PROJECTION METHODS

CFD SPRING 2013, A. DONEV

LAST CLASS WE discussed how to spatially discretize

$$u_t = P\left[-u\cdot\nabla u + \nu\nabla^2 u\right]$$

How about time integration?

If one uses an explicit method, say forward Euler, it is easy:

2

$$\boxed{u^{n+1} = \mathbb{P}\left[u^n - (u\cdot\nabla u)^n \Delta t + \nu \nabla^2 u^n \Delta t\right]}$$

This is the simplest <u>projection method</u>, as originated by Chorin.

It can be seen as a <u>first-order Lie splitting</u> algorithm for the equations

$$u_t = f_1 + f_2 = \underbrace{-\nabla p}_{f_1} \; \underbrace{u\cdot\nabla u + \nu\nabla^2 u}_{f_2}$$

More peeisely, they are
<u>fractional-step methods</u>
in which velocity and pressure updates
are handled separately and the
time step split into fractional
timesteps.

(1) $u^{*} = u^{n} + \left[-(u \cdot \nabla u)^{n} + \nu \nabla^{2} u^{n} \right] \Delta t$
(unconstrained step)

(2) $u^{n+1} = \mathbb{P}\, u^{*}$

(2a) Solve <u>Poisson</u> problem for Π^{n}

(2b) $u^{n+1} = u^{*} - \Delta t \left(G \Pi^{n} \right)$

this is a kind of projected Euler step that works for any constrained evolution equation and is first order accurate

It involves a Poisson solve:

$$\begin{cases} u^{n+1} = u^* - \Delta t \, G\pi^n \\ Du^{n+1} = Du^* - \Delta t \, (DG)\pi^n = 0 \end{cases}$$

$$\Rightarrow \boxed{(DG)\pi^n = \frac{Du^*}{\Delta t}}$$

$$\Rightarrow \quad -\Delta t \, G\pi^n = - \; G(DG)^{-1} D u^*$$

$$\Rightarrow \quad u^{n+1} = \left[I - G(DG)^{-1} D \right] u^* = \mathbb{P} u^*$$

More generally, e.g., for variable density flows, the projection operator is defined via momentum conservation without any forcing:

$$\begin{cases} \rho u_t + \nabla p = 0 \\ \nabla \cdot u = 0 \end{cases}$$

But what are the BCs ?!?

$$\begin{cases} \rho^n \left(\frac{u^{n+1} - u^*}{\Delta t} \right) + G \pi^n = 0 \\ D u^{n+1} = 0 \end{cases} \Rightarrow$$

<u>Defines</u>

$$u^{n+1} = \mathbb{P} u^*$$

How to do <u>second-order</u> ?

A Crank-Nicolson (implicit midpoint)

would be

$$\left(I - \nu\frac{\widetilde{\mathbb{P}L}\Delta t}{2}\right)u^{n+1} = \left(I + \nu\frac{\widetilde{\mathbb{P}L}\Delta t}{2}\right)u^{n} - \left[u\cdot\nabla u\right]^{n+1/2}$$

<u>implicit viscosity</u>

e.g. Adams-Bashforth

<u>explicit advection</u>

This looks complicated by the fact it involves $\mathbb{P}L$, but if one works it out using the fact that projection means solving

$$u_1 = \mathbb{P} u_2 \quad \text{means that } \exists \varphi$$ ⑦

$$\Updownarrow$$

$$\begin{cases} u_1 + \nabla \varphi = u_2 \\ \nabla \cdot u_1 = 0 \end{cases}$$

$$\Rightarrow \boxed{\begin{cases} \dfrac{u^{n+1} - u^n}{\Delta t} + \nabla p^{n+1/2} = -\left[u \cdot \nabla u\right]^{n+1/2} + \dfrac{\nu}{2} \nabla^2 \left(u^n + u^{n+1}\right) \\ \nabla \cdot u^{n+1} = 0 \qquad \text{with appropriate BCs} \end{cases}} \quad (*)$$

which is now simply seen as a direct discretization of the original NS equation!

How do we solve the linear system (*) ?

$$\left[\begin{array}{c|c} \frac{I}{\Delta t} - \frac{\nu}{2} L_\nu & G \\ \hline -D & 0 \end{array}\right] \begin{bmatrix} u^{n+1} \\ \pi^{n+1/2} \end{bmatrix} = \begin{bmatrix} \left(\frac{I}{\Delta t} + \frac{\nu}{2} L_\nu\right) u^n \\ - [u \cdot \nabla u]^{n+1/2} \\ \hline 0 \end{bmatrix}$$

"Saddle-point" problem $\rightarrow$ Stokes system

$$\left[\begin{array}{c|c} A & G \\ \hline G^* & 0 \end{array}\right] \begin{bmatrix} u^{n+1} \\ \pi^{n+1/2} \end{bmatrix} = rhs = \begin{bmatrix} b_u \\ b_\pi \end{bmatrix}$$

The widespread belief was/is that the saddle-point problem is hard to solve. This is actually wrong, but let's follow the historical path and try to do a second-order projection algorithm to approximate (*).

Step 1: Unconstrained step

$$\frac{u^* - u}{\Delta t} + \nabla q = -\left[(u \cdot \nabla) u\right]^{n+1/2} + \frac{\nu}{2} \nabla^2 (u^* + u^n) \quad \ldots (1)$$

subject to, for example, the same boundary conditions as for u, e.g. no slip

10

Following a seminal paper by Brown, Minion, Cortez (2001), we let the pressure approximation q be arbitrary, given.

2 Now project the result:

$$\begin{cases} u^* = u^{n+1} + \Delta t \, \nabla \varphi^{n+1} & \dots (2) \\ \nabla \cdot u^{n+1} = 0 \end{cases}$$

$$u^{n+1}\big|_{\partial\Omega} = 0 \quad \text{(for example)}$$

Solve for φ^{n+1} (Poisson problem)

What is the pressure?

Substitute (2)

$$u^* = u^{n+1} + \Delta t \nabla \varphi^{n+1}$$

into (1) and compare to (*),

to get:

$$\nabla p^{n+1/2} = \nabla q + \nabla \varphi^{n+1} - \frac{\nu \Delta t}{2} \nabla^2 (\nabla \varphi^{n+1})$$

$$G p^{n+1/2} = G (q + \varphi^{n+1}) - \frac{\nu \Delta t}{2} L_v G \varphi^{n+1}$$

This is an overdetermined equation, so in general there is no solution.

At the continuum level, ∇^2 and ∇ commute, so

$$\nabla p^{n+1/2} = \nabla\left[q + \varphi^{n+1} - \frac{\nu \Delta t}{2} \nabla^2 \varphi^{n+1}\right] \Rightarrow$$

$$\boxed{p^{n+1/2} = q + \varphi^{n+1} - \frac{\nu \Delta t}{2} \nabla^2 \varphi^{n+1}}$$

pressure correction due to viscosity

If we use this in the discrete setting, it is an approximation since L_ν and G do NOT commute except for periodic systems.

But it is a valid second-order approximation,

$$\boxed{p^{n+1/2} \approx q + \varphi^{n+1} - \frac{\nu \Delta t}{2} L \varphi^{n+1}} \quad \ldots (3)$$

where $L = DG$ is the scalar Laplacian used in the Poisson solve for φ^{n+1}.

Boyce Griffith (Courant / NYU) has found that using (2) + (3) as a preconditioner for a Krylov solver (GMRES) works well and helps solve (*) very efficiently

Multigrid is used for the linear solvers in (1) and (2), with inexact convergence criterion (e.g. 1-2 V-cycles only!)

Advantages

1. Any mistakes made in imposing artificial boundary conditions in (1) and (2) is corrected by the Krylove solver
2. Any kind of boundary condition can be handled
3. Works in the Stokes limit (steady flow)

But, for unsteady flow (not in Stokes limit), we can get the same <u>second-order</u> spatio-temporal accuracy for no-slip boundaries if we use this modified Bell-Collela-Glaz algorithm:

1. Solve (1) with $q = \Pi^{n-1/2}$ (<u>time-lag</u> pressure)
2. Solve for φ^{n+1} using <u>Neumann BCs</u> if any BCs are needed (NOT for staggered)
3. Calculate u^{n+1} from (2)
4. $$\nabla \Pi^{n+1/2} = \nabla \Pi^{n-1/2} + \nabla \varphi^{n+1} - \frac{\nu \Delta t}{2} \nabla^2 \left(\nabla \varphi^{n+1} \right)$$

Summary of <u>one</u> version of second-order projection for $u_{\partial\Omega} = 0$: (16)

(1) Solve (using multigrid or such)

$$\begin{cases} \dfrac{u^* - u^n}{\Delta t} + G p^{n-1/2} = -[u \cdot \nabla u]^{n+1/2} + \nu L_0 \left(\dfrac{u^n + u^*}{2} \right) \\ u^*_{\partial\Omega} = 0 \end{cases}$$

(2) Solve Poisson problem (multigrid)

$$L \varphi^{n+1} = (DG) \varphi^{n+1} = \frac{D u^*}{\Delta t}$$

with Neumann BCs

(3) Project velocity $u^{n+1} = u^* - \Delta t \cdot G \varphi^{n+1}$

(4) Correct pressure (gradient)

$$p^{n+1/2} = p^{n-1/2} + \varphi^{n+1} - \frac{\nu \Delta t}{2} \left(L \varphi^{n+1} \right)$$

For <u>viscous dominated</u> flows or if one wants to take large time steps, $\Delta t \gg \frac{\Delta x^2}{\nu}$, it is better to use Backward Euler (see HW3):

$$\left[\begin{array}{c|c} \frac{I}{\Delta t} - \nu L_v & G \\ \hline -D & 0 \end{array}\right] \begin{bmatrix} u^{n+1} \\ \pi^{n+1/2} \end{bmatrix} = \left[\begin{array}{c} \frac{I}{\Delta t} u^n + f^{n+1/2} \\ \hline 0 \end{array}\right]$$

and solve this using GMRES preconditioned by a Projection method.

<u>Note</u>: As $\Delta t \to \infty$, this solves the <u>Steady Stokes</u> problem: $\begin{cases} \nu \nabla^2 u + f = \nabla P \\ \nabla \cdot u = 0 \end{cases}$

as we will discuss next.