SESSION 2: MATLAB NUMERICS

htp://uiuc-cse.github.io/matlab-sp17/

OUTLINE

- Review: Matlab basics
  - Control Flow, Function
- *Example: Heat conduction (Numerics & linear algebra)*
- Example: Radioactive decay chain (system of 1st order ODEs)
- *Example: Radioactive decay scheme (System of 1st-order ODEs)*
- *Example: Shock waves (nonlinear PDE)*

FOR LOOP, IF-ELSE

- for loop :

  for *loop_index = vector*
  block of code
  end

- if-elseif – else statement:

  if *condition 1*
  do something 1
  elseif *condition 2*
  do something 2
  else
  do something 3
  end

EXAMPLE: HEAT CONDUCTION (1)

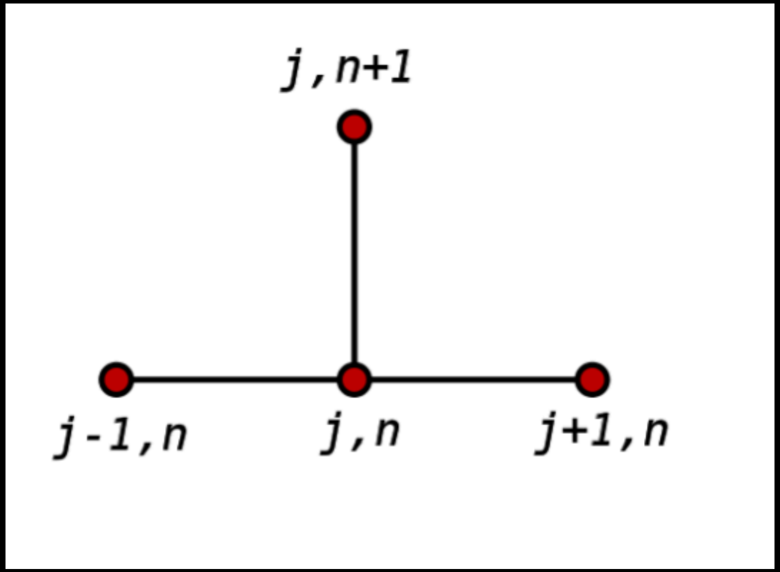
- Transient heat conduction problem

Initial condition u(x,t=0) = sin(πx/L),
Boundary conditions u(x=0,t) = u(x=L,t) = 0 and length L = 1

$$\boxed{\frac{1}{\alpha}\frac{du}{dt}=\frac{d^2u}{dx^2}} \quad \text{thus} \quad \boxed{\frac{1}{\alpha}\frac{u_i^m}{\delta t}=\frac{u_{i-1}^{m-1}-2u_i^{m-1}+u_{i+1}^{m-1}}{\delta x^2}}$$

Finite difference approximation

$$f'(x)\approx\frac{f(x+h)-f(x)}{h} \quad \text{thus} \quad f''(x)\approx\frac{f(x+h)-2\mathrm{f}(x)+f(x-h)}{h^2}$$



EXAMPLE: RADIOACTIVE DECAY CHAIN (1)

- $\frac{dN}{dt} = -\lambda N$

where "N" is the number of atoms and λ is the decay constant of the material.

- Now, this system has an analytic solution of the following form:

$$N(t) = N_0 e^{(-\lambda t)}$$

Write a function (dNdt) to return the analytical value for N given N0, λ, t.

- Matlab's built-in solvers: ode23, ode45, etc…
- Compare the results visually:
  - analytical vs. ode23
  - ode23 vs. ode 45

EXAMPLE: RADIOACTIVE DECAY CHAIN (2)

- Coupled Linear ODE:

$$
\begin{aligned}
\frac{dc_1}{dt} &= -\lambda_1\, c_1 \\
\frac{dc_2}{dt} &= \lambda_1\, c_1 - \lambda_2\, c_2 \\
\frac{dc_3}{dt} &= \lambda_2\, c_2 - \lambda_3\, c_3
\end{aligned}
$$

Matrix Form

$$
\begin{bmatrix} c_1^{(1)} \\ c_2^{(1)} \\ c_3^{(1)} \end{bmatrix} = \begin{bmatrix} -\lambda_1 & 0 & 0 \\ \lambda_1 & -\lambda_2 & 0 \\ 0 & \lambda_2 & -\lambda_3 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}
$$

- Re-write dNdt function with vectorized equtions

EXAMPLE: SHOCK WAVES (1)

- The inviscid Burgers' equation:

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} = -\frac{1}{2}\frac{\partial\left(u^2\right)}{\partial x}$$

- Approximate version of the equations:

$$u_j^{n+1} = u_j^n\left(1 - \Delta t\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}\right)$$

EXAMPLE: SHOCK WAVES(2)

- New function: burgers.m
- Specify the fundamental parameters as variables to make them easy to adjust.
- Set the initial condition:

  cosine-shaped wave at t = 0: u(x,0) = u0(x) = 1-cos(x).
- Define the boundary condtion.
- Main equation: u(n+1,j) = u(n,j) * (1 - dt * (u(n,j+1)-u(n,j-1))/(2*dx));

RECAP - FUNCTION

- Method 1: Explicitly define a function through m file.
- Method 2: Use a function handle

(ex)

```
fun = @(x) sin(x);
f1 = @(f,a,b,c) c*f(a+b);
f1(fun,2,3,4)
```