2D-Diffusion using ADI method

Vignesh Jeganathan

CONTENTS

1. Introduction

This report shows my effort to solve the given diffusion problem. I used the method of manufactured solution to validate my code. Then, I used that validated code to solve the given problem. This report is divided into two major parts. The first part deals with validation of the explicit and ADI scheme. The second part shows the results obtained from the validated code.

2. Numerical Scheme

I used second order discretization for space. For the time, its first order in the explicit scheme. The ADI method is 2nd order in time.

3. Verification and Validation

For the problem $D_t - D_{xx} - D_{yy} = f$, the manufactured solution is $u = e^{t/\tau}\sin(kx)\sin(ky)$ and RHS function is $f = e^{t/\tau}\sin(kx)\sin(ky)(\frac{1}{\tau} + 2k^2)$. The initial condition is $u(x, y, 0) = \sin(kx)\sin(ky)$. Regarding the boundary conditions (BC), there are two scenarios I considered: 1) all the edges are homogeneous Drichlet condition or 2) one Neumann condition at $u(x = bx) = -e^{t/\tau}\sin(ky)$ and the rest 3 are homogeneous Drichlet conditions. For uniformity, I used k=2 and $\tau = 2$ in all the cases.

3.1. *Explicit Scheme*

In order to validate the explicit scheme, I used both the scenarios stated before: all Drichelt and 3 Drichlet and 1 Neumann BCs. Since I already know the analytical solution for this problem, I calculated the L2 norm and used it to find the order p using :

$$p = \frac{\log E(n, t) - \log E(2n, t)}{\log 2} \tag{3.1}$$

where n in the above equation is the number of nodes in x or y direction as the mesh is

BC	*n*	*L1(n)*	L1(2n)	Order
Drichlet	50	1.52e-3	3.73e-4	2.03
	40	2.39e-3	5.87e-4	2.03
	20	9.80e-3	2.34e-2	2.04
Neumann	50	1.80e-4	3.77e-5	2.27
	40	3.04e-4	6.27e-5	2.25
	20	1.71e-3	3.04e-4	2.34

Table 1: Order of Explicit method

BC	*n*	*L1(n)*	L1(2n)	Order
Drichlet	50	1.11e-4	1.65e-5	2.45
	40	1.82e-4	3.41e-5	2.42
	20	7.9e-4	1.82e-4	2.12
Neumann	50	1.57e-4	3.11e-5	2.33
	40	2.57e-4	5.35e-5	2.26
	20	1.17e-3	2.57e-4	2.19

Table 2: Order of ADI method

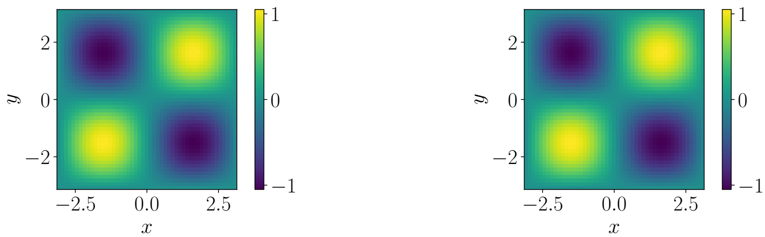

Figure 1: Contour plots for manufactured solutions k=2 *(a)* Analytical solution, *(b)* ADI with Drichlet and Neumann.

square. The table 1 shows the L2 norms after time t=0.1 for 2n and n meshes. The order is mostly 2, which is expected as I used the second order discretization. Also, one can see that the L2 norms are decreasing as we increase the number of points. This shows the criteria of grid convergence is met for explicit scheme.

3.2. *ADI Scheme*

In order to validate the ADI scheme, I followed the same procedure used in the explicit scheme. As expected I got the order 2 as shows in table 2. The L2 norm is decreasing as the mesh becomes finer. Hence, the grid convergence is met in this scenario too. Figure 1 compares the numerical solution (n=50, $\Delta t = 0.001$) with the analytical solution from method of manufactured solution after time t=0.1. We can see they look similar.

Δt	n	$L1(n)$	L1(2n)	Order
0.001	50	7.2e-4	2.9e-4	1.29
	30	1.8e-3	5.4e-4	1.73
	15	6.7e-3	1.8e-3	1.89
0.002	50	9.6e-4	5.4e-4	0.84
	30	2e-3	7.9e-4	1.32
	15	6.8e-3	2e-3	1.79
0.005	50	1.9e-3	1.3e-3	0.56
	30	2.9e-3	1.7e-3	0.75
	15	7.4e-3	2.9e-3	1.34

Table 3: Order of ADI for the given problem for various mesh size and time step sizes

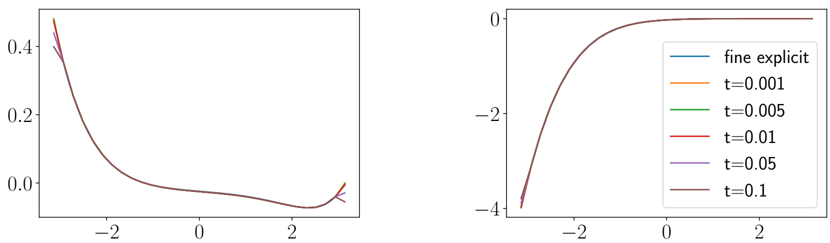

Figure 2: Comparing fine explicit solution with varying time step sizes for n= 30 after t=1 along (a)at x=0, (b) y=0.

4. Given Problem

4.1. *Comparing ADI with finely discretized explicit solution*

After validating both the implicit and explicit codes, I used the same codes for the given problem. Since the given problem's analytical solution is not known, I used finely discretized (n=300, $\Delta t = 1e - 5$) explicit scheme as the exact solution. The λ is set to 0.05 for this study. I calculated the L2 norms as I did during the validation process. We can see the order is nearly 2 for few cases from the table 3 when the time size is sufficiently smaller and mesh size is larger. I suspect, to increase the accuracy one should not simply make the mesh finer but also decrease the time step size. Since we are getting a significantly accurate results for the mesh size n= 30 for time step size of 0.001, we can fix mesh size as n=30 and study the effect of time step size now.

The contour plot is not an accurate way to study this system. Hence, I cut lines along x=0 and y=0 ie. the cut that splits domain into exact halves. Then, I plot the solution along those lines for n=30 and different time step sizes as shown in the figure. 2. This plot shows solution after time t=1. As stated previously, I used same n=30 and see how higher the time step size I can go for the ADI before my error starts becoming significant. Along x=0, the ADI solution's accuracy reduces as I decrease time steps. The error is visually apparent in the boundaries if Δt goes down from 0.05. This is expected because I assumed the $\lambda = 0.05$. In order to march in time properly the exponents of the boundary condition should not increase rapidly ie. λt should not increase faster. However, for the y=0 cut, there is not much noticeable error.

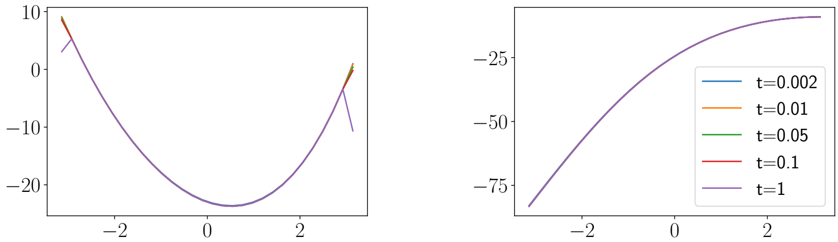

Figure 3: Comparing fine explicit steady state solution with varying time step sizes for n=30 along (a)at x=0, (b) y=0.

4.2. *Steady State Solution*

Since we are interested only in the steady state solution, we can safely use larger time steps to get to the steady state. This is good because we can get to the steady state with fewer time steps and hence save computation time. I used 1e-5 as the tolerance between the time steps ie. the moment the norm of the solution of current and the previous time step become less than 1e-5, the simulation will stop. The graphs 3 shows the comparison of the steady state solutions at (x=0) and (y=0) attained with different time step sizes. The time step can be as large as 0.1 to get reasonably accurate solution. The solution starts loses accuracy after Δt=1. This corresponds to CFL number of 22, which is much larger than what explicit method can offer.

5. Conclusion

We see that ADI scheme can use larger time steps and hence higher CFL which is much higher than what's possible with explicit scheme. I used git to record my code and the codes are written using python's Jupyter notebook, Both are attached in the end of this report