Scientific Computing Assignment 1

Kaixin Hu

April 2016

1 Introduction

Partial differential equation,i.e.PDE, is an equation involving partial derivatives of an unknown function with respect to more than one independent variables.

This report explores the finite difference method to solve PDEs. This method introduces a grid of mesh points throughout the problem domain in space and time, and replaces all the derivatives in the PDE by finite difference approximations, and then seeks an approximate solution values at each of the mesh points.[1] This report concentrates on two kind of PDE: time dependent diffusion equation and time independent diffusion equation. For the former, the Wave equation and two-dimensional time dependent diffusion equation are explored respectively. For the latter the two-dimensional time dependent diffusion equation with and without a sink object in the space domain are explored respectively.

2 Theory

2.1 Time Dependent Diffusion Equation

2.1.1 Wave equation

Wave equation is modelled to describe the vibrations of a uniform string, the transport of voltage and current along a lossless transmission line, the pressure and flow rate of a compressible liquid or gas in a pipe, sound waves in gases or liquids, and optical waves. In this example, the domain of the wave equation is a string.

The wave equation in one dimension space has the form:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}, 0 \leq x \leq L, t \geq 0, \tag{1}$$

with given initial conditions:

$$c_t(x, t = 0) = 0, c(x, t = 0) = g(x), 0 \leq x \leq L$$

and boundary conditions:

$$c(x=0,t)=0, c(x=L,t)=0, t\leq 0,$$

2.1.2 periodic boundary conditions

This part will analyse the two-dimensional time dependent diffusion equation in following form:

$$\frac{\partial c}{\partial t} = D \bigtriangledown^2 c \tag{2}$$

with given initial conditions:

$$c_t(x,y,0)=0, 0\leq x\leq L, 0\leq y<L$$

and boundary conditions:

$$c(x,y=0,t)=0, c(x,y=L,t)=1, t\leq 0,$$

and periodic boundary conditions:

$$c(x=0,y,t)=c(x=L,y,t), t\leq 0,$$

And it can be proven by analytic methods that for $t\rightarrow 1$ the concentration profile is simply a straight line:

$$\lim_{t\rightarrow\infty} c(x,y)=y$$

2.2 Time independent Diffusion Equation

If we are not very much interested in the time development of the concentration profile (i.e. the transient behavior), but only in the steady state. Then, we explore the Laplace Equation, which takes the form as follows:

$$\bigtriangledown^2 c=0 \tag{3}$$

And the boundary conditions and periodic boundary conditions are the same as those of periodic boundary conditions.

3 Methods

3.1 Time Dependent Diffusion Equation

Explicit time-stepping procedure is adopted to solve the time dependent diffusion equation. By explicit, it means the formular for the solution values at the next time step involves only past information. By time-stepping procedure, it means the method adopted for this part is a fully discrete finite difference method. I introduce a gird of mesh points through the problem domain in space and time, and seek an approximate values at each mesh point. Assuming

that there is N interval in the x- and y- directions, there is $\Delta x = 1/N$ and $\Delta y = 1/N$, and $x = i\Delta x, y = j\Delta x$, where i and j=0,1,2,...,N.

So as this method aims to find solution to the concentration at these discretized space and time points.So simplicity,it usually adopts the following definition:

$$c(i\Delta x; k\Delta t) = c_i^k$$

or

$$c(i\Delta x; j\Delta y; k\Delta t) = c_{i,j}^k$$

For the one-dimension space equation, the resulting array of approximate values represent a discrete sample of points in the (x,t) plane. For the two-dimension space equation, the the resulting array of approximate values represent a discrete sample of points in the (x,y,t) coordinate system.

3.1.1 Wave equation

To implement the **finite difference method**, Equation 1 can be transformed into Equation 4 and 5 to calculate the non-boundary points, using centered difference approximations for both $\frac{\partial^2 u}{\partial t^2}$ and $\frac{\partial^2 u}{\partial x^2}$. For boundary points, the initial conditions and boundary conditions can be implemented onto directly.

$$\frac{c_i^{k+1} - 2c_i^k + c_i^{k-1}}{(\Delta t)^2} = \frac{c_{i+1}^k - 2c_i^k + c_{i-1}^k}{(\Delta x)^2}, i = 1, 2, ..., N-1. \tag{4}$$

which can be rearranged into the explicit recurrence form:

$$c_i^{k+1} = 2c_i^k - c_i^{k-1} + c(\frac{\Delta t}{\Delta x})^2(c_{i+1}^k - 2c_i^k + c_{i-1}^k), i = 1, 2, ..., N-1. \tag{5}$$

This stencil is shown in Fig 1, as follows:

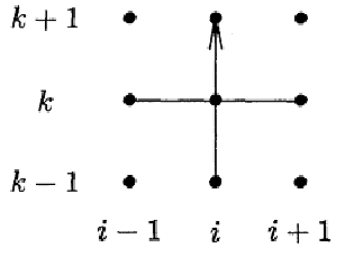

Figure 1: Explicit time stepping method for wave equation[1]

The parameters are as follows:
(1). Diffusion coefficient: c=1.
(2). Length: L=1. Time: T=1.

(3). For the initial conditions, the wave equation is explored with three versions of g(x) expressions: first, g(x)=sin(2x); Second, g(x)=sin(5x); And third, g(x)= sin(5x) if $1/5 \leq x \leq 2/5$; else g(x)= 0.

(4). Time step $\Delta t = 0.01$.

(5). Number of intervals in the x axis: N=100. So the interval length $\Delta x = L/N = 0.01$.

Things to explore:

First, the c(x,t) value will be plot at time 0.001, 0.01, 0.1 and 1 on the (x,c) plane.

Second, the c(x,t) value will be plot at time 0.001, 0.01, 0.1 and 1 on the (x,t) plane with various color representing the value of c(x,t).

Third, the animation of c(x,t) value on the (x,c) plane will shown with ".mp4" format.

Note, all these three explorations will be executed on the there initial conditions mentioned in the parameter setting part, respectively.

3.1.2 Two-dimensional time dependent diffusion equation

To implement the **finite difference method**, Equation 2 can be transformed into Equation 10 to calculate the non-boundary points, using centered difference approximations for $\frac{\partial^2 u}{\partial x^2}$, and forward difference approximation for both $\frac{\partial u}{\partial t}$. For boundary points on c(x,y=1,t), c(x,y=L,t), and (x,y,t=0) lines, the initial conditions and boundary conditions can be implemented onto directly. For boundary points on c(x=0,y,t) and c(x=L,y,t) lines, periodic boundary conditions is implemented by expanding the mesh points on (x,y) plane by one more column on the left side of the original matrix, as shown in Fig 2. And points value on this column is exactly the same as the Nth column (i.e. last but one column) in the original mesh point matrix. For the (N+1) column (i.e. last column), points value on it is exactly the same as the first column.

$$c_{i,j}^{k+1} = c_{i,j}^{k} + c\frac{\Delta t}{(\Delta x)^2}(c_{i+1,j}^{k} + c_{i-1,j}^{k} + c_{i,j-1}^{k} + c_{i,j+1}^{k} - 4c_{i,j}^{k}, \tag{6}$$

$$i = 1, 2, ..., N. j = 1, 2, .., N-1.$$

This scheme is stable if:

$$4c\frac{\Delta t}{(\Delta x)^2} \leq 1$$

The parameters are as follows:

(1). Diffusion coefficient: c=1.

(2). Length: L=1. Time: T=1.

(3). Time step is set as $\Delta t = 0.000025$, to ensure the stability of the scheme.

(4). Number of intervals in the x axis: N=100. So the interval length $\Delta x = L/N = 0.01$, $\Delta y = L/N = 0.01$.

Things to explore:

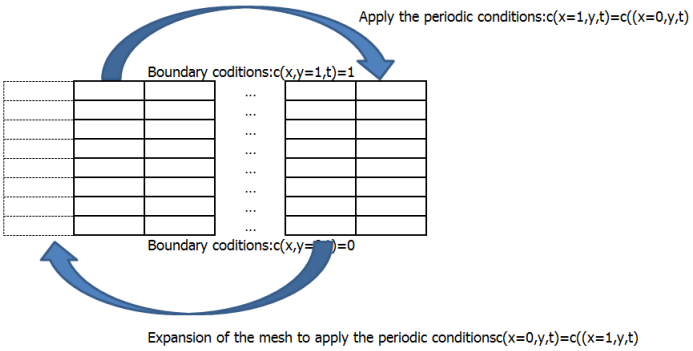

Figure 2: Application of the boundary and periodic boundary conditions

First, in (x,y) domain the concentration at each point will be plot with a color representing, and animation of c(x,t) value on the (x,y) plane will shown with".mp4" format through the time 0 to 1, with intervals of $T/\Delta = 40000$.

Second, test the the correctness of your simulation. c(y) is calculated at time: t = 0, 0.001, 0.01, 0.1, and 1 to compare with the analytic solutions. c(y) at those time will be plot on the (y,c) plane. Error=Simulated[c(y)]-analytic[c(y)] will also be plot on the (y,c) plane.

3.2 Time independent Diffusion Equation

Since there is no existence of the time, time-stepping procedure method is not useful in this case. Taking the same spatial discretization as before, and again applying the same 5-points stencil for the second order derivatives, Eq 3 is then transformed into the following form:

$$c_{i,j} = \frac{1}{4}(c_{i+1,j} + c_{i-1,j} + c_{i,j-1} + c_{i,j+1}), \tag{7}$$

For all the values of (i,j), except those on the y=0 or L boundaries. This part will concentrate on three iterative methods: the Jacobi Iteration, The Gauss-Seidel Iteration, and Successive Over Relaxation. The main difference between these three methods is the convergence efficiency due to the neighbor values it used whether or not involves that calculated in current iteration. And Successive Over Relaxation method further improves the efficiency by placing the weight of more 1 on the new calculated values, with respect to the previous calculated value for each mesh point.

For iteration method, superscript k is used to denote the k-th iteration. So for iteration k, the $c(i\Delta x; j\Delta y)$ is denoted as $c^k_{i,j}$.

3.2.1 Jacobi Iteration

The Jacobi iteration only uses the neighbour mesh point values calculated from previous iteration. So it is as following:

$$c_{i,j}^k = \frac{1}{4}(c_{i+1,j}^k + c_{i-1,j}^k + c_{i,j-1}^k + c_{i,j+1}^k), \tag{8}$$

The stopping condition is such that the solution is assumed to be converged if for all values of (i; j)

$$\delta = \underset{i,j}{max}|c_{i,j}^{k+1} - c_{i,j}^k| < \varepsilon$$

3.2.2 Gauss-Seidel Iteration

The Gauss-Seidel Iteration makes improvement over the Jacobi iteration, by making use of new values as soon as it has been calculated during the iteration. So it is as following:

$$c_{i,j}^k = \frac{1}{4}(c_{i+1,j}^k + c_{i-1,j}^{k+1} + c_{i,j-1}^{k+1} + c_{i,j+1}^k), \tag{9}$$

The Gauss-Seidel iteration is not a big improvement over the Jacobi iteration,in terms of the amount of iterations needed for convergence. However, the update can be performed in place, which is a big improvement in terms of storage. But Jacobi Iteration has the advantage of easy implementation on parallel computers, for the updating of the unknowns in current iteration can be done simultaneously. Gauss-Seidel Iteration can only be done in order, though this can also be solved by some tricks for the implementation of parallel computing.

The stopping condition is the same as Jacobi Iteration.

3.2.3 Successive Over Relaxation

The Gauss-Seidel iteration did not provide a huge improvement over the Jacobi iteration. For it introduces a weight, w, and it takes the following form:

$$c_{i,j}^k = \frac{w}{4}(c_{i+1,j}^k + c_{i-1,j}^{k+1} + c_{i,j-1}^{k+1} + c_{i,j+1}^k) + (1-w)c_{i,j}^k, \tag{10}$$

This method converges only for $0 < w < 2$. For $w < 1$ the method is called under relaxation. The new value is then the weighted average of the Gauss-Seidel method and the previous value. For w = 1 we recover the Gauss-Seidel iteration.

The stopping condition is the same as Jacobi Iteration and Gauss-Seidel Iteration.

The parameters are as follows:

(1). Length: L=1.

(2). Other parameters will be set in the results part, because in order to explore the convergence efficiency for three methods under different cases, various set of parameters are simulated.

Things to explore:

First, implement the Jacobi iteration, the Gauss-Seidel method and SOR, respectively.Test the methods by comparing the result to the analytical result, i.e. the linear dependence of the concentration on y.

Second, explore how the convergence measure δ depends on the number of iterations k for each of the tree methods, respectively.

Third, for the SOR method, explore the optimal w, with respect to the number of intervals the x and y axis are discretized into, i.e. N.

Forth, for the SOR method, one object is include into the computational domain. The object is sinks. And the infuence of this object on the resulting c value, as well as the convergence efficiency is explored and compared to the domain without such a sink object.

4 Results

4.1 Time Dependent Diffusion Equation

4.1.1 Wave equation

Fig 3 shows the amplitude on a vibrating string with three versions of initial condition of c(x,0)=g(x), in which first simulation, g(x)=sin(2x); Second simulation, g(x)=sin(5x); And third simulation, g(x)= sin(5x) if $1/5 \leq x \leq 2/5$; else g(x)= 0. And the amplitude along the x-axis is shown at various time: 0.001, 0.01, 0.1 and 1.

The animation of c(x,t) value on the (x,c) plane is also recorded with ".mp4" format, which can be found in the .zip file "time dependent diffusion" document.

4.1.2 Two-dimensional time dependent diffusion equation

Figure 4 shows the concentration at each point in the (x,y) space at time 0.001, 0.01,0.1,and 1 respectively, with a color representing. And animation of c(x,t) value on the (x,y) plane will shown with".mp4" format through the time 0 to 1, with intervals of $T/\Delta = 40000$, which can be found in the .zip file "time dependent diffusion" document.

And it can be observed that the source diffuses in space gradually through time, and at time 1, it seems that the concentration level is increasing with equal increment along the y direction from the sink to source, while in the x direction, it is homogenous. And this is proven true by the analytic solutions, in which c(y) is linearly dependent on y as t goes to infinity.

Therefore, c(y) is plot at time 0.001, 0.01, 0.1 and 1 on the (x,c) plane as shown in the left graph in Fig 4, and the error with respect to analytic results in the right graph.It can be observed that the value of concentration at time 1 is linear to the y value, and the error is very small for plots related to each time value at 0.001,0.01,0.1 and 1.

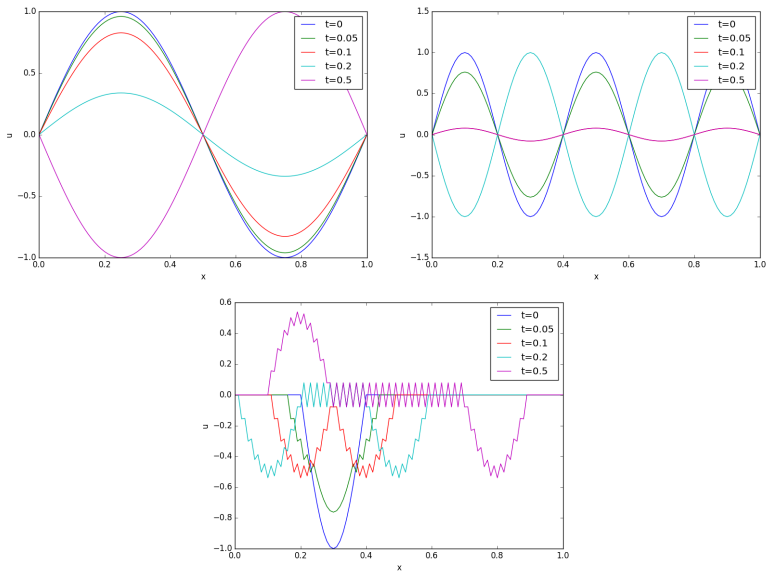

Figure 3: the amplitude on a vibrating string with different initial conditions: c(x,0) equals to sin(2x), sin(5x), and sin(5x) if $1/5 \leq x \leq 2/5$; else 0, respectively.

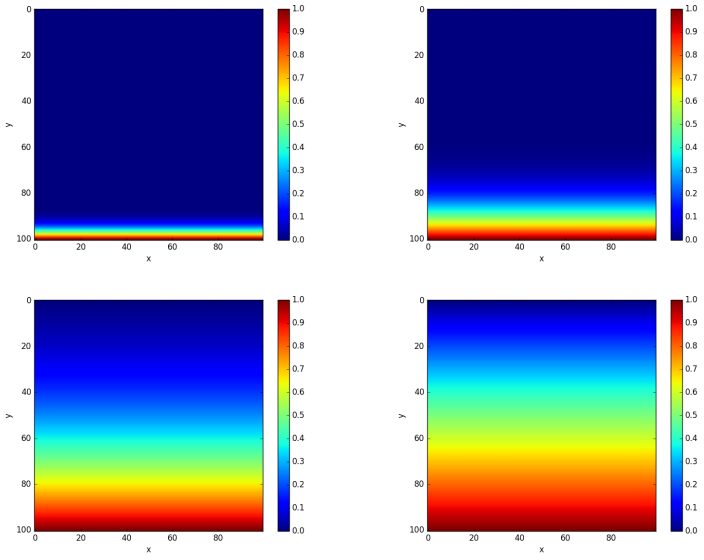

Figure 4: Two-dimensional time dependent diffusion equation: the concentration at each point in the (x,y) space at time 0.001, 0.01,0.1 and 1, respectively.

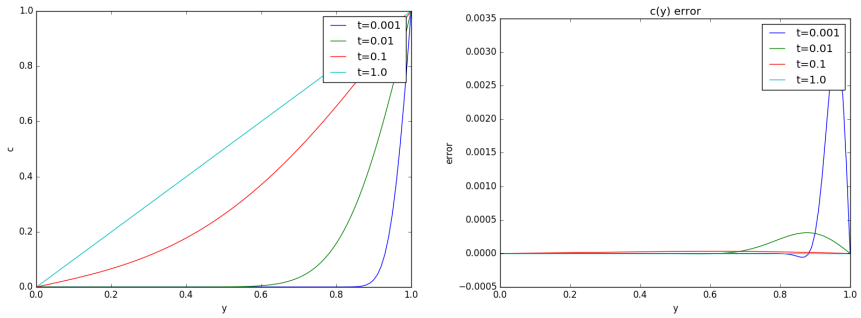

Figure 5: Two-dimensional time dependent diffusion equation: c(y) for the finite difference method and the error of it with the analytic results, at time 0.001, 0.01, 0.1 and 1, respectively.

4.2 Time independent Diffusion Equation

4.2.1 Exploration of different iteration methods for Laplace Equation

Fig 6 shows the linear dependence of the concentration on y for all the three methods, which is in consistent with the analytic results. The parameters are as follows:

(1). Length: L=1.

(2). $\varepsilon = 10^{-5}$

(3). Number of intervals in the x andy axis: N=50. So the interval length $\Delta x = L/N = 0.02$ and $\Delta y = L/N = 0.02$.

(4). Weight parameter for SOR method: w=1.8.

Note, that for Gauss-Seidel and SOR Iteration method there will be some inhomogenous distribution of concentration values along x axis due to the immediate use of the new values for non-boundary points, in contrast to the use of previous values for boundary points while no new values are ready when calculating. The difference between each mesh points column is shown in Fig 7. And it can be observed to be very small,within the order of magnitude of -6, which can be neglected.

Fig 8 shows the relationship between the convergence measure δ and the number of iterations k for each of the tree methods, respectively. For SOR, w is set as 0.5, 1.5 and 1.8 for three simulations. And it can be observed that SOR with w=0.5 has the lowest convergence rate, while w=1.8 has has the highest convergence rate. For other methods, the convergence efficiency in increasing order is Jacobi Iteration, Guass-Seidel Iteration, and SOR with w=1.5. For w = 1 , the SOR recover the Gauss-Seidel iteration, so it is not shown in the graph.

Fig 9 shows in the SOR method, how optimal w depends on the number of intervals in space discretization. By optimal, this reports measures the convergence rate. It can be observed that with N set to 30 and 40 , the optimal w is

1.8, and for N equals to 50 and 60, the optimal w is 1.9.

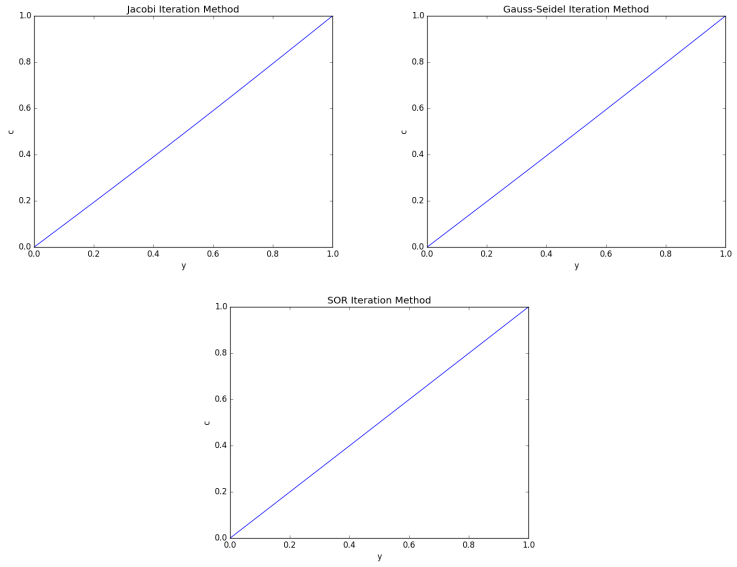

Figure 6: Laplace Equation: c(y) value with respect to y , solved with Jacobi Iteration, Gauss-Seidel Iteration, and SOR Iteration (w=1.8) method, respectively.

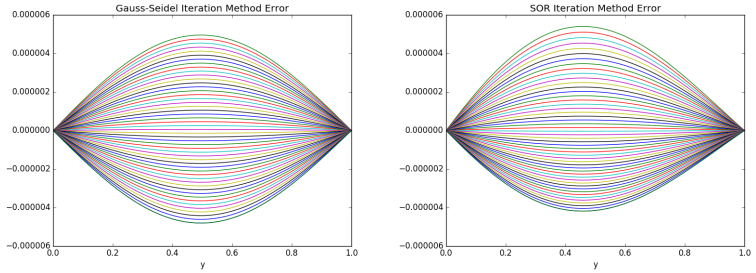

Figure 7: Laplace Equation: inhomogenous distribution of concentration values along x axis

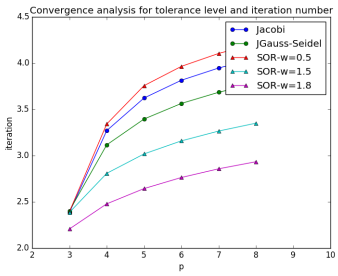

Figure 8: Laplace Equation: the relationship between the convergence measure δ and the number of iterations k for each of the tree methods, respectively.

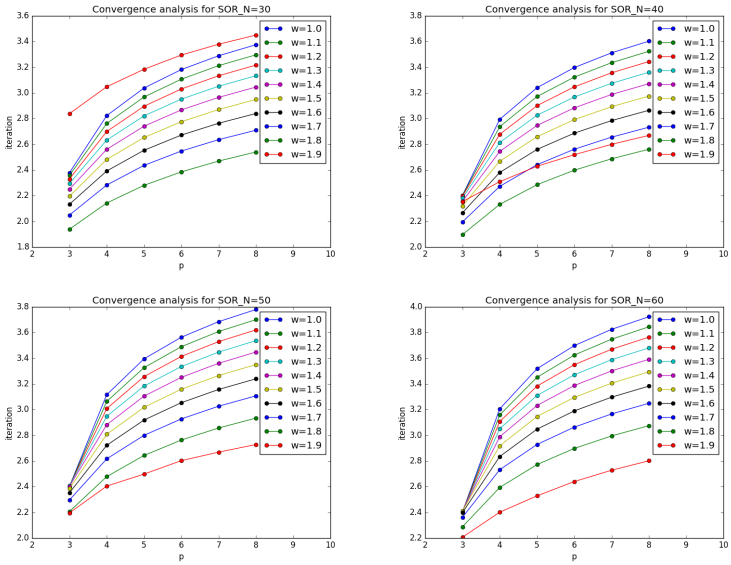

Figure 9: Laplace Equation: the relationship between the convergence measure δ and the number of iterations k for each of the SOR methods, with the interval for space coordinates N set as 30, 40, 50 and 60 respectively.

4.2.2 SOR Method with an object in the domain

Fig 10 shows the concentration at each point for a two-dimension domain with an object(sink) inserted. It can be observed that the concentration value depends on both x and y, and c(x,y) is no long a linear line with respect to y. This is due to the sink in the domain interrupt the diffusion process of the source from one boundary to another.

Fig 11 shows the impact of an inserted object(sink) on the optimal w for SOR method. It can be observed that the convergence efficiency improves a little compared with on objects in domain,which is shown in Fig 9. But the chosen of w by the object influence is not so clear under the set of parameters chosen this report.

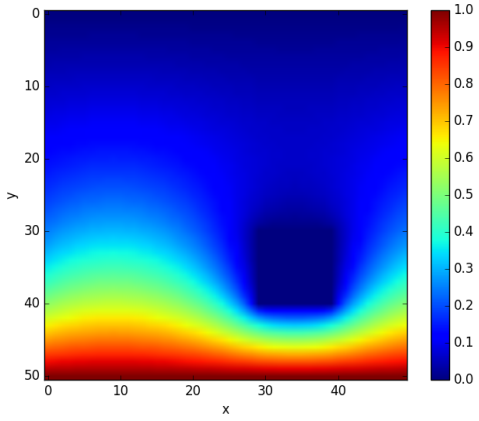

Figure 10: SOR Method: the resulting concentration for a two-dimension domain with an object(sink) inserted.

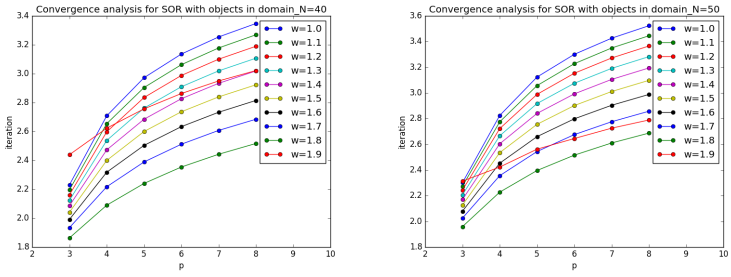

Figure 11: SOR Method: the influence on the optimal w respect to convergence rate of an inserted object(sink),with the SOR method.

5 Discussion

Most the explorations successfully settle the questions in the assignments.

For the time dependent diffusion equation, one dimension wave equation is solved with time stepping procedure and finite difference method, and the plot in (x,c) plane is plot at various time points. Animation of c value on x is also recorded through time 0 to 1. Two dimension time dependent diffusion equation is also solved with time stepping procedure and finite difference method, and the simulation is verified by comparing concentration value in respect to y with that resulting from an analytic method. And the errors are very small for at the chosen measuring time points. The concentration values for the mesh points on (x,y) plane is shown at chosen measuring time points, and the animation of concentration values through time form 0 to 1 is also recorded.

For the time independent diffusion equation, Laplace Equation is explored with three iteration methods, i.e. Jacobi Iteration, Gauss-Seidel Iteration, and SOR method. These methods are verified by plotting c(y) respectively to show its linear dependence on y, which is indicated by the analytic result. And these methods are further explored by comparing the convergence efficiency. For SOR method, the optimal w is analysed for various N (number of intervals for the space discretization). By optimal, it measures the convergence rate.

However there are some results indicating further work in order to completely figure out more accurate results. For the analysis of optimal w in SOR method for the time independent diffusion equation with an object inserted in. The influence of the object on the optimal w is not clear. Because with N set as 40 and 50, the optimal w is 1.8 for both cases with or without such objects. The solution may be to try more w between 1.70 and 1.99 with a smaller increment.

6 Extra section: Experience with Linux

My experience with Linux: I installed a virtual Machine, VMware Workstation Pro, and then run the Ubuntu on it. It is convenient to install some packages for python. But it is very different from the Window 7, which operating system I have been using for years. So I need some time to convert completely to Linux. To be honest, most of this assignment is completed on Windows Anaconda.But I will try to use Linux more as I am getting a little more familiar with it now.

References

[1] Michael T Heath. *Scientific computing*. McGraw-Hill New York, 2002.