Name: Lam Ling Ho Adrian
UID: 105211252

Report 3 for A&O 180

Problem: Modelling the solution of a simple one-dimensional linear wave equation via Leap-Frog and 3rd Order Runge-Kutta (RK3) scheme.

I. Documentation:

A) Leap Frog Scheme (Centered in Time, Centered in Space):

a) Recursive Equations:

Leap Frog Scheme Recursive Equation

$$u_i^{n+1} = u_i^{n-1} - \frac{c\,\Delta t}{\Delta x}\,(u_{i+1}^n - u_{i-1}^n)$$

A Leap-Frog-based numerical solution requires a different time advancement for the first time step, as u_i^{n-1} is inexistent for the first time step. For our case, Forward-In-Time-Backwards-In-Space (FTBS) scheme is used exclusively for the first time step. The recursive equation for FTBS scheme is as follows:

$$u_i^{n+1} = u_i^n - \frac{c\,\Delta t}{\Delta x}\,(u_i^n - u_{i-1}^n)$$

Curiously, the Forward in Time, Centered in Space (FTCS) Scheme was previously used for advancing solution in the first time step. FTCS's recursive equation is given by:

$$u_i^{n+1} = u_i^n - \frac{c\,\Delta t}{2\Delta x}\,(u_{i+1}^n - u_{i-1}^n)$$

However, the scheme was abandoned after encountering high errors in numerical solutions.

b) Boundary Conditions:

In all of the numerical experiments in this report, cyclic boundary conditions are implemented across the wave domain.

Right Boundary Condition:

$$u_{Nx-1}^{n+1} = u_{Nx-1}^{n-1} - \frac{c\,\Delta t}{\Delta x}\,(u_1^n - u_{Nx-2}^n)$$

Left Boundary Condition:

$$u_0^{n+1} = u_{Nx-1}^{n+1}$$

u^{n+1} is the numerical solution in new time step

u^n is the numerical solution in current time step

u^{n-1} is the numerical solution in previous time step

u_{Nx-1}^{n+1} is the numerical amplitude at the last grid point in new time step (right boundary condition)

u_1^n is the numerical amplitude at 2nd grid point in space in the current time step.

u_0^{n+1} is the numerical amplitude at 1nd grid point in space in the new time step.

In short, the right boundary condition is calculated via Leap Frog scheme, which requires information about both the grid points "in front" & "behind" of the right boundary point. In a cyclic boundary condition, the "in front" grid point is "recycled" to the left domain boundary. Since the definition of cyclic boundary is that left & right boundary points must be equal, the next "real" "in front" data point lies in the 2nd grid point, and thus u_1^n.

B) Runge-Kutta 3rd Order in Time (RK3)

a) Recursive equations:

$$\textit{Runge} - \textit{Kutta 3rd Order in Time}$$

$$u_i^{*} = u_i^n + \frac{\Delta t}{3} \times \frac{-c}{2\,\Delta\,x}\left(u_{i+1}^n - u_{i-1}^n\right) \quad (\textit{Implement Boundary Conditions})$$

$$u_i^{**} = u_i^n + \frac{\Delta t}{2} \times \frac{-c}{2\,\Delta\,x}\left(u_{i+1}^{*} - u_{i-1}^{*}\right) \quad (\textit{Implement Boundary Conditions})$$

$$u_i^{n+1} = u_i^n + \Delta t \times \frac{-c}{2\,\Delta\,x}\left(u_{i+1}^{**} - u_{i-1}^{**}\right) \quad (\textit{Implement Boundary Conditions})$$

u_i^{*} & u_i^{**} are 1st and 2nd temporary numerical estimation in RK3.

b) Boundary Conditions:

The boundary conditions implemented are very similar to Leap Frog. They are implemented for every stage for RK3. The right boundary value is first calculated from RK3, then the value is copied to left boundary value, fulfilling the cyclic domain condition.

Right Boundary Condition:

Calculated from RK3 (See above recursive equation)

Left Boundary Condition:

$$u_0^{n+1} = u_{Nx-1}^{n+1}$$

II. Experiments:

Exp.	Scheme	Δt	λ	Courant # (C)	Main Goal:
# 1	LF	2s	50m	1	Base case for code testing
# 1	RK3	2s	50m	1	Base case for code testing
# 2	LF	1s	50m	0.5	Assess effects of changing C to below 1 Determine amplitude (dampen) & phase errors
# 2	RK3	1s	50m	0.5	Assess effects of changing C to below 1 Determine amplitude (dampen) & phase errors
# 2	LF	4s	50m	2	Assess effects of changing C to above 1 (expected to "blow up) Determine properties of stability
# 2	RK3	4s	50m	2	Assess effects of changing C to above 1 (expected to "blow up) Determine properties of stability
# 3	LF	2s	10m	1	Assess effects of amplitude & phase errors on a less resolved wave
# 3	RK3	2s	10m	1	Assess effects of amplitude & phase errors on a less resolved wave
# 4	LF	1s	25m	0.5	Square wave Observe dispersive behavior and (amplitude & phase) errors
# 4	RK3	1s	25m	0.5	Square wave Observe dispersive behavior and (amplitude & phase) errors

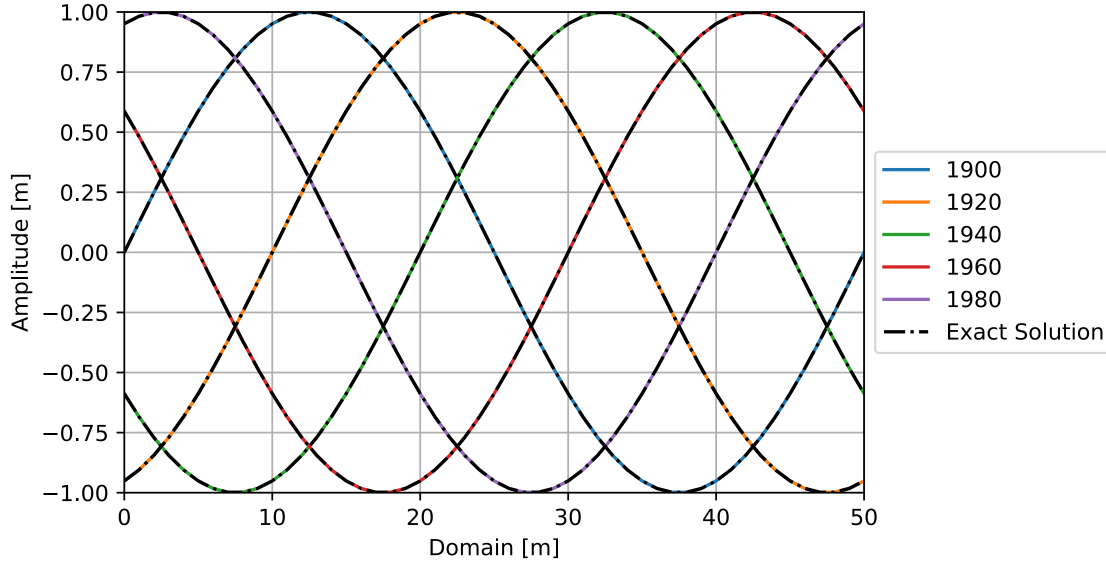
---

Experiment #1:
<u>Leap Frog Scheme, Δt =2s, λ = 50m , C = 1</u>

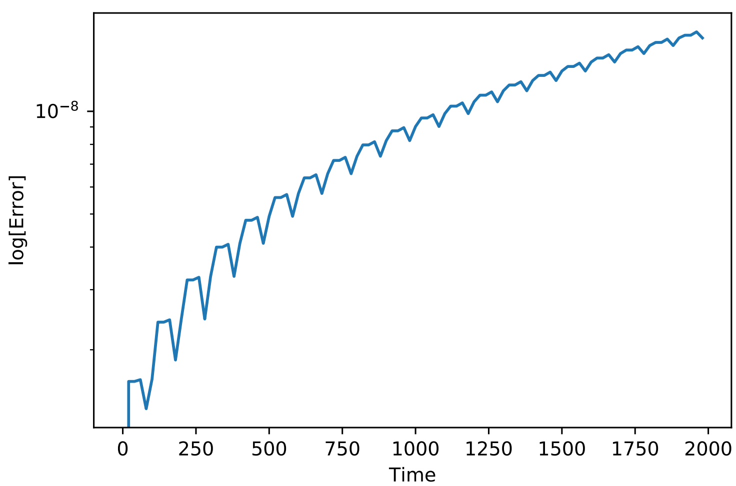
The purpose of this experiment is to evaluate the performance of the numerical solution at ideal C condition (C = 1). The following plot shows the numerical solution and exact solution for total integration time = 2000s. For clarity, the above plot shows the last 5 solutions taken at 20s intervals. This is done because numerical errors tend to accumulate and become more apparent as the solution advances near the end of integration time

One can observe that the numerical solutions overlap with the exact solutions, with both amplitude and phase preserved. Specifically, no computational mode is observed, most likely due to the large $\lambda/\Delta x$ ratio in this experiment.

Amplitude-Domain Diagram for Leap Frog with C = 1

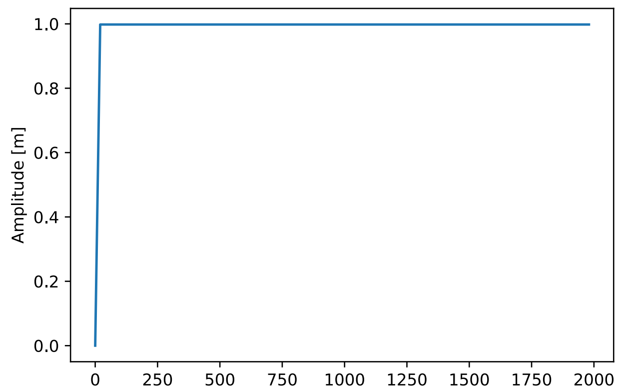


Error-Time Plot for Leap Frog with C = 1



The error is also plotted with time in a log-linear scale below. One can observe that the error increases gradually with time, but overall still remains very low at ~10E-8 throughout T = 2000.

Amplitude-Time Plot for Leap Frog with C = 1



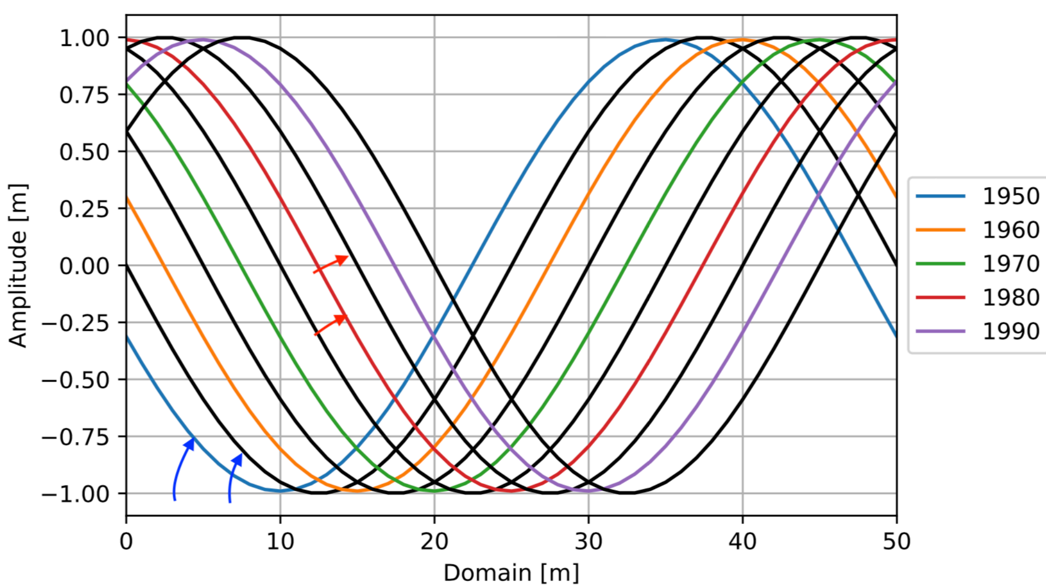
In the Amplitude-Time Plot, the amplitude is preserved and remains constant at 1m. This agrees with the previous observations.

In conclusion: Leap Frog demonstrates no observable damping and phase error for C = 1.

Experiment #1:

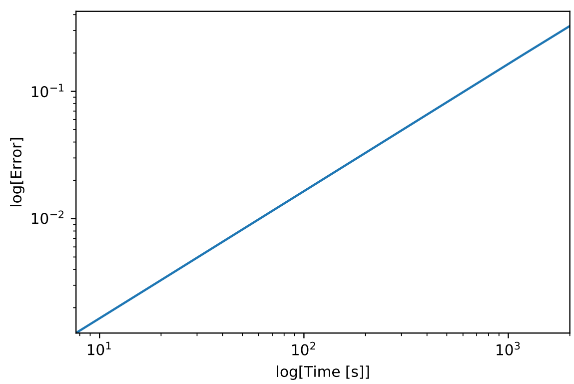
RK3 Scheme, Δt =2s, λ = 50m, C = 1

Amplitude-Domain Plot for RK3, C = 1

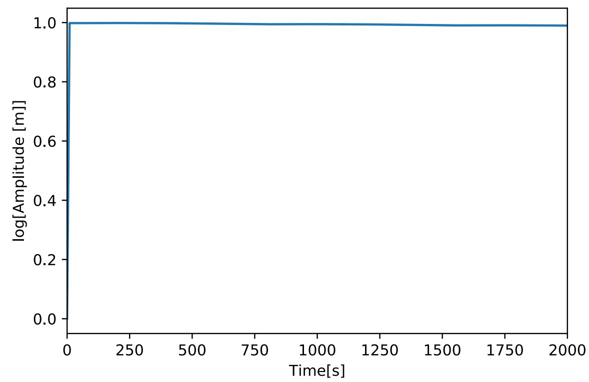


The same experiment is carried out with RK3 scheme, mainly to serve as a base case to evaluate code performance. However, one noticeable observation is that RK3 numerical solution tends to have phase errors even for C = 1, unlike the previous LF solution. For clarity, the above plot shows the last 5 solutions taken at 10s intervals. The red and blue arrow highlights the numerical solution and its respective numerical solution. One can observe that numerical solutions for RK3 tend to travel slower than the exact solutions.

Error-Time Plot for RK3, C = 1

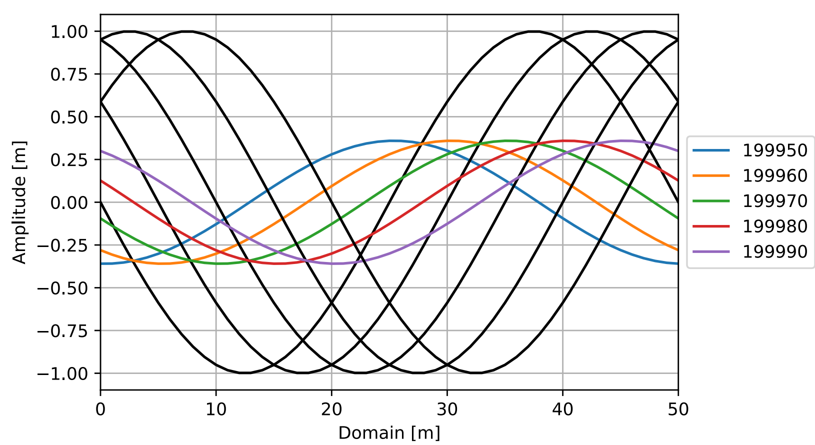


Amplitude-Time Plot for RK3, C = 1



In the above plots, one can see the error is much higher for RK3 and increases exponentially. This is most likely due to the phase error, as the amplitude is almost perfectly preserved. But under very close observation, the amplitude actually has a very small decrease over t = 2000s.

Amplitude-Time Plot for RK3, C = 1

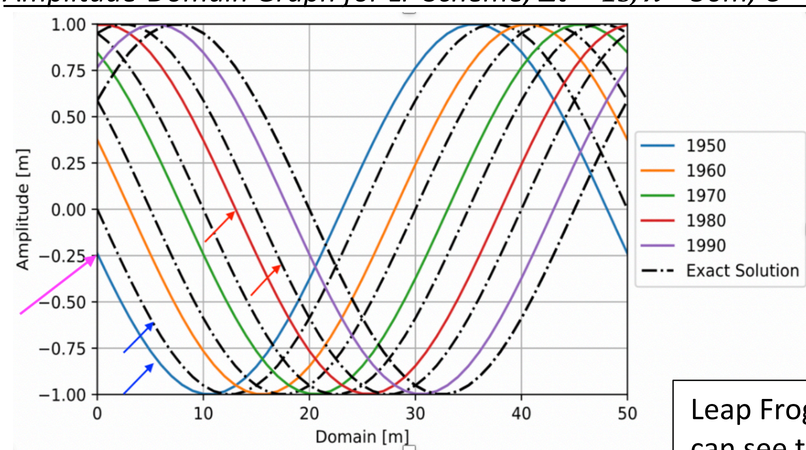


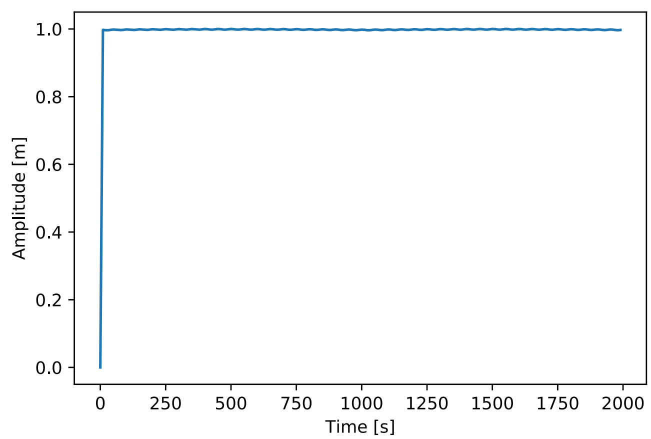
After extending the total integration time to T = 200000s (after 100000 time steps), the damping effect is much more apparent.

Experiment 2a) :

LF Scheme, Δt =1s, λ = 50m, C = 0.5   vs.   RK3 Scheme, Δt =1s, λ = 50m, C = 0.5

Amplitude-Domain Graph for LF Scheme, Δt =1s, λ = 50m, C = 0.5

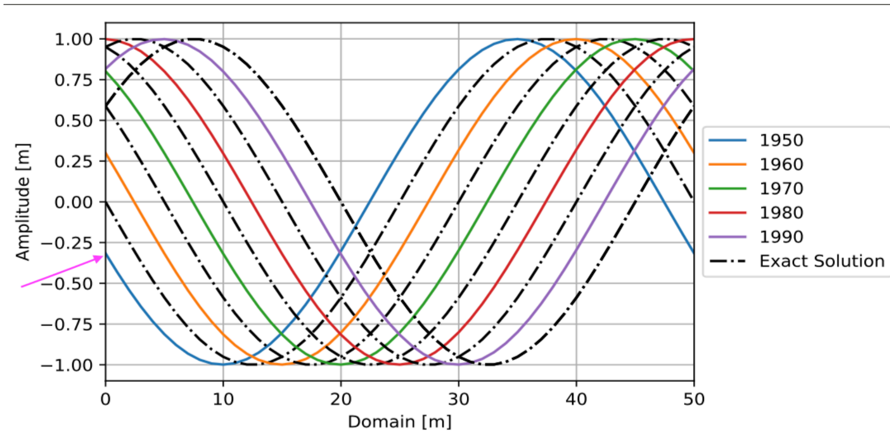




Leap Frog: In the above Amplitude-Domain plot, one can see that for LF scheme at C = 0.5, there is observable phase error but no amplitude error, as compared to the near perfect LF solution for C = 1 in Experiment #1. The corresponding exact and numerical solution are also labelled with arrows. Again, the numerical solutions tend to travel slower than exact solution, similar to RK3 for C = 1 (Exp. #1)

The lack in amplitude error is further supported by the Amplitude-Time graph to the right, which shows no wave damping over time.

Amplitude-Domain Graph for RK3 Scheme, Δt =1s, λ = 50m, C = 0.5



RK3: The results for RK3 is largely similar to Leap Frog numerical solutions. The amplitude is preserved and phase error is present.

Comparing qualitatively, the phase error seem to be slightly more severe for RK3. Labelled in both graphs by the pink arrow, the Y-intercept for the t = 1950s wave is slightly higher for RK3, indicating that the numerical solution is “lagging behind” more.

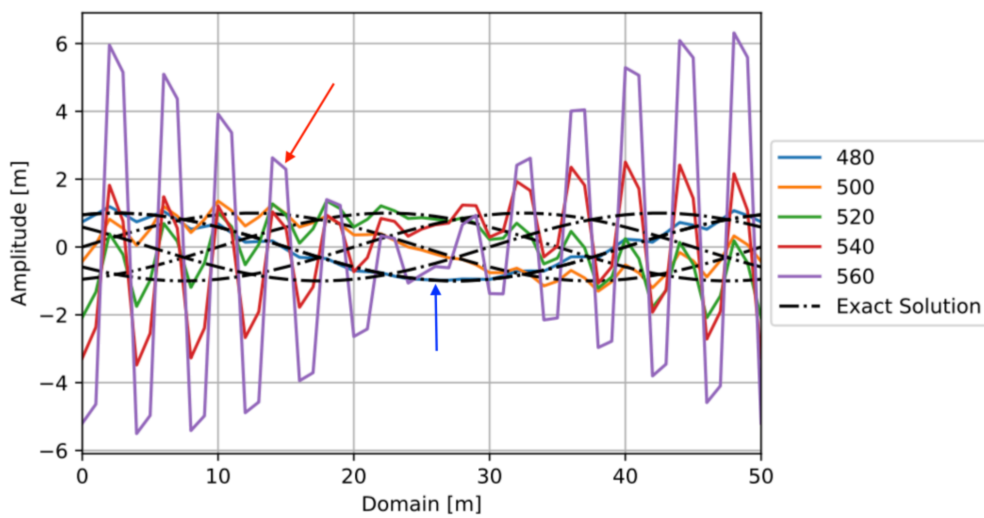
In conclusion, both schemes demonstrate negligible amplitude error but observable phase errors for C = 0.5.

Experiment 2b):

LF Scheme, Δt =4s, 𝜆 = 50m, C = 2 vs. RK3 Scheme, Δt =4s, 𝜆 = 50m, C = 2

The purpose of this experiment is to examine the instability behaviors of both numerical schemes. To do so, a large (i.e. > 1) Courant number is selected by changing the time step size to 4s. As expected, both solutions soon "blew up" and increased exponentially.

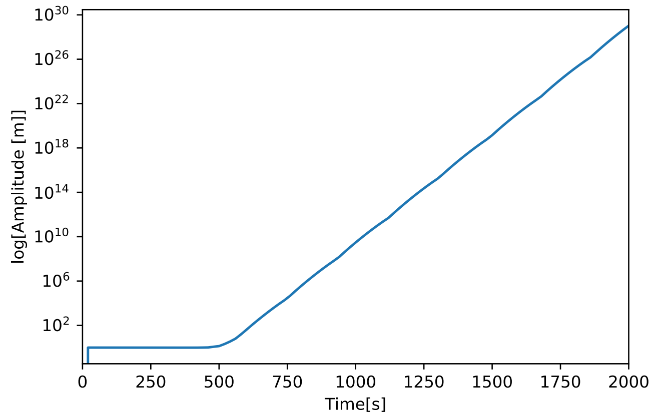
Amplitude-Domain Graph for RK3 Scheme, Δt =4s, 𝜆 = 50m, C = 2



In this experiment, both schemes demonstrate similar behavior at some time within the total integration time 2000s. To demonstrate this behavior, the Amplitude-Domain graph is plotted above with exact solutions for RK3 scheme.

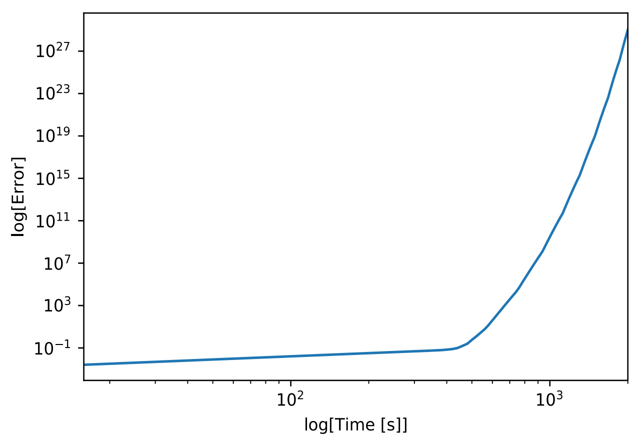
For the solution at t = 480s (shown by blue arrow), the numerical wave is mostly following the exact solution, with slight spikes over time. The waveform integrity progressive worsen, until the exponential increase in amplitude occurring at t ~560s, where the amplitude grows quickly and the solution flips signs every 2 grid steps.

Amplitude-Time Graph for RK3 Scheme, Δt =4s, λ = 50m, C = 2

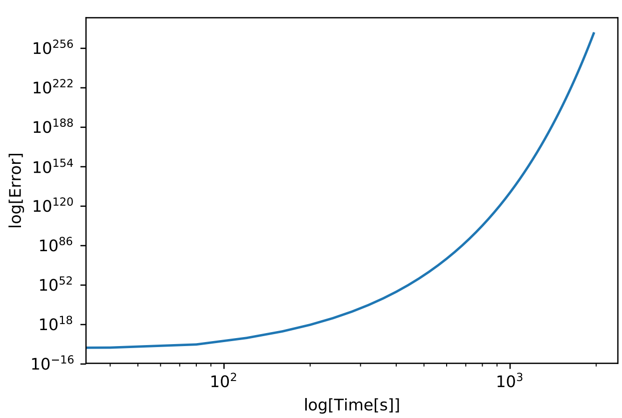


From the Amplitude-Time Graph, one can also confirm the amplitude increases exponentially with time

Error-Time Graph for RK3 Scheme, Δt =4s, λ = 50m, C = 2.0



Error-Time Graph for LF Scheme, Δt =4s, λ = 50m, C = 2



As shown in the Error-Time Graphs of RK3 & LF, one interesting observation is that RK3 "blows up" much later than LF, as the RK3 solution remains stable until t > 400s. On the other hand, LF shows instability behavior very early on.

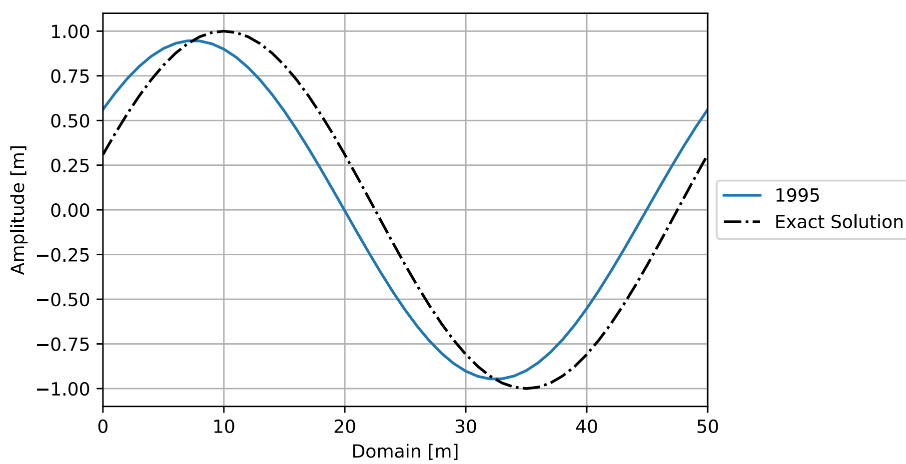
However, this is insufficient for concluding that RK3 is more stable than LF, since the time that it takes to "blow up" for LF is also affected by how is the first time step carried out (FTBS in our case).

So, the only definitive way of testing stability is to push the stability boundary for C by increasing Time Step Sizes.

- Stability Limits for RK3 & LF

  After testing different values for the stability boundary for both schemes, it is found out that Leap Frog quickly becomes unstable for $\Delta t > 2s$, $(C > 1)$. However, RK3 is still stable for Δt = 3.5s over T = 2000s.

  Amplitude-Domain Graph for RK3 Scheme, Δt =3.5s, λ = 50m, C = 1.75

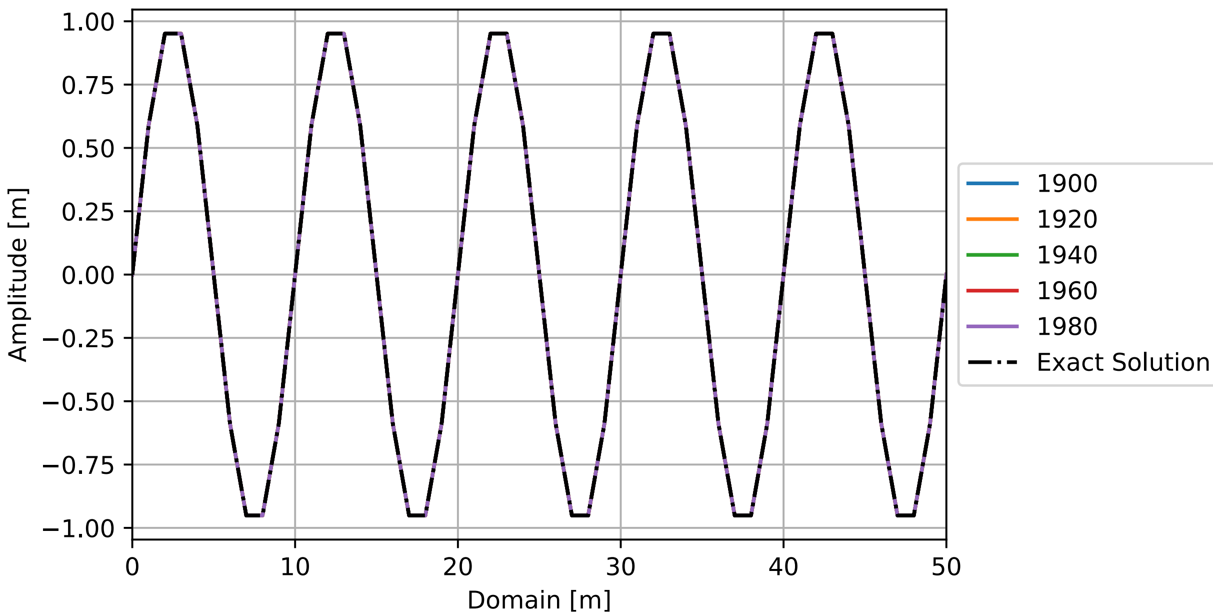


In the above graph, the numerical solution at t = 1995s is plotted with the exact solution. One can see that both amplitude and phase errors are present in this solution, with the amplitude being slightly damped.

Experiment 3:

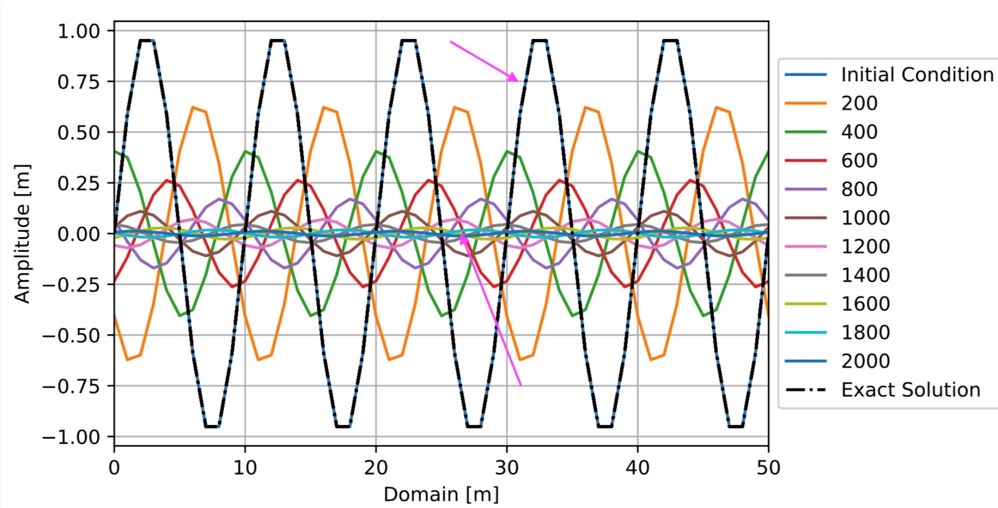
<u>LF Scheme, Δt =2s, λ = 10m, C = 1 vs. RK3 Scheme, Δt =2s, λ = 10m, C = 1</u>

<u>Amplitude-Domain Graph for LF Scheme, Δt =2s, λ = 10m, C = 1</u>

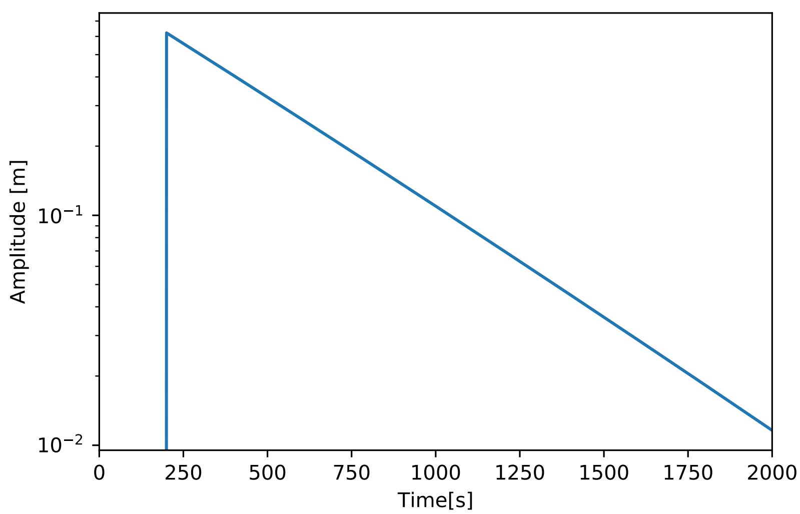


For Leap Frog scheme, the result is very similar to the results in Exp. #1 (same conditions except λ = 50m) In the above graph, the solutions from 5 time steps near T = 2000s are plotted, with 20 s intervals. As in Exp #1 (LF), there are negligible phase and amplitude errors for total integration time = 2000s. The fixed yet decreased amplitude (the waves' peaks appear as being "chopped off") is mostly likely due to the low λ/Δx ratio, which translates to poor wave resolution.

Amplitude-Domain Graph for RK3 Scheme, Δt =2s, λ = 10m, C = 1



From the above Amplitude Domain Graph for RK3, we can reach several interesting observations. For clarity, the graph only plots numerical solutions every 100s and their corresponding exact solution (which is the same wave). The first interesting observation is that the numerical solution in this case have both significant phase and amplitude errors (damping). For damping effect, the pink arrows show both the first initial wave and the final solution at t = 2000, which has a close-to-zero amplitude. This shows that in this implementation, RK3 is dissipative. From the amplitude-time plot below, one can also conclude that the amplitude dampening is an exponential decay.



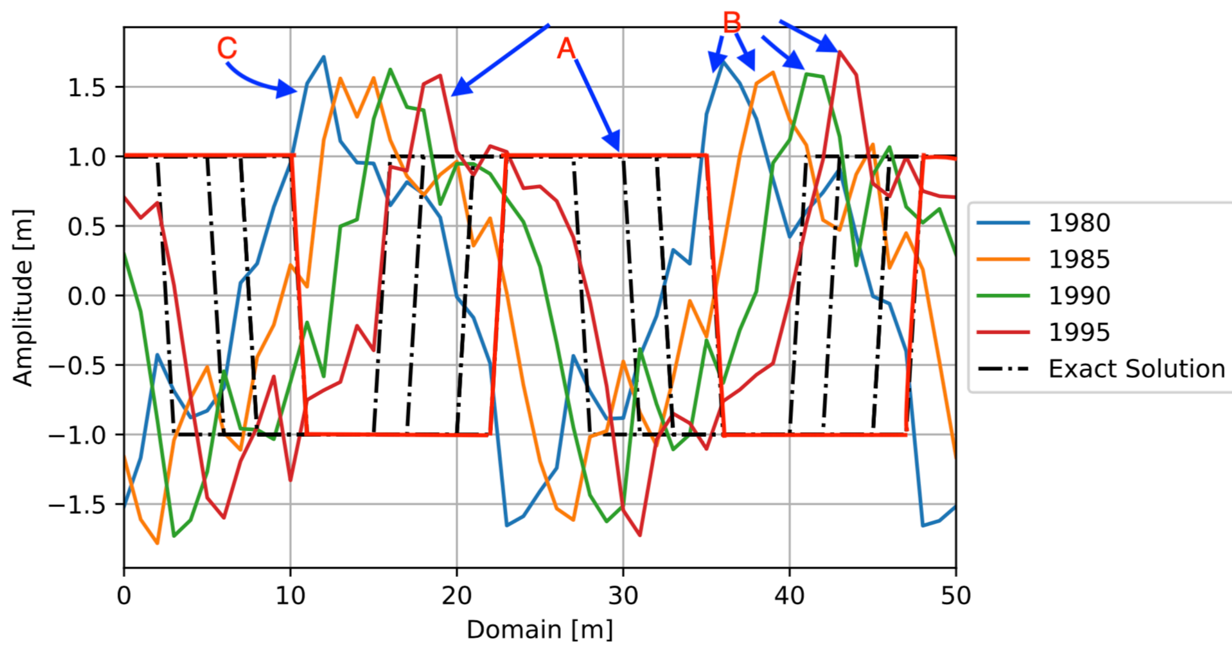
In conclusion, comparing this result with Experiment 1 (RK3), we can see that decreasing λ leads to much higher amplitude and phase errors. Also, the waveform in this experiment is highly distorted especially at later time steps, in comparison, the numerical solutions for

Experiment 1(RK3) remained a smooth sine wave overall. Qualitatively, this is because $\lambda/\Delta x$ determines how well a wave is resolved, a less resolved wave would lead to higher numerical errors.

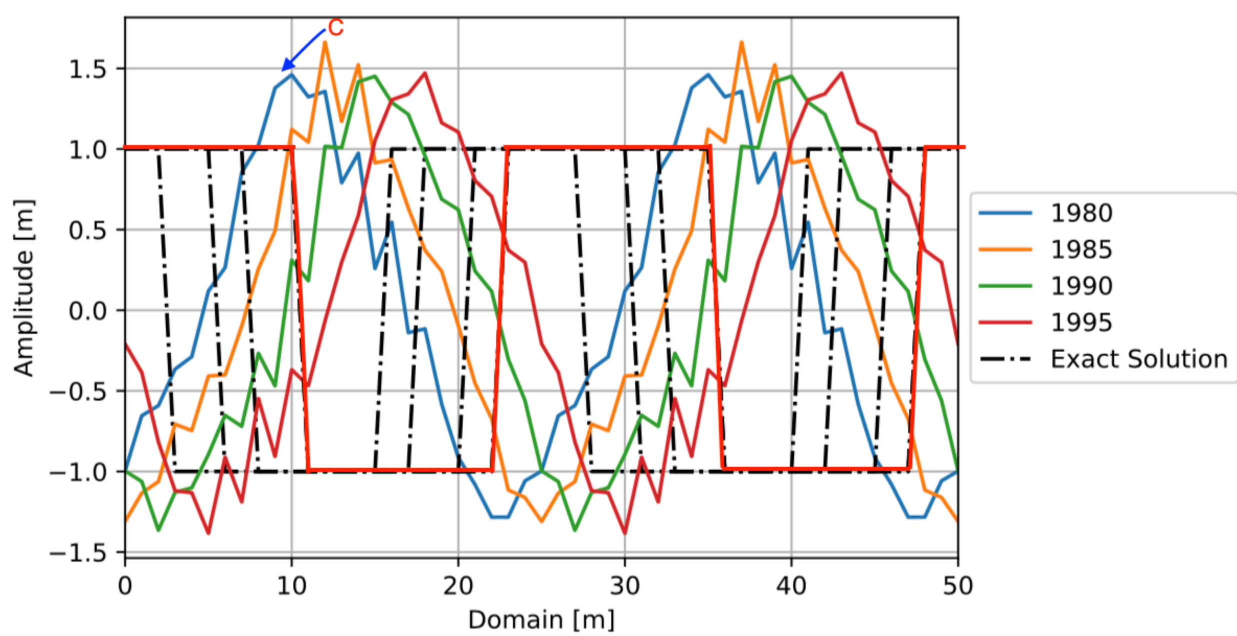
Experiment 4:

<u>LF Scheme, Δt =1s, λ = 25m, C = 0.5 Square Wave vs. RK3 Scheme, Δt =1s, λ = 25m, C = 0.5 Square Wave</u>

<u>Amplitude-Domain Graph for LF Scheme, Δt =1s, λ = 25m, C = 0.5</u>



<u>Amplitude-Domain Graph for RK3 Scheme, Δt =1s, λ = 25m, C = 0.5</u>

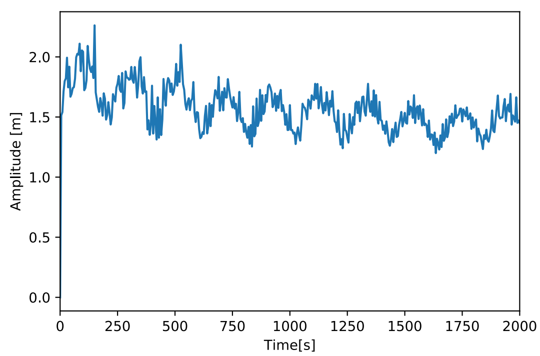


In the above Amplitude-Domain Graphs (RK3 & LF), 4 solutions are plotted near T = 2000s, with time interval of 5s. Firstly, both solutions share many similarities, including distortion in waveform and phase errors. The reason for the “jagged” spikes in the solution is as follows: a square wave consists of an infinite number of Fourier components, and due to each component has different wave numbers, each would then propagate in different velocities under both discretization scheme. In this case, both RK3 and LF are dispersive in nature. There are several interesting points labelled by the arrows:

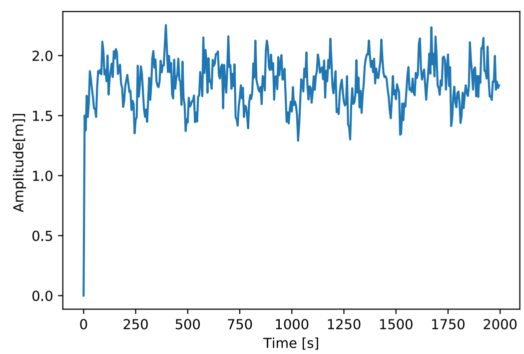
A) the numerical solution at t = 1995 and the corresponding exact solution, showing the phase error. Again, one can see the numerical solution tends to propagate slower than the exact solution.

B) For both solutions, we tend to see a fixed increase in amplitude (> 1) :

Amplitude-Time Graph for RK3



Amplitude-Time Graph for LF



In the amplitude-time graphs above, RK3 and LF solutions both oscillate between 1.5 & 2m amplitude at early time step. LF solution fluctuates around the same level throughout while RK3 shows a damping behavior, eventually oscillating around 1.5 as T approaches 2000.

For both schemes, this is most likely due to the dispersion of waves, where shorter waves (comprising of the “edges” of the square wave) slow down more (due to their larger wavenumbers) and superpose with the longer waves (comprising of the “center” of the square wave).

For LF, the presence of computational mode is also suspected, as $\lambda = 25m$ and thus the wave is less resolved.

C) By tracking the LF & RK3 solutions at the same time, RK3 solutions appears to have a larger phase error than LF by comparing the solution in the same time instance/time step (both are valid since RK3 & LF used the same time step sizes)

<u>**Comparison Between FTBS, RK3 & LF.**</u>

- Forward in Time Backwards in Space (FTBS) vs Leap Frog
    - As shown in the previous homework, Upstream FTBS has negligible phase error but significant amplitude error (amplitude damping).
    - On the other hand, Leap Frog doesn't have amplitude error as long as it is stable, yet it produces phase errors in its solutions except when C = 1.
- RK3 vs Leap Frog
    - RK3 remains <u>**stable**</u> for some Courant Number > 1 (C limit ~ 1.75 for stability over total integration time = 2000 / ~570 time steps). On the other hand, Leap Frog solutions tend to strictly "blow up" once C > 1. Thus, we can see that the RK3 is more stable and more tolerant to larger time steps.
    - In multiple experiments (i.e. Experiment 1, 2a, 3, 4), it Is observed that RK3 has higher phase errors than LF, even at C = 1 (Experiment 1).
    - RK3 doesn't have computational mode while LF does. This problem becomes significant if the wave is less resolved (low λ/dx ratio).
    - As an extra run, it was found out that LF Square Wave Solutions match with exact solution perfectly if C = 1. Thus, when C is ideal (C = 1), Leap Frog solutions tend to be highly accurate regardless of wavelength and wavenumber (not dispersive nor dissipative).
    - In multiple experiments, RK3 tends to be more dissipative (more amplitude damping) than LF under the same Δt.