

## Numerical solution of ODE & PDE

First order ordinary differential equation (ODE), with initial value

$$\frac{dy}{dx} = f(y(x), x), \quad \text{with initial value } y(x_0) = y_0$$

$dy/dx$  is tangent to solution curve  $y = y(x)$  at point  $x$  given  $x = x_0, y = y_0$ .

Possible to convert second-order ODE to two coupled first order ODE,

$$\frac{d^2x}{dt^2} = -\omega^2 x \Rightarrow v = \frac{dx}{dt} \text{ and } \frac{dv}{dt} = -\omega^2 x - \mu v$$

initial conditions being  $x(t_0) = x_0$  and  $v(t_0) = v_0$ .

To solve first order ODEs given an initial condition, the methods we discuss here are

1. Forward (explicit) and Backward (implicit) Euler's method
2. Predictor-Corrector method
3. Runge-Kutta 4th order

## Forward (explicit) Euler

Based on Taylor expansion of  $f(x_0 + h)$  about  $x_0$  assuming  $h$  is small

$$y(x_0 + h) = y(x_0) + h \frac{dy}{dx} \Big|_{x_0} + \frac{h^2}{2!} \frac{d^2y}{dx^2} \Big|_{x_0} + \dots \approx y(x_0) + h f(y(x_0), x_0) + \mathcal{O}(h^2)$$

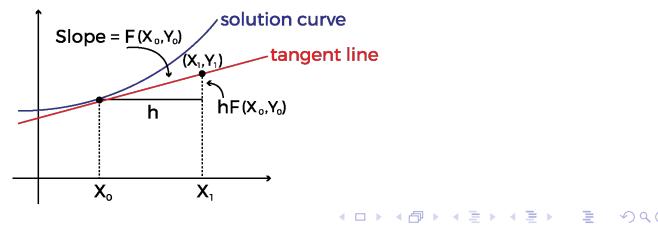
Say  $x_1 = x_0 + h$  a small  $h$  step away from  $x_0 \Rightarrow$  Forward Euler gives

$$y(x_1) = y(x_0) + h f(y(x_0), x_0) + \mathcal{O}(h^2)$$

Continue from  $x_1$  to  $x_2 = x_1 + h$ , to  $x_3$  etc. After  $n$ -th step

$$y(x_n + h) = y(x_n) + h f(y(x_n), x_n) \text{ or, equivalently } y_{n+1} = y_n + \kappa_1$$

where  $\kappa_1 = h f(y(x_n), x_n)$  implies **slope** or **tangent** at the beginning of the interval boundary  $\Rightarrow$   $dy/dx$  calculated at earlier point  $x_n$  to obtain solution at the end of interval  $x_n + h$ .

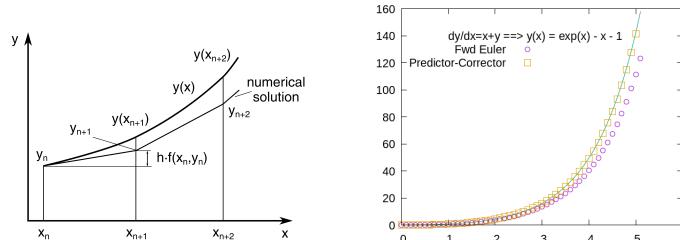


It is **Forward** because

$$\Rightarrow \frac{dy}{dx} = \lim_{\Delta x \rightarrow 0} \frac{\Delta y}{\Delta x} = \lim_{\Delta x \rightarrow 0} \frac{y(x + \Delta x) - y(x)}{\Delta x} = f(y(x), x)$$

Forward Euler often returns good approximation to actual solution, but it is extremely slow  $\rightarrow h$  has to be small to achieve desired accuracy.

Its biggest problem is **stability issue**, can easily veer away from solution.



Left plot  $\circ$  is for Forward Euler and  $\square$  for Predictor-Corrector.

Stability of Forward Euler is more of a problem than its sluggishness.

## Backward (implicit) Euler

Re-define  $dy/dx$  in terms of *backward derivative*,

$$\begin{aligned}\frac{\Delta y}{\Delta x} &\approx \frac{y(x) - y(x - \Delta x)}{\Delta x} = f(y(x), x) \Rightarrow y(x) = y(x - \Delta x) + \Delta x f(y(x), x) \\ &\Rightarrow y(x + \Delta x) = y(x) + f(y(x + \Delta x), x + \Delta x) \\ &\equiv y(x_n + h) = y(x_n) + h f(y(x_n + h), x_n + h)\end{aligned}$$

$y(x_n + h)$  is determined from the tangent at  $x_n + h$ , implying strangely that to calculate  $y(x_n + h)$  one needs to know  $y(x_n + h)!!$

This apparent conflict is resolved by looking it as a *linear equation* and solve it by [Newton-Raphson](#),

$$y^{NR}(x + h) = y(x) + h f(y^{NR}(x + h), x + h)$$

Hence, solving the differential equation by

$$y(x_n + h) = y(x_n) + h f(y^{NR}(x_n + h), x_n + h)$$

In spite of having an extra step of [Newton-Raphson](#), the [Backward Euler](#) is advantageous because of better stability.

## Predictor-Corrector method

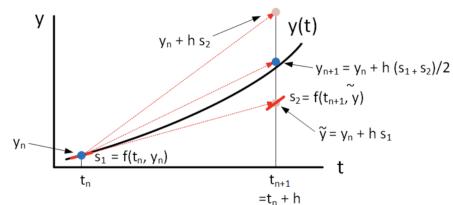
Predicts  $y^p(x_n + h)$  using Forward Euler and estimate  $f(y^p(x_n + h), x_n + h)$

$$y^P(x_n + h) = y(x_n) + h f(y(x_n), x_n) + \mathcal{O}(h^2)$$

Take the average of the two slopes to obtain correct value  $y^c(x_n + h)$ ,

$$y^c(x_n + h) = y(x_n) + \frac{h}{2} \left[ f(y(x_n), x_n) + f(y^p(x_n + h), x_n + h) \right] \equiv y(x_n) + \frac{1}{2} \left[ \kappa_1 + \kappa_2 \right]$$

where,  $\kappa_1 = h f(y(x_n), x_n)$ ,  $\kappa_2 = h f(y^p(x_n + h), x_n + h)$ . The  $\kappa_2$  denotes predicted slope at the end of interval boundary.



1. Compute slope  $\kappa_1 = hf(y(x_n), x_n)$  at  $x_n$ .
  2. Calculate the predicted  $y^p(x_n + h) = y(x_n) + \kappa_1$ , hence compute  $\kappa_2 = hf(y^p(x_n + h), x_n + h) = hf(y(x_n) + \kappa_1, x_n + h)$ .
  3. Calculate the corrected solution  $y^c(x_n + h) = y(x_n) + (\kappa_1 + \kappa_2)/2$ .

## Runge-Kutta method

Runge-Kutta (RK) methods are based on Taylor expansion and give better algorithms for solutions of ODE for same step size and stability.

$$\begin{aligned}\frac{dy}{dx} &= f(y(x), x) \Rightarrow \int_{y_n}^{y_{n+h}} dy = \int_{x_n}^{x_n+h} f(y(x), x) dx \\ y(x_n + h) &= y(x_n) + \int_{x_n}^{x_n+h} f(y(x), x) dx\end{aligned}$$

Numerical estimate of the integral can come from any of Midpoint, Trapezoidal or Simpson rule. Take Midpoint for instance,

$$\begin{aligned}\bar{x}_n &= [(x_n + h) + x_n]/2 = x_n + h/2 \\ y(x_n + h) &= y(x_n) + h f(y(x_n + h/2), x_n + h/2) + \mathcal{O}(h^3) \\ \text{where, } y(x_n + h/2) &= y(x_n) + \frac{h}{2} f(y(x_n), x_n)\end{aligned}$$

Last step above is Forward Euler, which leads to second order Runge-Kutta (RK2)

$$\begin{aligned}\kappa_1 &= h f(y(x_n), x_n) \\ \kappa_2 &= h f(y(x_n) + \kappa_1/2, x_n + h/2) \\ y(x_n + h) &\approx y(x_n) + \kappa_2 + \mathcal{O}(h^3)\end{aligned}$$

Note a couple of points on [RK2](#) –

1. Difference between previous one-step methods like Euler's and Predictor-Corrector methods is the addition of an intermediate **half-step**.
2. Order of error follows from the maximum error bound of [Midpoint method](#).
3. **Trapezoidal** rule reproduces Predictor-Corrector formula with the same error  $\mathcal{O}(h^3)$ .

Next obvious step is using [Simpson](#) rule as integrator to develop famous and by far the most popular method for solving ODE : [fourth-order Runge-Kutta \(RK4\)](#).

Unless stated or asked differently, it will always be assumed that [RK4](#) is used to solve ODE.

Consider the integral equation again, this time using [Simpson](#) rule,

$$\begin{aligned}
 y(x_n + h) &= y(x_n) + \int_{x_n}^{x_n + h} f(y(x), x) dx \\
 &= y(x_n) + \frac{h}{6} \left[ f(y(x_n), x_n) + 4f(y(x_n + h/2), x_n + h/2) + f(y(x_n + h), x_n + h) \right] \\
 &= y(x_n) + \frac{h}{6} \left[ f(y(x_n), x_n) + 2f(y(x_n + h/2), x_n + h/2) + \right. \\
 &\quad \left. 2f(y(x_n + h/2), x_n + h/2) + f(y(x_n + h), x_n + h) \right]
 \end{aligned}$$

Essentially, expression for slopes are splitted up at interval midpoint  $f(y(x_n + h/2), x_n + h/2)$  into two – one *predicts* the tangent at the interval and the later *corrects* it. Define the following,

$$\begin{aligned}
 \kappa_1 &= h f(y(x_n), x_n) & \kappa_2 &= h f(y(x_n) + \kappa_1/2, x_n + h/2) \\
 \kappa_3 &= h f(y(x_n) + \kappa_2/2, x_n + h/2) & \kappa_4 &= h f(y(x_n) + \kappa_3, x_n + h)
 \end{aligned}$$

Combine these to form the [RK4](#) solution

$$y(x_n + h) = y(x_n) + \frac{1}{6} (\kappa_1 + 2\kappa_2 + 2\kappa_3 + \kappa_4) + \mathcal{O}(h^5)$$

The error  $\mathcal{O}(h^5)$  follows from maximum error bound of Simpson and allow us to use relative coarser interval to arrive at very precise solution.

## Coupled ODE

Numerical methods like Euler and Runge-Kutta are all applied to first order ODE.

To solve higher order ODE : convert to coupled first order ODE. For instance, SHO

$$\begin{aligned}\frac{d^2x}{dt^2} &= -\mu \frac{dx}{dt} - \omega^2 x, \text{ with } x(t=0) = x_0, \quad v(t=0) = \frac{dx}{dt} \Big|_{t=0} = v_0 \\ \Rightarrow v &= \frac{dx}{dt} \text{ with } v(t=0) = v_0 \\ \frac{dv}{dt} &= -\mu v - \omega^2 x \text{ with } x(t=0) = x_0\end{aligned}$$

In addition, there can be just a set of coupled first order ODEs. For instance, Lorentz equations,

$$\frac{dx}{dt} = \sigma(y - x), \quad \frac{dy}{dt} = x(\rho - z) - y, \quad \frac{dz}{dt} = xy - \beta z$$

The equations relate the properties of a two-dimensional fluid layer uniformly warmed from below and cooled from above, and  $\sigma, \rho, \beta$  are three parameters whose certain values give rise to chaotic behavior.

The *RK4* for damped SHO takes the following appearance

```
k1x = dt*dxdt(x,v,t);
k1v = dt*dvdt(x,v,t);

k2x = dt*dxdt(x+k1x/2,v+k1v/2,t+dt/2);
k2v = dt*dvdt(x+k1x/2,v+k1v/2,t+dt/2);

⋮ = ⋮
x += (k1x + 2*k2x + 2*k3x + k4x)/6;
v += (k1v + 2*k2v + 2*k3v + k4v)/6;
t += dt;
```

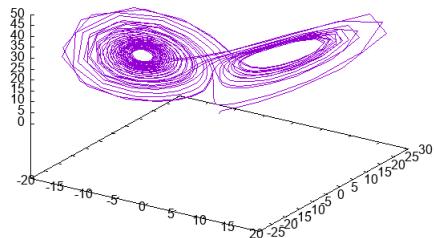
For Lorentz equations, the expressions are similar to SHO

```
k1x = dt*dxdt(x,y,z,t);
k1y = dt*dydt(x,y,z,t);
k1z = dt*dzdt(x,y,z,t);

k2x = dt*dxdt(x+k1x/2,y+k1y/2,z+k1z/2,t+dt/2);
k2y = dt*dydt(x+k1x/2,y+k1y/2,z+k1z/2,t+dt/2);
k2z = dt*dzdt(x+k1x/2,y+k1y/2,z+k1z/2,t+dt/2);
etc.
```

Just for fun : for  $\sigma = 10, \rho = 28, \beta = 8/3$ , the 3-dim plot of the solution shows the famous Lorentz attractor

'rk4lorentz.dat' u 2:3:4



Generalizing RK4 to  $n$  coupled first order ODE

$$\left. \begin{array}{lcl} \frac{dy_1}{dx} & = & f_1(y_1, y_2, \dots, y_n, x) \\ \frac{dy_2}{dx} & = & f_2(y_1, y_2, \dots, y_n, x) \\ \vdots & & \\ \frac{dy_n}{dx} & = & f_n(y_1, y_2, \dots, y_n, x) \end{array} \right\} \equiv \frac{d\vec{y}}{dx} = \vec{f}(\vec{y}, x)$$

where  $\vec{y} = (y_1, y_2, \dots, y_n)$  and  $\vec{f} = (f_1, f_2, \dots, f_n)$ . Vector sign simply implies collection of variables. In such case the **RK4** equations take the forms,

$$\begin{aligned}\vec{\kappa}_1 &= h \vec{f}(\vec{y}_i, x_i) \\ \vec{\kappa}_2 &= h \vec{f}(\vec{y}_i + \vec{\kappa}_1/2, x_i + h/2) \\ \vec{\kappa}_3 &= h \vec{f}(\vec{y}_i + \vec{\kappa}_2/2, x_i + h/2) \\ \vec{\kappa}_4 &= h \vec{f}(\vec{y}_i + \vec{\kappa}_3, x_i + h) \\ \vec{y}_{i+h} &= \vec{y}_i + \frac{1}{6} [\vec{\kappa}_1 + 2\vec{\kappa}_2 + 2\vec{\kappa}_3 + \vec{\kappa}_4]\end{aligned}$$

where  $\vec{y}_i$  are the values at the  $i$ -th interval boundary. Above set of equations have to be read only in terms of components and not as vector equations.

Use **RK4** to solve the following :

$$\begin{array}{lll} \dot{x} & = & -5x + 5y \\ \dot{y} & = & 14x - 2y - zx \\ \dot{z} & = & -3z + xy \end{array} \quad \begin{array}{lll} \dot{x} & = & -2y \\ \dot{y} & = & x + z^2 \\ \dot{z} & = & 1 + y - 2z \end{array}$$

## Boundary Value Problem : Shooting method

Many problems in physics are **boundary value problems**. Like Laplace equation in electrostatics or, more famously, Schrödinger equations.

In boundary value problems, we have conditions specified at two different space (and/or time) points. We can have either

$$\begin{array}{lll} \text{Dirichlet condition} & : & y(x_0) = Y_0 \quad \text{and} \quad y(x_N) = Y_N \\ \text{Neumann condition} & : & y'(x_0) = Y'_0 \quad \text{and} \quad y'(x_N) = Y'_N \end{array}$$

$$\begin{aligned} & \frac{d^2y}{dx^2} = f(x, y, y') \quad \text{where } a \leq x \leq b \quad \text{and} \quad y(a) = \alpha, \quad y(b) = \beta \\ \Rightarrow & \frac{dy}{dx} = z \quad \text{with} \quad y(a) = \alpha \\ & \frac{dz}{dx} = \frac{d^2y}{dx^2} = f(x, y, z) \quad \text{with} \quad z(a) = \zeta_h \quad (\text{guess}) \end{aligned}$$

the slope  $z(a) = \zeta_h$  at  $x = a$  is a guess. Solve the coupled ODEs with initial values  $y(a) = \alpha$  and  $z(a) = \zeta_h$ .

Solution obtained at end point  $x_N$  is compared with the boundary condition  $y(b) = \beta$ . If  $y_{\zeta_h}(b) = \beta$  within tolerance then the ODE is solved.

Suppose  $y_{\zeta_h}(b) \neq \beta$  but  $y_{\zeta_h}(b) > \beta$ . Change the guess initial value  $\zeta_h \rightarrow \zeta_l$  of course and solve it again.

Unless the choice lands bang on the solution, choose  $\zeta_l$  such that  $y_{\zeta_l}(b) < \beta$  implying

$$\zeta_l < z(a) < \zeta_h$$

Use [Lagrange's interpolation formula](#) to choose the next  $z(a) = \zeta$

$$\zeta = \zeta_l + \frac{\zeta_h - \zeta_l}{y_{\zeta_h}(b) - y_{\zeta_l}(b)} (y(b) - y_{\zeta_l}(b))$$

$z(a) = \zeta$  is our new guess value for initial slope and chances are this choice will lead us to the solution of the ODE i.e.  $y_{\zeta}(b) \approx \beta$ .

If not, go through the above procedure until  $y_{\zeta}(b)$  converges to  $\beta$ .

Let us study the following example,

$$\frac{d^2y}{dx^2} = 2y \quad \text{with} \quad y(x = 0.0) = \alpha = 1.2, \quad y(x = 1.0) = \beta = 0.9$$

Analytical solution of the above ODE is

$$y(x) = C_1 e^{\sqrt{2}x} + C_2 e^{-\sqrt{2}x}, \quad \text{where} \quad C_1 = 0.157, \quad C_2 = 1.043$$

Let our initial guess slope is  $z(x = 0.0) = -1.5$ . RK4 returns

$$y(x = 1.0) = 0.5614 < \beta = 0.9 \Rightarrow \zeta_l = -1.5, y_{\zeta_l}(1.0) = 0.5614$$

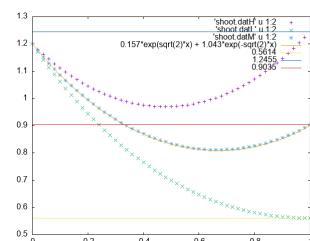
Next try  $z(x = 0.0) = -1.0$ ,

$$y(x = 1.0) = 1.2455 > \beta = 0.9 \Rightarrow \zeta_h = -1.0, y_{\zeta_h}(1.0) = 1.2455$$

Use Lagrange's linear interpolating formula,

$$\zeta = -1.5 + \frac{-1.0 - (-1.5)}{1.2455 - 0.5614} \times (0.9 - 0.5614) = -1.2525$$

Using  $z(x = 0.0) = -1.2525$ , we obtain  $y_{\zeta}(x = 1.0) = 0.9035 \approx \beta = 0.9$  thus solving the ODE. A graphical view of the process



## Partial differential equations

Many equations in physics are partial differential equations – Maxwell equations, Laplace and Poisson equations, wave equations, Schrödinger equation, diffusion equation and so on.

A general linear partial differential equation in 2-dimension *i.e.* second-order in two independent variables reads,

$$A(x, y) \frac{\partial^2 u}{\partial x^2} + B(x, y) \frac{\partial^2 u}{\partial x \partial y} + C(x, y) \frac{\partial^2 u}{\partial y^2} = F \left( x, y, u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y} \right)$$

Most common method employed to solve PDE is *finite difference*, converting *derivatives to differences*.

Two categories of method exist – *explicit* and *implicit*.

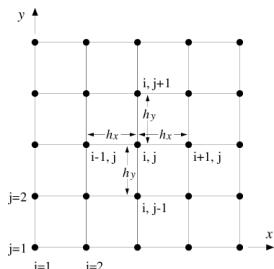
Consider *explicit* scheme in **1+1** dimension diffusion or heat equation,

$$\frac{\partial^2 u(x, t)}{\partial x^2} = \frac{\partial u(x, t)}{\partial t} \quad \equiv \quad u_{xx} = u_t$$

initial conditions at  $t = 0$  and boundary conditions at later time  $t > 0$ ,

$$\begin{aligned} u(x, 0) &= g(x) \quad \text{for } 0 < x < L \\ u(0, t) &= a(t) \quad \text{for } t \geq 0 \\ u(L, t) &= b(t) \quad \text{for } t \geq 0 \end{aligned}$$

Discretize space and time such that  $\Delta x = h_x$ ,  $\Delta t = h_t$  are differences between two space and time points respectively. (Take  $y \equiv t$ ,  $h_y \equiv h_t$ )



Position after  $i$  steps and time at  $j$  step are given by

$$\begin{cases} x_i = ih_x & 0 \leq i \leq n+1 \\ t_i = jh_t & j \geq 0 \end{cases}$$

Discretized forward derivatives for explicit scheme are

$$\begin{aligned}
 u_t &\approx \frac{u(x, t + h_t) - u(x, t)}{h_t} \equiv \frac{u(x_i, t_j + h_t) - u(x_i, t_j)}{h_t} = \frac{u_{i,j+1} - u_{i,j}}{h_t} \\
 u_{xx} &\approx \frac{u(x + h_x, t) + u(x - h_x, t) - 2u(x, t)}{h_x^2} \\
 &\equiv \frac{u(x_i + h_x, t_j) + u(x_i - h_x, t_j) - 2u(x_i, t_j)}{h_x^2} = \frac{u_{i+1,j} + u_{i-1,j} - 2u_{i,j}}{h_x^2}
 \end{aligned}$$

Define  $\alpha = h_t/h_x^2$ , results in explicit scheme

$$\begin{aligned} \frac{u_{i,j+1} - u_{i,j}}{h_t} &= \frac{u_{i+1,j} + u_{i-1,j} - 2u_{i,j}}{h_x^2} \\ u_{i,j+1} &= \alpha (u_{i+1,j} + u_{i-1,j}) + (1 - 2\alpha) u_{i,j} \end{aligned}$$

Given initial values  $u_{i,0} = g(x_i)$ , after one time step we get  $u_{i,1}$ ,

$$\begin{aligned} u_{i,1} &= \alpha (u_{i+1,0} + u_{i-1,0}) + (1 - 2\alpha) u_{i,0} \\ &= \alpha (g(x_{i+1}, 0) + g(x_{i-1}, 0)) + (1 - 2\alpha) g(x_{i,0}) \end{aligned}$$

For simplicity and without loss of generality, consider  $a(t) = b(t) = 0$  implying  $u_{0,i} = u_{L-n+1,i} = 0$  where the interval  $[0, L]$  is divided in  $n$  parts.

Then a vector  $v_j$  at the time  $t_j = j h_t$  is defined as,

$$V_j = \begin{pmatrix} u_{1,j} \\ u_{2,j} \\ \vdots \\ u_{n,j} \end{pmatrix}$$

with  $V_0 \equiv \mu_{i,0} = g(x_i)$ .

Therefore, the solution at a given time slice  $V_{j+1}$  proceeds as

$$V_{j+1} = A V_j \quad \text{where } A = \begin{pmatrix} 1-2\alpha & \alpha & 0 & 0 & \cdots \\ \alpha & 1-2\alpha & \alpha & 0 & \cdots \\ 0 & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \alpha & 1-2\alpha \end{pmatrix}$$

This implies that the solution is evolving in time using matrix-vector multiplication till the end of time, say  $u_{i,T}$

$$V_{j+1} = A V_j = A^2 V_{j-1} = \cdots = A^{j+1} V_0$$

Like explicit Euler, it too has weak stability condition given by  $\alpha = h_t/h_x^2 \leq 0.5$  and is overcome by implicit scheme using backward derivative in time, keeping second derivative unchanged.

$$\begin{aligned} u_t &\approx \frac{u_{i,j} - u_{i,j-1}}{h_t} \\ u_{i,j-1} &= -\alpha (u_{i+1,j} + u_{i-1,j}) + (1 + 2\alpha) u_{i,j} \end{aligned}$$

The corresponding evolution equation for implicit scheme is

$$V_{j-1} = A V_j \text{ where } A = \begin{pmatrix} 1+2\alpha & -\alpha & 0 & 0 & \cdots \\ -\alpha & 1+2\alpha & -\alpha & 0 & \cdots \\ 0 & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & -\alpha & 1+2\alpha \end{pmatrix}$$

$$V_j = A^{-1} V_{j-1} = A^{-2} V_{j-2} = \cdots = A^{-j} V_0$$

**Problem :** Solve the 1-dimension heat equation  $u_{xx} = u_t$  over a metal rod of length 2 units, with the initial conditions,

$$\begin{aligned} u(0, t) &= 0^\circ\text{C} = u(2, t) & \text{for } 0 \leq t \leq 4 \\ u(x, 0) &= 20 |\sin(\pi x)|^\circ\text{C} & \text{for } 0 \leq x \leq 2 \end{aligned}$$

Use *explicit scheme* taking number of position grid  $nx = 20$  and time grid  $nt = 5000$ . Show the temperature profile across the length of the rod at time steps 0, 10, 20, 50, 100, 200, 500 and 1000 in a plot. It should look something like

