University Course

Math 228B
Numerical Solution of Differential Equations

University of California, Davis

Spring quarter, 2011

My Class Notes

Nasser M. Abbasi

Spring 2011

Contents

Chapter 1

Introduction

I took this course in winter 2011 to learn about numerical solutions of PDE's.

1.1 course description from catalog

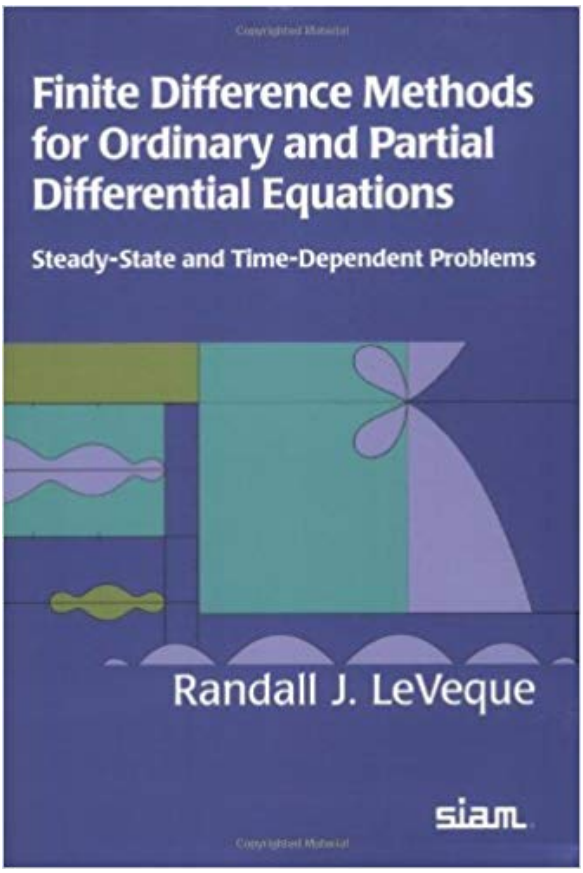
228A-228B-228C. Numerical Solution of Differential Equations (4-4-4) Lecture 3 hours; term paper or discussion 1 hour. Prerequisite: course 128C. Numerical solutions of initial-value, eigenvalue and boundary-value problems for ordinary differential equations. Numerical solution of parabolic and hyperbolic partial differential equations. Offered in alternate

years.

1.2 class syllabus

See Dr Guy's web page `http://www.math.ucdavis.edu/~guy/teaching/228b/index.html`

1.3 Text book



Chapter 2

my study notes

2.1 How to decide which fractional stepping to use?

Given a mixed PDE such as $u_t = Au + Bu$ where A, B are constant matrices.

Let standard stepping be

$$u^* = N_A(u^n, k)$$
$$u^{n+1} = N_B(u^*, k)$$

Where N_A and N_B are numerical schemes to solve the problem $u_t = Au$ and $u_t = Bu$ respectively. k in the above is the time step.

Let Strang splitting be

$$u^* = N_A(u^n, k/2)$$
$$u^{**} = N_B(u^*, k)$$
$$u^{n+1} = N_A(u^{**}, k/2)$$

Now, assuming that N_A and N_B are each second order accurate in time. Which of the above two schemes should one select?

Algorithm

```
---- standard stepping
IF A,B commute THEN
   standard stepping is second order in time
ELSE
   standard stepping is first order in time
END IF
```

```
---- Strang
IF A,B commute THEN
   strang gives second order accuracy in time
ELSE
   strang also gives second order accuracy in time
END IF
```

Hence, from the above, the conclusion is that

```
IF A,B commute THEN
   select standard stepping (simpler)
ELSE
   select Strang (more accurate)
END IF
```

some notes from the net HTML

Chapter 3

HWs

Local contents

3.1 Table summary

HW	description
1	refinement study 1D for diffusion, successive errors, diffusion-advection
2	Using Von Nuemann analysis on Peaceman-Rachford ADI. Numerical solution for diffusion pde using ADI scheme on 2D grid. Discrete conservation, refinment study on grid centered in 2D. Numerical solution of FitzHugh-Nagumo equations. Different initial conditions.
3	refinement study 1D for advection, Lax-Wendroff, C-N for advection, TVD, periodic boundary conditions.
4	Solve wave equation using Lax-Wendroff as system, cell centered grid. Proof that TVD scheme is monotone preserving, Finite volume solution for first order advection ode using different numerical flux functions.

3.2 HW 1

Local contents

3.2.1 Problem1

1. Consider the following PDE.

$$u_t = 0.01\, u_{xx} + 1 - \exp(-t), \quad 0 < x < 1$$
$$u(0,t) = 0 \quad u(1,t) = 0$$
$$u(x,0) = 0$$

   (a) Write a program to solve the problem using Crank-Nicolson up to time $t = 1$, and perform a refinement study that demonstrates that the method is second-order accurate in space and time.

   (b) Solve the problem using a forward Euler method up to time $t = 1$. Demonstrate in a refinement study that the method is first-order in time and second-order in space.

Figure 3.1: Problem description

The goal of a refinement study is to perform a numerical experiment to determine the order of accuracy of a given finite difference scheme. The appendix of this problem contain a review of the idea behind refinement study.

The problem asked us to determine the order of accuracy in time and in space. A program implementing the above scheme was run a number of times, each time with a different initial value for the space and time step. To verify order of accuracy for the C-N scheme, the space and the time step were divided by 2 simultaneously before the start of each run. To verify order of accuracy for the forward Euler scheme, the space step was divided by 2 but the time was divided by 4. For both schemes , the program generated ratios of successive errors between the numerical solutions at the end of each run (1 second long run).

Convergence of this ratio to the value 4 implied the results we are asked to demonstrate.

In the following, the C-N and the forward Euler finite difference schemes are derived, then the numerical results presented, followed by a conclusion.

3.2.1.1 Part (a)

The method of lines (MOL) was used to implement the C-N scheme to solve for the numerical solution u. The equations are solved using Matlab's $u = A\backslash b$ where A is a sparse matrix (the system update matrix) constructed based on the C-N discretization. An efficient algorithm to solve for u in this scheme is Thomas algorithm version of Gaussian elimination. It is understood that this will automatically be done by Matlab "\" operator when it recognizes that the A matrix is a tridiagonal giving an $O(n)$ order for the solver where n is the number of unknowns.

Let the PDE be

$$d\ u_t - Du_{xx} + au = g(x,t) \tag{1}$$

$g(x,t)$ is an internal source with initial conditions as $u(x,t) = u0(x)$. The Dirichlet boundary conditions are

$$\begin{cases} u(0,t) = \alpha(t) \\ u(L,t) = \beta(t) \end{cases}$$

and Neumann boundary conditions are

$$\begin{cases} u_t(0,t) = \alpha(t) \\ u_t(L,t) = \beta(t) \end{cases}$$

The terms d, a above are constants, and D is the diffusion constant. For the C-N scheme (1) was discretized at point x_j with space step as h and with time step as k resulting in

$$d\ \frac{u_j^{n+1} - u_j^n}{k} = \frac{1}{2}\left(f_j^n + f_j^{n+1}\right)$$

Where f^n is the RHS of the PDE at time $t_n = nk$, so the above becomes

$$d\ \frac{u_j^{n+1} - u_j^n}{k} = \frac{1}{2}\left[\left(Du_{xx} - au + g_j\right)^n + \left(Du_{xx} - au + g_j\right)^{n+1}\right]$$

$$= \frac{1}{2}\left(D\frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{h^2} - au_j^n + g_j^n\right) + \left(D\frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{h^2} - au^{n+1} + g_j^{n+1}\right)$$

$$= \frac{D}{2h^2}\left(u_{j-1}^n - 2u_j^n + u_{j+1}^n\right) + \frac{D}{2h^2}\left(u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}\right) - \frac{a}{2}\left(u_j^n + u_j^{n+1}\right) + \frac{1}{2}\left(g_j^n + g_j^{n+1}\right)$$

collecting all terms at time $n+1$ to the left gives

$$u_j^{n+1}\left(1 + \frac{kD}{dh^2} + \frac{ak}{2d}\right) - \frac{kD}{2dh^2}u_{j-1}^{n+1} - \frac{kD}{2dh^2}u_{j+1}^{n+1} = u_j^n\left(1 - \frac{kD}{dh^2} - \frac{ak}{2d}\right) + u_{j-1}^n\frac{kD}{2dh^2} + u_{j+1}^n\frac{kD}{2dh^2} + \frac{k}{2d}\left(g_j^n + g_j^{n+1}\right)$$

Let

$$r_1 = \left(1 + \frac{kD}{dh^2} + \frac{ak}{2d}\right)$$
$$r_2 = \frac{kD}{2dh^2}$$
$$r_3 = \left(1 - \frac{kD}{dh^2} - \frac{ak}{2d}\right)$$
$$r_4 = \frac{k}{2d}$$

Then the above becomes

$$r_1 u_j^{n+1} - r_2 u_{j-1}^{n+1} - r_2 u_{j+1}^{n+1} = r_3 u_j^n + r_2 u_{j-1}^n + r_2 u_{j+1}^n + r_4\left(g_j^n + g_j^{n+1}\right) \tag{2}$$

The above algebraic equation (2) is the C-N finite difference scheme for (1) and is valid for x_j at the internal points. Considering the case of both ends having Dirichlet boundary conditions, and using the following grid numbering[1]

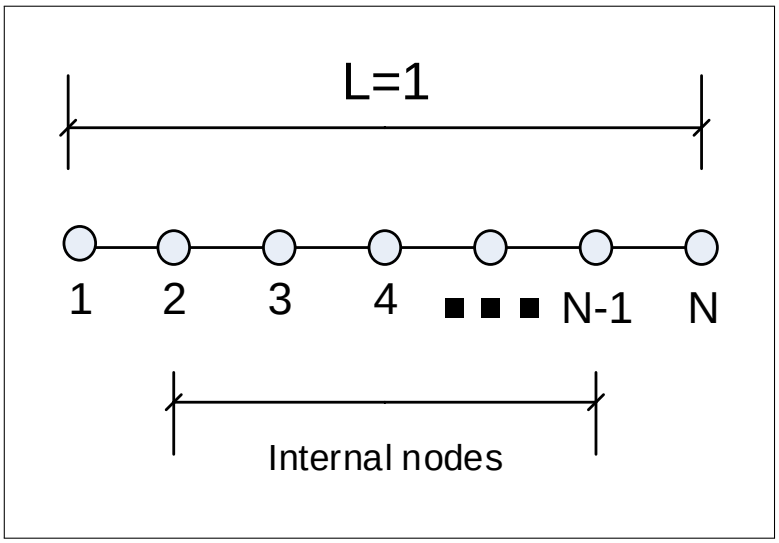


Figure 3.2: Problem grid format

Then (2) above is valid at the internal nodes numbered $j = 2 \cdots N-1$. Hence u_1^n will be the left boundary point and u_N^n will be the right boundary point. When the boundary conditions are Dirichlet, let $u_1^n = \alpha(t_n)$ and $u_N^n = \beta(t_n)$. Converting (2) to matrix form results in

$$\overbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_1 & -r_2 & 0 & 0 & 0 & 0 \\ 0 & -r_2 & r_1 & -r_2 & 0 & 0 & 0 \\ 0 & 0 & -r_2 & r_1 & -r_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & \vdots & 0 \\ 0 & 0 & 0 & 0 & -r_2 & r_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}^{\mathbf{A}} \overbrace{\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{N-2}^{n+1} \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix}}^{\mathbf{x}} = \overbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_3 & r_2 & 0 & 0 & 0 & 0 \\ 0 & r_2 & r_3 & r_2 & 0 & 0 & 0 \\ 0 & 0 & r_2 & r_3 & r_2 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & 0 & 0 & 0 & r_2 & r_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ u_4^n \\ \vdots \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} \alpha^{n+1} \\ r_2\tilde{\alpha} + r_4\tilde{g}_2 \\ r_4\tilde{g}_3 \\ r_4\tilde{g}_4 \\ \vdots \\ r_2\tilde{\beta} + r_4\tilde{g}_{N-1} \\ \beta^{n+1} \end{bmatrix}}^{\mathbf{b}}$$

Where in (3) $\alpha(t_{n+1}) + \alpha(t_n) \equiv \tilde{\alpha}$ and $\beta(t_{n+1}) + \beta(t_n) \equiv \tilde{\beta}$ and $g^n + g^{n+1} \equiv \tilde{g}$.

Equation (3) is in the form $Au^{n+1} = b$. And u^{n+1} is solved for using $A\backslash b$. Notice that (3) is in the same form shown in class notes, which is

$$\left(I - \frac{Dk}{2}L\right)u^{n+1} = \left(I + \frac{Dk}{2}L\right)u^n + kf^{n+\frac{1}{2}} \tag{4}$$

[1]This is slightly different from the standard numbering format we used before.

Where the update matrix $B = \left(I - \frac{Dk}{2}L\right)^{-1}\left(I + \frac{Dk}{2}L\right)$. L is the standard Laplace operator for $1D$ problem given by

$$\begin{bmatrix} -2 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & -2 \end{bmatrix}$$

Notice that (3) compared to (4), has additional terms included in the RHS in order to support the general form the parabolic PDE. Equation (4) represents the diffusion pde $u_t - Du_{xx} = 0$.

u_j^n when $n = 0$ is obtained from initial conditions. The first step solves for u_j^1 which is then used in the second second step to solve for u_j^2 and so on, until the maximum time to solve for is reached. Since Dirichlet boundary conditions are used, u_1^n (the solution at the left edge) and u_N^n, the solution at the right edge are always known. The above system is solved only for the internal nodes. Next section shows the numerical results.

3.2.1.1.1 Result for part(a) The above scheme was implemented with a GUI added to make it easier to use these algorithms. The following plot shows the numerical solution at $t = 1$.

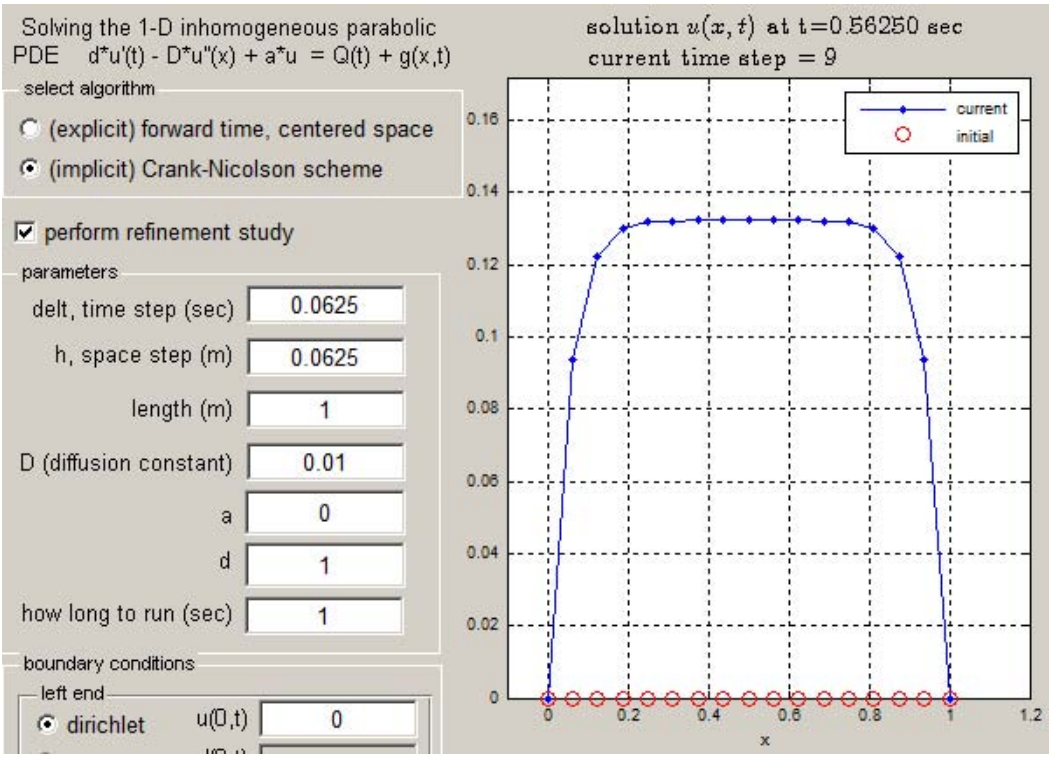


Figure 3.3: shows the numerical solution at $t = 1$

The following is the ratio error table. This table shows that the ratio converged to 4.

#	delt	h	ratio
2	0.2500000	0.2500000	1.0000e+000
3	0.1250000	0.1250000	2.9347e+000
4	0.0625000	0.0625000	3.6798e+000
5	0.0312500	0.0312500	4.2132e+000
6	0.0156250	0.0156250	4.1209e+000
7	0.0078125	0.0078125	4.0354e+000
8	0.0039063	0.0039063	4.0092e+000

The following is the loglog plot of the above result. The x-axis represents h and the y-axis the difference in errors (absolute). The slope of the line is seen to be 2 implying a second order accuracy.

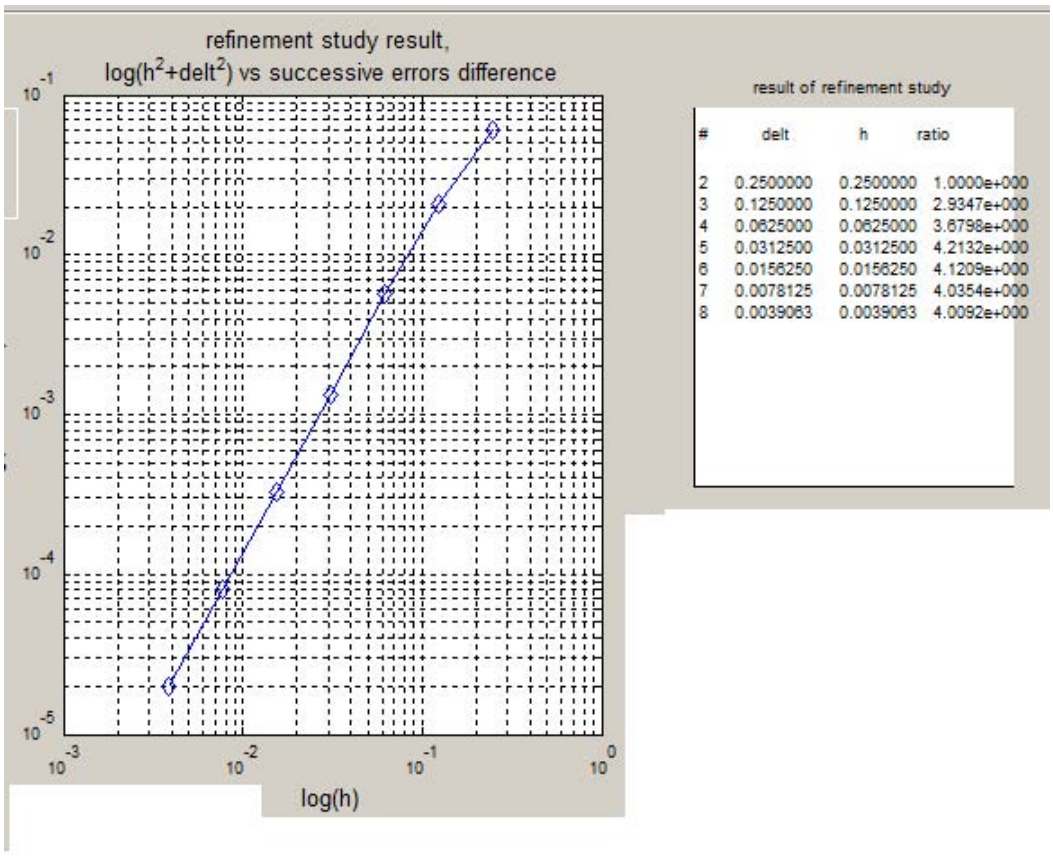


Figure 3.4: log log plot

Conclusion Since the ratio is 4 and since the time step and the space step were halved in each run, this implies C-N is second order in time and space.

3.2.1.2 Part(b)

Starting with the same PDE as in part (a)

$$d\ u_t - Du_{xx} + au = g(x,t) \tag{1}$$

All terms and boundary conditions and the solution domain are as shown in part (a).

For the forward Euler scheme (1) was discretized at point x_j with space step as h and with time step as k as follows

$$\begin{aligned} d\ \frac{u_j^{n+1} - u_j^n}{k} &= f_j^n \\ &= D\frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{h^2} - au_j^n + g_j^n \end{aligned}$$

Hence

$$\begin{aligned} u_j^{n+1} &= u_j^n + \frac{kD}{dh^2}\left(u_{j-1}^n - 2u_j^n + u_{j+1}^n\right) - \frac{ak}{d}u_j^n + \frac{k}{d}g_j^n \\ &= \frac{kD}{dh^2}u_{j-1}^n + u_j^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_{j+1}^n + \frac{k}{d}g_j^n \end{aligned}$$

Let $r_1 = \left(1 - 2\frac{kD}{dh^2} - \frac{ak}{d}\right), r_2 = \frac{kD}{dh^2}, r_3 = \frac{k}{d}$, the above becomes

$$u_j^{n+1} = r_2 u_{j-1}^n + r_1 u_j^n + r_2 u_{j+1}^n + r_3 g_j^n \tag{2}$$

The above algebraic equation (2) is the forward Euler finite difference scheme for (1) and is valid for x_j at the internal points.

Therefore, the stencil for the forward Euler scheme for the 1D parabolic PDE is

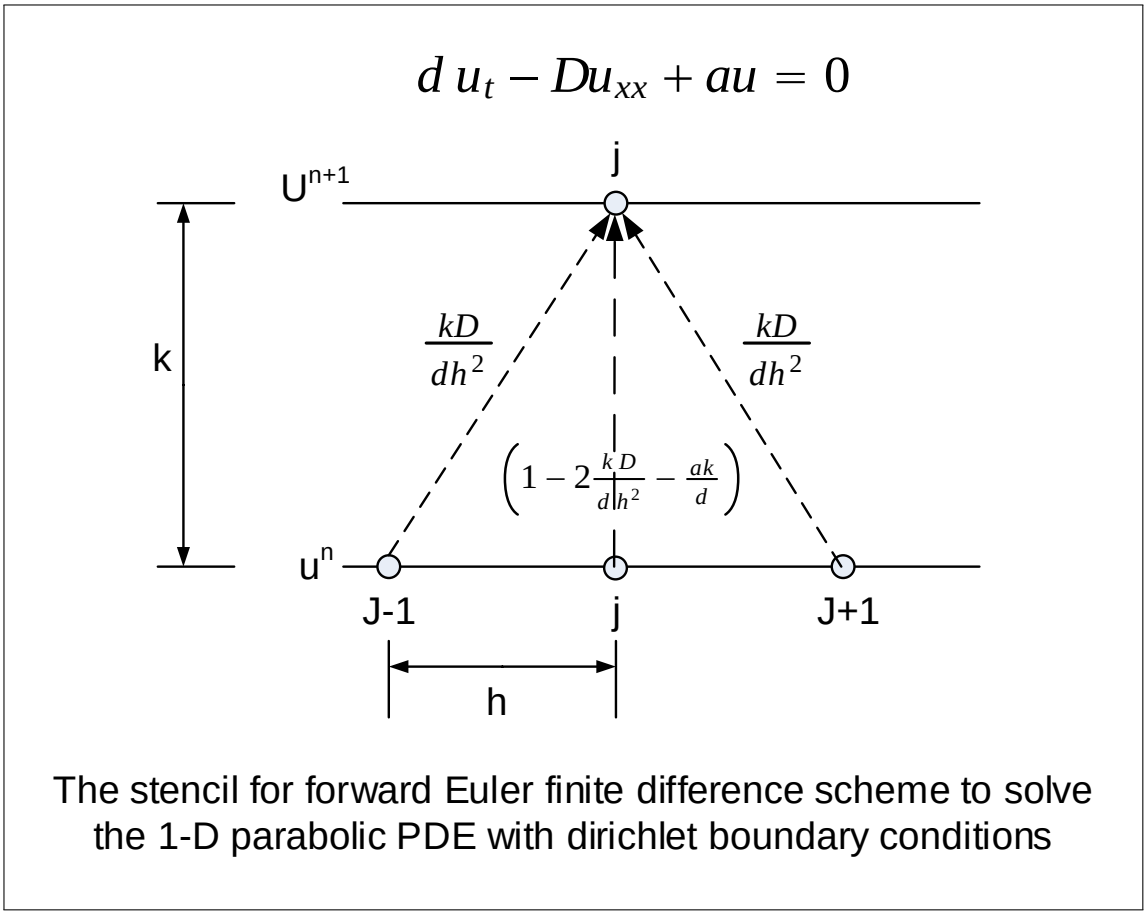


Figure 3.5: stencile for forward Euler

Considering the case of both ends having Dirichlet boundary conditions, and using the same numbering as in part (a) then (2) is valid at the internal nodes $j = 2 \cdots N-1$.

u_1^n will be the left boundary point and u_N^n will be the right boundary point. Let $u_1^n = \alpha(t_n)$ and $u_N^n = \beta(t_n)$. Converting (2) to matrix form results in

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ u_4^{n+1} \\ \vdots \\ u_{N-2}^{n+1} \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_1 & r_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_2 & r_1 & r_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & r_2 & r_1 & r_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & r_2 & r_1 & r_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & r_2 & r_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ u_4^n \\ \vdots \\ u_{N-2}^n \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} \alpha^{n+1} \\ r_2\alpha^n + r_3 g_2^n \\ r_3 g_3^n \\ r_3 g_4^n \\ \vdots \\ r_3 g_{N-2}^n \\ r_2\beta^n + r_3 g_{N-1}^n \\ \beta^{n+1} \end{bmatrix}$$

And now u^{n+1} is found by direct matrix/vector multiplication as shown above. No matrix inversion is required in this case since this is an explicit method.

Looking at the stencil above, an idea is now suggested to determine stability directly from the stencil diagram. By imposing that the weight on each edge in the directed graph not exceed unity, and that the total algebraic sum of the weight of the edge also not exceed unity. This includes any combination of edges involved. If this is always the case, then u_j^{n+1} will always have an amplitude $\le u_j^n$ since the weights are never more than 1 no matter what combinations are used. This idea is applied to this problem with $u_t + Du_{xx} = 0$, hence $a = 0$ and $d = 1$. This gives that following conditions on the edges shown in the stencil diagram above

$$\begin{cases} (1) & \frac{kD}{h^2} \le 1 & \text{Condition on } j-1 \text{ or } j+1 \text{ separately} \\ (2) & 2\frac{kD}{h^2} \le 1 & \text{Condition on } j-1 \text{ and } j+1 \text{ added together} \\ (3) & \left|1 - 2\frac{k\ D}{h^2}\right| \le 1 & \text{Condition on the } j \text{ edge} \\ (4) & \left|\frac{kD}{h^2} + 1 - 2\frac{k\ D}{h^2}\right| \le 1 & \text{Condition on the } j \text{ edge with either } j-1 \text{ or } j+1 \\ (5) & \left|2\frac{kD}{h^2} + 1 - 2\frac{k\ D}{h^2}\right| \le 1 & \text{Condition that all edges sum to less than 1} \end{cases}$$

Condition (1) is weaker than (2), hence not considered. Condition (3) results in $\frac{kD}{h^2} \le 1$ which is the same as (1). Condition (4) gives $\left|1 - \frac{k\ D}{h^2}\right| \le 1$ or $\frac{k\ D}{h^2} \le 2$ which is also weaker than (2). Condition (5) gives $1 \le 1$ hence no information is obtained from it. Therefore, condition (2) remains, and that condition says that $\frac{kD}{h^2} \le \frac{1}{2}$, which is the strongest condition. Hence, this is the absolute condition for stability for forward Euler. This agrees with the method to determine this using Von Neumann analysis.

3.2.1.2.1 Result for part(b) The forward Euler results are below. The space step was divided by 2 and the time step was divided by 4.

```
#       delt             h          ratio
2     0.0625000     0.2500000   1.0000e+000
```

3	0.0156250	0.1250000	3.0842e+000
4	0.0039063	0.0625000	3.7567e+000
5	0.0009766	0.0312500	4.1506e+000
6	0.0002441	0.0156250	4.0794e+000
7	0.0000610	0.0078125	4.0447e+000
8	0.0000153	0.0039063	3.9909e+000

The following is the corresponding loglog plot

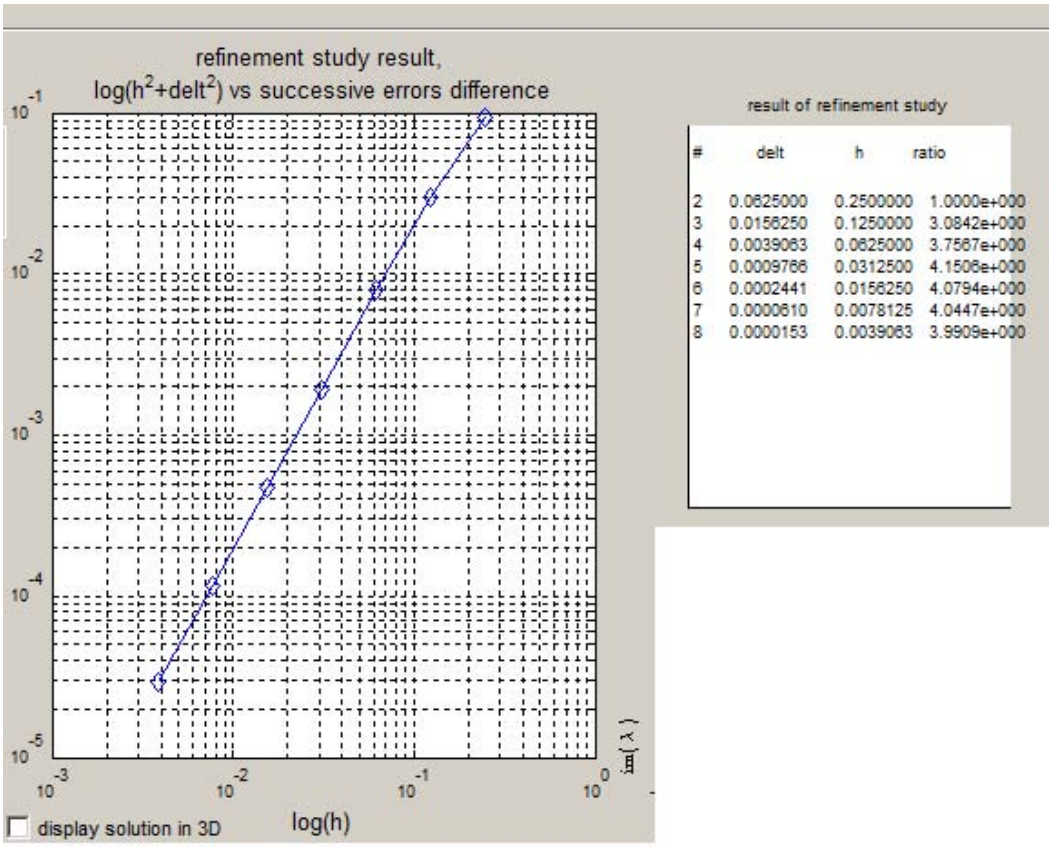


Figure 3.6: corresponding loglog plot

Conclusion Since the ratio is 4 and since the time step was divided by 4 and the space step by 2, this implies forward Euler is first order in time and second order in space.

The appendix of this problem show the steady state analytical solution to the above PDE derived using Laplace transform method.

3.2.1.3 Problem 1 appendix

3.2.1.3.1 Review of refinement study process The idea behind refinement study is reviewed briefly. Assume the goal is to find the order of accuracy of a finite difference scheme with respect to the space step. The finite difference formula is first derived, and the exact solution is substituted into this formula. Terms that contain $u(x \pm h)$, are replaced by Taylor series approximation. The result is simplified, and the error term is found. An small example is given to illustrate the idea.

To find the order of accuracy in space using forward Euler finite difference approximation to a derivative

$$u'(x) = \frac{u(x+h) - u(x)}{h}$$

Each term for u in the RHS above is replaced by its exact value, using Taylor series expansion where needed, resulting in

$$\begin{aligned} u'(x) &= \frac{\left[u(x) + hu'(x) + \frac{h^2}{2}u''(x) + \frac{h^3}{3!}u'''(x) + \cdots\right] - u(x)}{h} \\ &= u'(x) + \overbrace{\frac{h}{2}u''(x) + \frac{h^2}{3!}u'''(x) + \cdots}^{error} \end{aligned} \tag{1}$$

The error term is hence. It is the amount that the RHS differs from the LHS. The leading error term in (1) (the dominant term) is $\frac{h}{2}u''(x)$, but since x is a known value (the above is being evaluated at each grid point, hence x is known), then $u''(x)$ is some constant, and the leading error term in (1) is of the form Ch, where C is some constant. This is the same as saying that the error is of order h.

The above method can be used to find the order of the error in approximation when the exact solution is know. In problem (1), the exact solution is not given and was difficult to obtain. Hence, instead of finding the order of accuracy using the above method, it was found using a numerical experiment (refinement study).

In the refinement study the error itself is determined, and from the error profile (as h is changed), the order is determined. But this error is the error between successive numerical solutions.

Once the numerical error is found (after running the refinement study), then one method to find p (order of the error) is to take the logarithm resulting in

$$\begin{aligned} error &= Ch^p \\ \log(error) &= p\log(h) + \log C \\ &= p\log(h) + \text{constant} \end{aligned}$$

and this represents an equation of the line $Y = pX + k$, where p is the line slope which is the same as the order of accuracy. Hence, by generating different h values, and for each h determine the corresponding error, then p is found by measuring at the slope of line from

the plot generated. If the slope is $p = 1$, then it is first order accuracy, and if the slope is $p = 2$, it is second order.

The above is a graphical method. Another method is as follows: Starting with some h value, the error e_{n-1} is found, then h is divided by half and the error, now called e_n is found again. The ratio $\frac{e_{n-1}}{e_n}$ is found. If p happened to be 2, then the ratio will come out to be 4. This is because $\frac{e_{n-1}}{e_n} = \frac{(h_{n-1})^p}{\left(\frac{h_{n-1}}{2}\right)^p} = 2^p$ and so if $p = 2$, then the ratio will be 4. If $p = 1$, then the ratio will be 2.

In the above description, errors are found using differences between successive solutions as follows

$$e_{n-1} = |U_{n+1} - U_n|$$
$$e_n = |U_n - U_{n-1}|$$

The norm used to measure U, the approximate solution, is the Euclidean norm modified for the space grid

$$\|U\| = \sqrt{h}\,\|U\|_2$$

3.2.1.3.2 Steady state analytical solution to the PDE The following shows the steps used to determine the steady state solution for

$$u_t = a u_{xx} + f(t) \tag{1}$$

where $a = \frac{1}{100}$ and $f(t) = 1 - e^{-t}$ with initial conditions $u(0,t) = 0$ and boundary conditions $u(0,t) = 0$ and $u(1,t) = 0$.

The above is an inhomogeneous PDE (the source term $1 - e^{-t}$). The boundary conditions are homogeneous, and with zero initial conditions.

Since this is an inhomogeneous PDE, separation of variables can not be used. But the steady state solution (the particular solution) can be found using an integral transform approach. Integral transformation is first applied to the PDE, resulting in an ODE which is then solved in the new transformed space, and the solution in time domain is found by inverse transforming back.

Since the spatial domain in this problem is a bounded interval (from 0 to 1), Fourier transformation will not be used because the spatial domain is bounded and does not match the Fourier transformation domain (from $-\infty$ to $+\infty$), however, Laplace transformation (for $t > 0$) can be used as it matches the time domain of the problem.

Therefore, taking the Laplace transform of (1) w.r.t time gives

$$-u(x,0) + sU(x,s) = a\frac{d^2U(x,s)}{dx^2} + \frac{1}{s} - \frac{1}{1+s}$$

But $u(x,0) = 0$, hence the resulting ODE is

$$a\frac{d^2U(x,s)}{dx^2} - sU(x,s) = \frac{1}{1+s} - \frac{1}{s}$$

With the boundary conditions $U(0,s) = 0$ and $U(1,s) = 0$ obtained from the spatial domain. The above ODE is a second order, linear ODE, a inhomogeneous ODE that can be solved for $U(x,s)$, which results in the following (for the case $a = \frac{1}{100}$)

$$U(x,s) = \frac{-e^{-10x\sqrt{s}}\left(e^{10x\sqrt{s}-1} - 1\right)\left(e^{10x\sqrt{s}} - e^{10\sqrt{s}}\right)}{s^2(1+s)\left(1+e^{10\sqrt{s}}\right)}$$

The steady state solution can now be found using the limit theorem for Laplace transform, giving

$$u(x,\infty) = \lim_{s->0} sU(x,s) \tag{1}$$

$$= 50x(1-x) \tag{3.1}$$

Here is a plot of the particular solution

```
x=0:0.01:1; plot(x,50*x.*(1-x))
```

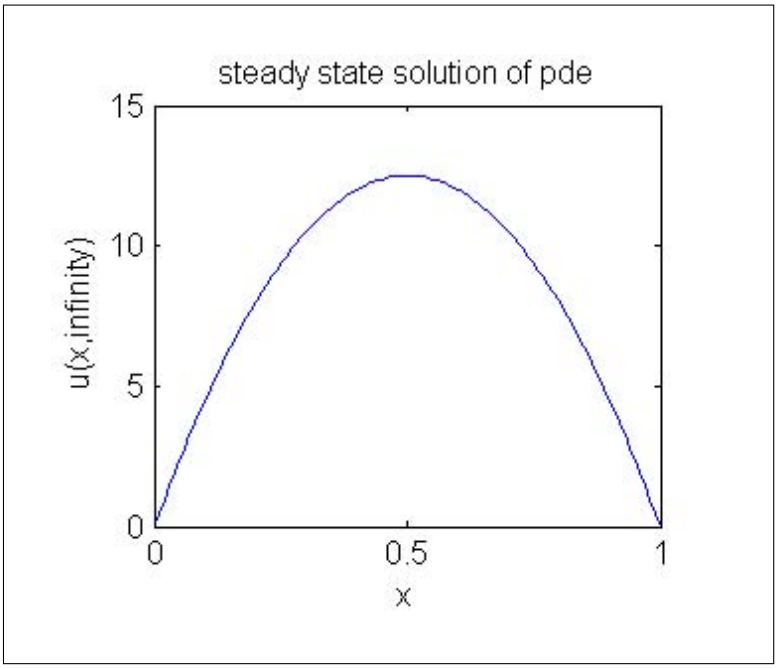


Figure 3.7: steady state plot. PDE solution

3.2.1.3.3 Derivation of forward Euler for periodic boundary conditions From part (b) above,

$$u_j^{n+1} = \frac{kD}{dh^2}u_{j-1}^n + u_j^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_{j+1}^n + \frac{k}{d}g_j^n$$

Periodic boundary conditions implies $u(0,t) = u(1,t)$, Hence u_{j-1} when j is the first note on the left is the same as node $N-1$. And u_{j+1} when j is the last node on the right is the same as node $j = 2$. As shown in the diagram below

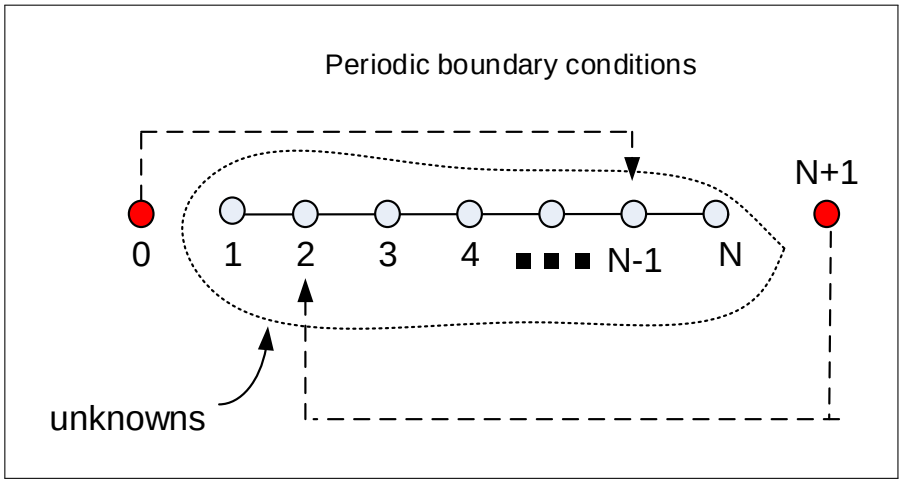


Figure 3.8: Grid format

Then

$$u_1^{n+1} = \frac{kD}{dh^2}u_{N-1}^n + u_1^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_2^n$$

And

$$u_N^{n+1} = \frac{kD}{dh^2}u_{N-1}^n + u_N^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_2^n$$

Let $r_1 = \left(1 - 2\frac{kD}{dh^2} - \frac{ak}{d}\right), r_2 = \frac{kD}{dh^2}, r_3 = \frac{k}{d}$, Hence the system can be written as

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ u_4^{n+1} \\ \vdots \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & 0 & 0 & 0 & r_2 & 0 \\ r_2 & r_1 & r_2 & 0 & 0 & 0 & 0 \\ 0 & r_2 & r_1 & r_2 & 0 & 0 & 0 \\ 0 & 0 & r_2 & r_1 & r_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & 0 & r_2 & r_1 & r_2 \\ 0 & r_2 & 0 & 0 & 0 & r_2 & r_1 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ u_4^n \\ \vdots \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} 0 \\ r_3 g_2^n \\ r_3 g_3^n \\ r_3 g_4^n \\ \vdots \\ r_3 g_{N-1}^n \\ 0 \end{bmatrix}$$

3.2.1.3.4 Derivation of forward Euler for Neumann boundary conditions both ends

Using this numbering

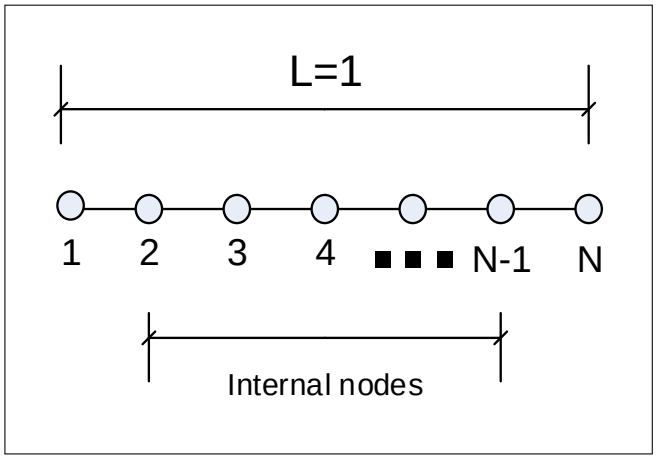


Figure 3.9: Grid format

Assume that $u_t = \alpha$ at node 1 and $u_t = \beta$ at node N (these are the Neumann boundary conditions).

Add a ghost node 0 to the left of node 1, and approximating $\alpha(t)$ gives

$$\alpha = \frac{u_0 - u_2}{2h}$$

hence

$$u_0 = 2h\alpha + u_2 \tag{1}$$

But the PDE for node 1 is

$$u_1^{n+1} = \frac{kD}{dh^2}u_0^n + u_1^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_2^n \tag{2}$$

Substitute (1) into (2) gives

$$\begin{aligned} u_1^{n+1} &= \frac{kD}{dh^2}(2h\alpha + u_2) + u_1^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_2^n \\ &= u_1^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + u_2^n\left(2\frac{kD}{dh^2}\right) + 2h\alpha\frac{kD}{dh^2} \end{aligned}$$

Similarly for the right end. Add a ghost node $N+1$ to the right of node N, and approximating $\beta(t)$ gives

$$\beta = \frac{u_{N-1} - u_{N+1}}{2h}$$

hence

$$u_{N+1} = 2h\beta + u_{N-1} \tag{3}$$

But the PDE for node N is

$$u_N^{n+1} = \frac{kD}{dh^2}u_{N-1}^n + u_N^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_{N+1}^n \tag{4}$$

Substitute (3) into (4) gives

$$\begin{aligned} u_1^{n+1} &= \frac{kD}{dh^2}u_{N-1}^n + u_N^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}\left(2h\beta + u_{N-1}\right) \\ &= 2\frac{kD}{dh^2}u_{N-1}^n + u_N^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + 2h\beta\frac{kD}{dh^2} \end{aligned}$$

Nodes $j = 2 \cdots N-1$ remain the same as before. In other words

$$u_j^{n+1} = \frac{kD}{dh^2}u_{j-1}^n + u_j^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_{j+1}^n + \frac{k}{d}g_j^n$$

Let $r_1 = \left(1 - 2\frac{kD}{dh^2} - \frac{ak}{d}\right), r_2 = \frac{kD}{dh^2},, r_3 = \frac{k}{d}$, Hence the system becomes (now nodes 1 and N are unknowns and added)

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ u_4^{n+1} \\ \vdots \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix} = \begin{bmatrix} r_1 & 2r_2 & 0 & 0 & 0 & 0 & 0 \\ r_2 & r_1 & r_2 & 0 & 0 & 0 & 0 \\ 0 & r2 & r_1 & r_2 & 0 & 0 & 0 \\ 0 & 0 & r_2 & \ddots & r_2 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & r_2 & r_1 & r_2 \\ 0 & 0 & 0 & 0 & 0 & 2r_2 & r_1 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ u_4^n \\ \vdots \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} 2h\alpha^n\ r_2 \\ r_3 g_2^n \\ r_3 g_3^n \\ r_3 g_4^n \\ \vdots \\ r_3 g_{N-1}^n \\ 2h\beta^n\ r_2 \end{bmatrix}$$

3.2.1.3.5 Derivation of forward Euler for Neumann on left and Dirichlet on right Using this numbering

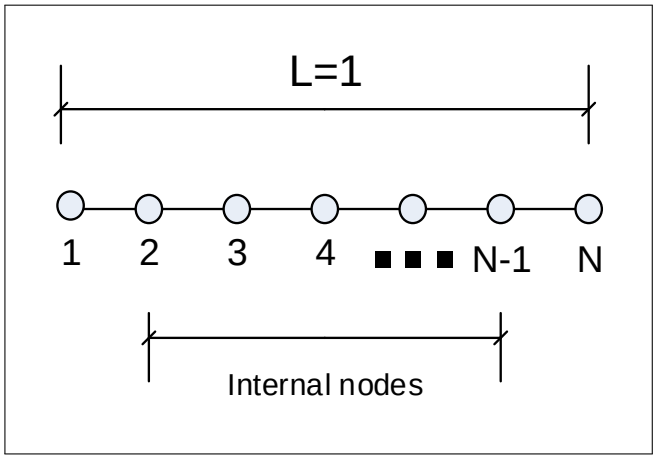


Figure 3.10: Grid format

Assume that $u_N = \beta$ at node N and $u_t = \alpha$ at node 1 (these are the Dirichlet and Neumann boundary conditions).

The unknowns in this case are nodes $1 \cdots N-1$, since u_N is known from Dirichlet boundary conditions. Add a ghost node 0 to the left of node 1, and approximating α gives

$$\alpha = \frac{u_0 - u_2}{2h}$$

hence

$$u_0 = 2h\alpha + u_2 \tag{3}$$

But the PDE for node 1 is

$$u_1^{n+1} = \frac{kD}{dh^2}u_0^n + u_1^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_2^n \tag{4}$$

Substitute (3) into (4) gives

$$\begin{aligned} u_1^{n+1} &= \frac{kD}{dh^2}(2h\alpha + u_2) + u_1^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_2^n \\ &= u_1^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \left(2\frac{kD}{dh^2}\right)u_2^n + 2\ h\ \alpha\frac{kD}{dh^2} \end{aligned}$$

Nodes $j = 2 \cdots N-1$ remain the same as before. In other words

$$u_j^{n+1} = \frac{kD}{dh^2}u_{j-1}^n + u_j^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_{j+1}^n + \frac{k}{d}g_j^n$$

Let $r_1 = \left(1 - 2\frac{kD}{dh^2} - \frac{ak}{d}\right), r_2 = \frac{kD}{dh^2}, r_3 = \frac{k}{d}$, Hence the system becomes

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix} = \begin{bmatrix} r_1 & 2r_2 & 0 & 0 & 0 & 0 & 0 \\ r_2 & r_1 & r_2 & 0 & 0 & 0 & 0 \\ 0 & r_2 & r_1 & r_2 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & 0 \\ 0 & 0 & 0 & r_2 & r_1 & r_2 & 0 \\ 0 & 0 & 0 & 0 & r_2 & r_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} 2\ h\ \alpha^n r_2 \\ r_3 g_2^n \\ r_3 g_3^n \\ r_3 g_4^n \\ \vdots \\ r_3 g_{N-1}^n + r_2\beta^n \\ \beta^{n+1} \end{bmatrix}$$

3.2.1.3.6 Derivation of forward Euler for Neumann on right and Dirichlet on left

Using this numbering

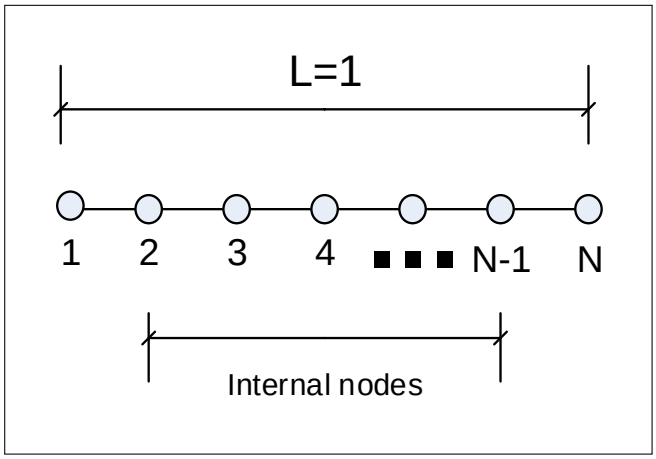


Figure 3.11: Grid format

Assume that $u_0 = \alpha$ at node 1 and $u_t = \beta$ at node N (these are the Dirichlet and Neumann boundary conditions).

The unknowns in this case are nodes $2 \cdots N$, since u_1 is known from Dirichlet boundary conditions. Add a ghost node $N+1$ to the right of node N, and approximating $\beta(t)$ gives

$$\beta = \frac{u_{N-1} - u_{N+1}}{2h}$$

hence

$$u_{N+1} = 2h\beta + u_{N-1} \tag{3}$$

But the PDE for node N is

$$u_N^{n+1} = \frac{kD}{dh^2}u_{N-1}^n + u_N^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_{N+1}^n \tag{4}$$

Substitute (3) into (4) gives

$$\begin{aligned} u_N^{n+1} &= \frac{kD}{dh^2}u_{N-1}^n + u_N^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}\left(2h\beta + u_{N-1}\right) \\ &= 2\frac{kD}{dh^2}u_{N-1}^n + u_N^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + 2h\beta\frac{kD}{dh^2} \end{aligned}$$

Nodes $j = 2 \cdots N-1$ remain the same as before. In other words

$$u_j^{n+1} = \frac{kD}{dh^2}u_{j-1}^n + u_j^n\left(1 - 2\frac{k\ D}{d\ h^2} - \frac{ak}{d}\right) + \frac{kD}{dh^2}u_{j+1}^n + \frac{k}{d}g_j^n$$

Let $r_1 = \left(1 - 2\frac{kD}{dh^2} - \frac{ak}{d}\right), r_2 = \frac{kD}{dh^2},, r_3 = \frac{k}{d}$, Hence the system becomes

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ u_4^{n+1} \\ \vdots \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_1 & r_2 & 0 & 0 & 0 & 0 \\ 0 & r_2 & r_1 & r_2 & 0 & 0 & 0 \\ 0 & 0 & r_2 & r_1 & r_2 & 0 & 0 \\ 0 & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & r_2 & r_1 & r_2 \\ 0 & 0 & 0 & 0 & 0 & 2r_2 & r_1 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ u_4^n \\ \vdots \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} \alpha^{n+1} \\ r_2\alpha^n + r_3 g_2^n \\ r_3 g_2^n \\ r_3 g_3^n \\ r_3 g_4^n \\ \vdots \\ r_3 g_{N-1}^n \\ 2h\beta^n\ r_2 \end{bmatrix}$$

3.2.1.3.7 Derivation of C-N for periodic boundary conditions From part(a) above,

$$u_j^{n+1}\left(1 + \frac{kD}{dh^2} + \frac{a\Delta t}{2d}\right) - \frac{kD}{2dh^2}u_{j-1}^{n+1} - \frac{kD}{2dh^2}u_{j+1}^{n+1} = u_j^n\left(1 - \frac{kD}{dh^2} - \frac{ak}{2d}\right) + u_{j-1}^n\frac{kD}{2dh^2} + u_{j+1}^n\frac{kD}{2dh^2} + \frac{k}{2d}\left(g_j^n + g_j^{n+1}\right)$$

Periodic boundary conditions implies $u(0,t) = u(1,t)$, Hence there is an extra one unknown (in addition to the internal nodes). Either $u(0,t)$ or $u(1,t)$ can be selected since they have

the same value. When selecting the right end node, then u_{N+1} becomes an unknown to be added to the internal nodes. Using the following diagram

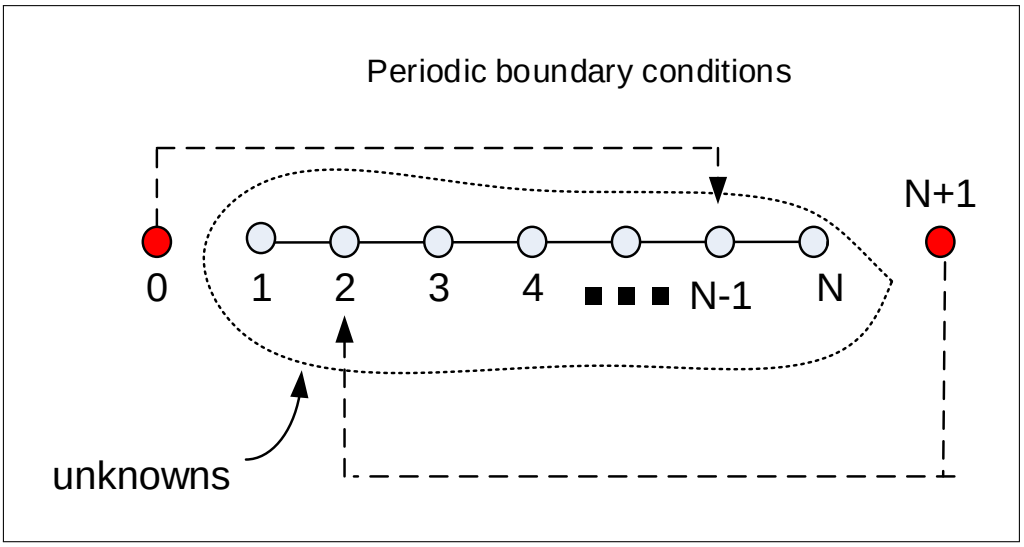


Figure 3.12: Grid format

Then for node 2

$$u_2^{n+1}\left(1+\frac{kD}{dh^2}+\frac{a\Delta t}{2d}\right)-\frac{kD}{2dh^2}u_1^{n+1}-\frac{kD}{2dh^2}u_3^{n+1}=u_2^n\left(1-\frac{kD}{dh^2}-\frac{ak}{2d}\right)+u_1^n\frac{kD}{2dh^2}+u_3^n\frac{kD}{2dh^2}+\frac{k}{2d}\left(g_2^n+g_2^{n+1}\right)$$

$$u_2^{n+1}\left(1+\frac{kD}{dh^2}+\frac{a\Delta t}{2d}\right)-\frac{kD}{2dh^2}u_N^{n+1}-\frac{kD}{2dh^2}u_3^{n+1}=u_2^n\left(1-\frac{kD}{dh^2}-\frac{ak}{2d}\right)+u_N^n\frac{kD}{2dh^2}+u_3^n\frac{kD}{2dh^2}+\frac{k}{2d}\left(g_2^n+g_2^{n+1}\right)$$

And for node N

$$u_N^{n+1}\left(1+\frac{kD}{dh^2}+\frac{a\Delta t}{2d}\right)-\frac{kD}{2dh^2}u_{N-1}^{n+1}-\frac{kD}{2dh^2}u_{N+1}^{n+1}=u_N^n\left(1-\frac{kD}{dh^2}-\frac{ak}{2d}\right)+u_{N-1}^n\frac{kD}{2dh^2}+u_{N+1}^n\frac{kD}{2dh^2}+\frac{k}{2d}\left(g_N^n+g_N^{n+1}\right)$$

$$u_N^{n+1}\left(1+\frac{kD}{dh^2}+\frac{a\Delta t}{2d}\right)-\frac{kD}{2dh^2}u_{N-1}^{n+1}-\frac{kD}{2dh^2}u_2^{n+1}=u_N^n\left(1-\frac{kD}{dh^2}-\frac{ak}{2d}\right)+u_{N-1}^n\frac{kD}{2dh^2}+u_2^n\frac{kD}{2dh^2}+\frac{k}{2d}\left(g_N^n+g_N^{n+1}\right)$$

Let

$$\begin{aligned}
r_1 &= \left(1+\frac{kD}{dh^2}+\frac{ak}{2d}\right)\\
r_2 &= \frac{kD}{2dh^2}\\
r_3 &= \left(1-\frac{kD}{dh^2}-\frac{ak}{2d}\right)\\
r_4 &= \frac{k}{2d}\\
g^n+g^{n+1} &\equiv \tilde{g}
\end{aligned}$$

Converting to matrix form gives

$$\overbrace{\begin{bmatrix} r_1 & -r_2 & 0 & 0 & -r_2 \\ -r_2 & r_1 & -r_2 & 0 & 0 \\ 0 & -r_2 & r_1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ -r_2 & 0 & 0 & -r_2 & r_1 \end{bmatrix}}^{\mathbf{A}} \overbrace{\begin{bmatrix} u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix}}^{\mathbf{x}} = \overbrace{\begin{bmatrix} r_3 & r_2 & 0 & 0 & r_2 \\ r_2 & r_3 & r_2 & 0 & 0 \\ 0 & r_2 & r_3 & r_2 & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ r_2 & 0 & 0 & r_2 & r_3 \end{bmatrix} \begin{bmatrix} u_2^n \\ u_3^n \\ u_4^n \\ \vdots \\ u_N^n \end{bmatrix} + \begin{bmatrix} r_4 \tilde{g}_2 \\ r_4 \tilde{g}_3 \\ r_4 \tilde{g}_4 \\ \vdots \\ r_4 \tilde{g}_N \end{bmatrix}}^{\mathbf{b}} \tag{3}$$

$$u_1^{n+1} = u_N^{n+1}$$

3.2.1.3.8 Derivation of C-N for Neumann boundary conditions both ends Using this numbering

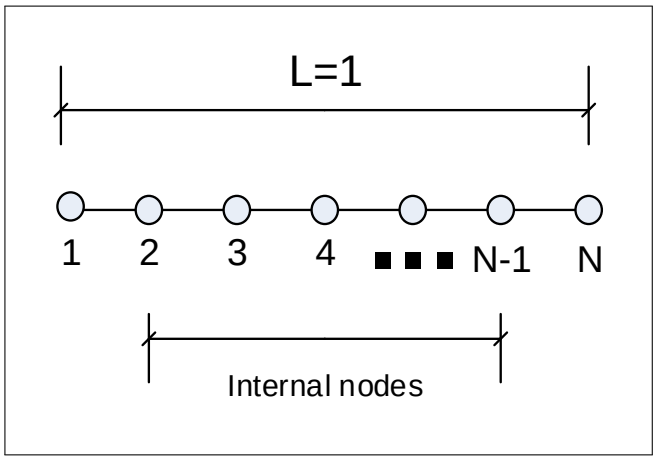


Figure 3.13: Grid format

Assuming that $u_t = \alpha$ at node 1 and $u_t = \beta$ at node N (these are the Neumann boundary conditions). Add a ghost node 0 to the left of node 1, and approximating α gives

$$\alpha = \frac{u_0 - u_2}{2h}$$

hence

$$u_0 = 2h\alpha + u_2 \tag{1}$$

But the PDE for node 1 is

$$r_1 u_1^{n+1} - r_2 u_0^{n+1} - r_2 u_2^{n+1} = r_3 u_1^n + r_2 u_0^n + r_2 u_2^n \tag{2}$$

substitute (1) into (2)

$$\begin{aligned} r_1 u_1^{n+1} - r_2\left(2h\alpha^{n+1} + u_2^{n+1}\right) - r_2 u_2^{n+1} &= r_3 u_1^n + r_2\left(2h\alpha^n + u_2^n\right) + r_2 u_2^n \\ r_1 u_1^{n+1} - 2r_2 u_2^{n+1} &= r_3 u_1^n + 2r_2 u_2^n + 2r_2 h\alpha^n + 2r_2 h\alpha^{n+1} \end{aligned}$$

Similarly for the right end. Add a ghost node $N+1$ to the right of node N, and approximating $\beta(t)$ gives

$$\beta = \frac{u_{N-1} - u_{N+1}}{2h}$$

hence

$$u_{N+1} = 2h\beta + u_{N-1} \tag{3}$$

But the PDE for node N is

$$r_1 u_N^{n+1} - r_2 u_{N-1}^{n+1} - r_2 u_{N+1}^{n+1} = r_3 u_N^n + r_2 u_{N-1}^n + r_2 u_{N+1}^n \tag{4}$$

substitute (3) into (4)

$$\begin{aligned} r_1 u_N^{n+1} - r_2 u_{N-1}^{n+1} - r_2\left(2h\beta^{n+1} + u_{N-1}^{n+1}\right) &= r_3 u_N^n + r_2 u_{N-1}^n + r_2\left(2h\beta + u_{N-1}^n\right) \\ r_1 u_N^{n+1} - 2r_2 u_{N-1}^{n+1} &= r_3 u_N^n + 2r_2 u_{N-1}^n + 2r_2 h\beta + 2r_2 h\beta^{n+1} \end{aligned}$$

Nodes $j = 2 \cdots N-1$ remain the same as before. In other words

$$u_j^{n+1}\left(1 + \frac{kD}{dh^2} + \frac{ak}{2d}\right) - \frac{kD}{2dh^2}u_{j-1}^{n+1} - \frac{kD}{2dh^2}u_{j+1}^{n+1} = u_j^n\left(1 - \frac{kD}{dh^2} - \frac{ak}{2d}\right) + u_{j-1}^n\frac{kD}{2dh^2} + u_{j+1}^n\frac{kD}{2dh^2} + \frac{k}{2d}\left(g_j^n + g_j^{n+1}\right)$$

Let

$$\begin{aligned} r_1 &= \left(1 + \frac{kD}{dh^2} + \frac{ak}{2d}\right) \\ r_2 &= \frac{kD}{2dh^2} \\ r_3 &= \left(1 - \frac{kD}{dh^2} - \frac{ak}{2d}\right) \\ r_4 &= \frac{k}{2d} \end{aligned}$$

gives

$$r_1 u_j^{n+1} - r_2 u_{j-1}^{n+1} - r_2 u_{j+1}^{n+1} = r_3 u_j^n + r_2 u_{j-1}^n + r_2 u_{j+1}^n + r_4\left(g_j^n + g_j^{n+1}\right)$$

Then the above becomes

$$\overbrace{\begin{bmatrix} r_1 & -2r_2 & 0 & 0 & 0 & 0 \\ -r_2 & r_1 & -r_2 & 0 & 0 & 0 \\ 0 & -r_2 & r_1 & -r_2 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & -r_2 & r_1 & -r_2 \\ 0 & 0 & 0 & 0 & -2r_2 & r_1 \end{bmatrix}}^{\mathbf{A}} \overbrace{\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix}}^{\mathbf{x}} = \overbrace{\begin{bmatrix} r_3 & 2r_2 & 0 & 0 & 0 & 0 \\ r_2 & r_3 & r_2 & 0 & 0 & 0 \\ 0 & r_2 & r_3 & r_2 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & r_2 & r_3 & r_2 \\ 0 & 0 & 0 & 0 & 2r_2 & r_3 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} 2r_2 h\alpha^n + 2r_2 h\alpha^{n+1} \\ r_4\left(g_2^n + g_2^{n+1}\right) \\ r_4\left(g_3^n + g_3^{n+1}\right) \\ \vdots \\ r_4\left(g_{N-1}^n + g_{N-1}^{n+1}\right) \\ 2r_2 h\beta + 2r_2 h\beta^{n+1} \end{bmatrix}}^{\mathbf{b}}$$

3.2.1.3.9 Derivation of C-N for Neumann on left and Dirichlet on right Using this numbering

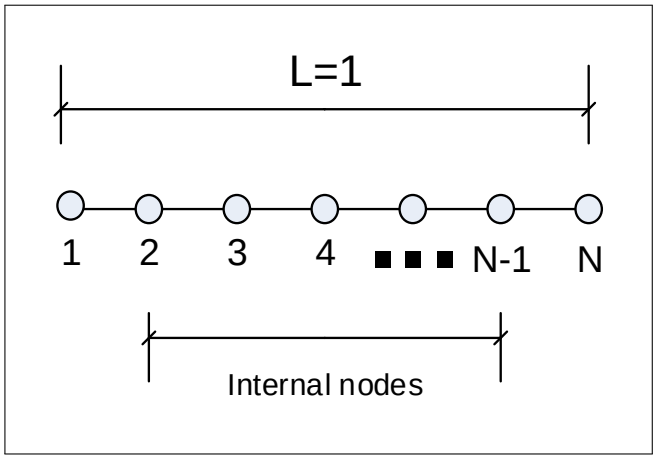


Figure 3.14: Grid format

Assume that $u_t = \alpha$ at node 1 and $u_N = \beta$ at node N (these are the Neumann and Dirichlet boundary conditions). Add a ghost node 0 to the left of node 1, and approximating α gives

$$\alpha = \frac{u_0 - u_2}{2h}$$

hence

$$u_0 = 2h\alpha + u_2 \tag{1}$$

But the PDE for node 1 is

$$r_1 u_1^{n+1} - r_2 u_0^{n+1} - r_2 u_2^{n+1} = r_3 u_1^n + r_2 u_0^n + r_2 u_2^n \tag{2}$$

substitute (1) into (2)

$$\begin{aligned} r_1 u_1^{n+1} - r_2\left(2h\alpha^{n+1} + u_2^{n+1}\right) - r_2 u_2^{n+1} &= r_3 u_1^n + r_2\left(2h\alpha^n + u_2^n\right) + r_2 u_2^n \\ r_1 u_1^{n+1} - 2r_2 u_2^{n+1} &= r_3 u_1^n + 2r_2 u_2^n + 2r_2 h\alpha^n + 2r_2 h\alpha^{n+1} \end{aligned}$$

Nodes $j = 2 \cdots N-1$ remain the same as before. In other words

$$u_j^{n+1}\left(1 + \frac{kD}{dh^2} + \frac{ak}{2d}\right) - \frac{kD}{2dh^2} u_{j-1}^{n+1} - \frac{kD}{2dh^2} u_{j+1}^{n+1} = u_j^n\left(1 - \frac{kD}{dh^2} - \frac{ak}{2d}\right) + u_{j-1}^n \frac{kD}{2dh^2} + u_{j+1}^n \frac{kD}{2dh^2} + \frac{k}{2d}\left(g_j^n + g_j^{n+1}\right)$$

Let

$$\begin{aligned} r_1 &= \left(1 + \frac{kD}{dh^2} + \frac{ak}{2d}\right) \\ r_2 &= \frac{kD}{2dh^2} \\ r_3 &= \left(1 - \frac{kD}{dh^2} - \frac{ak}{2d}\right) \\ r_4 &= \frac{k}{2d} \end{aligned}$$

gives

$$r_1 u_j^{n+1} - r_2 u_{j-1}^{n+1} - r_2 u_{j+1}^{n+1} = r_3 u_j^n + r_2 u_{j-1}^n + r_2 u_{j+1}^n + r_4\left(g_j^n + g_j^{n+1}\right)$$

Then the system becomes

$$\overbrace{\begin{bmatrix} r_1 & -2r_2 & 0 & 0 & 0 & 0 & 0 \\ -r_2 & r_1 & -r_2 & 0 & 0 & 0 & 0 \\ 0 & -r_2 & r_1 & -r_2 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & -r_2 & r_1 & -r_2 & 0 \\ 0 & 0 & 0 & 0 & -r_2 & r_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}^{\mathbf{A}} \overbrace{\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{N-2}^{n+1} \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix}}^{\mathbf{x}} = \overbrace{\begin{bmatrix} r_3 & 2r_2 & 0 & 0 & 0 & 0 & 0 \\ r_2 & r_3 & r_2 & 0 & 0 & 0 & 0 \\ 0 & r_2 & r_3 & r_2 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & r_2 & r_3 & r_2 & 0 \\ 0 & 0 & 0 & 0 & 2r_2 & r_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_{N-2}^n \\ u_{N-1}^n \\ u_N^n \end{bmatrix} + \begin{bmatrix} 2r_2 h\alpha^n + 2r_2 h\alpha^{n+1} \\ r_4\left(g_2^n + g_2^{n+1}\right) \\ r_4\left(g_3^n + g_3^{n+1}\right) \\ \vdots \\ r_4\left(g_{N-2}^n + g_{N-2}^{n+1}\right) \\ r_4\left(g_{N-1}^n + g_{N-1}^{n+1}\right) + r_2\beta^n \\ \beta^{n+1} \end{bmatrix}}^{\mathbf{b}}$$

3.2.1.3.10 Derivation of C-N for Neumann on right and Dirichlet on left Using this numbering

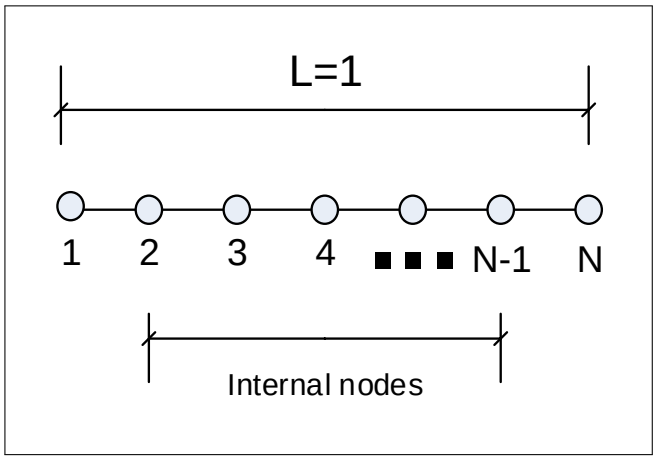


Figure 3.15: Grid format

Assume that $u_1 = \alpha$ at node 1 and $u_t = \beta$ at node N (these are the Dirichlet and Neumann boundary conditions). Add a ghost node $N+1$ to the right of node N, and approximating β gives

$$\beta = \frac{u_{N-1} - u_{N+1}}{2h}$$

hence

$$u_{N+1} = 2h\beta + u_{N-1} \tag{1}$$

But the PDE for node N is

$$r_1 u_N^{n+1} - r_2 u_{N-1}^{n+1} - r_2 u_{N+1}^{n+1} = r_3 u_N^n + r_2 u_{N-1}^n + r_2 u_{N+1}^n \tag{2}$$

substitute (1) into (2)

$$\begin{aligned} r_1 u_N^{n+1} - r_2 u_{N-1}^{n+1} - r_2\left(2h\beta^{n+1} + u_{N-1}^{n+1}\right) &= r_3 u_N^n + r_2 u_{N-1}^n + r_2\left(2h\beta^n + u_{N-1}^n\right) \\ r_1 u_N^{n+1} - 2r_2 u_{N-1}^{n+1} &= r_3 u_N^n + 2r_2 u_{N-1}^n + 2r_2 h\beta^n + 2r_2 h\beta^{n+1} \end{aligned}$$

Nodes $j = 2 \cdots N-1$ remain the same as before. In other words

$$u_j^{n+1}\left(1+\frac{kD}{dh^2}+\frac{ak}{2d}\right)-\frac{kD}{2dh^2}u_{j-1}^{n+1}-\frac{kD}{2dh^2}u_{j+1}^{n+1}=u_j^n\left(1-\frac{kD}{dh^2}-\frac{ak}{2d}\right)+u_{j-1}^n\frac{kD}{2dh^2}+u_{j+1}^n\frac{kD}{2dh^2}+\frac{k}{2d}\left(g_j^n+g_j^{n+1}\right)$$

Let

$$\begin{aligned}r_1&=\left(1+\frac{kD}{dh^2}+\frac{ak}{2d}\right)\\ r_2&=\frac{kD}{2dh^2}\\ r_3&=\left(1-\frac{kD}{dh^2}-\frac{ak}{2d}\right)\\ r_4&=\frac{k}{2d}\end{aligned}$$

gives

$$r_1u_j^{n+1}-r_2u_{j-1}^{n+1}-r_2u_{j+1}^{n+1}=r_3u_j^n+r_2u_{j-1}^n+r_2u_{j+1}^n+r_4\left(g_j^n+g_j^{n+1}\right)$$

Then the system becomes

$$\overbrace{\begin{bmatrix}1&0&0&0&0&0&0\\0&r_1&-r_2&0&0&0&0\\0&-r_2&r_1&-r_2&0&0&0\\0&0&-r_2&r_1&-r_2&0&0\\\vdots&\vdots&\vdots&\vdots&\ddots&\vdots&0\\0&0&0&0&-r_2&r_1&-r_2\\0&0&0&0&0&-2r_2&r_1\end{bmatrix}}^{\mathbf{A}}\overbrace{\begin{bmatrix}u_1^{n+1}\\u_2^{n+1}\\u_3^{n+1}\\u_4^{n+1}\\\vdots\\u_{N-1}^{n+1}\\u_N^{n+1}\end{bmatrix}}^{\mathbf{x}}=\overbrace{\begin{bmatrix}0&0&0&0&0&0&0\\0&r_3&r_2&0&0&0&0\\0&r_2&r_3&r_2&0&0&0\\0&0&r_2&r_3&r_2&0&0\\\vdots&\vdots&\vdots&\vdots&\ddots&\vdots&0\\0&0&0&0&r_2&r_3&r_2\\0&0&0&0&0&2r_2&r_3\end{bmatrix}\begin{bmatrix}u_1^n\\u_2^n\\u_3^n\\\vdots\\u_{N-2}^n\\u_{N-1}^n\end{bmatrix}+\begin{bmatrix}\alpha^{n+1}\\r_4\left(g_2^n+g_2^{n+1}\right)+r_2\alpha^n\\r_4\left(g_3^n+g_3^{n+1}\right)\\r_4\left(g_4^n+g_4^{n+1}\right)\\\vdots\\r_4\left(g_{N-2}^n+g_{N-2}^{n+1}\right)\\2r_2h\beta^n+2r_2h\beta^{n+1}\end{bmatrix}}^{\mathbf{b}}$$

3.2.2 Problem 2

2.

$$u_t = u_{xx}, \quad 0 < x < 1$$

$$u(0,t) = 1, \quad u(1,t) = 0$$

$$u(x,0) = \begin{cases} 1 & \text{if } x < 0.5 \\ 0 & \text{if } x \geq 0.5 \end{cases}$$

(a) Use Crank-Nicolson with grid spacing $h = 0.02$ and time step 0.1 to solve the problem up to time $t = 1$. Comment on your results. What is wrong with this solution?

(b) Give a mathematical argument to explain the unphysical behavior you observed in the numerical solution.

(c) Experiment with smaller time steps. How small does the time step need to be to get reasonable results?

(d) What happens to the numerical solution as $\Delta t \to 0$ with the ratio $\Delta t/h$ fixed? Explain. Would this same behavior occur using backward Euler in place of Crank-Nicolson? Explain.

Figure 3.16: Problem statement

3.2.2.1 Part(a)

The C-N scheme was programmed in Matlab and then run on the above problem. The following shows the result

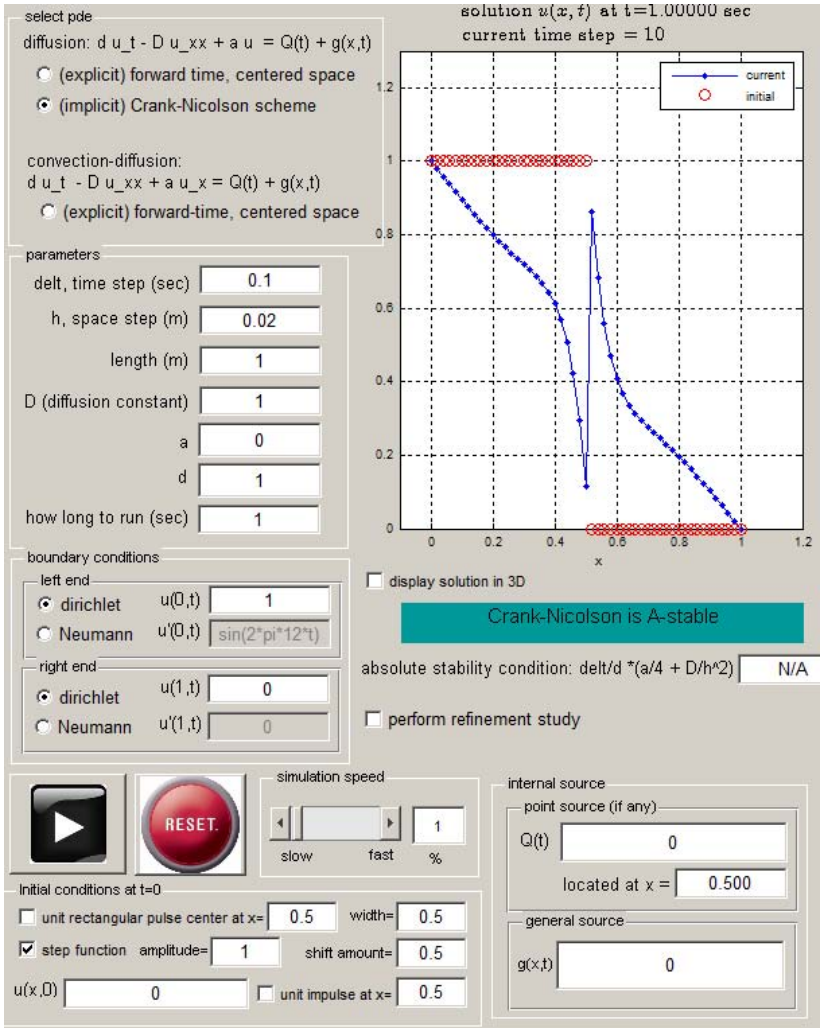


Figure 3.17: C-N scheme solution result

In the above plot, the red line represents initial conditions (the step function shifted to the right by 0.5) and the blue line represents the final numerical solution at time $t = 1$ second. The following plot is a closer look at the grid near $x = \frac{1}{2}$ showing the initial conditions

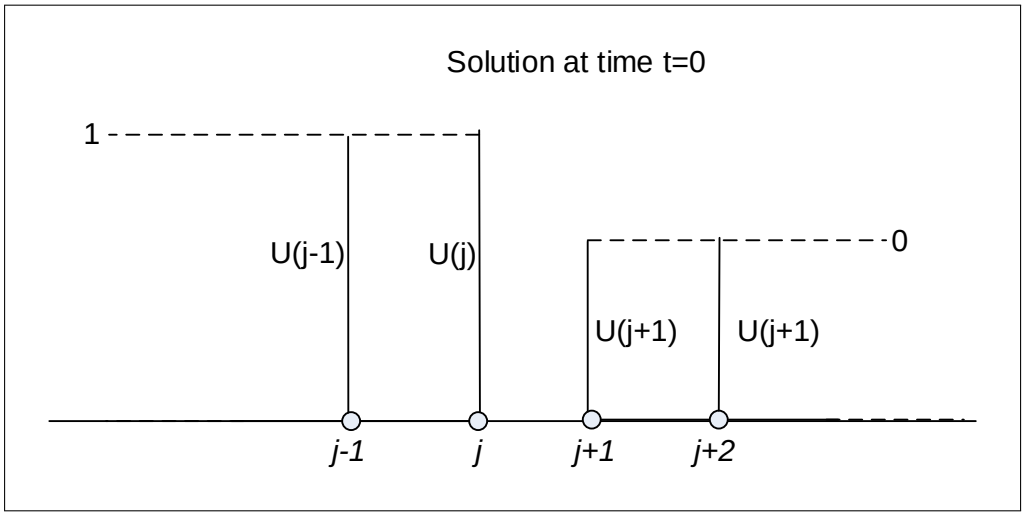


Figure 3.18: the grid near $x = \frac{1}{2}$ showing the initial conditions

It is clear the numerical solution is not accurate as it does not match what is expected to occur physically which is for initial data to diffuse. The initial data contained high spatial frequency that should have been smoothed out rapidly. The final numerical solution is not smooth and contain high spatial frequency components which should have been attenuated by the time the run is completed.

The exact solution in the Fourier space is $\hat{u}(\xi,t) = \hat{u}(\xi,0)e^{-D\xi^2 t}$, where $\hat{u}(\xi,0)$ is the spectrum or Fourier coefficients of the initial condition $u(x,0)$. This shows that modes with large spatial frequency (large wave number ξ) will attenuate the fastest due to the negative exponential decay effect. But this was not observed in the above numerical solution.

C-N is a stable scheme (A-stable), but can be inaccurate if the time step used is large relative to the space step or if initial conditions contain large spatial frequency components. In C-N, the time step needs to be about the same order of value as the space step for the scheme to give accurate numerical results. (This is because in C-N, the order of accuracy of space and time are the same, as was found in problem 1).

Therefore, it appears that C-N scheme does not handle discontinuities in initial conditions well as this result shows.

In the next part, the amplification factor for C-N is determined, and a mathematical explanation for the above result is given.

3.2.2.2 Part(b)

The C-N scheme for $u_t = Du_{xx}$ is given by

$$-ru_{j-1}^{n+1} + u_j^{n+1}(1+2r) - ru_{j+1}^{n+1} = ru_{j-1}^{n} + u_j^{n}(1-2r) + ru_{j+1}^{n} \tag{1}$$

Where $r = \frac{\Delta t D}{2h^2}$. Von Neumann analysis is used to determine the magnification factor[2]. Assume $u_j^n = e^{i\xi x_j}$, and $u_j^{n+1} = g(\xi)e^{i\xi x_j}$ then (1) becomes[3]

$$\begin{aligned} -r\left(g\ e^{i\xi x_j}e^{-i\xi h}\right) + g\ e^{i\xi x_j}(1+2r) - r\left(g\ e^{i\xi x_j}e^{i\xi h}\right) &= r\left(e^{i\xi x_j}e^{-i\xi h}\right) + e^{i\xi x_j}(1-2r) + re^{i\xi x_j}e^{i\xi h} \\ g(-2r\cos(\xi h)+1+2r) &= 2r\cos(\xi h)+1-2r \\ g(\xi) &= \frac{1+2r\cos(\xi h)-2r}{1-2r\cos(\xi h)+2r} \end{aligned}$$

Hence, the magnification factor is

$$g(\xi) = \frac{1+\frac{D\Delta t}{h^2}(\cos(\xi h)-1)}{1-\frac{D\Delta t}{h^2}(\cos(\xi h)-1)}$$

Let ξ be written as ξ_p in the above in order to examine g in terms of specific wave number ξ_p (which has units of radians per unit length). The above becomes

$$g\left(\xi_p\right) = \frac{1+\frac{D\Delta t}{h^2}\left(\cos\left(\xi_p h\right)-1\right)}{1-\frac{D\Delta t}{h^2}\left(\cos\left(\xi_p h\right)-1\right)} \tag{2}$$

But $\xi_p = p\pi$ and $p = 1\cdots N$, with $h = \frac{1}{N+1}$where the line was discretized using the standard grid convention

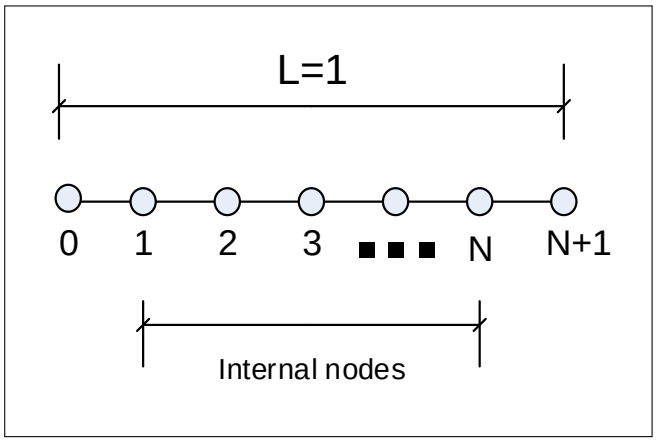


Figure 3.19: Grid format

Therefore $\cos\left(\xi_p h\right) = \cos\left(p\pi h\right) = \cos\left(\frac{p\pi}{N+1}\right)$. The largest frequency occurs when $p = N$, because then h is smallest, and the smallest frequency occurs when $p = 1$. Hence, there are a total of N Fourier modes when representing initial data as Fourier series. Now the magnification factor in (2) becomes

[2]The magnification factor is the term $g(\xi)$ in the expression relating $\hat{u}^{n+1}$ to $\hat{u}^n$ in the expression $\hat{u}^{n+1} = g(\xi)\,\hat{u}^n$

[3]It is possible to derive the amplification factor using direct application of fourier transform, but the procedure is longer. The final result will be the same. The appendix of this problem contain this derivation.

$$g\left(\xi_p\right) = \frac{1 + \frac{D\Delta t}{h^2}\left(\cos\left(p\pi h\right) - 1\right)}{1 - \frac{D\Delta t}{h^2}\left(\cos\left(p\pi h\right) - 1\right)} \tag{3}$$

But since $\left|\cos\left(p\pi h\right)\right| \leq 1$, then $g\left(\xi_p\right)$ is less than 1 in magnitude for any p, implying that C-N is stable. To determine the magnitude of $g\left(\xi_p\right)$ when the mode has the largest frequency, let $p\pi h = \frac{N}{N+1}\pi \approx \pi$ in (3), resulting in

$$g(\xi_N) = \frac{1 - 2\frac{D\Delta t}{h^2}}{1 + 2\frac{D\Delta t}{h^2}} \tag{4}$$

When the time step $\Delta t \gg h$, then $\frac{D\Delta t}{h^2} \gg 1$, and in the limit $\left|g(\xi_N)\right| \to 1$. This shows that large frequency modes will decay very slowly because $g\left(\xi_p\right)$ is now close to 1. No attenuation will occur between each application of the update matrix or between each time step.

The above explains the result seen in part (a). Large frequency components did not decay fast as was expected, because the time step used was much larger than the space step. The problem asked us to use $\Delta t = 0.1$ and $h = 0.01$, which gives $\frac{D\Delta t}{h^2} = \frac{0.1}{0.01^2} = 1000$ and hence $g(\xi_N) \to \left|\frac{1-2000}{1+2000}\right| \to 0.9999$, and since this is almost one, then large frequency modes did not attenuate with each time step. The amplification factor needs to be small for attenuation to occur fast.

The following is a plot of $g\left(\xi_p\right)$ showing how the amplification factor changes as function of p for the case of $\Delta t = 0.1$ and $h = 0.01$, and $D = 1$. It shows what was found above, that at large frequency where p is close to N will have a correspondingly large

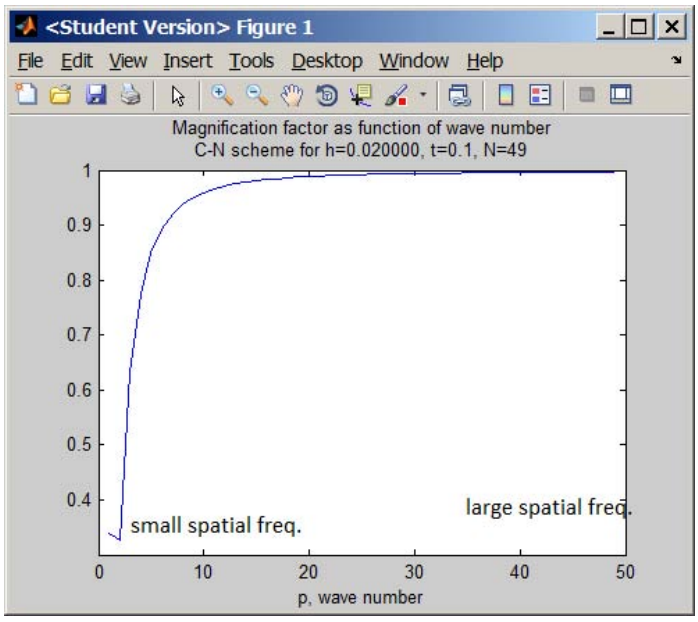


Figure 3.20: corrected plot of amplification factor

To determine what value of $\frac{D\Delta t}{h^2}$ is required to make the large frequency mode decay right away, let $\frac{D\Delta t}{h^2} = \frac{1}{2}$ in (4), this gives $g(\xi_N) = 0$, which implies that large frequency mode will be knocked out right away. Here is a plot of $g(\xi_N) = \frac{1-2\frac{D\Delta t}{h^2}}{1+2\frac{D\Delta t}{h^2}}$ as a function of $\frac{D\Delta t}{h^2}$ showing that when $\frac{D\Delta t}{h^2} = 0$ then the magnification factor is minimum. This is only for mode $p = N$.

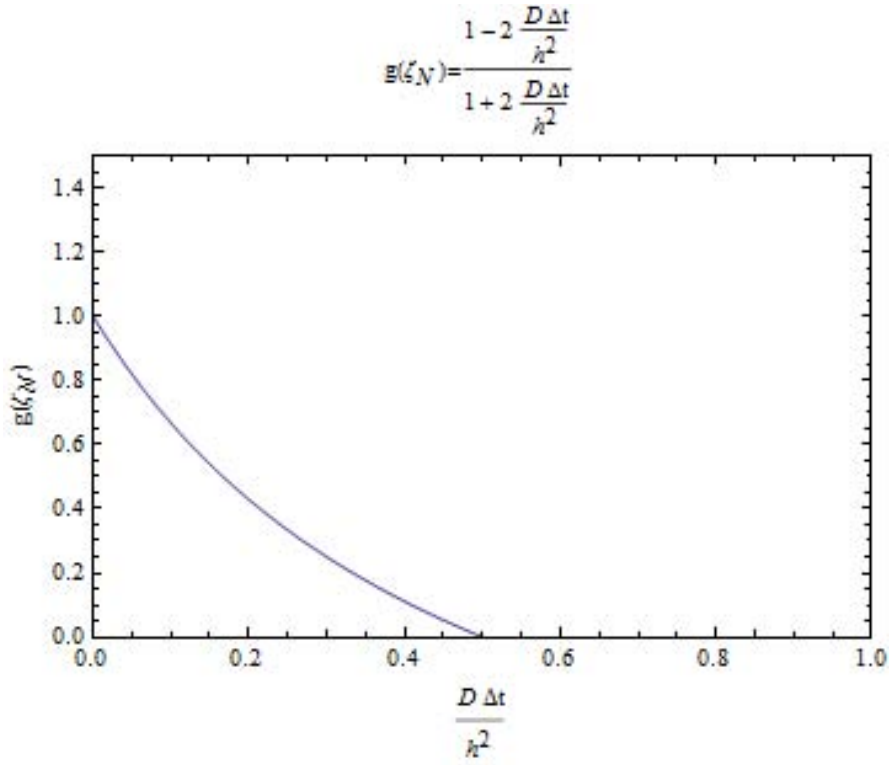


Figure 3.21: for mode $p = N$

Conclusion If initial data contained large difference in value over very short distances (in other words, large spatial frequencies) such as given in this problem, producing discontinuity in data and its space derivative, and when the time step is large compared to the space step, then the numerical solution produced by C-N will not be accurate since large frequency modes will not attenuate.

To compensate for large frequency present in initial data, the ratio $\frac{D\Delta t}{h^2}$ needs to be made close to 0.5 as possible. It might be better not to use C-N at all in such case and look for a scheme which does not have this problem.

notice that condition that $\frac{D\Delta t}{h^2} = \frac{1}{2}$ found above, is the same value for the upper limit for the absolute stability condition for the forward Euler discretization scheme for the 1D diffusion problem.

3.2.2.3 Part(c)

The time step was reduced and the program was run for 1 second. When the time step was reduced all the way to $\Delta t = 0.01$ then the final solution appeared smooth every where and in particular at $x = 0.5$. The following diagram illustrates this.

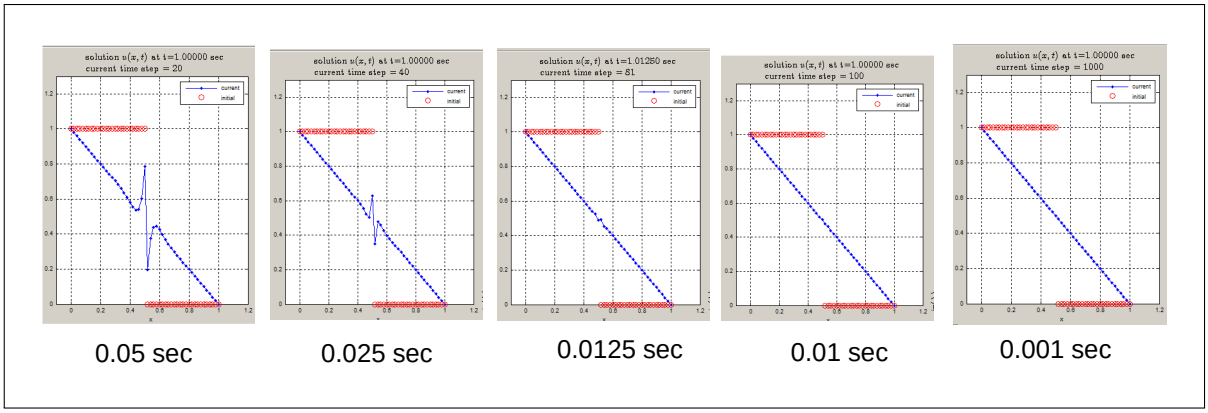


Figure 3.22: Final solution at $x = 0.5$

When using time step of 0.01 sec, 100 steps are used. From the last part it was found that $g(\xi_N) = \frac{1-2\frac{D\Delta t}{h^2}}{1+2\frac{D\Delta t}{h^2}}$, hence $|g| = \left|\frac{1-2\frac{0.01}{0.02^2}}{1+2\frac{0.01}{0.02^2}}\right| = 0.96078$, and therefore $|g|^{100} \rightarrow 0.96078^{100} \approx 0$ showing that by the 100^{th} iteration, the large mode frequency have completely smoothed out as verified by the above plots.

In the plot below, the magnification factor $|g(\xi)|$ is shown for $t = 0.01$ and $h = 0.02$ showing that at $p = 50$, $g(\xi) = 0.96078$. Compare this value with the one used in part(b) which was 0.999.

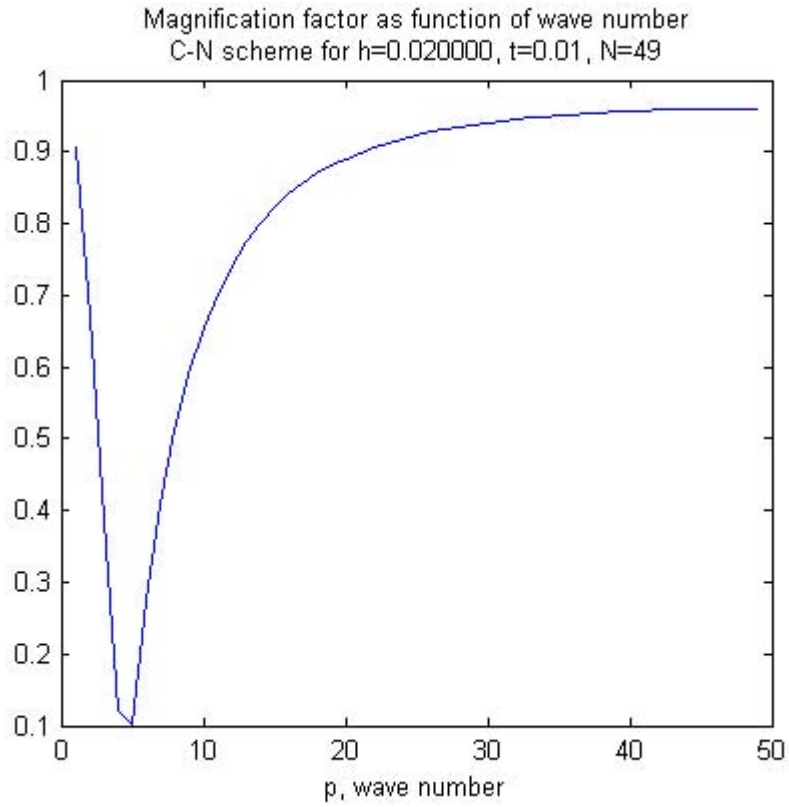


Figure 3.23: Comparing with the one used in part(b) which was 0.999.

To obtain smooth solution immediately after one time step, the required time step to accomplish this, can be determined from the condition for optimal amplification factor found in the last part which is given by $\frac{D\Delta t}{h^2} = \frac{1}{2}$. From this relation, and when $h = 0.02$ and $D = 1$, the time step will be $\Delta t = 0.0002$. This value of time step produces a smooth solution immediately (one step). This was confirmed, and here is the numerical solution, after only one time step, using $\Delta t = 0.0002, h = 0.02$ and $D = 1$

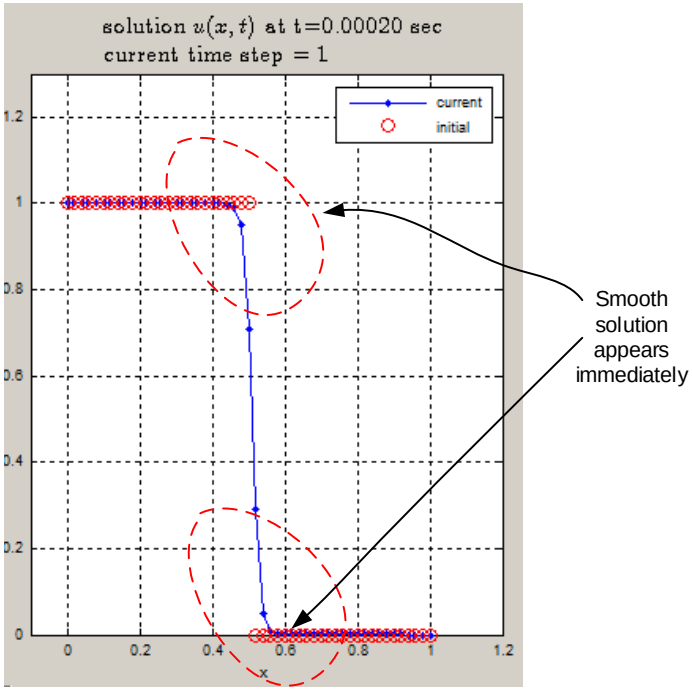


Figure 3.24: numerical solution, after only one time step, using $\Delta t = 0.0002, h = 0.02$ and $D = 1$

It is also possible to determine which Δt achieves a given specific attenuation of the high mode. Suppose it is required to attenuate the high mode to 0.001 of its initial amplitude at the end of 1 second run. Therefore, this means that

$$\left|g(\xi_N)\right|^{\frac{1}{\Delta t}} = 0.001$$

$$\left|\frac{1-2\frac{D\Delta t}{h^2}}{1+2\frac{D\Delta t}{h^2}}\right|^{\frac{1}{\Delta t}} = 0.001$$

Taking logs, and using $h = 0.02$ and $D = 1$ results in

$$\frac{1}{\Delta t}\log\left(\frac{1-5000\Delta t}{1+5000\Delta t}\right) = -3$$

$$\log(1-5000\Delta t) - \log(1+5000\Delta t) = -3\Delta t$$

The above is not a linear equation, but can be numerically solved for the root Δt. For the above example, Δt came out to be 0.00761 seconds. This means that when using $\Delta t = 0.00761$ sec, $h = 0.02$, and $D = 1$, then the largest frequency harmonic will have its amplitude attenuated to 0.01% of its original value after 1 second run.

3.2.2.4 Part(d)

Making the time step smaller and smaller, while keeping the ratio $\frac{\Delta t}{h}$ fixed, produces the following result (all runs are for one second). In this example, the ratio was kept at 5. The following sequence of ratios are used $\left\{\frac{0.1}{0.02}, \frac{0.01}{0.002}, \frac{0.001}{0.0002}, \frac{0.0001}{0.00002}\right\}$ to generate the following solution after one second run

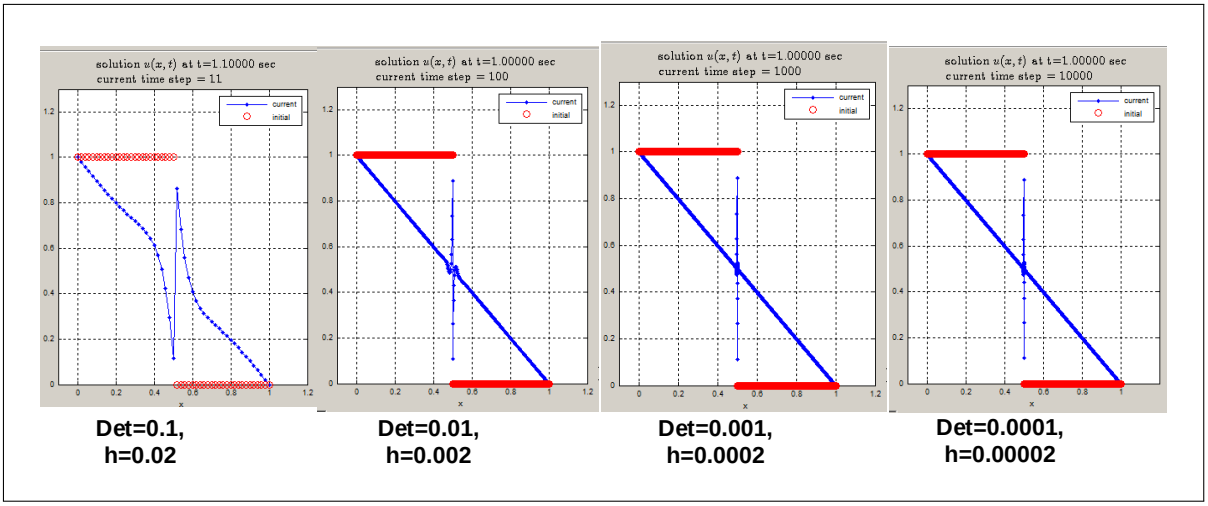


Figure 3.25: solution after one second run

Now, recall from part (b) that the magnification factor for the largest Fourier mode was given by

$$g(\xi_N) = \frac{1 - 2\frac{D\Delta t}{h^2}}{1 + 2\frac{D\Delta t}{h^2}}$$

When $\frac{\Delta t}{h}$ is held constant, say C, then the above becomes

$$g(\xi_N) = \frac{1 - Ch^{-1}}{1 + Ch^{-1}}$$

In the limit, as $h \to 0$ then $g(\xi_N) \to 1$, which implies, as was found in part(b), that large Fourier modes (high p values) will not be attenuated. This is confirmed by the plots above.

3.2.2.4.1 Using backward Euler in place of C-N When using backward Euler. The finite difference scheme for $u_t = Du_{xx}$ becomes

$$\begin{aligned} \frac{u_j^{n+1} - u_j^n}{\Delta t} &= f\left(u^{n+1}\right) \\ &= D\frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{h^2} \end{aligned} \tag{1}$$

Applying Von Neumann analysis, let $u_j^n = e^{i\xi x_j}$, and $u_j^{n+1} = g(\xi)e^{i\xi x_j}$, then (1) becomes

$$\begin{aligned} g\ e^{i\xi x_j}e^{i\xi h} - \ e^{i\xi x_j} &= \frac{D\Delta t}{h^2} g\left(e^{i\xi x_j}e^{-i\xi h} - 2e^{i\xi x_j} + e^{i\xi x_j}e^{i\xi h}\right) \\ g\ e^{i\xi h} - 1 &= \frac{D\Delta t}{h^2} g\left(e^{-i\xi h} - 2 + e^{i\xi h}\right) \\ g\left(e^{i\xi h} - \frac{D\Delta t}{h^2}(2\cos(\xi h) - 2)\right) &= 1 \\ g(\xi) &= \frac{1}{\left(e^{i\xi h} - \frac{D\Delta t}{h^2}(2\cos(\xi h) - 2)\right)} \end{aligned}$$

Therefore

$$g\left(\xi_p\right) = \frac{1}{\left(e^{i\xi_p h} - \frac{D\Delta t}{h^2}\left(2\cos\left(\xi_p h\right) - 2\right)\right)}$$

But $\xi_p = p\pi$ and $p = 1 \cdots N$, and to evaluate what happens to $g\left(\xi_p\right)$ at the largest spatial frequencies, let $\xi_p = N\pi$ and the above becomes

$$g(\xi_N) = \frac{1}{\left(e^{iN\pi h} - \frac{D\Delta t}{h^2}\left(2\cos\left(\frac{N}{N+1}\pi\right) - 2\right)\right)}$$

But $\frac{N}{N+1} \approx 1$ and $e^{iN\pi h} \approx e^{i\pi} = \cos(\pi) + i\sin\pi = -1$, then the above becomes

$$g(\xi_N) = \frac{1}{\left(-1 - \frac{D\Delta t}{h^2}(2\cos\pi - 2)\right)}$$

Therefore

$$g(\xi_N) = \frac{1}{4\frac{D\Delta t}{h^2} - 1}$$

When $\Delta t \gg h$ then $\left|g(\xi_N)\right| \rightarrow \frac{1}{1-1} \rightarrow 0$ which implies that large frequency modes will be knocked out fast. This is opposite to the situation observed using C-N. Hence backward Euler does not have the same problem with large spatial frequencies in initial data. But notice that when $\frac{D\Delta t}{h^2} = \frac{1}{2}$, now $g(\xi_N) = \frac{1}{2-1} = 1$, which means that large frequency mode will not decay or decay very slowly, This is also opposite of what was found for C-N.

3.2.2.5 Appendix for problem 2

3.2.2.5.1 Derivation of part (b) by direct application of DFT (the harder way) The C-N scheme for $u_t = Du_{xx}$ given by

$$-ru_{j-1}^{n+1} + u_j^{n+1}(1 + 2r) - ru_{j+1}^{n+1} = ru_{j-1}^n + u_j^n(1 - 2r) + ru_{j+1}^n \tag{2}$$

Assuming the problem is on the whole real line, then

$$u_j^n = \frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^n(\xi) e^{i\xi x_j} d\xi \tag{1A}$$

and

$$u_j^{n+1} = \frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^{n+1}(\xi) e^{i\xi x_j} d\xi \tag{1B}$$

Where $\hat{u}^n(\xi)$ is the discrete Fourier transform (DFT) of u_j^n. In what follows, $\hat{u}^n$ is written instead of $\hat{u}^n(\xi)$ to make it easier to read the equations. The C-N finite difference scheme for $u_t = Du_{xx}$ is given by

$$-ru_{j-1}^{n+1} + u_j^{n+1}(1+2r) - ru_{j+1}^{n+1} = ru_{j-1}^n + u_j^n(1-2r) + ru_{j+1}^n \tag{2}$$

Where $r = \frac{\Delta t D}{2h^2}$. Substitute (1A) and (1B) into (2), but leaving u_j^{n+1} as is gives

$$-r\left(\frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^{n+1} e^{i\xi\left(x_j-h\right)}\, d\xi\right) + u_j^{n+1}(1+2r) - r\left(\frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^{n+1} e^{i\xi\left(x_j+h\right)} d\xi\right) =$$

$$r\left(\frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^n e^{i\xi\left(x_j-h\right)} d\xi\right) + (1-2r)\left(\frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^n e^{i\xi x_j} d\xi\right) + r\left(\frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^n e^{i\xi\left(x_j+h\right)} d\xi\right)$$

or

$$\frac{-r}{\sqrt{2\pi}}\left(\int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^{n+1} e^{i\xi x_j}\, \left(e^{i\xi h} + e^{-i\xi h}\right) d\xi\right) + u_j^{n+1}(1+2r) =$$

$$\frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \left(r\ \hat{u}^n\ e^{i\xi x_j}\ e^{-i\xi h} + \hat{u}^n\ e^{i\xi x_j} - 2\ r\ \hat{u}^n\ e^{i\xi x_j} + r\ \hat{u}^n\ e^{i\xi x_j}\ e^{i\xi h}\right) d\xi$$

Simplify

$$\begin{aligned} u_j^{n+1}(1+2r) &= \frac{r}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^{n+1} e^{i\xi x_j}\, \left(e^{i\xi h} + e^{-i\xi h}\right) d\xi + \frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^n e^{i\xi x_j} \left[r\left(e^{i\xi h} + e^{-i\xi h}\right) + 1 - 2r\right] d\xi \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^{n+1}\ 2r\cos(\xi h) e^{i\xi x_j} d\xi + \frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \hat{u}^n (2r(\cos(\xi h) - 1) + 1) e^{i\xi x_j} d\xi \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\frac{\pi}{h}}^{\frac{\pi}{h}} \left(\hat{u}^n (2r(\cos(\xi h) - 1) + 1) + 2r\cos(\xi h)\hat{u}^{n+1}\right) e^{i\xi x_j} d\xi \end{aligned}$$

Hence

$$DFT\Big(u_j^{n+1}(1+2r)\Big) = \Big(\hat{u}^n(2r(\cos(\xi h)-1)+1) + 2r\cos(\xi h)\hat{u}^{n+1}\Big)$$
$$DFT\Big(u_j^{n+1}\Big) = \frac{1}{(1+2r)}\Big(\hat{u}^n(2r(\cos(\xi h)-1)+1) + 2r\cos(\xi h)\hat{u}^{n+1}\Big)$$

This implies that

$$\hat{u}^{n+1} = \frac{1}{(1+2r)}\Big(\hat{u}^n(2r(\cos(\xi h)-1)+1) + 2r\cos(\xi h)\hat{u}^{n+1}\Big)$$

Solving for $\hat{u}^{n+1}$ gives

$$\hat{u}^{n+1} - \frac{2r\cos(\xi h)\hat{u}^{n+1}}{(1+2r)} = \frac{1}{(1+2r)}\hat{u}^n(2r(\cos(\xi h)-1)+1)$$
$$\hat{u}^{n+1}\left(\frac{(1+2r)-2r\cos(\xi h)}{(1+2r)}\right) = \frac{\hat{u}^n(2r(\cos(\xi h)-1)+1)}{(1+2r)}$$

Hence

$$\hat{u}^{n+1} = \frac{(2r(\cos(\xi h)-1)+1)}{(1+2r)-2r\cos(\xi h)}\hat{u}^n$$
$$= \frac{1+2r\cos(\xi h)-2r}{1-2r\cos(\xi h)+2r}\hat{u}^n$$

Therefore $\hat{u}^{n+1} = g(\xi)\hat{u}^n$, where $g(\xi) = \frac{1+2r\cos(\xi h)-2r}{1-2r\cos(\xi h)+2r}$, and since $r = \frac{\Delta t D}{2h^2}$, then

$$g(\xi) = \frac{1+\frac{D\Delta t}{h^2}\Big(\cos\Big(p\pi h\Big)-1\Big)}{1-\frac{D\Delta t}{h^2}\Big(\cos\Big(p\pi h\Big)-1\Big)}$$

where $\xi_p = p\pi$, is the wave number.

3.2.2.5.2 Another derivation for the magnification factor The magnification factor $g(\xi)$ found above is the same as the eigenvalue of the update matrix of the C-N scheme. From

$$\left(I - \frac{D\Delta t}{2}L\right)u^{n+1} = \left(I + \frac{D\Delta t}{2}L\right)u^n$$

or

$$u^{n+1} = \left(I - \frac{D\Delta t}{2}L\right)^{-1}\left(I + \frac{D\Delta t}{2}L\right)u^n$$

Where L is the 1D Laplacian grid operator which has eigenvalues $\lambda_p = \frac{2}{h^2}\Big(\cos\Big(p\pi h\Big)-1\Big)$, hence, let μ_p be the eigenvalue of B above, then

$$\begin{aligned}\mu_p &= \frac{1+\frac{D\Delta t}{2}\lambda_p}{1-\frac{D\Delta t}{2}\lambda_p} \\ &= \frac{1+\frac{D\Delta t}{2}\frac{2}{h^2}\Big(\cos\big(p\pi h\big)-1\Big)}{1-\frac{D\Delta t}{2}\frac{2}{h^2}\Big(\cos\big(p\pi h\big)-1\Big)} \\ &= \frac{1+\frac{D\Delta t}{h^2}\Big(\cos\big(p\pi h\big)-1\Big)}{1-\frac{D\Delta t}{h^2}\Big(\cos\big(p\pi h\big)-1\Big)}\end{aligned}$$

Which is what was found in part(b)

3.2.3 Problem 3

3. Derive a stability restriction on the time step for solving the diffusion equation using the second-order accurate explicit Runge-Kutta method

$$y^* = y^n + \Delta t f(y^n)$$
$$y^{n+1} = y^n + \frac{\Delta t}{2}\big(f(y^n) + f(y^*)\big),$$

for time stepping. Does this scheme offer any practical advantage over Forward Euler for the diffusion equation?

Figure 3.26: Problem statement

The finite difference scheme for the diffusion problem is shown the appendix of this problem.

To obtain the absolute stability restriction, let

$$\begin{aligned}
y^{n+1} - y^n &= \frac{\Delta t}{2}\Big(f\big(y^n\big) + f\big(y^*\big)\Big) \\
y^{n+1} &= y^n + \frac{\Delta t}{2}\Big(\lambda y^n + \lambda y^*\Big) \\
&= y^n + \frac{\Delta t}{2}\Big(\lambda y^n + \lambda\Big(y^n + \Delta t \lambda y^n\Big)\Big) \\
&= y^n + \frac{\Delta t}{2}\lambda y^n + \frac{\Delta t}{2}\lambda\Big(y^n + \Delta t \lambda y^n\Big) \\
&= y^n + \frac{\Delta t}{2}\lambda y^n + \frac{\Delta t}{2}\lambda y^n + \frac{(\lambda \Delta t)^2}{2} y^n \\
&= \left(1 + \Delta t \lambda + \frac{(\lambda \Delta t)^2}{2}\right) y^n
\end{aligned}$$

Assuming $\Delta t \lambda = z$

$$y^n + 1 = \left(1 + z + \frac{1}{2}z^2\right) y^n$$

Hence $R(z) = 1 + z + \frac{1}{2}z^2$ and for absolute stability it is required that $|R(z)| \leq 1$ which leads to

$$-1 \leq 1 + z + \frac{1}{2}z^2 \leq 1$$
$$-2 \leq z + \frac{1}{2}z^2 \leq 0$$

A plot of $z + \frac{1}{2}z^2$ shows that $-2 \leq z \leq 0$

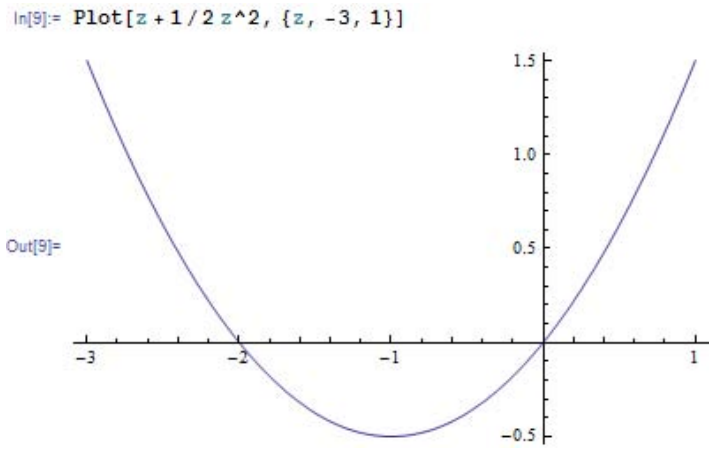


Figure 3.27: absolute stability region

The above gives the interval of absolute stability for the eigenvalues. To obtain the region of absolute stability, assume λ can be complex in general (complex eigenvalue), which results in a disk of radius 1 centered at -1. This is the same region of stability as Forward Euler.

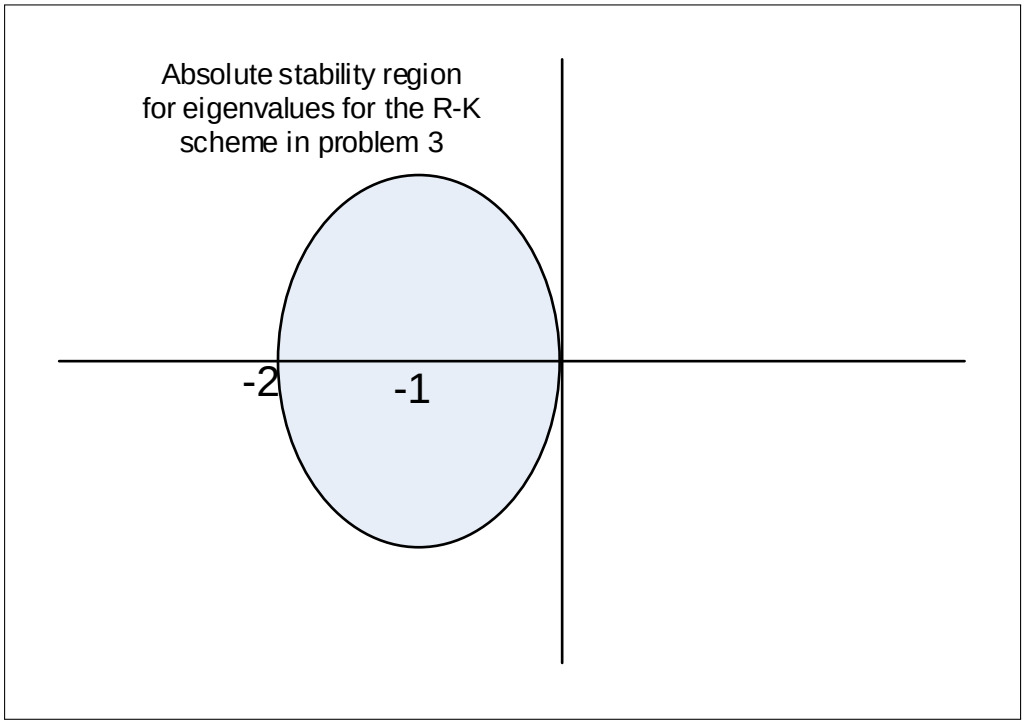


Figure 3.28: region of absolute stability

Notice that since $R(0) = 1$ and $\frac{dR(z)}{dz} = 1$, then this method is also consistent and first order accurate in time.

To answer the question about any advantage of this method over forward Euler. Recalling that In forward Euler

$$u^{n+1} = Bu^n$$

Where B is the update matrix given by $B = I + D\Delta t L_x$, where L_x is the 1-D Laplacian operator for u_{xx} with Dirichlet boundary conditions, with eigenvalue $\lambda = \frac{2}{h^2}\Big(\cos\big(p\pi h\big) - 1\Big)$ where $p = 1 \cdots N$ using the standard grid convention used before.

Let μ be the eigenvalue of the above update matrix B, hence

$$\mu = 1 + \frac{2D\Delta t}{h^2}\Big(\cos\big(p\pi h\big) - 1\Big)$$

For stability, $|u| \leq 1$ hence

$$\left|1 + \frac{2D\Delta t}{h^2}\Big(\cos\big(p\pi h\big) - 1\Big)\right| \leq 1$$
$$\left|1 - 4\frac{D\Delta t}{h^2}\right| \leq 1$$

Simplifying, this gives

$$0 \leq \frac{D\Delta t}{h^2} \leq \frac{1}{2}$$

To compare the above with the R-K scheme in this problem, since

$$\frac{u^* - u^n}{\Delta t} = Lu^n$$
$$\frac{u^{n+1} - u^n}{\Delta t} = \frac{1}{2}(Lu^n + Lu^*) \tag{1}$$

Expanding gives

$$\begin{aligned} u^{n+1} &= u^n + \frac{\Delta t}{2}(Lu^n + L(u^n + \Delta t L u^n)) \\ &= u^n + \frac{\Delta t}{2}\Big(Lu^n + Lu^n + \Delta t L^2 u^n\Big) \\ &= \left(I + \frac{\Delta t}{2}\Big(2L + \Delta t L^2\Big)\right)u^n \\ &= Bu^n \end{aligned}$$

The eigenvalue of L is $\frac{2}{h^2}\Big(\cos\big(p\pi h\big) - 1\Big)$ but since $\cos A - 1 = -2\sin^2\frac{A}{2}$, hence $\cos\big(p\pi h\big) - 1 = -2\sin^2\frac{p\pi h}{2}$ and the eigenvalue is written as $-\frac{4}{h^2}\sin^2\frac{p\pi h}{2}$.

let $\sin^2\frac{p\pi h}{2} \equiv \omega$, therefore the eigenvalue of L becomes $\lambda = -\frac{4\omega}{h^2}$.

Using this notation, the above update matrix B for this scheme will have the following eigenvalue

$$\begin{aligned} \mu &= 1 + \frac{\Delta t}{2}\left(-D\frac{8\omega}{h^2} + \Delta t\left(-D\frac{4\omega}{h^2}\right)^2\right) \\ &= 1 - \frac{4D\Delta t\omega}{h^2} + \Delta^2 t\frac{8D^2\omega^2}{h^4} \end{aligned}$$

For stability, $|\mu| \le 1$. Maximum μ will occur at $\omega = 1$ implying that $\sin^2 \frac{p\pi h}{2} = 1$ or $p = N$, resulting in

$$\left|1 - \frac{4D\Delta t}{h^2} + \Delta^2 t \frac{8D^2}{h^4}\right| \le 1$$

Therefore

$$\begin{aligned}
-1 \le 1 - \frac{4D\Delta t}{h^2} + \Delta^2 t \frac{8D^2}{h^4} &\le 1 \\
-2 \le -\frac{4D\Delta t}{h^2} + \Delta^2 t \frac{8D^2}{h^4} &\le 0 \\
0 \le \frac{4D\Delta t}{h^2} - \Delta^2 t \frac{8D^2}{h^4} &\le 2 \\
0 \le \frac{D\Delta t}{h^2} - \Delta^2 t \frac{2D^2}{h^4} &\le \frac{1}{2} \\
0 \le \frac{D\Delta t}{h^2} - 2\left(\frac{D\Delta t}{h^2}\right)^2 &\le \frac{1}{2}
\end{aligned}$$

This shows that with the given RK scheme, stability implies the condition $0 \le \frac{D\Delta t}{h^2} - \Delta^2 t \frac{2D^2}{h^4} \le \frac{1}{2}$. This is compared to $0 \le \frac{D\Delta t}{h^2} \le \frac{1}{2}$ for forward Euler.

What does this mean in terms of the time step? Will this allow the use of a larger time step than with FE while keep absolute stability?

Assume $D = 1$, This is a table showing the maximum value of Δt allowed for different h values

scheme	$h = 1$	$h = 0.1$	$h = 0.01$
FE $0 \le \frac{\Delta t}{h^2} \le \frac{1}{2}$	0.5	0.05	0.005
RK $0 \le \frac{\Delta t}{h^2} - \frac{2\Delta^2 t}{h^4} \le \frac{1}{2}$	0.5	0.05	0.005

Hence, the largest time step does not change with this scheme when compared to forward Euler. It seems based on the above, that the explicit Runge-Kutta scheme for solving the diffusion PDE does not offer any advantage in handling the stiffness of the PDE since the time step remained constrained by h^2 as with FE. It seems that explicit schemes are not suitable for stiff problems

3.2.3.1 Appendix for problem 3

3.2.3.2 Derivation of the update matrix for 2 step R-K for diffusion problem with Dirichlet B.C.

Given

$$y_t = Dy_{xx}$$

Then

$$\frac{y_i^* - y_i^n}{\Delta t} = D\frac{y_{i-1}^n - 2y_i^n + y_{i+1}^n}{h^2}$$

$$y_i^* = y_i^n + \Delta t\overbrace{\frac{D}{h^2}\left(y_{i-1}^n - 2y_i^n + y_{i+1}^n\right)}^{f\left(y^n\right)}$$

Hence

$$\begin{aligned}
2\frac{y_i^{n+1} - y_i^n}{\Delta t} &= \left(f\left(y^n\right) + f\left(y^*\right)\right) \\
&= \left[\frac{D}{h^2}\left(y_{i-1}^n - 2y_i^n + y_{i+1}^n\right)\right] + \left[\frac{D}{h^2}\left(y_{i-1}^* - 2y_i^* + y_{i+1}^*\right)\right]
\end{aligned}$$

$$\begin{aligned}
&= \frac{D}{h^2}\left(y_{i-1}^n - 2y_i^n + y_{i+1}^n\right) + \\
&\frac{D}{h^2}\left[y_{i-1}^n + \Delta t\frac{D}{h^2}\left(y_{i-2}^n - 2y_{i-1}^n + y_i^n\right)\right] - \\
&2\frac{D}{h^2}\left[y_i^n + \Delta t\frac{D}{h^2}\left(y_{i-1}^n - 2y_i^n + y_{i+1}^n\right)\right] + \\
&\frac{D}{h^2}\left[y_{i+1}^n + \Delta t\frac{D}{h^2}\left(y_i^n - 2y_{i+1}^n + y_{i+2}^n\right)\right]
\end{aligned}$$

Expand and simplify

$$\begin{aligned}
2\frac{y_i^{n+1} - y_i^n}{\Delta t} &= \frac{D}{h^2}y_{i-1}^n\left(1 + 1 - 2\Delta t\frac{D}{h^2} - 2\Delta t\frac{D}{h^2}\right) \\
&+ \frac{D}{h^2}y_i^n\left(-2 + \Delta t\frac{D}{h^2} - 2 - 4\Delta t\frac{D}{h^2} + \Delta t\frac{D}{h^2}\right) \\
&+ \frac{D}{h^2}y_{i+1}^n\left(1 - 2\Delta t\frac{D}{h^2} + 1 - 2\Delta t\frac{D}{h^2}\right) \\
&+ \frac{D}{h^2}y_{i+2}^n\left(\Delta t\frac{D}{h^2}\right)
\end{aligned}$$

$$= \frac{D}{h^2}\left[y_{i-1}^n\left(2 - 4\Delta t\frac{D}{h^2}\right) + y_i^n\left(-4 - 2\Delta t\frac{D}{h^2}\right) + y_{i+1}^n\left(2 - 4\Delta t\frac{D}{h^2}\right) + y_{i+2}^n\left(\Delta t\frac{D}{h^2}\right)\right]$$

Hence

$$\begin{aligned} y_i^{n+1} &= y_i^n + \frac{D\Delta t}{2h^4}\left[y_{i-1}^n\left(2h^2 - 4\Delta tD\right) + y_i^n\left(-4h^2 - 2\Delta tD\right) + y_{i+1}^n\left(2h^2 - 4\Delta tD\right) + y_{i+2}^n(\Delta tD)\right] \\ &= y_i^n + \frac{D^2\Delta^2 t}{2h^4}\left[2y_{i-1}^n\left(\frac{h^2}{\Delta tD} - 2\right) - 2y_i^n\left(\frac{2h^2}{\Delta tD} + 1\right) + 2y_{i+1}^n\left(\frac{h^2}{\Delta tD} - 2\right) + y_{i+2}^n\right] \end{aligned}$$

$$\begin{aligned} &= y_{i-1}^n\left(\frac{D^2\Delta^2 t}{2h^4}\left(\frac{h^2}{\Delta tD} - 2\right)\right) + y_i^n\left(1 - \frac{D^2\Delta^2 t}{h^4}\left(\frac{2h^2}{\Delta tD} + 1\right)\right) + \\ &y_{i+1}^n\left(\frac{D^2\Delta^2 t}{h^4}\left(\frac{h^2}{\Delta tD} - 2\right)\right) + y_{i+2}^n\frac{D^2\Delta^2 t}{2h^4} \end{aligned}$$

Let $\frac{D^2\Delta^2 t}{h^4} = r_1$ and let $\frac{h^2}{\Delta tD} = r_2$ then[4]

$$\begin{aligned} y_i^{n+1} &= y_{i-1}^n\left(\frac{r_1}{2}(r_2 - 2)\right) + y_i^n(1 - r_1(2r_2 + 1)) + y_{i+1}^n(r_1(r_2 - 2)) + y_{i+2}^n\frac{r_1}{2} \\ &= y_{i-1}^n\left(\frac{r_1 r_2}{2} - r_1\right) + y_i^n(1 - 2r_1 r_2 - r_1) + y_{i+1}^n(r_1 r_2 - 2r_1) + y_{i+2}^n\frac{r_1}{2} \end{aligned}$$

Using this 1-D grid

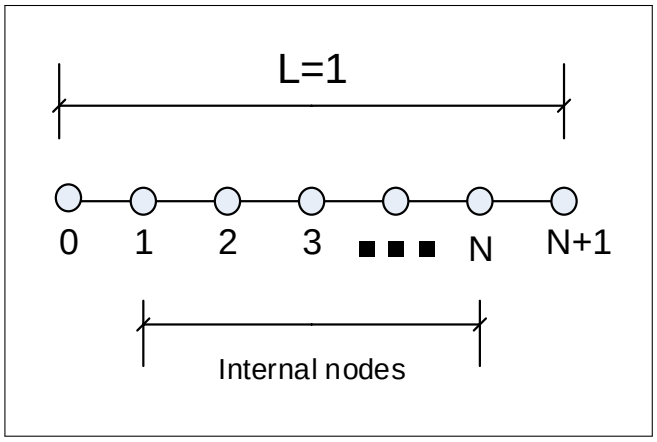


Figure 3.29: Grid format

[4]Notice that units of D are $meter^2$ per second, hence $\frac{h^2}{\Delta tD}$ is dimensionless, so we are ok.

The matrix form of the scheme becomes

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_{N-1}^{n+1} \\ u_N^{n+1} \end{bmatrix} = \begin{bmatrix} (1-2r_1r_2-r_1) & (r_1r_2-2r_1) & \frac{r_1}{2} & 0 & 0 & 0 \\ \left(\frac{r_1r_2}{2}-r_1\right) & (1-2r_1r_2-r_1) & (r_1r_2-2r_1) & \frac{r_1}{2} & 0 & 0 \\ 0 & \left(\frac{r_1r_2}{2}-r_1\right) & (1-2r_1r_2-r_1) & (r_1r_2-2r_1) & \frac{r_1}{2} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \left(\frac{r_1r_2}{2}-r_1\right) & (1-2r_1r_2-r_1) & (r_1r_2-2r_1) \\ 0 & 0 & 0 & 0 & \left(\frac{r_1r_2}{2}-r_1\right) & (1-2r_1r_2-r_1) \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_N^n \end{bmatrix} + \begin{bmatrix} \left(\frac{r_1r_2}{2}-r_1\right)u_0^n \\ 0 \\ 0 \\ \vdots \\ y_{N+1}^n \frac{r_1}{2} \\ (r_1r_2-2r_1)u_{N+1}^n \end{bmatrix}$$

There is a problem above at node N as two additional nodes are needed to its right, but only one node exist. Need to look more into this later, as this part is not required for this HW.

3.2.4 Problem 4

4. Consider the forward time, centered space discretization

$$\frac{u_j^{n+1}-u_j^n}{\Delta t}+a\frac{u_{j+1}^n-u_{j-1}^n}{2h}=b\frac{u_{j-1}^n-2u_j^n+u_{j+1}^n}{h^2},$$

to the convection-diffusion equation,

$$u_t+au_x=bu_{xx},\quad b>0.$$

(a) Let $\nu=a\Delta t/h$ and $\mu=b\Delta t/h^2$. Use von Neumann analysis to show that the scheme is stable if $\mu\le 1/2$.

(b) Let $a=80$, $b=1$, $h=0.05$. Generate a numerical solution on the spatial domain $[0,1]$ with periodic boundary conditions using $\Delta t=0.25h^2/b$ with initial condition $u(x,0)=\exp(-20(x-0.5)^2)$. What happens? Does your stability analysis predict this?

(c) Since the solution to the PDE does not grow in time, it seems reasonable to require that the numerical solution not grow in time. Show that the numerical solution does not grow (in 2-norm) if and only if $\nu^2\le 2\mu\le 1$. This is called strict or practical stability, and as the name suggests it is the restriction one would use in practice.

(d) Generate a numerical solution up to time $t=10^{-2}$.

Figure 3.30: Problem statement

3.2.4.1 Part (a)

The PDE is $u_t+au_x=bu_{xx}$, with $b>0$, the forward time, centered space discretization is

$$\frac{u_j^{n+1}-u_j^n}{\Delta t}+a\frac{u_{j+1}^n-u_{j-1}^n}{2h}=b\frac{u_{j-1}^n-2u_j^n+u_{j+1}^n}{h^2}\tag{1}$$

Applying Von Neumann analysis, let $u_j^n=e^{i\xi x_j}$ and $u_j^{n+1}=g(\xi)e^{i\xi x_j}$, then (1) becomes

$$\begin{aligned}
\frac{ge^{i\xi x_j}-e^{i\xi x_j}}{\Delta t}+a\frac{e^{i\xi x_j}e^{i\xi h}-e^{i\xi x_j}e^{-i\xi h}}{2h}&=b\frac{e^{i\xi x_j}e^{-i\xi h}-2e^{i\xi x_j}+e^{i\xi x_j}e^{i\xi h}}{h^2}\\
ge^{i\xi x_j}-e^{i\xi x_j}+\frac{a\Delta t}{2h}\left(e^{i\xi x_j}e^{i\xi h}-e^{i\xi x_j}e^{-i\xi h}\right)&=\frac{b\Delta t}{h^2}\left(e^{i\xi x_j}e^{-i\xi h}-2e^{i\xi x_j}+e^{i\xi x_j}e^{i\xi h}\right)\\
g-1+\frac{\nu}{2}\left(e^{i\xi h}-e^{-i\xi h}\right)&=\mu\left(e^{-i\xi h}-2+e^{i\xi h}\right)\\
g(\xi)&=1-\frac{\nu}{2}\left(e^{i\xi h}-e^{-i\xi h}\right)+\mu\left(e^{-i\xi h}-2+e^{i\xi h}\right)
\end{aligned}$$

Hence[5]

$$g(\xi)=1+2\mu(\cos(\xi h)-1)-i\nu\sin(\xi h)$$

[5]Notice that the first oder derivatives (or odd order in general) produces eigenvalues that are complex, and the even order ones produce real eigenvalues.

But $\cos A - 1 = -2\sin^2 \frac{A}{2}$, hence $\cos(\xi h) - 1 = -2\sin^2 \frac{\xi h}{2}$ and the above becomes

$$g(\xi) = 1 - 4\mu \sin^2\left(\frac{\xi h}{2}\right) - i\nu \sin(\xi h)$$

Therefore

$$\left|g(\xi)\right|^2 = \left[1 - 4\mu \sin^2\left(\frac{\xi h}{2}\right)\right]^2 + \nu^2 \sin^2(\xi h) \tag{2}$$

Using the trig identity

$$\sin^2(\xi h) = 4\sin^2\left(\frac{\xi h}{2}\right)\left(1 - \sin^2\left(\frac{\xi h}{2}\right)\right)$$

then (2) becomes

$$\left|g(\xi)\right|^2 = \left[1 - 4\mu \sin^2\left(\frac{\xi h}{2}\right)\right]^2 + 4\nu^2 \sin^2\left(\frac{\xi h}{2}\right)\left(1 - \sin^2\left(\frac{\xi h}{2}\right)\right)$$

Let $\sin^2\left(\frac{\xi h}{2}\right) \equiv \omega$ in the above

$$\left|g(\xi)\right|^2 = \left(1 - 4\mu\omega\right)^2 + 4\nu^2\omega(1 - \omega)$$

The maximum of $\left|g\left(\xi_p\right)\right|$ occurs when $\xi h \approx \pi$, making $\omega = 1$, hence from above, the maximum of $\left|g(\xi)\right|^2$ is reduced to $1 - 4\mu$, and then for stability

$$\left|1 - 4\mu\right| \leq 1$$

therefore

$$\begin{aligned} -1 \leq 1 - 4\mu &\leq 1 \\ -2 \leq -4\mu &\leq 0 \\ 0 \leq 4\mu &\leq 2 \end{aligned}$$

Hence

$$0 \leq \mu \leq \frac{1}{2}$$

The above result can also be derived using the stencil diagram method. The stencil diagram for the above scheme (for internal nodes only) is

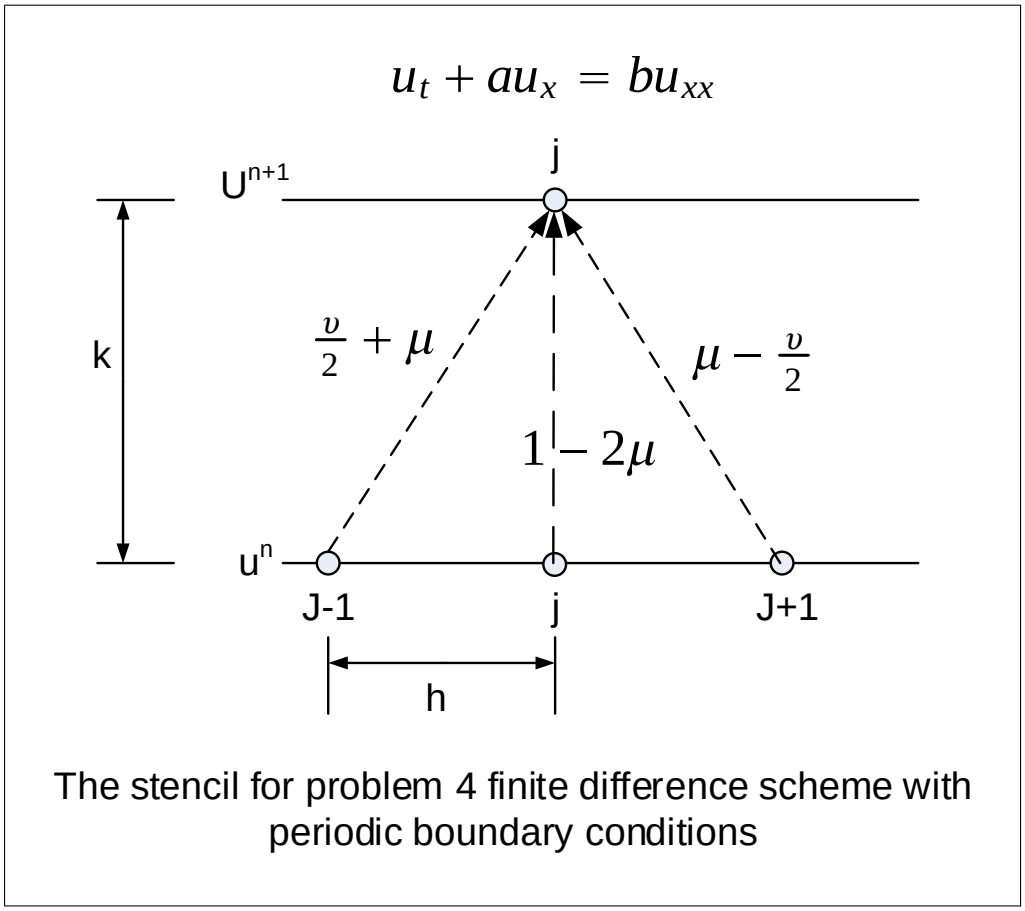


Figure 3.31: Grid used

As was done in problem 1, By imposing that the weight on each edge in the above directed graph not exceed unity, and that the total algebraic sum of the weight of the edge also not exceed unity. This includes any combination of edges involved. For if this was the case, then u_j^{n+1} will always have an amplitude $\leq u_j^n$. Applying this to the above diagram gives

$$\begin{cases} (1) & \frac{v}{2} + \mu \leq 1 & \text{Condition on } j-1 \text{ edge} \\ (2) & \left|\mu - \frac{v}{2}\right| \leq 1 & \text{Condition on } j+1 \text{ edge} \\ (3) & 2\mu \leq 1 & \text{Condition on } j-1 \text{ added to } j+1 \text{ edge} \\ (4) & |1-2\mu| \leq 1 & \text{Condition on the } j \text{ edge} \\ (5) & \left|1 + \frac{v}{2} - \mu\right| \leq 1 & \text{Condition on the } j \text{ edge added to } j-1 \\ (6) & \left|1 - \frac{v}{2} - \mu\right| \leq 1 & \text{Condition on the } j \text{ edge added to } j+1 \\ (7) & 1 \leq 1 & \text{Condition that all edges sum to less than 1} \end{cases}$$

Condition (7) gives no information. Condition (4) gives $\mu \leq 1$ and hence weaker than (3), condition (5) is the same as (2) and gives $\mu - \frac{v}{2} \leq 1$, condition (6) gives $\frac{v}{2} + \mu \leq 2$ which is

weaker than condition (1). Hence the following are remaining conditions (3),(1),(2).

$$\begin{cases} (1) & \frac{v}{2} + \mu \leq 1 & \text{Condition on } j-1 \text{ edge} \\ (2) & \left|\mu - \frac{v}{2}\right| \leq 1 & \text{Condition on } j+1 \text{ edge} \\ (3) & 2\mu \leq 1 & \text{Condition on } j-1 \text{ added to } j+1 \text{ edge} \end{cases}$$

From the above, only condition (3) can provide useful information, which is that $\mu \leq \frac{1}{2}$, which is what was found using Von Neumann analysis.

3.2.4.2 Part (b)

The scheme was implemented. The source code is in the appendix of this problem. The grid used is the standard grid

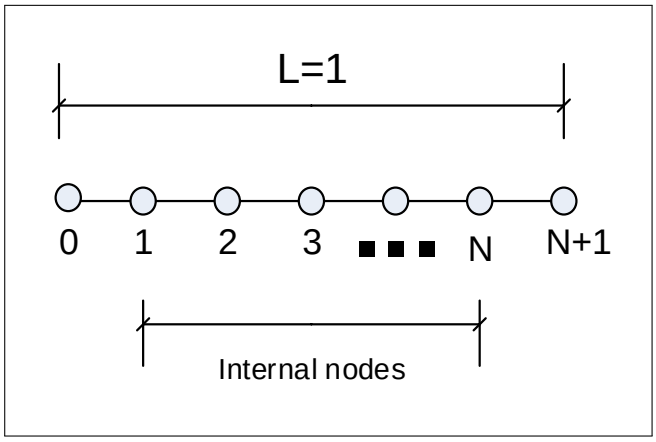


Figure 3.32: Grid used

In the following, I will be use the following PDE $cu_t + au_x = bu_{xx}$ (where$c$ was added a parameter for the advection term).

Periodic boundary conditions implies $u(0,t) = u(1,t)$, Hence there is an extra one unknown (in addition to the internal nodes). Either $u(0,t)$ or $u(1,t)$ can be selected since they have the same value. When selecting the right end node, then u_{N+1} becomes an unknown to be added to the internal nodes. Using the following diagram

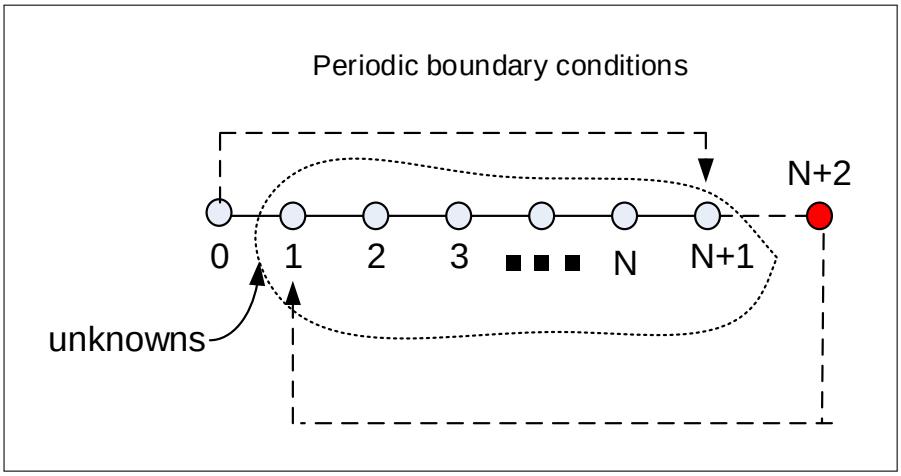


Figure 3.33: Grid used

The forward time, centered space discretization is

$$c\frac{u_j^{n+1}-u_j^n}{\Delta t}+a\frac{u_{j+1}^n-u_{j-1}^n}{2h}=b\frac{u_{j-1}^n-2u_j^n+u_{j+1}^n}{h^2} \tag{1}$$

Therefore, for nodes $2\cdots N$, the finite difference scheme is

$$\begin{aligned}
c\frac{u_j^{n+1}-u_j^n}{\Delta t}+a\frac{u_{j+1}^n-u_{j-1}^n}{2h}&=b\frac{u_{j-1}^n-2u_j^n+u_{j+1}^n}{h^2}\\
u_j^{n+1}&=u_j^n-a\Delta t\frac{u_{j+1}^n-u_{j-1}^n}{2ch}+b\Delta t\frac{u_{j-1}^n-2u_j^n+u_{j+1}^n}{ch^2}\\
u_j^{n+1}&=u_j^n-\frac{a\Delta t}{2ch}\left(u_{j+1}^n-u_{j-1}^n\right)+\frac{b\Delta t}{ch^2}\left(u_{j-1}^n-2u_j^n+u_{j+1}^n\right)\\
u_j^{n+1}&=u_{j-1}^n\left(\frac{a\Delta t}{2ch}+\frac{b\Delta t}{ch^2}\right)+u_j^n\left(1-2\frac{b\Delta t}{ch^2}\right)+u_{j+1}^n\left(-\frac{a\Delta t}{2ch}+\frac{b\Delta t}{ch^2}\right)
\end{aligned}$$

Let $v=\frac{a\Delta t}{ch}$, $\mu=\frac{b\Delta t}{ch^2}$, the above becomes

$$u_j^{n+1}=u_{j-1}^n\left(\frac{v}{2}+\mu\right)+u_j^n\left(1-2\mu\right)+u_{j+1}^n\left(\mu-\frac{v}{2}\right)$$

Node $j=1$ gives

$$\begin{aligned}
u_1^{n+1}&=u_0^n\left(\frac{v}{2}+\mu\right)+u_1^n\left(1-2\mu\right)+u_2^n\left(\mu-\frac{v}{2}\right)\\
&=u_{N+1}^n\left(\frac{v}{2}+\mu\right)+u_1^n\left(1-2\mu\right)+u_2^n\left(\mu-\frac{v}{2}\right)
\end{aligned}$$

And for node $j=N+1$

$$\begin{aligned}
u_{N+1}^{n+1}&=u_N^n\left(\frac{v}{2}+\mu\right)+u_{N+1}^n\left(1-2\mu\right)+u_{N+2}^n\left(\mu-\frac{v}{2}\right)\\
&=u_N^n\left(\frac{v}{2}+\mu\right)+u_{N+1}^n\left(1-2\mu\right)+u_1^n\left(\mu-\frac{v}{2}\right)
\end{aligned}$$

The full system can now be written in matrix form

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_N^{n+1} \\ u_{N+1}^{n+1} \end{bmatrix} = \begin{bmatrix} (1-2\mu) & \left(\mu - \frac{v}{2}\right) & 0 & 0 & 0 & \left(\frac{v}{2}+\mu\right) \\ \left(\mu + \frac{v}{2}\right) & (1-2\mu) & \left(\mu - \frac{v}{2}\right) & 0 & 0 & 0 \\ 0 & \left(\mu + \frac{v}{2}\right) & (1-2\mu) & \left(\mu - \frac{v}{2}\right) & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \left(\mu + \frac{v}{2}\right) & (1-2\mu) & \left(\mu - \frac{v}{2}\right) \\ \left(\mu - \frac{v}{2}\right) & 0 & 0 & 0 & \left(\mu + \frac{v}{2}\right) & (1-2\mu) \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_N^n \\ u_{N+1}^n \end{bmatrix}$$

The above can also be written as

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_N^{n+1} \\ u_{N+1}^{n+1} \end{bmatrix} = \overbrace{\left(I - \frac{a\Delta t}{2ch} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & -1 \\ -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ \vdots & \vdots & -1 & \ddots & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 0 & -1 & 0 \end{bmatrix} + \frac{b\Delta t}{ch^2} \begin{bmatrix} -2 & 1 & 0 & 0 & 0 & 1 \\ 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 1 & 0 & 0 & 0 & 1 & -2 \end{bmatrix} \right)}^{update\ matrix} \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_N^n \\ u_{N+1}^n \end{bmatrix}$$

u_j^0 are taken from initial conditions. The above is in the form

$$u^{n+1} = Bu^n$$

and is implemented directly as above in the code. Using the numerical values given in the problem

$$\Delta t = \frac{0.25h^2}{b} = 0.25\frac{(0.05)^2}{1} = 0.000625$$

and

$$v = \frac{a\Delta t}{h} = \frac{80(0.000625)}{0.05} = 1$$

and

$$\mu = \frac{b\Delta t}{h^2} = \frac{0.000625}{(0.05)^2} = 0.25$$

since $\mu \leq \frac{1}{2}$ the solution is expected to be stable since this is the condition derived in part (a).

The following is the result of running the program. The numerical solution grew with time. Here are few snap shots taken at increasing time steps showing the problem. After only about 10 time steps, the numerical solution can be seen to grow more than the initial conditions

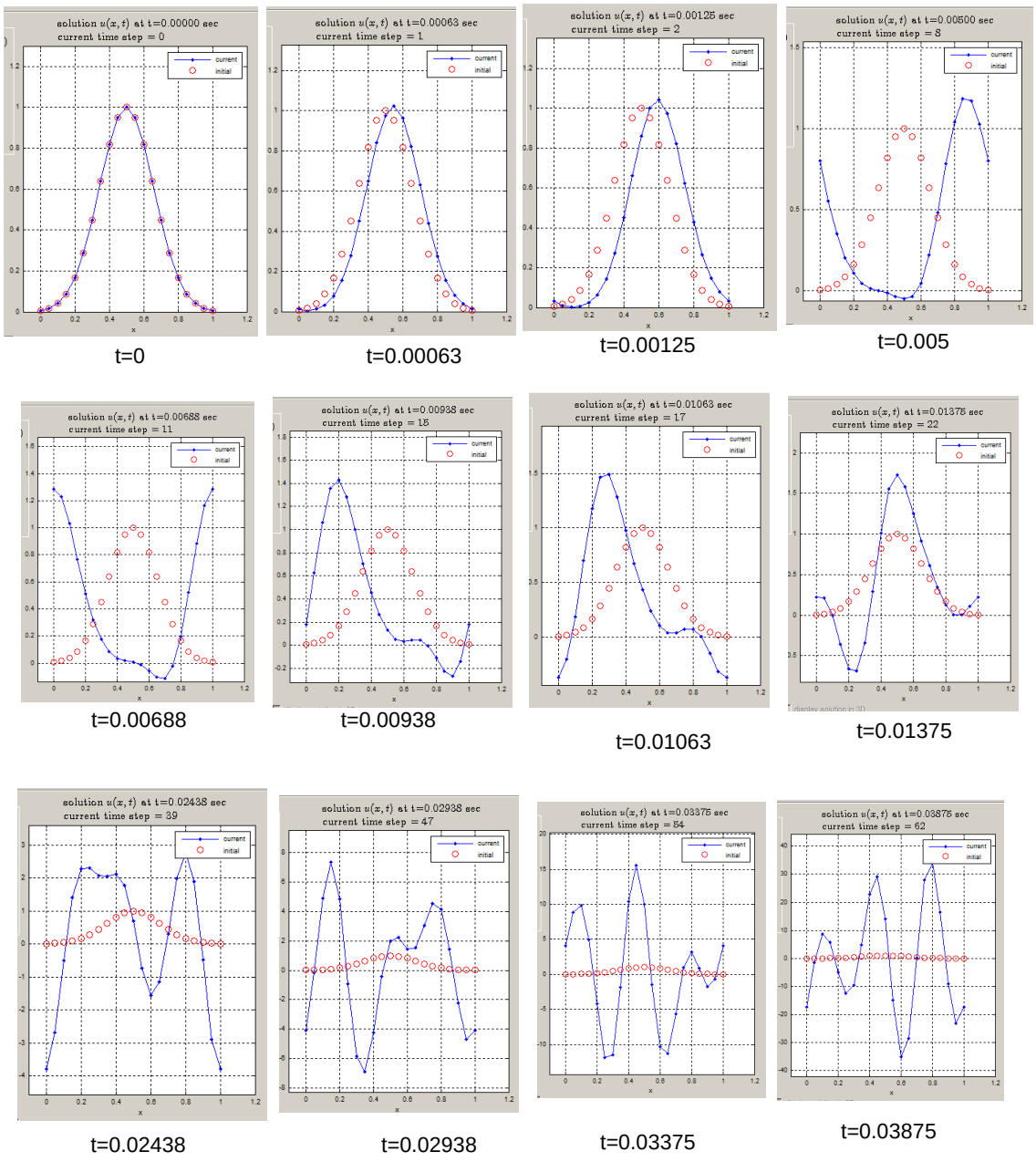


Figure 3.34: few snap shots taken at increasing time steps showing the problem

The stability analysis that was done did not predict this based on the value of μ which was $\leq \frac{1}{2}$.

3.2.4.3 Part(c)

The eigenvalues u_p of the update matrix derived in part (b) are

$$u_p = 1 - i\frac{a\Delta t}{h}\sin\left(\pi p h\right) + \frac{2b\Delta t}{h^2}\left(\cos\left(\pi p h\right) - 1\right)$$

But $v = \frac{a\Delta t}{h}$, $\mu = \frac{b\Delta t}{h^2}$, then the above becomes

$$u_p = 1 - iv\sin\left(\pi p h\right) + 2\mu\left(\cos\left(\pi p h\right) - 1\right)$$

Let $\sin^2\left(\frac{\xi h}{2}\right) \equiv \omega$, then the above can be written as

$$\left|u_p\right|^2 = \left(1 - 4\mu\omega\right)^2 + 4v^2\omega(1-\omega)$$

As was done in part (a).

From part (a), it was found that when $\omega = 1$, this resulted in the condition of stability being $0 \le \mu \le \frac{1}{2}$, and when $\omega = 0$, the maximum eigenvalue is 1. To find the condition of $\left|u_p\right| \le 1$ for the full range of ω, first expand $\left|u_p\right|^2$ into a quadratic in ω and minimize

$$\begin{aligned}\left|u_p\right|^2 &= 1 + 16\mu^2\omega^2 - 8\mu\omega + 4v^2\omega - 4v^2\omega^2 \\ &= 1 - 4\omega\left(2\mu - v^2\right) + 4\omega^2\left(4\mu^2 - v^2\right)\end{aligned}$$

Since $\omega = \sin^2\left(\frac{\pi p h}{2}\right)$, then ω values are from $0 \cdots 1$. Since the maximum eigenvalue occurs when $\omega = 0$, then $\omega = 0$ is the maximum point of the quadratic $1 - 4\omega\left(2\mu - v^2\right) + 4\omega^2\left(4\mu^2 - v^2\right)$, hence the slope of this quadratic at $\omega = 0$ must be negative. But the slope is

$$\begin{aligned}\frac{d}{d\omega}\left|u_p\right|^2 &= \frac{d}{d\omega}\left(1 - 4\omega\left(2\mu - v^2\right) + 4\omega^2\left(4\mu^2 - v^2\right)\right) \\ &= -4\left(2\mu - v^2\right) + 8\omega\left(4\mu^2 - v^2\right)\end{aligned}$$

For the above to be negative (so that the eigenvalue remain below 1) implies that $2\mu - v^2$ must be positive, i.e.

$$2\mu - v^2 \ge 0$$

or

$$v^2 \le 2\mu$$

And since from part(a) it was found that $\mu \le \frac{1}{2}$, or $2\mu \le 1$ then the above become

$$v^2 \le 2\mu \le 1$$

3.2.4.4 Part(d)

The solution for $\Delta t = 0.000625$, $h = 0.05$, $a = 80$, $b = 1$, with $u(x,0) = \exp\left(-20(x-0.5)^2\right)$ at $t = 0.01$ seconds is

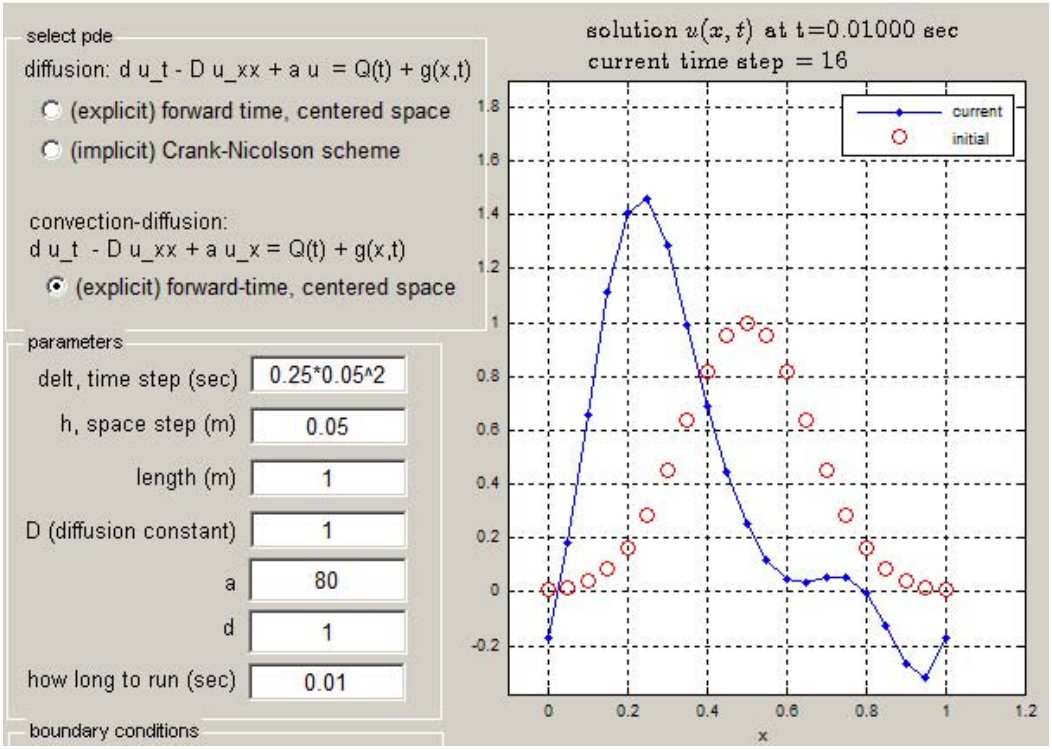


Figure 3.35: Part c final solution

The value of the numerical solution at that time is

```
-0.1727
 0.1817
 0.6603
 1.1158
 1.4092
 1.4609
 1.2868
 0.9910
 0.6912
 0.4441
 0.2527
 0.1168
 0.0451
 0.0337
 0.0532
```

```
 0.0563
-0.0013
-0.1275
-0.2680
-0.3162
-0.1727
```

3.2.4.5 Appendix for problem 4

3.2.4.5.1 derivation of the convection-diffusion using general terms The PDE is $cu_t + au_x = bu_{xx}$, with $b > 0$, the forward time, centered space discretization is

$$c\frac{u_j^{n+1} - u_j^n}{\Delta t} + a\frac{u_{j+1}^n - u_{j-1}^n}{2h} = b\frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{h^2} \tag{1}$$

Therefore, for nodes $2 \cdots N$, the finite difference scheme is

$$\begin{aligned}
c\frac{u_j^{n+1} - u_j^n}{\Delta t} + a\frac{u_{j+1}^n - u_{j-1}^n}{2h} &= b\frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{h^2} \\
u_j^{n+1} &= u_j^n - a\Delta t\frac{u_{j+1}^n - u_{j-1}^n}{2ch} + b\Delta t\frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{ch^2} \\
u_j^{n+1} &= u_j^n - \frac{a\Delta t}{2ch}\left(u_{j+1}^n - u_{j-1}^n\right) + \frac{b\Delta t}{ch^2}\left(u_{j-1}^n - 2u_j^n + u_{j+1}^n\right) \\
u_j^{n+1} &= u_{j-1}^n\left(\frac{a\Delta t}{2ch} + \frac{b\Delta t}{ch^2}\right) + u_j^n\left(1 - 2\frac{b\Delta t}{ch^2}\right) + u_{j+1}^n\left(-\frac{a\Delta t}{2ch} + \frac{b\Delta t}{ch^2}\right)
\end{aligned}$$

Let $v = \frac{a\Delta t}{ch}$, $\mu = \frac{b\Delta t}{ch^2}$, the above becomes

$$u_j^{n+1} = u_{j-1}^n\left(\frac{v}{2} + \mu\right) + u_j^n\left(1 - 2\mu\right) + u_{j+1}^n\left(\mu - \frac{v}{2}\right)$$

Node $j = 1$ gives

$$\begin{aligned}
u_1^{n+1} &= u_0^n\left(\frac{v}{2} + \mu\right) + u_1^n\left(1 - 2\mu\right) + u_2^n\left(\mu - \frac{v}{2}\right) \\
&= u_{N+1}^n\left(\frac{v}{2} + \mu\right) + u_1^n\left(1 - 2\mu\right) + u_2^n\left(\mu - \frac{v}{2}\right)
\end{aligned}$$

And for node $j = N + 1$

$$\begin{aligned}
u_{N+1}^{n+1} &= u_N^n\left(\frac{v}{2} + \mu\right) + u_{N+1}^n\left(1 - 2\mu\right) + u_{N+2}^n\left(\mu - \frac{v}{2}\right) \\
&= u_N^n\left(\frac{v}{2} + \mu\right) + u_{N+1}^n\left(1 - 2\mu\right) + u_1^n\left(\mu - \frac{v}{2}\right)
\end{aligned}$$

3.2.5 Screen shot of the GUI matlab application used for HW1

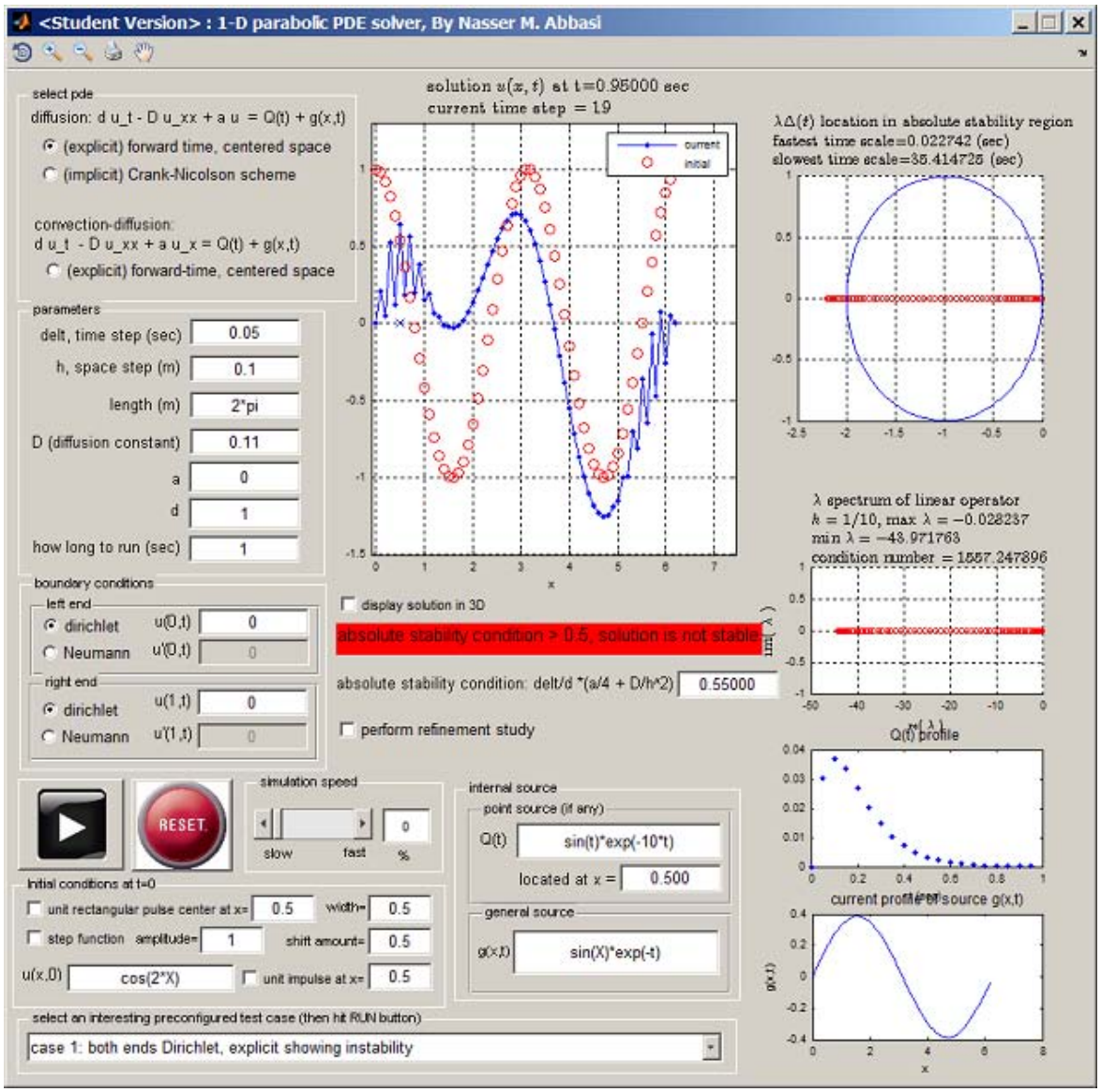


Figure 3.36: Matlab program I developed for this HW

3.2.6 Matlab Source code developed for this HW

3.2.6.1 nma_math228b_build_HW1.m

```
function nma_math228b_build_HW1()

list = dir('*.m');

if isempty(list)
    fprintf('no matlab files found\n');
    return
end

for i=1:length(list)
    name=list(i).name;
    fprintf('processing %s\n',name)
    p0 = fdep(list(i).name,'-q');
    [pathstr, name_of_matlab_function, ext] = fileparts(name);

    %make a zip file of the m file and any of its dependency
    p1=dir([name_of_matlab_function '.fig']);
    if length(p1)==1
        files_to_zip =[p1(1).name;p0.fun];
    else
        files_to_zip =p0.fun;
    end

    zip([name_of_matlab_function '.zip'],files_to_zip)

end

end
```

3.2.6.2 nma_math228b_HW1.m

```
function nma_math_228b_HW1

t=0:0.1:100;
x=0.01:0.01:1;
sol=zeros(length(x),1);

for i=1:length(t)
   for j=1:length(x)
      sol(j)=  uh(x(j),t(i)) + up(x(j))  ;
   end
   plot(x,sol);
   title(sprintf('t=%f',t(i)));
   drawnow();
```

```
   pause(.01);

end
end


%-------------------------
function v=up(x)

v= 50* x * (1-x) ;
end
%-----------------------
function v=uh(x,t)
a=0.01;
sum=0; N=50;

for n=1:N
   sum=sum+( 400* an(n)*exp( -a * (n*pi)^2 *t ) * sin(n*pi*x) );
end
v=sum;
end

%-----------------------
function v=an(n)
v=( (-1)^n-1)/(n^3*pi^3);
end
```

3.2.6.3 nma_math228b_plot.m

```
function nma_math228b_plot()
close all;

t=0.01;
D=1;

N=49;
h=1/(N+1);
p=1:N;
z=p*pi*(1/(N+1));
g= (D*t/h^2)*(cos(z)-1);
g=abs((1+g)./(1-g));


plot(p,g)
title(sprintf('Magnification factor as function of wave number\nC-N scheme for h=%f, t=0.01,
xlabel('p, wave number');
ylabel('g(wave number)');
```

```
%


close all;
h=0.02;
t=0.001;
D=1;
r=t*D/(2*h^2);
z=-pi/h:0.01:pi/h;
g= abs((1+2*r*cos(z*h)-2*r)./(1-2*r*cos(z*h)+2*r));

plot(z*h,g)
title(sprintf('Magnification factor, C-N scheme for h=0.02, t=0.001'))
xlabel('zeta(radians)');
ylabel('g(zeta)');
xlim([-pi,pi])
ylim([0,1.2]);
set(gca,'XTick',-pi:pi/2:pi)
set(gca,'XTickLabel',{'-pi/h','-pi/2h','0','pi/2h','pi/h'})


close all;
h=0.02;
t=0.001;
D=1;
r=t*D/(2*h^2);
z=-pi/h:0.01:pi/h;
g= abs(1-(4*D*t/h^2)*sin(z*h/2).^2);

plot(z*h,g)
title(sprintf('Magnification factor, FE scheme for h=0.02, t=0.001\ndelt*D/h^2=%3.3f',t*D/h^2))
xlabel('zeta(radians)');
ylabel('g(zeta)');
xlim([-pi,pi])
ylim([0,1.2]);
set(gca,'XTick',-pi:pi/2:pi)
%set(gca,'XTickLabel',{'-pi/h','-pi/2h','0','pi/2h','pi/h'})

end
```

3.3 HW 2

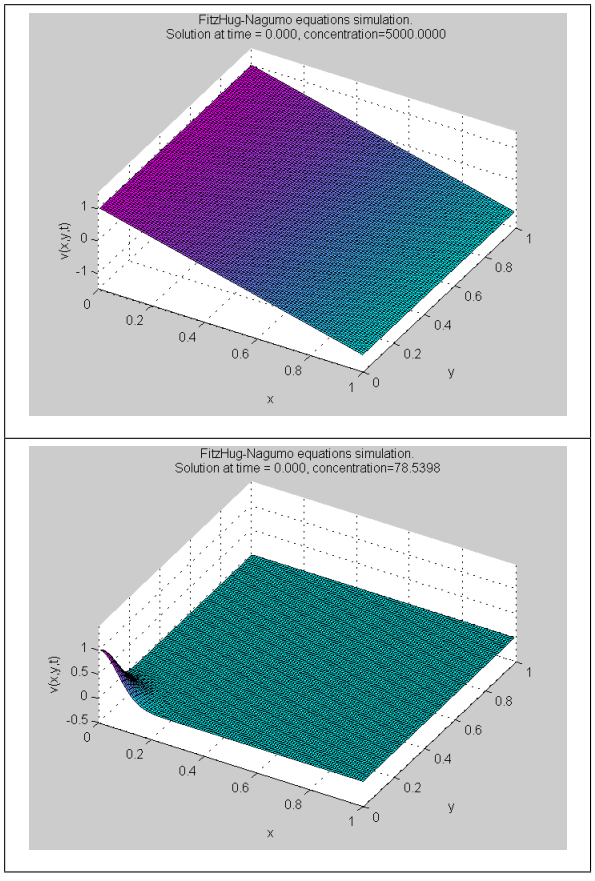
3.3.1 Animation of FitzHugh-Nagumo equations

The following are animated GIFs showing the solution to problem 3, parts (b) and (c). These will show only in the HTML version.

Assuming that $f(v) = (a - v)(v - 1)v$, the equations solved are the following

$$\frac{\partial v}{\partial t} = D\Delta v + f(v) - w + I$$
$$\frac{\partial w}{\partial t} = \epsilon\left(v - \gamma w\right)$$

Click on image to see the animation run, it will open in new window.



3.3.2 Problem1

Math 228B
Homework 2
Due Tuesday, February 15th

1. In class, we showed that the two-dimensional Peaceman-Rachford ADI scheme is unconditionally stable and second-order accurate in time. This can be thought of as either an approximate factorization or as a fractional step method. By adapting the fractional step idea to three-dimensions we get the scheme

$$\left(I - \frac{b\Delta t}{3}L_x\right)u^* = \left(I + \frac{b\Delta t}{3}L_y + \frac{b\Delta t}{3}L_z\right)u^n$$
$$\left(I - \frac{b\Delta t}{3}L_y\right)u^{**} = \left(I + \frac{b\Delta t}{3}L_x + \frac{b\Delta t}{3}L_z\right)u^*$$
$$\left(I - \frac{b\Delta t}{3}L_z\right)u^{n+1} = \left(I + \frac{b\Delta t}{3}L_x + \frac{b\Delta t}{3}L_y\right)u^{**}.$$

   (a) Use von Neumann analysis to show that this scheme is conditionally stable. This is an example of how certain desirable properties of a numerical scheme can be lost when using fractional stepping.

   (b) What temporal accuracy do you expect from this scheme? Explain.

Figure 3.37: Problem description

3.3.2.1 Part (a)

The diffusion PDE is given by

$$u_t - D\Delta u = 0$$

Where D is the diffusion constant. The ADI scheme in 3D [6] is given by

$$\left(I - \frac{D\Delta t}{3}L_x\right)u^* = \left(I + \frac{D\Delta t}{3}L_y + \frac{D\Delta t}{3}L_z\right)u^n \tag{1}$$
$$\left(I - \frac{D\Delta t}{3}L_y\right)u^{**} = \left(I + \frac{D\Delta t}{3}L_x + \frac{D\Delta t}{3}L_z\right)u^* \tag{2}$$
$$\left(I - \frac{D\Delta t}{3}L_z\right)u^{n+1} = \left(I + \frac{D\Delta t}{3}L_x + \frac{D\Delta t}{3}L_y\right)u^{**} \tag{3}$$

[6] Please see the appendix of this problem at the end of the HW report showing how these equations came about.

Where L_x, L_y, L_z are each the 1D Laplacian given by $\frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ \dots & \dots & \dots & \ddots & \dots & \dots \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & -2 \end{bmatrix}$

Assuming that the spatial frequencies in each of the three Cartesian directions (x, y, z) are given by ξ_x, ξ_y, ξ_z where $\frac{\pi}{h} \leq \xi_i \leq \frac{\pi}{h}$ and by setting $r = \frac{D\Delta t}{3h^2}$, $u^* = g^* e^{i\left(\xi_1 x + \xi_2 y + \xi_3 z\right)}$ and $u = e^{i\left(\xi_1 x + \xi_2 y + \xi_3 z\right)}$ and substituting these into Eq. (1) and dividing throughout by $e^{i\left(\xi_1 x + \xi_2 y + \xi_3 z\right)}$ gives the following

$$\begin{aligned} g^*\left(1 - r\left(e^{-i\xi_1 h} - 2 + e^{i\xi_1 h}\right)\right) &= 1 + r\left(e^{-i\xi_2 h} - 2 + e^{i\xi_2 h}\right) + r\left(e^{-i\xi_3 h} - 2 + e^{i\xi_3 h}\right) \\ g^* &= \frac{1 - 4r + r\left(e^{-i\xi_2 h} + e^{i\xi_2 h}\right) + r\left(e^{-i\xi_3 h} + e^{i\xi_3 h}\right)}{\left(1 + 2r - r\left(e^{i\xi_1 h} + e^{-i\xi_1 h}\right)\right)} \\ &= \frac{1 - 4r + 2r\cos(\xi_2 h) + 2r\cos(\xi_3 h)}{1 + 2r - 2r\cos(\xi_1 h)} \\ &= \frac{1 - 4r\left(\sin^2\left(\frac{\xi_2 h}{2}\right) + \sin^2\left(\frac{\xi_3 h}{2}\right)\right)}{1 + 4r\sin^2\left(\frac{\xi_1 h}{2}\right)} \end{aligned} \tag{4}$$

The last step above was obtained by the use of the relation $\cos A = 1 - 2\sin^2\left(\frac{A}{2}\right)$.

Applying the same method used above to Eq. (2), but now letting $u^* = g^* e^{i\left(\xi_1 x + \xi_2 y + \xi_3 z\right)}$ and $u^{**} = g^{**} e^{i\left(\xi_1 x + \xi_2 y + \xi_3 z\right)}$ and dividing throughout by $e^{i\left(\xi_1 x + \xi_2 y + \xi_3 z\right)}$ gives

$$\begin{aligned} g^{**}\left(1 - r\left(e^{-i\xi_2 h} - 2 + e^{i\xi_2 h}\right)\right) &= 1 + r\left(e^{-i\xi_1 h} - 2 + e^{i\xi_1 h}\right) + r\left(e^{-i\xi_3 h} - 2 + e^{i\xi_3 h}\right) \\ g^{**} &= \frac{1 - 4r + r\left(e^{-i\xi_1 h} + e^{i\xi_1 h}\right) + r\left(e^{-i\xi_3 h} + e^{i\xi_3 h}\right)}{1 + 2r - r\left(e^{i\xi_2 h} + e^{-i\xi_2 h}\right)} g^* \\ &= \frac{1 - 4r + 2r\cos(\xi_1 h) + 2r\cos(\xi_3 h)}{1 + 2r - 2r\cos(\xi_2 h)} g^* \\ &= \frac{1 - 4r\left(\sin^2\left(\frac{\xi_1 h}{2}\right) + \sin^2\left(\frac{\xi_3 h}{2}\right)\right)}{1 + 4r\sin^2\left(\frac{\xi_2 h}{2}\right)} g^* \end{aligned} \tag{5}$$

Again, applying the same method to Eq. (3), but now letting $u^{**} = g^{**} e^{i\left(\xi_1 x + \xi_2 y + \xi_3 z\right)}$ and

$u^{n+1} = ge^{i\left(\xi_1 x+\xi_2 y+\xi_3 z\right)}$ and dividing by $e^{i\left(\xi_1 x+\xi_2 y+\xi_3 z\right)}$ gives

$$\begin{aligned} g\left(1-r\left(e^{-i\xi_3 h}-2+e^{i\xi_3 h}\right)\right) &= 1+r\left(e^{-i\xi_1 h}-2+e^{i\xi_1 h}\right)+r\left(e^{-i\xi_2 h}-2+e^{i\xi_2 h}\right) \\ g &= \frac{1-4r+r\left(e^{-i\xi_1 h}+e^{i\xi_1 h}\right)+r\left(e^{-i\xi_2 h}+e^{i\xi_2 h}\right)}{1-r\left(e^{-i\xi_3 h}-2+e^{i\xi_3 h}\right)} g^{**} \\ &= \frac{1-4r+2r\cos(\xi_1 h)+2r\cos(\xi_2 h)}{1+2r-2r\cos(\xi_3 h)} g^{**} \\ &= \frac{1-4r\left(\sin^2\left(\frac{\xi_1 h}{2}\right)+\sin^2\left(\frac{\xi_2 h}{2}\right)\right)}{1+4r\sin^2\left(\frac{\xi_3 h}{2}\right)} g^{**} \end{aligned} \tag{6}$$

Substituting (4) into (5) and substituting the resulting expression into (6) gives the overall magnification factor for the ADI scheme:

$$g = \left[\frac{1-4r\left(\sin^2\left(\frac{\xi_1 h}{2}\right)+\sin^2\left(\frac{\xi_2 h}{2}\right)\right)}{1+4r\sin^2\left(\frac{\xi_3 h}{2}\right)}\right]\left[\frac{1-4r\left(\sin^2\left(\frac{\xi_1 h}{2}\right)+\sin^2\left(\frac{\xi_3 h}{2}\right)\right)}{1+4r\sin^2\left(\frac{\xi_2 h}{2}\right)}\right]\left[\frac{1-4r\left(\sin^2\left(\frac{\xi_2 h}{2}\right)+\sin^2\left(\frac{\xi_3 h}{2}\right)\right)}{1+4r\sin^2\left(\frac{\xi_1 h}{2}\right)}\right] \tag{7}$$

Letting $A \equiv \sin^2\left(\frac{\xi_1 h}{2}\right), B \equiv \sin^2\left(\frac{\xi_2 h}{2}\right), C \equiv \sin^2\left(\frac{\xi_1 h}{2}\right) = C$ in Eq. (7) results in

$$g(\xi_1, \xi_2, \xi_3) = \left(\frac{1-4r(A+B)}{1+4rC}\right)\left(\frac{1-4r(A+C)}{1+4rB}\right)\left(\frac{1-4r(B+C)}{1+4rA}\right) \tag{8}$$

The scheme is conditionally stable if $\left|g(\xi_1, \xi_2, \xi_3)\right| \leq 1$ for some value of r and $\left|g(\xi_1, \xi_2, \xi_3)\right| > 1$ for some other value of r (this is the same as using different values of Δt in place of r, since $r = \frac{D\Delta t}{3h^2}$ and h and D would be kept constant).

Now the scheme can be shown to be conditionally stable by letting $A = B = C = 1$ in Eq. (8) and then by finding one value of r which makes the magnification factor to become less than one and then by looking for another value of r which makes the magnification factor to becomes larger than one.

Therefore, when $A = B = C = 1$, Eq. (8) becomes

$$\left|g(\xi_1, \xi_2, \xi_3)\right| = \left(\frac{1-8r}{1+4r}\right)\left(\frac{1-8r}{1+4r}\right)\left(\frac{1-8r}{1+4r}\right) \tag{8A}$$

Now, putting $r = 2$ in the above gives $\left|g(\xi_1, \xi_2, \xi_3)\right| = 2.744 > 1$ implying that the scheme is unstable.

Putting $r = 0.5$ in Eq. (8A) gives $\left|g(\xi_1, \xi_2, \xi_3)\right| = 0.125 < 1$ implying that the scheme is stable.

Hence the scheme is conditionally stable, because by fixing h and D, it was possible to find a time step Δt which made some mode become unstable. If one mode is unstable, the overall scheme is also unstable. This result shows that the above given ADI scheme for 3D is conditionally stable.

3.3.2.2 Part (b)

Expectation: Temporal accuracy is expected to be $O(\Delta t)$ since at each 1/3 time step there is one implicit step compared to two explicit steps. Starting from the main equations shown in part (a)

$$\overbrace{(I - rL_x)u^*}^{\textit{implicit}\ \text{(backward Euler)}} = \overbrace{\Big(I + rL_y + rL_z\Big)}^{\text{explicit (2 forward Euler)}} u^n \tag{1}$$

$$\Big(I - rL_y\Big)u^{**} = (I + rL_x + rL_z)u^* \tag{2}$$

$$(I - rL_z)u^{n+1} = \Big(I + rL_x + rL_y\Big)u^{**} \tag{3}$$

There will be an $O(\Delta t)$ error resulting from the application of Euler approximation to each of the terms in each equation above. One of the implicit errors will cancel exactly with one of the errors from the explicit part of the equation (due to sign difference), leaving an extra $O(\Delta t)$ error after each third step. Hence at the completion of one full time step, the temporal error will be $3O(\Delta t)$ or $O(\Delta t)$.

Explanation: The derivation below follows the method explained in class for the 2D ADI case, but being applied to the 3D case. Starting by pre-multiplying Eq. (1) by the operator $(I + rL_x + rL_z)$ gives

$$(I + rL_x + rL_z)(I - rL_x)u^* = (I + rL_x + rL_z)\Big(I + rL_y + rL_z\Big)u^n$$

But since $(I + rL_x + rL_z)$ commutes[7] with $(I - rL_x)$, then the two terms in the LHS of the above equation can be interchanged giving

$$(I - rL_x)\ \overbrace{(I + rL_x + rL_z)u^*}^{\text{now replace this from (2)}} = (I + rL_x + rL_z)\Big(I + rL_y + rL_z\Big)u^n$$

Replacing the term marked above by its LHS value from Eq. (2) yields

$$(I - rL_x)\Big(I - rL_y\Big)u^{**} = (I + rL_x + rL_z)\Big(I + rL_y + rL_z\Big)u^n$$

Pre-multiplying the above by the operator $\Big(I + rL_x + rL_y\Big)$ gives

$$\Big(I + rL_x + rL_y\Big)(I - rL_x)\Big(I - rL_y\Big)u^{**} = \Big(I + rL_x + rL_y\Big)(I + rL_x + rL_z)\Big(I + rL_y + rL_z\Big)u^n$$

But since $\Big(I + rL_x + rL_y\Big)$ commutes with $(I - rL_x)\Big(I - rL_y\Big)$ the above can be written as

$$(I - rL_x)\Big(I - rL_y\Big)\overbrace{\Big(I + rL_x + rL_y\Big)u^{**}}^{\text{replace this from (3)}} = \Big(I + rL_x + rL_y\Big)(I + rL_x + rL_z)\Big(I + rL_y + rL_z\Big)u^n$$

Replacing the term marked above by its LHS value from Eq. (3) gives

$$(I - rL_x)\Big(I - rL_y\Big)(I - rL_z)u^{n+1} = \Big(I + rL_x + rL_y\Big)(I + rL_x + rL_z)\Big(I + rL_y + rL_z\Big)u^n$$

[7] To show these operators commute, similar argument can be made as was done for the 2D case in class, which is by saying that each operator L_x, L_y, L_z on its own commutes with the other 2, hence the result will follow.

Expanding all terms by multiplying all operators and simplifying the result and using $L = L_x + L_y + L_z$ gives the following

$$\begin{aligned}\Big(I - rL + r^2\Big(L_xL_z + L_yL_z\Big) + r^2L_xL_y - r^3L_xL_yL_z\Big)u^{n+1} = \\ \Big(I + rL + rL + 3r^2L_xL_y + 3r^2L_xL_z + 3r^2L_xL_y + 3r^2L_yL_z\Big)u^n + (H.O.T.)\end{aligned} \tag{4}$$

Where H.O.T. are terms from operators of order 2 and higher. These terms produce errors of order $O\Big(\Delta t^2\Big)$, $O\Big(\Delta t^3\Big)$ and higher. Moving all these terms to the RHS simplifies Eq. (4) to the following

$$\begin{aligned}(I - rL)u^{n+1} &= (I + rL + rL)u^n + O\Big(\Delta t^2\Big) + O\Big(\Delta t^3\Big) + \cdots \\ u^{n+1} - u^n &= rLu^{n+1} + 2rLu^n + O\Big(\Delta t^2\Big) + O\Big(\Delta t^3\Big) + \cdots\end{aligned}$$

Since $r = \frac{D\Delta t}{3}$ the above becomes

$$u^{n+1} - u^n = \frac{D\Delta t}{3}Lu^{n+1} + 2\frac{D\Delta t}{3}Lu^n + O\Big(\Delta t^2\Big) + O\Big(\Delta t^3\Big) + \cdots$$

Dividing the above equation by Δt gives

$$\frac{u^{n+1} - u^n}{\Delta t} = \frac{D}{3}Lu^{n+1} + \frac{2D}{3}Lu^n + O(\Delta t) + O\Big(\Delta t^2\Big) + \cdots$$

Now $\boxed{\text{adding } \frac{D}{6}Lu^{n+1} \text{ and subtracting } \frac{D}{6}Lu^{n+1}}$ and $\boxed{\text{subtracting } \frac{D}{6}Lu^{n} \text{ and adding } \frac{D}{6}Lu^{n}}$ from the RHS of the above equation gives

$$\begin{aligned}\frac{u^{n+1} - u^n}{\Delta t} &= \frac{D}{3}Lu^{n+1} + \frac{D}{6}Lu^{n+1} + \frac{2D}{3}Lu^n - \frac{D}{6}Lu^n + \frac{D}{6}Lu^n - \frac{D}{6}Lu^{n+1} + O(\Delta t) + O\Big(\Delta t^2\Big) + \cdots \\ \overbrace{\frac{u^{n+1} - u^n}{\Delta t}}^{C\text{–}N} &= \frac{1}{2}\Big(Lu^{n+1} + Lu^n\Big) + \frac{1}{6}\Big(Lu^n - Lu^{n+1}\Big) + O(\Delta t) + O\Big(\Delta t^2\Big) + \cdots\end{aligned}$$

The C-N scheme is known to be $O\Big(\Delta t^2 + h^2\Big)$. Multiplying the term $\frac{1}{6}\Big(Lu^n - Lu^{n+1}\Big)$ by $\frac{\Delta t}{\Delta t}$ in the above yields

$$u_t = \frac{1}{2}\Delta u + \overbrace{O\Big(\Delta t^2\Big) + O\Big(h^2\Big)}^{\text{from C-N part}} + \frac{\Delta t}{6}L\left(\frac{u^n - u^{n+1}}{\Delta t}\right) + O(\Delta t) + O\Big(\Delta t^2\Big) + \cdots$$

Taking the limits $\Delta t \to 0$ results in

$$\begin{aligned}u_t &= \frac{1}{2}\Delta u + \overbrace{O\Big(\Delta t^2\Big) + O\Big(h^2\Big)}^{\text{from C-N part}} + \overbrace{\frac{\Delta t}{6}\frac{\partial^7}{\partial^2 x \partial^2 y \partial^2 z \partial t} + O(\Delta t)}^{\textit{still } O(\Delta t)} + O\Big(\Delta t^2\Big) + \cdots \\ &= \frac{1}{2}\Delta u + \overbrace{O\Big(\Delta t^2\Big) + O\Big(h^2\Big)}^{\text{from C-N part}} + O(\Delta t) + O\Big(\Delta t^2\Big) + \cdots \\ &= \frac{1}{2}\Delta u + \overbrace{O\Big(h^2\Big) + O(\Delta t)}^{} + O\Big(\Delta t^2\Big) + \cdots\end{aligned}$$

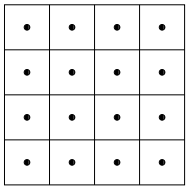
Since in the above, the dominant temporal error term is $O(\Delta t)$ the scheme is a first order in time accurate. It is also a second order in space accurate.

3.3.3 Problem 2

2. Consider

$$u_t = 0.1\,\Delta u \text{ on } \Omega = (0,1)\times(0,1)$$
$$\frac{\partial u}{\partial \vec{n}} = 0 \text{ on } \partial\Omega$$
$$u(x,y,0) = \exp\left(-10((x-0.4)^2 + (y-0.4)^2)\right)$$

(a) Write a program to solve this PDE using the Peaceman-Rachford ADI scheme on a cell-centered grid. Use a direct solver for the tridiagonal systems. In a cell-centered discretization the solution is stored at the grid points $(x_i, y_j) = (h(i-0.5), h(j-0.5))$ for $i, j = 1 \ldots N$ and $h = 1/N$. This discretization is natural for handling Neumann boundary conditions, and it is often used to discretize conservation laws. At the grid points adjacent to the boundary, the one-dimensional discrete Laplacian for homogeneous Neumann boundary conditions is



$$u_{xx}(x_1) \approx \frac{-u_1 + u_2}{h^2}.$$

(b) Perform a refinement study to show that your numerical solution is second-order accurate in space and time (refine time and space simultaneously using $\Delta t = h$) at time $t = 1$.

(c) Show that the spatial integral of the solution to the PDE does not change in time. That is

$$\frac{d}{dt}\int_\Omega u\,dV = 0.$$

(d) Show that the solution to the discrete equations satisfies the discrete conservation property

$$\sum_{i,j} u^n_{i,j} = \sum_{i,j} u^0_{i,j}$$

for all n. Demonstrate this property with your code.

Figure 3.38: Problem description

The following diagram shows the discretization using cell-centered scheme for the case of $N = 4$. The center of the cells moves closer to the physical boundaries of the unit square as N becomes larger.

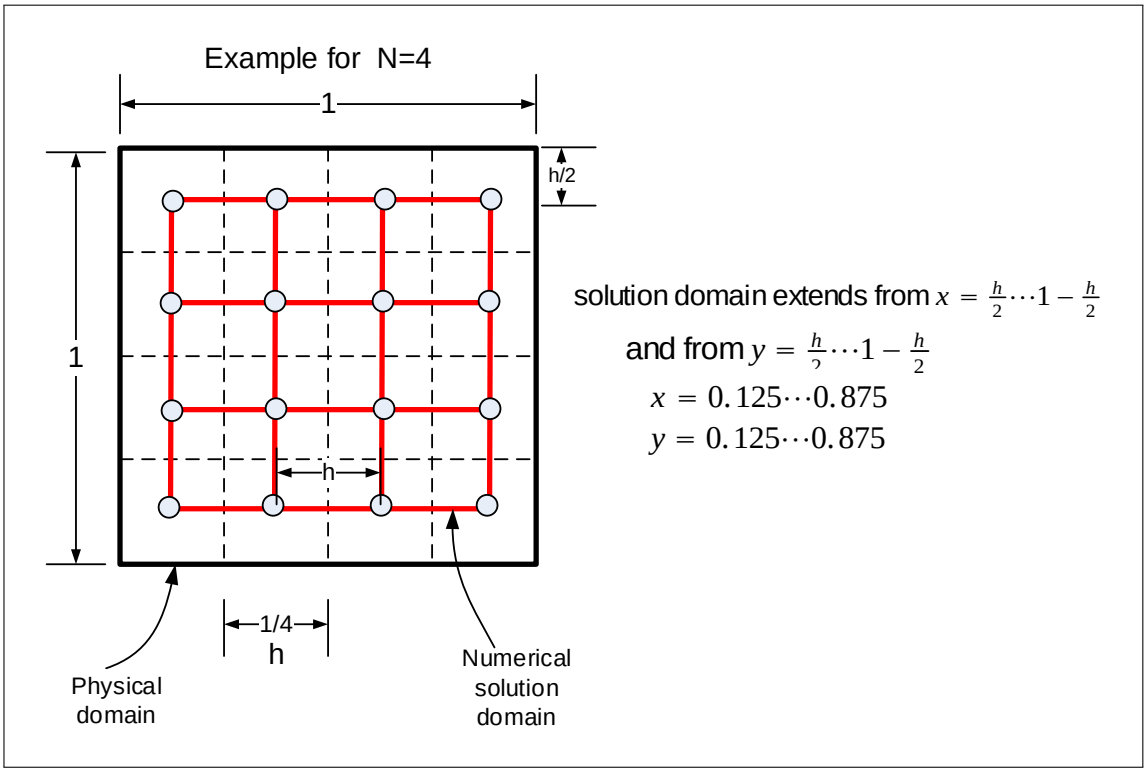


Figure 3.39: Grid used

The physical domain is always the unit square $x = 0 \cdots 1$, $y = 0 \cdots 1$, but the discrete solution domain is the one at corners of the red grid above. A small example is used below to help determine the layout of the operator used in the direct solver. The 2D ADI scheme for the diffusion problem is

$$\left(I - \frac{D\Delta t}{2} L_x\right)\mathbf{u}^* = \left(I + \frac{D\Delta t}{2} L_y\right)\mathbf{u}^n \tag{1A}$$
$$\left(I - \frac{D\Delta t}{2} L_y\right)\mathbf{u}^{n+1} = \left(I + \frac{D\Delta t}{2} L_x\right)\mathbf{u}^*$$

Where $L_x = L_y = \frac{1}{h^2}\begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ \cdots & \cdots & \cdots & \ddots & \cdots & \cdots \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$ for the case of homogenous Neumann boundary conditions. The solution given below uses an overall L operator which is used by the direct solver. Another approach would have been to use the above L_x, L_y operator, and iterate over each each row and column applying the direct solver each time.

The following derives the overall L operator used. Eq. (1A) can be written as

$$u^*_{ij} - \frac{D\Delta t}{2}\frac{u^*_{i-1,j} - 2u^*_{ij} + u^*_{i+1,j}}{h^2} = u^n_{ij} + \frac{D\Delta t}{2}\frac{u^n_{i,j-1} - 2u^n_{ij} + u^n_{i,j+1}}{h^2}$$
$$u^{n+1}_{ij} - \frac{D\Delta t}{2}\frac{u^{n+1}_{i,j-1} - 2u^{n+1}_{ij} + u^{n+1}_{i,j+1}}{h^2} = u^{n*}_{ij} + \frac{D\Delta t}{2}\frac{u^*_{i-1,j} - 2u^*_{ij} + u^*_{i+1,j}}{h^2}$$

letting $r = \frac{D\Delta t}{2h^2}$ and simplifying the above gives

$$u^*_{ij}(1+2r) - ru^*_{i-1,j} - ru^*_{i+1,j} = u^n_{ij}(1-2r) + ru^n_{i,j-1} + ru^n_{i,j+1} \tag{1}$$
$$u^{n+1}_{ij}(1+2r) - ru^{n+1}_{i,j-1} - ru^{n+1}_{i,j+1} = u^*_{ij}(1-2r) + ru^*_{i-1,j} + ru^*_{i+1,j} \tag{2}$$

The above finite difference equations are applied at all the grid points, except for those for the rows and columns at the boundaries. In order to determine the equations to use for the boundary grid points, the approximation $L_x \approx \frac{-u_1+u_2}{h^2}$ is used. Similar one is used for L_y. The result of using the above approximation is the following finite difference equations used for the boundary grid points

$$u^*_{ij} - \frac{D\Delta t}{2}\frac{-u^*_{ij} + u^*_{i+1,j}}{h^2} = u^n_{ij} + \frac{D\Delta t}{2}\frac{-u^n_{ij} + u^n_{i,j+1}}{h^2}$$
$$u^{n+1}_{ij} - \frac{D\Delta t}{2}\frac{-u^{n+1}_{ij} + u^{n+1}_{i,j+1}}{h^2} = u^{n*}_{ij} + \frac{D\Delta t}{2}\frac{-u^*_{ij} + u^*_{i+1,j}}{h^2}$$

Simplifying the above gives

$$u^*_{ij}(1+r) - ru^*_{i+1,j} = u^n_{ij}(1-r) + ru^n_{i,j+1} \tag{1A}$$
$$u^{n+1}_{ij}(1+r) - ru^{n+1}_{i,j+1} = u^*_{ij}(1-r) + ru^*_{i+1,j} \tag{2A}$$

To help obtain L, a small example is used to help see the structure of the matrix. This small example exhibits all the needed information to generate the pattern for L from. Using $n_x = n_y = 4$, the grid becomes

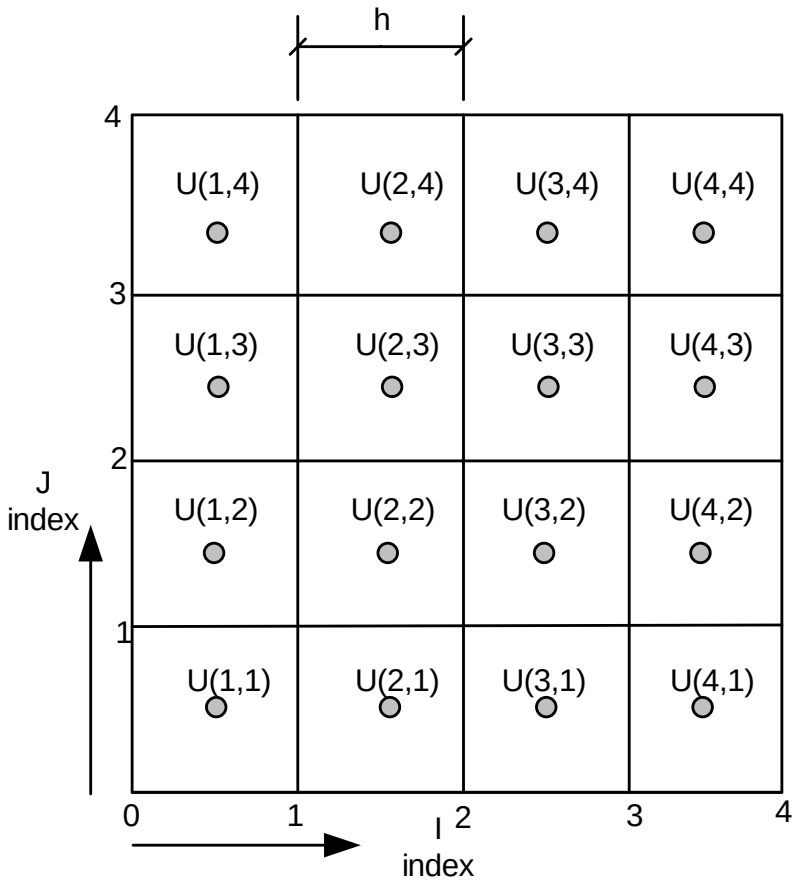


Figure 3.40: Updated Grid

Eq. (1) and (1A) are now written for all the nodes resulting in the following 16 equations

$$
\begin{aligned}
u^*_{11}(1+r) - ru^*_{21} &= u^n_{11}(1-r) + ru^n_{12} \\
u^*_{21}(1+2r) - ru^*_{1,1} - ru^*_{3,1} &= u^n_{21}(1-r) + ru^n_{22} \\
u^*_{31}(1+2r) - ru^*_{2,1} - ru^*_{4,1} &= u^n_{31}(1-r) + ru^n_{32} \\
u^*_{41}(1+r) - ru^*_{31} &= u^n_{41}(1-r) + ru^n_{42} \\
u^*_{12}(1+r) - ru^*_{22} &= u^n_{12}(1-2r) + ru^n_{1,1} + ru^n_{1,3} \\
u^*_{22}(1+2r) - ru^*_{1,2} - ru^*_{3,2} &= u^n_{22}(1-2r) + ru^n_{2,1} + ru^n_{2,3} \\
u^*_{32}(1+2r) - ru^*_{2,2} - ru^*_{4,2} &= u^n_{32}(1-2r) + ru^n_{3,1} + ru^n_{3,3} \\
u^*_{42}(1+r) - ru^*_{32} &= u^n_{42}(1-2r) + ru^n_{4,1} + ru^n_{4,3} \\
u^*_{13}(1+r) - ru^*_{23} &= u^n_{13}(1-2r) + ru^n_{1,2} + ru^n_{1,4} \\
u^*_{23}(1+2r) - ru^*_{1,3} - ru^*_{3,3} &= u^n_{23}(1-2r) + ru^n_{2,2} + ru^n_{2,4} \\
u^*_{33}(1+2r) - ru^*_{2,3} - ru^*_{4,3} &= u^n_{33}(1-2r) + ru^n_{3,2} + ru^n_{3,4} \\
u^*_{43}(1+r) - ru^*_{33} &= u^n_{43}(1-2r) + ru^n_{4,2} + ru^n_{4,4} \\
u^*_{14}(1+r) - ru^*_{24} &= u^n_{14}(1-r) + ru^n_{13} \\
u^*_{24}(1+2r) - ru^*_{1,4} - ru^*_{3,4} &= u^n_{24}(1-r) + ru^n_{23} \\
u^*_{34}(1+2r) - ru^*_{2,4} - ru^*_{4,4} &= u^n_{34}(1-r) + ru^n_{33} \\
u^*_{44}(1+r) - ru^*_{34} &= u^n_{44}(1-r) + ru^n_{43}
\end{aligned}
$$

In matrix form, the above gives $Au^* = b$ which is then used to solve for u^*. The matrix A is now written out. To save space and to allow the matrix to fit on the page, the following terms are used

$$
\begin{aligned}
r &= \frac{D\Delta t}{2h^2} \\
\alpha &\equiv 1 + r \\
\beta &\equiv 1 + 2r \\
\gamma &\equiv 1 - r \\
\theta &\equiv 1 - 2r
\end{aligned}
$$

$$
\overbrace{\begin{bmatrix}
\alpha & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -r & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \alpha & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -r & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha & -r & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \alpha & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha & -r & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \alpha
\end{bmatrix}}^{\mathbf{A}}
\overbrace{\begin{bmatrix} u^*_{11} \\ u^*_{21} \\ u^*_{31} \\ u^*_{41} \\ u^*_{12} \\ u^*_{22} \\ u^*_{32} \\ u^*_{4,2} \\ u^*_{13} \\ u^*_{2,3} \\ u^*_{3,3} \\ u^*_{4,3} \\ u^*_{14} \\ u^*_{2,4} \\ u^*_{3,4} \\ u^*_{4,4} \end{bmatrix}}^{\mathbf{x}} =
$$

$$
\overbrace{\begin{bmatrix}
\gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma
\end{bmatrix}}^{\mathbf{b}}
\begin{bmatrix} u^n_{11} \\ u^n_{21} \\ u^n_{31} \\ u^n_{41} \\ u^n_{12} \\ u^n_{22} \\ u^n_{32} \\ u^n_{4,2} \\ u^n_{13} \\ u^n_{2,3} \\ u^n_{3,3} \\ u^n_{4,3} \\ u^n_{14} \\ u^n_{2,4} \\ u^n_{3,4} \\ u^n_{4,4} \end{bmatrix} \tag{7}
$$

A sparse direct solver can now be used to solve for u^*.

Starting the second ADI step to find u^{n+1}, the process is similar to the one shown above, but the equations are written column-wise instead of row-wise as was the case earlier. For

non-boundary grid points the following equation is used

$$u_{ij}^{n+1}(1+2r) - ru_{i,j-1}^{n+1} - ru_{i,j+1}^{n+1} = u_{ij}^{*}(1-2r) + ru_{i-1,j}^{*} + ru_{i+1,j}^{*}$$

And for the boundary grid points the following equation is used

$$u_{ij}^{n+1}(1+r) - ru_{i,j+1}^{n+1} = u_{ij}^{*}(1-r) + ru_{i+1,j}^{*}$$

Applying the above to each grid point results in the the following 16 equations

$$\begin{aligned}
u_{11}^{n+1}(1+r) - ru_{12}^{n+1} &= u_{11}^{*}(1-r) + ru_{21}^{*} \\
u_{21}^{n+1}(1+r) - ru_{22}^{n+1} &= u_{21}^{*}(1-2r) + ru_{1,1}^{n} + ru_{3,1}^{n} \\
u_{31}^{n+1}(1+r) - ru_{32}^{n+1} &= u_{31}^{*}(1-2r) + ru_{2,1}^{n} + ru_{4,1}^{n} \\
u_{41}^{n+1}(1+r) - ru_{42}^{n+1} &= u_{41}^{*}(1-r) + ru_{31}^{*} \\
u_{12}^{n+1}(1+2r) - ru_{1,1}^{n+1} - ru_{1,3}^{n+1} &= u_{12}^{*}(1-r) + ru_{22}^{*} \\
u_{22}^{n+1}(1+2r) - ru_{2,1}^{n+1} - ru_{2,3}^{n+1} &= u_{22}^{*}(1-2r) + ru_{1,2}^{n} + ru_{3,2}^{n} \\
u_{32}^{*}(1+2r) - ru_{3,1}^{*} - ru_{3,3}^{*} &= u_{32}^{n}(1-2r) + ru_{2,2}^{n} + ru_{4,2}^{n} \\
u_{42}^{*}(1+2r) - ru_{4,1}^{*} - ru_{4,3}^{*} &= u_{42}^{n}(1-r) + ru_{32}^{n} \\
u_{13}^{*}(1+2r) - ru_{1,2}^{*} - ru_{1,4}^{*} &= u_{13}^{n}(1-r) + ru_{23}^{n} \\
u_{23}^{*}(1+2r) - ru_{2,2}^{*} - ru_{2,4}^{*} &= u_{23}^{n}(1-2r) + ru_{1,3}^{n} + ru_{3,3}^{n} \\
u_{33}^{*}(1+2r) - ru_{3,2}^{*} - ru_{3,4}^{*} &= u_{33}^{n}(1-2r) + ru_{2,3}^{n} + ru_{4,3}^{n} \\
u_{43}^{*}(1+2r) - ru_{4,2}^{*} - ru_{4,4}^{*} &= u_{43}^{n}(1-r) + ru_{33}^{n} \\
u_{14}^{*}(1+r) - ru_{13}^{*} &= u_{14}^{n}(1-r) + ru_{24}^{n} \\
u_{24}^{*}(1+r) - ru_{23}^{*} &= u_{24}^{n}(1-2r) + ru_{1,4}^{n} + ru_{3,4}^{n} \\
u_{34}^{*}(1+r) - ru_{33}^{*} &= u_{34}^{n}(1-2r) + ru_{2,4}^{n} + ru_{4,4}^{n} \\
u_{44}^{*}(1+r) - ru_{43}^{*} &= u_{44}^{n}(1-r) + ru_{34}^{n}
\end{aligned}$$

The above equations are now written as $Au = b$ but the unknowns are listed column-wise in

order to keep the tridiagonal form. The resulting matrix A is the following

$$
\overbrace{\begin{bmatrix}
\alpha & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -r & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \alpha & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -r & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha & -r & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \alpha & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha & -r & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \beta & -r \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -r & \alpha
\end{bmatrix}}^{\mathbf{A}}
\overbrace{\begin{bmatrix}
u_{11}^{n+1} \\ u_{12}^{n+1} \\ u_{13}^{n+1} \\ u_{14}^{n+1} \\ u_{21}^{n+1} \\ u_{22}^{n+1} \\ u_{23}^{n+1} \\ u_{2,4}^{n+1} \\ u_{31}^{n+1} \\ u_{32}^{n+1} \\ u_{3,3}^{n+1} \\ u_{3,4}^{n+1} \\ u_{41}^{n+1} \\ u_{4,2}^{n+1} \\ u_{4,3}^{n+1} \\ u_{4,4}^{n+1}
\end{bmatrix}}^{\mathbf{x}} =
$$

$$
\overbrace{\begin{bmatrix}
\gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \gamma & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \theta & 0 & 0 & 0 & r \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & 0 & \gamma
\end{bmatrix}}^{\mathbf{b}}
\begin{bmatrix}
u_{11}^{*} \\ u_{12}^{*} \\ u_{13}^{*} \\ u_{14}^{*} \\ u_{21}^{*} \\ u_{22}^{*} \\ u_{23}^{*} \\ u_{2,4}^{*} \\ u_{31}^{*} \\ u_{32}^{*} \\ u_{3,3}^{*} \\ u_{3,4}^{*} \\ u_{41}^{*} \\ u_{4,2}^{*} \\ u_{4,3}^{*} \\ u_{4,4}^{*}
\end{bmatrix} \tag{7}
$$

Now u^{n+1} is solved for using a direct sparse solver. The above 2 steps are repeated for each one time step. One can see that the A matrix is the same for both solving $Au^* = b$ and

$Au^{n+1} = b$. Therefore, in the implementation only one A and one B matrix was allocated initially and used for solving for u^* and u^{n+1}. Both matrices (A and B) are created as sparse matrices to save storage. The A matrix represent the implicit part of the scheme, while the B matrix for the explicit part.

Since the edges of the domain are insulated, no concentration will diffuse to the outside. Therefore the result of diffusion will be that the concentration will diffuse internally and will spread out. Therefore, at steady state as $t \to \infty$ the solution is known and given by

$$u\big(x,y,\infty\big) = \int_{h/2}^{1-\frac{h}{2}} \int_{h/2}^{1-\frac{h}{2}} u\big(x,y,0\big)dxdy$$

The following plot shows the solution at $t = 1$ second with the steady state solution displayed as the blue horizontal flat surface superimposed on the same plot. The steady state solution is what would result if run time was made to be very long.

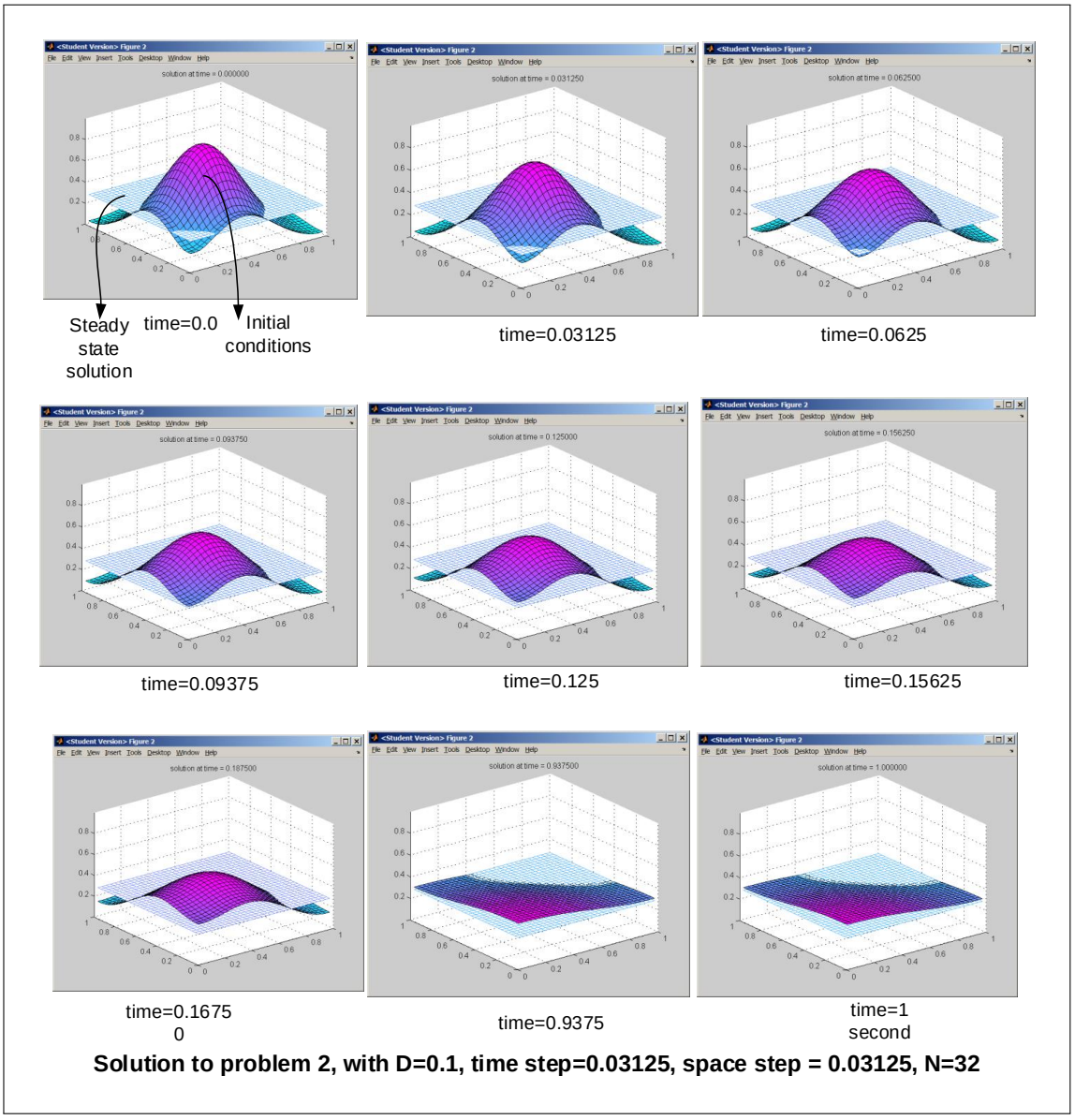


Figure 3.41: steady state solution

To verify that the numerical solution converges to the steady state solution, the plot below was generate which represents the solution of the above problem taken at $t = 4$ seconds. The gap in the diagram below is the difference between the steady state solution and the solution at $t = 4$ seconds. This gap became smaller the longer the time to run is made (keeping everything else fixed).

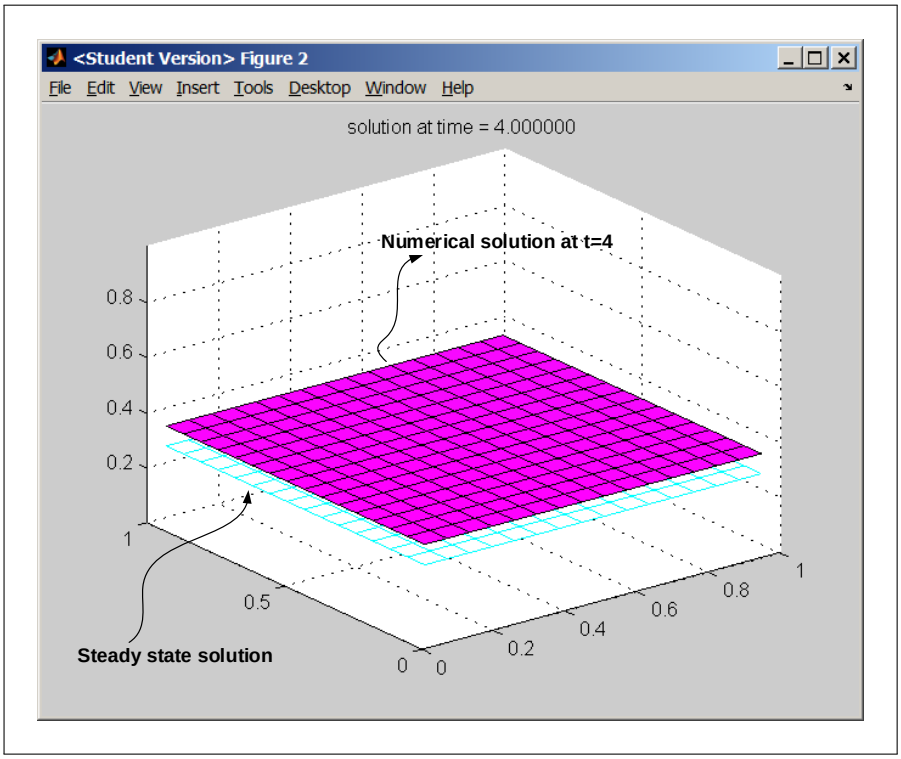


Figure 3.42: steady state solution

3.3.3.1 Part(b)

Refinement study was carried out to show that the 2D ADI scheme is a second order accurate in time and in space. The method used successive errors between numerical solutions. The algorithm of the refinement study is given below at the end of this part of the problem.

Recalling that In HW1, the spatial grid was divided by half each time. However, in this problem, since cell centered grid is used, h and Δt were divided by 3 each time. This was done so that the new grid will contain some grid locations that are still aligned in the same physical location as the previous step. The error between both solutions is obtained by taking the difference of only these points that are aligned. These points will be the grid point of the coarse grid. The following diagram illustrate this for the case of $n = 3$ and $n = 9$

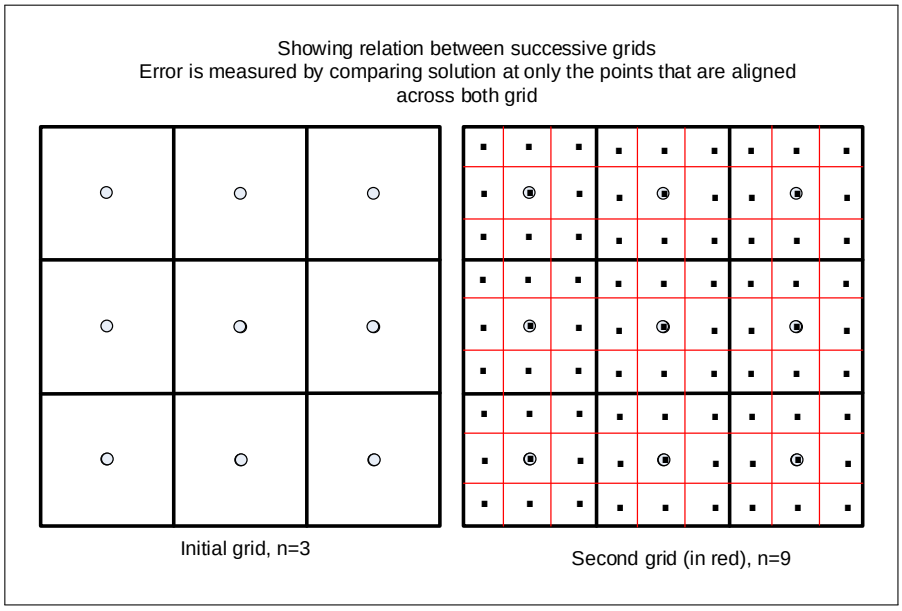


Figure 3.43: case of $n = 3$ and $n = 9$

The result of the refinement study shows second order accuracy as the error ratio came out to be 9.

Below is the result obtained. In addition to the ratio table, it can be seen that the slope of the line in the log plot is 2, implying the scheme is second order accurate.

Refinement study result, HW 2, problem 2, showing second order accuracy

#	N	delt	h	|u|	mean(u)	|e|	**ratio**	|u steady|
2	9	0.11111	0.111111	0.293146	0.290318	1.19490e-002	**1.000000**	0.289080
3	27	0.03704	0.037037	0.293146	0.289369	1.23779e-003	**9.653448**	0.289080
4	81	0.01235	0.012346	0.293146	0.289262	1.36668e-004	**9.056912**	0.289080
5	243	0.00412	0.004115	0.293146	0.289251	1.51752e-005	**9.006018**	0.289080
6	729	0.00137	0.001372	0.293146	0.289249	1.68601e-006	**9.000673**	0.289080

EDU>>

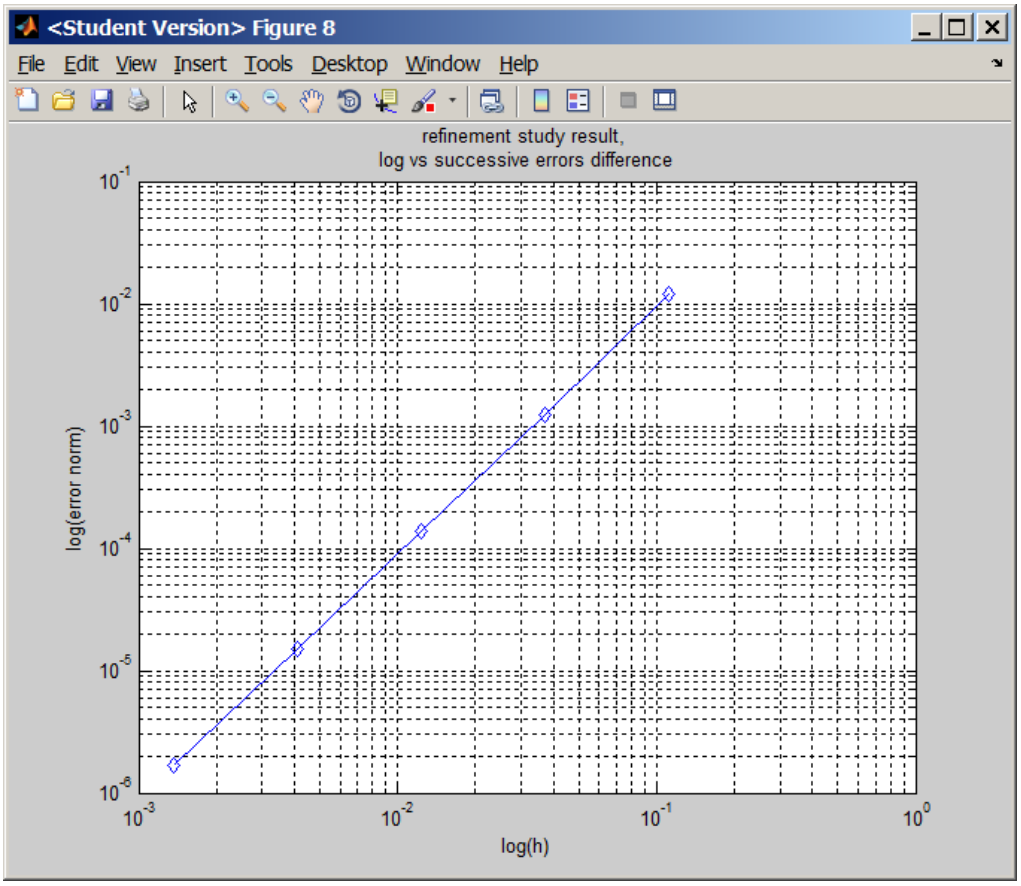


Figure 3.44: refinement study plot

3.3.3.2 Refinement algorithm

The following is the general outline of the algorithm used in the refinement study. The important part was to make sure when finding the error between the current and last solution, is to use the same physical locations that are aligned between both grids, and to use the coarse grid spacing when determining the grid norm of the error grid.

```
last_error    = 0
h             = 1/3
```

```
last_h       = h
delt         = h
last_u       = Solve_2D_ADI(h,delt)

LOOP
      h    = h/3
      delt = h

      current_u = Solve_2D_ADI(h,delt)

      -- now extract from current_u only locations that aligned with last_u grid
      current_u_mapped = extracted_u(last_u)

      error = last_h *  norm(last_u - current_u_mapped,2)

      ratio = last_error/error

      last_h = h
      last_error = error

      loop_counter++

      IF loop_counter > some_maximum THEN -- normally 5,6 iterations is enough
         EXIT LOOP
      END IF
END LOOP
```

3.3.3.3 Part(c)

The spatial integral represents the total concentration in the domain. Since the boundary are insulated, matter will only diffuse internally and no loss will occur to the outside. Hence, from the conservation of mass principle, initial concentration will remain the same, but will spread out to the mean in space. Therefore, it is known physically total concentration will not change with time

$$\frac{d}{dt}\int_{\Omega} u\big(x,y,t\big)dA = 0$$

The problem is asking to show this mathematically.

Since $u_t = D\Delta u$, then

$$\begin{aligned} I &= \frac{d}{dt}\int_{\Omega} u\big(x,y,t\big)dA \\ &= D\int_{\Omega} \Delta u \; dA \end{aligned}$$

But $\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$, hence

$$\left(\frac{1}{D}\right) I = \int_{y=0}^{1}\int_{x=0}^{1} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\, dxdy$$

To show that $I = 0$, the above is written as

$$\begin{aligned}\left(\frac{1}{D}\right) I &= \int_{y=0}^{1}\int_{x=0}^{1} \frac{\partial^2 u}{\partial x^2} dxdy + \int_{y=0}^{1}\int_{x=0}^{1} \frac{\partial^2 u}{\partial y^2}\, dxdy \\ &= \int_{y=0}^{1}\left(\int_{x=0}^{1} \frac{\partial^2 u}{\partial x^2} dx\right) dy + \int_{x=0}^{1}\left(\int_{y=0}^{1} \frac{\partial^2 u}{\partial y^2}\, dy\right) dx \end{aligned} \tag{1}$$

By applying the fundamental theory of calculus (or using integration by parts) results in

$$\int_{x=0}^{1} \frac{\partial^2 u}{\partial x^2} dx = \left.\frac{\partial u}{\partial x}\right|_{x=1} - \left.\frac{\partial u}{\partial x}\right|_{x=0}$$

However $\left.\frac{\partial u}{\partial x}\right|_{x=1}$ is the normal derivative at the right boundary, and $\left.\frac{\partial u}{\partial x}\right|_{x=0}$ is the normal derivative at the left boundaries. These are both zero due to the homogenous Neumann boundary conditions given in the problem statement. therefore

$$\int_{x=0}^{1} \frac{\partial^2 u}{\partial x^2} dx = 0 \tag{2}$$

Similar argument shows that

$$\int_{y=0}^{1} \frac{\partial^2 u}{\partial y^2}\, dy = 0 \tag{3}$$

Substituting Eqs. (2) and (3) into (1) gives

$$\left(\frac{1}{D}\right) I = 0$$

Therefore

$$\frac{d}{dt}\int_{\Omega} u\big(x, y, t\big) dA = 0$$

3.3.3.4 Part(d)

The finite difference equations for the 2D ADI scheme is given by

$$\left(I - \frac{D\Delta t}{2} L_x\right)\mathbf{u}^* = \left(I + \frac{D\Delta t}{2} L_y\right)\mathbf{u}^n \tag{1}$$

$$\left(I - \frac{D\Delta t}{2} L_y\right)\mathbf{u}^{n+1} = \left(I + \frac{D\Delta t}{2} L_x\right)\mathbf{u}^* \tag{2}$$

Summing the equations over all entries in the 2D solution domain gives

$$\sum_i \sum_j \left(I - \frac{D\Delta t}{2} L_x\right) u^* = \sum_j \sum_i \left(I + \frac{D\Delta t}{2} L_y\right) u^n \tag{1A}$$

$$\sum_j \sum_i \left(I - \frac{D\Delta t}{2} L_y\right) u^{n+1} = \sum_i \sum_j \left(I + \frac{D\Delta t}{2} L_x\right) u^* \tag{2A}$$

In the above i represents the row number and j represents the column number of the solution grid u. The above two equations can be rewritten as

$$\sum_i \sum_j u^*_{ij} - \frac{D\Delta t}{2} \sum_i \sum_j L_x u^*_{ij} = \sum_j \sum_i u^n_{ij} + \frac{D\Delta t}{2} \sum_j \sum_i L_y u^n_{ij} \tag{1B}$$

$$\sum_j \sum_i u^{n+1}_{ij} - \frac{D\Delta t}{2} \sum_j \sum_i L_y u^{n+1}_{ij} = \sum_i \sum_j u^*_{ij} + \frac{D\Delta t}{2} \sum_i \sum_j L_x u^*_{ij} \tag{2B}$$

Looking at the term $\sum_i \sum_j L_x u^*_{ij}$ from Eq. (1B) and rewriting this as follows

$$\sum_i \sum_j L_x u^*_{ij} = \sum_i \left(\sum_j L_x u^*_{ij} \right)$$

$$= \sum_i \overbrace{\left(\sum_j L_x u^*_{ij} \right)}^{L_x \text{ operator applied to } i^{th} \text{row}}$$

In other words, $\sum_j L_x u^*_{ij}$ is the result of applying L_x to each entry in the i^{th} row, then summing the result.

Therefore, L_x is applied to entry $u^*(i,1)$ then to entry $u^*(i,2)$ and so on, until the last entry in the row which is $u^*(i,n)$.

How to find the result of applying L_x on each row? Given that L_x for 1D with homogenous Neumann boundary conditions is

$$L_x = \frac{1}{h^2} \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ \cdots & \cdots & \cdots & \ddots & \cdots & \cdots \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

Then applying the operator to each entry in the i^{th} row gives the following

$$\begin{array}{cccccccccccc}
 & j=1 & j=2 & j=3 & 4 & 5 & 6 & 7 & \cdots & n-2 & n-1 & j=n \\
 & -u_{i1} & +u_{i2} & & & & & & & & & \\
 & +u_{i1} & -2u_{i2} & +u_{i3} & & & & & & & & \\
 & & +u_{i2} & -2u_{i3} & +u_{i4} & & & & & & & \\
 & & & +u_{i3} & -2u_{i4} & +u_{i5} & & & & & & \\
\left(\sum_j L_x u^*_{ij}\right) = & & & & +u_{i4} & -2u_{i5} & +u_{i6} & & & & & \\
 & & & & & +u_{i5} & -2u_{i6} & +u_{i7} & & & & \\
 & & & & & & & \ddots & & & & \\
 & & & & & & & & +u_{i,n-3} & -2u_{i,n-2} & +u_{i,n-1} & \\
 & & & & & & & & & +u_{i,n-2} & -2u_{i,n-1} & +u_{i,n} \\
 & & & & & & & & & & +u_{i,n-1} & -u_{i,n}
\end{array}$$

In the above, L_x was applied directly on the i^{th} row. The first line above shows the column index j which goes from $1 \cdots n$. The following diagram is made to help illustrate the above process, showing how L_x and L_y are applied to the solution in the u matrix.

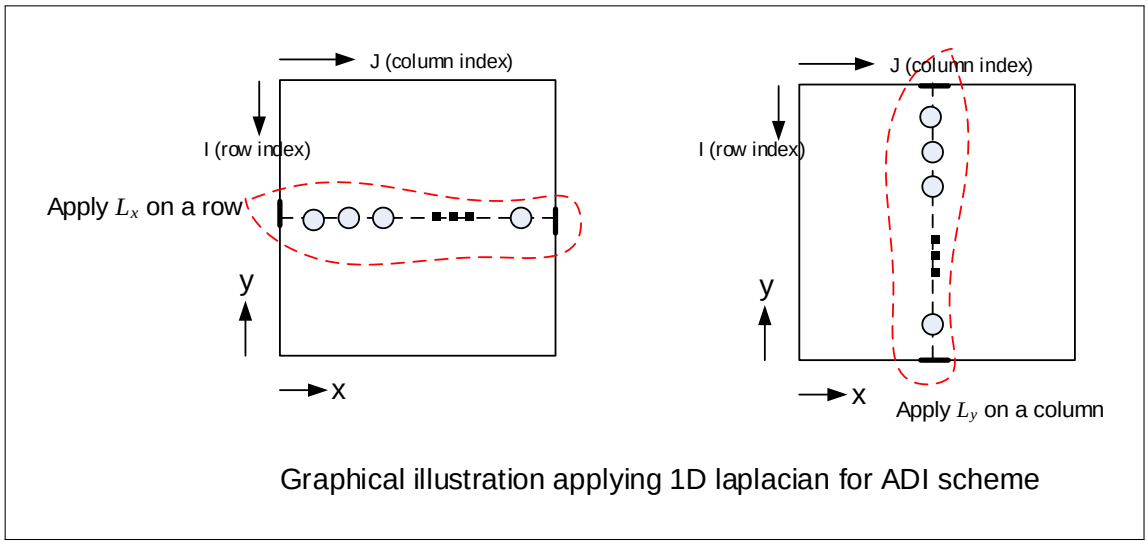


Figure 3.45: Illustrating the above process

One can see now that thesum is zero due to terms cancellation. The sum is zero in this case due to the homogenous Neumann boundary conditions which caused the first and last entries to cancel out.

Using the same procedure, then applying L_y to each column of u^n will also result in zero sum, since the north and south boundaries also have homogenous Neumann boundary conditions. Since boundary conditions do not change going from u^* to u^{n+1}, the same result is obtained when applying L_y operator to each column of u^{n+1}. From the above discussion, it is found

that

$$\sum_i \sum_j L_x u^*_{ij} = 0$$
$$\sum_j \sum_i L_y u^n_{ij} = 0$$
$$\sum_j \sum_i L_y u^{n+1}_{ij} = 0$$
$$\sum_i \sum_j L_x u^*_{ij} = 0$$

Substituting the above 4 equations back into Eqs. (1B),(2B) gives

$$\sum_i \sum_j u^*_{ij} = \sum_j \sum_i u^n_{ij} \tag{1C}$$
$$\sum_j \sum_i u^{n+1}_{ij} = \sum_i \sum_j u^*_{ij} \tag{2C}$$

Substituting Eq. (1C) into (2C) gives

$$\sum_i j u_i j^n + 1 = \sum_i j u_i j^n$$

Since the above is valid for any n (boundary conditions do not change with time) then setting $n = 0$ in the above results in

$$\sum_{ij} u^1_{ij} = \sum_{ij} u^0_{ij}$$

Similarly, setting $n = 1$ results in

$$\sum_{ij} u^2_{ij} = \sum_{ij} u^1_{ij}$$

and so on all the way any n value. Hence in general the following result is obtained

$$\sum_{ij} u^n_{ij} = \sum_{ij} u^{n-1}_{ij}$$

By repeated back substitution on the RHS, the following is obtained

$$\sum_{ij} u^n_{ij} = \sum_{ij} u^0_{ij}$$

Therefore, the discrete conservation property is satisfied.

3.3.3.4.1 Verification in code To verify part(d) in the code, a table was generated during one run, where $\sum_{ij} u^n_{ij}$ was calculated at the end of each time step using the Matlab command sum(sum(u)), and this value was printed at each time step. The result shows that this value is constant implying the discrete conservation property is satisfied. Here is the result below

```
current_time   sum(U(current_time)
  0.00000      1897.85094
  0.01235      1897.85094
  0.02469      1897.85094
  0.03704      1897.85094
  0.04938      1897.85094
  0.06173      1897.85094
  0.07407      1897.85094
  0.08642      1897.85094
  0.09877      1897.85094
  0.11111      1897.85094
....
  0.92593      1897.85094
  0.93827      1897.85094
  0.95062      1897.85094
  0.96296      1897.85094
  0.97531      1897.85094
  0.98765      1897.85094
```

3.3.4 Problem 3

for all n. Demonstrate this property with your code.

3. The FitzHugh-Nagumo equations

$$\frac{\partial v}{\partial t} = D\Delta v + (a - v)(v - 1)v - w + I$$
$$\frac{\partial w}{\partial t} = \epsilon(v - \gamma w).$$

are used in electrophysiology to model the cross membrane electrical potential (voltage) in cardiac tissue and in neurons. Assuming that the spatial coupling is local and passive results the term which looks like the diffusion of voltage. The state variables are the voltage v and the recovery variable w.

(a) Write a program to solve the FitzHugh-Nagumo equations on the unit square with homogeneous Neumann boundary conditions for v (meaning electrically insulated). Use a fractional step method to handle the diffusion and reactions separately. Use an ADI method for the diffusion solve. Describe what ODE solver you used for the reactions and what fractional stepping your chose.

(b) Use the following parameters $a = 0.1$, $\gamma = 2$, $\epsilon = 0.005$, $I = 0$, $D = 5 \cdot 10^{-5}$, for $h = 0.01$ and initial conditions

$$v(x, y, 0) = \exp\left(-100(x^2 + y^2)\right)$$
$$w(x, y, 0) = 0.0.$$

Note that $v = 0$, $w = 0$ is a stable steady state of the system. Call this the rest state. For these initial conditions the voltage has been raised above rest in the bottom corner of the domain. Generate a numerical solution up to time $t = 300$. What time step did you use and why? Visualize the voltage and describe the solution.

(c) Use the same parameters from part (b), but use the initial conditions

$$v(x, y, 0) = 1 - 2x$$
$$w(x, y, 0) = 0.05y,$$

and run the simulation until time $t = 600$. Show the voltage at several points in time (pseudocolor plot, or contour plot, or surface plot $z = V(x, y, t)$) and describe the solution.

The dynamics of excitable media is a fascinating subject from both the mathematical and physiological perspectives. The electrical patterns that you simulated in part (c) are related to cardiac arrhythmias. For more information see the book *Mathematical Physiology* by Keener and Sneyd.

Just for fun, (there is no need to turn this in or even do it) try to find an input current $I(x, y, t)$ in the form of a short pulse (e.g. $I(x, y, t) = f(x, y) \exp\left(-\kappa(t - t_p^2)\right)$) so that the normal electrical wave from part (b) degenerates into an arrhythmia like that from part (c). Then try to find a pulse of current that will eliminate the arrhythmia. This second task may be easier. What to the doctors on TV do?

Figure 3.46: Problem statement

3.3.4.1 Part(a)

The equations to solve are the following

$$\frac{\partial v}{\partial t} = D\Delta v + (a - v)(v - 1)v - w + I$$
$$\frac{\partial w}{\partial t} = \epsilon\Big(v - \gamma w\Big)$$

The first PDE $\frac{\partial v}{\partial t} = D\Delta v+(a - v)(v - 1)v-w+I$ was solved by the splitting method by solving the diffusion equation $\frac{\partial v}{\partial t} = D\Delta v$ using ADI method separately and then by solving the reaction (non-linear) equation $\frac{\partial v}{\partial t} = (a - v)(v - 1)v - w + I$ along with $\frac{\partial w}{\partial t} = \epsilon\Big(v - \gamma w\Big)$ separately. The following is a coupled first order non-linear differential equations system (the reaction ODE is nonlinear in votage v)

$$\frac{dv}{dt} = (a - v)(v - 1)v - w + I$$
$$\frac{dw}{dt} = \epsilon\Big(v - \gamma w\Big)$$

The above system was solved using Runge-Kutta order 4. The following diagram illustrates the time line for one full splitting step.

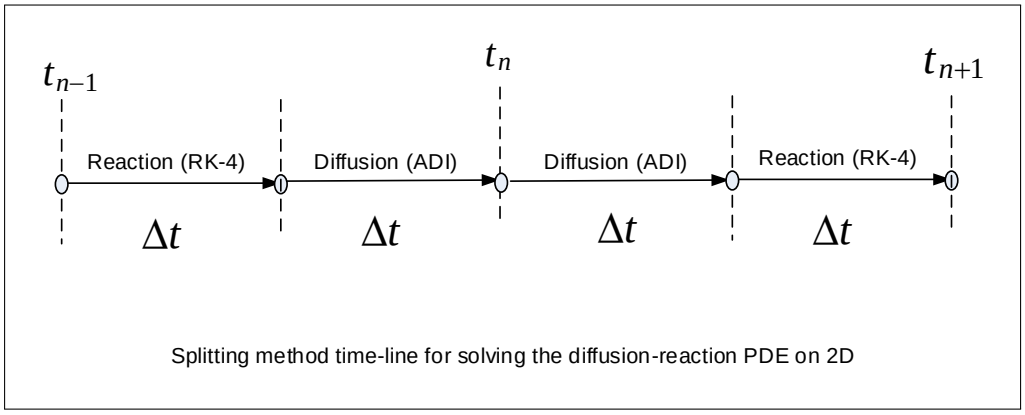


Figure 3.47: time line for one full splitting step

ADI was described in problem 2, and the same function was reused for this problem for the diffusion solver. For solving the reaction system of equations, RK4 was implemented as follows. define

$$f(v, w) = (a - v)(v - 1)v - w + I$$

and also define

$$g(v, w) = \epsilon\Big(v - \gamma w\Big)$$

Therefore, the RK4 solver for the above system becomes

$$v^{n+1} = v^n + \frac{1}{6}(m_1 + 2m_2 + 2m_3 + m_4)$$
$$w^{n+1} = w^n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Where

$$m_1 = \Delta t f(v, w)$$
$$m_2 = \Delta t f\left(v + \frac{1}{2}m_1, w + \frac{1}{2}k_1\right)$$
$$m_3 = \Delta t f\left(v + \frac{1}{2}m_2, w + \frac{1}{2}k_2\right)$$
$$m_4 = \Delta t f(v + m_3, w + k_3)$$

And

$$k_1 = \Delta t g(v, w)$$
$$k_2 = \Delta t g\left(v + \frac{1}{2}m_1, w + \frac{1}{2}k_1\right)$$
$$k_3 = \Delta t g\left(v + \frac{1}{2}m_2, w + \frac{1}{2}k_2\right)$$
$$k_4 = \Delta t g(v + m_3, w + k_3)$$

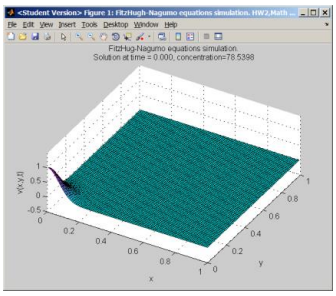
Another point regarding the splitting method. It was required to decide which splitting method to use. Should a simple splitting, Strang splitting or the 2-step splitting method which was described in class be used? To make sure the second order accuracy of ADI 2D in time is preserved, simple splitting was not used (unless the operators commute, this would have caused the scheme to become first order accurate in time). Instead, the two step splitting method was used, as it was found to be simpler than Strang method to implement.

3.3.4.2 Part(b)

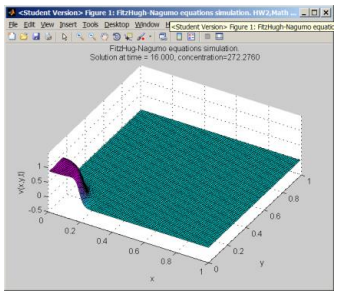
The program written in part(a) was run using the parameters given. The time step used was set to be the same as the space step. This time step is recommended for the ADI The diffusion solver as it is a second order accurate in time and space. This is the fast system (the stiff part of the system), hence making the time step larger than the space step would not give accurate results, even though it will remain a stable scheme. Keeping the time step the same as the space step seemed to be a good choice, as it kept the

time resolution and the space resolution the same. The same time step was then used for the reaction solver, as was required by the splitting method to keep each step the same length.

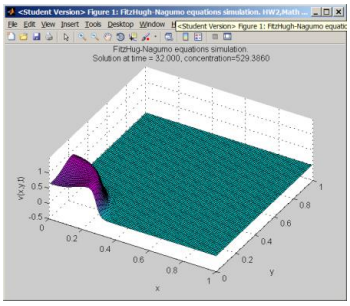
The following shows the visualization of the voltage solution for up to 300 seconds as required using the surf() command.



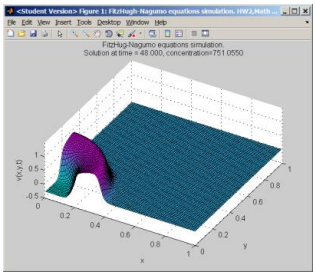
Time=0 sec



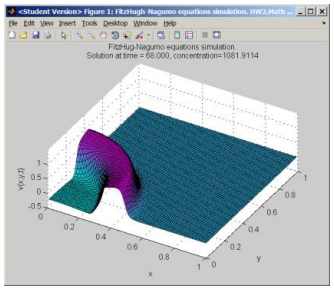
Time=16 sec



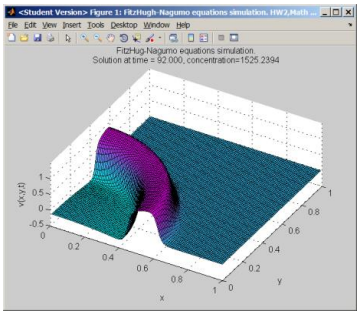
Time=32 sec



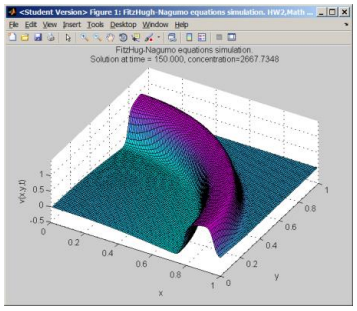
Time=48 sec



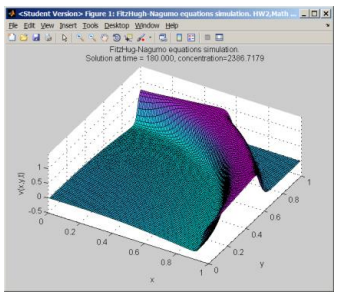
Time=68 sec



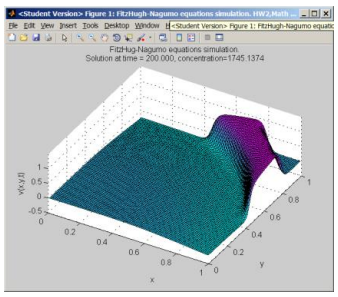
Time=92 sec



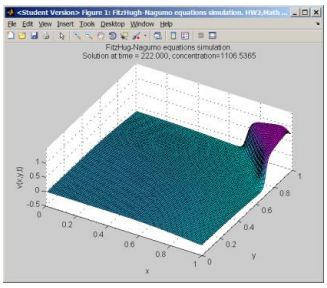
Time=150 sec



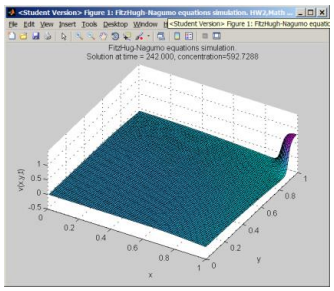
Time=180 sec



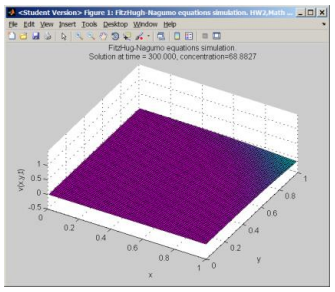
Time=200 sec



Time=220 sec



Time=242 sec



Time=300 sec

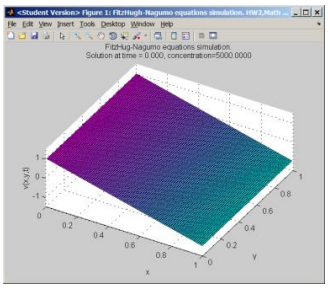
Result for Part(b), HW2. FitzHugh-Nagumo equations simulation

Figure 3.48: visualization of the voltage solution for up to 300 seconds

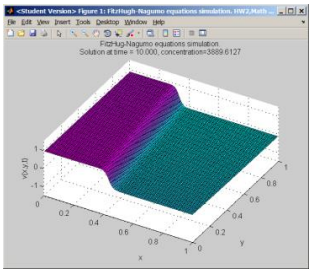
The solution $v(t)$ started from a peak value at one corner of the square. Shortly after, at about 50 seconds, a wave started to form, the wave front became large and it spread out and advanced with time. When the pulse reached the boundary on the other corner, it started to diffuse and by $t = 300$ seconds, the pulse has completely disappeared.

3.3.4.3 Part(c)

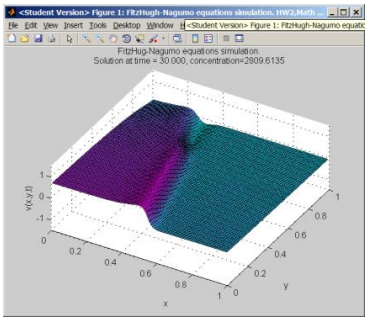
The following is the result of the simulation for this part.



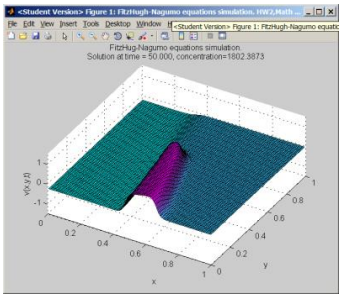
t=0 seconds



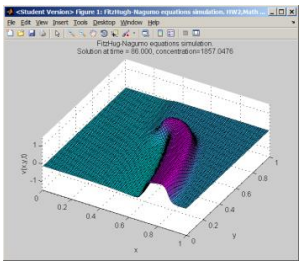
t=10 seconds



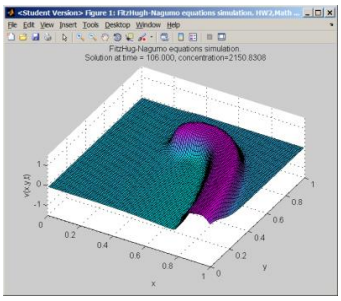
t=30 seconds



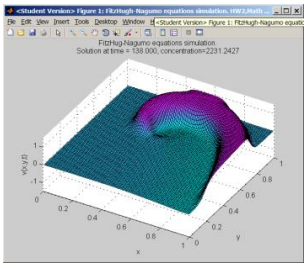
t=50 seconds



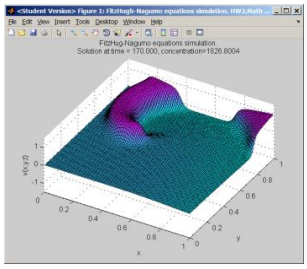
t=86 seconds



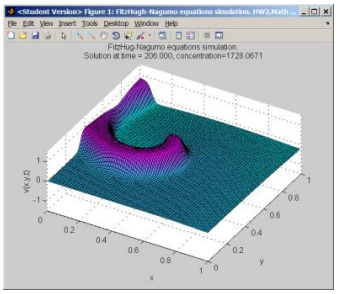
t=106 seconds



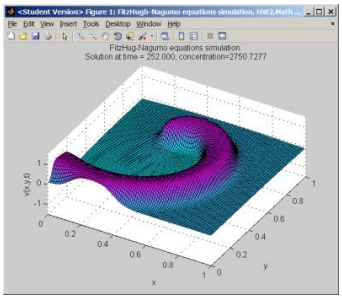
t=138 seconds



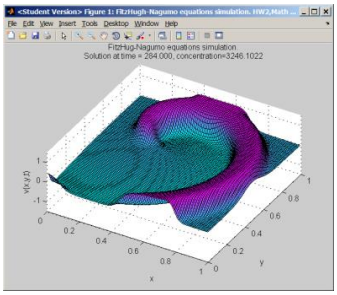
t=170 seconds



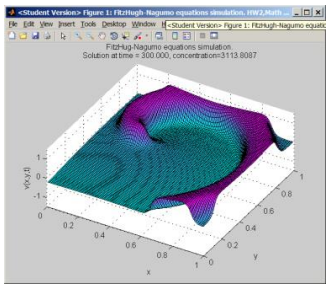
t=206 seconds



t=250 seconds



t=284 seconds



t=300 seconds

Result of part c problem 3. First 300 seconds

Figure 3.49: simulation of part c

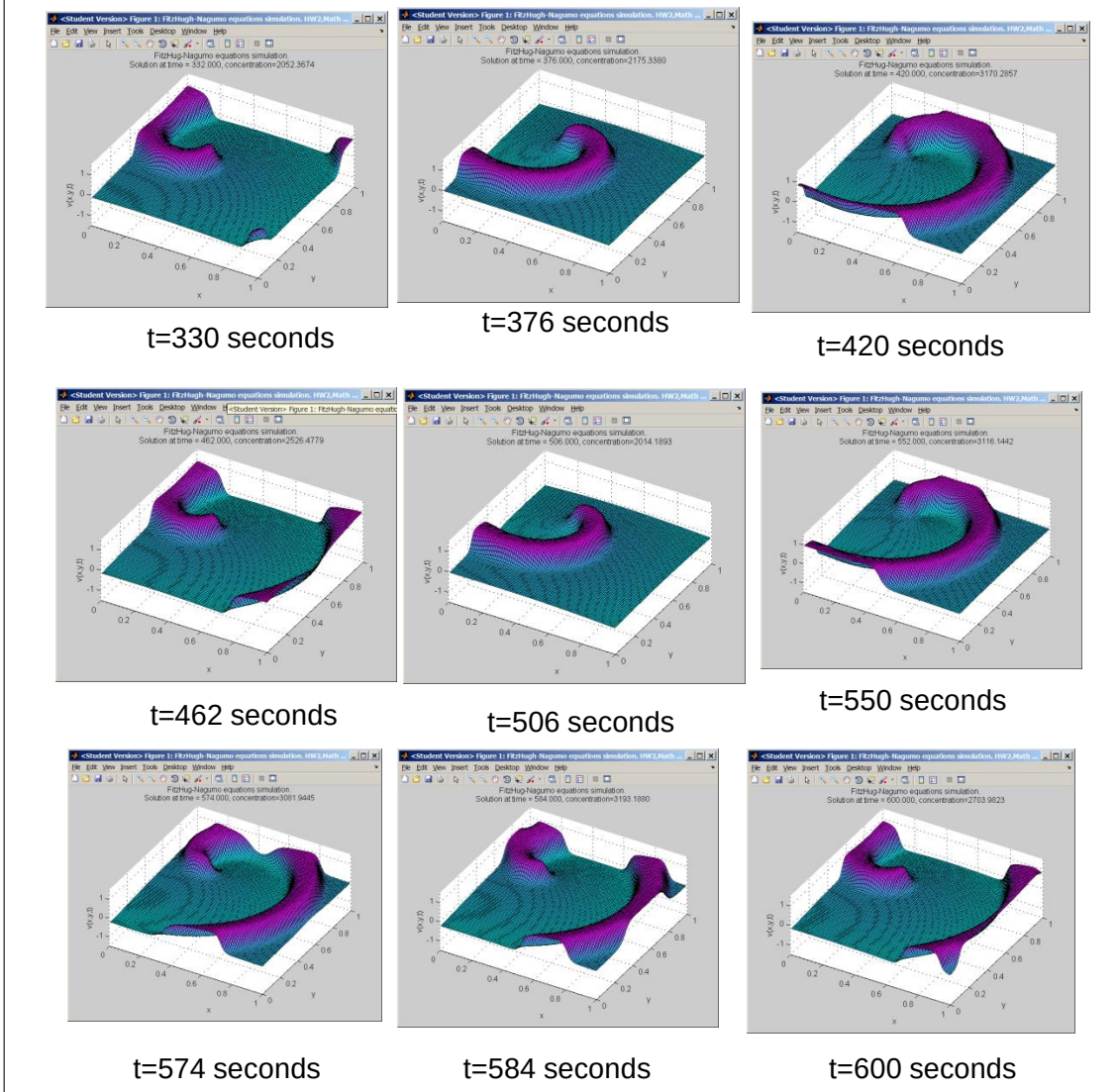


Figure 3.50: Up to 300 seconds

In this simulation, the pulse that appeared after about 50 second, quickly became a spiral, and did not appear to diffuse as was the case in part (d). At the end of the simulation, the pulse was continuing to spiral in the same rotation direction it started with. The above phenomena seem to be termed an arrhythmia pulse.

One common theme between part(c) and part (b), is the formation of a wave like motion that traveles across the domain. The difference was in the shape of the pulse, the direction it moves to and the amount of diffusion that occured.

3.3.5 Appendix

3.3.5.1 Problem 1 appendix

3.3.5.1.1 Derivation of ADI equations for 2D and 3D for the diffusion problem

Given $u_t = \Delta u$ (D is assumed 1), then in 2D forward Euler (explicit) gives

$$\frac{u_{ij}^{n+1} - u_{ij}^{n}}{k} = \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)_{ij}^{n}$$

While C-N method gives

$$\frac{u_{ij}^{n+1} - u_{ij}^{n}}{k} = \frac{1}{2} \left[\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)_{ij}^{n+1} + \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)_{ij}^{n} \right]$$

However, in ADI, the time step itself is divided by half, and in the first half step, one of the spatial second derivatives is implicit while and the other spatial second derivative is explicit. In the second half step, these are reversed.

$$\frac{u_{ij}^{n+\frac{1}{2}} - u_{ij}^{n}}{k/2} = \overbrace{\left(\frac{\partial^2 u}{\partial x^2} \right)_{ij}^{n+\frac{1}{2}}}^{implicit} + \overbrace{\left(\frac{\partial^2 u}{\partial y^2} \right)_{ij}^{n}}^{\text{explicit}}$$

$$\frac{u_{ij}^{n+1} - u_{ij}^{n+\frac{1}{2}}}{k/2} = \overbrace{\left(\frac{\partial^2 u}{\partial x^2} \right)_{ij}^{n+\frac{1}{2}}}^{\text{explicit}} + \overbrace{\left(\frac{\partial^2 u}{\partial y^2} \right)_{ij}^{n+1}}^{implicit}$$

Writing $\frac{\partial^2 u}{\partial x^2} = L_x = \frac{u_{i-1,j} - 2u_{ij} + u_{i+1,j}}{h^2}$ and $\frac{\partial^2 u}{\partial y^2} = L_y = \frac{u_{i,j-1} - 2u_{ij} + u_{i,j+1}}{h^2}$, the above 2 equations become

$$\frac{u_{ij}^{n+\frac{1}{2}} - u_{ij}^{n}}{k/2} = \frac{u_{i-1,j}^{n+\frac{1}{2}} - 2u_{ij}^{n+\frac{1}{2}} + u_{i+1,j}^{n+\frac{1}{2}}}{h^2} + \frac{u_{i,j-1}^{n} - 2u_{ij}^{n} + u_{i,j+1}^{n}}{h^2}$$

$$\frac{u_{ij}^{n+1} - u_{ij}^{n+\frac{1}{2}}}{k/2} = \frac{u_{i-1,j}^{n+\frac{1}{2}} - 2u_{ij}^{n+\frac{1}{2}} + u_{i+1,j}^{n+\frac{1}{2}}}{h^2} + \frac{u_{i,j-1}^{n+1} - 2u_{ij}^{n+1} + u_{i,j+1}^{n+1}}{h^2}$$

Moving all implicit terms to the LHS and rearranging results in

$$u_{ij}^{n+\frac{1}{2}} - \frac{k}{2} \frac{u_{i-1,j}^{n+\frac{1}{2}} - 2u_{ij}^{n+\frac{1}{2}} + u_{i+1,j}^{n+\frac{1}{2}}}{h^2} = u_{ij}^{n} + \frac{k}{2} \frac{u_{i,j-1}^{n} - 2u_{ij}^{n} + u_{i,j+1}^{n}}{h^2}$$

$$u_{ij}^{n+1} - \frac{k}{2} \frac{u_{i,j-1}^{n+1} - 2u_{ij}^{n+1} + u_{i,j+1}^{n+1}}{h^2} = u_{ij}^{n+\frac{1}{2}} + \frac{k}{2} \frac{u_{i-1,j}^{n+\frac{1}{2}} - 2u_{ij}^{n+\frac{1}{2}} + u_{i+1,j}^{n+\frac{1}{2}}}{h^2}$$

Hence, in operator form the above becomes

$$\left(I - \frac{k}{2}L_x\right)u_{ij}^{n+\frac{1}{2}} = \left(I + \frac{k}{2}L_y\right)u_{ij}^{n}$$
$$\left(I - \frac{k}{2}L_y\right)u_{ij}^{n+1} = \left(I + \frac{k}{2}L_x\right)u_{ij}^{n+\frac{1}{2}}$$

In the class notes, u_{ij}^{*} was used to represent $u_{ij}^{n+\frac{1}{2}}$ but they are the same. The above is the ADI scheme for 2D. The 3D equations are now derived. Since three different directions exist now, the time step is divided into 3. This results in

$$\frac{u_{ij}^{n+\frac{1}{3}} - u_{ij}^{n}}{\Delta t/3} = \overbrace{\left(\frac{\partial^2 u}{\partial x^2}\right)_{ij}^{n+\frac{1}{3}}}^{implicit} + \overbrace{\left(\frac{\partial^2 u}{\partial y^2}\right)_{ij}^{n}}^{\text{explicit}} + \overbrace{\left(\frac{\partial^2 u}{\partial z^2}\right)_{ij}^{n}}^{\text{explicit}}$$
$$\frac{u_{ij}^{n+\frac{2}{3}} - u_{ij}^{n+\frac{1}{3}}}{\Delta t/3} = \overbrace{\left(\frac{\partial^2 u}{\partial x^2}\right)_{ij}^{n+\frac{1}{3}}}^{\text{explicit}} + \overbrace{\left(\frac{\partial^2 u}{\partial y^2}\right)_{ij}^{n+\frac{2}{3}}}^{implicit} + \overbrace{\left(\frac{\partial^2 u}{\partial z^2}\right)_{ij}^{n+\frac{1}{3}}}^{\text{explicit}}$$
$$\frac{u_{ij}^{n+1} - u_{ij}^{n+\frac{2}{3}}}{\Delta t/3} = \overbrace{\left(\frac{\partial^2 u}{\partial x^2}\right)_{ij}^{n+\frac{2}{3}}}^{\text{explicit}} + \overbrace{\left(\frac{\partial^2 u}{\partial y^2}\right)_{ij}^{n+\frac{2}{3}}}^{\text{explicit}} + \overbrace{\left(\frac{\partial^2 u}{\partial z^2}\right)_{ij}^{n+1}}^{implicit}$$

Similar to what done in the 2D case, the above are rearranged resulting in

$$\left(I - \frac{D\Delta t}{3}L_x\right)\mathbf{u}^{n+\frac{1}{3}} = \left(I + \frac{D\Delta t}{3}L_y + \frac{D\Delta t}{3}L_z\right)\mathbf{u}^{n} \tag{1}$$
$$\left(I - \frac{D\Delta t}{3}L_y\right)\mathbf{u}^{n+\frac{2}{3}} = \left(I + \frac{D\Delta t}{3}L_x + \frac{D\Delta t}{3}L_z\right)\mathbf{u}^{n+\frac{1}{3}} \tag{2}$$
$$\left(I - \frac{D\Delta t}{3}L_z\right)\mathbf{u}^{n+1} = \left(I + \frac{D\Delta t}{3}L_x + \frac{D\Delta t}{3}L_y\right)\mathbf{u}^{n+\frac{2}{3}} \tag{3}$$

3.3.6 Matlab Source code developed for this HW

3.3.6.1 nma_math228b_HW2_prob2.m

```
function nma_math228b_HW2_prob2
% This function implements refinement study for HW2
% problem 2, Math 228B, Winter 2011, UC Davis
%
%
% By Nasser M. Abbasi

% set up initialization for the error table, such as headings
% and formating

close all;

% for formating of error table
titles = {'#','N','delt','h','|u|','mean(u)','|e|','ratio'};
fms    = {'d','d','.5f','.5f','.5f','.5f','.4e','.5f'};
wid    = 13;
fileID = 1;

% use 8 runs, and allocate the table to store the error and ratios
N=5;
table=zeros(N,8); % #, t, h, |u|,|u-u_last|, ratio, N, mean(u), |exact|

% Initialize space and time steps.
grid_size = 9;
h1 = figure();
D  = 0.1;  %diffusion constant
time_to_run = 1; % 1 second

for n = 1:N

   % Simulatiously divide space step and time step.

   grid_size = grid_size * 3;
   h = 1/grid_size;
   k = h;

   [u,u_steady_state] = solve_2D_diffusion(grid_size,h,k,D,time_to_run);

   % the numerical solution now is stored in u. Make
   % a new entry in the error table for this iteration.
   table(n,1) = n;
   table(n,7) = grid_size;
   table(n,8) = mean(mean(u));
```

```
   table(n,2) = k;
   table(n,3) = h;

   table(n,4) = h*norm(u(:),2);  % use grid 2-norm

   if n>1
      table(n,5) = abs(table(n-1,4)-table(n,4)); %e
      %table(n,5) = h*norm( u-u_steady_state,2); %e

      if n==2
         table(n,6)=1;
      else
         table(n,6) = table(n-1,5)/table(n,5); %e ratio
      end

      [hd,bdy]=nma_format_matrix(titles, ...
         [table(2:n,1) table(2:n,7) table(2:n,2) table(2:n,3) table(2:n,4)  table(2:n,8) tabl
         wid,fms,fileID,true );

      clf(h1);
      set(0,'CurrentFigure',h1);
      ax = axes();
      set(h1, 'CurrentAxes',ax);
      cla('reset');

      text(.1,.60,bdy,'FontSize',10);
      set(ax,'YTick',[]);
      set(ax,'XTick',[]);
      text(.1,.9,hd,'FontSize',10);
      title('result of refinement study');
      drawnow();

   end
end

% The refinment study is completed. Generate plots and error table

h2  = figure();
ax2 = axes();
set(0,'CurrentFigure',h2);
set(h2, 'CurrentAxes',ax2);
cla('reset');
set(0,'defaultaxesfontsize',8) ;

loglog(table(2:end,3),table(2:end,5),'-d');
xlabel('log(h)','FontSize',8);
title({'refinement study result, ';'log vs successive errors difference'},...
```

```
   'FontSize',8);
ylabel('log(error norm)','FontSize',8);
grid on;
end

%-----------------------
function [u,u_steady_state]=solve_2D_diffusion(...
   grid_size,...
   h,... % space step size
   k,... % time step size
   D,... % diffusion constant
   max_t... % maximum time to run solver for
   )

n  = grid_size-2; %internal nodes
ic = @(X,Y) exp( -10*((X-0.4).^2 + (Y-0.4).^2 ));   % initial data
[X,Y]  = meshgrid(h/2:h:1-h/2,1-h/2:-h:h/2); % coordinates
u_mean = quad2d(ic,h/2,1-(h/2),h/2,1-(h/2));
%u_mean = quad2d(ic,0,1,0,1);

%ic= @(X,Y) -exp( -(X-0.25).^2 - (Y-0.6).^2 );

% create sparse matrices for A and B (implicit and explicit) see HW report
A     = lap2D_diffusion_ADI_A(n,D,k,h);
A_RHS = lap2D_diffusion_ADI_A_RHS(n,D,k,h);

u = ic(X,Y); % initial U
u_steady_state = zeros(size(u));
u_steady_state(:,:)=u_mean;
u_max  = max(max(u));
u_min  = min(min(u));
h1        = figure();
current_t = 0;
done      = false;

while not(done)

   % solve for U*
   tmp = reshape(flipud(u(2:end-1,2:end-1))',n^2,1);
   sol = A\(A_RHS*tmp);
   u(2:end-1,2:end-1) = flipud(reshape(sol,n,n)');

   %update the boundaries
   u = update_BC(u);

   % solve for U_n+1
   tmp = reshape(flipud(u(2:end-1,2:end-1)),n^2,1);
```

```
   sol = A\(A_RHS*tmp);
   u(2:end-1,2:end-1) = flipud(reshape(sol,n,n));

   u = update_BC(u);

   set(0,'CurrentFigure',h1);
   surf(X,Y,u);
   colormap cool;
   title(sprintf('solution at time = %1.3f, D=%.3f, N=%d\nsteady state=%1.4f, h=%1.5f',...
      current_t,D,grid_size,u_mean,h));

   hold on;
   mesh(X,Y,u_steady_state);
   zlim([u_min u_max]);
   drawnow();
   hold off;

   %update current time and check if reached end of time
   current_t = current_t + k;
   if current_t > max_t
      done = true;
   end


end
close(h1);
end

%------------------------
function A=lap2D_diffusion_ADI_A(...
   n, ... %size of matrix (1D size)
   D, ... %diffusion constant
   k, ... %time step
   h)     %space_step

r  = D*k/(2*h^2);
e  = ones(n,1);
B  = [-r*e (1+2*r)*e -r*e];
Lx = spdiags(B,[-1 0 1],n,n);
Ix = speye(n);
A  = kron(Ix,Lx);

% adjust A, see HW report

pos = 1:n:n^2;

for i = 1:length(pos)
```

```
   A(pos(i),pos(i))=1+r;
end

pos = n:n:n^2;

for i=1:length(pos)
   A(pos(i),pos(i))=1+r;
end

end
%----------------------------------------
function A=lap2D_diffusion_ADI_A_RHS(...
   n, ... %size of matrix (1D)
   D, ... %diffusion constant
   k, ... %time step
   h)     %space_step

r = D*k/(2*h^2);

e = ones(n^2,1);
B = [r*e (1-r)*e r*e];
A = spdiags(B,[-n 0 n],n^2,n^2);

%adjust matrix, see HW report
pos = n+1:n^2-n;
for i=1:length(pos)
   A(pos(i),pos(i))=1-2*r;
end

end

%---------------------
function    u = update_BC(u)

   u(1,2:end-1)   = u(2,2:end-1);
   u(end,2:end-1) = u(end-1,2:end-1);
   u(2:end-1,1)   = u(2:end-1,2);
   u(2:end-1,end) = u(2:end-1,end-1);


   u(1,1)     = u(1,2);
   u(end,1)   = u(end-1,1);
   u(end,end) =  u(end,end-1);
   u(1,end)   =  u(1,end-1);

end
```

3.4 HW 3

3.4.1 Problem 1

Math 228B
Homework 3
Due Thursday, 3/03/11

1. Write programs to solve the advection equation

$$u_t + au_x = 0,$$

on $[0, 1]$ with periodic boundary conditions using upwinding and Lax-Wendroff. For smooth solutions we expect upwinding to be first-order accurate and Lax-Wendroff to be second-order accurate, but it is not clear what accuracy to expect for nonsmooth solutions.

(a) Let $a = 1$ and solve the problem up to time $t = 1$. Perform a refinement study for both upwinding and Lax-Wendroff with $\Delta t = 0.8h$ with a smooth initial condition. Compute the rate of convergence in the 1-norm, 2-norm, and max-norm. Note that the exact solution at time $t = 1$ is the initial condition, and so computing the error is easy.

(b) Repeat the previous problem with the discontinuous initial condition

$$u(x, 0) = \begin{cases} 1 & \text{if } |x - 1/2| < 1/4 \\ 0 & \text{otherwise} \end{cases}$$

Figure 3.51: Problem description

3.4.1.1 Part (a)

The advection PDE in 1D is given by

$$u_t + au_x = 0$$

Where a represents the speed of flow, which can be positive or negative. The Lax-Wendroff finite difference scheme for the above PDE is given by

$$u_j^{n+1} = u_j^n - \frac{ak}{2h}\left(u_{j+1}^n - u_{j-1}^n\right) + \frac{a^2k^2}{2h^2}\left(u_{j-1}^n - 2u_j^n + u_{j+1}^n\right)$$

where k is the time step and h is the space step. The upwind finite difference scheme for $a > 0$ is given by

$$u_j^{n+1} = u_j^n - \frac{ak}{h}\left(u_j^n - u_{j-1}^n\right)$$

The relation between a, k and h is given by

$$courant\ number = \nu = \frac{ak}{h}$$

Both schemes above are stable for $|v| \leq 1$. The problem asked to use $\nu = 0.8$ and $a = 1$, giving $k = 0.8h$.

A program was written to implement these schemes for both smooth and discontinues initial conditions. The exact solution was computed from $u = u_0(x - at)$ where $u_0(x)$ is the initial data. $\sin(4\pi x)$ was used for smooth initial conditions.

The boundary conditions are periodic. This means that the first grid point if physically the same as the last grid point as would be the case by viewing the domain as a closed ring. The following diagram illustrates the numbering used.

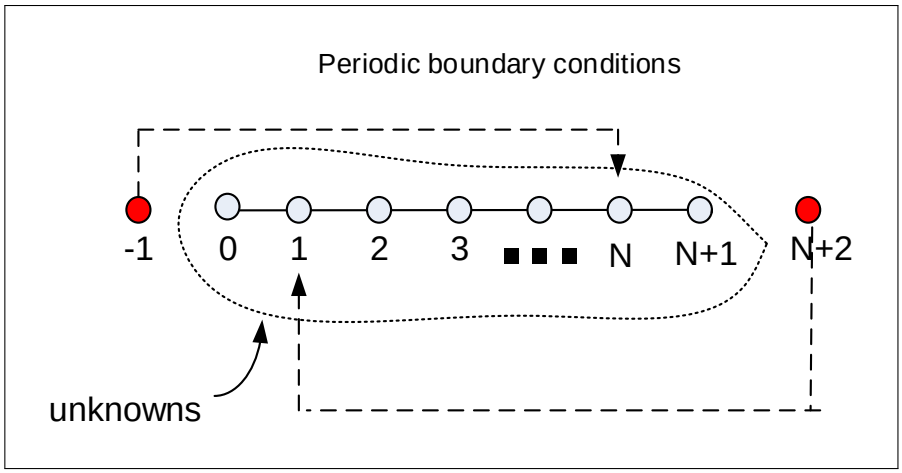


Figure 3.52: Grid used

Error norm calculation

The total error at a grid point j is given by

$$\mathbf{e}_j = \mathbf{U}_j - \mathbf{u}(x_j)$$

Where $\mathbf{U}_j$ is the numerical solution at the j^{th} grid point and $\mathbf{u}(x_j)$ is the exact solution evaluated at the same grid point location. $\mathbf{e}$ is a vector of length N where N is the number of grid points.

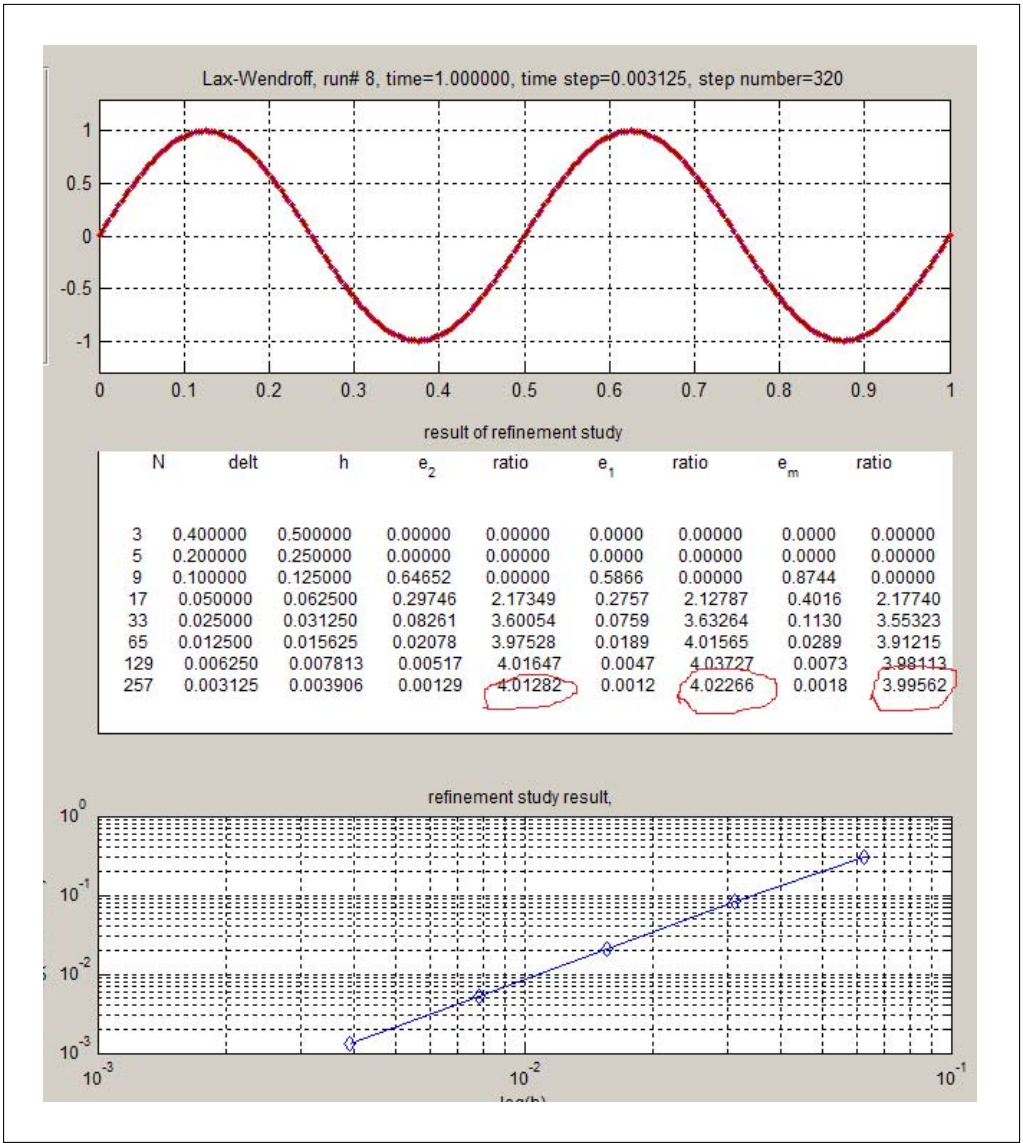
To measure the size of the error vector $\mathbf{e}$, a grid norm is used in place of the standard vector norm. The following are the definitions of the norms used.

1. max-norm (also called infinity norm) $\|\mathbf{e}^h\|_{\max} = \max_j |e_j|$
2. 1-norm $\|\mathbf{e}^h\|_1 = h\sum_{j=1}^{N} |e_j|$
3. 2-norm $\|\mathbf{e}^h\|_2 = \sqrt{h\sum_{j=1}^{N} |e_j|^2}$

3.4.1.2 Results of refinement study for Lax-Wendroff

The result of the refinement study for smooth data for Lax-Wendroff shows that the error ratio converged to 4, and since the space step was divided by 2 at each run, this indicates a second order accuracy in time and space

The following diagram shows the results obtained. All norms gave the same order of accuracy. In the diagram below, the first ratio column represent the error ratio found using norm-2, while the second ratio column represents the norm-1 result, and the third ratio column is for the max-norm. The log plot is generated only for 2-norm. The following parameters were used: $h = 0.01$, maximum time $= 1$ second, $\Delta t = 0.8h$, and initial conditions $u(x,0) = \sin(4\pi x)$ with periodic boundary conditions.

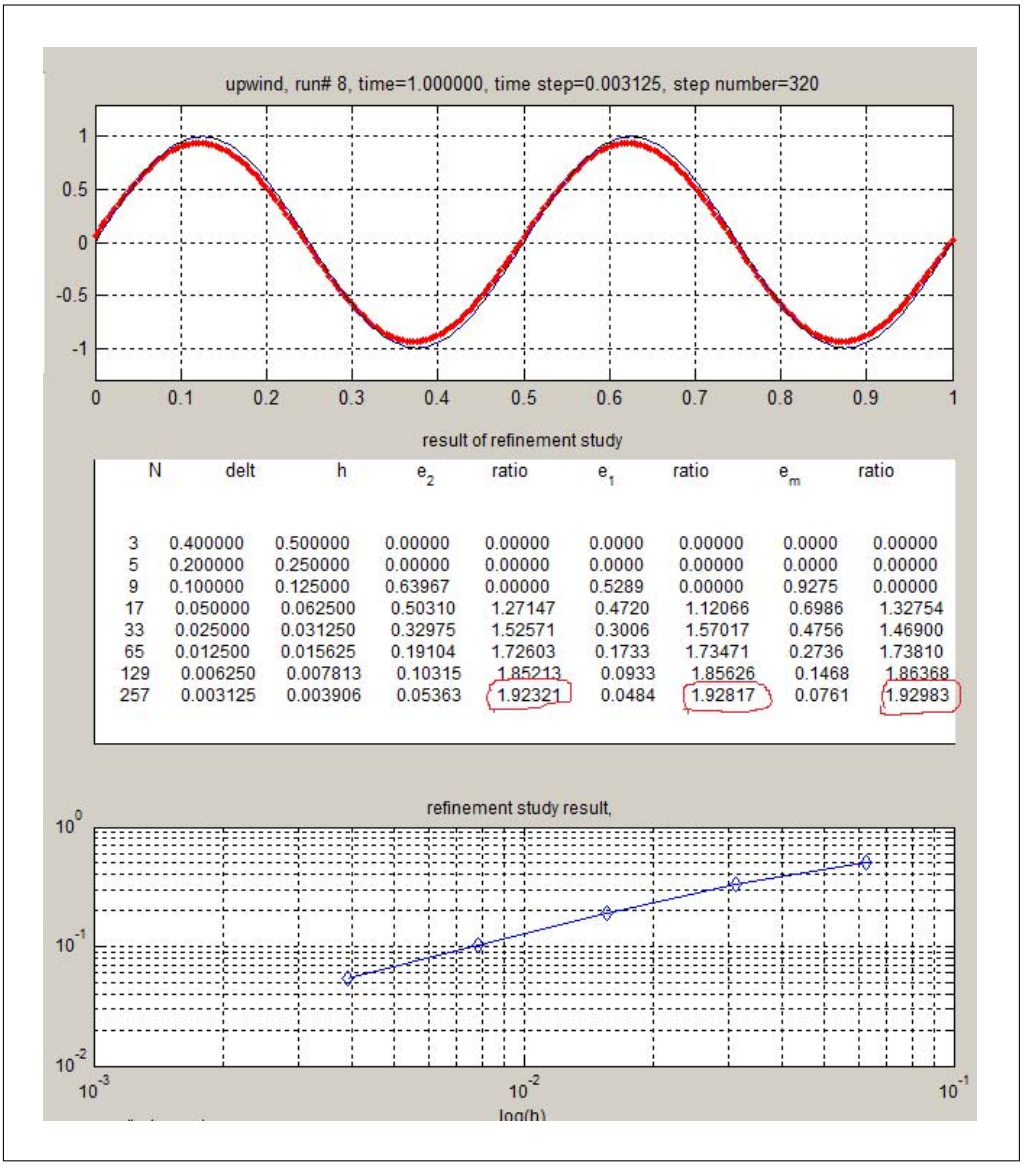


N	delt	h	e_2	ratio	e_1	ratio	e_m	ratio
3	0.400000	0.500000	0.00000	0.00000	0.0000	0.00000	0.0000	0.00000
5	0.200000	0.250000	0.00000	0.00000	0.0000	0.00000	0.0000	0.00000
9	0.100000	0.125000	0.64652	0.00000	0.5866	0.00000	0.8744	0.00000
17	0.050000	0.062500	0.29746	2.17349	0.2757	2.12787	0.4016	2.17740
33	0.025000	0.031250	0.08261	3.60054	0.0759	3.63264	0.1130	3.55323
65	0.012500	0.015625	0.02078	3.97528	0.0189	4.01565	0.0289	3.91215
129	0.006250	0.007813	0.00517	4.01647	0.0047	4.03727	0.0073	3.98113
257	0.003125	0.003906	0.00129	4.01282	0.0012	4.02266	0.0018	3.99562

Figure 3.53: refinement study part a LAX

Result of refinement study for Upwind

The result of the refinement study for smooth data for upwind showed that the error ratio converged to 2 indicating a first order accuracy in time and space. The following diagram shows the results obtained. All norms gave the same order of accuracy.



N	delt	h	e_2	ratio	e_1	ratio	e_m	ratio
3	0.400000	0.500000	0.00000	0.00000	0.0000	0.00000	0.0000	0.00000
5	0.200000	0.250000	0.00000	0.00000	0.0000	0.00000	0.0000	0.00000
9	0.100000	0.125000	0.63967	0.00000	0.5289	0.00000	0.9275	0.00000
17	0.050000	0.062500	0.50310	1.27147	0.4720	1.12066	0.6986	1.32754
33	0.025000	0.031250	0.32975	1.52571	0.3006	1.57017	0.4756	1.46900
65	0.012500	0.015625	0.19104	1.72603	0.1733	1.73471	0.2736	1.73810
129	0.006250	0.007813	0.10315	1.85213	0.0933	1.85626	0.1468	1.86368
257	0.003125	0.003906	0.05363	1.92321	0.0484	1.92817	0.0761	1.92983

Figure 3.54: refinement study upwind

3.4.1.3 Part (b)

The refinement study made in part (a) was repeated using the following initial conditions

$$u(x,0) = \begin{cases} 1 & \left|x - \frac{1}{2}\right| < \frac{1}{4} \\ 0 & otherwise \end{cases}$$

Which is a rectangular pulse of the following shape

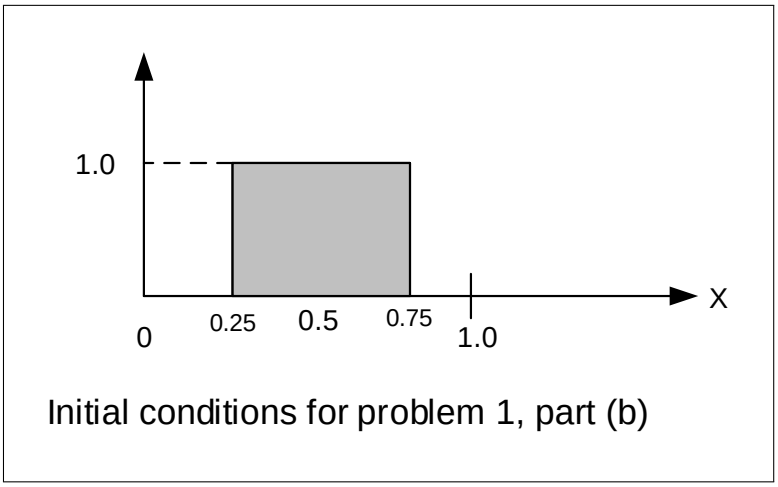


Figure 3.55: initial data

The following diagram shows the results obtained.

Results of refinement study for Lax-Wendroff

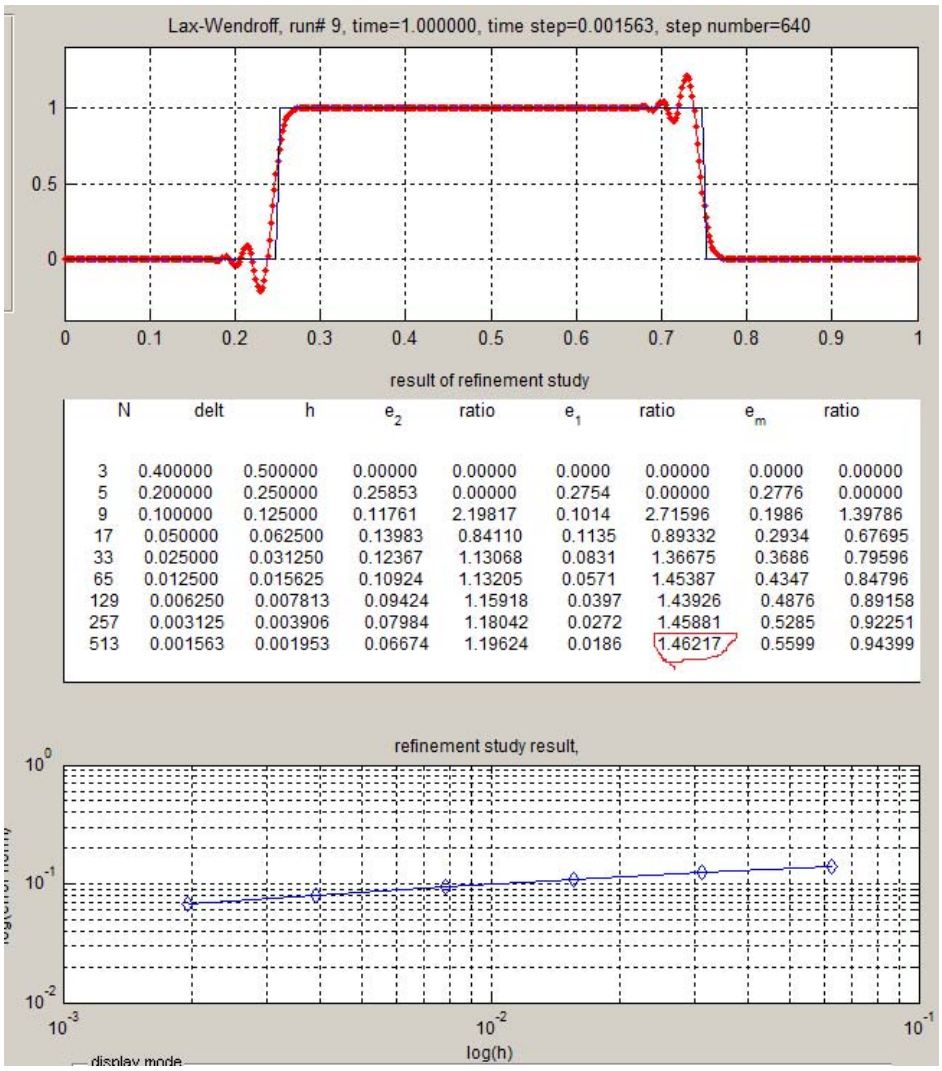


Figure 3.56: refinement study part b LAX

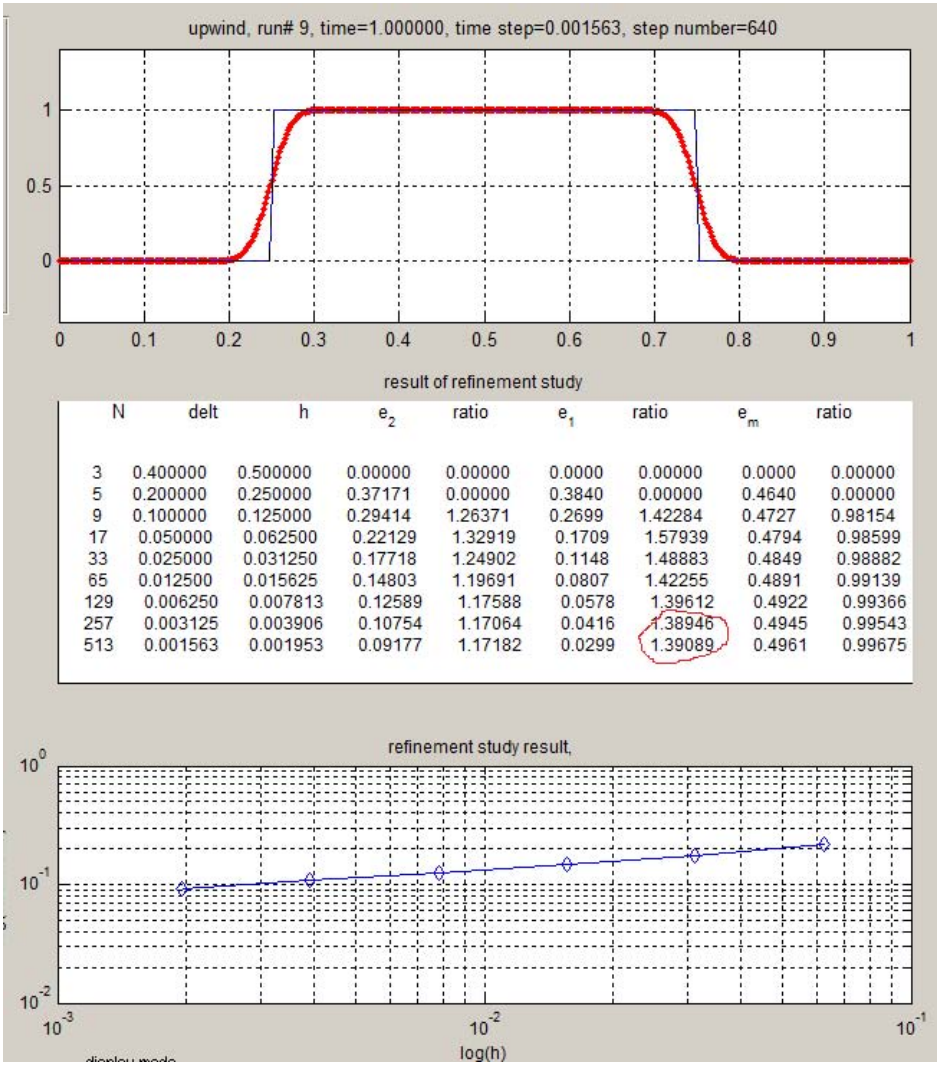
The following table is a summary of the results of the above refinement study for Lax-wendroff

Norm	ratio	order of accuracy $p = \log_2(ratio)$
1-norm	1.5	$\frac{1}{2}$
2-norm	1.2	$\frac{1}{4}$
max-norm	1	0

The maximum norm being zero order says that the largest error in absolute terms does not decrease. Hence for discontinues data, convergence will not occur in the max-norm, no matter how small h is made.

Results of refinement study for upwind

The following diagram shows the results obtained



N	delt	h	e_2	ratio	e_1	ratio	e_m	ratio
3	0.400000	0.500000	0.00000	0.00000	0.0000	0.00000	0.0000	0.00000
5	0.200000	0.250000	0.37171	0.00000	0.3840	0.00000	0.4640	0.00000
9	0.100000	0.125000	0.29414	1.26371	0.2699	1.42284	0.4727	0.98154
17	0.050000	0.062500	0.22129	1.32919	0.1709	1.57939	0.4794	0.98599
33	0.025000	0.031250	0.17718	1.24902	0.1148	1.48883	0.4849	0.98882
65	0.012500	0.015625	0.14803	1.19691	0.0807	1.42255	0.4891	0.99139
129	0.006250	0.007813	0.12589	1.17588	0.0578	1.39612	0.4922	0.99366
257	0.003125	0.003906	0.10754	1.17064	0.0416	1.38946	0.4945	0.99543
513	0.001563	0.001953	0.09177	1.17182	0.0299	1.39089	0.4961	0.99675

Figure 3.57: refinement study part b UPWIND

The following table is a summary of the results of the above refinement study for upwind. The results are similar to Lax-Wendroff.

Norm	ratio	order of accuracy $p = \log_2(ratio)$
1-norm	1.4	0.485
2-norm	1.2	$\frac{1}{4}$
max-norm	1	0

It is noticed that Lax-Wendroff is more accurate scheme than upwind, but only if the initial data is smooth. For discontinuous initial conditions, Lax-wendroff loses its advantage over upwind, and both schemes gave similar order of accuracy.

3.4.2 Problem 2

2. Consider three-point explicit schemes for the linear advection equation in the real line of the form

$$u_j^{n+1} = u_j^n - C\left(u_j^n - u_{j-1}^n\right) + D\left(u_{j+1}^n - u_j^n\right).$$

Show that

$$\sum_j \left|u_j^{n+1} - u_{j-1}^{n+1}\right| \leq \sum_j \left|u_j^n - u_{j-1}^n\right| \tag{1}$$

if $C \geq 0$, $D \geq 0$, and $C + D \leq 1$. When the numerical solution of a scheme satisfies (1) the scheme is total variation diminishing or TVD. Put upwinding and Lax-Wendroff into the above form, and show that upwinding is TVD when it is stable and that Lax-Wendroff is not TVD. Give an interpretation for the meaning of TVD and explain how this relates to the numerical solutions from problem 1b.

Figure 3.58: Problem statement

Given

$$u_j^{n+1} = u_j^n - C\left(u_j^n - u_{j-1}^n\right) + D\left(u_{j+1}^n - u_j^n\right)$$

Writing the above in the following form

$$\begin{aligned} u_j^{n+1} &= Cu_{j-1}^n + (1 - (C + D))u_j^n + Du_{j+1}^n \\ &= \begin{bmatrix} C & (1 - (C + D)) & D \end{bmatrix} \begin{bmatrix} u_{j-1}^n \\ u_j^n \\ u_{j+1}^n \end{bmatrix} \\ &= A \begin{bmatrix} u_{j-1}^n \\ u_j^n \\ u_{j+1}^n \end{bmatrix} \end{aligned} \tag{1}$$

Doing the same for u_{j-1}^{n+1} results in

$$u_{j-1}^{n+1} = A \begin{bmatrix} u_{j-2}^n \\ u_{j-1}^n \\ u_j^n \end{bmatrix} \tag{2}$$

Using the above gives

$$\begin{aligned}\sum_{j=-\infty}^{j=\infty}\left|u_j^{n+1}-u_{j-1}^{n+1}\right| &= \sum_{j=-\infty}^{j=\infty}\left|A\begin{pmatrix}u_{j-1}^n\\ u_j^n\\ u_{j+1}^n\end{pmatrix}-A\begin{pmatrix}u_{j-2}^n\\ u_{j-1}^n\\ u_j^n\end{pmatrix}\right| \\ &= \sum_{j=-\infty}^{j=\infty}\left|A\begin{pmatrix}u_{j-1}^n-u_{j-2}^n\\ u_j^n-u_{j-1}^n\\ u_{j+1}^n-u_j^n\end{pmatrix}\right| \\ &= \sum_{j=-\infty}^{j=\infty}\left|C\left(u_{j-1}^n-u_{j-2}^n\right)+(1-C-D)\left(u_j^n-u_{j-1}^n\right)+D\left(u_{j+1}^n-u_j^n\right)\right|\end{aligned}$$

Using the relation that $\sum|A+B| \le \sum(|A|+|B|)$, the above becomes

$$\sum_{j=-\infty}^{j=\infty}\left|u_j^{n+1}-u_{j-1}^{n+1}\right| \le \sum_{j=-\infty}^{j=\infty}\left|C\left(u_{j-1}^n-u_{j-2}^n\right)\right|+\sum_{j=-\infty}^{j=\infty}\left|(1-C-D)\left(u_j^n-u_{j-1}^n\right)\right|+\sum_{j=-\infty}^{j=\infty}\left|D\left(u_{j+1}^n-u_j^n\right)\right|$$

Given that that $C \ge 0, D \ge 0$ and that $(1-C-D) > 0$, where the last case follows from $(C+D) \le 1$, therefore C, D and $(1-C-D)$ can be taken from outside the absolute sign in the above expression leading to

$$\sum_{j=-\infty}^{j=\infty}\left|u_j^{n+1}-u_{j-1}^{n+1}\right| \le C\sum_{j=-\infty}^{j=\infty}\left|u_{j-1}^n-u_{j-2}^n\right|+(1-C-D)\sum_{j=-\infty}^{j=\infty}\left|u_j^n-u_{j-1}^n\right|+D\sum_{j=-\infty}^{j=\infty}\left|u_{j+1}^n-u_j^n\right|$$

Collecting terms with the same coefficient gives

$$\begin{aligned}\sum_{j=-\infty}^{j=\infty}\left|u_j^{n+1}-u_{j-1}^{n+1}\right| \le C&\left(\sum_{j=-\infty}^{j=\infty}\left|u_{j-1}^n-u_{j-2}^n\right|-\sum_{j=-\infty}^{j=\infty}\left|u_j^n-u_{j-1}^n\right|\right) \\ &+D\left(\sum_{j=-\infty}^{j=\infty}\left|\left(u_{j+1}^n-u_j^n\right)\right|-\sum_{j=-\infty}^{j=\infty}\left|\left(u_j^n-u_{j-1}^n\right)\right|\right) \\ &+\sum_{j=-\infty}^{j=\infty}\left|\left(u_j^n-u_{j-1}^n\right)\right|\end{aligned} \tag{3}$$

The first 2 expressions above in the RHS vanish, leading to the result required. To show this, Consider the first expression from the RHS above, and expanding it on the real line gives

$$C\left(\sum_{j=-\infty}^{j=\infty}\left|u_{j-1}^n-u_{j-2}^n\right|-\sum_{j=-\infty}^{j=\infty}\left|u_j^n-u_{j-1}^n\right|\right)=$$

$$\left(\cdots+\left|\left(u_{-2}^n-u_{-3}^n\right)\right|+\left|u_{-1}^n-u_{-2}^n\right|+\left|u_0^n-u_{-1}^n\right|+\left|u_1^n-u_0^n\right|+\left|u_2^n-u_1^n\right|+\left|u_3^n-u_2^n\right|+\left|u_4^n-u_3^n\right|+\cdots\right)-$$

$$\left(\cdots+\left|u_{-2}^n-u_{-3}^n\right|+\left|u_{-1}^n-u_{-2}^n\right|+\left|u_0^n-u_{-1}^n\right|+\left|u_1^n-u_0^n\right|+\left|u_2^n-u_1^n\right|+\left|u_3^n-u_2^n\right|+\left|u_4^n-u_3^n\right|+\cdots\right)$$

The above result shows that all terms cancel out. Each term in the first line above, has a corresponding term in the second line, but with a negative sign. Therefore

$$C\left(\sum_{j=-\infty}^{j=\infty}\left|u_{j-1}^n-u_{j-2}^n\right|-\sum_{j=-\infty}^{j=\infty}\left|u_j^n-u_{j-1}^n\right|\right)=0 \tag{4}$$

Similarly the following term vanish as well

$$D\left(\sum_{j=-\infty}^{j=\infty}\left|\left(u_{j+1}^n-u_j^n\right)\right|-\sum_{j=-\infty}^{j=\infty}\left|\left(u_j^n-u_{j-1}^n\right)\right|\right)=$$
$$\left(\cdots+\left|\left(u_{-2}^n-u_{-3}^n\right)\right|+\left|u_{-1}^n-u_{-2}^n\right|+\left|u_0^n-u_{-1}^n\right|+\left|u_1^n-u_0^n\right|+\left|u_2^n-u_1^n\right|+\left|u_3^n-u_2^n\right|+\left|u_4^n-u_3^n\right|\cdots\right)-$$
$$\left(\cdots+\left|u_{-2}^n-u_{-3}^n\right|+\left|u_{-1}^n-u_{-2}^n\right|+\left|u_0^n-u_{-1}^n\right|+\left|u_1^n-u_0^n\right|+\left|u_2^n-u_1^n\right|+\left|u_3^n-u_2^n\right|+\left|u_4^n-u_3^n\right|+\cdots\right)$$
$$=0 \tag{5}$$

Substituting Eqs. (4) and (5) into (3) gives

$$\sum_{j=-\infty}^{j=\infty}\left|u_j^{n+1}-u_{j-1}^{n+1}\right|\le\sum_{j=-\infty}^{j=\infty}\left|\left(u_j^n-u_{j-1}^n\right)\right|$$

Which is the result we are asked to show.

3.4.2.1 Second part

Lax-Wendroff is given by

$$\begin{aligned}u_j^{n+1}&=u_j^n-\frac{ak}{2h}\left(u_{j+1}^n-u_{j-1}^n\right)+\frac{a^2k^2}{2h^2}\left(u_{j-1}^n-2u_j^n+u_{j+1}^n\right)\\&=u_j^n\left(1-\frac{a^2k^2}{h^2}\right)+u_{j-1}^n\left(\frac{ak}{2h}+\frac{a^2k^2}{2h^2}\right)+u_{j+1}^n\left(\frac{a^2k^2}{2h^2}-\frac{ak}{2h}\right)\end{aligned} \tag{1}$$

Eq. (1) needs to be put in the following form $u_j^{n+1}=u_j^n-C\left(u_j^n-u_{j-1}^n\right)+D\left(u_{j+1}^n-u_j^n\right)$ or

$$u_j^{n+1}=u_j^n(1-C-D)+Cu_{j-1}^n+Du_{j+1}^n \tag{2}$$

Comparing Eqs. (1) and (2) leads to

$$\begin{aligned}1-C-D&=1-\frac{a^2k^2}{h^2}\\C+D&=\frac{a^2k^2}{h^2}\\C&=\frac{ak}{2h}+\frac{a^2k^2}{2h^2}\\D&=\frac{a^2k^2}{2h^2}-\frac{ak}{2h}=\frac{1}{2}\left(\frac{a^2k^2}{h^2}-\frac{ak}{h}\right)\end{aligned} \tag{3}$$

For the scheme to be TVD it is required that $C\ge 0$ and $D\ge 0$. Lax-Wendroff is stable when $|a|\frac{k}{h}\le 1$. Therefore this implies that $\frac{a^2k^2}{h^2}<|a|\frac{k}{h}$.Hence the constant D in Eq. (3) above will become negative. Therefore one of the conditions of TVD has been violated. Hence Lax-Wendroff is not TVD. Now consider upwind.

Consider the case $a > 0$. Upwind scheme is given by

$$\begin{aligned} u_j^{n+1} &= u_j^n - \frac{ak}{h}\left(u_j^n - u_{j-1}^n\right) \\ &= u_j^n\left(1 - \frac{ak}{h}\right) + \frac{ak}{h}u_{j-1}^n \end{aligned} \tag{4}$$

By comparing coefficients between Eqs. (4) and (2) results in

$$\begin{aligned} 1 - C - D &= 1 - \frac{ak}{h} \\ C + D &= \frac{ak}{h} \\ C &= \frac{ak}{h} \\ D &= 0 \end{aligned} \tag{5}$$

But when $a > 0$, upwind is stable when $0 \le \frac{ak}{h} \le 1$. Therefore $C > 0$ and all the TVD conditions above are now satisfied, Hence upwind is TVD for $a > 0$.

Now consider when $a < 0$, The upwind scheme is now given by

$$\begin{aligned} u_j^{n+1} &= u_j^n - \frac{ak}{h}\left(u_{j+1}^n - u_j^n\right) \\ &= u_j^n\left(1 + \frac{ak}{h}\right) + \left(-\frac{ak}{h}\right)u_{j+1}^n \end{aligned} \tag{6}$$

By comparing coefficients between Eqs. (6) and (2) results in

$$\begin{aligned} 1 - C - D &= 1 + \frac{ak}{h} \\ C + D &= -\frac{ak}{h} \\ C &= 0 \\ D &= -\frac{ak}{h} \end{aligned} \tag{5}$$

since $a < 0$, then upwind is now stable when $-1 \le \frac{ak}{h} \le 0$. Therefore $D > 0$ in the above, and $C + D > 0$ as well, and hence all the TVD conditions are satisfied, therefore upwind is TVD for $a < 0$ as well. Therefore upwind is TVD.

Interpretation of TVD: A scheme with this property implies that the numerical solution, starting with initial data that is monotone, will remain monotone as the solution is advanced in time. This implies that no new local extrema will be created and values of local minimum are nondecreasing while values of local maximum are nonincreasing. In part (b), when initial data was discontinuous, it was observed that Lax-wendroff produced wiggles where

non-existed before, meaning that new local maximum and new local minimum were created in that region. This agrees with the finding here that Lax-Wendroff is not TVD.

On the other hand, with upwind, no new wiggles were created near the discontinuity, and the numerical solution remained monotone. This agrees with the finding here that upwind is TVD scheme. A scheme which is TVD is also stable, since the TVD property will prevent any 'blow up' in the solution due to the above properties of being TVD scheme. TVD scheme is stable, but limited to first order accuracy. To obtain more accuracy and use a second order, the price to pay is that the scheme becomes non TVD.

3.4.3 Problem 3

3. For solving the heat equation we frequently use Crank-Nicolson. For the linear advection equation, Crank-Nicolson is

$$u_j^{n+1} - u_j^n + \frac{\nu}{4}(u_{j+1}^n - u_{j-1}^n) + \frac{\nu}{4}(u_{j+1}^{n+1} - u_{j-1}^{n+1}) = 0.$$

   (a) Show that Crank-Nicolson is unconditionally stable for the advection equation.

   (b) Use von Neumann analysis to show that for a periodic domain $\|u^n\|_2 = \|u^0\|_2$ for all n. This scheme is said to be nondissipative. This seems reasonable because this is a property of the PDE.

   (c) Solve the advection equation on the periodic domain $[0, 1]$ with the initial condition from problem 1b. Show the solution and comment on your results.

Figure 3.59: Problem statement

3.4.3.1 Part (a)

The PDE for linear advection equation is given by

$$u_t + au_x = 0$$

The Crank Nicholson finite difference scheme for the above is

$$u_j^{n+1} - u_j^n + \frac{\nu}{4}\left(u_{j+1}^n - u_{j-1}^n\right) + \frac{\nu}{4}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right) = 0 \tag{1}$$

Where $\nu = \frac{ak}{h}$, k is the time step and h is the space step. Applying Von Neumann stability analysis, let

$$u_j^{n+1} = g(\xi)e^{i\zeta x_j}$$

and let

$$u_j^n = e^{i\zeta x_j}$$

Substituting the above 2 equations into Eq. (1) gives

$$g(\xi)e^{i\zeta x_j} - e^{i\zeta x_j} + \frac{\nu}{4}\left(e^{i\zeta x_j}e^{i\zeta h} - e^{i\zeta x_j}e^{-i\zeta h}\right) + \frac{\nu}{4}\left(g(\xi)e^{i\zeta x_j}e^{i\zeta h} - g(\xi)e^{i\zeta x_j}e^{-i\zeta h}\right) = 0$$

Dividing throughout by $e^{i\zeta x_j}$ gives

$$g(\xi) - 1 + \frac{\nu}{4}\left(e^{i\zeta h} - e^{-i\zeta h}\right) + \frac{\nu}{4} g(\xi)\left(e^{i\zeta h} - e^{-i\zeta h}\right) = 0$$

Solving for $g(\xi)$ and applying Euler relation to convert exponential to trigonometry functions gives

$$g(\xi)\left(1 + \frac{\nu i}{2}\sin \zeta h\right) = 1 - \frac{\nu i}{2}\sin \zeta h$$

$$g(\xi) = \frac{1 - \frac{\nu i}{2}\sin \zeta h}{1 + \frac{\nu i}{2}\sin \zeta h}$$

Hence

$$\left|g(\xi)\right|^2 = \frac{\left|1 - i\frac{\nu}{2}\sin \zeta h\right|^2}{\left|1 + i\frac{\nu}{2}\sin \zeta h\right|^2} = \frac{1 + \frac{\nu^2}{4}\sin^2 \zeta h}{1 + \frac{\nu^2}{4}\sin^2 \zeta h}$$

Hence

$$\left|g(\xi)\right| = 1$$

Since $\left|g(\xi)\right| \leq 1$ the scheme is unconditionally stable[8].The stability of this scheme does not depend on CFL criteria, in other words, there is no dependency on Δt or h for the stability. In addition, it is seen that the amplitude of each Fourier mode remain constant at each time increment. High wave numbered modes as well as small wave numbered modes will remain in the numerical solution with same energy content. There is no dissipation in the numerical solution.

3.4.3.2 part (b)

The following relation, which is valid due to the periodic boundary conditions being used, will be utilized in the proof below

$$\sum_j u_j^n = \sum_j u_{j-1}^n = \sum_j u_{j+1}^n$$

The above relation can be more easily seen by viewing the domain as a ring, where the first grid point is physically the same as the last grid point. Therefore the location of where the sum starts is not important, since the same number of grid points will always be added as long as the sum is over the whole range. The following diagram illustrate this point.

[8]May be we should call this as marginally stable? Since there is no attenuation and also there is no magnification.

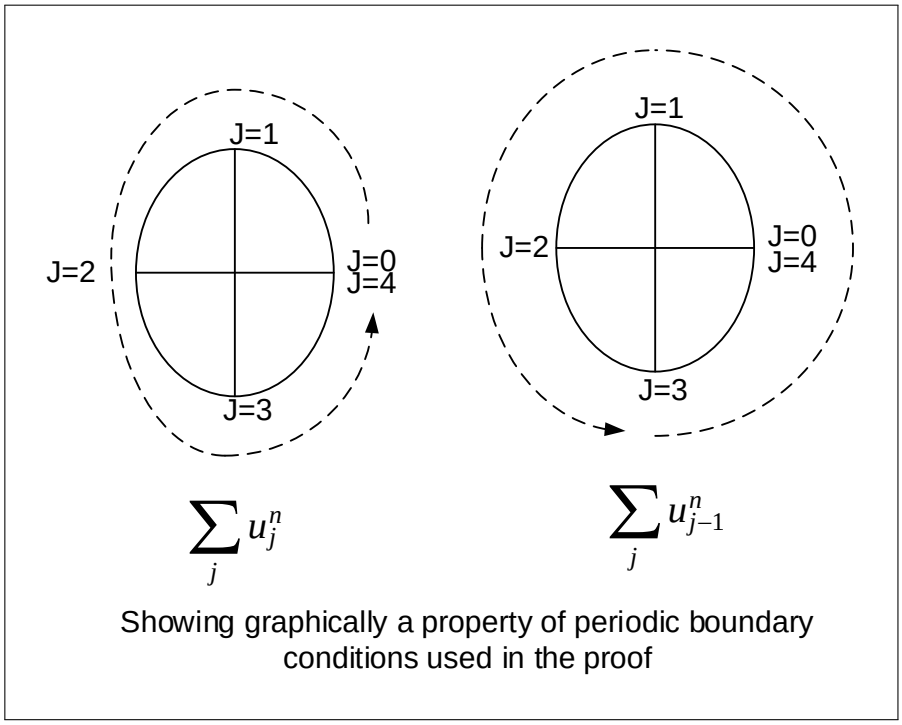


Figure 3.60: Grid layout

Now, the proof will start. Starting from the C-N scheme given by

$$u_j^{n+1} + \frac{\nu}{4}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right) = u_j^n - \frac{\nu}{4}\left(u_{j+1}^n - u_{j-1}^n\right)$$

And squaring each side, and summing over all j gives

$$\sum_j \left[u_j^{n+1} + \frac{\nu}{4}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right)\right]^2 = \sum_j \left[u_j^n - \frac{\nu}{4}\left(u_{j+1}^n - u_{j-1}^n\right)\right]^2$$

Expanding

$$\sum_j \left(u_j^{n+1}\right)^2 + \frac{\nu}{2}u_j^{n+1}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right) + \left(\frac{\nu}{4}\right)^2\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right)^2 =$$

$$\sum_j \left(u_j^n\right)^2 - \frac{\nu}{2}u_j^n\left(u_{j+1}^n - u_{j-1}^n\right) + \left(\frac{\nu}{4}\right)^2\left(u_{j+1}^n - u_{j-1}^n\right)^2$$

Moving all terms from LHS to RHS except for $\sum_j \left(u_j^{n+1}\right)^2$ the above becomes

$$\sum_j \left(u_j^{n+1}\right)^2 = \sum_j \left(u_j^n\right)^2 - \frac{\nu}{2}\left[u_j^n\left(u_{j+1}^n - u_{j-1}^n\right) + u_j^{n+1}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right)\right]$$
$$+ \left(\frac{\nu}{4}\right)^2\left[\left(u_{j+1}^n - u_{j-1}^n\right)^2 - \left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right)^2\right]$$

Using Von Neumann, let $u_j^{n+1} = |g| u_j^n$ in the above, where g is the magnification factor which was found from part (a) to be independent of ξ. The above becomes

$$\sum_j \left(u_j^{n+1}\right)^2 = \sum_j \left(u_j^n\right)^2 - \frac{\nu}{2}\left[u_j^n\left(u_{j+1}^n - u_{j-1}^n\right) + |g|^2 u_j^n\left(u_{j+1}^n - u_{j-1}^n\right)\right] + \left(\frac{\nu}{4}\right)^2\left[\left(u_{j+1}^n - u_{j-1}^n\right)^2 - |g|^2\left(u_{j+1}^n - u_{j-1}^n\right)^2\right]$$

Since $|g| = 1$ as was found in part(a), then the last term in the RHS above will vanish resulting in

$$\begin{aligned}\sum_j \left(u_j^{n+1}\right)^2 &= \sum_j \left(u_j^n\right)^2 - \frac{\nu}{2}\left[u_j^n\left(u_{j+1}^n - u_{j-1}^n\right) + u_j^n\left(u_{j+1}^n - u_{j-1}^n\right)\right] \\ &= \sum_j \left(u_j^n\right)^2 - \nu\ u_j^n\left(u_{j+1}^n - u_{j-1}^n\right) \\ &= \sum_j \left(u_j^n\right)^2 + \nu\left(\sum_j u_j^n u_{j-1}^n - \sum_j u_j^n u_{j+1}^n\right) \end{aligned} \tag{1}$$

Due to the periodic boundary[9] conditions $\sum_j u_j^n u_{j-1}^n = \sum_j u_j^n u_{j+1}^n$, Therefore Eq. (1) reduces to

$$\sum_j \left(u_j^{n+1}\right)^2 = \sum_j \left(u_j^n\right)^2 \tag{2}$$

But by definition

$$\|u\|_2^2 = \sum_j u^2$$

Hence Eq. (2) can be written as

$$\left\|u^{n+1}\right\|_2^2 = \|u^n\|_2^2$$

or

$$\left\|u^{n+1}\right\|_2 = \|u^n\|_2 \tag{3}$$

Similarly $\|u^n\|_2 = \left\|u^{n-1}\right\|_2 = \cdots = \left\|u^0\right\|_2$, therefore Eq. (3) becomes

$$\left\|u^{n+1}\right\|_2 = \left\|u^0\right\|_2$$

[9]Side note: Initially I thought I might have to use the Schawrz inequality $|u \cdot v| \le \|u\|\|v\|$, to write

$$\sum_j u_j^n u_{j+1}^n \le \sqrt{\sum_j \left(u_j^n\right)^2}\sqrt{\sum_j \left(u_{j+1}^n\right)^2}$$

And due to periodic boundary conditions, obtain $\sum_j \left(u_j^n\right)^2 = \sum_j \left(u_{j+1}^n\right)^2$, and so $\sum_j u_j^n u_{j+1}^n \le \sum_j \left(u_j^n\right)^2$ But this turned out not to be required.

3.4.3.3 Part(c)

The C-N scheme was implemented for the 1D advection PDE. The source code is shown in the appendix. The following diagram shows the result for the initial conditions as given in part (b). The result for C-N is shown next to the solution produced by Lax-Wendroff in order to compare the results

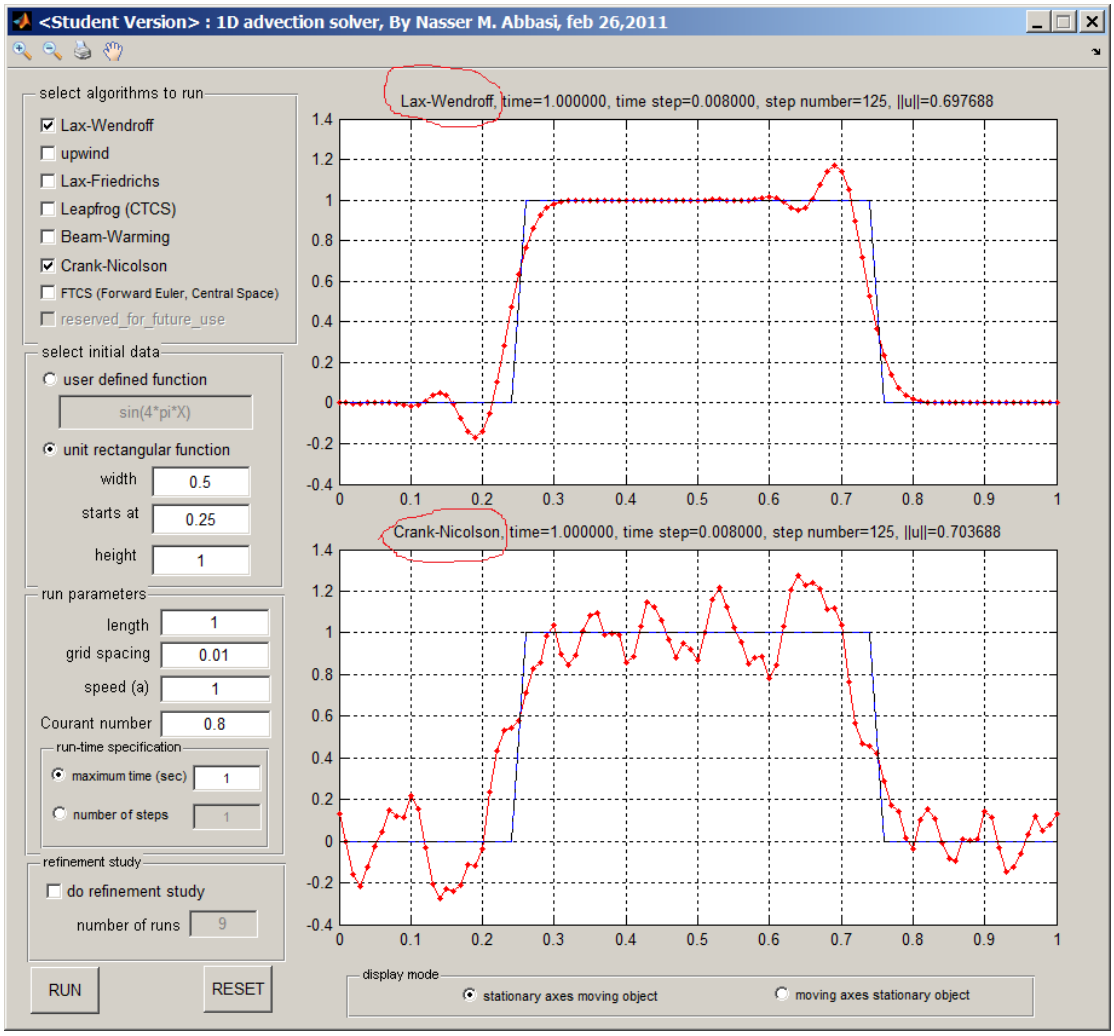


Figure 3.61: C-N result

C-N shows more wiggles near the boundaries than Lax-Wendroff. Both are not TVD schemes, so starting with non-smooth initial data, it was expected to see wiggles in both cases.

C-N scheme showed more wiggles and they appeared earlier in time as well, even though the solution was stable all the time, since these did not grow when the running time was made longer (It was shown in part(a) that the scheme is unconditionally stable). Norm-2 of the solution was displayed all the time and it remained the same value through the run-time, even as wiggles appeared in the solution. This was the case for both schemes shown.

This result shows that 2-norm of the numerical solution is stable as the 2-norm of the numerical solution did not grow with time. The scheme is non-dissipative at all, and high frequency modes did not attenuate, leading the solution observed. Using such non-dissipative schemes on non-smooth data does not appear to be a good idea. The following diagram below is another illustration to compare Lax-Wendroff with C-N with $h = 0.005$, showing the numerical solution at $t = 0.04$ seconds.

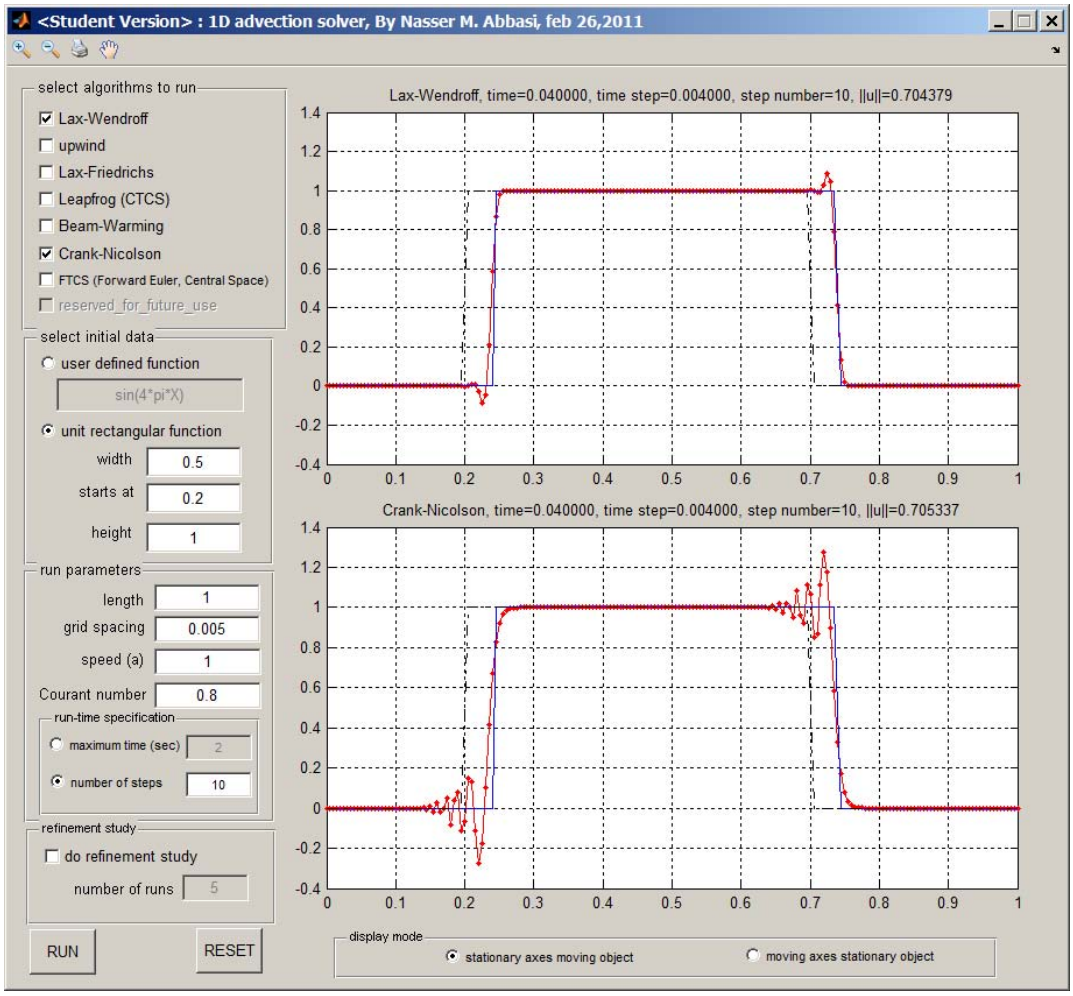


Figure 3.62: compare Lax-Wendroff with C-N with $h = 0.005$

The top diagram shows Lax-Wendroff, and the bottom one shows C-N. Notice that wiggles are larger in amplitude for C-N since its magnification factor is constant at 1, resulting in no attenuation at all in the large spatial frequency present in the initial data.

3.5 HW 4

3.5.1 Problem 1

Math 228B
Homework 4
Due Thursday, 3/17/11.

1. In one spatial dimension the linearized equations of acoustics (sound waves) are

$$p_t + Ku_x = 0$$
$$\rho u_t + p_x = 0,$$

where u is the velocity and p is the pressure, ρ is the density, and K is the bulk modulus of compressibility.

(a) Show that this system is hyperbolic and find the wave speeds.

(b) Write a program to solve this system using Lax-Wendroff in original variables on $(0, 1)$ using a cell centered grid $x_j = (j - 1/2)h$ for $j = 1 \dots N$. Write the code to use ghost cells, so that different boundary conditions can be changed by simply changing the values in the ghost cells.

Set the ghost cells at the left by

$$p_0^n = p_1^n$$
$$u_0^n = -u_1^n,$$

and set the ghost cells on the right by

$$p_{N+1}^n = \frac{1}{2}\left(p_N^n + u_N^n\sqrt{K\rho}\right)$$
$$u_{N+1}^n = \frac{1}{2}\left(\frac{p_N^n}{\sqrt{K\rho}} + u_N^n\right).$$

Run simulations with different initial conditions. Explain what happens at the left and right boundaries.

(c) Give a physical interpretation and a mathematical explanation of these boundary conditions.

Figure 3.63: Problem description

3.5.1.1 part(a)

The definitions and physical units of the variables used in the PDE's are given below. In the following table, L stands for length, T for time, M for mass and N for force.

term	meaning	dimensions	SI units
p	acoustic air pressure in medium	$\frac{N}{L^2}$ or $\frac{ML}{T^2}\frac{1}{L^2}$ or $\frac{M}{LT^2}$	$N/Meter^2$
u	acoustic perturbation velocity	L/T	$Meter/Second$
c	speed of sound in medium	L/T	$Meter/Second$
K	bulk modulus or modulus of bulk elasticity for gas[10]	$\frac{M}{T^2L}$	kg per $meter$ per $second^2$
ρ	air density	M/L^3	$kg/meter^3$

To show that the system is hyperbolic, the PDE's are written in matrix form

$$p_t + Ku_x = 0$$
$$\rho u_t + p_x = 0$$

Therefore

$$\begin{pmatrix} p \\ u \end{pmatrix}_t + \overbrace{\begin{pmatrix} 0 & K \\ 1/\rho & 0 \end{pmatrix}}^{A} \begin{pmatrix} p \\ u \end{pmatrix}_x = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$$\mathbf{q}_t + A\mathbf{q}_x = \mathbf{0}$$

If the eigenvalues of A are real and distinct, implying the existence of linearly independent eigenvectors for A, then the system is called strictly hyperbolic[11]. The eigenvalues of A are found by solving the following equation

$$Det(A - \lambda \mathbf{I}) = 0$$
$$(-\lambda)(-\lambda) - (k)\left(1/\rho\right) = 0$$
$$\lambda^2 = \frac{k}{\rho}$$
$$\lambda_{1,2} = \pm\sqrt{\frac{k}{\rho}}$$

The quantity $\frac{k}{\rho}$is positive and real because ρ is density (which is a real positive number) and k is bulk modulus of compressibility which is also real positive number.

Therefore both eigenvalues of A are real and distinct. Hence the system is strictly hyperbolic. The system is diagonalizable as well, since the transpose of A is a diagonal matrix, but this property was not needed to show the system is hyperbolic. The speed of sound in the medium is given by $\sqrt{\frac{k}{\rho}}$. Hence a sound wave will travel in one direction at speed $\sqrt{\frac{k}{\rho}}$ and another sound wave will travel in the same speed but in the opposite direction.

[11] Another method to show that the system is hyperbolic, is to show that A is real and symmetric, because this implies that A is diagonalizable. In this case, the system is called symmetric hyperbolic.

3.5.1.2 Part (b)

The following diagram illustrates the grid numbering used in the numerical solution

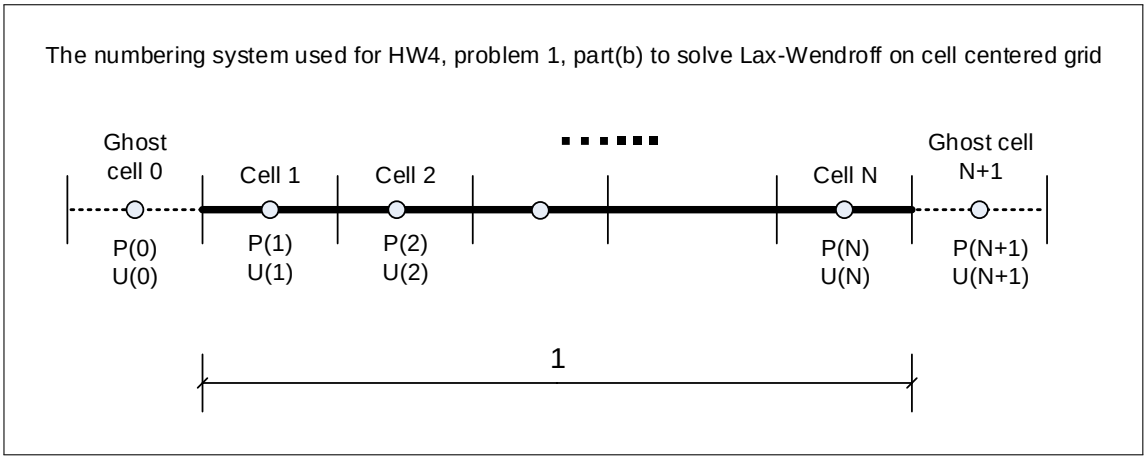


Figure 3.64: Grid used

The Lax-Wendroff scheme for the linear system $\mathbf{q}_t + A\mathbf{q}_x = \mathbf{0}$ is given by

$$\mathbf{q}_j^n + 1 = \mathbf{q}_j^n - \frac{\Delta t}{2h} A\left(\mathbf{q}_{j+1}^n - \mathbf{q}_{j-1}^n\right) + \frac{\Delta t^2}{2h^2} A^2\left(\mathbf{q}_{j-1}^n - 2\mathbf{q}_j^n + \mathbf{q}_{j+1}^n\right)$$

Where $A = \begin{pmatrix} 0 & K \\ 1/\rho & 0 \end{pmatrix}$ is a constant matrix.

In this problem, the solution at time n is

$$\mathbf{q}_j^n = \begin{pmatrix} p \\ u \end{pmatrix}_j^n = \mathbf{q}_j^n = \begin{pmatrix} p_j^n \\ u_j^n \end{pmatrix}$$

The following are the boundary conditions used

$$\mathbf{q}_0^n = \begin{pmatrix} p \\ u \end{pmatrix}_0^n = \begin{pmatrix} p_1^n \\ u_1^n \end{pmatrix}$$

$$\mathbf{q}_{N+1}^n = \begin{pmatrix} p \\ u \end{pmatrix}_{N+1}^n = \frac{1}{2}\begin{pmatrix} p_N^n + u_N^n \sqrt{k\rho} \\ \frac{p_N^n}{\sqrt{k\rho}} + u_N^n \end{pmatrix}$$

To find the time step Δt, Courant number $r = 0.8$ was used[12], and Δt found by using the CFL condition

$$r = \left| \frac{\Delta t}{h} \lambda \right|$$

[12]For stability, the Courant number must be less than 1.

Solving for Δt gives

$$\Delta t = \frac{rh}{|\lambda|}$$

The solution was implemented in Matlab and the result is given below. For each run, a number of plots are shown to illustrate the solution at different time instances. The following table describes the simulations done. Three different initial conditions are used with two different runs for each initial condition. The first run used the boundary conditions given in this problem, and the second run used different boundary conditions which caused the sound wave to reflect when it reached both the left and the right boundaries, and not just the left boundary. Therefore a total of 6 simulations were made, the first three used the following boundary conditions

$$\begin{aligned} p_0^n &= p_1^n \\ u_0^n &= -u_1^n \\ p_{N+1}^n &= \frac{1}{2}\left(p_N^n + u_N^n \sqrt{k\rho}\right) \\ u_{N+1}^n &= \frac{1}{2}\left(\frac{p_N^n}{\sqrt{k\rho}} + u_N^n\right) \end{aligned}$$

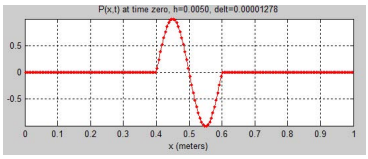
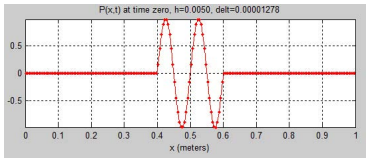
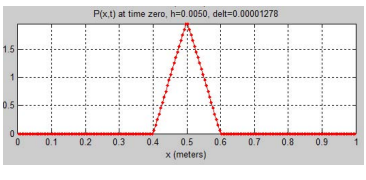
And the second three simulations used the following boundary conditions

$$\begin{aligned} p_0^n &= p_1^n \\ u_0^n &= -u_1^n \\ p_{N+1}^n &= p_N^n \\ u_{N+1}^n &= -u_N^n \end{aligned}$$

The images below show the three initial conditions for the pressure $p(x,0)$. The initial velocity $u(x,0)$ was set to zero for all simulations. The following section shows the simulation plots for each one of the 6 simulations. All snapshots were taken at the same time for each run in order to compare the results. All runs were made with the following parameters:

$$\begin{aligned} h &= 0.005 \ meter \\ \Delta t &= 0.1278 \ ms \\ Courant\ number &= 0.8 \\ maximum\ run\ time &= 0.005 \ \text{sec} \end{aligned}$$

Animations of these runs are available above (in HTML version only).

$\sin(10\pi x)$ from $x = 0.4$ to $x = 0.6$	$\sin(20\pi x)$ from $x = 0.4$ to $x = 0.6$	triangle function
		

Simulation using first initial data and reflect from left end only

This simulation used $p(x,0) = \sin(10\pi x)$ from $x = 0.4$ to $x = 0.6$. The pressure wave starts in the middle, and immediately starts to split into two smaller waves, each one became half the amplitude of the original wave. Each smaller wave traveled in opposite directions. The wave that reached the left boundary was reflected back while the wave that reached the right boundary was absorbed into the boundary. After the left wave reflected back and eventually reached the right boundary, it was also absorbed. This resulted in the original wave disappearing. As the left wave reflected from the left end, it also flipped upside down, such that the leading half of the wave remained with positive amplitude and the trailing half remained with the negative amplitude.

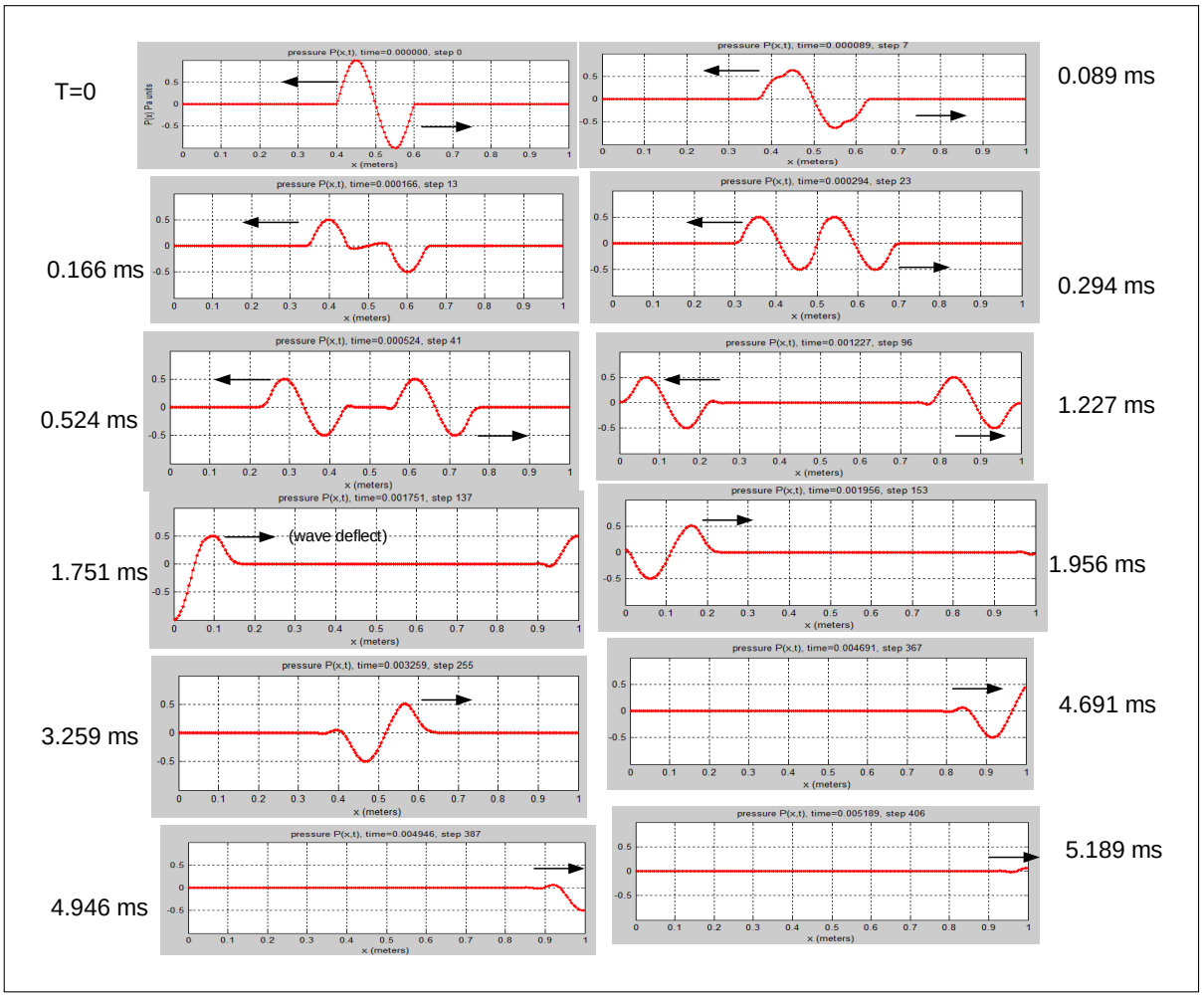


Figure 3.65: test typ0 BC 1

Simulation using second initial data and reflect from left end only

These images show the simulation result using $p(x,0) = \sin(20\pi x)$ from $x = 0.4$ to $x = 0.6$. Each frame is taken at the same time as the first simulation. The same result can be seen as described in the first simulation.

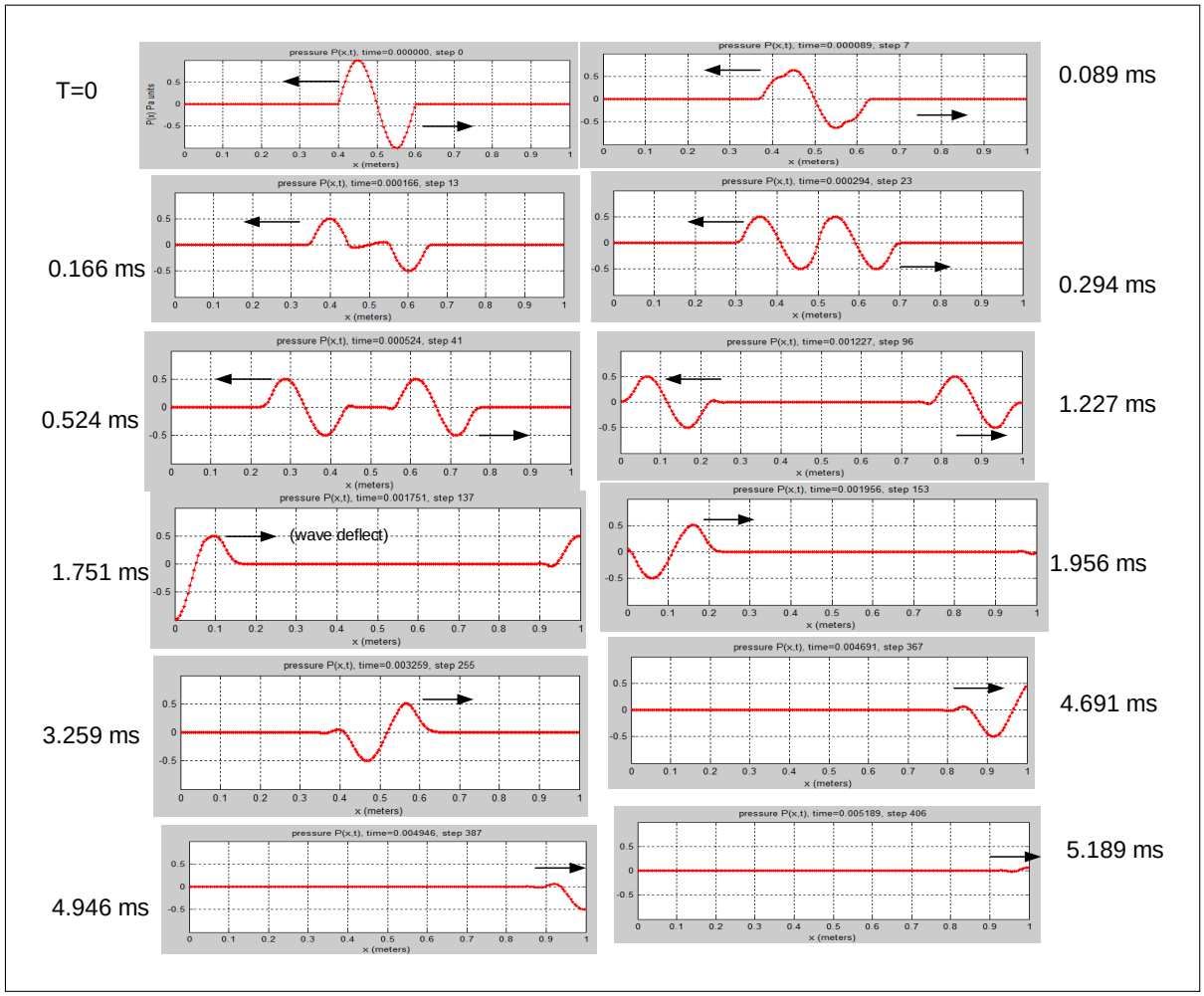


Figure 3.66: test typ2 BC 1

Simulation using third initial data and reflect from left end only

This simulation uses the triangle pulse as the initial data. Each frame is taken at the same time as the first simulation. The same result can be seen as was described in the first simulation.

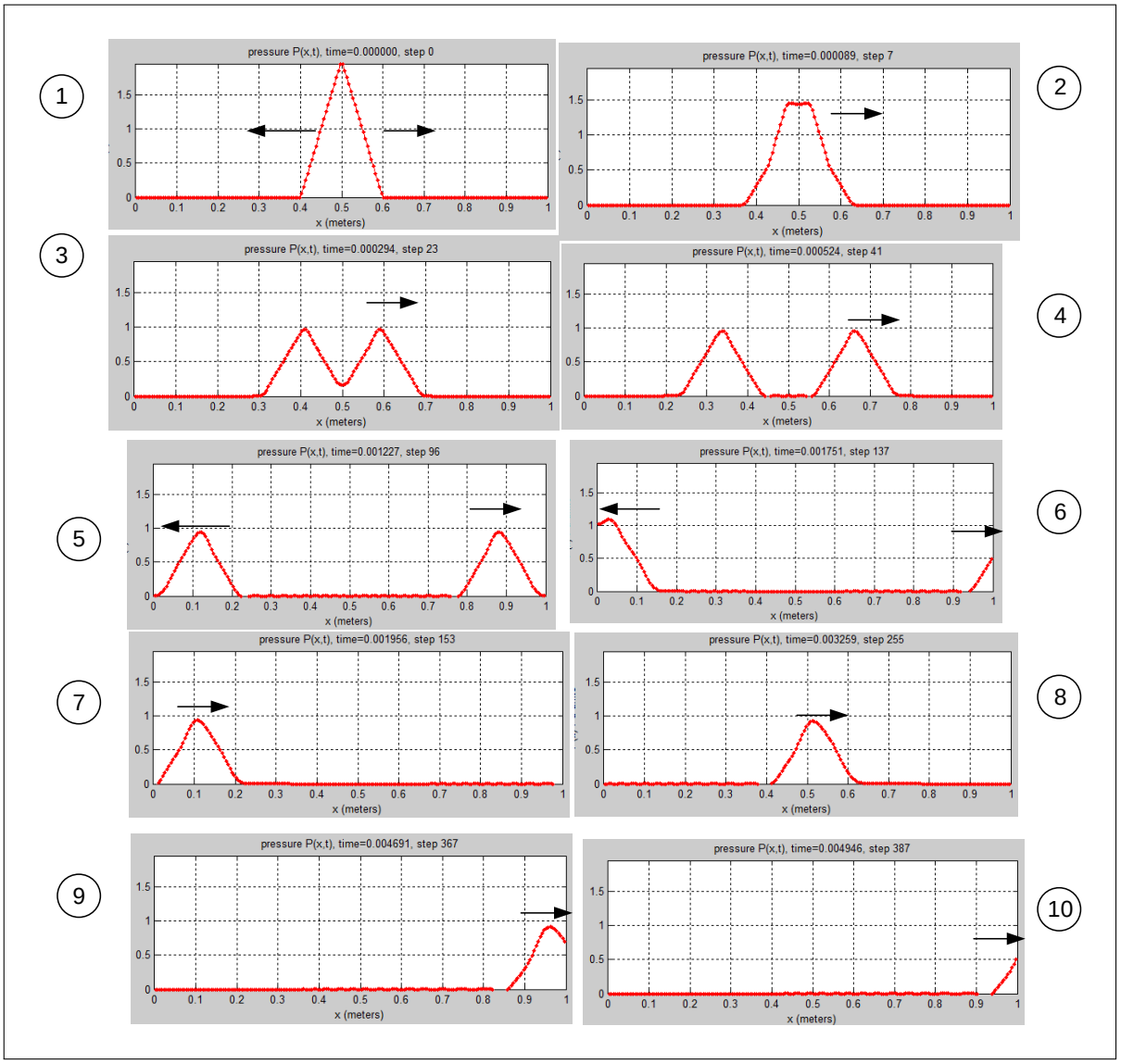


Figure 3.67: test type 4 BC 1

Simulation using first initial data and reflecting from both ends

The following 3 simulations are a repeat of the first 3, but using boundary conditions that caused the pressure wave to reflect from both the left and the right boundaries. This resulted in the wave reflecting back and forth all the time. When both waves met again at the middle, the original wave form was reconstructed for a very short time but in an upside down form compared to its original form, and then the whole cycle was repeated. When the waves met again for the second time in the middle, the original wave was reconstructed again, but this time with the same shape it was at the initial time. This process continued again. Since there was no diffusion term present in the PDE, this cycle repeated for the duration of the simulation and no energy was lost. The times of each frame is the same as was used in all the previous simulations.

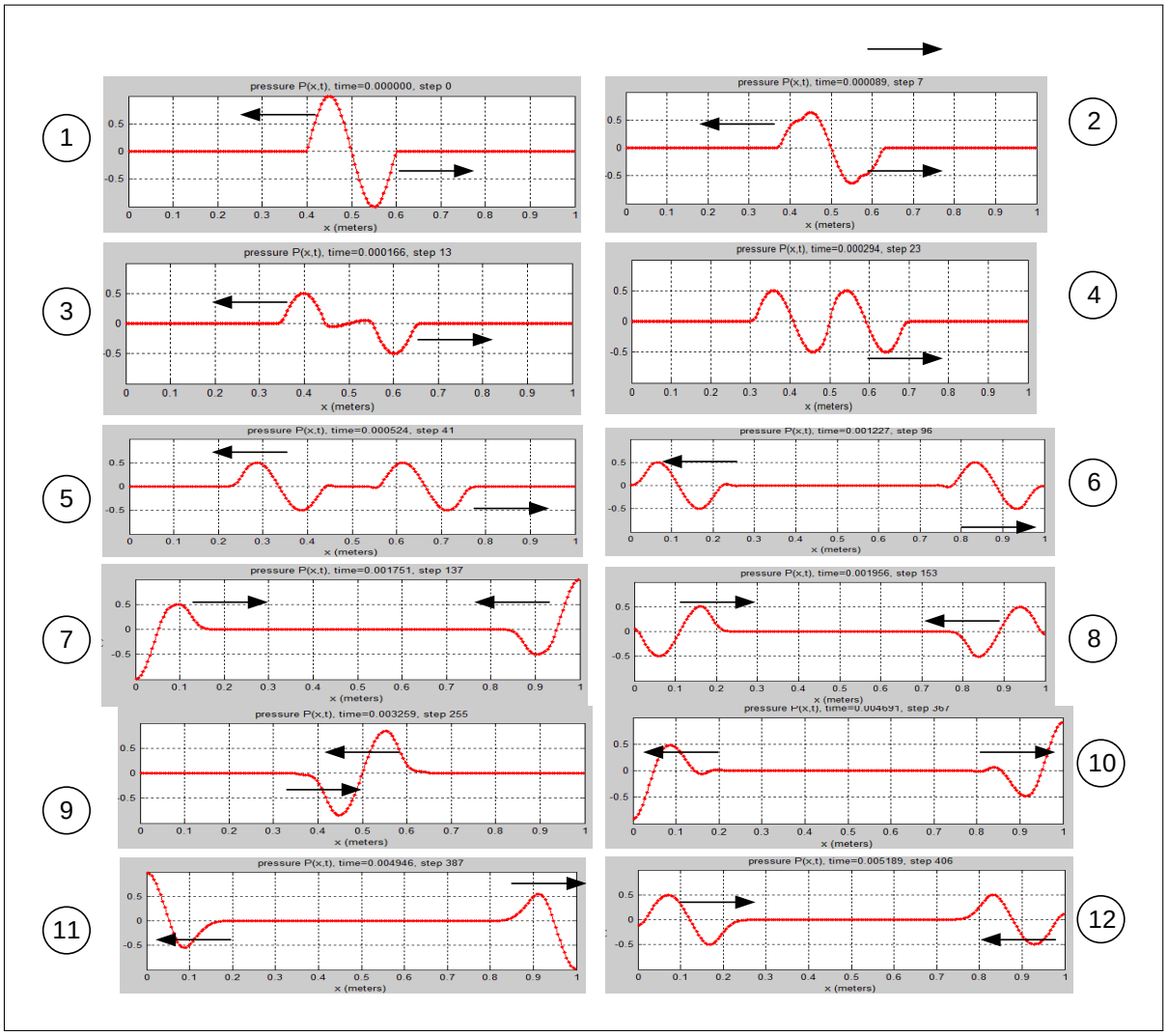


Figure 3.68: test type 0 BC 2

Simulation using second initial data and reflect from both ends

This simulation used $p(x,0) = \sin(20\pi x)$ from $x = 0.4$ to $x = 0.6$, but using boundary conditions that caused the pressure wave to reflect from both the left and the right boundaries. The same observation can be made as with the previous simulation.

Figure 3.69: test type 2 BC 2

Simulation using third initial data and reflect from both ends

This simulation used a triangle pressure wave as its initial data but using boundary conditions that caused the pressure wave to reflect from both the left and the right boundaries. The same observation can be made as with the previous simulation.

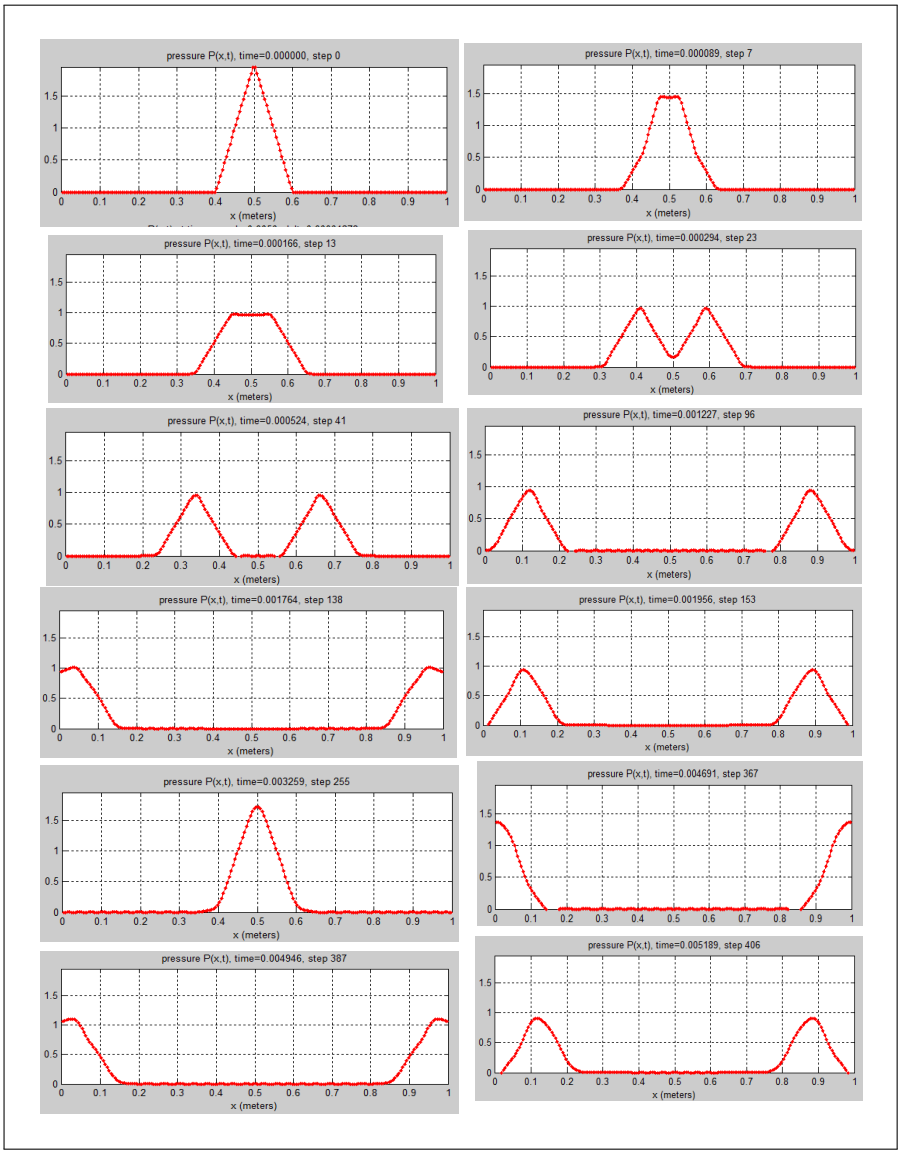


Figure 3.70: test type 4 BC 2

3.5.1.3 Part(c)

The boundary conditions given in the problem are

$$
\begin{aligned}
p_0^n &= p_1^n \\
u_0^n &= -u_1^n \\
p_{N+1}^n &= \frac{1}{2}\left(p_N^n + u_N^n \sqrt{k\rho}\right) \\
u_{N+1}^n &= \frac{1}{2}\left(\frac{p_N^n}{\sqrt{k\rho}} + u_N^n\right)
\end{aligned}
$$

At the left most cell (cell 0), the acoustic perturbation velocity u is negative its value on the inside cell, therefore the average value of u right at the left edge (start of the physical

domain) will be zero, as shown by the following diagram

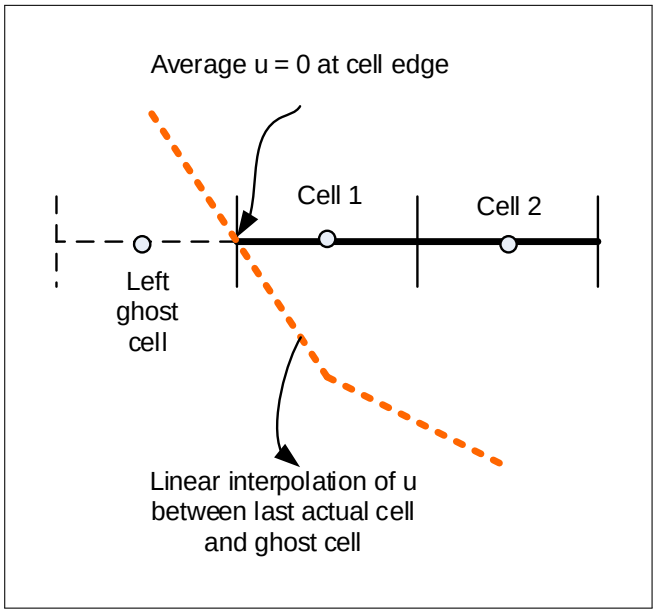


Figure 3.71: problem 1 left cell

Physically, this represent a barrier or a wall where perturbation velocity is zero at the wall resulting in deflection. Having zero velocity at the edge means that the momentum of the wave is zero at the left boundary. Since momentum is conserved, then it must have a direction which is opposite to what it was in the previous time step. This is similar to a ball hitting a perfectly elastic wall. For the pressure boundary conditions, having the acoustic pressure in the left most cell and the ghost cell being the same means that the pressure drop or gradient is zero between these two cells. Therefore, no sound will be transmitted through the boundary since sound is transmitted only due to presence of a pressure gradient between adjacent spatial points in the medium.

On the right side, when taking the average between the right-most cell and the ghost cell at the right results in

$$u_{right\_edge} = \frac{3}{4}u_N^n + \frac{1}{4}\frac{p_N^n}{\sqrt{k\rho}}$$
$$p_{right\_edge} = \frac{3}{4}p_N^n + \frac{1}{4}u_N^n\sqrt{k\rho}$$

Therefore, the perturbation velocity u at the right edge is no longer zero, but it has the same sign as the velocity at the right most cell. Physically this means the acoustic wave will continue to have momentum in the same direction and will not reflect. For the pressure, there exists now a pressure gradient, therefore sound will travel across the right boundary. Physically, this boundary can be thought of as a sound absorbing wall. (For example, a wall treated with special paint or covering).

3.5.2 Problem 2

2. A scheme is monotone preserving if the solution, u_j^n, is monotone in j for all n whenever the initial condition, u_j^0, is monotone in j. Show that if a scheme is TVD, then it is monotone preserving. Assume that the domain is the whole real line, that the solution satisfies the asymptotic boundary conditions $\lim_{j\to\pm\infty} u_j^n = U_{\pm\infty}$, and that the initial condition has bounded variation.

Figure 3.72: Problem statement

Given a sequence u_j^0 which is monotone in j, we need to show that when a TVD scheme is applied to this sequence, the resulting sequence u_j^n is also monotone at any n. This is the same as saying that a TVD is monotone preserving.

We are given that the sequence u_j^n has the fixed boundary conditions at $j = \pm\infty$ for any n.

A monotone sequence can be either monotone increasing or monotone decreasing but not both. A monotone increasing sequence u is one where $u_j \le u_{j+k}$ for any j and for any $k > j$. A monotone decreasing sequence is one where $u_j \ge u_{j+k}$ for any j and for any $k > j$. In the following discussion, a monotone sequence is taken to mean either an increasing or a decreasing sequence.

The following diagram illustrates this point. In this diagram the scheme is viewed as a system or an operator which transforms a sequence to a new sequence. We need to show that this transformation is monotone preserving when the operator is the TVD scheme.

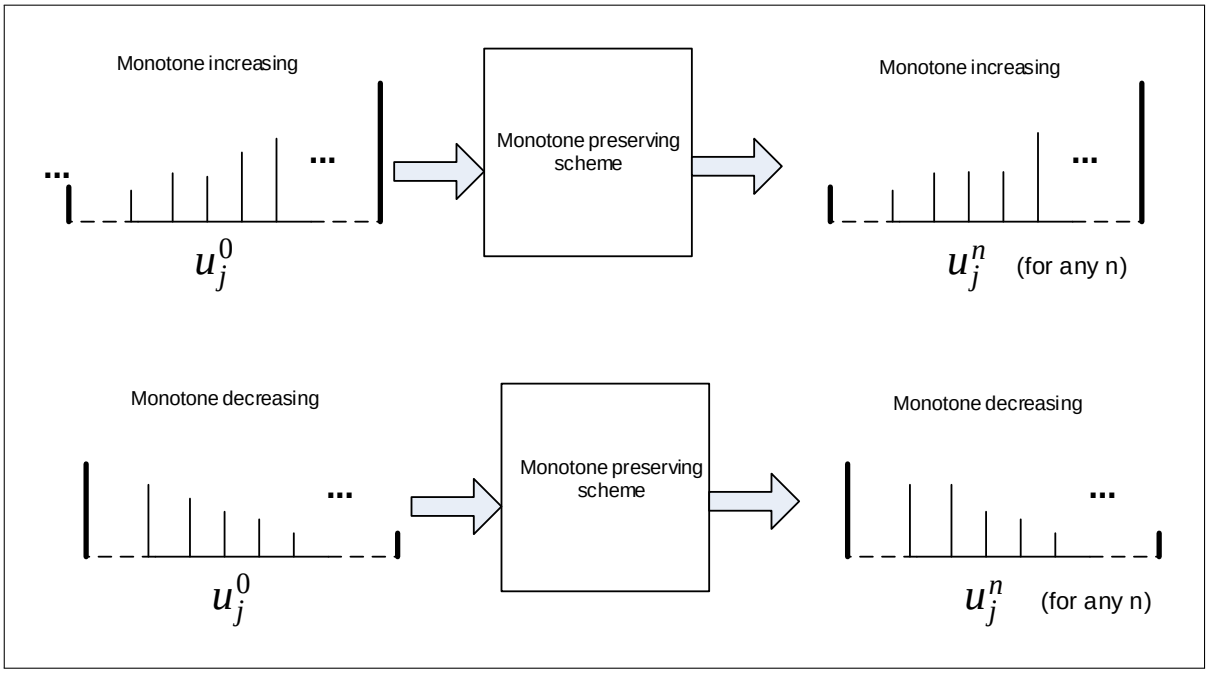


Figure 3.73: TVD 1 scheme

Since u_j^0 (the initial sequence) can be assumed to be monotone, then the total variation of u_j^0 is known, which is

$$TV\left(u_j^0\right) = |U_{+\infty} - U_{-\infty}|$$

The total variation is defined as the sum of the total amount the sequence change (in absolute values). In other words, the TV of the initial sequence is

$$\begin{aligned} TV\left(u^0\right) &= \sum_j \left|u_j^0 - u_{j-1}^0\right| \\ &= |U_{+\infty} - U_{-\infty}| \end{aligned}$$

$TV\left(u_j^0\right) = |U_{+\infty} - U_{-\infty}|$ is valid since u^0 is monotone. We could not have said this if u^0 was not monotone. The following diagram helps illustrate why this is the case, showing a monotone sequence, and showing that adding all the differences between successive values in the sequence is the same as the difference between the left-most value and the right-most values (in absolute values).

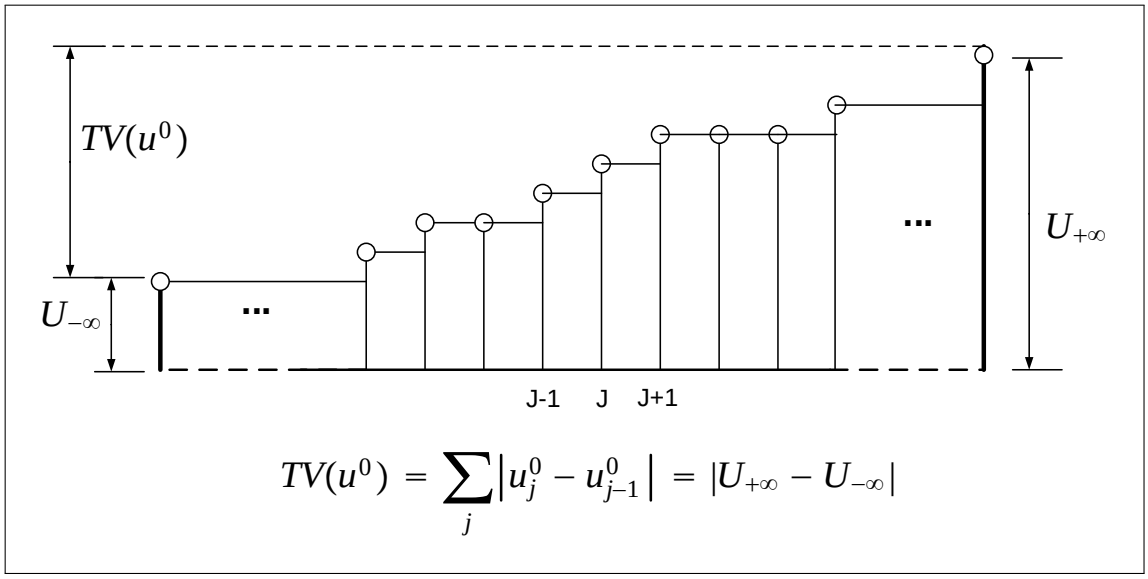


Figure 3.74: TVD 2 scheme

The above is similar to walking up a staircase. If we are told that each step could only go up (or remain flat), then the total height of the overall staircase is the total variation, which is the sum of the height difference between each 2 successive steps.

We know that a TVD scheme, by definition, is one in which satisfies the following relation for any n

$$TV(u^n) \leq TV\left(u^0\right) \tag{1}$$

We now need to show, that when u^0 is monotone, then u^n will also be monotone when applied to a TVD scheme.

The proof will be by contradiction. The idea is to assume that the scheme is TVD, hence Eq. (1) is true, and then to assume that the scheme, when applied to an initial monotone sequence u^0 has resulted in a sequence u^n which is no longer monotone. Then we show that this result is a contradiction to the assumption, meaning that u^n must be monotone.

The following proof below is for a monotone increasing sequence u^0, but the same idea of the proof can be used for a monotone decreasing sequence.

Proof

Let the scheme be TVD, therefore $TV\left(u^0\right) \leq TV(u^n)$, and let a monotone increasing sequence be u_j^0 with a total variation $TV\left(u^0\right) = \Delta$, where Δ is some constant that does not change with n. In this problem this constant is given as $|U_{+\infty} - U_{-\infty}|$.

Let result of applying the TVD scheme to u_j^0 be the sequence u_j^n. Now, assume that u_j^n is no longer a monotone increasing sequence. Since u_j^n is not monotone sequence, it must contain at least one local minimum and/or one local maximum. To illustrate this in a diagram,

assume u_j^n had one local minimum. The same idea would apply if we assumed a local maximum.

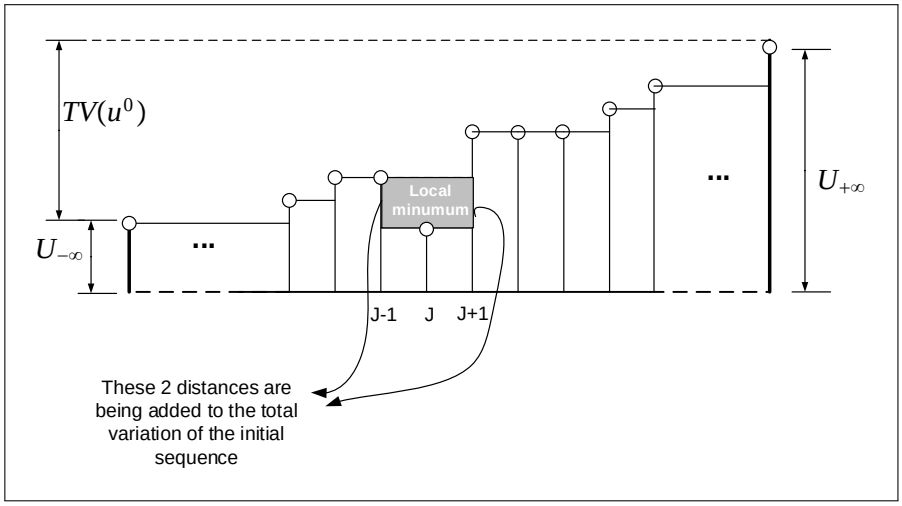


Figure 3.75: TVD 3 scheme

Since u^n has a local minimum, then the total variation of u_j^n is now larger than the total variation of what it would had been if it did not have this local minimum. In the above diagram, u_j^n is shown as being monotone increasing, except for the one local minimum which appeared as a result of applying the TVD scheme.

Due to the presence of this local minimum, the total variation has become larger than $|U_{+\infty} - U_{-\infty}|$. The extra amount added to $TV\left(u^0\right)$ is seen as $2\left|u_j^n - u_{j-1}^n\right|$, as this is the distance needed to be traversed in going down the local minimum and climbing back up the same level before meeting this local minimum.

Therefore, having a local minimum (or a local maximum) in a sequence increases its total variation. Therefore $TV(u^n) > TV\left(u^0\right)$. However, we started by assuming that the scheme is TVD, which means that $TV(u^n) \leq TV\left(u^0\right)$, so this result is a contradiction to our assumption.

Therefore u_j^n can not be a non monotone sequence, hence it must be a monotone sequence. This completes the proof.

3.5.3 Problem 3

3. Write a program to solve the linear advection equation,

$$u_t + au_x = 0,$$

on the unit interval using a finite volume method of the form

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{h}\left(F_{j+1/2} - F_{j-1/2}\right).$$

Use the numerical flux function

$$F_{j-1/2} = F^{\text{up}}_{j-1/2} + \frac{|a|}{2}\left(1 - \left|\frac{a\Delta t}{h}\right|\right)\delta_{j-1/2},$$

where $F^{\text{up}}_{j-1/2}$ is the upwinding flux,

$$F^{\text{up}}_{j-1/2} = \begin{cases} a\,u_{j-1} & \text{if } a > 0 \\ a\,u_j & \text{if } a < 0, \end{cases}$$

and $\delta_{j-1/2}$ is the limited difference. Let $\Delta u_{j-1/2} = u_j - u_{j-1}$ denote the jump in u across the edge at $x_{j-1/2}$. The limited difference is

$$\delta_{j-1/2} = \phi\left(\theta_{j-1/2}\right)\Delta u_{j-1/2},$$

where

$$\theta_{j-1/2} = \frac{\Delta u_{J_{\text{up}}-1/2}}{\Delta u_{j-1/2}},$$

and

$$J_{\text{up}} = \begin{cases} j-1 & \text{if } a > 0 \\ j+1 & \text{if } a < 0 \end{cases}.$$

Note that you will need two ghost cells on each end of the domain. Write your program so that you may choose from the different limiter functions listed below.

Upwinding	$\phi(\theta) = 0$
Lax-Wendroff	$\phi(\theta) = 1$
Beam-Warming	$\phi(\theta) = \theta$
minmod	$\phi(\theta) = \text{minmod}(1, \theta)$
superbee	$\phi(\theta) = \max(0, \min(1, 2\theta), \min(2, \theta))$
MC	$\phi(\theta) = \max(0, \min((1+\theta)/2, 2, 2\theta))$
van Leer	$\phi(\theta) = \dfrac{\theta + \lvert\theta\rvert}{1 + \lvert\theta\rvert}$

The first three are linear methods that we have already studied, and the last four are high-resolution methods.

Solve the advection equation with $a = 1$ with periodic boundary conditions for the different initial conditions listed below until time $t = 5$ at Courant number 0.9.

(a) Wave packet: $u(x, 0) = \cos(16\pi x)\exp(-50(x - 0.5)^2)$.

(b) Smooth, low frequency: $u(x, 0) = \sin(2\pi x)\sin(4\pi x)$.

(c) Step function: $u(x, 0)\begin{cases} 1 & \text{if } |x - 1/2| < 1/4 \\ 0 & \text{otherwise} \end{cases}$.

Compare the results with the exact solution, and comment on the solutions generated by the different methods. How do the different high-resolution methods perform in the different tests? What high-resolution method would you choose to use in practice?

Figure 3.76: Problem statement

The PDE

$$u_t + au_x = 0$$

was solved using finite volume method using the 7 flux limiter functions listed in the problem statement above. The following tables summarize the observations made after running the simulations using each of these limiter functions. Each method was given a letter grade based

on how close it was to the exact solution and how well the numerical solution appeared. Numerical solutions that showed ripples around the region of discontinuous data (corners) or showed more spatial lag relative to the exact solution, or had large amount of diffusion were graded lower than those which did not show any of these result.

3.5.3.1 part (a) wave packet as initial conditions

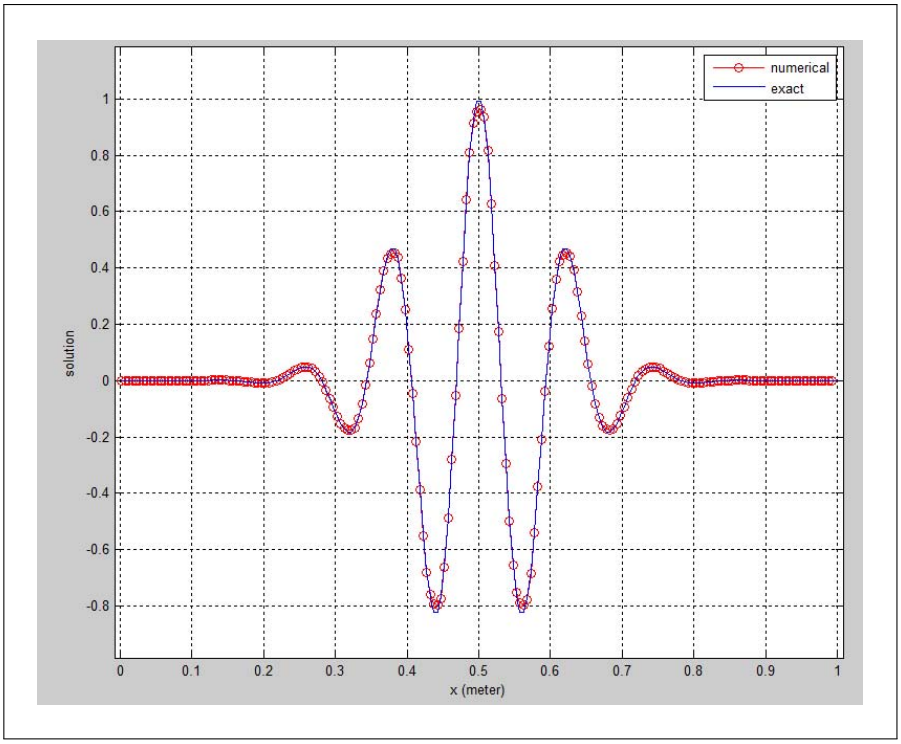


Figure 3.77: Initial conditions

method	comment	grade
Upwinding	Very large diffusion seen at wave crest and trough, but no shift (lag).	F
Lax-Wendroff	Some diffusion at wave crest and trough, in addition to significant shift to the left direction relative to exact solution.	B-
Beam-Warming	Similar to Lax-Wendroff, but shift was to the right relative to exact solution.	B
minmod	Diffusion was present at wave crest and trough, but no shifting.	C
superbee	No shifting and very small amount of diffusion at crest and trough.	B+
MC	Similar to superbee, but a little more diffusion at crest and trough.	B
Van Leer	Similar to MC limited, but much more diffusion at crest and trough.	B-

Among the high resolution limiter functions, superbee had the best numerical result.

3.5.3.2 part(b) smooth low frequency

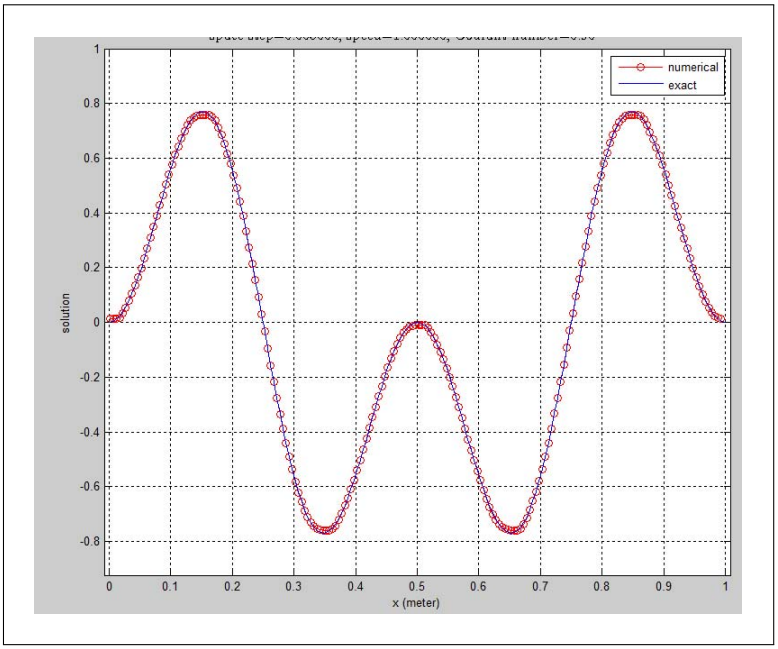


Figure 3.78: Initial conditions for part b

method	comment	grade
Upwinding	No shifting, but large amount of diffusion at crest and trough of the wave.	C
Lax-Wendroff	No shifting and no diffusion.	A
Beam-Warming	Very similar to Lax-Wendroff.	A
minmod	No shifting, but small amount of diffusion was present near crest and trough.	B
superbee	No shift and no diffusion, but at crest and trough, solution appeared to be less smooth than with Lax-Wendroff.	A-
MC	Similar to Lax-Wendroff, a little better than Superbee around crest and trough.	A
Van Leer	No diffusion and no shifting	A

Among the high resolution methods, MC and Van Lee had the best results. Among the non high resolutions method, Lax-Wendroff and Beam-Warming were the best.

3.5.3.3 Part (c) step function

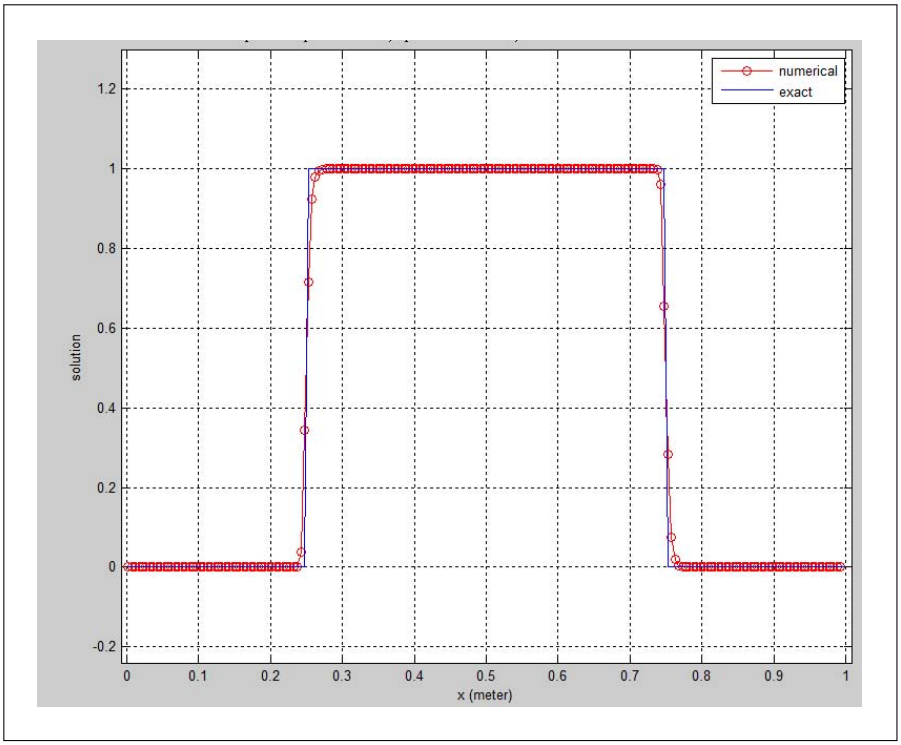


Figure 3.79: Initial conditions for part C

method	comment	grade
Upwinding	No ripples, solution followed the general form of the step function but there was large amount of diffusion near the corners.	C
Lax-Wendroff	Large ripples around the corners on the left of the step function. Less diffusion than upwinding.	C
Beam-Warming	The ripples are larger and have a larger extent than Lax-Wendroff.	C-
minmod	No ripples and little diffusion. An improved version of upwinding.	C+
superbee	The best scheme for the step function. No ripples, very closely followed the exact solution. Very small diffusion was seen.	A-
MC	Similar to superbee, but more diffusion.	B
Van Leer	Similar to MC limited.	B+

Among the high resolution methods, superbee was the best. Among the non high resolutions method, Lax-Wendroff and Beam-Warming are best.

3.5.3.4 Conclusion

Among the high resolutions methods, I would choose superbee. It handled discontinues data the best and did well for smooth data, even though MC and Van Leer did a little better on the low frequency data, superbee had less diffusion in the wave packet data. So, overall, and in particular since it handled discontinues data better than any other flux limiter function, it is the method I would choose in practice.

Among the non high resolution methods, Lax-Wendroff and Beam-Warming were very similar. Upwinding did not do well. All the non high resolution methods did relatively worst in the step function test compared to the high resolution methods, as they were not able to handle solution near the discontinues regions as well as the high resolution methods did.

Numerical solutions using all the above methods have been animated and available to run at my course web page. All the animations run for 5 seconds each.

3.5.4 References

1. Robert Guy, Lecture notes, Math 228B, Numerical Methods for PDEs. Winter 2011, UC Davis, CA

2. R. J. LeVeque. Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems. SIAM, 2007.

3. R. J. LeVeque. Finite Volume Methods for Hyperbolic Problems. Cambridge University Press; August 26, 2002.

3.6 HW 4 animations

These are HW4 animations.

3.6.1 Problem 1 results, Animation of acoustic wave equation solution using Lax-Wendroff

The following are animated GIFs showing the finite difference numerical solution to problem 1 as described in the above HW. The scheme used is Lax-Wendroff.

Clicking on an image will start the animation in a new window.

These simulations only show the pressure wave, $p(x,t)$ and not the acoustic perturbation velocity $u(x,t)$.

3.6.1.1 pressure Wave reflecting off both the left and the right boundary

This solution was run with boundary conditions which caused the sound wave to reflect from both boundaries. This is what would happen inside a room with reflective walls such as concerete or wood.

3.6.1.2 pressure Wave which reflects off the left boundary only and absorbed at the right boundary

This solution was run with boundary conditions which caused the sound wave to reflect from only the left boundary but absorbed into the right boundary. This is what would happen inside a room with one wall treated with material to absorbe the sound waves reaching it.

3.6.2 Problem 3 results, solving the advection 1-D using finite volume method

The following are animations of the numerical solution to $u_t + au_x = 0$. The solution used the finite volume method using 7 different numerical flux limiter functions to compare performance.

These 7 methods are defined in the problem statement in the report above.

The methods are

1. Upwinding
2. Lax-Wendroff
3. Beam-Warming
4. minmod (high resolution)
5. superbee (high resolution)

6. MC limited (high resolution)
7. Van Leer (high resolution)

The following tables show the results of the simulations. 4 tables are given. Each table is for a different initial conditions. In all of these results, the maximum run time was 5 seconds. In order to reduce the size of the animation file, not every frame was captured from the simulation run.

Courant number used was 0.9, the advection speed was set at $a = 1$ and grid spacing was $h = 0.005$ meters. The domain is $[0,1]$ using cell centered grid.

These animations will run only once and stop at 5 seconds. To run it again, simply reload the web page using the browser reload button, this will cause the animation to start from the beginning again.

3.6.2.1 Results for wave packet as initial conditions

The wave packet is defined as $u(x,0) = cos(16\pi x)exp(-50(x-0.5)^2)$

3.6.2.2 Results for smooth low frequency initial conditions

The wave packet is defined as $u(x,0) = sin(2\pi x)sin(4\pi x)$

3.6.2.3 Results for step function

A step function from $x = 0.25$ to $x = 0.75$.

3.6.2.4 Results for mixed step function and smooth function

The initial condition used for this test is $u(x,0) = (X > 0.1)(X < 0.3) + exp(-200(X-0.75)^2)$

This test just combines the step function with the low frequency smooth test done above. Hence, the same comments will apply as above.