Solving the Time-Independent Schrödinger Equation Computationally

Anne Stratman
Computational Lab in Quantum Mechanics, Spring 2018, University of Notre Dame

Introduction: Discrete Basis

The time-independent Schrodinger equation, given by

$$-\frac{\hbar^2}{2m}\frac{d^2}{dx^2}\psi(x)+V(x)\psi(x)=E\psi(x) \tag{1}$$

can be solved as a matrix diagonalization problem. This problem can be solved in a discrete basis, or a harmonic oscillator basis. The solution in the discrete basis is found using inner products of distinct points ($\langle x_i|H|x_j\rangle$). To solve the problem in the discrete basis, one rewrites the second derivative in the Schrodinger equation in terms of the value of ψ at three discrete points. The resulting equation can then be used to define diagonal and nondiagonal matrix elements. Each of these matrix elements is evaluated at a discrete points, and the resulting matrix, is diagonalized. This gives the Hamiltonian for the potential, or a set of eigenvalues and eigenvectors which correspond to the allowed energies and their wavefunctions. To begin this process, the second derivative can be rewritten as

$$f''=\frac{f(x+h)-2f(x)+f(x-h)}{h^2}+O(h^2) \tag{2}$$

where h is the step size. Next, we can define minimum and maximum values for the variable x, R_{min} and R_{max}, respectively. For a given number of steps N, the step size h can be defined by

$$h=\frac{R_{max}-R_{min}}{N} \tag{3}$$

Defining an arbitrary value of x as x=R_{min}+kh, k=1,2,...,N_{step}-1, the Schrodinger equation can be rewritten for x_k as

$$-\frac{\hbar^2}{2m}\frac{\psi(x_k+h)-2\psi(x_k)+\psi(x_k-h)}{h^2}+V(x)\psi(x_k) \\ =E\psi(x_k) \tag{4}$$

or, more compactly,

$$-\frac{\hbar^2}{2m}\frac{\psi_{k+1}-2\psi_k+\psi_{k-1}}{h^2}+V_k\psi_k=E\psi_k \tag{5}$$

where ψ_k=$\psi(x_k)$ and $\psi_{k\pm1}$=$\psi(x_k\pm h)$. Next, we can define diagonal matrix elements:

$$d_k=\frac{\hbar^2}{2m}\frac{2}{h^2}+V_k \tag{6}$$

and non-diagonal matrix elements:

$$e_k=-\frac{\hbar^2}{2m}\frac{1}{h^2} \tag{7}$$

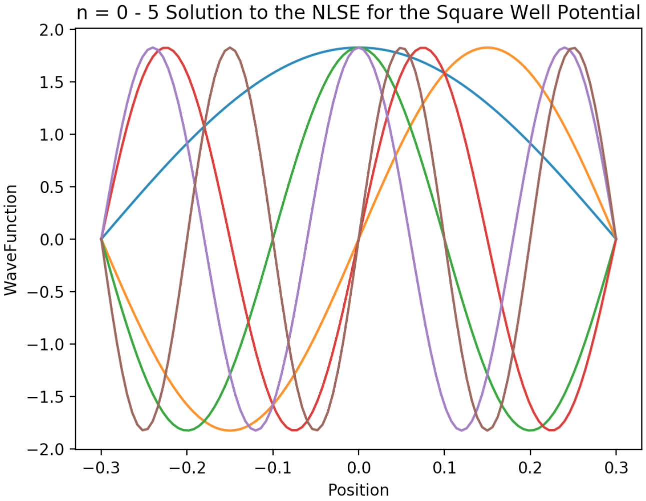

FIG. 1. Square well solved in the discrete basis

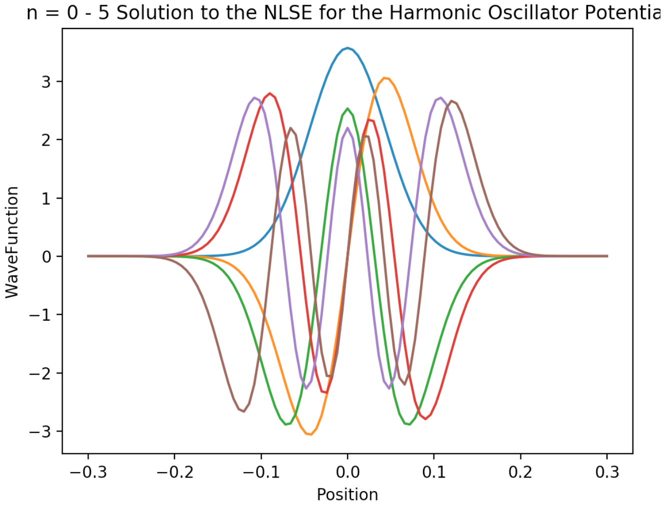

FIG. 2. Harmonic oscillator solved in the discrete basis

This gives the following form for the Schrodinger equation:

$$d_k\psi_k+e_{k-1}\psi_{k-1}+e_{k+1}\psi_{k+1}=E\psi_k \tag{8}$$

and allows us to solve the Schrodinger equation as a matrix eigenvalue problem.

Introduction: Harmonic Oscillator Basis

The harmonic oscillator basis solution to the Schrodinger equation uses inner products of wavefunctions

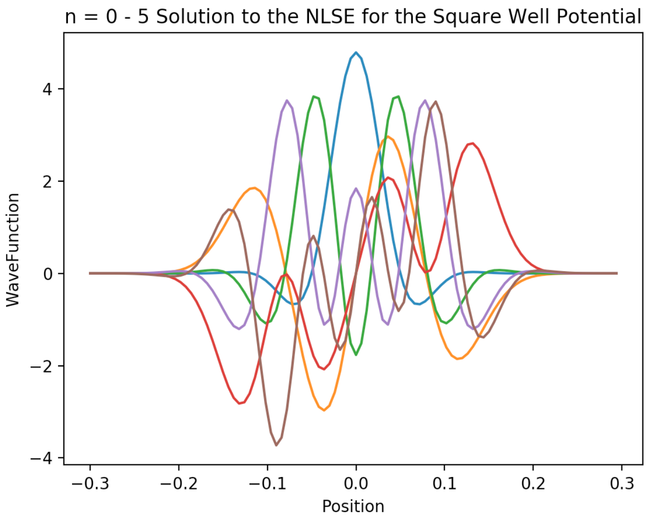

FIG. 3. Square well solved in the harmonic oscillator basis

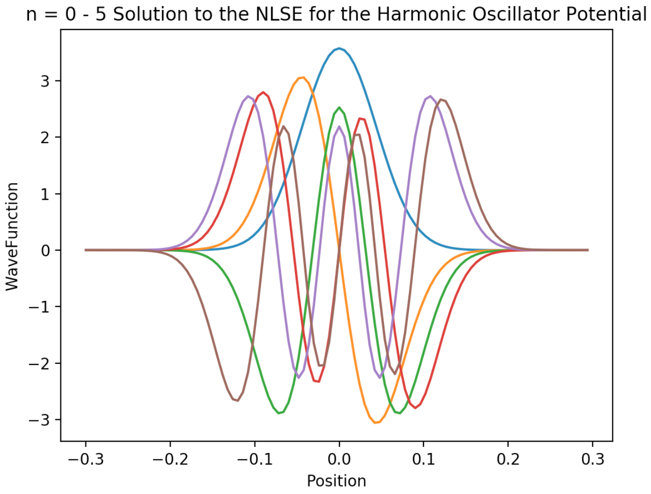

FIG. 4. Harmonic oscillator solved in the harmonic oscillator basis

($\langle\psi_m|H|\psi_n\rangle$) to find the matrix elements, and transforms the resulting Hamiltonian into the harmonic oscillator basis. The solutions to the harmonic oscillator are given by the equation

$$\psi_n(x) = \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} \frac{1}{\sqrt{2^n n!}} H_n(\xi) e^{-\xi^2/2} \tag{9}$$

where

$$\xi = \sqrt{\frac{m\omega}{\hbar}} x \tag{10}$$

The matrix elements are found by taking the inner product $\langle m|H|n\rangle$, which yields

$$\begin{aligned} &\frac{1}{2m}\langle n|p^2|m\rangle + \langle n|V|m\rangle \\ &= -\frac{\hbar\omega}{4}[\sqrt{m+1}\sqrt{m+2}\delta_{n,m+2} - \sqrt{m+1}\sqrt{m+1}\delta_{m,n} \\ &\qquad - \sqrt{m}\sqrt{m}\delta m, n + \sqrt{m}\sqrt{m+1}\delta_{n,m-2}] \\ &\qquad\qquad + \int_{-\infty}^{\infty} \psi_n^*(x)V(x)\psi_m(x)dx \end{aligned} \tag{11}$$

After the matrix elements are found, the resulting matrix is diagonalized to give the Hamiltonian, which has dimensions (number of wavefunctions, number of wavefunctions). This matrix is then multiplied by a transformation matrix with dimensions (number of steps, number of wavefunctions) to give the solution in the harmonic oscillator basis.

Programming

We structured our code by defining two classes, one for the discrete basis and one for the harmonic oscillator basis. Both classes have attributes *potential* (the potential function), x_{min}, x_{max}, *n steps* (number of steps - used for determining matrix dimensions), *particle mass*, *h* (to determine the step size), and *xPoints* (to determine the points to evaluate the wavefunction at). The harmonic oscillator class also has attributes *omega* and *hbar*.

The discrete class includes separate functions for finding each element of the matrix, creating the matrix, and diagonalizing the matrix. The “matrix element finder” function uses equations (6) and (7) to find the diagonal and off-diagonal matrix elements, respectively. The “matrix maker” function creates a zero matrix of dimensions (*n steps* + 1, *n steps* + 1). This function then calls “matrix element finder” to fill in the diagonal and off-diagonal elements of the matrix, giving the Hamiltonian. The “matrix solver” function uses Numpy’s “linalg.eigh” function to diagonalize the Hamiltonian, and then takes the transpose of this matrix, giving the eigenvectors and eigenvalues.

The harmonic oscillator class first includes a function for defining the harmonic oscillator wavefunction. This function calls a separate module to evaluate Hermite polynomials. The momentum and potential inner products corresponding to the Hamiltonian are defined in separate functions. In addition, a function for defining (NOT integrating) the integrand in the potential term is defined separately. A “matrix element finder” function adds the momentum operator and potential operator terms, and “matrix maker” and “matrix solver” functions are defined similarly to those in the discrete class. The zero matrix initially created by “matrix maker” has dimensions (number of functions, number of functions).

The plotting function, defined outside the two classes, includes a component for transforming the Hamiltonian given by the harmonic oscillator solver into the harmonic oscillator basis. The transformation component takes the Numpy dot product of a transformation matrix and the eigenvector matrix of given by the harmonic oscillator solver. The transformation matrix has dimensions (number of steps, number of functions), with elements representing each of the harmonic oscillator functions evaluated at a different point; thus multiplying the transformation matrix and the eigenvector matrix gives the solution in the harmonic oscillator basis.

Analysis of Correctness

To test whether this program gives the correct results, we used our code to solve problems with known solutions, specifically the infinite square well and the harmonic oscillator. Various plots of our solutions are shown throughout the report. *Figure 5* and *Figure 6*, which show the correct solutions to the infinite square well and the harmonic oscillator, respectively, are taken from Griffiths' *Introduction to Quantum Mechanics.*

As shown in *Figure 3*, the solution to the square well potential solved in the harmonic oscillator basis is not correct, which is expected. When solving the TISE in the harmonic oscillator basis, one should integrate the inner product out to positive and negative infinity (the SciPy "integrate.quad" function does this). When solving the square well, however, one should just integrate the inner products between the endpoints of the well. Therefore, the limits of integration in the harmonic oscillator basis are much too large for solving square well potentials, and should not yield the correct results.

Figure 2 and *Figure 4* show that certain solutions for the harmonic oscillator potential in the harmonic oscillator basis are negative solutions of the results from the discrete basis. Since any "negative" solution to an eigenvalue problem is still a solution, though, this is no cause for concern.

To achieve reasonable numerical precision, at least 100 points should be used when solving a potential in the discrete basis. This yields at least 100 points at which the code can evaluate the wavefunction solutions and plot the results. In the harmonic oscillator basis, SciPy's "integrate.quad" function integrates the inner products of wavefunctions and the potential out to positive and negative infinity, which effectively takes the place of evaluating the inner products at many discrete points, as is done in the discrete basis solver. Solutions to potentials in the harmonic oscillator basis vary depending on the number of basis functions used, and because the inner products are already integrated out to extremely large values, a much smaller number of basis functions yields an accurate solution. We found accurate solutions using as few as five basis functions.

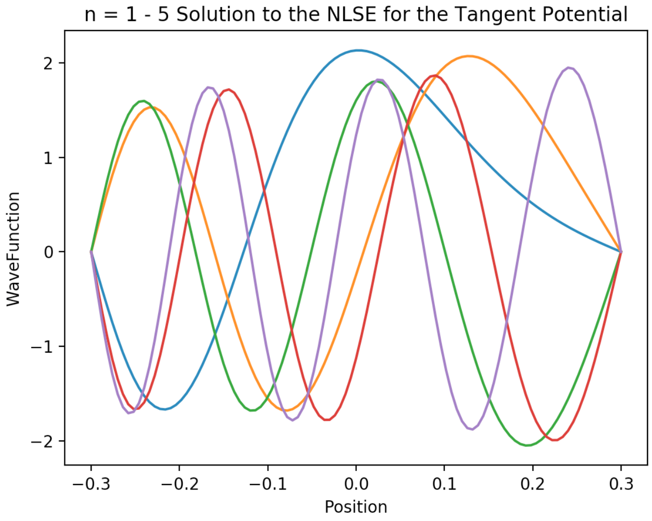

FIG. 5. Tangent potential solved in the discrete basis

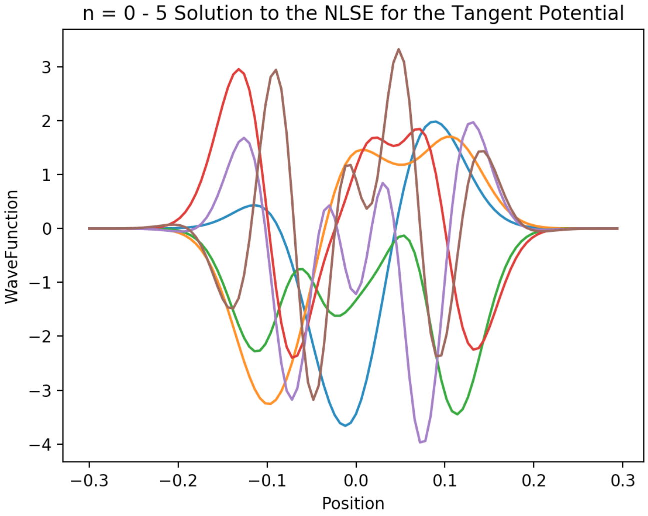

FIG. 6. Tangent potential solved in the harmonic oscillator basis

We also used this code to solve the tangent potential $V(x) = tan(x)$ in both the discrete and harmonic oscillator bases.

Units

The units of mass in this code are keV. In the harmonic oscillator basis, we worked in units such that $\hbar$ and ω were equal to 1.

Conclusion

The Schrodinger equation can be solved computationally through matrix diagonalization in multiple bases. The discrete basis solver evaluates the Schrodinger equation at many individual points