APMA 922: Homework Set 02

Joseph Lucero

October 8, 2019

Problem 1

Part A

We begin by discretizing the equation of motion. Let $\Theta_i \equiv \theta(i\Delta t)$ where $\Delta t = {}^{T}/_{n+1}$ and n is the number of internal time points that we wish to solve for

$$\ddot{\theta}(t) + \sin(\theta(t)) = \frac{1}{h^2}\left[\Theta_{i+1} - 2\Theta_i + \Theta_{i-1}\right] + \sin(\Theta_i). \tag{1}$$

We define

$$G_i(\theta) \equiv \frac{1}{h^2}\left[\Theta_{i+1} - 2\Theta_i + \Theta_{i-1}\right] + \sin(\Theta_i) \tag{2}$$

as well as the associated Jacobian

$$J_{ij}(\theta) = \frac{1}{h^2}\begin{cases} 1, & j = i+1 \text{ or } j = i-1 \\ -2 + h^2\cos\Theta_i, & j = i \\ 0, & \text{else.} \end{cases} \tag{3}$$

Then the kth iteration is given by

$$\theta^{k+1} = \theta^k + \boldsymbol{\delta} \tag{4}$$

where δ is the solution to the linear system

$$J(\theta)\boldsymbol{\delta} = -G(\theta^k). \tag{5}$$

This is implemented in the ipython notebook titled `A2Q1.ipynb`. The method is set up so that the algorithm continues the Newton refinement until either $\|\boldsymbol{\delta}\| < \varepsilon_{32}$ or when the number of iterations is greater than 500, after which I assume that no solution would be found. Here, ε_{32} denotes the machine epsilon for single-precision. As can be seen in Fig. 1, this implementation reproduces successfully Figs. 2.4 and 2.5 in **[LV]**.

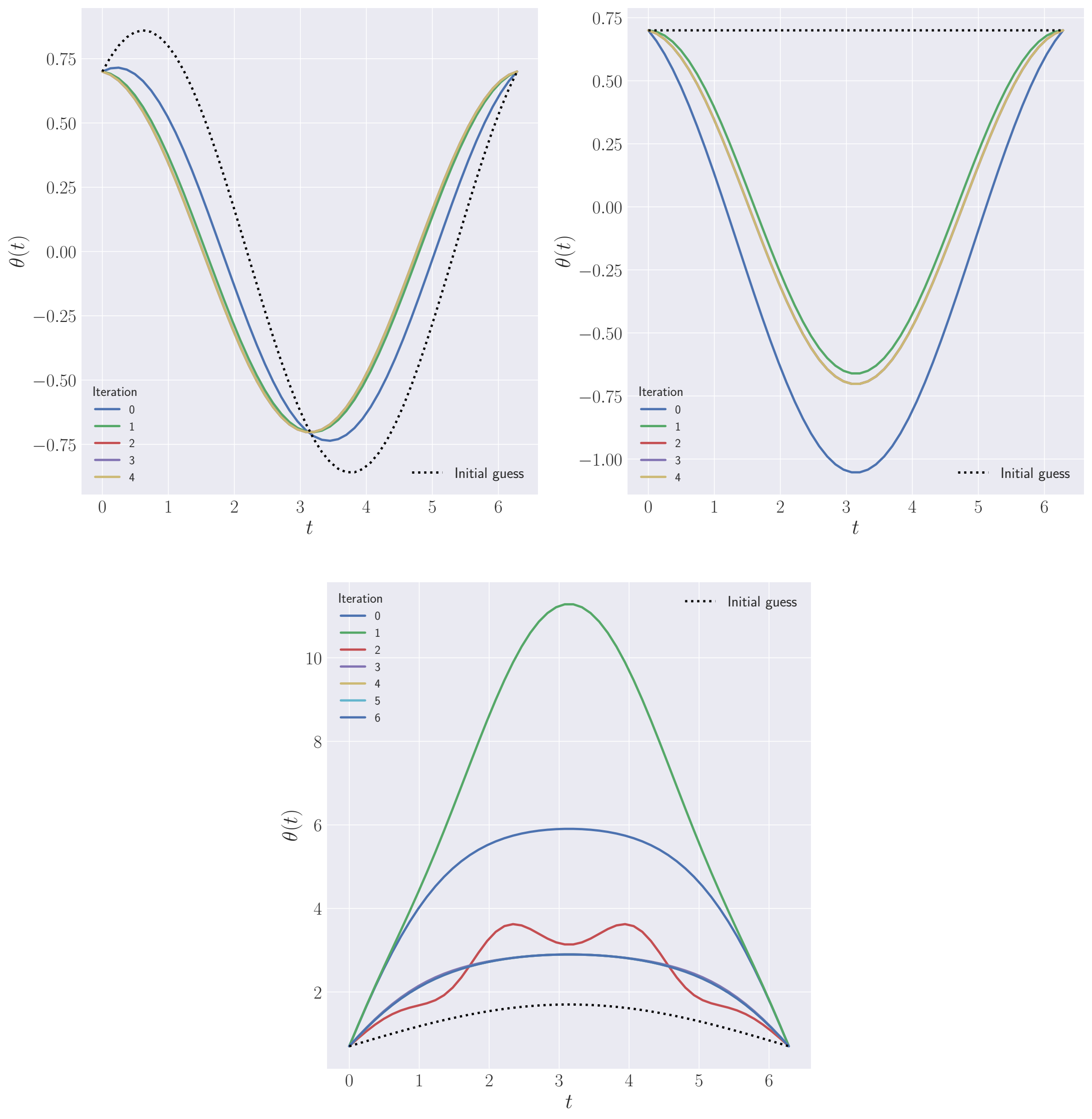

Figure 1: Recreation of Fig. 2.4 and Fig. 2.5 from **[LV]**. Angle θ as a function of time t. Initial guess is $\Theta_i^{[0]} = 0.7 + \sin(t_i/2)$. Different colors denote the iteration. Iteration counter starts from 0, not including initial guess. Initial guess is denoted by the black dotted line.

For an alternate initial solution $\theta^{[0]} = 0.7\cos(t_i) + 2\sin(t_i)$, I find an alternate solution shown in Fig. 2. This oscillation is similar to the one found in the top left-hand subplot of Fig. 1; however, this has a larger amplitude.

Part B

For longer time intervals, say $T = 20$, I find the following solution shown in Fig. 3 using the initial guess of $\theta^{[0]} = 0.9\cos(t_i) + 0.2\sin(t_i)$.

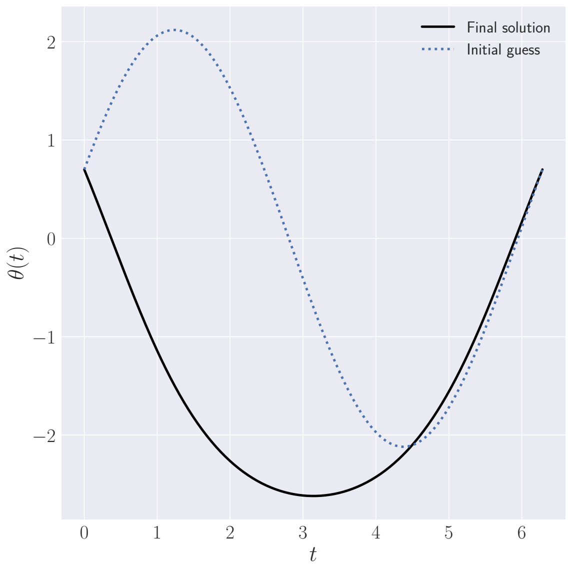

Figure 2: Solution of the Newton Iteration with initial guess $\theta^{[0]} = 0.7\cos(t_i) + 2\sin(t_i)$. Dashed blue line denotes the form of the initial guess, while the black solid line denotes the final solution found by the Newton iterator.

In general, as I anticipate the existence of the boundary layer, and from physical intuition I expect that such a solution to be slow moving near $\theta = \pi$, then propose an initial solution

$$\theta^{[0]} = \begin{cases} 0.7, & t = 0 \;\&\; t = T \\ \pi, & 0 < t < T. \end{cases} \tag{6}$$

Doing the Newton iteration with this guess gives a solution found in the left-hand subplot of Fig. 4. Then scanning across a bunch of different end times T, we find (right-hand subplot of Fig. 4) that the max θ asymptotes towards a value of $\theta = \pi$ as we can expect from physical intuition.

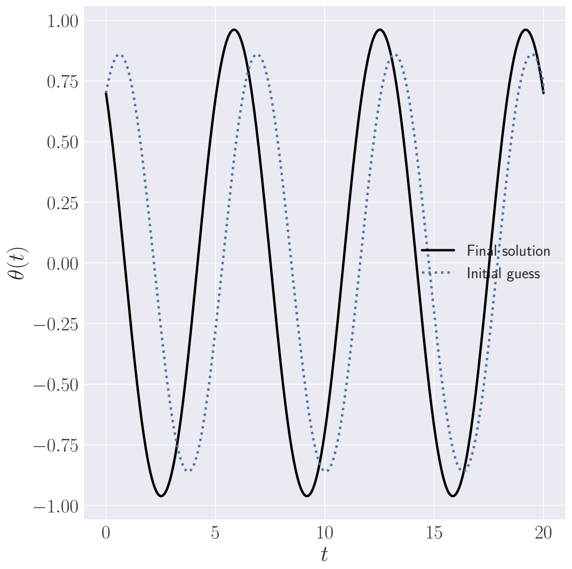

Figure 3: Solution found by the Newton Iteration for $T = 20$ using the initial solution $\theta^{[0]} = 0.9\cos(t_i) + 0.2\sin(t_i)$. Black solid line denotes the final solution after the convergence criterion was met. The dotted blue line denotes the initial guess input into the solver.

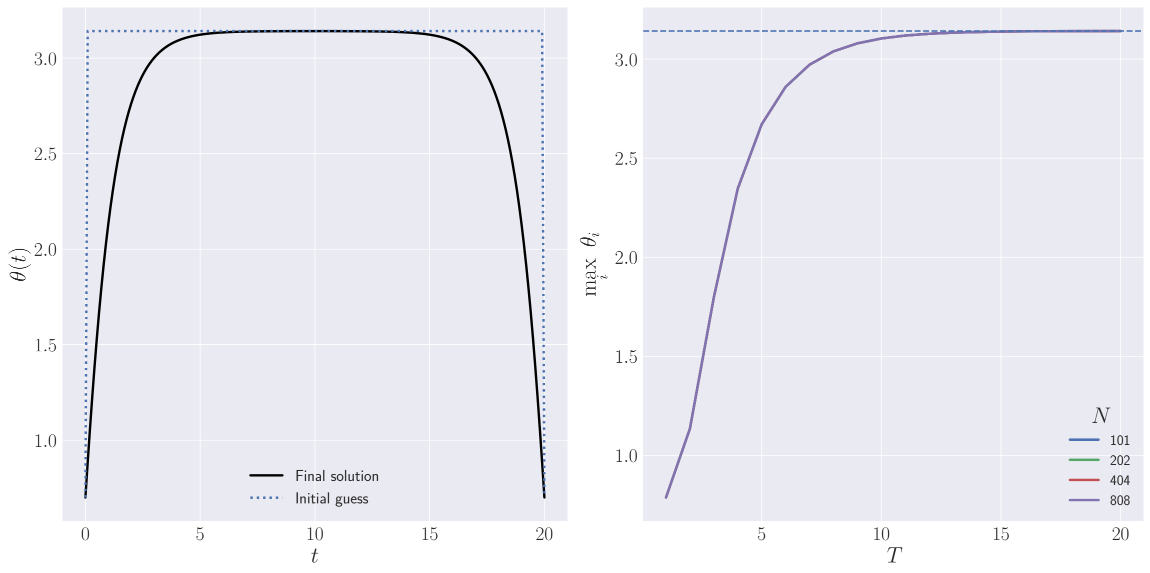

Figure 4: [Left Subplot] Solution found by the Newton Iteration for a range of T and for a variety of meshes with differing number of points m. [Right Subplot] Solution found by the Newton Iteration for a range of T and for a variety of meshes with differing number of points m. Using the initial solution given by (6). Different colors of lines denote different number of mesh points, which are indistinguishable in this case as each converges to the same answer. Blue dashed line denotes the value of π.

Problem 2

Part A

I implemented the five-point Laplacian by taking the code that was available on Canvas, translating it into Python, and then modifying the boundary conditions. This code can be found in the ipython notebook `A2Q2.ipynb`. Figure 5 shows the results of this implementation by reproducing all of the example figures in the provided MATLAB code.

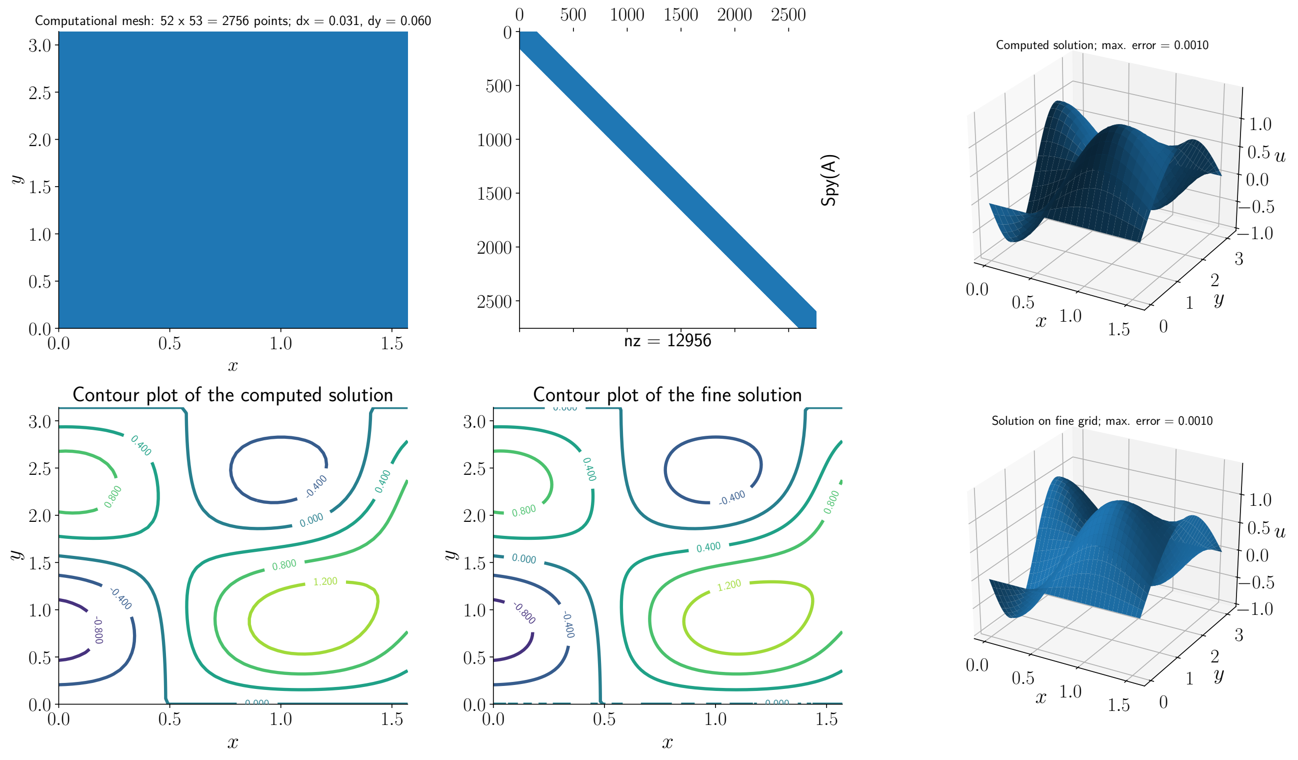

Figure 5: Recreation of the example plots for the differential equation $\nabla^2 u = 13\cos 3x \sin 2y$, using the five-point Laplacian.

Performing grid refinement using this implementation of the five-point Laplacian, we observe in Fig. 6 that this implementation has accuracy of order two. From the grid refinement procedure I estimate that the number of points needed to reach an inf-norm error of $< 10^{-5}$ is about 400 points

Part B

The nine-point Laplacian is given by the formula

$$\nabla_9^2 u_{ij} = \frac{1}{6h^2}\left[4(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}) + (u_{i+1,j-1} + u_{i+1,j+1} + u_{i-1,j-1} + u_{i-1,j+1}) - 20u_{ij}\right] \tag{7}$$

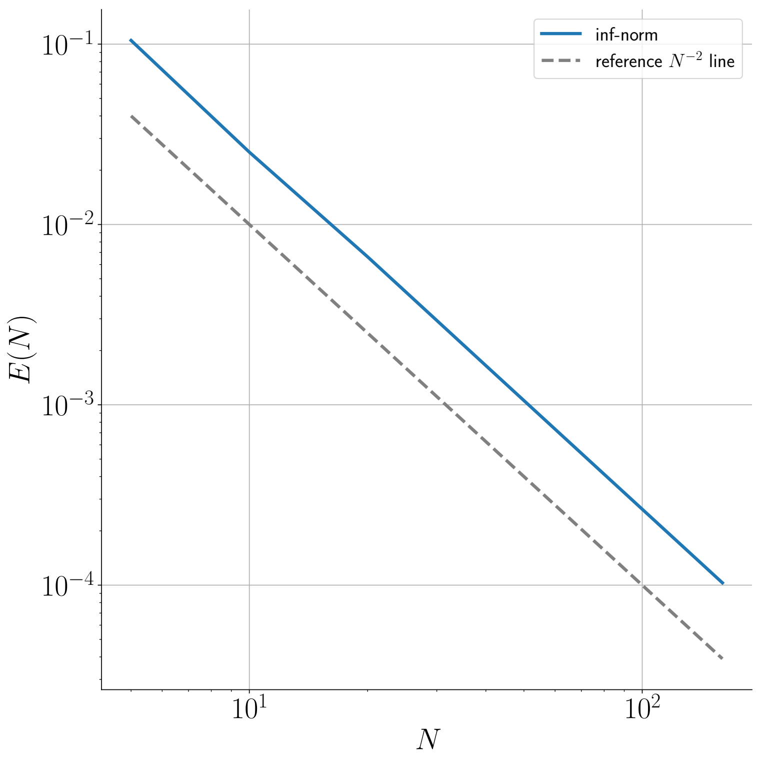

Figure 6: Error, $E(N)$, as a function of the number of mesh intervals, N, for the five-point Laplacian. We observe the desired order of two for this method.

To derive the truncation error we Taylor expand each of the u_{ij}. To simplify notation we define the shorthand,

$$u^{(m,n)}(\bar{x},\bar{y}) \equiv \frac{\partial^{m+n}}{\partial x^m \partial y^n} u(x,y)\Big|_{x=\bar{x},y=\bar{y}}. \tag{8}$$

Then, expanding each of the terms in (7), we have

$$\begin{aligned} u_{i-1,j} = u(\bar{x},\bar{y}) - hu^{(1,0)}(\bar{x},\bar{y}) + \frac{1}{2}h^2u^{(2,0)}(\bar{x},\bar{y}) - \frac{1}{6}h^3u^{(3,0)}(\bar{x},\bar{y}) + \frac{1}{24}h^4u^{(4,0)}(\bar{x},\bar{y}) \\ - \frac{1}{120}h^5u^{(5,0)}(\bar{x},\bar{y}) + \mathcal{O}\left(h^6\right) \end{aligned} \tag{9a}$$

$$\begin{aligned} u_{i+1,j} = u(\bar{x},\bar{y}) + hu^{(1,0)}(\bar{x},\bar{y}) + \frac{1}{2}h^2u^{(2,0)}(\bar{x},\bar{y}) + \frac{1}{6}h^3u^{(3,0)}(\bar{x},\bar{y}) + \frac{1}{24}h^4u^{(4,0)}(\bar{x},\bar{y}) \\ + \frac{1}{120}h^5u^{(5,0)}(\bar{x},\bar{y}) + \mathcal{O}\left(h^6\right) \end{aligned} \tag{9b}$$

$$\begin{aligned} u_{i,j-1} = u(\bar{x},\bar{y}) - hu^{(0,1)}(\bar{x},\bar{y}) + \frac{1}{2}h^2u^{(0,2)}(\bar{x},\bar{y}) - \frac{1}{6}h^3u^{(0,3)}(\bar{x},\bar{y}) + \frac{1}{24}h^4u^{(0,4)}(\bar{x},\bar{y}) \\ - \frac{1}{120}h^5u^{(0,5)}(\bar{x},\bar{y}) + \mathcal{O}\left(h^6\right) \end{aligned} \tag{9c}$$

$$u_{i,j+1} = u(\bar{x},\bar{y}) + hu^{(0,1)}(\bar{x},\bar{y}) + \frac{1}{2}h^2u^{(0,2)}(\bar{x},\bar{y}) + \frac{1}{6}h^3u^{(0,3)}(\bar{x},\bar{y}) + \frac{1}{24}h^4u^{(0,4)}(\bar{x},\bar{y})$$

$$+\frac{1}{120}h^5u^{(0,5)}(\bar{x},\bar{y})+\mathcal{O}\left(h^6\right) \quad (9d)$$

Adding these equations to compute the first group of terms inside the bracket of (7) we have that

$$\begin{aligned}4(u_{i-1,j}+u_{i+1,j}+u_{i,j-1}+u_{i,j+1}) = 16u(\bar{x},\bar{y}) &+ 4h^2\left(u^{(0,2)}(\bar{x},\bar{y})+u^{(2,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{3}h^4\left(u^{(0,4)}(\bar{x},\bar{y})+u^{(4,0)}(\bar{x},\bar{y})\right)+\mathcal{O}\left(h^6\right)\end{aligned} \quad (10)$$

Computing the expansions of the remaining terms we have

$$\begin{aligned}u_{i-1,j-1} = u(\bar{x},\bar{y}) &+ h\left(-u^{(0,1)}(\bar{x},\bar{y})-u^{(1,0)}(\bar{x},\bar{y})\right)+\frac{1}{2}h^2\left(u^{(0,2)}(\bar{x},\bar{y})+2u^{(1,1)}(\bar{x},\bar{y})+u^{(2,0)}(\bar{x},\bar{y})\right)\\ &-\frac{1}{6}h^3\left(u^{(0,3)}(\bar{x},\bar{y})+3\left(u^{(1,2)}(\bar{x},\bar{y})+u^{(2,1)}(\bar{x},\bar{y})\right)+u^{(3,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{24}h^4\left(u^{(0,4)}(\bar{x},\bar{y})+4u^{(1,3)}(\bar{x},\bar{y})+6u^{(2,2)}(\bar{x},\bar{y})+4u^{(3,1)}(\bar{x},\bar{y})+u^{(4,0)}(\bar{x},\bar{y})\right)\\ &-\frac{1}{120}h^5\left(u^{(0,5)}(\bar{x},\bar{y})+5\left(u^{(1,4)}(\bar{x},\bar{y})+2\left(u^{(2,3)}(\bar{x},\bar{y})+u^{(3,2)}(\bar{x},\bar{y})\right)+u^{(4,1)}(\bar{x},\bar{y})\right)+u^{(5,0)}(\bar{x},\bar{y})\right)\\ &+\mathcal{O}\left(h^6\right)\end{aligned} \quad (11a)$$

$$\begin{aligned}u_{i-1,j+1} = u(\bar{x},\bar{y}) &+ h\left(u^{(0,1)}(\bar{x},\bar{y})-u^{(1,0)}(\bar{x},\bar{y})\right)+\frac{1}{2}h^2\left(u^{(0,2)}(\bar{x},\bar{y})-2u^{(1,1)}(\bar{x},\bar{y})+u^{(2,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{6}h^3\left(u^{(0,3)}(\bar{x},\bar{y})-3u^{(1,2)}(\bar{x},\bar{y})+3u^{(2,1)}(\bar{x},\bar{y})-u^{(3,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{24}h^4\left(u^{(0,4)}(\bar{x},\bar{y})-4u^{(1,3)}(\bar{x},\bar{y})+6u^{(2,2)}(\bar{x},\bar{y})-4u^{(3,1)}(\bar{x},\bar{y})+u^{(4,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{120}h^5\left(u^{(0,5)}(\bar{x},\bar{y})-5u^{(1,4)}(\bar{x},\bar{y})+5\left(2u^{(2,3)}(\bar{x},\bar{y})-2u^{(3,2)}(\bar{x},\bar{y})+u^{(4,1)}(\bar{x},\bar{y})\right)-u^{(5,0)}(\bar{x},\bar{y})\right)\\ &+\mathcal{O}\left(h^6\right)\end{aligned} \quad (11b)$$

$$\begin{aligned}u_{i+1,j-1} = u(\bar{x},\bar{y}) &+ h\left(u^{(1,0)}(\bar{x},\bar{y})-u^{(0,1)}(\bar{x},\bar{y})\right)+\frac{1}{2}h^2\left(u^{(0,2)}(\bar{x},\bar{y})-2u^{(1,1)}(\bar{x},\bar{y})+u^{(2,0)}(\bar{x},\bar{y})\right)\\ &-\frac{1}{6}h^3\left(u^{(0,3)}(\bar{x},\bar{y})-3u^{(1,2)}(\bar{x},\bar{y})+3u^{(2,1)}(\bar{x},\bar{y})-u^{(3,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{24}h^4\left(u^{(0,4)}(\bar{x},\bar{y})-4u^{(1,3)}(\bar{x},\bar{y})+6u^{(2,2)}(\bar{x},\bar{y})-4u^{(3,1)}(\bar{x},\bar{y})+u^{(4,0)}(\bar{x},\bar{y})\right)\\ &-\frac{1}{120}h^5\left(u^{(0,5)}(\bar{x},\bar{y})-5u^{(1,4)}(\bar{x},\bar{y})+5\left(2u^{(2,3)}(\bar{x},\bar{y})-2u^{(3,2)}(\bar{x},\bar{y})+u^{(4,1)}(\bar{x},\bar{y})\right)-u^{(5,0)}(\bar{x},\bar{y})\right)\\ &+\mathcal{O}\left(h^6\right)\end{aligned} \quad (11c)$$

$$\begin{aligned}u_{i+1,j+1} = u(\bar{x},\bar{y}) &+ h\left(u^{(0,1)}(\bar{x},\bar{y})+u^{(1,0)}(\bar{x},\bar{y})\right)+\frac{1}{2}h^2\left(u^{(0,2)}(\bar{x},\bar{y})+2u^{(1,1)}(\bar{x},\bar{y})+u^{(2,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{6}h^3\left(u^{(0,3)}(\bar{x},\bar{y})+3\left(u^{(1,2)}(\bar{x},\bar{y})+u^{(2,1)}(\bar{x},\bar{y})\right)+u^{(3,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{24}h^4\left(u^{(0,4)}(\bar{x},\bar{y})+4u^{(1,3)}(\bar{x},\bar{y})+6u^{(2,2)}(\bar{x},\bar{y})+4u^{(3,1)}(\bar{x},\bar{y})+u^{(4,0)}(\bar{x},\bar{y})\right)\\ &+\frac{1}{120}h^5\left(u^{(0,5)}(\bar{x},\bar{y})+5\left(u^{(1,4)}(\bar{x},\bar{y})+2\left(u^{(2,3)}(\bar{x},\bar{y})+u^{(3,2)}(\bar{x},\bar{y})\right)+u^{(4,1)}(\bar{x},\bar{y})\right)+u^{(5,0)}(\bar{x},\bar{y})\right)\end{aligned}$$

$$+\mathcal{O}\left(h^6\right). \tag{11d}$$

Combining these terms to compute the remaining group of terms gives

$$\begin{aligned} u_{i+1,j-1} + u_{i+1,j+1} + u_{i-1,j-1} + u_{i-1,j+1} = 4u(\bar{x},\bar{y}) + 2h^2\left(u^{(0,2)}(\bar{x},\bar{y}) + u^{(2,0)}(\bar{x},\bar{y})\right) \\ + \frac{1}{6}h^4\left(u^{(0,4)}(\bar{x},\bar{y}) + 6u^{(2,2)}(\bar{x},\bar{y}) + u^{(4,0)}(\bar{x},\bar{y})\right) + \mathcal{O}\left(h^6\right). \end{aligned} \tag{12}$$

Substituting (10) and (12) into (7) and simplifying yields

$$\nabla_9^2 u(\bar{x},\bar{y}) = \left(u^{(0,2)}(\bar{x},\bar{y}) + u^{(2,0)}(\bar{x},\bar{y})\right) + \frac{1}{12}h^2\left(u^{(0,4)}(\bar{x},\bar{y}) + 2u^{(2,2)}(\bar{x},\bar{y}) + u^{(4,0)}(\bar{x},\bar{y})\right) + \mathcal{O}\left(h^4\right) \tag{13a}$$

$$= \nabla^2 u(\bar{x},\bar{y}) + \frac{1}{12}h^2\nabla^2\left(\nabla^2 u(\bar{x},\bar{y})\right) + \mathcal{O}\left(h^4\right) \tag{13b}$$

$$= \nabla^2 u(\bar{x},\bar{y}) + \frac{1}{12}h^2\nabla^4 u(\bar{x},\bar{y}) + \mathcal{O}\left(h^4\right) \tag{13c}$$

Computing the truncation error then gives

$$\nabla_9^2 u(\bar{x},\bar{y}) - \nabla^2 u(\bar{x},\bar{y}) = \frac{1}{12}h^2\nabla^4 u(\bar{x},\bar{y}) + \mathcal{O}\left(h^4\right). \tag{14}$$

The nine-point Laplacian is only a valid approximation if the mesh width in both dimensions is equivalent. As such here we set the number of point m to specify the number of internal mesh points in the x-direction and from this we compute the necessary n number of internal mesh points in the y-direction such that the grid spacing in both dimensions are equivalent:

$$\Delta y = \Delta x \tag{15a}$$

$$\frac{L_y}{n+1} = \frac{L_x}{m+1} \tag{15b}$$

$$\implies n = \frac{L_y}{L_x}\left[m+1\right] - 1. \tag{15c}$$

Implementing this in in the ipython notebook `A2Q2.ipynb`, in the section marked “Part B”. Figure 7, shows the recreation of the example plots using the nine-point Laplacian. We see that the implementation successfully reproduces the results acquired from the 5-point Laplacian.
Like the 5-point Laplacian, this implementation of the 9-point Laplacian yields an observed accuracy order of two, which can be seen in Fig. 8.

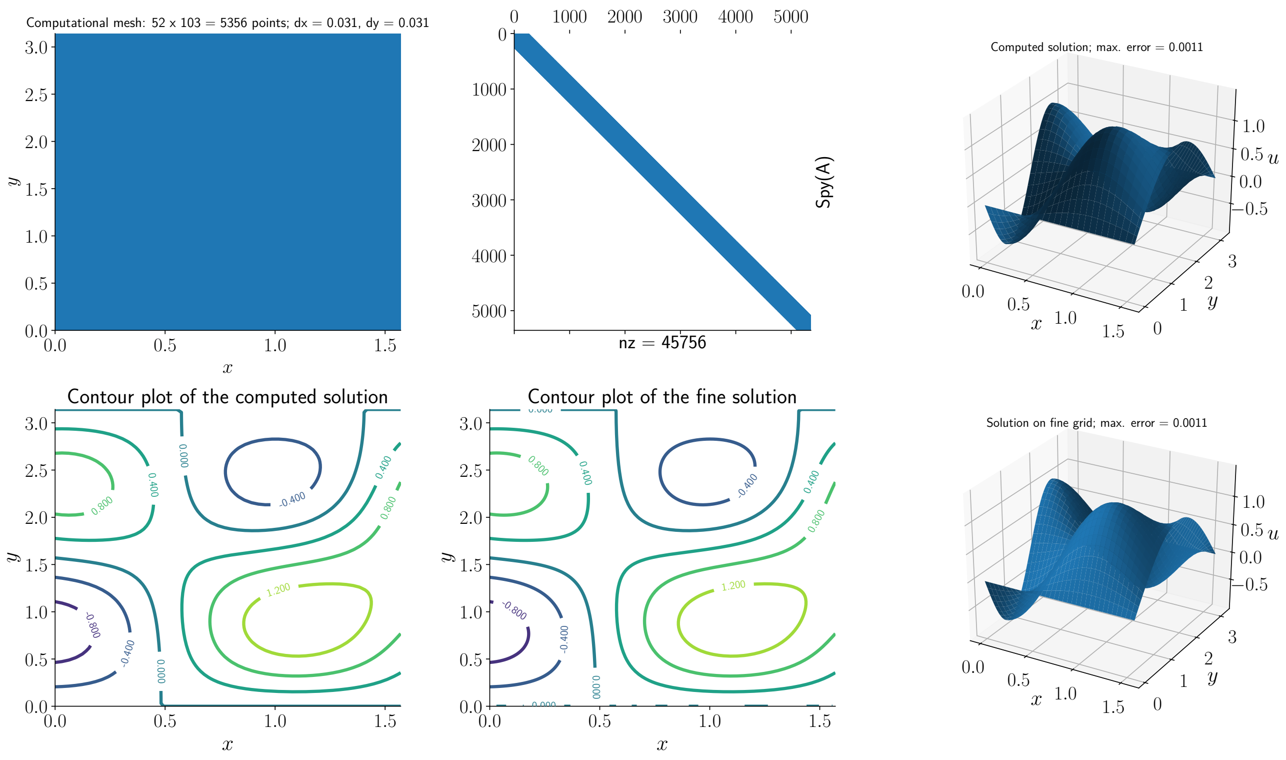

Figure 7: Recreation of the example plots using the 9-point Laplacian.

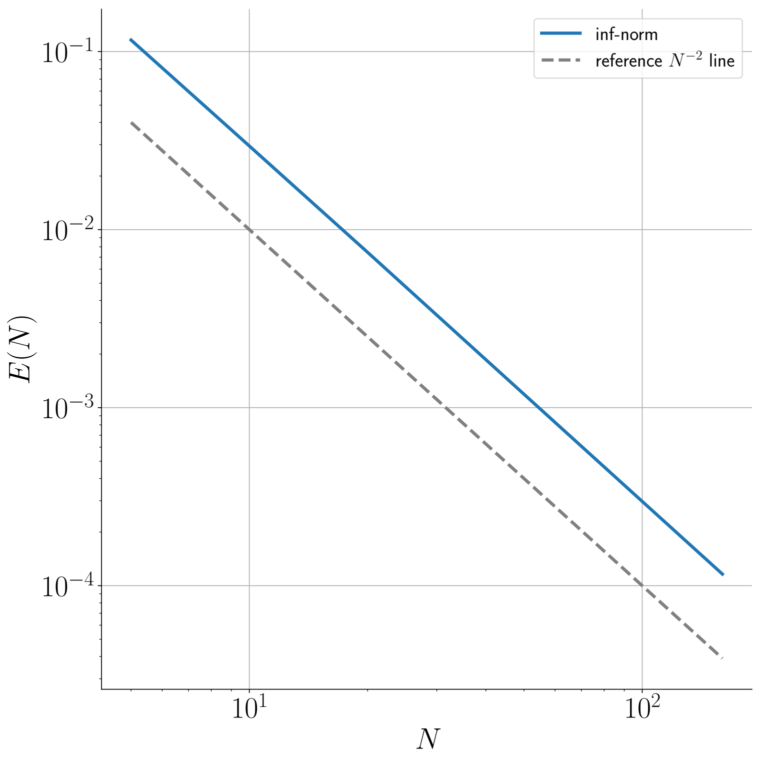

Figure 8: Error, $E(N)$, as a function of the number of mesh intervals, N, for the nine-point Laplacian. We observe an order of two for the accuracy for this method.

Implementing the deferred corrections by changing the right hand side of the PDE from,

$$\begin{aligned} f(x_i, y_j) &\to \tilde{f}(x_i, y_j) && (16a)\\ &= f(x_i, y_j) + \frac{h^2}{12}\nabla^2 f(x_i, y_j) && (16b)\\ &= 13\cos(3x)\sin(2y) - \frac{h^2}{12}169\cos(3x)\sin(2y), && (16c)\end{aligned}$$

we acquire the error scaling observed in Fig. 9. We observe that this correction increases the order of the method by two from two to four. From the grid refinement (see ipython notebook), I estimate that we would need 350 grid points in the x-direction and correspondingly 700 grid points in the y-direction to acquire an inf-norm error of $< 10^{-5}$.

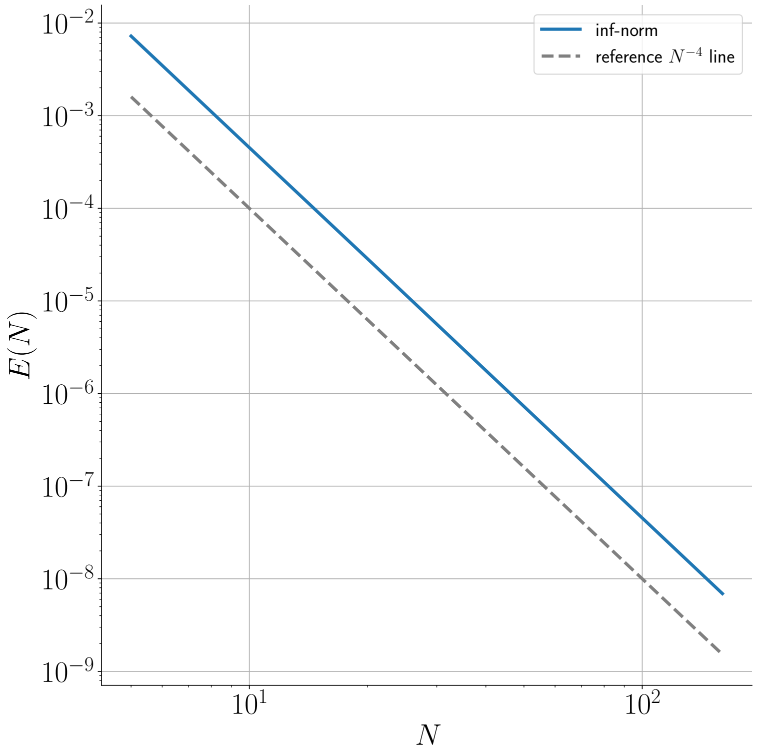

Figure 9: Error, $E(N)$, as a function of the number of mesh intervals, N, for the nine-point Laplacian with deferred corrections. We observe the desired order of four for this method.

Part C

We implement the Neumann Boundary condition using a second-order one-sided difference approximation so that

$$u_x(x = 0, y_j) \approx \frac{1}{h}\left[-\frac{3}{2}U_{0,j} + 2U_{1,j} - \frac{1}{2}U_{2,j}\right] = \frac{1}{2}\sin y_j. \qquad (17)$$

Performing grid refinement on the implementation of the Neumann boundary condition using the five-point stencil, gives the desired order of two as can be seen in Fig. 10. I suspect that the reason why this method deviates from the order 2 scaling at low N is because in those cases, the stencil of this boundary condition takes up the majority of points on the grid.

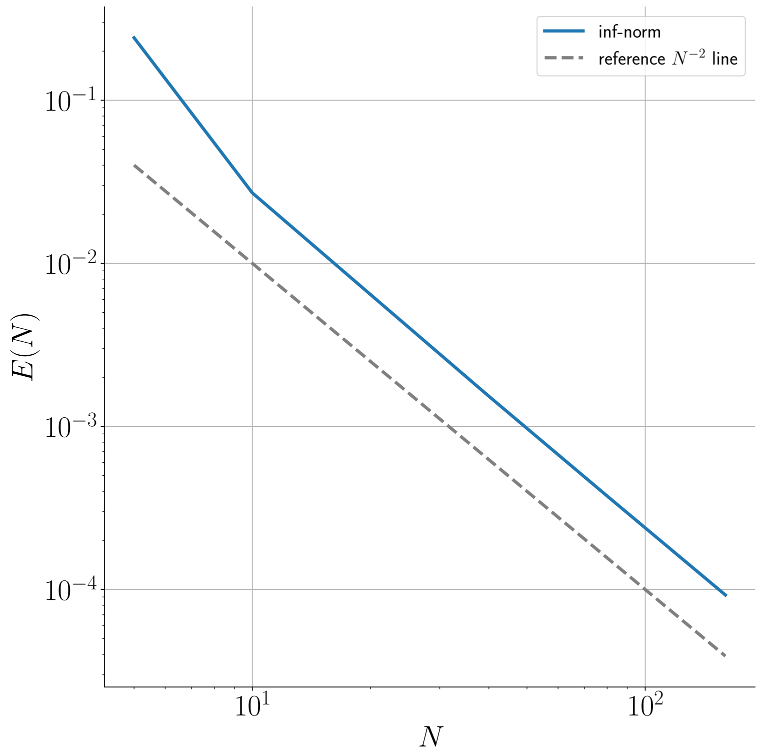

Figure 10: Error, $E(N)$, as a function of the number of mesh intervals, N, for the five-point Laplacian with Neumann boundary conditions at $x = 0$. We observe the desired order of two for this method.