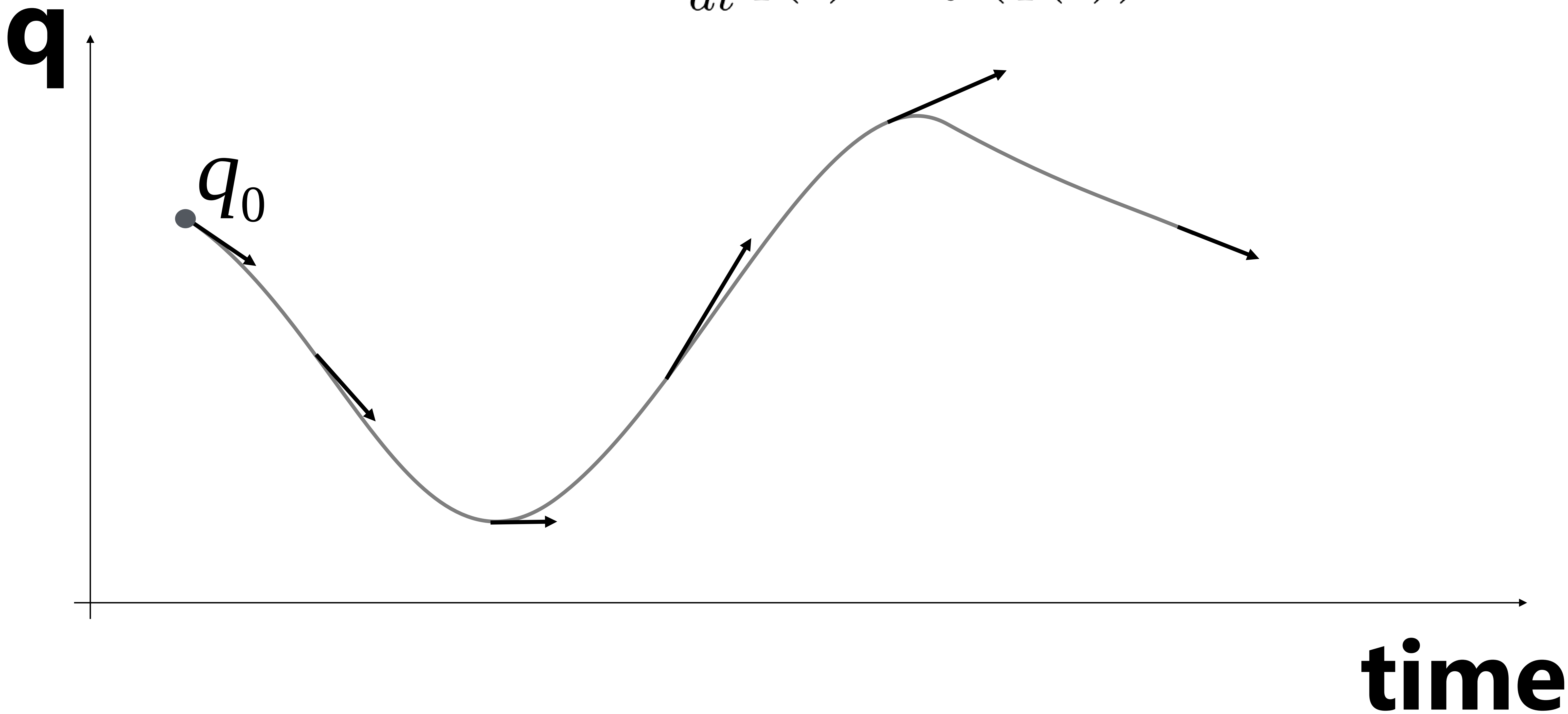
An introduction to Physically-based animation: ODEs and PDEs

ODEs

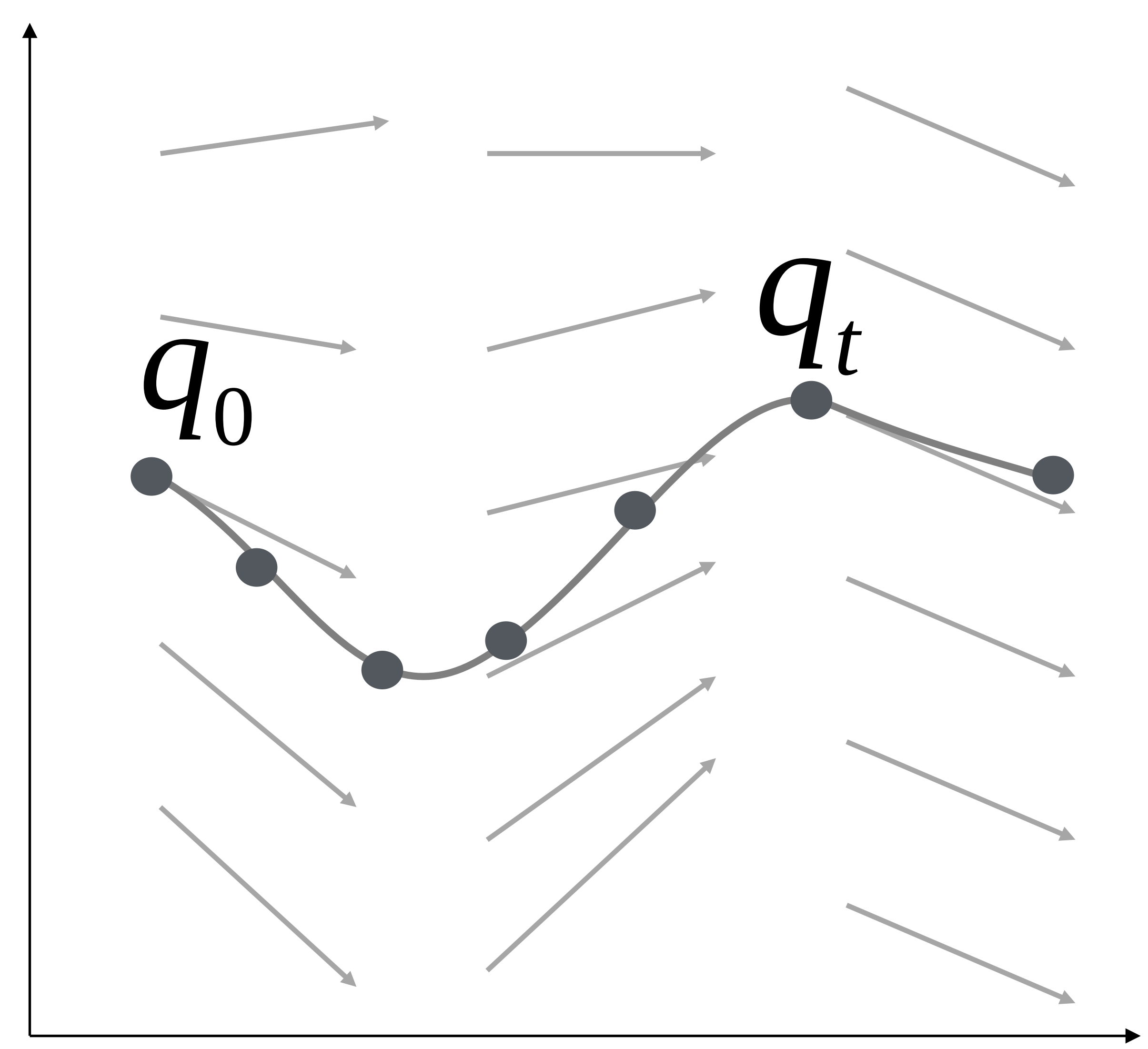
- ODEs implicitly define an unknown function through its derivative
- Only a single parameter, e.g., time

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



Solving ODEs Numerically

- Given initial conditions $q(0), \dot{q}(0)$, find function $q(t)$
- Replace time-continuous function $q(t)$ with discrete samples q_i at time t_i



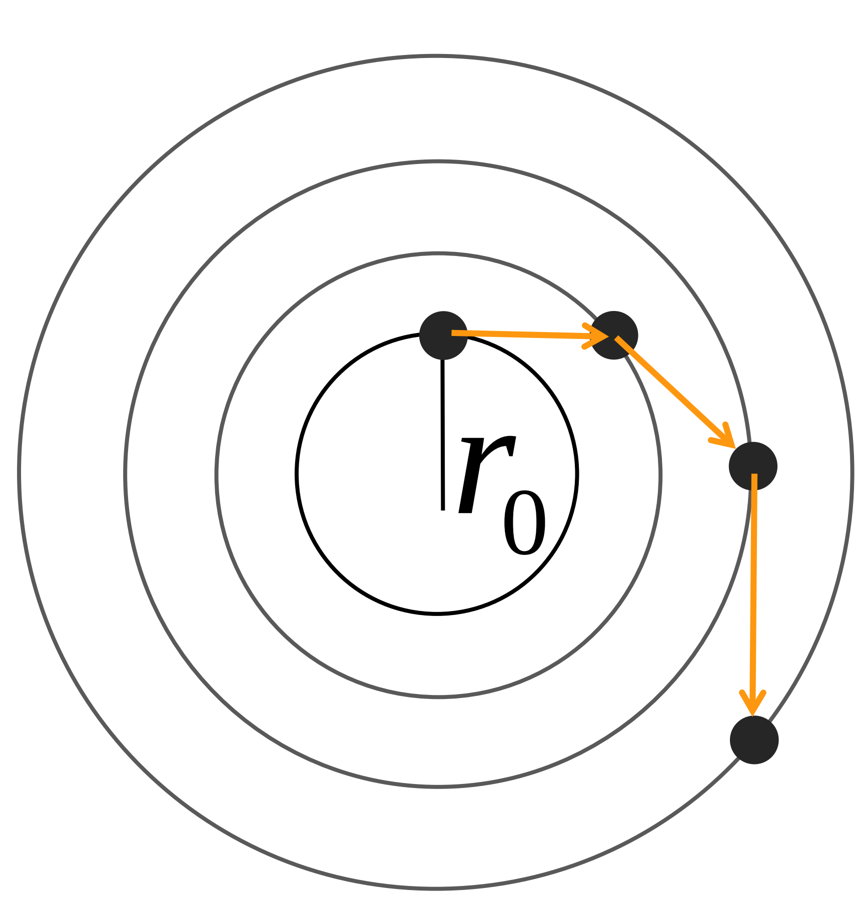
Solving ODEs Numerically

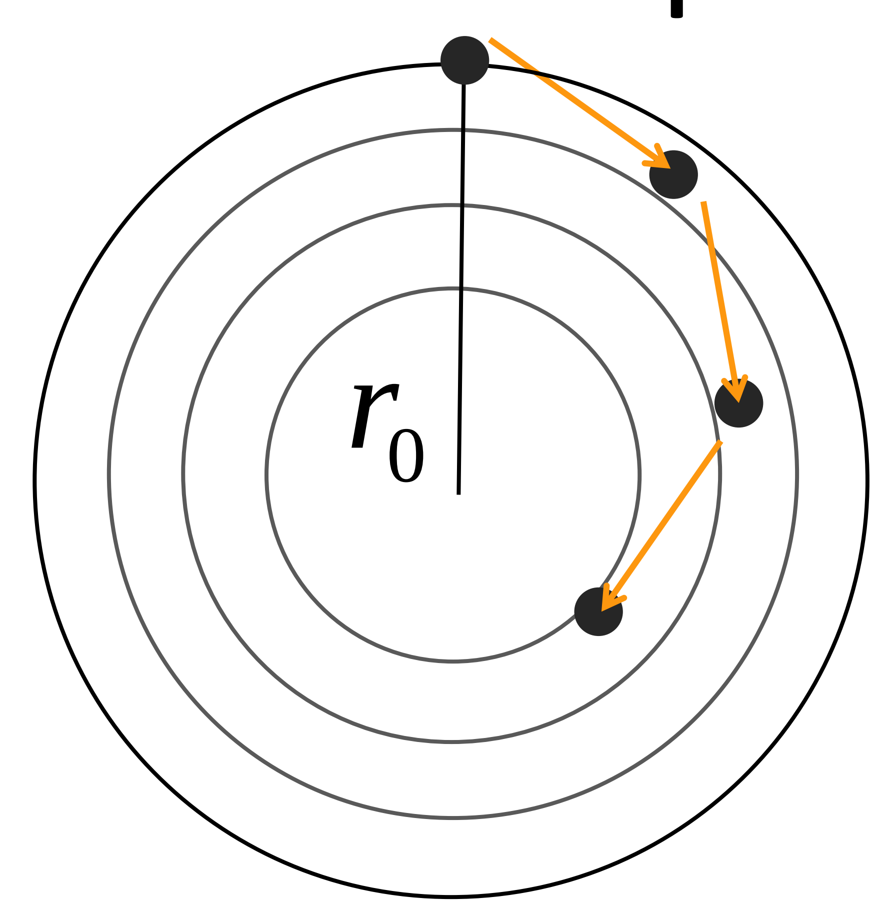
- Numerical time integration approximates

$$q(t+h) = q(t) + \int_t^{t+h} \dot{q}(t)\,dt$$

- Different 'quadrature rules' lead to different behaviors
  - Explicit Euler is simple but only conditionally stable
  - Implicit Euler is unconditionally stable but expensive

$$\dot{q} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} q$$





Partial Differential Equations (PDEs)

Differential Equations

An Ordinary Differential Equation (ODE) describes an unknown function through its derivatives with respect to a single variable.

Example ODE: $$m\frac{d^2x(t)}{dt^2} = F(x(t))$$

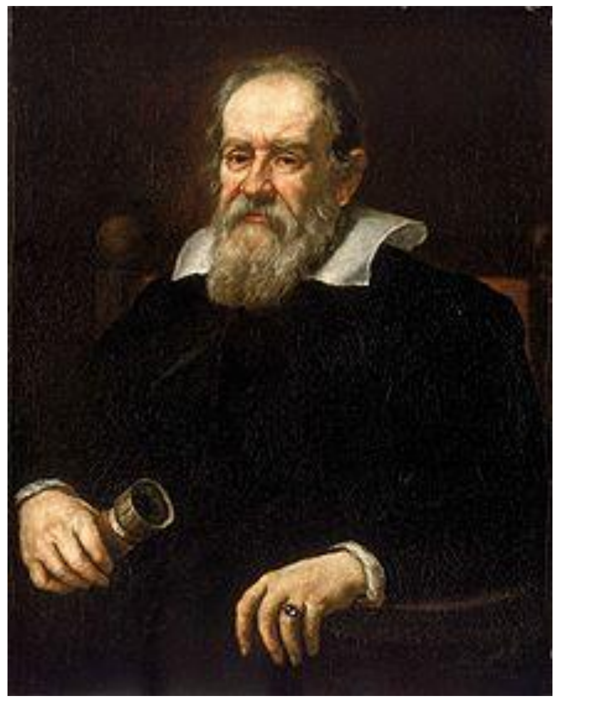
A Partial Differential Equation (PDE) describes an unknown function through its partial derivatives with respect to **multiple** variables.

Example PDE: $$\frac{\partial u(t,x)}{\partial t^2} = c^2 \frac{\partial u(t,x)}{\partial x^2}$$

Why PDEs?

The book of nature is written in the language of mathematics.

Galileo Galilei



- Many physical phenomena and processes can be described by partial differential equations (PDEs)
- PDEs arise naturally from
  - conservation laws (mass, momentum, energy)
  - equilibrium conditions, …

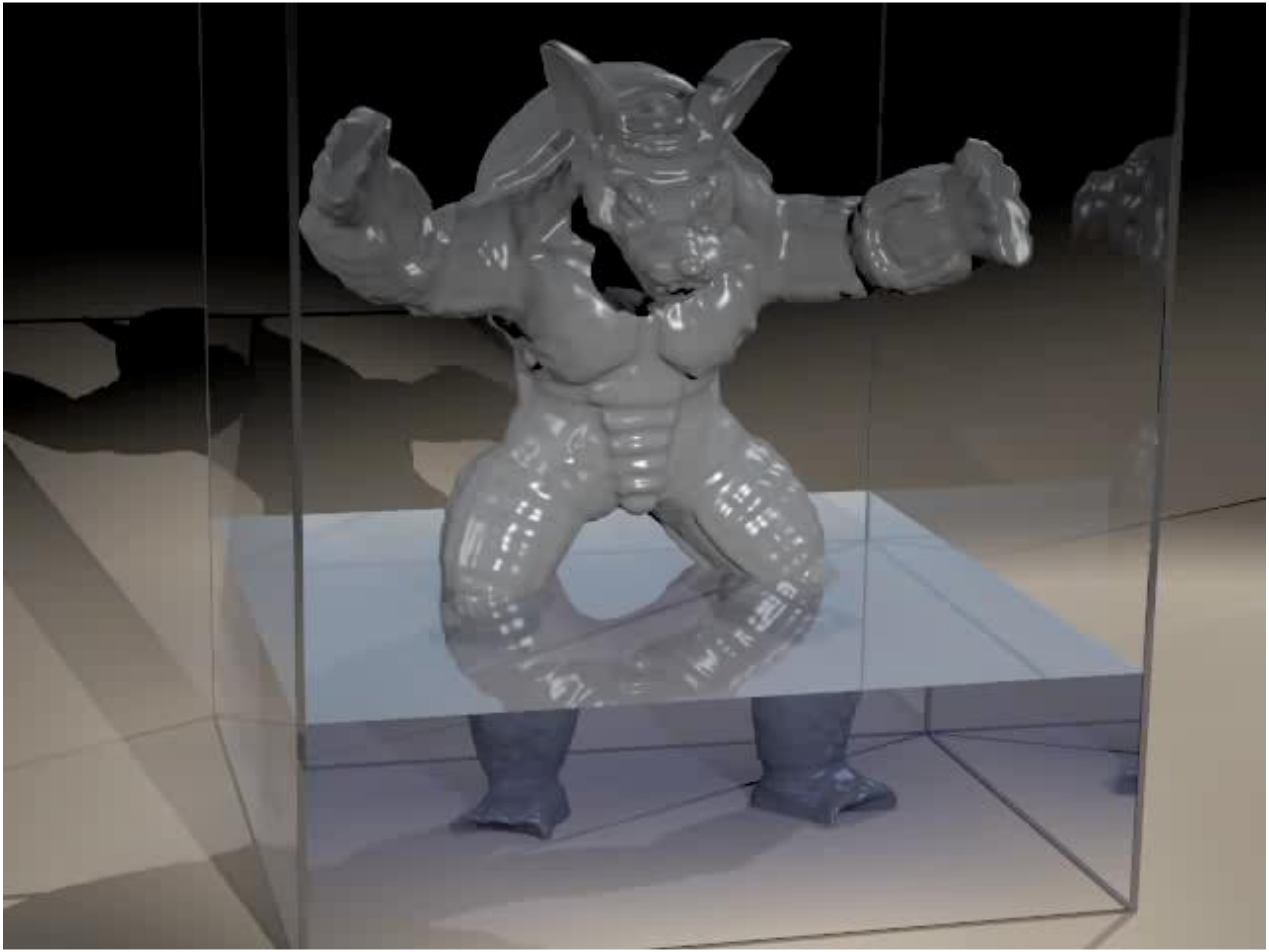
Fluid Simulation in Graphics

Incompressible Navier Stokes Equations

$$\nabla \cdot \boldsymbol{u} = \mathbf{0}$$

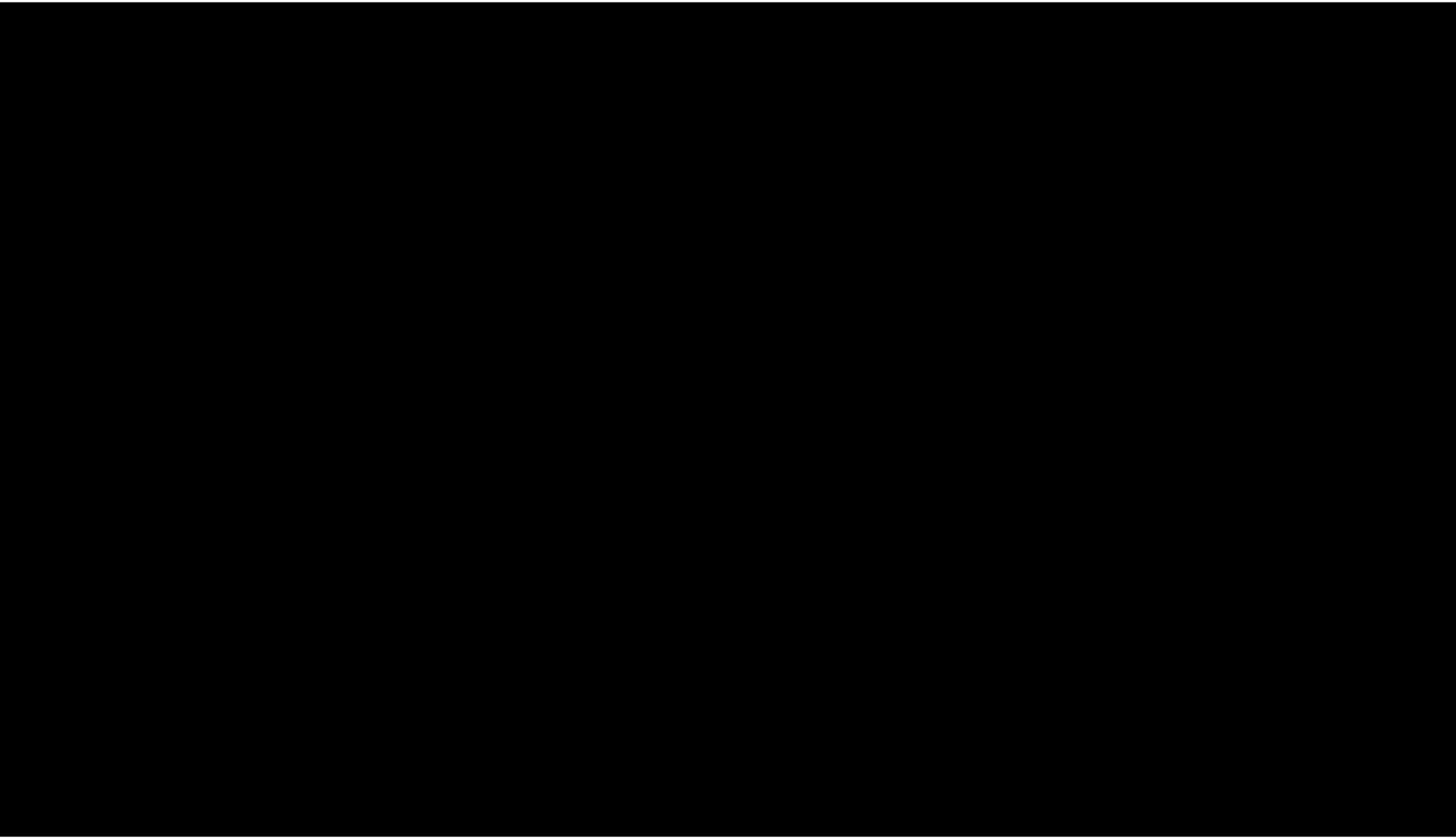
$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} - \nu \nabla^2 \mathbf{u} = -\nabla w + \mathbf{g}.$$

Fluid Simulation in Graphics



[Losasso, F., Shinar, T. Selle, A. and Fedkiw, R., "*Multiple Interacting Liquids*"](#)

Fluid Simulation in Graphics



https://jzehnder.me/publications/advectionReflection/

Elasticity in Graphics in Graphics

Governing Equations of Continuum Mechanics

$$\boxed{\nabla \cdot \boldsymbol{\sigma} + \boldsymbol{f} = \boldsymbol{m} \cdot \boldsymbol{a}}$$

Elasticity in Graphics

[Irving, G., Schroeder, C. and Fedkiw, R., "Volume Conserving Finite Element Simulation of Deformable Models"]

Cloth Simulation in Graphics

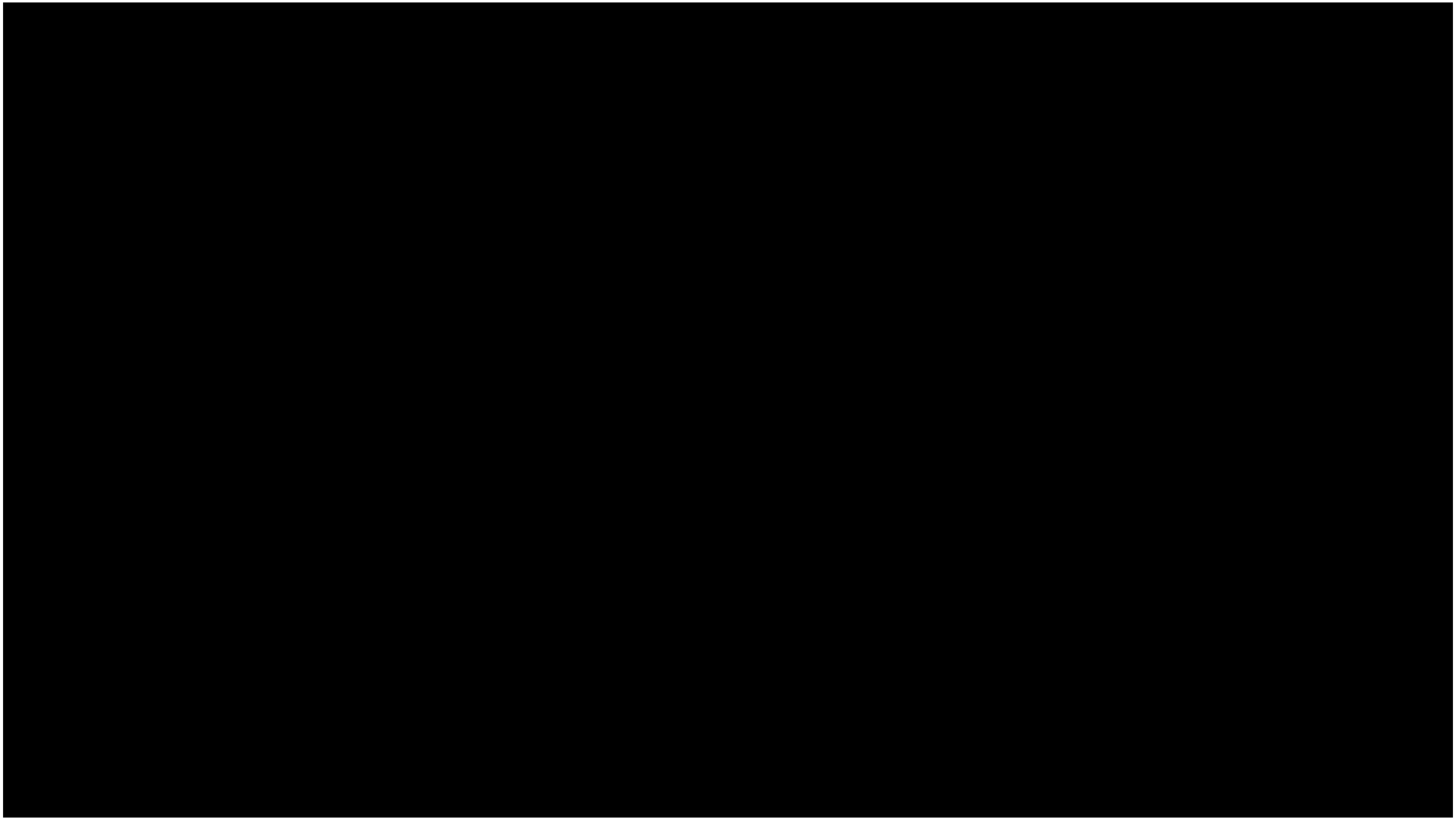
Zhili Chen, Renguo Feng and Huamin Wang, *"Modeling friction and air effects between cloth and deformable bodies"*

Magnetism in Graphics

Maxwell Equations (static case)

$$\boxed{\begin{gathered}\nabla \cdot \mathbf{B} = 0, \qquad \nabla \times \mathbf{H} = \mathbf{J} \\ \mathbf{H} = \frac{1}{\mu_0}\mathbf{B} - \mathbf{M}\,,\end{gathered}}$$

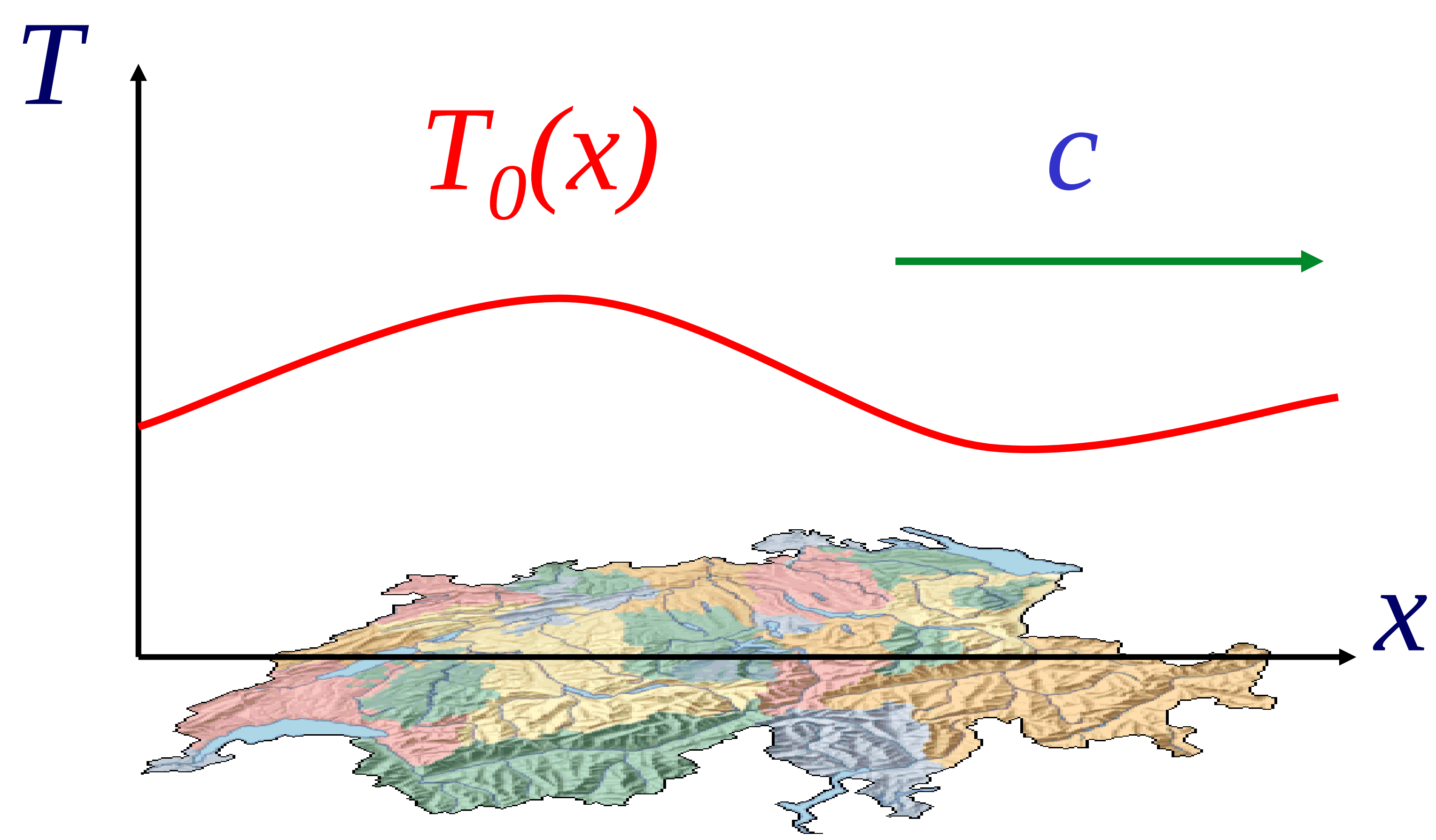
Magnetism in Graphics



https://www.youtube.com/watch?v=NTlbGGcsYek

A Simple Example: 1D Advection

Weather forecast: simulate temperature evolution.



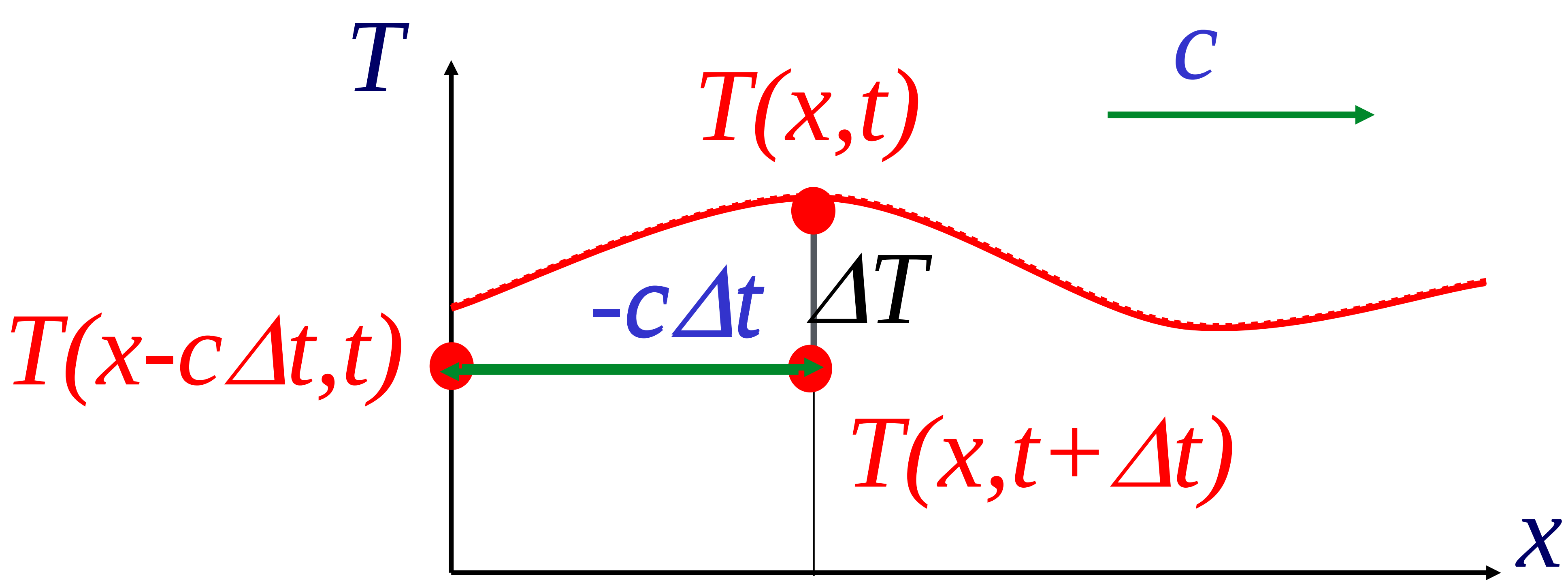
One space dimension + time

Given: initial temperature distribution $T_0(x) = T(x, 0)$
and wind speed c.

Find: temperature distribution $T(x, t)$ for any t.

A Simple Example: 1D Advection

- How does the temperature change over a time interval Δt?



$$\boxed{T(x,t+\Delta t)} = \boxed{T(x-c\Delta t,t)}$$

$$\boxed{\Delta T} = T(x,t+\Delta t) - T(x,t)$$

$$T(x-c\Delta t,t) = \boxed{T(x,t)} - \frac{\partial T}{\partial x} c\Delta t + O(\Delta t^2) = \boxed{T(x,t+\Delta t)}$$

1D advection equation

$$\frac{\Delta T}{\Delta t} \approx -c\frac{\partial T}{\partial x} \quad \Delta t \to 0 \quad \frac{\partial T}{\partial t} = -c\frac{\partial T}{\partial x}$$

Analytical Solution

- Any $T(x,t)$ of the form $T(x,t) = f(x-ct)$ solves $\dfrac{\partial T}{\partial t} = -c\dfrac{\partial T}{\partial x}$

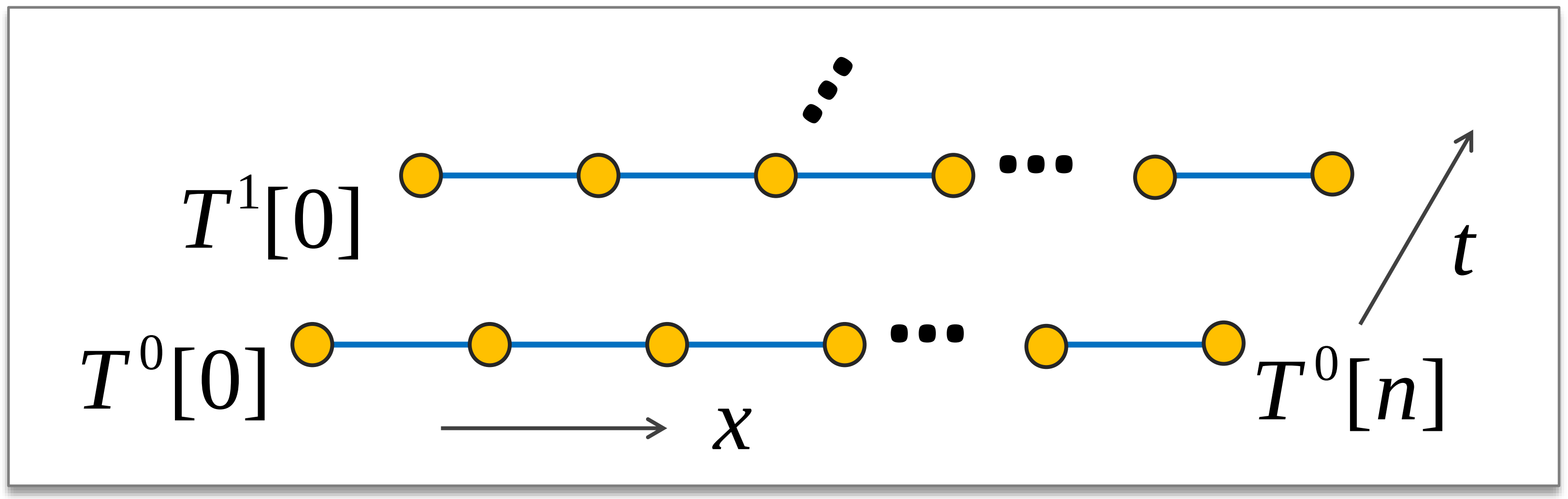
- The solution also needs to satisfy the initial condition $T(x,0) = T_0(x)$

- The solution is thus $T(x,t) = T_0(x-ct)$

Note: *only simple PDEs can be solved analytically!*

Numerical Solution

- Sample temperature $T(x,t)$ on 1D grids $T^t[i] = T(i \cdot h, t \cdot \Delta t)$ with $i \in (1,..,n), t \in (0,1,2..)$



- Discretize derivatives with **finite differences** (*space & time*)

$$\frac{\partial T}{\partial t} = -c\frac{\partial T}{\partial x} \quad \Rightarrow \quad \frac{T^{t+1}[i] - T^t[i]}{\Delta t} = -c\frac{T^t[i] - T^t[i-1]}{h}$$

- Solving for $T^{t+1}[i]$ yields update rule

$$T^{t+1}[i] = T^t[i] - \Delta t \cdot c\frac{T^t[i] - T^t[i-1]}{h}$$

- Provide initial values $T^0[i]$
- Set boundary conditions, e.g. *periodic* $\quad T^t[0] = T^t[n]$

Some Notation

- Abbreviation $$u_{tt} = \frac{\partial^2}{\partial t^2} u(t,..), \quad u_{xy} = \frac{\partial^2}{\partial x \, \partial y} u(x, y,..)$$
- Spatial variables $$\mathbf{x} = \left(x_1, \ldots, x_d\right)^t$$
- Nabla operator $$\nabla = \left(\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_d}\right)^t \quad \nabla s = \left(\frac{\partial s}{\partial x_1}, \ldots, \frac{\partial s}{\partial x_d}\right)^t$$
- Laplace operator $$\Delta = \nabla^t \cdot \nabla = \nabla_{\mathbf{x}}^2 = \sum_{k=1}^{d} \frac{\partial^2}{\partial x_k^2}$$

(in *d* dimensions)

PDE Classification

- **Order** of PDE = order of highest partial derivative
- A PDE is **linear** if the unknown function u and its partial derivatives only occur linearly

Linear example

$$u_t + c \cdot u_x = 0$$

Advection equation

Nonlinear example

$$u_t + u \cdot u_x = 0$$

Burgers' equation

- Coefficients of linear PDEs can be nonlinear functions

$$y^2 \cdot u_{yy} + x^2 \cdot u_{yy} = 0$$

PDE Classification

- 2^{nd} order linear PDEs are of high practical relevance

  $\Rightarrow$ *dedicated classification*

- A 2^{nd} order linear PDE in 2 variables has the form

$$Au_{xx} + 2Bu_{xy} + Cu_{yy} = F(x, y, u, u_x, u_y)$$

- A 2^{nd} order linear PDE in 2 variables is
  - **Hyperbolic** $B^2\text{-}AC > 0$ *(Wave equation)*
  - **Parabolic** $B^2\text{-}AC = 0$ *(Heat equation)*
  - **Elliptic** $B^2\text{-}AC < 0$ *(Laplace equation)*

Model Equations

- Fundamental behavior of many important PDEs is well-captured by three model linear equations:

Solve numerically?

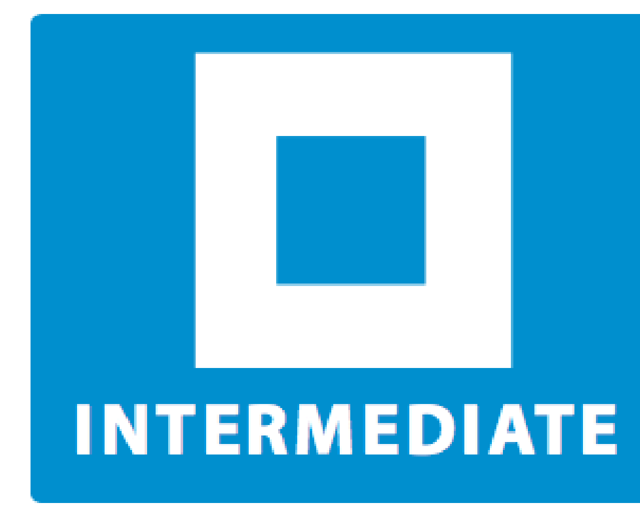
"Laplacian" (more later!)

LAPLACE EQUATION ("ELLIPTIC") $$\Delta u = 0$$

"what's the smoothest function interpolating the given boundary data"

HEAT EQUATION ("PARABOLIC") $$\dot{u} = \Delta u$$

"how does an initial distribution of heat spread out over time?"



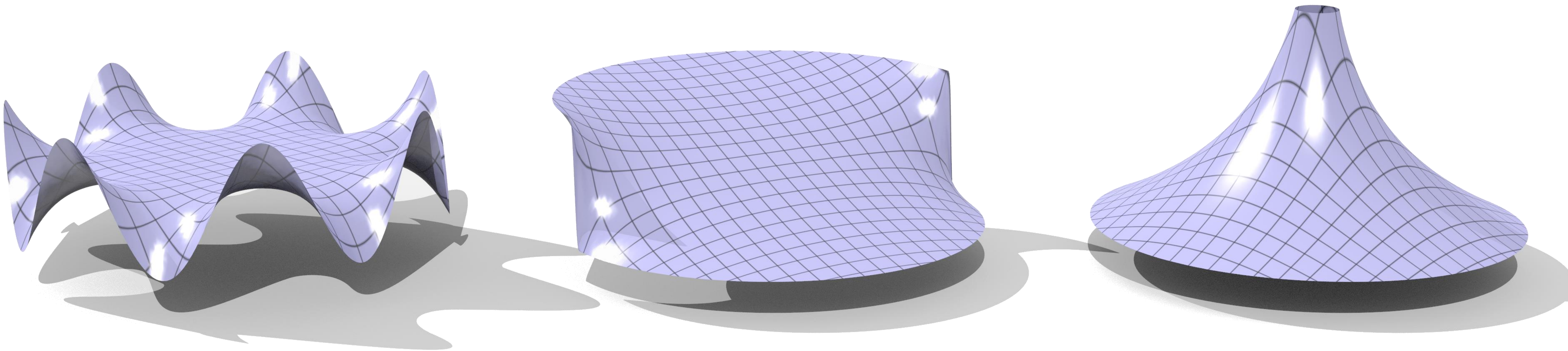
WAVE EQUATION ("HYPERBOLIC") $$\ddot{u} = \Delta u$$

"if you throw a rock into a pond, how does the wavefront evolve over time?"

[NONLINEAR + HYPERBOLIC + HIGH-ORDER]

Elliptic PDEs / Laplace Equation

What's the smoothest function interpolating given boundary data?

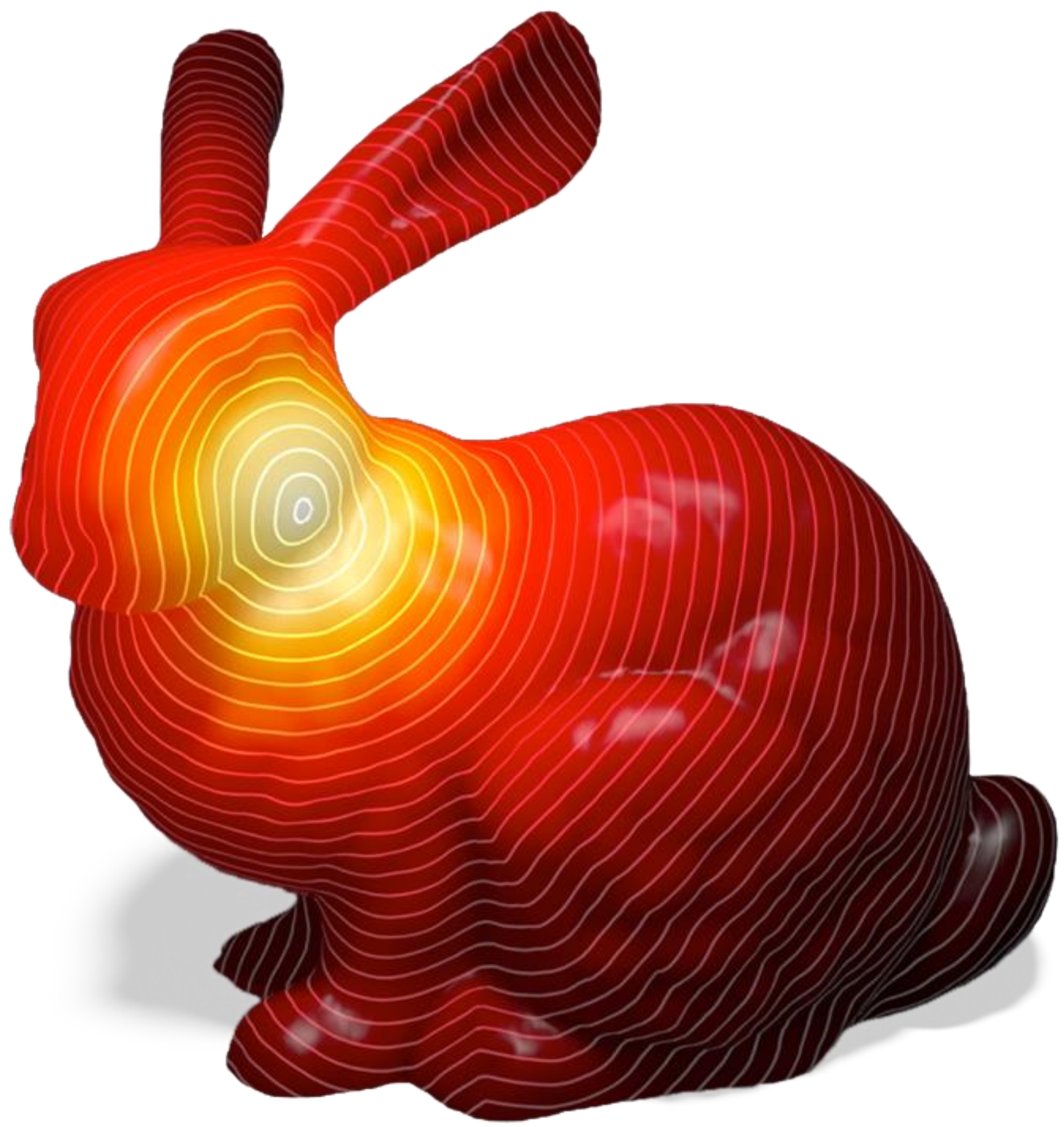


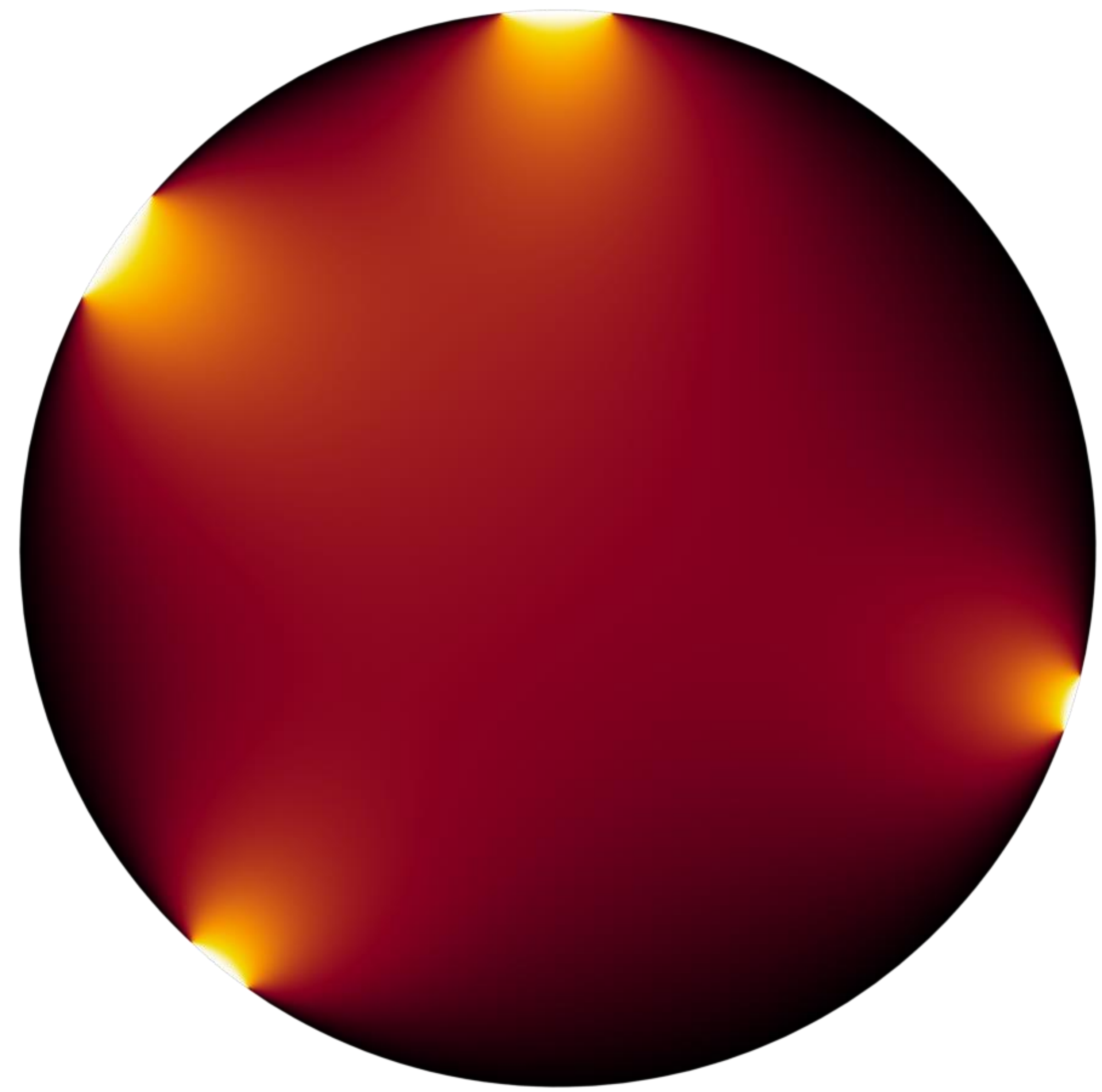
- Conceptually: each value is at the average of its "neighbors"
- Very robust to errors: just keep averaging with neighbors!

Image from Solomon, Crane, Vouga, "Laplace-Beltrami: The Swiss Army Knife of Geometry Processing"

Parabolic PDEs / Heat Equation

How does an initial distribution of heat spread out over time?





- After a long time, solution is same as Laplace equation!
- Models damping / viscosity in many physical systems

Hyperbolic PDEs / Wave Equation

If you throw a rock into a pond, how does the wavefront evolve over time?

- No steady state solution
- Errors made at the beginning will persist for a long time

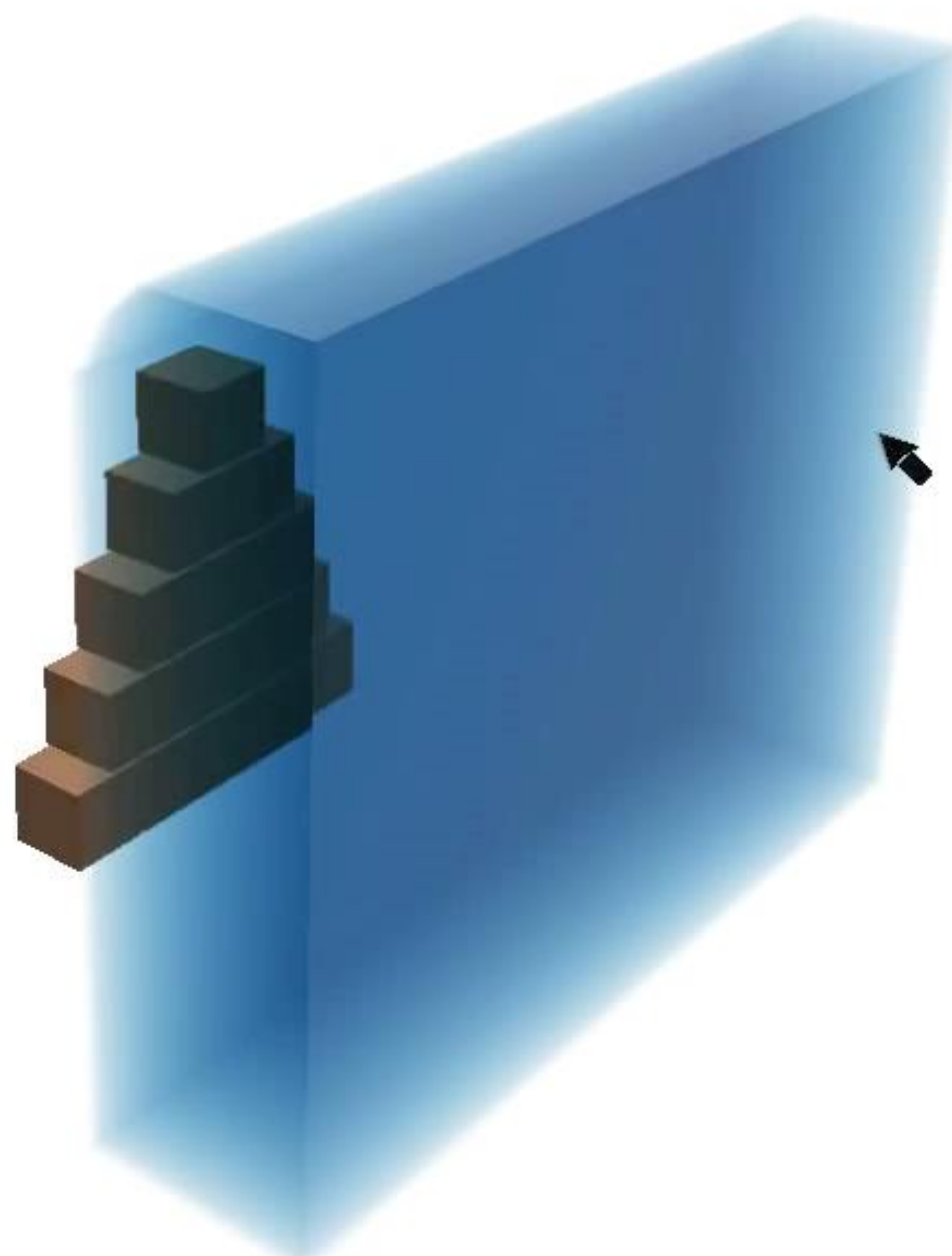
How can we solve PDEs?

Numerical Solution of PDEs—Overview

- Like ODEs, many interesting PDEs are difficult or impossible to solve analytically
- Must instead use numerical integration
- Basic strategy:
  - pick a spatial discretization
  - pick a time discretization (forward Euler, backward Euler...)
  - as with ODEs, run a time-stepping algorithm

Real Time PDE-Based Simulation (Fire)

Real Time PDE-Based Simulation (Water)

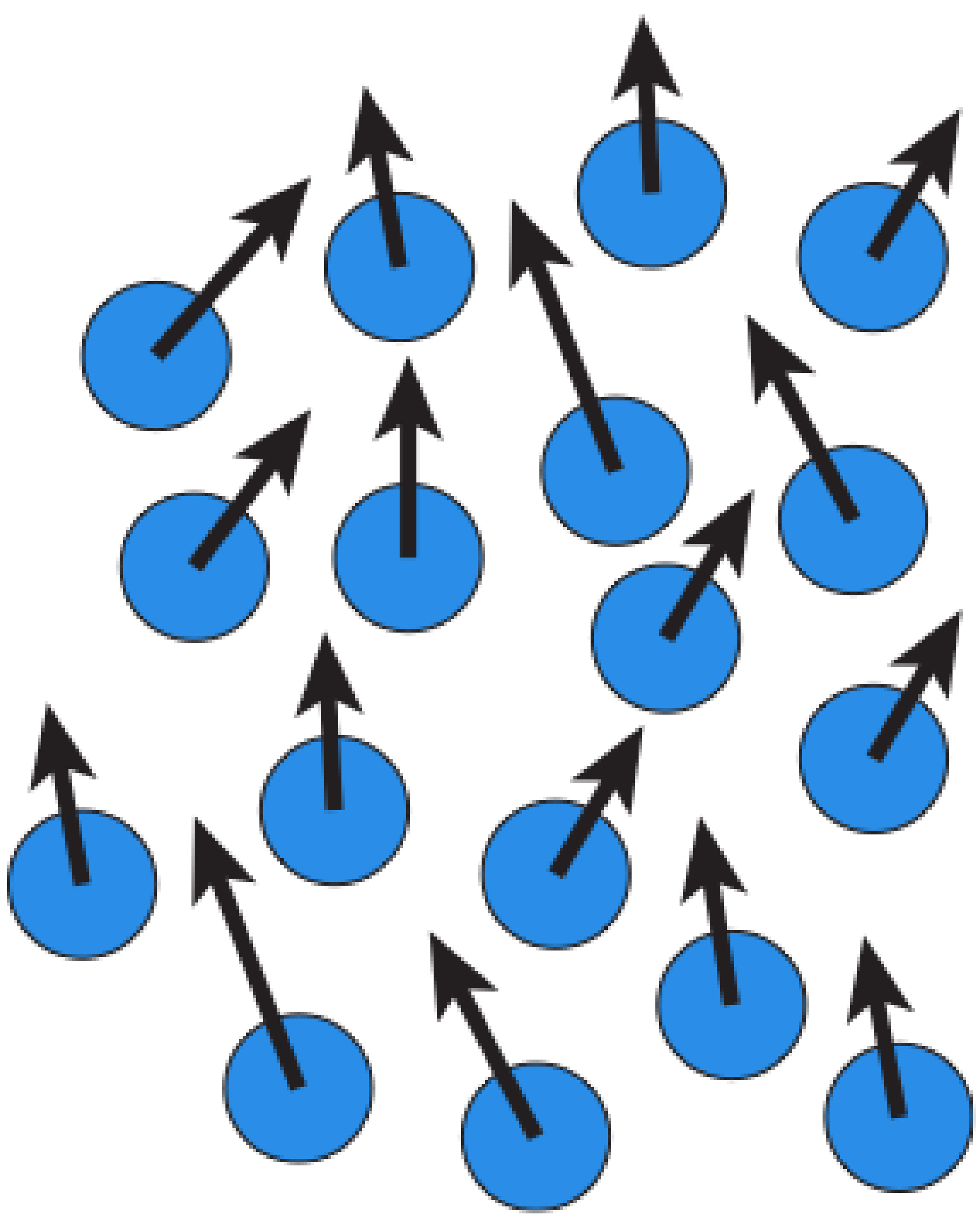


[Nuttapong Chentanez](), [Matthias Müller](), ***"Real-time Eulerian water simulation using a restricted tall cell grid"***

Lagrangian vs. Eulerian

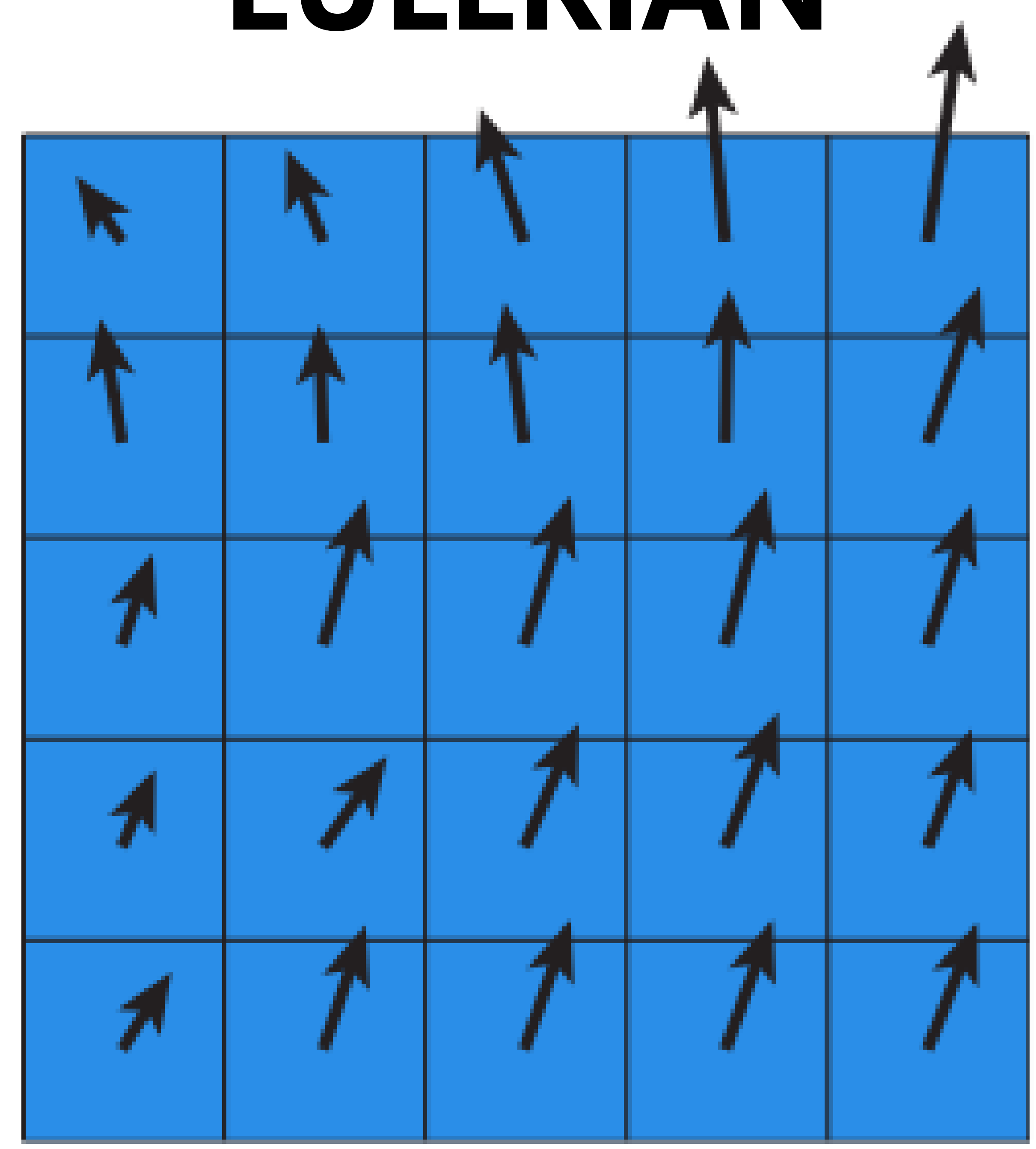
- Two basic ways to discretize space: Lagrangian & Eulerian (particle-based & grid-based)
- Suppose you want to keep track of the weather…

LAGRANGIAN



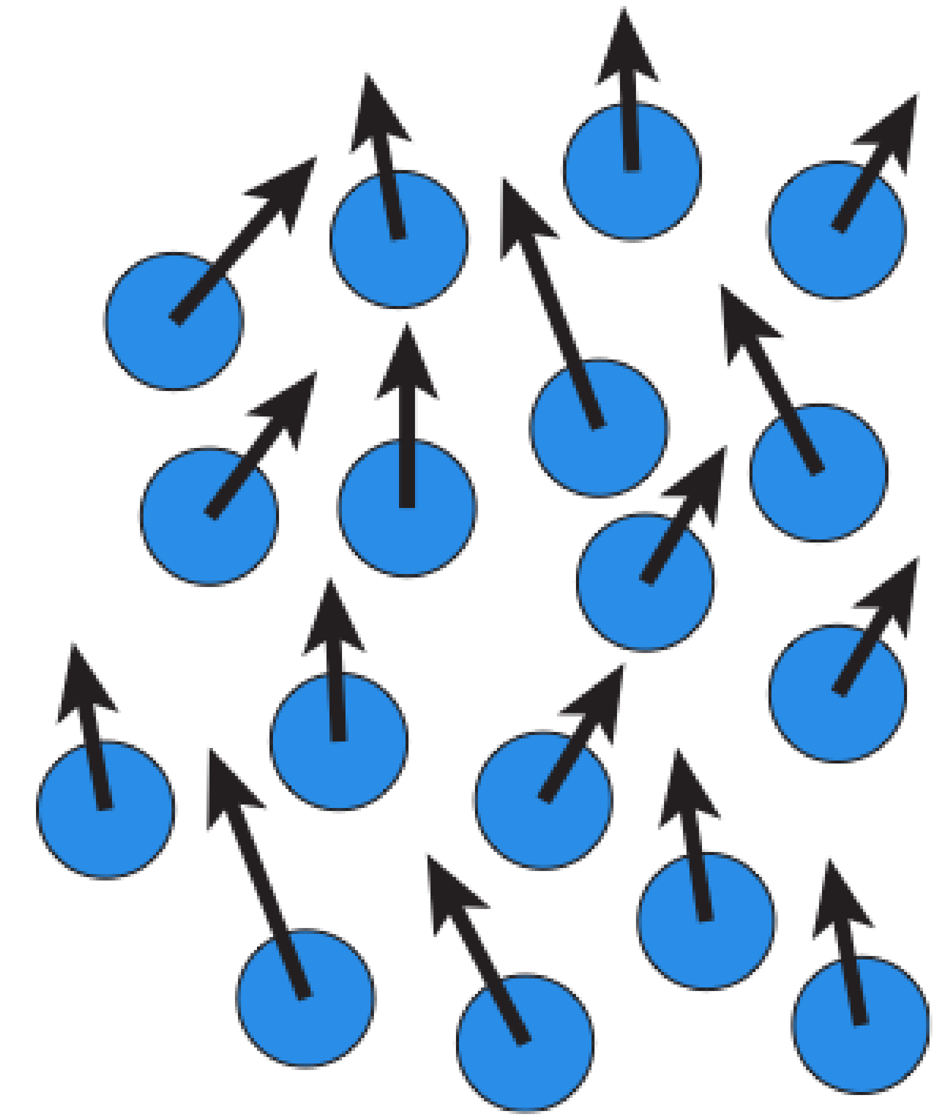
track moving particles and read what they are measuring

EULERIAN

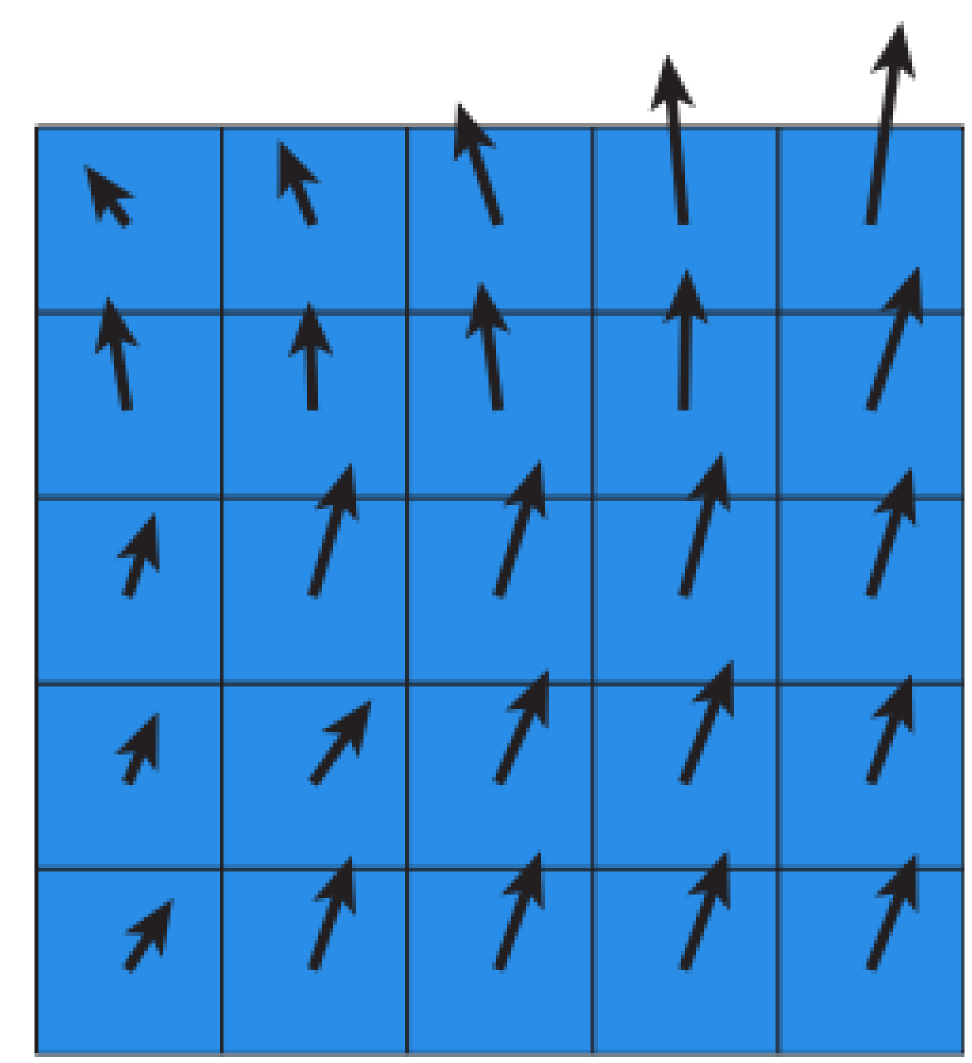


record temperature at fixed locations in space

Lagrangian vs. Eulerian—Trade-Offs



- **Lagrangian**
  - conceptually easy (like polygon soup!)
  - resolution/domain not limited by grid
  - good particle distribution can be tough
  - finding neighbors can be expensive
- **Eulerian**
  - fast, regular computation
  - easy to represent, e.g., smooth surfaces
  - simulation “trapped” in grid



Mixing Lagrangian & Eulerian

- Of course, no reason you have to choose just one!
- Many modern methods mix Lagrangian & Eulerian:
  - PIC/FLIP, particle level sets, mesh-based surface tracking, Voronoi-based, arbitrary Lagrangian-Eulerian (ALE), ...
- *Pick the right tool for the job!*

Maya *Bifrost*

How do we solve *easy* PDEs?

Numerical PDEs—Basic Strategy

- Pick PDE that models phenomenon of interest
  - Which quantity do we want to solve for?
- Pick spatial discretization
  - How do we approximate derivatives in space?
- Pick time discretization
  - How do we approximate derivatives in time?
  - When do we evaluate forces?
  - Forward Euler, backward Euler, symplectic Euler, ...
- Finally, we have an *update rule*
- Repeatedly solve to generate an animation

The Laplace Operator

- All of our model equations used the Laplace operator

Nabla operator

$$\nabla = \left(\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_d}\right)^t \qquad \nabla u = \left(\frac{\partial u}{\partial x_1}, \dots, \frac{\partial u}{\partial x_d}\right)^t$$

Laplace operator

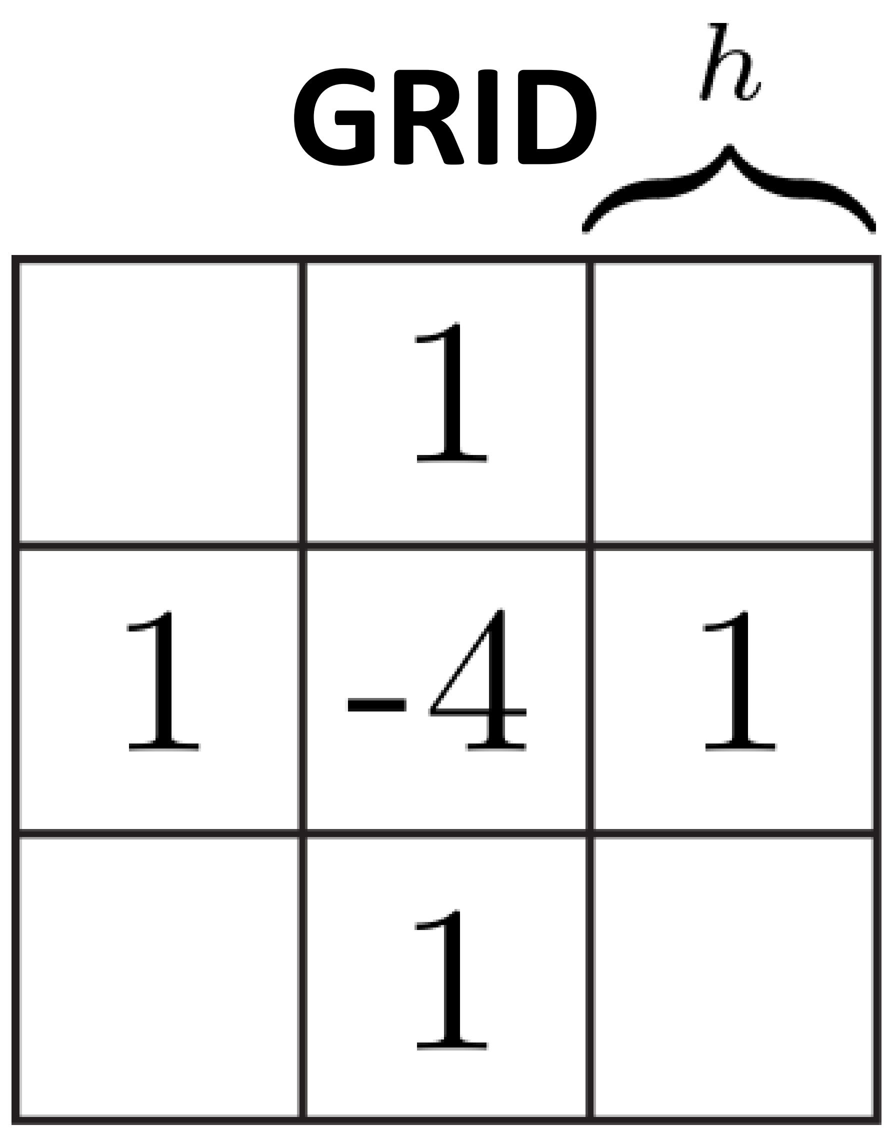
$$\Delta = \nabla \cdot \nabla = \sum_{k=1}^{d} \frac{\partial^2}{\partial x_k^2} \qquad \Delta u = \underbrace{\nabla}_{\text{div}} \cdot \underbrace{\nabla u}_{\text{grad}}$$

$$\Delta u = \frac{\partial u^2}{\partial x_1^2} + \cdots + \frac{\partial u^2}{\partial x_n^2}$$

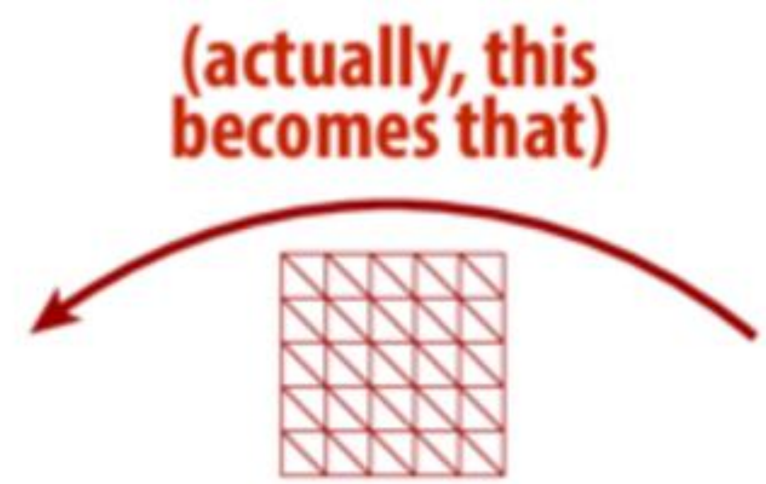
Discretizing the Laplacian

- How do we approximate the Laplacian?
- Depends on discretization (Eulerian, Lagrangian, grid, mesh, ...)
- Two extremely common ways in graphics:

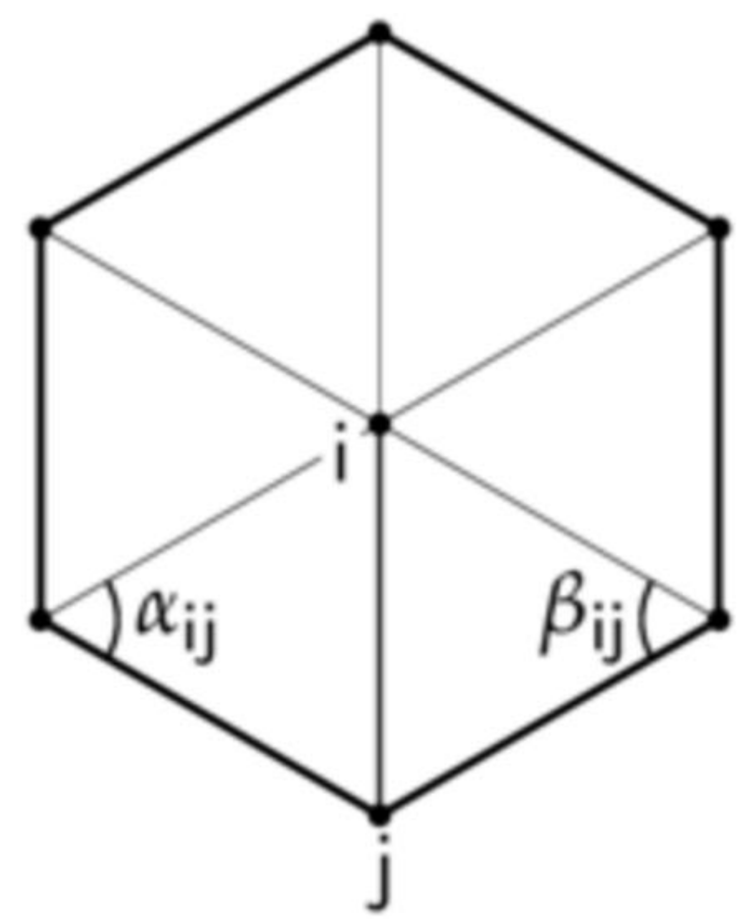
GRID



$$\frac{4u_{ij} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}}{h^2}$$



TRIANGLE MESH



$$\frac{1}{2}\sum_j (\cot\alpha_{ij} + \cot\beta_{ij})(u_j - u_i)$$

Numerically Solving the Laplace Equation

- Want to solve $\Delta u = 0$
- Plug in one of our discretizations, e.g.,

	c	
d	a	b
	e	

$$\frac{4a - b - c - d - e}{h} = 0$$

$$\iff \quad a = \tfrac{1}{4}(b + c + d + e)$$

- At solution that solves the Laplace Equation, each value is the average of neighboring values.
- How do we solve this?
- One idea: keep averaging with neighbors! (“Jacobi method”)
- Correct, but *slow* convergence

Boundary Conditions for Discrete Laplace

- What values do we use to compute averages near the boundary?

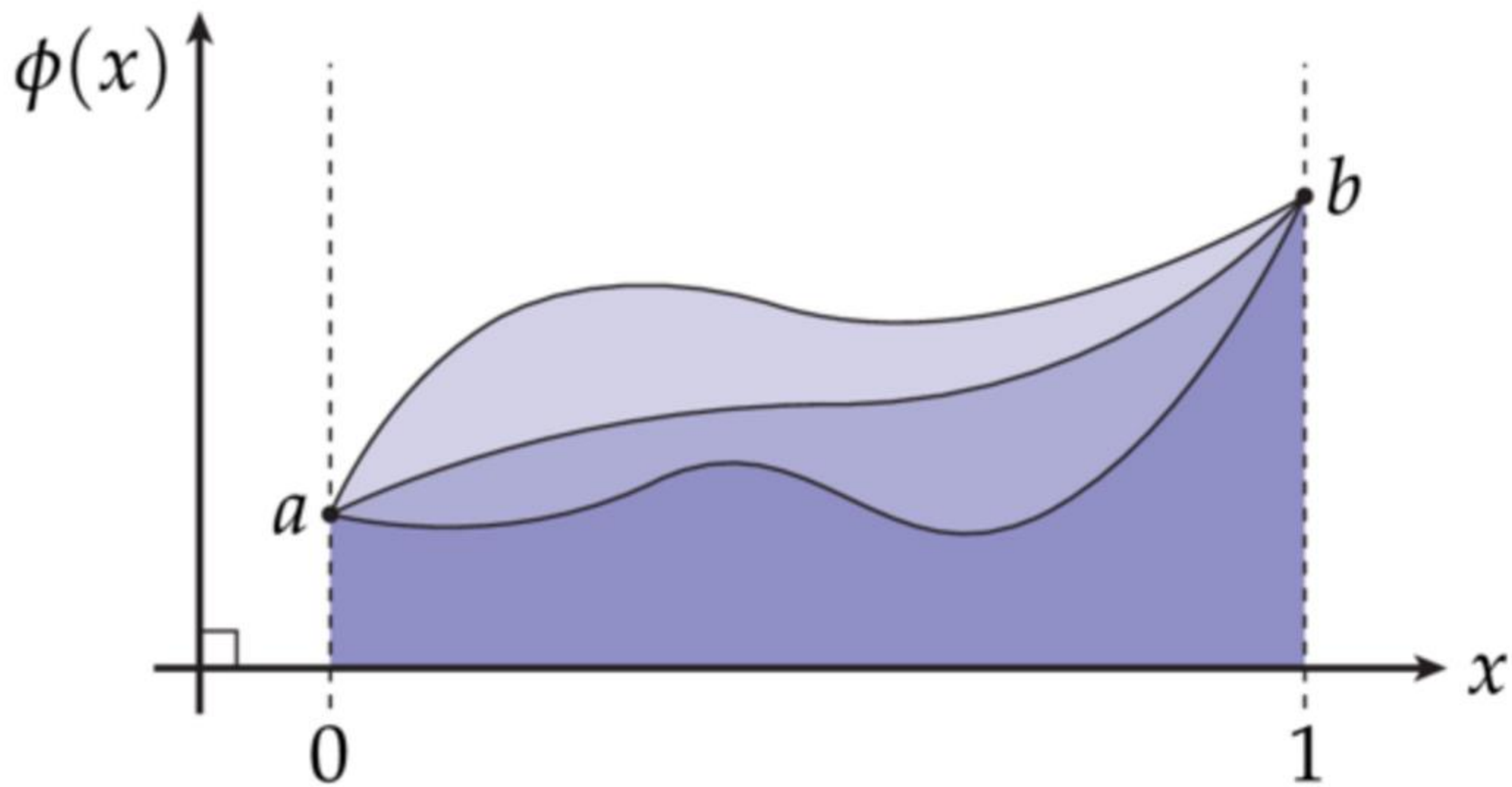
	c	
?	a	b
	e	

$$a = \frac{1}{4}(b + c + ? + e)$$

- Two basic boundary conditions:
  1. *Dirichlet*—boundary data always set to fixed values
  2. *Neumann*—specify derivative (difference) across boundary

Dirichlet Boundary Conditions

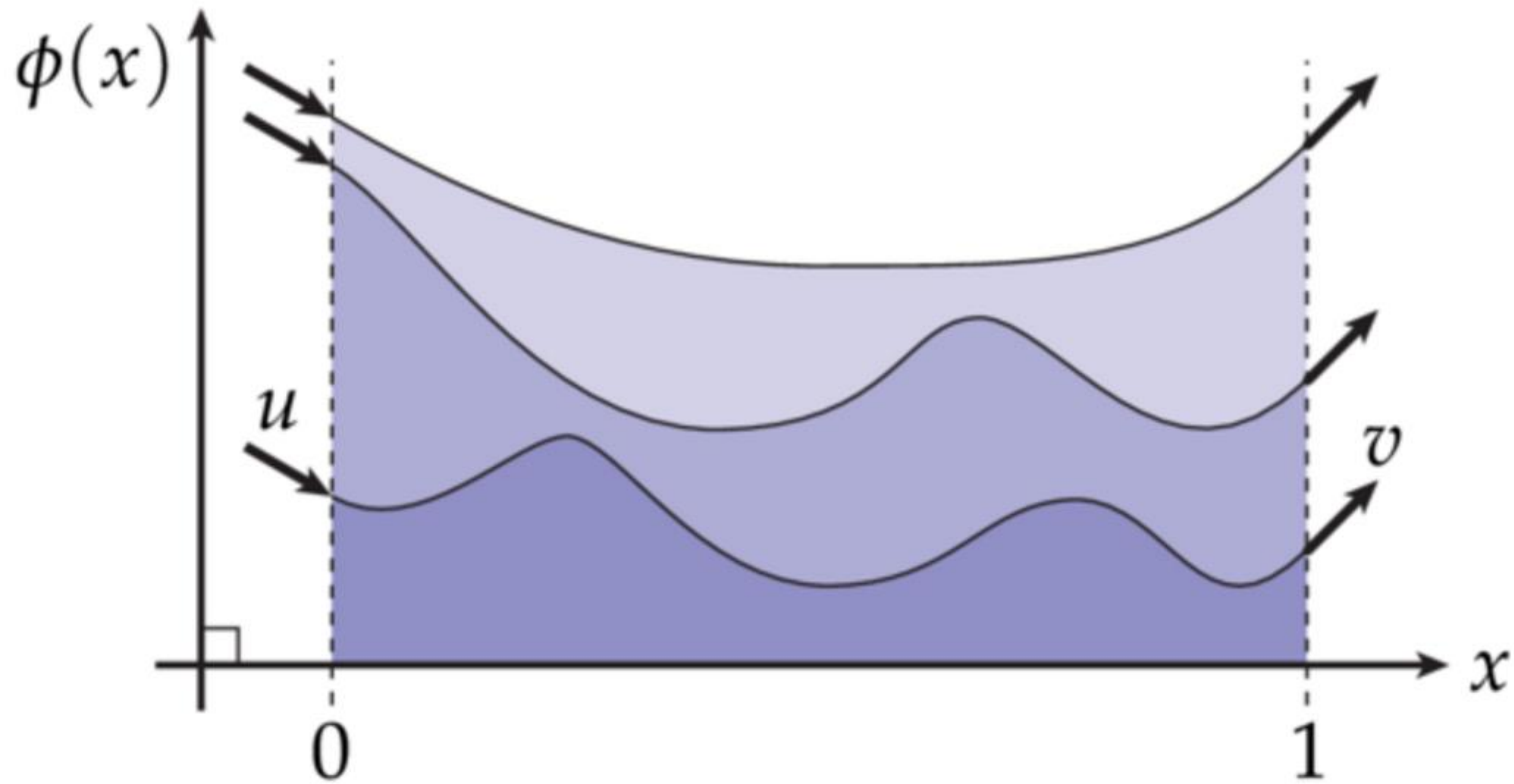
- *Dirichlet* means “prescribe values”
- E.g., $\Phi(0)=a$, $\Phi(1) = b$



- Many possible functions “in between”!

Neumann Boundary Conditions

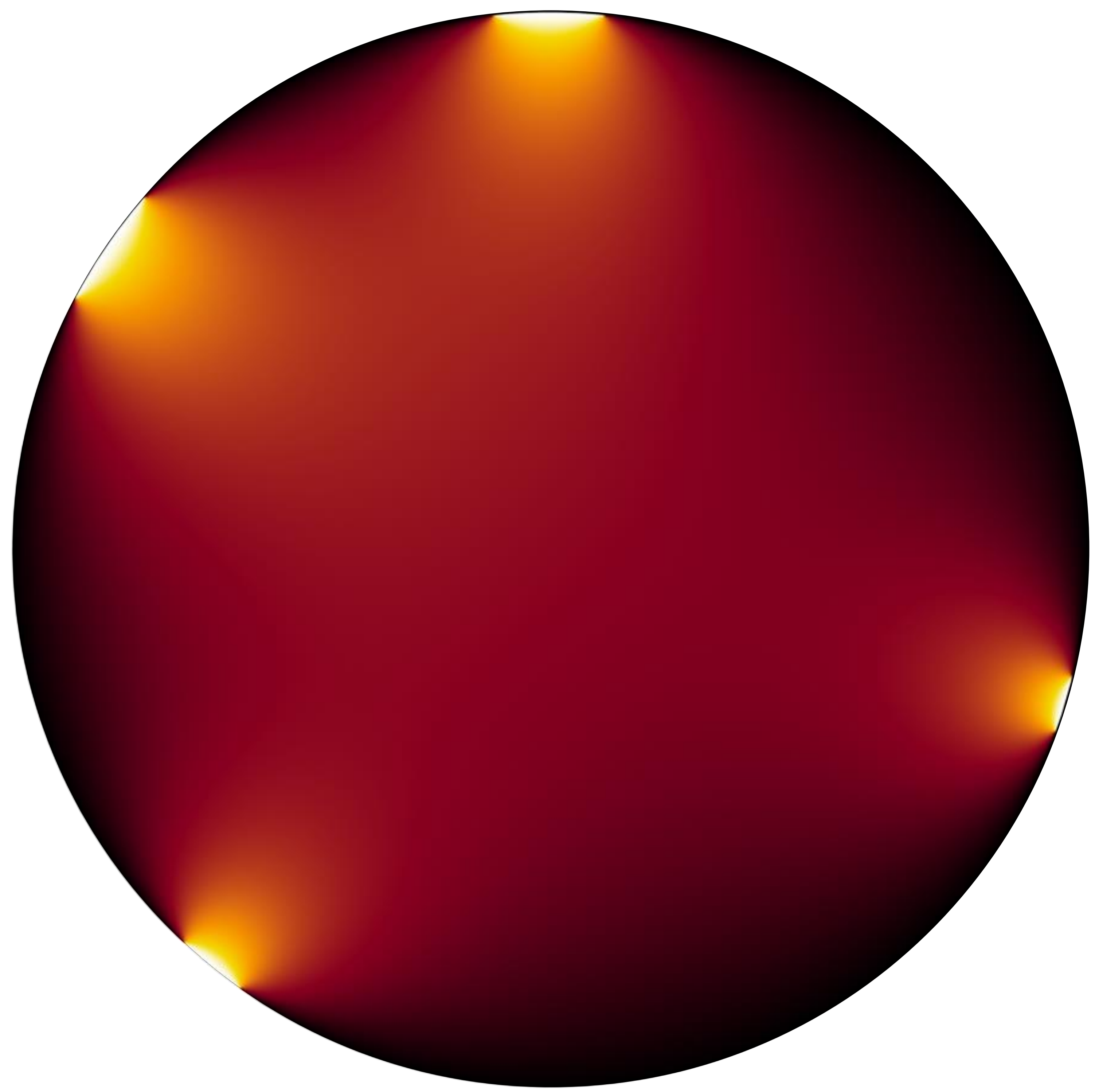
- *Neumann* means "prescribe derivatives"
- E.g., $\Phi'(0)=u$, $\Phi'(1) = v$



- Again, many possible functions

2D Laplace w/ Dirichlet BCs

- 2D Laplace: $\Delta\ \Phi=0$
- Q: Can satisfy any Dirichlet BCs? (given data along boundary)



- Yes: Laplace is long-time solution to heat flow
- Data is “heat” at boundary. Then just let it flow...

Solving the Heat Equation

- Back to our three model equations, want to solve *heat diffusion equation*

$$\dot{u} = \Delta u$$

- Just saw how to discretize Laplacian
- Also know how to do time (forward Euler, backward Euler, ...)
- E.g., forward Euler:

$$u^{k+1} = u^k + \Delta u^k$$

- Q: On a grid, what's our overall update now at $u_{i,j}$?

$$u_{i,j}^{k+1} = u^k + \frac{\tau}{h^2}(4u_{i,j}^k - u_{i+1,j}^k - u_{i-1,j}^k - u_{i,j+1}^k - u_{i,j-1}^k)$$

- *Not* hard to implement! Loop over grid, add up some neighbors.

Solving the Wave Equation

- Finally, wave equation:

$$\ddot{u} = \Delta u$$

- Not much different; now have 2nd derivative in time
  - Convert to two 1st order (in time) equations:

$$\dot{u} = v, \quad \dot{v} = \Delta u$$

  - Evaluate spatial derivative:

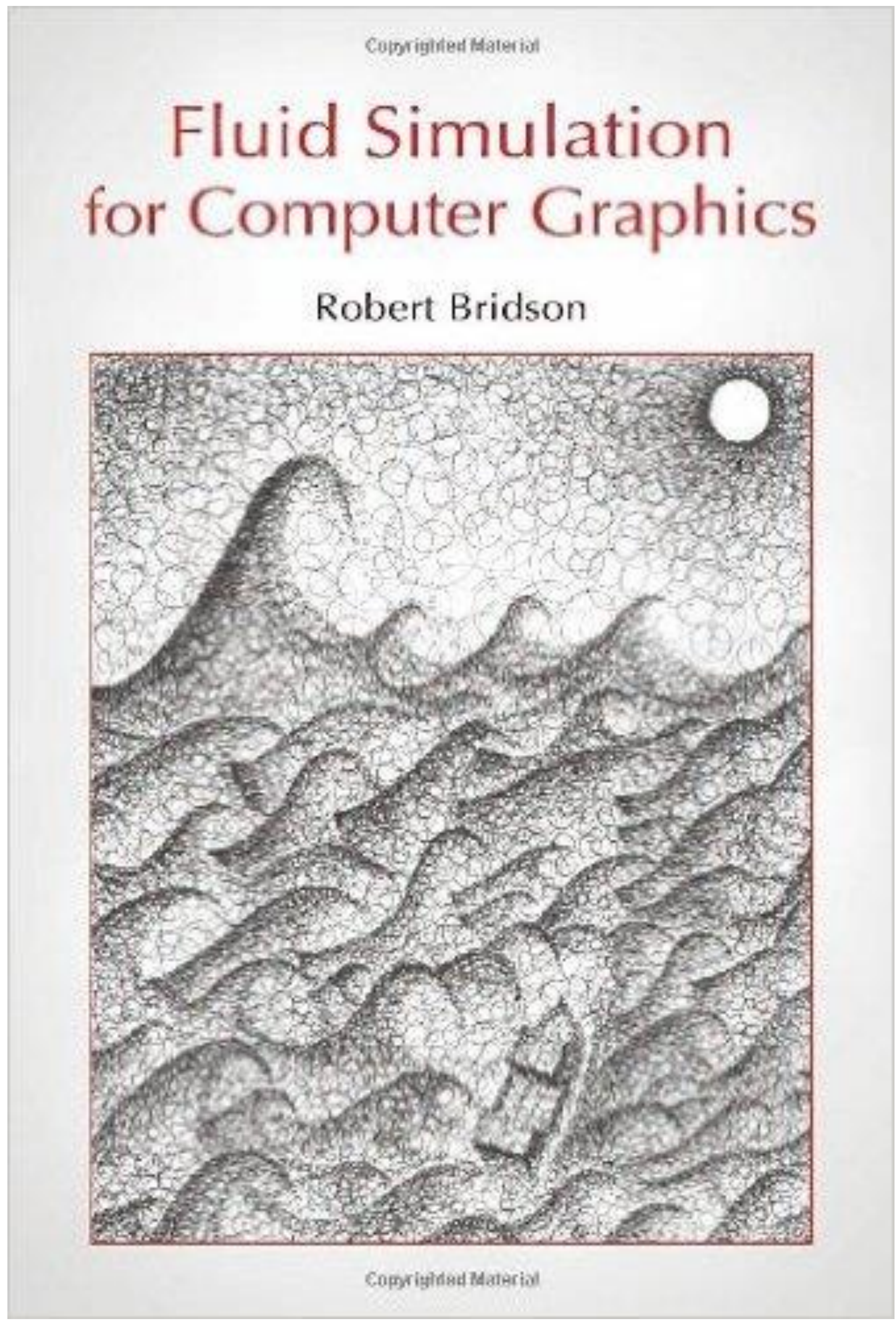
$$\frac{u^{k+1} - 2u^k + u^{k-1}}{\tau^2} = \Delta u^k$$

  - Integrate forward in time using, for example, symplectic Euler
- This is just one way to solve this PDE. There are *many* other choices!

Want to Know More?

- There are some good books:
- And papers:

http://www.physicsbasedanimation.com/



- Also, what did the folks who *wrote* these books & papers read?