Numerical solution to the Woods-Saxon potential

We want to find numerical solutions to the Schrödinger equation

\[ -\frac{\hbar^2}{2m} \frac{d^2}{dx^2} + V(x) \psi(x) = E \psi(x), \]

in particular for the “Woods-Saxon potential,”

\[ V(x) = \frac{-V_0}{1 + \exp\left((|x| - R)/a\right)} \]

where \( V_0, R, \) and \( a \) are all parameters of the potential (the “depth,” “radius,” and “surface thickness,” respectively.

The Basic Idea: We will discretize the wavefunction \( \psi \) on a lattice. On the lattice, the potential energy will be a diagonal matrix, while the kinetic energy is a tridiagonal matrix. We can then use the Numerical Recipes routine TQLI (or my master routine eig) to find the eigenvalues and eigenvectors of the resulting tridiagonal matrix.

Your ultimate goal is to find the eigenenergies and eigenfunctions for the Woods-Saxon potential, for values that approximate the nuclear potential, as well as excited states. For simplicity, let \( \hbar = 197.3 \) MeV-fm (let \( c = 1 \)). Let the mass \( m = 939 \) MeV (you may hardwire this in) (technically \( m = 939 \) MeV/\( c^2 \), but as \( c = 1 \) mass and energy are interconvertible) and let \( V_0 = 50 \) MeV, a typical value. You may “hardwire” them in, although I recommend you introduce \( m \) and \( V_0 \) as variables and then internally set the values. The typical value of the surface thickness \( a \) is 0.2 fermi (1 fm = \( 10^{-15} \) m) while that of the radius \( R \) is usually several fm.

Your wavefunction will be discretized in a “box” from \(-L\) to \(+L\), with \( N \) points, with the step size \( dx = (2L)/(N-1) \). Then your wavefunction will be a vector \( \psi(i) \) with \( \psi(1) = \psi(-L) \) and \( \psi(N) = \psi(+L) \). Hence \( x = -L + dx*(i-1) \).

Your Hamiltonian will be a matrix, but in this case it will be a tridiagonal matrix. The Hamiltonian will have two parts. The potential energy is easy, it is just diagonal, and \( V(i) = V(x) \) with \( x = -L + dx*(i-1) \).

The kinetic energy is slightly subtler, but not much. Remember that it is

\[ -\frac{\hbar^2}{2m} \frac{d^2}{dx^2} \psi, \]

and on a lattice of equally spaced points we can discretize the second derivate with a symmetric three-point formula: \( = (\psi(i+1) + \psi(i-1) - 2*\psi(i))/dx**2 \).

Hence the kinetic energy contributes to the diagonal a constant

\[ +hbar**2/mass/dx**2 \]

and along the off-diagonal also a constant

\[ -0.5*hbar**2/mass/dx**2 \]

Adding to the potential energy, and you have a discretized representation of the Hamiltonian.

You will allow the user to input the following:
(a) The size of the “box”, $L$, that is, from $-L$ to $+L$. You input the value of $L$.
(b) The number of points, $N$, on your lattice. A typical number should be 100-200; however, you should check that your results are not sensitive to the choice of $N$.
(c) A way to choose $R$. (You may “hardwire” $m$, $a$, and $V_0$ if you wish).

Hints for constructing and validating your program.
Step 1. Write a routine that generate the kinetic energy (both off-diagonal and diagonal) matrix. Diagonalize this matrix and sort (very important!) the eigenvalues/eigenvectors. You may choose any $L$. Plot the g.s. wavefunction and first excited state. It should be cosine and sine functions, specifically, $\cos(\pi x/2L)$ for the g.s. and $\sin(\pi x/L)$ for the first excited state. The energy should be $\hbar^2 \pi^2 / 8mL^2$ for the g.s. (here $\hbar = 1$) and 4 times this for the first excited state. This is the infinite well/“particle in a box” problem from quantum mechanics.
Step 2. Now try the harmonic oscillator. Add a potential energy $1/2 k x^2$, with a value of $k = \hbar^2 / m$. Then the g.s. energy should be $= 0.5 \hbar^2 / m$ and the first excited energy should be $1.5 \hbar^2 / m$. The g.s. wfn should be a gaussian.
Step 3. Finally, put in the Woods-Saxon potential. (Do a test here and plot your Woods-Saxon potential and make sure it looks like the figure given on the class web page.)
Step 4. To be sure, try increasing both $L$ and $N$. Your answers, at least for any negative energies, should not change very much as you increase either. (Positive energy, which are “unbound” can indeed change with $L$.)

The program should ask the user for:
$R$, radius of the well, $L$, the size of the box (should be at least twice $R$, but you will want to vary and check it) and the number of integration points $N$. (It could optionally ask for $m$, $V_0$, etc; you may “hardwire” them it but in this case I recommend you have them as variables that are set internally, that is, do not be writing out 939.0 everywhere you would normally put $m$.) Be sure to check that your results do not depend upon $N$.

The program should write out the energies of the first few states.

For this assignment, produce the following plot: The g.s. energies for the Woods-Saxon potential in units of MeV as the radius $R$ varies from 2 to 10 fm. (Your program does not need to loop over $R$; you may run it for different values of $R$ and plot the results.) Also as you go from $R$= 2 to 10 fm, plot the energies (in MeV) of the $1^{\text{st}}$, $2^{\text{nd}}$, and $3^{\text{rd}}$ excited states as a function of $R$.

Program notes: (1) You will need to call the subroutine eigsrtto sort the eigenvalues into proper order. (2) You can either call the program eig (which is easier) or the program tqli in the file eiglib.f. I call eig in one of my demo programs, but here it is again:
call eig(A,n,np,e,vec,work)
Here A is a matrix set with dimension np x np; e is the vector of eigenvalues, dimensioned np; and vec is a matrix np x np of the eigenvectors; and work is a real vector of dimension np. Although np is the physical dimension, n is the "actual" dimension of your matrix. If you dynamically allocate your arrays, you can easily set $n = np$. Otherwise choose np to be some large value (say 1000) and (important) put in an error trap if the
user chooses $n > n_p$ (otherwise I will grade you down). While $n_p$ can be hardwired, do not hardwire $n$; let the user choose $n$.

If you call the latter, note that in the calling line
\[
\text{call } \text{tqli}(d,e,n,n_p,z)
\]
that $d$ is the diagonal and $e$ the vectors of off-diagonal matrix elements, each dimensioned $n_p$; $z$ is the matrix of eigenvectors but **must** be initialized as a unit matrix (i.e., zero everywhere but $= 1$ along the diagonal). In principle skipping \text{tred2} (which does the Householder reduction) and going directly to \text{tqli} (which you can do because your matrix is already in tridiagonal form) saves time, but in practice I find it doesn't save much time.

Note: you must also call \text{eigsrt} afterwards.

C coders: I've put the routines \text{tred2.c} and \text{tqli.c} from Numerical Recipes, along with a sample calling code \text{speedeigen.c}. You will need a sorting routine (not included, sorry). You may also need \text{nrutil.h} and \text{nrutil.c}

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

To make identification easier for me please use the convention \text{lastname_proj4.f} for your program. Also, your program should print out the following at the beginning:

You must submit the following:
1. The program itself, using the convention \text{lastname_proj4.f}.
2. A sample input batch file and output runs
3. A plot of the energies (in MeV) as a function of $R$ (in fm).
4. To put these all together, make a directory \text{lastname} and put the above files in it. Then tar and gzip the files and send them to me.

Due: 11:59 pm Monday Nov 14, 2016. This project is worth a full “advanced” project.

By this time you know what I look for in a code. So I will not automatically send you projects with comments. However, *if you want feedback*, (not fixing but comments on style, etc.) turn in by 11:59 pm Wednesday, Nov 9. I will get back to you by Friday Nov 11.

**HINT: TRY TO GET A WORKING VERSION OF YOUR CODE TO COMPILE AND RUN BY NOV 9. DO NOT WAIT UNTIL THE LAST MOMENT TO GET IT TO WORK.** (This goes especially for C/C++ programmers.) I will be happy to help...but the best thing is to come with me with *specific* questions.

Also keep in mind we have at least 2 or 3 more projects to go, which I will assign as we go through the material!