Finding Energy Eigenvalues for Symmetric Potentials from Numerical Integration of Schrodinger's Equation

Integrating the Time-Independent Schrodinger Equation

The basic idea behind numerical integration is to replace continuous integration with computations that divide the calculation into small, fixed, intervals of space (Δx) or time (Δt) in order to do the calculation in *steps*. In this project, one uses spatial integration, starting from some position, such as $x_i = 0$, and stepping to some final position, x_f, solving for one or more quantities of interest at each step of the calculation, and ultimately determining the solutions that correspond to the energy eigenvalues of a given static potential, *V(x)*.

The procedure used here starts with the Time-Independent Schrodinger Equation (TISE), rearranged to solve for the *curvature* of the wavefunction, then integrates this to solve for the *slope* of the wavefunction, and then integrates again to determine the *value* (amplitude) of the wavefunction as a function of x. At each step Δx it will be possible to find the curvature, slope, and value of the wavefunction, and to tabulate or plot these quantities for the region between x_i and x_f.

The starting point for this study is to choose a static potential *V(x)*, and in this case we will restrict the options to potentials that are symmetric about $x = 0$. [Examples we have studied include the harmonic oscillator potential ($\frac{1}{2}kx^2$) or the symmetric square well (extending from $-a/2$ to $a/2$).]

Equations for the Numerical Integration Procedure

First we rearrange the TISE to solve for the curvature of the wavefunction at some location x:

$$\frac{d^2\psi}{dx^2} = \frac{2m}{\hbar^2}\left[V(x) - E\right]\psi(x) \qquad (1)$$

Since we will be integrating in finite intervals Δx, we shall determine the slope of the wavefunction using the definition of the derivative – but for finite intervals of small Δx instead of infinitesimal dx:

$$\left(\frac{d^2\psi}{dx^2}\right)_{x_0} = \frac{\left[\left(\frac{d\psi}{dx}\right)_{x_0+\Delta x} - \left(\frac{d\psi}{dx}\right)_{x_0}\right]}{\Delta x}$$

Rather than taking the limit $\Delta x \rightarrow 0$, we rearrange this equation to solve for the approximate *slope* of the wavefunction at $x_0 + \Delta x$:

$$\left(\frac{d\Psi}{dx}\right)_{x_0+\Delta x} = \left(\frac{d\psi}{dx}\right)_{x_0} + \left(\frac{d^2\psi}{dx^2}\right)\Delta x \qquad (2)$$

Next, to find the *value* of the wavefunction at $x_0 + \Delta x$, we do a Taylor series expansion of the wavefunction about the point x_0:

$$\psi(x_0 + \Delta x) = \psi(x_0) + \left(\frac{d\psi}{dx}\right)_{x_0} \Delta x + \left(\frac{d^2\psi}{dx^2}\right)_{x_0} \frac{\Delta x^2}{2} \qquad (3)$$

If we know all the values on the right-hand side of this equation from equations (1) and (2) we can find the value of the wavefunction at the *beginning of the next step*, starting at $x_0 + \Delta x$. We could then compute the next step by letting $x_0 \rightarrow x_0 + \Delta x$ and passing again through equations (1), (2), and (3) to extend the integration by another Δx. The calculation progresses in this way until x reached the specified endpoint, x_f.

You can see that this requires a program structure that loops through equations (1), (2), and (3) until x is greater than or equal to x_f, acting on program variables such as the current curvature, slope, and value of the wavefunction. In addition, the potential $V(x)$ should be defined as a callable function so that it is a simple matter to change from one potential to another. Before running the program on a "new" potential of your choice, it should be debugged and tested on a potential with known energy eigenvalues, such as the symmetric square well.

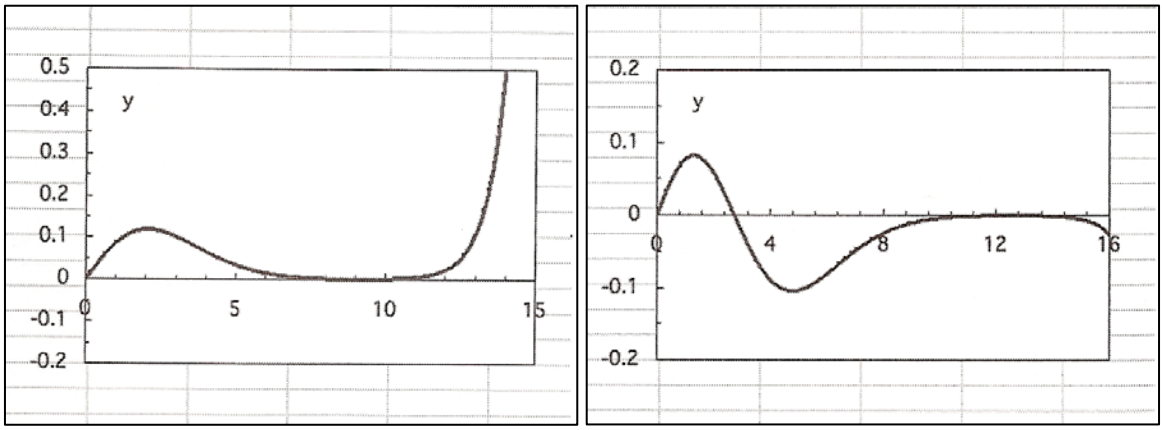
The Core of the Method

Using this program to find the energy eigenvalues E_n for some new $V(x)$ is a "fishing expedition", where you pick trial values for E_n to see if they yield a reasonable wavefunction. In this case, "reasonable" means that you are looking for solutions where the wavefunctions go to zero at infinity – as they must in order to be normalizable. How can you "home in" on these solutions? You will find that if your trial value for E_n is too high or too low, rather than going to zero as you integrate toward x_f, at some point the wavefunction will diverge rapidly to the + or – direction. This occurs because the calculation will have gone beyond the classical turning point, x_{TP}, in the $+x$ direction, where $V(x)$ is greater than E_n, and the only allowed solutions are "exponential" in character. In fact, given the numerical accuracy of these calculations, you can never get the solutions to avoid this behavior! But in the process of tuning E_n to keep the wavefunction as close to zero as possible for some reasonable distance beyond x_{TP} you will find that E_n has been determined to several decimal places of accuracy.

You can see what this implies for the program: that you should have it loop back repeatedly to the beginning of the calculation to ask for the next trial E_n in order to quickly cycle through the possibilities. You will also want to be able to view the wavefunction behavior by using a table or plot that updates quickly at the end of each trial, to see the wavefunction has the right nodal structure inside the well and if it is approaching zero to reasonable accuracy beyond the turning point.

The two plots below show the trial solutions for some symmetric potential well centered at $x = 0$, with the integration proceeding from that point to the right, until the calculation diverges. Since these wavefunctions have a node at $x = 0$, and the one on the right has three nodes in total, they must be the two lowest lying antisymmetric states for this

potential. (For the ground state and all other symmetric states the wavefunction would be finite at the origin and of zero slope at that point.) You can see that beyond the classical turning point (where there is an inflection in the wavefunction), these solutions stay reasonably close to zero before diverging to + or – infinity. Note that even the slightest variation of the trial energies for these solutions could have caused the divergences to "go the other way" and still yield a very accurate determination of, in this case, E_2 and E_4.



Setup of Stepping and the Initial Conditions

From the equations and plots above, you can see that there are a number of constants that need to be defined as part of the stepping process and to establish the initial conditions.

To set up the stepping you will need to choose a step size, Δx, and the upper limit for the integration, x_f. Note that you can also choose to stop the calculation when the wavefunction has diverged to a certain point in the + or – direction. You can develop a quick intuition for these settings by varying them and noting the behavior and accuracy of the calculation. (More below in the hints on debugging.)

The setup of initial conditions for the wavefunction itself is much easier. If you are calculating a wavefunction of *even* symmetry, at the first step you will need to set the wavefunction to some finite value and the slope to zero. The linearity of the Schrodinger equation allows you to choose any convenient value without affecting the E_n's in the end, so you can set $\psi = +1.000$ for convenience. For *odd* symmetry, the wavefunction must be set initially to zero, and the slope given some finite value. Also because of linearity, $d\psi/dx$ can be set to +1.000 for these cases.

The Energy Scale

Since the constants associated with quantum systems always involve very small numbers in S.I. units, it is difficult to do stable calculations and understand the results directly in S.I. units. So before plowing into the calculation it is necessary to identify the natural energy scale of the problem and then set up to do the analysis in the appropriate units. Fortunately the natural units can be established in one place: in the TISE of equation (1).

We'll illustrate the choice of natural units for the case of typical problems at the atomic scale (eg. hydrogen atom.). For easier understanding, and for convenience in coding, we

collect the constants appearing in front of the bracket on the right-hand side into one constant, β:

$$\frac{d^2\psi}{dx^2} = \frac{2m}{\hbar^2}[V(x) - E]\psi(x) = \beta[V(x) - E]\psi(x) \qquad \text{where} \qquad \beta = \frac{2m}{\hbar^2}$$

The natural units for atomic calculations are angstroms [L], electron mass [M], and electron-volts [E]. We can convert β to these units using (1) 1 angstrom = 10^{-10} m, (2) 1 electron mass = 9.1049 x 10^{-31} kg, and (3) 1 eV = 1.6022 x 10^{-19} Joule. And the result is: β_e = 1.6831 x 10^{38} (J m^2)$^{-1}$ = 0.26246 (eV angstroms^2)$^{-1}$, for one electron mass. So if you were doing a calculation using these units, you would use angstroms for distances, and eV for $V(x)$ and E_n. For accuracy, your step-size, Δx, should be a small fraction of an angstrom, and your tables and plots would be labeled in angstroms. This will assure that you are not seeing extremely small or large numbers in tables or plots (except when the wavefunction diverges).

If you were doing a nuclear physics problem the natural units would probably be femtometers, the proton mass, and MeV. Each problem will have a set of units which makes the calculation doable and easily understandable. You should set this up ahead of time to avoid confusion.

Computer Language, System, and Tools

You have probably used C in an introductory computer course, or in some other environment. But you are free to do this in any language you prefer – Basic, Fortran, C++, Java, Excel equations, etc – as long as you write the whole program yourself. You will need to include the source code in your writeup, along with a typical output, a typical plot, and the full results of your study.

Similarly, any method to process the output of your program and make plots is acceptable. For this type of step-wise computation, there are two "universal" stages to plot creation:

(1) If, for example, you wanted to plot the wavefunction vs. x as shown in the plots above, at each step in x you will need to give the current x and $\psi(x)$ to the plotting program. (If the steps are very closely spaced you may prefer to plot only every nth point.)

(2) If the plotting program is linked to the main program, you can do this on the fly. If not, you can write an output file with x and $\psi(x)$ in two columns, which the plotting program can read back and plot as a "side" process.

For those of you using Unix/Linux, I will give an example that illustrates the use of a particular package, gnuplot, as a side process. (gnuplot is available for all Linux machines, and already installed on most.) Suppose that you write a two-column file, myplot.dat, as described in (2) above. If want to produce the most basic plot, and have gnuplot do all the work, all you need to do is type: "gnuplot", and then when it pops up with "gnuplot>", type "plot ./myplot.dat". It will open an X-window and show you the plot. You can exit gnuplot by typing "q".

If you want to customize the plot, and be able to produce hard-copy, here is a typical gnuplot script (eg. gnuplot.scpt) that would do the job:

```
set timestamp
set terminal postscript        #Comment out for X terminal
set title  "This is the main title for the plot"
set ylabel "This is the y axis label"
set xlabel "This is the x axis label"
set xrange [0.0) to 10.0]      #Tune this to your specs
set yrange [-3.0 to 3.0]       #Tune this to your specs
set output "myplot.ps"         #Comment out for X terminal
plot "./myplot.dat" with lines
quit
```

Edit this text into the file gnuplot.scpt, then type "gnuplot ./gnuplot.scpt" to have it use this script instead of your typed input. As you can see, the "#" is the comment symbol, and if you put one at the beginning of any line, it will ignore that line. This script will produce the postscript file myplot.ps. If you want to create a PDF file from this, just type "ps2pdf myplot.ps".

Again, if you already have a plotting program you can use for this project, feel free to use it instead. This also applies to the tools you use to create the project report. The final report should be one electronic file (probably PDF). But to create it you can use any combination that works for you: Windows(Word,Powerpoint)/Linux/TeX/Mac/Etc…..

Validation steps to help in debugging the program

Use a symmetric square well to debug and calibrate your program. At the beginning you can do a very basic check to see if you program is stepping. Just set β to 1.0, the trial energy to 5.0, the initial slope to 0.0, the initial wavefunction to 1.00, the stepsize to 0.01, and the potential to zero. Then integrate in x far enough to see that the wavefunction has a cosine form. This is like simulating a high-n quantum state in an infinite square well and just looking at the first few cycles of the stationary wavefunction. You can investigate the accuracy by looking at the values of successive maxima and seeing what happens when you tune the stepsize. Then to see how the system behaves beyond the classical turning point, introduce a step at $x = 0.6$ and height 118.6. At this point you should be able to move the trial energy up and down in the neighborhood of 5.0 and see the onset the wavefunction divergence (+ and –) for $x > \sim 2.0$.

Nature of the project

Pick a new potential and solve for the energies of the four lowest-lying eigenstates. For this part you will need to choose the natural units and convert β to these unit.