** All codes are wriittern and compiled by Matlab R2017b.*

1 Question 1

Consider the following system of Partial Differential Equations

$$\begin{cases} \dfrac{\partial u(x,t)}{\partial t} = \beta_u \dfrac{\partial^2 u(x,t)}{\partial^2 x} + f(u,v), \text{ in } \Omega = [0,L] \times (0,T] \\ \dfrac{\partial v(x,t)}{\partial t} = \beta_v \dfrac{\partial^2 v(x,t)}{\partial^2 x} - f(u,v), \text{ in } \Omega = [0,L] \times (0,T] \end{cases} \tag{1}$$

with Neumann boundary conditions

$$\begin{cases} \dfrac{\partial u(x,t)}{\partial x} = 0|_{x=0,L} \\ \dfrac{\partial v(x,t)}{\partial x} = 0|_{x=0,L} \end{cases} \tag{2}$$

We first define grids in both the x and t directions, in the usual way:

$$0 = x_0 < x_1 < x_2 < \cdots < x_j < \cdots < x_{N-1} < x_N = L$$

and

$$0 = t_0 < t_1 < t_2 < \cdots < t_i < \cdots < t_{M-1} < t_M = T$$

which is shown in Figure(1)

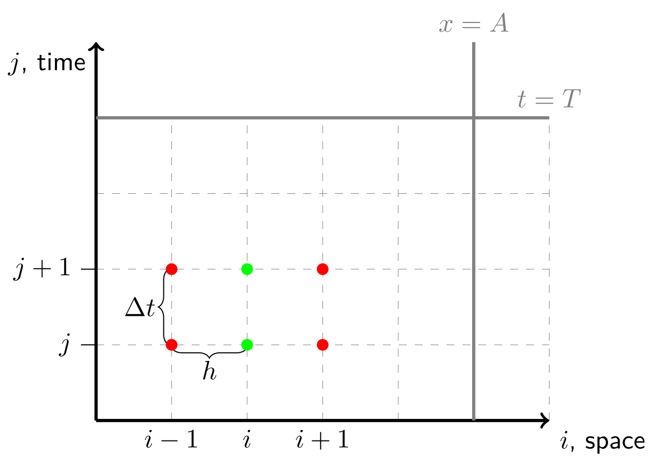


Figure 1: The Crank-Nicolson scheme

By replacing

$$\begin{cases} \dfrac{\partial u}{\partial t} = \dfrac{u_j^{i+1} - u_j^i}{\Delta t} \\ \dfrac{\partial u}{\partial x} = \dfrac{1}{2h^2}\left(u_{j+1}^{i+1} - 2u_j^{i+1} + u_{j-1}^{i+1}\right) + \dfrac{1}{2h^2}\left(u_{j+1}^i - 2u_j^i + u_{j-1}^i\right) \end{cases} \tag{3}$$

We get Crank-Nicolsan scheme for this PDEs problem

$$\begin{cases} u_j^{i+1} - \beta_u \dfrac{\Delta t}{2h^2}\left(u_{j+1}^{i+1} - 2u_j^{i+1} + u_{j-1}^{i+1}\right) = \dfrac{\Delta t}{2h^2}\beta_u\left(u_{j+1}^{i} - 2u_j^{i} + u_{j-1}^{i}\right) + u_j^i + \Delta t f(u,v) \\ v_j^{i+1} - \beta_v \dfrac{\Delta t}{2h^2}\left(v_{j+1}^{i+1} - 2v_j^{i+1} + v_{j-1}^{i+1}\right) = \dfrac{\Delta t}{2h^2}\beta_v\left(v_{j+1}^{i} - 2v_j^{i} + v_{j-1}^{i}\right) + v_j^i - \Delta t f(u,v) \end{cases} \tag{4}$$

where $u_j^i = u(x_j, t_i)$,$v_j^i = v(x_j, t_i)$

If we treat forcing term $f(u,v)$ explicitly, which means $f(u,v) = f\left(u_j^i, v_j^i\right)$, and use first-approach for Neumann boundary conditions

$$\begin{cases} \dfrac{\partial u(x,t)}{\partial x} = 0|_{x=0,L} \Rightarrow u_0^t = u_1^t, u_N^t = u_{N-1}^t \\ \dfrac{\partial v(x,t)}{\partial x} = 0|_{x=0,L} \Rightarrow v_0^t = v_1^t, v_N^t = v_{N-1}^t \end{cases} \tag{5}$$

then we write system equation(3) in matrix form

$$\begin{gathered} A(\beta_u)\begin{pmatrix} u_1^{i+1} \\ u_2^{i+1} \\ u_3^{i+1} \\ \vdots \\ u_{N-2}^{i+1} \\ u_{N-1}^{i+1} \end{pmatrix} = B(\beta_u)\begin{pmatrix} u_1^{i} \\ u_2^{i} \\ u_3^{i} \\ \vdots \\ u_{N-2}^{i} \\ u_{N-1}^{i} \end{pmatrix} + \Delta t\begin{pmatrix} f(u_1^i, v_1^i) \\ f(u_2^i, v_2^i) \\ f(u_3^i, v_3^i) \\ \vdots \\ f(u_{N-2}^i, v_{N-2}^i) \\ f(u_{N-1}^i, v_{N-1}^i) \end{pmatrix} \\ A(\beta_v)\begin{pmatrix} v_1^{i+1} \\ v_2^{i+1} \\ v_3^{i+1} \\ \vdots \\ v_{N-2}^{i+1} \\ v_{N-1}^{i+1} \end{pmatrix} = B(\beta_v)\begin{pmatrix} v_1^{i} \\ v_2^{i} \\ v_3^{i} \\ \vdots \\ v_{N-2}^{i} \\ v_{N-1}^{i} \end{pmatrix} - \Delta t\begin{pmatrix} f(u_1^i, v_1^i) \\ f(u_2^i, v_2^i) \\ f(u_3^i, v_3^i) \\ \vdots \\ f(u_{N-2}^i, v_{N-2}^i) \\ f(u_{N-1}^i, v_{N-1}^i) \end{pmatrix} \end{gathered} \tag{6}$$

where

$$A(\beta) = \begin{pmatrix} 1+\beta\frac{\Delta t}{2h^2} & -\beta\frac{\Delta t}{2h^2} & & & & \\ -\beta\frac{\Delta t}{2h^2} & 1+\beta\frac{\Delta t}{h^2} & -\beta_u\frac{\Delta t}{2h^2} & & & \\ & -\beta\frac{\Delta t}{2h^2} & 1+\beta\frac{\Delta t}{h^2} & -\beta_u\frac{\Delta t}{2h^2} & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\beta\frac{\Delta t}{2h^2} & 1+\beta\frac{\Delta t}{h^2} & -\beta\frac{\Delta t}{2h^2} \\ & & & & -\beta\frac{\Delta t}{2h^2} & 1+\beta\frac{\Delta t}{2h^2} \end{pmatrix}$$

$$B(\beta) = \begin{pmatrix} 1-\beta\frac{\Delta t}{2h^2} & \beta\frac{\Delta t}{2h^2} & & & & \\ \beta\frac{\Delta t}{2h^2} & 1-\beta\frac{\Delta t}{h^2} & \beta\frac{\Delta t}{2h^2} & & & \\ & \beta\frac{\Delta t}{2h^2} & 1-\beta\frac{\Delta t}{h^2} & \beta_u\frac{\Delta t}{2h^2} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \beta\frac{\Delta t}{2h^2} & 1-\beta\frac{\Delta t}{h^2} & \beta_u\frac{\Delta t}{2h^2} \\ & & & & \beta\frac{\Delta t}{2h^2} & 1-\beta\frac{\Delta t}{2h^2} \end{pmatrix}$$

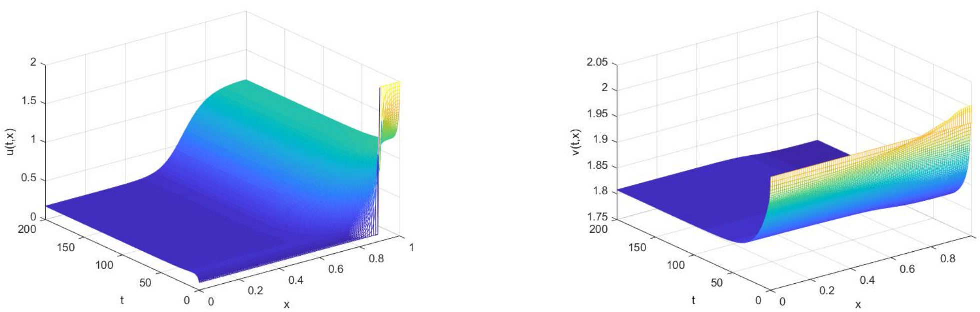
The initial conditon is given by

$$u(x,0) = \begin{cases} 0.1 \text{ for } 0 \le x \le 0.9 \\ 2.0 \text{ for } 0.9 < x \le 1.0 \end{cases} \tag{7}$$

and

$$v(x,0) = \frac{P - \int_0^L u(x,0)\mathrm{d}x}{L} = 1.97 \tag{8}$$

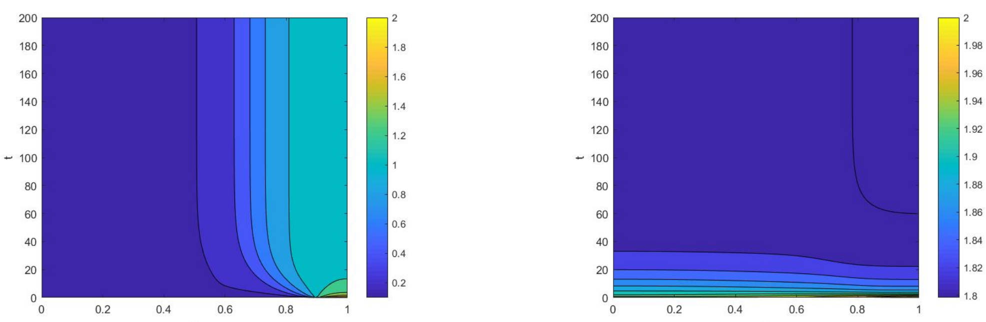
where $P = 2.26, \beta_v = 1/10, \beta_u = 0.01$, using$L = 1, T = 200, h = /99, \Delta t = T/999$ we solve this system using equation(6), the results are shown in Figuer 2, and data are saved in q1 folder/u.xlsx, v.xlsx.



(a) Result of u (b) Result of v

Figure 2: PDEs,Crank-Nicolsan

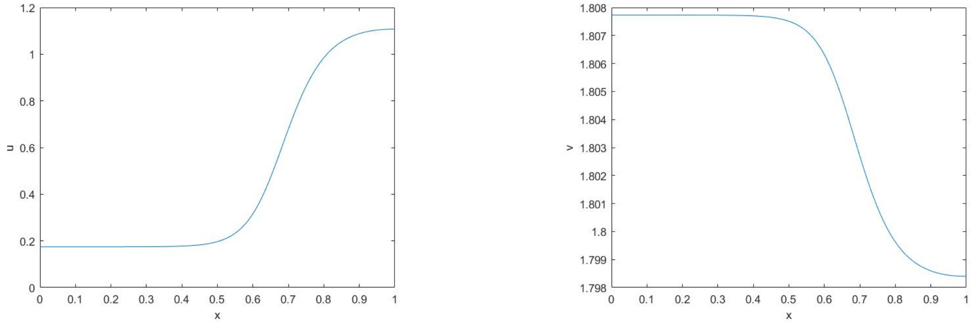
Draw the heatmap on plane (x,t), the results are shown in Figuer 3



(a) u (b) v

Figure 3: heatmap

And at final simulation time t=200, plot u(x,200) and v(x,200), the results are shown in Figure 4



(a) u (b) v

Figure 4: final time simulation

Let $\mathcal{J}_1$ be the index set of all nodes (grid points) for a discretization with $h_1 = L/99$, $\mathcal{J}_2$ for $h_2 = h_1/2$, $\mathcal{J}_3$ for $h_3 = h_2/2$, and $\mathcal{J}_4$ for $h_4 = h_3/2$. Compute

$$e_j = \left| h_{j+1} \sum_{i \in \mathcal{J}_{j+1}} u_{h_{j+1},\Delta t}\left(x_i, T\right) - h_j \sum_{i \in \mathcal{J}_j} u_{h_j,\Delta t}\left(x_i, T\right) \right|, \quad \text{for } 1 \leq j \leq 3 \tag{9}$$

The result is shown below

	h=1/99	h=1/198	h=1/396
e_j	5.364741674559639	2.721062447602719	1.399457366138449
$log(e_j)$	1.679848224794728	1.001022409777639	0.336084565871224

Plot $log(e_j)$ versus $log(h_j)$, the result is shown in Figure 5

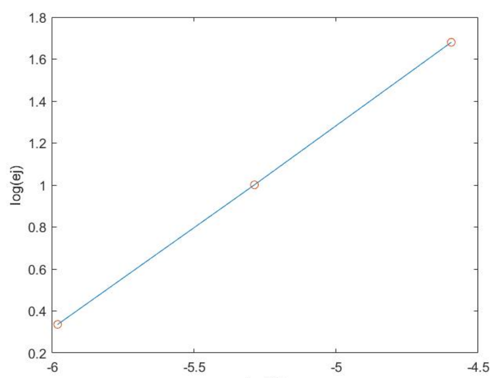


Figure 5: $log(e_j)$ versus $log(h_j)$

We use OLS method to fit the line in Figure 5, we get

$$log(e) = p_1 h + p_2 = 0.9693 log(h) + 6.132$$

Theoretically, Crank-Nicolsan scheme is unconditionally stable, which means

$$\begin{aligned}
log(e_j) &= log\left(\left|h_{j+1}\sum_{i\in\mathcal{J}_{j+1}} u_{h_{j+1},\Delta t}\left(x_i,T\right) - h_j\sum_{i\in\mathcal{J}_j} u_{h_j,\Delta t}\left(x_i,T\right)\right|\right)\\
&= log\left(h_j\left|\frac{1}{2}\sum_{i\in\mathcal{J}_{j+1}} u_{h_{j+1},\Delta t}\left(x_i,T\right) - \sum_{i\in\mathcal{J}_j} u_{h_j,\Delta t}\left(x_i,T\right)\right|\right)\\
&= log(h_j) + log\left(\left|\frac{1}{2}\sum_{i\in\mathcal{J}_{j+1}} u_{h_{j+1},\Delta t}\left(x_i,T\right) - \sum_{i\in\mathcal{J}_j} u_{h_j,\Delta t}\left(x_i,T\right)\right|\right)
\end{aligned}$$

should be equal to

$$log(e_j) = log(h_j) + C,$$

where C is a positive constant number which is independent on h

Actually, parameter $p_1 = 0.9693$ with 95% confidence interval $(0.8958, 1.043)$

2 Question 2

Poisson's equation

$$\begin{cases} -(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}) = f, & (x,y) \in \Omega, \\ u = g, & (x,y) \in \partial\Omega, \end{cases} \tag{10}$$

We first define grids in both the x and y directions, in the usual way:

$$x_0 < x_1 < x_2 < \cdots < x_i < \cdots < x_N < x_{N+1}$$

and

$$y_0 < y_1 < t_2 < \cdots < y_j < \cdots < y_M < y_{M+1}$$

We use central finite differences to approximate this PDE problem, which can be get from Figure 6

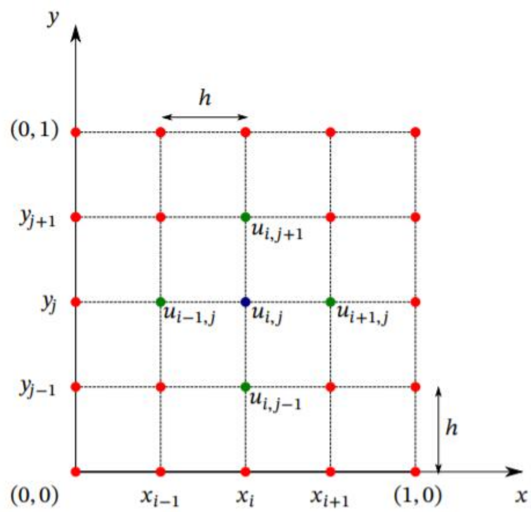


Figure 6: Central finite differences

Then we get

$$-\left[\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}\right] - \left[\frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}\right] = f\left(x_i, y_j\right) \tag{11}$$

which can be simplified somewhat to yield

$$-u_{i+1,j} - u_{i-1,j} + 4u_{i,j} - u_{i,j+1} - u_{i,j-1} = h^2 f_{i,j} \tag{12}$$

where $u_{i,j} = u(x_i, y_j)$, $f_{i,j} = f(x_i, y_j)$

Take M=N=4(5 points at each direction) as example, write equation(12) in matrix form, we get

$$
\left[\begin{array}{ccc|ccc|ccc}
4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\
\hline
-1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\
0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\
0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\
\hline
0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4
\end{array}\right]
\begin{bmatrix} u_{1,1} \\ u_{1,2} \\ u_{1,3} \\ u_{2,1} \\ u_{2,2} \\ u_{2,3} \\ u_{3,1} \\ u_{3,2} \\ u_{3,3} \end{bmatrix}
= h^2 \begin{bmatrix} f_{1,1} \\ f_{1,2} \\ f_{1,3} \\ f_{2,1} \\ f_{2,2} \\ f_{2,3} \\ f_{3,1} \\ f_{3,2} \\ f_{3,3} \end{bmatrix}
+ \begin{bmatrix} g(x_1)+g(y_1) \\ g(x_2) \\ g(x_3)+g(y_1) \\ g(y_2) \\ 0 \\ g(y_2) \\ g(x_1)+g(y_3) \\ g(x_2) \\ g(x_3)+g(y_3) \end{bmatrix} \tag{13}
$$

In order to save the time when N is large, we use sparse command in Matlab to save the system (13). For example,

```
row=[0, 0, 1, 1, 2, 1]+1;
%plus 1 because index in matlab begins from 1 not 0
col=[0, 1, 2, 0, 2, 2]+1;
data=[1, 4, 5, 8, 9, 6];
A=sparse(row,col,data);
B=full(A);%B is matrix form of A
```

output

$$
B = \begin{pmatrix} 1 & 4 & 0 \\ 8 & 0 & 11 \\ 0 & 0 & 9 \end{pmatrix}
$$

which is the same result as Python.

When $u(x, y) = cos(4\pi x)cos(4\pi y)$, $\Omega = [0, 1]^2$

$$
\frac{\partial^2 u}{\partial x^2} = \frac{\partial^2 u}{\partial y^2} = -16\pi^2 \cos(4\pi x)\cos(4\pi y) \tag{14}
$$

then

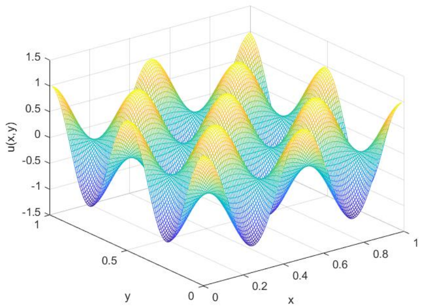
$$
-\Delta u = f(x, y) = 32\pi^2 \cos(4\pi x)\cos(4\pi y) \tag{15}
$$

and

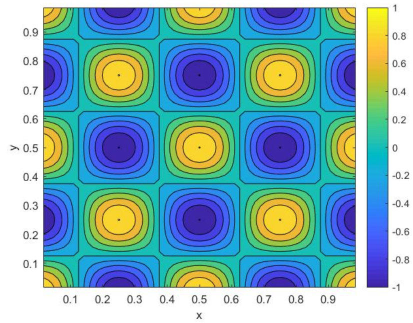
$$
g(z) = cos(4\pi z) \tag{16}
$$

which means $u(x, 0) = u(x, 1) = g(x), u(0, y) = u(1, y) = g(y)$

Using N=64, we solve this PDE by using system (13), result is shown in Figure 7, and data is saved in q2 folder/u64.xlsx



(a) u



(b) heatmap

Figure 7: Poisson N=64

Compute error $\|\boldsymbol{u}-\hat{\boldsymbol{u}}\|_{h,2}$ using function

$$\|\boldsymbol{u}-\hat{\boldsymbol{u}}\|_{h,2}=\left(h\sum_{j=\infty}^{\infty}|\boldsymbol{u}_j-\hat{u}_j|^2\right)^{\frac{1}{2}}$$

The result is shown below

	N=16	N=32	N=64	N=128	N=256	N=512	N=1024
e_j	0.0927164	0.0318224	0.0111657	0.0039401	0.0013924	0.00049222	0.00017402
$log(e_j)$	-2.378210	-3.447585	-4.494907	-5.536544	-6.576744	-7.616585	-8.656336

Plot $log(error)$ versus $log(N)$, the result is shown in Figure 8

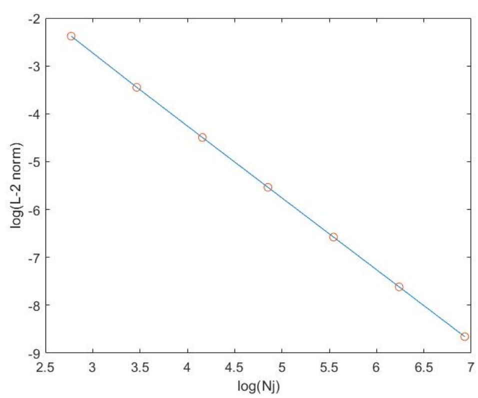


Figure 8: $log(e_j)$ versus $log(h_j)$

Theoretically, central finite defference method is $O\left(h^2\right)$ local truction error since

$$
\begin{aligned}
\tau_{i,j} &= -\left[\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h^2}\right]-\left[\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{h^2}\right]-f\left(x_i,y_j\right)\\
&= -(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+f(x_i,y_j))+\frac{h^2}{12}(\frac{\partial^4 u}{\partial x^4})+O\left(h^4\right)\\
&= O\left(h^2\right)=O\left(\frac{1}{N^2}\right)
\end{aligned}
$$

And the error $\boldsymbol{u}-\hat{\boldsymbol{u}}$ is given by

$$A^h(\boldsymbol{u}-\hat{\boldsymbol{u}})^h=-\tau^h$$

Then $\|\boldsymbol{u}-\hat{\boldsymbol{u}}\|_{h,2}$ is $O\left(h^{1.5}\right)$ error since

$$\|\boldsymbol{u}-\hat{\boldsymbol{u}}\|_{h,2}=h^{\frac{1}{2}}(\sum(\boldsymbol{u}-\hat{\boldsymbol{u}})^2)^{\frac{1}{2}}=O\left(h^{1.5}\right)$$

We use OLS method to fit the line in Figure 8, we get

$$log(error)=p_1 log(N)+p2=-1.507log(N)+1.784$$

where 95$\%$ confidence interval of p_1 is $(-1.515,-1.5)$

3 Question 3

3.1 One-dimensional Hyperbolic PDE

Consider the hyperbolic PDE (scalar transport)

$$\begin{cases} \partial_t u + a\partial_x u = 0, \text{ with } (x,t) \in [0,2\pi] \times (0,1] \text{ (one-dimensional in space),} \\ \qquad u(x,0) = g(x) \end{cases} \tag{17}$$

Theoretically, PDE(17) remains constant along the straight line x(t)=x(0)+at on plane (x,t) since

$$\frac{du}{dt} = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x}\frac{dx}{dt} = 0 \quad \text{ on } (x(t),t)$$

Consequently, it has analytical solution given by

$$u(x,t) = g(x-at), \quad t \geq 0 \tag{18}$$

We first define grids in both the x and t directions. However, the finite difference method for Hyperbolic is different, which is shown in Figure 9:

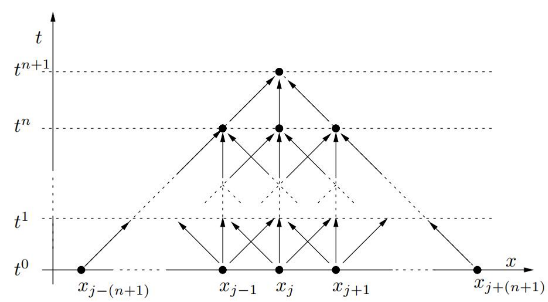


Figure 9: Finite difference method for Hyperbolic

Thus we choose to compute approximation with $(x,t) \in [-2\pi, 4\pi] \times (0,1]$ instead of $(x,t) \in [0,2\pi] \times (0,1]$ to get the approximation with $x \in [0,2\pi]$ when t=1.

Then the grids should be

$$-2\pi = x_0 < x_1 < x_2 < \cdots < x_j < \cdots < x_{N-1} < x_N = 4\pi$$

and

$$0 = t_0 < t_1 < t_2 < \cdots < t_n < \cdots < t_{M-1} < t_M = 1$$

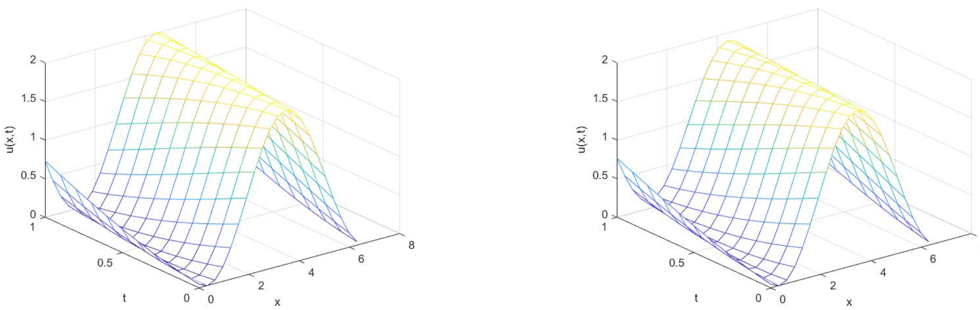
Let $a = 1, g(x) = 1 - \cos x, \Delta x = 2\pi/20$, and $\Delta t = 1/10$, we use Lax-Wendroff scheme and Upwind scheme to solve this PDE problem, which are given by

$$u_j^{n+1} = u_j^n - \frac{\lambda}{2}a\left(u_{j+1}^n - u_{j-1}^n\right) + \frac{\lambda^2}{2}a^2\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right) \qquad \text{(Lax-Wendroff)} \tag{19}$$

$$u_j^{n+1} = u_j^n - \frac{\lambda}{2}a\left(u_{j+1}^n - u_{j-1}^n\right) + \frac{\lambda}{2}|a|\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right) \qquad \text{(Upwind)} \tag{20}$$

where $\lambda = \Delta t/\Delta x$, $u_j^n = u(x_j, t_n)$

The results are shown in Figure 10, data is stored in q3 folder/one dimension folder/ulax.xlsx and uwind.xlsx.



(a) Lax-Wendroff

(b) Upwind

Figure 10: One-dimensional Hyperbolic PDE

We compute L^2 using grid norm function

$$\|\mathbf{u^n}\|_{\Delta,2} = \left(\Delta x \sum_{j=-\infty}^{\infty} \left| u_j^n \right|^2 \right)^{\frac{1}{2}}$$

Then in one dimensional case, the result is shown below

	Lax-Wendroff	Upwind
$\|\mathbf{u^0}\|_{\Delta,2}$	9.4255	9.4255
$\|\mathbf{u^1}\|_{\Delta,2}$	9.4263	9.3608
$\|\mathbf{u^2}\|_{\Delta,2}$	9.4286	9.2996
$\|\mathbf{u^3}\|_{\Delta,2}$	9.4330	9.2424
$\|\mathbf{u^4}\|_{\Delta,2}$	9.4403	9.1900
$\|\mathbf{u^5}\|_{\Delta,2}$	9.4513	9.1431
$\|\mathbf{u^6}\|_{\Delta,2}$	9.4668	9.1020
$\|\mathbf{u^7}\|_{\Delta,2}$	9.4874	9.0674
$\|\mathbf{u^8}\|_{\Delta,2}$	9.5140	9.0393
$\|\mathbf{u^9}\|_{\Delta,2}$	9.5469	9.0180
$\|\mathbf{u^{10}}\|_{\Delta,2}$	9.5864	9.0033

A numerical method for a hyperbolic problem is said to be stable if, for any time T, there exist two constants $C_T > 0$ and $\delta_0 > 0$, such that

$$\|\mathbf{u}^n\|_\Delta \leq C_T \left\|\mathbf{u}^0\right\|_\Delta$$

for any n such that $n\Delta t \leq T$ and for any $\Delta t, \Delta x$ such that $0 < \Delta t \leq \delta_0$, $0 < \Delta x \leq \delta_0$.

Consequently, Upwind method is stable in this case.

3.2 two-dimensional Hyperbolic PDE

Consider the hyperbolic PDE

$$\begin{cases} \partial_t u + \boldsymbol{a} \cdot \nabla u = 0, \text{ with } (x, y, t) \in [0, 2\pi]^2 \times (0, 1] \text{ (two-dimensional in space).} \\ \quad u(x, y, 0) = g(x, y) \end{cases} \tag{21}$$

where $\boldsymbol{a} \cdot \nabla u = a_1 \partial_x u + a_2 \partial_y u$

Theoretically, solution of PDE(21) is given by

$$u(x, y, t) = g(x - a_1 t, y - a_2 t), \quad t \geq 0 \tag{22}$$

Similarly, We first define grids in x, y and t directions.

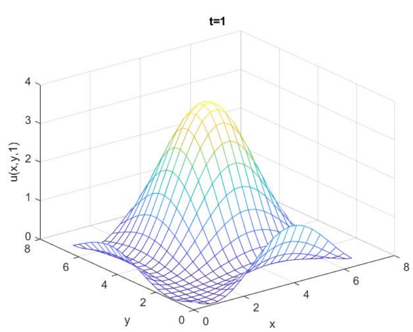
$$\begin{gathered} -2\pi = x_0 < x_1 < x_2 < \cdots < x_j < \cdots < x_{N-1} < x_N = 4\pi \\ -2\pi = y_0 < y_1 < y_2 < \cdots < y_k < \cdots < y_{N-1} < y_N = 4\pi \\ 0 = t_0 < t_1 < t_2 < \cdots < t_n < \cdots < t_{M-1} < t_M = 1 \end{gathered}$$

In this case, $\Delta x = \Delta y = \Delta$

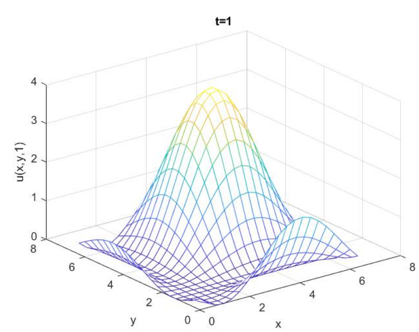
We build a three-dimensional array U(i,k,n) to store the data. Let $\boldsymbol{a} = (1, 0.5), g(x, y) = (1 - \cos x)(1 - \cos y), \Delta x = \Delta y = 2\pi/20$, and $\Delta t = 1/10$, we use Upwind scheme to solve this problem, which is given by

$$\begin{aligned} u_{j,k}^{n+1} = & u_{j,k}^n - \frac{\lambda}{2} a_1 \left(u_{j+1,k}^n - u_{j-1,k}^n \right) + \frac{\lambda}{2} |a_1| \left(u_{j+1,k}^n - 2u_{j,k}^n + u_{j-1,k}^n \right) \\ & - \frac{\lambda}{2} a_2 \left(u_{j,k+1} - u_{j,k-1}^n \right) + \frac{\lambda}{2} |a_2| \left(u_{j,k+1}^n - 2u_{j,k}^n + u_{j,k-1}^n \right) \end{aligned} \tag{23}$$

The result of final simulation time(t=1) is shown in Figure 11



(a) Upwind



(b) Analytical solution

Figure 11: Two-dimensional Hyperbolic PDE

Then we compute L^2 using grid norm function.

$$\|\mathbf{u^n}\|_{\Delta,2} = \Delta x^{\frac{1}{2}} \Delta y^{\frac{1}{2}} \|\mathbf{u^n}\|_2$$

where $\|\mathbf{u^n}\|_2$ is L^2 norm of matrix $U(:,:,t_n)$.

The result is shown below.

	Upwind
$\|\mathbf{u^0}\|_{\Delta,2}$	9.4255
$\|\mathbf{u^1}\|_{\Delta,2}$	9.3730
$\|\mathbf{u^2}\|_{\Delta,2}$	9.3227
$\|\mathbf{u^3}\|_{\Delta,2}$	9.2750
$\|\mathbf{u^4}\|_{\Delta,2}$	9.2303
$\|\mathbf{u^5}\|_{\Delta,2}$	9.1889
$\|\mathbf{u^6}\|_{\Delta,2}$	9.1512
$\|\mathbf{u^7}\|_{\Delta,2}$	9.1174
$\|\mathbf{u^8}\|_{\Delta,2}$	9.0876
$\|\mathbf{u^9}\|_{\Delta,2}$	9.0620
$\|\mathbf{u^{10}}\|_{\Delta,2}$	9.0405

Then in this case, Upwind method is stable.

4 Appendix

4.1 Code guide q1

In q1 folder, run triproduct.m to add the function to matlab path

```
%Function to compute A*x where A is a tridiagnal matrix, A=(abc)
function y=triproduct(a,b,c,x)
n=length(x);
y(1)=b(1)*x(1)+c(1)*x(2);
y(2:n-1)=b(2:n-1).*x(2:n-1)+c(2:n-1).*x(3:n)+a(1:n-2).*x(1:n-2);
y(n)=a(n-1)*x(n-1)+b(n)*x(n);
end
```

Run PDE.m to add the function to matlab path, This function is a bit long so I didn't put it here.

Then Run PDEmain.m, we get result of the PDEs. In workspace, u and v is the data of PDEs when h=L/99.

```
L=1;M=1000;N=100;h=1/99;deltat=200/999;P=2.26;
betav=0.1;betau=0.01*betav;n=N-2;k0=0.067;
pde=@PDE;
[u,v]=pde(L,N,M,h,deltat,betau,betav,k0,P);

%final time plot, delete'%' if you wanna see
%plot(0:h:1,v(end,:))

%contour plot, delete'%' if you wanna see
%x1=0:h:1;t1=0:deltat:200;
%[X,Y]=meshgrid(x1,t1);
%contourf(X,Y,v);

hj=[h,h/2,h/4,h/8];
e=zeros(3,1);
[u2,~]=pde(L,199,M,hj(2),deltat,betau,betav,k0,P); % h1/2
[u3,~]=pde(L,397,M,hj(3),deltat,betau,betav,k0,P); % h2/2
[u4,~]=pde(L,793,M,hj(4),deltat,betau,betav,k0,P); % h3/2
e(1)=abs(hj(2)*sum(u2(:))-hj(1)*sum(u(:)));
e(2)=abs(hj(3)*sum(u3(:))-hj(2)*sum(u2(:)));
e(3)=abs(hj(4)*sum(u4(:))-hj(3)*sum(u3(:)));

lge=log(e);lgh=log(hj(1:3));
plot(log(hj(1:3)),log(e))
hold on
scatter(log(hj(1:3)),log(e))
```

4.2 Code guide q2

In q2 folder, run f.m and g.m(boundary condition) to add the function to matlab path

```
function y=f(x,y)
y=32*pi^2*cos(4*pi*x)*cos(4*pi*y);
end
```

```
function y=g(z)
y=cos(4*pi*z);
end
```

Run PDE.m to add the function to matlab path, This function is a bit long so I didn't put it here. // and then run Poissonmain.m. In work space, u is the result of PDE with N=1024 because the loop which is used to compute error. if you wanna see the result with 64, delete codes from row 14.

```
1 fx=@f;gx=@g;P=@Poisson;
2 N=64;h=1/N;
3 [u,U,Freal]=P(N,fx,gx);
4 
5 %contour plot, delete'%' if you wanna see
6 %x1=h:h:1-h;
7 %y1=h:h:1-h;
8 %[X,Y]=meshgrid(x1,y1);
9 %contourf(X,Y,U);
10 %ylabel('y')
11 %xlabel('x')
12 %zlabel('u(x,y)')
13 
14 Nj=[16,32,64,128,256,512,1024];
15 error=zeros(length(Nj),1);
16 for i=1:length(Nj)
17 [u,~,Freal]=P(Nj(i),fx,gx);
18 error(i)=sum((Freal-u).^2);
19 end
20 
21 lge=log(error);
22 plot(log(Nj),lge)
23 hold on
24 scatter(log(Nj),lge)
25 xlabel('log(Nj)')
26 ylabel('log(L-2 norm)')
```

4.3 Code guide q3

In q3 folder/One dimension folder, run runlaxwind.m for one-dimensional Hyperbolic PDE. In q3 folder/two dimension folder, run runwind.m for two-dimensional Hyperbolic PDE