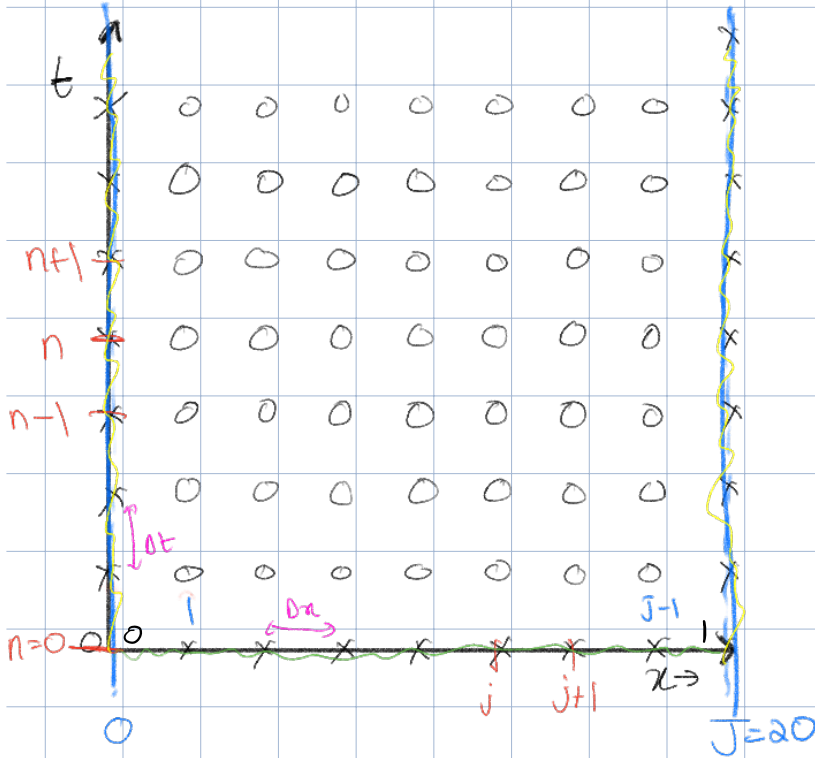


Question 1 (explicit scheme)



$x = \text{knowns from BCs}$

0 = unknown vals

$$x_j = j\alpha \quad ; \quad j = 0, 1, 2, \dots, J$$

$$t_n = n \Delta t \quad ; \quad n = 0, 1, \dots$$

$$u_j^n = v(x_j, t_n)$$

Boundary conditions

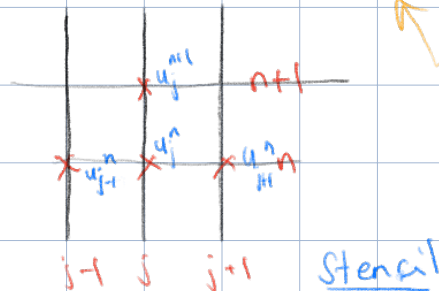
$$u(x, 0) = u_0(x)$$

$$u(0, t) = u(J, t) = 0$$

So there are $J+1$ grid points

for the explicit method

$$u_j^{n+1} = u_j^n + \mu (u_{j+1}^n - 2u_j^n + u_{j-1}^n) \quad (+)$$



$$v = \frac{Dt}{Dn^2}$$

for this scheme,

$$J=20 \quad \Delta x = 0.05$$

$$\Delta t_1 = 0.002 \quad \Delta t_2 = 0.0013$$

$$u^0(x) = \begin{cases} 2x & 0 \leq x \leq 1/2 \\ 2-2x & 1/2 \leq x \leq 1 \end{cases}$$

The code for this then use a vector set up in Python, initially finding the vector u_0 using the given initial condition. Then, this vector is used to find the next timestep vector using (+).

$$\text{ie } u_j^1 = u_j^0 + \mu (u_{j+1}^0 - 2u_j^0 + u_{j-1}^0)$$

$$u_j^{50} = u_j^{49} + \mu (u_{j+1}^{49} - 2u_j^{49} + u_{j-1}^{49})$$

=> The code is written in file
"gulana - q4.py"

\Rightarrow Figure 2.2 is proved and
saved in the github folder
as "gulana-q4-dt-0.002.jpeg" for $dt=0.002$
and "gulana-q4-dt-0.0013.jpeg" for $dt=0.0013$

If the fourier analysis is performed
on the explicit scheme dealt with
above, as shown in M&M 2.7,
we find that the stability criterion
is

$$N \leq \frac{1}{2}$$

and thus,

$$N = \frac{\Delta t}{\Delta x^2} \Rightarrow \frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}$$

$$\Rightarrow \Delta t \leq \frac{1}{2} \Delta x^2 \quad (++)$$

if we calculate with $\frac{1}{2} \Delta x^2 = \frac{1}{2} D (0.05)^2 = 1.25 \times 10^{-3}$

condition 1 $\Delta t = 0.0012$

$$0.0012 \leq 0.00125 \text{ (satisfied ++)}$$

→ stable as shown in "gulana-q4-at-0.0012.jpg"

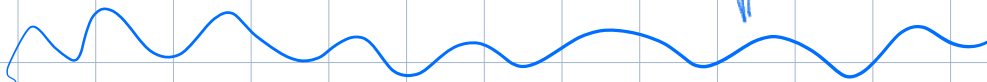
condition 2 $\Delta t = 0.0013$

$$0.0013 > 0.0012$$

↑ does not satisfy (++)

→ thus this solution is unstable, as shown
the figure in file "gulana-q4-at-0.0013.jpg"

extending explicit scheme to
linear advection diffusion:



$$\frac{U_j^{n+1} - U_j^n}{\Delta t} - \alpha(t) \frac{U_{j+1}^n - U_j^n}{\Delta x} - \left(\epsilon \frac{U_j^n - 2U_j^n + U_{j-1}^n}{\Delta x^2} \right) = 0$$

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = +\alpha(t) \frac{U_{j+1}^n - U_j^n}{\Delta x} + \epsilon \left(\frac{U_j^n - 2U_j^n + U_{j-1}^n}{\Delta x^2} \right)$$

$$U_j^{n+1} = U_j^n + \frac{\Delta t}{\Delta x} \alpha(t) U_{j+1}^n - U_j^n + \epsilon \frac{\Delta t}{\Delta x^2} (U_j^n - 2U_j^n + U_{j-1}^n)$$

↑ in my code I will take

$\epsilon = 0.001$ and $\alpha(t) = 1$ and
no. of timesteps (T) = 50

The implementation is shown in code:

"gulana-q4.py" $\left\{ \begin{array}{l} dt = 0.0012, T = 50 \\ \epsilon = 0.001 \\ \alpha(t) = 1, dx = 0.05 \end{array} \right.$

and the graph output is shown in

"gulana-q4 - advection-diffusion.jpeg"