# 26 Oct 2023 - Activity: 1-Dimensional Travelling Waves

[Wave equations](https://en.wikipedia.org/wiki/Wave_equation) are common to all branches of physics. In fact, many modern research topics include some investigations that use wave theory. So useful is the concept of a wave, that we adapted it for quantum mechanics in the form of the [Schrodinger Equation](https://en.wikipedia.org/wiki/Schr%C3%B6dinger_equation) and the position representation of the wavefunction (). While wildly useful and broadly applicable, we need to build up some intuition and use cases for waves. We will start with the 1D wave equation:

As we discussed, any function of the form:

solves the 1D wave equation. One way to think about this what this solution is showing is that some smooth function and another smooth function individually solve the wave equation if they just move to the left for and to the right for . Furthermore because this PDE is linear (containing only linear derivatives of and , i.e., no squares or functions of them), [superposition](https://en.wikipedia.org/wiki/Superposition_principle) holds. And we can simply add solutions together linearly to get new solutions!

## Travelling Wave Solutions

That is all well and good, but we need a common language. A grammar to understand our solutions. These are basis functions, they are the things we all agree on that describe the “space” of the solutions. You have seen these already in our study of normal modes. Our sinusoidal solutions, the frequencies and amplitudes of the modes themselves, provide a set of basis functions that can describe the general motion of the coupled SHOs. What’s nice about that choice is that we can reproduce it because we all agree to solve the same eigenvalue problem.

For the 1D wave equation, these functions are [Travelling Wave Solutions](https://en.wikipedia.org/wiki/Periodic_travelling_wave). The describe the simplest motion of a travelling wave: single frequency oscillation over an infinite domain. There’s no end points to worry about, and only one frequency. Also, it just moves to the left or to the right. You can probably see how multiple dimensions result in much more complexity. A basic traveling wave solution is:

You can show that this equation satisfies the wave equation with .

More importantly, does a linear sum do so?

**✅ Do this**

Demonstrate that this proposed general solution:

solves the wave equation.

## Looking at Travelling Wave Solutions

Let’s investigate several scenarios for travelling wave solutions:

1. A single travelling wave (done for you with different representations)
2. The superposition of two waves (not close frequencies; close amplitudes)
3. The superposition of two waves (close frequencies; close amplitudes)
4. The superposition of two waves (not close frequencies; not close amplitudes)
5. The superposition of two waves (close frequencies; not close amplitudes)
6. The superposition of three waves (make adjustments to test your intuition from two waves)

Make your choices, but also vary them a bit. You can investigate things like widgets (e.g., [sliders](https://matplotlib.org/stable/gallery/widgets/slider_demo.html)) which make this stuff more interactive, but regular plots are totally fine.

**✅ Do this**

Here is my suggestion and a challenge:

1. Plot your solutions in 2D for a fixed location (e.g., ) for all time ( vs )
2. Plot your solutions in 2D for a given time (e.g., ) for all x ( vs )
3. Plot your solutions in 3D using the plane as positions vs time and using for (what does this representation buy you? Is it more or less useful than the above two?)
4. Challenge (animate your plots). Matplotlib can use the animation api. [Here’s a simple script](https://matplotlib.org/2.0.2/examples/animation/simple_anim.html).

import numpy as np  
import matplotlib.pyplot as plt  
from mpl\_toolkits.mplot3d import Axes3D  
from matplotlib.animation import FuncAnimation  
from IPython.display import HTML  
%matplotlib inline

### Single Travelling Wave

Here’s the plots for a single travelling wave with , , , and .

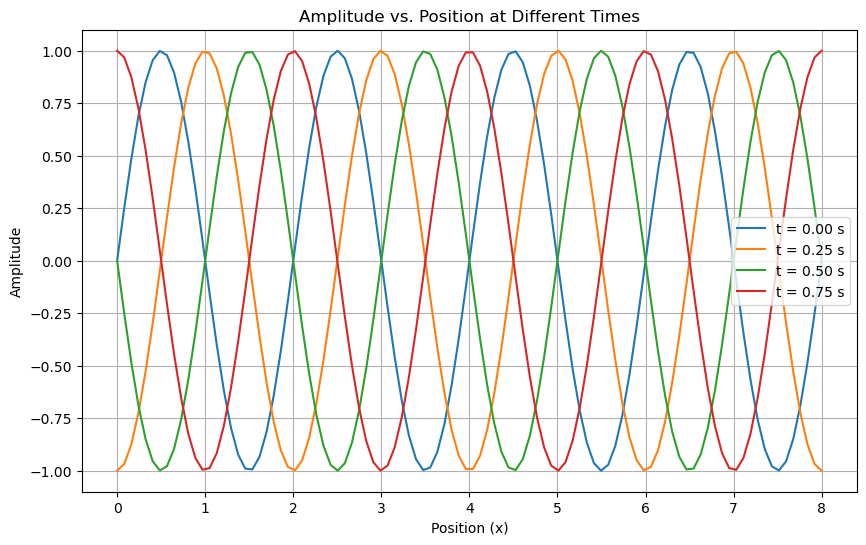
First we will create the wave. This is a little bit difficult because both and can vary, but we’ve made functions of two dimensions before using `meshgrid, so we can just employ that with a little function to complete the wave.

# Parameters  
A = 1.0 # Amplitude  
lambda\_ = 2.0 # Wavelength (underscore needed because lambda is a reserved word)  
f = 1.0 # Frequency in Hz  
phi = 0.0 # Phase constant in radians  
  
k = 2 \* np.pi / lambda\_  
omega = 2 \* np.pi \* f  
  
# Define the wave equation  
def wave(x, t):  
 return A \* np.sin(k \* x - omega \* t + phi)  
  
# Generate x and t values  
x = np.linspace(0, 4 \* lambda\_, 100)  
t = np.linspace(0, 2/f, 100)  
  
X, T = np.meshgrid(x, t) # Create a meshgrid for x and t values  
Y = wave(X, T) # Compute the amplitude for every combination of x and t

#### Plotting amplitide vs location

Every location is waving, so here we look at all location at a single time. And then we plot the amplitude of the wave at that location. We can do this for different times.

# Plot amplitude vs. position for different times  
times\_to\_plot = [0, 0.25/f, 0.5/f, 0.75/f] # Chosen specific times to show one complete cycle  
  
plt.figure(figsize=(10, 6))  
  
for plot\_time in times\_to\_plot:  
 plt.plot(x, wave(x, plot\_time), label=f't = {plot\_time:.2f} s')  
  
plt.title("Amplitude vs. Position at Different Times")  
plt.xlabel('Position (x)')  
plt.ylabel('Amplitude')  
plt.legend()  
plt.grid(True)

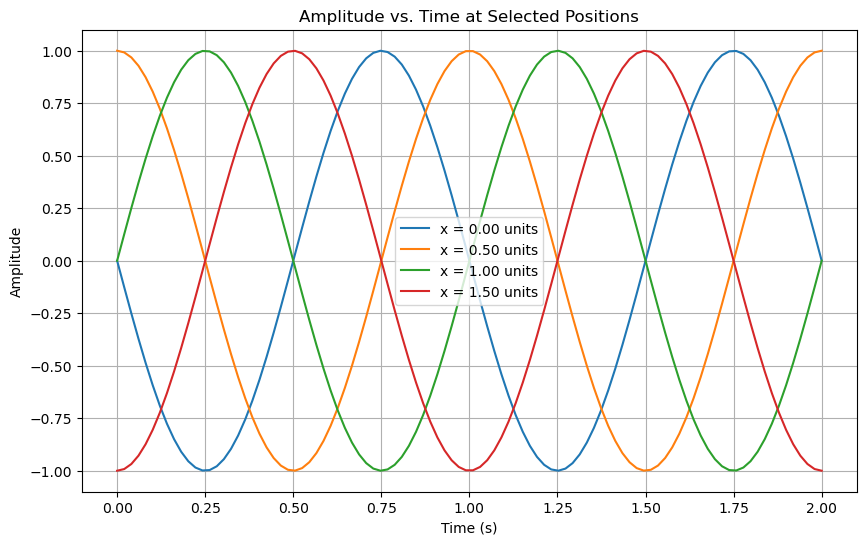


png

#### Plotting amplitude vs. time for different positions

At all times the wave is waving, so we can look at the amplitude of the wave at all times for a single location. We can do this for different locations.

# Selected positions to plot  
positions\_to\_plot = [0, lambda\_/4, lambda\_/2, 3\*lambda\_/4]  
  
plt.figure(figsize=(10, 6))  
  
for pos in positions\_to\_plot:  
 plt.plot(t, wave(pos, t), label=f'x = {pos:.2f} units')  
  
plt.title("Amplitude vs. Time at Selected Positions")  
plt.xlabel('Time (s)')  
plt.ylabel('Amplitude')  
plt.legend()  
plt.grid(True)  
plt.show()

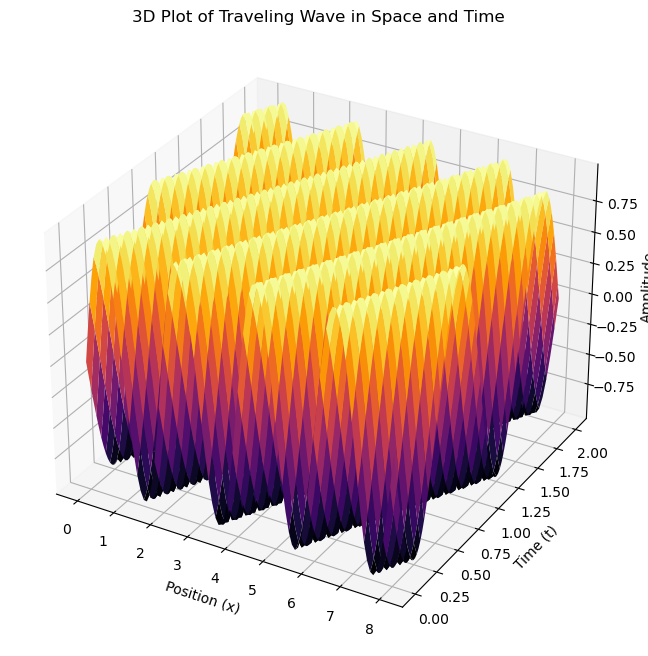


png

### Plotting space and time together

It’s a bit hard to see what this 1D wave is doing, so we can instead plot position and time on the plane and let the height represent the amplitude.

# Create a 3D plot  
fig = plt.figure(figsize=(12, 8))  
ax = fig.add\_subplot(111, projection='3d')  
  
ax.plot\_surface(X, T, Y, cmap='inferno') # Using viridis colormap for better clarity  
ax.set\_title("3D Plot of Traveling Wave in Space and Time")  
ax.set\_xlabel("Position (x)")  
ax.set\_ylabel("Time (t)")  
ax.set\_zlabel("Amplitude")  
ax.view\_init(elev=30, azim=-60) # Adjust viewing angle for better visualization

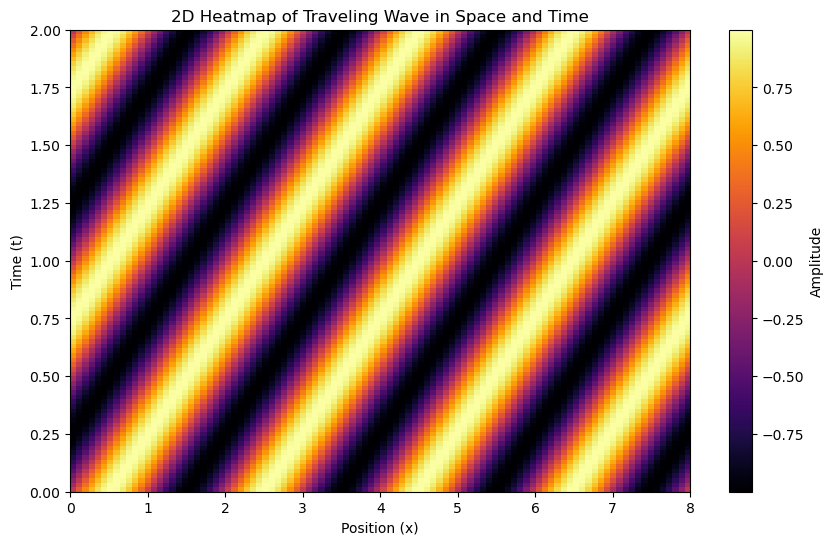


png

#### Ok, but make it 2D.

This isn’t the easiest representation to view. You might be able to adjust it a bit with changing the viewing angle. But, we can flatten this plot into a heat map. This is the more common way of viewing these kind of data.

plt.figure(figsize=(10, 6))  
plt.imshow(Y, aspect='auto', origin='lower', extent=[x.min(), x.max(), t.min(), t.max()], cmap='inferno')  
plt.colorbar(label="Amplitude")  
plt.title("2D Heatmap of Traveling Wave in Space and Time")  
plt.xlabel("Position (x)")  
plt.ylabel("Time (t)")  
  
plt.show()



png

#### But I really want to animate it

Sure. Here’s that code.

fig, ax = plt.subplots(figsize=(10, 6))  
line, = ax.plot(x, wave(x, 0)) # Initial plot (t=0)  
ax.set\_ylim([-A, A]) # Setting the y limits to the amplitude range  
ax.set\_xlabel('Position (x)')  
ax.set\_ylabel('Amplitude')  
ax.grid(True)  
ax.set\_title("Amplitude vs. Position over Time")  
  
def update(frame):  
 line.set\_ydata(wave(x, frame)) # Update the y data of the line plot for the current time frame  
 ax.set\_title(f"Amplitude vs. Position at t = {frame:.2f} s")  
 return line,  
  
# Animate over the given times using FuncAnimation  
ani = FuncAnimation(fig, update, frames=t, blit=True, interval=50)  
plt.close(fig) ## needed to stop another static image from being displayed  
  
# Display the animation in the notebook  
HTML(ani.to\_jshtml())

<input id="\_anim\_slider3a4ba7a62c13475c8b9940f195fcfc0e" type="range" class="anim-slider"  
 name="points" min="0" max="1" step="1" value="0"  
 oninput="anim3a4ba7a62c13475c8b9940f195fcfc0e.set\_frame(parseInt(this.value));">  
<div class="anim-buttons">  
 <button title="Decrease speed" aria-label="Decrease speed" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.slower()">  
 <i class="fa fa-minus"></i></button>  
 <button title="First frame" aria-label="First frame" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.first\_frame()">  
 <i class="fa fa-fast-backward"></i></button>  
 <button title="Previous frame" aria-label="Previous frame" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.previous\_frame()">  
 <i class="fa fa-step-backward"></i></button>  
 <button title="Play backwards" aria-label="Play backwards" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.reverse\_animation()">  
 <i class="fa fa-play fa-flip-horizontal"></i></button>  
 <button title="Pause" aria-label="Pause" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.pause\_animation()">  
 <i class="fa fa-pause"></i></button>  
 <button title="Play" aria-label="Play" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.play\_animation()">  
 <i class="fa fa-play"></i></button>  
 <button title="Next frame" aria-label="Next frame" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.next\_frame()">  
 <i class="fa fa-step-forward"></i></button>  
 <button title="Last frame" aria-label="Last frame" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.last\_frame()">  
 <i class="fa fa-fast-forward"></i></button>  
 <button title="Increase speed" aria-label="Increase speed" onclick="anim3a4ba7a62c13475c8b9940f195fcfc0e.faster()">  
 <i class="fa fa-plus"></i></button>  
</div>  
<form title="Repetition mode" aria-label="Repetition mode" action="#n" name="\_anim\_loop\_select3a4ba7a62c13475c8b9940f195fcfc0e"  
 class="anim-state">  
 <input type="radio" name="state" value="once" id="\_anim\_radio1\_3a4ba7a62c13475c8b9940f195fcfc0e"  
 >  
 <label for="\_anim\_radio1\_3a4ba7a62c13475c8b9940f195fcfc0e">Once</label>  
 <input type="radio" name="state" value="loop" id="\_anim\_radio2\_3a4ba7a62c13475c8b9940f195fcfc0e"  
 checked>  
 <label for="\_anim\_radio2\_3a4ba7a62c13475c8b9940f195fcfc0e">Loop</label>  
 <input type="radio" name="state" value="reflect" id="\_anim\_radio3\_3a4ba7a62c13475c8b9940f195fcfc0e"  
 >  
 <label for="\_anim\_radio3\_3a4ba7a62c13475c8b9940f195fcfc0e">Reflect</label>  
</form>

## your code here  
  
## REMEMBER THAT YOU ARE WORKING ON SUPERPOSITION OF WAVES

## Implications of Superposition: Diffraction

One of the key indicators of wave like behavior is [diffraction](https://en.wikipedia.org/wiki/Diffraction). Diffraction is the bending of waves around obstacles and it is a consequence of superposition.

In the case of [single-slit diffraction](https://en.wikipedia.org/wiki/Diffraction#Single-slit_diffraction), the wave is diffracted around a slit of width . a forms circular wave fronts that interfere with each other.

|  |
| --- |
| Diagram of single slit diffraction |

Diagram of single slit diffraction

A screen placed a distance away from the slit will show a diffraction pattern. We can show the intensity of the diffraction pattern is:

where

and is the wavelength of the wave and is the angle of the diffracted wave from the center of the slit ( where is the location along the screen).

**✅ Do this**

1. Plot the intensity of the diffraction pattern for a single slit of width mm and wavelength nm.
2. Find the slit width needed to form a diffraction pattern for a typical water wave. You will need to make some estimates.
3. For atomic spacing ( m), use your plotting tool to estimate the wavelength needed to form a diffraction pattern. You will need to make some estimates.

## your code here