Computational Fluid Dynamics: A crash course and minimal implementation

Cilliers Pretorius

August 2, 2023

Overview

- Differences between equations of motion for solids and fluids
- Navier-Stokes for incompressible flow
- Implementing it in practice
- A quick look at a spherical variant
- Examining an audience member's shape

Equations of motion

- Equations used to describe the behaviour of a physical system in terms of movement as a function of time.
- Classically, they derive from Newton's Second Law ($F = m\vec{a}$).
- For a particle in linear acceleration, its position $\vec{x}$ at time $t + 1$ is calculated as follows:

$$\vec{x}_{t+1} = \vec{x}_t + \vec{v}_t \Delta t + \frac{1}{2}\vec{a}\Delta t^2 \tag{1}$$

- This also works well for larger, rigid objects.
- But fluids are specifically not rigid... what now?

Modelling fluid systems

- Can model all the molecules in a system, using equation 1.
- But then you have to do it for all molecules in your system.
- 6.02×10^{23} particles for each 18 grams of water, or 30.72 grams of air.
- The biggest particle simulation ever contained "only" 6×10^{11} particles and could do "only" 0.5 simulation ticks per second, utilising the full potential of the biggest supercomputer in the world (in 2021).
- Clearly, that won't work.

History of fluid equations

- Initial work came from Daniel Bernoulli and Leonard Euler in the 1740s.
- Euler equations (one of first partial differential equations) were published in 1757.
- However, Euler equations do not take any viscosity into account.
- Euler equations gets you most of the behaviour, but it is still not quite right.

Messrs. Navier and Stokes

- Over a few decades of work in the first half of the 19th century, mainly by Claude-Louis Navier and Sir George Stokes, the more accurate Navier-Stokes equations were formulated.
- They correctly model viscosity and are usable for both compressible and incompressible flow.
- Despite the centuries of work both theoretical and applied on these equations, they still arene't fully understood mathematically.
- The *Navier-Stokes existence and smoothness* problem is one of the Clay Mathematics Institute's Millenium Prize Problems.
- In three dimensions, does there exist a smooth (i.e., infinitely differentiable) and globally defined solution for all possible initial velocity fields?
- It has been proven true for two dimensions, but not three.

Navier-Stokes for incompressible flow

Fluid equations for any incompressible fluid:

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u} \tag{2}$$

$$\nabla \cdot \vec{u} = 0 \tag{3}$$

The equations are derived from the principles of conservation of mass, momentum, and energy. It is left to the reader to look up the precise derivations that lead to the above.

Definitions

- $\vec{u}$ is the velocity vector field.
- $\nabla \vec{u}$ is the gradient of the velocity.
- ρ is the density of the fluid.
- ∇p is the gradient of the pressure in the fluid.
- $\vec{g}$ is the sum of the body forces acting on the fluid.
- ν is the viscosity of the fluid.
- $\nabla \cdot \vec{u}$ is the divergence of the velocity, i.e., the change in volume.

Navier-Stokes for incompressible flow

Fluid equations for any incompressible fluid:

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

$$\nabla \cdot \vec{u} = 0 \tag{4}$$

The equations are derived from the principles of conservation of mass, momentum, and energy. It is left to the reader to look up the precise derivations that lead to the above.

Simplifications and considerations for a computer implementation

- What is the level of detail that we need? What resolution do we operate on, what is our total area?
- Do the Navier-Stokes equations care about resolution? Not really.
- Do we need an analytical solution, or is approximate good enough?
- How do we model the fluid in our program?
- Do we need a solution at every point of our system, or can we discretize?

Fluid equations in our case

- We discard body forces (instead applying them directly to the velocity field) and viscosity.
- It turns out that the errors from discretizing the equations happen to mimic viscosity quite nicely for most fluids.
- Thus, our fluid equations are:

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = 0 \tag{5}$$

$$\nabla \cdot \vec{u} = 0 \tag{6}$$

Eulerian vs Lagrangian methods

- In computational simulations, two broad classes of methods exist.
- *Eulerian* methods discretize the simulation space into a mesh and update each cell at each timestep.
- *Lagrangian* methods discretize the material/fluid being simulated, updating each particle at each timestep.
- Eulerian methods are coarser and more complicated to implement, but have less variance in resolution across the system and is faster.
- Lagrangian methods are accurate and easier to implement, but are slower for equivalent accuracy across a large space.

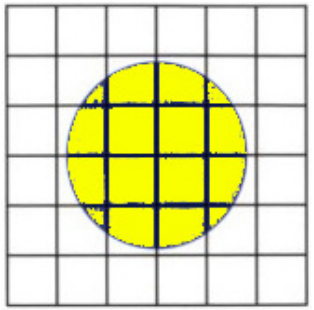


Figure: An Eulerian discretization.

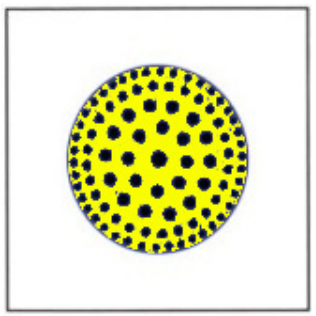


Figure: A Lagrangian discretization.

Marker-and-Cell Grid

- The MAC grid has scalar quantities at centre of cell, and velocity components on the cell edges.
- This is beneficial for pressure projection later.
- Velocity at centre of the cell is simply the bilinear interpolation of the components on the cell edges.

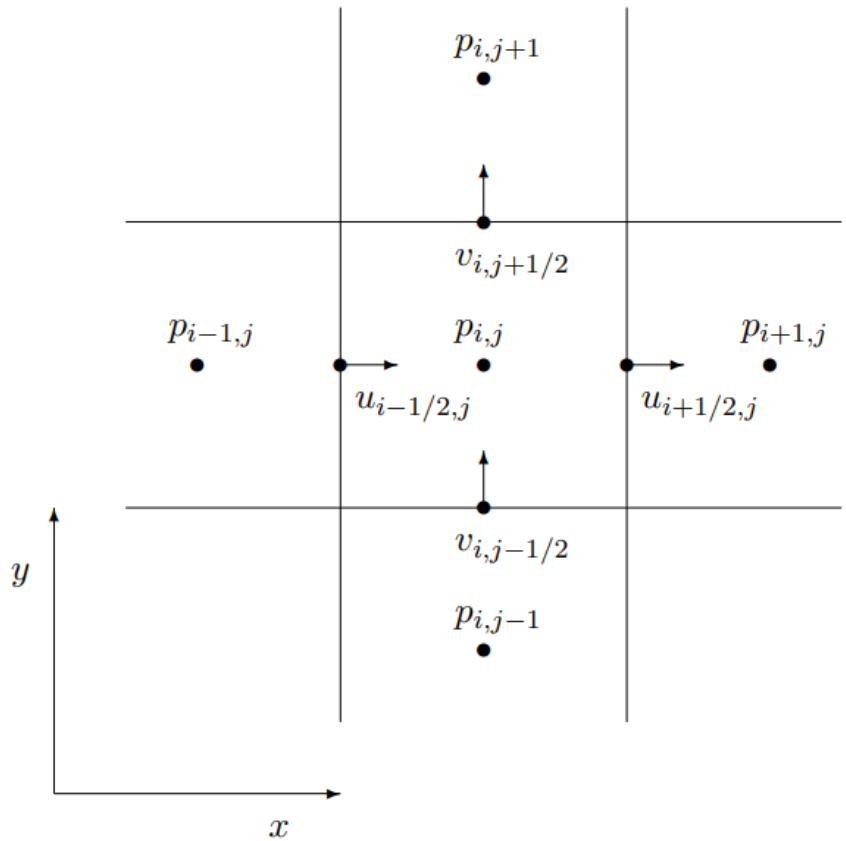


Figure: A sample cell i,j in the MAC grid discretization.

Semi-Lagrangian Advection

- Traditional Eulerian solutions to the advection equation ($\frac{Dq}{Dt} = 0$) are unconditionally unstable.
- In comparison, traditional particle-based (Lagrangian) advection is utterly trivial.
- For each grid point $Q_{i,j}$, semi-Lagrangian advection creates a virtual particle P such that the particle is at exactly $Q_{i,j}$ at time $t + \Delta t$.
- We trace the virtual particle back to time t and sample the advected values at that 'old' position with bilinear interpolation.
- Update $Q_{i,j}$ with the values sampled at the previous position.
- This method is **unconditionally stable** regardless of Δt, though numerical inaccuracies tend to limit it to $\Delta t \leq \frac{5\Delta x}{\vec{u}_{max}}$.

Semi-Lagrangian Advection

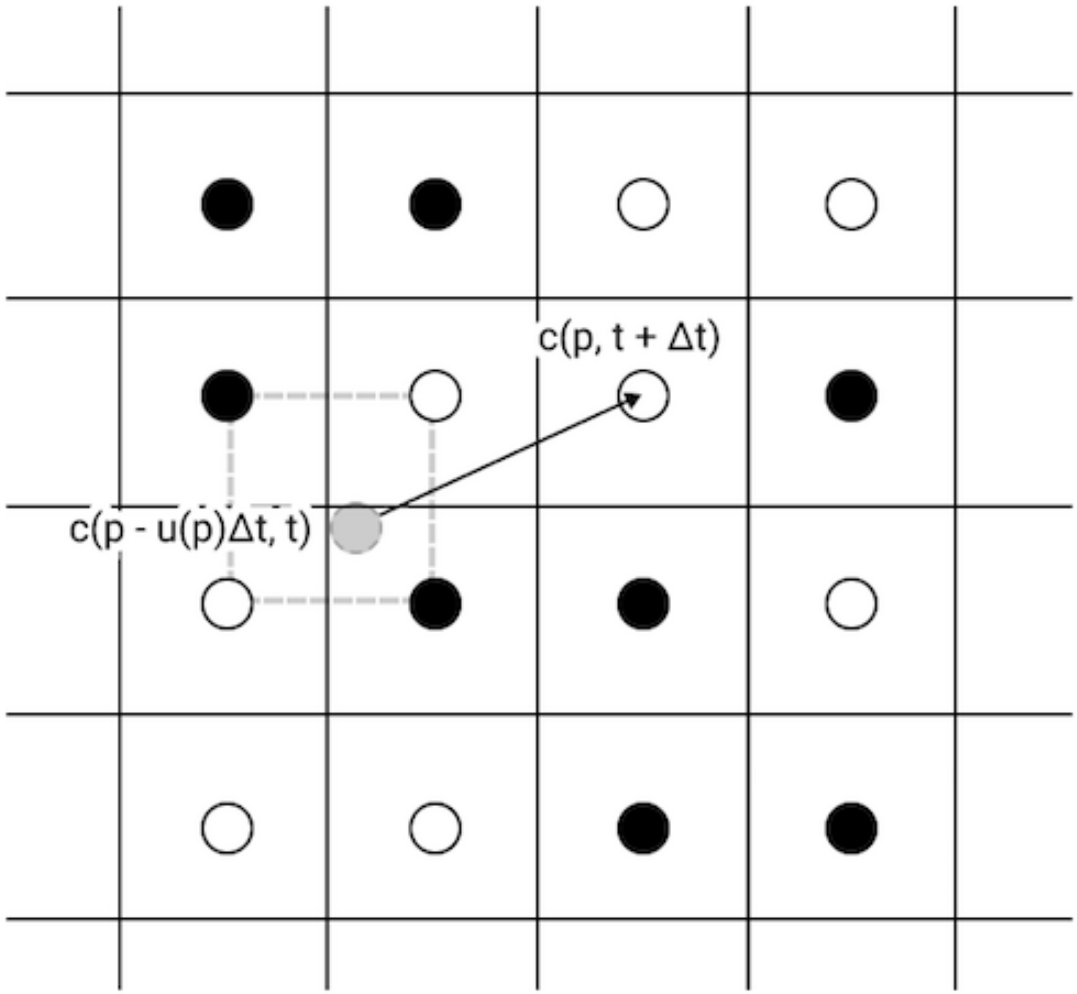


Figure: Trace the virtual particle back to time t to update $Q_{i,j}$ at time $t + \Delta t$.

Solving divergence

- Discretizing the Navier-Stokes equations violates the second equation, necessitating a correction step.
- Divergence d at a point (i, j) is:

$$d_{i,j} = \frac{(u_x(i, j + \frac{1}{2}) - u_x(i, j - \frac{1}{2}))}{\Delta x} + \frac{u_y(i + \frac{1}{2}, j) - u_y(i - \frac{1}{2}, j)}{\Delta y} \tag{7}$$

- Solving for pressure (substituting equations 9 and 10 into 7) gives a linear system of the form:

$$\frac{\Delta t}{\rho} \frac{4p_{i,j} - p_{i+1,j} - p_{i-1,j} - p_{i,j+1} - p_{i,j-1}}{\Delta x \Delta y} = -d_{i,j} \tag{8}$$

Solving linear system – 2D Poisson problem

- This is a sparse linear system of the form $Ap = d$, where A is a $N \times N$ symmetric (positive definite) matrix, and p and d are vectors of size N, where N is the total amount of grid points.
- We solve for p using a linear solver of our choice, conjugate gradient generally being the best compromise between accuracy and speed.
- For each velocity grid point (i, j), update $\vec{u}_{i,j}$ using central differences of the pressure gradient.

$$u^*_{i+1/2,j} = u_{i+1/2,j} - \Delta t \frac{1}{\rho} \frac{p_{i+1,j} - p_{i,j}}{\Delta x} \tag{9}$$

$$v^*_{i,j+1/2} = v_{i,j+1/2} - \Delta t \frac{1}{\rho} \frac{p_{i,j+1} - p_{i,j}}{\Delta y} \tag{10}$$

Boundary Conditions

- Solid boundaries add an additional constraint to the equations in the form $\vec{u}^{n+1} \cdot \hat{n} = \vec{u}_{solid} \cdot \hat{n}$.
- For any cell (i,j), any solid neighbours mean that velocity components on the solid boundary will be replaced by u_{solid}.
- The coefficient for $p_{i,j}$ in our pressure equation is the number of non-solid neighbours, while solid neighbours' coefficients are 0.
- For example, the pressure in a cell with its bottom and left neighbours solid becomes the following:

$$\frac{\Delta t}{\rho} \frac{2p_{i,j} - p_{i-1,j} - p_{i,j+1}}{\Delta x^2} = -\left(\frac{u_{solid} - u_\theta(i,j)}{\Delta x} + \frac{u_\phi(i,j+1) - u_{solid}}{\Delta x}\right) \tag{11}$$

Grid on a sphere

- Traditional grids on a sphere cause severe discontinuities and artefacts at the poles.
- Instead of x and y, we use θ and ϕ on a rectangular grid of $(\pi, 2\pi)$.
- Cells are consistent size in θ direction, but gets smaller towards the poles in the ϕ direction, causing inaccuracies.
- The smaller cells require additional terms in the advection equations to compensate.

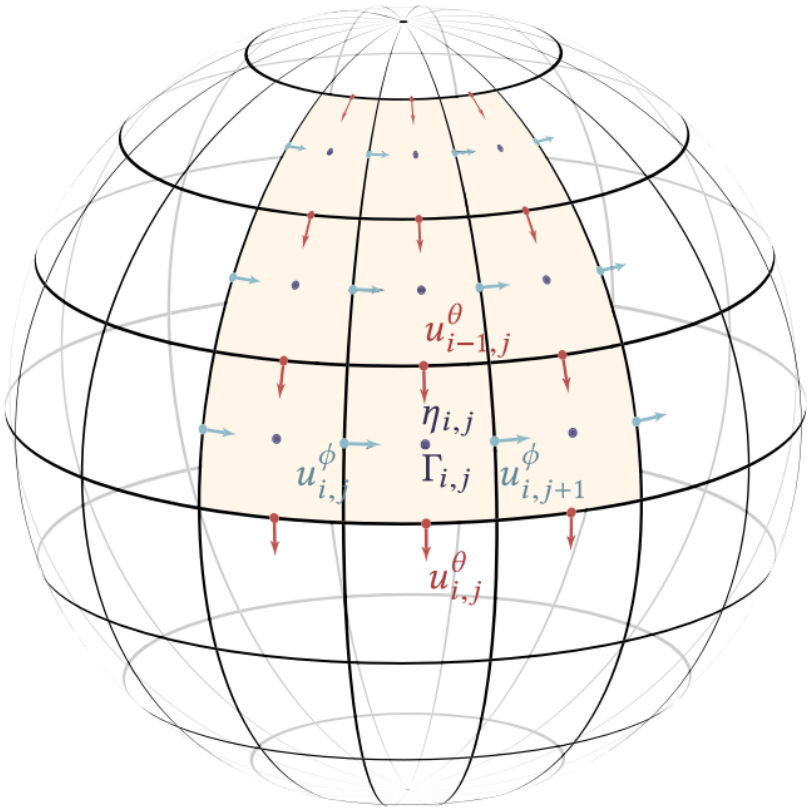


Figure: The staggered spherical grid with directions θ and ϕ.

Spherical case

- The equations for pressure solves on the sphere requires additional factors taking geometry into account.
- The spherical equivalents of the pressure projection equations (equations 9 and 10) are:

$$u_{\theta E}^{n+1} = u_{\theta E}^{*} - \frac{\Delta t}{\rho R}\left(p_{CB} - p_{CT}\right) \tag{12}$$

$$u_{\phi E}^{n+1} = u_{\phi E}^{*} - \frac{\Delta t}{\rho R \sin\theta_C}\left(p_{CR} - p_{CL}\right) \tag{13}$$

- u^{n+1} is the velocity at the next step, while u^* is the velocity after advection (before correction).

Divergence on the sphere

- The divergence for a cell C on the surface of the sphere is:

$$(\nabla \cdot \vec{u})_C = \frac{1}{R \sin\theta_C}\left(\frac{\sin\theta_{EB}\, u_{EB} - \sin\theta_{ET}\, u_{ET}}{\Delta\theta} + \frac{v_R - v_L}{\Delta\phi}\right) \quad (14)$$

- If we then substitute equations 12 and 13 into 14, we get the following:

$$0 = \frac{1}{R \sin\theta_C}[\frac{\sin\theta_{EB}\left(u^*_{\theta EB} - \frac{\Delta t}{\rho R}\left(p_{CB} - p_C\right)\right) - \sin\theta_{ET}\left(u^*_{\theta ET} - \frac{\Delta t}{\rho R}\left(p_C - p_{CT}\right)\right)}{\Delta\theta} + \frac{\left(u^*_{\phi ER} - \frac{\Delta t}{\rho R \sin\theta_C}\left(p_{CR} - p_C\right)\right) - \left(u^*_{\phi EL} - \frac{\Delta t}{\rho R \sin\theta_C}\left(p_C - p_{CL}\right)\right)}{\Delta\phi}] \quad (15)$$

Poisson problem on the sphere

- To create a problem of the form $Ap = d$, where A is a symmetric positive definite matrix of size $N \times N$, p and d are vectors of size N, we rearrange equation 15 to be the following:

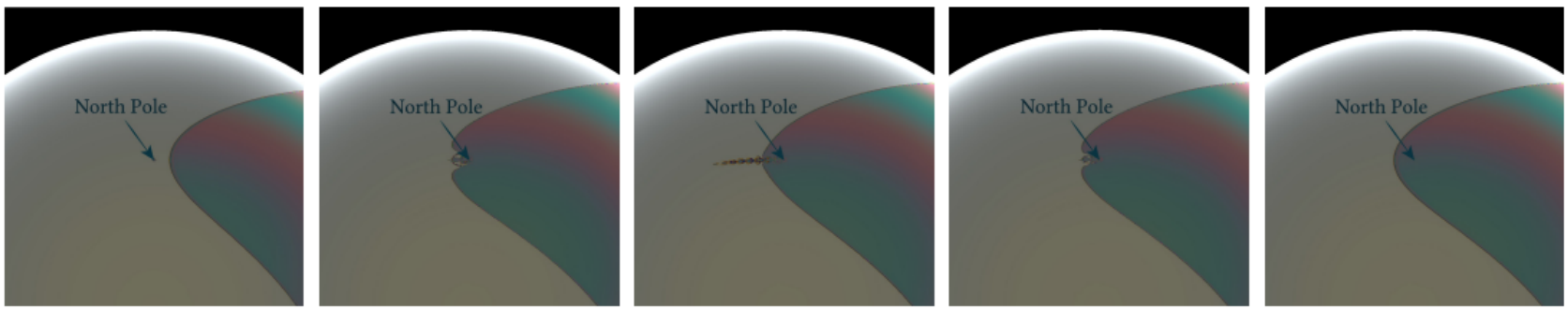
$$\frac{\Delta t}{\rho R}\Bigg(P_C\left(\frac{\sin\theta_{EB} + \sin\theta_{ET}}{\Delta\theta} + \frac{2}{\sin\theta_C \Delta\phi}\right) - \frac{P_{CR}}{\sin\theta_C \Delta\phi} - \frac{P_{CL}}{\sin\theta_C \Delta\phi} - \frac{\sin\theta_{EB} P_{CB}}{\Delta\theta} - \frac{\sin\theta_{ET} P_{CT}}{\Delta\theta}\Bigg) = -\left(\frac{u_{EB}\sin\theta_{EB} - u_{ET}\sin\theta_{ET}}{\Delta\theta} + \frac{u_{ER} - u_{EL}}{\Delta\phi}\right) \quad (16)$$

Questions?

Pretty pictures!

Great circle method

- To avoid artefacts, Huang et al. proposed an advection scheme utilising great circles on the sphere.
- At each grid point, a great circle passing through the point and tangent to $\vec{u}$ at the point is constructed.
- The virtual particle is traced back along this great circle for a timestep of Δt and sampled to update the grid point values.
- The coordinate transformation allows all points to be treated equally as if on the equator, with no additional terms required.



(a) Before | (b) Single step (order-1, global) | (c) RK3 (order-3, global) | (d) [Yang et al. 2019] (order-2, global) | (e) Ours (order-2, aligned)

Figure: Comparing different advection schemes at the North Pole, (e) being Huang et al.'s scheme.