Solving the Navier-Stokes Equations in Primitive Variables

Grétry Trygvaason
Spring 2013

The projection method—review
Methods for the Navier-Stokes Equations
Moin and Kim
Bell, et al
Collocated grids

Summary of discrete vector equations

\[ \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = -A_{i,j} - \nabla h P_{i,j} + D_{i,j} \]

Evolution of the velocity

\[ \nabla h \cdot u_{i,j}^{n+1} = 0 \]

Constraint on velocity

No explicit equation for the pressure!

To derive an equation for the pressure we take the divergence of

\[ u_{i,j}^{n+1} = u_{i,j}^n - \Delta t \nabla h P_{i,j} \]

and use the mass conservation equation

\[ \nabla h \cdot u_{i,j}^{n+1} = 0 \]

The result is

\[ \nabla h \cdot u_{i,j}^{n+1} = \nabla h \cdot u_{i,j}^n - \Delta t \nabla h \cdot \nabla h P_{i,j} \]

\[ \nabla h \cdot u_{i,j}^{n+1} = \frac{1}{\Delta t} \nabla h \cdot u_{i,j}^n \]

Split

\[ \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = -A_{i,j} - \nabla h P_{i,j} + D_{i,j} \]

into

\[ \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = -A_{i,j} - \nabla h P_{i,j} \Rightarrow u_{i,j}^{n+1} = u_{i,j}^n + \Delta t(-A_{i,j} - D_{i,j}) \]

and

\[ \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = -\nabla h P_{i,j} \Rightarrow u_{i,j}^{n+1} = u_{i,j}^n - \Delta t \nabla h P_{i,j} \]

by introducing the temporary velocity \( u_t \)

Projection Method

1. Find a temporary velocity using the advection and the diffusion terms only:

\[ u_t_{i,j} = u_{i,j}^n + \Delta t(-A_{i,j} + D_{i,j}) \]

2. Find the pressure needed to make the velocity field incompressible

\[ \nabla h \cdot u_t_{i,j} = \frac{1}{\Delta t} \nabla h \cdot u_{i,j}^n \]

3. Correct the velocity by adding the pressure gradient:

\[ u_{i,j}^{n+1} = u_{i,j}^n - \Delta t \nabla h P_{i,j} \]
\[
\begin{align*}
v(2:nx+2,1:ny+1) + v(1:nx+1,1:ny+1) &= \frac{(2 \times h)}{2} \\
v(1:nx+1,1:ny+1) &= \frac{u(1:nx+1,2:ny+2) - u(1:nx+1,1:ny+1) - \nu v(1:nx+1,1:ny+1)}{2 \\
v(1:nx+1,1:ny+1) &= \frac{v(2:nx+2,1:ny+1) + v(1:nx+1,1:ny+1)}{2} \\
\text{Plots the results} \quad uu(1:nx+1,1:ny+1) &= \frac{u(1:nx+1,2:ny+2) + u(1:nx+1,1:ny+1)}{2} \\
\end{align*}
\]

\[
\begin{align*}
vt(2:nx+1,2:ny) &= vt(2:nx+1,2:ny) - \left(\frac{dt}{h}\right) \left(\frac{p(2:nx+1,3:ny+1) - p(2:nx+1,2:ny)}{2} \right) \\
u(2:nx,2:ny+1) &= ut(2:nx,2:ny+1) - \left(\frac{dt}{h}\right) \left(\frac{p(3:nx+1,2:ny+1) - p(2:nx,2:ny+1)}{2} \right) \\
\end{align*}
\]

\[
\begin{align*}
p(i,j) &= \beta c(i,j) (p(i+1,j) + p(i-1,j) + p(i,j+1) + p(i,j-1) - \frac{h}{dt} (ut(i,j) - ut(i-1,j) + vt(i,j) - vt(i,j-1))) \\
&\quad + (1 - \beta) p(i,j) \\
\end{align*}
\]

\[
\begin{align*}
vt(i,j) &= v(i,j) + \frac{dt}{2} (2.5 \left(\frac{u(i+1,j)+u(i,j)}{2} \frac{v(i+1,j)+v(i,j)}{2} \right) - \frac{u(i,j+1)+u(i,j)}{2} \frac{v(i,j+1)+v(i,j)}{2} ) \\
u(2:nx,2:ny+1) &= 2u - u(1:nx,2:ny+1); v(nx+2,1:ny+1) = 2v - v(nx+1,1:ny+1); \\
\end{align*}
\]

\[
\begin{align*}
\text{Time step limitations} \quad \Delta t \leq \frac{2V}{U} \quad \text{and} \quad \Delta t \leq \frac{h^2}{4V} \\
\text{where} \quad U^2 = \max(u^2 + v^2) \\
\end{align*}
\]
Forward in time, centered in space:
\[ \Delta t \to 0 \quad \text{for} \quad \text{Re} \to 0 \quad \text{and} \quad \text{Re} \to \infty \]

What is the maximum timestep?

\[ \text{Re}_{\text{max}} = \sqrt{\frac{\text{L}}{h}} \]

Increasing \( h \) \[ \Delta t \to 0 \] for \( \text{Re} \to 0 \) and \( \text{Re} \to \infty \)

For low \( \text{Re} \), use implicit methods for diffusion term

For high \( \text{Re} \), use stable advection schemes

Combine both for schemes intended for all \( \text{Re} \)

Higher Order in Time

Predictor-Corrector

A second order method can be developed by first taking a forward step, then a backward step and average the results:

\[
\begin{align*}
\text{Predictor Step:} & \\
\frac{u^* - u}{\Delta t} &= -u \cdot \nabla u + \nabla \cdot \nabla u \\
\frac{\nabla p}{\Delta t} &= \frac{1}{\Delta t} \alpha \cdot u \\
u^* &= u - \Delta t \cdot \nabla p
\end{align*}
\]

Backward step using the predicted velocity:

\[
\begin{align*}
\frac{u^* - u}{\Delta t} &= -u \cdot \nabla u + \nabla \cdot \nabla u \\
\frac{\nabla p}{\Delta t} &= \frac{1}{\Delta t} \alpha \cdot u \\
u^* &= u - \Delta t \cdot \nabla p
\end{align*}
\]

Then average the results:

\[ u^{**} = \frac{1}{2} (u^* + u) \]

Then compute

\[ u = u^* - \Delta t \cdot \nabla p \]

A complete Runge-Kutta time integration (continued)

First a half step:

\[
\begin{align*}
\frac{u^* - u}{\Delta t} &= -u \cdot \nabla u + \nabla \cdot \nabla u \\
\frac{\nabla p}{\Delta t} &= \frac{1}{\Delta t} \alpha \cdot u \\
u^* &= u - \Delta t \cdot \nabla p
\end{align*}
\]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

\[ u^* \]

Then compute

\[ k = \Delta t \left( -u^* \cdot \nabla u + \nabla \cdot \nabla u \right) \]

And finally

\[
\begin{align*}
\frac{u^* - u^*}{\Delta t} &= -u^* \cdot \nabla u + \nabla \cdot \nabla u \\
\frac{\nabla p}{\Delta t} &= \frac{1}{\Delta t} \alpha \cdot u \\
u^* &= u^* - \Delta t \cdot \nabla p
\end{align*}
\]
Simplified Fourth order method

\[
\frac{u^*_n - u^*}{\Delta t} = -\nabla \cdot (u^* \nabla u^*) + \nu \nabla^2 u^*
\]

\[
\frac{u^* - u_{n-1}}{\Delta t} = -\nabla \cdot (u_{n-1} \nabla u_{n-1}) + \nu \nabla^2 u_{n-1}
\]

\[
\frac{u^*_{n-1} - u^*_{n-2}}{2\Delta t} = -\nabla \cdot (u^*_{n-2} \nabla u^*_{n-2}) + \nu \nabla^2 u^*_{n-2}
\]

\[
u^*_{n-2} = (1 - u^*_{n-1}) + \frac{\nu}{6}(-u^*_{n-1} + u^*_{n-2})
\]

\[
\nabla \cdot u^*_{n-2} = -V \phi
\]

\[
\frac{u^*_{n-2} - u^*_{n-3}}{\Delta t} = -V \phi
\]

The first equation is implicit and must be solved by an iteration in the same way as the pressure equation.

Other Methods

Fully Implicit

\[
\frac{u_{n+1}^* - u^*}{\Delta t} = \frac{1}{2}[-(A(u^*) + A(u^*)) + \nu(\nabla^2 u^* + \nabla^2 u^*)] - V \phi
\]

\[
\nabla \cdot u_{n+1}^* = 0
\]

Solve by iteration

Rarely used due to the complications of the nonlinear system that must be solved for the advection terms

Adam-Bashford/Crank-Nicolson

\[
\frac{u^*_{n+1} - u^*}{\Delta t} = \left(\frac{3}{2}A(u^*) - \frac{1}{2}A(u^*_{n-1})\right) + \frac{\nu}{2}(\nabla^2 u^* + \nabla^2 u^*_{n-1}) - V \phi
\]

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

Split:

\[
\frac{u^*_{n+1} - u^*_n}{\Delta t} = \left(\frac{3}{2}A(u^*) - \frac{1}{2}A(u^*_{n-1})\right) + \frac{\nu}{2}(\nabla^2 u^* + \nabla^2 u^*_{n-1})
\]

\[
\nabla \cdot u_{n+1}^* = 0
\]

The correction equation is implicit and must be solved by an iteration in the same way as the pressure equation

Method of Kim and Moin (JCP 59 (1985), 8-23)

\[
\frac{u^*_{n+1} - u^*_n}{\Delta t} = \left(\frac{3}{2}A(u^*) - \frac{1}{2}A(u^*_{n-1})\right) + \frac{\nu}{2}(\nabla^2 u^* + \nabla^2 u^*_{n-1})
\]

\[
\frac{u^* - u^*}{\Delta t} = -V \phi
\]

\[
\nabla \cdot u_{n+1}^* = 0
\]

The first equation is implicit and must be solved by an iteration in the same way as the pressure equation.

Notice that \( \phi \) is not exactly \( p \). Adding the first two equations gives

\[
\frac{u^*_{n+1} - u^*_n}{\Delta t} = \left(\frac{3}{2}A(u^*) - \frac{1}{2}A(u^*_{n-1})\right) + \frac{\nu}{2}(\nabla^2 u^* + \nabla^2 u^*_{n-1}) + \frac{\nu}{2}(\nabla^2 u^*_{n-1} - \nabla^2 u^*_{n-2}) - V \phi
\]

Where we have added and subtracted an implicit diffusion term.

Using \( \frac{u^*_{n+1} - u^*_n}{\Delta t} = -V \phi \)

we can rewrite the last terms as:

\[
\frac{\nu}{2}(\nabla^2 u^* - \nabla^2 u^*_{n-2}) - V \phi = \frac{\nu}{2}(V \phi - V \phi) = V \phi
\]
Method of Bell, Colella and Glaz (JCP 85 (1989), 7-83)

\[
\frac{u_i^{n+1} - u_i^n}{\Delta t} = -A(u_i^{n+1/2}) + \frac{1}{2}(\nabla u_i^{n+1/2} + \nabla u_i^{n-1/2}) - \nabla p
\]

A Godunov method is used for the advection terms.

Many other methods have been proposed


PISO (Pressure Implicit with Split Operator). Similar to the projection method but iterates to enforce incompressibility.

Colocated grids

Although staggered grids have been very successful, in some cases it is desirable to use co-located (or colocated) grids where all variables are located at the same physical point.

Colocated grids

Staggered grids

Colocated grids

All variables are stored at the same location.

\[
\begin{align*}
\bar{u}_{i,j} &= u_{i,j} - \Delta t A(u_i^{n+1/2}) \\
\bar{u}_{i,j}^{n+1} &= \bar{u}_{i,j} + \frac{\Delta t}{\Delta x} (p_e - p_w) \\
\bar{u}_{i,j}^{n+1} &= \bar{u}_{i,j}^{n+1} + \frac{1}{2} (v_{i+1/2,j}^{n+1} - v_{i-1/2,j}^{n+1}) = 0
\end{align*}
\]

First idea: use averaging for the variables on the edges:

\[
\begin{align*}
\bar{u}_{i,j} &= \frac{1}{2} (p_{i+1,j} + p_{i,j}) \\
u_{i,j} &= \frac{1}{2} \left( u_{i+1,j} + u_{i,j} \right) \\
\bar{u}_{i,j}^{n+1} &= \bar{u}_{i,j} - \Delta t \left( \frac{1}{2} (p_{i+1,j} + p_{i-1,j}) - \frac{1}{2} (v_{i,j}^{n+1} + v_{i,j}^{n-1}) \right) \\
\bar{u}_{i,j}^{n+1} &= \bar{u}_{i,j} - \Delta t \frac{1}{2\Delta x} (p_{i+1,j} - p_{i-1,j})
\end{align*}
\]
Substituting

\[ u_i = \frac{1}{2}(u_{i+1} + u_i) \quad \text{and} \quad u_i^{*1} = u_i^{*1} - \frac{\Delta t}{\Delta x} (p_{i+1} - p_i) \]

into

\[ u_i^{*1} - u_i^{*1} + v_i^{*1} - v_i^{*1} = 0 \]

yields

\[ p_{i+1} + p_{i+2} + p_{i+2} - 4 p_{i+1} = \frac{2\Delta t}{\Delta x^2} (u_{i+1}^{*1} - u_i^{*1} + v_{i+1}^{*1} - v_{i+1}^{*1}) \]

A straightforward application discretization on colocated grids results in a very wide stencil for the pressure

The pressure points are also uncoupled and the pressure field can develop oscillations.

The remedy is to find the pressures that make the edge velocities incompressible.


The Rhie and Chow method

Instead of interpolating (the final velocity)

\[ u_i^{*1} = \frac{1}{2}(u_{i+1}^{*1} + u_i^{*1}) \]

interpolate (the intermediate velocity)

\[ u_i^{*1} = \frac{1}{2}(u_{i+1}^{*1} + u_i^{*1}) \]

and then find

\[ u_i^{*+} = u_i^{*1} - \frac{\Delta t}{\Delta x} (p_{i+1} - p_i) \]

In effect, “pretend” we are using a staggered grid!

Substituting for the velocities:

\[ p_{i+1} + p_{i+1} + p_{i+2} - 4 p_{i+1} = \frac{h}{2\Delta t} (u_{i+1}^{*1} - u_i^{*1} + v_{i+1}^{*1} - v_{i+1}^{*1}) \]

For the correction of the momentum equation we still use the average of the pressures

\[ p_i = \frac{1}{2} (p_{i+1} + p_{i+1}) \]

giving

\[ u_i^{*1} = u_i^{*1} - \frac{\Delta t}{\Delta x} (p_{i+1} - p_i) \]
The algorithm is therefore:

1. First find predicted velocities:
   \[ u^*_{ij} = u_{ij} - \Delta t \Delta \Omega^* (u) \]  
   \[ v^*_{ij} = v_{ij} - \Delta t \Delta \Omega^* (v) \]

2. Find pressure by solving:
   \[ p_{n+1,j} + p_{n+1,i} + p_{n+1,i} - 4 p_{n+1,j} = \frac{h}{2 \Delta t} \left( u^*_{ij} - u^*_{i,j+1} + v^*_{i,j+1} - v^*_{i,j} \right) \]
   suitably modified at the boundaries

3. Correct the velocities:
   \[ u^{n+1}_{ij} = u_{ij} - \frac{\Delta t}{2h} (p_{n+1,j} - p_{n+1,i}) \]  
   \[ v^{n+1}_{ij} = v_{ij} - \frac{\Delta t}{2h} (p_{n+1,i} - p_{n+1,j}) \]

The pressure equation for \( j = 2 \)

\[ p_{n+2,j} + p_{n+2,j-1} + p_{n+2,i+1} - 4 p_{n+2,j} = \frac{h}{2 \Delta t} \left( u^*_{ij} - u^*_{i,j+1} + v^*_{i,j+1} - v^*_{i,j} \right) \]

And use

\[ p_{n+2,j} = \frac{h}{2 \Delta t} v_{1,j} \]

For the pressure at \( j = 1 \)

\[ p_{n+2,1} = \frac{h}{2 \Delta t} v_{1,1} \]

The pressure boundary is more complex:

Simply the wall velocity.

At the wall, most of the terms are zero

\[ \frac{\partial p}{\partial y} = \frac{\partial^2 v}{\partial y^2} \rightarrow \Delta \frac{p_n}{h} \frac{\partial v}{\partial y} = \mu \frac{\partial^2 v}{\partial y^2} \]

Evaluated by one-sided differences

The boundary conditions for the velocity are now very simple: The velocity at nodes on the wall is simply the wall velocity.

The pressure boundary is more complex:

\[ \frac{\partial p}{\partial y} = \frac{\partial^2 v}{\partial y^2} \rightarrow \Delta \frac{p_n}{h} \frac{\partial v}{\partial y} = \mu \frac{\partial^2 v}{\partial y^2} \]
Computational Fluid Dynamics

Colocated grids

Why colocated grids:
- Sometimes simpler for body fitted grids
- Easy to use methods for hyperbolic equations
- Easier to implement AMR
- Some people just don’t like staggered grids!

The two-dimensional programs developed in the project and shown here can be extended to fully three-dimensional flows in a relatively straightforward way, replacing \( u(j) \) by \( u(i,j,k) \), etc. The time required to run the code increases significantly and visualizing the output becomes more challenging.