Simulations
• Physically-Based Animation

• ODE Solvers

• PDE Solvers
What natural phenomenon can we simulate?
Flocking Simulation
Crowd Simulation
Crowd Simulation
Fluid Simulation

Sph particle fluid
300,000 particles
71 min bake
3.5 min per renderframe
Granular Material Simulation
Molecular Dynamics Simulation
Cosmological Simulation
Cloth Simulation
Hair Simulation
Elasticity Simulation
Fracture Simulation
Snow Simulation
Ok, simulation is cool,
How can we solve them analytically?
• Physically-Based Animation

• ODE Solvers

• PDE Solvers
• **Ordinary Differential Equations (ODEs)** have a derivative with respect to one other variable
  - Ordinary - involves derivatives in time but not space

• Many dynamical systems can be described via an ODE in generalized coordinates:

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

• ODEs can also be used to model rates of growth proportional to some original value:

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

  • **Solution:** \(u(t) = be^{at}\)
  • Describes exponential decay \((a < 1)\), or stock \((a > 1)\)
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:

$$\ddot{q} = \frac{g}{m}$$

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

$$q(t) = q_0 + tv_0 + \frac{t^2}{2m}g$$

** Yes, the rock is spherical and has uniform density

Easy! We don’t need a computer for simulation!
Example: Pendulum

- Mass on end of a bar, swinging under gravity
- What are the equations of motion?
  - Same as “rock” problem, but constrained
  - Response tension $T(q)$ now varies based on configuration $q$
- Could use a “force diagram”
  - You probably did this for many hours in high school/college

![Diagram of pendulum with forces](image)

Ok, maybe bring back the computer...
Lagrangian Mechanics

- Beautifully simple recipe:
  - Write down kinetic energy $K$
  - Write down potential energy $U$
  - Write down Lagrangian
    \[ \mathcal{L} := K - U \]
  - Dynamics then given by Euler-Lagrange equation

- 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)
Lagrangian Mechanics: Pendulum

Simple configuration parameterization:

\[ q = \theta \]

Kinetic energy:

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

Potential energy:

\[ U = mgh = -mgL \cos \theta \]

Euler-Lagrange equations:

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

\[ \frac{\partial \mathcal{L}}{\partial \dot{q}} = \frac{\partial \mathcal{L}}{\partial \dot{\theta}} = mL^2 \dot{\theta} \quad \frac{\partial \mathcal{L}}{\partial q} = \frac{\partial \mathcal{L}}{\partial \theta} = -mgL \sin \theta \]

\[ \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}} = \frac{\partial \mathcal{L}}{\partial q} \quad \Rightarrow \quad \ddot{\theta} = -\frac{g}{L} \sin \theta \]
Solving The Pendulum

Simple equation for the pendulum:

\[ \ddot{\theta} = -\frac{g}{L} \sin \theta \]

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

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

\[ \sin \theta = \theta \text{ for small angles} \]
\[ \frac{d^2}{d\theta^2} \cos \theta = -\cos \theta \]

In general, there is no closed form solution!
Hence, we must use a numerical approximation

And pendulums are supposed to be easy to simulate!
Harder: Double Pendulum

- Blue ball swings from pendulum
  - Green ball swings from blue ball
  - Forces will act on each other
    - Newton’s 3rd law

- Simple system...not-so-simple motion
  - Chaotic: perturb input, wild changes to output
  - Must again use numerical approximation
Even Harder: N-Body Problem

- Consider the Earth, moon, and sun
  - Where do they go?
  - Solution is trivial for two bodies
    - Assume one is fixed, solve for the other

- As soon as \( n \geq 3 \), gets chaotic
  - No closed form solution

- **Fun Fact**: this is a 15-418 homework assignment
  - Glad you aren’t taking 15-418…
Ok, so solving solutions analytically is hard, How can we solve them numerically?

closed-form solution

guess-and-check
Numerical Integration

- **Key idea:** replace derivatives with differences
  - With ODEs, only need to worry about derivative in **time**

- Replace time-continuous configuration function $q(t)$ with samples $q_k$ in time

\[
v(q(t)) = \frac{d}{dt} q(t) = \frac{q_{k+1} + q_k}{\tau}
\]

- Deriving Forward & Backward Euler (2022) Steve Brunton
• **Idea:** evaluate velocity at current configuration

• New configuration can then be written explicitly in terms of known data:

\[
q_{k+1} = q_k + \tau \cdot v(q_k)
\]

• Very intuitive: walk a tiny bit in the direction of the velocity

Where did all this energy come from?
Forward Euler Analysis

Let’s consider behavior of forward Euler for a simple linear ODE:

\[ \dot{q} = -aq, \quad a > 0 \]

\( q \) should decay over time (loss of energy to global system).

Forward Euler approximation is:

\[ q_{k+1} = q_k - \tau a q_k \]

\[ q_{k+1} = (1 - \tau a)q_k \]

Which means after \( n \) steps, we have:

\[ q_n = (1 - \tau a)^n q_0 \]

Decays only if \(|1 - \tau a| < 1\), or equivalently, if \( \tau < 2/a \)

**In practice:** need very small time steps if \( a \) is large
• **Idea:** evaluate velocity at next configuration

• New configuration implicit, output depends on input:

\[
q_{k+1} = q_k + \tau \ast \nu(q_{k+1})
\]

• Much harder to solve, since in general \( \nu \) can be very nonlinear!
  • More of a constraint problem: find a \( q_{k+1} \) that satisfies the above equation
  • Generally expensive to solve

Where did all this energy go?
Backward Euler Analysis

Again, let’s consider a simple linear ODE:

\[ \dot{q} = -aq, \quad a > 0 \]

\( q \) should decay over time (loss of energy to global system).

Backward Euler approximation is:

\[
\frac{(q_{k+1} - q_k)}{\tau} = -aq_{k+1} \\
\frac{q_{k+1}}{\tau} + aq_{k+1} = \frac{q_k}{\tau} \\
(1 + \tau a)q_{k+1} = q_k \\
q_{k+1} = \frac{1}{1+\tau a} q_k
\]

Which means after \( n \) steps, we have:

\[
q_n = \left(\frac{1}{1+\tau a}\right)^n q_0
\]

Decays only if \( |1 + \tau a| > 1 \), which is always true!

Backwards Euler is **unconditionally stable** for linear ODEs!
Symplectic Euler

- Nice alternative is Symplectic Euler
  - Update velocity using current configuration $q_k$
  - Update configuration using new velocity $v_{k+1}$

$$
\begin{align*}
    v_{k+1} &= v_k + \tau \cdot a(q_k) \\
    q_{k+1} &= q_k + \tau \cdot v_{k+1}
\end{align*}
$$

- Pendulum now conserves energy \textit{almost exactly}, forever

Proof? The analysis isn’t very easy…
Explicit Euler Methods

[ Forward ]
\begin{align*}
v_{k+1} &= v_k + \tau \ast a(q_k) \\
q_{k+1} &= q_k + \tau \ast v_k
\end{align*}

[ Symplectic ]
\begin{align*}
v_{k+1} &= v_k + \tau \ast a(q_k) \\
q_{k+1} &= q_k + \tau \ast v_{k+1}
\end{align*}

[ Verlet ]
\begin{align*}
v_{k+1} &= v_{k+0.5} + \frac{\tau}{2} \ast a(q_k) \\
q_{k+1} &= q_k + \tau \ast v_{k+1} \\
v_{k+1.5} &= v_{k+1} + \frac{\tau}{2} \ast a(q_k)
\end{align*}

[ RK2 ]
\begin{align*}
v'_{k+1} &= \tau \ast a(q_k) \\
v''_{k+1} &= \tau \ast a(q_k + \frac{v'_{k+1}}{2}) \\
q_{k+1} &= q_k + v''_{k+1}
\end{align*}

[ RK4 ]
\begin{align*}
v'_{k+1} &= \tau \ast a(q_k) \\
v''_{k+1} &= \tau \ast a(q_k + \frac{v'_{k+1}}{2}) \\
v'''_{k+1} &= \tau \ast a(q_k + \frac{v''_{k+1}}{2}) \\
v''''_{k+1} &= \tau \ast a(q_k + v''''_{k+1}) \\
q_{k+1} &= q_k + \frac{1}{6}(v'_{k+1} + 2v''_{k+1} + 2v'''_{k+1} + v''''_{k+1})
\end{align*}
• Physically-Based Animation

• ODE Solvers

• PDE Solvers
Partial Differential Equations

- **Partial Differential Equations (PDEs)** have a derivative with respect to multiple variables
  - ODE Simulations take derivatives w.r.t time
    \[ q_{k+1} = q_k + \tau f(q) \]
  - PDE Simulations take derivatives w.r.t time + space
    \[ q_{k+1} = q_k + \tau f(q) \]
  - Same function! But depends on how we parameterize the configuration \( q \)

- PDEs described with implicit definitions
  - Need to solve for the actual function
ODEs vs. PDEs

[ ODE ] throwing a rock

\[ \frac{d^2}{dt^2} x(t) = g \]

[ PDE ] throwing rock lands in pond

\[ \frac{d^2}{dt^2} h(t, x, y) = \Delta h(t, x, y) \]
ODEs vs. PDEs

Aren’t both a function of space and time?

A single object (rock) in time

 Millions of objects (droplets) in time

No additional space parameter

The region of droplets we want to solve over is our space

\[ \frac{d^2}{dt^2} x(t) = g \]

\[ \frac{d^2}{dt^2} h(t, x, y) = \Delta h(t, x, y) \]

[ ODE ] throwing a rock

[ PDE ] throwing rock lands in pond
Moving forward, we will denote:

\[ u(t, x) \]

As the values for which our PDE will solve for, and:

\[ \dot{u}, \ddot{u}, \dddot{u} \ldots \]

As temporal derivatives, and:

\[ u', u'', u''' \ldots \]

As spatial derivatives
The Laplacian Operator

• All of our model equations used the Laplace operator
  • Laplace Equation \( \Delta u = 0 \)
  • Heat Equation \( \dot{u} = \Delta u \)
  • Wave Equation \( \ddot{u} = \Delta u \)

• Unbelievably important object showing up everywhere across physics, geometry, signal processing, and more

• What does the Laplacian mean?
  • **Differential operator**: eats a function, spits out its 2nd derivative
  • What does that mean for a function: \( u : \mathbb{R}^n \rightarrow \mathbb{R} \)?
    • Divergence of gradient
      \[ \Delta u = \nabla \cdot \nabla u \]
    • Sum of second derivatives
      \[ \Delta u = \frac{\partial u^2}{\partial x_1^2} + \cdots + \frac{\partial u^2}{\partial x_n^2} \]
    • Deviation from local average
    • …
Modeling PDE Equations

Laplace Equation [Elliptic]
“What’s the smoothest function interpolating the given boundary data?”
\[ \Delta u = 0 \]

Heat Equation [Parabolic]
“How does an initial distribution of heat spread out over time?”
\[ \dot{u} = \Delta u \]

Wave Equation [Hyperbolic]
“If you throw a rock into a pond, how does the wavefront evolve over time?”
\[ \ddot{u} = \Delta u \]

Nonlinear + Hyperbolic + High-Order
“A lot of real life phenomenon”

???
Modeling PDE Equations

Laplace Equation [Elliptic]
“What’s the smoothest function interpolating the given boundary data?”

\[ \Delta u = 0 \]

Heat Equation [Parabolic]
“How does an initial distribution of heat spread out over time?”

\[ \dot{u} = \Delta u \]

Wave Equation [Hyperbolic]
“If you throw a rock into a pond, how does the wavefront evolve over time?”

\[ \ddot{u} = \Delta u \]

Nonlinear + Hyperbolic + High-Order
“A lot of real life phenomenon”

???
Laplace Equation

“What’s the smoothest function interpolating the given boundary data?”

- Conceptually each value is at the average of its “neighbors”
  - Very robust to errors: just keep averaging with neighbors
  - Errors will eventually get averaged out/diminish
Modeling PDE Equations

Laplace Equation [Elliptic]
“What’s the smoothest function interpolating the given boundary data?”
\[ \Delta u = 0 \]

Heat Equation [Parabolic]
“How does an initial distribution of heat spread out over time?”
\[ \dot{u} = \Delta u \]

Wave Equation [Hyperbolic]
“If you throw a rock into a pond, how does the wavefront evolve over time?”
\[ \ddot{u} = \Delta u \]

Nonlinear + Hyperbolic + High-Order
“A lot of real life phenomenon”
???
Heat Equation

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

- After a long time, solution is same as Laplace equation
  - Treat 3D problem over a mesh as a 2D surface problem
- Models damping/viscosity in many physical system
Modeling PDE Equations

Laplace Equation [Elliptic]
“What’s the smoothest function interpolating the given boundary data?”
\[ \Delta u = 0 \]

Heat Equation [Parabolic]
“How does an initial distribution of heat spread out over time?”
\[ \dot{u} = \Delta u \]

Wave Equation [Hyperbolic]
“If you throw a rock into a pond, how does the wavefront evolve over time?”
\[ \ddot{u} = \Delta u \]

Nonlinear + Hyperbolic + High-Order
“A lot of real life phenomenon”
???
Wave Equation

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

• Difficult! Errors made at the beginning will persist for a long time
  • Errors may even compound and explode/break simulation
Modeling PDE Equations

Laplace Equation [Elliptic]
“What’s the smoothest function interpolating the given boundary data?”
\[ \Delta u = 0 \]

Heat Equation [Parabolic]
“How does an initial distribution of heat spread out over time?”
\[ \dot{u} = \Delta u \]

Wave Equation [Hyperbolic]
“If you throw a rock into a pond, how does the wavefront evolve over time?”
\[ \ddot{u} = \Delta u \]

Nonlinear + Hyperbolic + High-Order
“A lot of real life phenomenon”

? ? ?
PDE Anatomy

• How are derivatives combined?
  • **Linear**: derivatives are not multiplied with each other
  • **Nonlinear**: derivatives multiplied with each other

\[ \dot{u} + uu' = au'' \]  \quad \text{[ Burger’s equation ]}

\[ \dot{u} = au'' \]  \quad \text{[ diffusion equation ]}

• What is the highest order derivative in space and time?

\[ \frac{\partial u}{\partial t} + uu' = au'' \]  \quad \text{[ Burger’s equation ]}

\[ \frac{\partial u}{\partial t} = au'' \]  \quad \text{[ diffusion equation ]}

• The higher the order, the harder to solve!
Great, but how do we solve PDEs?
Numerically Solving a PDE

• PDEs are (near) impossible to solve analytically
  • Need to solve numerically

• **Algorithm:**
  • Pick a time discretization to compute temporal derivatives
    • Forward Euler, Symplectic, ...
  • Pick a spatial discretization to compute spatial derivatives
    • Lagrangian, Eulerian, ...
  • Perform time-stepping to advance solution

• Historically, very expensive
  • Only for “hero shots” in movies

• Computers are even faster nowadays
  • Can solve PDEs in real-time

Titanic (1997) James Cameron
Lagrangian vs. Eulerian

[ Lagrangian ]
track position & velocity of moving particles

[ Eulerian ]
track velocity (or flux) at fixed grid locations
Lagrangian vs. Eulerian

- **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
  - [+] Good cache coherence
  - [+] Easy to represent
  - [-] Simulation “trapped” in grid
  - [-] Grid causes “numerical diffusion” (blur/aliasing)
  - [-] Need to understand PDEs (but you will!)

- Where have we seen these formats before?
  - Rasterization!
    - **Lagrangian** is the primitives in our scene
    - **Eulerian** is the pixel representations on our displays
Mixing Lagrangian & Eulerian

• 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!
  • If you can’t pick one, pick them all!
The Laplacian Operator

• All of our model equations used the Laplace operator
  • Laplace Equation \( \Delta u = 0 \)
  • Heat Equation \( \dot{u} = \Delta u \)
  • Wave Equation \( \ddot{u} = \Delta u \)

• Unbelievably important object showing up everywhere across physics, geometry, signal processing, and more

• What does the Laplacian mean?
  • **Differential operator:** eats a function, spits out its 2nd derivative
  • What does that mean for a function: \( u: \mathbb{R}^n \rightarrow \mathbb{R} \)?
    • Divergence of gradient
      \[
      \Delta u = \nabla \cdot \nabla u
      \]
    • Sum of second derivatives
      \[
      \Delta u = \frac{\partial u^2}{\partial x_1^2} + \cdots + \frac{\partial u^2}{\partial x_n^2}
      \]
    • Deviation from local average
    • ...
Discretizing The Laplacian

Consider the Laplacian as a sum of second derivatives:

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

How do we compute this numerically?

Consider a non-differentiable function with evaluated samples \( x_0, x_1, \ldots \)

- The first derivative approximated is:

\[ u'(x_i) \approx \frac{u_{i+1} - u_i}{h} \]

- The second derivative approximated is:

\[ u''(x_i) \approx \frac{u'_i - u'_{i-1}}{h} \approx \frac{(u_{i+1} - u_i) - (u_i - u_{i-1})}{h^2} = \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} \]

- Known as the finite difference approach to PDEs
Discretizing The Laplacian

What if \( u \) is not a 1D function...

If the mesh is a grid, equations become the same

- \( [ \text{Grid} ] \)
  \[
  4u_{ij} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = \frac{h^2}{h^2}
  \]

- \( [ \text{Triangle Mesh} ] \)
  \[
  \frac{1}{2} \sum_j \left( \cot \alpha_{ij} + \cot \beta_{ij} \right) (u_j - u_i)
  \]
Numerically Solving The Laplacian

Want to solve $\Delta u = 0$:

$$\frac{4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1}}{h^2} = 0$$

Can isolate for $u_{i,j}$:

$$\iff u_{i,j} = \frac{1}{4} (u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1})$$

• If $u$ is a solution, then each value must be the average of the neighboring values

• How do we solve this?
  • **Idea:** keep averaging with neighbors! (“Jacobi method”)
  • Correct, but slow
    • Much better to use modern linear solver
Linearly Solving The Laplacian

- We have a bunch of equations of the form:
  \[4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = 0\]

- Index 2D grid using 1D indices
  - Create a matrix with all equations (these are our constraints)

\[
\begin{bmatrix}
-4 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & -4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & -4 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & -4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -4 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -4 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & -4 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4 \\
u_5 \\
u_6 \\
u_7 \\
u_8 \\
u_9 \\
u_{10} \\
u_{11} \\
u_{12} \\
u_{13} \\
u_{14} \\
u_{15} \\
u_{16} \\
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
\end{bmatrix}
\]

What is the issue with this?

Make sure to use a sparse solver!
Boundary Conditions

- We need boundary conditions that make our solution non-zero
  - Essentially, what is the data we want to interpolate?

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

- Three types of boundary conditions:
  - **Dirichlet**: boundary data always set to fixed values
  - **Neumann**: specify derivatives across boundary
  - **Robin**: mix of fixed and derivative values
    - Many more in general, but this is all we will cover
Dirichlet Boundary Conditions

**Dirichlet:** boundary data always set to fixed values

Example: \( \phi(0) = a, \phi(1) = b \)

Many possible functions interpolate values in between
Neumann Boundary Conditions

**Neumann:** specify derivatives across boundary

Example: \( \phi'(0) = u, \; \phi'(1) = v \)

Again, many possible functions
Dirichlet + Neumann Boundary Conditions

**Dirichlet**: boundary data always set to fixed values
**Neumann**: specify derivatives across boundary

\[ \phi'(0) = u, \ \phi(1) = b \]

Still, many possible functions

What about (Robin): \[ \phi'(0) + \phi(0) = p, \ \phi'(1) + \phi(1) = q \]
We can generate a continuous function for any of the boundary conditions, But does there exist a Laplacian solution for any set of boundary conditions?
Solution To The Laplacian

- Consider a 1D function
  - What is the solution to:
    \[ \Delta u = 0 \]
  - Any function who’s second derivative is 0
    \[ \partial^2 \phi / \partial x^2 = 0 \]
  - Any function that is linear
    \[ \phi(x) = cx + d \]
- Makes sense conceptually
  - The Laplacian gives us the resting state of diffusion
  - The resting state is a linear function between boundary conditions
1D Laplacian With Dirichlet

Can we always satisfy Dirichlet boundary conditions in 1D?

Yes! A line can always interpolate two points
Can we always satisfy Neumann boundary conditions in 1D?

No! A line can only have one slope
Not always guaranteed that a PDE has a solution for given boundary conditions...
2D Laplacian With Dirichlet

Can we always satisfy Dirichlet boundary conditions in 2D?

Yes! Laplacian is a long-time solution to heat flow
Data is “heat” at boundary. Will eventually diffuse to equilibrium
Can we always satisfy Neumann boundary conditions in 2D?

Neumann BCs prescribe derivative in normal direction: $n \cdot \nabla \phi$

Want to solve for $\Delta \phi = 0$

In 2D, we have the **divergence theorem**:

$$\int_{\partial \Omega} n \cdot \nabla \phi = \int_{\Omega} \nabla \cdot \nabla \phi = \int_{\Omega} \Delta \phi \overset{!}{=} 0$$

integrating $\phi$ over boundary
integrating divergence of $\phi$ over interior

what goes in, must come out!

Can’t have a solution unless the net flux through the boundary is zero!
Numerical libraries will not always tell you that there is a problem with your boundary conditions.
Need to verify yourself. If solving $Ax = b$, verify $\|b - Ax\|$.
## Modeling PDE Equations

<table>
<thead>
<tr>
<th>Level</th>
<th>Equation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easier</td>
<td>Laplace Equation ( \Delta u = 0 )</td>
<td>“What’s the smoothest function interpolating the given boundary data?”</td>
</tr>
<tr>
<td>Intermediate</td>
<td>Heat Equation ( \dot{u} = \Delta u )</td>
<td>“How does an initial distribution of heat spread out over time?”</td>
</tr>
<tr>
<td>Advanced</td>
<td>Wave Equation ( \ddot{u} = \Delta u )</td>
<td>“If you throw a rock into a pond, how does the wavefront evolve over time?”</td>
</tr>
<tr>
<td>Experts Only</td>
<td>Nonlinear + Hyperbolic + High-Order ( \text{??} )</td>
<td>“A lot of real life phenomenon”</td>
</tr>
</tbody>
</table>
Solving The Heat Equation

Heat equation tells us the Laplacian is the first temporal derivative:

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

Compute the Laplacian as normal (Ex: on a grid):

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

Propagate using the first temporal derivative $\Delta u$ (Ex: forward Euler):

$$u^{k+1} = u^k + \Delta u^k$$
Solving The Wave Equation

Wave equation tells us the Laplacian is the second temporal derivative:

\[ \ddot{u} = \Delta u \]

Compute the Laplacian as normal (Ex: on a grid):

\[ u_{i,j}^{k+1} = u_{i,j}^k + \frac{\tau}{\Delta t^2} \left( 4u_{i,j}^k - u_{i+1,j}^k - u_{i-1,j}^k - u_{i,j+1}^k - u_{i,j-1}^k \right) \]

Propagate using the second temporal derivative \( \Delta u \) (Ex: forward Euler):

\[ \dot{u} = v, \quad \ddot{v} = \Delta u \]
Wave Equation On A Triangle Mesh

Wave Equation On Surfaces (2016) Alec Jacobson
Wave Equation On A Triangle Mesh

https://www.adultswim.com/etcetera/elastic-man/
Want To Know More?

Plenty of books and papers on Simulation

What did the folks who wrote these papers/books read?