Report 4 --- Advection-Diffusion Problem in 2D

Name: Lam Ling Ho Adrian
UID: 105211252

I. Recursive Equations:

In this problem, a simple operator splitting scheme is used with leapfrog for advection and Euler forward (FTCS) with a timestep size of 2Δt for diffusion.

The ***operator splitting*** scheme is carried out as follows:

$$\textit{Operator Splitting}:$$
$$u_{new} = u_{old} + advection$$
$$u_{new} = u_{new} + diffusion$$

Advection:

The recursive equation for Leapfrog in a 2-Dimensional domain is as follows:

$$\textit{Leap Frog Scheme Recursive Equation}$$
$$u_{i,j}^{n+1} = u_{i,j}^{n-1} - \frac{c_x \Delta t}{\Delta x}\left(u_{i+1,j}^{n-1} - u_{i-1,j}^{n-1}\right) - \frac{c_y \Delta t}{\Delta y}\left(u_{i,j+1}^{n-1} - u_{i,j-1}^{n-1}\right)$$

For time step = 0, Leap Frog requires information at time step = -1. Thus, a different time advance scheme is needed for advancing the first time step. In this problem, Forward-In-Time-Backward-In-Space (FTBS) scheme is used for this purpose. The FTBS recursive equation in 2-D domain is as follows:

$$\textit{Forward in Time, Backwards in Space Scheme Recursive Equation}$$
$$u_{i,j}^{n+1} = u_{i,j}^{n-1} - \frac{c_x \Delta t}{\Delta x}\left(u_{i,j}^{n-1} - u_{i-1,j}^{n-1}\right) - \frac{c_y \Delta t}{\Delta y}\left(u_{i,j}^{n-1} - u_{i,j-1}^{n-1}\right)$$

Diffusion:

The Euler Forward Scheme is used for modelling the diffusion component of our solution, advancing with a time step equal to 2Δt.

$$\textit{Euler Forward Scheme (Advance in 2 Time Steps) Recursive Equation}$$
$$u_{i,j}^{n+1} = u_{i,j}^{n-1} + \frac{K\Delta 2t}{\Delta x^2}\left(u_{i+1,j}^{n-1} - 2u_{i,j}^{n-1} + u_{i-1,j}^{n-1}\right) + \frac{K\Delta 2t}{\Delta y^2}\left(u_{i,j+1}^{n-1} - 2u_{i,j}^{n-1} + u_{i,j-1}^{n-1}\right)$$

One interesting result of this is that the diffusion solutions would have 2 set of co-evolving solutions that never interact with each other. (i.e. Solutions at time step N, N+2, N+4… will evolve independently from N+1, N+3, N+5 etc.)

II. Boundary Conditions:

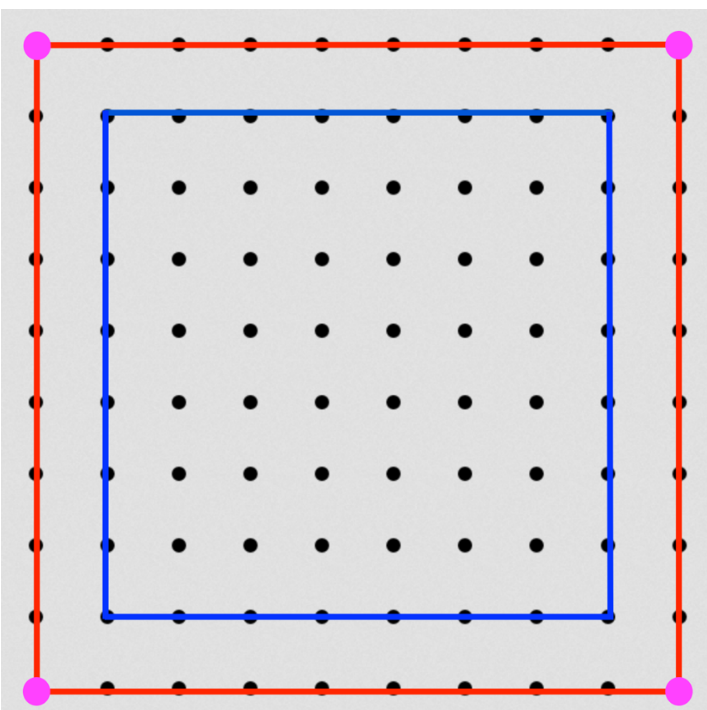
A. Periodic Boundary Conditions:

Periodic/Cyclic boundary conditions are implemented for this advection diffusion model, detailed as follows:

$$u(x=0,y,t) = u(x = L_x,y,t)$$
$$and$$
$$u(x,y=0,t) = u(x,y=L_y,t)$$

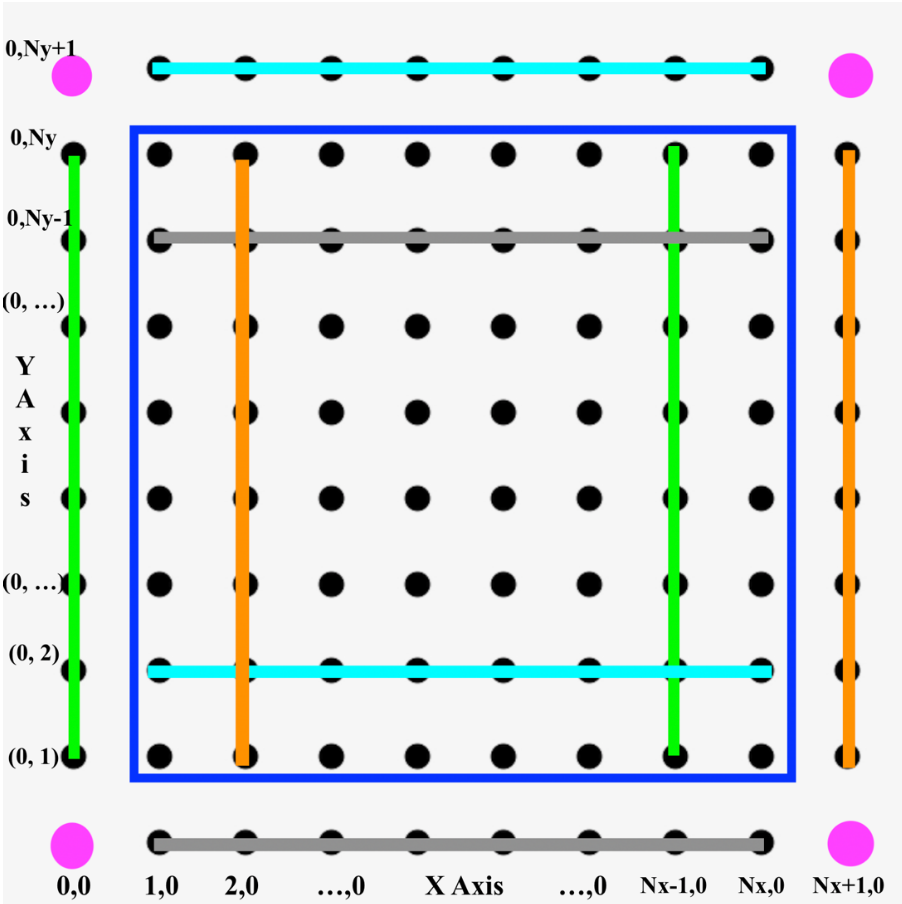
In essence, the upper and lower boundaries are set equal for each X and Y axis.

B. Ghost Nodes



The boundary conditions are enforced by implementing a layer of ghost nodes surrounding the entire physical domain. The right figure illustrates the idea of the ghost node: the blue square denotes the physical domain, containing all the physical nodes. The nodes under the red lines are the ghost nodes.

In essence, ghost nodes are virtual, non-physical nodes that allows physical boundary nodes (nodes under blue lines) to nonetheless be calculated according to the 5 point stencil diagram.



After updating all physical nodes in each time step, the ghost nodes are updated as such:

Each colored line of ghost nodes will copy values from the line of physical boundary node with the identical color. (e.g. the values of the physical nodes in the **lower light blue line** will be copied to the corresponding ghost nodes in the **upper light blue line**)

As in the 5-point stencil, the corner ghost nodes (pink dots in diagram) are actually never used nor updated. Thus, they are initialized with a value of -999 for error detection.

update horizontal boundaries

$$u^n_{(i,\,Nx+1)} = u^n_{(i,2)}$$

$$u^n_{(i,0)} = u^n_{(i,\,Nx-1)}$$

update vertical boundaries

$$u^n_{(Nx+1,\,j)} = u^n_{(2,\,j)}$$

$$u^n_{(0,\,j)} = u^n_{(Nx-1,\,j)}$$

At a particular time step n, the equations for updating ghost nodes are shown to the right. In the equations, each index ***i*** and ***j*** are looped from 1 to Nx.

Lastly, after all calculations in every time step, **the upper and lower physical boundary nodes are made equal for both X and Y axis.**

This is to remove the computer's rounding error at the physical boundaries by directly reinforcing the cyclic boundary conditions at every iteration.

III. Choice for Δt:

Courant Number and Von Neumann Number provides basic constraints on Δt for stability.

A. Considering Courant Number:

The stability conditions for Leap Frog Advection is as follows:

$$\begin{aligned}&\textit{CFL Condition for 2D}\\&\textit{For a wave travelling at diagonal}:\end{aligned}$$

$$\sqrt{C_x^2 + C_y^2} = C$$

$$\frac{C\Delta t}{\delta} \leqslant \frac{\sqrt{2}}{2}$$

$$\textit{Thus}: \ \Delta t \leqslant \frac{\sqrt{2}}{2} \ \textit{for stability}$$

$$\Delta x = \Delta y = \delta = \textit{grid spacing}$$

B. Considering Von Neuman Number:

$$K_x = K_y = K \ (\textit{isotropic diffusion})$$

$$\Delta x = \Delta y = \delta = \textit{grid spacing}$$

$$\frac{K\,2\Delta t}{\delta^2} \leqslant 0.25 \ (0.125 \ \textit{to prevent oscillating solutions})$$

$$\textit{Thus, for EF scheme that advances } 2\Delta t \textit{ per timestep}:$$

$$\Delta t \leqslant 0.625 \ \textit{for stable, non-oscillating solutions}$$

Supposedly, the smaller Δt from the above two calculations should be sufficient to yield a stable advection-diffusion solution. Interestingly, even for Δt = 0.5, advection and diffusion seem to be stable individually, but advection-diffusion solution is found to be unstable.

As a result, **Δt = 0.1** was chosen for the time step size for the numerical experiments.

IV. Numerical Experiment A – Advection Only:

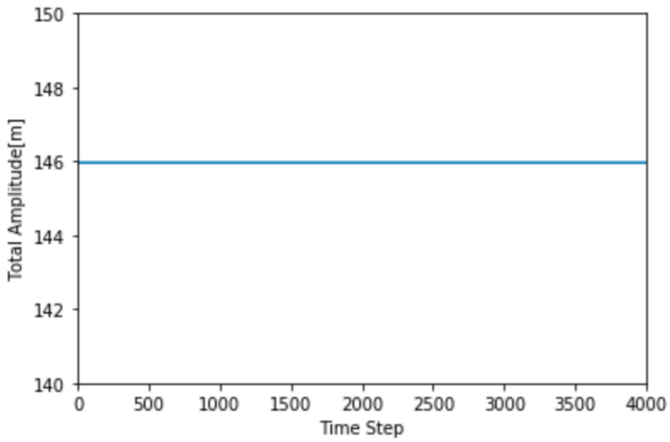
i. Basic Configuration:

<u>*Δt = 0.1, r = 12.5 (initial radius of disturbance), cx = cy = 1 (x,y velocities). Total Integration Time = 400s*</u>

ii. Total Amplitude Integral:

$$U(t) = \int_0^{Lx}\int_0^{Ly} u(x,y,t)\,dx\,dy = const.$$

<u>Total Amplitude-Time Step Graph</u>

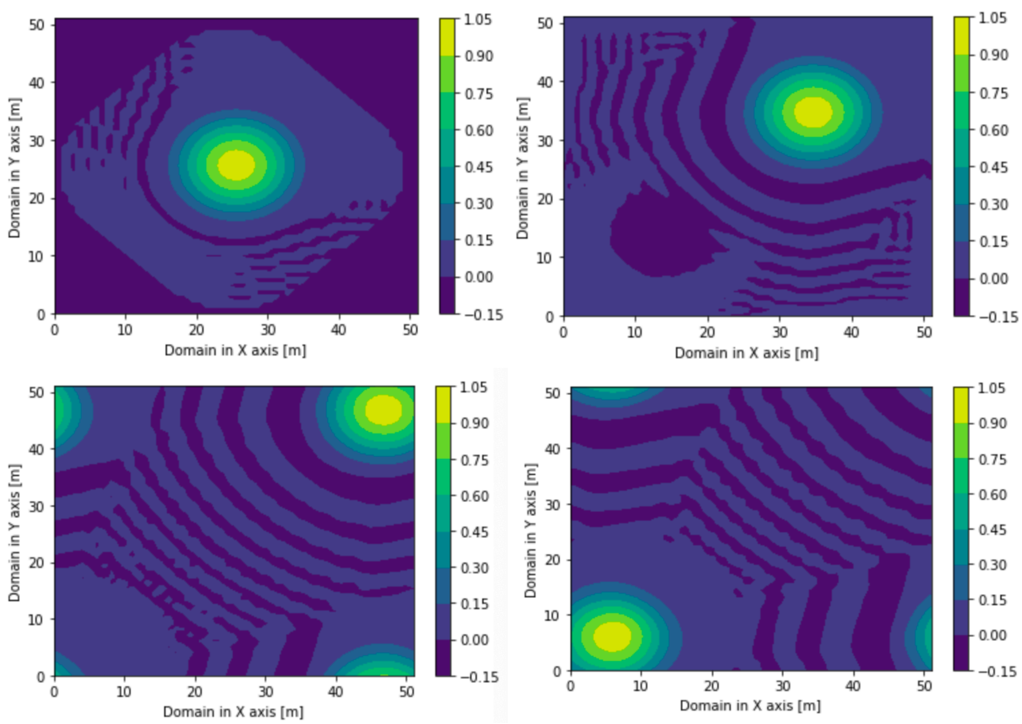


The total amount of amplitude within the 2D domain should be conserved over the entire integration time. (i.e. no net amplitude leakage/influx within the domain) The exact value, however, should vary with different Δx/Δy grid sizes.

From the graph, we could see that the total amplitude is clearly conserved over time.

iii. Advection in Action:

<u>Contour Plot at T = 10s (Top left), 100s (Top Right), 200s (Bottom Left), 300s (Bottom right)</u>

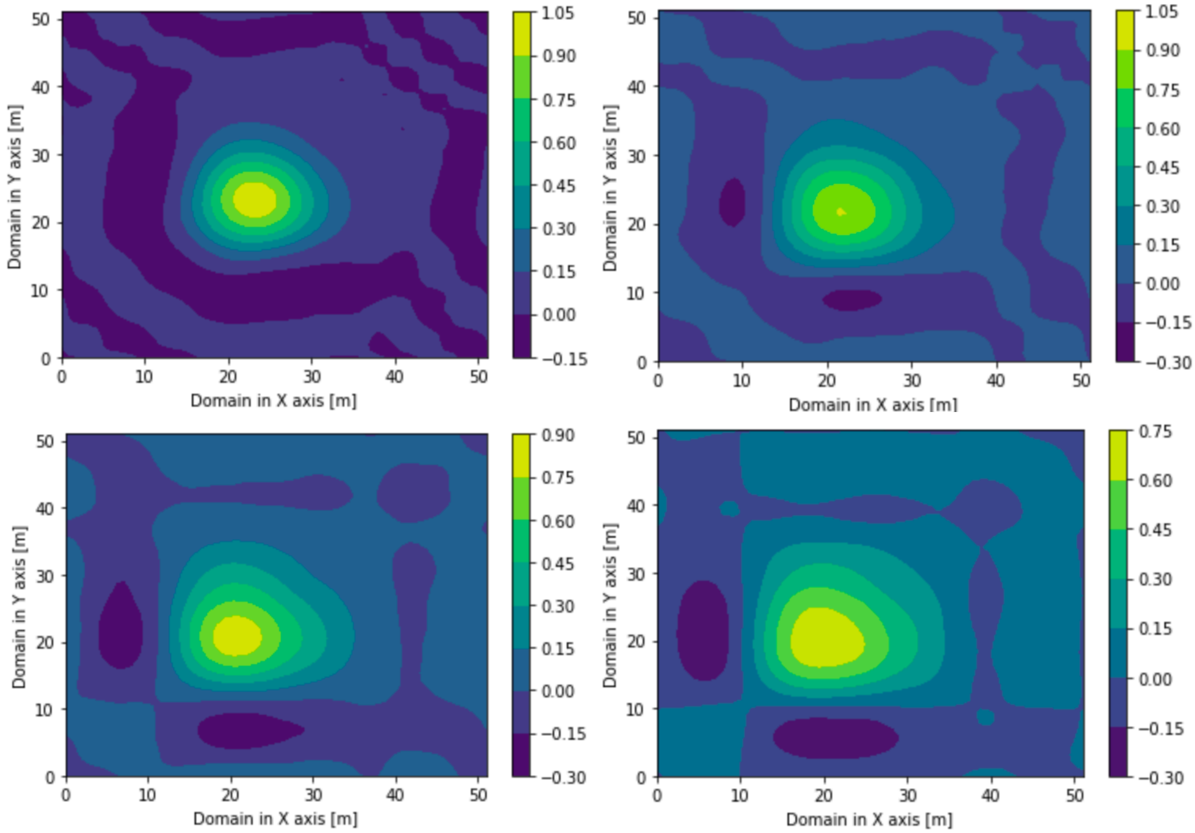


The goal of the contour plot is to visualize the wave advection process.

An interesting point to notice in the plots is the ripple-like patterns that propagated in the opposite direction of the main wave. The patterns show a +/- oscillation, hinting that it should be the computational mode. As time progresses, we see the computational modes pass through the boundaries and interact, forming interfere patterns.

iv. Phase Error:

Contour Plot at T = 100s (Top left), 200s (Top Right), 300s (Bottom Left), 400s (Bottom right)



The Leapfrog scheme is known to cause phase errors in the advection solutions, thus, we expect to see the numerical solution propagating at the wrong velocity (usually slower than the analytical solution). In our 2D advection only model, 4 contour plots are taken at 100s intervals, during which the waves are supposed to traverse the entire domain 2 times and return to the same initial starting position (25,25).

From the above plots we can see several interesting features:

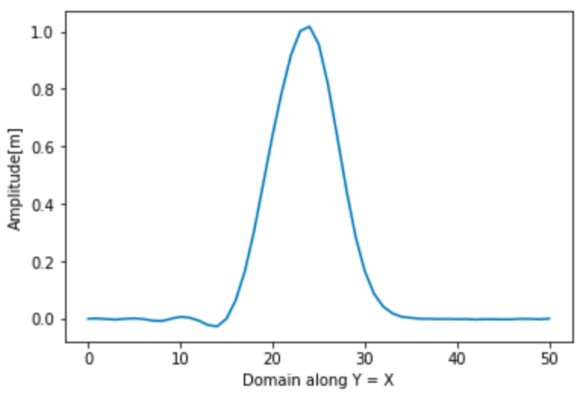
1. As integration time increases, the peak disturbance wave tends to deviate from the center towards bottom left, which shows that it is indeed lagging behind the analytical solution.
2. The shape of the disturbance is deformed as time progressed. This is due to the 2D nature of the wave. As it is travelling along the Y = X diagonal, the gaussian disturbance actually is composed of different wave number of waves travelling in parallel. Leapfrog's dispersive nature would cause different phase errors for different wavenumbers. Thus, some waves in the disturbance would travel faster than others, forming the asymmetric pattern at the later time steps.
3. The amplitude of the disturbance actually decreases over time, this will be addressed in detail under the "Challenge Question" section.

4. The minimum amplitude increases in amplitude with time. This is mostly associated with the computational mode and will be discussed in the following section.

   v. Computational Mode

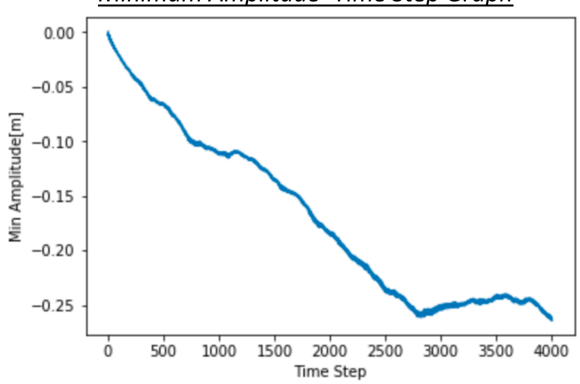
   The presence of computational mode is intrinsic in the leapfrog advection solutions. These computational mode waves should emerge from behind the disturbance and travel at the opposite direction, since computational mode has a phase opposite to the physical mode. The computational mode also changes sign every time step (2Δt period oscillations).

Amplitude-Domain Graph across Y = X line at T = 50s



In the graph to the right, one can see the computational mode at the left of the main wave peak. This illustrates the fact that the computational mode is the source of negative amplitudes found the numerical solutions.

Minimum Amplitude–Time Step Graph

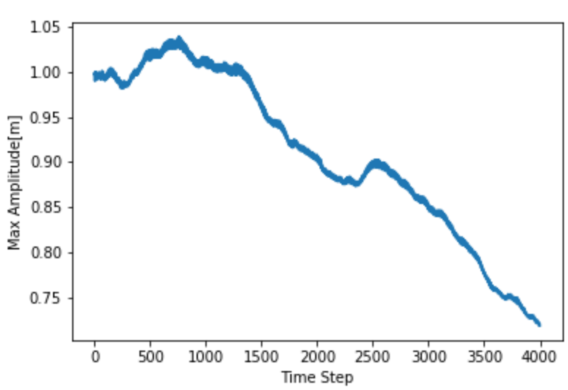


The graph to the left shows that the minimum amplitude actually decreases with time. In other words, the negative amplitude resulting from computational mode accumulates over the integration timespan.

Another interesting feature is that the "thick" texture of the graph's line, most likely due to the 2Δt oscillations of computational mode.

Challenge Question:

Maximum Amplitude–Time Step Graph



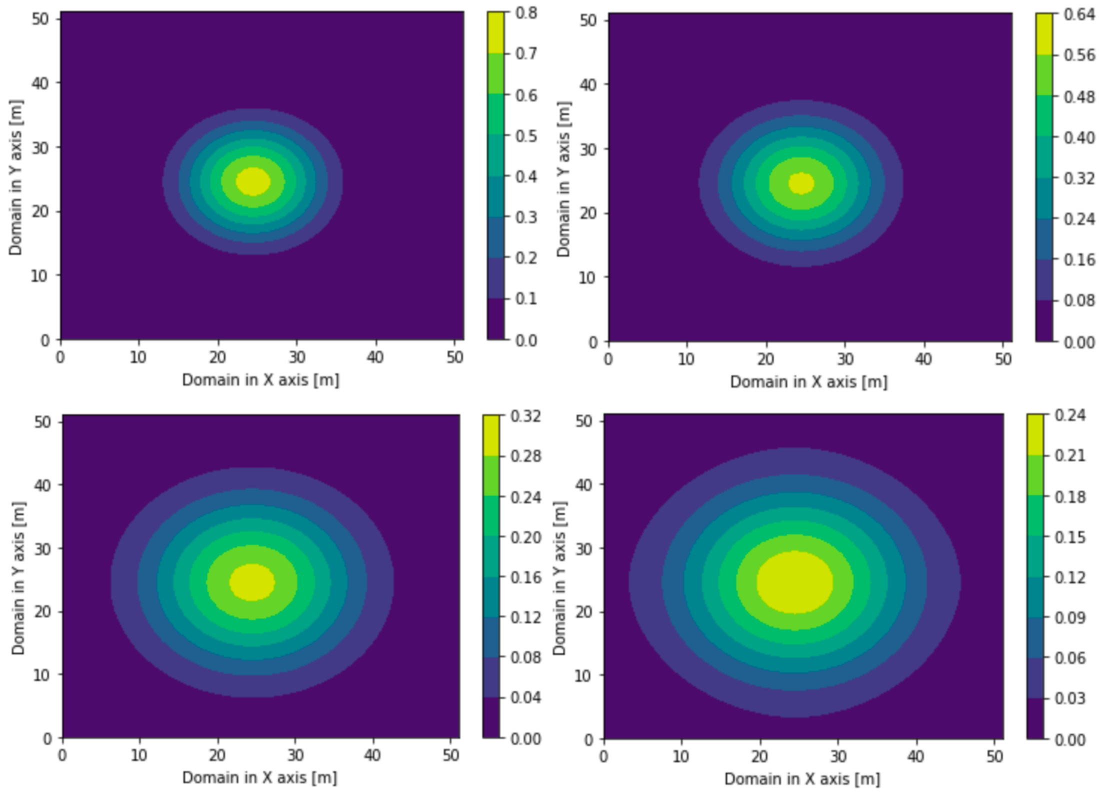
In principle, Leapfrog solutions should not have amplitude error as long as it is stable.
(i.e. if Courant Number ≤ 1, Amplification factor = 1)
However, we see a significant amplitude decay of the 2D disturbance over T = 400s.

One interesting observation is that, by comparing maximum and minimum amplitudes, it can be deduced that the "missing amplitude" actually is cancelled out by the negative minimum amplitude. In fact, the difference of maximum and minimum amplitudes tend to be roughly equal to 1, the analytical amplitude.
(e.g. @Time step ~4000,
Minimum = ~-0.25, Maximum = ~+0.75)

This discussion will be continued at the end of the report.

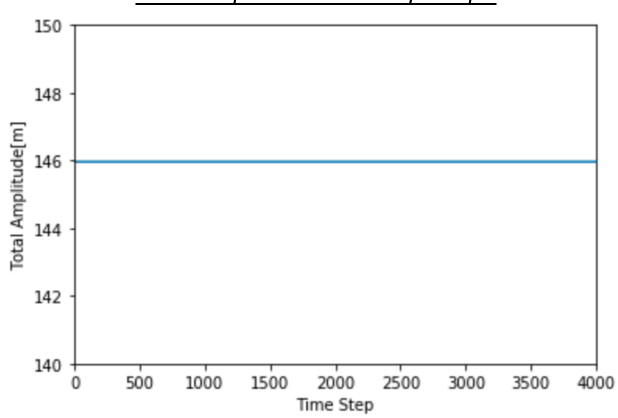
V. Numerical Experiment B, Diffusion Only (K = 0.1):
Basic Configurations: *<u>K = 0.1, cx = cy = 0 (x,y velocities), Δt = 0.1</u>*

<u>Contour Plot at T = 100s (Top left), 200s (Top Right), 300s (Bottom Left), 400s (Bottom right)</u>



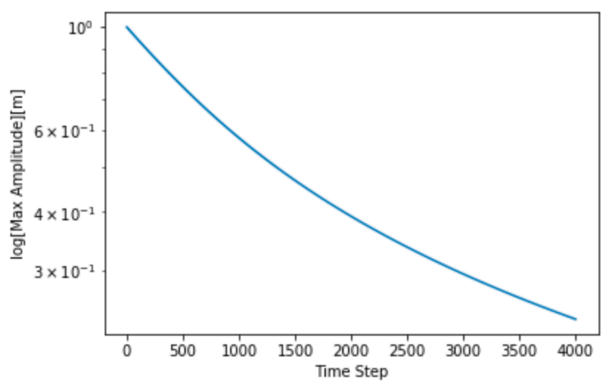
In the above plots, one can see the diffusion-only solution evolves with time. In particular, one can notice the wave's amplitude decreasing and spreading over a larger area, while remaining at its initial position (no advection). Comparing with the previous advection-only contour plots, a noticeable feature is the absence of computational mode, which is logical because Euler Forward shouldn't produce computational modes (since it only calculates new solutions based on solutions from one previous time step).

<u>Total Amplitude-Time Step Graph</u>

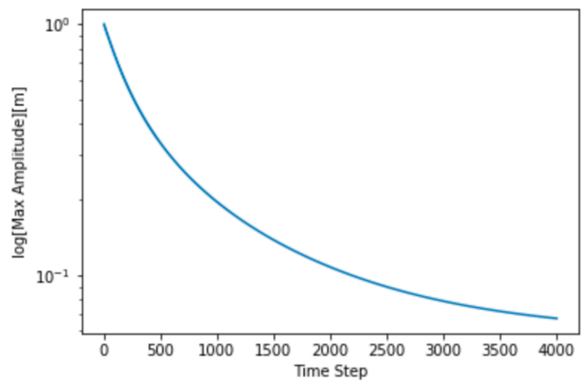


Again, the total amount of amplitude within the 2D domain is conserved over the entire integration time for diffusion-only solutions.

Maximum Amplitude–Time Step Graph (K = 0.1)



Maximum Amplitude–Time Step Graph (K = 0.5)



On both graphs, the amplitude is plotted against the time step in a log-linear scale. As anticipated, one can see a strong damping effect on the wave. By increasing the values of the diffusion coefficient, one can see a steeper drop in the wave's amplitude over time, as shown above for K = 0.1 & 0.5.

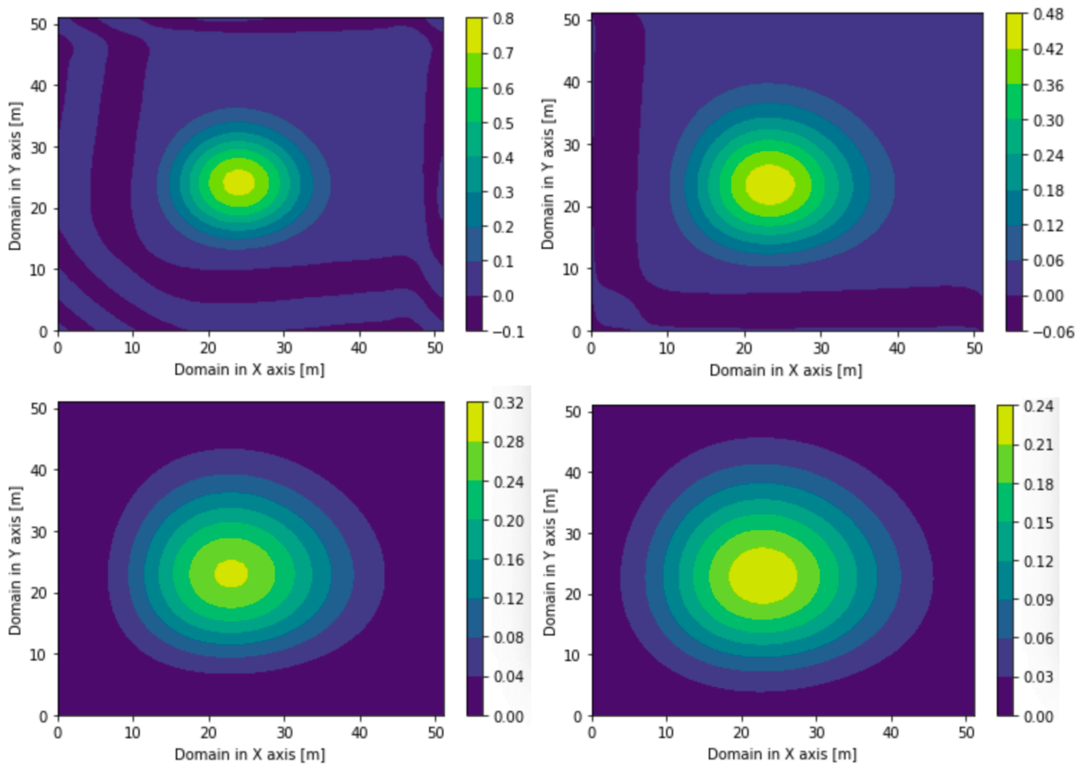
From both graphs, we cannot see the thick, oscillating lines found in advection-only's amplitude-time graphs. This further supports the absence of computational modes in diffusion solutions.

VI. Numerical Experiment C. Advection-Diffusion:

Basic Configurations: *K = 0.1, cx = cy = 1 (x,y velocities), Δt = 0.1*

Basic information: Total integral conserved

Contour Plot at T = 100s (Top left), 200s (Top Right), 300s (Bottom Left), 400s (Bottom right)

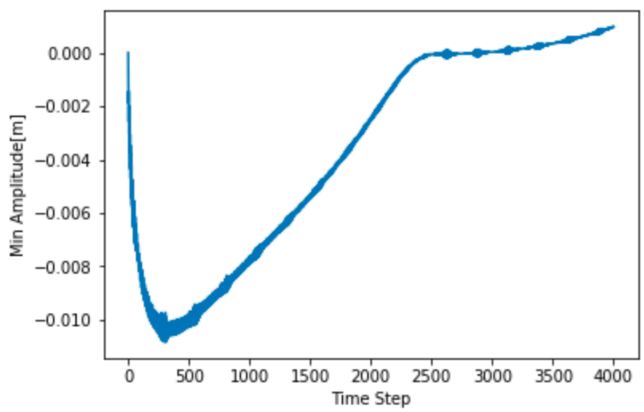


There are a number of interesting observations from the above contour plots. Similar to above, these plots are taken with 100s intervals, intending to show the waves' evolution over multiple periods.

1. In contrast to the advection only experiment, the wave's central peak appears to have less phase error compared to advection-only experiment. Instead, the wave peaks tend to remain close to the starting position. A likely explanation is that the advection-diffusion wave has a less severe phase error.
2. In previous experiments, the center wave appears to have a stronger phase error, lagging behind the other "side waves" that are propagating parallel to, but not on the diagonal. This resulted in asymmetric distortions in the wave contour. Although we can still see some asymmetries in the contour plots for advection diffusion, this is much less severe compared to advection only. The presence of diffusion likely decreases the wavenumber gradient of the disturbance's constituent waves, thus reducing phase differences caused by the Leapfrog scheme. Hence, diffusion might indirectly counter/hide the Leapfrog's dispersion effects.

3. Absence of negative amplitudes is another surprising observation in this run. At the early time steps, patterns of computational modes (see Top Left contour plot) are clearly visible. However, as T > 300s, the ripple-like patterns by the computational mode disappeared. Due to the intrinsic nature of computational modes in Leapfrog, the computational mode must be masked in some way. One possible explanation is that the diffusion acts on the oscillating computational modes, creating a cancellation effect.

Minimum Amplitude–Time Step Graph



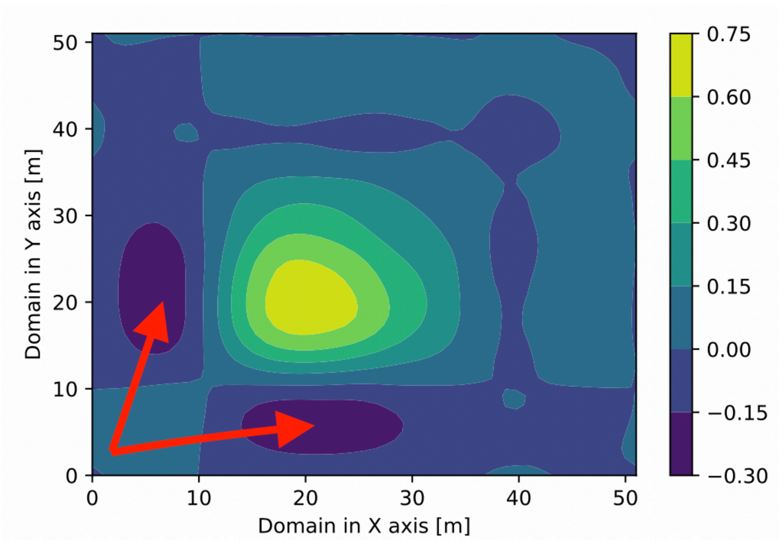
The minimum amplitude-time step graph of the advection diffusion solution is very interesting, showing some new features of the solutions. The most apparent feature is the rapid drop in minimum amplitude during the first 250 time steps, followed by a gradual increase that levels off at 0 amplitude. This confirms our previous observations with contour plots: the computational mode ripples were present during T< 200s (i.e. Time Step < 2000). But after T > 300s (i.e. Time Step > 3000), the contour plot no longer showed the computational mode ripples.

Another signature in this graph is the small, periodic bumps along the curve. They seem to have a period of 250 time steps, corresponding to 25 seconds. In my opinion, this pattern is due to the computational modes travelling in the opposite direction and meeting the main wave front again. Since the main wave needs 50 seconds to traverse the entire domain, the main wave and computation modes would intersect and overlap every 25 seconds.

Challenge Question: (Continued)

As previously discussed, the main wave's amplitude "loss" is not exactly an amplitude leakage, since the overall amplitude integral over domain is conserved for all numerical experiments performed. Instead, the apparent amplitude loss should be due to the main wave's peak super-positioning with some computational mode with negative amplitude. At the beginning of the advection-only numerical run, we see some ripple-like computational mode patterns with narrow wavelengths. However, as the computational modes enters and exits boundaries, they form interference patterns with one another, creating relatively deep peaks and troughs around the main wave.

Contour Plot at T = 400s



As pointed in the graph with red arrows, near the end of integration time, the negative amplitude troughs actually moved along with the main wave. This is quite interesting as these troughs are also the locations of minimum amplitudes. My hypothesis is that a trough similar to these overlaps with the main peak, thus causing a constant amplitude decay.