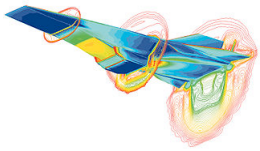
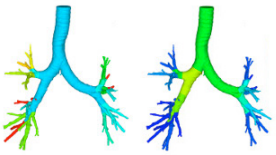
Hyper-X at Mach 7, NASA image in the public domain

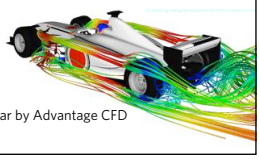


ME 702
Computational Fluid Dynamics, CFD

Prof. Lorena A. Barba



Human airways, by FuiDA nv



F1 car by Advantage CFD

BU

Accuracy of the FD approximations

- Write a Taylor series expansion for $u(x)$ around point x_i
- replace x by x_{i+1}
- solve for $\dfrac{\partial u_i}{\partial x}$

Order of accuracy

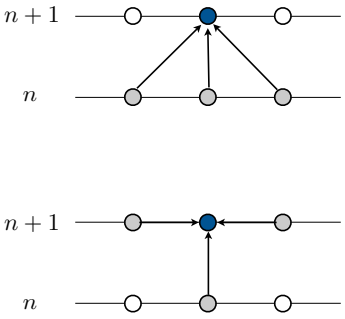
The power of Δx with which the truncation error tends to zero for a finite-difference approximation.

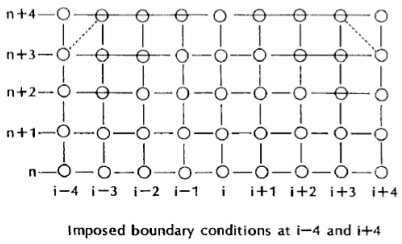
Second-order derivatives

- slope of the line tangent to $\dfrac{\partial u}{\partial x}$
- combine FD and BD for 1st derivative

$$\frac{\partial^2 u_i}{\partial x^2} = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} - \mathcal{O}(\Delta x^2)$$

- Explicit / Implicit FD formulas
  - *recall the 1D diffusion eqn.* $\dfrac{\partial u}{\partial t} = \nu \dfrac{\partial^2 u}{\partial x^2}$

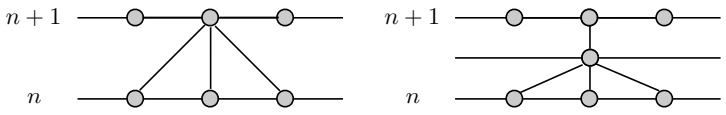




Imposed boundary conditions at i−4 and i+4

Crank-Nicholson method

- Average of explicit and implicit schemes



Practical module:

"The 12 steps to computing Navier-Stokes"

③ — 1D diffusion

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial x^2}$$

FD in time & CD in space

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \nu \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}$$

Pseudocode Step 3

```
nx = 20, nt = 50
dt = 0.01, vis = 0.1
dx = 2/(nx-1)
```

```
for i = 1:nx
 if 0.5 <= x(i) <= 1
  u(i)=2
 else
  u(i)=1
 end
end
```

```
for it = 1:nt
 un = u
 for i = 2:nx-1
  u(i) = un(i) + vis*dt/dx/dx...
    *( un(i+1)-2*un(i)+un(i-1) )
 end
end
```


④ — 1D Burger's equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}$$

Pseudocode Step 4

```
nx = 20, nt = 50
dt = 0.01, vis = 0.1
dx = 2/(nx-1)

for i = 1:nx
 ip1(i) = i+1
 im1(i) = i-1
   x(i) = (i-1)*dx
end

ip1(nx) = 1
 im1(1) = nx
```

```
for i = 1:nx
 phi = exp( -x(i)^2/4/vis )+ exp( -(x(i)-2*pi).^2/4/vis )

 dphi = -0.5/vis*x(i)* exp( -x(i)^2/4/vis )- 0.5/vis*( x(i)-2*pi )...
      * exp( -(x(i)-2*pi)^2/4/vis )

 u(i) = -2*vis*dphi/phi + 4
end
```

④ — 1D Burger's equation

Pseudocode Step 4 (cont'd)

```
for it = 1:nt
 t = (it-1)*dt
 for i = 1:nx
  phi = exp( -(x(i)-4*t)^2/4/vis/(t+1) )+ exp( -(x(i)-4*t-2*pi)^2...
             /4/vis/(t+1) )

  dphi = - 0.5/vis/(t+1)*( x(i)-4*t )*exp(-(x(i)-4*t)^2/4/vis/(t+1) )...
         - 0.5/vis/(t+1)*( x(i)-4*t-2*pi )*exp( -(x(i)-4*t-2*pi).^2...
             /4/vis/(t+1) )

  ua(i) = -2*vis*dphi./phi + 4;
 end

 un = u
 for i = 1:nx
  u(i) = un(i)-un(i)*dt/dx*( un(i)- un(im1(i)) )+ vis*dt/dx^2*...
         ( un(ip1(i))- 2*un(i)+ un(im1(i)) )
 end
end
```