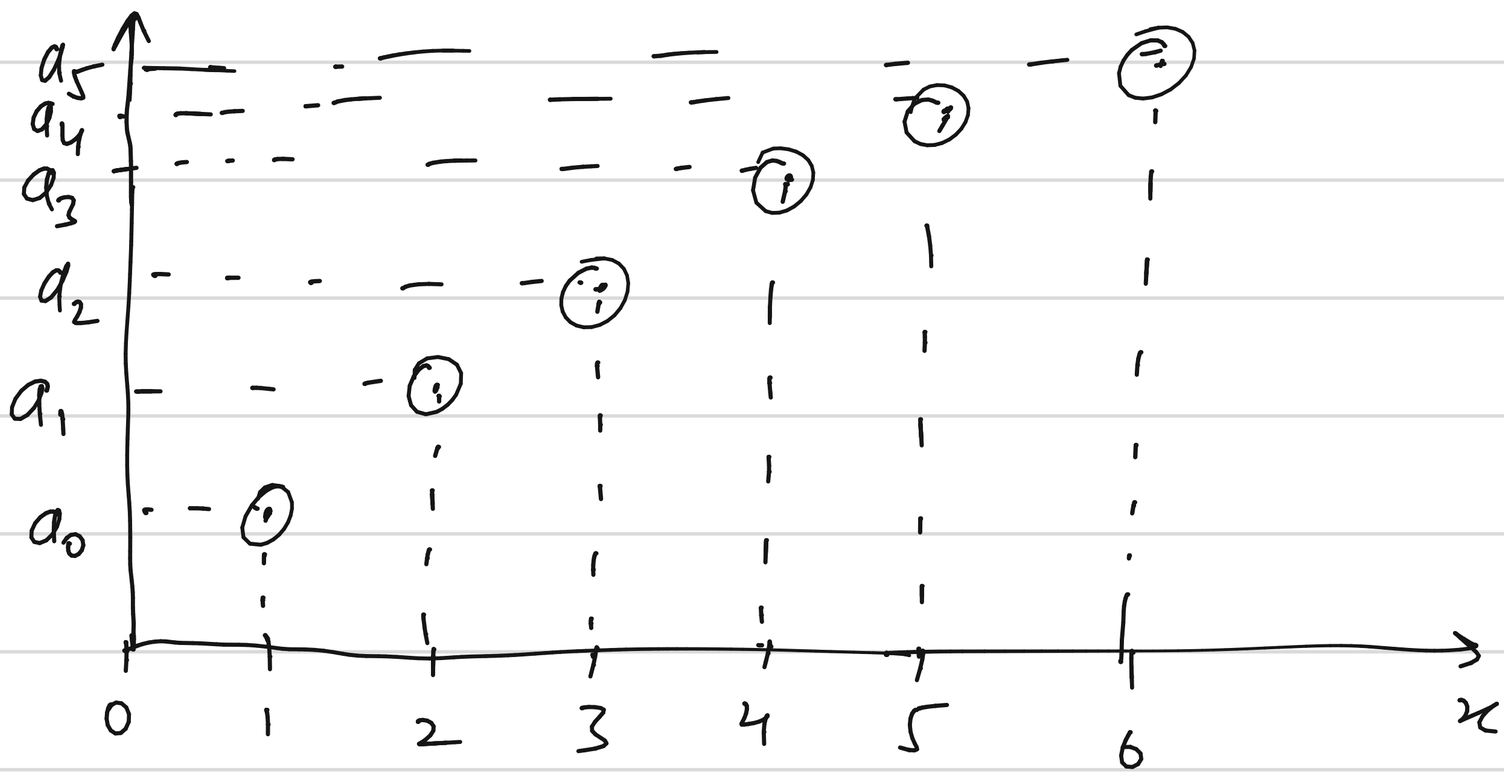
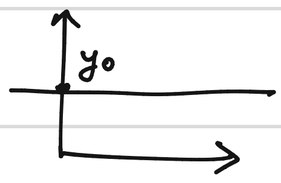
Assignment -1

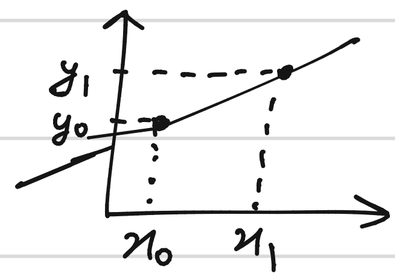
Newton's divided-difference Method :



① 1 pt : $f_0(x) = a_0 \quad (= y_0)$



② 2 pts : $f_1(x) = f_0(x) + a_1(x - x_0)$



$\frac{y - y_0}{x - x_0} = \frac{y_0 - y_1}{x_0 - x_1}$

At $x = x_0$: $f_1(x_0) = f_0(x_0)$

$= a_0 \quad (= y_0)$

At $x = x_1$: $\frac{f_1(x_1) - f_0(x_1)}{x_1 - x_0} = a_1$

$\Rightarrow \boxed{\frac{y_1 - y_0}{x_1 - x_0} = a_1}$ ✓ (We calculate a_1)

$\therefore$ We now know $f_1(x)$ ✓

③ 3 pts : $f_2(x) = f_1(x) + a_2(x - x_0)(x - x_1)$

At $x = x_0$: $f_2(x_0) = f_1(x_0) = y_0$ ✓

At $x = x_1$: $f_2(x_1) = f_1(x_1) = y_1$ ✓

At $x = x_2$: $\boxed{a_2 = \frac{f_2(x_2) - f_1(x_2)}{(x_2 - x_0)(x_2 - x_1)}}$ ✓ (We calculate a_2)

$\therefore$ We now know $f_2(x)$ ✓

④ 4 pts : ⋮

Continues ...

∴ for 6 pts

$$f_5(x) = f_4(x) + a_5(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)$$

$$= f_3(x) + a_4(x-x_0)(x-x_1)(x-x_2)(x-x_3) + a_5(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)$$

$$= f_2(x) + a_3(x-x_0)(x-x_1)(x-x_2) + a_4(x-x_0)(x-x_1)(x-x_2)(x-x_3) + a_5(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)$$

$$= f_1(x) + a_2(x-x_0)(x-x_1) + a_3(x-x_0)(x-x_1)(x-x_2) + a_4(x-x_0)(x-x_1)(x-x_2)(x-x_3) + a_5(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)$$

$$= f_0(x) + a_1(x-x_0) + a_2(x-x_0)(x-x_1) + a_3(x-x_0)(x-x_1)(x-x_2) + a_4(x-x_0)(x-x_1)(x-x_2)(x-x_3) + a_5(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)$$

$$= a_0 + a_1(x-x_0) + a_2(x-x_0)(x-x_1) + a_3(x-x_0)(x-x_1)(x-x_2) + a_4(x-x_0)(x-x_1)(x-x_2)(x-x_3) + a_5(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4)$$

Lagrange Interpolation

3deg poly for 4 pts :-

$$f_3(x) = a_0(x-x_1)(x-x_2)(x-x_3) + a_1(x-x_0)(x-x_2)(x-x_3) + a_2(x-x_0)(x-x_1)(x-x_3) + a_3(x-x_0)(x-x_1)(x-x_2)$$

→ For $x = x_0$: $f_3(x_0) = a_0(x_0-x_1)(x_0-x_2)(x_0-x_3)$

$$\Rightarrow a_0 = \frac{y_0}{(x_0-x_1)(x_0-x_2)(x_0-x_3)}$$

→ For $x = x_1$: $f_3(x_1) = a_1(x_1-x_0)(x_1-x_2)(x_1-x_3)$

$$\Rightarrow a_1 = \frac{y_1}{(x_1-x_0)(x_1-x_2)(x_1-x_3)}$$

Similarly, obtain a_2 & a_3 ✓

Assignment-2

Natural cubic spline Interpolation:

- (x_i, y_i) $i = 0, 1, 2, \dots n$ $\Rightarrow (n+1)$ points

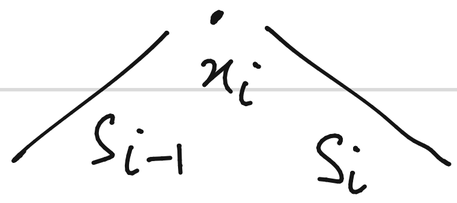
  $\Leftrightarrow$ n cubic polynomials

- $S_0(x) = a_0 x^3 + b_0 x^2 + c_0 x + d_0 \rightarrow$ 4 parameters

- n cubic polynomials $\Rightarrow$ $4n$ parameters

- $$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1] \\ S_1(x), & x \in [x_1, x_2] \\ \vdots \\ S_{n-1}(x), & x \in [x_{n-1}, x_n] \end{cases}$$

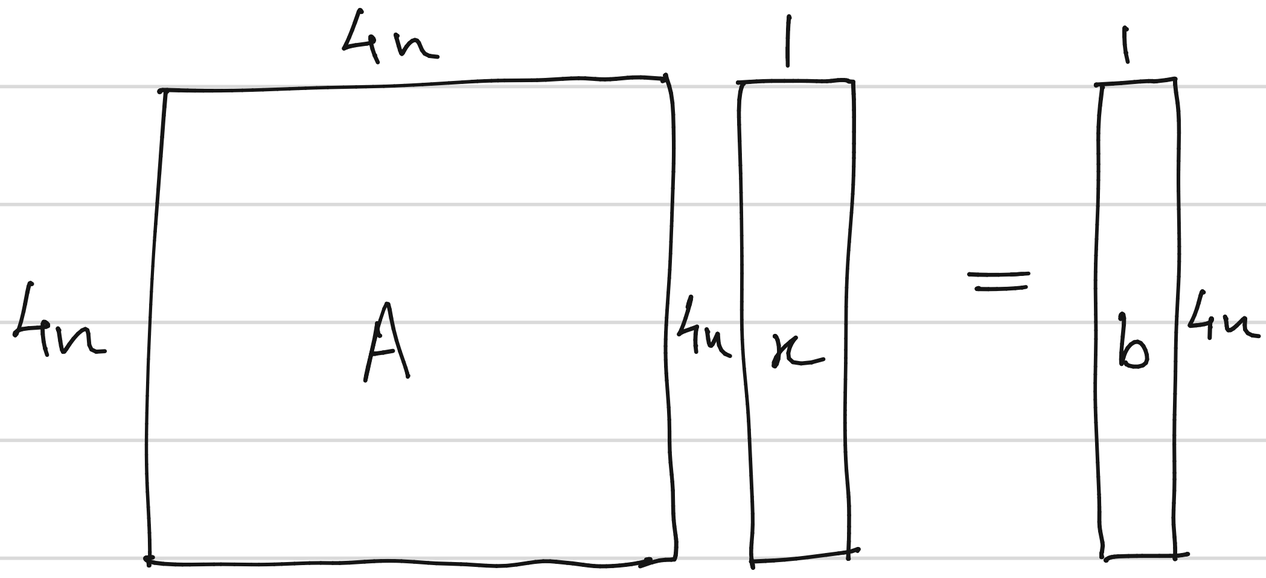
$$S_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i$$
$$S_i'(x) = 3a_i x^2 + 2b_i x + c_i$$
$$S_i''(x) = 6a_i x + 2b_i x$$

- 

$S = S$
$S' = S'$
$S'' = S''$

$$\begin{cases} 2n \begin{cases} S_{i-1}(x_i) = y_i \;;\; S_i(x_i) = y_i \\ n-1 \quad S_{i-1}'(x_i) = S_i'(x_i) \\ n-1 \quad S_{i-1}''(x_i) = S_i''(x_i) \end{cases} \\ \hline 4n-2 \end{cases} \qquad \begin{cases} 1 \quad S_0''(x_0) = 0 \\ 1 \quad S_{n-1}''(x_n) = 0 \\ \hline 2 \end{cases}$$

$\therefore$ $4n$ parameters & $4n$ equations


4m
1
1
4m
A
4m
x
=
b
4m

Assignment -3

We have looking into Polynomial Interpolation since assign 1.

① Direct Interpolation: n pts

↳ (n-1) deg poly.

↳ n coeffs.

We make n eq^n by satisfying n pts into the poly. & solve to get n unknown coeffs.

$$\begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \\ 1 & x_4 & x_4^2 & x_4^3 \end{bmatrix} \begin{bmatrix} d \\ c \\ b \\ a \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

Eg. n = 4

4 pts

↳ 3 deg poly

↳ 4 coeff.

$$A \qquad x \qquad b$$

↳ $\boxed{x = A^{-1} b}$ ✓

② The above method is computationaly expensive, so we use Newton's divided difference interpolation polynomial or simply Newton Polynomial or Lagrange Polynomial as discussed in Assign-1.

③ Sometimes, this single polynomial fitting is not good as this may lead to bumps!. ∴ we use piecewise poly. i.e. Splines.. We discussed cubic spline in Assign-2

④ Sometimes, we need to fit a lower deg poly as compared to the number of points.

The direct interpolation, Newton interpolation or Lagrange interpolation can't be used in this case as the always fit the $(n-1)$ deg poly if n pts are given.
Even if we do direct interpolation, bcz of deg $< (n-1)$ our sys. of eq^n will become over-determined.

So, we use Least Square Method to obtain coeff. such that we minimize the least sq. error.

- The equation which gives the parameters (θ) corresponding to the global minima point directly for a linear regression problem is called the Normal Equation.

- At minima, $\nabla J(\theta) = 0$
  ↳ Solve for $[\theta]$

To keep it clean & easy, we will use matrices.

- Let

$$X = \begin{matrix} x_0 & x_1 & x_2 & \cdots & x_n \end{matrix}$$
$$\begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ 1 & x_1^{(3)} & x_2^{(3)} & \cdots & x_n^{(3)} \\ 1 & x_1^{(4)} & x_2^{(4)} & \cdots & x_n^{(4)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{bmatrix}; \quad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}; \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$$

(design matrix)

$m \times (n+1)$ $\quad$ $m \times 1$ $\quad$ $(n+1) \times 1$

Now, $$J(\theta) = \frac{1}{2m} (X\theta - y)^T (X\theta - y)$$

$X\theta$
$m \times (n+1)$ $\quad$ $(n+1) \times 1$ $\longrightarrow$ $m \times 1$

$(X\theta - y)^T (X\theta - y)$
$1 \times m$ $\quad$ $m \times 1$ $\longrightarrow$ 1×1

$$\nabla J(\theta) = \frac{1}{m} X^T (X\theta - y) = 0$$

$$\Rightarrow X^T (X\theta - y) = 0$$
$$\Rightarrow X^T X\theta - X^T y = 0$$
$$\Rightarrow X^T X\theta = X^T y$$
$$\Rightarrow \boxed{\theta = (X^T X)^{-1} X^T y}$$

Assignment-4

→ Solving BVP ODEs :

To solve ODEs numerically, we use finite differences (FD)

① Forward FD: $f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{\Delta x}$

② Backward FD: $f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{\Delta x}$

③ Central FD:

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2\Delta x}$$

$\Delta x = h$ = step size

Q1 BVP, ODE: $y'' + e^x y' - xy = (-x^2 + 2x - 3)e^{-x} - x + 2$, $0 \le x \le 1$

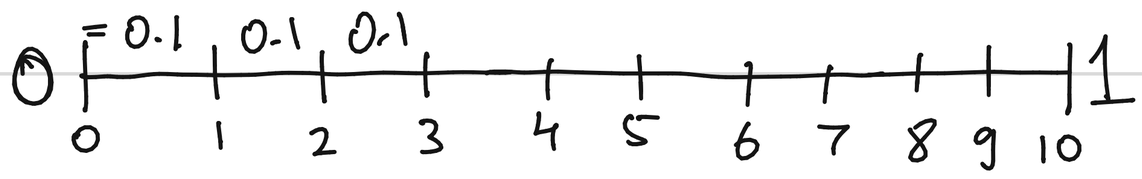
BV $\begin{cases} y(0) = -1 \\ y(1) = 0 \end{cases}$

Let $y_i \equiv y(x_i)$

$$y_i' = \frac{y_{i+1} - y_{i-1}}{2h} \; ; \; y_i'' = \frac{\frac{y_{i+1} - y_i}{h} - \frac{y_i - y_{i-1}}{h}}{h}$$

$$\Rightarrow y_i'' = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}$$

$\Delta x = h$



$(0.1)\,10 = 1$ ✓

$$\therefore \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} + e^{x_i}\left(\frac{y_{i+1} - y_{i-1}}{2h}\right) - x_i y_i = \boxed{(\quad)e^{-x_i} - x_i + 2}$$

$$\Rightarrow \left(1 - \frac{e^{x_i} h}{2}\right) y_{i-1} - (2 + x_i h^2) y_i + \left(1 + \frac{e^{x_i} h}{2}\right) y_{i+1} = (-x_i^2 + 2x_i - 3)e^{-x_i} h^2 - x_i h^2 + 2h^2$$

$$\bigcirc\, y_{i-1} + \bigcirc\, y_i + \bigcirc\, y_{i+1} = \bigcirc$$

$\therefore$ for $i = 1, 2, \ldots 9$

$$\begin{cases} 0\,y_0 + 0\,y_1 + 0\,y_2 = 0 \\ 0\,y_1 + 0\,y_2 + 0\,y_3 = 0 \\ \vdots \\ 0\,y_8 + 0\,y_9 + 0\,y_{10} = 0 \\ y_0 = -1 \\ y_{10} = 0 \end{cases}$$

We formulate it into a matrix:

$$\underset{(n+1)\times(n+1)}{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ \sim & \sim & \sim & 0 & 0 \\ 0 & \sim & \sim & \sim & 0 \\ 0 & 0 & \sim & \sim & \sim \\ - & - & - & - & - \\ - & - & - & - & - \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \underset{(n+1)\times 1}{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_9 \\ y_{10} \end{bmatrix}} = \begin{bmatrix} -1 \\ \vdots \\ 0 \end{bmatrix}$$

→ Shooting Method:

We have seen how to solve BVP. We can also solve IVP using various methods, one of them being Forward Euler. In IVP y & y' at initial pt are given.

In shooting method, we convert a BVP to an IVP and solve it using any method used to solve IVP.

- We guess a value for y' at initial pt, integrate & get the end BC. If it is not close to the given BC, we adjust the guess and repeat.

Q3 $y'' - \frac{3}{x}y' + \frac{3}{x^2}y = 2x^2e^x \quad ; \quad 1 \le x \le 2$

BC $\begin{cases} y(1) = 0 \\ y(2) = 4e^2 \end{cases}$ Lets take $\underline{y'(1) = p}$

$\therefore \; y(1) = 0 \;$ & $\; y'(1) = p \;$: IVP

→ Lets use **Forward Euler Method** to solve this IVP:

① First convert the eq^n into 1st order ODEs:

$$\text{Let} \begin{cases} z_1 = y \\ z_2 = y' \end{cases} \Rightarrow \begin{cases} z_1' = z_2 \\ z_2' = \underbrace{\frac{3}{x}z_2 - \frac{3}{x^2}z_1 + 2x^2e^x}_{\text{✓}} \end{cases} \text{✓}$$

② $z_1(1) = 0$, $z_2(1) = p$

$$\begin{cases} z_1(1+\Delta t) = z_1(1) + \Delta t \cdot z_2(1) \\ z_2(1+\Delta t) = z_2(1) + \Delta t \cdot z_2'(1) \end{cases}$$

and so on...

③ We get $z_1(2)$ and compare it with $4e^2$.

④ Repeat till they get close with a new 'p'.

Q4 Non-linear ?

Assignment-5

Solving PDEs :

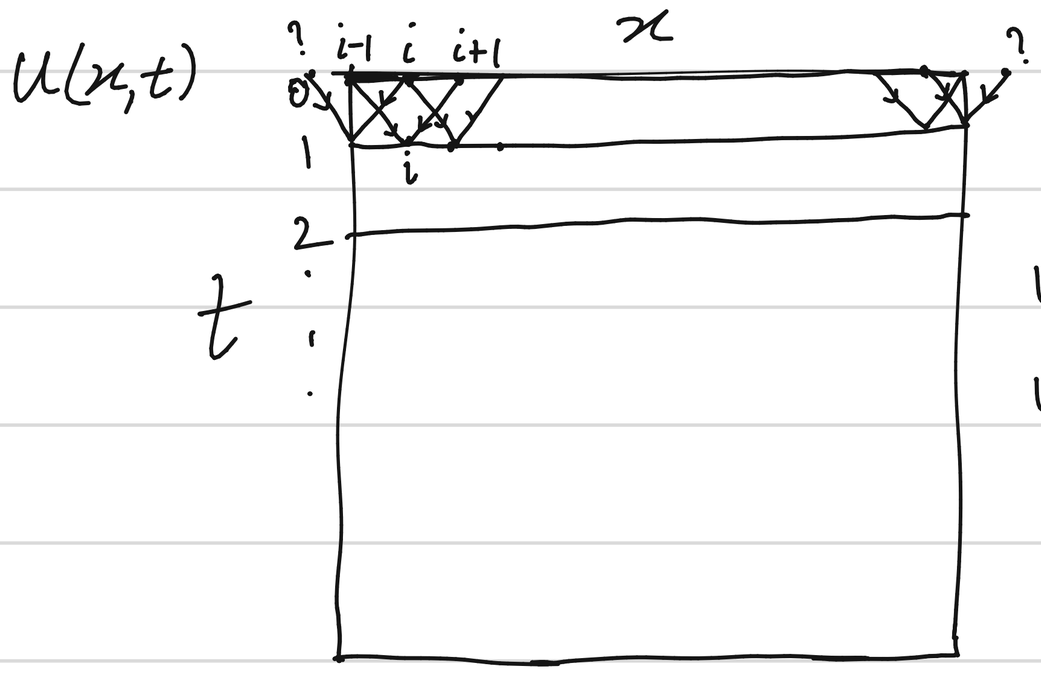
→ Lex-Friedrich Method

- $\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + \frac{f(u_{i+1}^n) - f(u_{i-1}^n)}{2\Delta x} = 0$$

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

Lex-Fred ↦ $$u_i^{n+1} = \left(\frac{u_{i+1}^n + u_{i-1}^n}{2}\right) - \frac{\Delta t}{2\Delta x}\left(f(u_{i+1}^n) - f(u_{i-1}^n)\right)$$



$u_{i-2}^t = u_{n-1}^t$ (Periodic Boundary conditions)

$u_{n+1}^t = u_0^t$

→ Now using initial condition given at $t=0$ i.e. $u(x,0)$, calculate u for further time steps using the formula.

Assignment - 6 :

Neumann Boundary Condition: $u_1 = u_0$

$u_{n+1} = u_n$

- $\frac{\partial u}{\partial t} + \frac{\partial F(u)}{\partial x} = 0$

$$U = \begin{bmatrix} h \\ hu \end{bmatrix} \qquad F = \begin{bmatrix} hu \\ hu^2 + \frac{1}{2}gh^2 \end{bmatrix}$$

Q1 Jacobian:

$$U = \begin{pmatrix} h \\ v \end{pmatrix} \begin{matrix} -x \\ -y \end{matrix} \qquad F = \begin{pmatrix} v \\ \frac{v^2}{h} + \frac{1}{2}gh^2 \end{pmatrix} \begin{matrix} -F_1 \\ -F_2 \end{matrix}$$

	x	y
F_1	$\frac{\partial F_1}{\partial x}$	$\frac{\partial F_1}{\partial y}$
F_2	$\frac{\partial F_2}{\partial x}$	$\frac{\partial F_2}{\partial y}$

Let $hu = v$ and now h & v are considered indep.

$$\frac{\partial F_1}{\partial x} = \frac{\partial v}{\partial h} = 0 \quad ; \quad \frac{\partial F_1}{\partial y} = \frac{\partial v}{\partial v} = 1$$

$$\frac{\partial F_2}{\partial x} = \frac{\partial\left(\frac{v^2}{h} + \frac{1}{2}gh^2\right)}{\partial h} = -\frac{v^2}{h^2} + gh \qquad \frac{\partial F_2}{\partial y} = \frac{\partial(\ )}{\partial v} = \frac{2v}{h} = 2u$$

$$= -\frac{h^2u^2}{h^2} + gh$$

$$= gh - u^2$$

$$\therefore \quad J = \begin{bmatrix} 0 & 1 \\ gh - u^2 & 2u \end{bmatrix}$$

- Stability condition:

$$\Delta t = \frac{(0.8)\,\Delta x}{\max f}$$

We adapt Δt according to this condition.

- $$\begin{cases} \dfrac{\partial h}{\partial t} + \dfrac{\partial (hu)}{\partial x} = 0 & \text{—①} \\ \dfrac{\partial (hu)}{\partial t} + \dfrac{\partial}{\partial x}\left(hu^2 + \dfrac{1}{2}gh^2\right) = 0 & \text{—②} \end{cases}$$

$$\rightarrow \begin{cases} h_i^{n+1} = \dfrac{1}{2}\left(h_{i+1}^n + h_{i-1}^n\right) - \dfrac{\Delta t}{2\Delta x}\left(h_{i+1}^n u_{i+1}^n - h_{i-1}^n u_{i-1}^n\right) \\ h_i^{n+1} u_i^{n+1} = \dfrac{1}{2}\left(h_{i+1}^n u_{i+1}^n + h_{i+1}^n u_{i+1}^n\right) - \dfrac{\Delta t}{2\Delta x}\left(h_{i+1}^n \left(u_{i+1}^n\right)^2 + \dfrac{1}{2}g\left(h_{i+1}^n\right)^2 - h_{i-1}^n \left(u_{i-1}^n\right)^2 - \dfrac{1}{2}g\left(h_{i-1}^n\right)^2\right) \end{cases}$$

$$\rightarrow u_i^{n+1} = \frac{\dfrac{1}{2}\left(h_{i+1}^n u_{i+1}^n + h_{i+1}^n u_{i+1}^n\right) - \dfrac{\Delta t}{2\Delta x}\left(h_{i+1}^n \left(u_{i+1}^n\right)^2 + \dfrac{1}{2}g\left(h_{i+1}^n\right)^2 - h_{i-1}^n \left(u_{i-1}^n\right)^2 - \dfrac{1}{2}g\left(h_{i-1}^n\right)^2\right)}{h_i^{n+1}}$$

Assignment - 7

Same as assignment - 6

Q1

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} = 0$$

$$U = \begin{bmatrix} \rho \\ \rho u \end{bmatrix} \qquad F = \begin{bmatrix} \rho u \\ \rho u^2 \end{bmatrix}$$

Let $\rho u = v$

$$\therefore U = \begin{pmatrix} \rho \\ v \end{pmatrix} \qquad F = \begin{pmatrix} v \\ \frac{v^2}{\rho} \end{pmatrix}$$

$$\therefore \frac{\partial F_1}{\partial x} = \frac{\partial v}{\partial \rho} = 0 \; ; \; \frac{\partial F_1}{\partial y} = \frac{\partial v}{\partial v} = 1$$

$$\frac{\partial F_2}{\partial x} = \frac{\partial v^2/\rho}{\partial \rho} = \frac{-v^2}{\rho^2} \; ; \; \frac{\partial F_2}{\partial y} = \frac{\partial v^2/\rho}{\partial v} = \frac{2v}{\rho} = 2u$$

$$= -u^2$$

$$\therefore J = \begin{bmatrix} 0 & 1 \\ -u^2 & 2u \end{bmatrix}$$

Assignment - 8 : (Explicit Method)

- $$\frac{\partial^2 u}{\partial x^2} = \frac{\partial u}{\partial t} \quad ; \; 0 < x < 1 \, , \; t > 0$$

FTCS

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

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

Assignment - 9 :

↳ Crank - Nicolson Method (Implicit Method)

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

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

In this explicit formula, replace $(u_{i+1}^n - 2u_i^n + u_{i-1}^n)$ with a forward time avg.

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

Let $\frac{\alpha \Delta t}{(\Delta x)^2} = \lambda$

$$\Rightarrow \boxed{(-\lambda)\, u_{i-1}^{n+1} + (2(1+\lambda))\, u_i^{n+1} + (-\lambda)\, u_{i+1}^{n+1} = (\lambda)\, u_{i-1}^n + (2(1-\lambda))\, u_i^n + (\lambda)\, u_{i+1}^n}$$

Now, solve it like BVP in assign-4. for each time step increment.

We formulate it into a matrix :

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ \sim & \sim & \sim & \cdots & 0 \\ 0 & \sim & \sim & \cdots & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}_{11 \times 11} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_9 \\ u_{10} \end{bmatrix}_{11 \times 1} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}_{11 \times 1}$$

$$\begin{cases} 0\, u_0 + 0\, u_1 + 0\, u_2 = \sim \\ 0\, u_1 + 0\, u_2 + 0\, u_3 = \sim \\ \vdots \\ 0\, u_8 + 0\, u_9 + 0\, u_{10} = \sim \end{cases}$$

$$u_0 = \sim$$

$$u_{10} = \sim$$