VI. TIME-DEPENDENT SCHRÖDINGER EQUATION

Physics 580, SDSU. Due 11:59 pm, Friday, December 9, 2011

In this project, you will solve the time-dependent Schrödinger equation,

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\Psi(x,t) + V(x)\Psi(x,t), \tag{14}$$

using the Crank-Nicolson method. First, we introduce the Hamiltonian operator,

$$\hat{\mathcal{H}} = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x),$$

which is the same as we had in Project 4. Then, the Schrödinger equation can be re-written as

$$\frac{\partial}{\partial t}\Psi(x,t) = -\frac{i}{\hbar}\hat{\mathcal{H}}\Psi(x,t).$$

Next, we discretize $\Psi(x,t)$ as a vector, $\Psi(x,t_n) \rightarrow \vec{\psi}^{(n)}$, where the components of the vector obey $\Psi(x_j,t_n) = \psi_j^{(n)}$. In this discretization of the wavefunction, we can similarly discretize the Hamiltonian operator as a matrix, exactly as we did in Project 4 (in fact, since you wrote modular code in doing Project 4, it should be straight-forward for you to reuse code from Project 4 here).

The Crank-Nicolson method is to split up the Hamiltonian, half explicit and half implicit:

$$\frac{\vec{\psi}^{(n+1)} - \vec{\psi}^{(n)}}{\Delta t} = \frac{1}{2}\left(-\frac{i}{\hbar}\hat{\mathcal{H}}\vec{\psi}^{(n+1)} - \frac{i}{\hbar}\hat{\mathcal{H}}\vec{\psi}^{(n)}\right),$$

or rewritten in the half-explicit/half-implicit form

$$\left(1 + \frac{i\Delta t}{2\hbar}\hat{\mathcal{H}}\right)\vec{\psi}^{(n+1)} = \left(1 - \frac{i\Delta t}{2\hbar}\hat{\mathcal{H}}\right)\vec{\psi}^{(n)}. \tag{15}$$

The problem requires the use of complex variables. One can either code the problem using complex variables, or by recasting the problem as a purely real problem. Below is outlined the real variable approach, though either approach is acceptable.

Rewrite the wavefunction vector as the sum of a real part and an imaginary part,

$$\vec{\psi}^{(n)} = \mathrm{Re}\vec{\psi}^{(n)} + \mathrm{Im}\vec{\psi}^{(n)}.$$

Placing this decomposition into the Crank-Nicolson formulation, and noting that the Hamiltonian matrix is purely real (*see, e.g.*, Project 4), we get two coupled equations:

$$\mathrm{Re}\vec{\psi}^{(n+1)} - \frac{\Delta t\hat{\mathcal{H}}}{2\hbar}\mathrm{Im}\vec{\psi}^{(n+1)} = \mathrm{Re}\vec{\psi}^{(n)} + \frac{\Delta t\hat{\mathcal{H}}}{2\hbar}\vec{\psi}^{(n)}$$

$$\mathrm{Im}\vec{\psi}^{(n+1)} + \frac{\Delta t\hat{\mathcal{H}}}{2\hbar}\mathrm{Re}\vec{\psi}^{(n+1)} = \mathrm{Im}\vec{\psi}^{(n)} - \frac{\Delta t\hat{\mathcal{H}}}{2\hbar}\vec{\psi}^{(n)}$$

These equations can be combined into a larger set of coupled linear equations in the form

$$\begin{pmatrix} 1 & -\frac{\Delta t}{2\hbar}\hat{\mathcal{H}} \\ \frac{\Delta t}{2\hbar}\hat{\mathcal{H}} & 1 \end{pmatrix} \begin{pmatrix} \mathrm{Re}\vec{\psi}^{(n+1)} \\ \mathrm{Im}\vec{\psi}^{(n+1)} \end{pmatrix} = \begin{pmatrix} 1 & \frac{\Delta t}{2\hbar}\hat{\mathcal{H}} \\ -\frac{\Delta t}{2\hbar}\hat{\mathcal{H}} & 1 \end{pmatrix} \begin{pmatrix} \mathrm{Re}\vec{\psi}^{(n)} \\ \mathrm{Im}\vec{\psi}^{(n)} \end{pmatrix}. \tag{16}$$

If the wavefunction is discretized as a complex vector with dimension N, we can replace it with real vectors with dimension $2N$.

To evolve in time from time t_n to t_{n+1}, start with the $2N$-dimensional vector,

$$\begin{pmatrix} 1 & \frac{\Delta t}{2\hbar}\hat{\mathcal{H}} \\ -\frac{\Delta t}{2\hbar}\hat{\mathcal{H}} & 1 \end{pmatrix} \begin{pmatrix} \mathrm{Re}\vec{\psi}^{(n)} \\ \mathrm{Im}\vec{\psi}^{(n)} \end{pmatrix}.$$

Finally, we must solve the equation

$$\begin{pmatrix} 1 & -\frac{\Delta t}{2\hbar}\hat{\mathcal{H}} \\ \frac{\Delta t}{2\hbar}\hat{\mathcal{H}} & 1 \end{pmatrix} \begin{pmatrix} \mathrm{Re}\vec{\psi}^{(n+1)} \\ \mathrm{Im}\vec{\psi}^{(n+1)} \end{pmatrix} = \begin{pmatrix} 1 & \frac{\Delta t}{2\hbar}\hat{\mathcal{H}} \\ -\frac{\Delta t}{2\hbar}\hat{\mathcal{H}} & 1 \end{pmatrix} \begin{pmatrix} \mathrm{Re}\vec{\psi}^{(n)} \\ \mathrm{Im}\vec{\psi}^{(n)} \end{pmatrix}.$$

Because our matrix is now real, we can use LU decomposition to either solve the linear equation of invert the matrix on the left hand side of the equation. If we do the latter, it is more efficient to invert the matrix once and store it, because the matrix does not change as time evolves. Finally, you will also need to calculate the probability density,

$$\rho(x_j, t_n) = |\Psi(x_j, t_n)|^2 = |\mathrm{Re}\vec{\psi}^{(n)}|^2 + |\mathrm{Im}\vec{\psi}^{(n)}|^2.$$

A. Basic Project (worth 75%)

Place your wavefunction in a box from $-L$ to $+L$. By using the same discretization of the second derivative as in Project 4, we implicitly have the boundary condition that the wavefunction vanishes at the boundaries. For simplicity, you can set $\hbar = m = 1$ (a departure from Project 4).

Have the user input: the size of the box, L, the number of lattice points, the time step and the number of time steps to take.

Create a Gaussian wave function somewhere in the box. The user should be able to input the center and width of the Gaussian. Let this wavefunction evolve under the free Schrödinger equation (*i.e.*, $V(x) = 0$). Your code should output the following for each time step either into three individual files or one file with a header that makes it clear what information is in each column.

- The normalization $\sum_{j=1}^{N} \rho_j \, dx$; we expect this to be constant.
- The expectation value of x: $\langle x \rangle = \sum_{j=1}^{N} x_j \rho_j \, dx / \sum_{j=1}^{N} \rho_j \, dx$.
- The variance ("width"), $\sigma^2 = \langle x^2 \rangle - \langle x \rangle^2$, where $\langle x^2 \rangle = \sum_{j=1}^{N} x_j^2 \rho_j \, dx / \sum_{j=1}^{N} \rho_j \, dx$.

After the last time step, print to a file the value of the probability density $\rho(x, t_f) = |\Psi(x, t_f)|^2$.

In quantum mechanics, we can analytically compute the spreading of a Gaussian. Suppose we start with a Gaussian wavefunction,

$$\Psi(x, t = 0) = (2\pi\sigma^2)^{-1/4} \exp\left[-\frac{(x - x_0)^2}{4\sigma^2}\right].$$

The functional form of the wavefunction is not a pure Gaussian because the probability density is determined by taking the magnitude squared. One can show that the solution to the Schrödinger equation is

$$\Psi(x, t) = (2\pi\sigma^2)^{-1/4} \frac{1}{\sqrt{1 + i\frac{\hbar^2 t}{2m\sigma^2}}} \exp\left[-\frac{(x - x_0)^2}{4\sigma^2\left(1 + i\frac{\hbar^2 t}{2m\sigma^2}\right)}\right],$$

where the speed of light, $c = 1$. The probability density is thus

$$\rho(x, t) = |\Psi(x, t)|^2 = \frac{1}{\sqrt{2\pi\left(\sigma^2 + \frac{\hbar^4 t^2}{4m^2\sigma^4}\right)}} \exp\left[-\frac{(x - x_0)^2}{2\left(\sigma^2 + \frac{\hbar^4 t^2}{4m^2\sigma^4}\right)}\right].$$

The width of the Gaussian wavefunction grows as

$$\langle (x - x_0)^2 \rangle = \sigma^2 + \frac{\hbar^4 t^2}{4m^2\sigma^2}$$

To validate your code, confirm that your width grows, at least initially, according to the analytic result. After some time, it will deviate from the analytical solution because the boundary conditions are those of an infinite potential well. Figure 2 shows an example solution. At time $t \sim 2$, the numerical solution deviates from the analytical solution as is expected.

If we take a snapshot at different points in time in the evolution, Figure 3, then we can see how the hard boundary conditions we impose on the wave functions result in reflections off the boundary changing the wavefunction at large times from a Gaussian, to another beast completely.

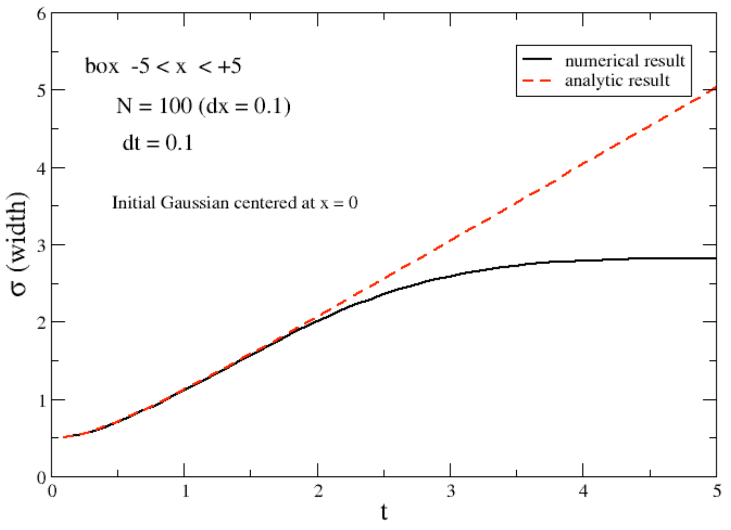

FIG. 2: The evolution of the width of a Gaussian wavefunction, initially centered at $x = 0$ with width 0.5.

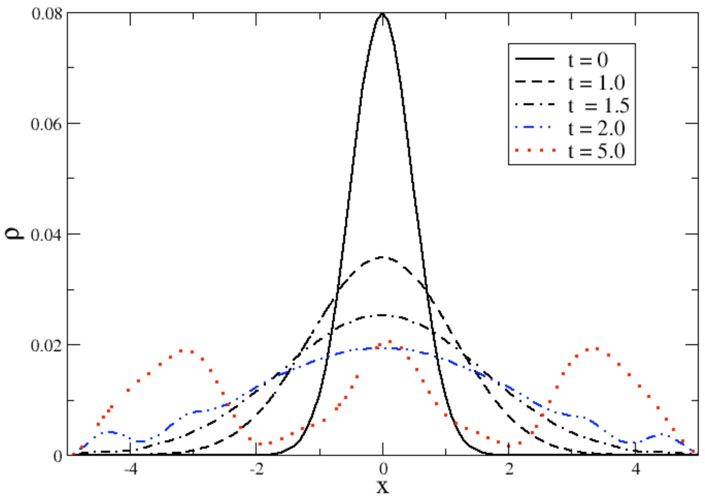

FIG. 3: The evolution of a Gaussian wavefunction, initially centered at $x = 0$ with width 0.5. Plotted is the probability density as a function of location in the box.

Finally, check the dependence on dt. The normalization should be stable, even for reasonably large values of dt, this is expected because the Crank-Nicolson method is a unitary method which should conserve the normalization. However, when looking at the width, it is much more sensitive to values of dt. Figure 4 shows a sensitivity study of the width of the wavefunction to chosen value of dt. In the figure, we see that for $dt = 0.1$ and $dt = 0.2$, the results are pretty similar.

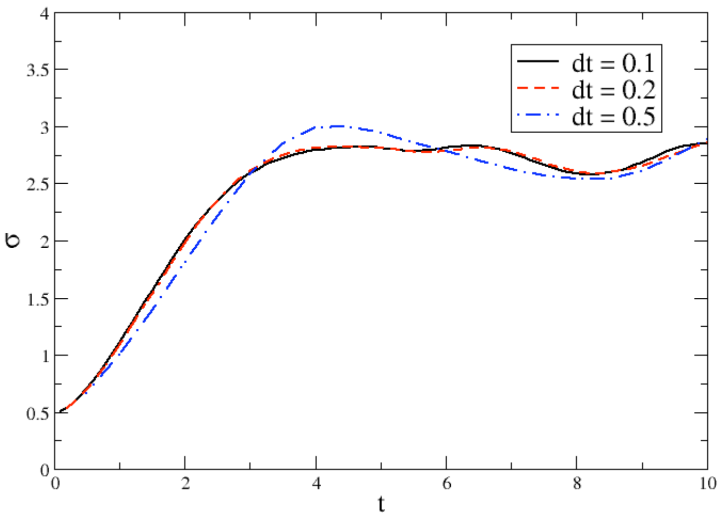

FIG. 4: The evolution of a Gaussian wavefunction, initially centered at $x = 0$ with width 0.5. Plotted is the evolution of the width of the wavefunction as a function of time. A sensitivity study is shown, illustrating the effects of different choices of dt on the results.

For the basic project, produce a figure similar to Figure 2 to show that you can validate your program by having a similar width evolution at least for short times. Second, produce a figure similar to Figure 3 that shows the wavefunction at your final time. Finally, choose illustrative values of dt in a plot similar to Figure 4 to show that your result is converged.

B. Advanced Project

Now place your wavefunction in a harmonic well with $V(x) = 1/2kx^2$. The program should allow the user to input k (it is recommended to use a relatively small value of k, around 0.01 or 0.02). If you follow $\langle x \rangle$ as a function of time, you would expect to find get the classical result, $\langle x(t) \rangle = x_0 \cos(t\sqrt{k/m})$. Is this the case? What happens as the initial width is changed?

Your code should ask for the same inputs as the basic project, and include an input for k. Produce a plot of $\langle x(t) \rangle$ for a value of x_0 and some value of k (make sure your parameters are evident on your plot) for two different initial widths. It is advised to make sure that x_0 is sufficiently far away from the boundary so that the tail of the wavefunction is small at the boundaries. You may find that for even small values of dt, the normalization grows with time, do not worry about this and keep moving forward.