`Note file for Task 2`

This project simulates the time evolution of a single-particle Gaussian wave packet on a tight-binding chain with a gradient field. We use numerical methods to solve the time-dependent Schrödinger equation on a one-dimensional lattice, by which we can finally observe the core features of Bloch oscillation.

1. Basic Principle

The simulation models a particle confined to a one-dimensional tight-binding lattice, subjected to a linear potential. The system's dynamics are governed by the time-dependent Schrödinger equation, $i\hbar\frac{\partial}{\partial t}|\psi(t)\rangle = H|\psi(t)\rangle$, where H is the system's Hamiltonian.

The Hamiltonian is composed of two parts:

- A **hopping term** (`-t_0`) that allows the particle to tunnel between neighboring lattice sites.
- A **linear potential** (`F*j`) representing a constant external force, which creates a potential difference across the lattice sites.

The total Hamiltonian is given by:

$$H = -\sum_{j=1}^{N-1}(|j\rangle\langle j+1| + |j+1\rangle\langle j|) + F\sum_{j=1}^{N} j\cdot|j\rangle\langle j|$$

The simulation proceeds in three main steps:

1. **Diagonalization**: The Hamiltonian matrix `H_matrix` is constructed and numerically diagonalized to find its eigenvalues and eigenvectors. The eigenvectors form a complete basis, which allows us to represent any quantum state as a linear combination of these eigenstates.
2. **Time Evolution**: The initial wave packet, `psi0` , is projected onto the eigenbasis. The time evolution of each component is then calculated analytically using the time evolution operator e^{-iE_nt}.
3. **Reconstruction**: At each time step, the time-evolved components are recombined to reconstruct the wave packet in the real-space basis. The probability density, $|\psi(j,t)|^2$, is then calculated as the absolute square of the wave function.

2. Source Code Analysis (`task2.py`)

(1) Hamiltonian Construction & Eigenvalues Computation

The script builds the Hamiltonian as a tridiagonal matrix. The diagonal terms `F*j` are constructed using `np.arange(1, N+1)` for `j` to ensure a consistent 1-based indexing system across the simulation, while (`-np.ones`) on the off-diagonal terms represents the hopping term in the tight-binding model.

(2) Time Evolution & Probability Density

- **Initial State (`psi0`)**: The code computes and normalizes the initial Gaussian wave packet `psi0` to ensure the total probability is 1.
- **Eigenbasis Projection**: The `psi0` state is projected onto the eigenbasis of the Hamiltonian using matrix multiplication (`np.matmul`), yielding the initial coefficients `c_n0` .
- **Time Propagation**: The time evolution is performed in a loop, where the coefficients `c_n0` are multiplied by the analytical time evolution factor `np.exp(-1j * eigenvalues * t)` .
- **Reconstruction**: The time-evolved state is transformed back to the real-space basis, and the probability density is calculated.
- **Results**: The script extracts the probability density values at a specific time (t=42) and for specific lattice sites (j=10, 20, 30, 40, 50).

(3) Visualization

A heatmap is generated using `matplotlib.pyplot.imshow()` . This visualization displays the probability density as a function of both time (x-axis) and position (y-axis). The `jet` colormap is used, where warmer colors (red) represent higher probability density and cooler colors (blue) represent lower density.

3. Simulation Results

The final output values and the generated heatmap accurately represent the physical behavior of a quantum particle under the specified conditions:

- **Eigenvalues**: The calculated eigenvalues represent the discrete energy levels of the system under the given potential.
- **Wave Packet Cohesion**: The wave packet remains localized during its motion, demonstrating the effect of the external force on its momentum distribution.
- **Bloch Oscillation Period**: The heatmap shows the wave packet oscillating back and forth across the lattice, while the period of this oscillation is determined by the external force `F` .