Younus Jamal

N254, Engineering Building 1,

Cullen College of Engineering

University of Houston

Hello Dr. AA and Dr. AP,

With due respect and honor, commensurate to your professional privileges, my name is Younus Jamal. Appended below is the cover letter which briefly entails the specifics of the Project: B02-5 – Diffusion Equation.

The diffusion equation is a partial differential equation which describes density fluctuations in a material undergoing diffusion. The system could easily be solved considering some parameters like time, temperature, diffusivity constant, specific heat, and space increments. The results could either be shown in plots or mesh; it depends upon the choice of the user. Accordingly, the system could be solved by either taking implicit discretization, explicit discretization, or Crank-Nicholson and Ranga-Kutta scheme of solution.

I adopted implicit and explicit discretization, and I appreciate your keen interest and co-operation to help us in completing every phase of this project. You can always contact me for any inquiries or concerns at yjamal@uh.edu.

Thanks,

Sincerely,

Younus Jamal

**Project Report-Spring 2018 Semester**

**April 20th, 2018 – May 9th, 2018**

**Project B – Diffusion Equation**

**Student Name: Younus Jamal**

**Course: Scientific Computing for Mechanical Engineers: 5397**

**Instructors: Dr. AA and Dr. AP**

**Date of Submission: May 5th, 2018**

# Abstract

The primary goal of this report is to examine diffusion equation and how different parameters play their part in solving this 2D diffusion equation. This report covers this aspect very amicably. It entails a half page description of 2D – Diffusion Equation in Mat-lab solver. Further mathematical statement covering the prompt has been mentioned. Then discretized version of the problem of the equations have been cast. A thorough description of the numerical methods (in this case: Implicit and Explicit Discretization) has been stipulated. Some technical specification of the computer with salient features have been included. Then there is a portion of results allocated specifically to address some key postulates like specifications of the parameters used in the simulations, evaluation of the effects of number of points used for discretization, performance of grid-convergence study, evaluation of the effects of courant number referring to CFL behavior, comparison of results between implicit and explicit discretization with individual modulation in space and time increments. In the last, verification and validation of the spatial accuracy of the discretization has been included.

Table of Contents

[Abstract i](#_Toc513798438)

[List of Figures iii](#_Toc513798439)

[Mathematical Statement of The Problem 5](#_Toc513798440)

[Discretized Version of the Problem 5](#_Toc513798441)

[Description of the Numerical Methods (Pseudo Code included) 6](#_Toc513798442)

[Pseudo Code | Implicit and Explicit Discretization 6](#_Toc513798443)

[Technical Specifications of the Computer Used 9](#_Toc513798444)

[Results (Include Graphs and Comments) | Implicit and Explicit 9](#_Toc513798445)

[Specifications of the Parameters Used in the simulation | Explicit and Implicit 9](#_Toc513798446)

[Evaluate the effect of Number of Points Used for Discretization | Explicit Discretization 10](#_Toc513798447)

[Evaluate the effect of Number of Points Used for Discretization | Implicit Discretization 12](#_Toc513798448)

[Time Integration to Steady State, Results Becoming independent of Time | Explicit Discretization 14](#_Toc513798449)

[Time Integration to Steady State, Results Becoming independent of Time | Implicit Discretization 15](#_Toc513798450)

[Perform Grid Convergence Study | Explicit Discretization 16](#_Toc513798451)

[Perform Grid Convergence Study | Implicit Discretization 18](#_Toc513798452)

[Evaluate the effect of Diffusive CFL | Explicit Discretization 20](#_Toc513798453)

[Evaluate the effect of Diffusive CFL | Implicit Discretization 22](#_Toc513798454)

[Comparison of Results with Expected Theoretical Behavior | Explicit and Implicit |Adaptive Integration 23](#_Toc513798455)

[Verification of the Order of Spatial Accuracy of Discretization | Explicit and Implicit 26](#_Toc513798456)

# List of Figures

[Figure 1: Time spent vs nt 12](#_Toc513798393)

[Figure 2: NT = 250………………………………………………………………………………………………………………………………….13 Figure 3: NT = 500 13](#_Toc513798394)

[Figure 4: Nt = 600 13](#_Toc513798395)

[Figure 5: Nt = 600…………………………………………………………………………………………………………………………………..13 Figure 6: NT = 800 14](#_Toc513798396)

[Figure 7: NT = 1000 15](#_Toc513798397)

[Figure 8: Nt = 600…………………………………………………………………………………………………………………………………..15 Figure 9: Nt = 800 16](#_Toc513798398)

[Figure 10: NT = 1000 16](#_Toc513798399)

[Figure 11: Time speNT FOR TEMPERATURE TO RISE VS GRIDDING 18](#_Toc513798400)

[Figure 12: nX = 80………………………………………………………………………………………………………………………………….18 Figure 13: nX = 100 19](#_Toc513798401)

[Figure 14: nx = 120…………………………………………………………………………………………………………………………………19 Figure 15: nx = 140 19](#_Toc513798402)

[Figure 16: velocity, C = 1.5………………………………………………………………………………………………………………………19 Figure 17: Velocity, c= 2 20](#_Toc513798403)

[Figure 18: Velocity, C = 2.5 21](#_Toc513798404)

[Figure 19: tIME-SPENT VS VELOCITY MAGNITUDE 21](#_Toc513798405)

[Figure 20: C = 1…………………………………………………………………………………………………………………………………….21 Figure 21: C = 1.5 22](#_Toc513798406)

[Figure 22: C= 2 22](#_Toc513798407)

[Figure 23: explicit descretization for 5 x 5 nodes matrix 23](#_Toc513798408)

[Figure 24:Implicit descretization for 5x5 nodes matrix 24](#_Toc513798409)

[Figure 25: Relative error between implicit and explicit 25](#_Toc513798410)

[Figure 26: verification of the code 26](#_Toc513798411)

# Mathematical Statement of The Problem

We are required to produce a computer code to solve the two-dimensional diffusion equation

Where the domain of the interest is the rectangle

And the boundary conditions are

We will apply ghost nodes for Newman Boundary conditions.

Further, we will increase the number of nodes for time-steps until the results become independent of time| Steady State

# Discretized Version of the Problem

The explicit discretization of the equation is as follows:

Whereas the implicit discretization of the equation is as follows:

Here i, j refers to the node numbers in x and y directions, while delta-x represents the spacing in x and y directions **(x and y spacing being equal**)

# Description of the Numerical Methods (Pseudo Code included)

Two numerical methods will be discussed for comparison: Implicit and Crank-Nicholson discretization. For Implicit discretization, the Von Neumann criterion shows the solution is unconditionally stable, although with an error of order delta-t in time and delta-x^2 and delta-y^2 in space. The node (i, j) are connected to the four surrounding nodes i +- 1, j and i, j+-1 so that the linear system has 5 diagonals matrix rather than. 3. The Crank-Nicholson discretization is unconditionally stable for Dirichlet Boundary conditions. As the implicit scheme, it also gives rise to a penta-diagonal matrix, although it has a higher accuracy with an error of order delta-t^2 in time and delta-x^2 and delta-y^2 in space.

## **Pseudo Code | Implicit and Explicit Discretization**

**% Solving two-dimensional diffusion equation without any heat-source**

**% Solve Ut = Uxx + Uyy**

**% Parameters**

**% Domain and Steps**

**bx = 2\*pi;**

**by = 2\*pi**

**% Now comes the node numbers for x and y in spacing and t in time**

**Nx = 51 (This will differ to observe behavior)**

**Ny = 51 (This will differ to observe behavior)**

**Nt = 500 (this will also change to observe behavior in time)**

**% Now Comes the increment size**

**𝞓x = lx/(Nx - 1);**

**𝞓y = ly/(Ny - 1);**

**% Note dx and dy are the same: dx = dy for ease of use**

**% Satisfy time-speed condition | Studying the Effects of Diffusive CFL**

**c = 1; % speed**

**C = 0.05; % courant number (CFL condition C < 1);**

**dt = C\*dx/c;**

**% Now define the field variables**

**Un = zeros (Ny, Nx); % temperature**

**x = linspace (0, bx, Nx); % x distance**

**y = linspace (0, by, Ny); % y distance**

**[X, Y] = meshgrid(x, y); % what this meshing would do, is to visualize our psuedo-code into interactive results**

**% Now define the initial and boundary conditions as stated in Project-prompt**

**Un(:, :) = 0; % This defines the time initiation**

**t = 0;**

**% now looping**

**for n = -1 : Nt**

**Uc = Un+1**

**t = t + dt; % new time**

**for i = 2:Nx-1**

**for j = 2:Ny-1**

**Un(j,i) = Uc(j, i) + ((dt/(dx.^2)).\*((Un(j, i+1) + Un(j+1, i) - 4\*Un(j, i) + Un(j, i-1) + Un(j-1, i)))); % implicit**

**Un(j,i) = Uc(j, i) + ((dt/(dx.^2)).\*((Uc(j, i+1) + Uc(j+1, i) - 4\*Uc(j, i) + Uc(j, i-1) + Uc(j-1, i)))); % explicit**

**end**

**end**

**% Now comes the boundary conditions**

**% given are**

**% 1 (ax = 0 < x < bx = 2pi) , 2 ( ay = 0 < y < by = 2pi) ,**

**% u(x = ax, y) = phi\_ab(y) , u(x = bx, y) = si\_ab(y) ,**

**% First apply the Dirichlet Boudary conditions**

**Un (:, 1) = (cos(y\*pi)-1).\*cosh(2\*pi-y);**

**Un (:, end) = (y.\*y).\*(sin(y/4));**

**% Now apply the Newman Boundary Conditions**

**Un (end, :) = Un (end-1, :);**

**Un (1, :) = Un (2, :);**

**% Visualize**

**Plot (x, y, Un); axis ([0 bx 0 by]);**

**mesh (x, y, Un); axis ([0 lx 0 ly]);**

**title(sprintf('Time = % f seconds', t));**

**end**

# Technical Specifications of the Computer Used

Following are some technical specifications of the computer used for solving this problem:

* Processor: Intel® Core™ i5-4300 CPU @ 1.0 Ghz 2.50 Ghz
* Installed RAM Memory: 4.00 GB
* System Type: 64-bit Operating System

# Results (Include Graphs and Comments) | Implicit and Explicit

## **Specifications of the Parameters Used in the simulation | Explicit and Implicit**

As mentioned in the pseudo code, I specified the following parameters were used for the simulation:

**bx:** It is max limit of x-axis

**by:** It is max limit of y-axis

**Nx, Ny:** It is the number of nodes used for simulation for x and y

**Nt:** Max limit of time steps set for running simulation

**𝞓x** = lx/ (Nx - 1): It is the increment size in x-axis

**𝞓y =** ly/ (Ny - 1): it is the increment size in y-axis

**c:** It is the constant used to compute dt

**C:** It is the courant number used for CFL behavior in diffusion equation

## **Evaluate the effect of Number of Points Used for Discretization | Explicit Discretization**

Number of points for time spacing play a crucial role in determining the mesh of the solution. As mentioned in the diagrams below:

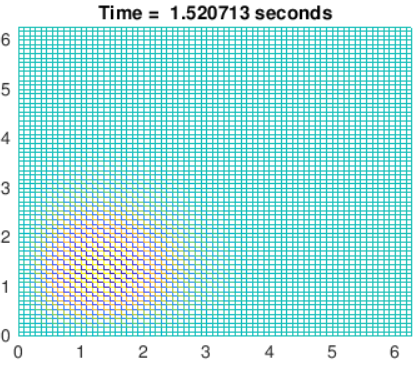
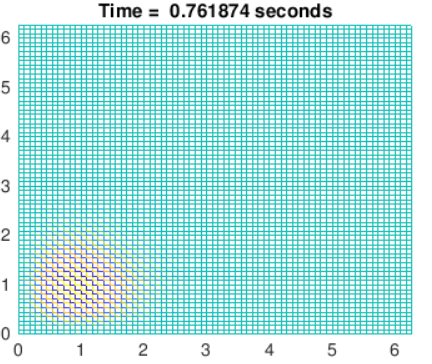


FIgure 1; NT = 250 Figure 2; Nt = 500

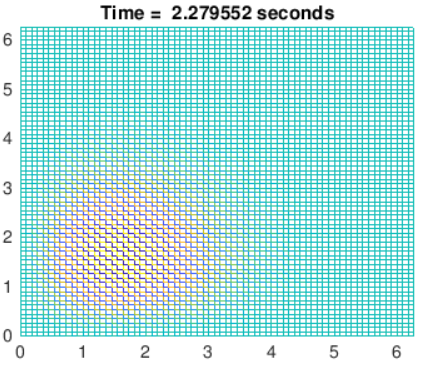


Figure 3: Nt = 600

|  |  |  |
| --- | --- | --- |
| **Figure Number** | **Number of Points** | **Time Spent For temperature to rise** |
| 1 | 250 | 0.761 |
| 2 | 500 | 1.520 |
| 3 | 600 | 2.279 |

Thus, we see that the greater the number of points used for time-spacing, the greater the time spent to generate the results and show the heat penetrated through the rectangle. Implicit and explicit discretization showed similar behavior.

Figure : Time spent vs nt

Thus, according to the graphs above, we see that the up-to 600 number of nodes, the results changes, however, things get stable as we increased the nodes from 600.

## **Evaluate the effect of Number of Points Used for Discretization | Implicit Discretization**

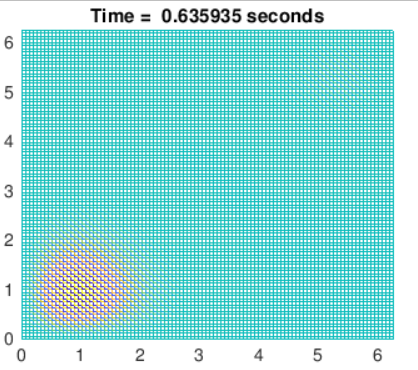
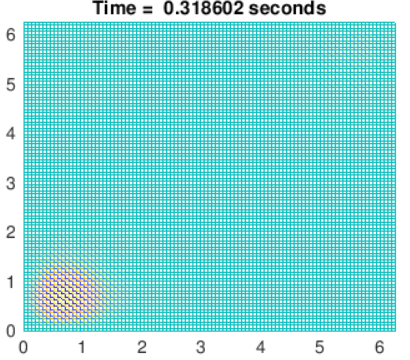


Figure : NT = 250 Figure : NT = 500

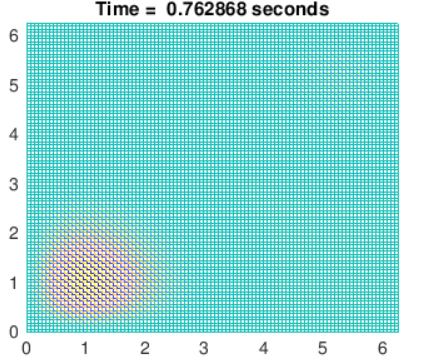


Figure : Nt = 600

As shown in here, implicit is comparably better than the explicit, since the figure does not show much changes when time-spacing were increased.

## **Time Integration to Steady State, Results Becoming independent of Time | Explicit Discretization**

One thing could be shown that as we increase the time from Nt = 600, we notice that there no change in heat perpetration with respect to time, that-is the conditions gotten stable. This can be visualized by the following figures:

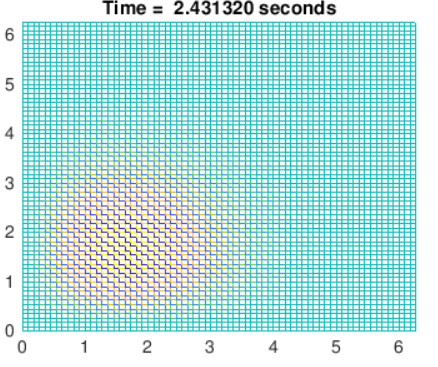
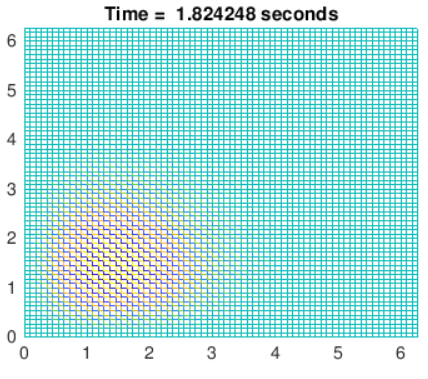


Figure : Nt = 600 Figure : NT = 800

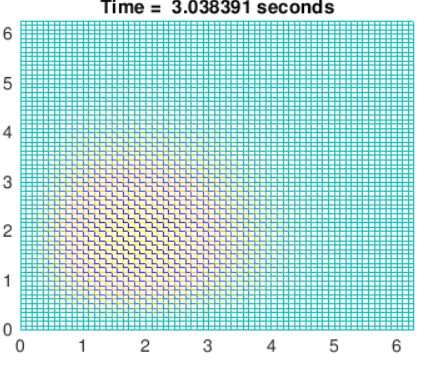


Figure : NT = 1000

## **Time Integration to Steady State, Results Becoming independent of Time | Implicit Discretization**

Like explicit discretization, one thing could be shown that as we increase the time from Nt = 600, we notice that there no change in heat perpetration with respect to time, that-is the conditions gotten stable. This can be visualized by the following figures:

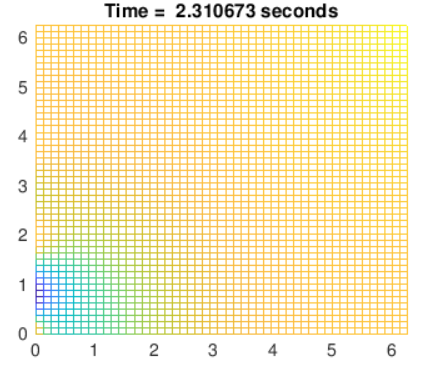
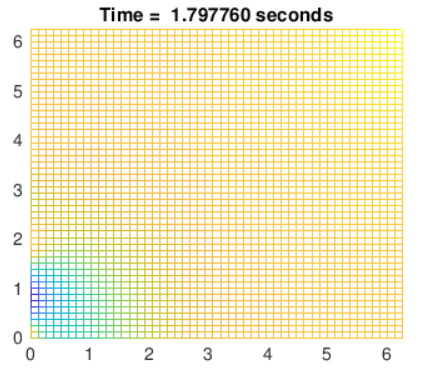


Figure : Nt = 600 Figure : Nt = 800

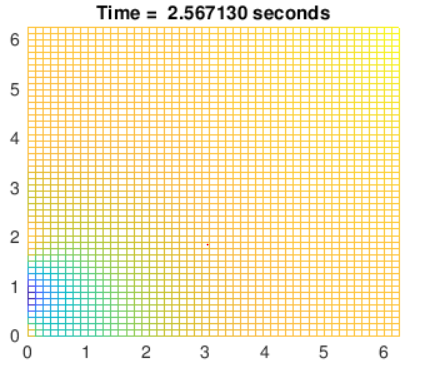


Figure : NT = 1000

## **Perform Grid Convergence Study | Explicit Discretization**

At the most basic level, before any results can be taken trustworthy, make sure the code is insensitive to time and space steps, used to obtain the result. Here in my code, initially the performance differs up-to a certain level, but then when you increase the space steps then code becomes stable; that is, it not fluctuating.

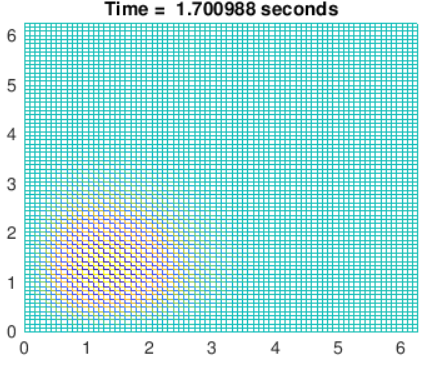
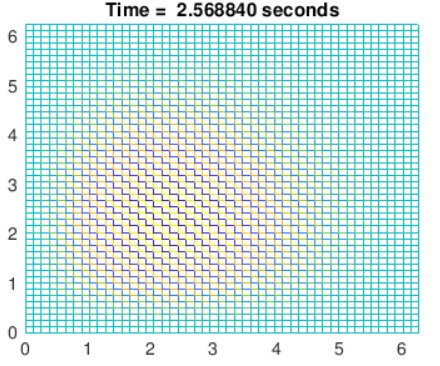


Figure 4: Nx = 50 Figure 5: Nx = 75

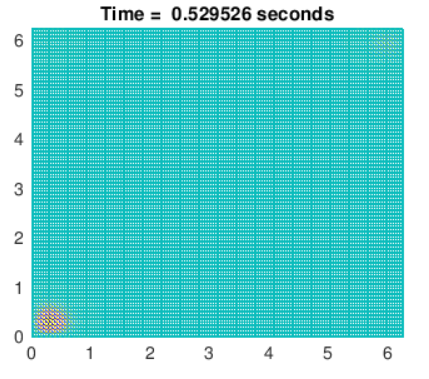
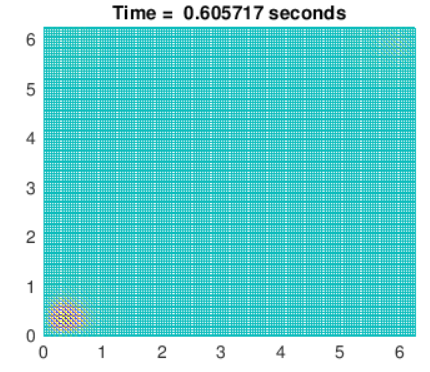


Figure 6: Nx = 140 Figure 6: Nx = 160

|  |  |  |
| --- | --- | --- |
| **Figure Number** | **NX** | **Time Spent For temperature to rise** |
| 4 | 50 | 2.568 |
| 5 | 75 | 1.701 |
| 6 | 100 | 1.271 |

Figure : Time speNT FOR TEMPERATURE TO RISE VS GRIDDING

## **Perform Grid Convergence Study | Implicit Discretization**

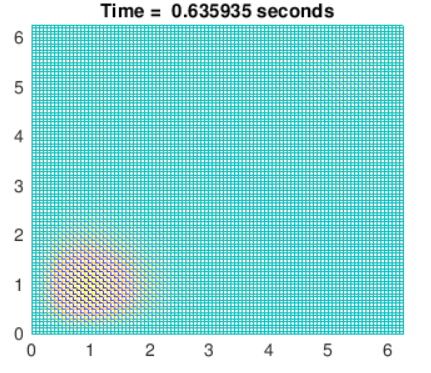
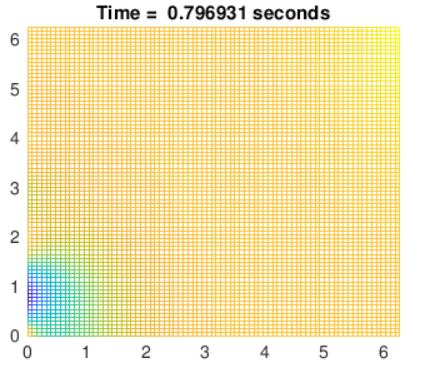


Figure : nX = 80 Figure : nX = 100

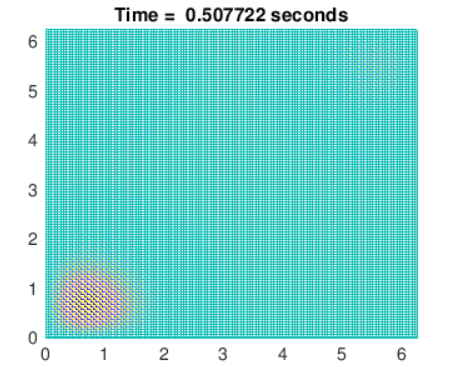
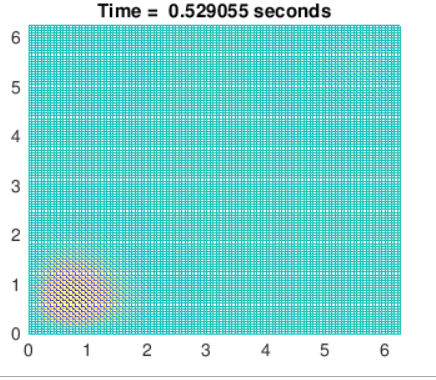


Figure : nx = 120 Figure : nx = 140

As seen from the figures, from the implicit descretization too! The behaviour of the code becomes stable after grid increased from

120 number of nodes.

## **Evaluate the effect of Diffusive CFL | Explicit Discretization**

The Courant–Friedrichs–Lewy or CFL condition is a condition for the stability of unstable numerical methods that model convection or wave phenomena. As such, it plays an important role in CFD (computational fluid dynamics). The CFL relation is given as

Where C is the courant number, which is < 1 in explicit schemes. The courant number is less than 1 for explicit condition to maintain stability, whereas for implicit condition, it is not required to impose this restriction so an increase in value of C is tolerated.

As the figure shows, if you increase the c, magnitude of the velocity for the x and y spacing, the time spent to complete the simulation would be less compared to the velocity with the lesser magnitude.

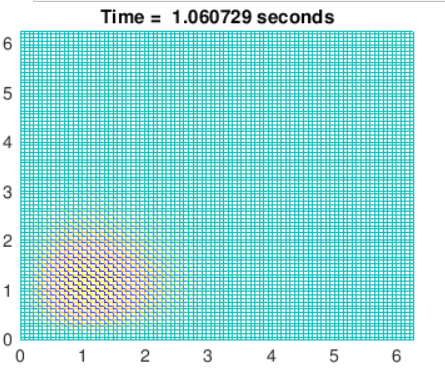
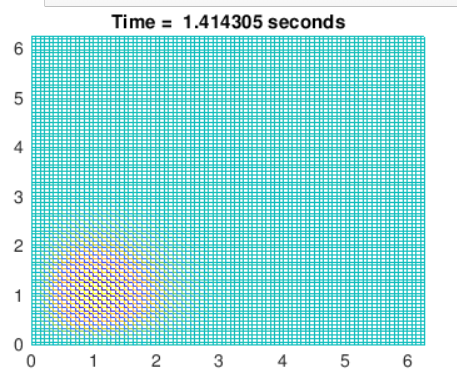


Figure : velocity, C = 1.5 Figure : Velocity, c= 2

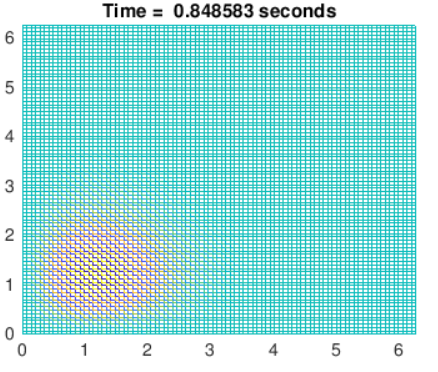


Figure : Velocity, C = 2.5

|  |  |
| --- | --- |
| Velocity Magnitude, C | Time-Spent |
| 1.5 | 1.414 |
| 2.0 | 1.060 |
| 2.5 | 0.848 |

Figure : tIME-SPENT VS VELOCITY MAGNITUDE

## **Evaluate the effect of Diffusive CFL | Implicit Discretization**

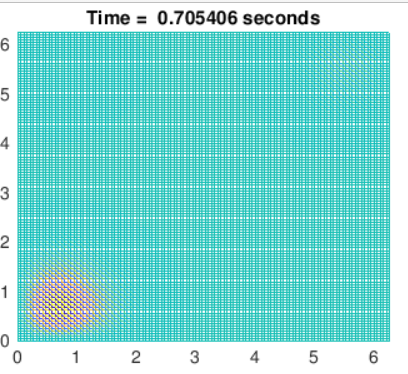
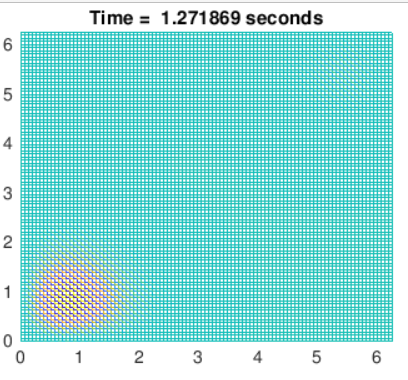


Figure : C = 1 Figure : C = 1.5

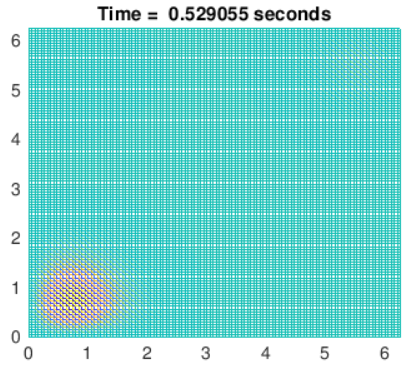


Figure : C= 2

It is seen from the from the figure, a greater courant number decrease the time required to complete a certain iteration.

## **Comparison of Results with Expected Theoretical Behavior | Explicit and Implicit |Adaptive Integration**

The comparison of Implicit and Explicit Discretization is done with minimal number of nodes, let’s say between 5x5 numbers of nodes to actually ‘see’ the difference between the results:

Figure : explicit descretization for 5 x 5 nodes matrix

Figure :Implicit descretization for 5x5 nodes matrix

Referencing one of the discretization as a reference to another could be good way to observe how the code is doing. If we use this equation

This is helpful to consider the deviation of one discretization to the other.

Table :Error Analysis

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Node (x=1, y), Relative Error** | | | **Node (x=2, y) Relative Error** | | | **Node (x=3, y), Relative Error** | | |
| Explicit | Implicit | **Error** | Explicit | Implicit | **Error** | Explicit | Implicit | **Error** |
| 12.2785 | 12.26745 | **0.000901** | 12.2758 | 12.21442 | **0.005025** | -10.4639 | -10.4011 | **0.006036** |
| 5.7158 | 5.710656 | **0.000901** | 5.7158 | 5.687221 | **0.005025** | 0.2117 | 0.21043 | **0.006036** |
| 4.6572 | 4.653009 | **0.000901** | 4.6572 | 4.633914 | **0.005025** | 4.216 | 4.190704 | **0.006036** |
| 4.0398 | 4.036164 | **0.000901** | 4.0398 | 4.019601 | **0.005025** | 6.518 | 6.478892 | **0.006036** |
| 0.9442 | 0.94335 | **0.000901** | 0.9442 | 0.939479 | **0.005025** | 6.9789 | 6.937027 | **0.006036** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node (x=4, y), Relative Error** | | | **Node (x=5, y), Relative Error** | | |
| -1.5527 | -1.53873 | **0.009082** | -1.5572 | -1.54007 | **0.011122** |
| 1.3788 | 1.366391 | **0.009082** | 1.3788 | 1.363633 | **0.011122** |
| 5.4773 | 5.428004 | **0.009082** | 5.4773 | 5.41705 | **0.011122** |
| 10.8372 | 10.73967 | **0.009082** | 10.8372 | 10.71799 | **0.011122** |
| 20.5162 | 20.33155 | **0.009082** | 20.5162 | 20.29052 | **0.011122** |

Figure : Relative error between implicit and explicit

Adaptive integration is desirable when comparing different discretization schemes. The key ingredients is to have a way ‘sense’ how accurate a solution the integration process is producing “in real time”, i-e as the integration processes. A standard to achieve this objective is to take one step of length delta-t and t^n and two half-steps of lengths 0.5\*delta-t, also starting at t^n, and to compare the two solutions thus found, which are two different approximations to u^n+1.

|

## **Verification of the Order of Spatial Accuracy of Discretization | Explicit and Implicit**

GRID INDEPENDENCE:

At the most basic level, before any result can be “believed”, it must be proven that the results be insensitive to time and spatial steps used for calculating the performance of the problem. We have already seen from the figures (in the section: grid-convergence study) that our code does imply to be insensitive to space, albeit some dependency on time up-to 600 time-nodes.

Figure : verification of the code

Error is less than 1% for a 5x5 node pattern. Similar behavior was observed for 10x10 and higher order. How-ever, it needs to be kept in mind that, for some specific ‘areas’ code there might appear some unexpected behavior.