Numerical Solution of the Schrödinger Equation Using the RK4 Method

1. Introduction

The Python code provided numerically solves the one-dimensional time-independent Schrödinger equation using the fourth-order Runge–Kutta (RK4) integration method. The potential chosen is the harmonic oscillator, which allows us to illustrate how numerical wavefunctions are computed for different trial energies.

2. Schrödinger Equation

The time-independent Schrödinger equation in one dimension is

$$-\frac{1}{2}\frac{d^2\psi(x)}{dx^2} + V(x)\psi(x) = E\psi(x), \tag{1}$$

where we have chosen units such that $\hbar = m = 1$.

Rearranging, we obtain a second-order ordinary differential equation (ODE):

$$\frac{d^2\psi(x)}{dx^2} = -2\left(E - V(x)\right)\psi(x). \tag{2}$$

In the program, the harmonic oscillator potential is used:

$$V(x) = \frac{1}{2}x^2. \tag{3}$$

3. Reduction to a First-Order System

To apply RK4, the second-order ODE must be rewritten as a system of two first-order equations. Define

$$\phi(x) = \psi(x), \qquad \phi'(x) = \frac{d\phi}{dx}. \tag{4}$$

Then we have

$$\frac{d\phi}{dx} = \phi', \tag{5}$$

$$\frac{d\phi'}{dx} = -2\left(E - V(x)\right)\phi. \tag{6}$$

These two equations form the system that RK4 evolves at each step.

4. Boundary Conditions

The code allows two types of boundary conditions at $x = 0$, corresponding to the parity of the wavefunction:

- **Even (symmetric) solutions:**
$$\phi(0) = 1, \qquad \phi'(0) = 0,$$

- **Odd (antisymmetric) solutions:**
$$\phi(0) = 0, \qquad \phi'(0) = 1.$$

The parameter `p` in the program selects which one is used.

5. Fourth-Order Runge–Kutta Method

For a step size h, define the four RK4 increments as follows:

$$k_1 = \phi', \qquad p_1 = -2\,(E - V(x))\,\phi,$$

$$k_2 = \phi' + \frac{h}{2}p_1, \qquad p_2 = -2\left(E - V(x + \tfrac{h}{2})\right)\left(\phi + \frac{h}{2}k_1\right),$$

$$k_3 = \phi' + \frac{h}{2}p_2, \qquad p_3 = -2\left(E - V(x + \tfrac{h}{2})\right)\left(\phi + \frac{h}{2}k_2\right),$$

$$k_4 = \phi' + hp_3, \qquad p_4 = -2\,(E - V(x + h))\,(\phi + hk_3)\,.$$

The RK4 update formulas become

$$\phi(x + h) = \phi(x) + \frac{h}{6}\,(k_1 + 2k_2 + 2k_3 + k_4)\,, \tag{7}$$

$$\phi'(x + h) = \phi'(x) + \frac{h}{6}\,(p_1 + 2p_2 + 2p_3 + p_4)\,. \tag{8}$$

These are exactly the recurrence relations implemented in the Python code.

6. Interpretation of Results

The code integrates from $x = 0$ to a user-defined maximum value $x_{\max}$. For a chosen trial energy E, it generates the numerical wavefunction $\phi(x)$ and plots it.

Only when E is close to a true energy eigenvalue

$$E_n = n + \frac{1}{2}, \qquad n = 0, 1, 2, \ldots \tag{9}$$

will the numerical solution remain finite. Otherwise the wavefunction diverges, indicating that the trial energy is not an eigenvalue.

7. Summary

This numerical approach illustrates how the Schrödinger equation can be solved by reformulating it into first-order ODEs and applying the RK4 method. The produced wavefunctions depend strongly on the choice of energy, enabling the shooting method to determine quantum eigenvalues.