Dynamics and Time Integration

Computer Graphics
CMU 15-462/15-662

Last time: animation

- Added motion to our model
- Interpolate keyframes
- Still a lot of work!
- Today: physically-based animation
  - often less manual labor
  - often more compute-intensive
- Leverage tools from physics
  - dynamical descriptions
  - numerical integration
- Payoff: beautiful, complex behavior from simple models
- Widely-used techniques in modern film (and games!)

Dynamical Description of Motion

"A change in motion is proportional to the motive force impressed and takes place along the straight line in which that force is impressed."

—Sir Isaac Newton, 1687

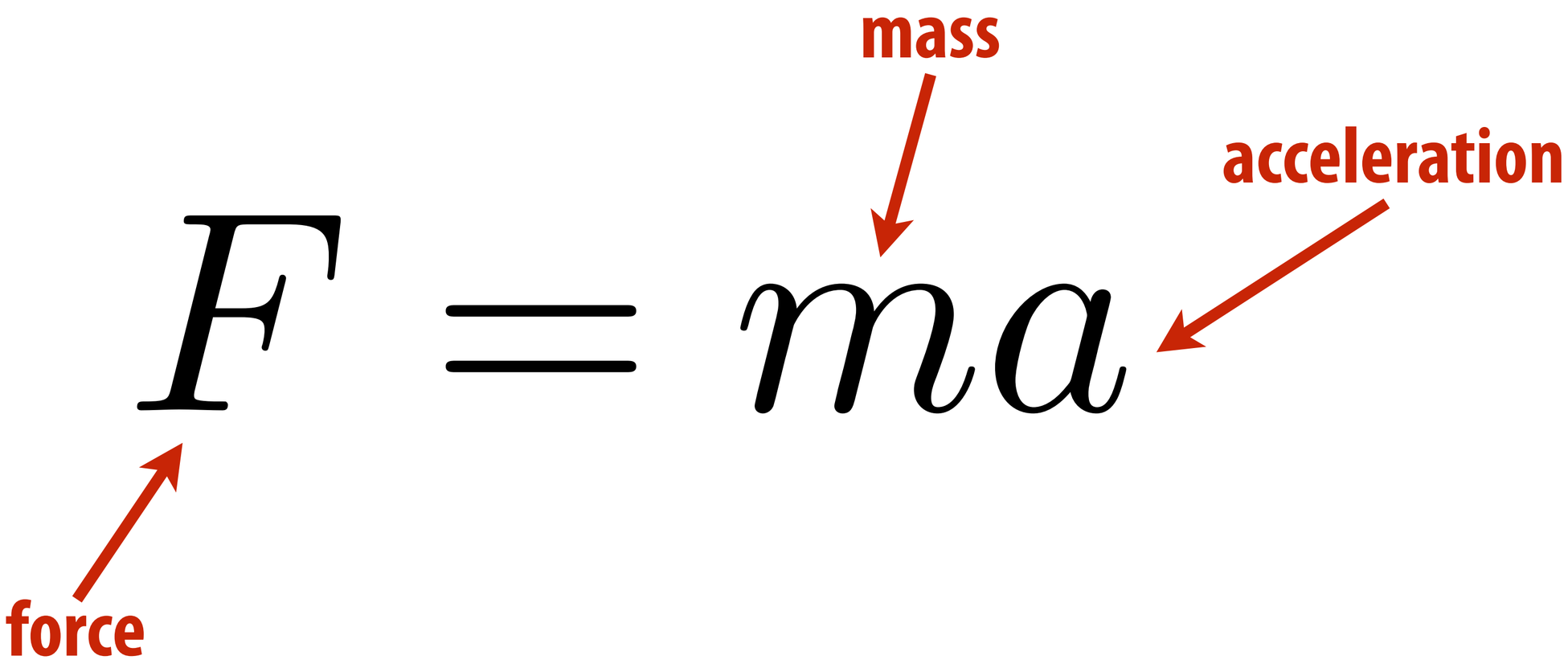
"Dynamics is concerned with the study of forces and their effect on motion, as opposed to kinematics, which studies the motion of objects without reference to its causes."

—Sir Wiki Pedia, 2015

(Q: Is keyframe interpolation dynamic, or kinematic?)

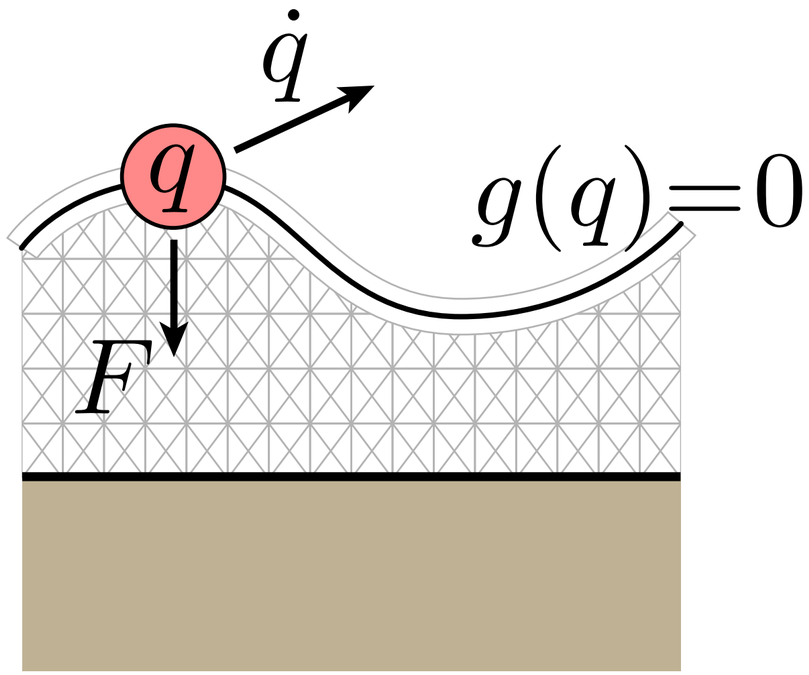
The Animation Equation

- Already saw the rendering equation
- What's the animation equation?



The "Animation Equation," revisited

- Well actually there are some more equations...
- Let's be more careful:
  - Any system has a configuration $q(t)$
  - It also has a velocity $\dot{q} := \frac{d}{dt} q$
  - And some kind of mass M
  - There are probably some forces F
  - And also some constraints $g(q, \dot{q}, t) = 0$

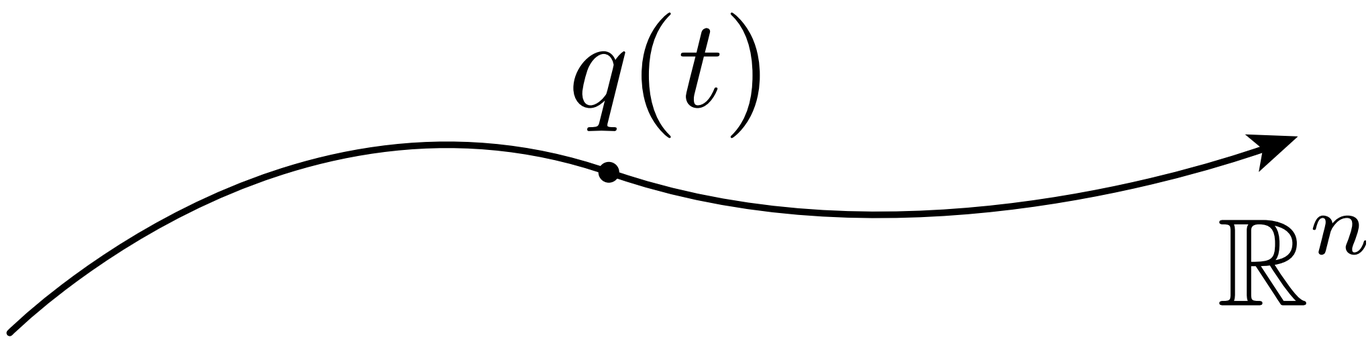


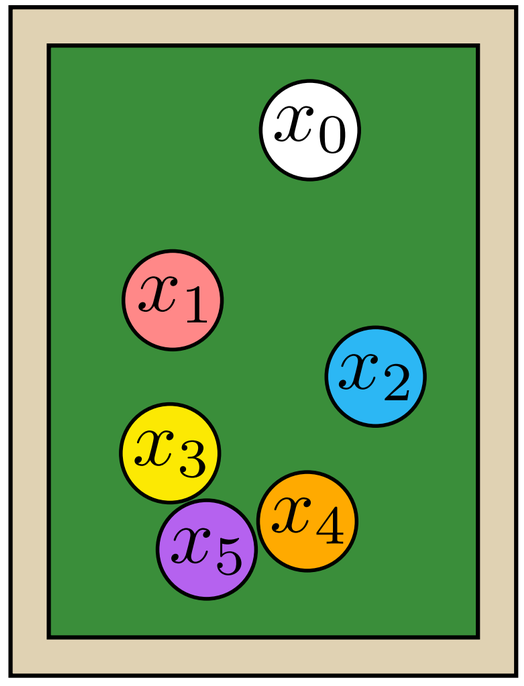
- E.g., could write Newton's 2nd law as $\ddot{q} = F/m$
- Makes two things clear:
  - acceleration is 2nd time derivative of configuration
  - ultimately, we want to solve for the configuration q

Generalized Coordinates

- Often describing systems with many, many moving pieces
- E.g., a collection of billiard balls, each with position x_i
- Collect them all into a single vector of generalized coordinates:

$$q = (x_0, x_1, \ldots, x_n)$$



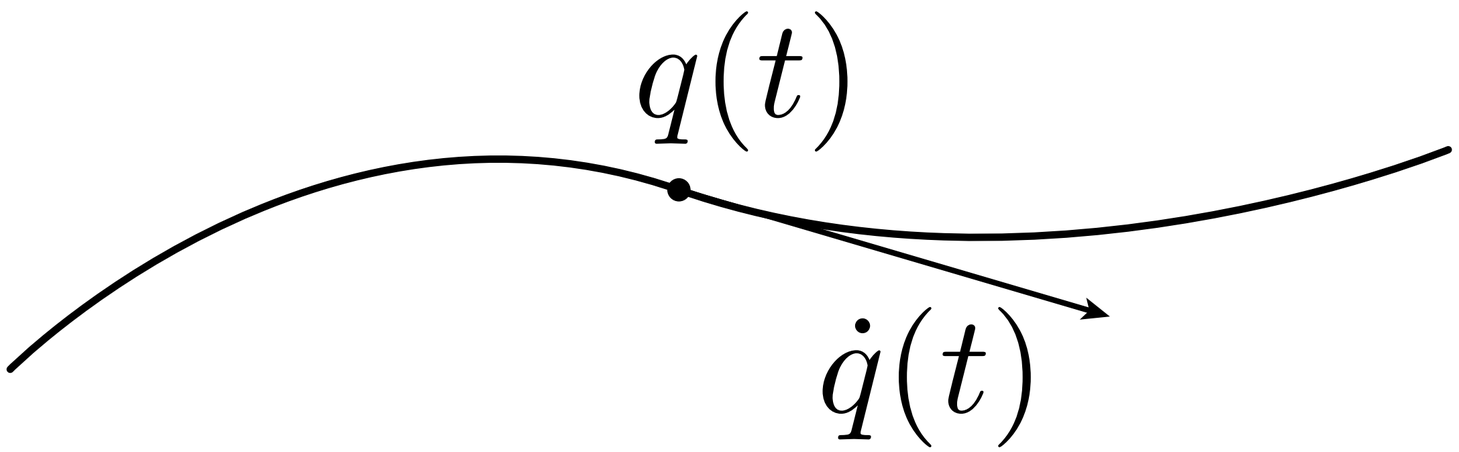


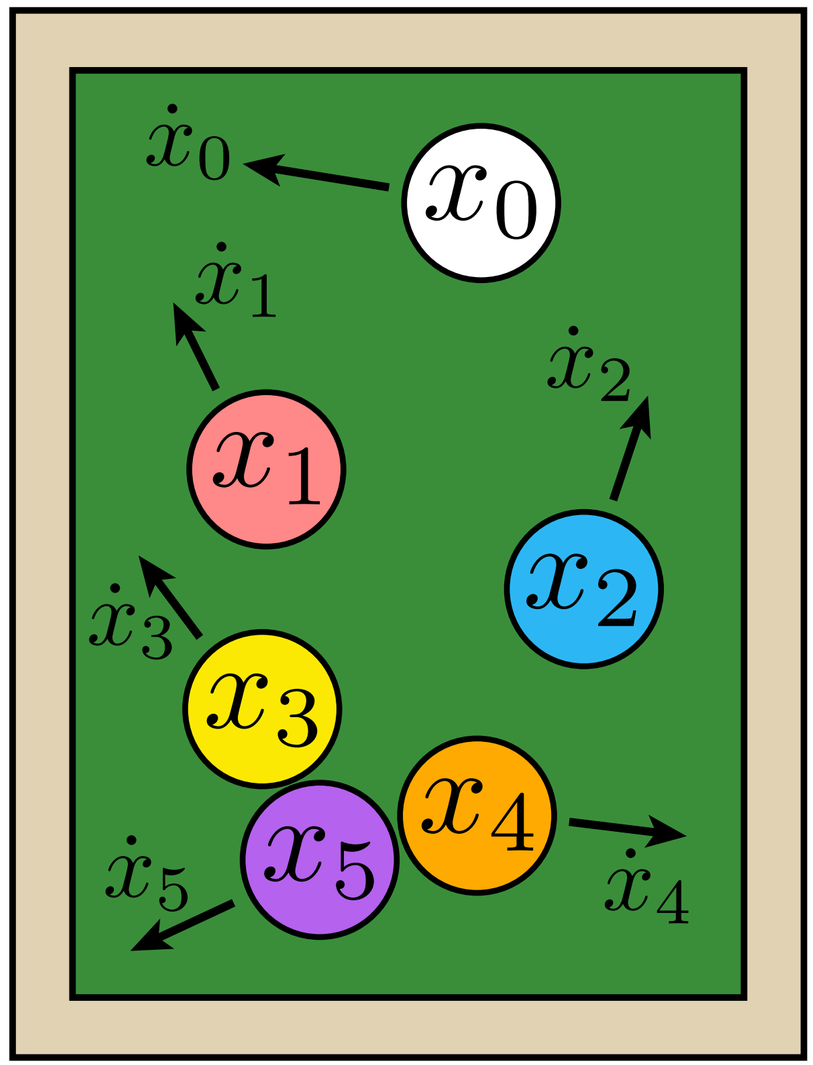
- Can think of q as a single point moving along a trajectory in R^n
- This way of thinking naturally maps to the way we actually solve equations on a computer: all variables are often "stacked" into a big long vector and handed to a solver.
- (…So why not write things down this way in the first place?)

Generalized Velocity

- **Not much more to say about generalized velocity: it's the time derivative of the generalized coordinates!**

$$\dot{q} = (\dot{x}_0, \dot{x}_1, \ldots, \dot{x}_n)$$





All of life (and physics) is just traveling along a curve...

Ordinary Differential Equations

- Many dynamical systems can be described via an ordinary differential equation (ODE) in generalized coordinates:

change in configuration over time

velocity function

$$\frac{d}{dt}q = f(q, \dot{q}, t)$$

- ODE doesn't have to describe mechanical phenomenon, e.g.,

$$\frac{d}{dt}u(t) = au$$

"rate of growth is proportional to value"

- Solution? $u(t) = be^{at}$
- Describes exponential decay (a < 1), or really great stock (a > 1)
- "Ordinary" means "involves derivatives in time but not space"
- We'll talk about spatial derivatives (PDEs) in another lecture...

Dynamics via ODEs

- Another key example: Newton's 2nd law!

$$\ddot{q} = F/m$$

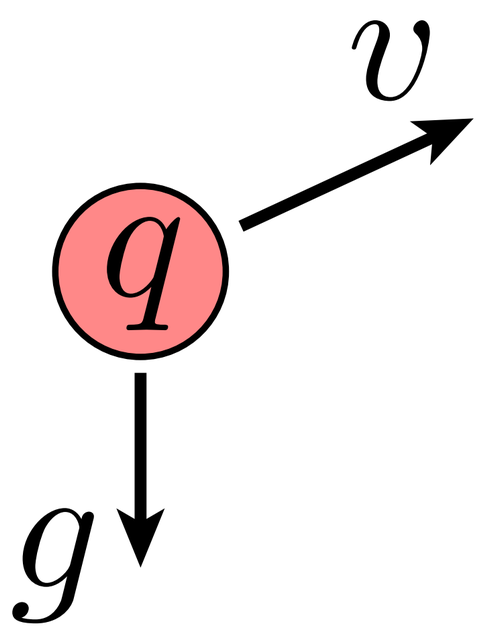
- "Second order" ODE since we take two time derivatives
- Can also write as a system of two first order ODEs, by introducing new "dummy" variable for velocity:

$$\dot{q} = v$$
$$\dot{v} = F/m$$

- Splitting things up this way will make it easy to talk about solving these equations numerically (among other things)

Simple Example: Throwing a Rock

- Consider a rock* of mass m tossed under force of gravity g
- Easy to write dynamical equations, since only force is gravity:



$$\ddot{q} = g/m \quad \textbf{or} \quad \begin{aligned} \dot{q} &= v \\ \dot{v} &= g/m \end{aligned}$$

Solution:

$$v(t) = v_0 + \frac{t}{m} g$$

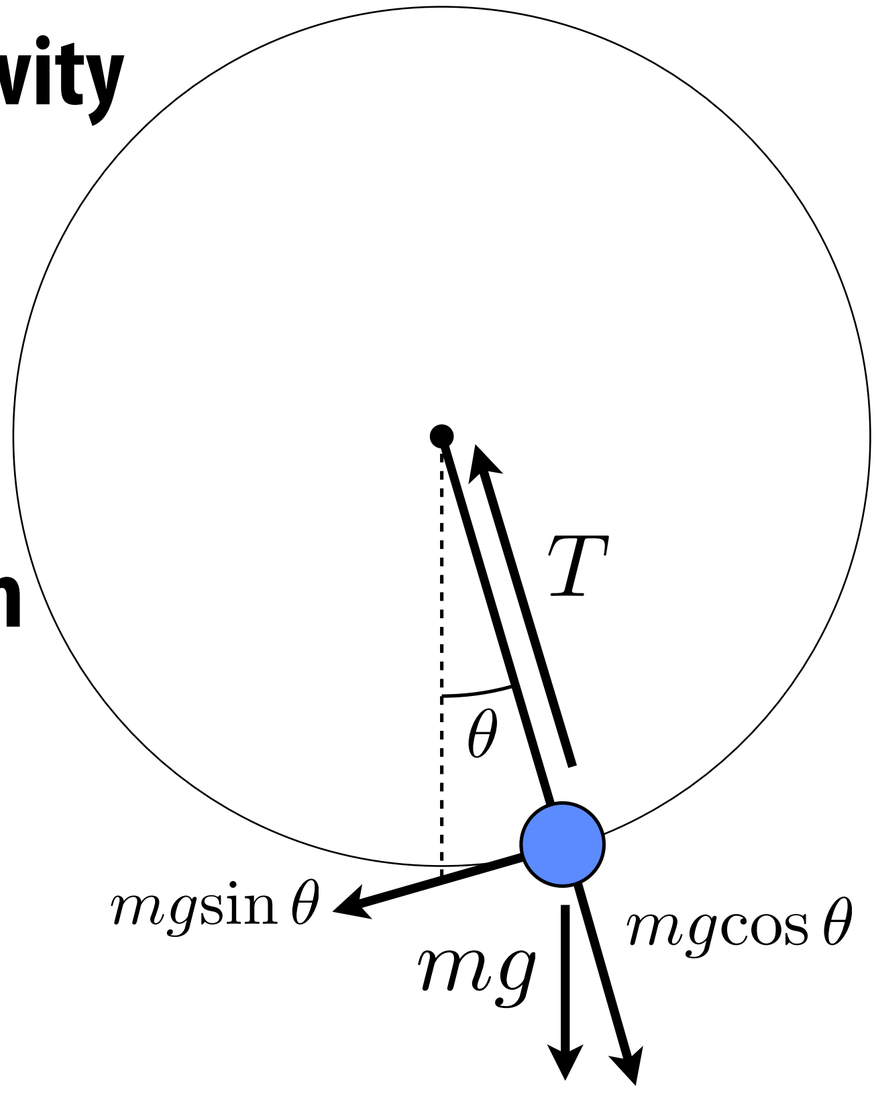
$$q(t) = q_0 + t v_0 + \frac{t^2}{2m} g$$

(What do we need a computer for?!)

*Yes, this rock is spherical and has uniform density.

Slightly Harder Example: Pendulum

- Mass on end of a bar, swinging under gravity
- What are the equations of motion?
- Same as "rock" problem, but constrained
- Could use a "force diagram"
  - You probably did this for many hours in high school/college
  - Let's do something new & different!



Lagrangian Mechanics

Joe Lagrange

- **Beautifully simple recipe:**
  1. **Write down kinetic energy** K
  2. **Write down potential energy** U
  3. **Write down Lagrangian** $\mathcal{L} := K - U$
  4. **Dynamics then given by Euler-Lagrange equation**

becomes (generalized) "MASS TIMES ACCELERATION"

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} = \frac{\partial \mathcal{L}}{\partial q}$$

becomes (generalized) "FORCE"

- **Why is this useful?**
  - **often easier to come up with (scalar) energies than forces**
  - **very general, works in any kind of generalized coordinates**
  - **helps develop nice class of numerical integrators (symplectic)**

Great reference: Sussman & Wisdom, "Structure and Interpretation of Classical Mechanics"

Lagrangian Mechanics - Example

- **Generalized coordinates for pendulum?**

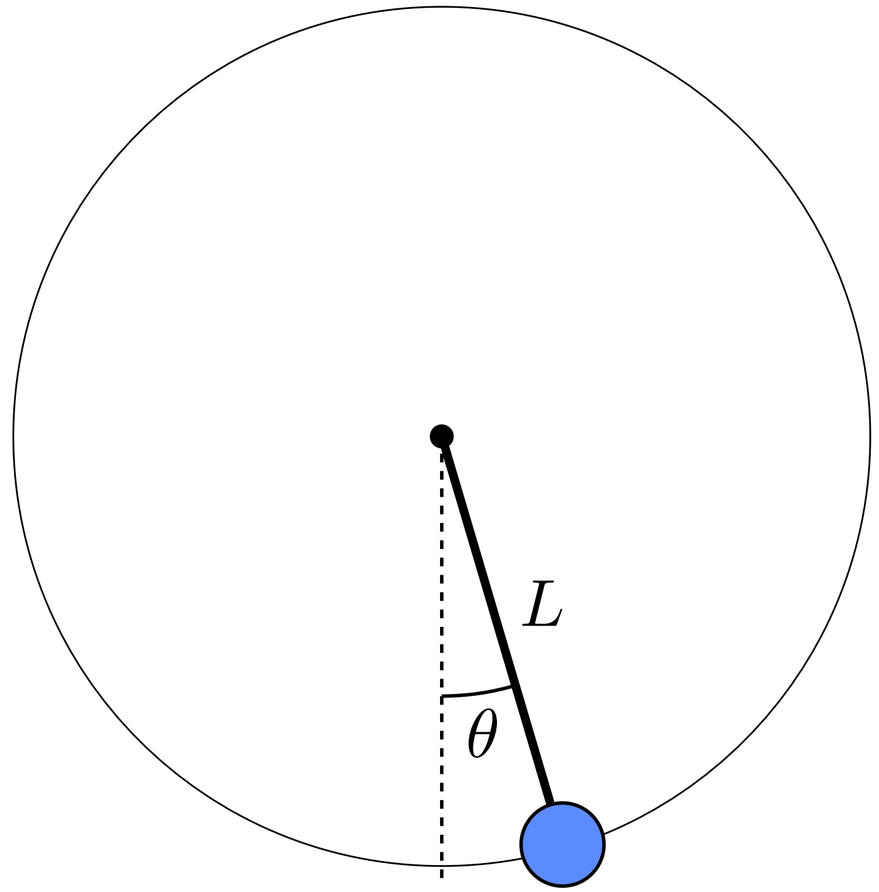
$$q = \theta$$ ← just one coordinate: angle with the vertical direction

- **Kinetic energy (mass m)?**

$$K = \frac{1}{2} I \omega^2 = \frac{1}{2} m L^2 \dot{\theta}^2$$

- **Potential energy?**

$$U = mgh = -mgL\cos\theta$$



- **Euler-Lagrange equations?** (from here, just "plug and chug"—even a computer could do it!)

$$\mathcal{L} = K - U = m(\frac{1}{2} L^2 \dot{\theta}^2 + gL\cos\theta)$$

$$\frac{\partial \mathcal{L}}{\partial \dot{q}} = \frac{\partial \mathcal{L}}{\partial \dot{\theta}} = mL^2\dot{\theta} \qquad \frac{\partial \mathcal{L}}{\partial q} = \frac{\partial \mathcal{L}}{\partial \theta} = -mgL\sin\theta$$

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} = \frac{\partial \mathcal{L}}{\partial q} \quad \Rightarrow \quad \boxed{\ddot{\theta} = -\frac{g}{L}\sin\theta}$$

Solving the Pendulum

- Great, now we have a nice simple equation for the pendulum:

$$\ddot{\theta} = -\frac{g}{L}\sin\theta$$

- For small angles (e.g., clock pendulum) can approximate as

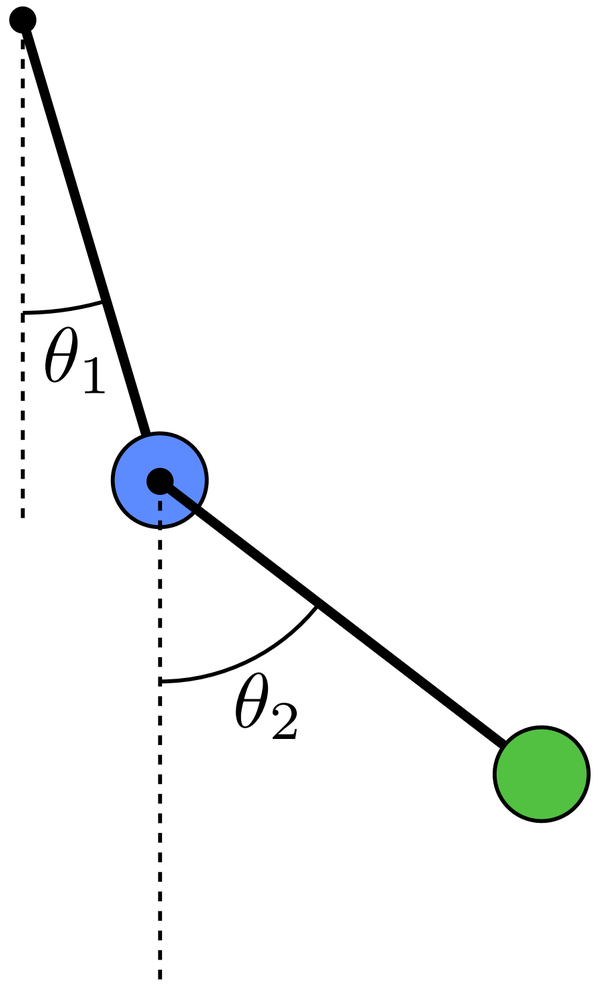
$$\ddot{\theta} = -\frac{g}{L}\theta \qquad \Rightarrow \theta(t) = a\cos(t\sqrt{g/L} + b)$$

"harmonic oscillator"

- In general, there is no closed form solution!
- Hence, we must use a numerical approximation
- ...And this was (almost) the simplest system we can think of!
- (What if we want to animate something more interesting?)

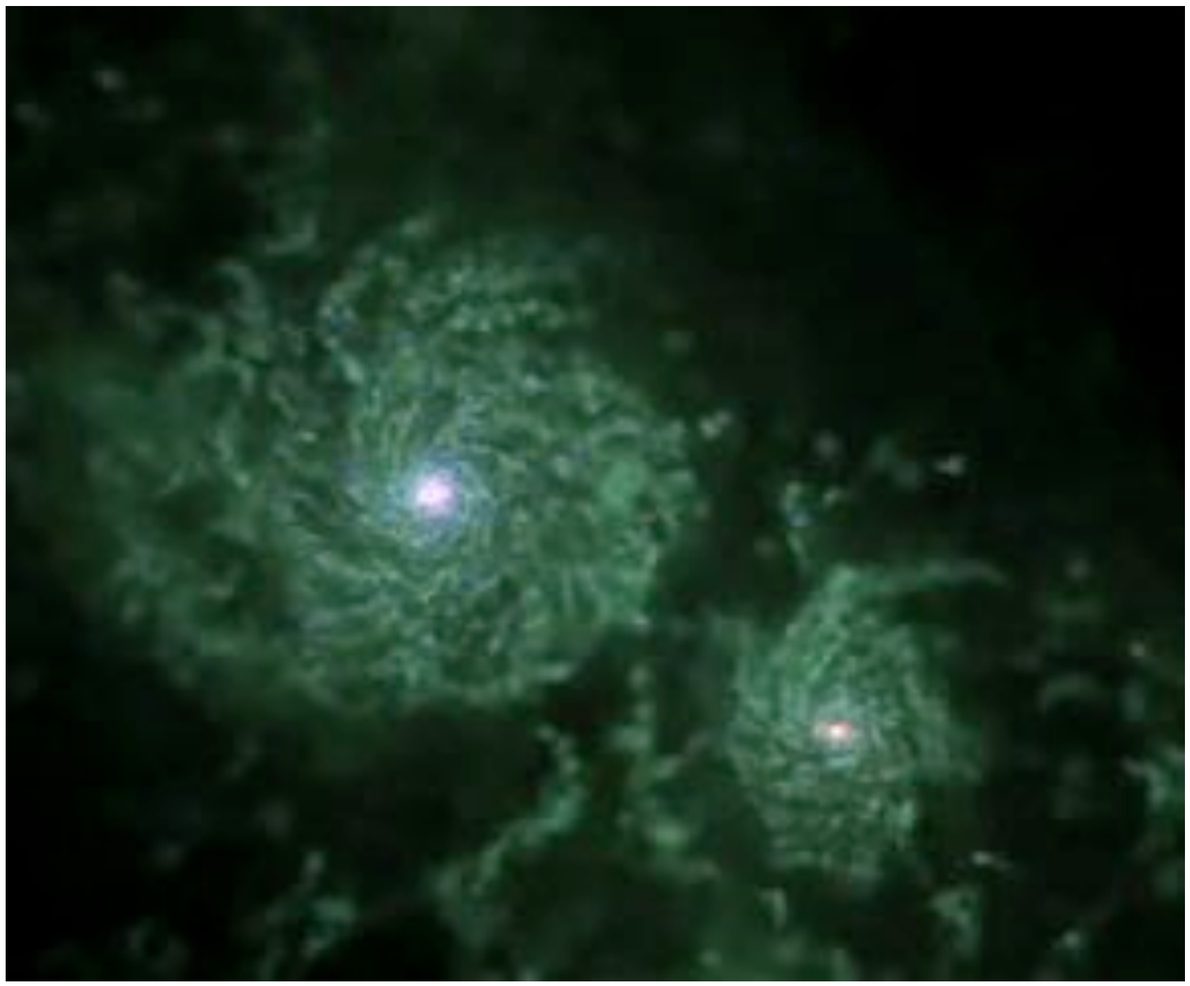
Not-So-Simple Example: Double Pendulum

- Blue ball swings from fixed point; green ball swings from blue one
- Simple system… not-so-simple motion!
- Chaotic: perturb input, wild changes to output
- Must again use numerical approximation

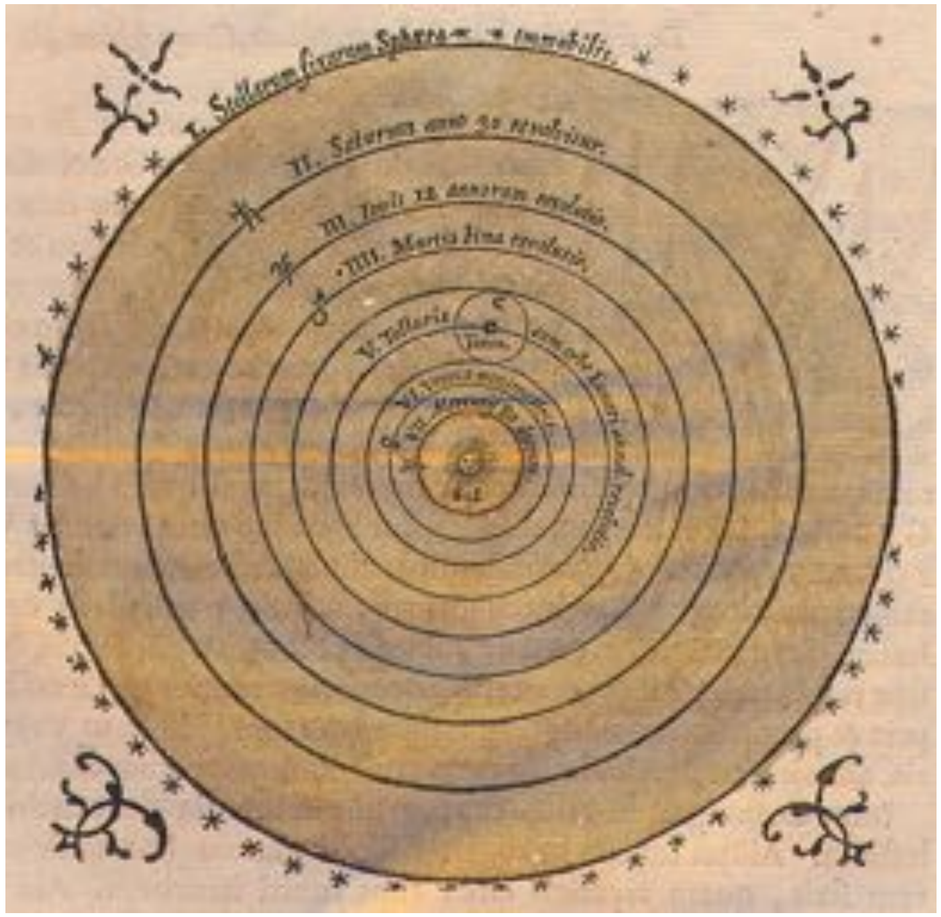


Not-So-Simple Example: n-Body Problem

- Consider the Earth, moon, and sun—where do they go?
- Solution is trivial for two bodies (e.g., assume one is fixed)
- As soon as $n \geq 3$, again get chaotic solutions (no closed form)
- What if we want to simulate entire galaxies?



Credit: Governato et al / NASA

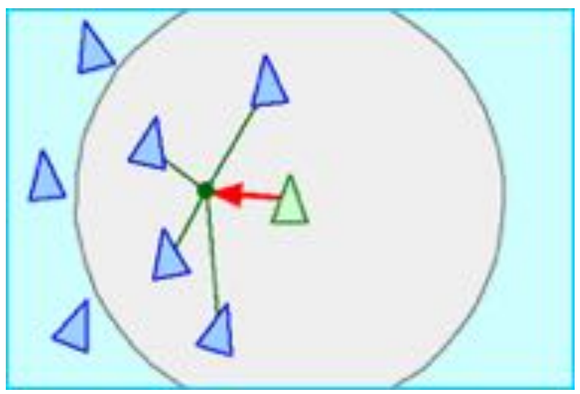


For animation, we want to simulate these kinds of phenomena!

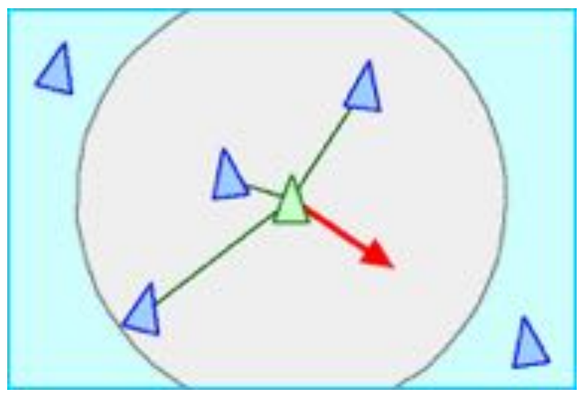
Example: Flocking

Simulated Flocking as an ODE

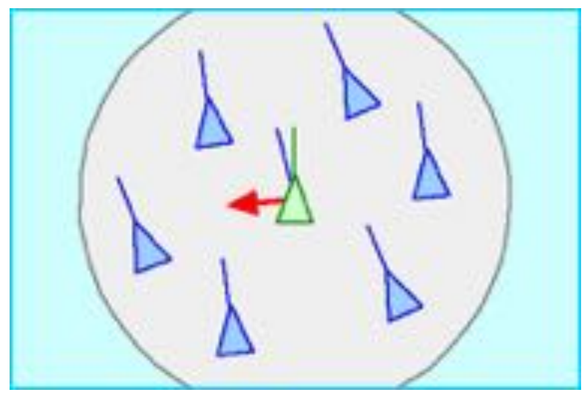
- Each bird is a particle
- Subject to very simple forces:
  - attraction to center of neighbors
  - repulsion from individual neighbors
  - alignment toward average trajectory of neighbors
- Solve large system of ODEs (numerically!)
- Emergent complex behavior (also seen in fish, bees, …)



attraction



repulsion



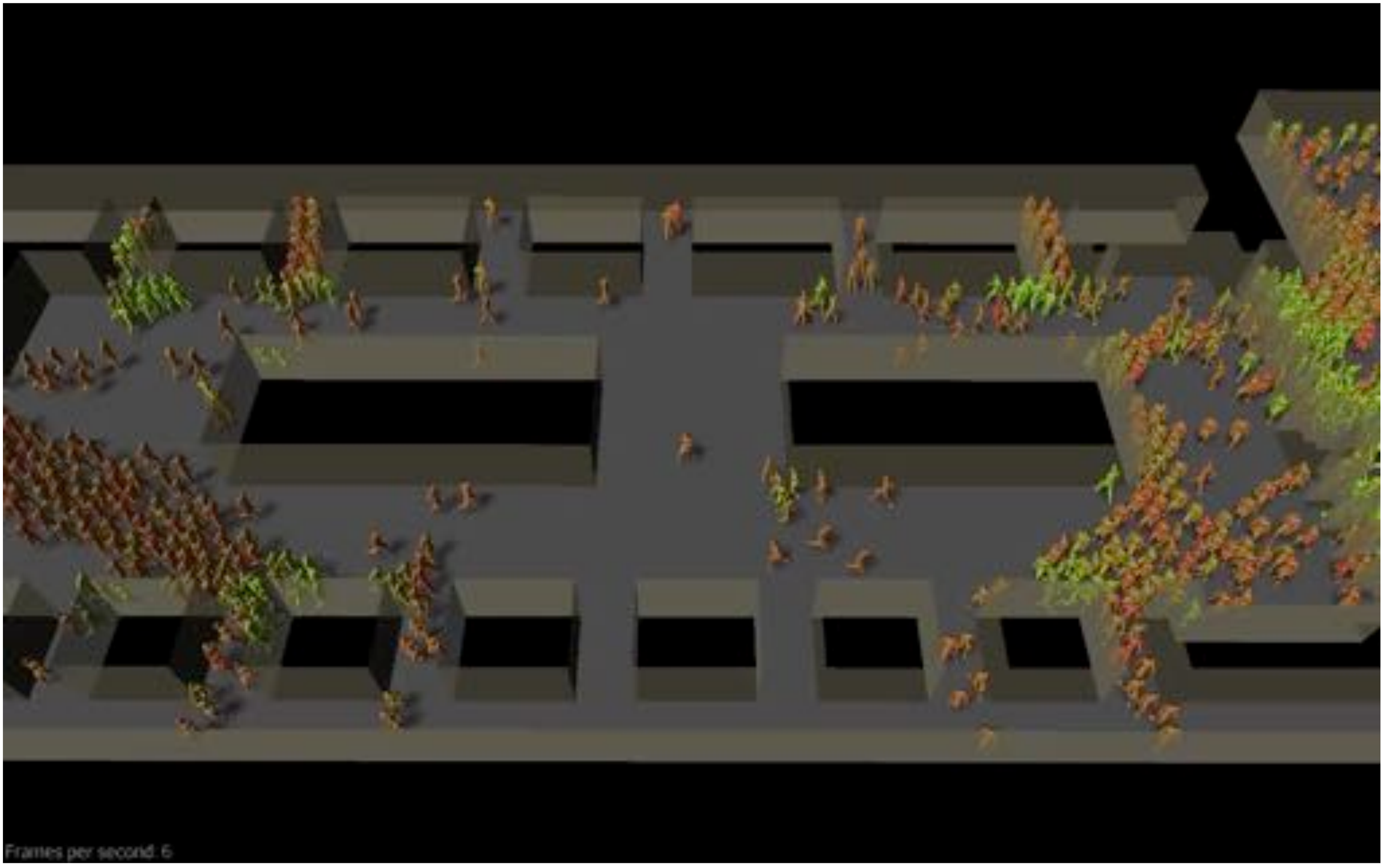
alignment

Credit: Craig Reynolds (see http://www.red3d.com/cwr/boids/)

Particle Systems

- More generally, model phenomena as large collection of particles
- Each particle has a behavior described by (physical or non-physical) forces
- Extremely common in graphics/games
  - easy to understand
  - simple equation for each particle
  - easy to scale up/down
- May need many particles to capture certain phenomena (e.g., fluids)
  - may require fast hierarchical data structure (kd-tree, BVH, ...)
  - often better to use continuum model

Example: Crowds



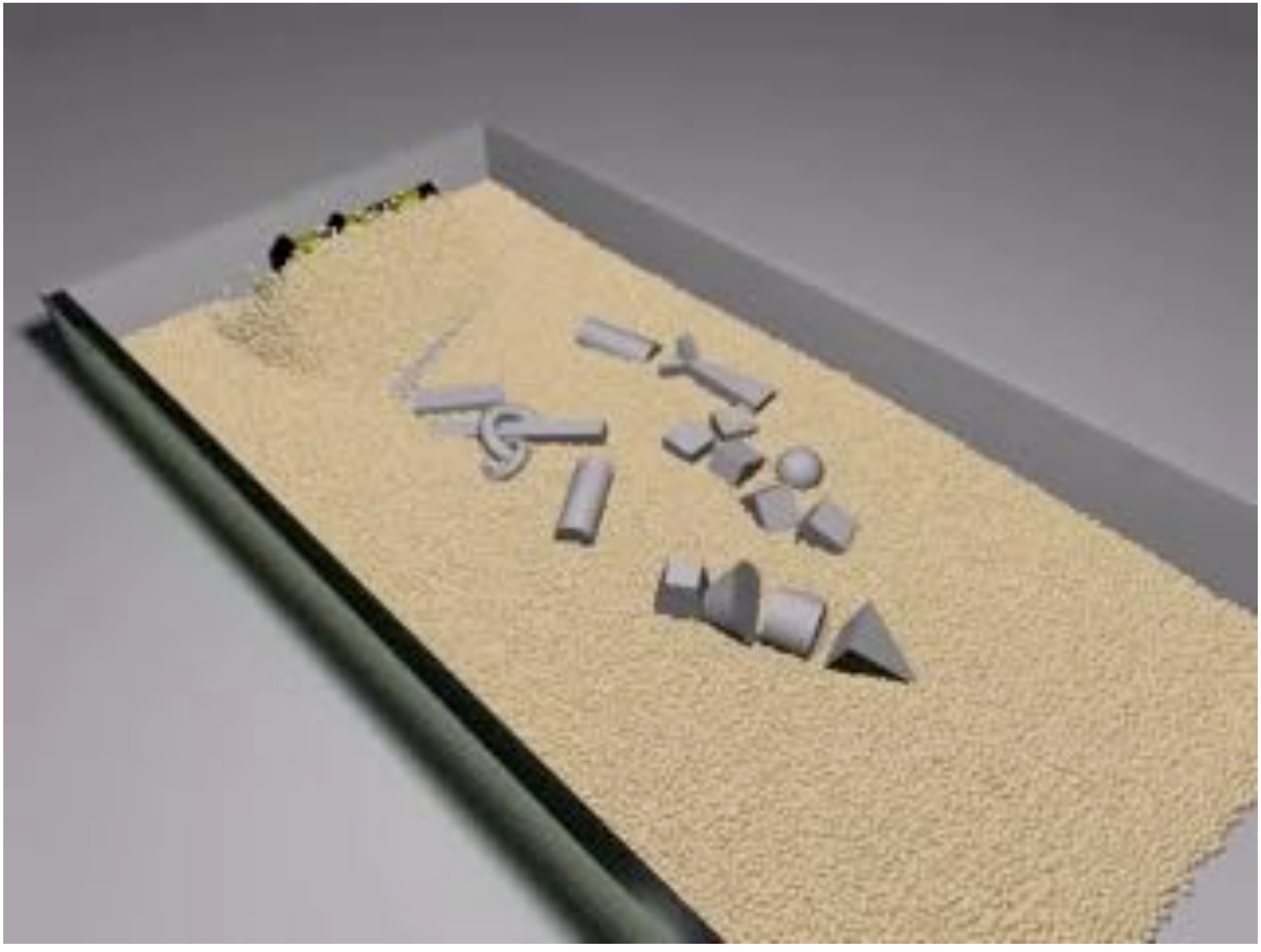
Where are the bottlenecks in a building plan?

Example: Crowds + “Rock” Dynamics

Example: Particle-Based Fluids

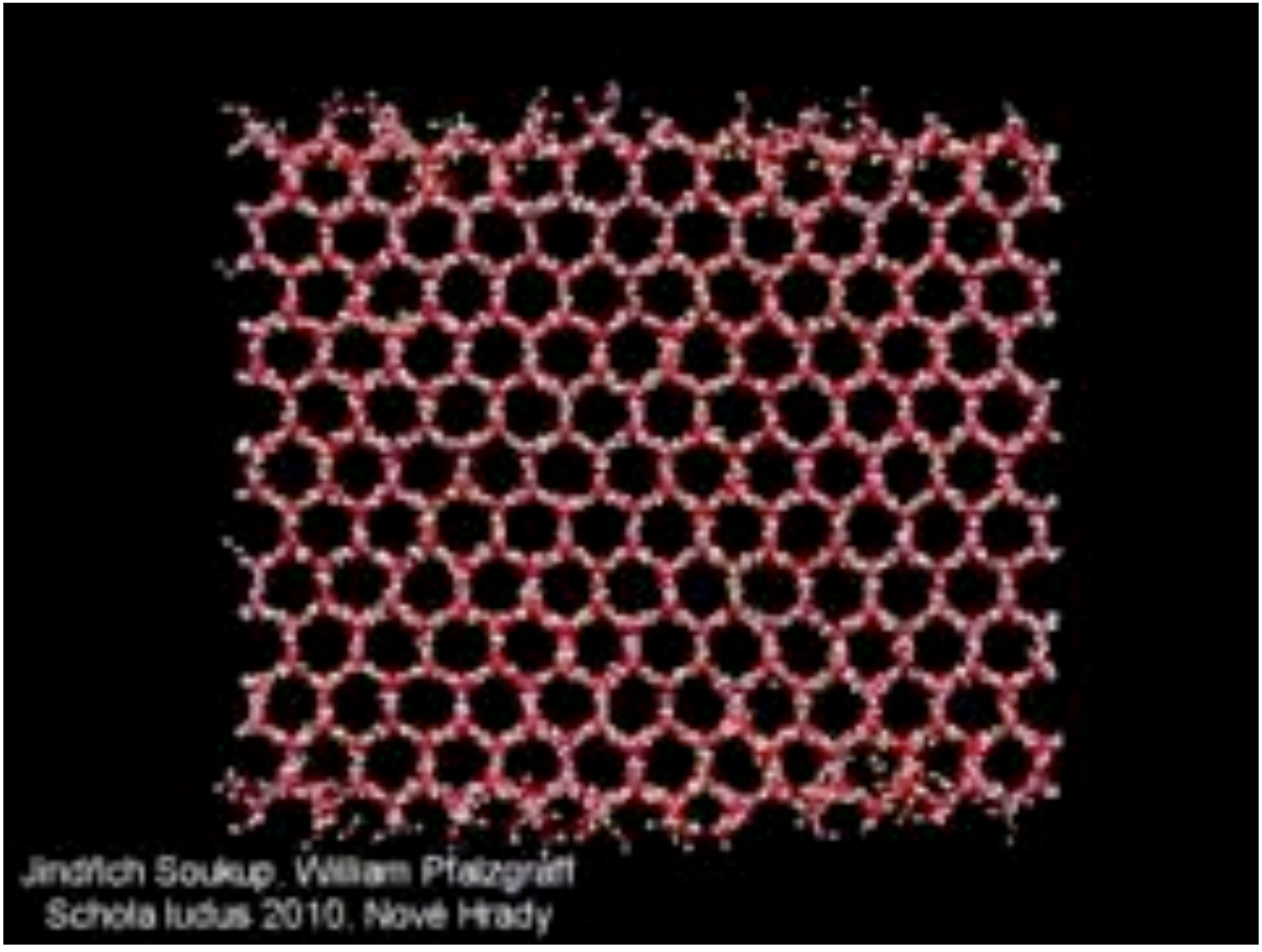
(Fluid: particles or continuum?)

Example: Granular Materials



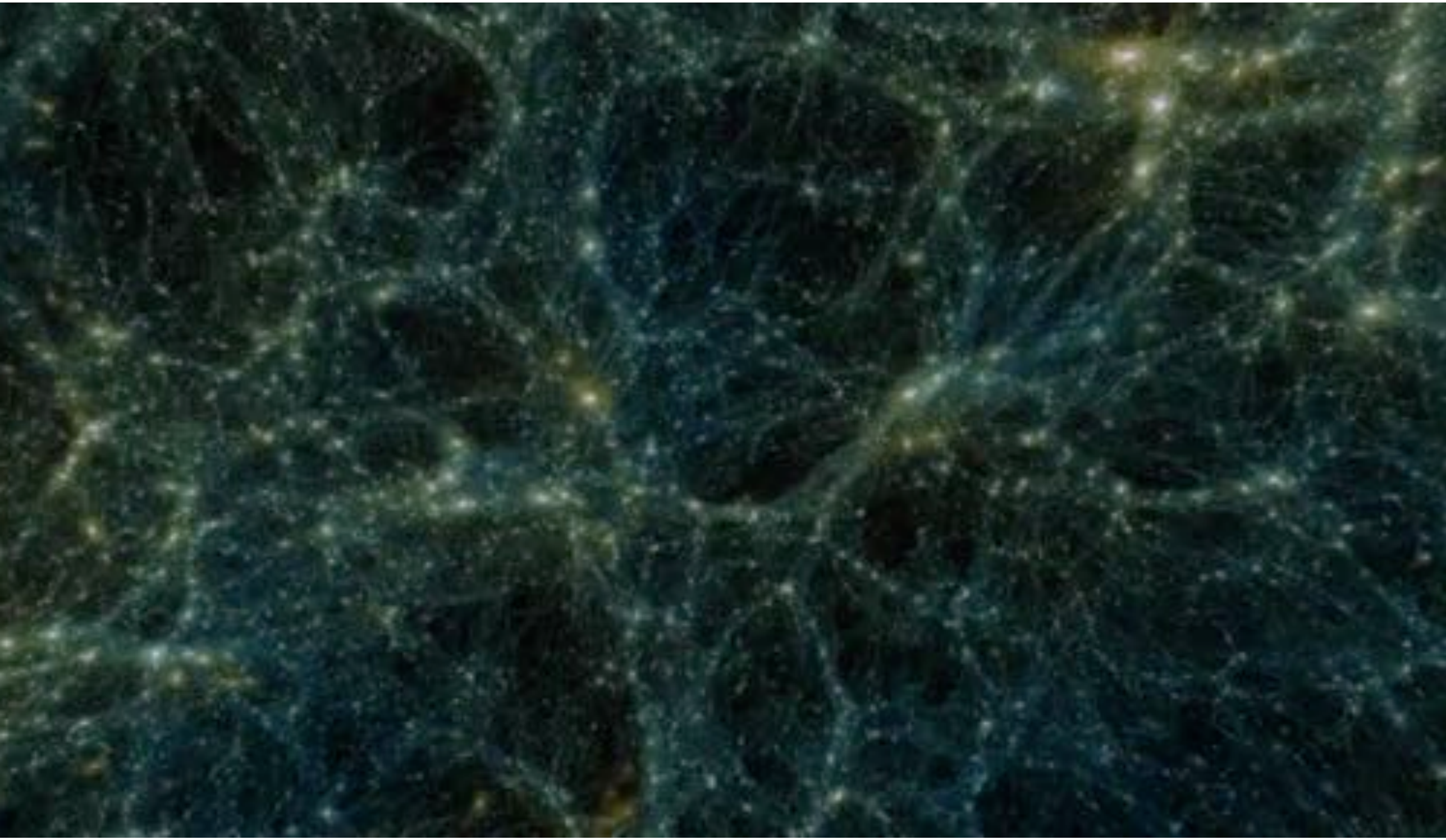
Bell et al, "Particle-Based Simulation of Granular Materials"

Example: Molecular Dynamics



(model of melting ice crystal)

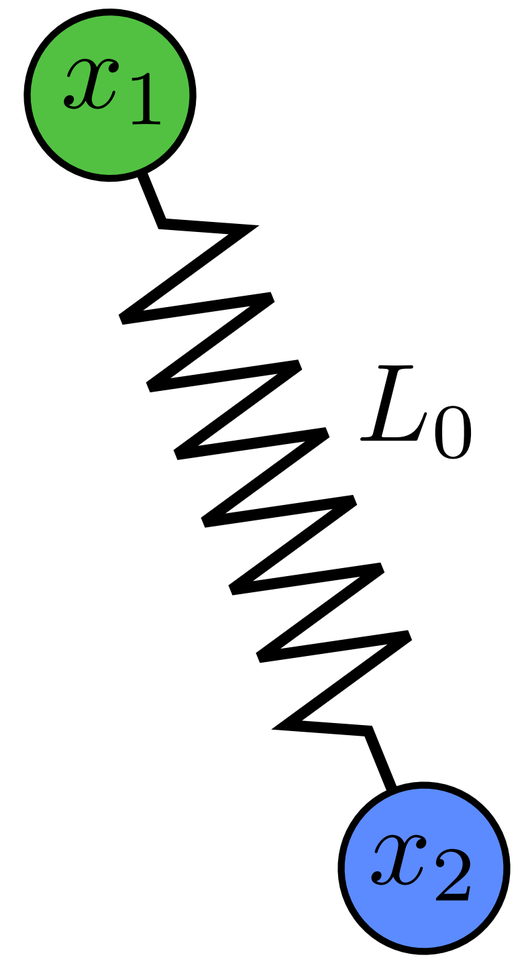
Example: Cosmological Simulation



Tomoaki et al - v²GC simulation of dark matter (~1 trillion particles)

Example: Mass-Spring System

- Connect particles x_1, x_2 by a spring of length L_0
- Potential energy is given by



rest length

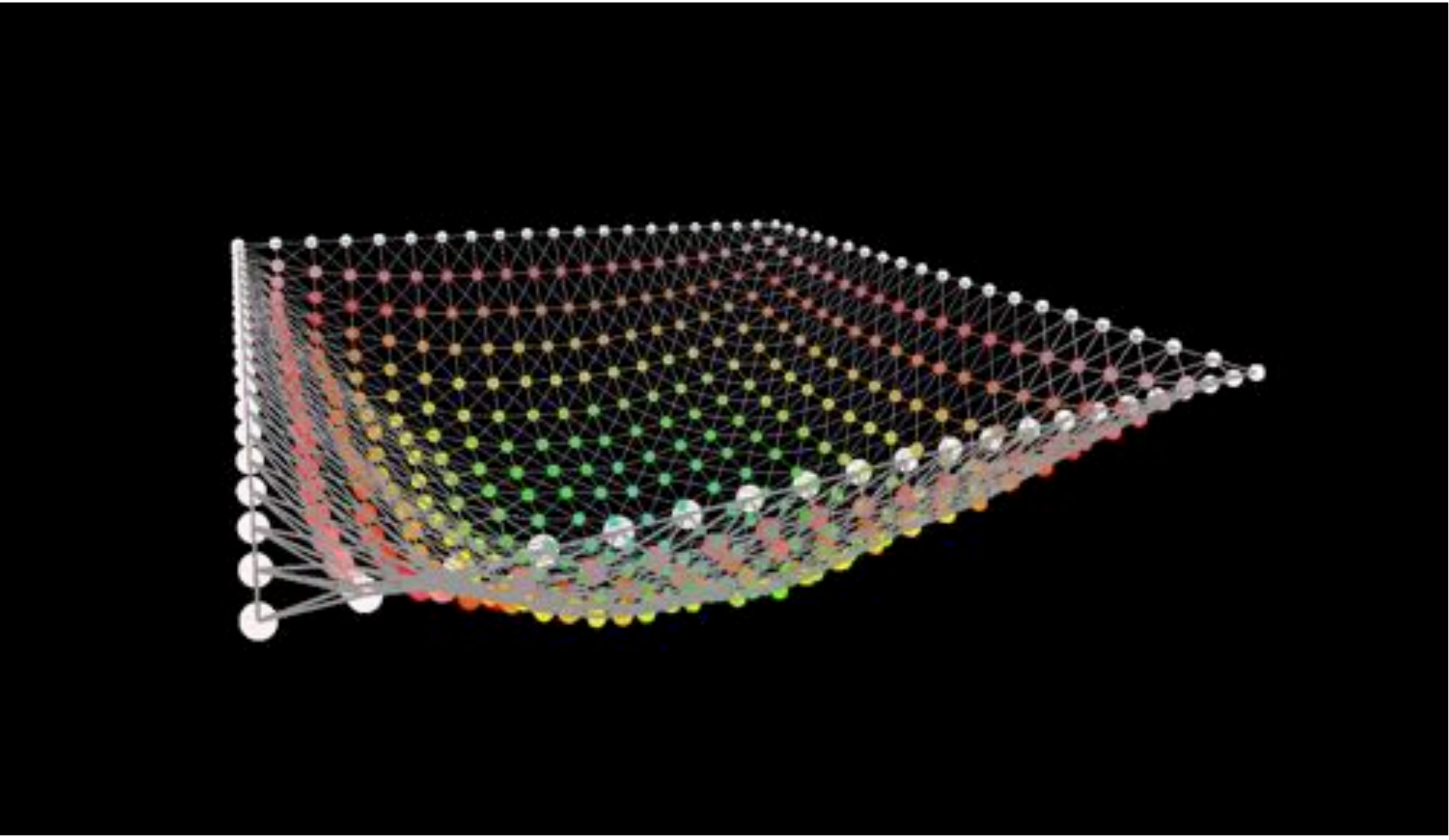
$$U = \frac{1}{2}k(L - L_0)^2$$

stiffness current length

$$= \frac{1}{2}k(|x_1 - x_2|^2 - L_0)^2$$

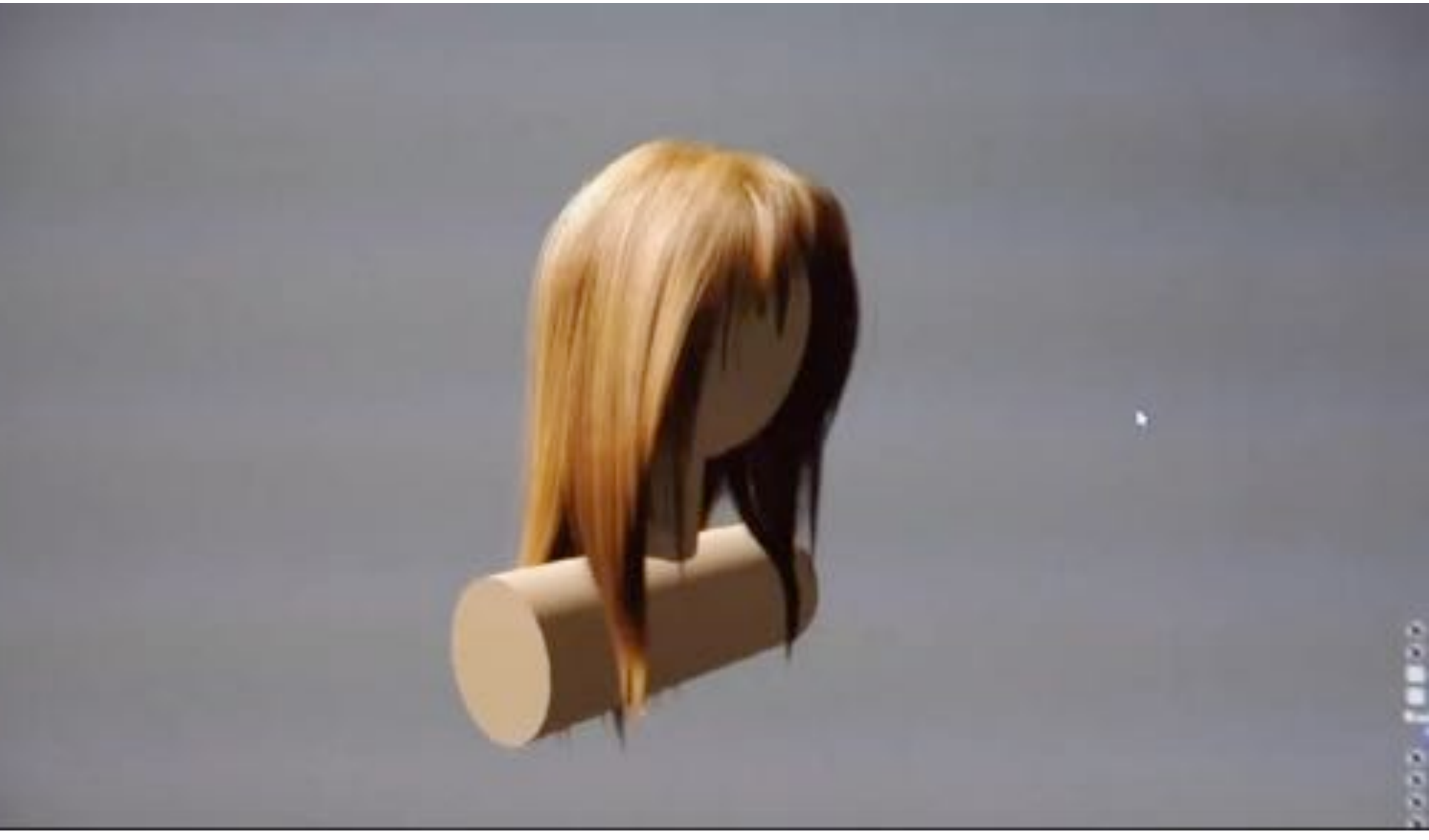
- Connect up many springs to describe interesting phenomena
- Extremely common in graphics/games
  - easy to understand
  - simple equation for each particle
- Often good reasons for using continuum model (PDE)

Example: Mass Spring System



Example: Mass Spring + Character

Example: Hair



Ok, I'm convinced.
So how do we solve these things numerically?

Numerical Integration

- Key idea: replace derivatives with differences
- In ODE, only need to worry about derivative in time
- Replace time-continuous function q(t) with samples q_k in

$$\frac{d}{dt}q(t) = f(q(t))$$

$$\Downarrow$$

$$\frac{q_{k+1} - q_k}{\tau} = f(q)$$

new configuration (unknown—want to solve for this!)

current configuration (known)

"time step," i.e., interval of time between q_k and q_{k+1}

Wait... where do we evaluate the velocity function? At the new or old configuration?

Forward Euler

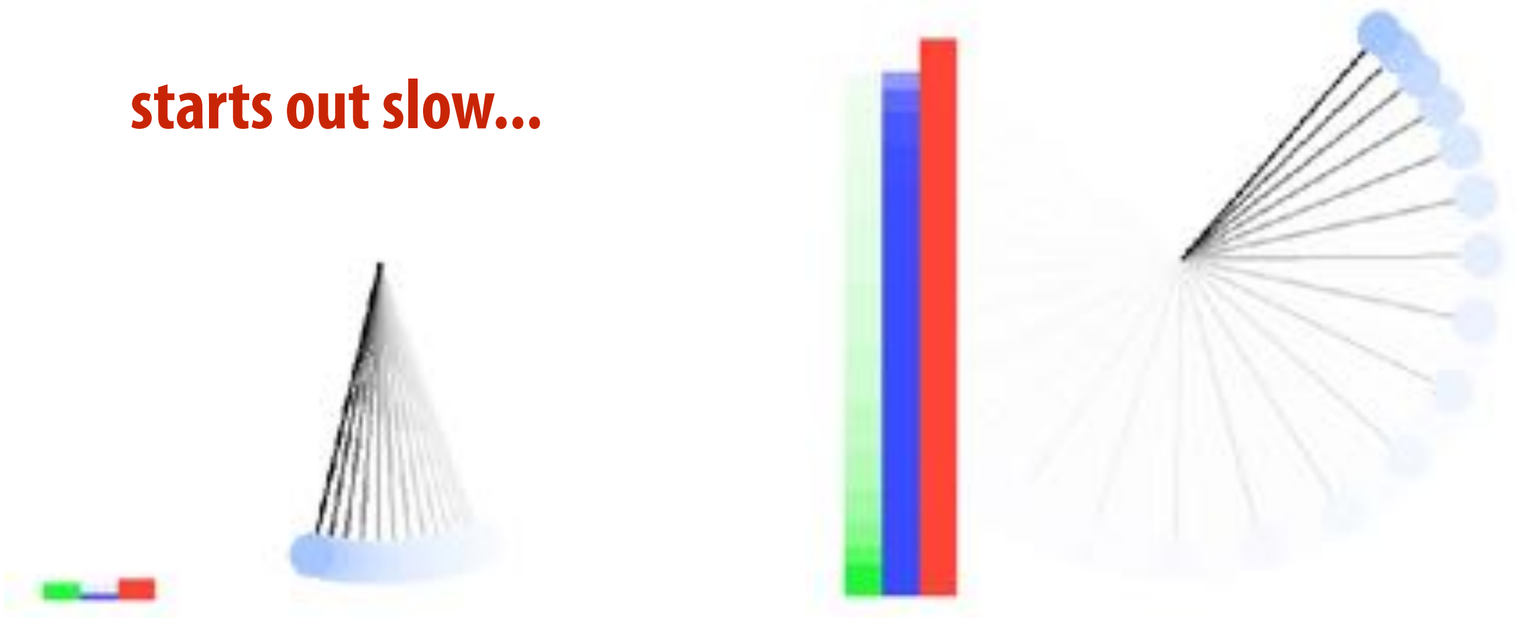
- Simplest scheme: evaluate velocity at current configuration
- New configuration can then be written explicitly in terms of known data:

new configuration — current configuration — velocity at current time

$$q_{k+1} = q_k + \tau f(q_k)$$

- Very intuitive: walk a tiny bit in the direction of the velocity
- Unfortunately, not very stable—consider pendulum:

starts out slow...



...gradually moves faster & faster!

Where did all this extra energy come from?

Forward Euler - Stability Analysis

- Let's consider behavior of forward Euler for simple linear ODE:

$$\dot{u} = -au, \quad a > 0$$

- Importantly: u should decay (exact solution is $u(t)=e^{-at}$)
- Forward Euler approximation is

$$\begin{aligned} u_{k+1} &= u_k - \tau a u_k \\ &= (1 - \tau a) u_k \end{aligned}$$

- Which means after n steps, we have

$$u_n = (1 - \tau a)^n u_0$$

- Decays only if $|1-\tau a| < 1$, or equivalently, if $\tau < 2/a$
- In practice: need very small time steps if a is large ("stiff system")

Backward Euler

- Let's try something else: evaluate velocity at next configuration
- New configuration is then implicit, and we must solve for it:

new configuration

current configuration

velocity at next time

$$q_{k+1} = q_k + \tau f(q_{k+1})$$

- Much harder to solve, since in general f can be very nonlinear!

- Pendulum is now stable... perhaps too stable?

starts out slow...

Where did all the energy go?

...and eventually stops moving completely.

Backward Euler - Stability Analysis

- Again consider a simple linear ODE:

$$\dot{u} = -au, \quad a > 0$$

- Remember: u should decay (exact solution is $u(t)=e^{-at}$)
- Backward Euler approximation is

$$(u_{k+1} - u_k)/\tau = -au_{k+1}$$

$$\iff \quad u_{k+1} = \frac{1}{1+\tau a} u_k$$

- Which means after n steps, we have

$$u_n = \left(\frac{1}{1+\tau a}\right)^n u_0$$

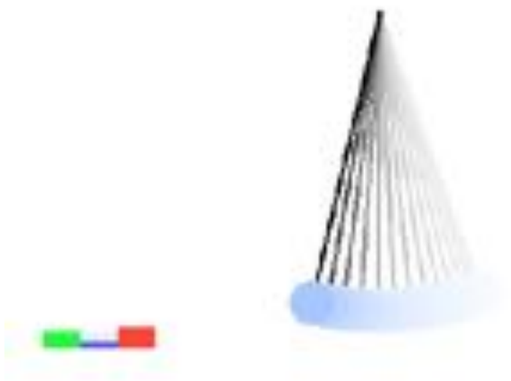
- Decays if $|1+\tau a| > 1$, which is always true!
- $\Rightarrow$Backward Euler is unconditionally stable for linear ODEs

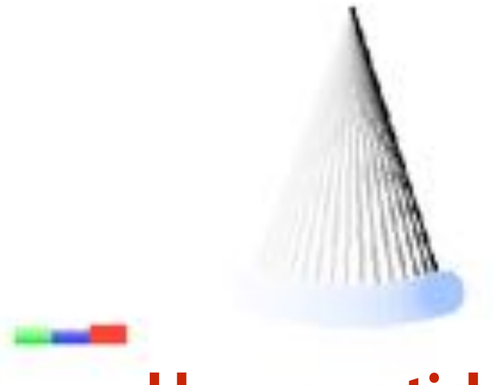
Symplectic Euler

- Backward Euler was stable, but we also saw (empirically) that it exibits numerical damping (damping not found in original eqn.)
- Nice alternative is symplectic Euler
  - update velocity using current configuration
  - update configuration using new velocity
- Easy to implement; used often in practice (or leapfrog, Verlet, ...)

- Pendulum now conserves energy almost exactly, forever:

starts out slow...



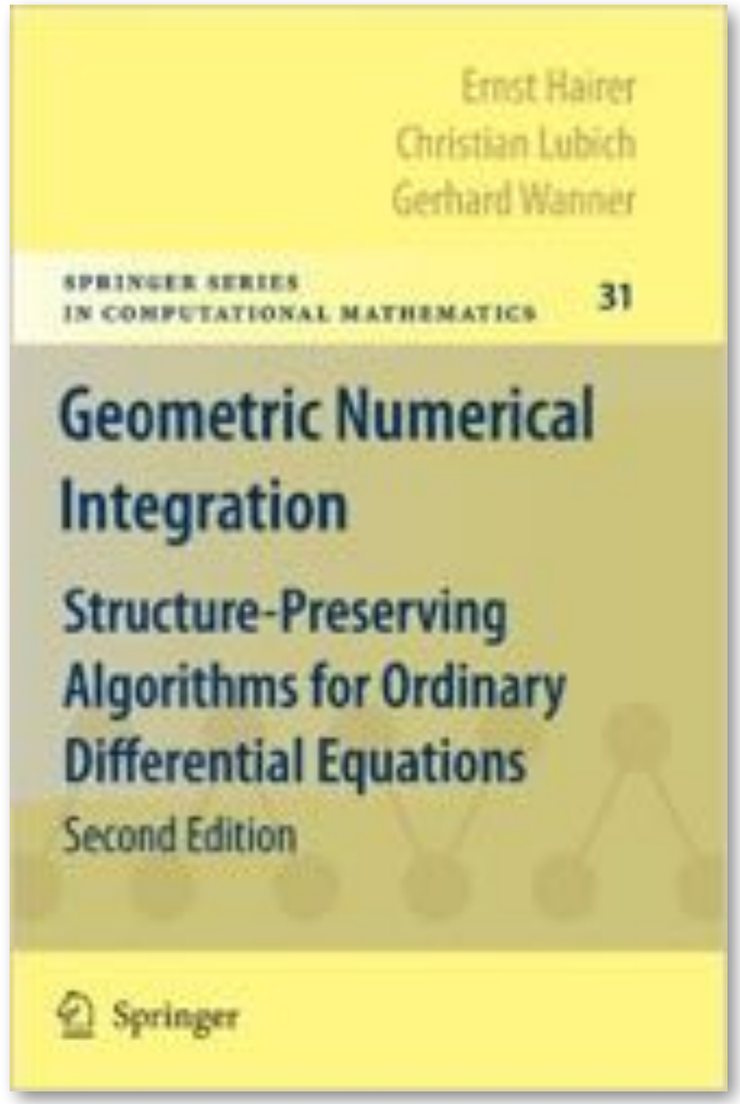


...and keeps on ticking.

(Proof? The analysis is not quite as easy...)

Numerical Integrators

- Barely scratched the surface
- Many different integrators
- Why? Because many notions of "good":
  - stability
  - accuracy
  - consistency/convergence
  - conservation, symmetry, ...
  - computational efficiency (!)
- No one "best" integrator—pick the right tool for the job!
- Could do (at least) an entire course on time integration...
- Great book: Hairer, Lubich, Wanner



Intoduction to Optimization (Part 1 of 2)

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, ...

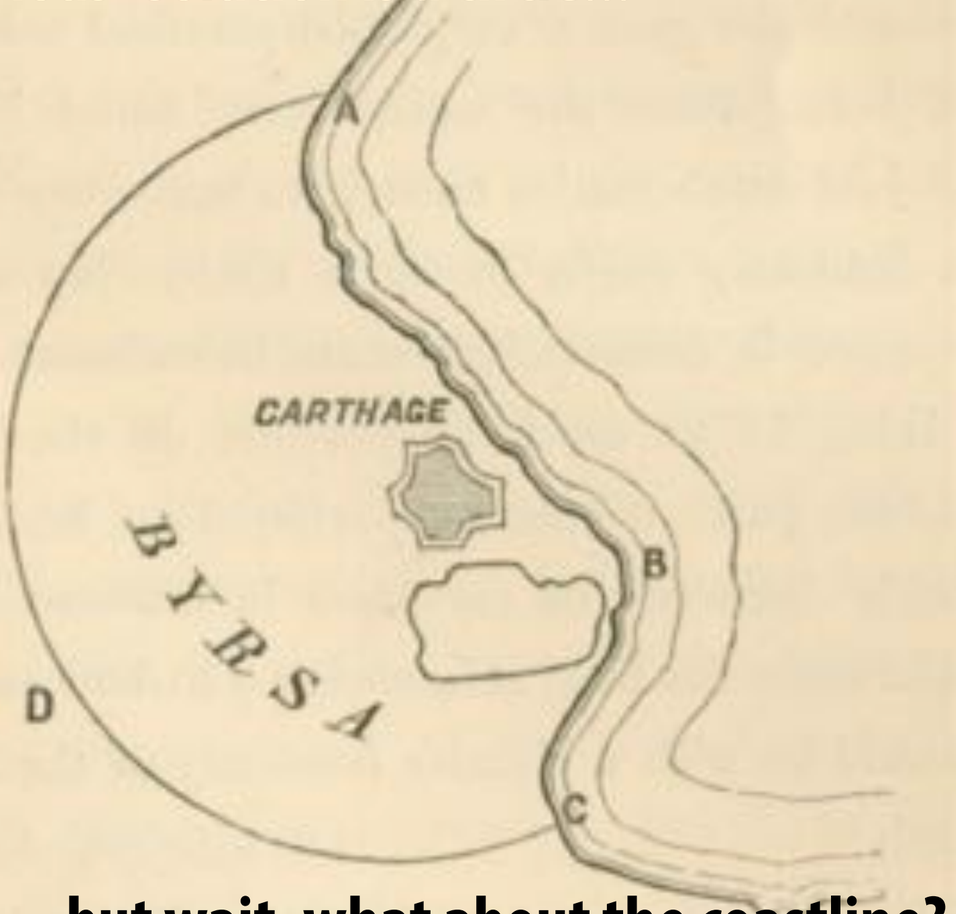
What is an optimization problem?

- 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."

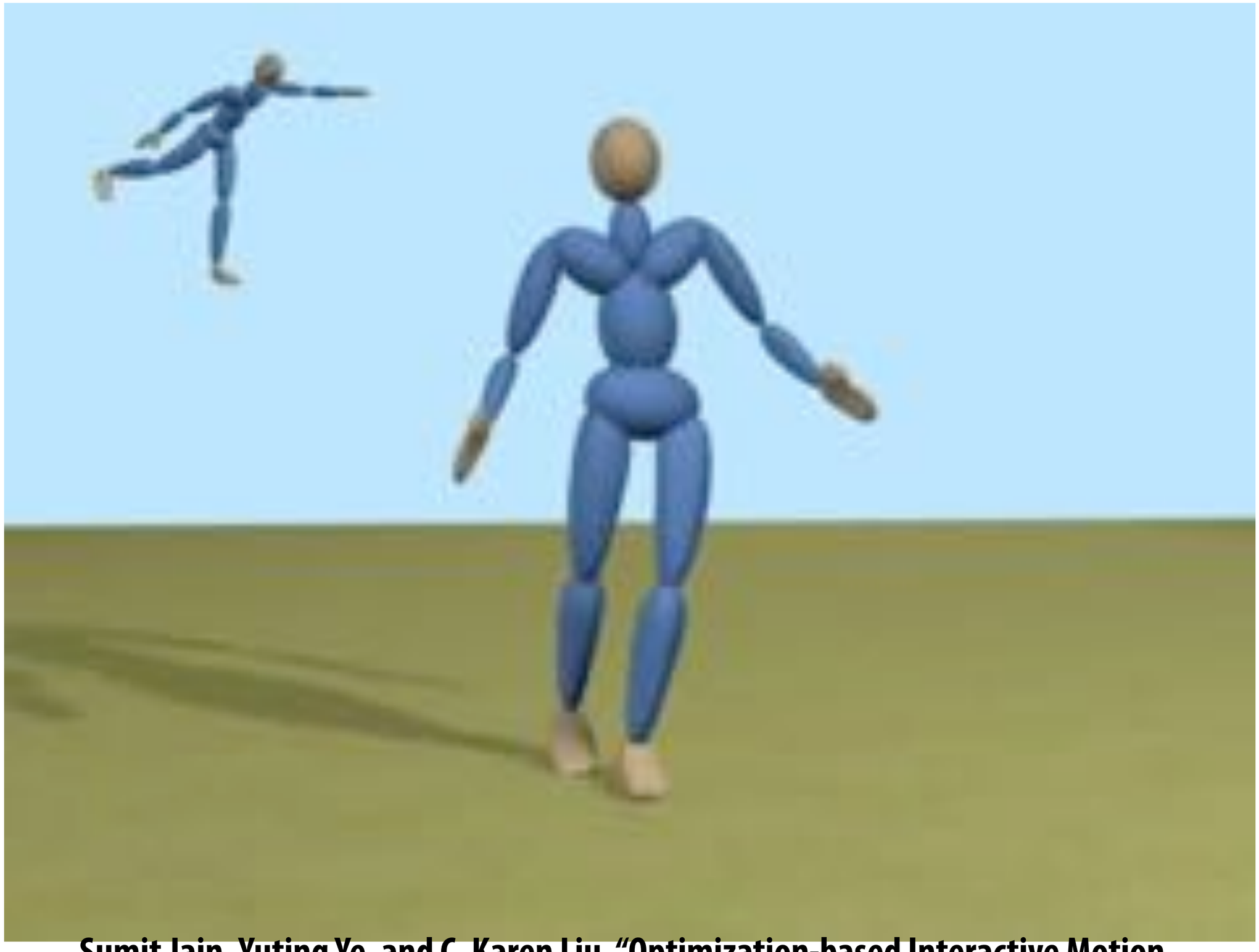
—Lord Kelvin, 1893

"Obvious" solution is a circle...



...but wait, what about the coastline?

Optimization in Graphics

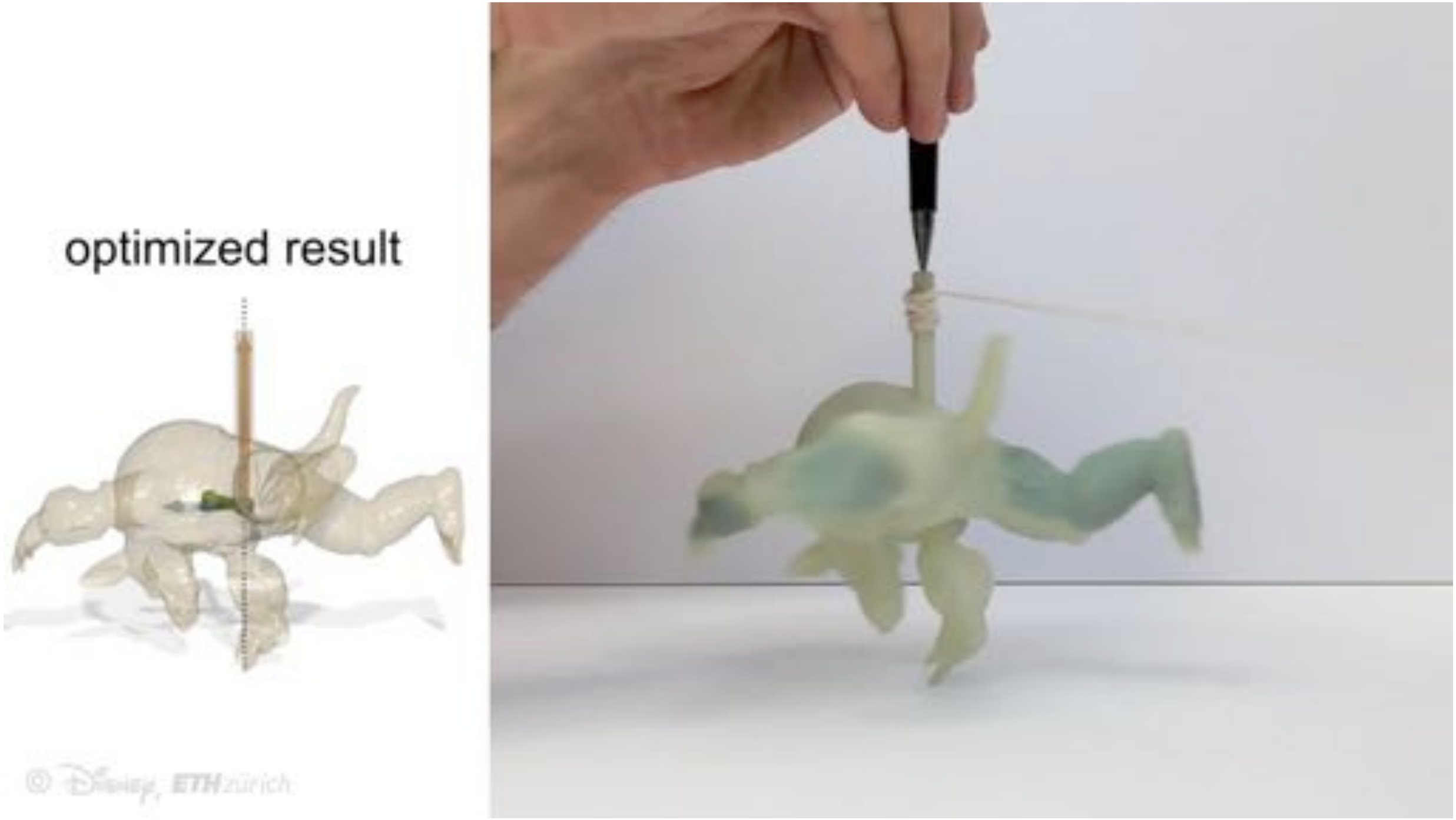


Sumit Jain, Yuting Ye, and C. Karen Liu, "Optimization-based Interactive Motion Synthesis"

Optimization in Graphics

Niloy J. Mitra, Leonidas Guibas, Mark Pauly, “Symmetrization”

Optimization in Graphics



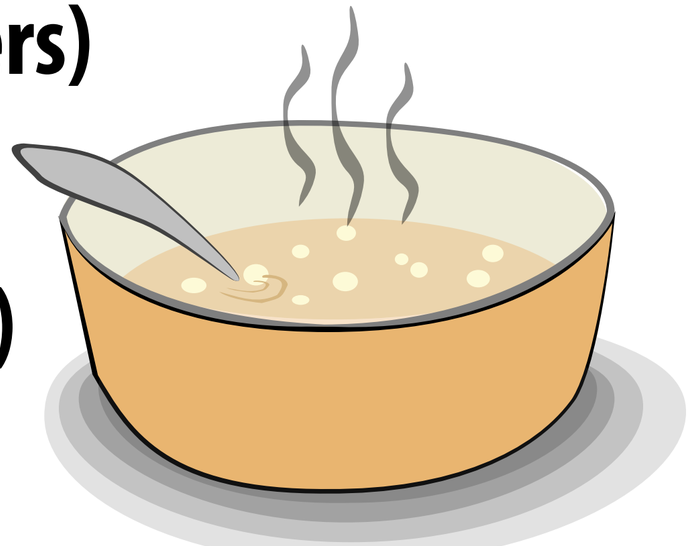
Moritz Bächer, Emily Whiting, Bernd Bickel, Olga Sorkine-Hornung,
“Spin-It: Optimizing Moment of Inertia for Spinnable Objects”

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?

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

$(f_i : \mathbb{R}^n \to \mathbb{R}, \;\; i = 0, \ldots, m)$

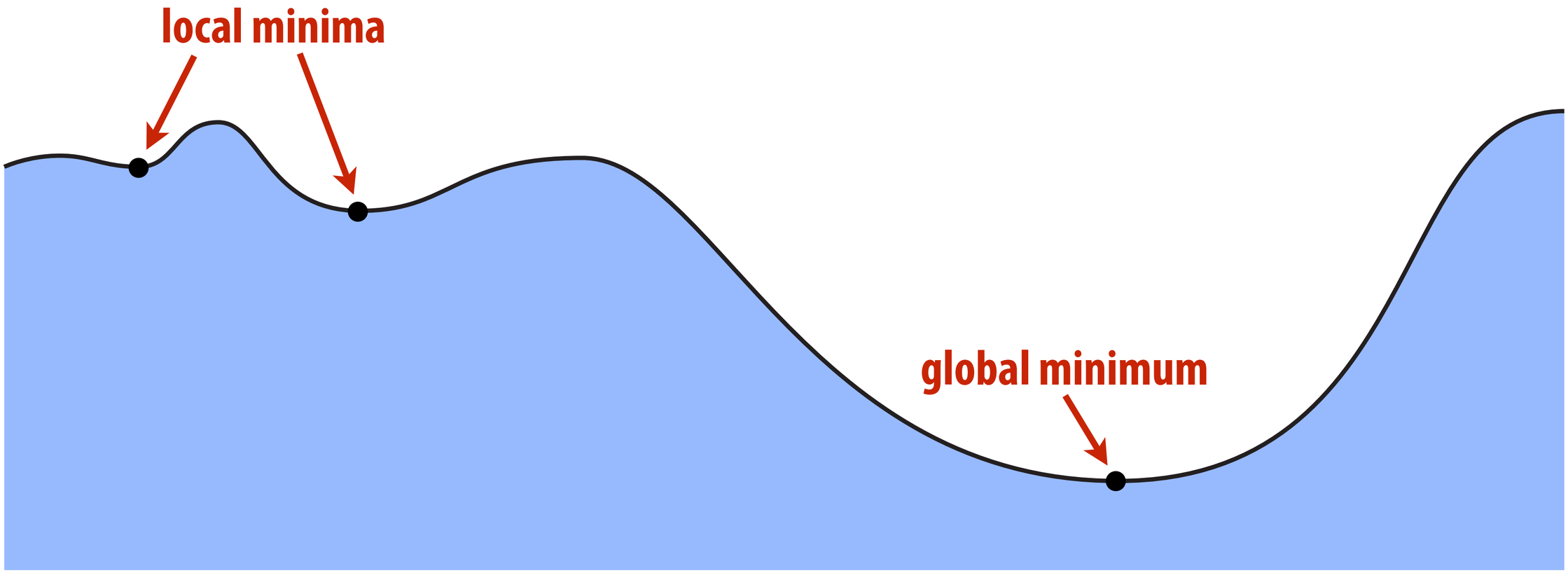
often (but not always) continuous, differentiable, ...

"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"



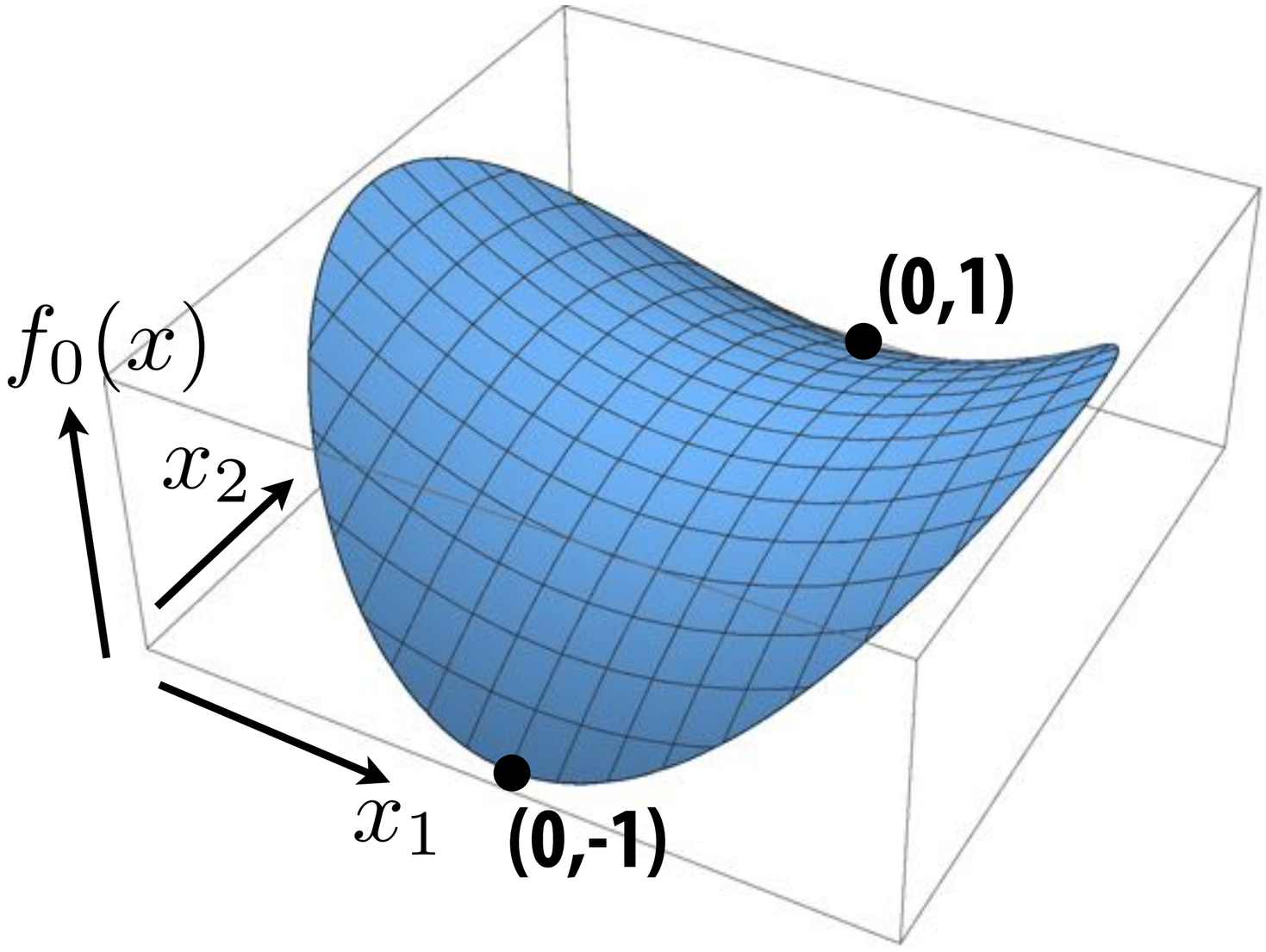
Philosophical question: does a local minimum "solve" the problem?

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

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

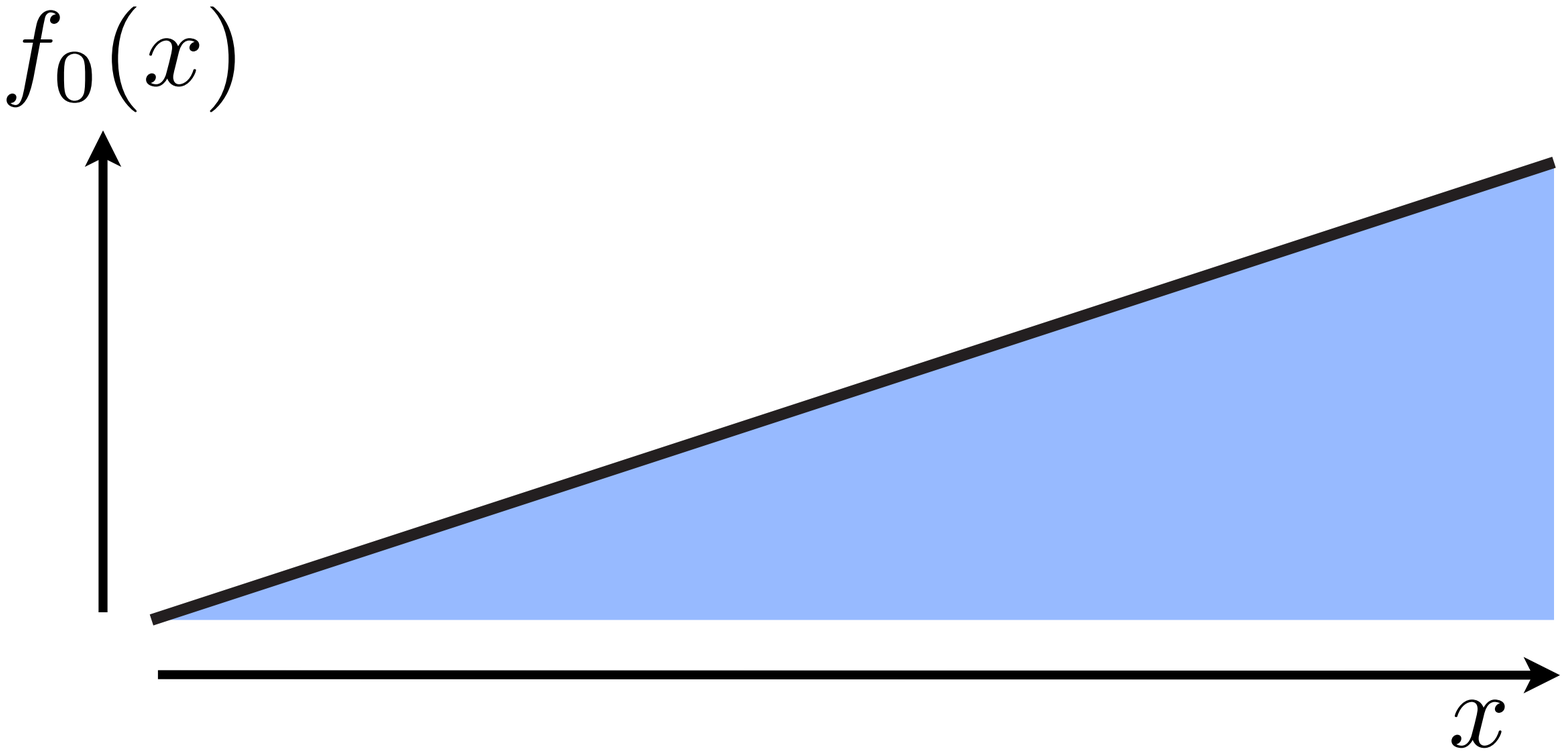


Q: Is this an optimization problem in standard form? **A: Yes.**

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?



perfectly reasonable optimization problem

$$\min_{x \in \mathbb{R}} x$$

clearly has no solution (can always pick smaller x)

- 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

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

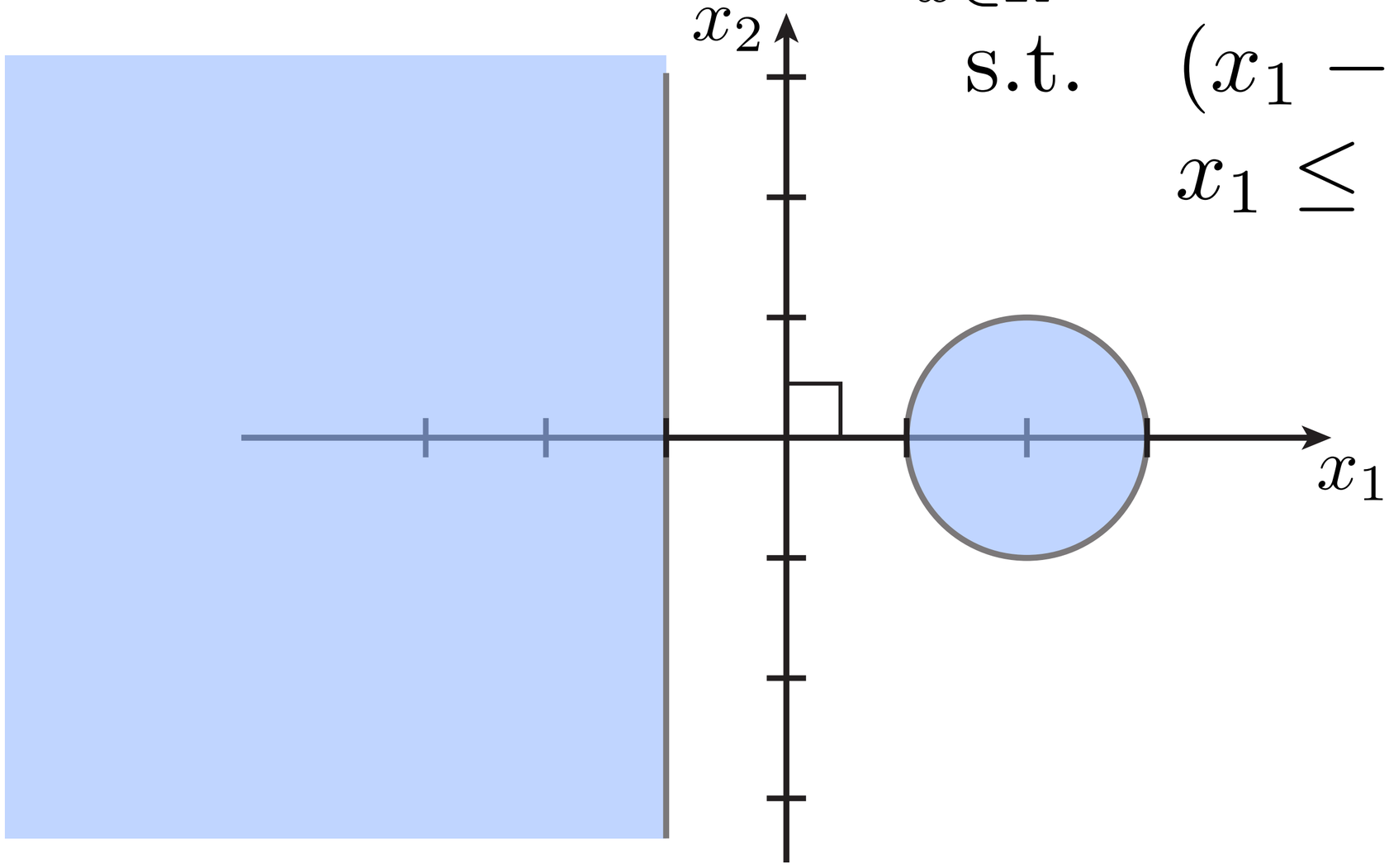
problem now is just finding a feasible solution—
which can be really hard (or impossible!)

- 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?

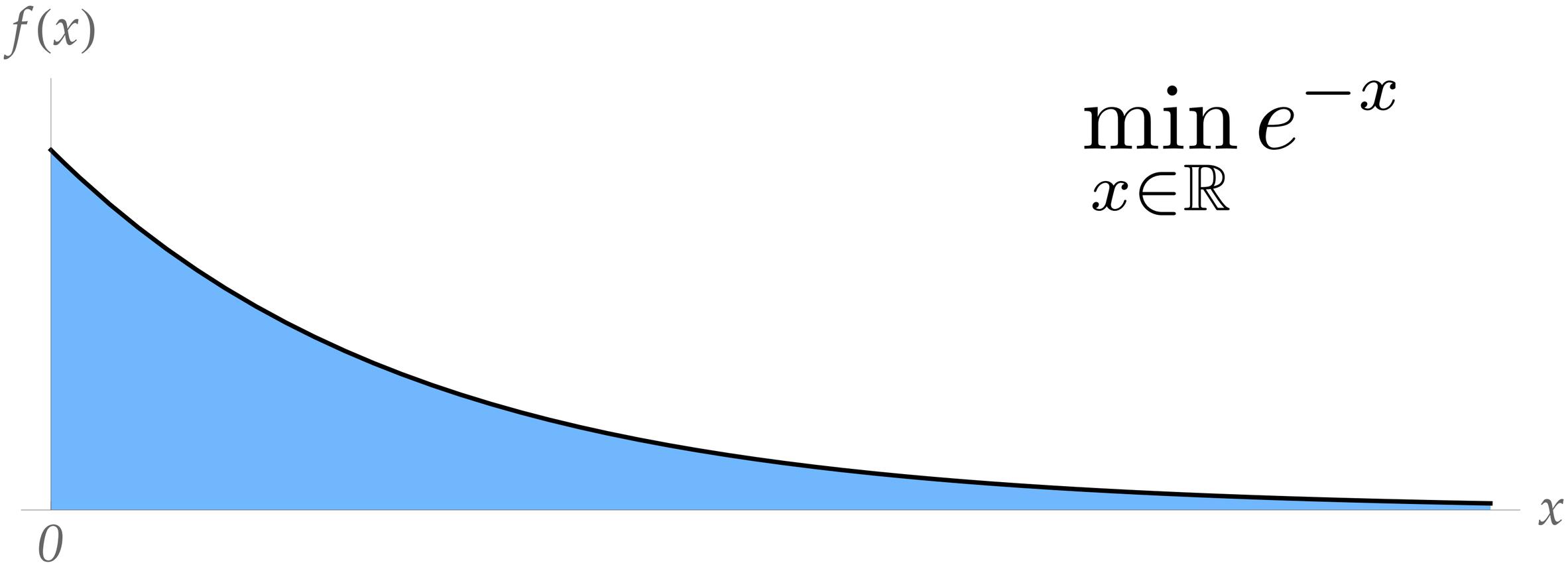
$$\begin{aligned} \min_{x \in \mathbb{R}^2} \quad & \sin(x_1) + x_2^2 \\ \text{s.t.} \quad & (x_1 - 2)^2 + x_2^2 \leq 1, \\ & x_1 \leq -1 \end{aligned}$$



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

Existence & Uniqueness of Minimizers, cont.

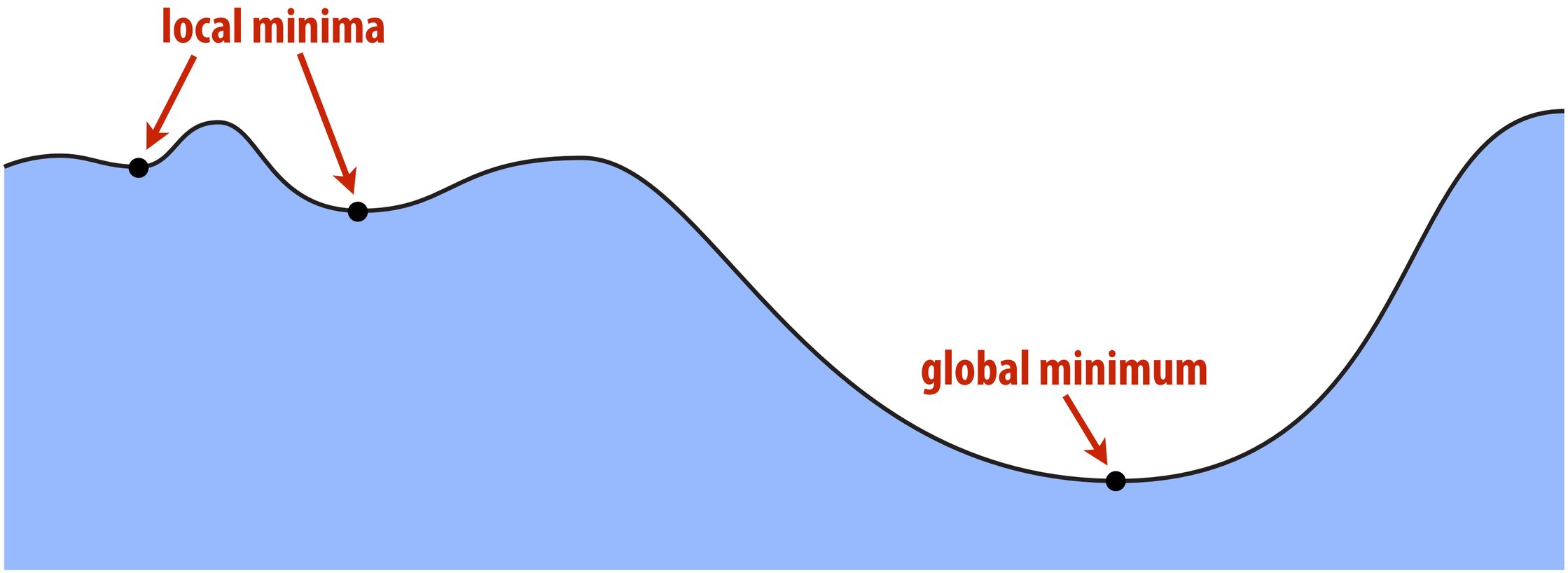
- Even being bounded from below is not enough:



- 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!)

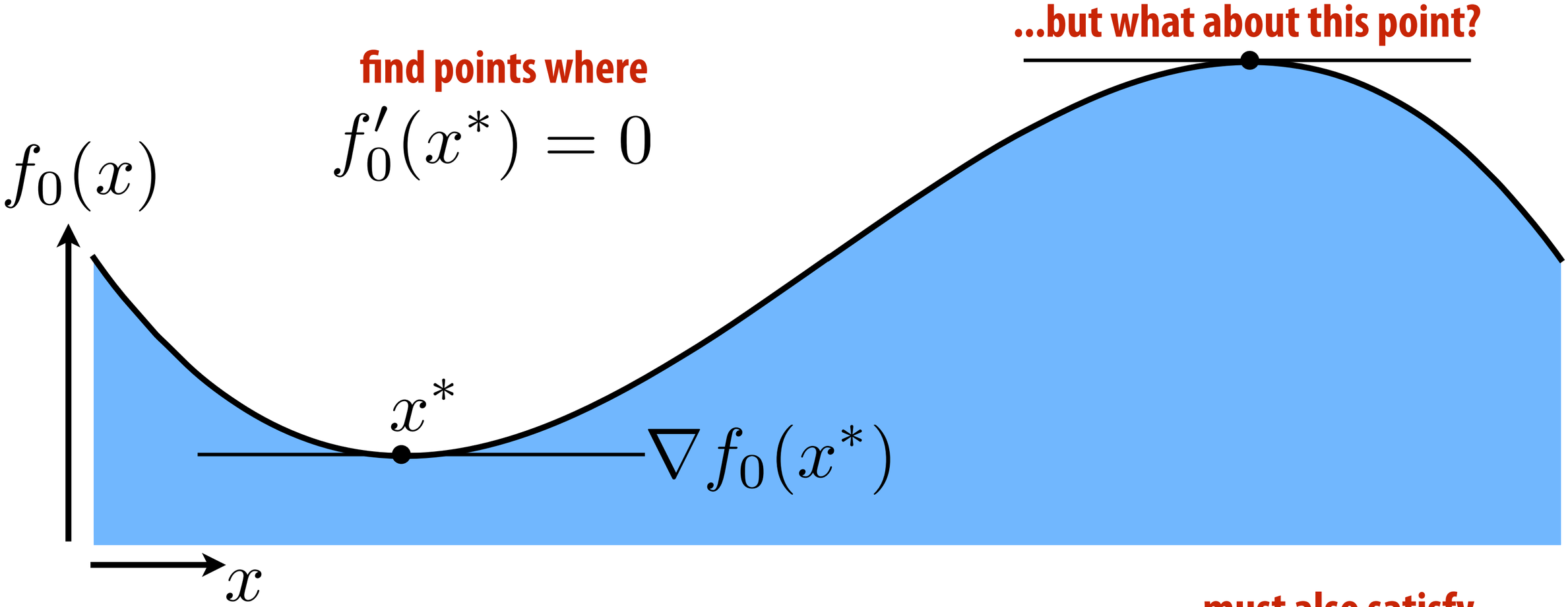
Characterization of Local Minima

- Consider an objective f_0: R → R. How do you find a minimum?
- (Hint: you may have memorized this formula in high school!)

find points where

$$f_0'(x^*) = 0$$

...but what about this point?



- Also need to check second derivative (how?)
- Make sure it's positive
- Ok, but what does this all mean for more general functions f_0?

must also satisfy

$$f_0''(x^*) \geq 0$$

Optimality Conditions (Unconstrained)

- In general, our objective is f0: R → R^n (goes to R^n, not just R)
- How do we test for a local minimum?
- 1st derivative becomes gradient; 2nd derivative becomes Hessian

$$\nabla f := \begin{bmatrix} \partial f/\partial x_1 \\ \vdots \\ \partial f/\partial x_n \end{bmatrix}$$

GRADIENT
(measures "slope")

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

HESSIAN
(measures "curvature")

- Optimality conditions?

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

1st order

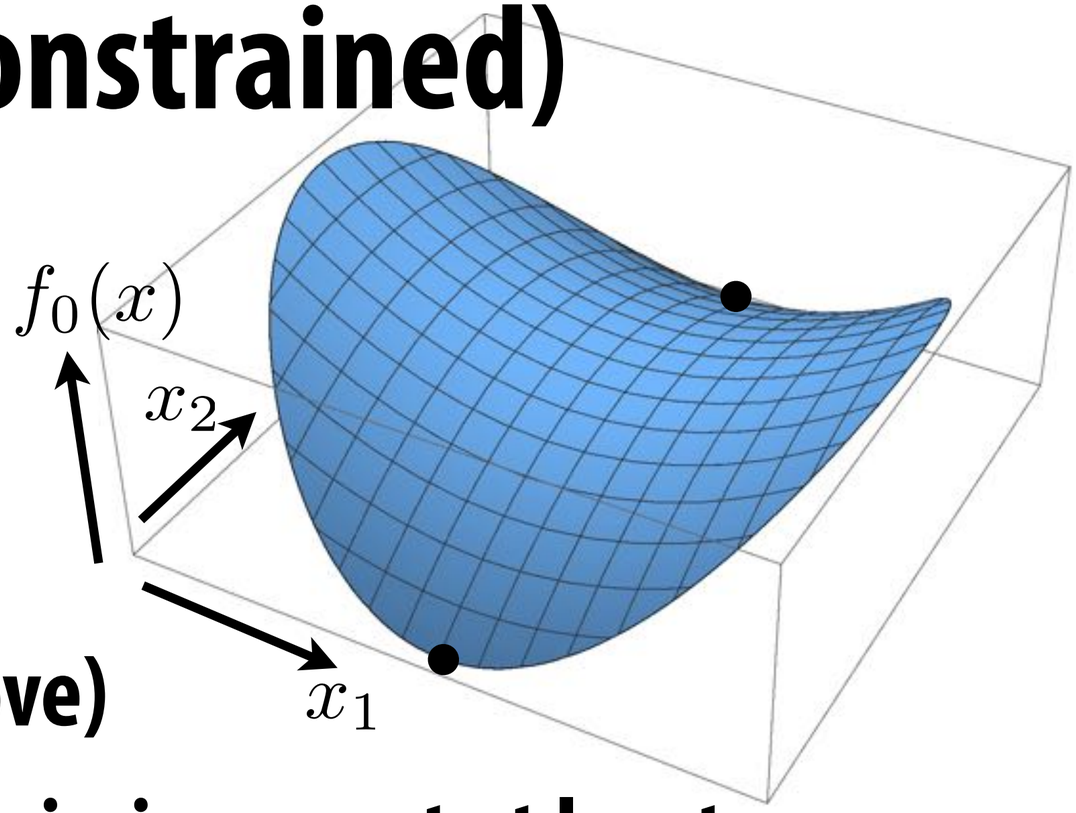
positive semidefinite (PSD)
($u^T A u \geq 0$ for all u)

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

2nd order

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:

$$\exists \lambda_i \text{ s.t.} \quad \nabla f_0(x^*) = -\sum_{i=1}^{n} \lambda_i \nabla f_i(x^*) \qquad \text{stationarity}$$

$$f_i(x^*) \leq 0, \;\; i = 1, \ldots, n \qquad \text{primal feasibility}$$

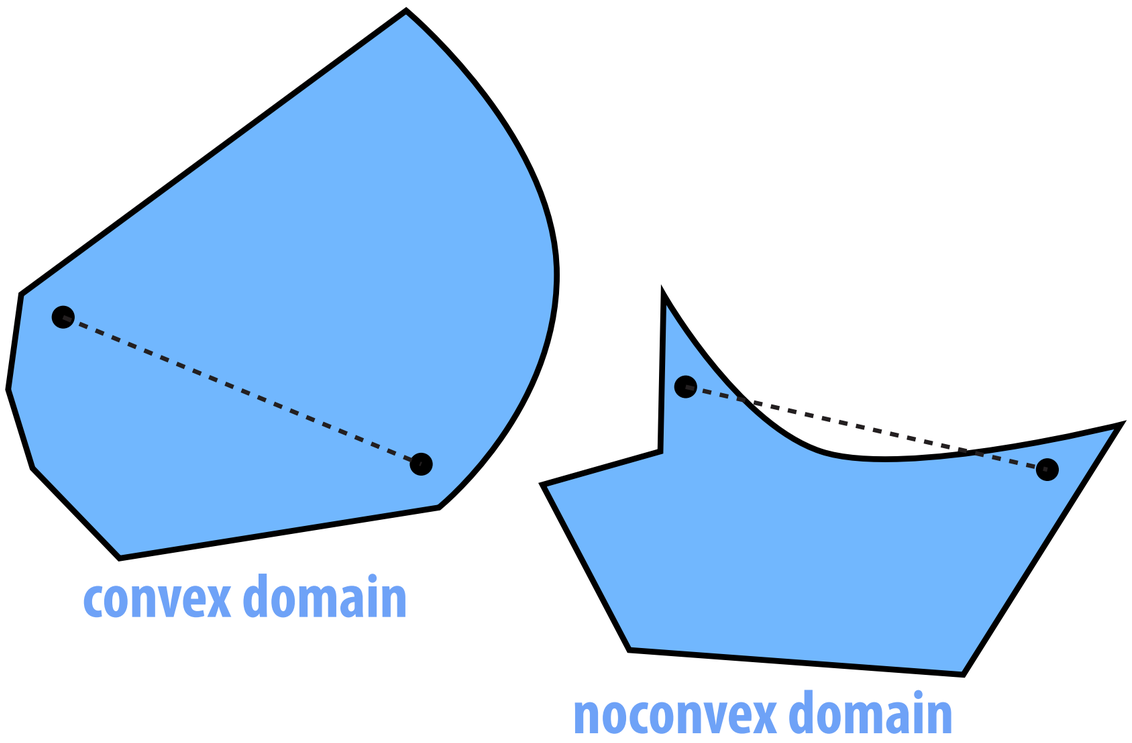
$$\lambda_i \geq 0, i = 1, \ldots, n \qquad \text{dual feasibility}$$

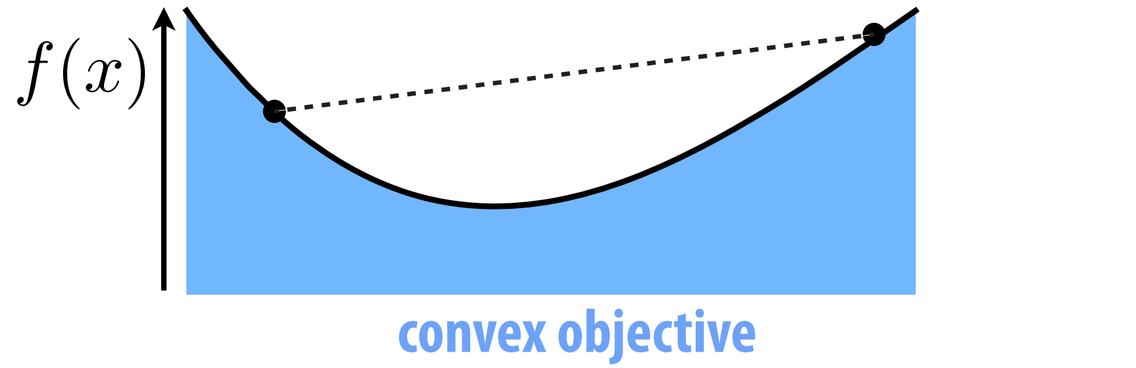
$$\lambda_i f_i(x^*) = 0, \;\; i = 1, \ldots, n \qquad \text{complementary slackness}$$

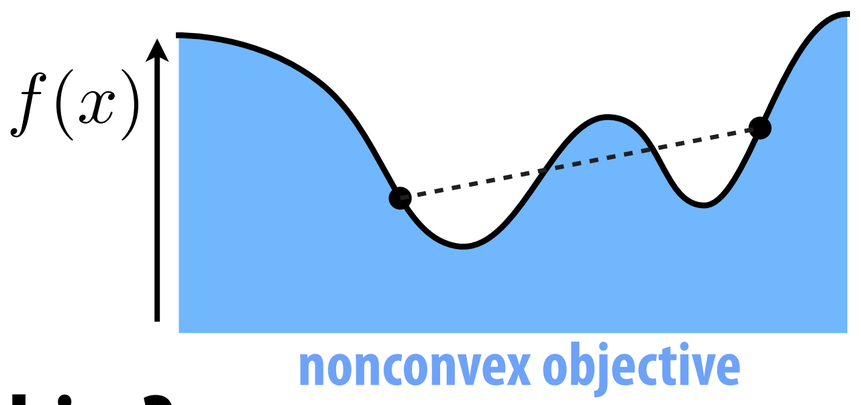
- ...we won't work with these in this class! (But good to know where to look.)

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