Istanbul Technical University
Faculty of Aeronautical and Astronautical Engineering

Computational Fluid Dynamics

Project #1

Ümit Yelken
511161187
April 15, 2018

List of Figures

Contents

1 Introduction

[0,1]x[0,1] two dimensional laplace equation is given like

$$\frac{\partial^2\psi}{\partial x^2} + \frac{\partial^2\psi}{\partial x^2} = 0 \tag{1}$$

Boundary conditions are given

$$\psi(x,0) = 0 \tag{2}$$
$$\psi(x,1) = (x - x^2) \tag{3}$$
$$\psi(0,y) = 0 \tag{4}$$
$$\psi(1,y) = 0 \tag{5}$$

We need to discretizate our domain 41x41, 81x81 and 161x161 format to solve it. In this project two dimensional laplace equation is solved with iteratives methods and direct numerical method. In order to obtain solution laplace equation is discretized by second-order finite difference method with uniform mesh.

2 Analytical Solution

We have a problem which has dirichlet boundary conditions. To solve this problem we need to use seperation of variable method.

$$\psi(x,y) = X(x)Y(y) \tag{6}$$

From the three homogeneous boundary conditions, we see that

$$Y(0) = 0 \tag{7}$$
$$X(1) = 0 \tag{8}$$
$$Y(0) = 0 \tag{9}$$
$$\frac{\partial}{\partial x^2}XY + \frac{\partial}{\partial y^2}XY = 0 \tag{10}$$
$$X''Y + XY'' = 0 \tag{11}$$
$$\frac{X''}{X} = -\frac{Y''}{Y} = -\lambda^2 \tag{12}$$
$$\tag{13}$$

λ must be + in this equation
Hence;

$$X'' - \lambda^2 X = 0 \tag{14}$$
$$X(x) = Acos(\lambda x) + Bsin(\lambda x) \tag{15}$$
$$X(0) = 0 \Longrightarrow A = 0 \tag{16}$$
$$X(1) = 0 \mapsto Bsin(\lambda) = 0 \Longrightarrow \lambda = n\pi \tag{17}$$
$$\tag{18}$$

$$X_n = B_n sin(n\pi x) \tag{19}$$
$$Y'' - \lambda^2 Y = 0 \tag{20}$$
$$Y(y) = Ce^{\lambda y} + De^{-\lambda y} \tag{21}$$
$$Y_n = C_n sinh(n\pi y) \tag{22}$$
$$\psi_n(x,y) = X_n(x)Y_n(y) = D_n sin(n\pi x)sinh(n\pi y) \tag{23}$$
$$\psi_n(x,y) = \sum_{n=1}^{\infty} \psi_n(x,y) = \sum_{n=1}^{\infty} D_n sin(n\pi x)sinh(n\pi y) \tag{24}$$
$$y = 1 \Longrightarrow x - x^2 = \sum_{n=1}^{\infty} D_n sin(n\pi x)sinh(n\pi y) \tag{25}$$

From orthogonal property we will multiply it with $sin(m\pi x)$

$$\int_0^1 (x - x^2)sin(m\pi x)dx = D_m sinh(m\pi) \int_0^1 sin^2(m\pi x)dx$$
$$D_m = \frac{2(-\pi m sin(m\pi) - 2cos(m\pi + 2)}{sinh(m\pi)\pi^3 m^3}$$
$$m = 2n - 1$$
$$\psi_n(x,y) = \frac{8}{\pi^3} \sum_{n=1}^{\infty} \frac{sin((2n-1)\pi x)sinh((2n-1)\pi y}{sinh((2n-1)\pi)(2n-1)^3}$$

These calculations on matlab are presented in my homework folder

3 Fully implicit solution algorithm and a direct solver (LU factorization)

We will use second order finite difference for each term to solve it.

Our domain is [0.1]x[0,1]. We will use z_{ij} to find others. Also BCs are given in the problem. Starting point will be our boundary conditions.

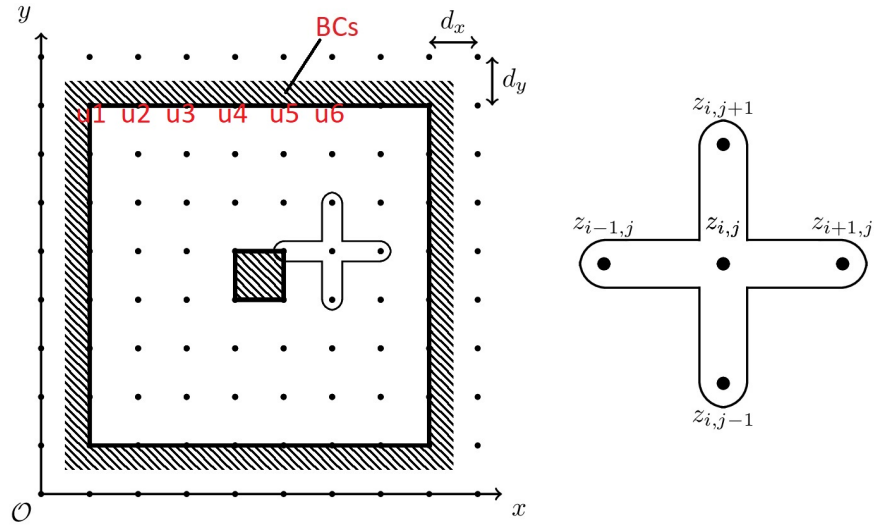

Figure 1: Domain

$$\frac{\partial^2\psi}{\partial x^2} = \frac{\psi(x+h,y) - 2\psi(x,y) + \psi(x-h,y)}{(\Delta h)^2} \tag{26}$$

$$\frac{\partial^2\psi}{\partial y^2} = \frac{\psi(x,y+h) - 2\psi(x,y) + \psi(x,y-h)}{(\Delta h)^2} \tag{27}$$

$$\tag{28}$$

For these equations because of our mesh h values will be equal.

$$\Delta h = \Delta h \tag{29}$$

$$\frac{\partial^2\psi}{\partial x^2} + \frac{\partial^2\psi}{\partial x^2} = 0 = \frac{\psi(x,y+h) + \psi(x+h,y) - 4\psi(x,y) + \psi(x-h,y) + \psi(x,y-h)}{(\Delta h)^2} \tag{30}$$

$$\psi(x,y+h) + \psi(x+h,y) - 4\psi(x,y) + \psi(x-h,y) + \psi(x,y-h) = 0 \tag{31}$$

In the question [0,1][0,1] domain will discretizate into 41x41, 81x81, 161x161.
41x41 after discretization 1681x1681 matrix.
81x81 after discretization 6561x6561 matrix.
161x161 after discretization 25921x25921 matrix.

3.1 LU Decomposition

In order to use LU depomposition we will define triangular matrix and diagonal element.

$$M = L * U$$

$$y = L^{-1} f$$

$$x = U^{-1} y$$

I used [L,U] = lu(A) decomposition code in matlab.In above equation, x symbolize u_{ij} values. To find ψ values u_{ij} values must be find first. Spy of L and U components are shown in figures.
$[L, U] = lu(A)$ returns an upper triangular matrix in U and a permuted lower triangular matrix in L such that $A = L*U$. Returned value L is a product of lower triangular and permutation matrices.

$$X^{n+1} = -D^{-1}[LX^n + UX^n] + D^{-1}b$$

Above equation presents how we define X values with using previous values of it.
Sample of A matrix is shown in Fig.2 for 5x5.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	1	0	0	0	1	-4	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	1	0	0	0	1	-4	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	1	0	0	0	1	-4	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	1	0	0	0	1	-4	1	0	0	0	1	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	1	0	0	0	1	-4	1	0	0	0	1	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	1	0	0	0	1	-4	1	0	0	0	1	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	-4	1	0	0	0	1	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	-4	1	0	0	0	1	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	-4	1	0	0	0	1	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure 2: Sample A Matrix For 5x5

After LU decomposition our L and U component should look like:

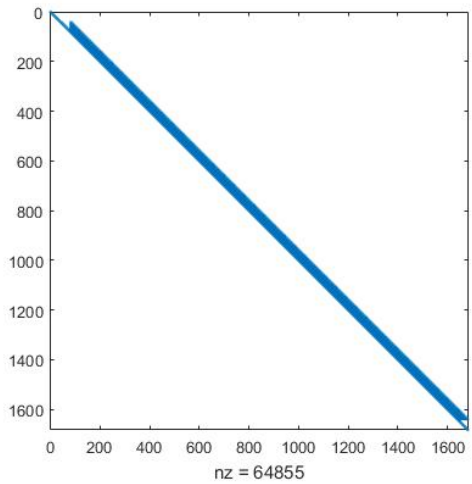

Figure 3: U Component

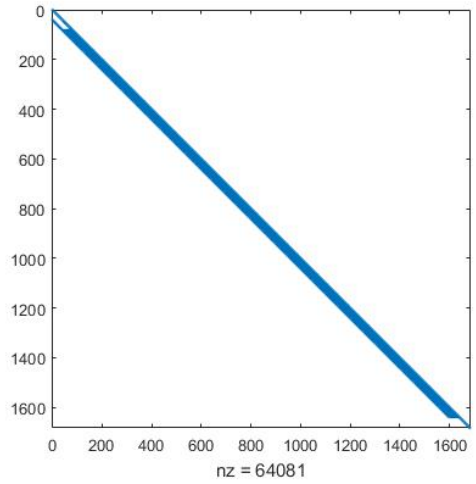

Figure 4: L Component

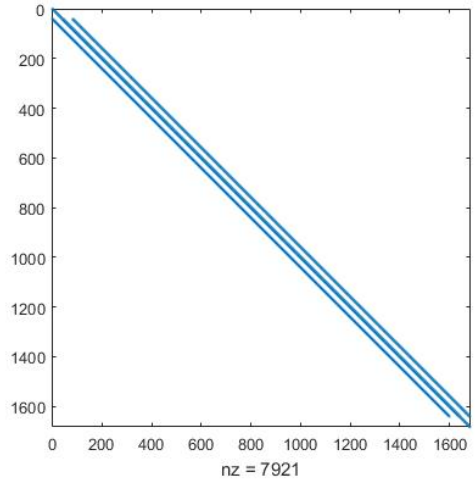

Figure 5: M Component

For question 1 matlab code is presented in my homework folder.
After LU decomposition meshes are shown in figures. Our domain is [0.1]x[0,1].

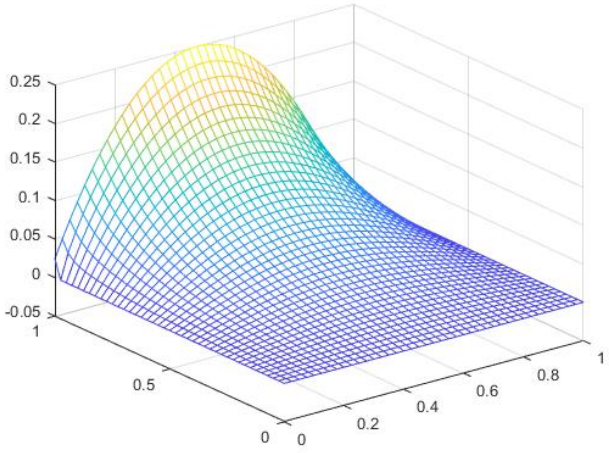

Figure 6: Generated Mesh for 41x41

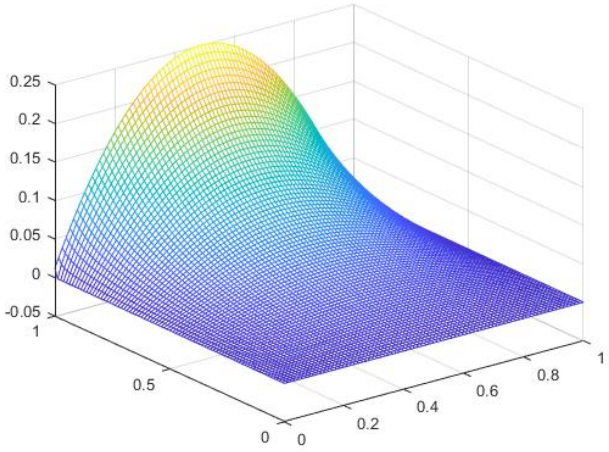

Figure 7: Generated Mesh for 81x81

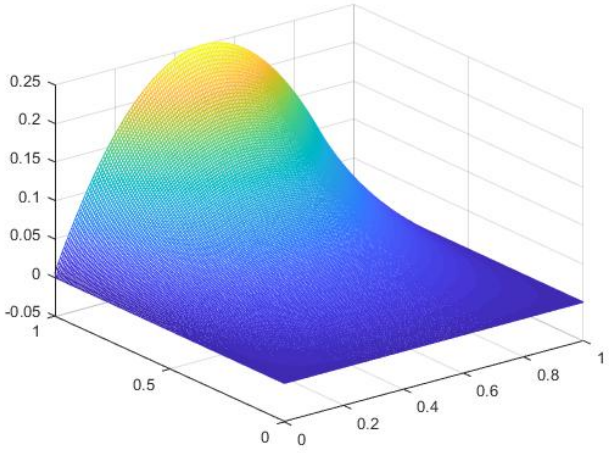

Figure 8: Generated Mesh for 161x161

After LU decomposition, error matrix is exported in csv format in my homework folder.

3.2 log-log Scale

In my algorithm analytic solution calculated for each 41x41, 81x81, and 161x161. The solution which coming from these calculations is exported a file. They are in my homework folder. After LU decomposition, with using these datas mesh space and log-log scale calculated and plotted.

Matrix	log(Δx)	log(error)	Δx
41x41	1.602059991	4.638544901	0.025
81x81	1.903089987	5.244053266	0.0125
161x161	2.204119983	5.890655994	0.00625

Figure 9: Log-Log Data

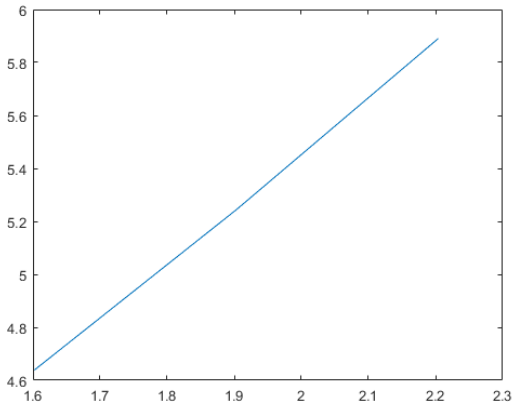

Figure 10: Log(Δx) vs Log(error)

Slope of this graph is 2.07

4 Jacobi, Gauss-Seidel, SOR Algorithms

4.1 Jacobian Algorithm

Jacobian algorithm is using previous values of point. For jacobian algorithm we will use diagonal elements of M matrix so we will define residual matrix and diagonal matris of M. First value of x is chosend randomly. With using matlab code of iterative solution the value of x is closing right value of x. Iteration solver stops working when error rate is smaller than error which is defined in the matlab code. The solver define new value of x with using previous value of it.
Our equation for matlab code

$$A = L + D + U$$

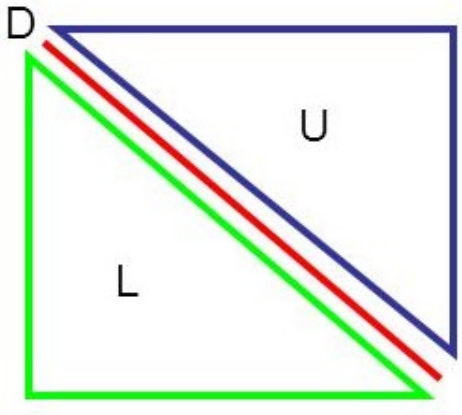

Figure 11: Jacobi Matrix

When we multiply x^{k+1} with diagonal of A this will be equal multiply of b+(D-A) with last value of x
So

$$Dx^{k+1} = b + (D - A)x^k$$

Rearrange it

$$x^{k+1} = D^{-1}b - D^{-1}(Lx^k + Ux^k)$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ a_{21} & 0 & 0 \\ a_{31} & a_{32} & 0 \end{bmatrix} + \begin{bmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{bmatrix} + \begin{bmatrix} 0 & a_{12} & a_{13} \\ 0 & 0 & a_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

Figure 12: (L+D+U)x=b

4.2 Gauss Seidel Algorithm

Unlike Jacobi Method, Gauss-Seidel Algorithm is using less memory. For gauss seidel algorithm we will use LU. Our equation is shown.

$$Ax = b$$

$$(L + D + U)x = b$$

$$x^{k+1} = -(D + L)^{-1}Ux^k + (D + L)^{-1}b$$

$$E = -(D + L)^{-1}U$$

$$f = -(D + L)^{-1}b$$

Gauss-Seidel algorithm converges more rapidly when comparing with jacobi algorithm.

4.3 Succesive Over Relaxation Algorithm

We need to decompose matrix to Lower, upper and diagonal element.

$$M = L + U + D$$

For SOR algorithm we will define ω which needs to be greater than 0 and smaller than 2. I take three values to see convergence of SOR algorithm, these are $\omega = 1.4$, $\omega = 1.7$ and $\omega = 1.9$.
To see convergence rates comparison of these different values please look Fig.16

$0 < \omega < 1$ Under Relaxation
$1 < \omega < 2$ Over Relaxation
$2 < \omega$ It does not converge

$$x^{k+1} = (D + \omega L)^{-1}(\omega b - (\omega U + (\omega - 1)D)x^k)$$

$$0 < \omega < 2$$

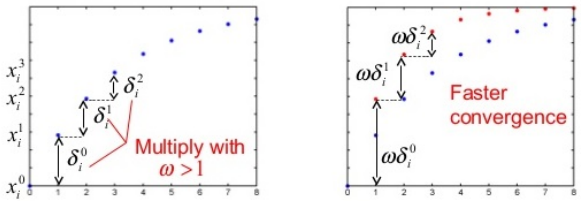

Figure 13: SOR Convergence

Source code written for all cases(41x41, 81x81, 161x161)

4.4 Error Function vs Mesh Space

Calculated value is shown in figure.

GRID SIZE	MESH	JACOBIAN	GAUSS-SEIDEL	SOR
41x41	0.0250	3.68500E-05	3.68500E-05	3.68500E-05
81x81	0.0125	3.26500E-05	2.69500E-05	2.69500E-05
161x161	0.0063	2.24500E-05	2.24500E-05	2.24500E-05

Figure 14: Spatial Convergence Rate

Increasing mesh value provides more accurate value.
Initial errors are identical for all methods like shown in figure 14.

4.5 Comparison Their Convergence Rates with The Iteration Number

When compare their convergence rates with iteration number, SOR has minimum time to reach under error. Their iteration numbers to reach under error:

Method	41x41	81x81	161x161
Jacobi	6197	23903	101586
Gauss Seidel	3091	11934	43624
SOR	265	526	3062

Figure 15: Comparison their convergence rates

And their graps for 41x41

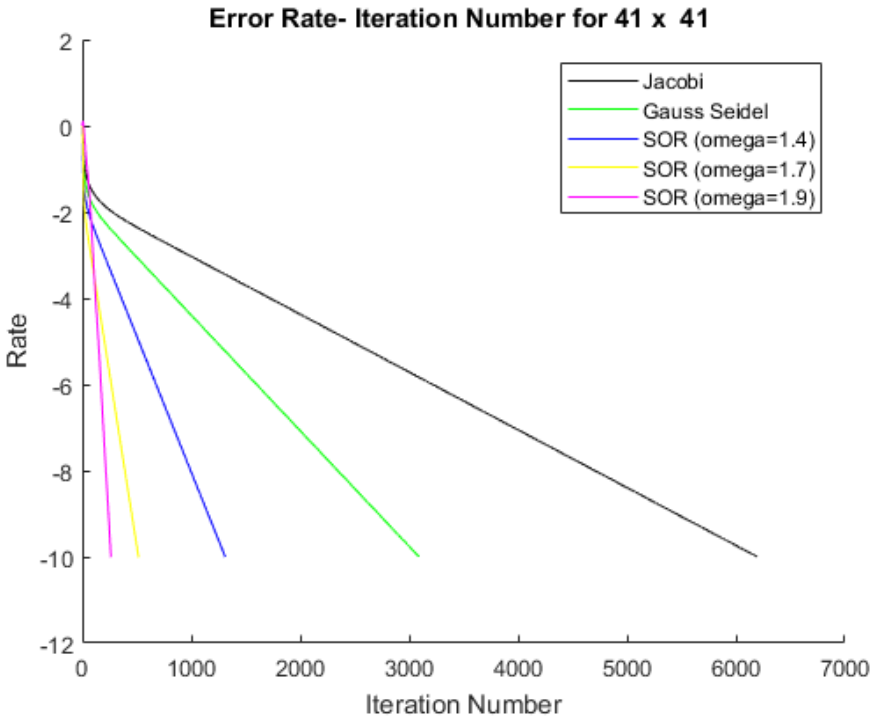

Figure 16: 41x41 Convergence Rates

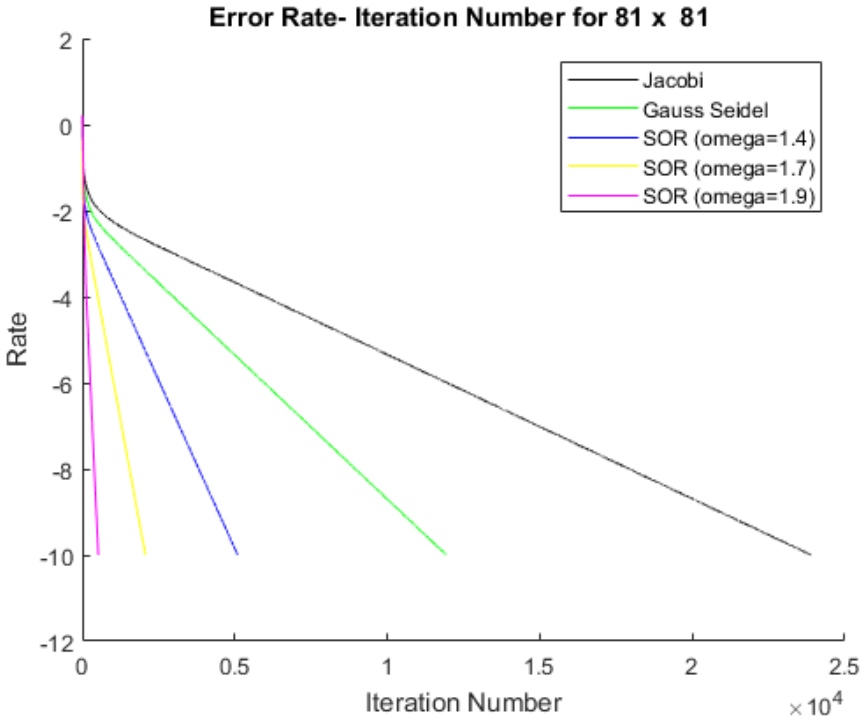

Figure 17: 81x81 Convergence Rates

5 4th Order Approximation

To solve 4th order equation we use Taylor series considering second order derivatives. Solving:

$$\psi(x_0-\Delta x)=\psi(x_0)-\Delta x\frac{\partial\psi}{\partial x}+(\frac{(\Delta x)^2}{2!})\frac{\partial^2\psi}{\partial x^2}-(\frac{(\Delta x)^3}{3!})\frac{\partial^3\psi}{\partial x^3}+ \tag{32}$$

$$(\frac{(\Delta x)^4}{4!})\frac{\partial^4\psi}{\partial x^4}-(\frac{(\Delta x)^5}{5!})\frac{\partial^5\psi}{\partial x^5}+(\frac{(\Delta x)^6}{6!})\frac{\partial^6\psi}{\partial x^6} \tag{33}$$

$$\psi(x_0+\Delta x)=\psi(x_0)+\Delta x\frac{\partial\psi}{\partial x}+(\frac{(\Delta x)^2}{2!})\frac{\partial^2\psi}{\partial x^2}+(\frac{(\Delta x)^3}{3!})\frac{\partial^3\psi}{\partial x^3}+ \tag{34}$$

$$(\frac{(\Delta x)^4}{4!})\frac{\partial^4\psi}{\partial x^4}+(\frac{(\Delta x)^5}{5!})\frac{\partial^5\psi}{\partial x^5}+(\frac{(\Delta x)^6}{6!})\frac{\partial^6\psi}{\partial x^6} \tag{35}$$

For $+2\Delta x$ difference:

$$\psi(x_0+2\Delta x)=\psi(x_0)+2\Delta x\frac{\partial\psi}{\partial x}+(\frac{(2\Delta x)^2}{2!})\frac{\partial^2\psi}{\partial x^2}+(\frac{(2\Delta x)^3}{3!})\frac{\partial^3\psi}{\partial x^3}+ \tag{36}$$

$$(\frac{(2\Delta x)^4}{4!})\frac{\partial^4\psi}{\partial x^4}+(\frac{(2\Delta x)^5}{5!})\frac{\partial^5\psi}{\partial x^5}+(\frac{(2\Delta x)^6}{6!})\frac{\partial^6\psi}{\partial x^6} \tag{37}$$

For -$2\Delta x$ difference:

$$\psi(x_0-2\Delta x)=\psi(x_0)-2\Delta x\frac{\partial\psi}{\partial x}+(\frac{(2\Delta x)^2}{2!})\frac{\partial^2\psi}{\partial x^2}-(\frac{(2\Delta x)^3}{3!})\frac{\partial^3\psi}{\partial x^3}+ \tag{38}$$

$$(\frac{(2\Delta x)^4}{4!})\frac{\partial^4\psi}{\partial x^4}-(\frac{(2\Delta x)^5}{5!})\frac{\partial^5\psi}{\partial x^5}+(\frac{(2\Delta x)^6}{6!})\frac{\partial^6\psi}{\partial x^6} \tag{39}$$

Hence we have totally:

$$\frac{\partial^2\psi}{\partial x^2}=\frac{-\psi_{i-2,j}+16\psi_{i-1,j}-30\psi_{i,j}+16\psi_{i+1,j}-\psi_{i+2,j}}{12\Delta x^2} \tag{40}$$

When we calculate it for y direction and consequently we have:

$$\frac{\partial^2\psi}{\partial y^2}=\frac{-\psi_{i-2,j}+16\psi_{i-1,j}-30\psi_{i,j}+16\psi_{i+1,j}-\psi_{i+2,j}}{12\Delta y^2} \tag{41}$$

$\Delta x=\Delta y=h$ Summing of Eq.40 and Eq.41

$$\frac{\partial^2\psi}{\partial x^2}+\frac{\partial^2\psi}{\partial y^2}=\frac{-\psi_{i-2,j}+16\psi_{i-1,j}-\psi_{i,j-2}+16\psi_{i,j-1}-60\psi_{i,j}+16\psi_{i,j+1}-\psi_{i,j+2}+16\psi_{i+1,j}-\psi_{i+2,j}}{12h^2} \tag{42}$$

4th order solution file is presented in my hw folder.

6 Homework Folder

In my homework folder ‘jacobigausssortogether.m‘ file is showing all iterative solutions datas like iteration numbers and errors. In addition to show these iterative solutions seperately i created files for each iterative solutions. These are ‘jacobi.m‘,‘ SOR.m‘ and ‘gaussseidel.m‘ . ‘sparseandLU‘ file is showing how i create matrix A and how i make LU decomposition. ‘loglogcomparisonplot.m‘ file is to create log(Δx)-log(error) also this file uses data which is coming from analytical calculation. This analytical solution creates by ‘analyticcalcu4141.m‘ , ‘analyticcalcu8181.m‘ and ‘analyticcalcu161161.m‘ also these matlab files export analytical solution to ‘4141analtics.mat‘ , ‘8181analytic.mat‘ and ‘161161analytic.mat‘ in the matrix form. ‘4thorder.m‘ is for 4th Order Approximation calculation.