Finite Difference Method for Dirichlet Problem

Tianyu Sun

October 26, 2021

1 Background and scheme

The Dirichlet problem satisfy

$$-\Delta u + \lambda u = f \quad u \in (0,1) \times (0,1) \tag{1}$$

with $u = 0$ on the boundary. The scheme I used is a kind of central space scheme. According to the problem, we have

$$-\frac{\partial^2}{\partial x^2}u - \frac{\partial^2}{\partial y^2}u + \lambda u = f \tag{2}$$

Let $u_{i,j}, f_{i,j}$ be u, f at i,j-th position. Partition x, y in $(0,1) \times (0,1)$. Let $\frac{\partial^2}{\partial x^2}u = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{\delta x^2}$ where δx is the mesh size of x. And similarly for y. Then we get

$$-\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{\delta x^2} - \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{\delta y^2} + \lambda u_{i,j} = f_{i,j} \tag{3}$$

Rearrange the terms, we have

$$u_{i,j} = \frac{f_{i,j}(\delta x)^2(\delta y)^2 + (u_{i+1,j} + u_{i-1,j})(\delta y)^2 + (u_{i,j+1} + u_{i,j-1})(\delta x)^2}{2(\delta y)^2 + 2(\delta x)^2 + \lambda(\delta x)^2(\delta y)^2} \tag{4}$$

2 Numerical result

2.1 Example 1

Let $\lambda = 1$ and let $u = \sin(2\pi x)\sin(2\pi y)$. Then we can see $u = 0$ when $x, y = 0, 1$. So it satisfy the boundary condition. Then

$$\begin{aligned}
\Delta u &= \frac{\partial^2}{\partial x^2}u + \frac{\partial^2}{\partial y^2}u \\
&= -4\pi^2 \sin(2\pi x)\sin(2\pi y) - 4\pi^2 \sin(2\pi x)\sin(2\pi y) \\
&= -8\pi^2 \sin(2\pi x)\sin(2\pi y) \\
f = -\Delta u + u &= (8\pi^2 + 1)\sin(2\pi x)\sin(2\pi y)
\end{aligned} \tag{5}$$

The result is shown below with mesh size 1/60 and 10000 iterations.

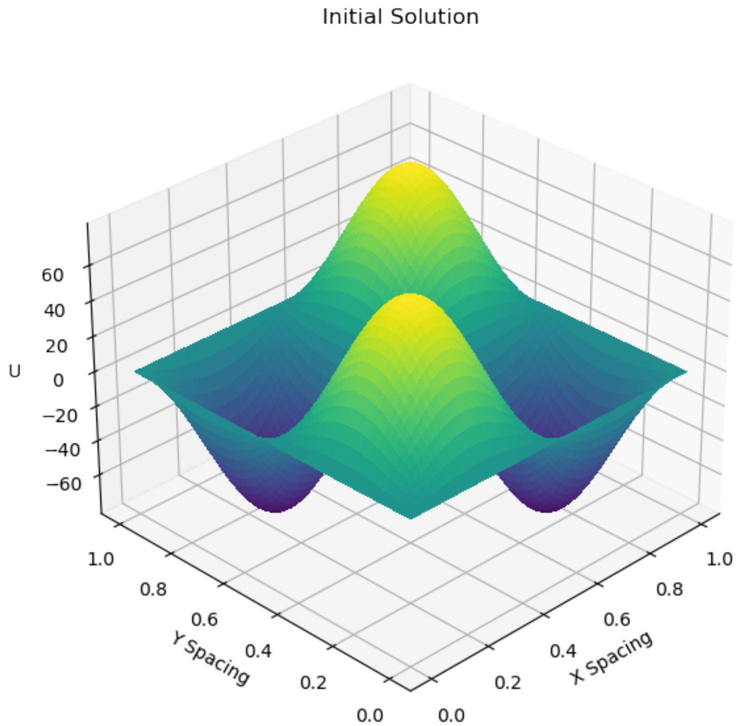

Figure 1: Initial Solution with mesh size 1/60 and 10^4 iterations

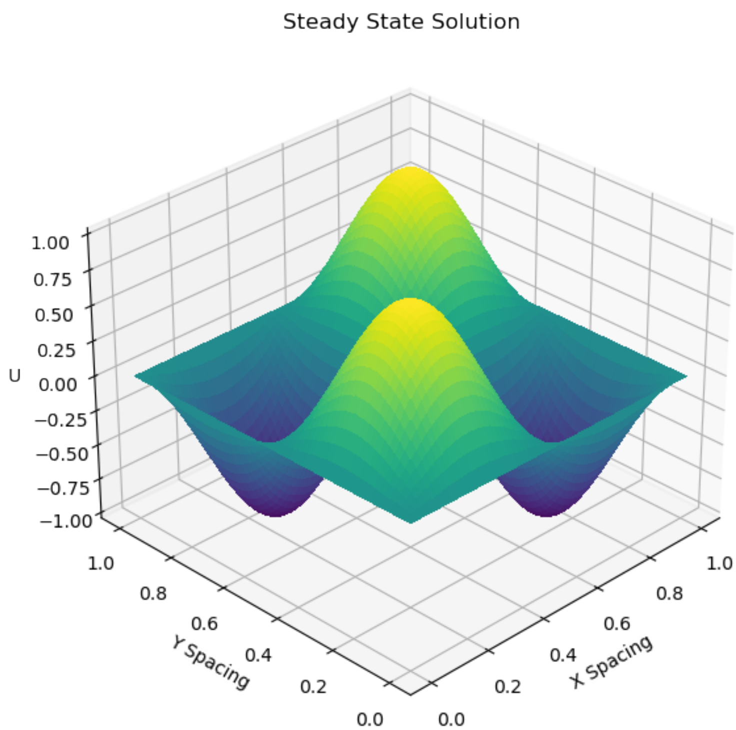

Figure 2: Steady State Solution with mesh size 1/60 and 10^4 iterations

The L^2 error with 10000 iterations and various number of mesh points is shown below.

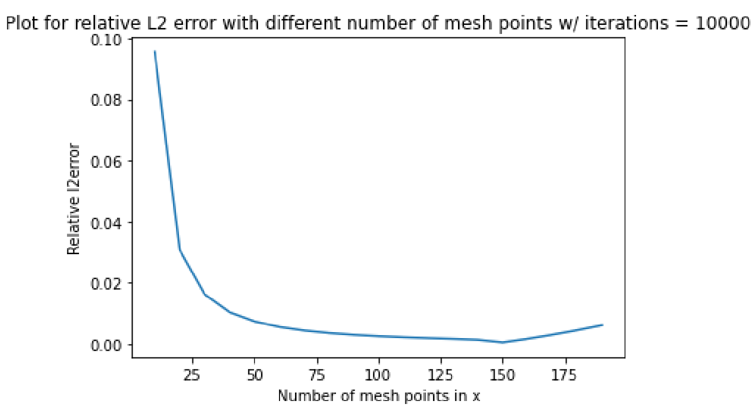

Figure 3: L^2 error vs number of mesh points at 10^4 iterations

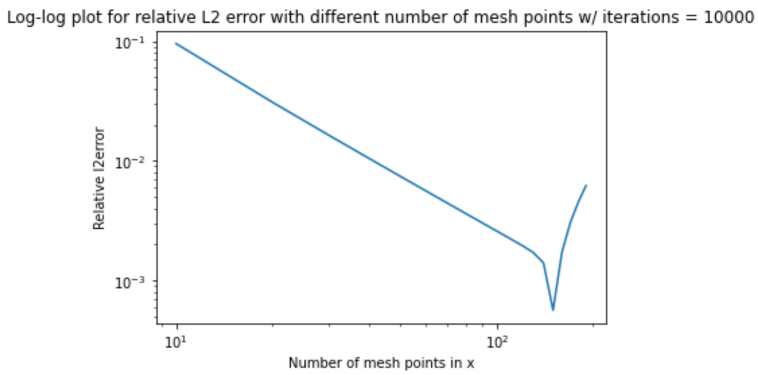

Figure 4: Log-log plot for L^2 error vs number of mesh points at 10^4 iterations

Interestingly, the scheme doesn't always converge as mesh points increases. From the graph, the error is lowest when there are 150 mesh points. Also, when I change the number of iterations, the mesh size corresponding to the lowest L^2 error also changes. Here is a graph or L^2 error vs iterations with mesh size 1/150.

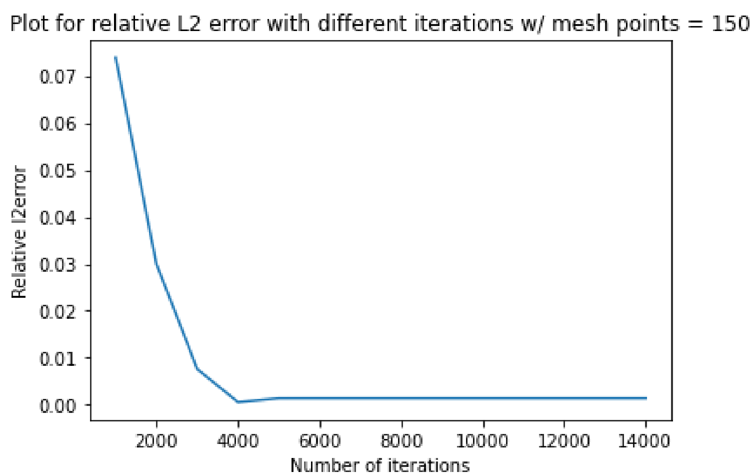

Figure 5: L^2 error vs number of iterations at mesh size 1/150

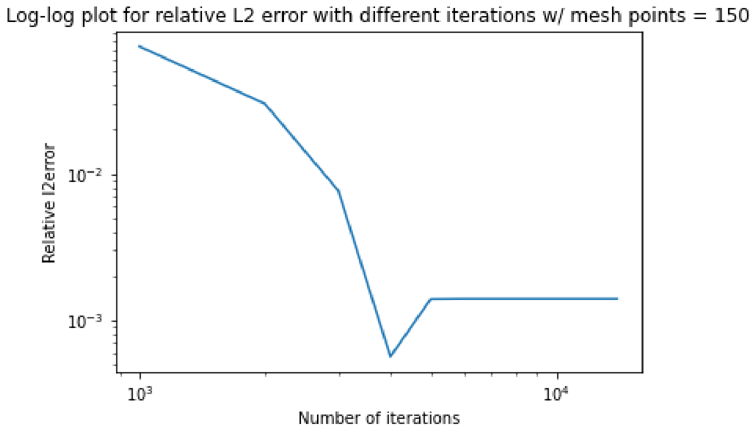

Figure 6: Loglog plot for L^2 error vs number of iterations at mesh size 1/150

The L^2 error is actually at lowest around 5000 iterations and stay the same afterwards.

2.2 Example 2

Let $\lambda = 1$ and let $u = \sin^2(\pi x)\sin(\pi y)$. Then we can see $u = 0$ when $x, y = 0, 1$. So it satisfy the boundary condition. Then

$$\begin{aligned}
\Delta u &= \frac{\partial^2}{\partial x^2}u + \frac{\partial^2}{\partial y^2}u \\
&= 2\pi^2 \sin(\pi y)(\cos^2(\pi x) - \sin^2(\pi x)) - \pi^2 \sin^2(\pi x)\sin(\pi y) \\
&= 2\pi^2 \sin(\pi y)\cos^2(\pi x) - 3\pi^2 \sin(\pi y)\sin^2(\pi x) \\
f = -\Delta u + u &= -2\pi^2 \sin(\pi y)\cos^2(\pi x) + (3\pi^2 + 1)\sin(\pi y)\sin^2(\pi x)
\end{aligned} \tag{6}$$

The L^2 error with 10000 iterations and various number of mesh points is shown below.

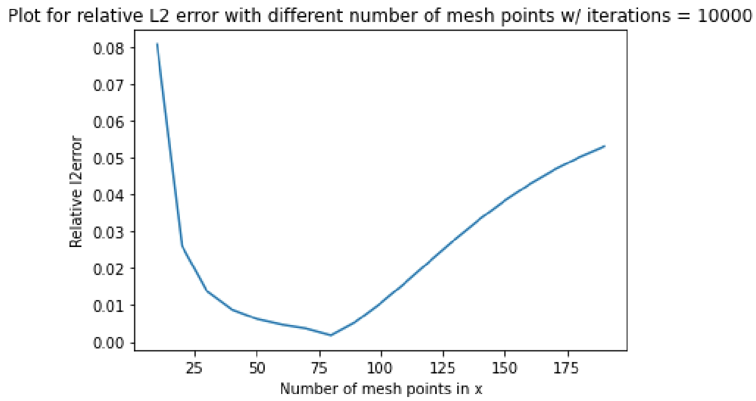

Figure 7: L^2 error vs number of mesh points at 10^4 iterations

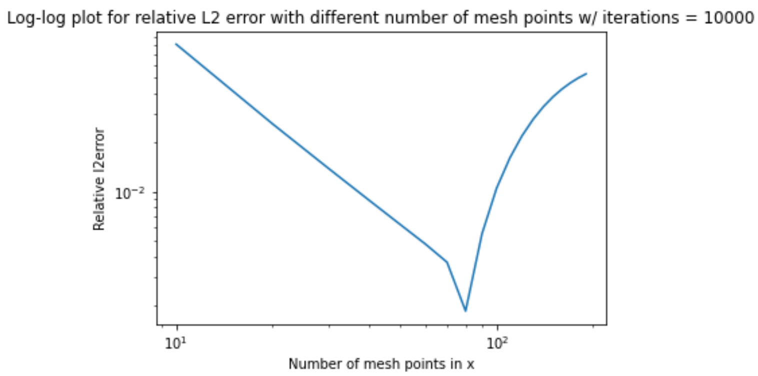

Figure 8: Log-log plot for L^2 error vs number of mesh points at 10^4 iterations

Interestingly, the scheme doesn't always converge as mesh points increases. From the graph, the error is lowest when there are 80 mesh points. Also, when I change the number of iterations, the mesh size corresponding to the lowest L^2 error also changes. Here is a graph or L^2 error vs iterations with mesh size $1/150$.

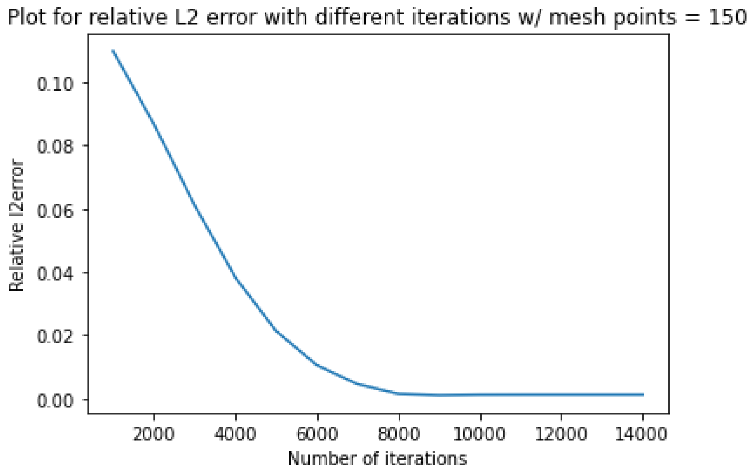

Figure 9: L^2 error vs number of iterations at mesh size 1/150

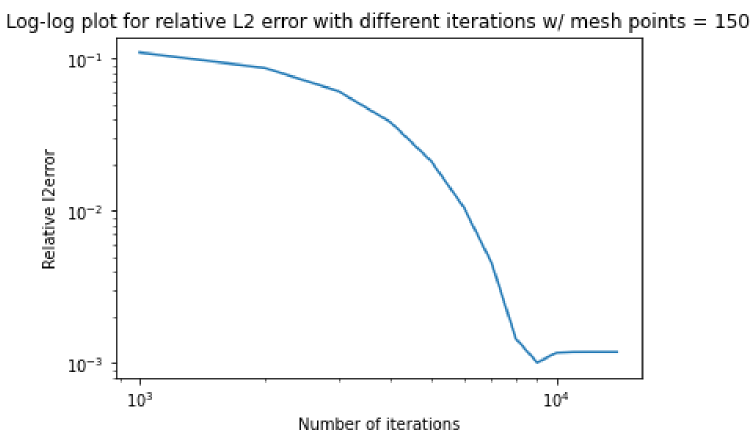

Figure 10: Loglog plot for L^2 error vs number of iterations at mesh size 1/150

The L^2 error is actually at lowest around 9000 iterations and stay the same afterwards. Here is a graph of L^2 error vs iterations at mesh size 1/80.

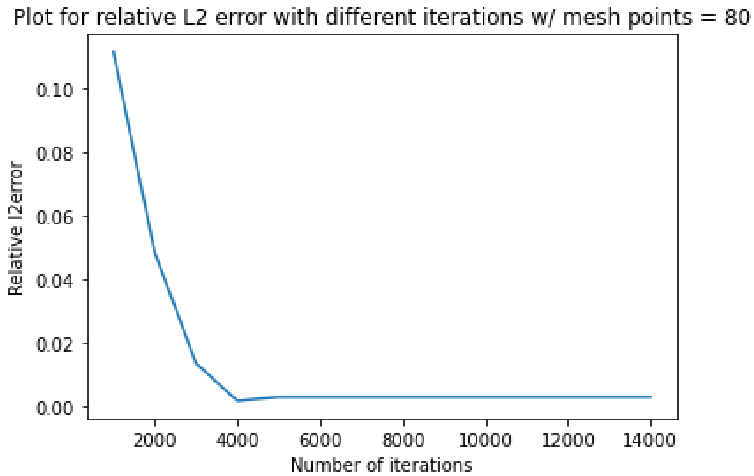

Figure 11: L^2 error vs number of iterations at mesh size 1/80

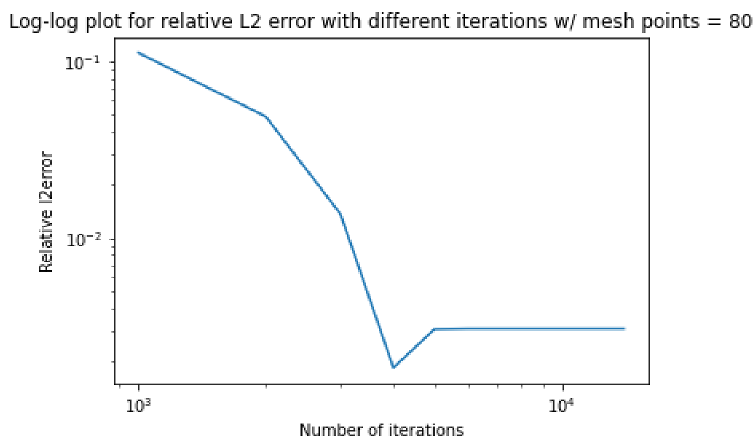

Figure 12: Loglog plot for L^2 error vs number of iterations at mesh size 1/80

The L^2 error is actually at lowest around 4000 iterations and stay the same afterwards.

2.3 Example 3

Let $\lambda = 1$ and let $u = x^2 \sin(\pi x) \sin(\pi y)$. Then we can see $u = 0$ when $x, y = 0, 1$. So it satisfy the boundary condition. Then

$$
\begin{aligned}
\Delta u &= \frac{\partial^2}{\partial x^2}u + \frac{\partial^2}{\partial y^2}u \\
&= \sin(\pi y)(4\pi x\cos(\pi x) - (\pi^2 x^2 - 2)\sin(\pi x)) - \pi^2 x^2 \sin(\pi x)\sin(\pi y) \\
&= \sin(\pi y)(4\pi x\cos(\pi x) - (2\pi^2 x^2 - 2)\sin(\pi x)) \\
f = -\Delta u + u &= \sin(\pi y)((2\pi^2 x^2 - 2 + x^2)\sin(\pi x) - 4\pi x\cos(\pi x))
\end{aligned} \tag{7}
$$

The L^2 error with 10000 iterations and various number of mesh points is shown below.

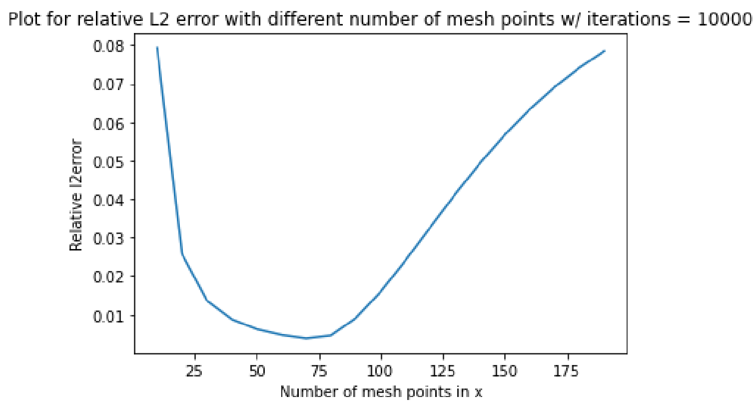

Figure 13: L^2 error vs number of mesh points at 10^4 iterations

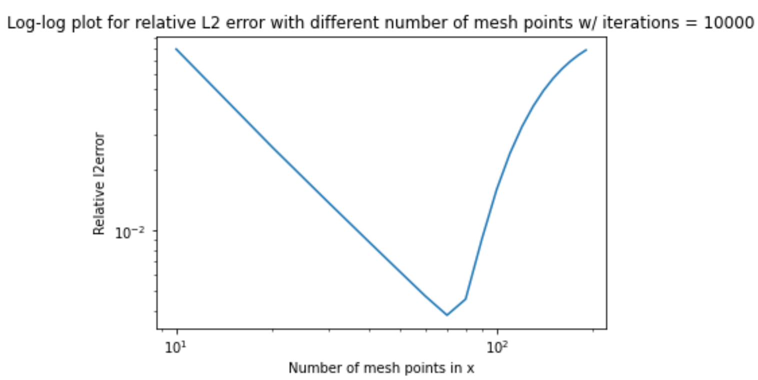

Figure 14: Log-log plot for L^2 error vs number of mesh points at 10^4 iterations

Interestingly, the scheme doesn't always converge as mesh points increases. From the graph, the error is lowest when there are 80 mesh points. Also, when I change the number of iterations, the mesh size corresponding to the lowest L^2 error also changes. Here is a graph or L^2 error vs iterations with mesh size 1/80.

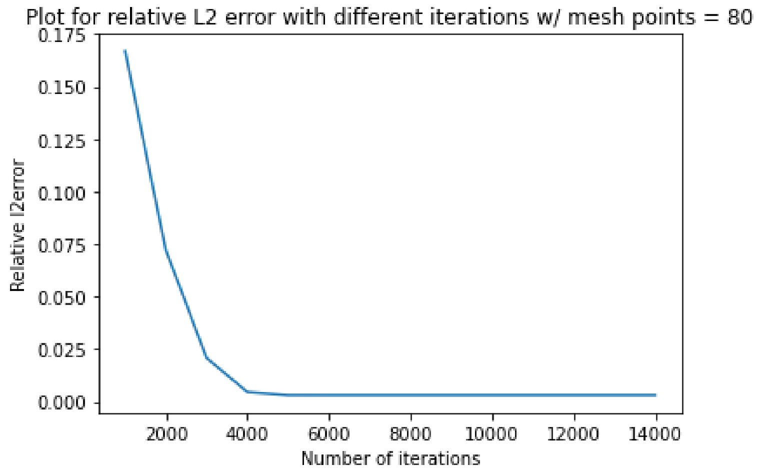

Figure 15: L^2 error vs number of iterations at mesh size 1/80

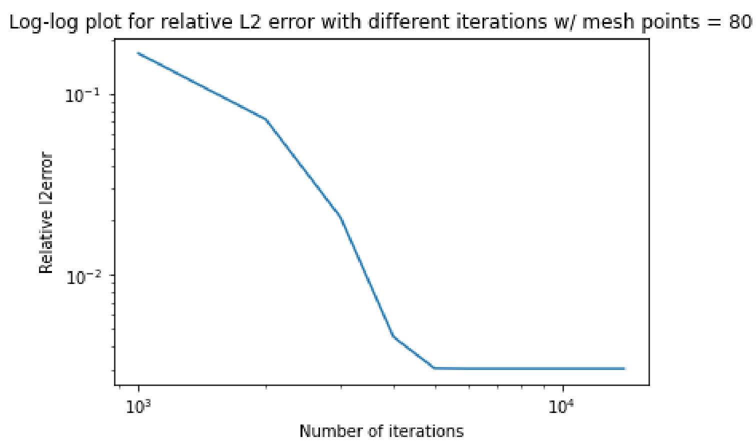

Figure 16: Loglog plot for L^2 error vs number of iterations at mesh size 1/80

The L^2 error is actually at lowest around 5000 iterations and stay the same afterwards.

The code is attached. It was partially adapted from https://github.com/daleroberts/poisson/blob/m with slight modification.

3 Conclusion

From the graph, one can observe that the relative L^2 decrease with respect to number of iterations. But after certain threshhold, the scheme won't perform better and the relative L^2 error stays the same. But the L^2 error doesn't increase with respect to the number of mesh points. And the model doesn't converge with very fine mesh.