Computational Partial Differential Equations
Project 3: Hyperbolic Systems

Tudor Trita Trita
CID 01199397

April 10, 2020

This is my own work unless stated otherwise.

Structure of Coursework:

The coursework .zip file contains the following:

- partA.py: Code used to generate figures and output for Part A.
- partB.py: Code used to generate figures and output for Part B.
- waves.py: Script containing classes for generating numerical schemes for the 1D and 2D waves.
- Proj3_01199397.pdf: (This document) Compiled LaTeX report.

Software Requirements:

The code for the coursework has been written in Python 3.7 and the following third-party packages have been used:

- Numpy (1.18.1): Used for access to fast array operations.
- Scipy (1.4.1): Used for to sparse arrays and operations.
- Matplotlib (3.1.3): Used for plotting capabilities.

1 Part A

1.1 Introduction to Problem Statement

In this question, we are asked to investigate how the solution to the one-dimensional wave equation for $u(x,t)$ given by

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2}, \tag{1}$$

behaves, where x represents a spatial coordinate and t a temporal coordinate. We are given the initial conditions

$$u(x,0) = \cos\left(\frac{\pi x}{2\delta}\right), \quad \frac{\partial u}{\partial t}(x,0) = 0, \quad \text{for } (-\delta \leq x \leq \delta), \tag{2}$$

and $u(x,0) = \partial u(x,0)/\partial t = 0$ everywhere else.

1.2 Discretisation using Leap-Frog Scheme

The discretised leap-frog scheme given in the question is given by the equation

$$\frac{U_j^{k+1} - 2U_j^k + U_j^{k-1}}{(\Delta t)^2} = \frac{U_{j-1}^k - 2U_j^k + U_{j+1}^k}{(\Delta x)^2}. \tag{3}$$

If we let $h := \Delta x,\ k := \Delta t,\ r = k/h$ equation (3) can be expressed to solve for the next time-step U^{k+1} to give the equation

$$U_j^{k+1} = r^2(U_{j-1}^k + U_{j+1}^k) + 2(1-r^2)U_j^k - U_j^{k-1} \tag{4}$$

If we let our grid consist of N points in the x-direction and $M+1$ points in the t-direction, then we can solve the scheme (4) at the x_j points for $j = 1, \ldots, N$ and $\Delta x = h = 4/(N-1), \Delta t = k = T/M$, where T is the time we wish to compute the numerical solution at.

1.3 Initial Conditions

We now need to discretise the initial conditions given. We are given the initial condition $u(x,0) = \cos\left(\frac{\pi x}{2\delta}\right)$, and the discretised version is given by $U_j^0 = \cos\left(\frac{\pi x_j}{2\delta}\right)$ for $j = 1, \ldots, N$.

We can use a second-order finite difference approximation to the first derivative to obtain the other set of initial conditions,

$$\frac{U_j^1 - U_j^{-1}}{2k} = 0 \quad \Longrightarrow \quad U_j^1 = U_j^{-1},$$

which we can substitute into equation (4) to solve for the first time-step as follows

$$U_j^1 = \frac{r^2}{2}(U_{j-1}^0 + U_{j+1}^0) + (1-r^2)U_j^0. \tag{5}$$

1.4 Boundary Conditions

At our boundaries, i.e. at $x = \pm 2$, corresponding to the discretised points $x_1 = -2, x_N = 2$, we will need to take care with the corresponding boundary conditions.

For part (a), the boundaries satisfy the property that there are minimal numerical reflections off the outer boundaries. This can be viewed as waves passing and leaving the domain $|x| \leq 2$ or the waves being absorbed at the boundaries.

At the left boundary, only a left-moving wave can leave our domain of interest, and we can impose the following approximation to the open boundary conditions at $x = -2$

$$\frac{\partial u}{\partial t} - \frac{\partial u}{\partial x} = 0. \tag{6}$$

If we discretise this equation using 2nd order central finite differences we obtain

$$\frac{U_1^{k+1} - U_1^k}{2k} - \frac{U_2^k - U_0^k}{2h} = 0, \tag{7}$$

$$\implies U_0^{k+1} = U_2^k + \frac{1}{r}(U_1^{k-1} - U_1^{k+1}), \tag{8}$$

leaving us with the ghost point U_0, which upon substituting into (4) we obtain the boundary condition

$$U_1^{k+1} = \frac{1}{1+r}(2r^2 U_2^k + 2(1-r^2)U_1^k + (r-1)U_1^{k-1}). \tag{9}$$

At the right boundary, only a right-moving wave can leave our domain of interest, and we can impose the following approximation to the open boundary conditions at $x = 2$

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0. \tag{10}$$

If we discretise this equation using 2nd order central finite differences we obtain

$$\frac{U_N^{k+1} - U_N^k}{2k} - \frac{U_{N+1}^k - U_{N-1}^k}{2h} = 0, \tag{11}$$

$$\implies U_{N+1}^{k+1} = U_{N-1}^k + \frac{1}{r}(U_N^{k-1} - U_N^{k+1}), \tag{12}$$

leaving us with the ghost point U_{N+1}, which upon substituting into (4) we obtain the boundary condition

$$U_N^{k+1} = \frac{1}{1+r}(2r^2 U_{N-1}^k + 2(1-r^2)U_N^k + (r-1)U_N^{k-1}). \tag{13}$$

At $k = 1$, we can use the Neumann conditions in time to obtain the modified Equations

$$U_1^1 = r^2 U_2^0 + (1-r^2)U_1^0, \tag{14}$$

$$U_N^1 = r^2 U_{N-1}^0 + (1-r^2)U_N^0. \tag{15}$$

We can use these equations in our numerical scheme to solve for the boundaries for the case where minimal reflections of the waves is to be achieved preserving 2nd order accuracy.

For part (b), we are given a solid wall and the waves reflect off this wall as soon as they touch it. This happens if and only if $u_x = 0$ at the boundaries, i.e.

$$\left.\frac{\partial u}{\partial x}\right|_{x=-2} = 0, \quad \left.\frac{\partial u}{\partial x}\right|_{x=2} = 0.$$

Using a second-order finite difference approximation to this partial derivative, we can obtain the discretised conditions

$$\frac{U_{j+1}^k - U_{j-1}^k}{2h} = 0 \quad \Longrightarrow \quad U_{j+1}^k = U_{j-1}^k,$$

for $j = 1, j = N$. Hence, we can substitute the points $U_0^k = U_2^k$, $U_{N+1}^k = U_{N-1}^k$, to obtain the discretised equations for the solid-wall condition at the boundaries

$$U_1^{k+1} = 2r^2 U_2^k + 2(1 - r^2)U_1^k - U_1^{k-1}, \tag{16}$$
$$U_N^{k+1} = 2r^2 U_{N-1}^k + 2(1 - r^2)U_N^k - U_N^{k-1}. \tag{17}$$

for $k > 1$. For $k = 1$, we need to use our initial conditions, so the formulas for the boundaries at the first time-step become

$$U_1^1 = r^2 U_2^0 + (1 - r^2)U_1^0, \tag{18}$$
$$U_N^1 = r^2 U_{N-1}^0 + (1 - r^2)U_N^0. \tag{19}$$

1.5 Example Waves

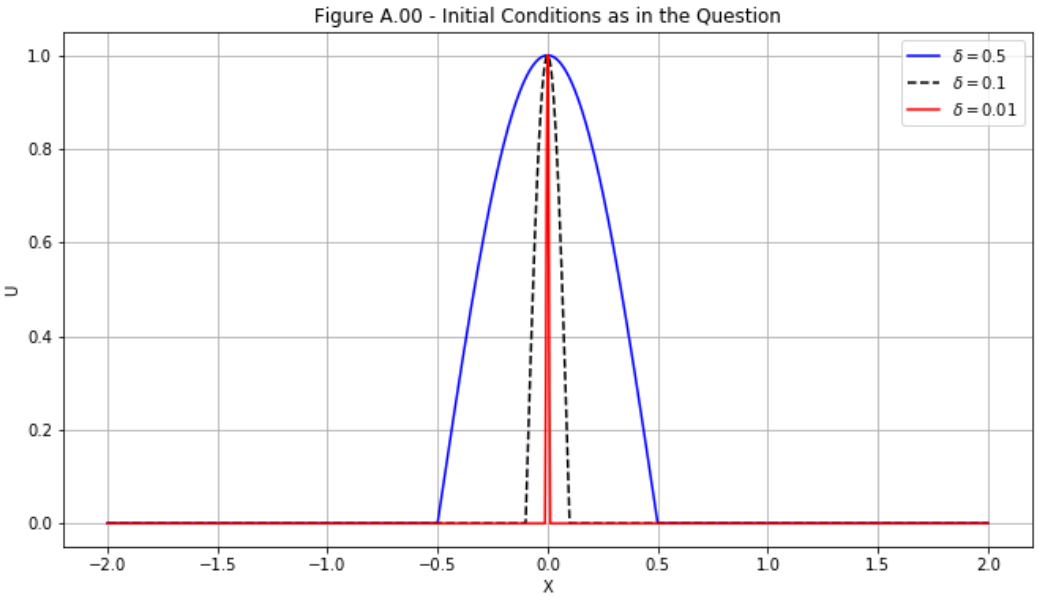
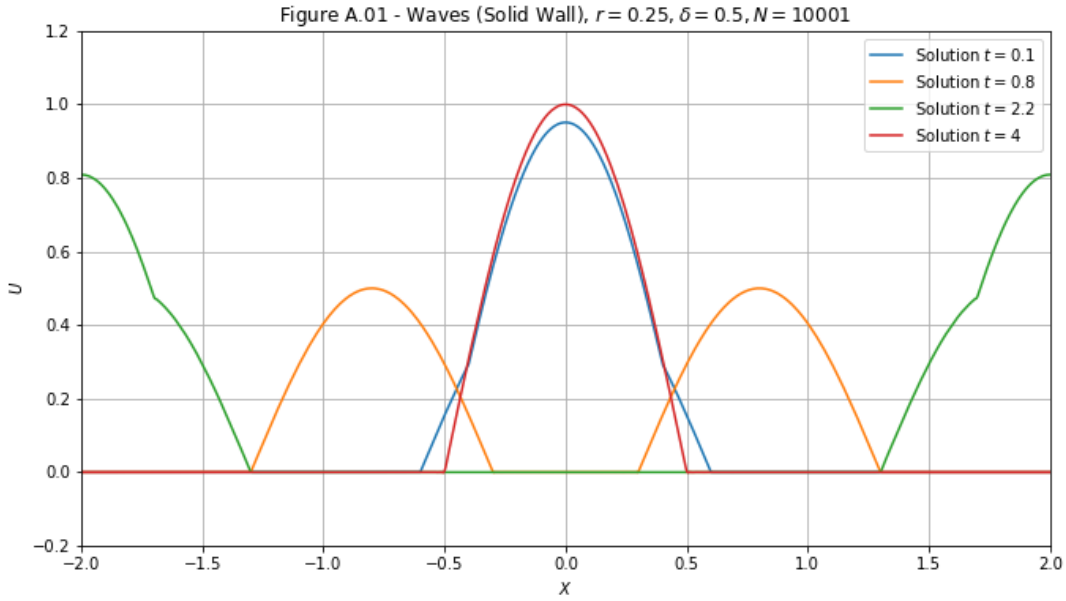
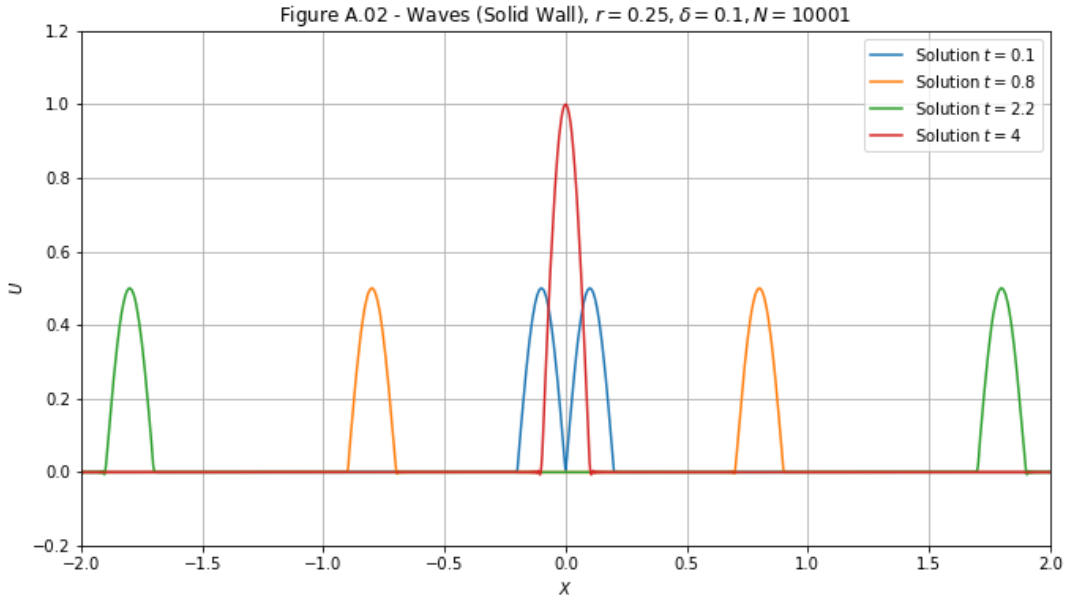
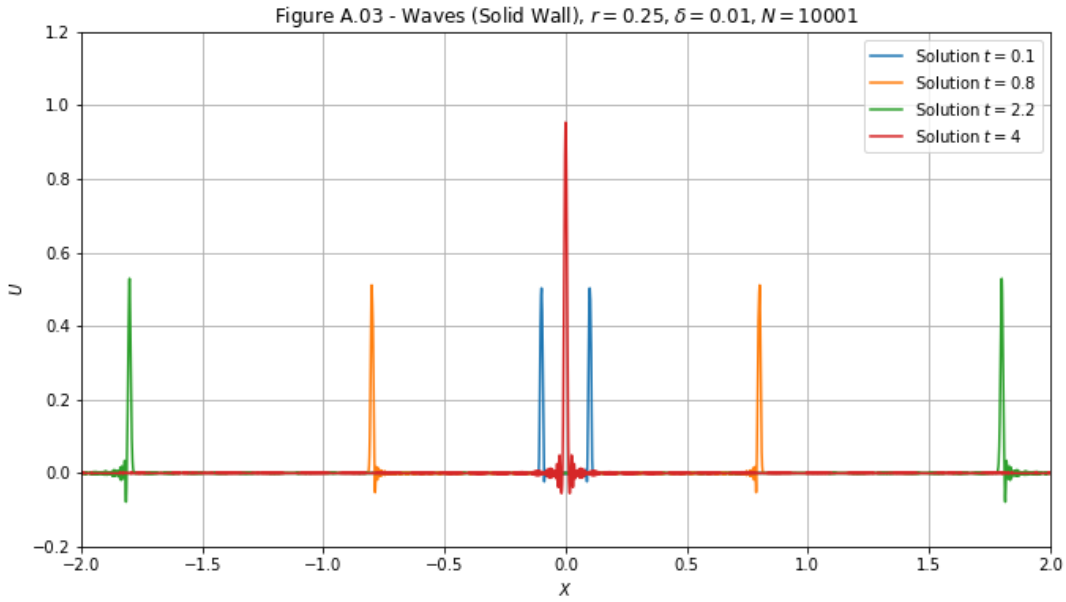


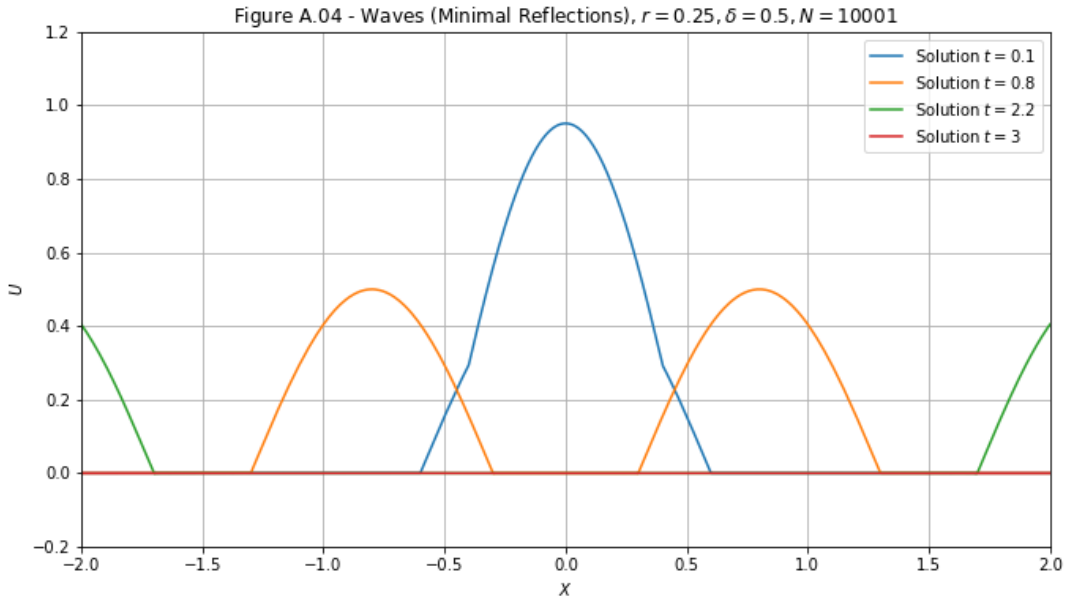
Figure A.00 shows the initial conditions as specified in the question. As we can see, the initial wave gets sharper and shrinks towards the center as we decrease δ. As we can see, at the points $x = \pm\delta$, the function becomes steeper as we decrease δ, which we expect to become more challenging 'numerically', in the sense that the discretisation may produce higher dissipation/dispersion effects as δ becomes very small.

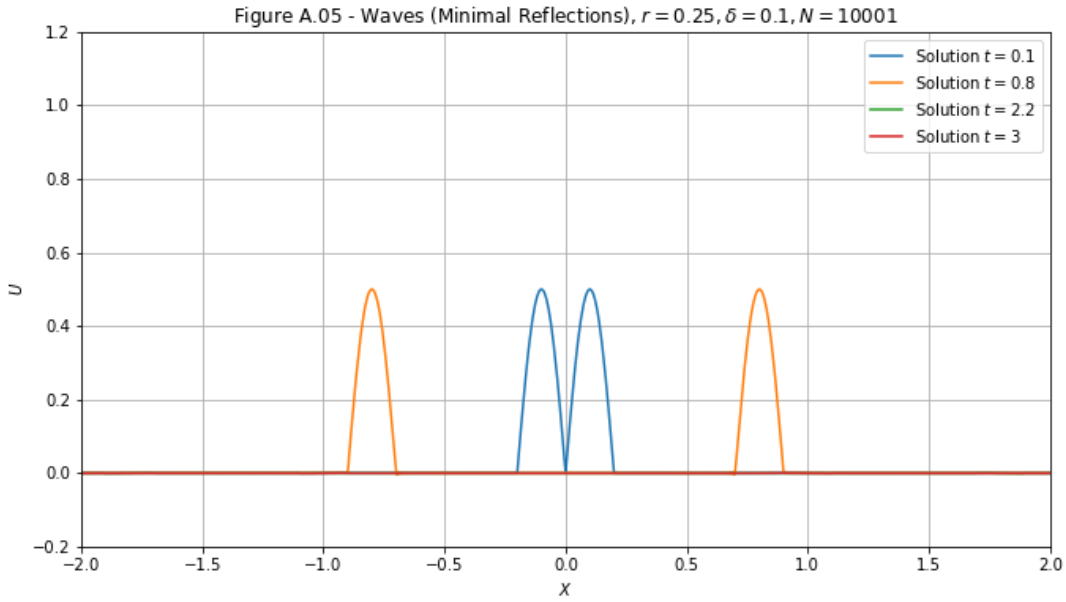


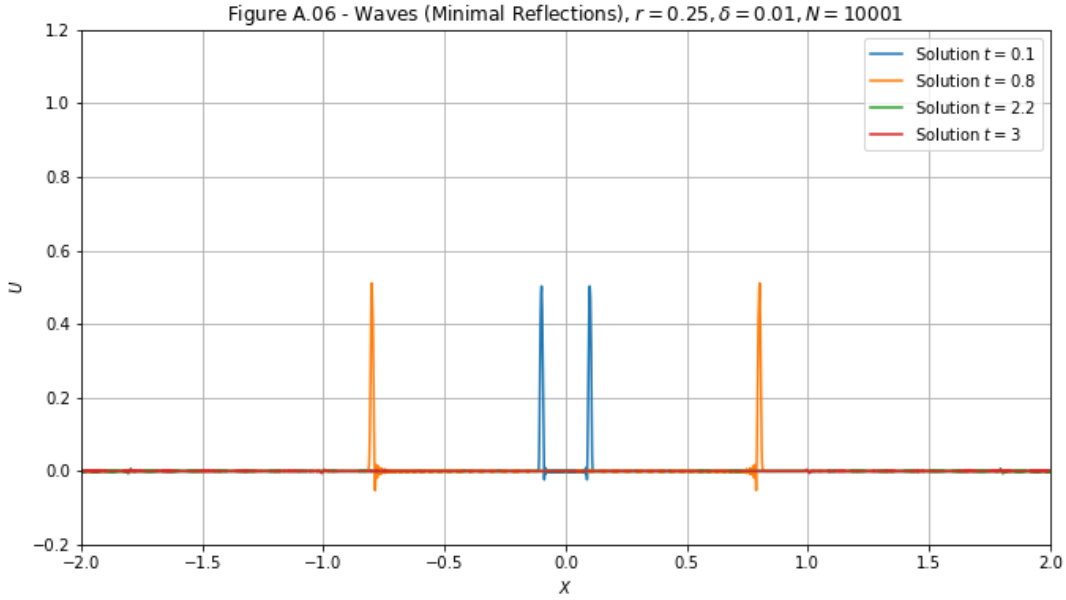




Figures A.01, A.02 and A.03 show how the solution behaves in time with the reflecting boundary conditions. As it can be seen, the solutions reflect off the boundaries once they hit it. As we can see, the smaller δ creates sharper discontinuities, introducing dispersive effects near these as seen by the oscillatory behaviour in Figure A.03. We will later explore how this depends on the Courant Number $C = c\Delta t/\Delta x = \Delta t/\Delta x = r$, where $c = 1$ in the wave equation. Since at $t = 4$, the wave returns to its initial condition, we can be confident that the implementation is correct, since the wave speed is equal to 1, the wave peak is at the expected point $x = 0$ and the shape is the same as the initial condition.







Figures A.01, A.02 and A.03 show how the solution behaves in time with the open boundary conditions. Solutions at $t = 2.2, t = 3$ are outside of the boundaries $x = -2, x = 2$ hence do not appear in our solution. Similarly to before, we can see dispersive effects appearing in Figure A.06.

Whilst we do not have an analytical solution for this problem, we can still be sure that our numerical scheme is accurate, since it behaves as we would expect. For example, the wave 'splits' in to two identical parts moving left and right, each with a peak that is half the size

of the initial condition's peak, as well as being > 0, all verifying that our numerical solution behaves as we would expect.

1.6 Verification of Grid Independence

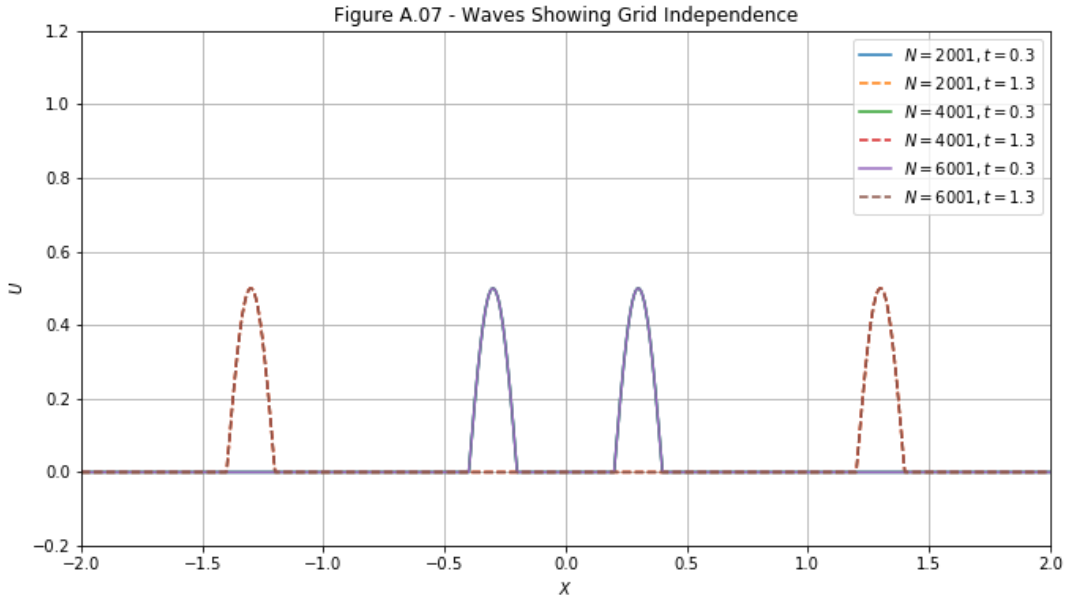


Figure A.07 shows grid independence since the solutions match up for different values of N. This shows that the solutions are independent of the coarseness of the grid. In this figure, we have taken $\delta = 0.1, r = 1$.

1.7 Definition of Dispersion & Dissipation

Dissipation effects of the numerical solution appear as stretching of sharp gradients in the solution. These gradients tend to appear flatter in the solution. Dissipation effects are due to the presence of extra even derivatives in the modified PDE ($u_{xx}, u_{xxxx}, \ldots$),

Dispersion effects of the numerical solution appear as oscillatory behaviour near the points of sharp gradients in the solution. Dispersion effects are due to the presence of extra odd derivatives in the modified PDE ($u_x, u_{xxx}, \ldots$)

Both dissipation and dispersion effects appear due to the Gibbs effect.

1.8 Modified PDE

Our original PDE reads $u_{tt} = u_{xx}$. If we Taylor expand the leapfrog discretisation around the points $U^k_{j+1} = u(j+\Delta x, k), U^k_{j-1} = u(j-\Delta x, k), U^{k+1}_j = u(j, k+\Delta t), U^{k-1}_j = u(j, k-\Delta t)$ we can obtain the following expression:

$$\frac{1}{(\Delta t)^2}\left[\left(u+\Delta t u_t+\frac{(\Delta t)^2}{2!}u_{tt}+\dots\right)\Big|_j^k-2u\Big|_j^k+\left(u-\Delta t u_t+\frac{(\Delta t)^2}{2!}u_{tt}+\dots\right)\Big|_j^k\right]$$
$$=\frac{1}{(\Delta x)^2}\left[\left(u+\Delta x u_x+\frac{(\Delta x)^2}{2!}u_{xx}+\dots\right)\Big|_j^k-2u\Big|_j^k+\left(u-\Delta x u_x+\frac{(\Delta x)^2}{2!}u_{xx}+\dots\right)\Big|_j^k\right].$$

After collecting terms up to fourth order and simplifying, we obtain the equation

$$u_{tt}+\frac{(\Delta t)^2}{12}u_{tttt}=u_{xx}+\frac{(\Delta x)^2}{12}u_{xxxx}. \tag{20}$$

Now, since $u_{tt}=u_{xx}$, $u_{tttt}=u_{ttxx}=u_{xxtt}=u_{xxxx}$, hence, equation (20) becomes

$$u_{tt}=u_{xx}+\frac{((\Delta x)^2-(\Delta t)^2)}{12}u_{xxxx}. \tag{21}$$

The u_{xxxx} accounts for dissipation in our numerical scheme, with the condition for this to disappear being $(\Delta x)^2=(\Delta t)^2 \implies \Delta x=\Delta t$. As we can see, there are no odd terms in the expansion, hence dispersion is not accounted by the modified PDE.

1.9 Stability of Wave Equation

From lectures, we know that the numerical solution is stable iff $\Delta t/\Delta x \leq 1$. This is seen in the code, since if we let $\Delta t/\Delta x > 1$, our solution blows up and we cannot march forward in time, verifying the theoretical properties.

1.10 Wave Speed and Dispersion

The other source of dispersion to account for the behaviour in Figures A.03, A.06 is the wave speed, or more precisely, the discrete-wave speed. For a general wave equation $u_{tt}=c^2u_{xx}, u(x,0)=u_0(x)$, it is well known that the solutions take the form

$$u(x,t)=\frac{1}{2}(u_0(x-ct)+u_0(x+ct)) \tag{22}$$

where $u_0(x-ct)$ is the left-moving wave with wave speed $\omega \propto -c$ and $u_0(x+ct)$ is the right-moving wave with wave speed $\omega \propto +c$.

We can express $u_0(x)$ in its Fourier series representation as follows

$$u_0(x)=\sum_n a_n e^{inx} \tag{23}$$

for some Fourier coefficients $a_n, n \geq 0$. Therefore, our solution becomes

$$u(x,t) = \frac{1}{2}\left(\sum_n a_n e^{in(x-ct)} + \sum_n a_n e^{in(x+ct)}\right) \tag{24}$$

and hence we see that our solution is a sum of individual left-moving waves $e^{in(x-ct)}$ and right-moving waves $e^{in(x+ct)}$, with speeds $\omega = -nc, \omega = +nc$ respectively.

Now, we can look at equation (3) and see that we can represent the discrete solution as the sum of individual discrete left-moving waves $e^{i(x_j - \hat{\omega} t_k)}$ and right-moving waves $e^{i(x_j + \hat{\omega} t_k)}$, with speeds $\hat{\omega} \neq \omega$ in general. After some algebra, we can solve for $\hat{\omega}$ to obtain the expression

$$\hat{\omega} = \frac{2}{\Delta t} \arcsin\left(c\frac{\Delta t}{\Delta x} \sin\left(\frac{n\Delta x}{2}\right)\right) \tag{25}$$

Now, if $c\Delta t/\Delta x = 1$, then 25 simplifies to $\hat{\omega} = \omega$, and so the discretised wave speed is exactly equal to the exact solution wave speed. If this condition does not hold, we expect discrepancies between the wave speeds and this will introduce dispersion in our solution.

1.11 Empirical Dissipation & Dispersion Effects

Since we know from figures A.04, A.05, A.06 that the dissipation/dispersion effects are strongest for small delta, we focus on showing these results for the case $\delta = 0.01$.

We begin by showing dissipation/dispersion on a different set of initial conditions, namely $u(x,0) = 1$, for $(-\delta \leq x \leq \delta)$. This serves as a control to make sure our solution is accurate as well. We expect the solution to be a rectangular-shaped wave moving to the left and right of the plane. Since the solution is symmetric, we focus on looking at the left-moving wave only and zoom into the solution.

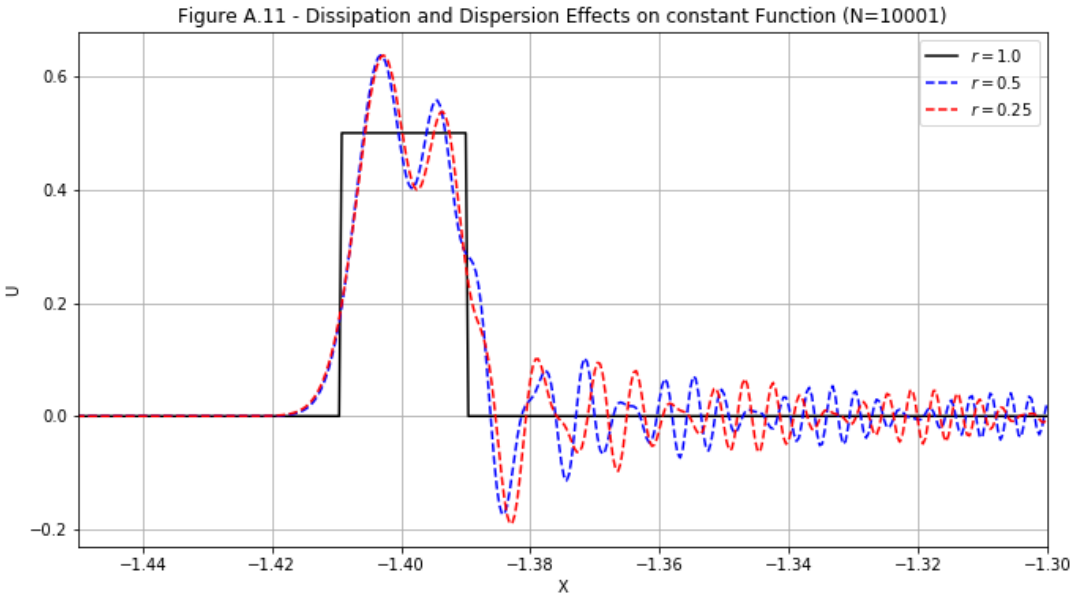
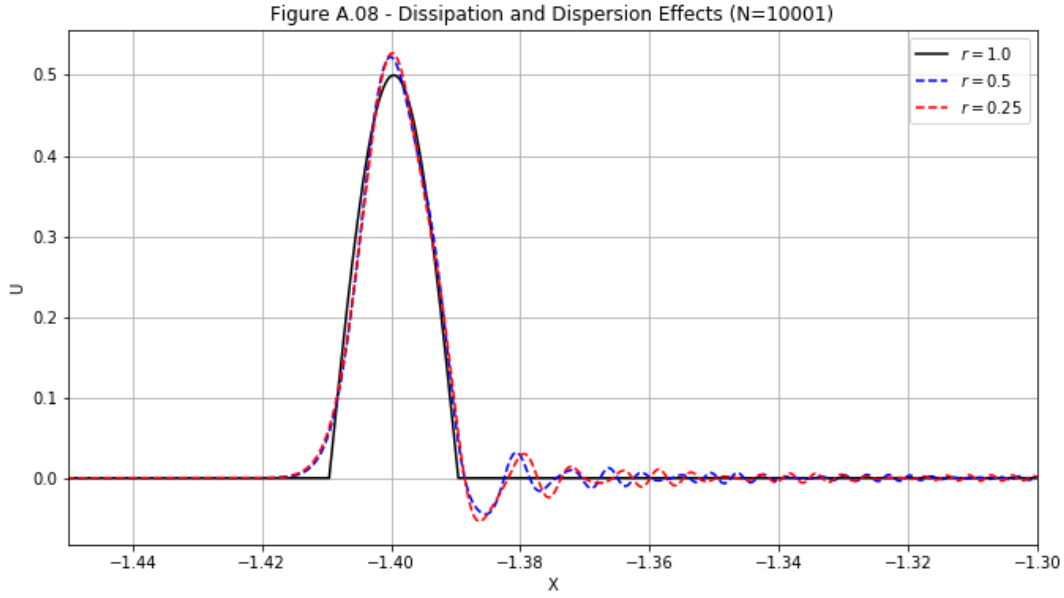
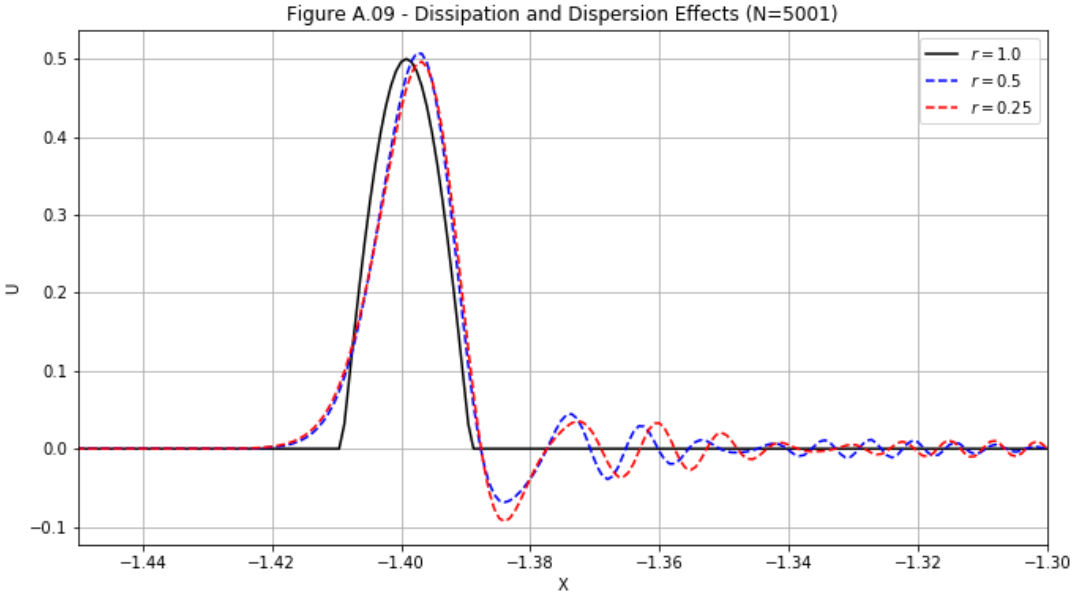
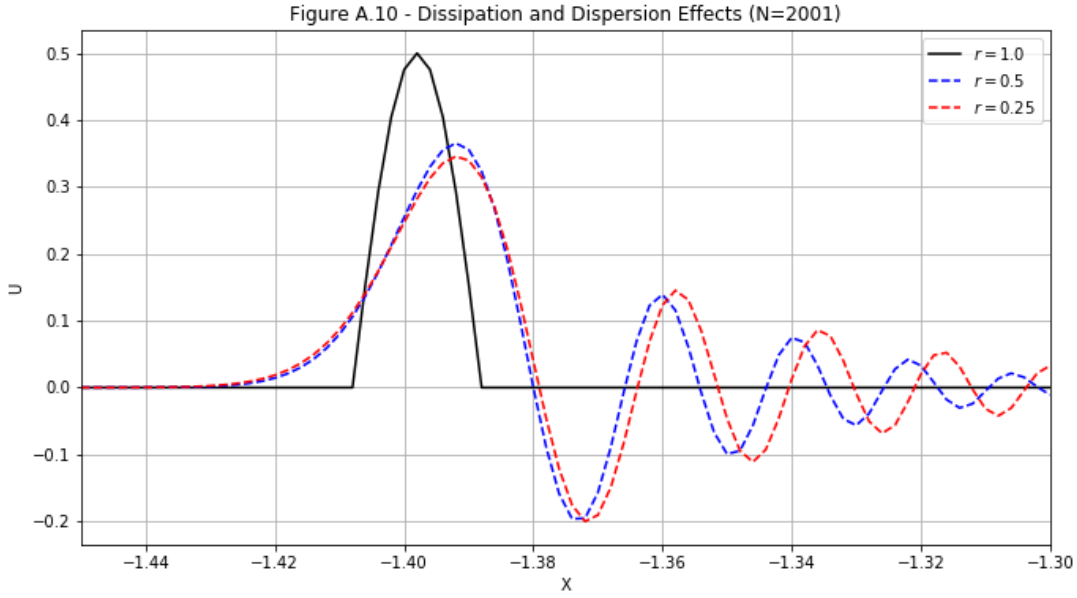


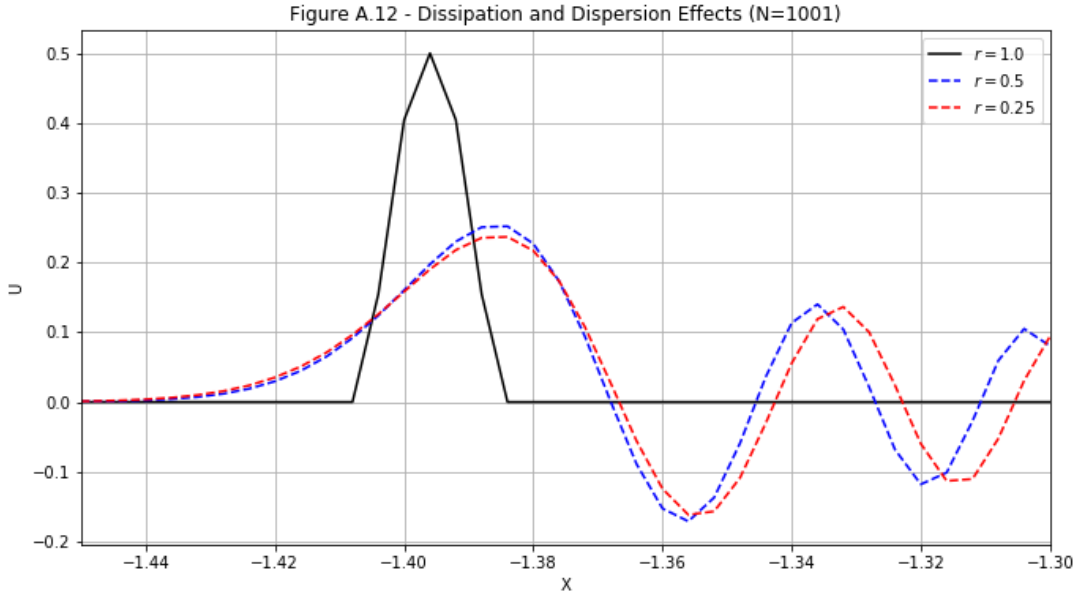
Figure A.11: When $r = 1$, we can clearly see that the solution is the correct one, since it is a rectangular wave with $u = 0.5$ and width $\delta = 0.01$. The dashed lines represent lowering the value of r. The dispersive effects can be seen around the point $x = -1.41$, since the jump from $u = 0$ to $u = 0.5$ appears as a smooth curve. The dispersion (and dissipation) effects can be seen for $x \geq -1.4$, since we have highly oscillatory solutions in that region. The wave-speed is also verified, since we are taking $T = 1.4, c = 1$, we know that the left-moving wave should be at $x = -1.4$ at this time, which is exactly where it is. This verifies the accuracy of our implementation, together with the shape, for $r = 1$.





Figures A.08, A.09 show the same effects as those seen in figure A.11, now using the original initial condition $u(x, 0) = \cos(50\pi x)$.





Figures A.10, A.12 show the effects of lowering our grid size N. We can see that by lowering the grid coarseness, the changes in r produce more dramatic effects, not only leading to more severe dissipation and dispersive effects, but also a change in the wave speed, since we see that the peaks of the $r = 0.5, r = 0.25$ solutions are shifted to the right of the expected peak at $x = -1.4$.

1.12 Verification of Accuracy of Implementation

To properly verify the accuracy of the implementation, we would usually need the exact solution and compare the numerical vs exact solution. Since we don't have the exact solution here, we can use another trick. Since the solid wall conditions should theoretically reflect our wave perfectly, we can use the fact that at times $t = 4n, n \in \mathbb{Z}$, the wave should return to its initial condition state, i.e. $u(x, 0) = u(x, 4n)$. Hence, if we let $u_0 = u(x, 0), u_4 = u(x, 4)$ we can compute the accuracy of our method by looking at the following quantity

$$error = \log_{10} \|u_4 - u_0\|_\infty = \log_{10} \max_i |(u_4 - u_0)_i|. \tag{26}$$

and seeing how it varies with δ. Since we know that the optimal $r = 1$, we take $\Delta t = \Delta x$, otherwise the dissipation/dispersive effects will kick in.

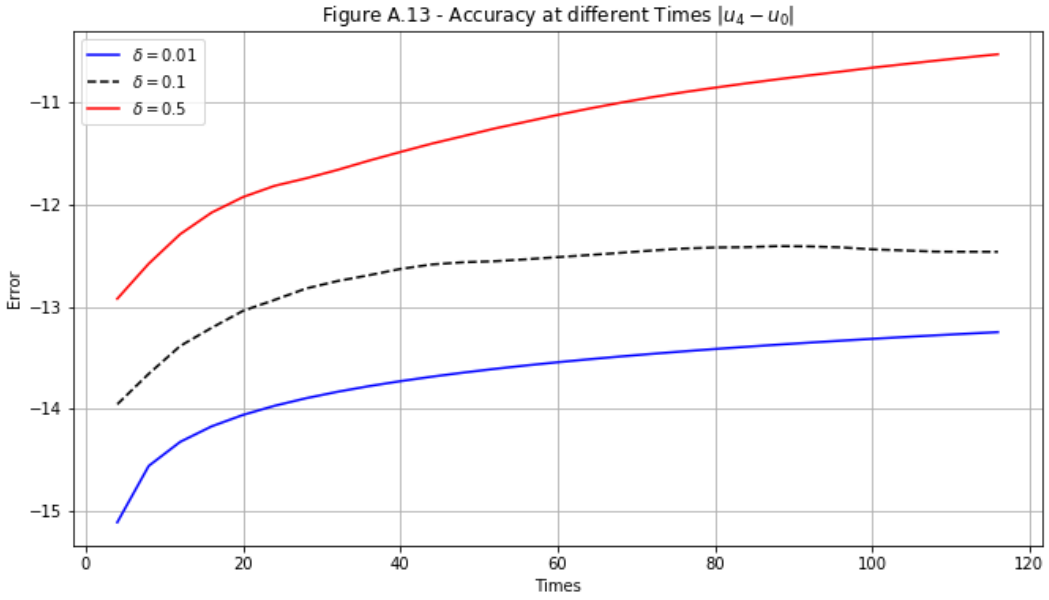


Figure A.13 shows the error value at different times for $N = 1001$ when we expect the wave to return back to the initial condition. As we can see, the values of errors are very low, implying that the accuracy of our implementation is correct.

From the plot, we see that the accuracy increases the lower we make δ.

2 Part B

2.1 Question 1

2.1.1 Derivation of 2D Scheme

We can discretise the 2D wave equation $u_{tt} = u_{xx} + u_{yy}$ using centred finite differences in the leap-frog method to obtain

$$\frac{U_{ij}^{k+1} - 2U_{ij}^{k} + U_{ij}^{k-1}}{(\Delta t)^2} = \frac{U_{i-1j}^{k} - 2U_{ij}^{k} + U_{i+1j}^{k}}{(\Delta x)^2} + \frac{U_{ij-1}^{k} - 2U_{ij}^{k} + U_{ij+1}^{k}}{(\Delta y)^2}. \tag{27}$$

For simplicity, we take $\Delta x = \Delta y := h$ and let $\Delta t := k,\ r = k/h$. Solving for $U_i j^k$, we can obtain the scheme

$$U_{ij}^{k+1} = 2(1 - 2r^2)U_{ij}^{k} + r^2(U_{i+1j}^{k} + U_{i-1j}^{k} + U_{ij+1}^{k} + U_{ij-1}^{k}) - U_{ij}^{k-1}. \tag{28}$$

From lectures, we know that the condition of stability for this scheme, taking $\Delta x = \Delta y$ is that

$$\frac{\Delta t}{\Delta x} = \frac{\Delta t}{\Delta y} \leq \frac{1}{\sqrt{2}}.$$

2.1.2 Open Boundary Conditions

To obtain the non-reflecting boundary conditions, we would need to impose the appropriate conditions at the boundaries similarly to the case of part A. From lectures, we know that the general solution to the 2D wave equation can be written as a superposition of waves $\exp(ilx + imy + i\omega t)$, where l, m, ω are related by $l^2 + m^2 = \omega^2$. From lectures, we know that a good approximation for the open boundary condition at the bottom boundary $y = -2$ is

$$u_{yt} = u_{tt} - \frac{1}{2}u_{xx}, \tag{29}$$

and due to the symmetrical nature of the solution, since we can take the negative root to obtain the second condition and then swap x and y in the equations since the problem is symmetrical, the other conditions are

$$u_{yt} = -u_{tt} + \frac{1}{2}u_{xx}, \quad \text{on top border } y = 2 \tag{30}$$

$$u_{xt} = u_{tt} - \frac{1}{2}u_{yy}, \quad \text{on left border } x = -2 \tag{31}$$

$$u_{xt} = -u_{tt} + \frac{1}{2}u_{yy}, \quad \text{on right border } x = 2. \tag{32}$$

These conditions are not perfect since they do not entirely stop all reflections. Another way of improving the open boundary condition would be implementing a perfectly matched layer outside of the computational domain where the waves would dissipate exponentially,

thus when they return to the computational domain they are exponentially small. For this coursework however, we will be exploring the previously mentioned conditions.

At the boundary, we cannot use centered finite differences in one of the spatial dimensions, hence we will need to use forward and backward finite differences for one spatial direction and centered finite differences for the other spatial direction and the time direction.

After computing the required finite difference for the mixed derivatives, the boundary conditions, for $i = 1, \ldots, N, j = 1, \ldots, N$, are

Left Boundary:

$$U_{1,j}^{k+1} = \frac{r}{2+r}\left(\frac{2}{r}(2U_{1,j}^{k} - U_{1,j}^{k-1}) + r(U_{1,j+1}^{k} - 2U_{1,j}^{k} + U_{1,j-1}^{k}) + U_{1,j}^{k-1} - U_{2,j}^{k-1} + U_{2,j}^{k+1}\right) \quad (33)$$

Top Boundary:

$$U_{i,N}^{k+1} = \frac{r}{2+r}\left(\frac{2}{r}(2U_{i,N}^{k} - U_{i,N}^{k-1}) + r(U_{i+1,N}^{k} - 2U_{i,N}^{k} + U_{i-1,N}^{k}) + U_{i,N}^{k-1} - U_{i,N-1}^{k-1} + U_{i,N-1}^{k+1}\right) \quad (34)$$

Right Boundary:

$$U_{N,j}^{k+1} = \frac{r}{2+r}\left(\frac{2}{r}(2U_{N,j}^{k} - U_{N,j}^{k-1}) + r(U_{N,j+1}^{k} - 2U_{N,j}^{k} + U_{N,j-1}^{k}) + U_{N,j}^{k-1} - U_{N-1,j}^{k-1} + U_{N-1,j}^{k+1}\right) \quad (35)$$

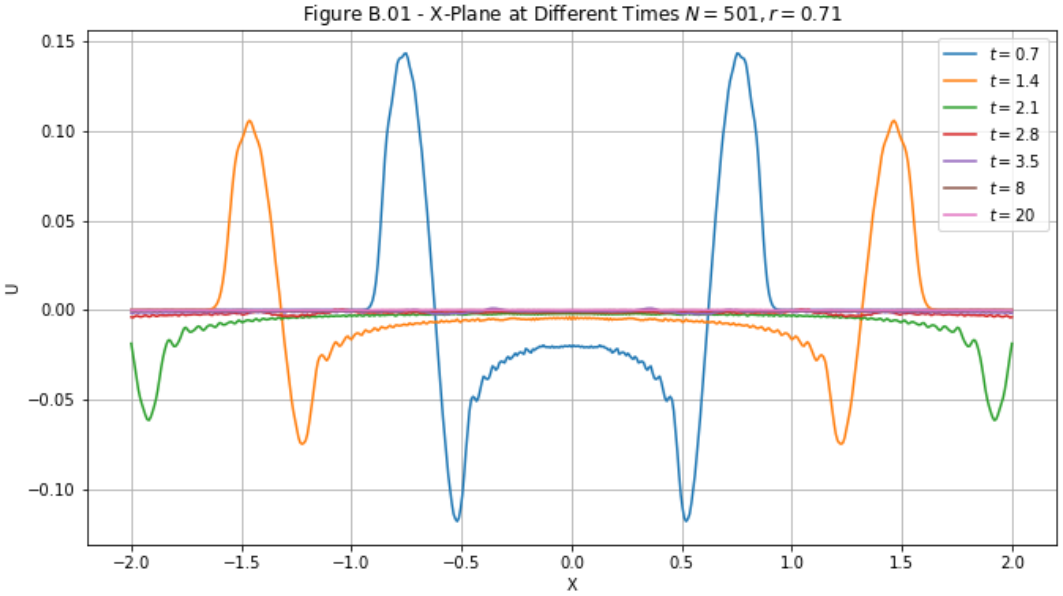
Bottom Boundary:

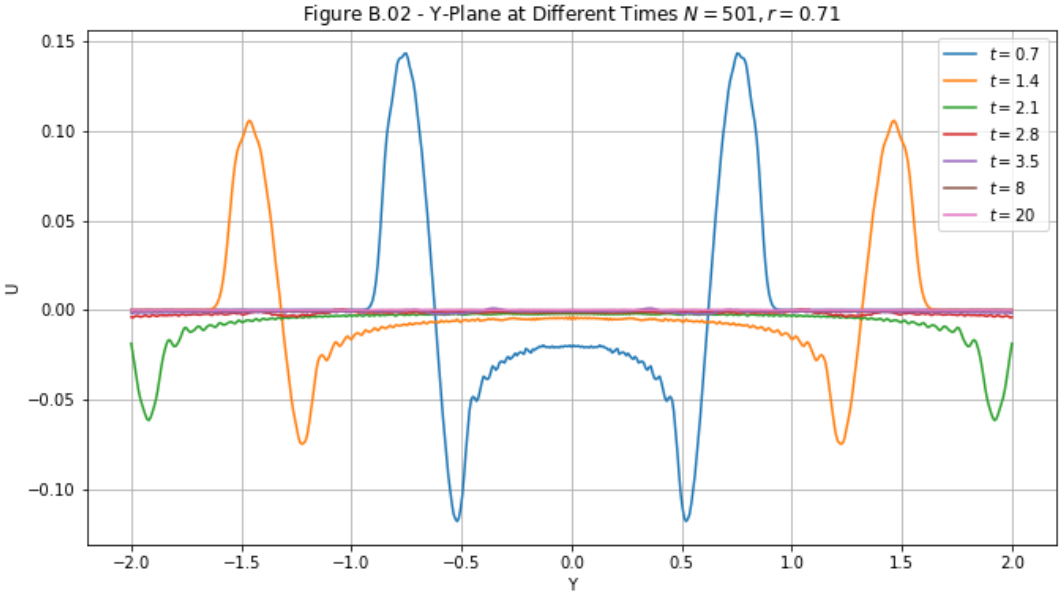
$$U_{i,1}^{k+1} = \frac{r}{2+r}\left(\frac{2}{r}(2U_{i,1}^{k} - U_{i,1}^{k-1}) + r(U_{i+1,1}^{k} - 2U_{i,1}^{k} + U_{i-1,1}^{k}) + U_{i,1}^{k-1} - U_{i,2}^{k-1} + U_{i,2}^{k+1}\right) \quad (36)$$

We can use these four equation to impose the open boundary conditions. At the corners, we will need to use forward and backwards finite differences in the other spatial direction as well. This simply involves changing the terms $U_{i,j+1}^{k} - 2U_{ij}^{k} + U_{i,j-1}^{k}$ and $U_{i+1,j}^{k} - 2U_{ij}^{k} + U_{i-1,j}^{k}$ in the equations above to the respective forwards/backwards differences, i.e. $2U_{i,j}^{k} - 5U_{i,j+1}^{k} + 4U_{i,j+2}^{k} - U_{i,j+3}^{k}$ and $2U_{i,j}^{k} - 5U_{i+1,j}^{k} + 4U_{i+2,j}^{k} - U_{i+3,j}^{k}$ for forward differences, $2U_{i,j}^{k} - 5U_{i,j-1}^{k} + 4U_{i,j-2}^{k} - U_{i,j-3}^{k}$ and $2U_{i,j}^{k} - 5U_{i-1,j}^{k} + 4U_{i-2,j}^{k} - U_{i-3,j}^{k}$ for backward differences.

It is worth noting that these boundary conditions require solutions at the current time-step $U_{i,2}^{k+1}, U_{i,N-1}^{k+1}, U_{2,j}^{k+1}, U_{N-1,j}^{k+1}, i = 1, \ldots, N, j = 1, \ldots, N$. Since these points lie *inside* the domain, i.e. do not lie on the boundaries, we can solve our system first on the inside of our domain and then compute the boundary conditions using the values just computed. This does mean that our boundary conditions have become implicit in some sense, however, this is the best way of keeping 2nd order accuracy in the temporal finite difference discretisations.

2.1.3 Plots of X and Y Axes





Figures B.01 and B.02 show the evolution of the solution with time along both axes as required. Now, since the solution is symmetrical for the case of $q = 1$, we can see that both plots are virtually identical up to round off error. As we can see, there are some dispersion effects happening, similarly to part A.

2.1.4 Grid Independence

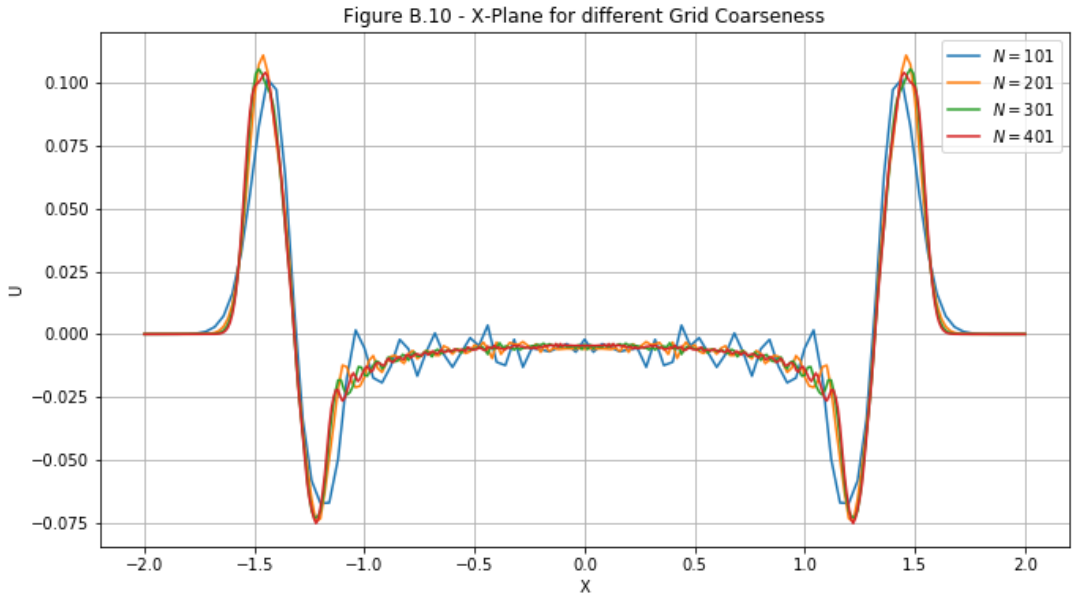
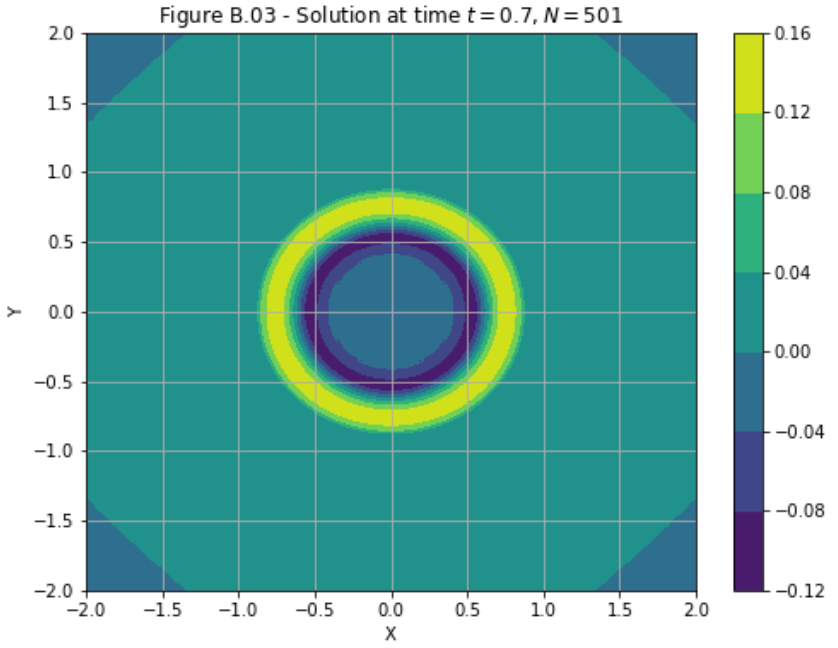
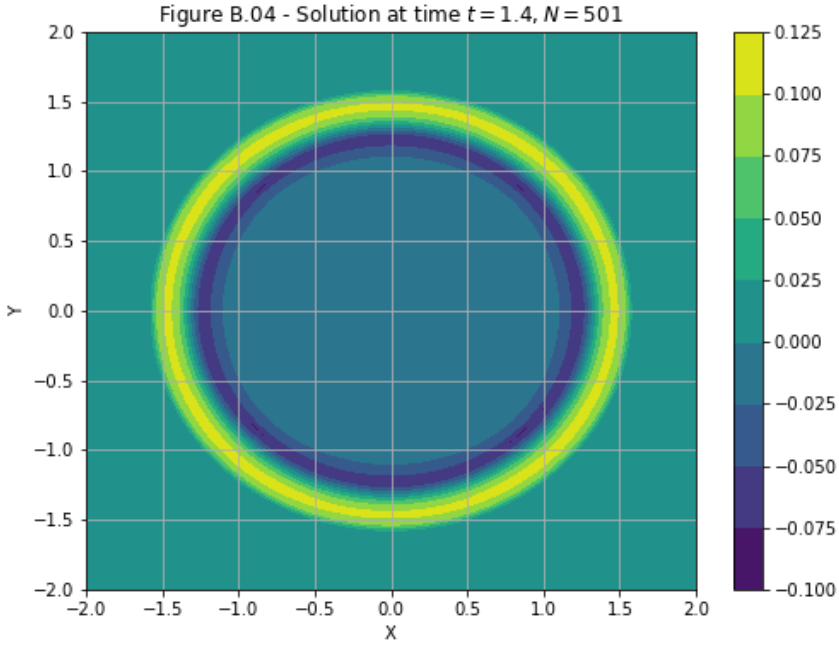


Figure B.10 shows that the solution is grid independent, since we arrive at the same solution independent of coarseness of the grid, apart from increased dispersion effects for lower N. These dispersion effects are expected for the exact same reasons as in the 1D case, i.e. the lower N is, the more severely the solution will be impacted by dispersion. For large time, we see that the waves have passed through our domain, and all that remains are small reflections that are due to the imperfect nature of the boundary conditions.

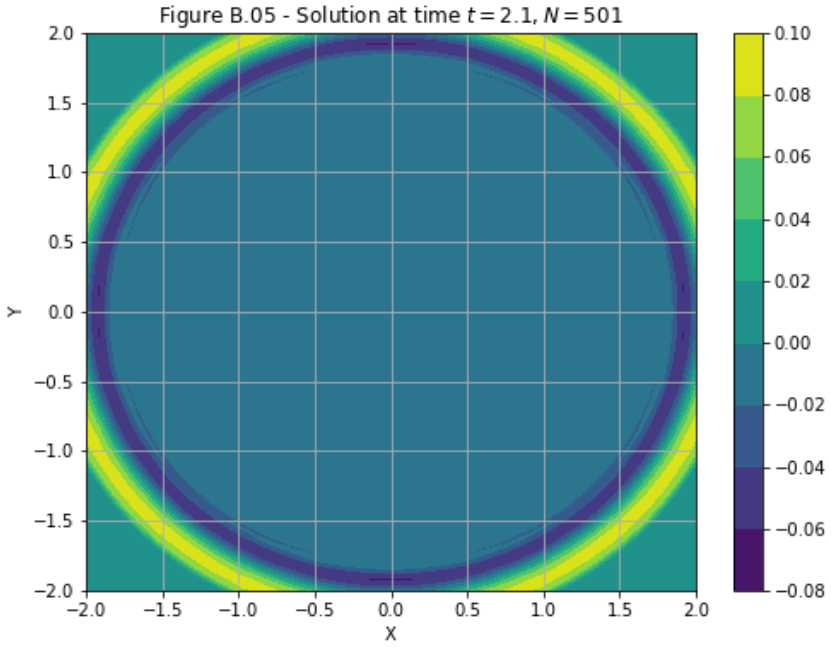
2.1.5 Contour Plots of the Entire Solution

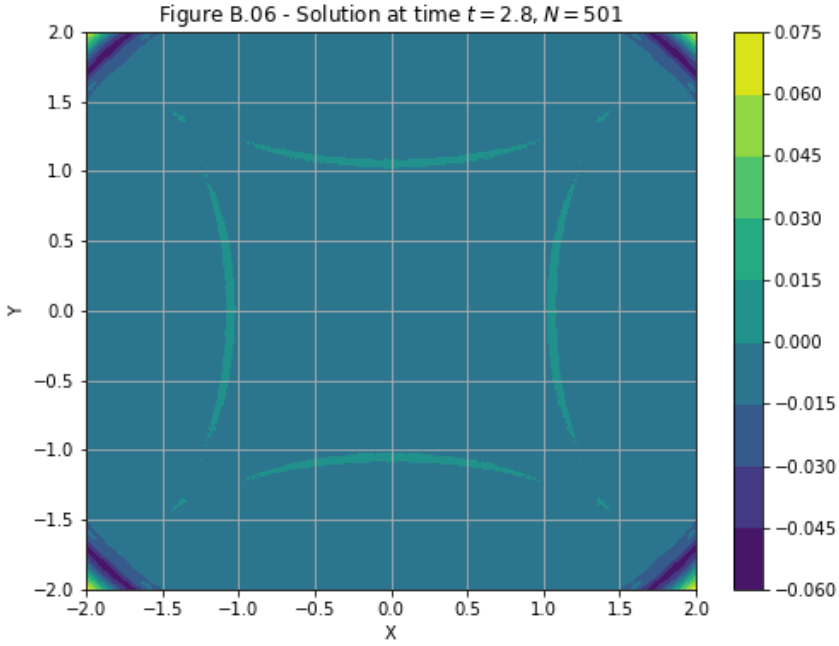
Countour plots give a good visual representation of how our scheme behaves across the whole computational domain. We can use them to judge whether the solution makes sense as well as seeing whether the boundary conditions behave as we would expect.





Figures B.03 and B.04 show how the solution evolves before it hits the boundary. The solution is symmetrical in both x and y directions and evolves as we would expect.





Figures B.05, B.06 show what happens to the wave as it hits the boundary. As we can see, the wave passes through the boundaries with minimal reflections as seen in figure B.06. We can expect these minimal reflections since the boundary conditions are not perfect. However, their magnitude is close to zero, indicating the boundary conditions are a pretty good approximation to open boundaries.

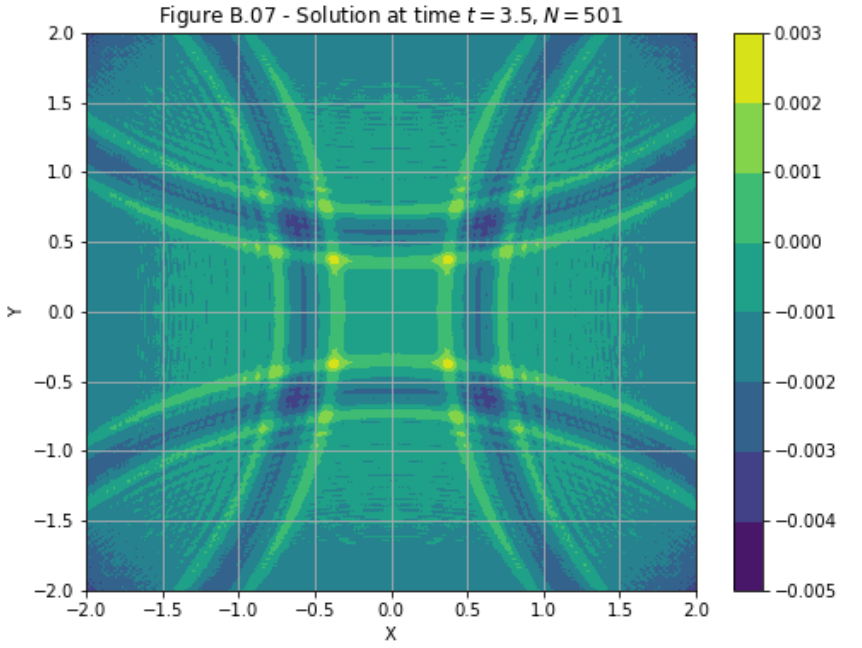


Figure B.07 - Solution at time $t = 3.5, N = 501$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
Y
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
X
0.003
0.002
0.001
0.000
−0.001
−0.002
−0.003
−0.004
−0.005

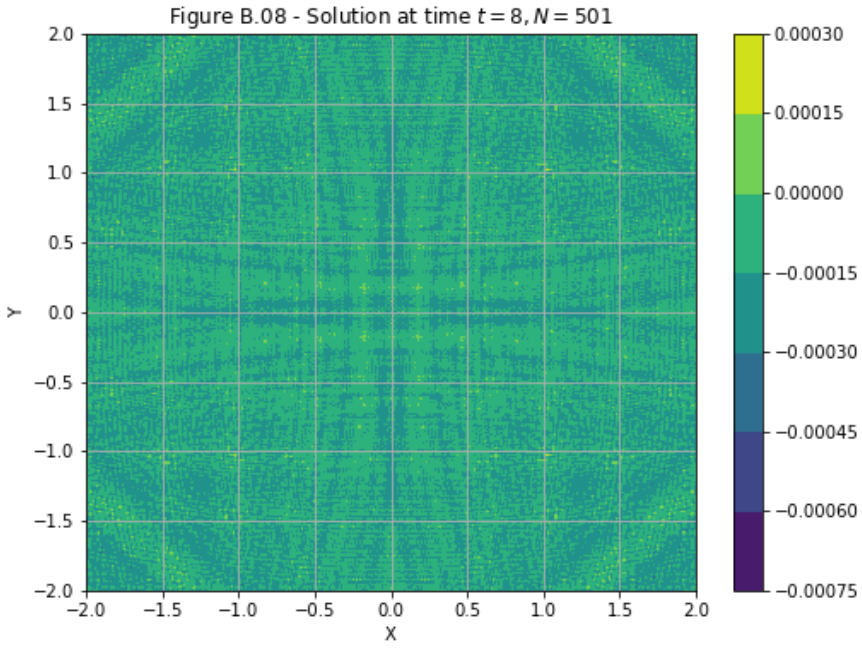


Figure B.08 - Solution at time $t = 8, N = 501$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
Y
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
X
0.00030
0.00015
0.00000
−0.00015
−0.00030
−0.00045
−0.00060
−0.00075

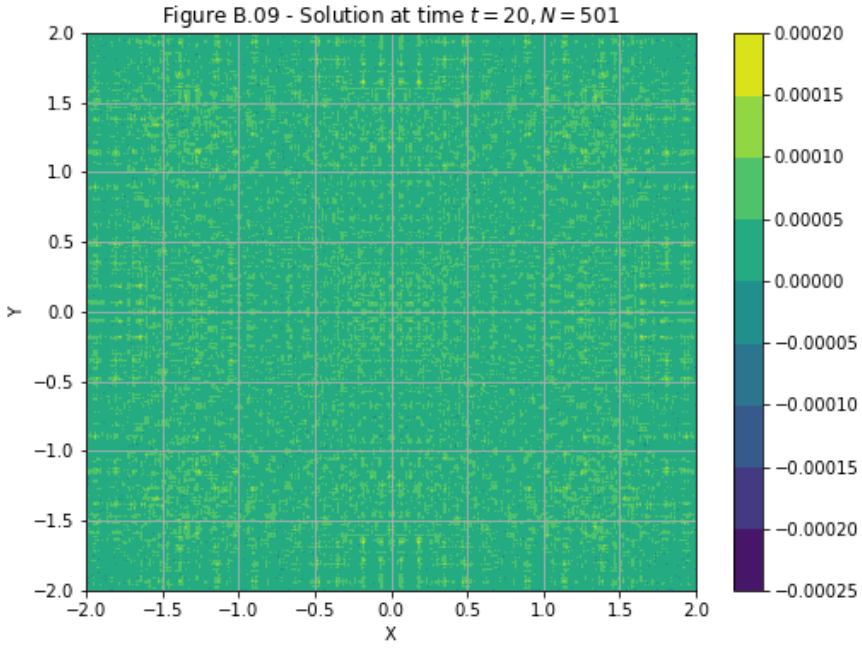


Figure B.07 captures the first reflections generated by our approximations to the open boundary conditions. By first reflections, we mean the period of time between the moment the main wave has convected through our boundaries and the moment that the first set of (unwanted) reflections reaches the boundaries again. The scale on the right of the plots shows that the reflections are in the order of 10^{-3}, i.e. small. In the ideal case, we would like the domain to be equal to zero throughout.

Figures B.08 and B.09 captures the behaviour of our domain for much larger times $t = 8, t = 20$. As we can see, the reflections become smaller and smaller every time they hit the boundaries, since most of their magnitude convects through. After large enough time, they will eventually diffuse throughout the whole domain with decreasing amplitude.

2.2 Question 2

2.2.1 Modified Equations

By having $q \neq 1$, we affect the governing dynamics of the PDE greatly. Our wave equation discretisation needs to be modified to the following

$$U_{ij}^{k+1} = 2(1 - r^2(1+q))U_{ij}^k + r^2(U_{i+1j}^k + U_{i-1j}^k + q(U_{ij+1}^k + U_{ij-1}^k)) - U_{ij}^{k-1}. \tag{37}$$

The parameter q affects the wave propagation in the y-direction. For large q, the wave speed will increase in the y-direction, whereas for small q, the wave speed will be smaller in the y-direction. This will mean that our wave will behave propagate in an elliptical way instead of circularly as before.

2.2.2 Boundary Conditions

In lectures, we found that for the case $q = 1$, we can take Fourier transforms to show that the general solution to the wave equation can be written as a superposition of waves $\exp(ilx + imy + i\omega t)$ and we can relate l, m, ω using $l^2 + m^2 = \omega^2$. For $q \neq 1$, we can obtain the analogue condition that the equation can be written as a superposition of waves $\exp(ilx + imy + i\omega t)$ and we can relate l, m, ω using $l^2 + qm^2 = \omega^2$.

So, similarly to before, we can obtain the modified approximations to the open boundary conditions, when $q \neq 0$ by

$$\sqrt{q}u_{yt} = u_{tt} - \frac{1}{2}u_{xx}, \tag{38}$$

and due to the symmetrical nature of the solution as previously seen, the other conditions are

$$\sqrt{q}u_{yt} = -u_{tt} + \frac{1}{2}u_{xx}, \quad \text{on top border } y = 2 \tag{39}$$

$$u_{xt} = u_{tt} - \frac{q}{2}u_{yy}, \quad \text{on left border } x = -2 \tag{40}$$

$$u_{xt} = -u_{tt} + \frac{q}{2}u_{yy}, \quad \text{on right border } x = 2. \tag{41}$$

Similarly to before, we can discretise these using 2nd order finite differences.

Left Boundary:

$$U_{1,j}^{k+1} = \frac{r}{2+r}\left(\frac{2}{r}(2U_{1,j}^k - U_{1,j}^{k-1}) + qr(U_{1,j+1}^k - 2U_{1,j}^k + U_{1,j-1}^k) + U_{1,j}^{k-1} - U_{2,j}^{k-1} + U_{2,j}^{k+1}\right) \tag{42}$$

Top Boundary:

$$U_{i,N}^{k+1} = \frac{r}{2/\sqrt{q}+r}\left(\frac{2}{r\sqrt{q}}(2U_{i,N}^k - U_{i,N}^{k-1}) + \frac{r}{\sqrt{q}}(U_{i+1,N}^k - 2U_{i,N}^k + U_{i-1,N}^k) + U_{i,N}^{k-1} - U_{i,N-1}^{k-1} + U_{i,N-1}^{k+1}\right) \tag{43}$$

Right Boundary:

$$U_{N,j}^{k+1} = \frac{r}{2+r}\left(\frac{2}{r}(2U_{N,j}^{k} - U_{N,j}^{k-1}) + qr(U_{N,j+1}^{k} - 2U_{N,j}^{k} + U_{N,j-1}^{k}) + U_{N,j}^{k-1} - U_{N-1,j}^{k-1} + U_{N-1,j}^{k+1}\right) \tag{44}$$

Bottom Boundary:

$$U_{i,1}^{k+1} = \frac{r}{2/\sqrt{q}+r}\left(\frac{2}{r\sqrt{q}}(2U_{i,1}^{k} - U_{i,1}^{k-1}) + \frac{r}{\sqrt{q}}(U_{i+1,1}^{k} - 2U_{i,1}^{k} + U_{i-1,1}^{k}) + U_{i,1}^{k-1} - U_{i,2}^{k-1} + U_{i,2}^{k+1}\right) \tag{45}$$

2.2.3 Contour Plots of Implementation

The following contour plots illustrate the behaviour of the solution for the different values of q specified in the question $q = (0.2, 0.5, 2)$

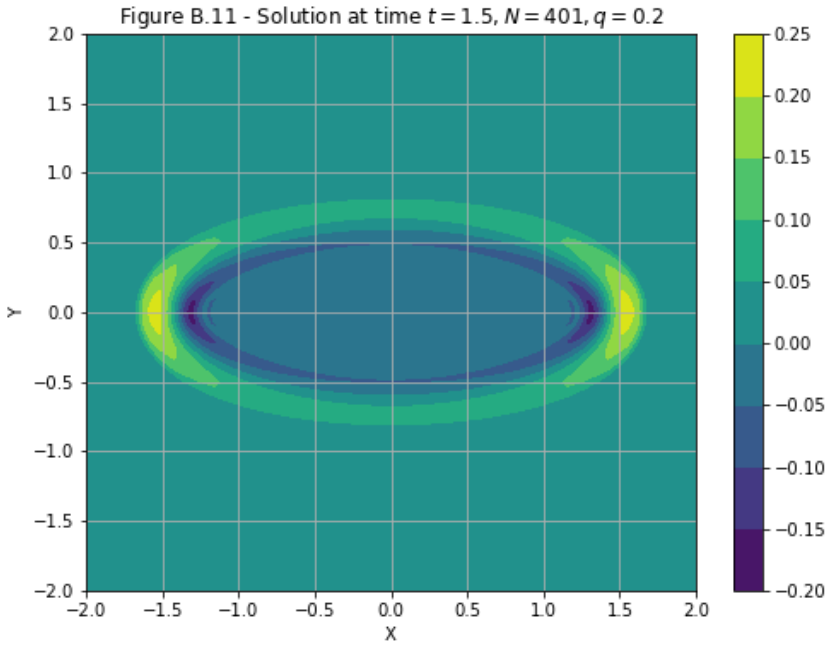


Figure B.12 - Solution at time $t = 3.5, N = 401, q = 0.2$

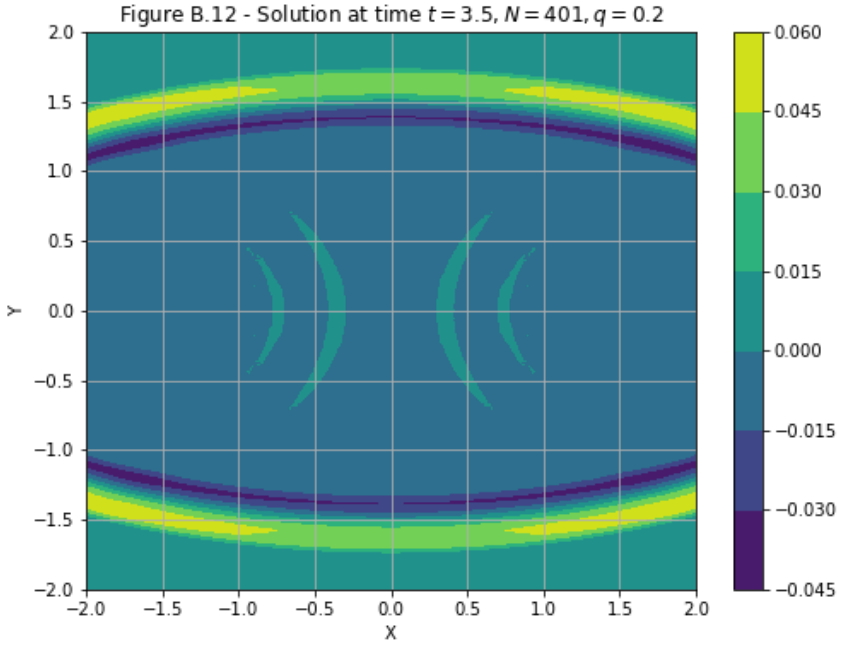


Figure B.13 - Solution at time $t = 5, N = 401, q = 0.2$

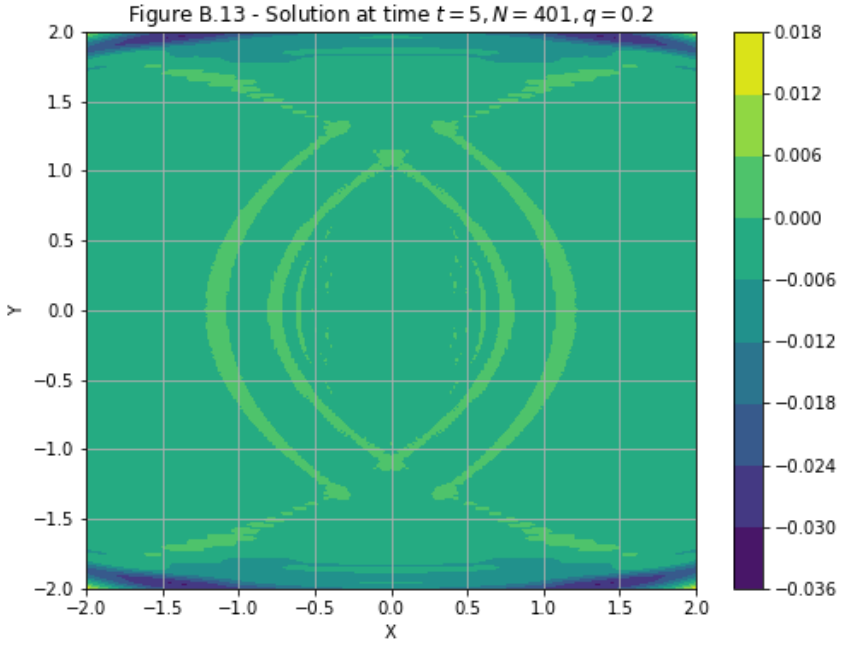


Figure B.14 - Solution at time $t = 8, N = 401, q = 0.2$

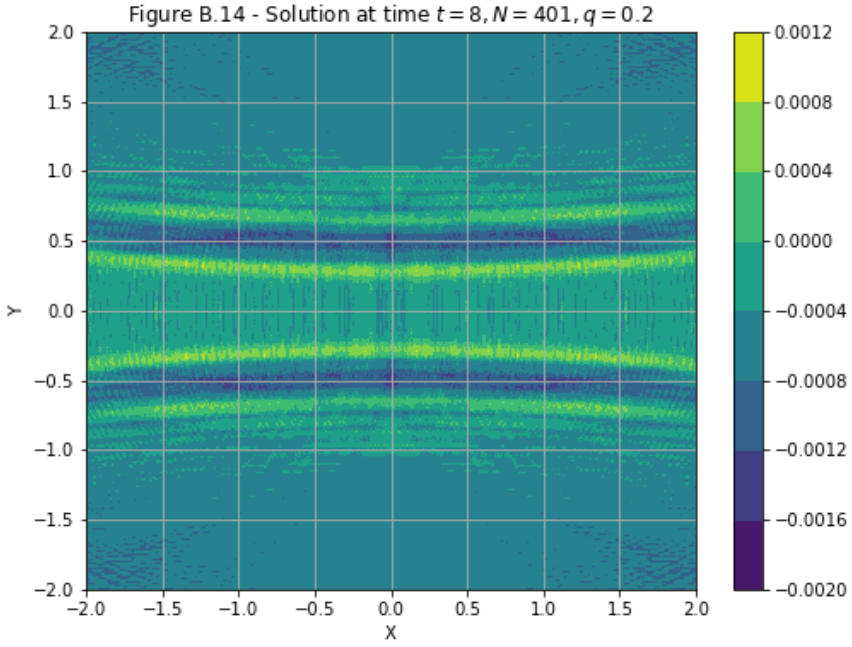


Figure B.15 - Solution at time $t = 1.5, N = 401, q = 0.5$

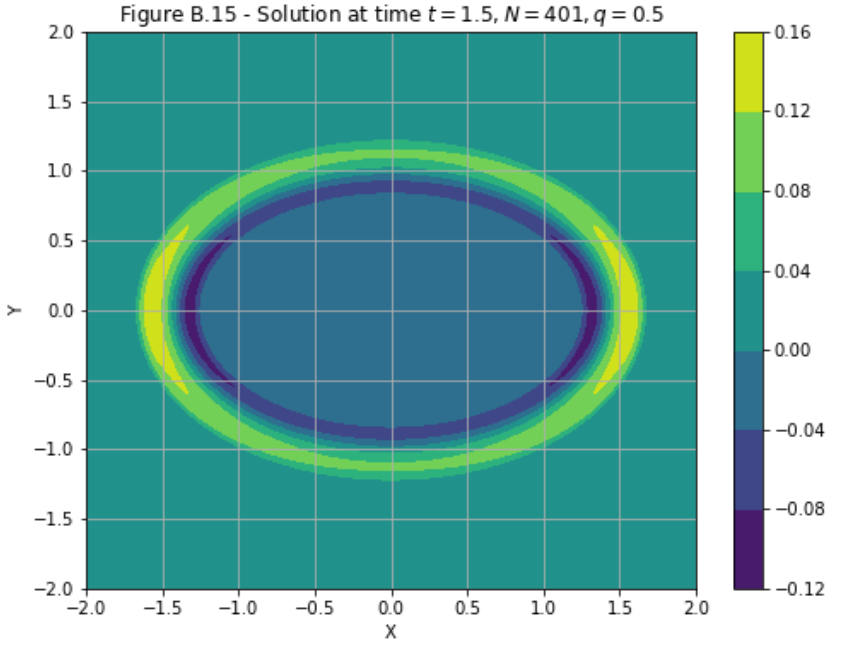


Figure B.16 - Solution at time $t = 3.5, N = 401, q = 0.5$

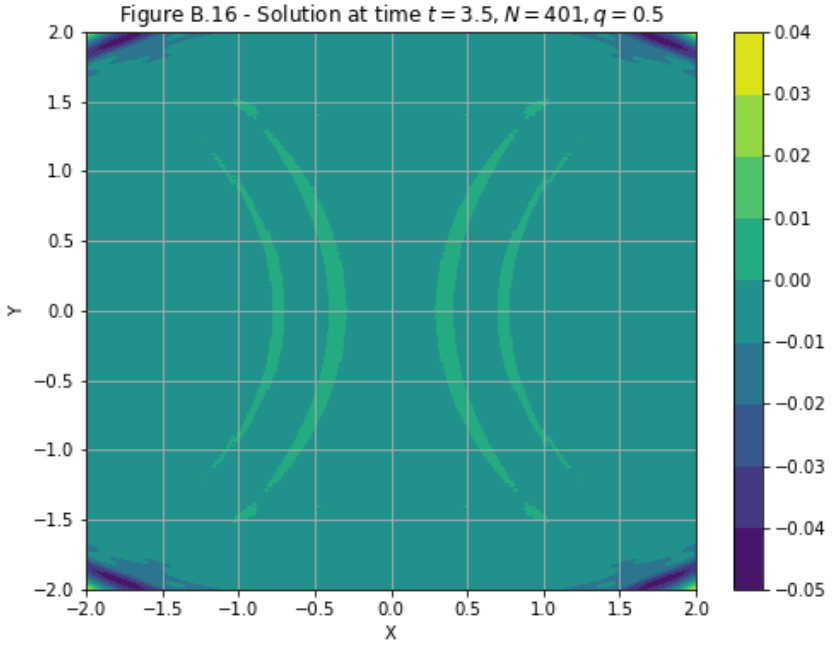


Figure B.17 - Solution at time $t = 5, N = 401, q = 0.5$

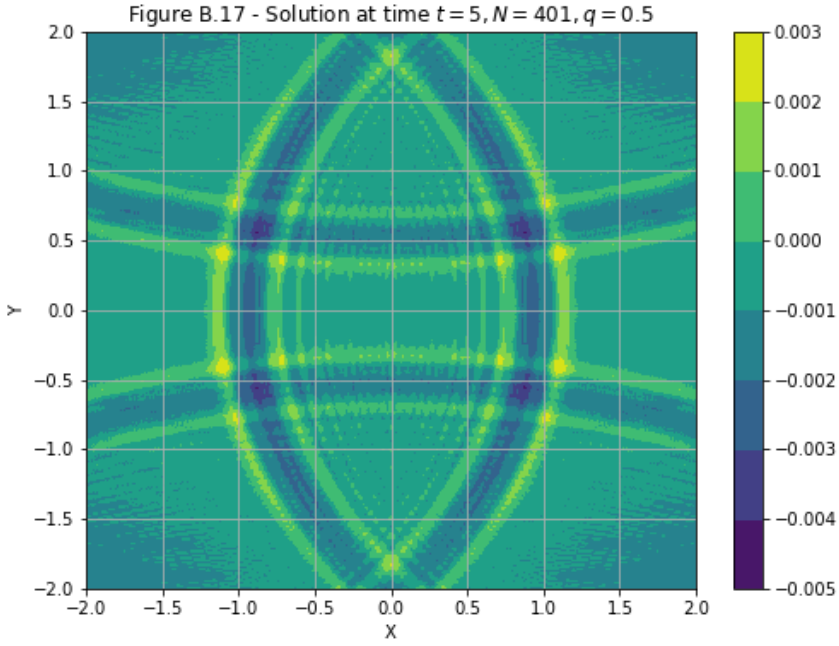


Figure B.18 - Solution at time $t = 1.2$, $N = 401$, $q = 2$

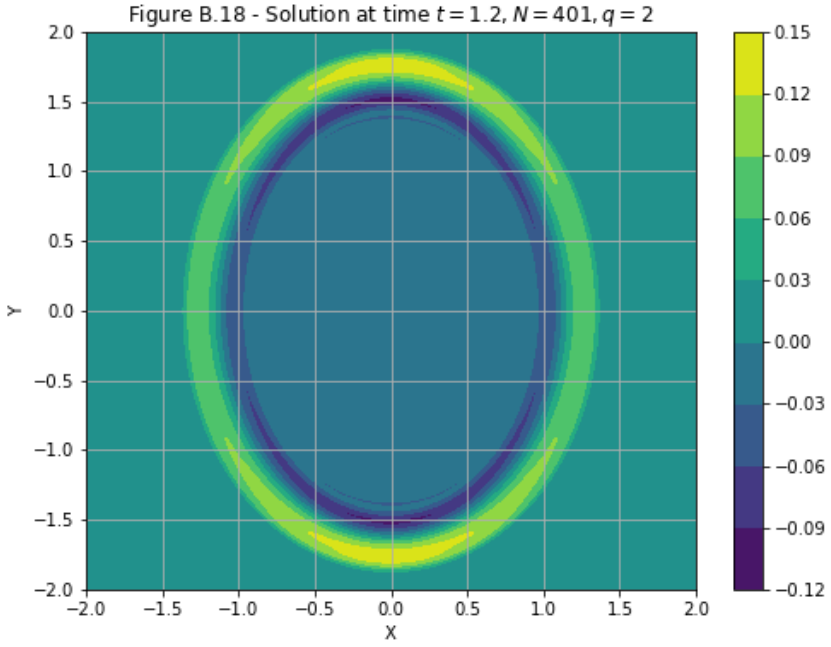
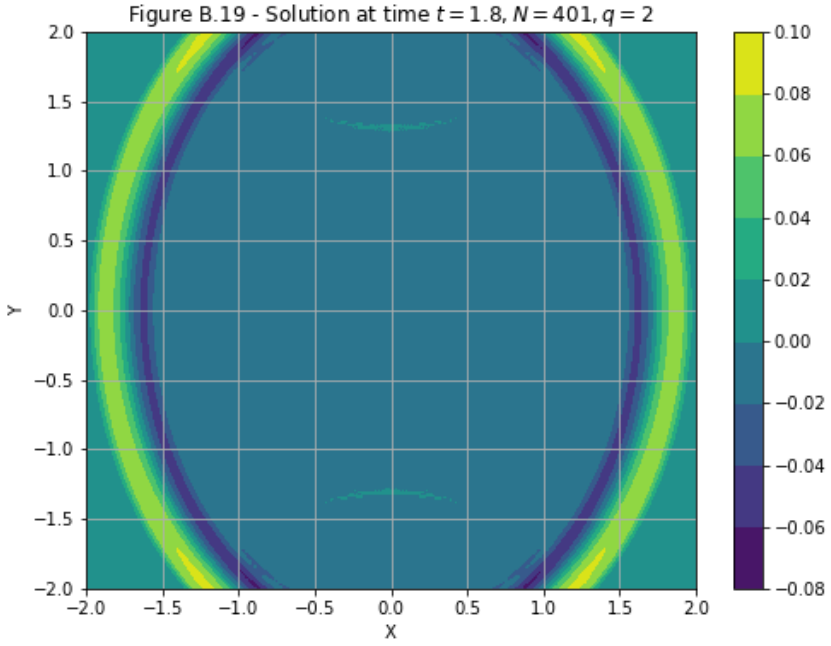
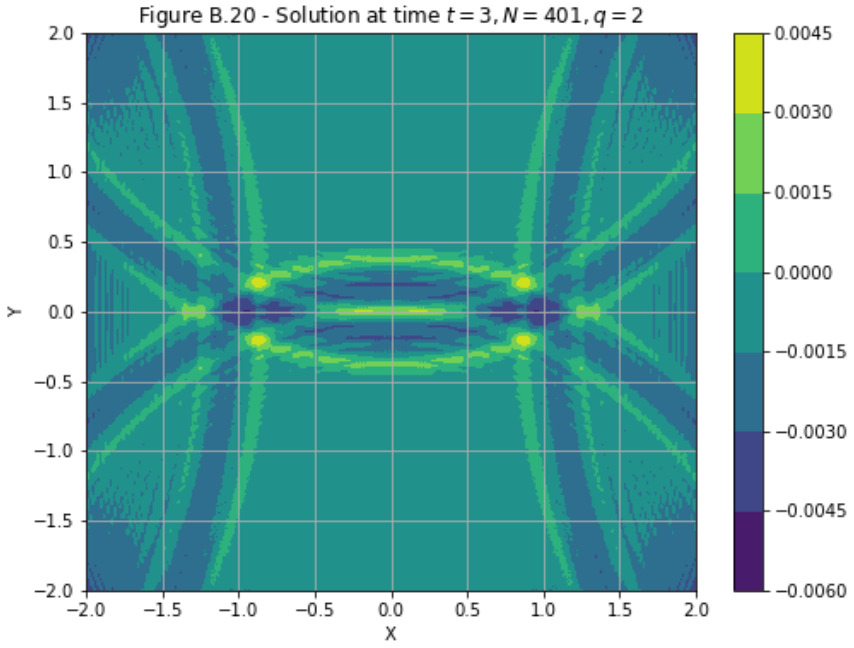


Figure B.19 - Solution at time $t = 1.8$, $N = 401$, $q = 2$





From the previous contour plots, we can observe that the parameter q is proportional to the speed of the wave in the y-direction, creating elliptic shapes when $q \neq 1$. The additional numerical issues that arise were due to modifying the wave equation discretisation as well as the boundary conditions, since the nature of the PDE is changed when $q \neq 1$.

3 References

1. Course 18.086: Mathematical Methods for Engineers II (Spring 2006): https://math.mit.edu/classes/18.086/2006/
2. H. P. Langtangen and S. Linge: Finite Difference Computing with PDEs - A Modern Software Approach
3. M4N9 Course Lecture Notes