Project 1 Diffusion Equation

1 Project Objective

A Implement different numerical methods for diffusion equations

B Compare the results of these methods

2 Equation specification & Exact Solution

The project will focus on one specific diffusion equation:

$$u_t = \frac{1}{2} u_{xx} \quad t > 0,\ -\infty < x < +\infty$$

which is with 2π periodic both initial data $u(x,0) = \theta(x)$ and boundary conditions. Without loss of Generality, we assume that the boundary conditions are $u(0, t) = u(2\pi, t) = 0$ and $u_x(0, t) = u_x(2\pi, t)$; the initial data $\theta(x) = \sin(x)$

Therefore, the exact solution for this equation:

$$u(x,t) = A_0 + \sum_{n=1}^{\infty} \left[A_n \cos(nx) + B_n \sin(nx)\right] e^{-\frac{1}{2}n^2 t}, 0 \le x \le 2\pi$$

where

$$A_0 = \frac{1}{2\pi} \int_0^{2\pi} \theta(x) dx$$

$$A_n = \frac{1}{\pi} \int_0^{2\pi} \cos(nx)\theta(x) dx$$

$$B_n = \frac{1}{\pi} \int_0^{2\pi} \sin(nx)\theta(x) dx$$

By taking into account the exact form of $\theta(x)$, which is $\sin x$ as we have assumed, we get the solution for this equation:

$$u(x,t) = e^{-\frac{1}{2}t} \sin x$$

3 Notation

All of the following numerical methods are based on the grid notation below:

- For spacial variables x, the step of it is Δx ,...; the grid points are x_i , i = 0, 1, 2, ...,N

- For temporal variable t, the step of it is Δt; the grid points are t_n, n = 1, 2, 3, ..
- Denote $u(x_i, t_n)$ as u_i^n.
- Denote $\frac{\Delta t}{(\Delta x)^2}$ as f.

4 Dufort-Frankel Method

4.1 Difference Scheme

This method is a modification of Richardson method to provide stability. The exact formula is shown as below:

$$\frac{\partial u_i^n}{\partial t} = \frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t};$$

$$\frac{\partial^2 u_i^n}{\partial x^2} = \frac{u_{i+1}^n + u_{i-1}^n - u_i^{n+1} - u_i^{n-1}}{(\Delta x)^2}$$

Thus we get the difference scheme of this diffusion equation:

$$\frac{u_i^{n+1} - u_i^{n-1}}{\Delta t} = \frac{u_{i+1}^n + u_{i-1}^n - u_i^{n+1} - u_i^{n-1}}{(\Delta x)^2}$$

By substituting $\frac{\Delta t}{(\Delta x)^2}$ with f and taking into account boundary conditions and initial data, we get the final version of the explicit scheme:

$$\begin{cases} (1+f)u_i^{n+1} = f\left(u_{i+1}^n + u_{i-1}^n\right) + (1-f)u_i^{n-1}, i = 1,2,\dots, N-1 \\ u_i^0 = \sin(i\Delta x), i = 0,1,\dots, N \\ u_0^n = u_N^n = 0, n = 0,1,2,\dots \end{cases}$$

4.2 Numerical Results

4.2.1 Time step specification

The truncation error of this scheme is

$$\tau = \frac{(\Delta t)^2}{(\Delta x)^2} u_{tt} + O((\Delta t)^2 + (\Delta x)^2 + \frac{(\Delta t)^4}{(\Delta x)^2})$$

Let $k = c(\Delta x)^{1+\delta} (\delta > 0)$, then τ is of $O((\Delta x)^{2\delta})$.

Therefore, when $\Delta t = c(\Delta x)^{1.625}$, τ is order of 1.25;

when $\Delta t = c(\Delta x)^{1.75}$, τ is order of 1.5;

when $\Delta t = c(\Delta x)^{2}$, τ is order of 2.

4.2.2 Results

Set N = 500, which means x and t are both equally cut into 500 intervals.

Set c = 1.

A. Error of u(x, t)

order of 1.25 order of 1.5 order of 2

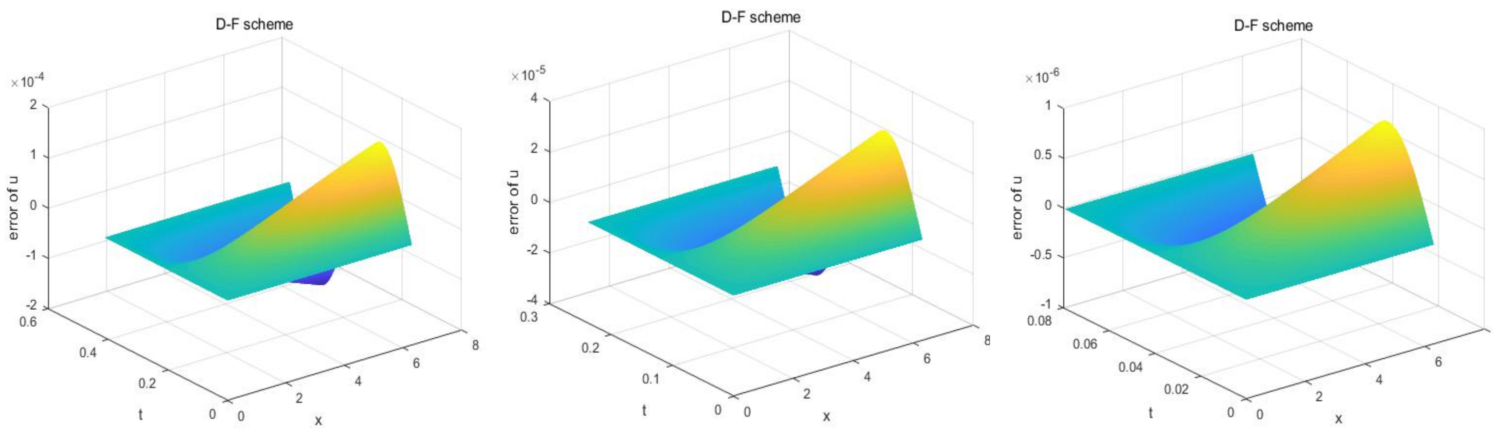
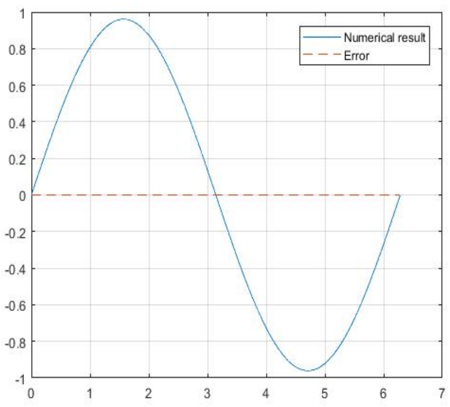


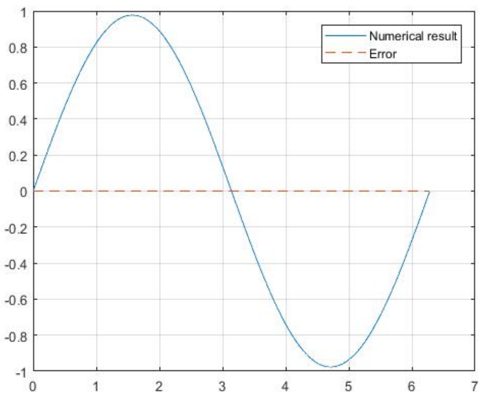
Fig 1: Error of D-F scheme

The error here means the difference of u between the exact solution and numerical results at the same value of both x and t. According to these figures above, we can see that there are still some minor errors of this difference scheme. When the order goes higher, the error seems to diminish at first glance. This might due to that we didn't settle the range of temporal variable t, so when order goes higher, time step Δt goes down, making the numerical results more accurate.

If we look at errors more closely, taking a cross-section of the 3D figures above, say, setting time t at the 100th node, we get:

order of 1.25 order of 1.5 order of 2





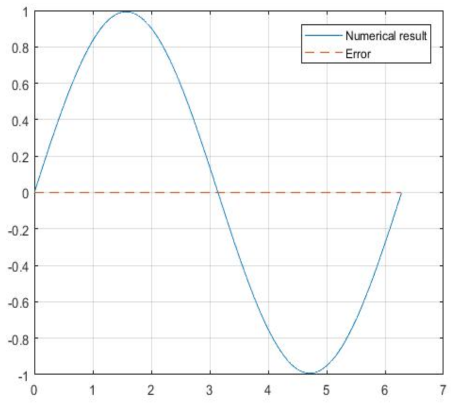
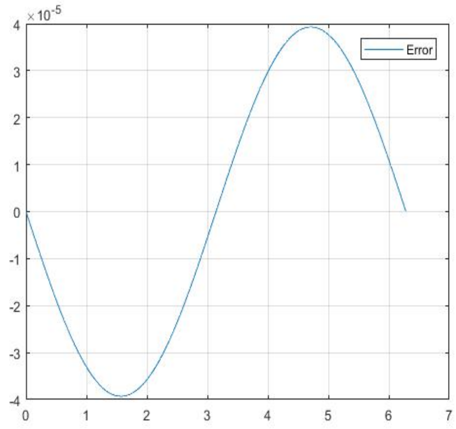
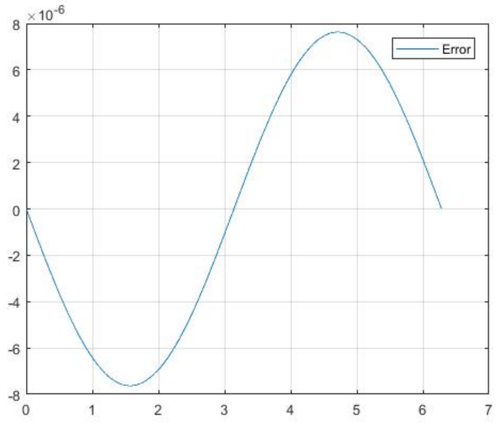


Fig 2: Error of D-F scheme when t is at the 100th node

By observing the dashed error line, we can hardly tell the difference of errors between the three orders, so to get a clearer idea of the errors, we only show the error line:

order of 1.25 order of 1.5 order of 2





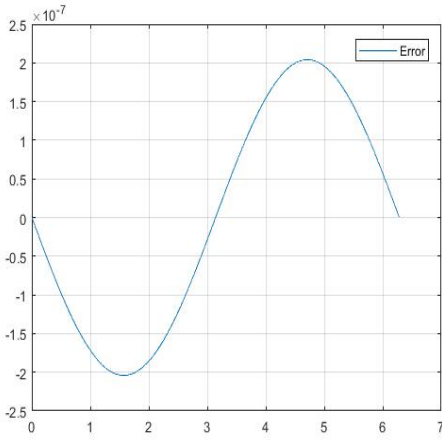


Fig 3: Error of D-F scheme when t is at the 100th node

When t is settled, u is actually a function of x, the above figures indicate that errors are slightly more obvious when u is at local extreme value. Meanwhile, the axis range indicates that when order goes higher, results seem to be more accurate, just as we have mentioned previously.

B. CPU running time

Order	1.25	1.5	2
Running time (seconds)	0.233	0.275	0.273

Table 1: CPU running time of the three orders

5 Crank-Nicholson Method

5.1 Difference Scheme

Instead of estimating u_i^n, this method estimate $u_i^{n+\frac{1}{2}}$:

$$\frac{\partial u_i^{n+\frac{1}{2}}}{\partial t} = \frac{u_i^{n+1} - u_i^n}{\Delta t}$$

$$\frac{\partial^2 u_i^{n+\frac{1}{2}}}{\partial x^2} = \frac{1}{2}\left[\frac{\partial^2 u_i^{n+1}}{\partial x^2} + \frac{\partial^2 u_i^n}{\partial x^2}\right] = \frac{1}{2}\left[\frac{u_{i+1}^{n+1} + u_{i-1}^{n+1} - 2u_i^{n+1}}{(\Delta x)^2} + \frac{u_{i+1}^n + u_{i-1}^n - 2u_i^n}{(\Delta x)^2}\right]$$

Thus we get the implicit difference scheme of this diffusion equation:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{u_{i+1}^{n+1} + u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^n + u_{i-1}^n - 2u_i^n}{4(\Delta x)^2}$$

By substituting $\frac{\Delta t}{(\Delta x)^2}$ with f, we get the simpler version of the implicit scheme:

$$-\frac{1}{2} f u_{i-1}^{n+1} + (2+f)u_i^{n+1} - \frac{1}{2} f u_{i+1}^{n+1} = \frac{1}{2} f (u_{i+1}^n + u_{i-1}^n) + (2-f)u_i^n$$

Rewrite the scheme in matrix form to show tridiagonal structure:

First let $\frac{1}{2} f (u_{i+1}^n + u_{i-1}^n) + (2-f)u_i^n = R_i^n$

Since boundary potentials u_0 (where x = 0)and u_N (where x = 2π) are specified, we get

$$\begin{bmatrix} 2+f & -\frac{1}{2}f & 0 & 0 & \cdots & 0 \\ -\frac{1}{2}f & 2+f & -\frac{1}{2}f & 0 & \ddots & \vdots \\ 0 & -\frac{1}{2}f & 2+f & \ddots & \ddots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 2+f & -\frac{1}{2}f \\ 0 & 0 & 0 & 0 & -\frac{1}{2}f & 2+f \end{bmatrix} \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ \vdots \\ \vdots \\ u_{N-1}^{n+1} \end{bmatrix} = \begin{bmatrix} R_1^n + \frac{1}{2}fu_0^n \\ R_2^n \\ R_3^n \\ \vdots \\ R_{N-2}^n \\ R_{N-1}^n + \frac{1}{2}fu_N^n \end{bmatrix}$$

The coefficient matrix above is tridiagonal, so we can use chasing method to solve the 'Ax=b'.

5.2 Numerical Results

5.2.1 Time step specification

Crank-Nicholson Method is unconditionally stable, so it doesn't matter a lot how we choose the time step. In order to provide a relatively fair comparison with Dufort-Frankel Method, specify time step $\Delta t = (\Delta x)^2$, which is the same as when truncation order is 2 in Dufort-Frankel Method.

5.2.2 Results

A. Error of u(x, t)

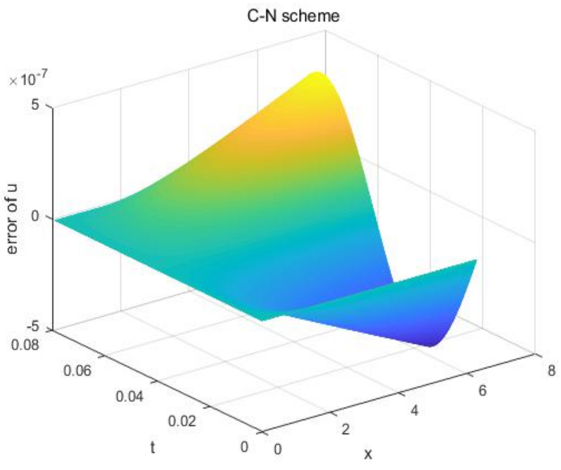


Fig 4: Error of C-N scheme

If we look at errors more closely, taking a cross-section of the 3D figures above, say, setting time t at the 100th node, just as in the Dufort-Frankel Method, we get:

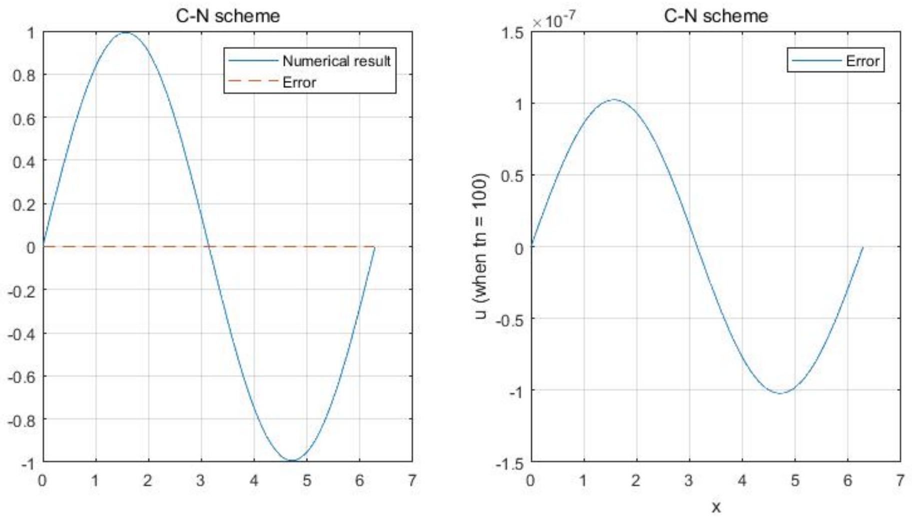


Fig 5: Error of C-N scheme when t is at the 100th node

According to the left-sided figure above, the error of this method can barely be recognized by naked eyes, so we provide the error plot alone, which is the right-sided figure above, to show more details. In order to provide more thorough idea about the accuracy of the D-F and C-N method, we calculate the mean and standard deviation of their absolute errors when t is at the 100th node:

	mean	standard deviation
D-F	1.2970e-07	6.3095e-08
C-N	6.4842e-08	3.1545e-08

Table 2: Comparison of absolute errors of D-F scheme and C-N scheme

The table above shows that C-N scheme is slightly more accurate than D-F scheme since both its mean the standard deviation are smaller than that of D-F.

B. CPU running time

The CPU running time of this scheme is 2.337 seconds, this might due to that this scheme requires to use chasing method to solve 'Ax=b' at each level of time thus increasing the overall running time.

6 Other high-order scheme

In this section, we study and present the results of another high-order scheme, which is the Forward Time Centered Space Scheme.

6.1 Difference Scheme

$$\frac{\partial u_i^n}{\partial t} = \frac{u_i^{n+1} - u_i^n}{\Delta t};$$

$$\frac{\partial^2 u_i^n}{\partial x^2} = \frac{u_{i+1}^n + u_{i-1}^n - 2u_i^n}{(\Delta x)^2}$$

By substituting $\frac{\Delta t}{(\Delta x)^2}$ with f and taking into account boundary conditions and initial data, we get the final version of the explicit scheme:

$$\begin{cases} u_i^{n+1} = u_i^n + \frac{1}{2} f(u_{i+1}^n + u_{i-1}^n - 2u_i^n), i = 1,2,...,N \\ u_i^0 = \sin(i\Delta x), i = 0,1,2,...,N \\ u_0^n = u_N^n, n = 1,2,... \end{cases}$$

6.2 Numerical Results

6.2.1 Time step specification

This scheme is conditionally stable and the condition is:

$$\frac{1}{2}\frac{\Delta t}{(\Delta x)^2} \le \frac{1}{2}$$

So f needs to be less than or equal to 1. In order to provide a relatively fair comparison with the other two schemes, we specify f = 1, so $\Delta t = (\Delta x)^2$, which is same as in the previous C-N scheme.

6.2.2 Results

A. Error of u(x, t)

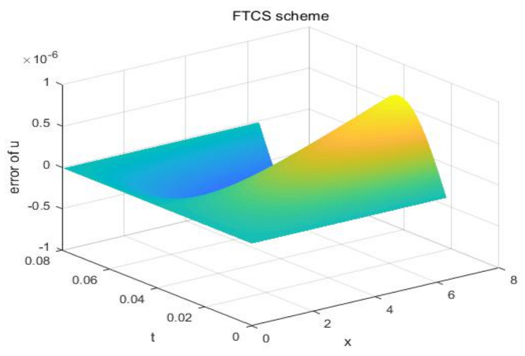


Fig 6: Error of FTCS scheme

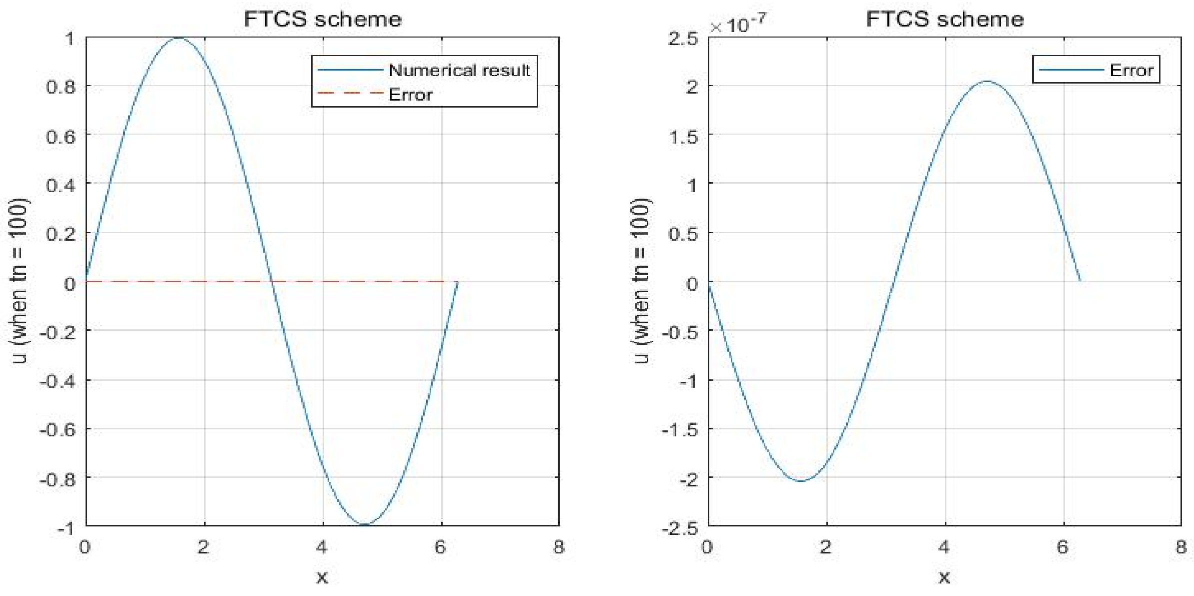


Fig 7: Error of FTCS scheme when t is at the 100th node

Above figures show that the error of this scheme bears the same characteristic as previous two schemes. The mean and standard deviation of its absolute errors are shown below:

	mean	standard deviation
D-F	1.2970e-07	6.3095e-08
C-N	6.4842e-08	3.1545e-08
FTCS	1.2070e-07	6.3148e-08

Table 2: Comparison of absolute errors of D-F scheme, C-N scheme and FTCS scheme

The table shows that FTCS scheme is almost at the same level of accuracy as D-F scheme.

B. CPU running time

The CPU running time of this scheme is 0.307 seconds -- slightly longer than D-F scheme, much shorter than C-N scheme.

7 Summary

This project studied properties of numerical methods for one specific diffusion equation and made comparison among these difference schemes with regards to their CPU running time and accuracy.

D-F and C-N are both implicit, but they treat spatial variable x differently, thus yielding the difference of their results. FTCS is explicit and conditionally stable.

Among these three schemes, CPU running time of D-F and FTCS are almost at the same level, while that of C-N is longer; Accuracy of D-F and FTCS also look alike, a bit lower than that of C-N.