Class 7

- von Neumann analysis
- implicit vs explicit schemes
- iterative methods for linear systems
- implicit operators in OpenFOAM

Remainder of semester

- Incompressible N-S, projection methods, meshing (Class 8)
- Final projects, lift/drag, turbulence modeling
- Multiphase flow

Return HW this week.

- Feedback on reading period

$$\begin{cases} \partial_t u + a\partial_x u = \nu \partial_{xx} u \\ u(0) = u(2\pi) \end{cases}$$

- Recall our FD scheme

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -a\left(\frac{u_i^n - u_{i-1}^n}{\Delta x}\right) + \nu\left(\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2}\right)$$

$$u_i^{n+1} = u_i^n - K_1\left(u_i^n - u_{i-1}^n\right) + K_2\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)$$

- From PDE, we know that linear PDE take the form for a single Fourier mode

$$u(x,t) = \hat{u}(t)\exp(ikx)$$

- Plugging this in

$$u_i^n = u(x_i, t^n)$$

$$u_i^n - u_{i-1}^n = \hat{u}(t^n)\left[\exp(ikx_n) - \exp(ik(x_n - \Delta x))\right]$$

$$= \hat{u}(t^n)\exp(ikx_n)\left(1 - \underbrace{\exp(-ik\Delta x)}_{\cos(\Delta x k) - i\sin(k\Delta x)}\right)$$

$$u_{i+1}^n - 2u_i^n + u_{i-1}^n = \hat{u}(t^n)\left[\exp(ik(x_n + \delta x)) - 2e^{ikx_n} + e^{ik(x_n - \delta x)}\right]$$

$$= \hat{u}(t^n)e^{ikx_n}\left(\underbrace{e^{k\Delta x} - 2 + e^{-k\Delta x}}_{2\cos k\Delta x - 2}\right)$$

$$\hat{u}(t^{n+1})\cancel{e^{ikx}} = \hat{u}(t^n)\cancel{e^{ikx}}\left[1 - K_1\left(1 - \cos k\Delta x + i \sin k\Delta x\right) + 2K_2\left(\cos k\Delta x - 1\right)\right]$$

What condition such that mode doesn't grow?

$$\left|\frac{\hat{u}(t^{n+1})}{\hat{u}(t^n)}\right| < 1 \quad ?$$

(Take $K_1 = 0$ for simplicity)

$$= \Bigg|1 - 2K_2\underbrace{\left(\cos k\Delta x - 1\right)}_{\text{between } 0 \text{ and } -2}\Bigg| < 1$$

$$\leq |1 - 4K_2|$$

$$\boxed{K_2 \leq \frac{1}{2}} \Rightarrow \boxed{\frac{\nu \Delta t}{\Delta x^2} < \frac{1}{2}}$$

- Similarly (HW) $\boxed{\frac{a\Delta t}{\Delta x} < 1}$

- We can see the big issue... what if ν is large? Then we need a tiny timestep.

Implicit schemes

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -a\left(\frac{u_i^{n+1} - u_{i-1}^{n+1}}{\Delta x}\right) + \nu\left(\frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{\Delta x^2}\right)$$

Take $a = 0$, Let $K_2 = \frac{\nu \Delta t}{\Delta x^2}$

$$u_i^{n+1} = u_i^n + K_2\left(u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}\right)$$

Again, ansatz $u(x,t) = \hat{u}(t)\, e^{ikx}$

$$\hat{u}^{n+1} e^{ikx} = \hat{u}^n e^{ikx} + K_2\, \hat{u}^{n+1}\left(e^{ik(x+\Delta x)} - 2e^{ikx} + e^{ik(x-\Delta x)}\right)$$

$$\hat{u}^{n+1} = \hat{u}^n + K_2 \hat{u}^{n+1} \underbrace{\left(e^{ik\Delta x} - 2 + e^{-ik\Delta x}\right)}_{2\cos k\Delta x - 2}$$

$$\hat{u}^{n+1}\left(1 + K_2(2 - 2\cos k\Delta x)\right) = \hat{u}^n$$

$$\left|\frac{\hat{u}^{n+1}}{\hat{u}^n}\right| \leq \left|\frac{1}{1 + 2K_2(\underbrace{1 - \cos k\Delta x}_{\text{between } 0,\, 2})}\right| \leq 1$$

$\longrightarrow$ unconditionally stable

Almost there - how to solve?

- $\vec{u}^{n+1} = A\backslash\vec{u}^n$ in Matlab will be extremely expensive (depending on how careful you are)
- Gauss elimination is $O(N^3)$ (dense solver)
- Need it to be fast enough that we're better off than using explicit scheme

explicit $\rightarrow$ steps $\rightarrow \frac{t_{final}}{\delta t}$

$\delta t \sim 1/N^2 \Rightarrow N^2$ steps, each w/ $O(N)$ updates $\Rightarrow O(N^3)$

implicit $\rightarrow \delta t \sim 1/N \Rightarrow N$ steps, need $O(N^2)$ update or better to break even

- even storing the matrix is expensive $(O(N^2))$

Iterative Sparse Solvers

While direct solvers provide an exact (to machine precision) solution in $O(N^3)$ time, iterative solvers take an initial guess x_0 and generate a sequence $\{x_i\}_{i=1,2,\ldots}$ such that the residual $r_i = b - Ax_i$ satisfies

$$\|r_i\| \searrow 0 \text{ for large } i.$$

Key idea: Start with a good guess, terminate when $\frac{\|r_i\|}{\|b\|} < \varepsilon$ (e.g. 0.1%)

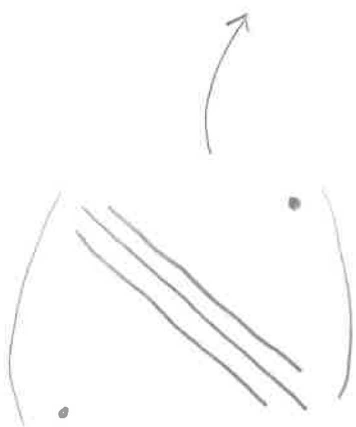
So how do we implement this? Each point of the stencil relies on other stencils information

Collect terms

$$u_{i+1}^{n+1}\underbrace{\left(-K_2\right)}_{\alpha_{i,i+1}} + u_i^{n+1}\underbrace{\left(1+K_1+2K_2\right)}_{\alpha_{i,i}} + u_{i-1}^{n+1}\underbrace{\left(-K_1-K_2\right)}_{\alpha_{i,i-1}} = u_i^n$$

Build a big system of equations

$$A\,u^{n+1} = u^n$$



$$\begin{pmatrix} \alpha_{00} & \alpha_{01} & & & \alpha_{0N} \\ \alpha_{10} & \alpha_{11} & \alpha_{12} & & \\ & \alpha_{21} & \alpha_{22} & \alpha_{23} & \\ & & \ddots & \ddots & \ddots \\ \alpha_{N,0} & & & \alpha_{N,N-1} & \alpha_{NN} \end{pmatrix} \begin{pmatrix} u_0^{n+1} \\ u_1^{n+1} \\ \vdots \\ u_N^{n+1} \end{pmatrix} = \begin{pmatrix} u_0^n \\ \vdots \\ u_N^n \end{pmatrix}$$

Jacobi's method

idea Want an update formula

$$x_{n+1} = x_n + B r_n$$

what if we had A^{-1}? $\qquad A x_{exact} = b$

$$x_{n+1} = x_n + A^{-1}(b - Ax_n)$$

$$x_{n+1} = x_n + x_{exact} - x_n$$

$$= x_{exact}$$

But if we knew A^{-1} we'd be done already...

Split $Ax = b$

$$Bx^{n+1} + (A-B)x^n = b$$

$$x^{n+1} = B^{-1}\left(b - (A-B)x^n\right)$$

Want B thats cheap to invert

Jacobi $\Rightarrow$ $B = \mathrm{diag}(A)$

Take $x_0 = x^n$ $\qquad$ w/

for

$$x_i^{n+1} = \frac{1}{a_{ii}}\left(x_i^n - a_{i,i-1}x_{i-1}^n - a_{i,i+1}x_{i+1}^n\right)$$

Algorithm (what goes in your update function)

update() {

last timestep

compute this without storing matrix

- $x^n = x_{old}$
- $r = b - Ax_{old}$
- $\varepsilon = \|r\|_2$

while $\varepsilon < TOL$

for $i = 1 : N$

$$x_i^{n+1} = \frac{1}{a_{ii}}\left(x_i^n - a_{i,i+1}\, x_{i+1}^n - a_{i,i-1}\, x_{i-1}^n\right)$$

end

$r = b - A x^{n+1}$

$\varepsilon = \|r\|_2$

$x_i^n = x_i^{n+1}$

end

Some choices for stopping criteria

$$\varepsilon = \|r\|_2 \qquad \varepsilon = \frac{\|r\|}{\|b\|} \qquad \varepsilon = \frac{\|x_{n+1} - x_n\|}{\|x_n\|}$$