Emrullah Enes VAROL

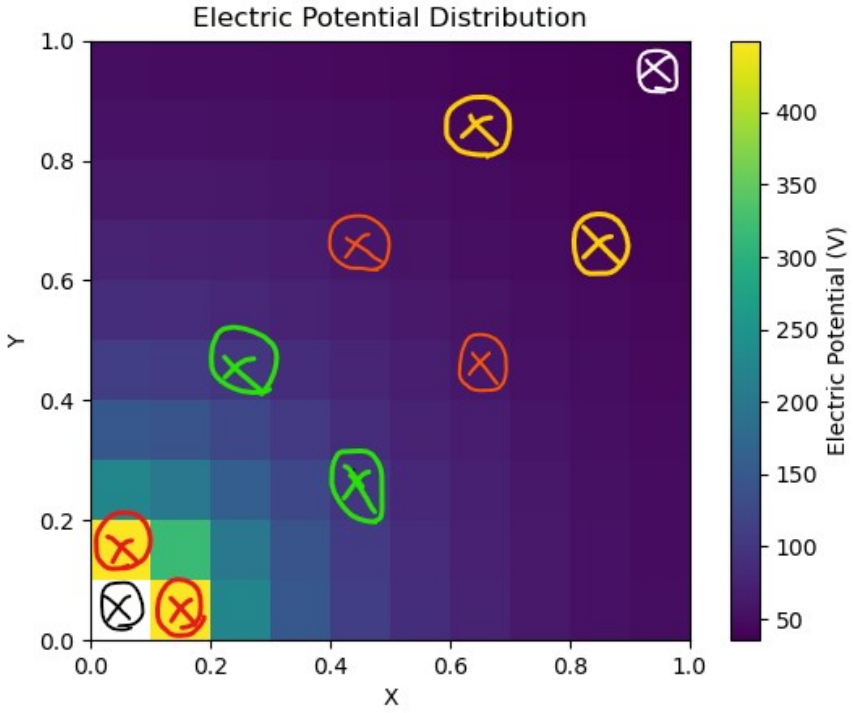
20210805065

2024 Physics 2 Homework 1

Firsly, we open the “aPart.py” to start the explanation.

```
import numpy as np
import matplotlib.pyplot as plt

# Create a 10x10 matrix
matrix = np.zeros((10, 10))

# Constants
k = 8.99e9  # Coulomb's constant in N m^2/C^2
q = 5e-9    # Charge in C

# Calculate potential for each cell
for i in range(10):
    for j in range(10):
        r = np.sqrt(i**2 + j**2) * 0.1  # Distance in meters
        matrix[i, j] = k * q / r

plt.imshow(matrix, cmap='viridis', origin='lower', extent=[0, 1, 0, 1])
plt.colorbar(label='Electric Potential (V)')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Electric Potential Distribution')
plt.show()

x_values = np.arange(0, 1, 0.1)
v_values = matrix[:, 0]  # Potential values for j=0

plt.plot(x_values, v_values)
plt.xlabel('X')
plt.ylabel('Electric Potential (V)')
plt.title('Electric Potential vs. X')
plt.grid()
plt.show()

r_values = np.sqrt(np.arange(0, 10)**2 + np.arange(0, 10)**2) * 0.1

plt.plot(r_values, np.diag(matrix))
plt.xlabel('Diagonal Distance (m)')
plt.ylabel('Electric Potential (V)')
plt.title('Electric Potential vs. Diagonal Distance')
plt.grid()
plt.show()


# In this electric potential distribution, points that have the same electric potential
value are those located on the same equipotential surface.
# An equipotential surface is a surface where the electric potential is constant.
# Electric Potential Distribution Plot:
# The first plot shows the electric potential distribution across the 10x10 matrix.
Darker regions represent higher potential values.
# The potential decreases as you move away from the central charge (located at the
origin).
# The circular contours indicate equipotential surfaces.


# The f part is here:
```

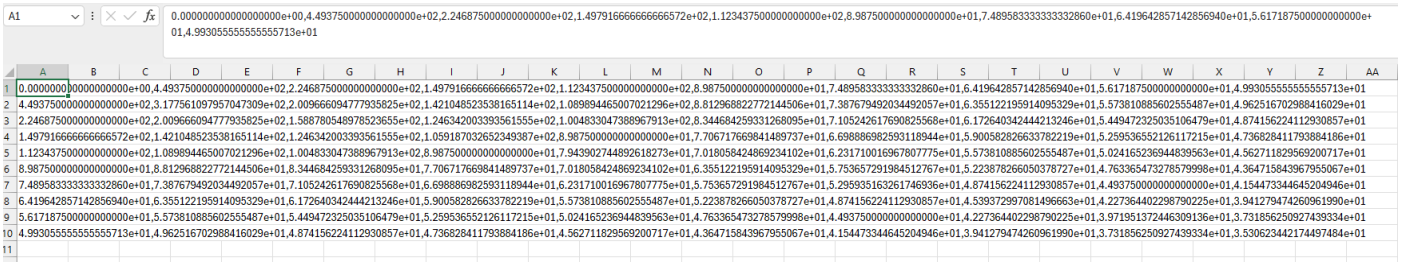
```
# By using a finer grid (100x100), the equipotential lines become smoother and more
detailed, providing a clearer representation of the potential distribution.

# Create a 100x100 matrix for a finer grid
matrix_fine = np.zeros((100, 100))

# Calculate potential for each cell in the finer grid
for i in range(100):
    for j in range(100):
        r = np.sqrt(i**2 + j**2) * 0.01  # Adjusted distance for finer grid
        if r != 0:
            matrix_fine[i, j] = k * q / r
        else:
            matrix_fine[i, j] = np.inf

# Plotting with finer grid
plt.imshow(matrix_fine, cmap='viridis', origin='lower', extent=[0, 1, 0, 1])
plt.colorbar(label='Electric Potential (V)')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Electric Potential Distribution with Finer Grid')
plt.contour(np.linspace(0, 1, 100), np.linspace(0, 1, 100), matrix_fine,
colors='white', levels=np.linspace(0, np.max(matrix_fine[matrix_fine != np.inf]), 20))
plt.show()


# Below is the code that has been modified and edited so that we can draw equipotential
lines:
# aPartF.py
with open('aPartF.py') as file:
    exec(file.read())
```



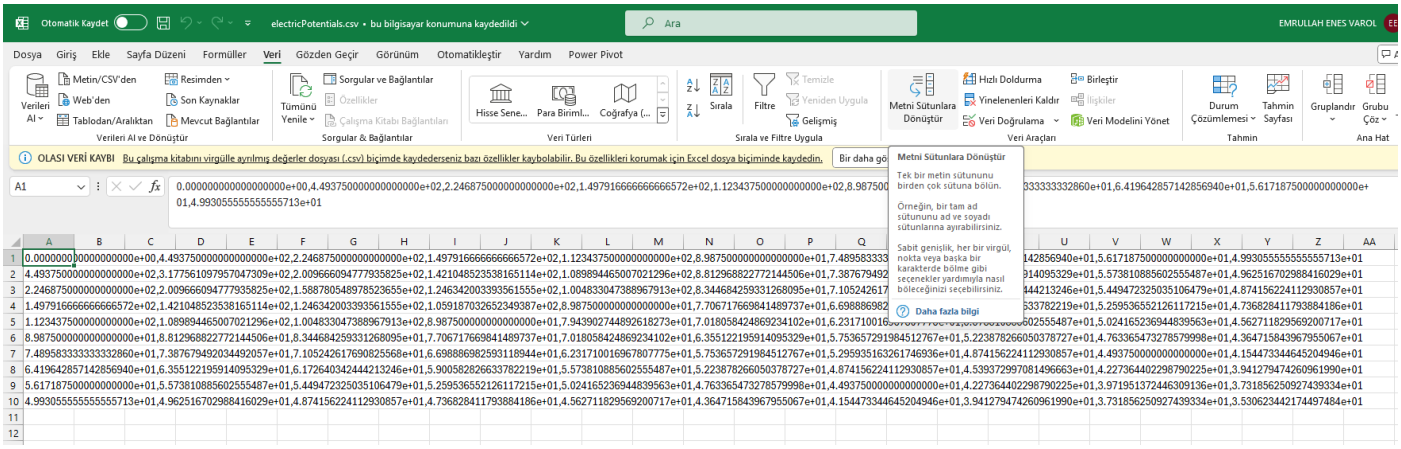
Points with equal electric potential are shown as the same colored markings on the matrix. For example, four identical points are marked: red, green, orange and yellow. Black and white markings symbolize maximum and minimum values.

If we run the “savingMatrixData.py” program, it saves matrix datas on “csv” format in same folder like “electricPotentials.csv” so we can open that file wish we want. At the same time if we run matrix datas it saves it as a new file and writes on the terminal.

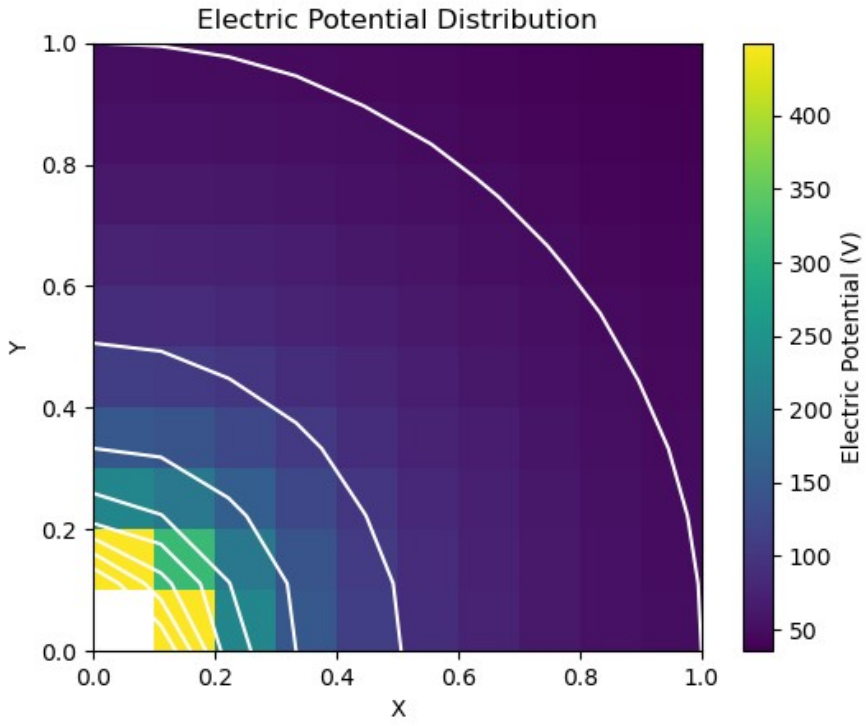
```
C:\ProgramData\anaconda3\python.exe "C:\Users\enesi\OneDrive - Akdeniz Üniversitesi\Masaüstü\PhysicsHM\firstPart\savingMatrixData.py"
[[  0.         449.375      224.6875     149.79166667 112.34375
   89.875       74.89583333  64.19642857  56.171875    49.93055556]
 [449.375      317.7561098  200.96660948 142.10485235 108.9894465
   88.12968823  73.87679492  63.55122196  55.73810886  49.62516703]
 [224.6875     200.96660948 158.8780549  124.63420034 100.48330474
   83.44684259  71.05242618  61.72640342  54.49472325  48.74156224]
 [149.79166667 142.10485235 124.63420034 105.91870327  89.875
   77.0671767   66.98886983  59.00582827  52.59536552  47.36828412]
 [112.34375    108.9894465  100.48330474  89.875       79.43902745
   70.18058425  62.31710017  55.73810886  50.24165237  45.6271183 ]
 [ 89.875      88.12968823  83.44684259  77.0671767   70.18058425
   63.55122196  57.53657292  52.23878266  47.63365473  43.64715844]
 [ 74.89583333  73.87679492  71.05242618  66.98886983  62.31710017
   57.53657292  52.95935163  48.74156224  44.9375      41.54473345]
 [ 64.19642857  63.55122196  61.72640342  59.00582827  55.73810886
   52.23878266  48.74156224  45.39372997  42.27364402  39.41279474]
 [ 56.171875    55.73810886  54.49472325  52.59536552  50.24165237
   47.63365473  44.9375      42.27364402  39.71951372  37.31856251]
 [ 49.93055556  49.62516703  48.74156224  47.36828412  45.6271183
   43.64715844  41.54473345  39.41279474  37.31856251  35.30623442]]

Process finished with exit code 0
```



To view these datas as it is shown on the terminal we can use this part of Excell application.

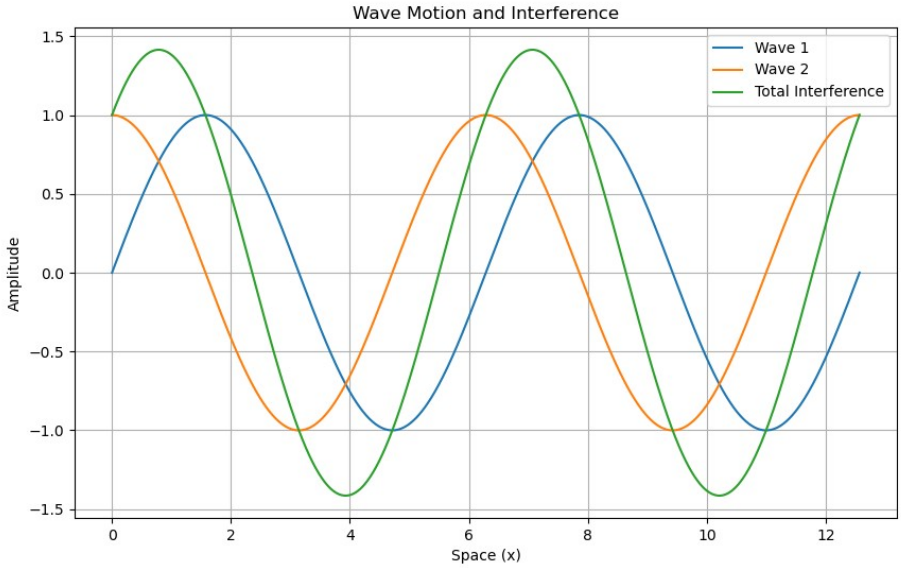


In the f part of the homework it runs “aPartF.py” as a compiler.

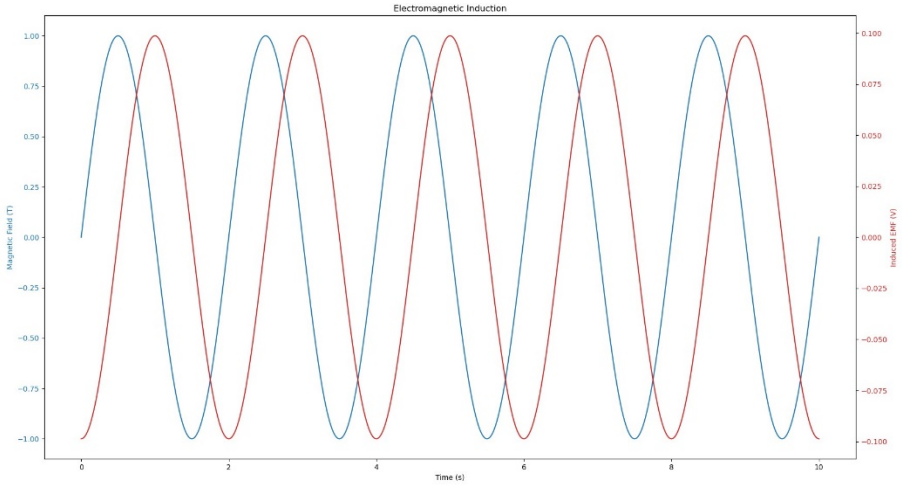


Then, when we begin the question 2. I want to do multiple simulation.

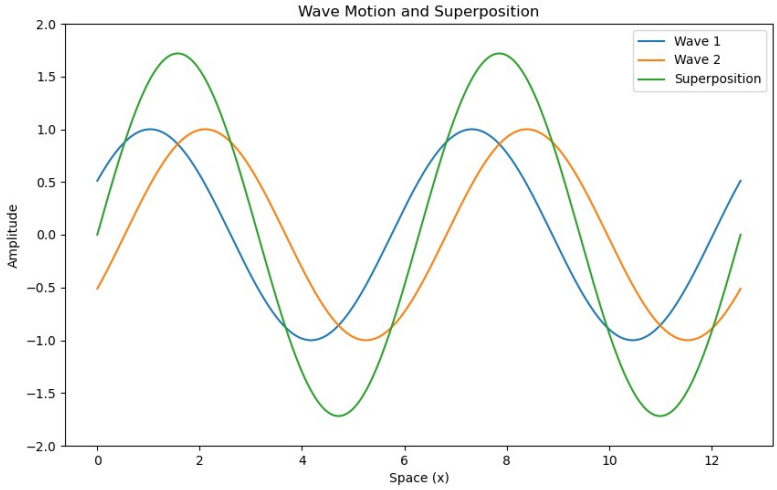
1. Wave Motion and Interference



2. Electromagnetic Induction

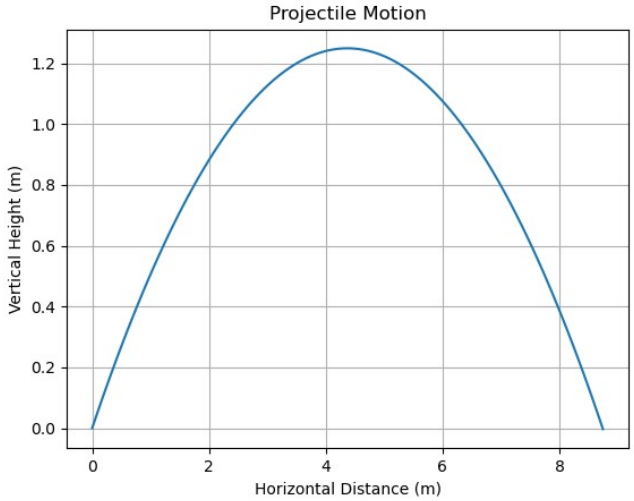


3. Wave Motion and Superposition



4. Shooting Movements

```
C:\ProgramData\anaconda3\python.exe "C:\Users\enesi\OneDrive - Akdeniz Üniversitesi\MasaüstÜ\PhysicsHM\bonusPart\1\bonusPart.py"
Enter the force at launch (Newtons): 10
Enter the launch angle (0 to 90 degrees): 30

Process finished with exit code 0
```



You can reach the codes of this homework by the following link:

https://github.com/dahaii/Physics2HM