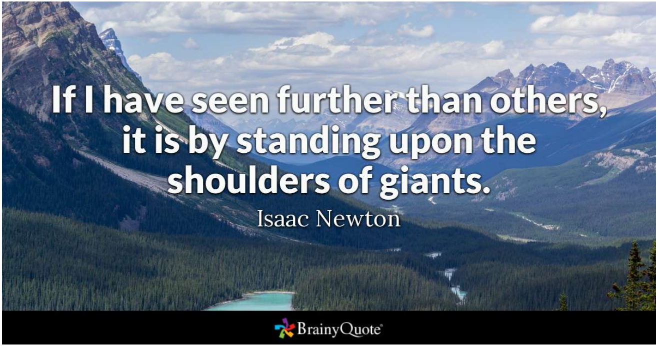
ENT441 Computational Fluid Dynamics

Muhammad Izham, PhD

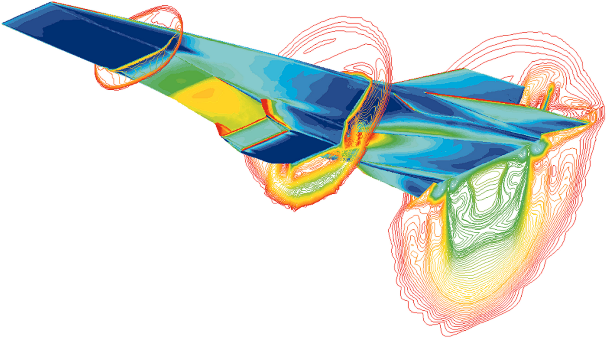
Universiti Malaysia Perlis

izham@unimap.edu.my
sugita5019@gmail.com
https://github.com/izham-sugita

Why Computational Fluid Dynamics?



COLORFUL FLUID DYNAMICS



Overview

Model Partial Differential Equations

- Advection Equation
- Diffusion/Heat Equation
- Poisson Equation

Finite Difference Method

Finite Volume Method

Grid generation

Advection equation 1

- Linear advection equation.

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0, \qquad c \geq 0 \tag{1}$$

$$u\left(x, 0\right) = u_0$$

Advection equation 2

- Initial condition and analytical solution at t=2.0000.

$$u_0 = \begin{cases} 1.0 & 2.0 \leqslant x \leqslant 4.0 \\ 0.0 & other \end{cases}$$

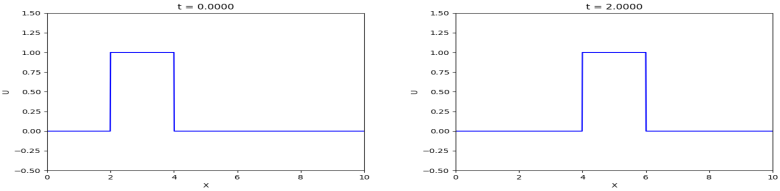


Figure: Left is at t=0.0000 and right is at t=2.0000

Advection equation 3

- Sample of upwind discretization in Python.

```
import numpy as np
import matplotlib.pyplot as plt
```

```
c = 1.0
alpha = c*dt/dx
for iter in range(itermax):

    for i in range(1,imax-1):
        iup = i - int(np.sign(c))
        un[i] = u[i] - alpha*( u[i] - u[iup] )

    #update, make sure its a copy; not the same object
    u[:] = un[:]
```

Diffusion/Heat equation 1

- Diffusion equation or heat equation.

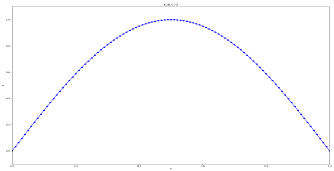
$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2} \tag{2}$$

- The boundary condition

$$T(x,0) = \begin{cases} f(x) & \textit{Neumann BC} \\ const. & \textit{Dirichlet BC} \end{cases} \qquad \forall x \in [0, L] \tag{3}$$

Diffusion/Heat equation 2

- Diffusion/Heat equation in one-dimension



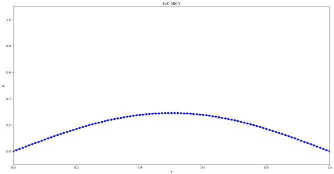


Figure: Left is at t=0.0000 and right is at t=0.5000

Diffusion/Heat equation 3

- Sample of the heat equation discretization in Python.

```
dt = np.float64(input("Enter dt, dx=%s (dt is
    propotional to dx*dx)"%dx ))
itermax = np.int64( 0.5/dt ) +1
print("Maximum iteration: ", itermax)
kappa = 0.25 #diffusion coefficient
alpha = kappa * dt /(dx**2)
steps = itermax/1000
for iter in range(itermax):
    for i in range(1,imax-1):
        un[i] = u[i] + alpha*( u[i-1] - 2.0*u[i] +
    u[i+1] )
    #update
    u[:] = un[:]
```

Poisson/Laplace equation 1

- Poisson equation/ Laplace equation.

$$-\frac{\partial^2 u}{\partial x^2} = \begin{cases} f(x) & \textit{Poisson equation} \\ 0 & \textit{Laplace equation} \end{cases} \qquad \forall x \in [0, L] \quad (4)$$

- The boundary condition

$$T(x, 0) = \begin{cases} g(x) & \textit{Neumann BC} \\ const. & \textit{Dirichlet BC} \end{cases} \qquad \forall x \in [0, L] \quad (5)$$

Poisson/Laplace equation 2

- Diffusion/Heat equation in one-dimension

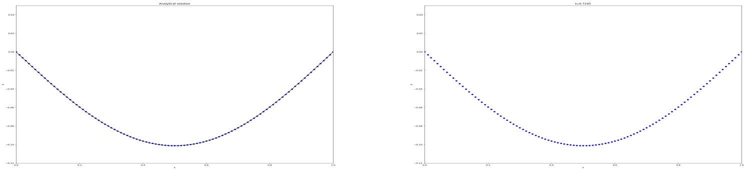


Figure: Left is comparison with analytical solution and right is the converged solution.

Poisson/Laplace equation 3

- Sample of Poisson equation discretization in Python.

```
for iter in range(1,itermax):
    for i in range(1,imax-1):
        un[i] = u[i] + alpha*( u[i-1] - 2.0*u[i] + u[i+1] ) - dt*source[i]
```

- The source term refers to $f(x)$

Standard template

- This is a standard template slide.
- Modify by adding items.

Standard template

- This is a standard template slide.
- Modify by adding items.

Standard template

- This is a standard template slide.
- Modify by adding items.

Thank You!

Questions?