Numerov's Method

Ben Pearson

November 17, 2021

Contents

1 The Physics

1.1 Summary of Problem

We are asked to consider a nucleon trapped in a one-dimensional squared well potential of depth $V_0 = 83MeV$ and $|x| \leq a$ where $a = 2fm$ ($1fm = 10^{-15}m$). We then wish to find the wave function for this particle and the energy of the quantum states it inhabits. We also say for this problem that $\frac{2m}{\hbar^2} = 0.04829MeV^{-1}fm^{-2}$. This project has many similarities to Transcendental Equations, however our goal with this project is to generalize this solution so we can begin considering varying wells of potential.

1.2 Analytical Solution

For our particle, we have the wave function

$$\begin{aligned}\frac{d^2\psi}{dx^2} + \frac{2m}{\hbar^2}\left(E+V_0\right)\psi = 0 \\ \frac{d^2\psi}{dx^2} + k^2(x)\psi = 0\end{aligned} \tag{1}$$

Where $k(x) = \sqrt{\frac{2m}{\hbar^2}(E+V_0)}$ Notice that 1 is of the form of a second order differential equation, missing a first order derivative. This means that we can apply Numerov's Method. For a more in depth analysis on the analytical portion of this problem, refer to Section I of Transcendental Equations. We will delve more deeply into the analytical solution here and how it might be further generalized.

2 Numerical Solution

2.1 Numerov's Method

Numerov's method is used to solve second order differential equations when it lacks a first order term - as we have showed in 1. If we have a differential equation of the form

$$\frac{d^2y}{dx^2} = -g(x)y(x) + s(x) \quad x \in [a,b] \tag{2}$$

Where $g(x)$ and $s(x)$ are continuous from a to b. If we take a look at some step size h, then we can expand $\psi(x)$ as a Taylor series

$$\begin{aligned}\psi(x+h) &= \psi(x) + h\psi^{'}(x) + \frac{h^2}{2}\psi^{''}(x) + \frac{h^3}{3!}\psi^{'''}(x) + ... \\ \psi(x-h) &= \psi(x) - h\psi^{'}(x) + \frac{h^2}{2}\psi''(x) - \frac{h^3}{3!}\psi^{'''}(x) + ... \\ \psi(x+h) + \psi(x-h) &= 2\psi(x) + h^2\psi^{''}(x) + \frac{h^4}{12}\psi^{iv}(x) + ...\end{aligned} \tag{3}$$

$$\psi^{''} = \frac{\psi(x+h) + \psi(x-h) - 2\psi(x)}{h^2} - \frac{h^2}{12}\psi^{iv}(x) + O(h^4) \tag{4}$$

Since we end up with a final term of the order of h^4, if we pick a small enough step size, those terms will drop off quickly and can be ignored. Next, we plug this back into the 1 to get

$$\frac{\psi(x+h)+\psi(x-h)-2\psi(x)}{h^2}+k^2\psi+\frac{h^2}{12}\frac{d^2}{dx^2}\left(k^2\psi\right)=0 \tag{5}$$

Now, we can take the last term of 5 and expand it out as

$$\frac{h^2}{12}\frac{d^2}{dx^2}(k^2\psi)=\frac{\left[k^2(x+h)\psi(x+h)-k^2(x)\psi(x)\right]+\left[k^2(x-h)\psi(x-h)-k^2(x)\psi(x)\right]}{h^2} \tag{6}$$

Which we can then rearrange and collect terms to get

$$\psi(x+h)=\frac{2\left[1-\frac{5}{12}h^2k^2(x)\right]\psi(x)-\left[1+\frac{h^2}{12}k^2(x-h)\right]\psi(x-h)}{\left[1+\frac{h^2}{12}k^2(x+h)\right]} \tag{7}$$

Or rather, simplifying our notation

$$\psi_{i+1}=\frac{\left[1-\frac{5}{12}h^2k_i^2\right]\psi_i-\left[1+\frac{h^2}{12}k_{i-1}^2\right]\psi_{i-1}}{1+\frac{h^2}{12}k_{i+1}^2} \tag{8}$$

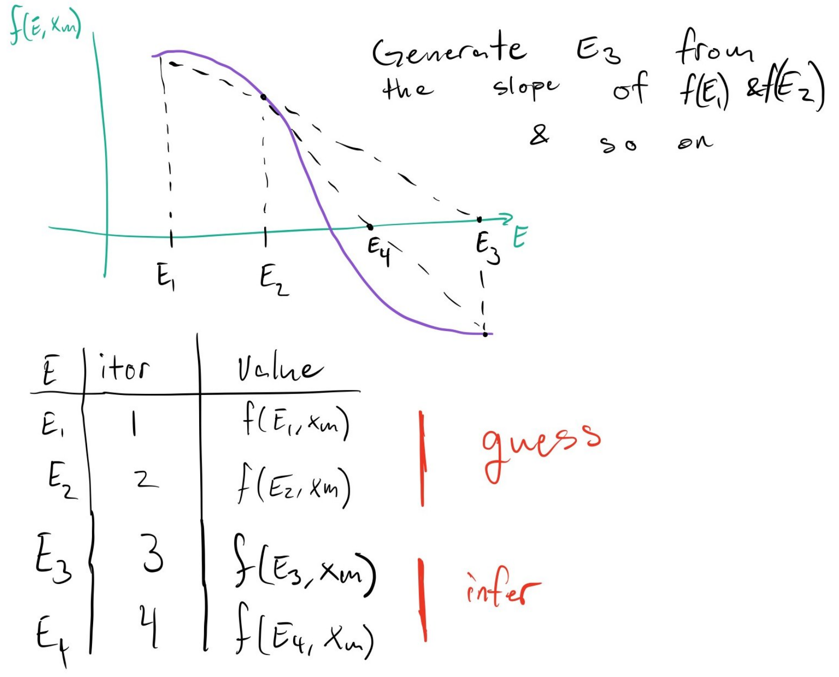
Which gives us an equation to find the following value of the Schrodinger equation. Notice that this relies upon the previous two points (that is i and $i-1$), so during our initialization phase we will have to find the first two points before beginning our calculations.

2.2 Other Methods to be Used

First, we will initialize the first two points using Euler's method as demonstrated in my Pendulum Simulation. Then, in order to find the Energy E we will guess their initial starting values and then check for convergence around some matching point in the middle - both the derivatives and the values should be identical coming from either side. In other words

$$f(E,x_m)=\frac{d\psi_l}{dx}|_{x_m}-\frac{d\psi_r}{dx}|_{x_m}=0 \tag{9}$$

Here we will be choosing a matching point of $x=\pm 2fm$ which corresponds with a. If our guessed values bracket the root, we can bisect the values until we find convergence. If they do not we use the slope between the values evaluations to generate a new Energy value - ideally this will then be on the opposite side of the root and then bracketing can begin, however if it does not we simply continue. Refer to 1 for a visual on how this process works.


$f(E, x_m)$
E
E_1
E_2
E_4
E_3
Generate E_3 from the slope of $f(E_1)$ & $f(E_2)$ & so on
guess
infer

E	itor	Value
E_1	1	$f(E_1, x_m)$
E_2	2	$f(E_2, x_m)$
E_3	3	$f(E_3, x_m)$
E_4	4	$f(E_4, x_m)$

Figure 1: A graph and table of finding the roots iteratively until bracketing is achieved.

3 Program

3.1 Flow chart

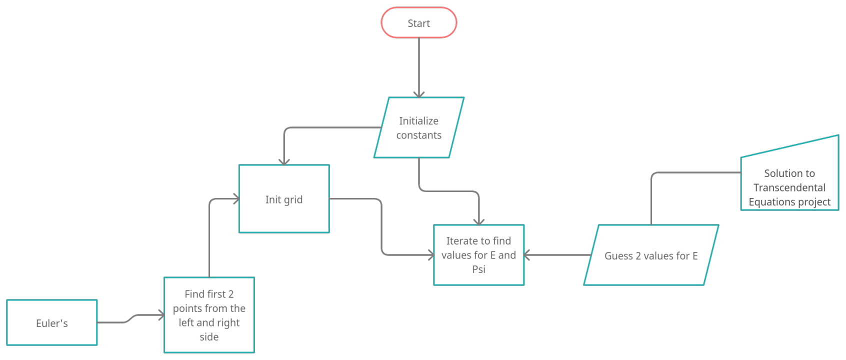

Start
Initialize constants
Init grid
Solution to Transcendental Equations project
Iterate to find values for E and Psi
Guess 2 values for E
Euler's
Find first 2 points from the left and right side

Figure 2: Flow chart of program

3.2 Numerov Method

This program was written in one file because I was interested with whether or not I would prefer it, it ended up clouding my thought process so I may rewrite this into several files for the next project.

```
#!/usr/bin/env python3
"""Iteratively find the wave funciton and energy for a particle."""

# Imports
import numpy as np
import matplotlib.pyplot as plt

# Constants
V0 = 83  # MeV
A = 2  # fm
mesh = np.linspace(
    -4, 4, 500
)  # grid is from -4fm to 4fm with 100 equally spaced pieces
C = 0.04829  # MeV^-1 fm^-2
# Only used prior to implementation of Energy finding function
E = -74.876956  # MeV
```

```
def numerov_method_negative(energy, grid):
    """Implement Numerov Method at index n to find n+1. From R -> L."""
    global A
    dx = grid[1] - grid[0]

    k = np.zeros(grid.size)

    # this will be the wave function when it is run
    val = np.zeros(grid.size)
    val[0] = 0
    val[1] = dx

    h_12 = dx ** 2 / 12

    # to simplify this, I am absorbing h**2/12 into this term
    k = []
    for i in range(grid.size):
        if abs(grid[i]) > A:
            k.append(h_12 * C * energy)
        else:
            k.append(h_12 * C * (energy + V0))

    print(k[0])

    # iterate through the array
    i = 2
    while grid[i] < - A:
        val[i] = 2 * (1 - 5 * k[i-1]) * val[i-1]\
            - (1 + k[i-2]) * val[i-2]
        val[i] = val[i]/(1 + k[i+1])
        i += 1

    val[val == 0] = np.nan
    return val


def numerov_method_positive(energy, grid):
    """Implement Numerov Method at index n to find n+1. From R -> L."""
    dx = grid[1] - grid[0]

    k = np.zeros(grid.size)

    # this will be the wave function when it is run
    val = np.zeros(grid.size)
    val[-1] = 0
    val[-2] = dx

    h_12 = dx ** 2 / 12

    # to simplify this, I am absorbing h**2/12 into this term
    k = []
    for i in range(grid.size):
        if abs(grid[i]) > A:
            k.append(h_12 * C * energy)
        else:
            k.append(h_12 * C * (energy + V0))

    print(k[0])

    # iterate through the array
    i = grid.size - 3
    while grid[i] > -A:
        val[i] = 2 * (1 - 5 * k[i+1]) * val[i+1]\
            - (1 + k[i+2]) * val[i+2]
        val[i] = val[i]/(1 + k[i-1])
        i -= 1

    val[val == 0] = np.nan
    return val



# data processing

# graphs
plt.style.use('ggplot')

fig = plt.figure(
    figsize=(9, 6),
    dpi=300
)

test_energies = [-80, -70, -74, -75]
axes = [plt.subplot(2, 2, i+1) for i in range(4)]
for i in range(4):
    ax = axes[i]
    ax.set_title(f"E = {test_energies[i]}MeV")
    psi = numerov_method_positive(test_energies[i], mesh)
    psi2 = numerov_method_negative(test_energies[i], mesh)
    ax.plot(
        mesh, psi,
        ls='none',
        ms=4,
        marker='.',
        mew=1.0,
        zorder=1,
    )
    ax.plot(
        mesh, psi2,
        ls='none',
        ms=4,
        marker='.',
        mew=1.0,
        zorder=1,
    )
    ax.set_xlabel('x', labelpad=5)
    ax.set_ylabel(r'$\Psi$', labelpad=5)
fig.tight_layout()
fig.suptitle("n = 500")
plt.savefig('500-mismatch-energies.png')
plt.show()
```

4 Analysis

Seeing varying timesteps around the correct energy doesn't do as much justice as to the effect that the time step carries. To see that we need to consider the guesses. Here we have three separate counts, 500, 100, and 10.

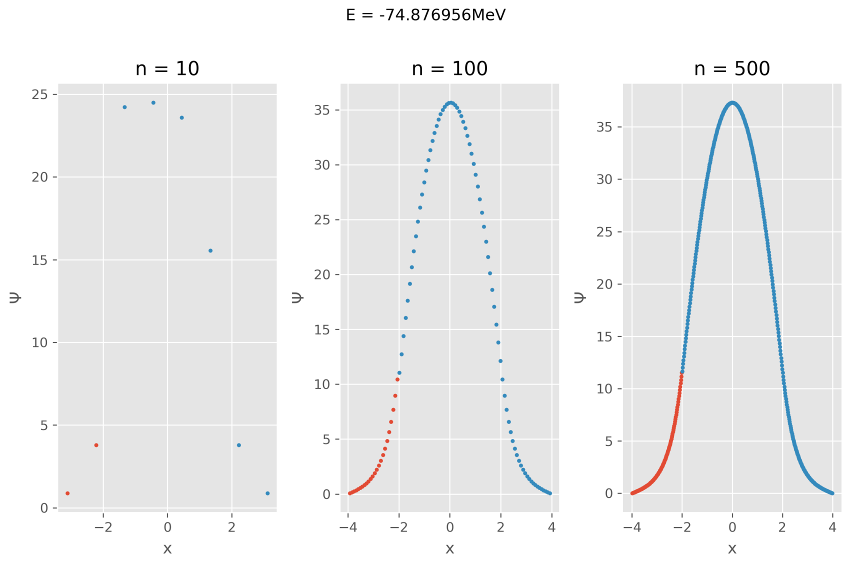


Figure 3: Correct energy across 3 different time steps

Each one is graphed across 4 different energy guesses. As is shown, the more timesteps we use not only the clearer the graph but also the easier it is to tell the accuracy. This makes larger timesteps easier to converge and easier to guess correct energies because they are clearer. However, once we surpass 100 timesteps (in this case), we find that there is little to be gained in terms of clarity, unless we desire even more accuracy.

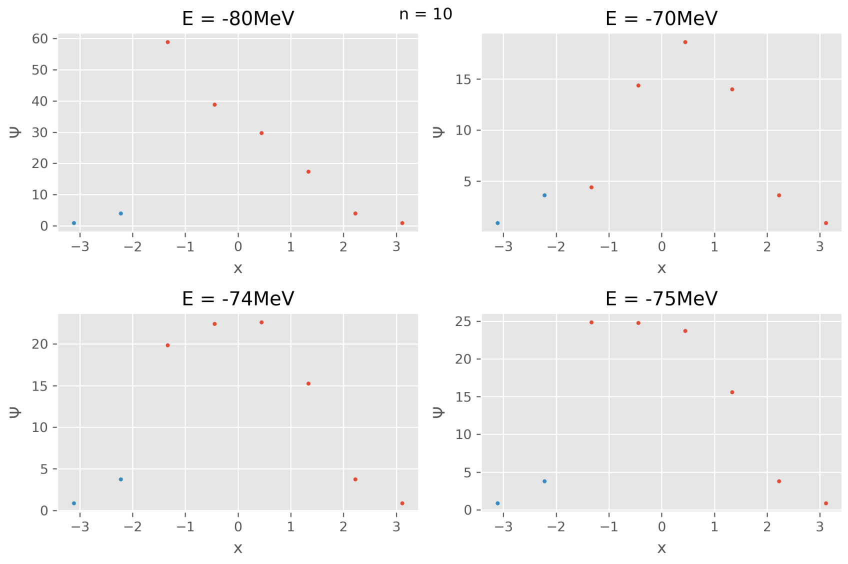


Figure 4: Energies when the space is divided 10 different times

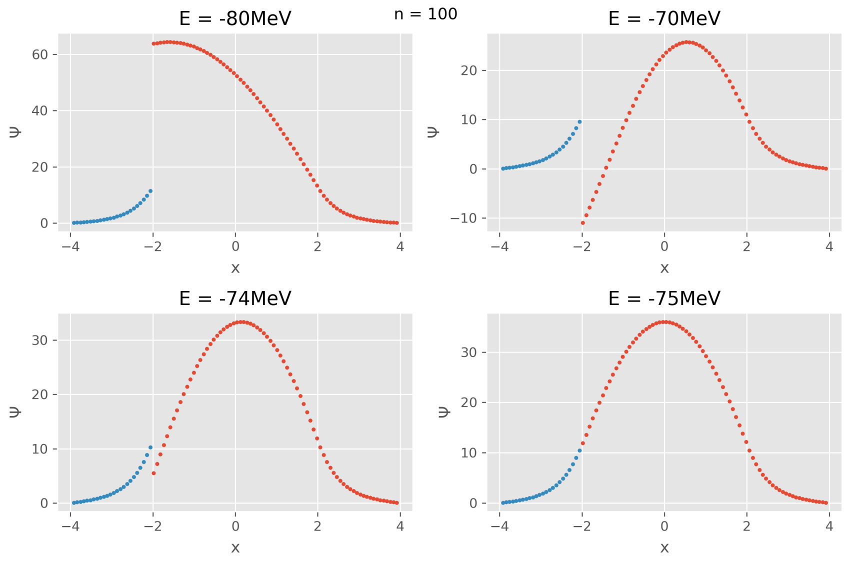


Figure 5: Energies when the space is divided 100 different times

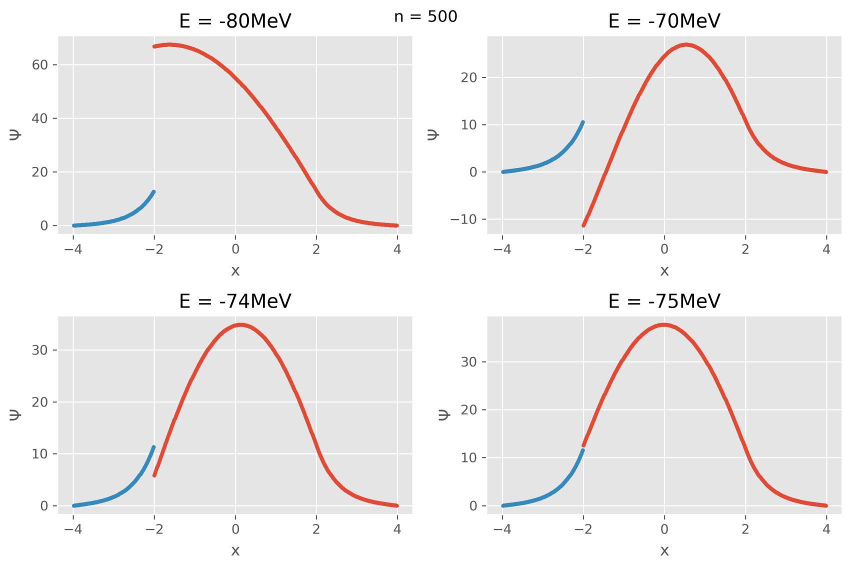


Figure 6: Energies when the space is divided 500 different times