Aim:

To solve and visualize the Time-Dependent Schrödinger Equation using two numerical methods and to compare their results through animation.

Objective:

1. Implement the Finite Difference Method and Eigenstate Evolution to solve the Schrödinger Equation.
2. Analyze and visualize the wave function evolution over time.
3. Compare the accuracy and efficiency of the two numerical techniques.

Summary:

This project focuses on solving the Time-Dependent Schrödinger Equation in a one-dimensional potential well using two distinct numerical methods: the Finite Difference Method and Eigenstate Evolution. The project involves discretizing the Schrödinger Equation, computing the wave function at successive time steps, and comparing the results of both methods. The final output is an animation that illustrates the evolution of the wave function over time, highlighting the equivalence of the two approaches.

Tools and Libraries Used:

- **Jupyter Lab**: Integrated development environment.
- **NumPy**: Numerical computations.
- **Matplotlib**: Data visualization and animation.
- **Numba**: Just-In-Time compilation to accelerate the Finite Difference Method.
- **SciPy**: Linear algebra operations, specifically for eigenvalue computation.

Procedure:

1. **Setup and Imports**: Import necessary libraries like NumPy, Matplotlib, Numba, and SciPy.
2. **Define Parameters**: Set up the spatial and temporal grids, initial wave function, and potential energy.
3. **Finite Difference Method**:
   - Discretize the Schrödinger Equation using a finite difference approach.
   - Implement the algorithm to compute the wave function at each time step using Numba for acceleration.
4. **Eigenstate Evolution**:
   - Compute eigenstates and eigen-energies by solving the time-independent Schrödinger Equation.
   - Use these to reconstruct the time-dependent wave function.

5. **Visualization**:
    - Create a Matplotlib plot to compare the wave function from both methods at a specific time.
    - Develop an animation that shows the evolution of the wave function over time using both techniques.
6. **Save Animation**: Save the resulting animation as a GIF.

Highlights:

- **Numba Optimization**: Accelerates the Finite Difference Method, making the computation feasible for a large number of time steps.
- **Comparative Analysis**: Demonstrates the equivalence of two numerical methods for solving the Schrödinger Equation.
- **Visualization**: The animation effectively illustrates the wave function evolution, providing a clear comparison between methods.

Conclusion:

The project successfully demonstrates the use of two numerical techniques to solve the Time-Dependent Schrödinger Equation. Both methods yield consistent results, as evidenced by the animation. The Finite Difference Method, optimized with Numba, proves efficient for real-time computation, while the Eigenstate Evolution method offers a robust alternative. The animation serves as a valuable tool for visualizing quantum dynamics, enhancing understanding of wave function behavior in quantum systems.