

**Lecture 22:**

# **Introduction to Optimization**

---

**Computer Graphics**  
**CMU 15-462/15-662**

# Last time: physically-based animation

- **Use dynamics to drive motion**
- **Complexity from simple models**
- **Technique: numerical integration**
  - **formulate equations of motion**
  - **take little steps forward in time**
  - **general, powerful tool**
- **Today: numerical optimization**
  - **another general, powerful tool**
  - **basic idea: “ski downhill” to get a better solution**
  - **used everywhere in graphics (not just animation)**
  - **image processing, geometry, rendering, ...**

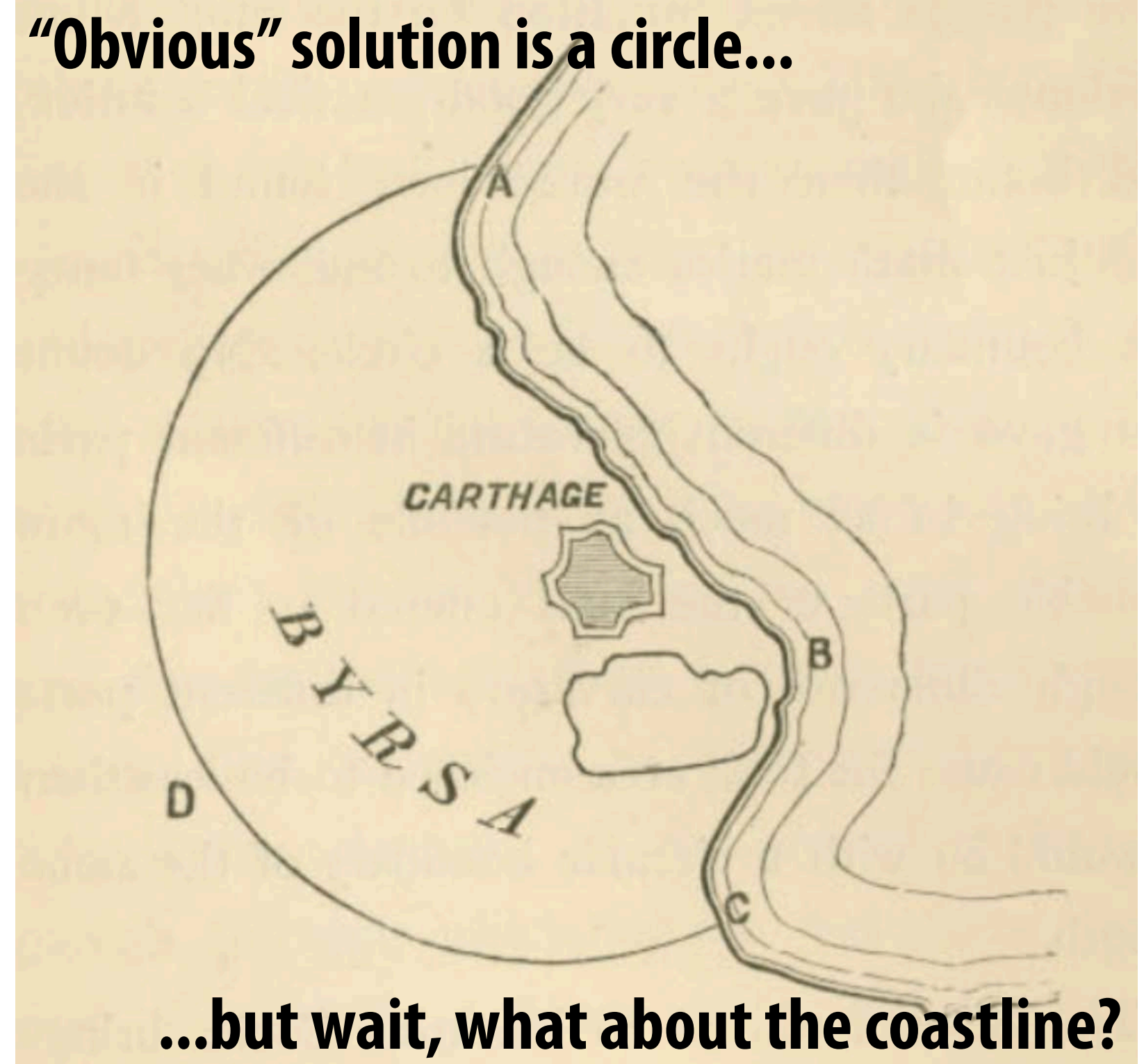


# What is an optimization problem?

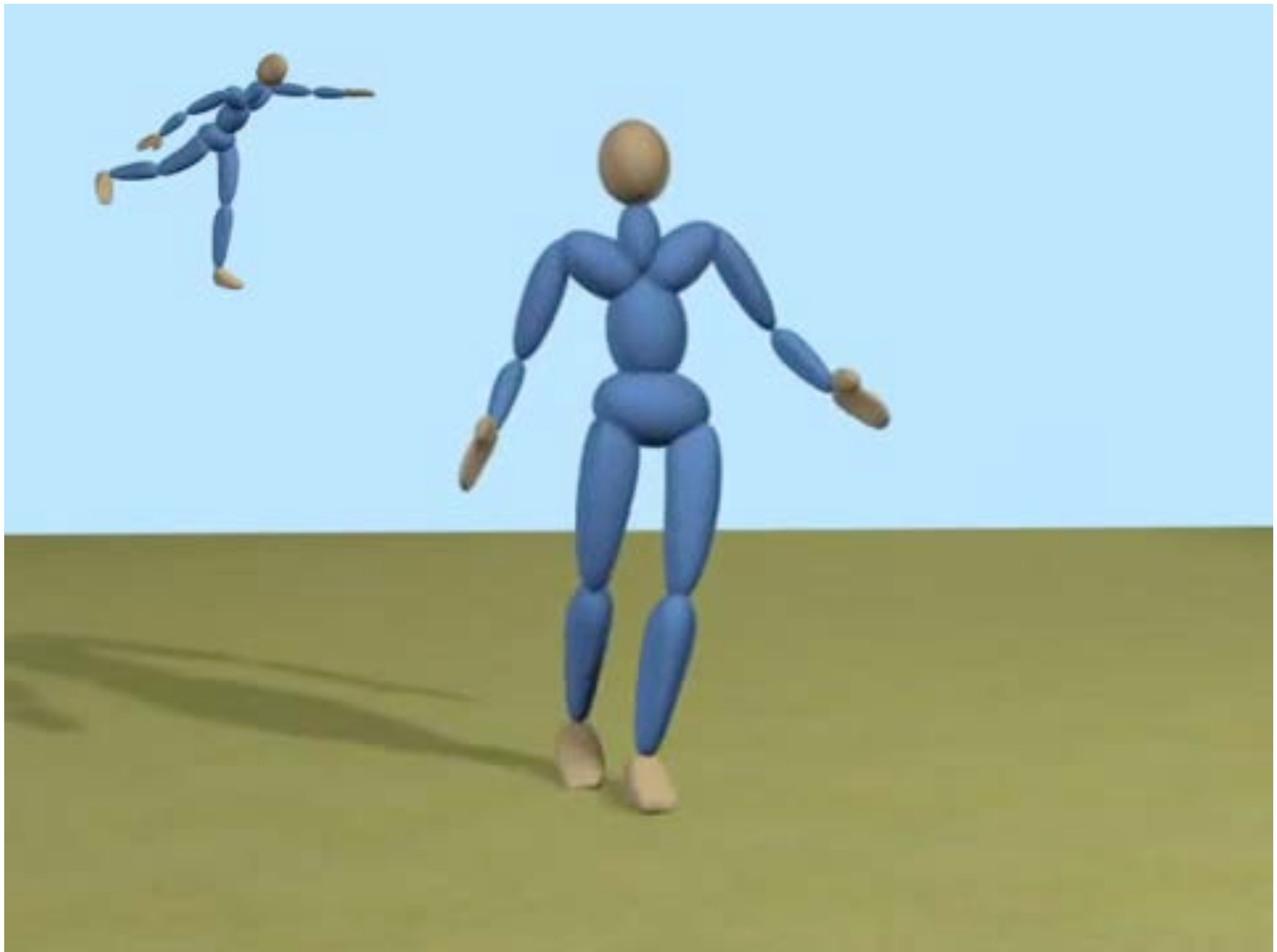
- **Natural human desire: find the best solution among all possibilities (subject to certain constraints)**
- **E.g., cheapest flight, shortest route, tastiest restaurant ...**
- **Has been studied since antiquity, e.g., isoperimetric problem:**

*“The first optimization problem known in history was practically solved by Dido, a clever Phoenician princess, who left her Tyrian home and emigrated to North Africa, with all her property and a large retinue, because her brother Pygmalion murdered her rich uncle and husband Acerbas, and plotted to defraud her of the money which he left. On landing in a bay about the middle of the north coast of Africa she obtained a grant from Hiarbas, the native chief of the district, of as much land as she could enclose with an ox-hide. She cut the ox-hide into an exceedingly long strip, and succeeded in enclosing between it and the sea a very valuable territory on which she build Carthage.”*

**—Lord Kelvin, 1893**



# Optimization in Graphics



Sumit Jain, Yuting Ye, and C. Karen Liu, *"Optimization-based Interactive Motion Synthesis"*

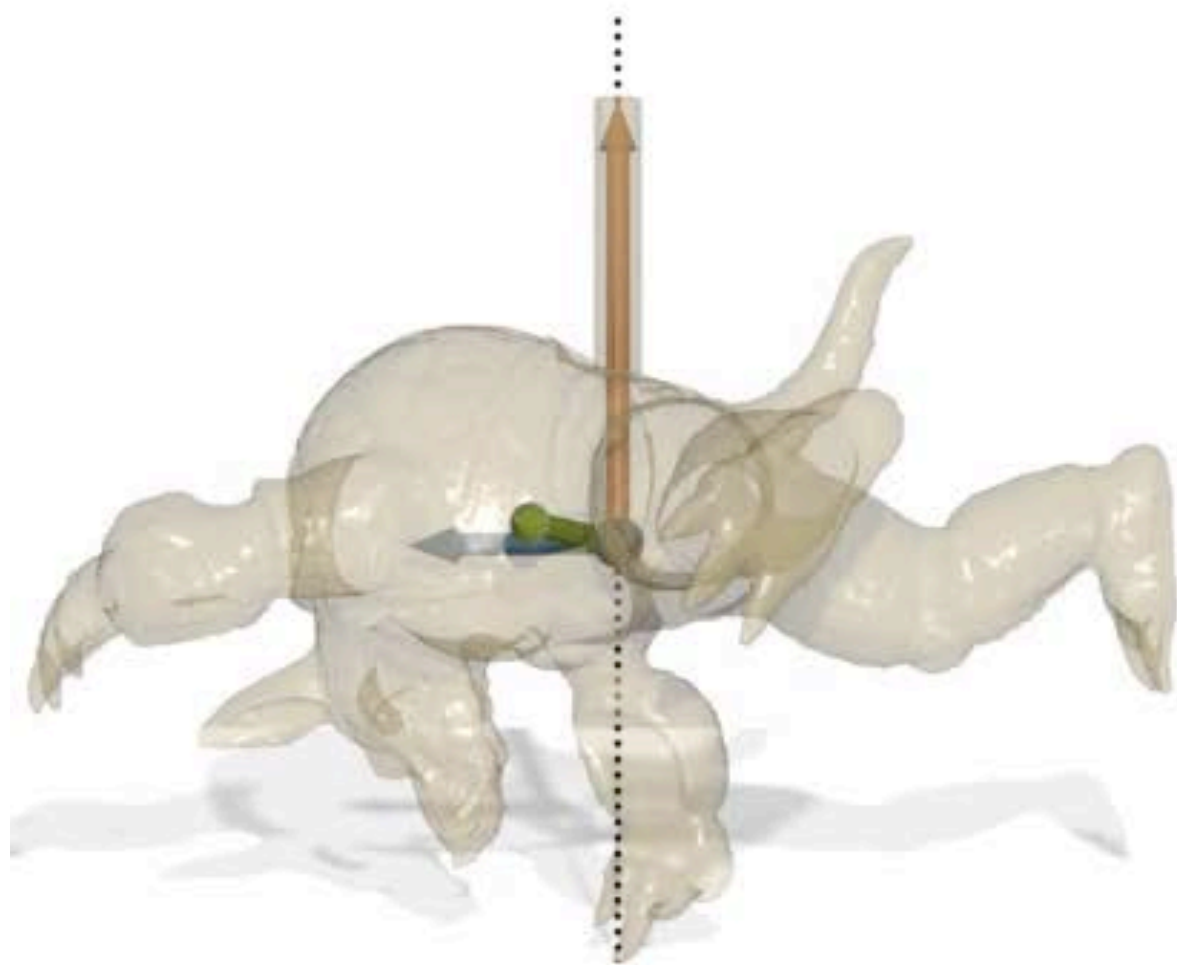
# Optimization in Graphics



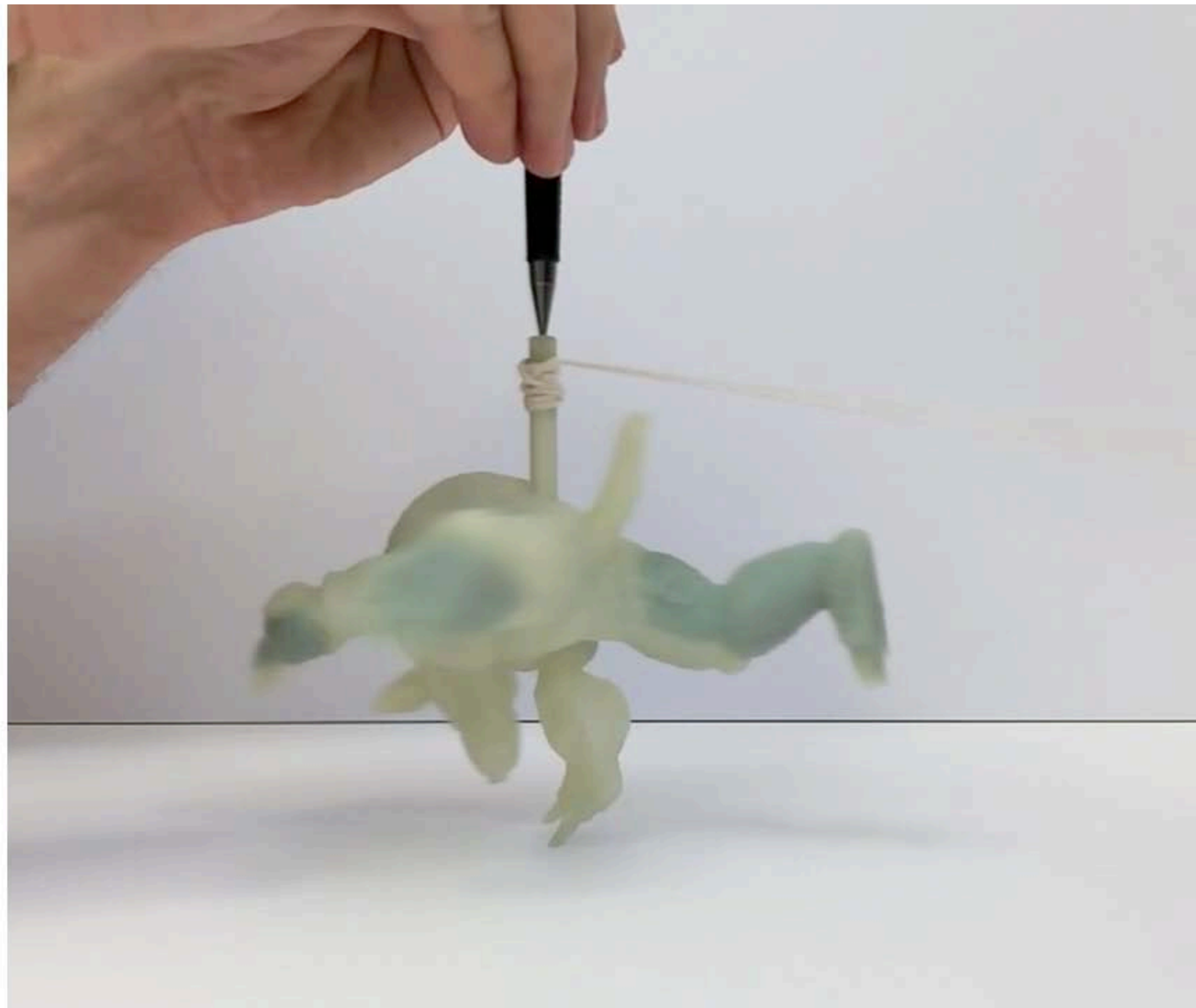
**Niloy J. Mitra, Leonidas Guibas, Mark Pauly, *"Symmetrization"***

# Optimization in Graphics

optimized result



© Disney, ETH zürich



**Moritz Bächer, Emily Whiting, Bernd Bickel, Olga Sorkine-Hornung,  
"Spin-It: Optimizing Moment of Inertia for Spinnable Objects"**

# Optimization in Graphics

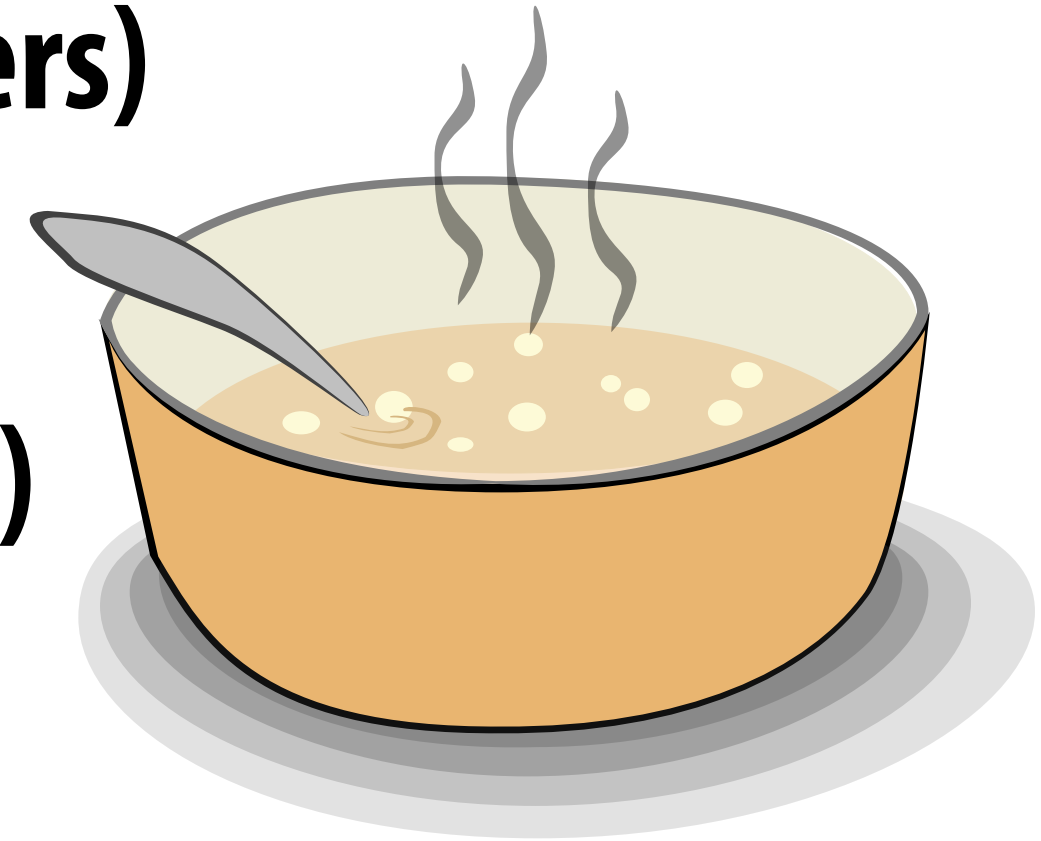


**Nobuyuki Umetani, Yuki Koyama, Ryan Schmidt & Takeo Igarashi,**  
*"Pteromys: Interactive Design and Optimization of Free-formed Free-flight Model Airplanes"*

# Continuous vs. Discrete Optimization

## ■ DISCRETE:

- domain is a discrete set (e.g., finite or integers)
- Example: best vegetable to put in a stew
  - Basic strategy? Try them all! (exponential)
  - sometimes clever strategy (e.g., MST)
  - more often, NP-hard (e.g., TSP)



## ■ CONTINUOUS:

- domain is not discrete (e.g., real numbers)
- Example: best temperature to cook an egg
- still many (NP-)hard problems, but also large classes of "easy" problems (e.g., convex)





# Optimization Problem in Standard Form

- Can formulate most continuous optimization problems this way:

“objective”: how much does solution  $x$  cost?

$$\begin{array}{ll} \min_{x \in \mathbb{R}^n} & f_0(x) \\ \text{subject to} & f_i(x) \leq b_i, \quad i = 1, \dots, m \end{array}$$

$$(f_i : \mathbb{R}^n \rightarrow \mathbb{R}, \quad i = 0, \dots, m)$$

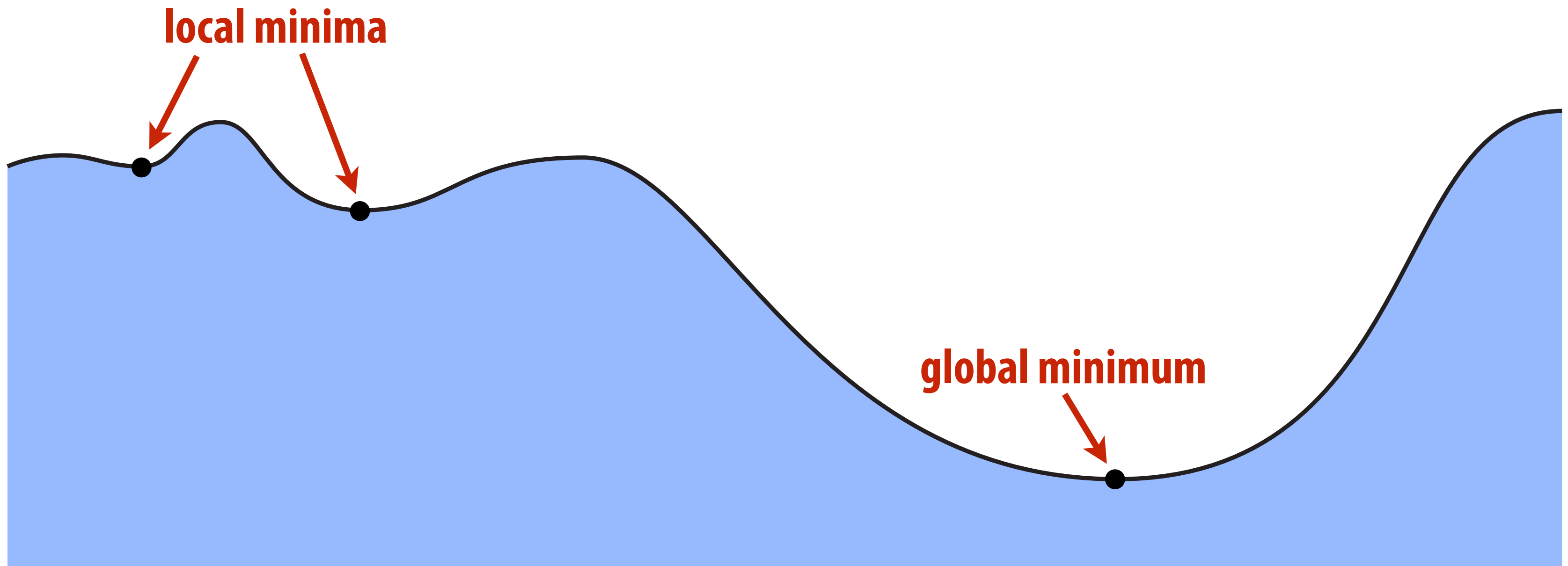
often (but not always) continuous, differentiable, ...

“constraints”: what must be true about  $x$ ? (“ $x$  is feasible”)

- **Optimal solution  $x^*$**  has smallest value of  $f_0$  among all feasible  $x$
- **Q:** What if we want to *maximize* something instead?
- **A:** Just flip the sign of the objective!
- **Q:** What if we want *equality* constraints, rather than inequalities?
- **A:** Include two constraints:  $g(x) \leq c$  and  $g(x) \leq -c$

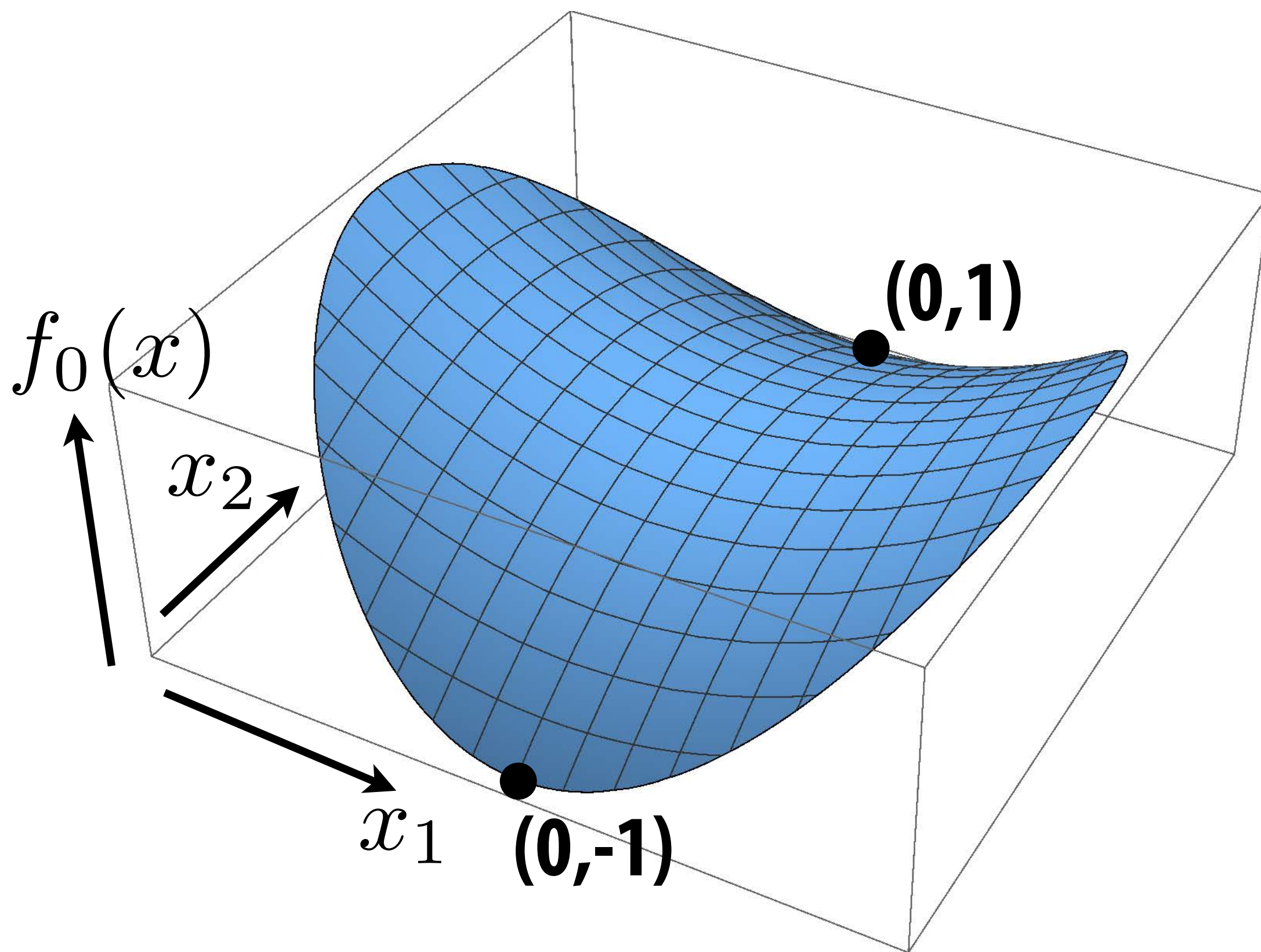
# Local vs. Global Minima

- ***Global* minimum is absolute best among all possibilities**
- ***Local* minimum is best “among immediate neighbors”**



**Philosophical question: does a local minimum “solve” the problem?  
Depends on the problem! (E.g., real protein folding is *local* minimum)  
Other times, local minima can be really bad (e.g., path planning)**

# Optimization Problem, Visualized



$$\begin{aligned} \min_{x \in \mathbb{R}^2} \quad & x_1^2 - x_2^2 \\ \text{s.t.} \quad & x_1^2 + x_2^2 - 1 \leq 0 \end{aligned}$$

**Q: Is this an optimization problem in standard form?**

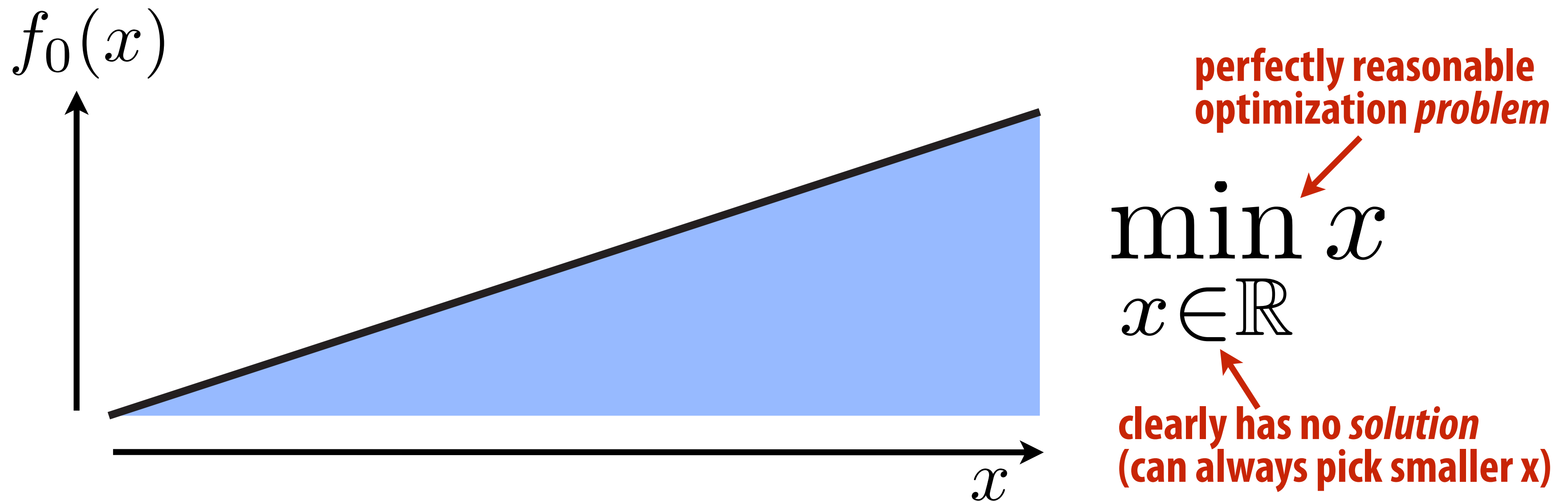
**A: Yes.**

**Q: Where is the optimal solution?**

**A: There are two,  $(0, 1)$ ,  $(0, -1)$ .**

# Existence & Uniqueness of Minimizers

- Already saw that (global) minimizer is not unique.
- Does it always exist? Why?
- Just consider all possibilities and take the smallest one, right?



- **WRONG!** Not all objectives are bounded from below.
- It's like that old adage: *"no matter how good you are, there will always be someone better than you."*

# Feasibility

- Ok, but suppose the objective is bounded from below.
- Then we can just take the best feasible solution, right?

value of objective doesn't depend on  $x$ ;  
all feasible solutions are equally good

$$\begin{array}{l} \min_{x \in \mathbb{R}^n} \quad 0 \\ \text{subject to} \quad f_i(x) \leq b_i, \quad i = 1, \dots, m \end{array}$$

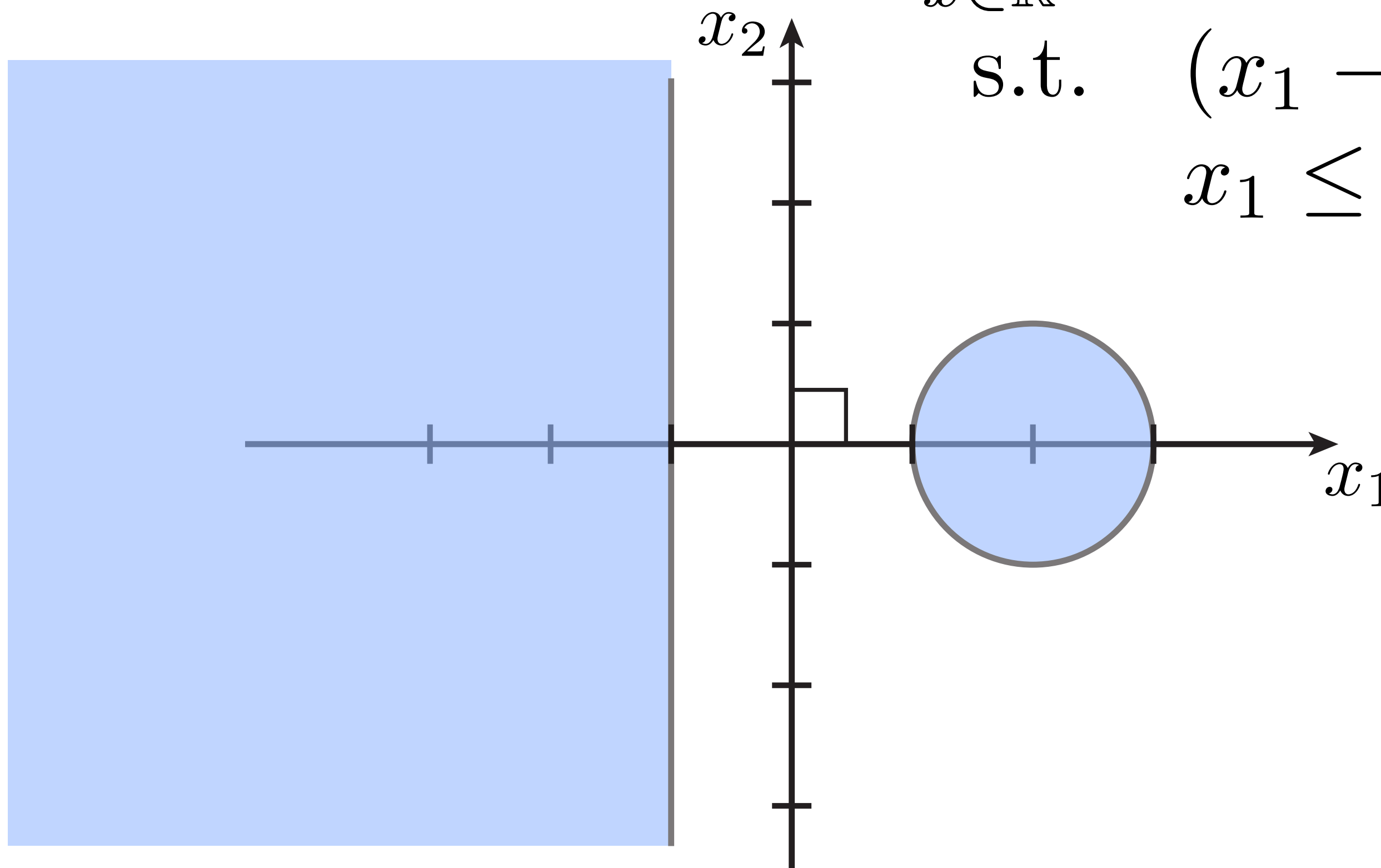
problem now is just finding a feasible solution—  
which can be really hard (or impossible!)

- Not if there aren't any!
- Every system of equations is an optimization problem.
- But not all problems have solutions!
- (You'll appreciate this more as you get older.)

# Feasibility - Example

**Q: Is this problem feasible?**

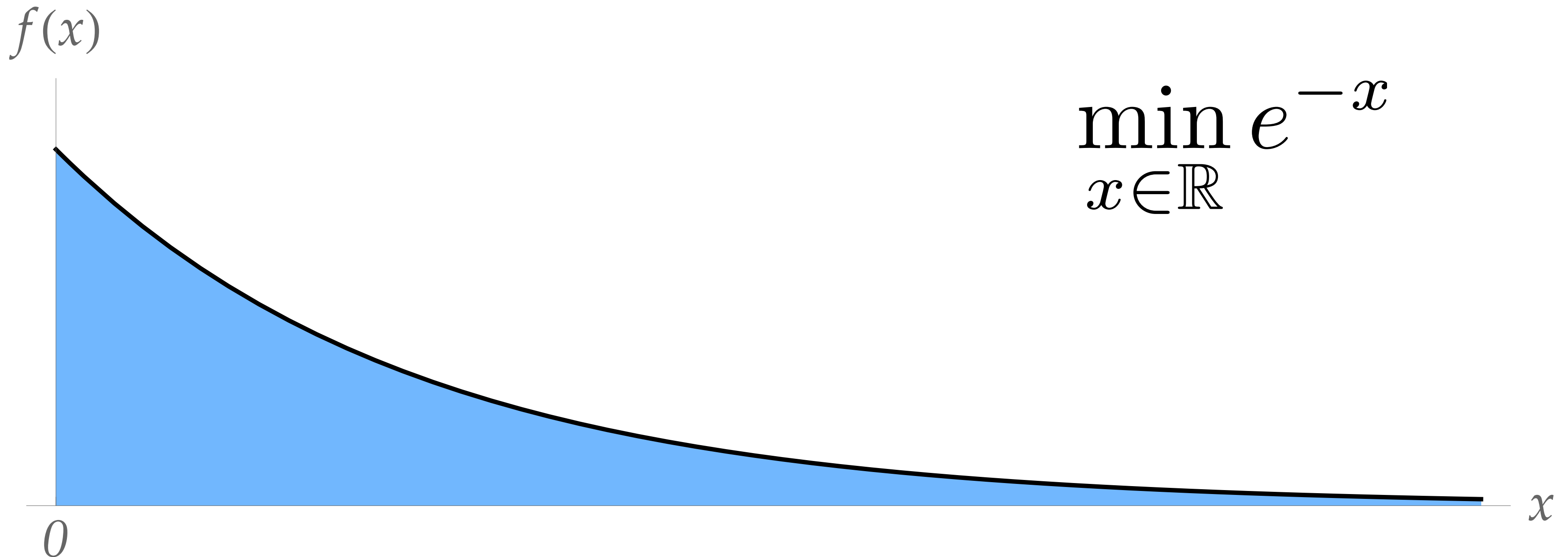
$$\begin{aligned} \min_{x \in \mathbb{R}^2} \quad & \sin(x_1) + x_2^2 \\ \text{s.t.} \quad & (x_1 - 2)^2 + x_2^2 \leq 1, \\ & x_1 \leq -1 \end{aligned}$$



**A: No**—the two sublevel sets (points where  $f_i(x) \leq 0$ ) have no common points, i.e., they do not overlap.

# Existence & Uniqueness of Minimizers, cont.

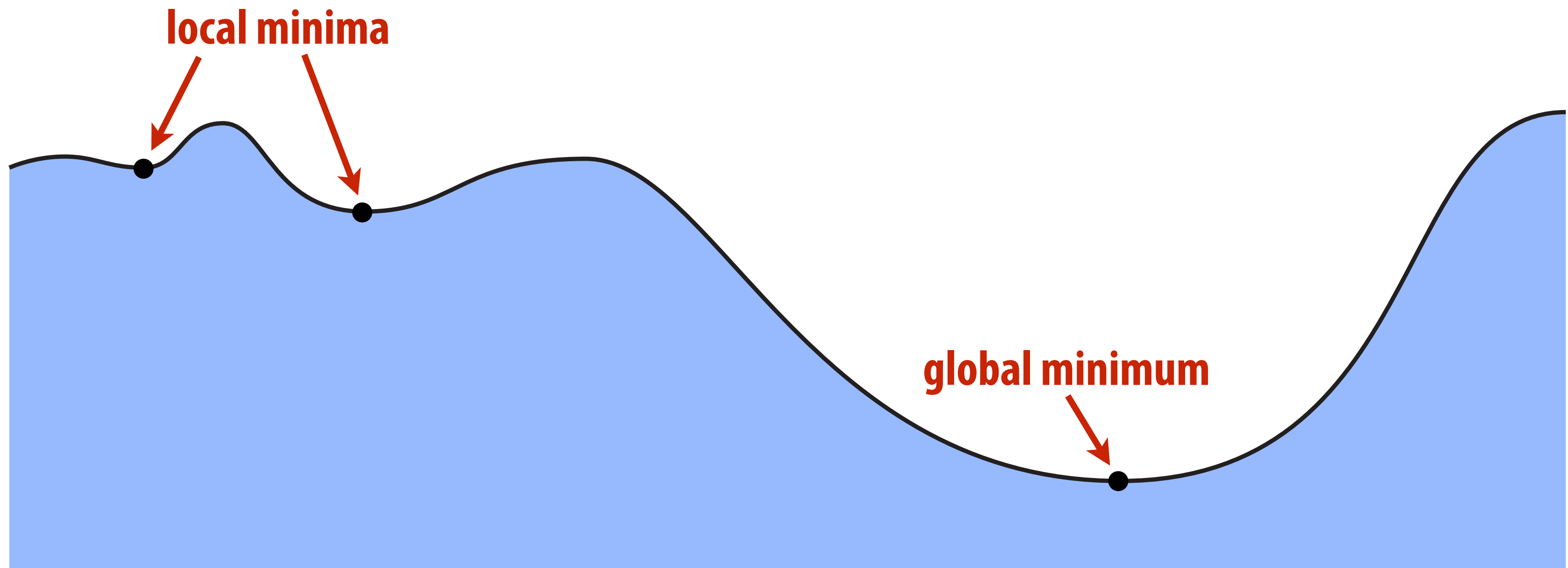
- Even being bounded from below is not enough:



- No matter how big  $x$  is, we never achieve the lower bound (0)
- So when does a solution exist? Two *sufficient* conditions:
- *Extreme value theorem*: continuous objective & compact domain
- *Coercivity*: objective goes to  $+\infty$  as we travel (far) in any direction

# Characterization of Minimizers

- Ok, so we have some sense of when a minimizer might *exist*
- But how do we know a given point  $x$  is a minimizer?

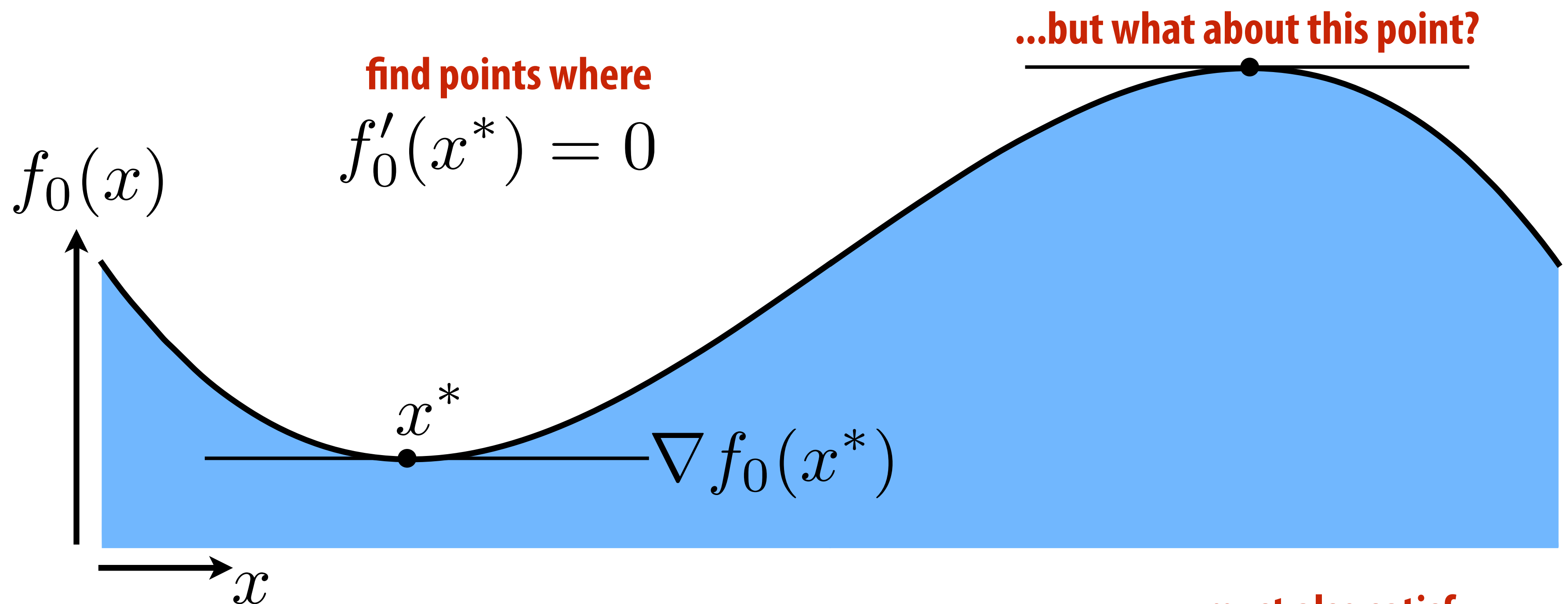


- Checking if a point is a global minimizer is (generally) hard
- But we can certainly test if a point is a local minimum (ideas?)
- (Note: a global minimum is also a local minimum!)



# Characterization of Local Minima

- Consider an objective  $f_0: \mathbb{R} \rightarrow \mathbb{R}$ . How do you find a minimum?
- (Hint: you may have memorized this formula in high school!)



- Also need to check *second* derivative (how?)  $f''_0(x^*) \geq 0$  must also satisfy
- Make sure it's *positive*
- Ok, but what does this all mean for more general functions  $f_0$ ?

# Optimality Conditions (Unconstrained)

- In general, our objective is  $f_0: \mathbb{R}^n \rightarrow \mathbb{R}$
- How do we test for a local minimum?
- 1st derivative becomes *gradient*; 2nd derivative becomes *Hessian*

$$\nabla f := \begin{bmatrix} \partial f / \partial x_1 \\ \vdots \\ \partial f / \partial x_n \end{bmatrix} \quad \nabla^2 f := \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial f}{\partial x_n^2} \end{bmatrix}$$

**GRADIENT**  
(measures "slope")

**HESSIAN**  
(measures "curvature")

- Optimality conditions?

$$\nabla f_0(x^*) = 0$$

1st order

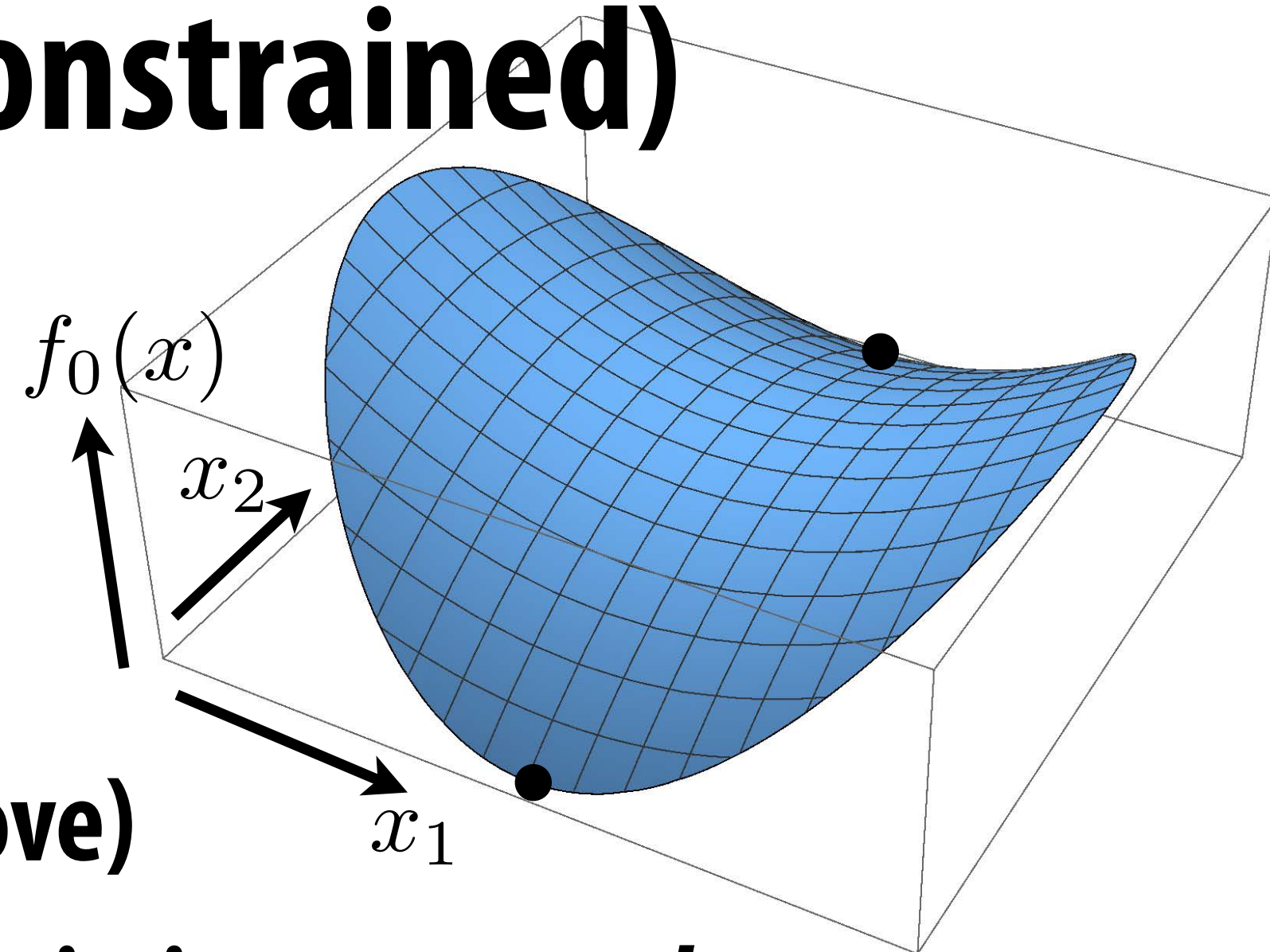
$$\nabla^2 f_0(x^*) \succeq 0$$

2nd order

*positive semidefinite (PSD)*  
( $u^T A u \geq 0$  for all  $u$ )

# Optimality Conditions (Constrained)

- What if we have constraints?
- Is gradient at minimizer still zero?
- Is Hessian at minimizer still PSD?
- Not necessarily! (See example above)
- In general, any (local or global) minimizer must *at least* satisfy the *Karush–Kuhn–Tucker (KKT)* conditions:

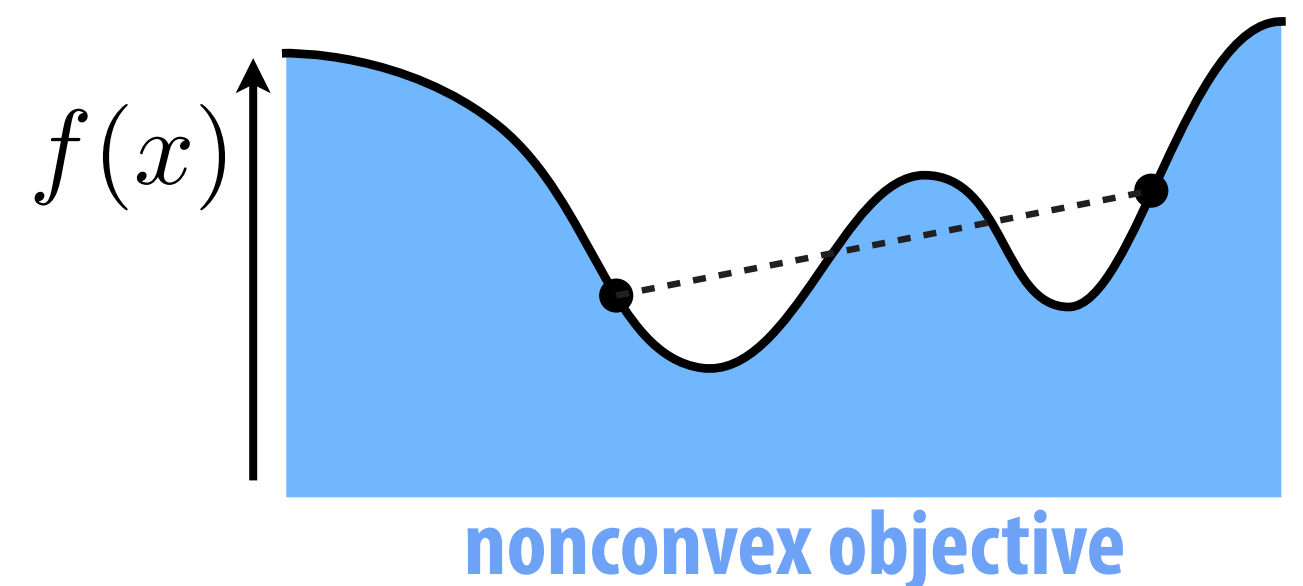
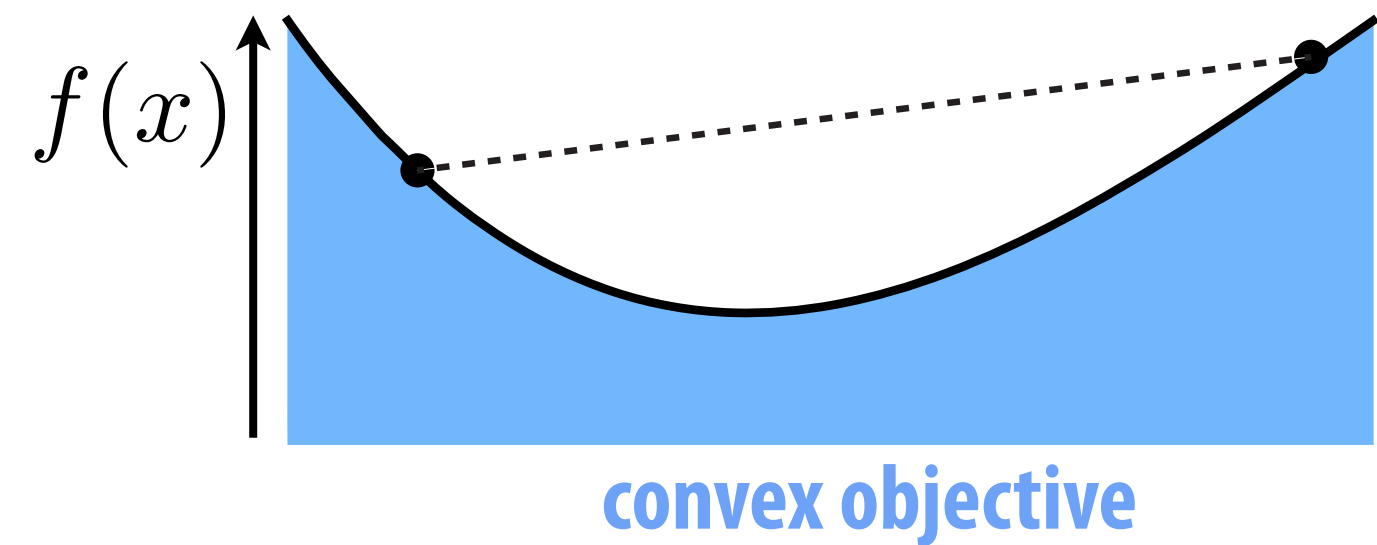
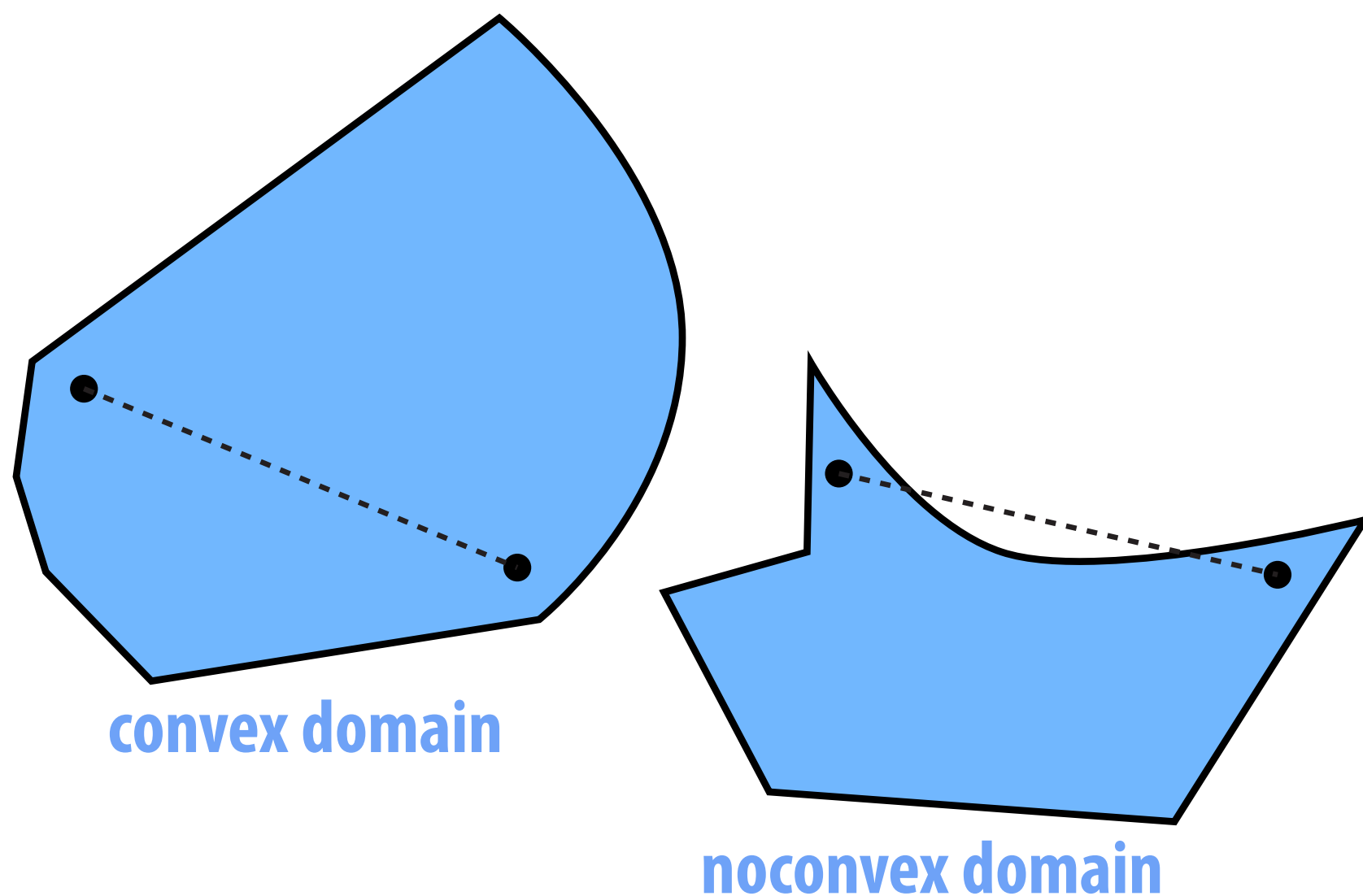


$$\begin{aligned} \exists \lambda_i \text{ s.t. } \quad \nabla f_0(x^*) &= - \sum_{i=1}^n \lambda_i \nabla f_i(x^*) && \text{stationarity} \\ f_i(x^*) &\leq 0, \quad i = 1, \dots, n && \text{primal feasibility} \\ \lambda_i &\geq 0, \quad i = 1, \dots, n && \text{dual feasibility} \\ \lambda_i f_i(x^*) &= 0, \quad i = 1, \dots, n && \text{complementary slackness} \end{aligned}$$

- ...we won't work with these in this class!  
(But good to know where to look.)

# Convex Optimization

- Special class of problems that are almost always “easy” to solve (polynomial-time!)
- Problem *convex* if it has a convex domain *and* convex objective



- Why care about convex problems in graphics?
  - can make guarantees about solution (always the best)
  - doesn't depend on initialization (strong convexity)
  - often quite efficient, but not always

# Convex Quadratic Objectives & Linear Systems

- Very important example: convex *quadratic* objective
- Already saw this with, e.g., quadric error simplification
- Valuable “*variational*” way of looking at many common equations
- Can be expressed via positive-semidefinite (PSD) matrix:

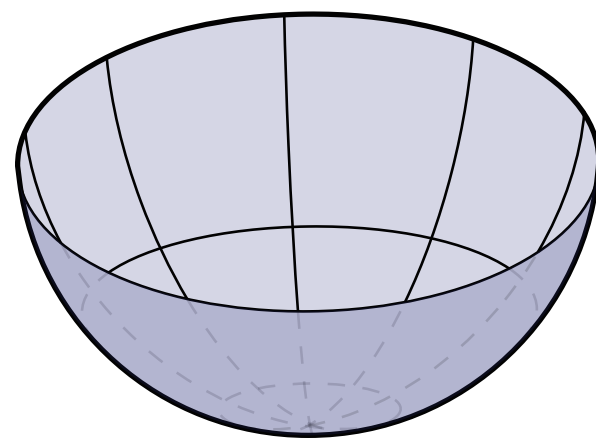
$$f_0(x) = \frac{1}{2}x^T A x - x^T b, \quad A \succeq 0$$

just solve a linear system!

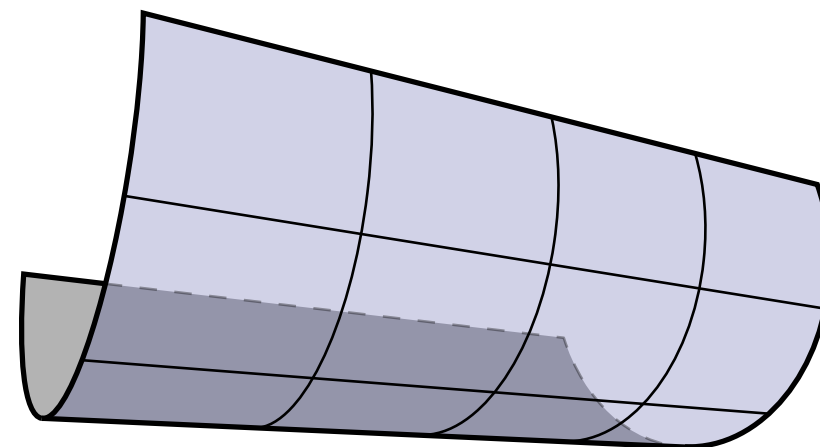
- Q: 1st-order optimality condition?  $Ax = b$

satisfied by definition

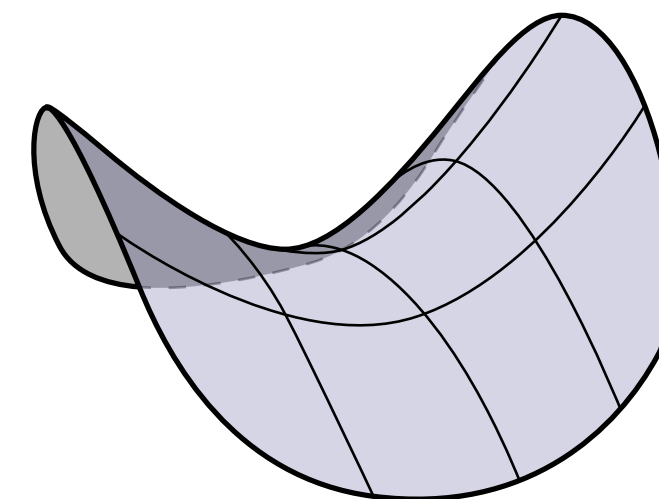
- Q: 2nd-order optimality condition?  $A \succeq 0$



positive definite



positive semidefinite



indefinite

**Sadly, life is not usually that easy.  
How do we solve optimization  
problems in general?**

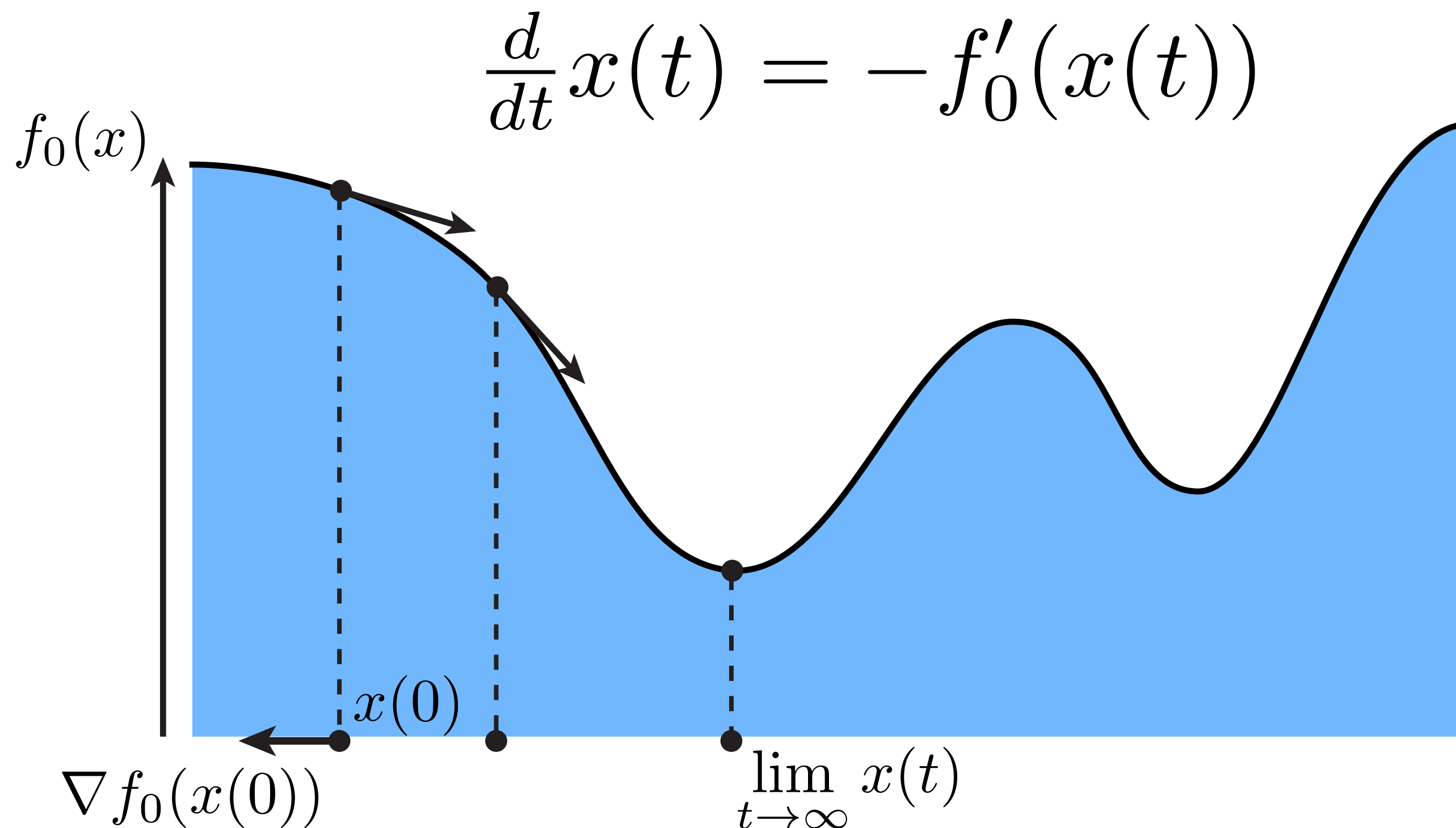
# Descent Methods

An idea as old as the hills:



# Gradient Descent (1D)

- Basic idea: follow the gradient “downhill” until it’s zero
- (Zero gradient was our 1st-order optimality condition)



- Do we always end up at a (global) minimum?
- How do we compute gradient descent in practice?



# Gradient Descent Algorithm (1D)

- Did you notice that gradient descent equation is an ODE?

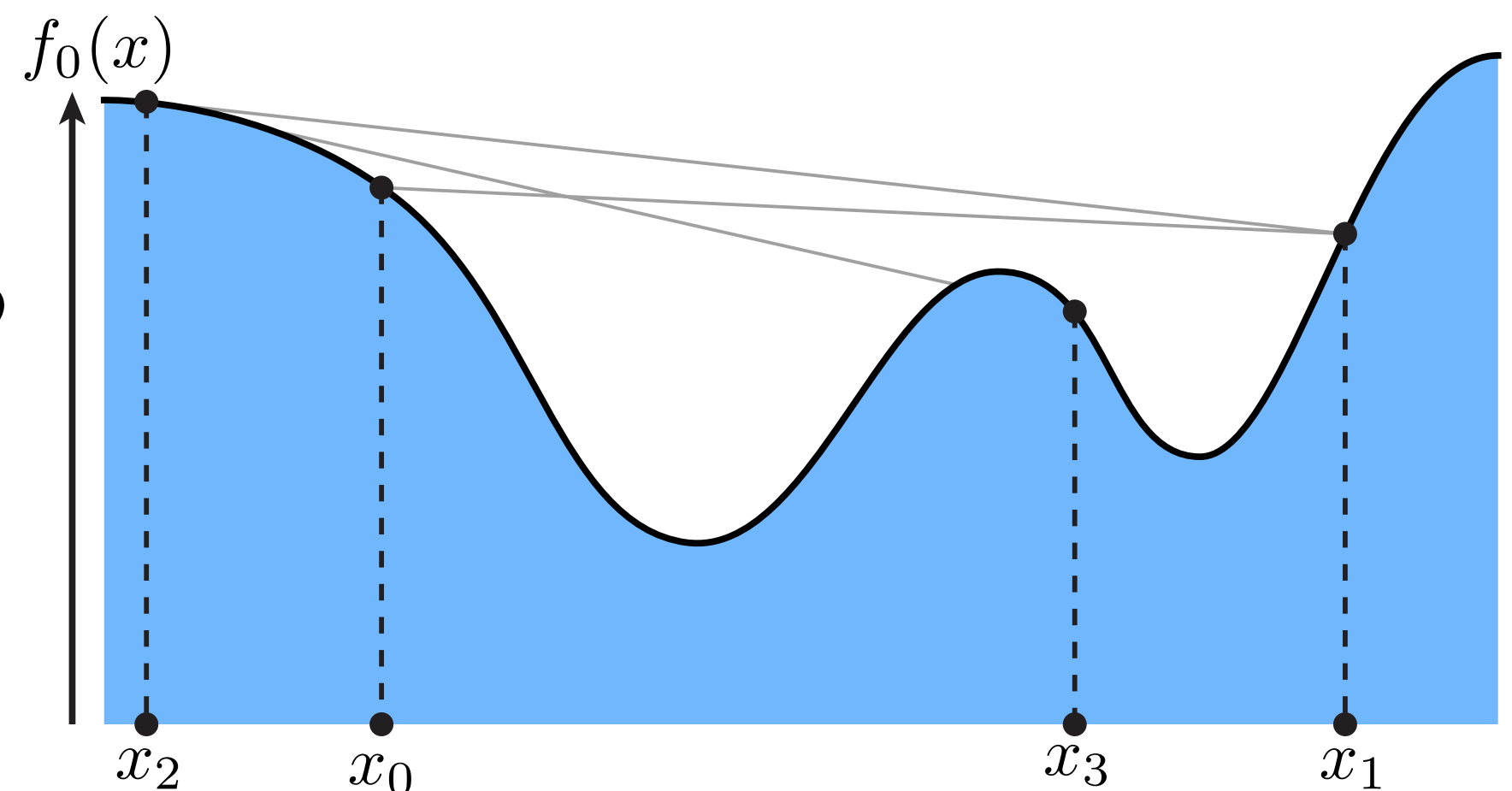
- Q: How do we solve it numerically?  $\frac{d}{dt}x(t) = -f'_0(x(t))$

- One way: forward Euler:

$$x_{k+1} = x_k + \tau f'_0(x_k)$$

- Q: How do we pick the time step?

- If we're not careful, we'll go zipping all over the place; won't make any progress.



- Basic idea: use “*step control*” to determine step size based on value of objective & derivatives.

- A careful strategy (e.g., Armijo-Wolfe) can guarantee convergence at least to a *local* minimum.

- For now we will do something simpler: make  $\tau$  *really small!*

# Gradient Descent Algorithm (nD)

- Q: How do we write gradient descent equation in general?

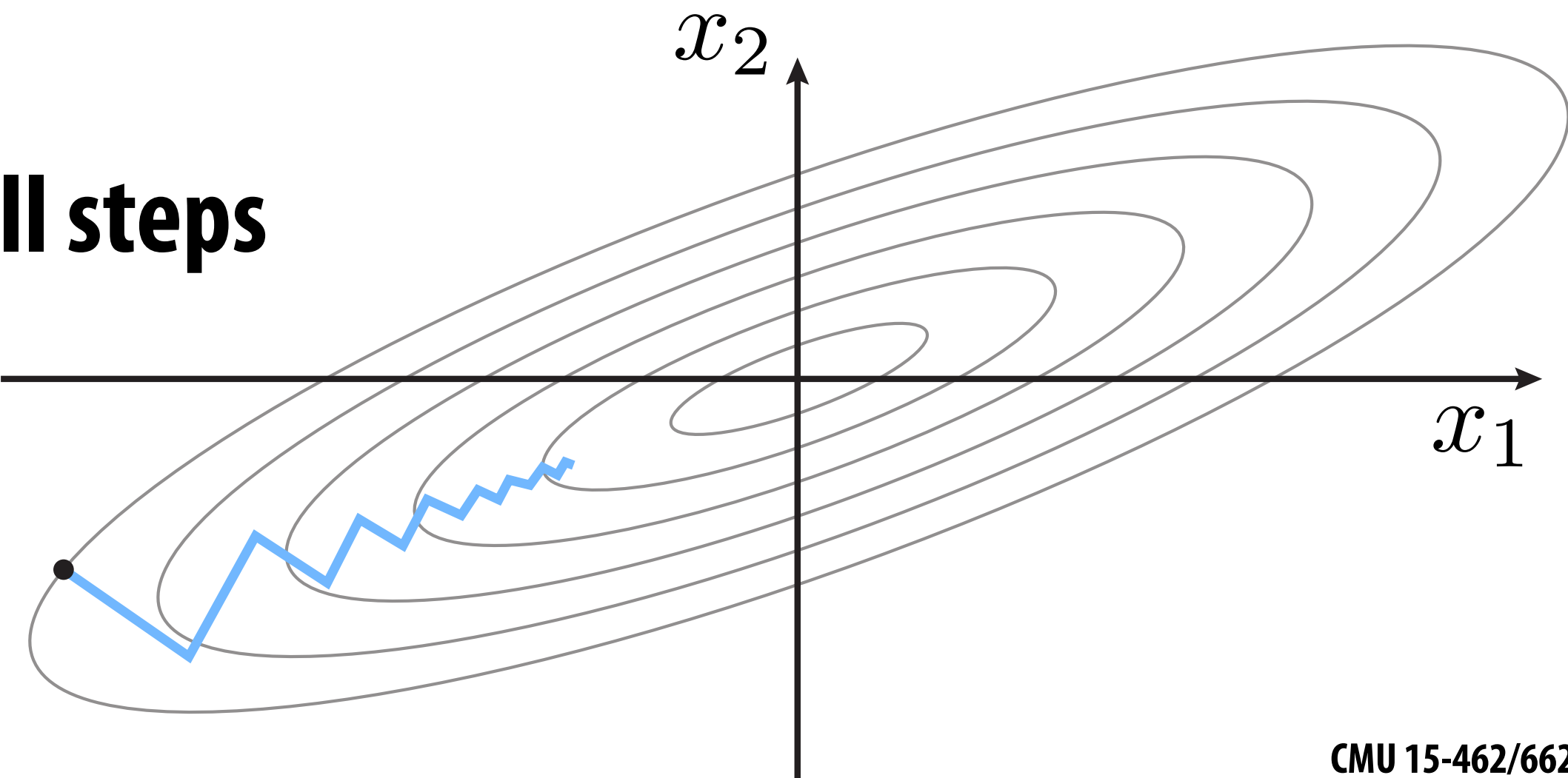
$$\frac{d}{dt}x(t) = -\nabla f_0(x(t))$$

- Q: What's the corresponding discrete update?

$$x_{k+1} = x_k - \tau \nabla f_0(x_k)$$

- Basic challenge in nD:

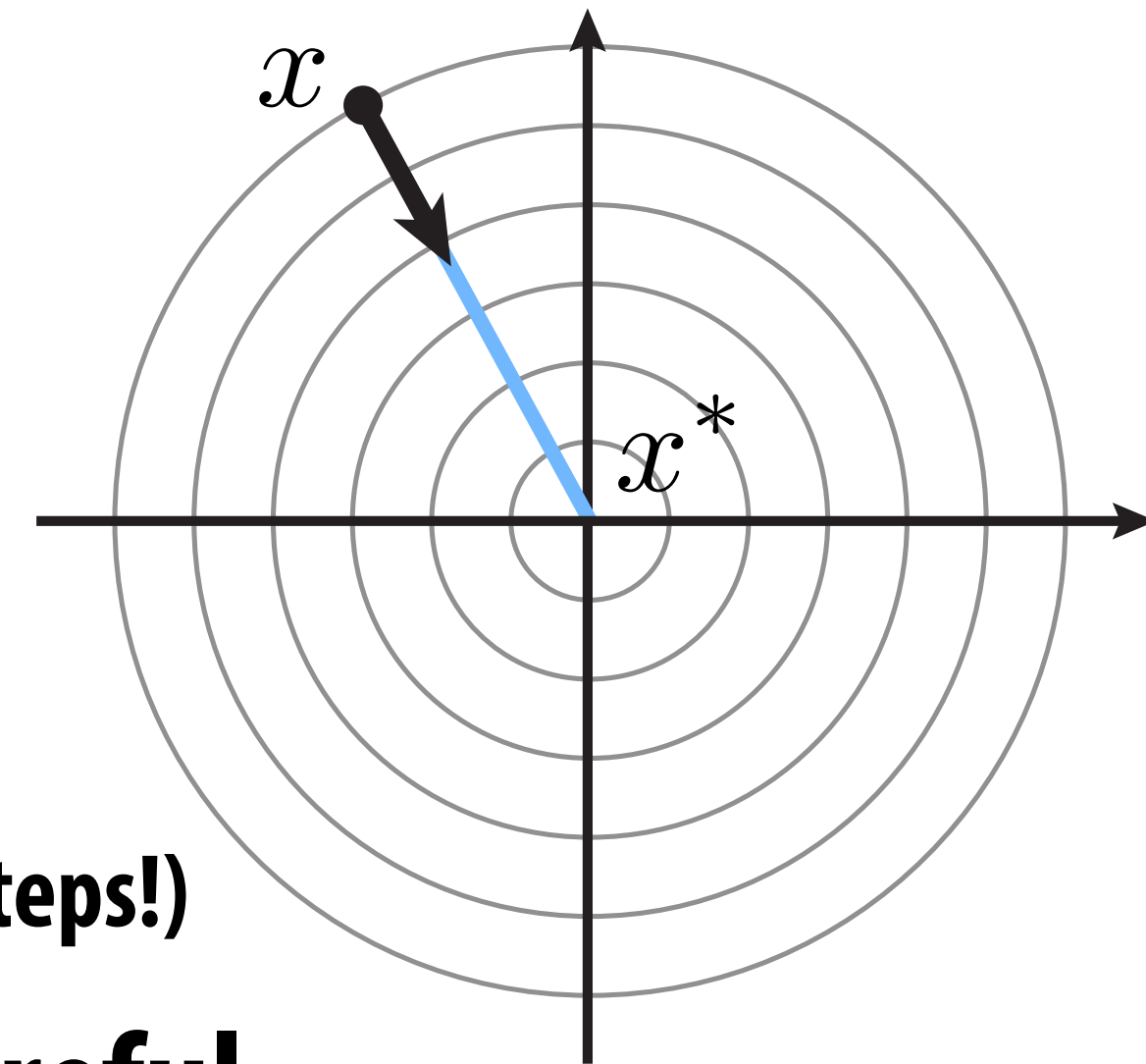
- solution can “oscillate”
- takes many, many small steps
- very slow to converge



# Higher Order Descent

- **General idea: apply a coordinate transformation so that the local energy landscape looks more like a “round bowl”**
- **Gradient now points directly toward nearby minimizer**
- **Most basic strategy: Newton’s method:**

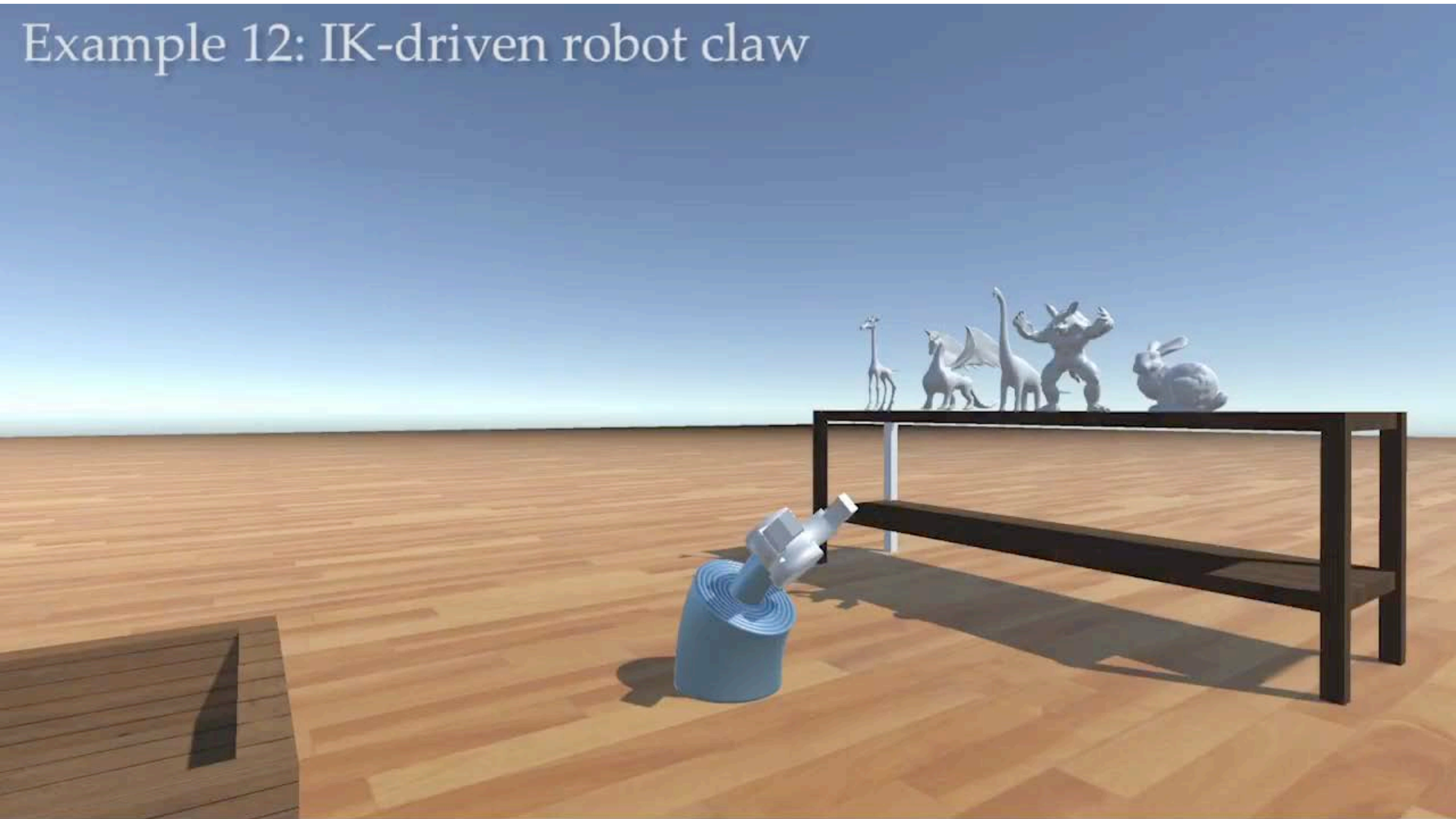
$$x_{k+1} = x_k - \underbrace{\tau (\nabla^2 f_0(x_k))^{-1}}_{\text{Hessian inverse}} \underbrace{\nabla f_0(x_k)}_{\text{gradient}}$$



- **Great for convex problems** (even proofs about # of steps!)
- **For nonconvex problems, need to be more careful**
- **In general, nonconvex optimization is a *BLACK ART***
- **Meta-strategy: try lots of solvers, see what works!**
  - **quasi-Newton, trust region, L-BFGS, ...**

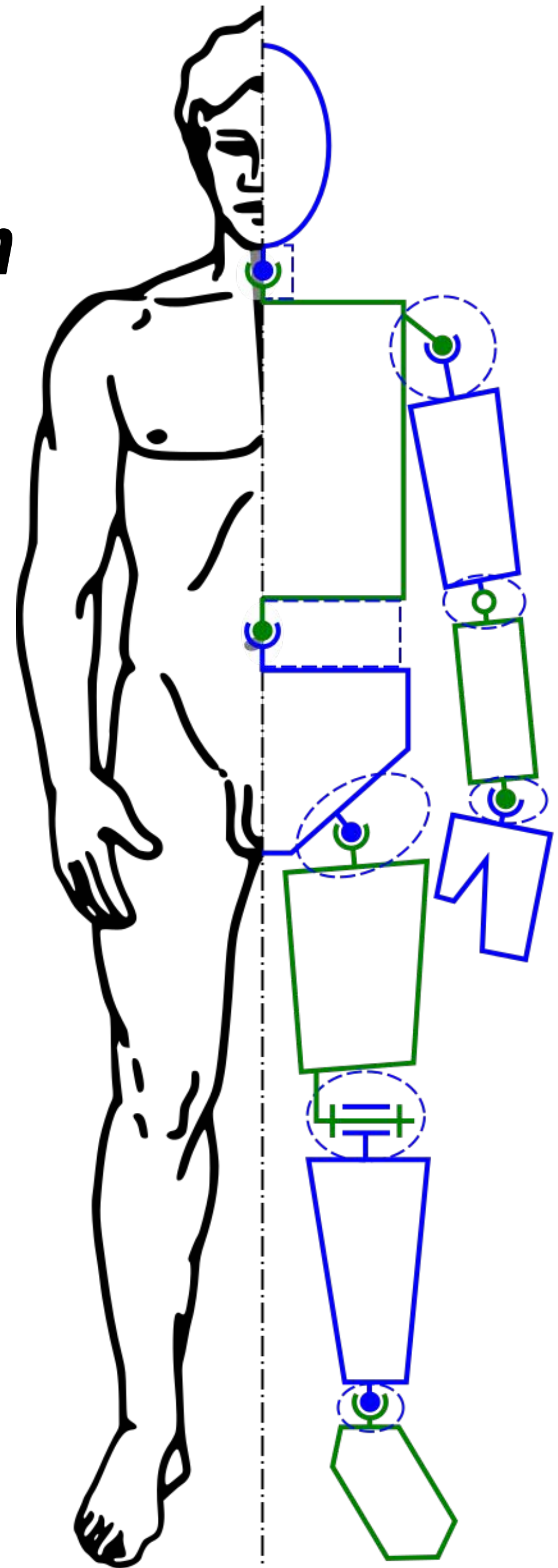
# Example: Inverse Kinematics

Example 12: IK-driven robot claw

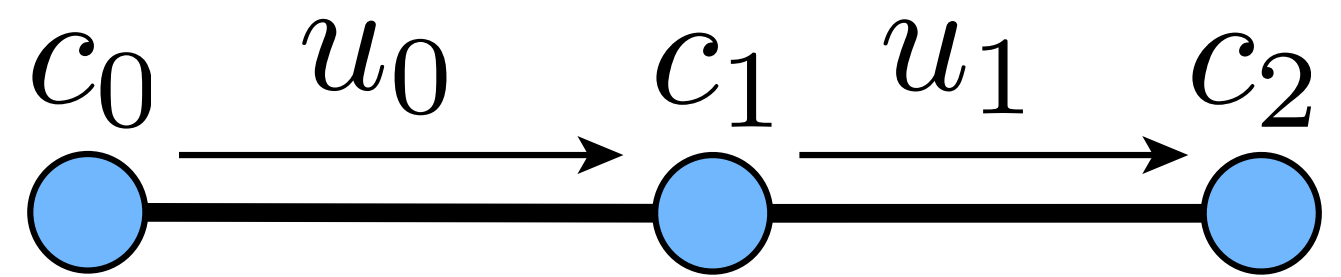


# Forward Kinematics

- **Many systems well-described by a *kinematic chain***
  - **collection of rigid bodies, connected by joints**
  - **joints have various behaviors (ball, piston, ...)**
  - **also have constraints (e.g., range of angles)**
  - **hierarchical structure (body → leg → foot)**
- **In animation, often called a *rig***
- **How do we specify the configuration of a “rig”?**
  - **One way: artist sets each joint individually**
  - **Another way: ...optimization!**



# Simple Kinematic Chain



- Consider a simple path-like chain in 2D
- Q: How do we write  $p_1$  in terms of the root position  $p_0$ , angles, & vectors  $u := c_{i+1} - c_i$ ?

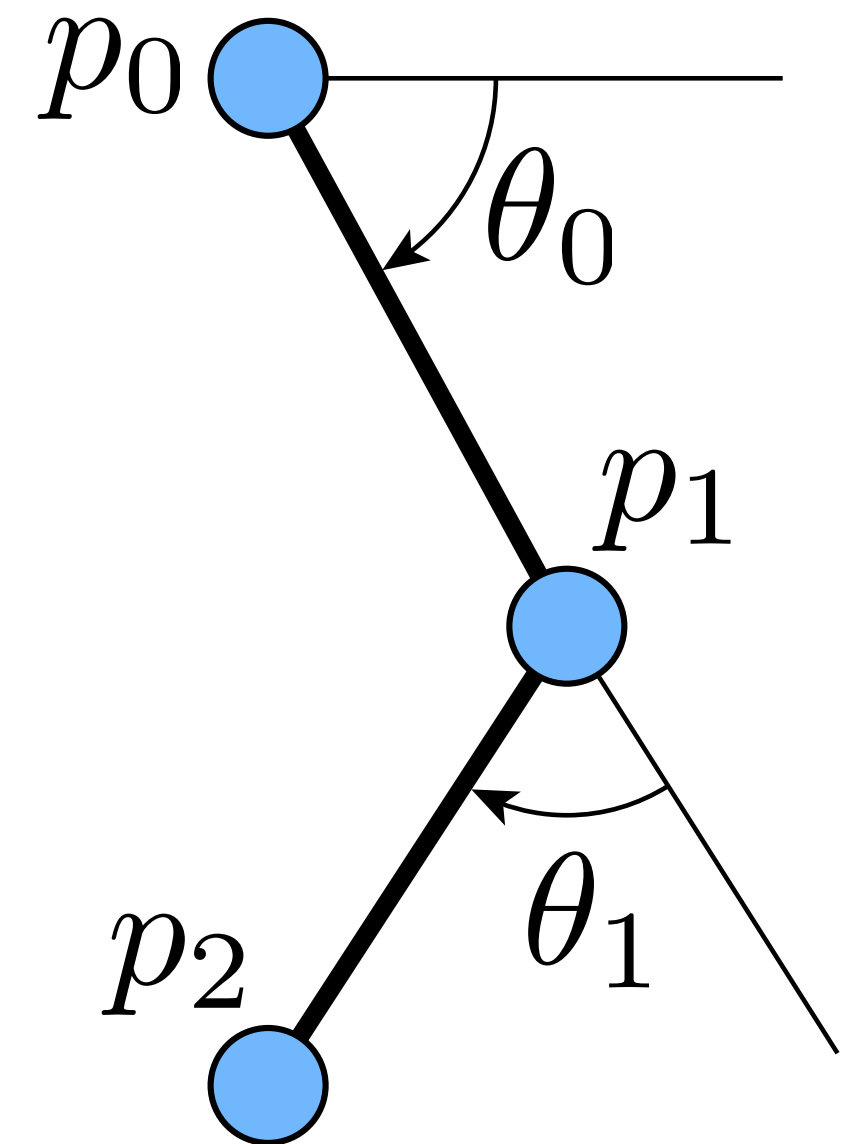
$$p_1 = p_0 + \begin{bmatrix} \cos \theta_0 & \sin \theta_0 \\ -\sin \theta_0 & \cos \theta_0 \end{bmatrix} u_0$$

- (For brevity, can use complex numbers:)

$$p_1 = p_0 + e^{i\theta_0} u_0$$

- Q: How about  $p_2$ ?

$$p_2 = p_0 + e^{i\theta_0} u_0 + e^{i\theta_0} e^{i\theta_1} u_1$$



# Simple IK Algorithm

- **Basic idea behind our IK algorithm:**
  - **write down distance between final point and “target”**
  - **compute gradient with respect to angles**
  - **apply gradient descent**

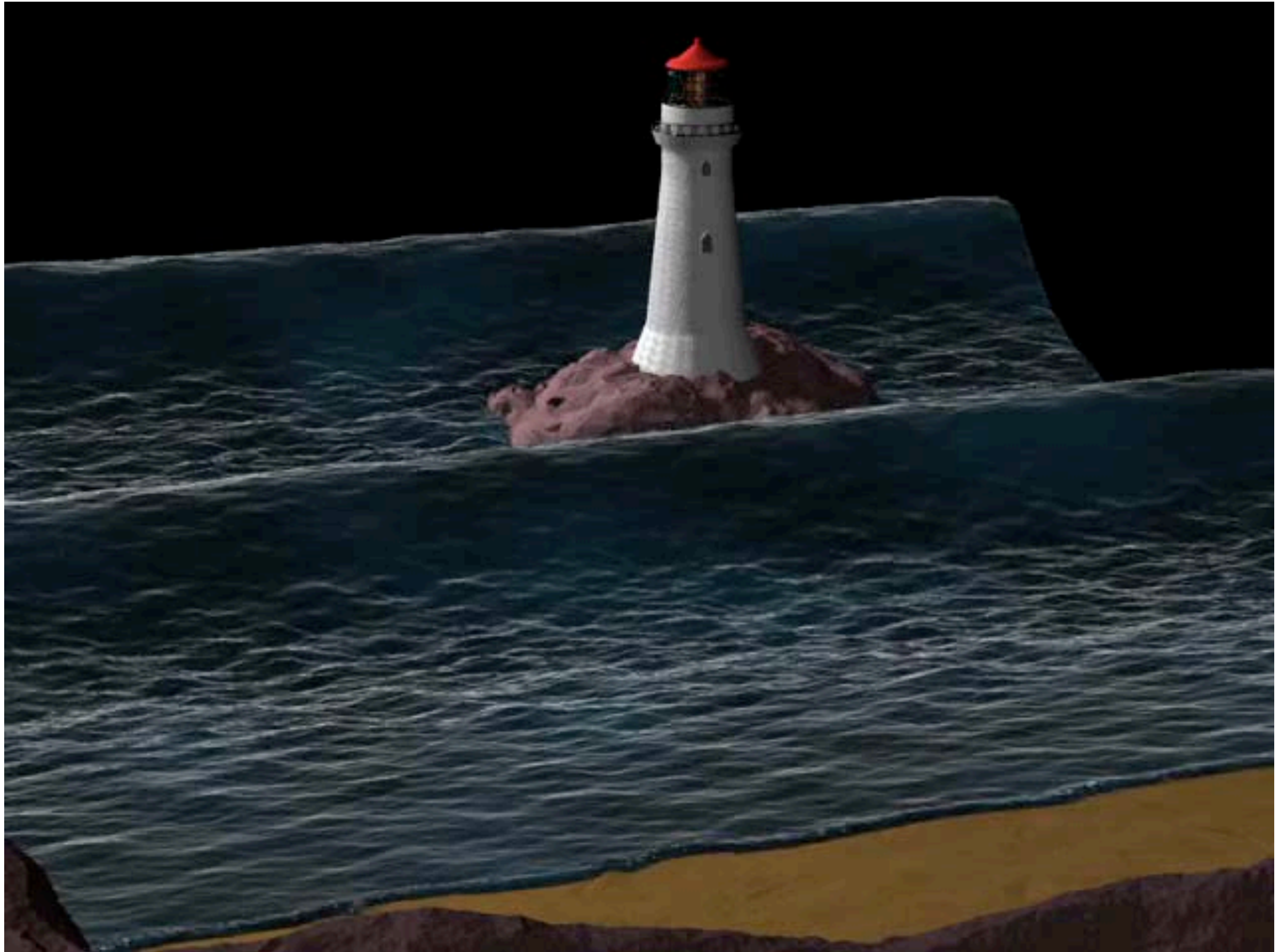
- **Objective?**

$$f_0(\theta) = \frac{1}{2} |\tilde{p}_n - p_n|^2$$

- **Constraints?**

- **None! The joint angle can take any value.**
- **Though we could limit joint angles (for instance)**

# Coming up next: PDEs in Computer Graphics



Frank Losasso, Jerry O. Talton, Nipun Kwatra, and Ron Fedkiw, *"Two-Way Coupled SPH and Particle Level Set Fluid Simulation"*