Assignment 3

Zeyuan Wang

February 14, 2024

1 Problems

1.1

First I choose the manufactured solution $u(x) = cos(\frac{\pi}{2}x)$ which satisfied the Dirichlet BC $u(-1) = u(1) = 0$. So $f(x) = -\frac{\partial^2 u}{\partial x^2} = \frac{\pi^2}{4}cos(\frac{\pi}{2}x)$. The idea of coefficient-based Chebyshev spectral method is to approximate solution u by linear combination of different orders Chebyshev polynomials:

$$u(x) = \sum_{k=0}^{n} c_k T_k(x),\ x \in [-1, 1] \tag{1}$$

where T_k is the kth order Chebyshev polynomial of the first kind, $T_k(x) = cos(kcos^{-1}(x))$. We also want the boundary condition and original PDE to be exactly satisfied at n sample points, for example, the Chebyshev extrema $x_0, x_1, \dots, x_n \in [-1, 1]$. Then we get N equations:

$$\begin{aligned}
u(x_0) = \sum_{k=0}^{n} c_k T_k(x_0) &= 0 \\
\left.\frac{\partial^2 u}{\partial x^2}\right|_{x=x_1} = \sum_{k=0}^{n} c_k \left.\frac{\partial^2 T_k}{\partial x^2}\right|_{x=x_1} &= -f(x_1) \\
\left.\frac{\partial^2 u}{\partial x^2}\right|_{x=x_2} = \sum_{k=0}^{n} c_k \left.\frac{\partial^2 T_k}{\partial x^2}\right|_{x=x_2} &= -f(x_2) \\
\dots \\
\left.\frac{\partial^2 u}{\partial x^2}\right|_{x=x_{n-1}} = \sum_{k=0}^{n} c_k \left.\frac{\partial^2 T_k}{\partial x^2}\right|_{x=x_{n-1}} &= -f(x_{n-1}) \\
u(x_n) = \sum_{k=0}^{n} c_k T_k(x_n) &= 0
\end{aligned} \tag{2}$$

Here the unknowns are the coefficients $c_1, c_2, \cdots, c_n$. The equivalent matrix representation of the system can be constructed as follow:

$$\begin{bmatrix} T_0(x_0) & T_1(x_0) & \cdots & T_{n-1}(x_0) & T_n(x_0) \\ \frac{\partial^2 T_0}{\partial x^2}(x_1) & \frac{\partial^2 T_1}{\partial x^2}(x_1) & \cdots & \frac{\partial^2 T_{n-1}}{\partial x^2}(x_1) & \frac{\partial^2 T_n}{\partial x^2}(x_1) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial^2 T_0}{\partial x^2}(x_{n-1}) & \frac{\partial^2 T_1}{\partial x^2}(x_{n-1}) & \cdots & \frac{\partial^2 T_{n-1}}{\partial x^2}(x_{n-1}) & \frac{\partial^2 T_n}{\partial x^2}(x_{n-1}) \\ T_0(x_n) & T_1(x_n) & \cdots & T_{n-1}(x_n) & T_n(x_n) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} = \begin{bmatrix} 0 \\ -f(x_1) \\ \vdots \\ -f(x_{n-1}) \\ 0 \end{bmatrix} \tag{3}$$

As mentioned $T_k(x) = cos(kcos^{-1}(x))$. The Chebyshev polynomials of the first kind are obtained from the recurrence relation:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_2(x) &= 2x^2 - 1 \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x) \end{aligned} \tag{4}$$

The benefit for the recurrence formula is we can easily get the derivatives:

$$\begin{aligned} T_0'(x) &= 0 \\ T_1'(x) &= 1 \\ T_2'(x) &= 4x \\ T_{n+1}'(x) &= 2xT_n'(x) + 2T_n(x) - T_{n-1}'(x) \end{aligned} \tag{5}$$

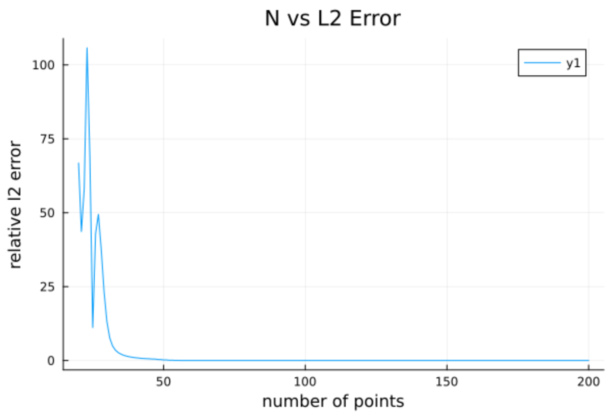
$$\begin{aligned} T_0''(x) &= 0 \\ T_1''(x) &= 0 \\ T_2''(x) &= 4 \\ T_{n+1}''(x) &= 2xT_n''(x) + 4T_n'(x) - T_{n-1}''(x) \end{aligned} \tag{6}$$

I found in practice the recurrence form is not very efficient to evaluate especially when a large n is chosen. So instead I use the "explicit" kind of Chebyshev polynomials in code

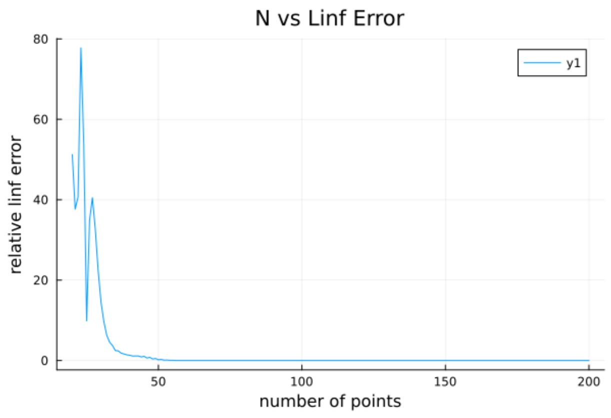
$$T_n(x) = \cos(n\cos^{-1}(x)) \tag{7}$$

$$T_n''(x) = \frac{-n^2\cos(n\cos^{-1}(x))}{1-x^2} + \frac{nx\sin(n\cos^{-1}(x))}{(1-x^2)\sqrt{1-x^2}} \tag{8}$$

The manufactured solution I choose is $u(x) = sin(\pi x) + sin(2\pi x) + sin(4\pi x) + sin(8\pi x) + sin(16\pi x)$. I choose the number of Chebyshev extrema points from 20 to 200 and 1000 equal spaced test points ranging in $[-1, 1]$ to measure the decrease rate for relative L_2 and L_∞ error.



(a) relative L_2 error



(b) relative L_∞ error

Figure 1: N vs error

As the plot shown, both L_2 and L_∞ error decrease exponentially fast.

1.2

Replace u with $sin(\pi x)cos(\pi y)e^{-\pi t}$ we get

$$-\pi sin(\pi x)cos(\pi y)e^{-\pi t} = \nu(-\pi^2 sin(\pi x)cos(\pi y)e^{-\pi t} - \pi^2 sin(\pi x)cos(\pi y)e^{-\pi t}) + f(x,t) \quad (9)$$

$$f(x,t) = (2\pi^2\nu - \pi)sin(\pi x)cos(\pi y)e^{-\pi t} \quad (10)$$

Use solution $u(x,y,t)$ to impose Dirichlet Boundary condition:

$$\begin{aligned} u(0,y,t) &= 0 \\ u(1,y,t) &= 0 \\ u(x,0,t) &= sin(\pi x)e^{-\pi t} \\ u(x,1,t) &= sin(\pi x)e^{-\pi t} \end{aligned} \quad (11)$$

Use Crank-Nicolson scheme to discretize the PDE in time we get

$$u^{n+1} - u^n = \frac{\Delta t}{2}[\nu\Delta u^{n+1} + f^{n+1} + \nu\Delta u^n + f^n] \quad (12)$$

Use second-order central finite difference to discretize the space, we get

$$\begin{aligned} u_{i,j}^{n+1} = u_{i,j}^n + \frac{\Delta t}{2}[&\frac{\nu}{h^2}(-4u_{i,j}^{n+1} + u_{i-1,j}^{n+1} + u_{i+1,j}^{n+1} + u_{i,j-1}^{n+1} + u_{i,j+1}^{n+1}) + f_{i,j}^{n+1} \\ &+ \frac{\nu}{h^2}(-4u_{i,j}^n + u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n) + f_{i,j}^n], \quad u_{i,j} \in \Omega/\partial\Omega \end{aligned} \quad (13)$$

$$\begin{aligned} u_{0,j}^{n+1} &= 0 \\ u_{n,j}^{n+1} &= 0 \\ u_{i,0}^{n+1} &= sin(\pi ih)e^{-\pi(n+1)\Delta t} \\ u_{i,n}^{n+1} &= sin(\pi ih)e^{-\pi(n+1)\Delta t} \end{aligned} \quad (14)$$

$$\begin{aligned}(1+\frac{2\nu\Delta t}{h^2})u_{i,j}^{n+1} - \frac{\nu\Delta t}{2h^2}[u_{i-1,j}^{n+1}+u_{i+1,j}^{n+1}+u_{i,j-1}^{n+1}+u_{i,j+1}^{n+1}] \\ = (1-\frac{2\nu\Delta t}{h^2})u_{i,j}^{n} + \frac{\nu\Delta t}{2h^2}[u_{i-1,j}^{n}+u_{i+1,j}^{n}+u_{i,j-1}^{n}+u_{i,j+1}^{n}] \\ + \frac{\Delta t}{2}(f_{i,j}^{n+1}+f_{i,j}^{n})\end{aligned} \tag{15}$$

Let's denote $1+\frac{2\nu\Delta t}{h^2}$ as α_1, $1-\frac{2\nu\Delta t}{h^2}$ as α_2, $\frac{\nu\Delta t}{2h^2}$ as β. Then a linear system can be constructed by forming all the u^{n+1} as a unknown vector $\mathbf{u^{n+1}}$, all the history u^n and the forcing term as the right hand side $\mathbf{b}$, and forming all the coefficient α_1 and β to a matrix L. Note that since the Dirichlet boundary condition is used, and we assume that the value of u at the boundary can be evaluated at any given time, then u_b^{n+1} do not need to be included in the unknown vector. Rather, it should be included in the right hand side as parts of the known information. Bear this in mind, the size of our matrix L is $N_L \times N_L$, where $N_L = (N_{grid}-2)(N_{grid}-2)$, and N_{grid} is the length of the grid edge.(Here the length of grid edge is defined as the number of points, instead of the number of intervals).

$$\mathrm{L} = \begin{bmatrix} \alpha_1 & -\beta & 0 & \cdots & 0 & -\beta & 0 & \cdots & 0 \\ -\beta & \alpha_1 & -\beta & \cdots & 0 & 0 & -\beta & \cdots & 0 \\ 0 & -\beta & \alpha_1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_1 & -\beta & 0 & \cdots & 0 \\ -\beta & 0 & 0 & \cdots & -\beta & \alpha_1 & -\beta & \cdots & 0 \\ 0 & -\beta & 0 & \cdots & 0 & -\beta & \alpha_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & \alpha_1 \end{bmatrix} \tag{16}$$

$$\mathrm{u^{n+1}} = \begin{bmatrix} u_{1,1}^{n+1} \\ u_{1,2}^{n+1} \\ \vdots \\ u_{1,grid\_size-1}^{n+1} \\ u_{2,1}^{n+1} \\ u_{2,2}^{n+1} \\ \vdots \\ u_{2,grid\_size-1}^{n+1} \\ \vdots \\ u_{grid\_size-1,1}^{n+1} \\ u_{grid\_size-1,2}^{n+1} \\ \vdots \\ u_{grid\_size-1,grid\_size-1}^{n+1} \end{bmatrix} \tag{17}$$

$$\mathrm{u}_1^{\mathrm{n}} = \begin{bmatrix} u^n_{1,1} \\ u^n_{1,2} \\ \vdots \\ u^n_{1,grid\_size-1} \\ u^n_{2,1} \\ u^n_{2,2} \\ \vdots \\ u^n_{2,grid\_size-1} \\ \vdots \\ u^n_{grid\_size-1,1} \\ u^n_{grid\_size-1,2} \\ \vdots \\ u^n_{grid\_size-1,grid\_size-1} \end{bmatrix} \tag{18}$$

$$\mathrm{u}_2^{\mathrm{n}} = \begin{bmatrix} u^n_{0,1} + u^n_{2,1} + u^n_{1,0} + u^n_{1,2} \\ u^n_{0,2} + u^n_{2,2} + u^n_{1,1} + u^n_{1,3} \\ \vdots \\ u^n_{0,grid\_size-1} + u^n_{2,grid\_size-1} + u^n_{1,grid\_size-2} + u^n_{1,grid\_size} \\ u^n_{1,1} + u^n_{2,1} + u^n_{2,0} + u^n_{2,2} \\ u^n_{1,2} + u^n_{3,2} + u^n_{2,1} + u^n_{2,3} \\ \vdots \\ u^n_{1,grid\_size-1} + u^n_{3,grid\_size-1} + u^n_{2,grid\_size-2} + u^n_{2,grid\_size} \\ \vdots \\ u^n_{grid\_size-2,1} + u^n_{grid\_size,1} + u^n_{grid\_size-1,0} + u^n_{grid\_size-1,2} \\ u^n_{grid\_size-2,2} + u^n_{grid\_size,2} + u^n_{grid\_size-1,1} + u^n_{grid\_size-1,3} \\ \vdots \\ u^n_{grid\_size-2,grid\_size-1} + u^n_{grid\_size,grid\_size-1} + u^n_{grid\_size-1,grid\_size-2} + u^n_{grid\_size-1,grid\_size} \end{bmatrix} \tag{19}$$

$$\mathrm{f} = \begin{bmatrix} f^n_{1,1} + f^{n+1}_{1,1} \\ f^n_{1,2} + f^{n+1}_{1,2} \\ \vdots \\ f^n_{1,grid\_size-1} + f^{n+1}_{1,grid\_size-1} \\ f^n_{2,1} + f^{n+1}_{2,1} \\ f^n_{2,2} + f^{n+1}_{2,2} \\ \vdots \\ f^n_{2,grid\_size-1} + f^{n+1}_{2,grid\_size-1} \\ \vdots \\ f^n_{grid\_size-1,1} + f^{n+1}_{grid\_size-1,1} \\ f^n_{grid\_size-1,2} + f^{n+1}_{grid\_size-1,2} \\ \vdots \\ f^n_{grid\_size-1,grid\_size-1} + f^{n+1}_{grid\_size-1,grid\_size-1} \end{bmatrix} \tag{20}$$

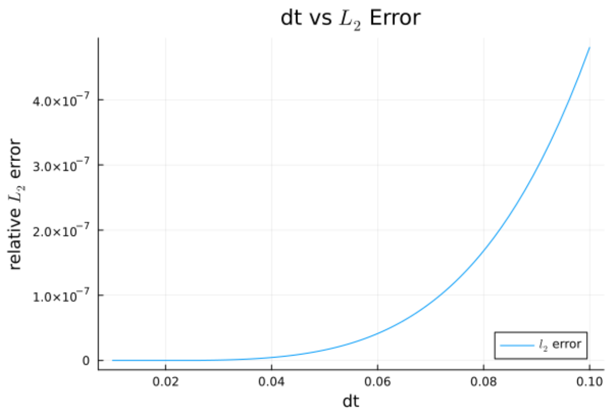
$$\mathrm{u_b^{n+1}} = \begin{bmatrix} u_{0,1}^{n+1} + u_{1,0}^{n+1} \\ u_{0,2}^{n+1} \\ \vdots \\ u_{0,grid\_size-1}^{n+1} + u_{1,grid\_size}^{n+1} \\ u_{2,0}^{n+1} \\ 0 \\ \vdots \\ u_{2,grid\_size}^{n+1} \\ \vdots \\ u_{grid\_size,1}^{n+1} + u_{grid\_size-1,0}^{n+1} \\ u_{grid\_size,2}^{n+1} \\ \vdots \\ u_{grid\_size,grid\_size-1}^{n+1} + u_{grid\_size-1,grid\_size}^{n+1} \end{bmatrix} \tag{21}$$

$$\mathrm{b} = \alpha_2 \mathrm{u_1^n} + \beta \mathrm{u_2^n} + \beta \mathrm{u_b^{n+1}} + \frac{\Delta t}{2}\mathrm{f} \tag{22}$$

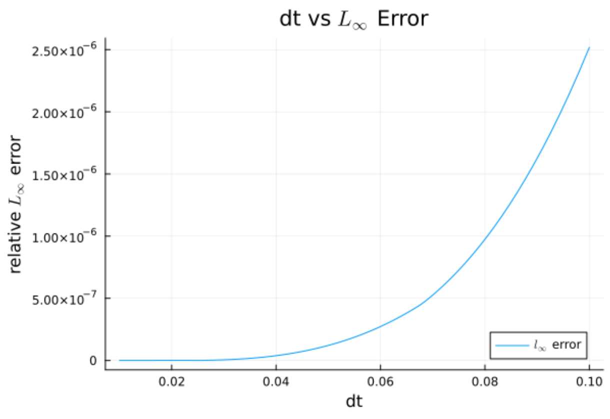
Note that while b need to be reconstructed at every time-step, L only need to constructed once. To solve the system $\mathrm{Lu^{n+1}} = \mathrm{b}$, note that L is positive definite and really well conditioned, Conjugate Gradient can be a good choice. I use the conjugate gradient method in Julia's IterativeSolvers package, with Incomplete Cholesky Factorization as the pre-conditioner.

The accuracy respect to dt and h has been tested respectively. I slightly change the manufactured solution to $sin(\pi x)cos(\pi y)e^{-\pi t} + 1$ to make the relative error computation more robust.(Avoid divide by 0)

The accuracy to time is studied for dt range from 0.01 to 0.1, while h is set as 0.01 and the error is measured at $T = 5.0$

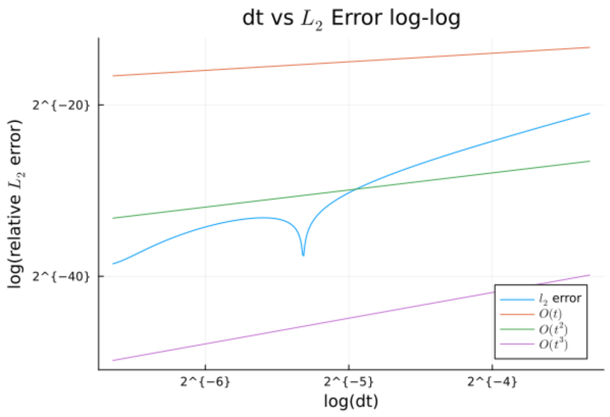


(a) relative L_2 error

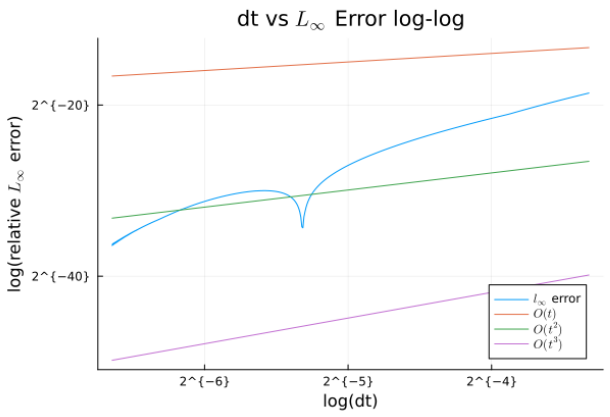


(b) relative L_∞ error

Figure 2: dt vs error



(a) relative L_2 error $\log - \log$

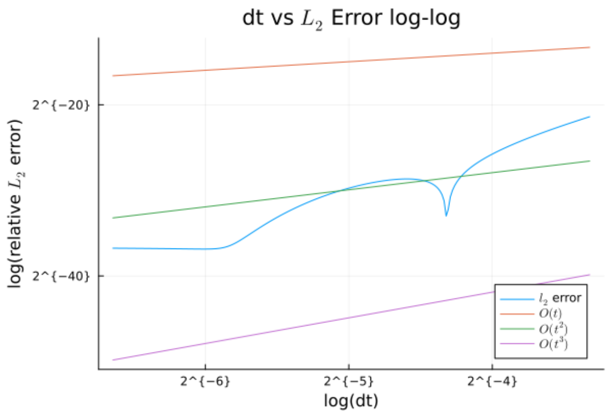


(b) relative L_∞ error $\log - \log$

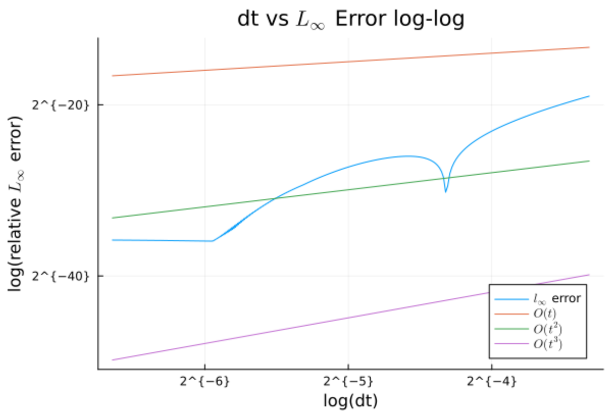
Figure 3: dt vs error $\log - \log$, h = 0.01

It seems that when $dt < 0.025$, the error starts to be dominated by other factor like the machine precision and the size of grid. This can be evidenced by looking the result for $h = 0.02$. The singular point shifts to right since the error introduced by space become prominent earlier.

Inside the asymptotic region, It seems that both type of L error decrease as third order, which is better than the theoretic second order convergence speed.



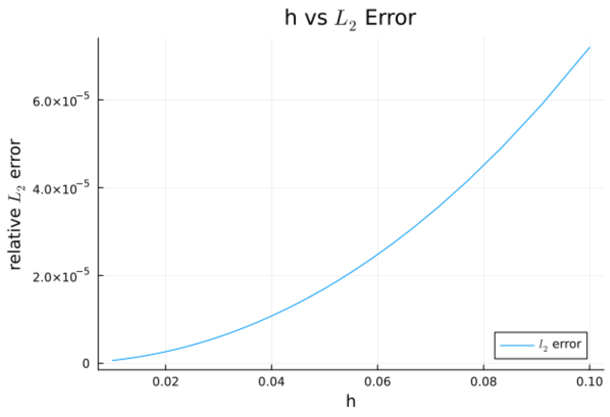
(a) relative L_2 error $\log - \log$



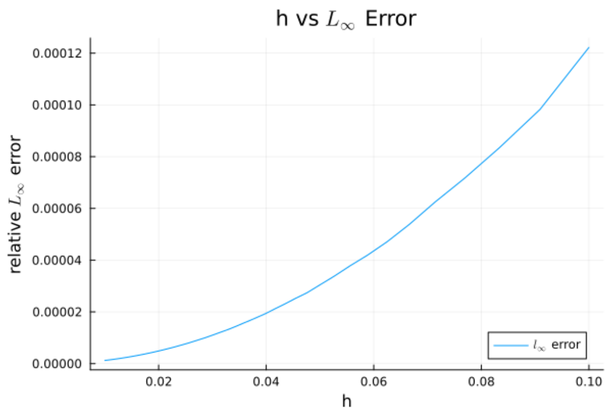
(b) relative L_∞ error $\log - \log$

Figure 4: dt vs error $\log - \log$, h $= 0.02$

The accuracy to space is studied for h range from 0.1 to 0.01, while dt is set as 0.01 and the final time is 1.0



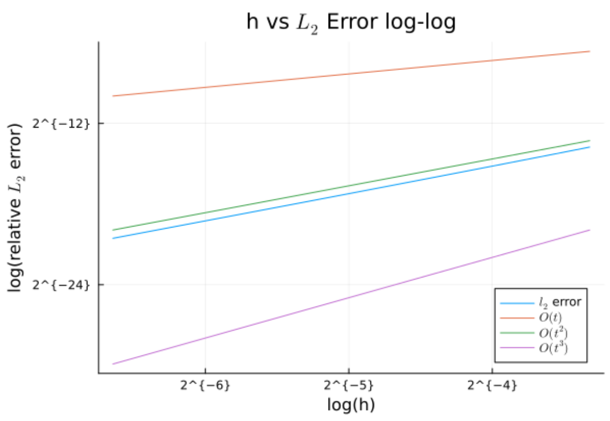
(a) relative L_2 error



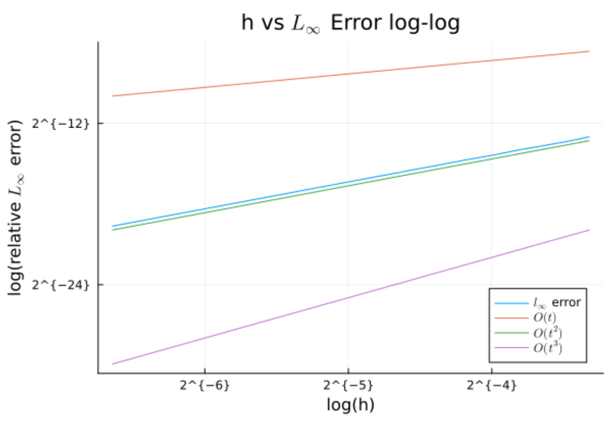
(b) relative L_∞ error

Figure 5: h vs error

A $\mathbf{log} - \mathbf{log}$ scale plot is also provided to show that the method obtains second order accuracy in space for both L_2 and L_∞ error.



(a) relative L_2 error $\log - \log$



(b) relative L_∞ error $\log - \log$

Figure 6: h vs error $\log - \log$

1.3

We would expect a stable result should gradually decay to 0 instead of blowing up or oscillating. A stronger statement to ensure the stability is for any given $t > 0$, $|u(t + \Delta t)| \leq |u(t)|$ always holds. Or equivalently, there exists a growing factor G, where $|G| \leq 1$, let $|u(t + \Delta t)| = G|u(t)|$, .

A powerful tool to study the stability of PDE is called Von Neumann stability analysis, also known as Fourier stability analysis. As its name implies, the idea behind the Fourier stability analysis is to replace the discrete solution with its Fourier mode in space.

$$u = \sum \sum e^{ikx} e^{ily} \tag{23}$$

Here, i is the imaginary unit.

$$u_{p,q} = e^{ikph} e^{ilqh} \tag{24}$$

Similarity we have

$$
\begin{aligned}
u_{p-1,q} &= e^{ik(p-1)h}e^{ilqh} \\
u_{p+1,q} &= e^{ik(p+1)h}e^{ilqh} \\
u_{p,q-1} &= e^{ikph}e^{il(q-1)h} \\
u_{p,q+1} &= e^{ikph}e^{il(q+1)h}
\end{aligned} \tag{25}
$$

$$
\begin{aligned}
u_{p,q}^{n+1} = u_{p,q}^{n} + \frac{\nu\Delta t}{2h^2}(&-4u_{p,q}^{n+1} + u_{p-1,q}^{n+1} + u_{p+1,q}^{n+1} + u_{p,q-1}^{n+1} + u_{p,q+1}^{n+1} \\
&- 4u_{p,q}^{n} + u_{p-1,q}^{n} + u_{p+1,q}^{n} + u_{p,q-1}^{n} + u_{p,q+1}^{n})
\end{aligned} \tag{26}
$$

Remember we still have the assumption that $|u^{n+1}| = G|u^n|$, so we can rewrite the Eq.(18) as

$$
\begin{aligned}
Gu_{p,q} = u_{p,q} + \frac{\nu\Delta t}{2h^2}(&-4Gu_{p,q} + Gu_{p-1,q} + Gu_{p+1,q} + Gu_{p,q-1} + Gu_{p,q+1} \\
&- 4u_{p,q} + u_{p-1,q} + u_{p+1,q} + u_{p,q-1} + u_{p,q+1})
\end{aligned} \tag{27}
$$

Replace $u_{p,q}$ by its Fourier mode

$$
\begin{aligned}
Ge^{ikph}e^{ilqh} = {} & e^{ikph}e^{ilqh} \\
& + \frac{\nu\Delta t}{2h^2}(-4Ge^{ikph}e^{ilqh} + Ge^{ik(p-1)h}e^{ilqh} + Ge^{ik(p+1)h}e^{ilqh} + Ge^{ikph}e^{il(q-1)h} + Ge^{ikph}e^{il(q+1)h} \\
& - 4e^{ikph}e^{ilqh} + e^{ik(p-1)h}e^{ilqh} + e^{ik(p+1)h}e^{ilqh} + e^{ikph}e^{il(q-1)h} + e^{ikph}e^{il(q+1)h})
\end{aligned} \tag{28}
$$

Divide both side by $e^{ikp}e^{ilq}$, we get

$$\begin{aligned} G = 1 + \frac{\nu\Delta t}{2h^2}(&-4G + Ge^{-ikh} + Ge^{ikh} + Ge^{-ilh} + Ge^{ilh} \\ &- 4 + e^{-ikh} + e^{ikh} + e^{-ilh} + e^{ilh}) \end{aligned} \tag{29}$$

$$G[1 + \frac{\nu\Delta t}{2h^2}(4 - e^{-ikh} - e^{ikh} - e^{-ilh} - e^{ilh})] = 1 - \frac{\nu\Delta t}{2h^2}(4 - e^{-ikh} - e^{ikh} - e^{-ilh} - e^{ilh}) \tag{30}$$

From the Euler formula, we have

$$e^{-i\omega} + e^{i\omega} = 2cos(\omega) \tag{31}$$

Denote α as $\frac{\nu\Delta t}{2h^2}$,

$$\begin{aligned} \alpha(4 - e^{-ikh} - e^{ikh} - e^{-ilh} - e^{ilh}) &= \alpha[4 - 2(cos(kh) + cos(lh))] \\ &= 2\alpha[2 - (cos(kh) + cos(lh)] \end{aligned} \tag{32}$$

Denote $2\alpha[2 - (cos(kh) + cos(lh)]$ as β. Despite that the zeros of β do exist, for example when $kh = lh = n\pi$, for most of the points in space β always strictly greater than zero. Divide both side by β, we get

$$G = \frac{1-\beta}{1+\beta}, \quad \beta > 0 \tag{33}$$

$$\begin{aligned} G^2 &= \frac{(1-\beta)^2}{(1+\beta)^2} \\ &= \frac{(1+\beta)^2 - 4\beta}{(1+\beta)^2} \\ &= 1 - 4\frac{\beta}{(1+\beta)^2} \\ &< 1 \end{aligned} \tag{34}$$

$0 <= G^2 < 1 \rightarrow |G| < 1$. Therefore we can conclude that Crank-Nicolson on 2D heat equation is unconditionally stable, when kh and lh is not equal to $n\pi$.