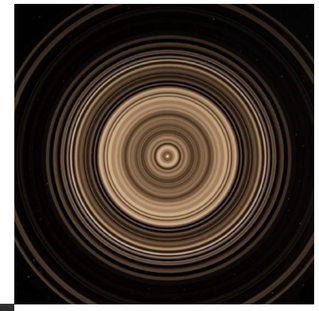
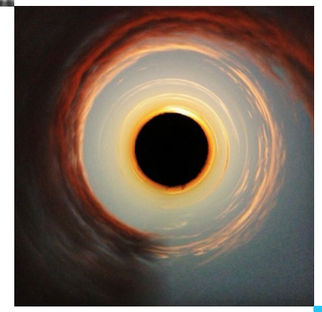
SOLVING SCHRÖDINGER'S EQUATION USING MATRIX NUMEROV METHOD IN PYTHON

PRESENTATION





Solving

SCHRÖDINGER'S EQUATION

Schrödinger equation is a fundamental equation in quantum mechanics that describes the behavior of quantum systems, such as atoms and molecules. It is named after Erwin Schrodinger, who developed the equation in 1925.

$$-\frac{\hbar^2}{2m}\frac{d^2}{dx^2}\psi(x) + V(x)\psi(x) = E\psi(x)$$

WHY THIS PROJECT

The Schrödinger equation being a second order differential function, it's not always possible to find an analytical solution. Hence we employ numerical methods that involves high level of computation to solve the Schrödinger Equation.

This has already been achieved by Mohandas Pillai, Joshua Goglio, and Thad G. Walkera (Department of Physics, University of Wisconsin-Madison) using Mathematica and they published it in American Journal of Physics.

Wolfram Mathematica is a premium paid software system with built-in libraries for several areas of technical computing. Higher cost of Mathematica makes it inaccessible to a student.Hence alternate choice is always preferred.

The aim of this project is to make a python code to plot the Stationary States of a quantum system in the presence of a linear potential.

NUMEROV METHOD

- more complex
- require more lines
- more accurate

MATRIX NUMEROV METHOD

- Simple
- few lines of code
- less accurate

If the programming is too difficult or time consuming the focus shifts from the physics to the programming.

Therefore, we prefer the Matrix Numerov method.

SOLVING SCHRODINGER'S EQUATION

DIFFERENT METHODS

NUMERICAL METHODS

1 Finite Difference Method

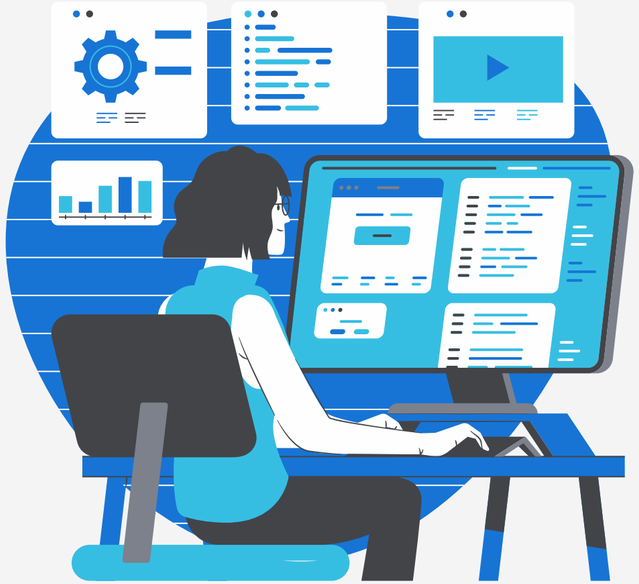
This method involves approximating the differential equation by discrete differences, and then solving the resulting algebraic equations using matrix algebra.

2 Numerov method

This is a variant of the finite difference method that is particularly well-suited for solving second-order differential equations like the Schrodinger equation.

3 Shooting method

In this method, the Schrodinger equation is transformed into a system of first-order differential equations, which can then be solved using standard numerical methods like the Runge-Kutta method.



ADVANTAGES AND DISADVANTAGES

SEPARATION METHOD

- **Advantage**: it is simple, direct, easy to understand, and easy to solve.
- **Disadvantage**: this is not always the case in complex problems like the boundary value. The separation of variables method becomes convoluted and hard to solve in such instances.

PERTURBATION METHOD

Advantage: perturbation theory is that they provide analytical formulas that describe the influence and role of system parameters on the response.

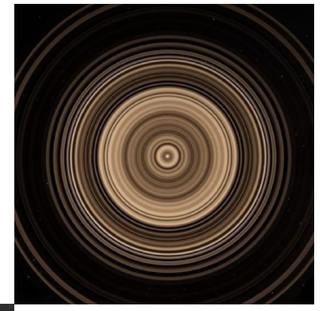
Disadvantage: they are limited to relatively narrow, but important, classes of mathematical problems.

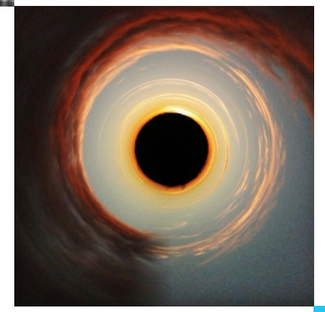
SHOOTING METHOD

Advantage: it takes advantage of the speed and adaptivity of methods for initial value problems.

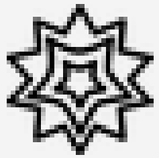
Disadvantage: it is not as robust as finite difference or collocation methods

WHY PYTHON OVER MATHEMATICA?





MATHEMATICA



Wolfram Mathematica is a versatile and powerful computational system that provides a wide range of functionality for solving complex problems in various domains. Its interactive notebook interface and rich library of functions make it a popular tool for researchers, educators, and practitioners alike.

PYTHON

Python is a high level programming lang and gained popularity in recent years. Python has become an essential tool for scientific computing due to its simplicity, versatility, and broad support for scientific libraries and tools. Its popularity among researchers and data scientists is expected to continue to grow in the future, as more and more organizations adopt Python for their scientific computing

WHY PYTHON OVER MATHEMATICA

PROPERTIES	PYTHON	MATHEMATICA
. EASE OF USE	Simpler and more straightforward	steeper learning curve
. FUNCTIONALITY	Vast array of third party libraries and modules	broad range of built-in functions and libraries
. VISUALIZATION	External module	In built
. COMMUNITY	Large & active	Closed-source nature can limit its accessibility
. PRICE	Free	Premium software with a high price tag

MATRIX NUMEROV METHOD

DERIVATION

$$\frac{-\hbar^2}{2m}\frac{\partial^2 \psi}{\partial x^2} + V\,\psi = E\,\psi$$

$$\hat{H}\,\psi = E\,\psi$$

Where $\hat{H} = \frac{-\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V$

$$A\,\mathbf{x} = \lambda\,\mathbf{x} \qquad \mathbf{x} = eig(A)$$

$$\hat{H}\,\psi = E\,\psi \qquad \psi = eig(H)$$

$$\frac{\partial^2 \psi}{\partial x^2} = \frac{-2m}{\hbar^2}[E - V(x)]\,\psi(x)$$

$$\psi^{(2)}x = f(x)\,\psi(x)$$

$$\frac{-\hbar^2}{2m}\frac{\psi_{i-1} - 2\psi_i + \psi_{i+1}}{d^2} + \frac{V_{i-1}\psi_{i-1} + 10V_i\psi_i + V_{i+1}\psi_{i+1}}{12} = E\frac{\psi_{i-1} + 10\psi_i + \psi_{i+1}}{12}$$

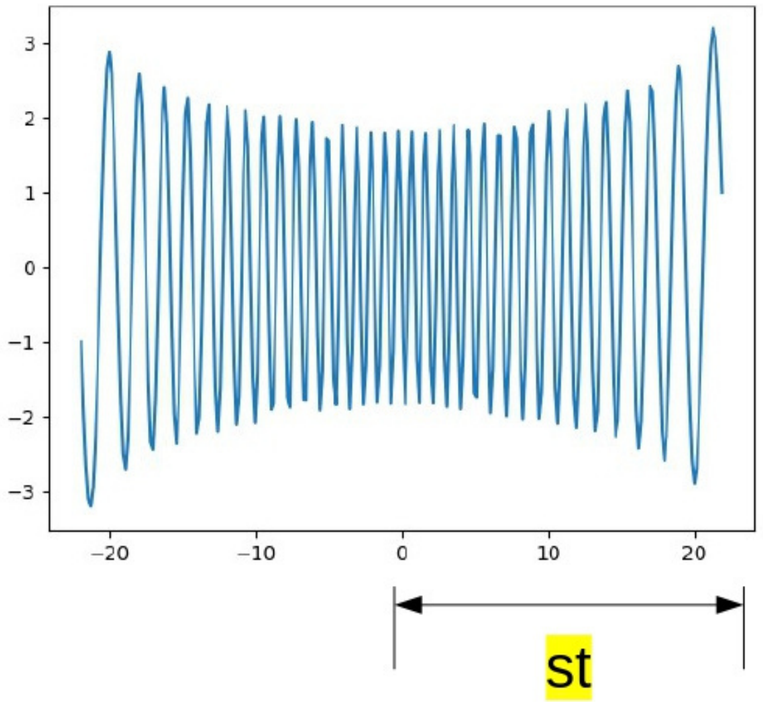
$$\begin{pmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{pmatrix}\begin{pmatrix} \psi_{i-1} \\ \psi_i \\ \psi_{i+1} \end{pmatrix}$$

$$\frac{-\hbar^2}{2m}A\,\psi + BV\,\psi = E\,B\,\psi$$

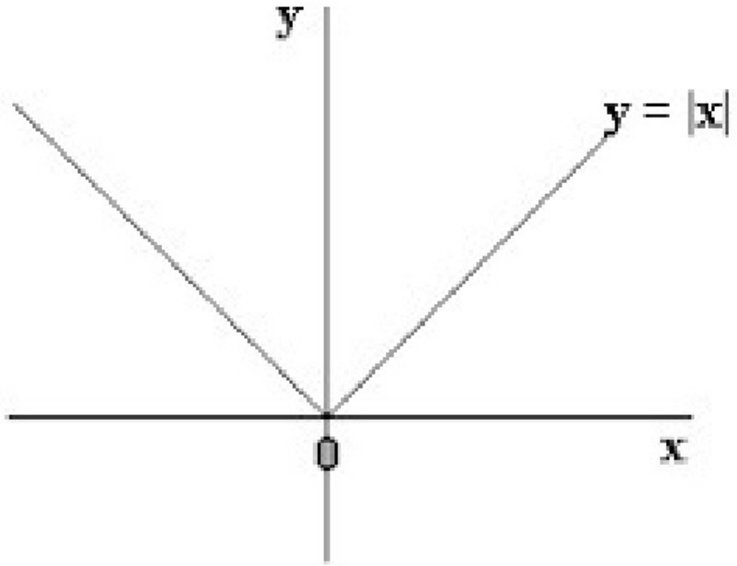
$$\frac{-\hbar^2}{2m}B^{-1}A\,\psi + V\,\psi = E\,\psi$$

$$\begin{pmatrix} 10 & 1 & 0 \\ 1 & 10 & 1 \\ 0 & 1 & 10 \end{pmatrix}\begin{pmatrix} \psi_{i-1} \\ \psi_i \\ \psi_{i+1} \end{pmatrix}$$

Determining the grid



$$V(x)=|x|$$



$$V(x_{max})=E$$

$$V(st)=em$$

$$st=fsolve(V,em)$$

Determining the grid

$$\lambda = \frac{h}{P} \qquad E = \frac{P^2}{2m}$$

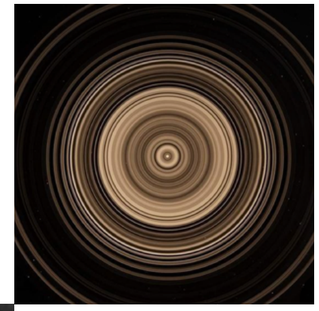
$$\lambda = \frac{h}{\sqrt{2Em}} \qquad P = \sqrt{2Em}$$

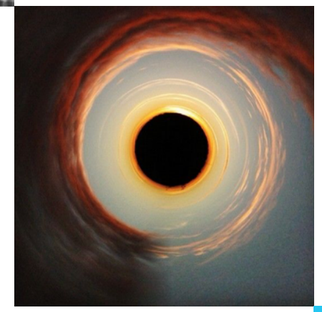
$$ds = \frac{1}{\sqrt{2em}}$$

$$n = 2\left(\frac{st}{ds} + 4\pi\right)$$

ALGORITHM AND CODE

PYTHON CODE





Algorithm

DEFINE THE POTENTIAL FUNCTION

Define Potential function and Energy value

DETERMINE THE GRID

Calculate the number of gridpoints needed and set the x axis

CALCULATE KE MATRIX

Contruct the matrces A and B and find the KE Energy operator

CALCULATE HAMILTONIAN AND EIGENVECTOR

Convert the potential to a matrix and add the calculated KE matrix to get the Hamilton matrix. Find the Eigenvector to get the wavefunction

TRANSPOSE AND PLOT THE GRAPH

Transpose the eigen vector matrix
associated with the given Energy value in the Y-axis.

Code

1 Import Libraries and functions

```
from math import sqrt,pi
from scipy.optimize import fsolve
import numpy as np
from numpy.linalg import eig
from numpy.linalg import inv
import matplotlib.pyplot as plt
```

Code

2 Define the Potential function and Energy value

```
#Define the potential and Energy
def v(s): return abs(s)
em = 20.
```

Code

3 Determine the grid

```
#determine grid
def f(y): return abs(y)-em
st = float(fsolve(f,em)) #findroot
ds= 1/(sqrt(2*em))
n = int(2*((st/ds)+4*pi))
s = [(-ds*(n+1)/2)+ds*i for i in range(1,n+1)]
```

Code

4 Calculate KE Matrix

```
#calculate KE matrix
def one(n,d): return np.diagflat([1 for i in range (n-abs(d))], d)
A = (one(n,-1)- 2*one(n,0) +one(n,1))/ds**2
B = (one(n,-1)+ 10*one(n,0) +one(n,1))/12.
Bi = inv(B)
K= np.matmul(A, -Bi)/2
```

Code

5 Calculate Hamiltonian matrix

```
#Hamiltonian
V= np.diagflat([v(s[i]) for i in range(n)])
H= V+K
eval, evec = eig(H)
```

Code

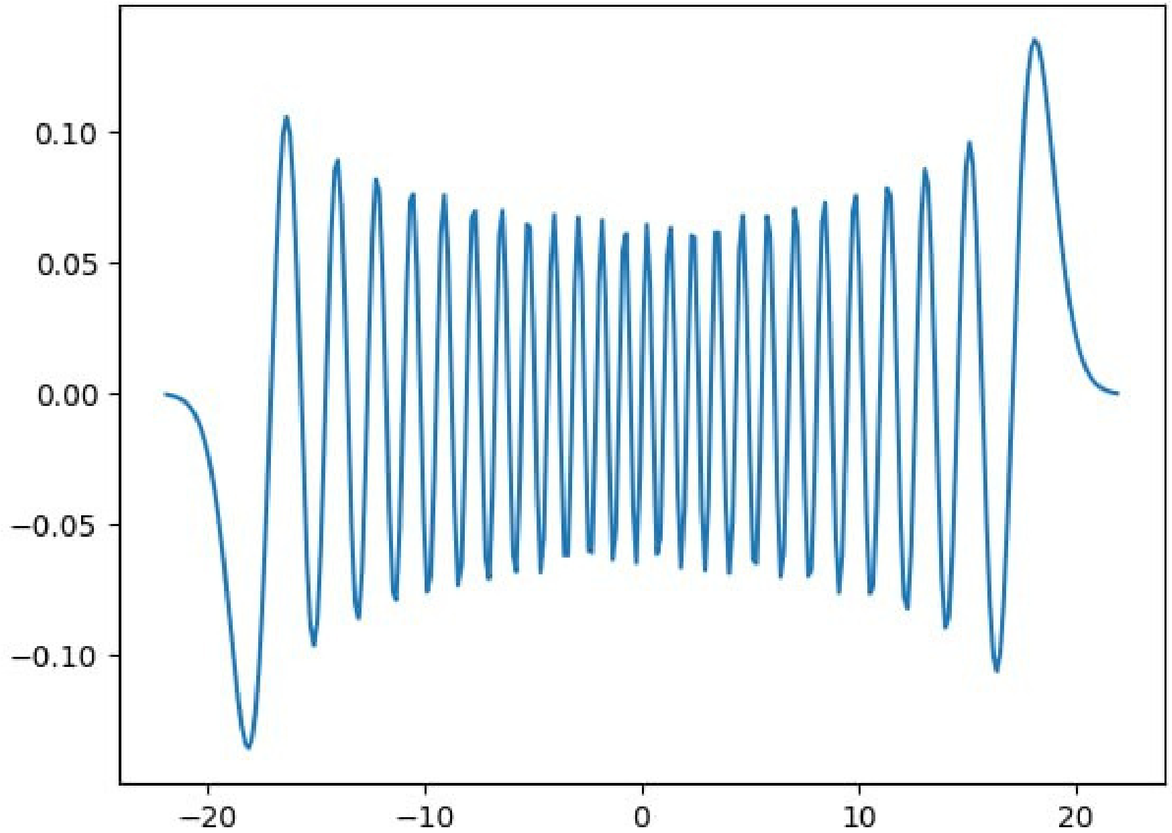
6 Show the plot

```
#Show the plot
plt.plot(s,evec[-20])
plt.show()
```

PYTHON CODE

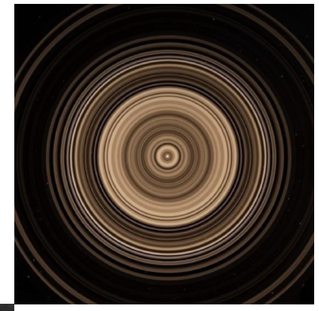
```
from math import sqrt,pi
from scipy.optimize import fsolve
import numpy as np
from numpy.linalg import eig
from numpy.linalg import inv
import matplotlib.pyplot as plt


#Define the potential and Energy
def v(s): return abs(s)
em = 20.

#determine grid
def f(y): return abs(y)-em
st = float(fsolve(f,em)) #findroot
ds= 1/(sqrt(2*em))
n = int(2*((st/ds)+4*pi))
s = [(-ds*(n+1)/2)+ds*i for i in range(1,n+1)]

#calculate KE matrix
def one(n,d): return np.diagflat([1 for i in range (n-abs(d))], d)
A = (one(n,-1)- 2*one(n,0) +one(n,1))/ds**2
B = (one(n,-1)+ 10*one(n,0) +one(n,1))/12.
Bi = inv(B)
K= np.matmul(-Bi, A)/2

#Hamiltonian
V= np.diagflat([v(s[i]) for i in range(n)])
H= V+K
eval, evec = eig(H)

#Denormalize the vector

evec = np.transpose(evec)

#Show the plot
plt.plot(s,evec[-20])
plt.show()
```

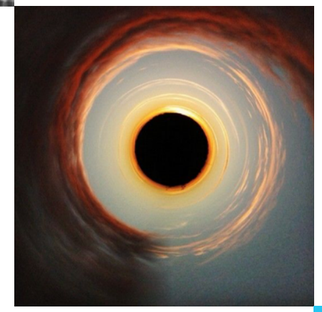
RESULT GRAPH



RESULT AND CONCLUSION

OUR FINDINGS





CONCLUSION

Schrödinger Equation is the central equation of Modern Physics and the foundation of Quantum Mechanics.

The equation being so complex and yet so elegant has been applied to solve breakthrough phenomenons of the Physical world like Quantum Tunneling and Quantum Entanglement.

The ability to plot it's complex wave functions in a programming language that is so simple and free can create a much easier opportunity for students around the world to get into the world of Quatumn Physics.

Python code for plotting the wave function in the Schrödinger Equation can be modified to fit various potential functions such as Simple Harmonic Oscillator or Particle in a Box problems. By adding higher-order terms from the Matrix Numerov Method, more accurate results can be obtained.