RD Method for Unsteady Scalar Conservation Law

Computational Fluid Dynamics Group

School of Aerospace Engineering

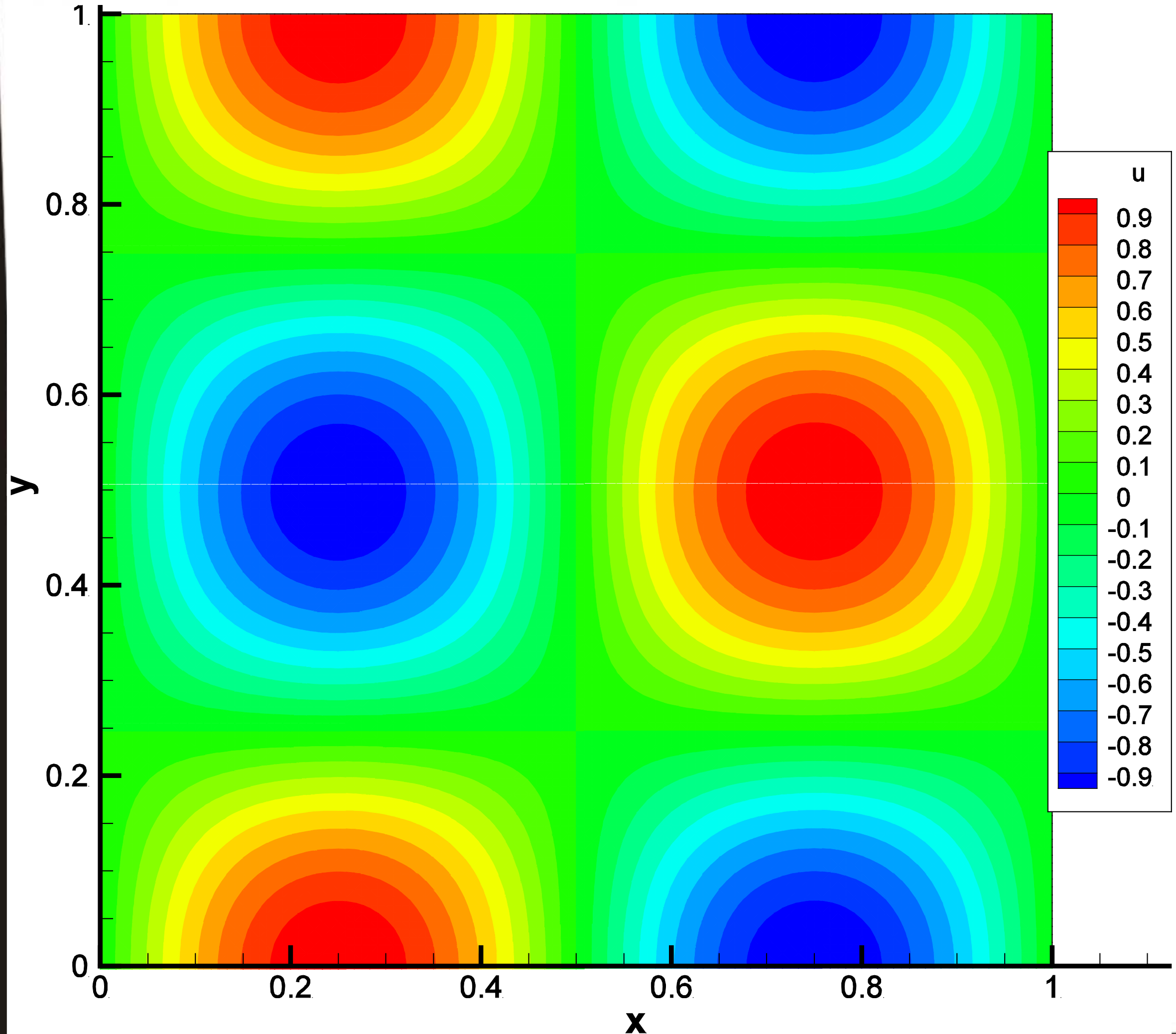
Universiti Sains Malaysia

21st October 2015

Presenter : Neoh Soon Sien

A Quick Review

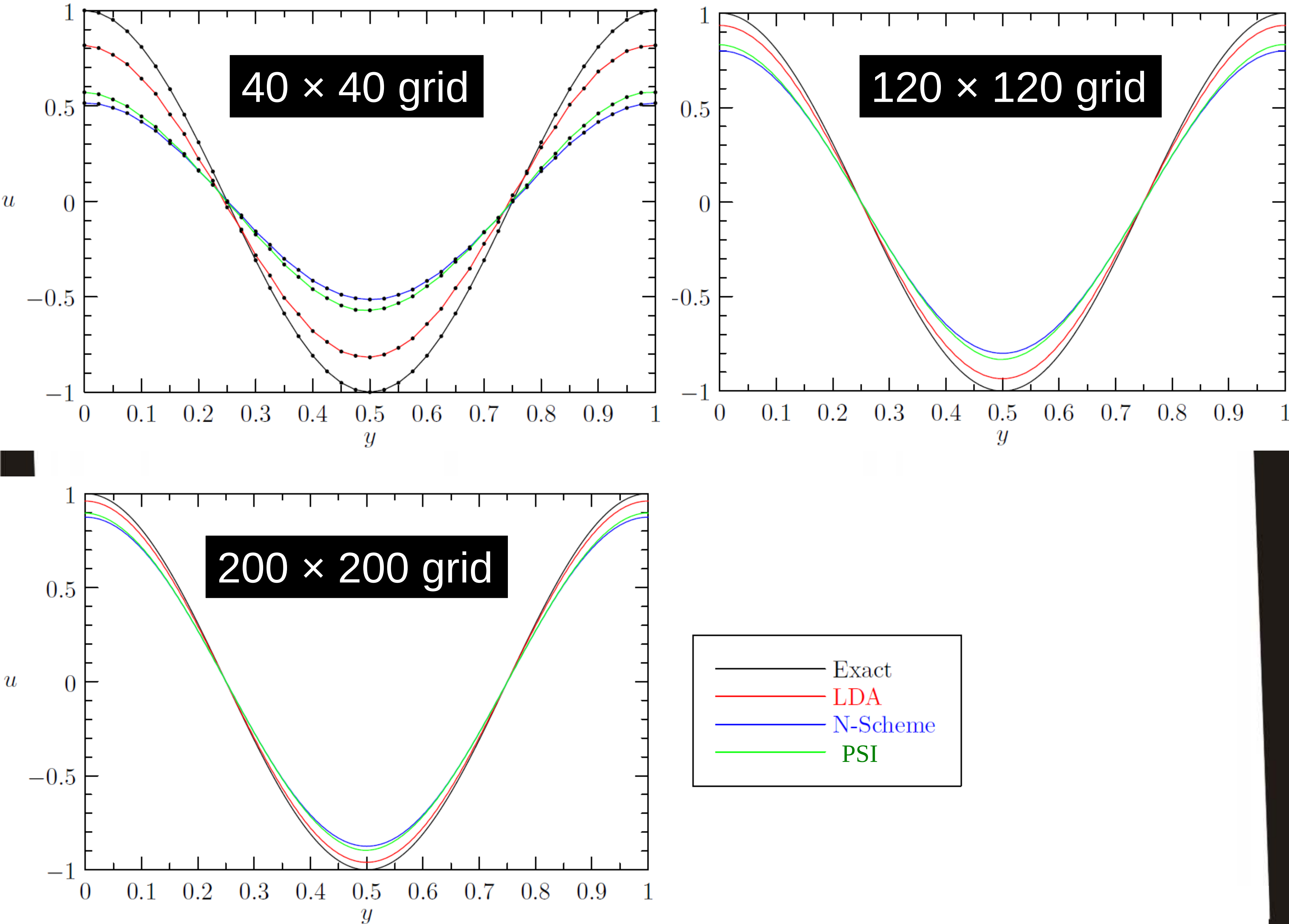
$$u(x, y, t = 0) = \sin(2\pi x)\ \cos(2\pi y)$$

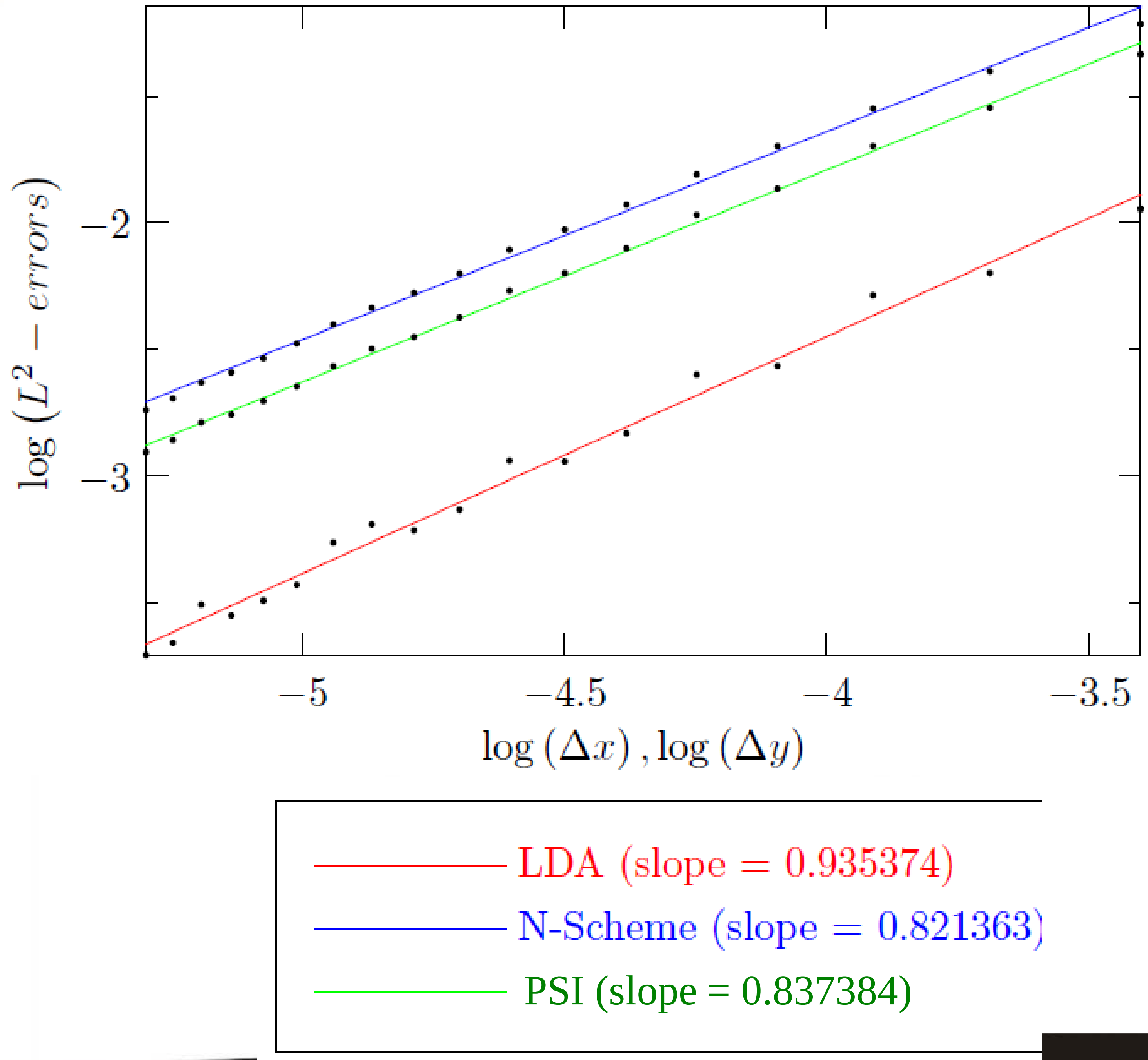


Initial profile at t = 0

&

Exact result at t = 1


40 × 40 grid
120 × 120 grid
200 × 200 grid
u
y
Exact
LDA
N-Scheme
PSI



$\log(L^2 - errors)$
−2
−3
−5
−4.5
−4
−3.5
$\log(\Delta x), \log(\Delta y)$
LDA (slope = 0.935374)
N-Scheme (slope = 0.821363)
PSI (slope = 0.837384)

Explicit RD Solver

The conservation law is given as

$$\boxed{\frac{\partial u}{\partial t} + a \cdot \nabla u = 0} \qquad 1$$

and the weak formulation is obtained as follows:

$$\omega_i(x, y) \times \left\{ \frac{\partial u^h}{\partial t} + \vec{a} \cdot \nabla u^h = 0 \right\}$$

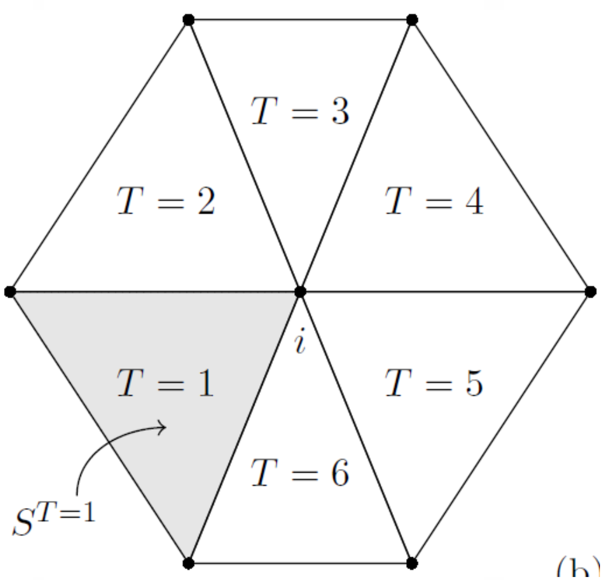
$$\int_{\Delta_T} \omega_i \left(\frac{\partial u^h}{\partial t} + \vec{a} \cdot \nabla u^h \right) dxdy = 0$$

$$\int_{\Delta_T} \omega_i \frac{\partial u}{\partial t} dxdy = - \int_{\Delta_T} \omega_i (\vec{a} \cdot \nabla u^h) dxdy$$

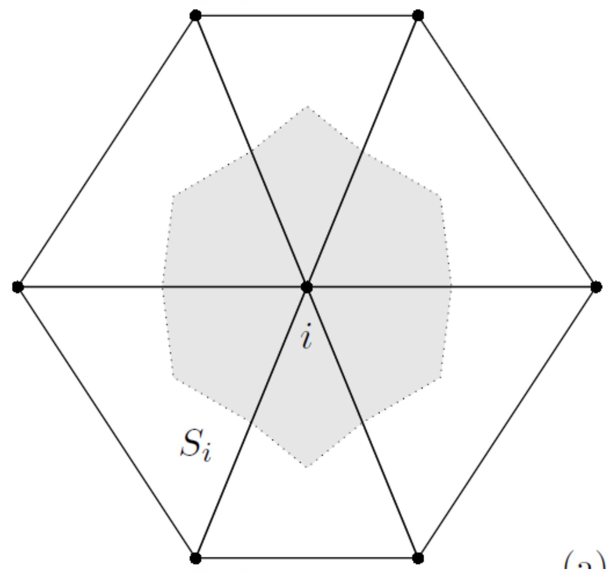
The value of primary variable – u at node i is given as:

$$\frac{S_T}{3}\left(\frac{\partial u}{\partial t}\right)_i = \phi_i^T$$

$$\boxed{\sum_{T \in \cup \Delta_i} \frac{S_T}{3}\left(\frac{\partial u}{\partial t}\right)_i = \sum_{T \in \cup \Delta_i} \phi_i^T} \quad 2$$



Element area
(primal cell area)
S^T

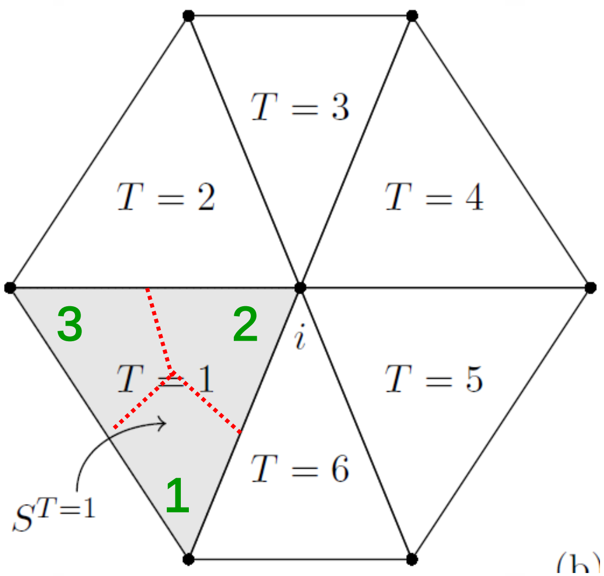


Median dual cell area

$$S_i = \sum_{T \in \cup \Delta_i} \frac{S^T}{3}$$

Implicit RD Solver

It emphasises on the integration of the time derivatives over the control volume.



Element area (primal cell area)

S^T

The weak formulation following from the previous section is

$$\int_{\Delta_T} \omega_i \frac{\partial u}{\partial t} dxdy = -\int_{\Delta_T} \omega_i (\vec{a} \cdot \nabla u^h) dxdy$$

$$\int_{\Delta_T} \omega_i \frac{\partial u}{\partial t} dxdy = \phi_i^T$$

The integration of the time derivative over the control volume can be split into 3 parts,

$$\int_{\Delta_T} \omega_i \frac{\partial u}{\partial t} dxdy$$

$$= \int_{A_1} \omega_1 \left(\frac{\partial u}{\partial t}\right)_1 dxdy + \int_{A_2} \omega_2 \left(\frac{\partial u}{\partial t}\right)_2 dxdy + \int_{A_3} \omega_3 \left(\frac{\partial u}{\partial t}\right)_3 dxdy$$

Using the **two time-level** discretisation for the time derivative,

$$\left(\frac{\partial u}{\partial t}\right)_i \cong \frac{u_i^{n+1} - u_i^n}{\Delta t}$$

the nodal update for explicit RD scheme reads

$$\sum_{T \in \cup\Delta_i} \frac{S_T}{3}\left(\frac{\partial u}{\partial t}\right)_i = \sum_{T \in \cup\Delta_i} \phi_i^T \Rightarrow \boxed{S_i\left(\frac{\partial u^h}{\partial t}\right)_i = \sum_{T \in \cup\Delta_i} \phi_i^T} \qquad \boxed{3}$$

while the nodal update for implicit RD scheme are

$$\boxed{\sum_{T \in \cup\Delta_i} \sum_{j=1}^{3} m_{ij}^T \left(\frac{\partial u^h}{\partial t}\right)_i = \sum_{T \in \cup\Delta_i} \phi_i^T} \qquad \boxed{4}$$

Explicit Lax-Wendroff

Explicit Lax-Wendroff Scheme

It is expected to maintain its second-order accuracy in both space and time.

Two-time-level:

$$u_i^{n+1} = u_i^n + \frac{\Delta t}{S_i} \sum_{T \in \cup \Delta_i} \left(\frac{1}{3} + \frac{\Delta t}{2} \frac{k_i}{S_T} \right) \phi^{T,n}$$

Three-time-level:

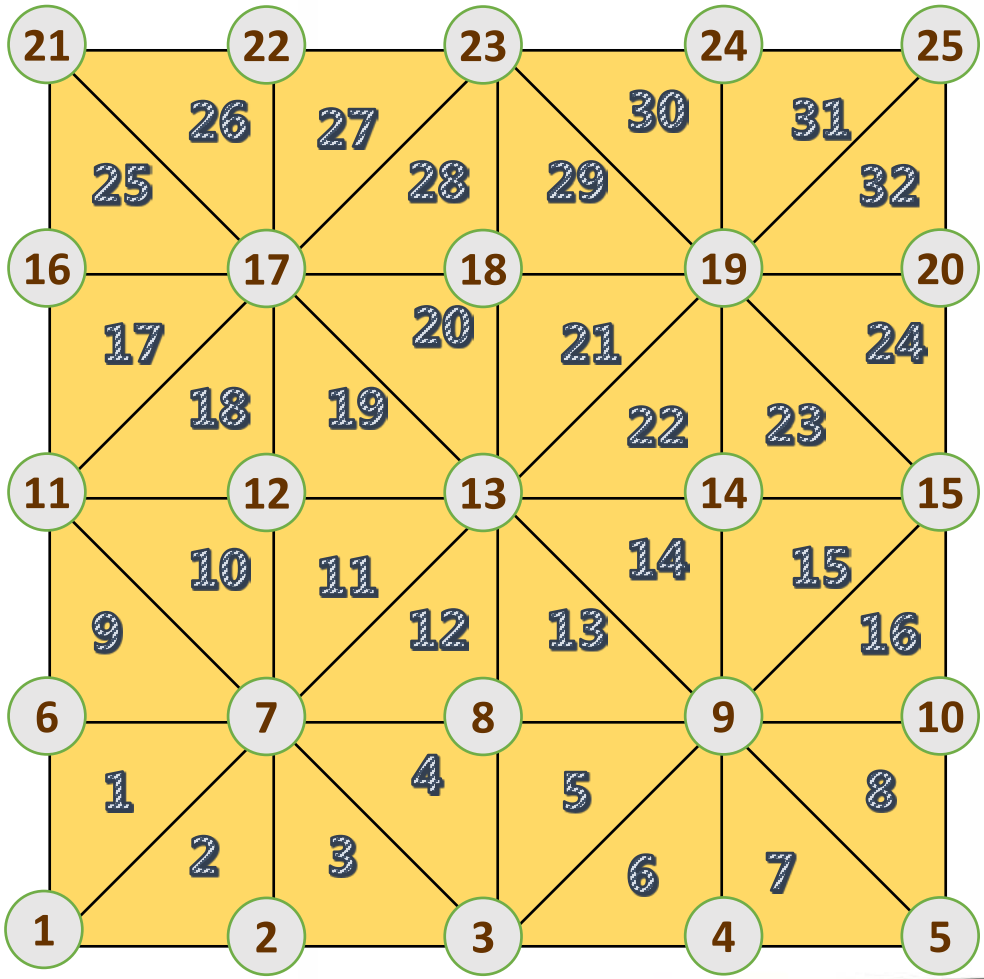
$$S_i \left(\frac{3u_i^{n+1} - 4u_i^n + u_i^{n-1}}{2\Delta t} \right) = \sum_{T \in \cup \Delta_i} \left(\frac{1}{3} + \Delta t \frac{k_i}{S_T} \right) \phi^{T,n}$$

$$u_i^{n+1} = \frac{1}{3} \left(4u_i^n - u_i^{n-1} + \frac{2\Delta t}{S_i} \sum_{T \in \cup \Delta_i} \left(\frac{1}{3} + \Delta t \frac{k_i}{S_T} \right) \phi^{T,n} \right)$$

Implicit RD Solver

To obtain second-order-accuracy both in space and time.

The discretisation of spatial domain

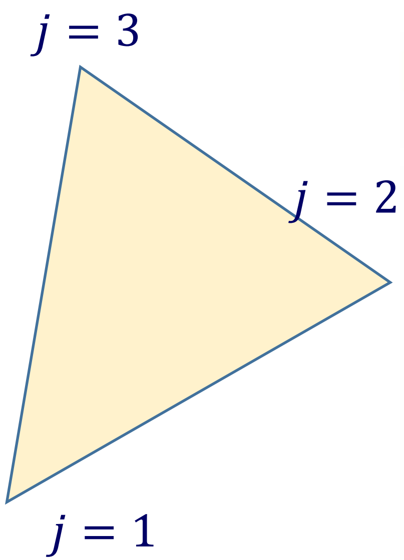


Constructing Mass Matrix

$$\sum_{T\in\cup\Delta_i}\sum_{j=1}^{3} m_{ij}^T\left(\frac{\partial u^h}{\partial t}\right)_i = \sum_{T\in\cup\Delta_i}\phi_i^T$$

For a single element,

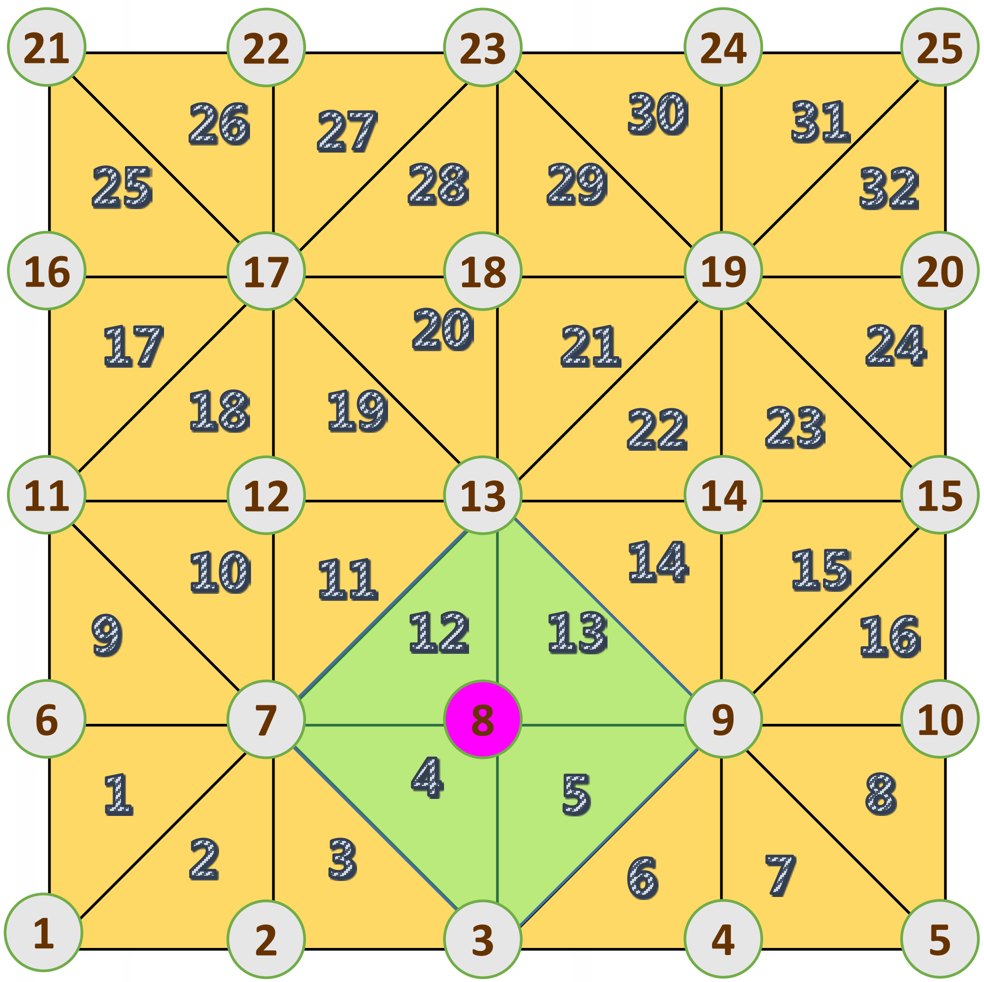
$$\sum_{T\in\cup\Delta_i}\frac{S_T}{3}\begin{bmatrix} m_{11}^T & m_{12}^T & m_{13}^T \\ m_{21}^T & m_{22}^T & m_{23}^T \\ m_{31}^T & m_{32}^T & m_{33}^T \end{bmatrix}\begin{bmatrix}(u_t)_1 \\ (u_t)_2 \\ (u_t)_3\end{bmatrix} = \sum_{T\in\cup\Delta_i}\begin{bmatrix}\phi_1^T \\ \phi_2^T \\ \phi_3^T\end{bmatrix}$$



Mass matrix – consistent upwind

$$m_{ij}^T = \frac{S_T}{3}\begin{bmatrix} \beta_1^T(2-\beta_1^T) & \beta_1^T(1-\beta_2^T) & \beta_1^T(1-\beta_3^T) \\ \beta_2^T(1-\beta_1^T) & \beta_2^T(2-\beta_2^T) & \beta_2^T(1-\beta_3^T) \\ \beta_3^T(1-\beta_1^T) & \beta_3^T(1-\beta_2^T) & \beta_3^T(2-\beta_3^T) \end{bmatrix}$$

Since there are 25 nodes, therefore, it needs a 25 × 25 matrix to represent the mass matrix m_{ij}

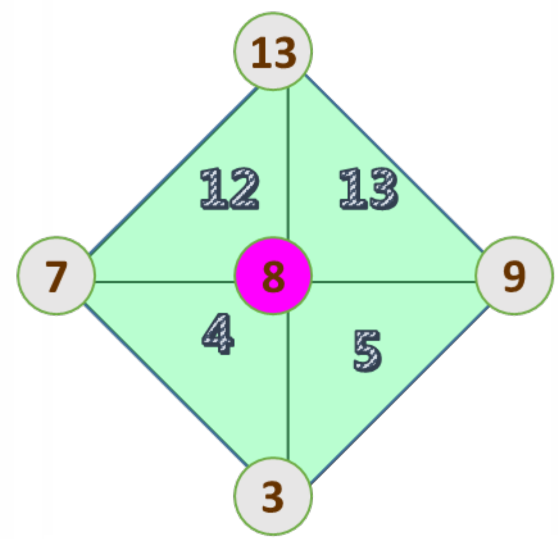


$$\begin{bmatrix} m_{1,1} & \cdots & m_{1,25} \\ \vdots & \ddots & \vdots \\ m_{25,1} & \cdots & m_{25,25} \end{bmatrix}$$

Element 4

$$m_{ij}^T = \frac{S_T}{3}\begin{bmatrix} \beta_1^T(2-\beta_1^T) & \beta_1^T(1-\beta_2^T) & \beta_1^T(1-\beta_3^T) \\ \beta_2^T(1-\beta_1^T) & \beta_2^T(2-\beta_2^T) & \beta_2^T(1-\beta_3^T) \\ \beta_3^T(1-\beta_1^T) & \beta_3^T(1-\beta_2^T) & \beta_3^T(2-\beta_3^T) \end{bmatrix}$$

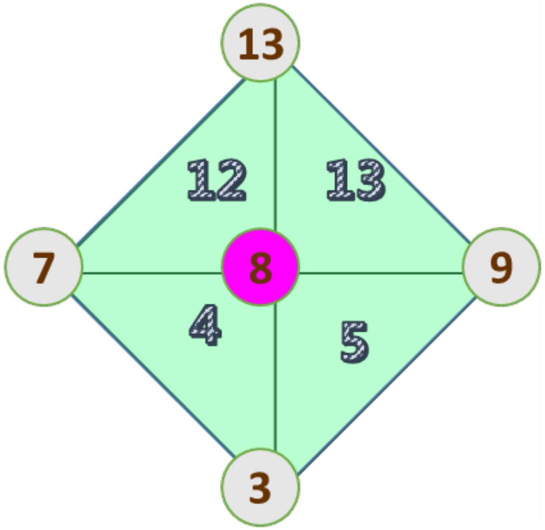
	3	7	8
3	■	■	■
7	■	■	■
8	■	■	■



Element 5

$$m_{ij}^{T} = \frac{S_T}{3}\begin{bmatrix} \beta_1^T(2-\beta_1^T) & \beta_1^T(1-\beta_2^T) & \beta_1^T(1-\beta_3^T) \\ \beta_2^T(1-\beta_1^T) & \beta_2^T(2-\beta_2^T) & \beta_2^T(1-\beta_3^T) \\ \beta_3^T(1-\beta_1^T) & \beta_3^T(1-\beta_2^T) & \beta_3^T(2-\beta_3^T) \end{bmatrix}$$

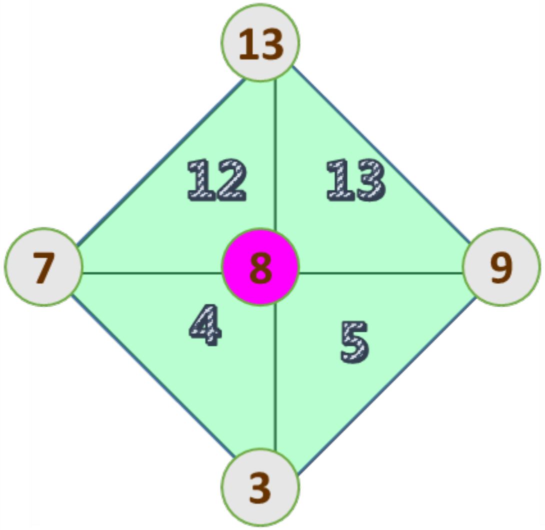
	3	8	9
3	■	■	■
8	■	■	■
9	■	■	■



Element 12

$$m_{ij}^{T} = \frac{S_T}{3}\begin{bmatrix} \beta_1^T(2-\beta_1^T) & \beta_1^T(1-\beta_2^T) & \beta_1^T(1-\beta_3^T) \\ \beta_2^T(1-\beta_1^T) & \beta_2^T(2-\beta_2^T) & \beta_2^T(1-\beta_3^T) \\ \beta_3^T(1-\beta_1^T) & \beta_3^T(1-\beta_2^T) & \beta_3^T(2-\beta_3^T) \end{bmatrix}$$

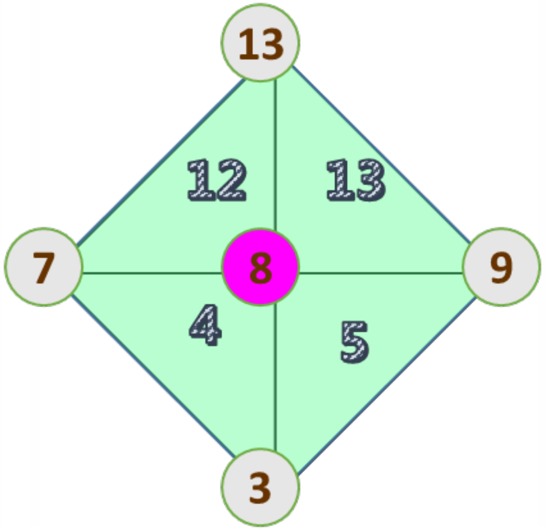
$$\begin{array}{c|ccc} & 7 & 8 & 13 \\ \hline 7 & \blacksquare & \blacksquare & \blacksquare \\ 8 & \blacksquare & \blacksquare & \blacksquare \\ 13 & \blacksquare & \blacksquare & \blacksquare \end{array}$$



Element 13

$$m_{ij}^T = \frac{S_T}{3}\begin{bmatrix} \beta_1^T(2-\beta_1^T) & \beta_1^T(1-\beta_2^T) & \beta_1^T(1-\beta_3^T) \\ \beta_2^T(1-\beta_1^T) & \beta_2^T(2-\beta_2^T) & \beta_2^T(1-\beta_3^T) \\ \beta_3^T(1-\beta_1^T) & \beta_3^T(1-\beta_2^T) & \beta_3^T(2-\beta_3^T) \end{bmatrix}$$

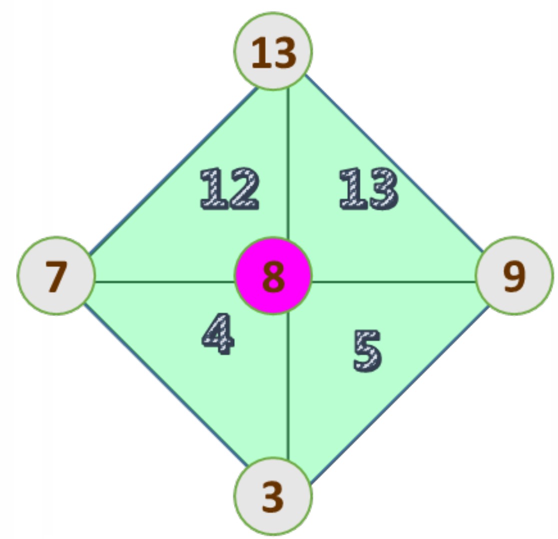
$$\begin{array}{c|ccc} & 8 & 9 & 13 \\ \hline 8 & \blacksquare & \blacksquare & \blacksquare \\ 9 & \blacksquare & \blacksquare & \blacksquare \\ 13 & \blacksquare & \blacksquare & \blacksquare \end{array}$$



Element 4 + 5 + 12 + 13

$$\sum_{T\in\cup\Delta_i}\frac{S_T}{3}\begin{bmatrix} m_{11}^T & m_{12}^T & m_{13}^T \\ m_{21}^T & m_{22}^T & m_{23}^T \\ m_{31}^T & m_{32}^T & m_{33}^T \end{bmatrix}\begin{bmatrix}(u_t)_1\\(u_t)_2\\(u_t)_3\end{bmatrix}=\sum_{T\in\cup\Delta_i}\begin{bmatrix}\phi_1^T\\\phi_2^T\\\phi_3^T\end{bmatrix}$$

	3	7	8	9	13
3	■	■	■	■	
7	■	■	■		■
8	■	■	■	■	■
9	■		■	■	■
13		■	■	■	■



Solving the system of equations

Gauss-Elimination Method will be used in this work.

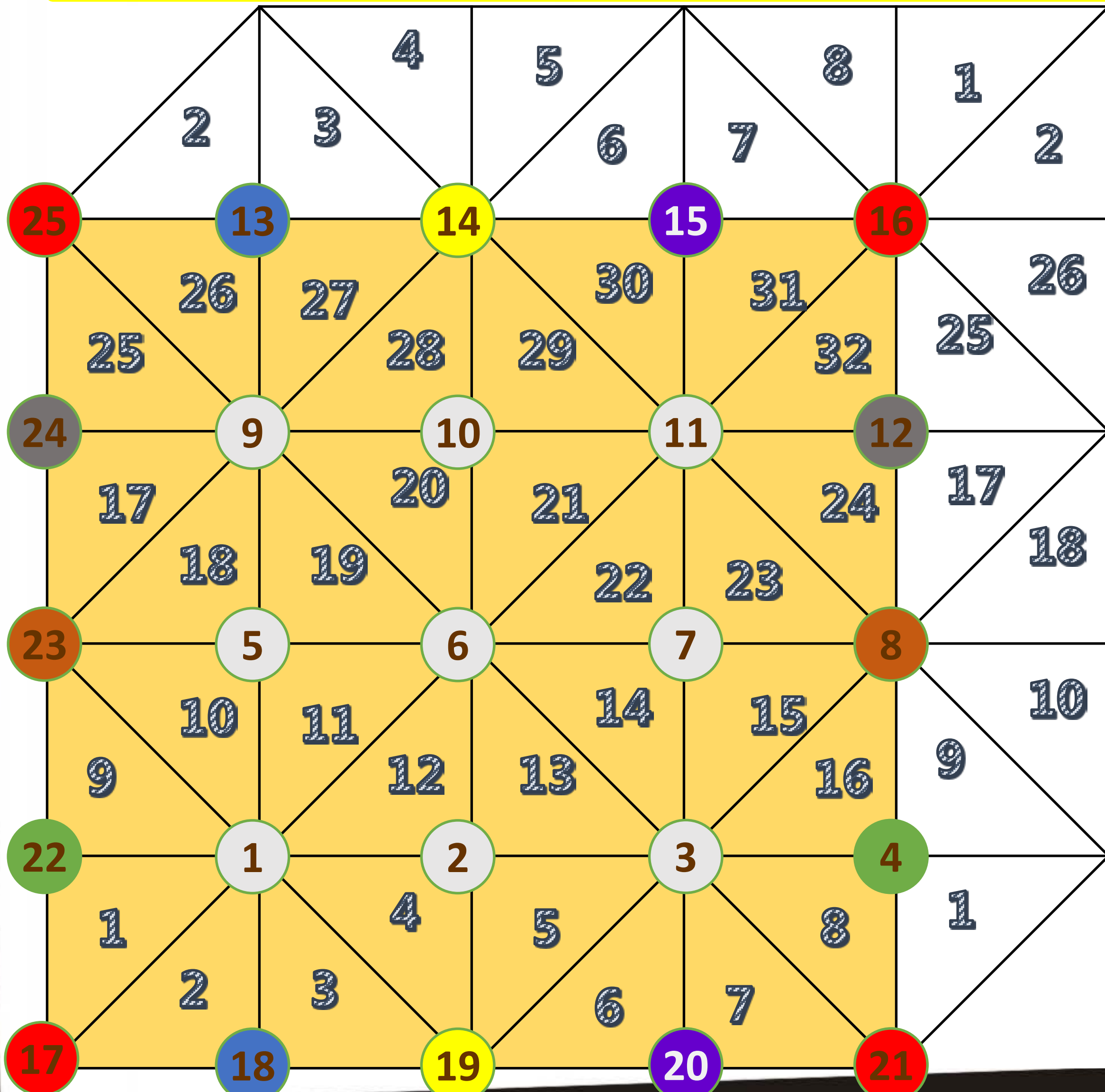
For example, take a 5×5 matrix,

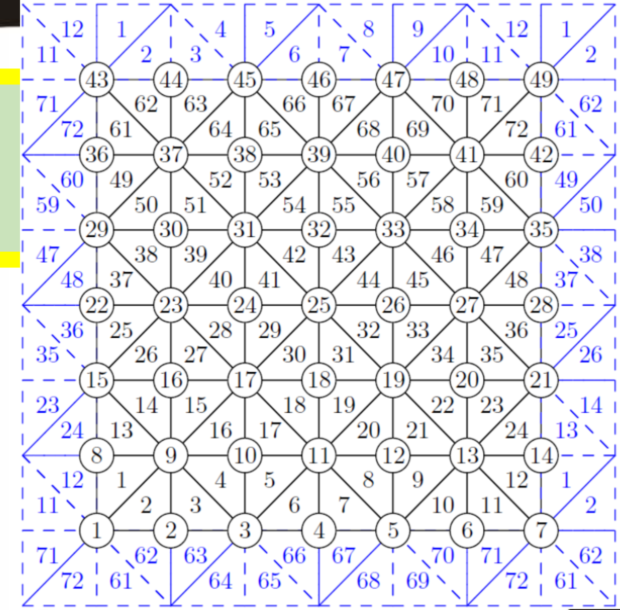
$$\left[\begin{array}{ccccc|c} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & \phi_1 \\ c_{21} & c_{22} & c_{23} & c_{24} & c_{25} & \phi_2 \\ c_{31} & c_{32} & c_{33} & c_{34} & c_{35} & \phi_3 \\ c_{41} & c_{42} & c_{43} & c_{44} & c_{45} & \phi_4 \\ c_{51} & c_{52} & c_{53} & c_{54} & c_{55} & \phi_5 \end{array}\right] \Rightarrow \left[\begin{array}{ccccc|c} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & \phi_1 \\ 0 & c_{22} & c_{23} & c_{24} & c_{25} & \phi_2 \\ 0 & 0 & c_{33} & c_{34} & c_{35} & \phi_3 \\ 0 & 0 & 0 & c_{44} & c_{45} & \phi_4 \\ 0 & 0 & 0 & 0 & c_{55} & \phi_5 \end{array}\right]$$

$$\Rightarrow \left[\begin{array}{ccccc|c} c_{11} & 0 & 0 & 0 & 0 & \phi_1 \\ 0 & c_{22} & 0 & 0 & 0 & \phi_2 \\ 0 & 0 & c_{33} & 0 & 0 & \phi_3 \\ 0 & 0 & 0 & c_{44} & 0 & \phi_4 \\ 0 & 0 & 0 & 0 & c_{55} & \phi_5 \end{array}\right]$$

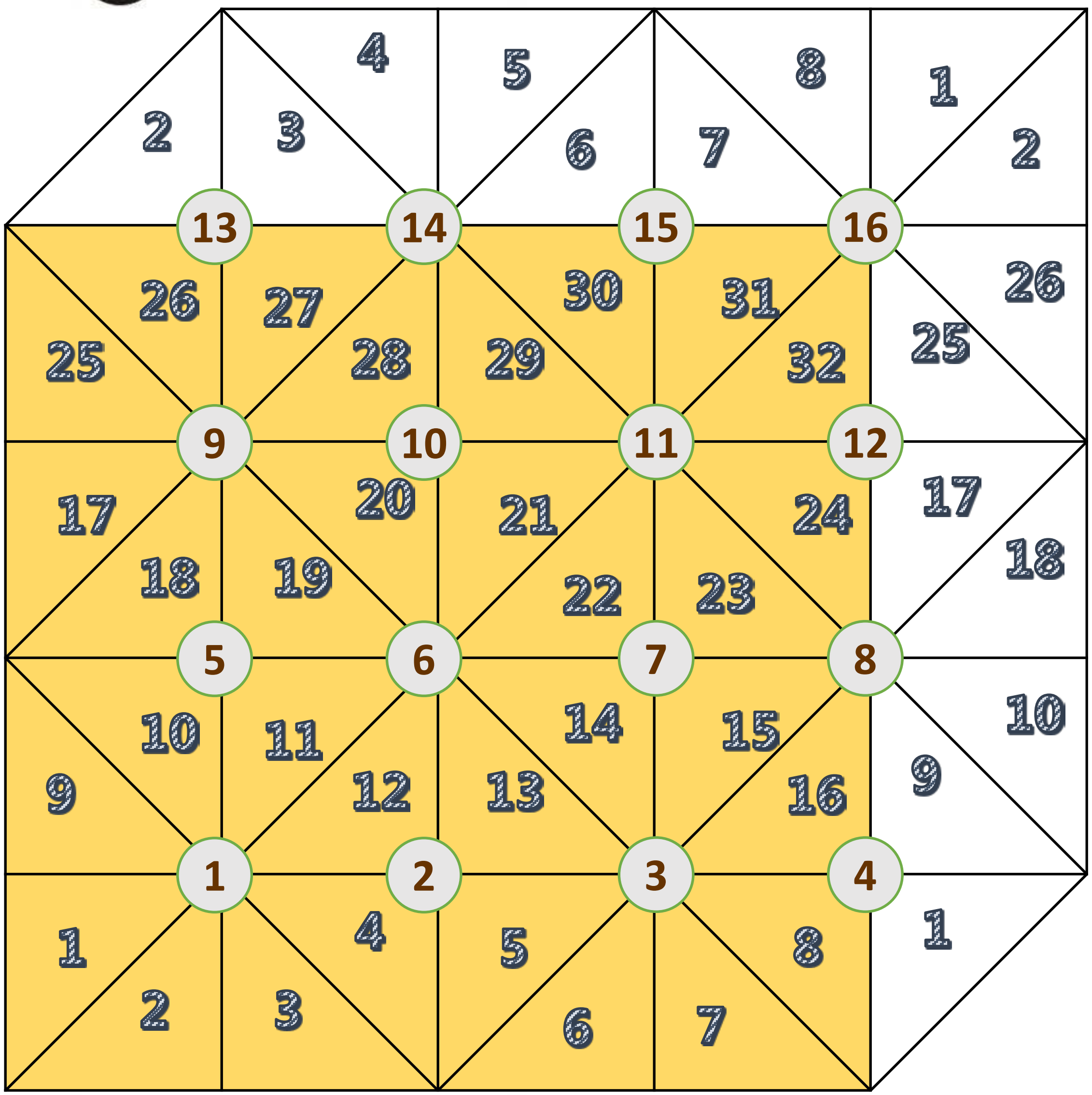
$$\sum_{T \in \cup \Delta_i} \frac{S_T}{3} \begin{bmatrix} m_{11}^T & m_{12}^T & m_{13}^T \\ m_{21}^T & m_{22}^T & m_{23}^T \\ m_{31}^T & m_{32}^T & m_{33}^T \end{bmatrix} \begin{bmatrix} (u_t)_1 \\ (u_t)_2 \\ (u_t)_3 \end{bmatrix} = \sum_{T \in \cup \Delta_i} \begin{bmatrix} \phi_1^T \\ \phi_2^T \\ \phi_3^T \end{bmatrix}$$

Periodic BCs





Nodes located along the boundaries are identical to their corresponding counterparts on the opposite side.



When those matching nodes are being removed, the system of equations will just become

$$16 \times 16$$

instead of

$$25 \times 25$$

Precaution:

If one does not remove away the matrix components of those repeating identical nodes, then one might get row of zeros which lead to singularity.

For example, nodes 16, 17, 21 and 25 all refer to the same point. If the 25×25 matrix is used, then rows 17, 21 and 25 will all go to zero under Gauss Eliminating procedure.

Dual-time Stepping

Dual-time stepping

It is done by performing fictitious time or pseudo-time iteration in between two subsequent physical time step.

```
time = 0.0;
while (time ≤ 1.00)
{
        :
        :
        time = time + Δt;
};
```

$$\Delta t \le \frac{2}{3} CFL \min_{T \in \cup \Delta_i} \left(\frac{S_T}{\max_{j \in \Delta_T} \left| k_j \right|} \right)$$

$$S_i \frac{\partial u}{\partial \tau} + \left(S_i \left(\frac{\partial u^h}{\partial t} \right)_i - \sum_{T \in \cup \Delta_i} \phi_i^T \right) = 0$$

The derivative of the variable-u with respect to fictitious time τ is discretised using forward Euler finite difference formula.

```
time = 0.0;
while (time ≤ 1.00)
{
        :
        :
        time = time + Δt;
};
```

The fictitious time is iterated for each physical time step.

$$\boxed{u_i^{n+1,k+1} = u_i^{n+1,k} + \frac{\Delta\tau}{S_i} \sum_{T \in \cup \Delta_i} \phi_i^T}$$

Criteria for the fictitious time iteration

The fictitious time step chosen must be much smaller than the physical time step to ensure stability.
In this case, the fictitious time step in use is

$$\Delta\tau = \frac{\Delta t}{10^6}$$

And the criteria for it to stop is when

$$\boxed{\max_{i\in\Omega}\left|u_i^k - u_i^{k-1}\right| < tolerance}$$

$$tolerance = 10^{-8}$$

or when

$$\boxed{k \geq kmax}$$

$$kmax = 20$$

Three level time derivative discretisation

Three level time discretisation

The time derivative at the nodes is evaluated as

$$\left(\frac{\partial u}{\partial t}\right)_i \cong \frac{3u_i^{n+1} - 4u_i^n + u_i^{n-1}}{2\Delta t}$$

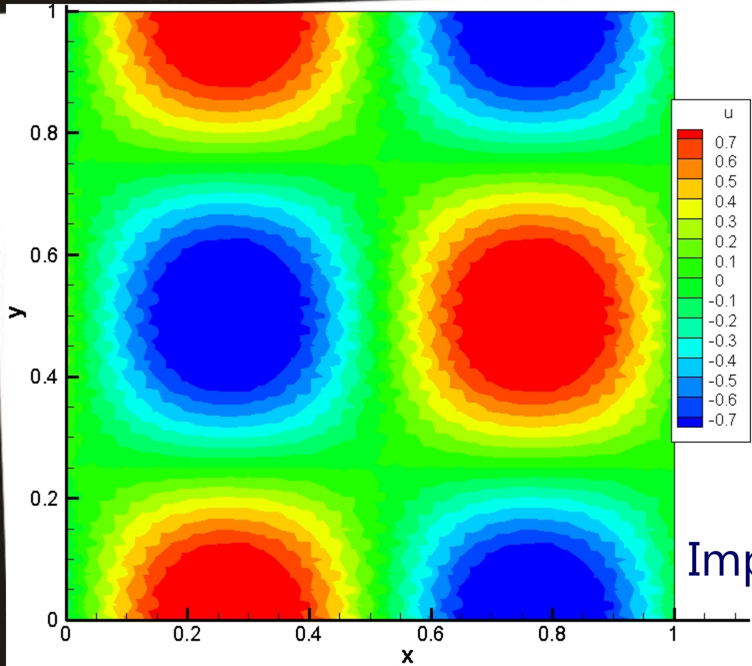
However, during the first time step t_1, whereby only the initial condition at t_0 is known, then the two-level time discretisation is used.

$$\left(\frac{\partial u}{\partial t}\right)_i \cong \frac{u_i^{n+1} - u_i^n}{\Delta t}$$

Results

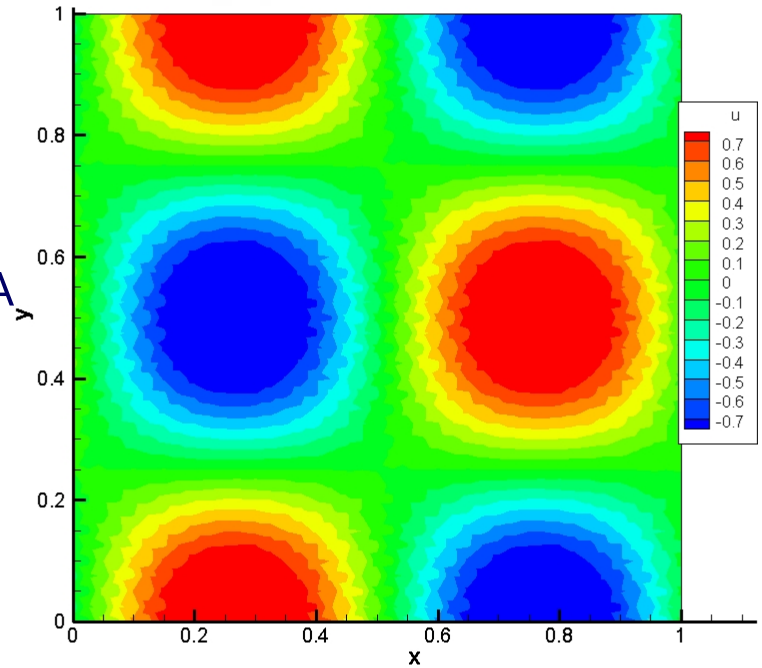
Two level time step only

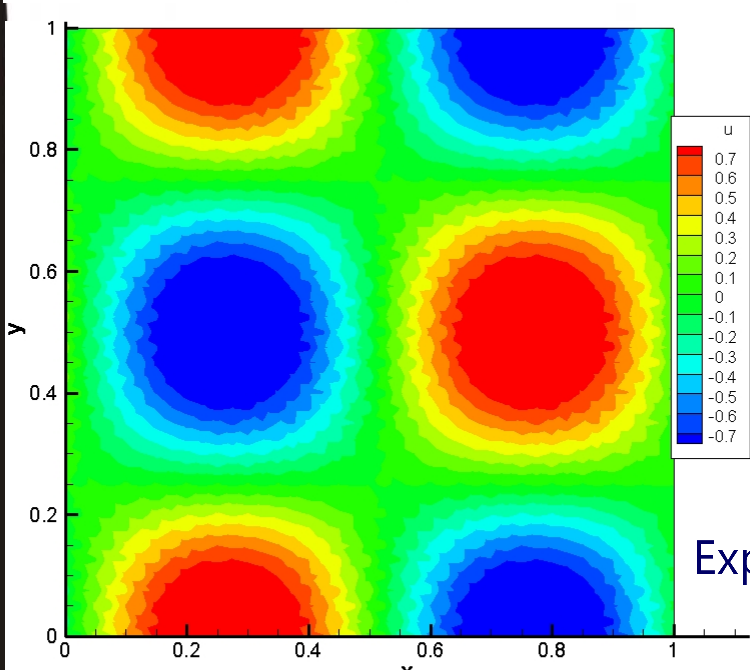
40 x 40 grid



Implicit LDA

Explicit LDA

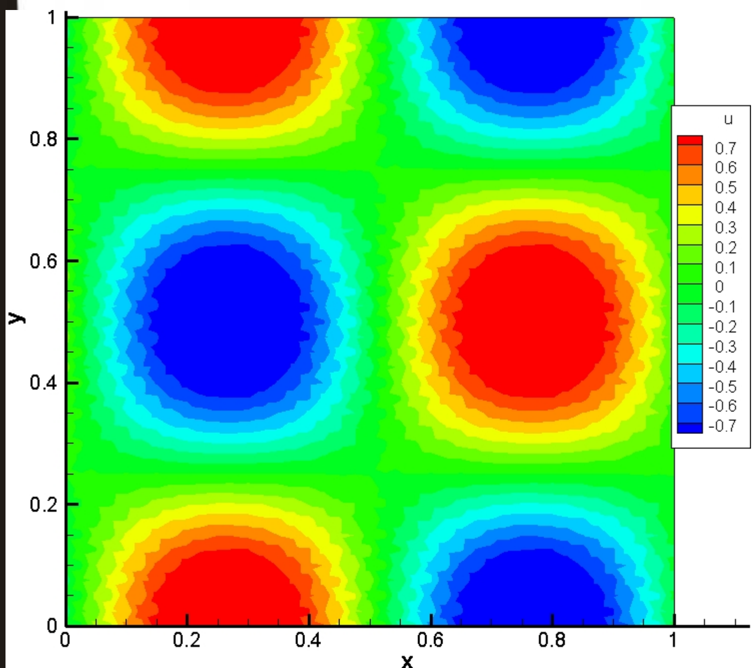




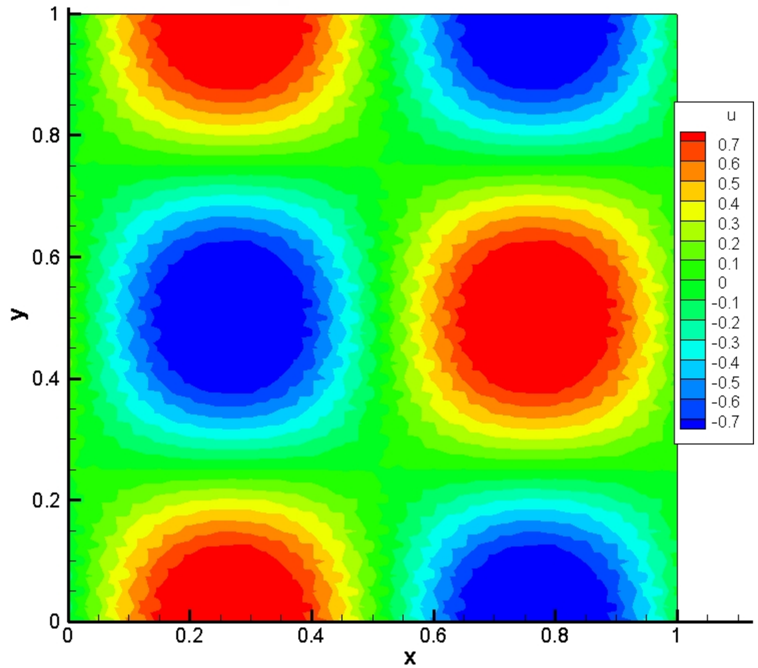
Explicit Lax-Wendroff

Two level time step & dual time stepping

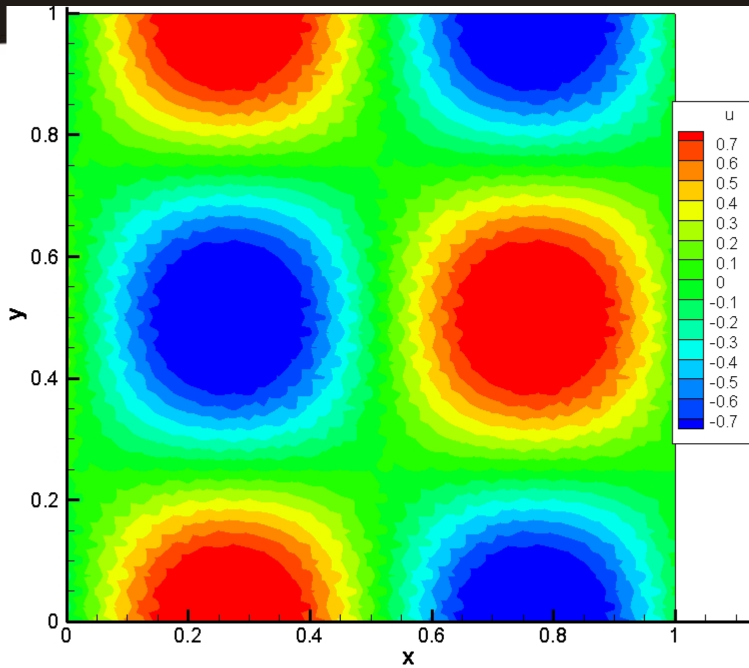
40 x 40 grid



Implicit LDA



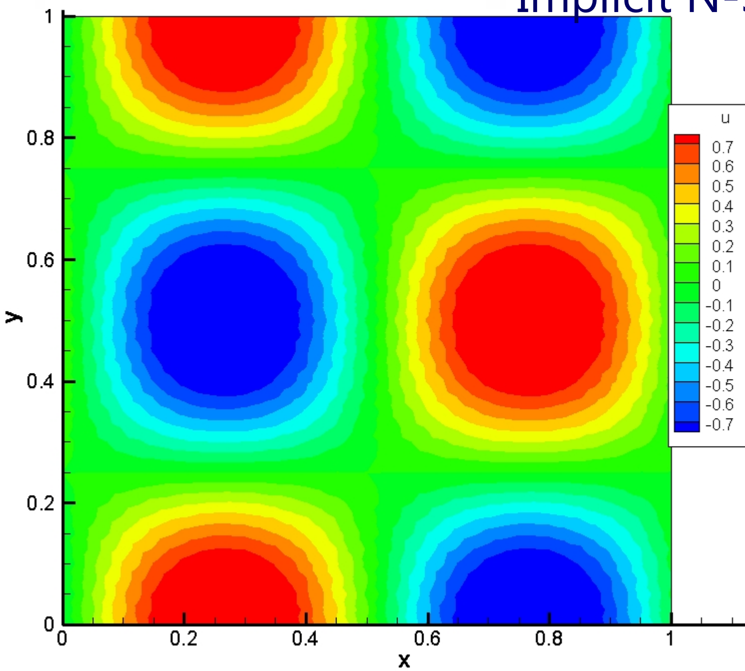
Explicit LDA

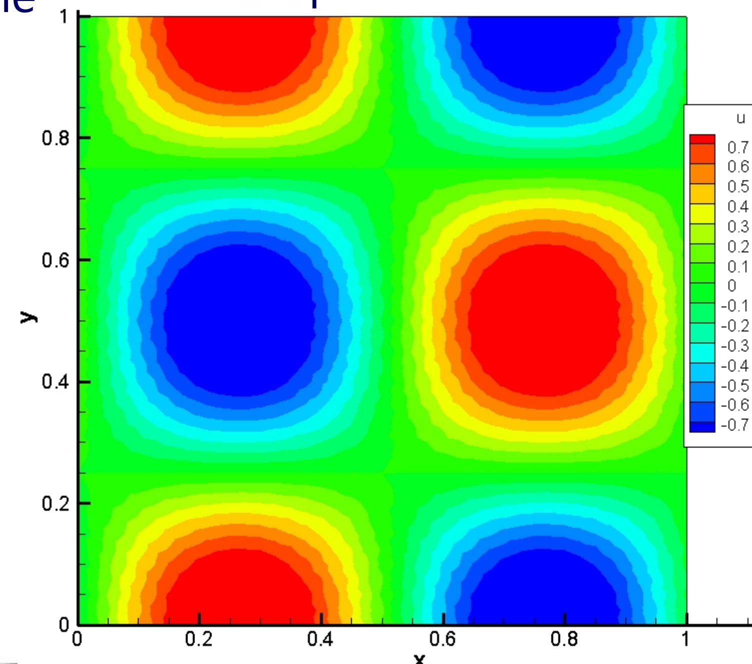


Explicit Lax-Wendroff

Implicit N-Scheme

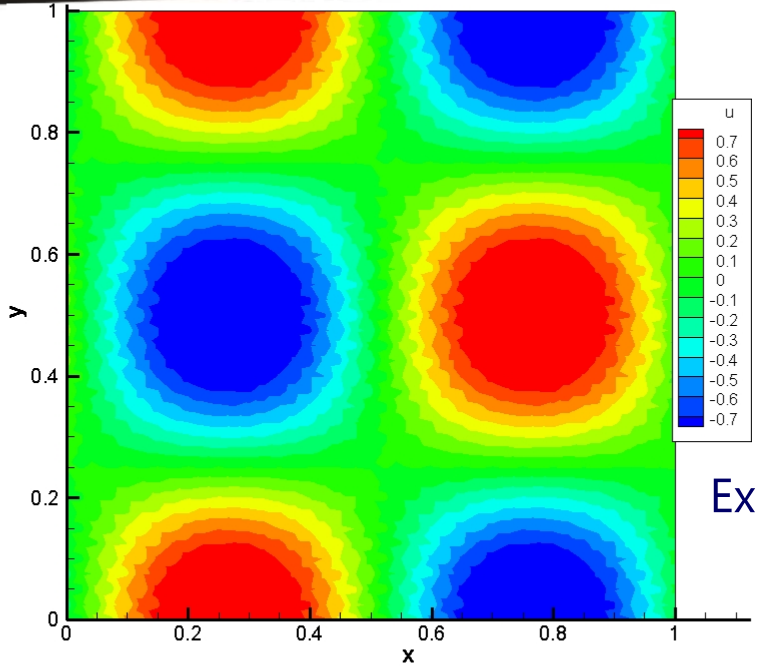
Explicit N-Scheme



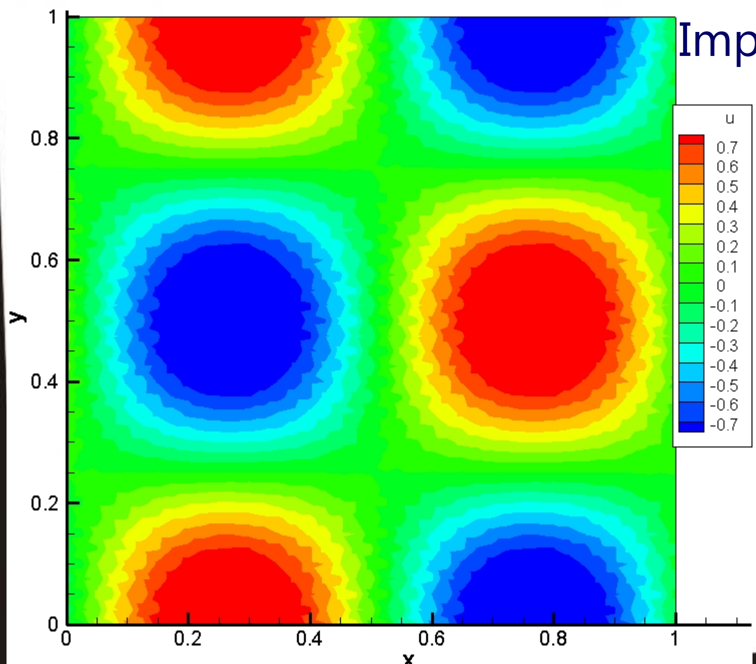


Three level time step only

40 x 40 grid

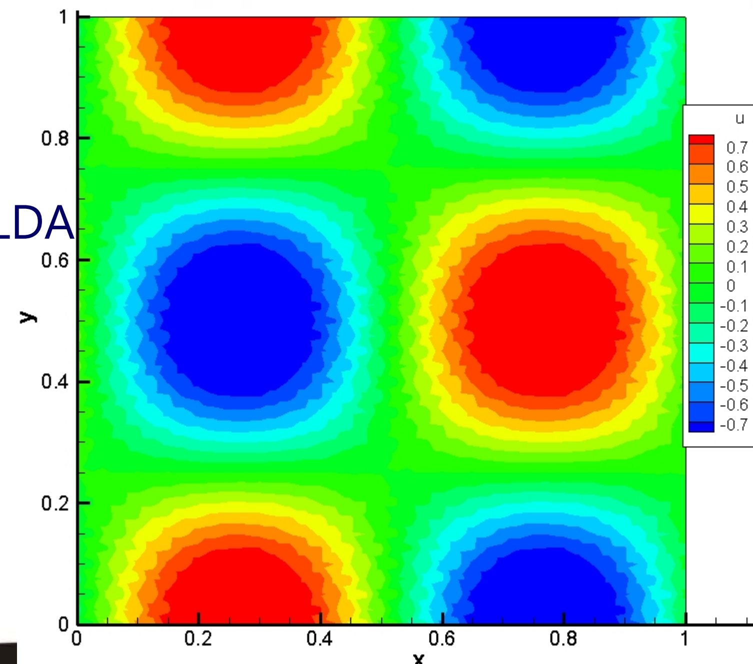


Explicit Lax-Wendroff

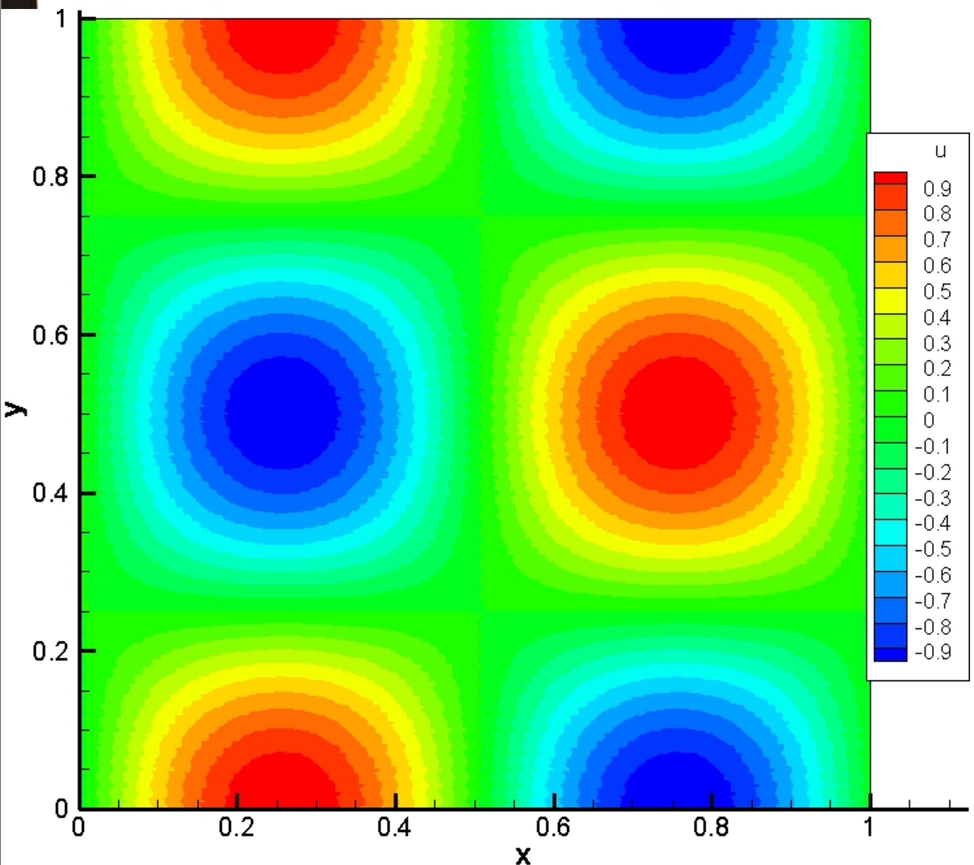


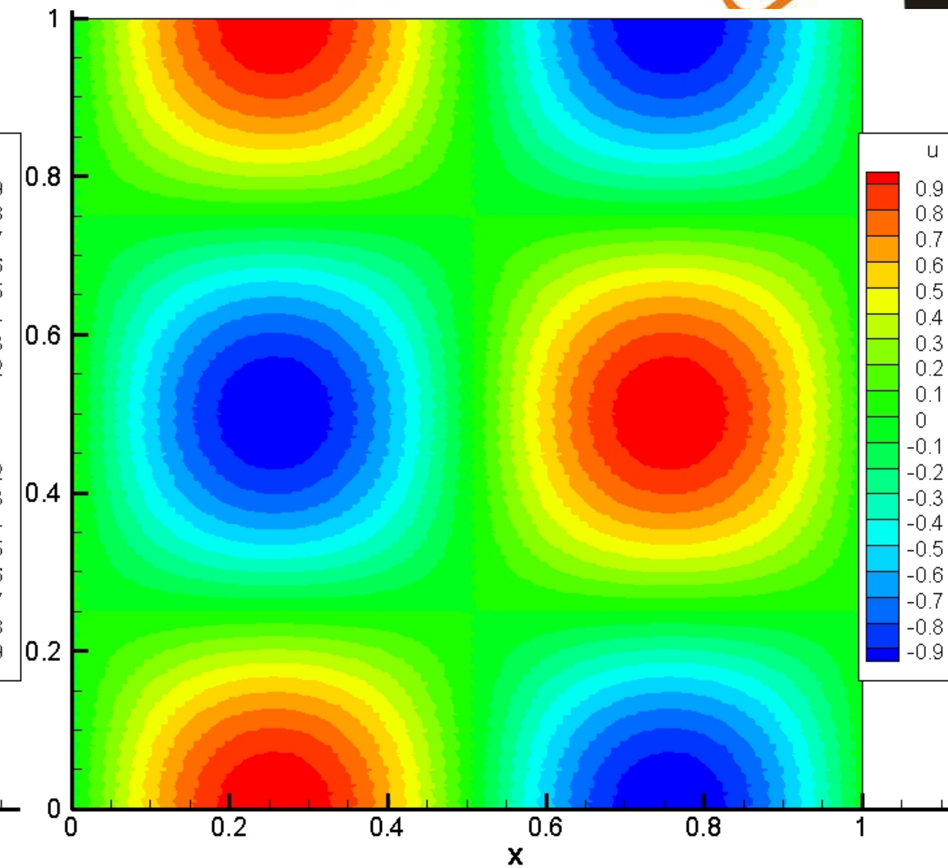
Implicit LDA

Explicit LDA



Grid 100 x 100

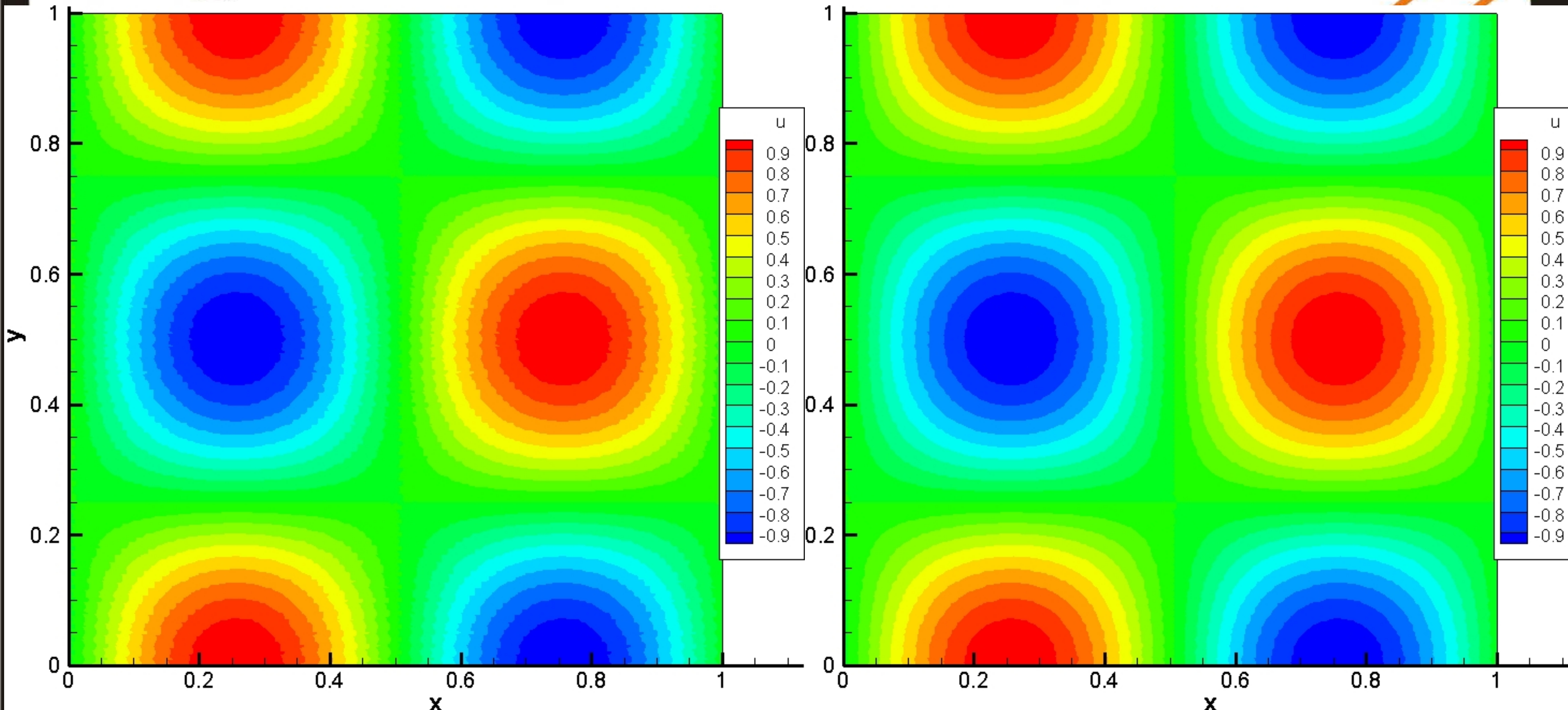




Implicit LDA
(Two level time discretisation)

Implicit LDA
(Three level time discretisation)

Grid 100 x 100

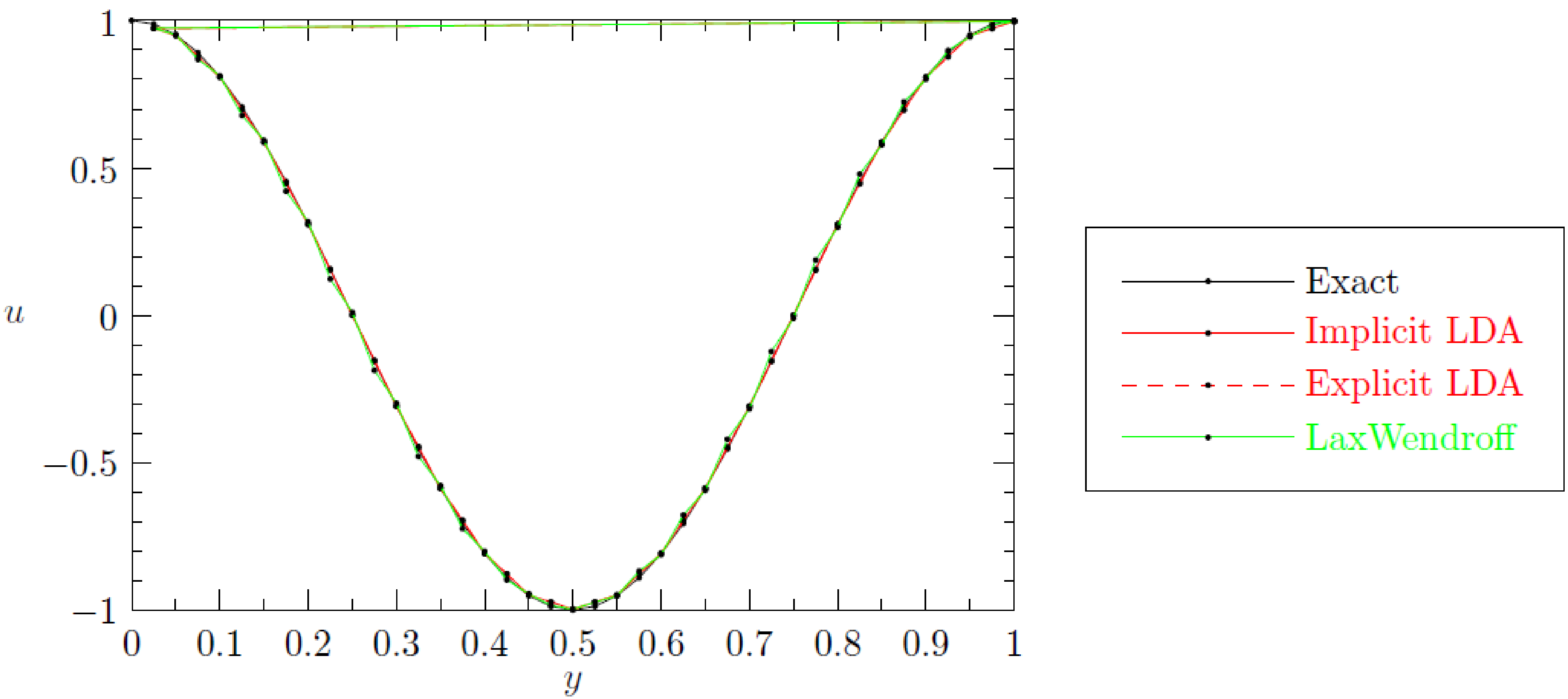


Implicit LDA
(Two level time discretisation
&
Dual-time-stepping)

Implicit N-Scheme
(Two level time discretisation
&
Dual-time-stepping)

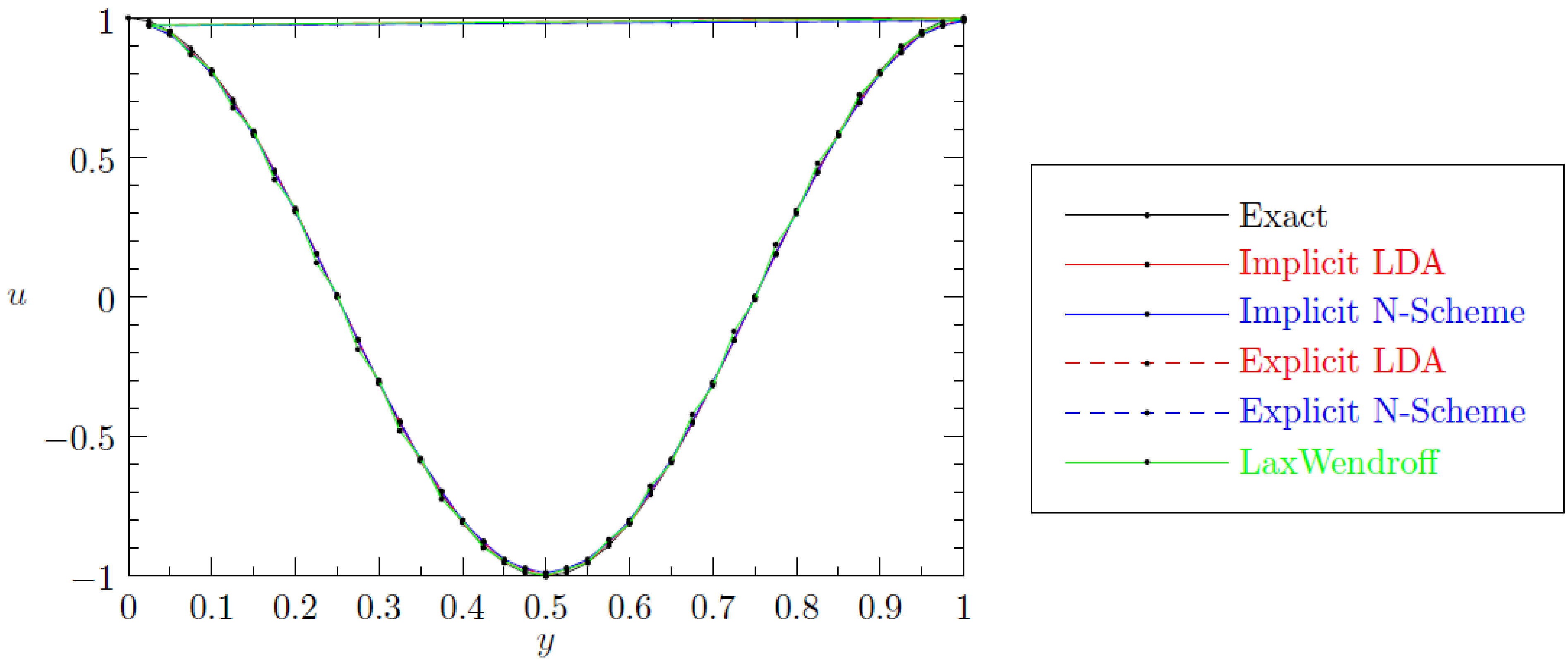
Two level time step only

40 x 40 grid



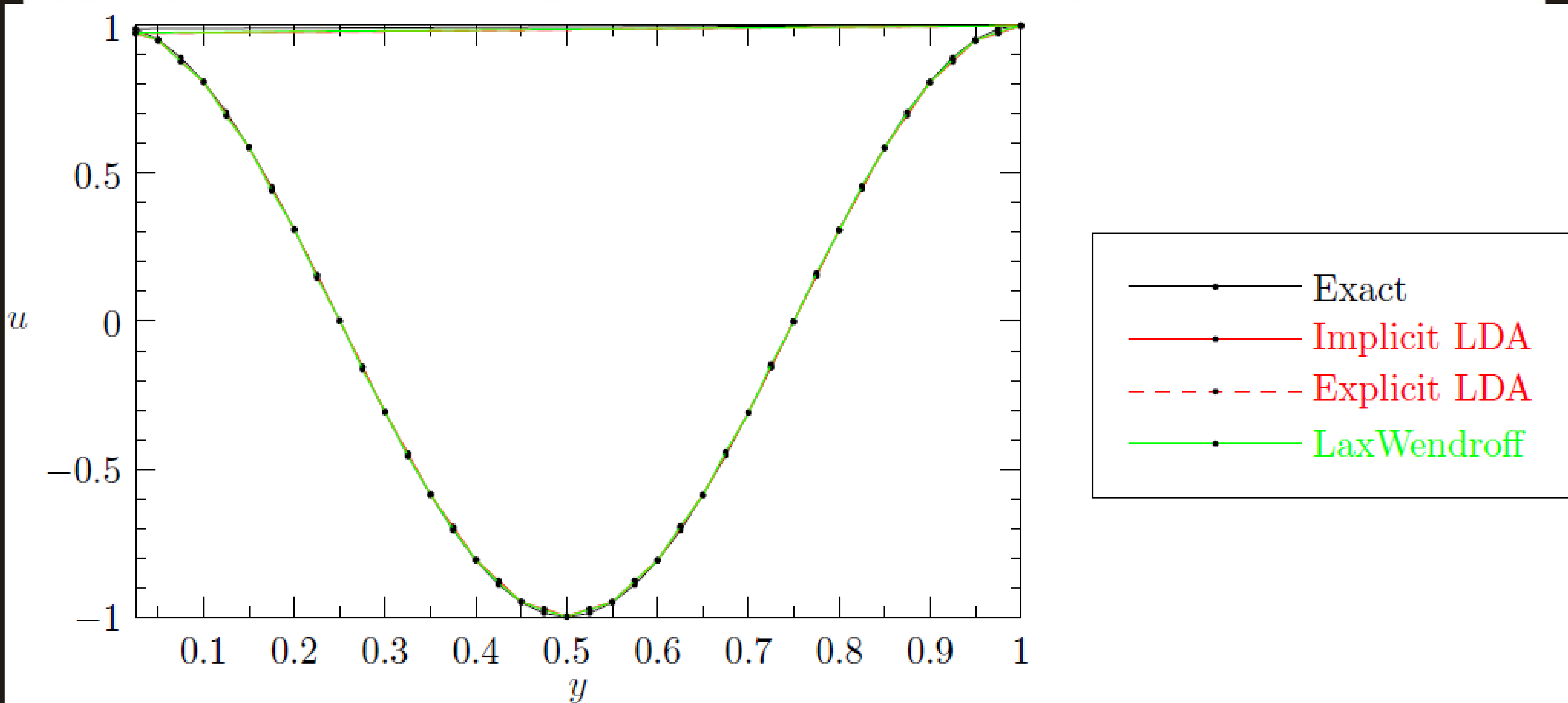
Two level time step & dual time stepping

40 x 40 grid



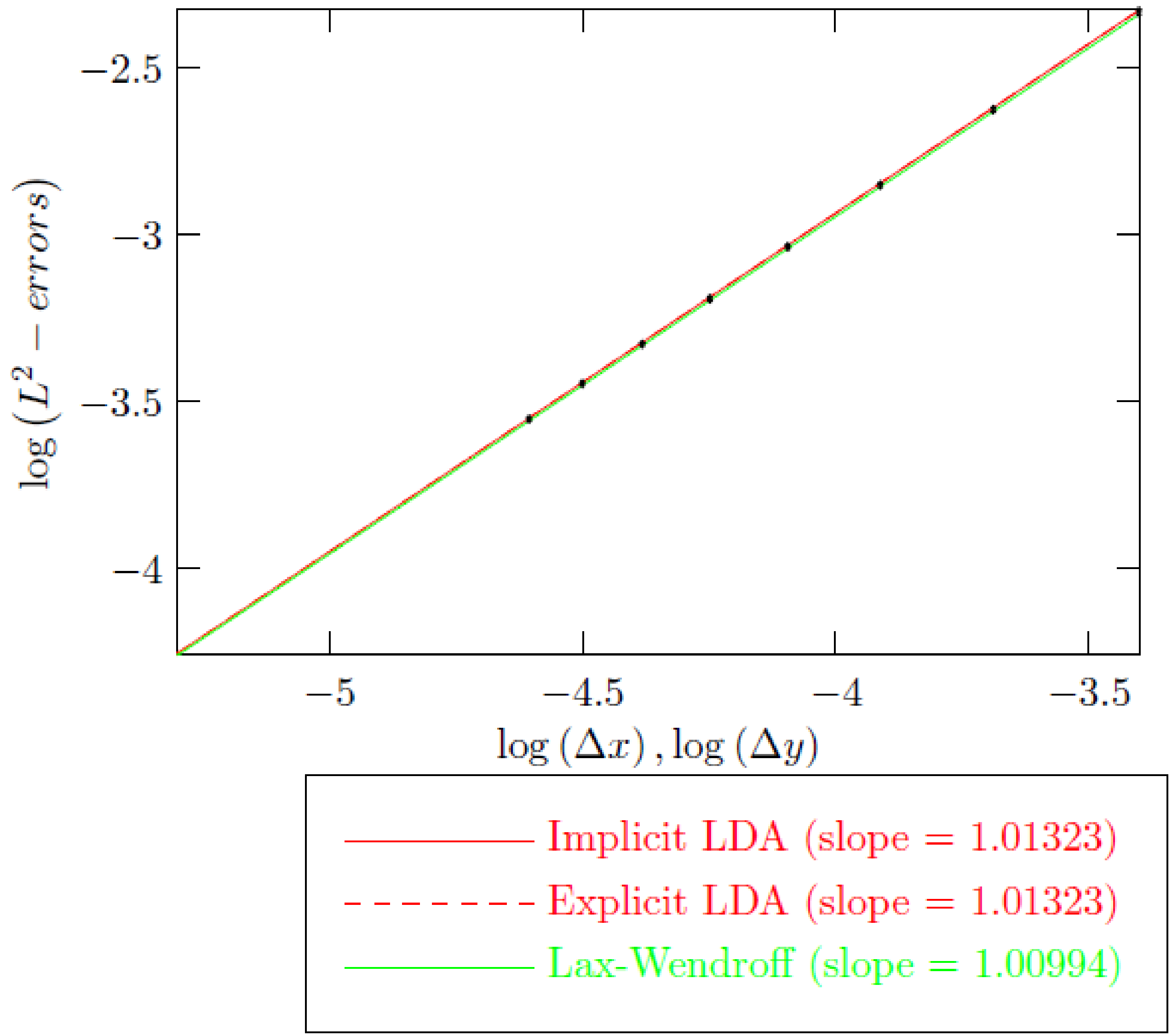
Three level time step only

40 x 40 grid



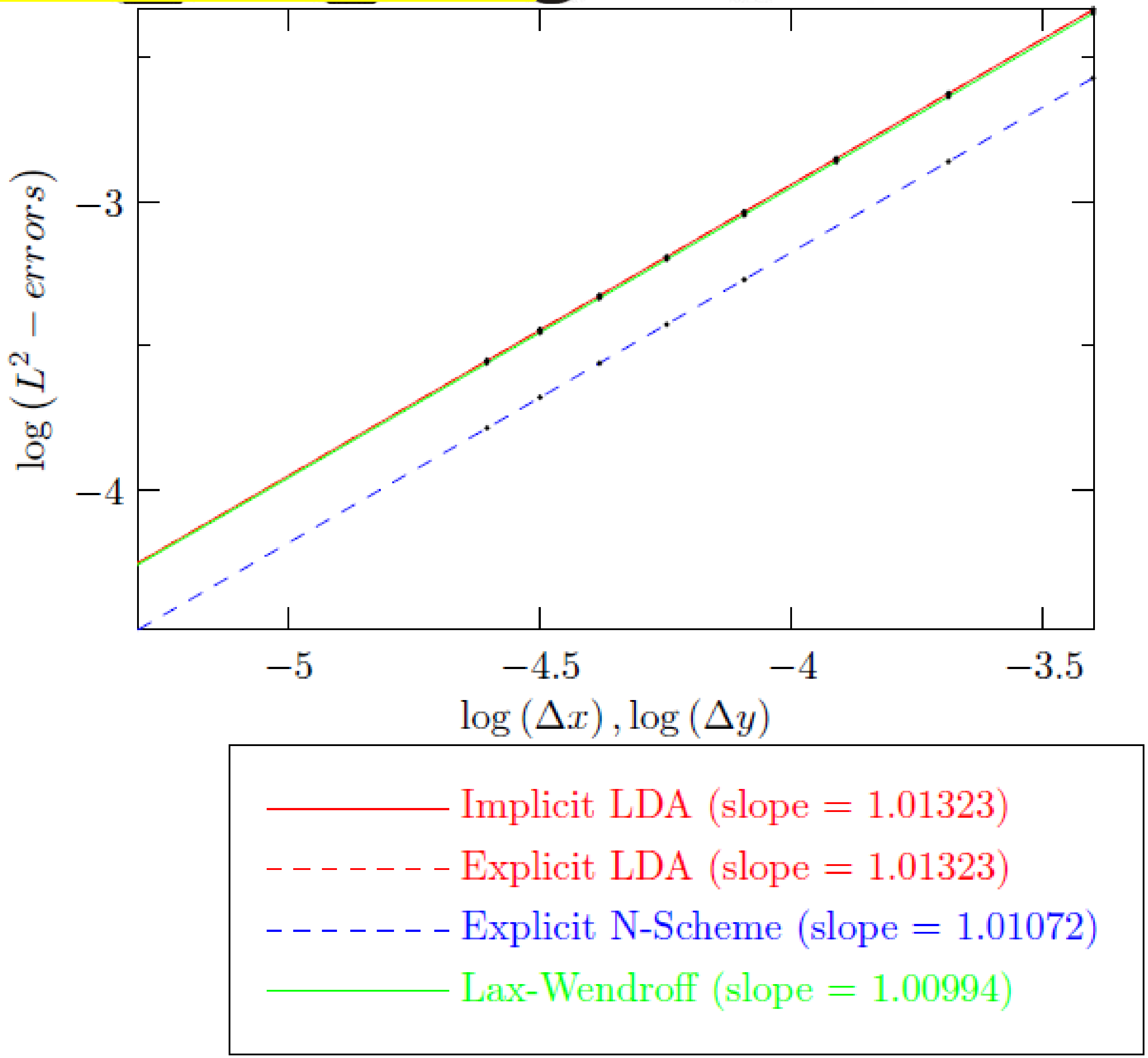
Two level time step only

40 x 40 grid



Two level time step & dual time stepping

40 x 40 grid



Three level time step only

40 x 40 grid