PHY407 Computational Lab
Solving PDEs, Part 2

This lab has only *two* questions but they are each challenging in their own way. It would be helpful for you and your partner to discuss both questions in developing your pseudocode and solution strategy.

This week's lab starts with applying the spectral method to the water wave problem, presenting this method as an alternative to the leapfrog-plus-relaxation we used in Lab08 Q2 last week. The lab continues with applying Crank Nicholson time stepping to the time-dependent Schrödinger equation. These topics are handled in a somewhat reverse order in the book: Newman presents Crank-Nicholson time stepping and then presents the spectral method as an alternative to C-N. I suggest reading the C-N material carefully and skimming the spectral method part in the book, and also reading this assignment's background carefully.

Computational Background

- **Spectral method (for Q1):** Section 9.3.4 of the Newman text discusses the spectral method as applied to the time dependent Schrödinger equation. This method is quite general and works very well for linear PDEs with constant coefficients in simple geometries. Pages 435-438 discuss the spectral method as applied to the Schrödinger wave equation. In this lab you will use it to solve the water wave equation that we looked at in Lab08 Q2.

  The basic idea is to do more analysis up front before using the computer (seems like a good idea, right?). You start by writing the solution in a Fourier series and then solve for the coefficients of the Fourier series. In the end, you use the computer to carry out the Fourier transform, carrying out operations in spectral space, and then find the solution doing an inverse Fourier transform.

  Unlike the example in the book, we will not be doing a discrete Fourier sine transform. Instead we will consider the evolution of the Gaussian initial condition in η, which was the problem we were trying to solve in Lab08 Q2. However, it is worth reading the book because the basic idea is very similar. We will introduce the theory in the Physics Background and provide more guidance in Q1.

  If you understand the principles of this question there is actually very little coding involved. You will see, in fact, that the code you have runs almost instantantly, in contrast to the leapfrog plus relaxation code you ran in Lab08. The spectral method is great as long as the conditions are right to apply it!

- **Crank-Nicholson method (for Q2):** Remember that the FTCS method suffered from stability problems. The Crank-Nicolson (C-N) scheme involves a combined implicit/explicit scheme that is numerically stable. Its used quite often for time-evolution of PDEs. Section 9.3.3 of the Newman text discusses the method in detail as applied to the wave equation. For the wave equation, C-N is stable (i.e. stable for all time step sizes) because the magnitudes of the eigenvalues always equal 1. This means that initial fourier modes neither grow nor decay in the evolution, precisely the behaviour one expects for the solution to

the wave equation. For the time-dependent Schrodinger equation, the C-N equation is given at the top of page 440. It results in a set of simultaneous equations, one for each grid point. This means that we can solve the system using our methods for solving linear systems (from Chapter 6). One particular aspect we didn't discuss yet this year, but is explored in the problem, is taking advantage of the banded structure of matrices to solve for a linear set of equations. The module `banded.py`, provided by Newman in his online materials, has the functionality to do this.

- **Plotting wave functions and animations (for Q2):** In Q2 I ask you to plot and animate the real part of the wavefunction ψ and the wave function envelope given by $\pm|\psi|$. For Q2d) it is also nice to plot the potential barrier as well (just to give the eye a point of reference for the scattering problem). Some code to accomplish the steps might look like this:

```
while t < tend:
    clf()
    #calculate psi ...
    # update animation with psi at current timestep
    plot(x, abs(psi),'-b',x,psi.real,'-r',x,-abs(psi),'-b')
    plot(x,V(x)/(eV*80.0),'g') #this scales V to go between 0 and 0.5
    ylim((-1,1))
    my_title="Time: %e" % (t)
    title(my_title)
    pause(0.01)
    t += h
```

Physics Background

- **Spectral methods:** Equations (1)-(3) of Lab08 can be solved using Fourier transforms. Let's express ϕ and η as Fourier series as follows:

$$\eta(x,t) = \sum_{k=-\infty}^{\infty} e^{ikx}\tilde{\eta}_k(t),$$

$$\phi(x,z,t) = \sum_{k=-\infty}^{\infty} e^{ikx}\hat{\phi}_k(z)\tilde{\phi}_k(t).$$

We are using Fourier series instead of discrete Fourier transforms at this stage, to make some of the math easier for the question. For example, we can apply derivatives to these equations. Time derivatives are expressed as a Fourier series of time derivatives on the coefficients:

$$\frac{\partial \eta}{\partial t} = \sum_{k=-\infty}^{\infty} e^{ikx}\frac{d\tilde{\eta}_k}{dt},$$

and the Laplacian in x and z can be expressed as a series of operations on the vertical structure functions $\hat{\phi}_k(z)$:

$$\nabla^2\phi = \sum_{k=-\infty}^{\infty} e^{ikx}\left[\frac{d^2\hat{\phi}_k(z)}{dz^2} - k^2\hat{\phi}_k(z)\right]\tilde{\phi}_k(t).$$

This gives us a useful result right away: because for this particular problem

$$\nabla^2 \phi \equiv 0,$$

we have that

$$\frac{d^2\hat{\phi}_k(z)}{dz^2} - k^2\hat{\phi}_k(z) = 0$$

for all k, which requires that $\phi_k(z) \propto e^{kz}$, $-\infty < z < 0$. Proceeding in this way leads to Fourier series expressions for the solution to the water wave problem which we will explore.

- **Time-dependent Schrödinger Equation (for Q2):** Here we numerically solve the time-DEPENDENT Schrodinger equation. In 1 dimension, the Schrodinger equation for a particle of mass M with no potential energy reads:

$$-\frac{\hbar^2}{2M}\frac{\partial^2\psi}{\partial x^2} = i\hbar\frac{\partial\psi}{\partial t} \tag{1}$$

Notice that the derivatives in this equation are 1st order in time and 2nd order in space, which means that its form is similar to the diffusion equation. However, it is complex and therefore leads to more complicated behaviour. Allowing the time derivative in the equation means that the wave function ψ can evolve in time. This will therefore allow us to study the evolution of quantum wave packets. We will also look at the packets in the presence of a potential barrier $V(x)$. In this case, the TDSE becomes

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

Lab Instructions

For all codes mentioned, it is always recommended to write pseudocode first!

For grading purposes, only hand in the following parts. Ensure that your codes are well commented and readable by an outsider:

- For Q1: Hand in your analytic solution for Q1a (can be scanned or photographed as long as they are included in the pdf). Hand in other written answers and plots.
- For Q2: Hand in code, plots, and written answers.

Lab Questions

1. **Spectral approach to the water wave problem. (40% of the lab)** Consider the water wave problem of Lab08 Q2, consisting of equations (1)-(3) and an initial condition in which $\phi(x, z, t = 0) \equiv 0$ and $\eta(x, t = 0) = \eta_0(x)$ is the same Gaussian as given in Lab08. Denote by $\hat{\eta}_k(t = 0)$ as the initial distribution of Fourier coefficients through

$$\eta_0(x) = \sum_{k=-\infty}^{\infty} e^{ikx}\tilde{\eta}_k(t = 0).$$

(a) Show that the general solution for this problem in the same domain as in Lab08 Q2, is

$$\eta(x,t) = \sum_{k=-\infty}^{\infty} e^{ikx}\tilde{\eta}_k(t=0)\cos\omega_k t, \tag{3}$$

$$\phi(x,z,t) = -\sum_{k=-\infty}^{\infty} e^{ikx}e^{kz}\frac{(g\tilde{\eta}_k(t=0))}{\omega_k}\sin\omega_k t, \tag{4}$$

$$\text{where } \omega_k = \sqrt{gk}. \tag{5}$$

Equations (3)-(5) provide, without any numerical time stepping, a series solution for any time t. Show also that the $k = 0$ solution doesn't change in time (so the x averaged part of the solution is constant)

(b) Using this information and the `numpy.fft.rfft` and `numpy.fft.irfft` routines of previous labs, generate and plot the solution for $t = 0, 2, 10, 40$s, for both η and ϕ. Compare this new solution to your previous solutions. You can use a very large number of Fourier coefficients in this calculation - since you only have to do one Fourier transform.

Say a few words about the different approaches, and the relative advantages and disadvantages. Have fun zooming in on the fine structure of the solution.

Hint: one trick here is to translate the expressions above into the discrete Fourier transform expressions you've used before. A useful change of variables is

$$k \to 2\pi k'/L,$$

where k is a wavenumber (equal to $2\pi/\lambda$) and k' is an integer index, which corresponds to a wavenumber when multiplied by $2\pi/L$, where L is the length of the domain.

2. **The Schrödinger equation and the Crank–Nicolson method (60% of the lab):** This is based on Newman Exercise 9.8 but there are differences, so read carefully.

Perhaps the most important partial differential equation, at least for physicists, is the Schrödinger equation. This exercise uses the Crank–Nicolson method to solve the full time-dependent Schrödinger equation and hence develop a picture of how a wavefunction evolves over time.

We will look at the Schrödinger equation in one dimension. The techniques for calculating solutions in two or three dimensions are basically the same as for one dimension, but the calculations take much longer on the computer, so in the interests of speed we'll stick with one dimension. In one dimension the Schrödinger equation for a particle of mass M with no potential energy reads

$$-\frac{\hbar^2}{2M}\frac{\partial^2\psi}{\partial x^2} = \mathrm{i}\hbar\frac{\partial\psi}{\partial t}.$$

For simplicity, let's put our particle in a box with impenetrable walls, so that we only have to solve the equation in a finite-sized space. The box forces the wavefunction ψ to be zero at the walls, which we'll put at $x = 0$ and $x = L$.

Replacing the second derivative in the Schrödinger equation with a finite difference and applying Euler's method, we get the FTCS equation

$$\psi(x,t+h) = \psi(x,t) + h\frac{\mathrm{i}\hbar}{2ma^2}\bigl[\psi(x+a,t) + \psi(x-a,t) - 2\psi(x,t)\bigr],$$

where a is the spacing of the spatial grid points and h is the size of the time-step. (Be careful not to confuse the time-step h with Planck's constant $\hbar$.) Performing a similar step in reverse, we get the implicit equation

$$\psi(x,t+h) - h\frac{\mathrm{i}\hbar}{2ma^2}\bigl[\psi(x+a,t+h) + \psi(x-a,t+h) - 2\psi(x,t+h)\bigr] = \psi(x,t).$$

And taking the average of these two, we get the Crank–Nicolson equation for the Schrödinger equation:

$$\begin{aligned}\psi(x,t+h) - h\frac{\mathrm{i}\hbar}{4ma^2}&\bigl[\psi(x+a,t+h) + \psi(x-a,t+h) - 2\psi(x,t+h)\bigr]\\ &= \psi(x,t) + h\frac{\mathrm{i}\hbar}{4ma^2}\bigl[\psi(x+a,t) + \psi(x-a,t) - 2\psi(x,t)\bigr].\end{aligned}$$

This gives us a set of simultaneous equations, one for each grid point.

The boundary conditions on our problem tell us that $\psi = 0$ at $x = 0$ and $x = L$ for all t. In between these points we have grid points at a, $2a$, $3a$, and so forth. Let us arrange the values of ψ at these interior points into a vector

$$\boldsymbol{\psi}(t) = \begin{pmatrix}\psi(a,t)\\ \psi(2a,t)\\ \psi(3a,t)\\ \vdots\end{pmatrix}.$$

Then the Crank–Nicolson equations can be written in the form

$$\mathbf{A}\boldsymbol{\psi}(t+h) = \mathbf{B}\boldsymbol{\psi}(t),$$

where the matrices $\mathbf{A}$ and $\mathbf{B}$ are both symmetric and tridiagonal:

$$\mathbf{A} = \begin{pmatrix}a_1 & a_2 & & & \\ a_2 & a_1 & a_2 & & \\ & a_2 & a_1 & a_2 & \\ & & a_2 & a_1 & \\ & & & & \ddots\end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix}b_1 & b_2 & & & \\ b_2 & b_1 & b_2 & & \\ & b_2 & b_1 & b_2 & \\ & & b_2 & b_1 & \\ & & & & \ddots\end{pmatrix},$$

with

$$a_1 = 1 + h\frac{\mathrm{i}\hbar}{2ma^2}, \qquad a_2 = -h\frac{\mathrm{i}\hbar}{4ma^2}, \qquad b_1 = 1 - h\frac{\mathrm{i}\hbar}{2ma^2}, \qquad b_2 = h\frac{\mathrm{i}\hbar}{4ma^2}.$$

(Note the different signs and the factors of 2 and 4 in the denominators.)

The equation $\mathbf{A}\boldsymbol{\psi}(t+h) = \mathbf{B}\boldsymbol{\psi}(t)$ has precisely the form $\mathbf{A}\mathbf{x} = \mathbf{v}$ of the simultaneous equation problems we studied in Chapter 6 and can be solved using the same methods. One special feature of this system is that the matrix $\mathbf{A}$ is tridiagonal. This means that we can use a special form of Gaussian elimination that takes advantage of the tridiagonal structure. The fast tridiagonal version of Gaussian elimination is discussed in Section 6.1.6 and it will save you a lot of time if you use this method.

Consider an electron (mass $M = 9.109 \times 10^{-31}$ kg) in a box of length $L = 10^{-8}$ m. Suppose that at time $t = 0$ the wavefunction of the electron has the form

$$\psi(x,0) = \exp\left[-\frac{(x-x_0)^2}{2\sigma^2}\right] e^{i\kappa x},$$

where

$$x_0 = \frac{L}{2}, \qquad \sigma = 1 \times 10^{-10}\,\text{m}, \qquad \kappa = 5 \times 10^{10}\,\text{m}^{-1},$$

and $\psi = 0$ on the walls at $x = 0$ and $x = L$. (This expression for $\psi(x,0)$ is not normalized—there should really be an overall multiplying coefficient to make sure that the probability density for the electron integrates to unity. It's safe to drop the constant, however, because the Schrödinger equation is linear, so the constant cancels out on both sides of the equation and plays no part in the solution.)

(a) Write a program to perform a single step of the Crank–Nicolson method for this electron, calculating the vector $\boldsymbol{\psi}(t)$ of values of the wavefunction, given the initial wavefunction above and using $N = 1000$ spatial slices with $a = L/N$. Your program will have to perform the following steps. First, given the vector $\boldsymbol{\psi}(0)$ at $t = 0$, you will have to multiply by the matrix $\mathbf{B}$ to get a vector $\mathbf{v} = \mathbf{B}\boldsymbol{\psi}$. Because of the tridiagonal form of $\mathbf{B}$, this is fairly simple. The ith component of $\mathbf{v}$ is given by

$$v_i = b_1\psi_i + b_2(\psi_{i+1} + \psi_{i-1}).$$

You will also have to choose a value for the time-step h. A reasonable choice is $h = 10^{-18}$ s.

Second you will have to solve the linear system $\mathbf{A}\mathbf{x} = \mathbf{v}$ for $\mathbf{x}$, which gives you the new value of $\boldsymbol{\psi}$. You could do this using a standard linear equation solver like the function `solve` in `numpy.linalg`, but since the matrix $\mathbf{A}$ is tridiagonal a better approach would be to use the fast solver for banded matrices given in Appendix E of Newman, which can be imported from the file `banded.py` (which you can find in the on-line resources).[1]

Third, once you have the code in place to perform a single step of the calculation, extend your program to perform repeated steps and hence solve for ψ at a sequence of times a separation h apart. Note that the matrix $\mathbf{A}$ is independent of time, so it doesn't change from one step to another. You can set up the matrix just once and then keep on reusing it for every step.

[1] In the text at this point there are a couple of sentences about the wavefunction not needing to be of type complex. In fact `banded.py` in my solution works regardless of whether its input is complex or real, and my solution took advantage of the complex type, so I thought this text was just confusing and deleted it.

(b) Extend your program to make an animation of the solution by displaying the real part of the wavefunction[2] at each time-step and, as an overlay, plus and minus the absolute value of the wave function as well. That is, plot both Reψ and $\pm|\psi|$. You can use matplotlib or visual animation to do this (the former is easier but some of you may prefer the latter).

(c) Run your animation for a while and describe what you see. Write a few sentences explaining in physics terms what is going on in the system. It is likely you will see some noise in your solution - you can try to eliminate some of this noise by playing with numerical parameters, but this will be a challenge.

(d) Alter the problem to introduce a potential jump of the form

$$V(x) = \begin{cases} 0 & x \leq L/2 \\ V_0 & x > L/2 \end{cases}.$$

Use $V_0 = 40\,\text{eV}$. To do this you will need to change the diagonal entries of the matrices **A** and **B**. Once your program is ready , initialize the wave packet to the left of the barrier by setting $x_0 = 0.4L$. Observe what happens in your animation and describe it in words and with a couple of figures.

[2]This is the wording from the text.