• Math & Physics
• MAC Grid
• 2x2x1 Example
• Data Structure
• Starter Kit Workflow
• Extras
• Tips
• Math & Physics
• MAC Grid
• 2x2x1 Example
• Data Structure
• Starter Kit Workflow
• Extras
• Tips
• if $\vec{f}$ is vector:
  ◦ Divergence: $\text{div}(\vec{f}) = \nabla \cdot \vec{f}$
  ◦ Curl (3D): $\text{curl}(\vec{f}) = \nabla \times \vec{f}$
Vector differential operator $\nabla$ in 3D Euclidian space

$$\nabla = \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix}$$
Divergence of vector $\vec{f} = (u, v, w)^T$ in 3D Euclidian space

$$\nabla \cdot \vec{f} = \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} \cdot \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}$$
Divergence of vector $\vec{f} = (u, v, w)^T$ in 3D Euclidian space

\[
\nabla \cdot \vec{f} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}
\]
- Divergence of vector \( \vec{f} = (u, v, w)^T \) in 3D Euclidian space

\[
\nabla \cdot \vec{f} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}
\]

- \( \nabla \cdot \vec{f} > 0 \): source
- \( \nabla \cdot \vec{f} < 0 \): sink
- \( \nabla \cdot \vec{f} = 0 \): “divergence free”
**Curl of vector** $\mathbf{\vec{f}} = (u, v, w)^T$ in 3D Euclidean space

\[ \nabla \times \mathbf{\vec{f}} = \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} \times \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} \frac{\partial w}{\partial y} - \frac{\partial v}{\partial z} \\ \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} \\ \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \end{pmatrix} \]
• Example:

\[ \vec{f}(x, y, z) = y \cdot \vec{x} - x \cdot \vec{y} \]

\[ \begin{align*}
u &= y \\
v &= -x \\
w &= 0
\end{align*} \]

[Figure from http://en.wikipedia.org/wiki/Curl_(mathematics)]
Example:
\[ \vec{f}(x, y, z) = y \cdot \hat{x} - x \cdot \hat{y} \]
\[ \nabla \times \vec{f} = \begin{pmatrix} \frac{\partial 0}{\partial y} - \frac{\partial y}{\partial z} \\ \frac{\partial (-x)}{\partial z} - \frac{\partial 0}{\partial x} \\ \frac{\partial (-x)}{\partial x} - \frac{\partial y}{\partial y} \end{pmatrix} \]
\[ \nabla \times \vec{f} = (0, 0, -2)^T \]
Summary:

- **Divergence:** \( \text{div}(\vec{f}) = \nabla \cdot \vec{f} \)
  - Measures the magnitude of a vector field’s source or sink at a given point.

- **Curl:** \( \text{curl}(\vec{f}) = \nabla \times \vec{f} \)
  - Measure the rotation of a 3d vector field
Fluid equation:

\[
\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u}
\]

\[
\nabla \cdot \vec{u} = 0
\]
Physics: Navier-Stokes Eq

- Momentum equation: \( f = ma \)
  - \( \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p = \mathbf{g} + \nu \nabla \cdot \nabla \mathbf{u} \)
  - Rearrange:
    - \( \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \mathbf{g} + \left( -\frac{1}{\rho} \nabla p \right) + \nu \nabla \cdot \nabla \mathbf{u} \)

Material Derivative
External Force
Pressure Force
Viscosity Force
• Constraint equation: divergence free
  \[ \nabla \cdot \vec{u} = 0 \]

- Divergence of vector \( \vec{f} = (u, v, w)^T \) in 3D Euclidian space
- \( \nabla \cdot \vec{f} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \)
  - \( \nabla \cdot \vec{f} > 0 \): source
  - \( \nabla \cdot \vec{f} < 0 \): sink
  - \( \nabla \cdot \vec{f} = 0 \): "divergence free"
Fluid equation:

\[ \frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u} \]

\[ \nabla \cdot \vec{u} = 0 \]
Solve PDE by splitting:

- Toy example:
  - \(\frac{\partial q}{\partial t} + 1 + 2 = 3 + 4\)
  - Given \(q^n\), how do we find \(q^{n+1}\)?
  - Step 1: rearrange \(\frac{\partial q}{\partial t} = -1 - 2 + 3 + 4\)
  - Step 2: split and solve
    \[
    \begin{align*}
    q^{(1)} &= q^n + dt \times (-1) \\
    q^{(2)} &= q^{(1)} + dt \times (-2) \\
    q^{(3)} &= q^{(2)} + dt \times (3) \\
    q^{n+1} &= q^{(3)} + dt \times (4)
    \end{align*}
    \]
  - Step 3: get result \(q^{n+1} = q^n + 4dt\)
• Solve PDE by splitting:
  ◦ N-S equation
    • \[ \frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u}, \nabla \cdot \vec{u} = 0 \]
    • Given \( \vec{u}^n \), how do we find \( \vec{u}^{n+1} \)?
  ◦ Step 1, rearrange:
    \[ \frac{\partial \vec{u}}{\partial t} = -\vec{u} \cdot \nabla \vec{u} + \vec{g} + \nu \nabla \cdot \nabla \vec{u} - \frac{1}{\rho} \nabla p \]
Solve ODE by splitting:

- N-S equation
  \[
  \frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u}, \quad \nabla \cdot \vec{u} = 0
  \]
- Given \( \vec{u}^n \), how do we find \( \vec{u}^{n+1} \)?
- Step 2, split:
  \[
  \begin{cases}
    \text{solve } \frac{\partial \vec{u}}{\partial t} = -\vec{u} \cdot \nabla \vec{u}, \text{ using } \vec{u}^n, \text{ save the result to } \vec{u}^{(1)} \\
    \text{solve } \frac{\partial \vec{u}}{\partial t} = \vec{g}, \text{ using } \vec{u}^{(1)}, \text{ save the result to } \vec{u}^{(2)} \\
    \text{solve } \frac{\partial \vec{u}}{\partial t} = \nu \nabla \cdot \nabla \vec{u}, \text{ using } \vec{u}^{(2)}, \text{ save the result to } \vec{u}^{(3)} \\
    \text{solve } \frac{\partial \vec{u}}{\partial t} = -\frac{1}{\rho} \nabla p, \text{ using } \vec{u}^{(3)}, \text{ save the result to } \vec{u}^{n+1} \\
    \text{keep } \nabla \cdot \vec{u}^{n+1} = 0
  \end{cases}
  \]
Overview

- Math & Physics
- MAC Grid
- 2x2x1 Example
- Data Structure
- Starter Kit Workflow
- Extras
- Tips
Marker and Cell Grid / Staggered Grid
- Detail described in Bridson’s notes, page 15
- If I say any thing about page number in today’s class, I’m referring the Bridson’s notes.
2D case, center quantities:
2D case, staggered quantities:
• 2D case, staggered quantities: (Programming)
3D case:

- $P_{i,j,k}$
- $U_{i-1/2,j,k}$
- $W_{i,j,k-1/2}$
- $V_{i,j-1/2,k}$

Diagram showing a 3D grid with vectors $P_{i,j,k}$, $U_{i-1/2,j,k}$, $W_{i,j,k-1/2}$, and $V_{i,j-1/2,k}$.
3D case: (Programming)
All the elements inside are **SCALAR**.
- Pressure, temperature, density... etc
- Also velocity: U, V, W.

We solve the bad ½ cases by **ADDING** another ½.
- \( U_{i-1/2, j, k} \rightarrow U_{i, j, k} \)
- \( U_{i+1/2, j, k} \rightarrow U_{i+1, j, k} \)
• Simple case of interpolation
  ◦ What’s the velocity at the red point?
  ◦ \( \text{Vec3}(U_{i,j,k}, \frac{1}{4}*(V_{i,j,k} + V_{i-1,j,k} + V_{i,j+1,k} + V_{i-1,j+1,k}), \frac{1}{4}*(W_{i,j,k} + W_{i-1,j,k} + W_{i,j,k+1} + W_{i-1,j,k+1})) \)
• Math & Physics
• MAC Grid
• 2x2x1 Example
• Data Structure
• Starter Kit Workflow
• Extras
• Tips
Example of simple simulation

- Dropped the external and viscosity force

\[
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p = \mathbf{g} + \nu \nabla \cdot \nabla \mathbf{u}
\]

\[
\nabla \cdot \mathbf{u} = 0
\]
A good start...

2x2x1 Example
How many y-velocities (v)?
What are the indices?

1, 2, 0
0, 2, 0
1, 1, 0
0, 1, 0
1, 0, 0
0, 0, 0
• Ditto for x and z
  ◦ x velocities: $u$
    (2, 1, 0) (1, 1, 0) (0, 1, 0)
    (1, 0, 0) (1, 0, 0) (0, 0, 0)
  ◦ z velocities: $w$
    • front
      (1, 1, 0) (0, 1, 0)
      (0, 1, 0) (0, 0, 0)
    • back
      (1, 1, 1) (0, 1, 1)
      (0, 1, 1) (0, 0, 1)
• Source

\[ V(0, 1, 0) = 1.0 \]
• After setting up the source
  ◦ What is the current u?
  ◦ All zero
  ◦ What is the current v?
    ◦ 1 at v(0, 1, 0), 0 at the rest of them
  ◦ What is the current w?
    ◦ All zero
N-S Equation

\[ \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p = 0 \]

\[ \nabla \cdot \mathbf{u} = 0 \]
Advection Part Split from N-S Equation

\[ \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = 0 \]
Change a view (Page 5)

- Lagrangian: treat fluid as particles, $q$ can be any quantity carried by one particle (temperature, density, or velocity).

$$q(t, \vec{x})$$

- In the advection step, we assume that there's no outside force affecting the particle. (Traveling with the velocity field)

$$\frac{\partial}{\partial t} q(t, \vec{x}) = 0$$

2x2x1 Example
Change a view (Cont’d)

\[
\frac{\partial}{\partial t} q(t, \vec{x}) = 0
\]

\[
= \frac{\partial q}{\partial t} + \nabla q \cdot \frac{d\vec{x}}{dt}
\]

\[
= \frac{\partial q}{\partial t} + \nabla q \cdot \vec{u}
\]
Advect Velocity

\[ \frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = 0 \]

\[ \vec{u} = (u, v, w) \]

\[ \frac{\partial \vec{v}}{\partial t} + \vec{u} \cdot \nabla \vec{v} = 0 \]

\[ \frac{\partial \vec{w}}{\partial t} + \vec{u} \cdot \nabla \vec{w} = 0 \]
• Advect Velocity (Trace back)
  ◦ Assume $dt = 0.1$, $dx = dy = dz = 1.0$

\[
\begin{align*}
\text{vel}(0, 1, 0) &= (0.0, 1.0, 0.0) \\
V(0, 1, 0) &= 1.0 \\
\text{vel}(0, 0.9, 0) &= (0.0, 0.9, 0.0)
\end{align*}
\]
Advect Velocity (Update)
- Assume $dt = 0.1$, $dx = dy = dz = 1.0$

$V(0, 1, 0) = 0.9$
N-S Equation

\[
\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = 0
\]

\[
\nabla \cdot \vec{u} = 0
\]
Projection Part Split from N-S Equation

\[ \frac{\partial \vec{u}}{\partial t} + \frac{1}{\rho} \nabla \rho = 0 \]

\[ \nabla \cdot \vec{u} = 0 \]
Projection

\[ \vec{u}^{n+1} = \vec{u}^n - \Delta t \frac{1}{\rho} \nabla p \]

\[ \nabla \cdot \vec{u}^{n+1} = 0 \]
**Projection**

\[
\nabla \cdot \mathbf{u}^{n+1} = \frac{u_{i+1/2,j,k}^{n+1} - u_{i-1/2,j,k}^{n+1}}{\Delta x} + \frac{v_{i,j+1/2,k}^{n+1} - v_{i,j-1/2,k}^{n+1}}{\Delta x} + \frac{w_{i,j,k+1/2}^{n+1} - w_{i,j,k-1/2}^{n+1}}{\Delta x} = 0
\]

\[
\frac{1}{\Delta x} \left[ \begin{array}{c}
\left( u_{i+1/2,j,k} - \frac{\Delta t}{\rho} \frac{p_{i+1,j,k} - p_{i,j,k}}{\Delta x} \right) - \left( u_{i-1/2,j,k} - \frac{\Delta t}{\rho} \frac{p_{i,j,k} - p_{i-1,j,k}}{\Delta x} \right) \\
+ \left( v_{i,j+1/2,k} - \frac{\Delta t}{\rho} \frac{p_{i,j+1,k} - p_{i,j,k}}{\Delta x} \right) - \left( v_{i,j-1/2,k} - \frac{\Delta t}{\rho} \frac{p_{i,j,k} - p_{i,j-1,k}}{\Delta x} \right) \\
+ \left( w_{i,j,k+1/2} - \frac{\Delta t}{\rho} \frac{p_{i,j,k+1} - p_{i,j,k}}{\Delta x} \right) - \left( w_{i,j,k-1/2} - \frac{\Delta t}{\rho} \frac{p_{i,j,k} - p_{i,j,k-1}}{\Delta x} \right) \end{array} \right] = 0
\]
• Projection (Page 29-30, general case)

\[
\frac{\Delta t}{\rho} \left( \frac{6p_{i,j,k} - p_{i+1,j,k} - p_{i,j+1,k} - p_{i,j,k+1} - p_{i-1,j,k} - p_{i,j-1,k} - p_{i,j,k-1}}{\Delta x^2} \right)
\]

\[
= -\left( \frac{u_{i+1/2,j,k} - u_{i-1/2,j,k}}{\Delta x} + \frac{v_{i,j+1/2,k} - v_{i,j-1/2,k}}{\Delta x} + \frac{w_{i,j,k+1/2} - w_{i,j,k-1/2}}{\Delta x} \right)
\]
Projection (Solid Wall boundary on (i+1,j,k))

\[
\frac{\Delta t}{\rho} \left( 5 \left( p_{i,j,k} - p_{i+1,j,k} - p_{i,j,k+1} - p_{i,j-1,k} - p_{i,j,k-1} \right) \right)
= -\left( \frac{u_{i+1/2,j,k} - u_{i-1/2,j,k}}{\Delta x} + \frac{v_{i,j+1/2,k} - v_{i,j-1/2,k}}{\Delta x} + \frac{w_{i,j,k+1/2} - w_{i,j,k-1/2}}{\Delta x} \right)
\]
2x2x1 Example

Projection (Free Surface boundary on (i+1,j,k))

\[
\frac{\Delta t}{\rho} \left( 6p_{i,j,k} - p_{i+1,j,k} - p_{i,j+1,k} - p_{i,j,k+1} - p_{i-1,j,k} - p_{i,j-1,k} - p_{i,j,k-1} \right) \Delta x^2 = 0
\]

\[
= -\left( \frac{u_{i+1/2,j,k} - u_{i-1/2,j,k}}{\Delta x} + \frac{v_{i,j+1/2,k} - v_{i,j-1/2,k}}{\Delta x} + \frac{w_{i,j,k+1/2} - w_{i,j,k-1/2}}{\Delta x} \right)
\]
Projection (Matrix Equation)

\[
\begin{pmatrix}
6 & -1 & \ldots & 0 \\
-1 & 5 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & 6
\end{pmatrix}
\begin{pmatrix}
p_{0,0,0} \\
p_{1,0,0} \\
\vdots \\
p_{I,J,K}
\end{pmatrix}
= - \frac{(\Delta x)^2 \ast \rho}{dt}
\begin{pmatrix}
\nabla \cdot \vec{u}_{0,0,0} \\
\nabla \cdot \vec{u}_{1,0,0} \\
\vdots \\
\nabla \cdot \vec{u}_{I,J,K}
\end{pmatrix}
\]

\[Ap = d\]
• Projection
  ◦ What is dimension of the A matrix?
  ◦ What does the A matrix look like?

\[
\begin{pmatrix}
2 & -1 & -1 & 0 \\
-1 & 2 & 0 & -1 \\
-1 & 0 & 2 & -1 \\
0 & -1 & -1 & 2
\end{pmatrix}
\]
**Projection**
- What is the divergence column?

\[
\begin{pmatrix}
0.9 \\
0 \\
-0.9 \\
0
\end{pmatrix}
\]
• Projection
  ◦ Multiply by the constant...

\[
- \frac{(\Delta x)^2 \cdot \rho}{dt} = -10
\]

Here I use \( dx = 1 \) and \( dt = 0.1 \)
But in your framework, they are different
• Projection
  ◦ Solve for $p$

\[
\begin{pmatrix}
2 & -1 & -1 & 0 \\
-1 & 2 & 0 & -1 \\
-1 & 0 & 2 & -1 \\
0 & -1 & -1 & 2 \\
\end{pmatrix}
\begin{pmatrix}
p_{0,0,0} \\
p_{1,0,0} \\
p_{0,1,0} \\
p_{1,1,0} \\
\end{pmatrix}
=
\begin{pmatrix}
-9 \\
0 \\
9 \\
0 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
p_{0,0,0} \\
p_{1,0,0} \\
p_{0,1,0} \\
p_{1,1,0} \\
\end{pmatrix}
=
\begin{pmatrix}
-3.375 \\
-1.125 \\
3.375 \\
1.125 \\
\end{pmatrix}
\]
Projection

Update velocity $u^{n+1}$:

$$
\vec{u}^{n+1} = \vec{u}^n - \Delta t \frac{1}{\rho} \nabla p
$$

$$
u_{1,0,0}^{n+1} = u_{1,0,0}^n - \Delta t \frac{1}{\rho} \frac{p_{1,0,0} - p_{0,0,0}}{\Delta x}
$$

$$= 0 - 0.1 \frac{1}{1} \frac{-1.125 - (-3.375)}{\Delta x}
$$

$$= -0.225
$$

$$u_{1,1,0}^{n+1} = 0.225
$$

$$v_{0,1,0}^{n+1} = 0.225
$$

$$v_{1,1,0}^{n+1} = -0.225
$$
- Math & Physics
- MAC Grid
- 2x2x1 Example
- **Data Structure**
- Starter Kit Workflow
- Extras
- Tips
• Classes
  ◦ GridData
  ◦ GridDataMatrix
  ◦ MACGrid

• Useful Constants and Macros
- **GridData**
  - Key member:
    - `std::vector<double> mData;`
  - Key methods:
    - `virtual double interpolate(const vec3& pt)`
    - Indexing with “()” notation.
    - Arithmetic operations defined in MACGrid.h
  - Subclasses:
    - GridDataX
    - GridDataY
    - GridDataZ
• GridDataMatrix
  ◦ Key members:
    • GridData diag;
    • GridData plusI;
    • GridData plusJ;
    • GridData plusK;
  ◦ Key methods:
    • Arithmetic operations defined in MACGrid.h
MACGrid

- **Key Members:**
  - GridDataX mU;
  - GridDataY mV;
  - GridDataZ mW;
  - GridData mP
  - GridData mD;
  - GridData mT;
  - GridDataMatrix AMatrix;

- **Key Methods:**
  - Main work flow functions.
  - Drawing functions.
• **Useful Constants**
  ◦ Mostly defined in constants.cpp
    • `const int theDim[3]`
  ◦ But `dt` defined in `SmokeSim::step()`

• **Useful Macros**
  ◦ Mostly defined in `mac_grid.cpp`
  ◦ `FOR_EACH_CELL`
  ◦ `FOR_EACH_FACE`
  ◦ Recommended to write your own:
    • `FOR_EACH_FACE_X`
    • `FOR_EACH_FACE_Y` ... etc.
• Math & Physics
• MAC Grid
• 2x2x1 Example
• Data Structure
• Starter Kit Workflow
• Extras
• Tips
Main simulation loop

- Update Sources
- Advect Velocity
- Add External Forces
- Projection
- Advect Density and Temperature
• Update Sources
  ◦ Set temperature/density: Think it as a filling some cubic spaces will smoke
  ◦ Set velocity: Think it as a fan blowing at some faces.
  ◦ Note: do not set invalid velocity on boundaries.
Advevt Velocity
  ◦ Save the current velocity as a copy. (target)
  ◦ Trace back and update the target velocity.
  ◦ Copy it back to the current velocity.

Quiz:
  ◦ Why are we using a copy?
    ◦ We are using current velocity field to change velocities
  ◦ How many velocities do I need to advect in the 2x2x1 case?
    ◦ 6+6+8=20
• External Forces
  ◦ Get a copy (target)
  ◦ Calculate forces and update the copy (Explicit Euler integration)
  ◦ Copy it back
• **External Forces**
  ◦ **Kind of forces:**
    - Buoyancy *(Typo in our code...)*
    - Vorticity Confinement Forces
    - Gravity
    - Viscosity
    - Surface Tension

---

**Smoke Sim**

**Water Sim**
External Forces
  ◦ Buoyancy+Gravity: Page 45
    • Only acts on vertical velocities.

\[ f_{\text{buoy}} = (0, -\alpha s + \beta (T - T_{\text{amb}}), 0) \]
• External Forces
  ◦ Vorticity Confinement Force: Page 46,47
    • Goal: enhance the vortices
    • vorticity measurement: curl
    \[
    \vec{\omega} = \nabla \times \vec{u}
    \]
    • Find the local maximum (vortex center):
    \[
    \vec{N} = \frac{\nabla |\vec{\omega}|}{\|\nabla |\vec{\omega}|\|}
    \]
    • Add force:
    \[
    f_{\text{conf}} = \epsilon \Delta x (\vec{N} \times \vec{\omega})
    \]
• **Projection**
  - **Build A Matrix.**
    - `setUpAMatrix()` defined in MACGrid.
    - Need to modify `setUpAMatrix()` if you are putting other solids into the scene.
  - **Build d vector.**
  - Solve for p from $Ap = d$.
    - (call `conjugateGradient()`) 
  - **Update velocity.**
  - **Sanity check. (Divergence Free).**

\[
\begin{bmatrix}
6 & -1 & \cdots & 0 \\
-1 & 5 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 6
\end{bmatrix}
\begin{bmatrix}
 p_{0,0,0} \\
p_{1,0,0} \\
\vdots \\
p_{I,J,K}
\end{bmatrix}
= - \frac{(dx)^2}{dt \cdot \rho}
\begin{bmatrix}
\nabla \cdot \vec{u}_{0,0,0} \\
\nabla \cdot \vec{u}_{1,0,0} \\
\vdots \\
\nabla \cdot \vec{u}_{I,J,K}
\end{bmatrix}
\]
• Advect Other Quantities:
  ◦ Temperature
    • For buoyancy
  ◦ Density
    • For gravity / For display
Overview

- Math & Physics
- MAC Grid
- 2x2x1 Example
- Data Structure
- Starter Kit Workflow
- Extras
- Tips
• Sharper interpolation
  ◦ Hermite Cubic Interpolation: Described in page 47.
• Add Objects:
  ◦ Interactive objects. (Cell Sized, like Tetris)
  ◦ Arbitrary objects. (Deal with half fill cases, curved boundaries. (Page 38))
• Fluid Viscosity:
  ◦ Adding viscosity to external force.
  ◦ Document it and compare the difference.
Preconditioner for faster conjugate gradient solver.
- Described in page 32 with conjugate gradient solver.
- Document which preconditioner you use.
- Compare the speed before/after you apply your preconditioner.

Extras (make it faster)
• Port parts of your solve to the GPU using CUDA. (cg_solver is probably the best part to accelerate!)

OR

• Make a totally different solver.
  ◦ Sparse Matrix Solvers on the GPU: Conjugate Gradients and Multigrid
  ◦ A parallel multigrid Poisson solver for fluids simulation on large grids

Extras (make it faster)
• Port the data out.
  ◦ Inside SmokeSim::grabScreen() Function, you can uncomment lines of code to port your data out.
• Use volumetric renderer you made in 560.
• Use Maya.
• Etc.

[Peter Kutz, 2011]
• Math & Physics
• MAC Grid
• 2x2x1 Example
• Data Structure
• Starter Kit Workflow
• Extras
• Tips
• How to start this assignment
  ◦ Find the place that you need to fill by searching “TODO”
  ◦ Start with a 2x2x1 sample, because you can verify everything by hand!

  ◦ Start with
    • void MACGrid::updateSources()
    • void MACGrid::advectVelocity(double dt)
    • void MACGrid::project(double dt)
• Save some time for simulation and rendering
  ◦ For instance, under a 50x50x20 setting, simulating one frame may cost you 2 minutes, rendering it using naive volumetric renderer may cost you 10 minutes...
• Typical Length: less than 600 frames is fine.
  ◦ If your smoke fills the entire space, you can hardly see anything, even if it is flowing.
  ◦ Rendering costs more time than simulation.
• If you still have late days, use it...
  ◦ Late days can not be used on your final project
Acknowledgement

- Thanks to Aline Normoyle, Peter Kutz and Harmony Li for setting up the starter code.
- The 2x2x1 example is taken from Aline’s recitation in 2011.
Good Luck 😊