PROJECT REPORT :Scientific computing

Project 1: Solving Poisson equation with given Dirichlet and Nuemaan Bc.B02-2

Project 2: Solving Diffusion equation with given Dirichlet and Nuemaan Bc.AP01-3

Submitted by : Jasdeep makkar

1. **Outline**
2. **Abstract for Poisson and diffusion problem**
3. **Mathematical statement of the problem**
4. **Discretized version of the equation**
5. **Description of numerical methods used and pseudo code**
6. **Results and discussion**
    a. **Specification and parameters used in the problems**
    b. **Effect of number of points used for discretization**
    c. **Grid convergence study**
    d. **Effect of diffusive CFL**
    e. **Comparison of results with the expected theory**
    f. **Verify the order of special accuracy with discretization.**

Abstract

For the project given an approximate solution of the diffusion equation for the temperature at the surface of a 2D domain and an approximate solution of the poisson equation using iterative methods at a 2D domain is improvised.

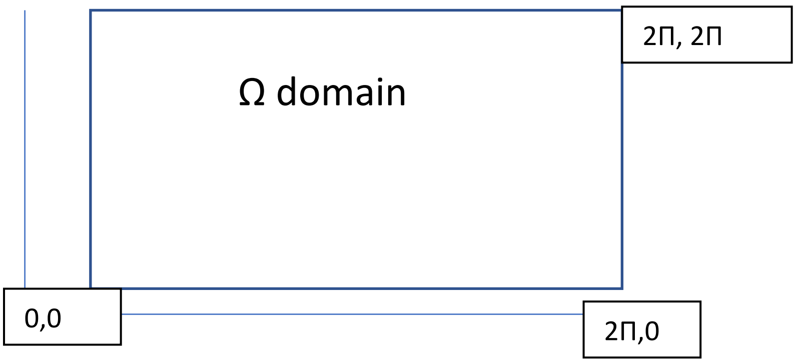
To solve DE scheme such as Explicit and ADI have been used and to solve for PE iterative methods such Gauss seidel and SOR has been implemented.

A final solution of the equation has been posed as a matrix of N+2,N+2 dimension with N referring to the point used to discretize the domain.

Grid convergence study along and accuracy of special discretization has been performed.

Results with plots, contours and tables have been provided supporting the relevant work.

Mathematical statement of the problem DE and PE



$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -f(x, y)$$

Omega is the domain of interest, with directions x and y magnitude provided with their coordinate values. To solve above PE for below given BC and DE with Bc along with the initial condition.

The domain of interest is the rectangle

$$a_x < x < b_x\,, \qquad a_y < y < b_y$$

and the boundary conditions

$$u(x = a_x, y) = f_b(y)\,, \qquad u(x = b_x, y) = g_b(y)\,,$$

$$\left.\frac{\partial u}{\partial y}\right|_{y=b_y} = 0\,, \qquad u(x, y = a_y) = f_b(a_y) + \frac{x - a_x}{b_x - a_x}\left[g_b(a_y) - f_b(a_y)\right]$$

$$a_x = a_y = 0\,, \qquad b_x = b_y = 2\pi$$

$$f_b(y) = (b_y - y)^2 \cos\frac{\pi y}{b_y}\,, \qquad g_b(y) = y(b_y - y)^2$$

$$F(x, y) = \cos\left[\frac{\pi}{2}\left(2\frac{x - a_x}{b_x - a_x} + 1\right)\right] \sin\left[\pi\frac{y - a_y}{b_y - a_y}\right]$$

Figure 1 . Condition for PE

The domain of interest is the rectangle

$$a_x < x < b_x\,, \qquad a_y < y < b_y$$

and the boundary conditions

$$u(x, y = b_y) = f_a(x)\,, \qquad u(x, y = a_y) = g_a(x)\,,$$

$$\left.\frac{\partial u}{\partial x}\right|_{x=a_x} = 0\,, \qquad u(x = b_x, y) = g_a(b_x) + \frac{y - a_y}{b_y - a_y}\left[f_a(b_x) - g_a(b_x)\right]$$

$$a_x = a_y = 0\,, \qquad b_x = b_y = 2\pi$$

$$g_a(x) = (x - a_x)^2 \cos x\,, \qquad f_a(x) = x(x - a_x)^2$$

The initial condition is

$$u(x, y, t = 0) = 0\,.$$

Figure 2 : Condition for DE

Discretized version of the equation

Poisson equation

For problem one method which is selected is and the discretization is done central difference scheme for both x and y coordinate but due to the structure of domain it seems suited to take number of point similar on x and y axis.

$$\left[\left(\frac{\partial^2 u}{\partial x^2}\right)_y\right]_{x=x_j, y=y_k} = \frac{u_{j-1,k} - 2u_{jk} + u_{j+1,k}}{\Delta x^2} + O(\Delta x^2) .$$

$$\left[\left(\frac{\partial^2 u}{\partial y^2}\right)_x\right]_{x=x_j, y=y_k} = \frac{u_{j,k-1} - 2u_{jk} + u_{j,k+1}}{\Delta y^2} + O(\Delta y^2) .$$

$$\frac{u_{j-1,k} - 2u_{jk} + u_{j+1,k}}{\Delta x^2} + \frac{u_{j,k-1} - 2u_{jk} + u_{j,k+1}}{\Delta y^2} = -f_{jk}$$

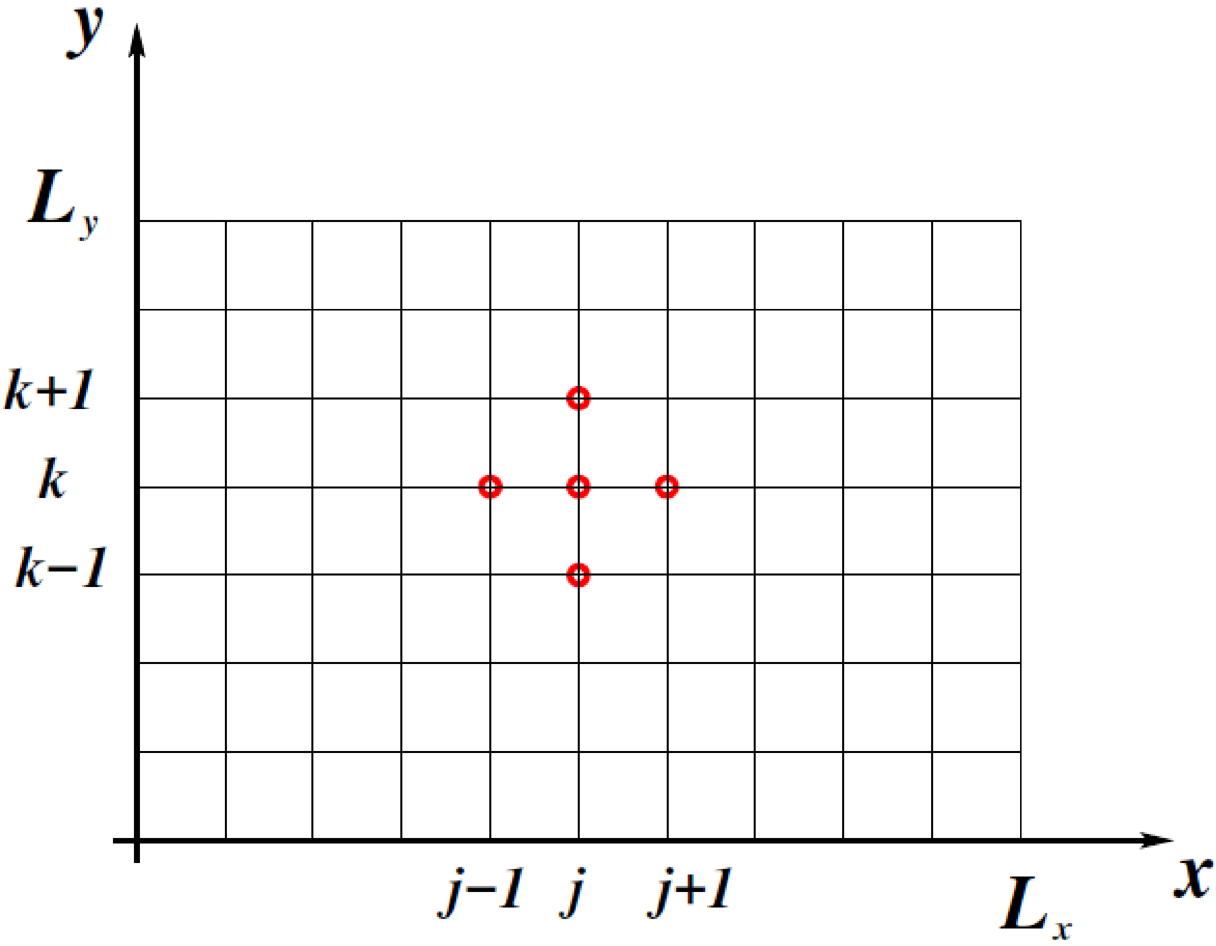
$$u_{j-1,k} + u_{j+1,k} + u_{j,k-1} + u_{k,k+1} - 4u_{jk} = -\Delta^2 f_{jk} .$$

$$u_{j,k}^{(n+1)} = \frac{1}{4}\left[u_{j-1,k}^{(n+1)} + u_{j+1,k}^{(n)} + u_{j,k-1}^{(n+1)} + u_{j,k+1}^{(n)}\right] + \frac{\Delta^2}{4} f_{jk}$$

The above discretization is for gauss seidel method

$$u_{j,k}^{(n+1)} = (1-\omega)u_{j,k}^{(n)} + \frac{\omega}{4}\left[u_{j-1,k}^{(n+1)} + u_{j+1,k}^{(n)} + u_{j,k-1}^{(n+1)} + u_{j,k+1}^{(n)}\right] + \omega\frac{\Delta^2}{4} f_{jk}$$

This is Successive over relation method.



This is the discretized version of the scheme which says while moving in the forward the ones which have calculated before will be updated and the rest of the values will be used from the previous matrix.

Diffusion equation

$$\frac{\partial u}{\partial t} = D\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

$$\nabla_x^2 u_{jk} = \frac{u_{j-1,k} - 2u_{j,k} + u_{j+1,k}}{\Delta x^2}, \qquad \nabla_y^2 u_{jk} = \frac{u_{j,k-1} - 2u_{j,k} + u_{j,k+1}}{\Delta y^2}$$

Where D==1

$$\frac{u_{jk}^{n+1} - u_{jk}^{n}}{\Delta t} = D\left(\nabla_x^2 u_{jk}^n + \nabla_y^2 u_{jk}^n\right)$$

This is for the explicit scheme used

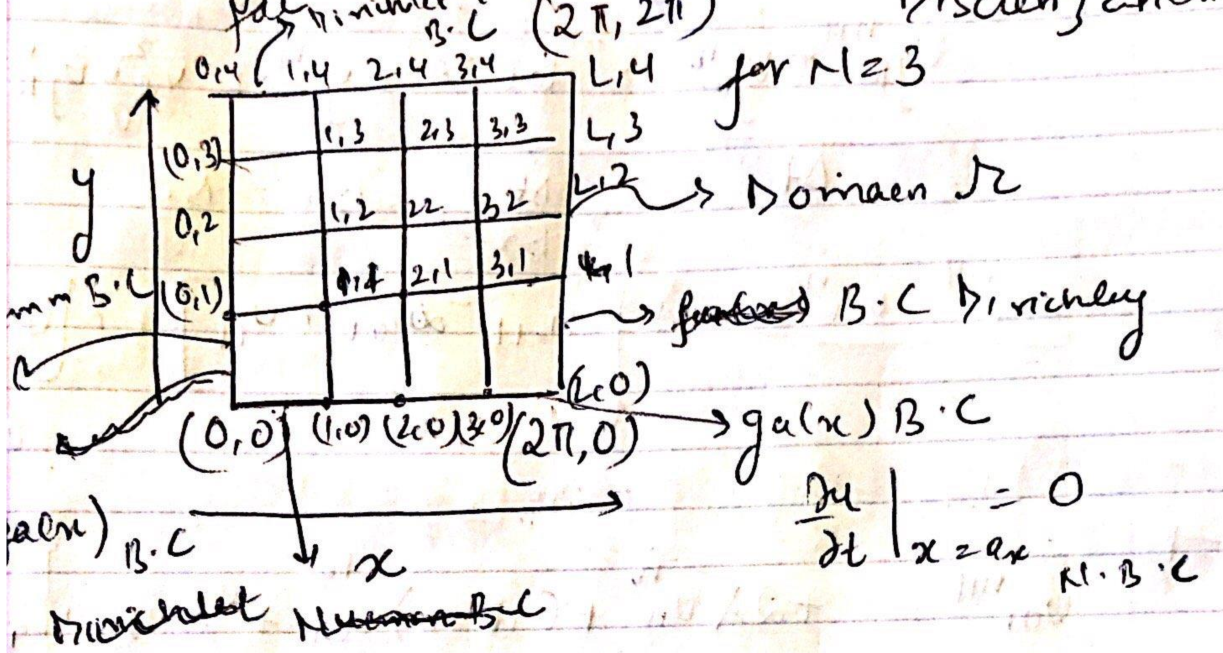
$$\frac{u_{jk}^{n+1/2} - u_{jk}^{n}}{\Delta t/2} = D\left(\nabla_x^2 u_{jk}^{n+1/2} + \nabla_y^2 u_{jk}^{n}\right) ,$$

$$\frac{u_{jk}^{n+1} - u_{jk}^{n+1/2}}{\Delta t/2} = D\left(\nabla_x^2 u_{jk}^{n+1/2} + \nabla_y^2 u_{jk}^{n+1}\right) ,$$

This for ADI scheme used again D==1

Diffusion Equation

Discretization for N=3



for Nueman we will use $u_{t,N,a}$

Point 1 to solve $v_{0,1}$, but due to Nueman B.C, we will use ghost node such as

$$\frac{v_{1,1} - v_{-1,1}}{2h} = 0 \Rightarrow v_{1,1} = v_{-1,1}$$

So if we need to solve for 4 nodes, starting from $u(0,1)$ to $v(3,1)$, we will get a Block tridiagonal matrix.

$$\begin{bmatrix} a_1 & c_1 & 0 & 0 \\ b_1 & a_2 & c_2 & 0 \\ 0 & b_2 & a_3 & c_3 \\ 0 & 0 & b_3 & a_4 \end{bmatrix}$$

used for ADI.

$\swarrow$ N $\qquad \swarrow$ Dirichlet B.C

Solved for point (0,1) & (1, 1)

Discretization for Explicit Scheme

$$\frac{U_{jk}^{n+1} - U_{jk}^{n}}{\Delta t} = D\left(\nabla^2_x U_{jk}^{n} + \nabla^2_y U_{jk}^{n}\right]$$

$\lambda = \frac{\Delta t}{\Delta x^2} \qquad \Delta x = \Delta y$

$$\frac{U_{01}^{n+1} - U_{01}^{n}}{} = \lambda\left[U_{-11} - 2U_{01} + U_{11}\right] + \lambda\left[U_{00} - 2U_{01} + U_{02}\right]$$

$\Rightarrow$

$$U_{01}^{n+1} = +2\lambda U_{11}^{n} + (1-4\lambda)U_{01}^{n} + \lambda U_{00} + \lambda U_{02}$$

$\Rightarrow$ for dirichlet condition it will be

$$U_{11}^{n+1} = (1-4\lambda)U_{11}^{n} + \left[U_{01}^{n} + U_{21}^{n} + U_{10}^{n} + U_{12}^{n}\right]$$

for ADI scheme

[for the Neuman Node]

$$U_{01}^{n+1/2} - U_{01}^{n} = \frac{\lambda}{2}\Big[\underbrace{U_{-11} - 2U_{01} + U_{11}}_{\text{Implicit in x}}\Big]^{n+1/2} + \Big[\underbrace{U_{00} - 2U_{01} + U_{02}}_{\text{Explicit in y}}\Big]$$

$$U_{01}^{n+1/2}(1+\lambda) - \lambda U_{11}^{n+1/2} = \frac{\lambda}{2}\left[U_{00} + U_{02}\right]^{n} + (1-\lambda)U_{01}^{n}$$

~~Discreti~~

for ~~Neumann~~ Dirichlet B.C

$$u_{11}^{n+1/2} - u_{11}^{n} = \frac{\lambda}{2}\left[u_{01} - 2u_{11} + u_{21}\right]^{n+1/2} + \frac{\lambda}{2}\left[u_{10} - 2u_{11} + u_{12}\right]^{n}$$

$$u_{11}^{n+1/2}(1+\lambda) - \frac{\lambda}{2}[u_{01}] - \frac{\lambda}{2}[u_{21}] = (1-\lambda)u_{011} + \frac{\lambda}{2}[u_{10} + u_{12}]$$

first we keep on solving for horizontal points keeping implicit in x & Explicit in y, & then (we also use tri-diagonal (Block matrix) at Each step, and then solve for vertical points keeping Explicit in x solved previously.

Discretization for y direction.

$$u_{01}^{n} - u_{01}^{n+1/2} = \frac{\lambda}{2}\left[u_{-11} - 2u_{01} \; u_{11}\right]^{n+1/2} + \frac{\lambda}{2}\left[u_{00} - 2u_{01} + u_{02}\right]^{n}$$

$$(1+\lambda)u_{01}^{n} - \frac{\lambda}{2}u_{02}^{n} = \frac{\lambda}{2}\left\{\lambda u_{11}^{n} + (1-\lambda)u_{01}^{n} + \frac{\lambda u_{00}^{n+1/2}}{2}\right.$$

$\hookrightarrow$ B.C

Similar for u_{02}, u_{03}

Discretization for U_{12} for vertical direction

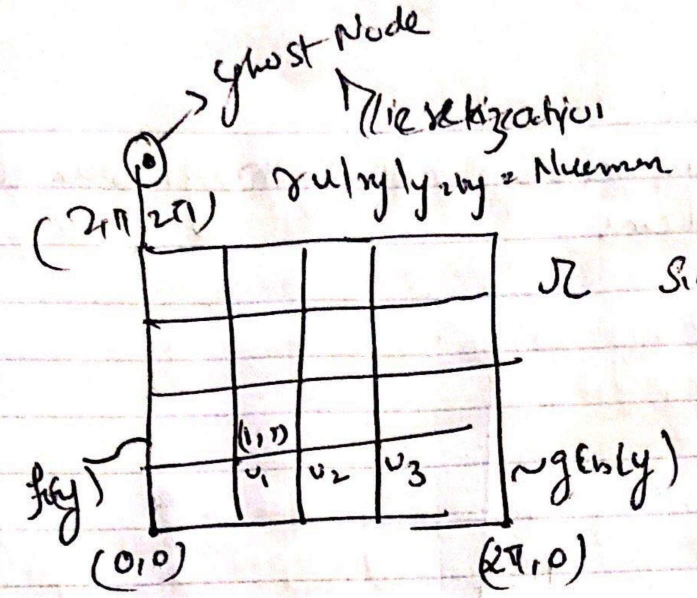
$$U_{12}^{n} - U_{12}^{n+1/2} = \frac{\lambda}{2}\left(U_{02} - 2U_{12} + U_{22}\right)^{n+1/2}$$

$$+ \frac{\lambda}{2}\left(U_{11} - 2U_{12} + U_{13}\right)^{n}$$

$$(1+\lambda)U_{12}^{n} - \frac{\lambda}{2}\left[U_{11}\right]^{n} + \frac{\lambda}{2}U_{13}^{n} = \frac{\lambda}{2}\left[U_{0,2} + U_{22}\right]^{n+1/2}$$

$$+ (1-\lambda)U_{12}^{n+1/2}$$

For Poisson equation using ghost node



Similar points as previous discretization.

for gauss seidel ~~theta~~ method when $\Delta x = \Delta y$

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 y}{\partial y^2} = f(x,y)$$

$$\frac{1}{\Delta x^2}\left[u_{j-1,k} - 2u_{j,k} + u_{j+1,k} + u_{j,k-1} + u_{j,k+1} - 2u_{jk}\right] = f(x_j, y_k)$$

$$-4u_{jk} = -\Delta x^2 f(x_j, y_k) - [u_{j-1,k} + u_{j+1,k} + u_{j,k-1} + u_{j,k+1}]$$

$$u_{jk} = \frac{1}{4}\left[\Delta x^2 f(x_j, y_k) + [\ \text{"}\ ]\right]$$

$u_{j,k+1} - u_{j,k-1} = 0$

$$\frac{u_{j,k+1} - u_{j,k-1}}{2\Delta y} = 0$$

$(u_{j,k-1} = u_{j,k+1})$

for neuman B.C, we will have

$$u_{jk} = \frac{1}{4}\left[\Delta^2 x\, f(x_j, y_k)\right) + \left[u_{j-1,k} + u_{j+1,k} + 2u_{j,k-1}\right]$$

$$u_{jk}^{n+1} = \frac{1}{4}\left[u_{j-1,k}^{(n+1)} + u_{j+1,k}^{n} + u_{j,k-1}^{(n+1)} + u_{j,k+1}^{(n)} \right] + \frac{\Delta^2}{4} f_{jk}$$

Thus updating values while moving in x direction

& for SOR it will be

$$u_{jk}^{n+1} = (1-\omega) u_{j,k}^{n} + \frac{\omega}{4}\left[u_{j-1,k}^{n+1} + u_{j+1,k}^{n} + u_{j,k-1}^{n+1} + u_{j,k+1}^{n} \right] + \frac{\omega \Delta^2}{4} f_{jk}$$

7. Description of numerical method used and pseudo code

For Poisson equation as mentioned earlier gauss seidel and sor method is used, both can be evaluated using the tri diagonal algorithm, if we are solving for the nodes in the x direction then we can update those nodes while using the method to solve for the next node, so updated values are used when jumping from one point to other in one direction for u_sol.

Algorithm used Gauss seidel and SOR

1.Tri diagonal matrix is created using the discretization of the equation keeping in mind the changes to be made for the ghost node.

2.Matrix of the solution is initiated using number of discretized points and Bc is updated and a grid is made for the number of discretized points to be solved

3. This particular code calculates each column using direct equations created while implementing tri diagonal algorithm and then calculate the maximum error comparing it with provided and tolerance and iterate until updated.

4. Right hand function is calculated separately and to be called for the time when solving for similar node.

5.Spectral radius is calculated for the grid convergence

6. Loop iterates unless error is less then tolerance which is calculated using the method provided in the notes.

7. For SOR method discretized equation changes and algorithm is similar apart from the SOR factor added for the calculations of the results and the spectral radius.

For diffusion equation explicit scheme and ADI method is used direct equation solving technique is used while updating the solution at every time step, for the ADI scheme tri diagonal algorithm is used for first solving in x direction keeping y nodes explicit and then solving in y direction keeping x nodes explicit.

Algorithm for ADI and explicit

1. Explicit method differentiate the equation for Neumann and Dirichlet bc and try to solve for each node using the time loop , and iteration goes on for the time where temperate is independent of time .
2. For ADI scheme tridiagonal algorithm is used for which it calculates all the values in x direction using tridiagonal matric and then in y direction again

using tridiagonal matrix by making sure each bc is used on the right side by creating a funvction f and r for x and y direction .

Results and discussion

1. Gauss seidel method and SOR method to evaluate solution for Poisson equation.

   a. Specification of the parameters used in the analysis

```
Ax=0;                          upper left corner of the matrix u_new
Bx=2*pi;                       maximum length of the domain in the x direction
By=2*pi;                       maximum length of the domain in the x direction
Ay=0;                          upper left corner of the matrix u_new
N=3                            Number of discretized points
dx                             DELTA x
dy                             DELTA Y
del_x                          Value at each discretization in x direction
del_y                          Value at each discretization in y direction
u_loop_sol                     initial value of the solution
f_sol                         Right hand side of the equation computed at each iteration
tolerance=10^-06;              Provided Tolerance
cal_u_vec                     solution vector for each column update
error                         error calculated at each iteration
cal_values_norm               Norm of the values to calculate the error
matrix_A                      Tri diagonal matrix for each iteration
K                             Diagonal of tri-diagonal matrix
U                             Lower matrix
L                             upper matrix + diagonal matrix
L_iv                          Inverse of N
N_ins_p                       N_inverse*P
spectral_radius               Spectral radius of the N_iverse*p
D1                            tridiagonal without major diagonal
D                             Inverse of major diagonal
B                             Norm of that matrix_value calculated less then 1
u_bc1                         Dirichlet Boundary Condition 1
u_bc2                         Dirichlet Boundary Condition 2
u_bc3                         Dirichlet Boundary Condition 3

u_sol(N+2,N+2)                Initial matrix with boundary condition
u_new                         Updated solution
f(N+2,N+2)                    Function on right hand side of the matrix
u_step_update(N+2,N+2)        Extra values storage to check the error at each iteration
iteration                     Number of iterations
K                             Value to verify grid convergence
Sor_fac                       Sor factor used for SOR method
```

b. Effect of number of points used for discretization

Initially for both of the solution methods used number of points used to discretize the domain was 3 and then it was updated to 10 to observe the difference , number of points will increase the number of iterations but which will help us reach closer to the true solution. Increase in number of points will eliminate the truncation error , this is due to the fact that the equation we are solving for is not exactly the same to the one for which solution needs to be found, they all contain truncation error with the order of delta_x square, where delta_x is the distance between two neighboring points, as we increase N ,delta_x will certainly decrease and so will the error which is observed while Taylor series expansion of the discretized equation.

Computational cost will increase as well if we increase the discretization points.

This is array shown above is generated where domain is discretized with points N=3, and when discretized with N=10 the array is shown below ,as we see the more number of discretized points the more precise/true values are obtained with decreased truncation error, some time what happens is when we use course discretization solution tend to jump any non-linearities available but for more discretized domain we will get smooth curves with non- linearity captured to the best. which can observed in the contour plot used as a comparison for poisson equation results solved using Gauss seidel method.

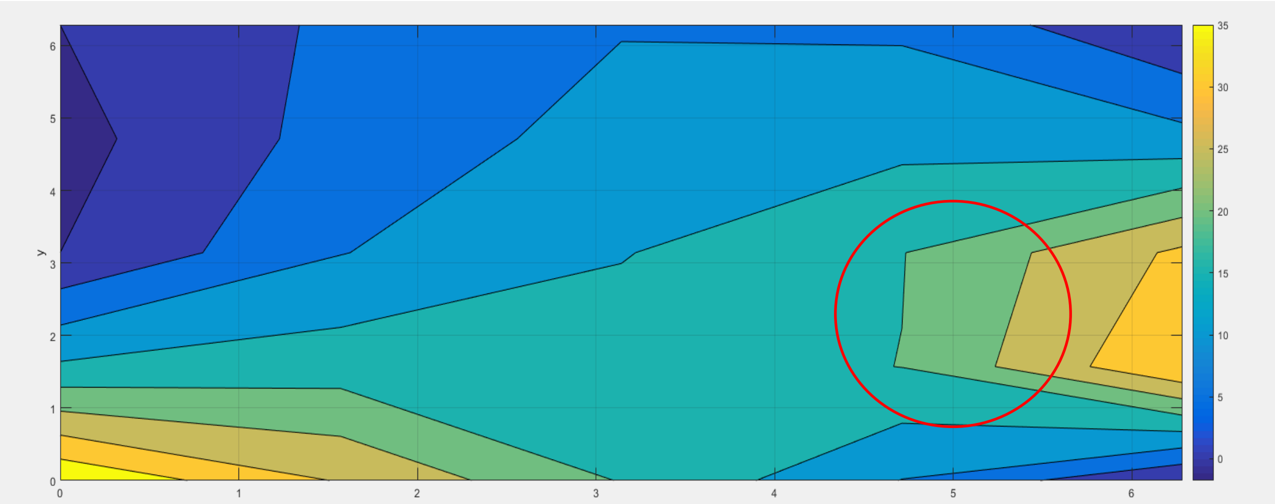


Fig -1b Solution of the domain with 3 points for the last increment after error has decrease to a tolerance of10e-06

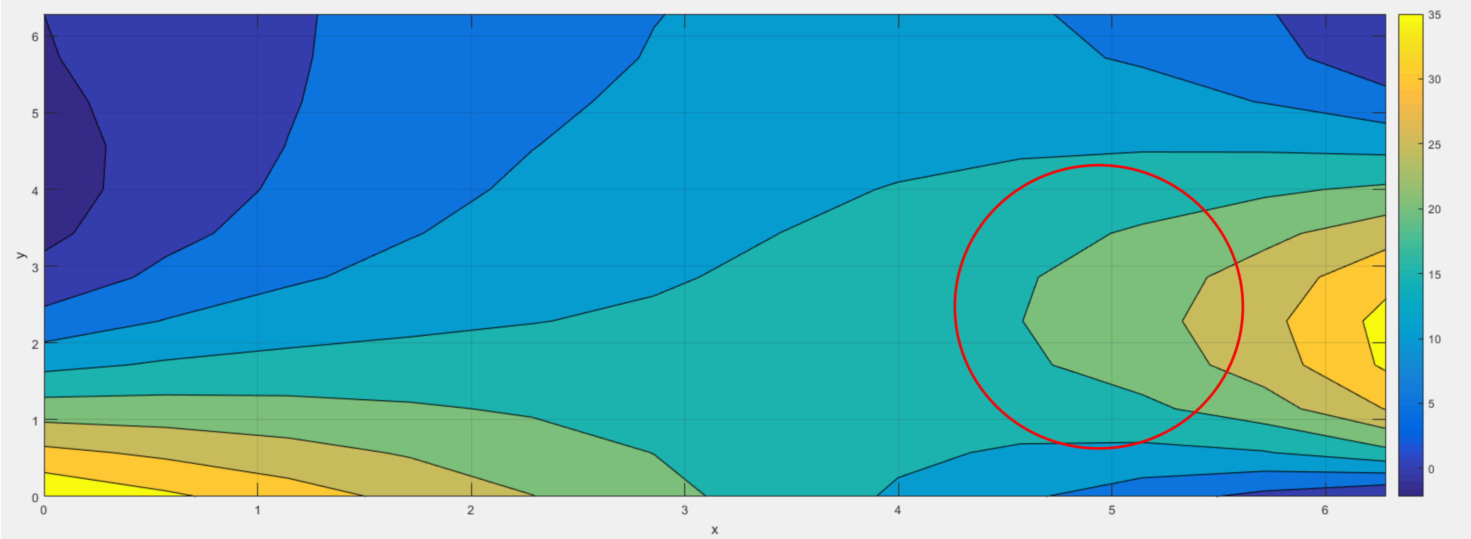


Fig -2b Solution of the domain with 3 points for the last increment after error has decrease to a tolerance of10e-06

This is the solution for the domain when discretized with less number of point and non smoothness in the results is observed which speaks loud for the larger truncation error .Marked region in red shows the comparison and how fine discretization capture non linearities better then course discretization.

c. Grid convergence study

For both the cases GS and SOR grid convergence study was done using the tridiagonal matrix which is used at every iteration to compute the results of all the points in every column of the solution matrix.

First Separation of N and P matrix was done which is inturn used to find the spectral radius of N_inverse*P.

N for the case Gauss seidel was D+L which is (diagonal+ lower triangle)of the tridiagonal matrix and for the SOR case N was w^-1*(D+L).

P for the GS was -U which is the upper triangle of the tri diagonal matrix and for the SOR case it was w^-1*(1-w)*(D)-U

SOR factor changes with the number of points used for discretization as it depends on delta_x, after going through literature it was states it has to be between 1 and 2 but what will be the optimal value for the SOR factor was a big question and best optimal option was

W= 2/1+sin(pi*dx) where dx stands for h or delta_x
Spectral radius value for SOR case for the N=3 discretization was "0.165719246397235" which is less then 1 which shows that the calculations done further are valid and convergent, value for spectral radius for the GS case was "0.213388347648318"
Which also shows that it is convergent.

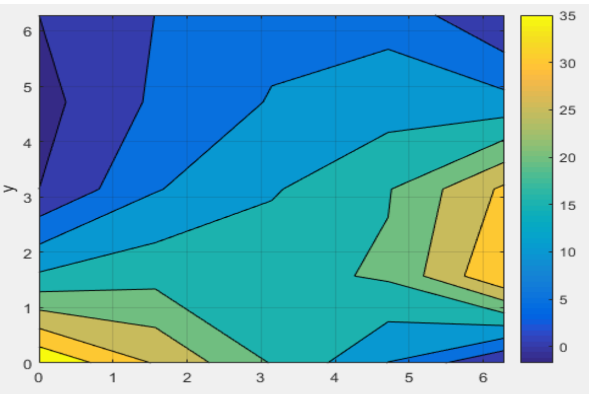
To check convergence we can always use spectral radius such as K> m/-log10(rho), where K stands for least number of iterations and m stands for the integer in the tolerance which was 6 for our case and rho is the spectral radius.

Both satisfies this condition and the values for iteration for both cases is higher then generated K value which proofs that the grid is convergent, and solution can be trusted.

e) Comparison of results with expected theory.

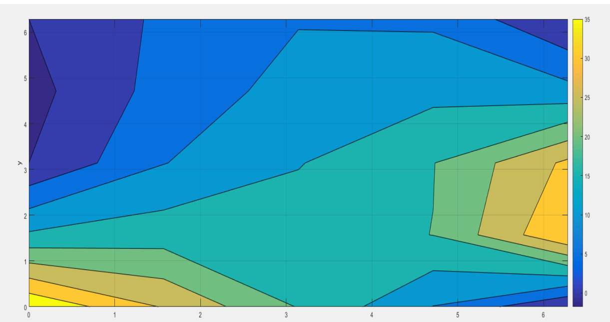
1.Expected theory says that Sor uses weighted residual which is its SOR_factor this factor help us to converge the solution at faster rate hence less number of iterations are required, which can be observed with the code. For GS method total number of iteration was 40 and for the case of SOR total iteration were close to 25 when discretized using 3 points.

2. Solution was asked to be computed for F==0 and the contour plot is shown below with the comparison with the solution of GS method with a non zero rhs function. Results are satisfactory.



39.4784	29.6088	19.7392	9.8696	0
15.7024	18.3201	18.3093	20.6833	34.8821
6.0434e-16	9.6597	14.4947	19.6724	31.0063
-1.7447	5.8239	10.3373	12.5053	11.6274
0	5.0433	8.5255	8.3840	0

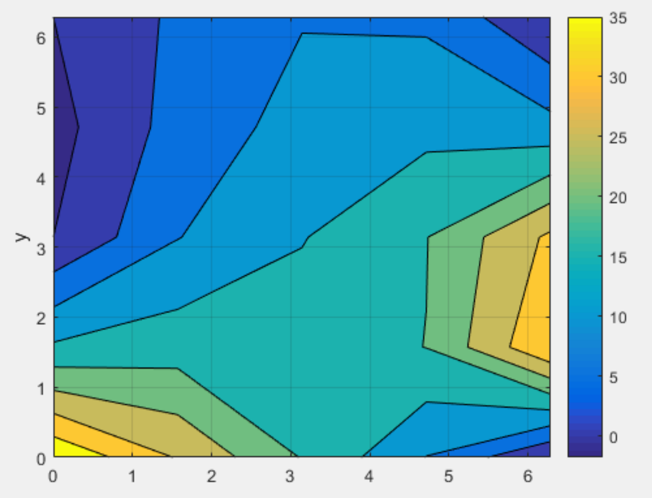
Figure e1: with forcing function equivalent to zero, highlighted values are the ones effected by the non zero values of the forcing function.



1	2	3	4	5
39.4784	29.6088	19.7392	9.8696	0
15.7024	17.7129	17.4507	20.0762	34.8821
6.0434e-16	9.8345	14.7419	19.8472	31.0063
-1.7447	6.8831	11.8352	13.5644	11.6274
0	5.8625	9.6840	9.2032	0

Figure e2: forcing function not equivalent to zero

3 Whatever may be the iterative method used, the solution must not be far from reality and neither the solution must be different for any of the iterative method used, results must match. Hereby contour plot and the discretized solution for both the methods used is being provided, which proofs the capability of the codes to produce valid results.



39.4784	29.6088	19.7392	9.8696	0
15.7024	17.7126	17.4503	20.0759	34.8821
6.0434e-16	9.8340	14.7413	19.8469	31.0063
-1.7447	6.8826	11.8346	13.5641	11.6274
0	5.8621	9.6836	9.2029	0

Figure e3: solution using SOR method

Compare e2 and e3 for verification of both codes producing right results with SOR using less number of iteration.

f) Verify order of special accuracy with discretization.

To verify this we know that the error generated between true solution and approximate solution should be equivalent of the square if the del_x, where delta_x is the total length/ number of segments.

But due to unavailability of the true solution other way to do it is to check the solution at two successive iteration and the max of the error between the two must not exceed square of the distance between the neighboring points. Below is solution matrix provided for N=3 and for iteration K=2 and K=3 using matrix A and C and their difference represented by matrix C, we can observe that max value is “2.779” which is lower than the prescribed one, which can be utilized as am assurance that yes GRID IS SPACIALLY ACCURATE.

A =

39.478417604357404	29.608813203268099	19.739208802178702	9.869604401089360	0
15.702444449187499	12.750629475245200	10.473907306028799	14.758362180770400	34.882061265337299
0.000000000000001	3.861228459871480	6.838299881403910	14.898004578374600	31.006276680299798
-1.744716049909720	1.399515214297970	4.605424040247400	9.178821502626480	11.627353755112400
0	0.902647490284099	3.492907746649200	5.508631026867730	0

Matrix at iteration ==2

C =

39.478417604357404	29.608813203268099	19.739208802178702	9.869604401089360	0
15.702444449187499	14.517904414262100	12.964652006046100	17.152264492338499	34.882061265337299
0.000000000000001	5.733949869832580	9.617314800446570	16.784008675433402	31.006276680299798
-1.744716049909720	2.845260898505590	6.956957077080470	10.691788178896200	11.627353755112400
0	2.330174970957640	5.486096039381990	6.747192976206530	0

Matrix at iteration=3

0	0	0	0	0
0	1.767274939016900	2.490744700017300	2.393902311568100	0
0	1.872721409961100	2.779014919042660	1.886004097058802	0
0	1.445745684207620	2.351533036833070	1.512966676269720	0
0	1.427527480673541	1.993188292732790	1.238561949338800	0

Difference between matrix A and C and maximum value marked with red is close to the value of the squared discretization. Which is “2.466”.

RESULTS AND DISCUSSION FOR DIFFUSION EQUATION using Explicit and ADI method

a. Specification of the parameters used in the analysis

```
Ax=0;                          upper left corner of the matrix u_new
Bx=2*pi;                       maximum length of the domain in the x direction
By=2*pi;                       maximum length of the domain in the x direction
Ay=0;                          upper left corner of the matrix u_new
N=3                            Number of discretized points
dx                             DELTA x
dy                             DELTA Y
del_x                          Value at each discretization in x direction
del_y                          Value at each discretization in y direction
T                              Total time provided
Lamda                          del_t/det_x^2
Alpha                          increment
u_bc1                          Dirichlet Boundary Condition 1
u_bc2                          Dirichlet Boundary Condition 2
u_bc3                          Dirichlet Boundary Condition 3
u_sol(N+2,N+2)                 Initial matrix with boundary condition
u_new                          Updated solution
tolerance=0;                   to stop when change is temp with respect tp t is zero
iteration=0;                   iteration count
increment=0;                   time increment
Matrix_A                       Tridiagonal matrix
Timestep                       Value of each time step
```

b. Effect of number of points used in the iteration.

Comparison for the change of grid size is done explicit method which again shows that a solution is closer to a true solution when fine grid is use for discretization as compared to the course grid where non linearities can be showed in proper way. Comparison of the N=3 and N=10 is done below.

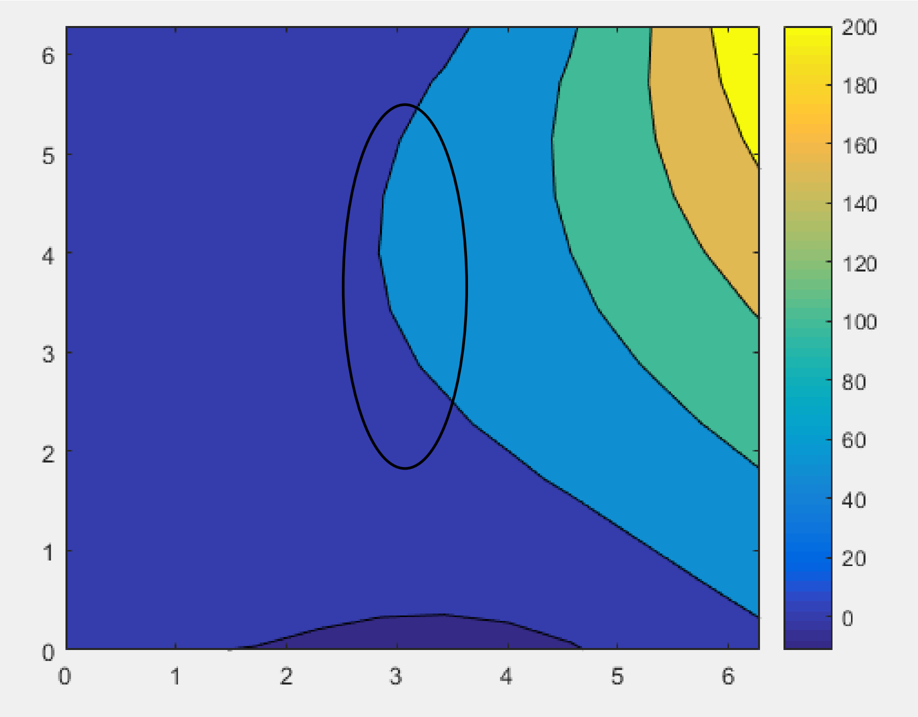


Figure b1 when N=10 for explicit case

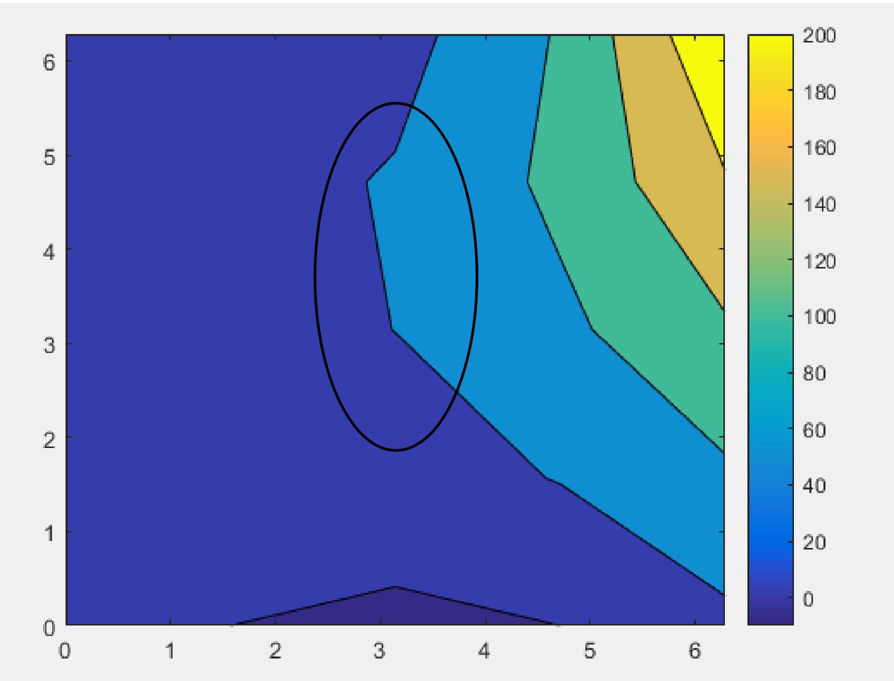
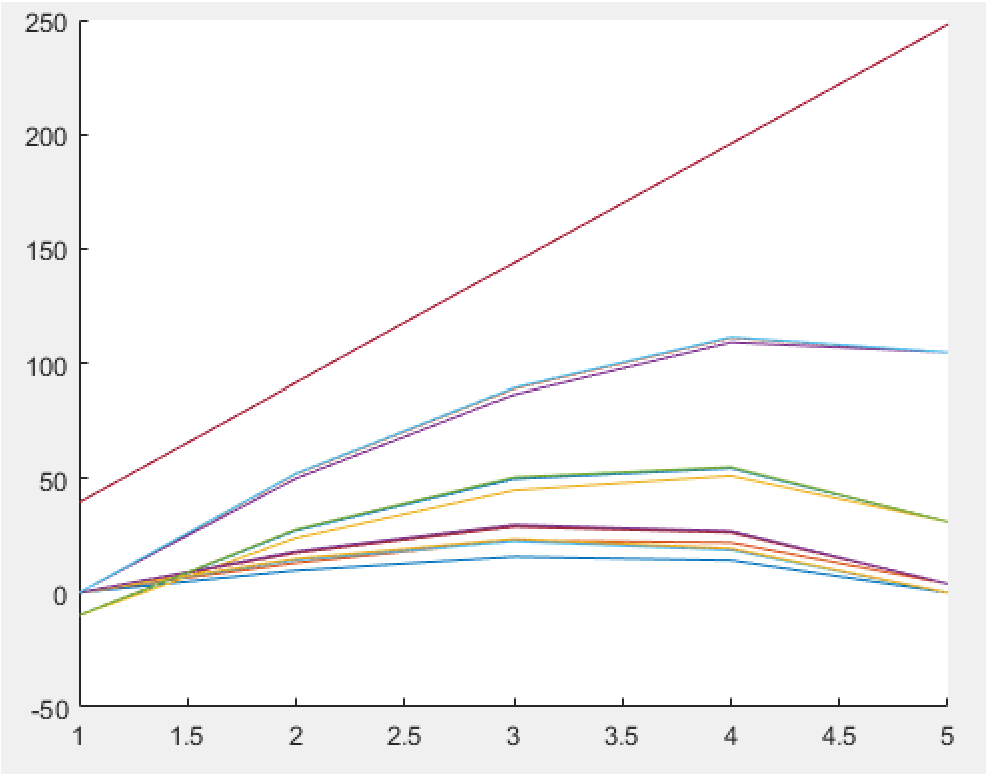


Figure b2 when N=3 for explicit case

D. effect of diffusive CFL

Plot show here shows the effect of diffusion as the values can be seen to respond for the calculated solution matrix.



e) comparison of the results

Scheme used below don't have an exact match bcoz of the error produced in explicit scheme is more as the truncation error is of order 1 in time discretization and in scheme such as ADI it is of order of the special discretization, thus making it more closer to the true solution, explicit scheme do not update the values due to which it deviates from the true solution more as compared to any two level or three level time scheme, these schemes can be made stable by using appropriate lamda, but for ADI it is unconditionally unstable.

Here, surface plot of the final solution is displayed which seems valid with the Boundary and initial conditions provided.

1.0e+02 *

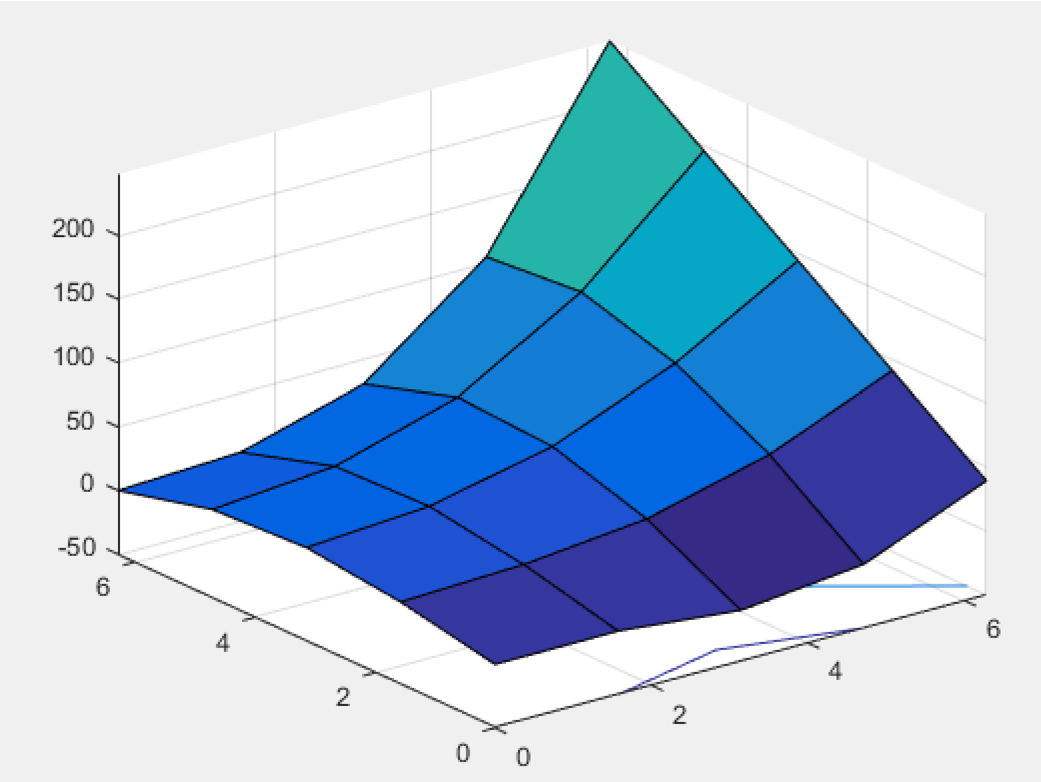
0	0.000000000000000	-0.098696044010894	-0.000000000000000	0.394784176043574
0.148951681778609	0.180810314760478	0.277062974073260	0.521813858836450	0.916213665638677
0.234186427961340	0.297227108847172	0.504324000312361	0.893979005083905	1.437643155233780
0.193340586398831	0.269588124000922	0.549027460564438	1.112135184746826	1.959072644828882
0	0.038757845850375	0.310062766802998	1.046461837960119	2.480502134423986

Final solution of the explicit method

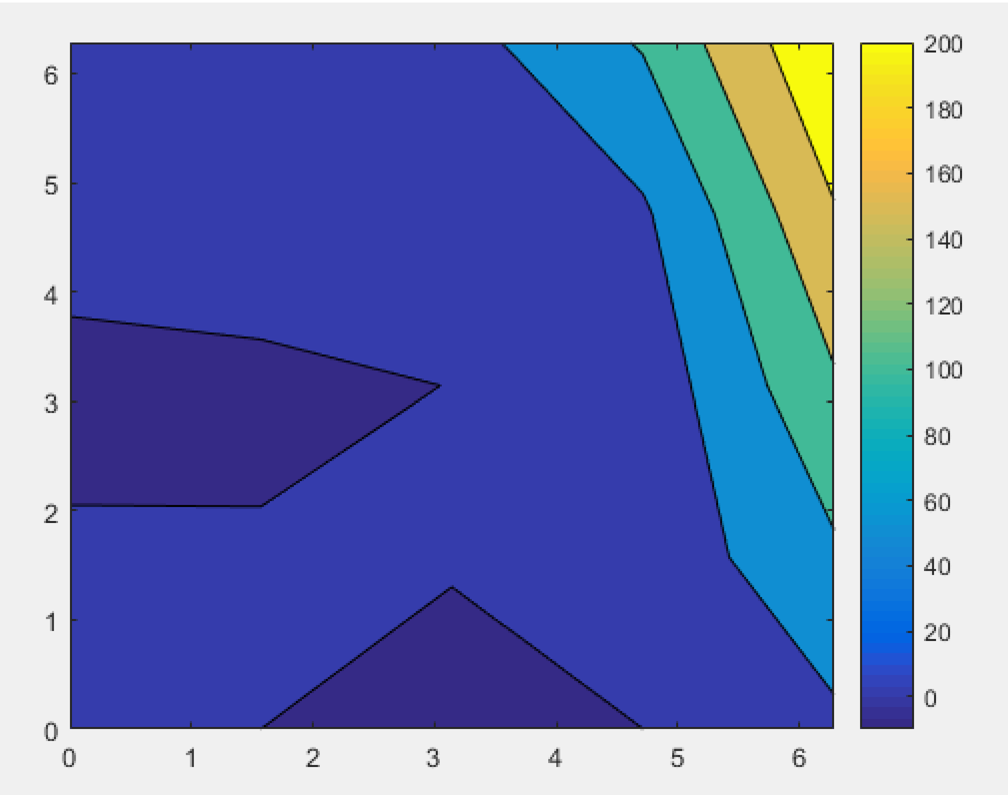
1.0e+02 *

0	0.000000000000000	-0.098696044010894	-0.000000000000000	0.394784176043574
0.001605384025441	0.002186314682898	0.020446080850017	0.158107641037127	0.916213665638677
-0.003651452111712	-0.005184005620884	0.000325881635466	0.188088658105634	1.437643155233780
0.005436306636227	0.014116181122503	0.073170899223547	0.427633681463266	1.959072644828882
0	0.038757845850375	0.310062766802998	1.046461837960119	2.480502134423986

Final solution of the ADI method.



Surface plot for the Explicit scheme



Contour plot for ADI scheme

f) special accuracy

For explicit scheme we know the error must be of order of det_t which is 0.6 in the case with T=10 and N=3 the error check was done from the multi dimensional array, which can be observed to be of same order, which stands for one of the proof for special accuracy.

```
ans =

                   0                   0                   0                   0                   0
  -0.447954611909937  -0.250687544866377  -0.320286847234410  -0.105358692998237                   0
  -0.382619681692578  -0.587321057236462  -0.273362511426072  -0.247102716534613                   0
  -0.448148964384878  -0.250749672871862  -0.320424276329916  -0.105384427392664                   0
                   0                   0                   0                   0                   0
```

Specification of the computer used

Item	Value
OS Name	Microsoft Windows 7 Enterprise
Version	6.1.7601 Service Pack 1 Build 7601
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	DFRIM-04
System Manufacturer	Dell Inc.
System Model	Precision Tower 5810
System Type	x64-based PC
Processor	Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz, 3601 Mhz, 6 Core(s), 12 Logical ...
BIOS Version/Date	Dell Inc. A16, 11/17/2016
SMBIOS Version	2.8
Windows Directory	C:\Windows
System Directory	C:\Windows\system32
Boot Device	\Device\HarddiskVolume1
Locale	United States
Hardware Abstraction Layer	Version = "6.1.7601.24408"
User Name	Not Available
Time Zone	Central Daylight Time
Installed Physical Memory (RAM)	64.0 GB
Total Physical Memory	63.9 GB
Available Physical Memory	52.1 GB
Total Virtual Memory	128 GB
Available Virtual Memory	102 GB
Page File Space	63.9 GB
Page File	C:\pagefile.sys