Quantum Tunneling Simulation Project README

1 Overview

This repository contains a full numerical simulation of quantum tunneling in one and two dimensions using Python. The project visualizes a Gaussian wavepacket encountering a potential barrier and demonstrates tunneling, reflection, and transmission phenomena using:

- Eigen-decomposition of the Hamiltonian (1D)
- Split-step Fourier method (2D)
- Matplotlib animations for visualization

The code is intended for students, researchers, and educators working with computational quantum mechanics.

2 Features

- Simulates 1D Gaussian wavepacket tunneling through a rectangular barrier.
- Computes reflection and transmission probabilities numerically.
- Sweeps barrier height and width to generate tunneling curves.
- Includes a 2D tunneling demo using FFT-based propagation.
- Generates animations directly in Jupyter Notebook.

3 Repository Structure

- `main.ipynb` – Notebook containing simulations, plots, and animations.
- `GaussianWavePacket` class – Handles 1D Hamiltonian construction and evolution.
- Parameter sweep utilities – Functions for computing transmission vs. barrier parameters.
- 2D simulation module – Split-step Fourier method implementation.

4 Requirements

Ensure the following Python packages are installed:

- NumPy
- Matplotlib
- IPython (for animation display)
- FFmpeg (optional, for video export)

Install the required packages using:

```
pip install numpy matplotlib
```

If animations fail due to missing FFmpeg, install it from:

- Windows: `https://www.gyan.dev/ffmpeg/`
- Linux: `sudo apt install ffmpeg`
- Mac: `brew install ffmpeg`

5 Running the Simulation

Open `main.ipynb` in Jupyter Notebook:

```
jupyter notebook main.ipynb
```

Each section of the notebook is organized as:

1. Define and simulate the 1D wavepacket.
2. Compute transmission and reflection numerically.
3. Sweep barrier parameters.
4. Run the 2D tunneling demo.
5. Display animations.

6 Usage Examples

6.1 1D Simulation

```
wavepacket = GaussianWavePacket(
    num_grid_intervals=750,
    domain_length=750,
    barrier_start=420,
    barrier_height=0.1,
    barrier_width=50,
    initial_center=100,
    initial_momentum=0.4,
    sigma_width=15,
    time_array=np.linspace(0, 2500, 1000)
)
ani = wavepacket.animation()
HTML(ani.to_jshtml())
```

6.2 Compute Transmission

```
transmission = compute_transmission(bheight=0.1, bwidth=20)
print(transmission)
```

6.3 2D Simulation

```
ani2d = run_2d_tunneling()
HTML(ani2d.to_jshtml())
```

7 Troubleshooting

Animation does not play:

- Use `to_jshtml()` instead of `to_html5_video()`.
- Install FFmpeg if saving to MP4.

Simulation runs slowly:

- Reduce the grid size.
- Reduce the number of animation frames.
- Use NumPy vectorization where possible.

8 License

This project may be freely used for educational and research purposes.

9 Citation

If you use this repository in academic work, please cite:

```
Your Name, Quantum Tunneling Simulation Project (2025)
```