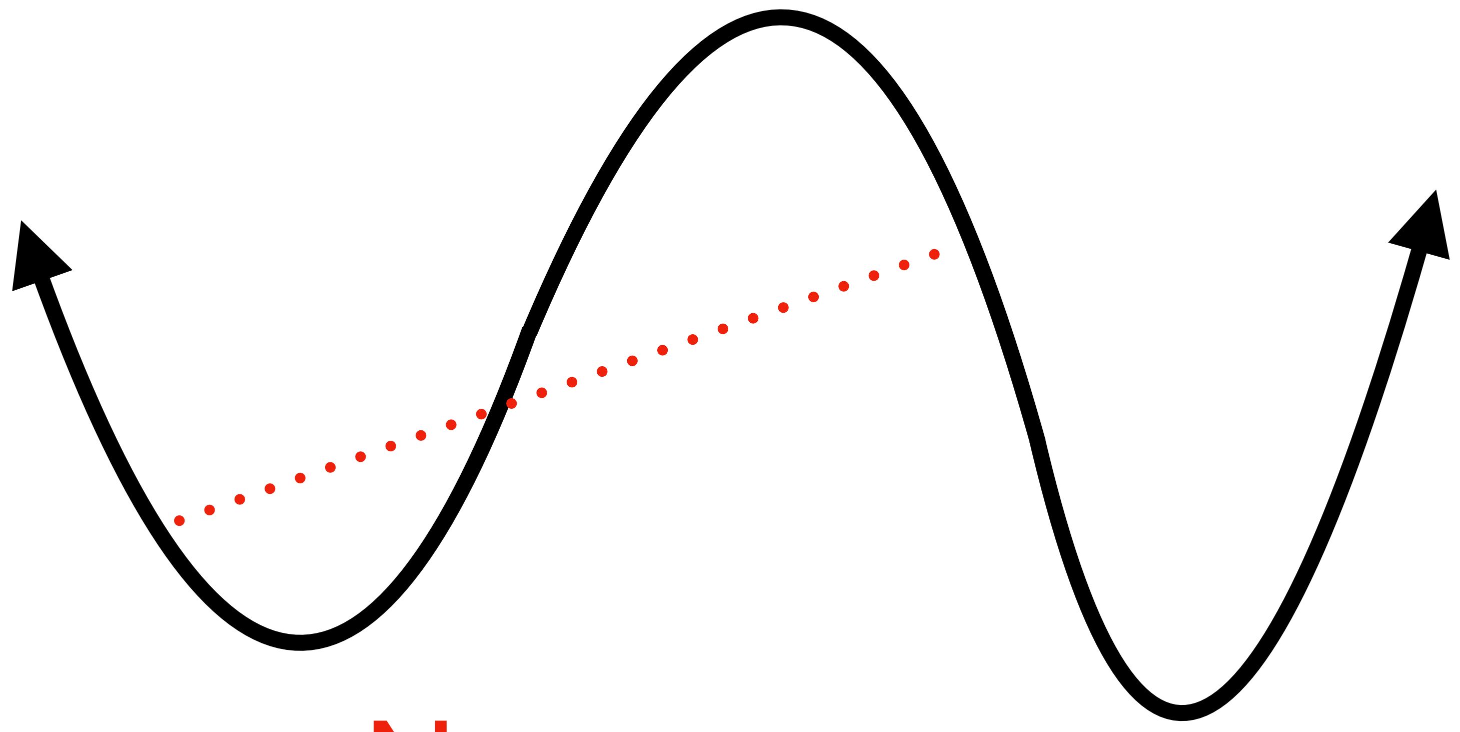
$$f(\theta x_1 + (1-\theta)x_2) \leq \theta f(x_1) + (1-\theta)f(x_2)$$

$f(x) = x^2$



Convex

$f(x) = x^4 - 2 * x^3 - 10x^2 + 7x - 17$



Non-convex

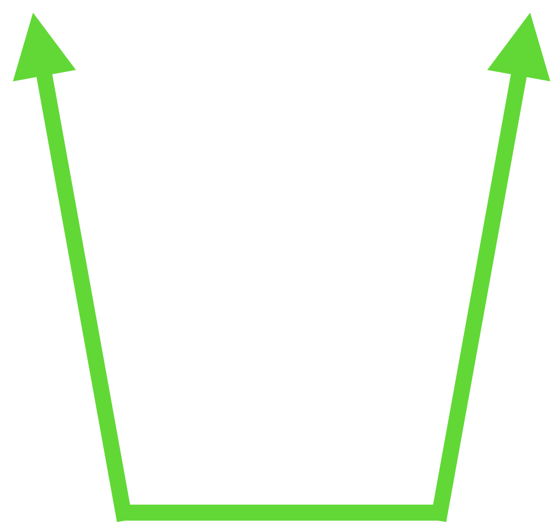
Convexity

Convex functions

- A function $f(x)$ is convex if $\forall x_1, x_2 \in \operatorname{dom} f$ and $\theta \in [0,1]$

$$f(\theta x_1 + (1-\theta)x_2) \leq \theta f(x_1) + (1-\theta) f(x_2)$$

- Note: f need not be smooth (i.e., $\nabla_x f(x)$ might not exist)



Convex

Non-convex

Convexity

Convex functions and sets

- The epigraph of a function $f : \mathbb{R}^n \to \mathbb{R}$ is:

$$\text{epi}\, f = \{(x, t)\,|\, x \in \text{dom}\, f, f(x) \leq t\}$$

- f is a convex function if and only if epi f is a convex set

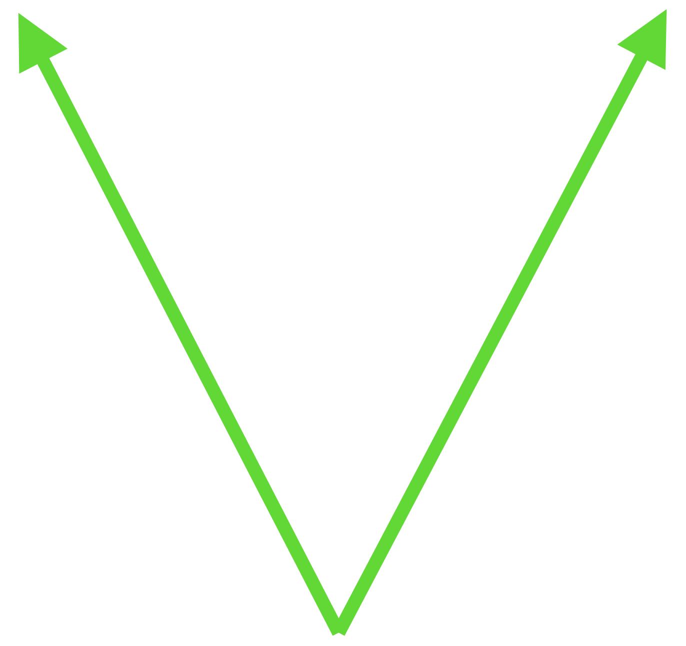
$$f \text{ is convex function} \Leftrightarrow \text{epi}\, f \text{ is convex set}$$

Convexity

Convex functions and sets

- The epigraph of a function $f : \mathbb{R}^n \to \mathbb{R}$ is:

$$\mathrm{epi}\, f = \{(x, t) \,|\, x \in \mathrm{dom}\, f, f(x) \leq t\}$$



Convexity

Convex functions and sets

- The epigraph of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is:

$$\text{epi}\, f = \{(x, t) \,|\, x \in \text{dom}\, f, f(x) \leq t\}$$

- Informally: is the space above a convex function a convex set?

Convexity

Why does this matter?

- A *local* solution to a convex problem is also the *global* solution
  - For smooth unconstrained optimization, if $f \in \mathscr{C}^2$:

$$x^* \text{ is global optimizer} \Leftrightarrow \nabla_x f(x^*) = 0$$

  - For smooth constrained optimization, if $f, h, g \in \mathscr{C}^2$:

$$x^* \text{ is global optimizer} \Leftrightarrow \nabla_x \mathscr{L}(x^*, y^*, z^*) = 0$$

Convexity

Completeness guarantees

- A function $f : \mathbb{R} \to \mathbb{R}$ is called *self-concordant* if f is convex and:

$$f'''(x) \leq 2f''(x)^{\frac{3}{2}}$$

- Loosely, "third derivative doesn't change too quickly"

Convexity

Completeness guarantees

INTERIOR POINT

POLYNOMIAL TIME METHODS

IN

CONVEX PROGRAMMING

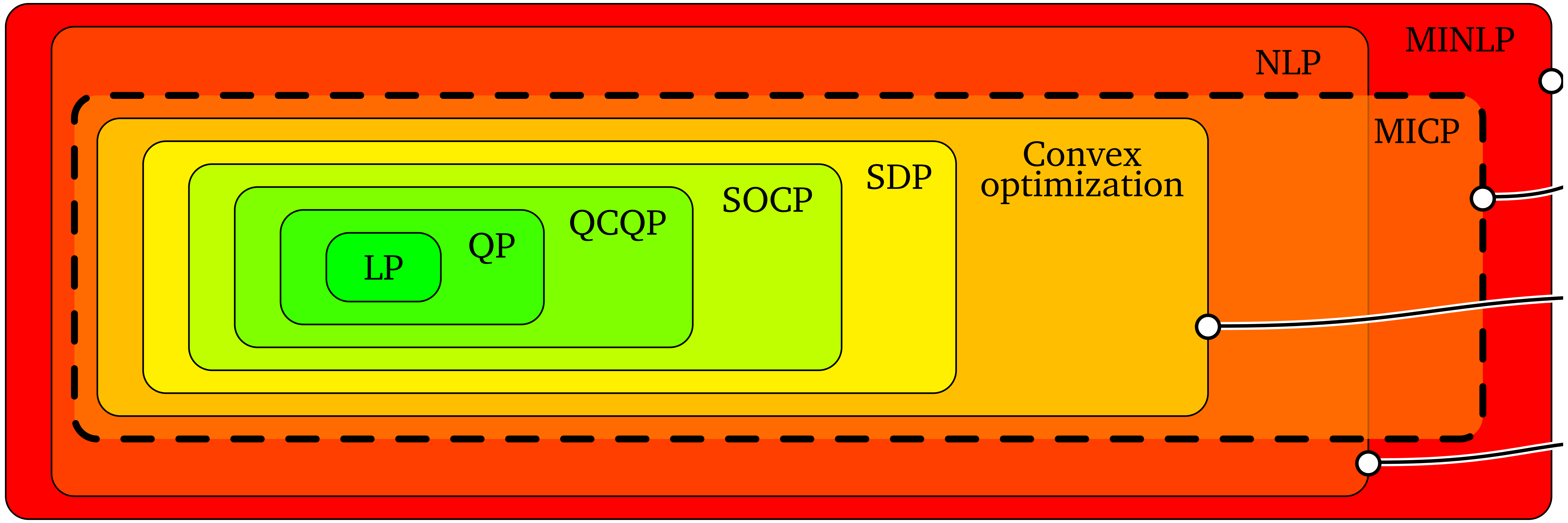
A. Nemirovski

Spring Semester 1996

Convexity

In summary

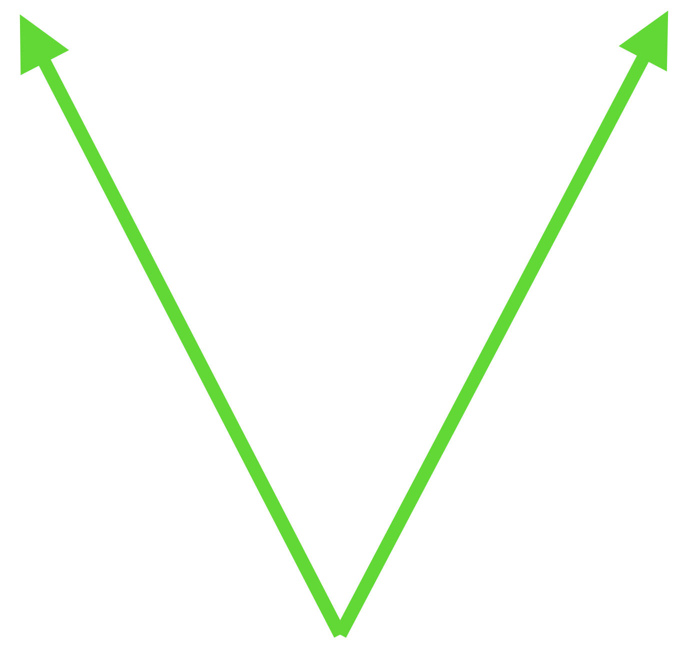
- Convexity does not mean the problem is *easy*



Convexity

In summary

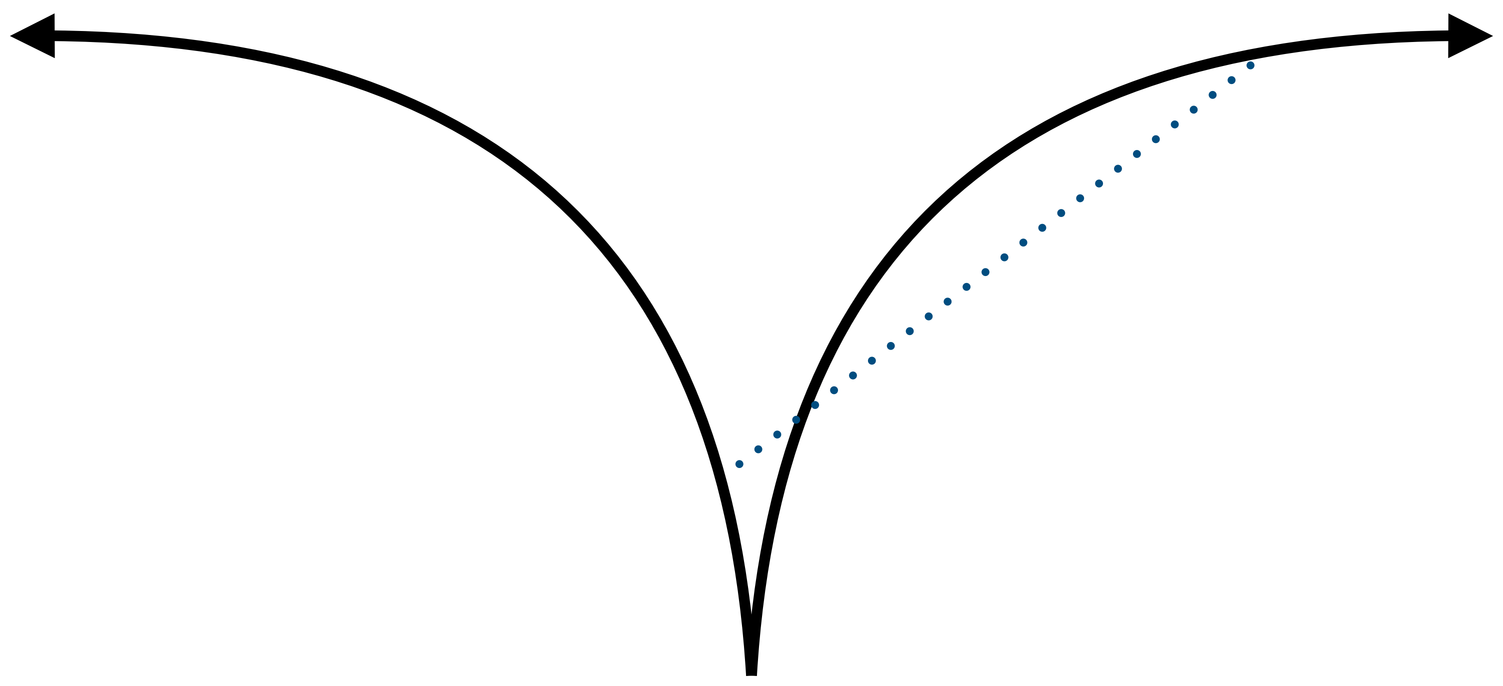
- Convexity does not mean the problem is *easy*
- Convex programs do not have to be smooth



Convexity

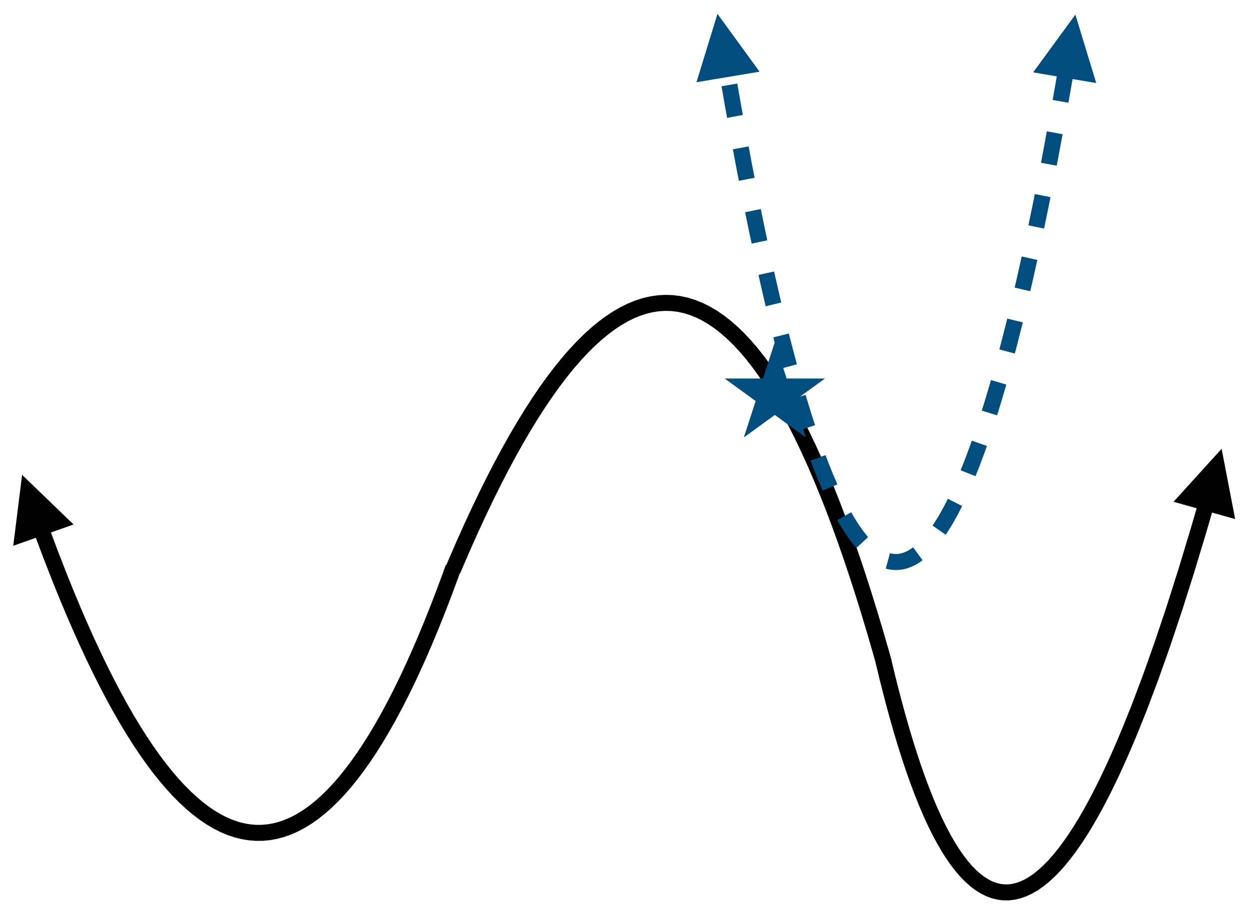
In summary

- Convexity does not mean the problem is *easy*
- Convex programs do not have to be smooth
- Convex programs are not the only ones with global optima



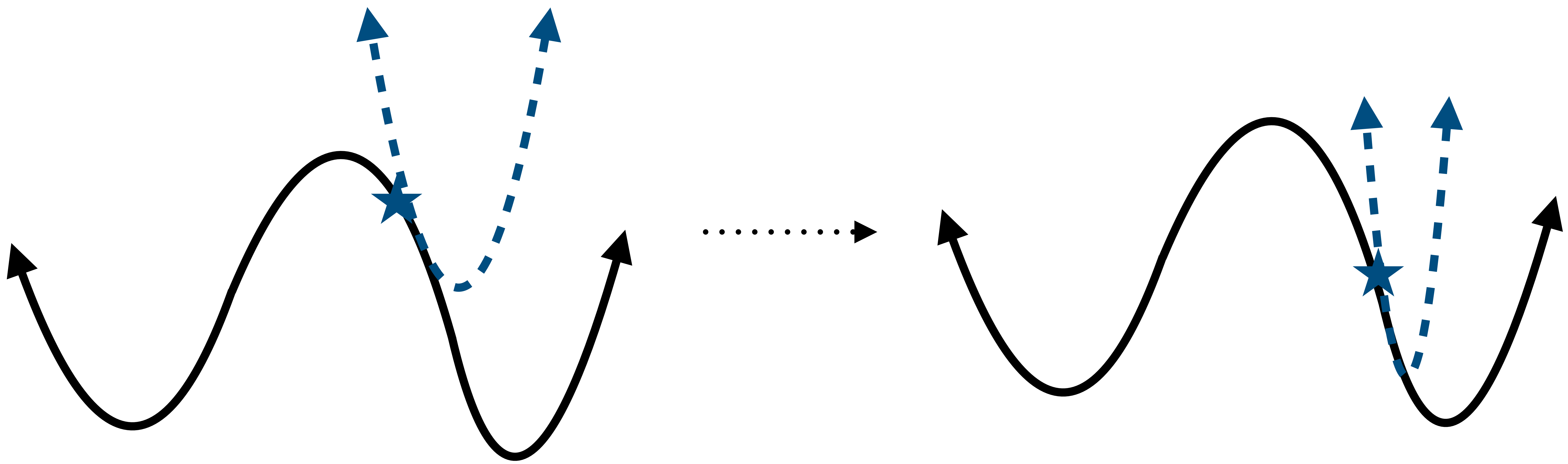
Relevance to trajectory design

- Most practical systems of interest are non-convex
- For *smooth* non-convex optimization problems, most approaches rely on solving a sequence of convex approximations to the problem



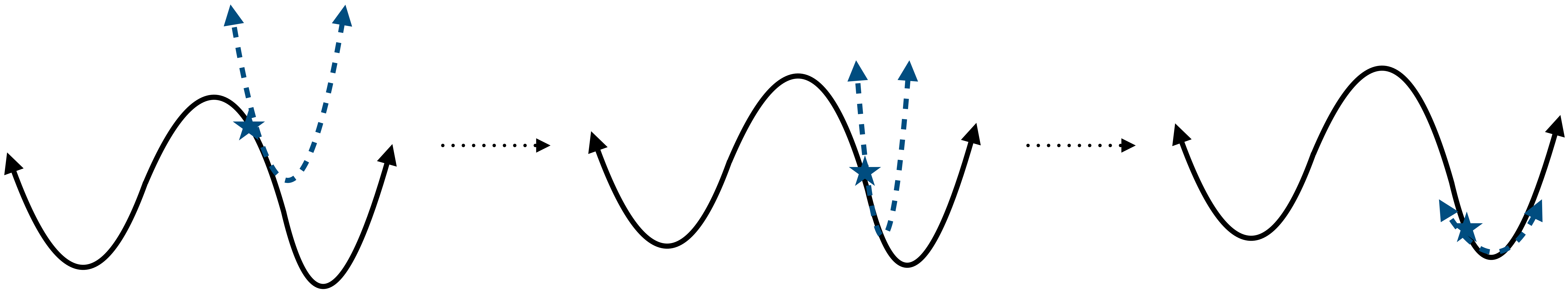
Relevance to trajectory design

- Most practical systems of interest are non-convex
- For *smooth* non-convex optimization problems, most approaches rely on solving a sequence of convex approximations to the problem



Relevance to trajectory design

- Most practical systems of interest are non-convex
- For *smooth* non-convex optimization problems, most approaches rely on solving a sequence of convex approximations to the problem



Relevance to trajectory design

Sequential quadratic programming

$$\min_x f(x)$$

subj. to:

$$h(x) = 0$$

$$g(x) \leq 0$$

$$f, h, g \in \mathscr{C}^2$$

Relevance to trajectory design

Sequential quadratic programming

$$\min_x f(x)$$

subj. to: $h(x) = 0$

$g(x) \leq 0$

$f, h, g \in \mathscr{C}^2$

$\longrightarrow$

1. Given $w^0 = (x^0, y^0, z^0)$

2. Construct Lagrangian using w

for $k \in [1, N_{\texttt{max}}]$

3. Solve local QP approximation

$$\min_{\Delta x} \frac{1}{2} \Delta x^T \nabla_x \mathscr{L}(\bar{x}) \Delta x + \nabla_x \mathscr{L}(\bar{x})$$

subj. to: $h(\bar{x}) + \nabla_x h(\bar{x}) \Delta x = 0$

$g(\bar{x}) + \nabla_x g(\bar{x}) \Delta x \leq 0$

4. Line search to find α

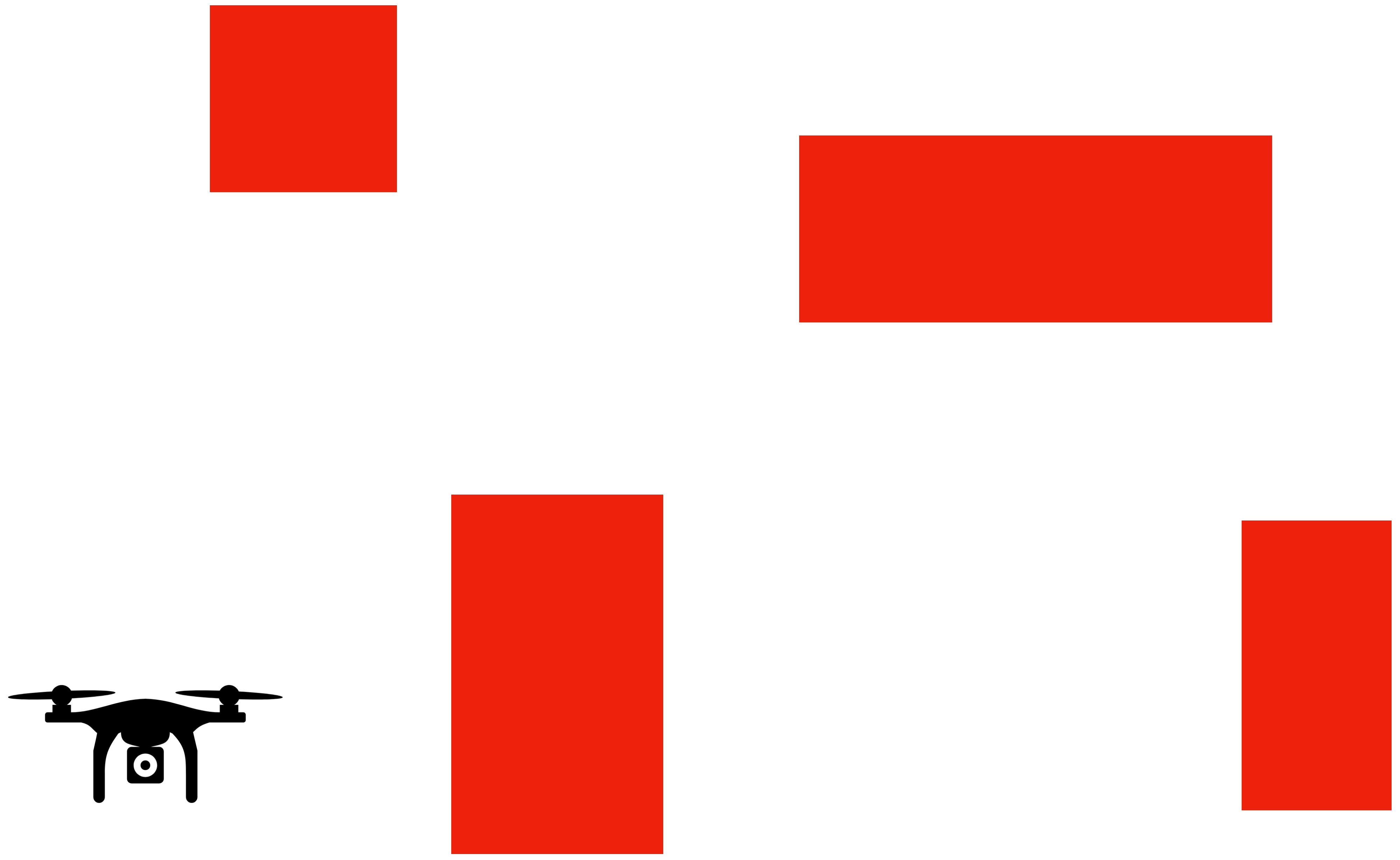
5. $w^k = w^{k-1} + \alpha \Delta w^k$

5. Terminate when necessary conditions satisfied

Convexity of practical constraints

Convexity of practical constraints

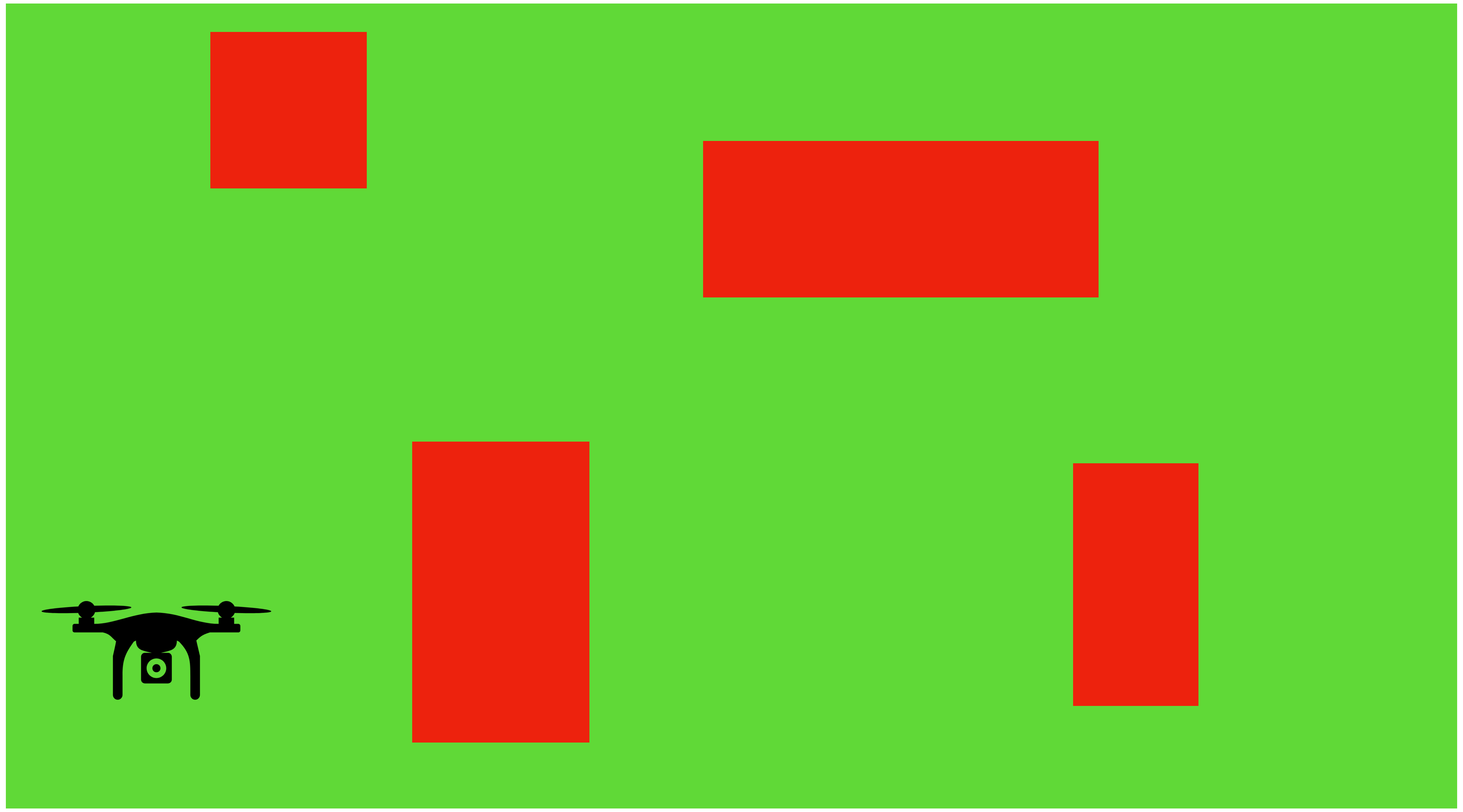
Collision avoidance



$$x_k \notin \mathcal{X}_{\text{obs}}$$

Convexity of practical constraints

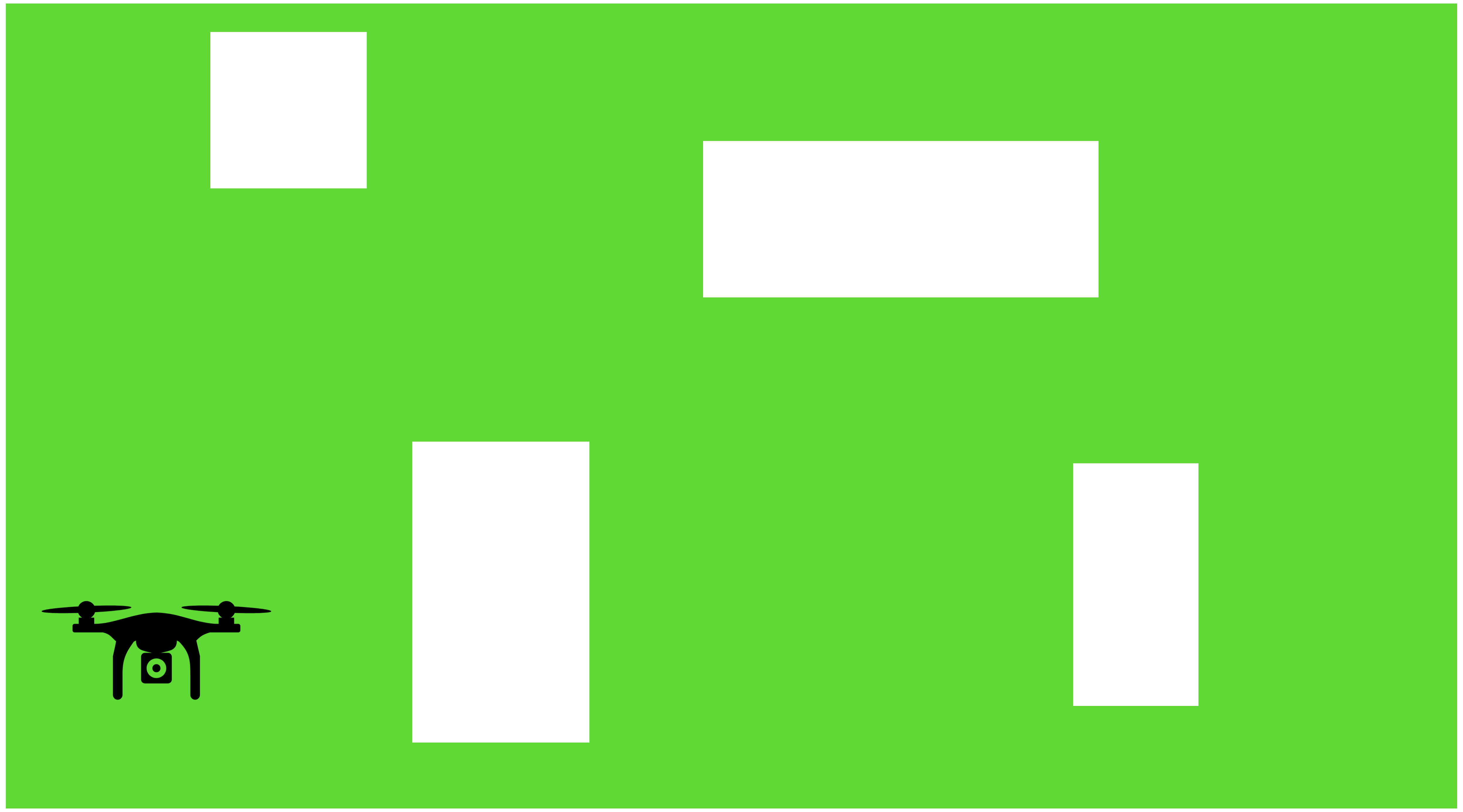
Collision avoidance



$$x_k \in \mathcal{X}_{\text{safe}}$$

Convexity of practical constraints

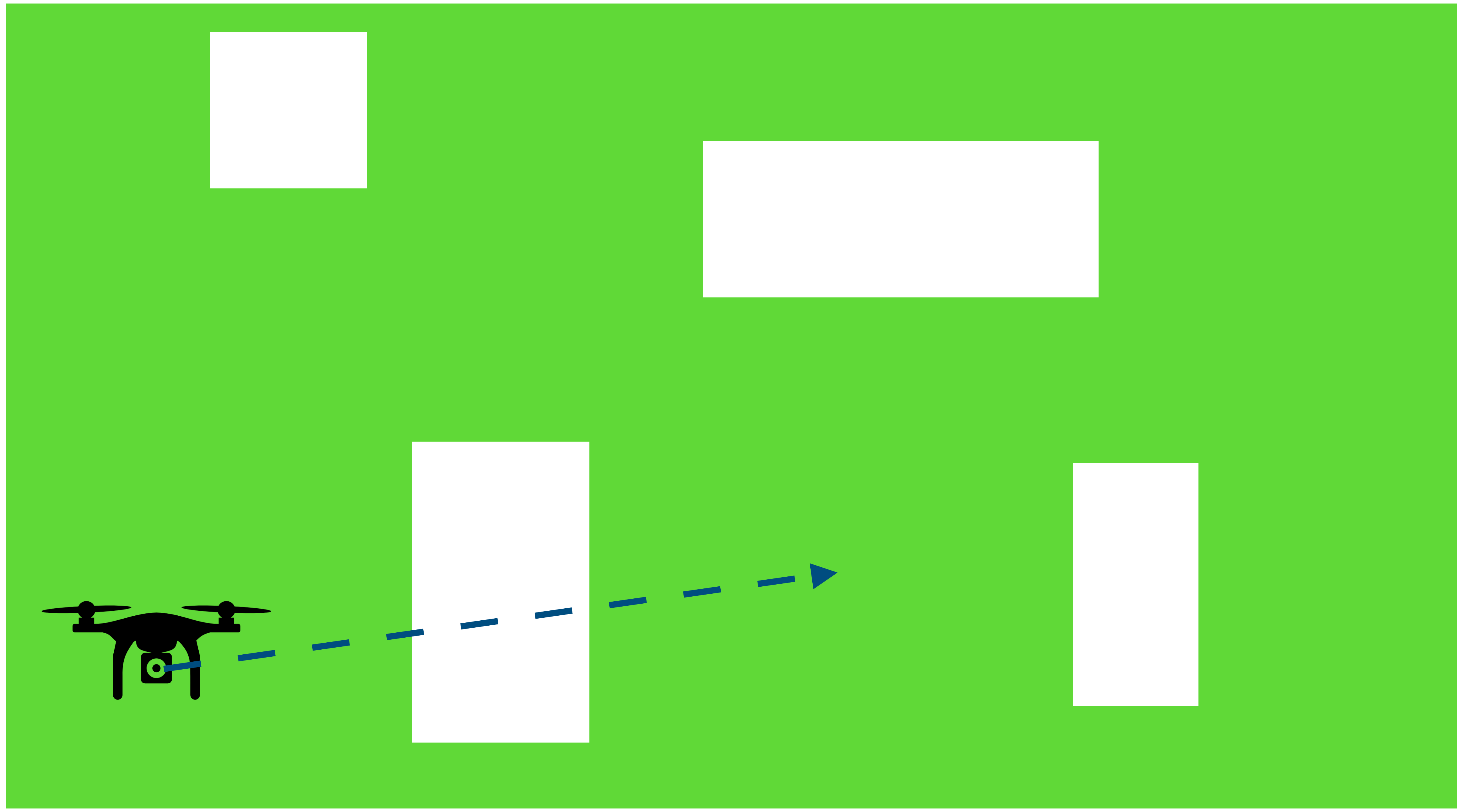
Collision avoidance



$$x_k \in \mathcal{X}_{\text{safe}}$$

Convexity of practical constraints

Collision avoidance



$x_k \in \mathcal{X}_{\text{safe}}$ Non-convex

Convexity of practical constraints

Dynamics constraints

- An equality constraint can be written as a two-sided inequality

$$x_{k+1} = f(x_k, u_k) \quad \longleftrightarrow \quad \begin{aligned} x_{k+1} &\leq f(x_k, u_k) \\ x_{k+1} &\geq f(x_k, u_k) \end{aligned}$$

Convexity of practical constraints

Dynamics constraints

- An equality constraint can be written as a two-sided inequality

$$x_{k+1} = f(x_k, u_k) \quad \longleftrightarrow \quad \begin{aligned} x_{k+1} &\leq f(x_k, u_k) \\ -x_{k+1} &\leq -f(x_k, u_k) \end{aligned}$$

Convexity of practical constraints

Dynamics constraints

- An equality constraint can be written as a two-sided inequality

$$x_{k+1} = f(x_k, u_k) \quad \longleftrightarrow \quad \begin{aligned} x_{k+1} &\leq f(x_k, u_k) \\ -x_{k+1} &\leq -f(x_k, u_k) \end{aligned}$$

- For both a function $f(x_k, u_k)$ and its negative $-f(x_k, u_k)$ to be convex with respect to x_k and u_k, $f(\cdot)$ must be an affine function

$$x_{k+1} = Ax_k + Bu_k + c$$

Convexity of practical constraints

Dynamics constraints

- An equality constraint can be written as a two-sided inequality

$$x_{k+1} = f(x_k, u_k) \quad \longleftrightarrow \quad \begin{aligned} x_{k+1} &\leq f(x_k, u_k) \\ -x_{k+1} &\leq -f(x_k, u_k) \end{aligned}$$

- For both a function $f(x_k, u_k)$ and its negative $-f(x_k, u_k)$ to be convex with respect to x_k and u_k, $f(\cdot)$ must be an affine function

$$x_{k+1} = Ax_k + Bu_k + c$$

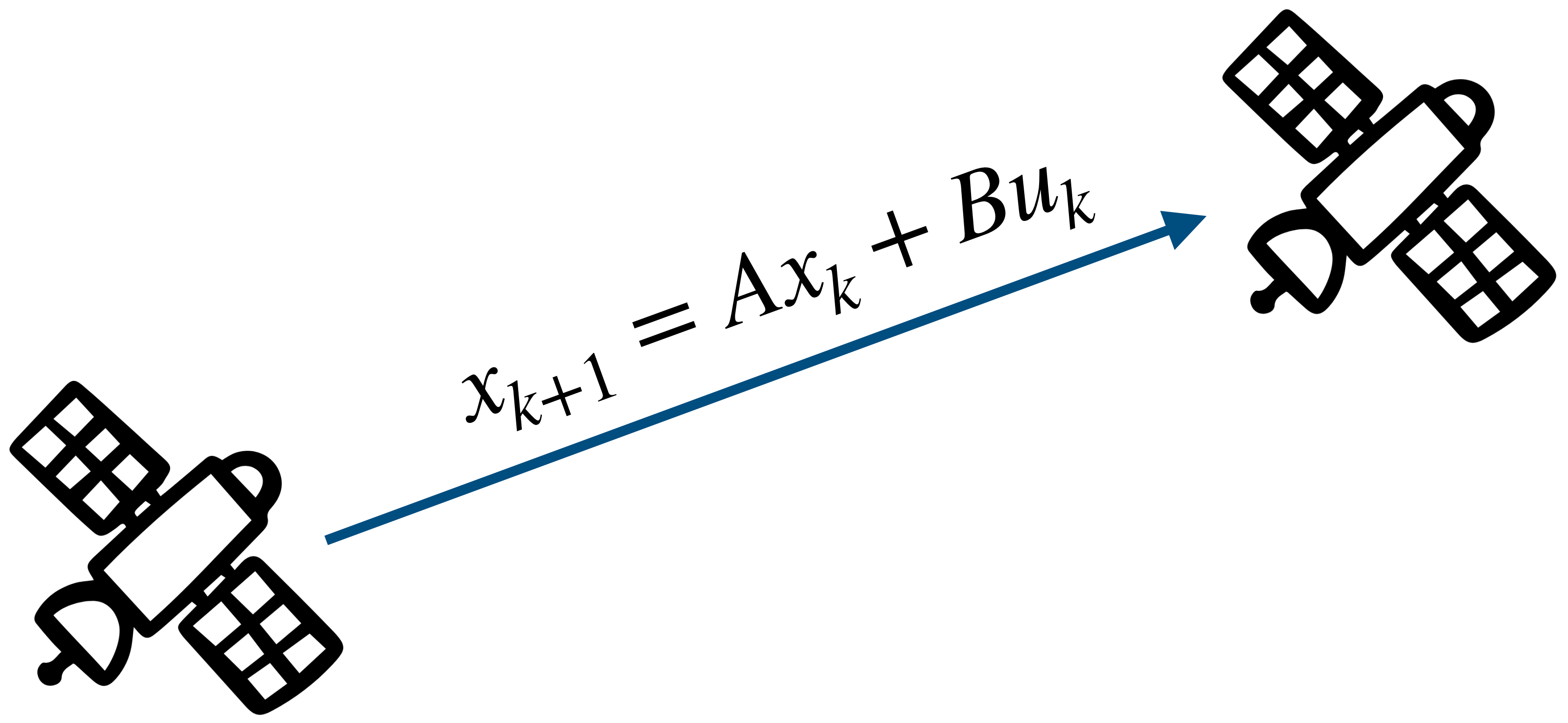
- Any system with nonlinear dynamics is non-convex

Convexity of practical constraints

Dynamics constraints

Spacecraft double integrator (convex)

$$x_{k+1} = \begin{pmatrix} I & \Delta t I \\ 0 & I \end{pmatrix} x_k + \begin{pmatrix} \Delta t^2 I \\ \Delta t I \end{pmatrix} u_k$$

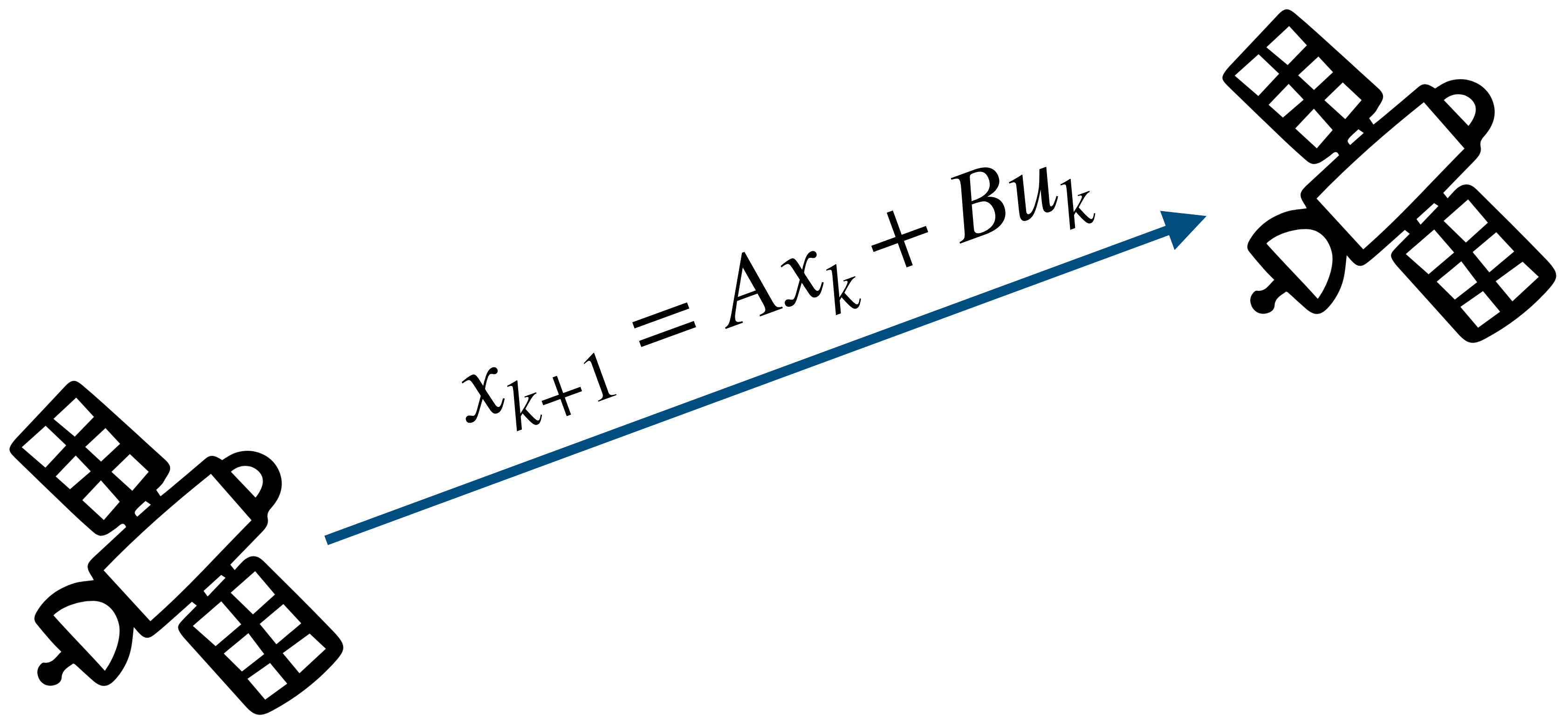


Convexity of practical constraints

Dynamics constraints

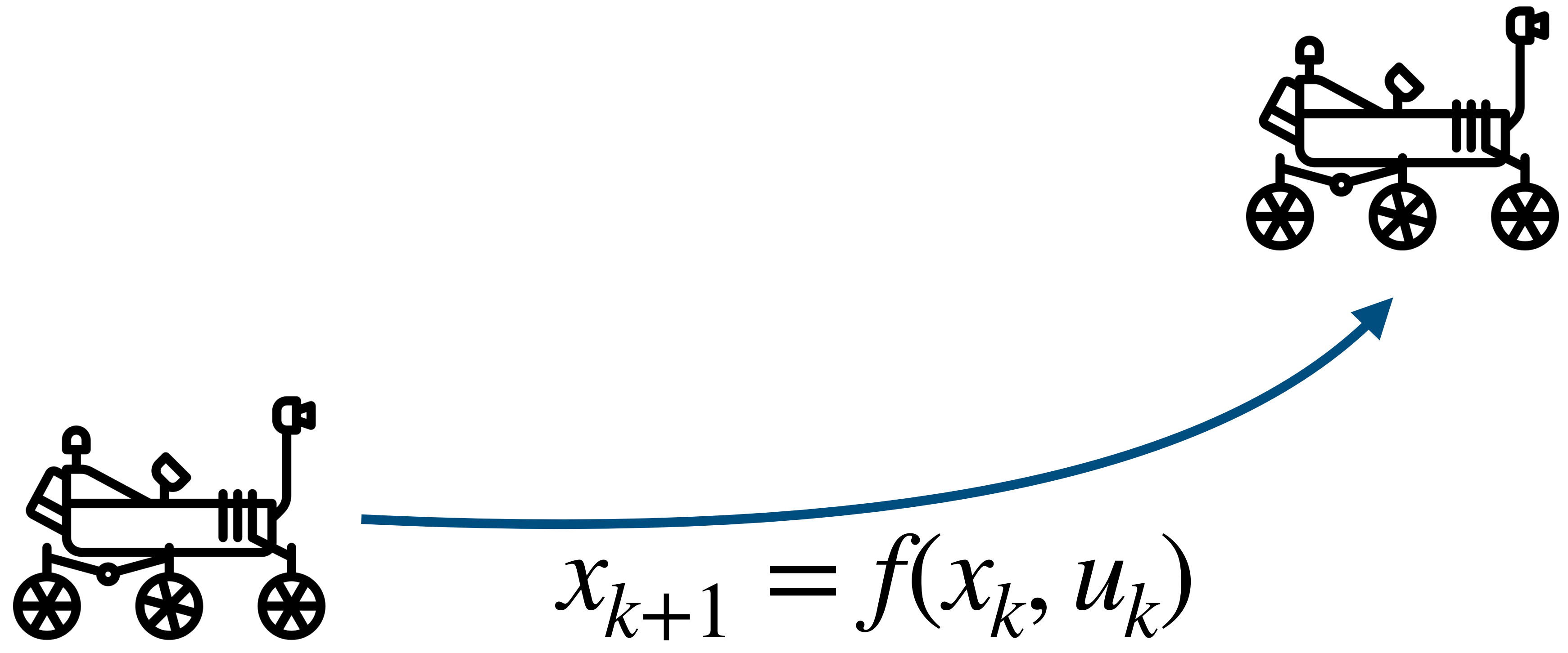
Spacecraft double integrator (convex)

$$x_{k+1} = \begin{pmatrix} I & \Delta t I \\ 0 & I \end{pmatrix} x_k + \begin{pmatrix} \Delta t^2 I \\ \Delta t I \end{pmatrix} u_k$$



Mars rover (non-convex)

$$x_{k+1} = x_k + \Delta t \begin{pmatrix} v_k \cos \theta_k \\ v_k \sin \theta_k \\ \omega_k \end{pmatrix}$$



Optimal control as nonlinear programs

How to convert a trajectory generation problem into standard optimization form?

Solve for trajectory $(x_{0:N}, u_{0:N})$

$$\min_{x_{0:N}, u_{0:N}} g_T(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

`subject to:`

$$x_0 = x_{\text{init}}$$

$$x_{k+1} = f(x_k, u_k)$$

$$x_k \in \mathcal{X}_{\text{safe}}$$

$$u_k \in \mathcal{U}$$

Optimal control as nonlinear programs

How to convert a trajectory generation problem into standard optimization form?

Solve for trajectory $(x_{0:N}, u_{0:N})$

$$\min_{x_{0:N}, u_{0:N}} \; g_T(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

subject to:

$$x_0 = x_{\text{init}}$$

$$x_{k+1} = f(x_k, u_k)$$

$$x_k \in \mathcal{X}_{\text{safe}}$$

$$u_k \in \mathcal{U}$$

1. Convert to standard form with inequality constraints written as $g_i(x_k, u_k) \leq 0$ and equality constraints as $h_i(x_{k+1}, x_k, u_k) = 0$

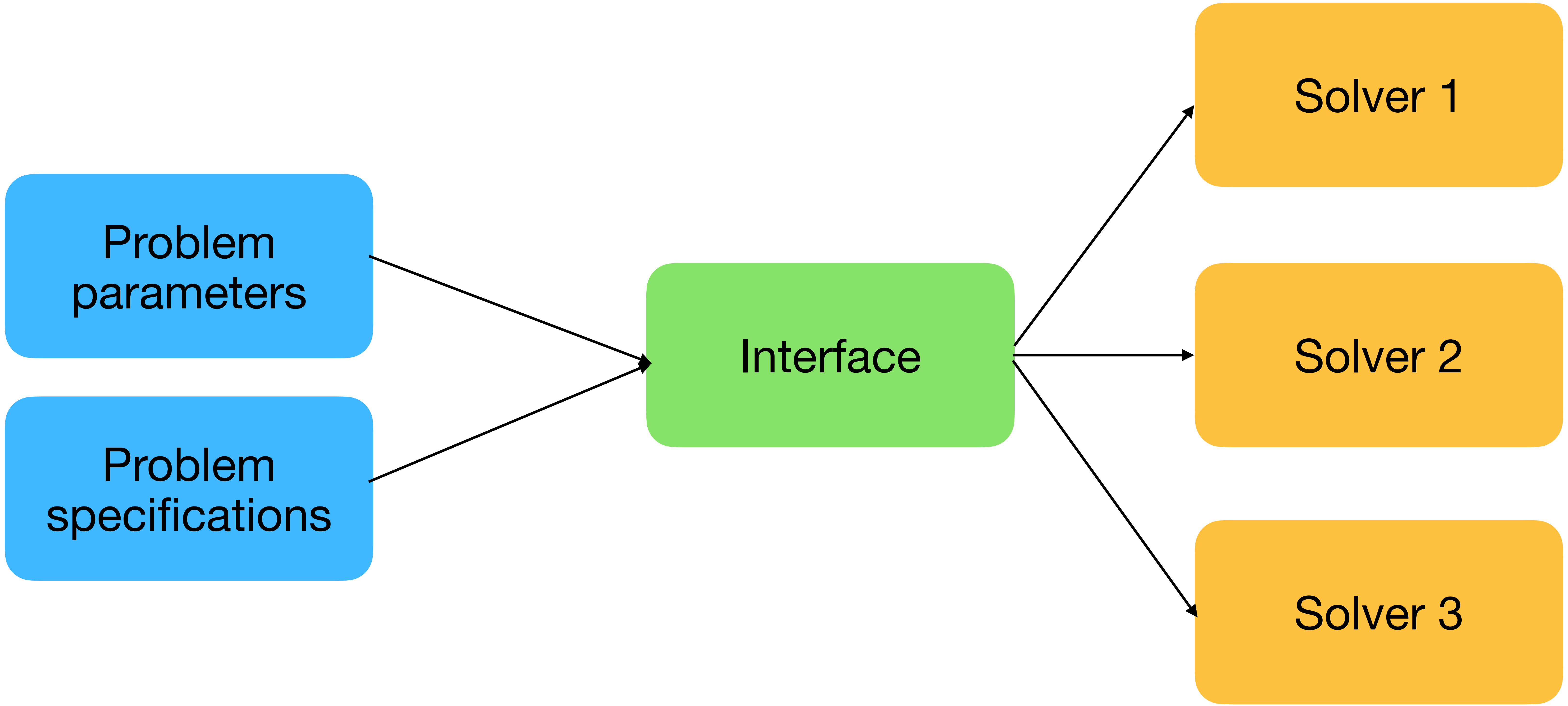
2. Provide gradient and Hessian for each constraint and cost

Converting an optimal control problem into standard form can be <u>cumbersome</u>!

Off-the-shelf trajectory optimization

Interface vs. solver

- An interface allows for solving the same optimization problem with different backend solvers
- Allows for rapid prototyping and testing



Off-the-shelf trajectory optimization

Interface vs. solver

```python
X = cp.Variable(shape=(nx, N+1))
U = cp.Variable(shape=(nu, N))

cost = 0.0
constraints = []

# Initial condition for system
constraints += [X[:,0] == x_init]

for cp_idx in range(N):
    # Add stage-wise cost
    cost += cp.quad_form(X[:, cp_idx+1] - x_goal, Q) + cp.quad_form(U[:, cp_idx], R)

    # Dynamics constraint
    constraints += [X[:, cp_idx+1] == Ak @ X[:,cp_idx] + Bk @ U[:,cp_idx]]

    # State upper and lower bounds
    constraints += [X[:, cp_idx+1] <= x_max]
    constraints += [X[:, cp_idx+1] >= x_min]

    # Control upper and lower bounds
    constraints += [U[:, cp_idx] <= u_max]
    constraints += [U[:, cp_idx] >= u_min]

prob = cp.Problem(cp.Minimize(cost), constraints)
prob.solve()
print(prob.value)
```

```
4677.378851800433
```

CVXPY

Solver interfaces

Python

acados PYOMO CVXPY CasADi

Julia

JUMP

jump-dev/**Convex.jl**

A Julia package for disciplined convex programming

C++

CasADi

acados

MATLAB

yalmip/**YALMIP**

MATLAB toolbox for optimization modeling

CVX

Convex solvers

Linear programs: GLPK, SCIP

Quadratic programs: OSQP, HPIPM, QPOASES

Second-order cone programs: SCS, ECOS, Clarabel

Semidefinite programs: SEDUMI, SDPT3

General nonlinear programs: IPOPT, SNOPT, KNITRO

Mixed integer linear programs: CPLEX, KNITRO, MOSEK, Gurobi

Mixed integer quadratic programs: MOSEK, Gurobi

Mixed integer nonlinear programming

NLLS solvers

Popular for SLAM

- Many problems in SLAM yield nonlinear least squares (NLLS) problems

$$\min_x \|f(x) - y\|_2^2$$

- Constraints are managed by “penalizing” violations
- Weaker guarantees and recent work in SLAM has sought to connect convex optimization with such problems
- Solvers: g2o, GTSAM, ceres, symforce, theseus

References

- D. Malyuta, Y. Yu, P. Elango, and B. Acikmese, “Advances in trajectory optimization for space vehicle control,” *Annual Reviews in Control*, vol. 52, pp. 282 — 315, 2021.
- M. Kelly, “An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation,” *SIAM Review*, vol. 59, no. 4, pp. 849 —904, 2017.
- D. Malta, T. P. Reynolds, M. Szmuk, T. Lew, R. Bonalli, M. Pavone, and B. Acikmese, “Convex Optimization for Trajectory Generation,” *IEEE Control Systems Magazine*, vol. 42, no. 5, pp. 40 — 113, 2022.