Physics 514 – Single Particle Quantum Mechanics

Emanuel Gull

November 1, 2016

Chapter 1

Single Particle Quantum Mechanics

1.1 The time-independent one-dimensional Schrödinger equation

We will start with the time-independent one-dimensional Schrödinger equation for a particle with mass m in a potential $V(x)$. For this problem the time-dependent Schrödinger equation

$$i\hbar\partial_t\psi = \frac{-\hbar^2}{2m}\partial_x^2\psi + V(x)\psi \tag{1.1}$$

simplifies to an ordinary differential equation using the ansatz $\psi(x,t) = \psi(x)\exp(-iEt)$:

$$E\psi = \frac{-\hbar^2}{2m}\partial_x^2\psi + V(x)\psi \tag{1.2}$$

1.1.1 The Numerov algorithm

After rewriting this second order differential equation to a coupled system of two first order equations, any ODE solver such as the Runge-Kutta method can be applied. However, there are special algorithms which perform better for this particular form of differential equation. We write the equation as

$$\psi''(x) + k(x)\psi(x) = 0 \tag{1.3}$$

with $k = 2mV(x)/\hbar^2$ and start from a Taylor series of $\psi_n = \psi(n\Delta x)$:

$$\psi_{n\pm1} = \psi_n \pm \Delta x\psi_n' + \frac{\Delta x^2}{2}\psi_n'' \pm \frac{\Delta x^3}{3!}\psi_n^{(3)} + \frac{\Delta x^4}{24}\psi_n^{(4)} \pm \frac{\Delta x^5}{120}\psi_n^{(5)} + O(\Delta x^6) \tag{1.4}$$

thus, adding ψ_{n+1} and ψ_{n-1} we obtain

$$\psi_{n+1} + \psi_{n-1} = 2\psi_n + (\Delta x)^2 \psi_n'' + \frac{\Delta x^4}{12}\psi_n^{(4)} \tag{1.5}$$

Replacing the fourth derivatives by a finite difference second derivative of the second derivatives:

$$\psi_n^{(4)} = \frac{\psi_{n+1}'' + \psi_{n-1}'' - 2\psi_n''}{\Delta x^2} \tag{1.6}$$

and substituting the differential equation $-k(x)\psi(x)$ for $\psi''(x)$ we obtain the Numerov algorithm:

$$\left(1 + \frac{\Delta x^2}{12} k_{n+1}\right)\psi_{n+1} = 2\left(1 - \frac{5\Delta x^2}{12} k_n\right)\psi_n - \left(1 + \frac{\Delta x^2}{12} k_{n-1}\right)\psi_{n-1} + O(\Delta x^6) \tag{1.7}$$

This means that the Numerov algorithm is locally of 6^{th} order!

The Numerov algorithm will need the wave function at two initial points. Depending on the problem and its symmetry, different choices are ideal:

For potentials $V(x)$ with reflection symmetry $V(x) = V(-x)$ the wave functions need to be either even ($\psi(x) = \psi(-x)$) or odd ($\psi(x) = -\psi(-x)$). We can use this to find initial values:

- For the even solution we use a half-integer mesh with mesh points $x_{n+1/2} = (n+1/2)\Delta x$ and pick initial values $\psi(x = -1/2) = \psi(x = 1/2) = 1$.

- For the odd solution we know that $\psi(0) = -\psi(0)$ and therefore $\psi(0) = 0$. This gives one starting value. Using an integer mesh with mesh points $x_n = n\Delta x$ we pick $\psi(x_1) = 1$ as the second starting value.

In general potentials we need to use another approach. If the potentials vanish for large distance, *i.e.* $V(x) = 0$ for $|x| \geq a$ we can use the exact solution of the Schrödinger equation at large distances to define starting points, *e.g.*

$$\psi(-a) = 1 \tag{1.8}$$

$$\psi(-a - \Delta x) = \exp(-\Delta x\sqrt{2mE}/\hbar) \tag{1.9}$$

Finally, if the potential never vanishes we need to begin with a single starting value $\psi(x_0)$ and obtain the second starting value $\psi(x_1)$ by performing an integration of the first spatial step Δx with an Euler scheme or a Runge Kutta scheme.

1.1.2 The 1d scattering problem

The scattering problem is the easiest quantum problem to solve numerically. Solutions exist for all energies $E > 0$ if the potential vanishes at large distance

($V(x) \to 0$ for $x \to \infty$). The solution becomes particularly simple if the potential is nonzero only on a finite interval $[0, a]$. For a particle approaching the potential barrier from the left ($x < 0$) we can make the following ansatz for the free propagation when $x < 0$:

$$\psi_L(x) = A \exp(-ikx) + B \exp(ikx) \tag{1.10}$$

where A is the amplitude of the incoming wave and B the amplitude of the reflected wave. On the right hand side, once the particle has left the region of finite potential ($x > a$), we can again make a free propagation ansatz,

$$\psi_R(x) = C \exp(-ikx) \tag{1.11}$$

The coefficients A, B, and C have to be determined self-consistently by matching to a numerical solution of the Schrödinger equation in the interval $[0, a]$. This is best done in the following way:

- Set $C = 1$ and use the two points a and $a + \Delta x$ as starting points for a Numerov integration.
- Integrate the Schrödinger equation numerically – backwards in space, from a to 0, using the Numerov algorithm.
- Match the numerical solution of the Schrödinger equation for $x < 0$ to the free propagation ansatz to determine A and B.

Once A and B are known, the reflection and transmission probabilities R and T are given by

$$R = |B|^2/|A|^2 \tag{1.12}$$
$$T = 1/|A|^2 \tag{1.13}$$

1.1.3 Bound states in 1d and the solution of the eigenvalue problem

While there exist scattering states for all energies $E > 0$, bound state solutions of the Schrödinger equation with $E < 0$ exist only for discrete energy eigenvalues. Integrating the Schrödinger equation from $-\infty$ to ∞ the solution will diverge to $\pm\infty$ as $x \to \infty$ for almost all values. These functions cannot be normalized and thus do not constitute a solution to the Schrödinger equation. Only for some special eigenvalues E will the solution go to zero as $x \to \infty$.

A simple eigensolver can be implemented by using the shooting method, where we again assume that the potential is zero outside an interval $[0, a]$:

- Start with an initial guess E
- Integrate the Schrödinger equation for $\psi_E(x)$ from $x = 0$ to $x_f \gg a$ and determine the value $\psi_E(x_f)$.

- use a root solver, such as the bisection method, to look for an energy E with $\psi_E(x_f) \approx 0$.

This algorithm is pretty bad, since the divergence of the wave function for $x \pm \infty$ will cause roundoff errors to proliferate. A better solution is to integrate the Schrödinger equation from both sides towards the center:

- We search for a point b with $V(b) = E$
- Starting from $x = 0$ we integrate the left hand side solution $\psi_L(x)$ to a chosen point b and obtain $\psi_L(b)$ and a numerical estimate for $\psi'_L(b) = (\psi_L(b) - \psi_L(b - \Delta x))/\Delta x$.
- Starting from $x = a$ we integrate the right hand solution $\psi_R(x)$ backwards to the same point b and obtain $\psi_R(b)$ and a numerical estimate for $\psi'_R(b) = (\psi_R(b + \Delta x) - \psi_R(b))/\Delta x$.
- At the point b the wave functions and their first two derivatives have to match, since solutions to the Schrödinger equation have to be twice continuously differentiable. Keeping in mind that we can multiply the wave functions by an arbitrary factor we obtain the conditions

  $$\psi_L(b) = \alpha\psi_R(b) \tag{1.14}$$

  $$\psi'_L(b) = \alpha\psi'_R(b) \tag{1.15}$$

  $$\psi''_L(b) = \alpha\psi''_R(b) \tag{1.16}$$

  The last condition is automatically fulfilled since $\psi''(b) = 0$ by the choice of $V(b) = E$. The first two conditions can be combined to the condition that the logarithmic derivatives vanish:

  $$\left.\frac{d\log\psi_L}{dx}\right|_{x=b} = \frac{\psi'_L(b)}{\psi_L(b)} = \frac{\psi'_R(b)}{\psi_R(b)} = \left.\frac{d\log\psi_R}{dx}\right|_{x=b} \tag{1.17}$$

- Solve this last equation for b using the shooting method, *e.g.* using a bisection algorithm.

1.2 Time-independent Schrödinger equations in more than one dimension

The time-independent Schrödinger equation in more than one dimension is a partial differential equation and cannot, in general, be solved using a simple ODE solver like the Numerov algorithm. Before employing a PDE solver we should always try to reduce the problem to a one-dimensional problem, *e.g.* by making use of symmetries or factorizations.

A first example is the three-dimensional Schrödinger equation in a cubic box with potential $V(r) = V(x)V(y)V(z)$ with $\mathbf{r} = (x, y, z)$. Using the product ansatz

$$\psi(\mathbf{r}) = \psi_x(x)\psi_y(y)\psi_z(z) \tag{1.18}$$

the PDE factorizes into three ODEs which can be solved as above.

Another trick is possible for spherically symmetric potentials with $V(\mathbf{r}) = V(|\mathbf{r}|)$. An ansatz using spherical harmonics,

$$\psi_{lm}(\mathbf{r}) = \psi_{lm}(r, \phi, \theta) = \frac{u(r)}{r} Y_{lm}(\theta, \phi) \tag{1.19}$$

can be used to reduce the three-dimensional Schrödinger equation to a one-dimensional one for the radial wave function $u(r)$:

$$\left[\frac{-\hbar^2}{2m}\frac{d^2}{dr^2} + \frac{\hbar^2 l(l+1)}{2mr^2} + V(r)\right] u(r) = Eu(r) \tag{1.20}$$

in the interval $[0, \infty[$. Given the singular character of the potential for $r \to 0$ the numerical integration should start at large distances r and integrate towards $r = 0$, so that the largest errors are accumulated only at the last steps of the integration.

1.2.1 Variational solutions using a finite basis set

In the case of general potentials, or for more than two particles, it is not possible in general to reduce the Schrödinger equation to a one-dimensional problem and we need to employ a PDE solver. One approach will be, as before, to use a finite difference method and discretize the Schrödinger equation on a mesh using a finite difference approximation. A better solution is to expand the wave functions in terms of a finite set of basis functions

$$|\phi\rangle = \sum_{i=1}^{N} a_i |u_i\rangle, \tag{1.21}$$

as with finite element methods.

To estimate the ground state energy we want to minimize the energy of the variational wave function

$$E^* = \frac{\langle\phi|H|\phi\rangle}{\langle\phi|\phi\rangle} \tag{1.22}$$

Since the variational basis set is finite, the variational estimate E^* will always be larger than the true ground state E_0, but will converge towards E_0 as the size of the basis set is increased*e.g.* by reducing the mesh size in a finite element basis.

To perform the minimization we denote by

$$H_{ij} = \langle u_i|H|u_j\rangle = \int d\mathbf{r} u_i(\mathbf{r})^* \left(\frac{-\hbar^2}{2m}\nabla^2 + V\right) u_j(\mathbf{r}) \tag{1.23}$$

the matrix elements of the Hamiltonian H and by

$$S_{ij} = \langle u_i|u_j\rangle = \int d\mathbf{r} u_i(\mathbf{r})^* u_j(\mathbf{r}) \tag{1.24}$$

the overlap matrix. Note that for an orthogonal basis set, S_{ij} is the identity matrix δ_{ij}. Minimizing equation 1.22 we obtain a generalized eigenvalue problem

$$\sum_j H_{ij} a_j = E \sum_k S_{ik} a_k \tag{1.25}$$

or in a compact notation with $\mathbf{a} = (a_1, \ldots, a_N)$

$$H\mathbf{a} = ES\mathbf{a} \tag{1.26}$$

If the basis set is orthogonal this reduces to an ordinary eigenvalue problem and we can use an eigensolver (typically a sparse algorithm like the Lanczos algorithm, see next section).

In the general case we have to find orthogonal matrices U such that $U^T SU$ is the identity matrix. Introducing a new vector $\mathbf{b} = U^{-1}\mathbf{a}$ we can rearrange the problem into

$$H\mathbf{a} = ES\mathbf{a} \tag{1.27}$$

$$HU\mathbf{b} = ESU\mathbf{b} \tag{1.28}$$

$$U^T HU\mathbf{b} = EU^T SU\mathbf{b} = E\mathbf{b} \tag{1.29}$$

This is a standard eigenvalue problem for $U^T HU$. Mathematica, Python (scipy) and LAPACK contain general purpose eigensolvers for generalized eigenvalue problems.

The precision of this method is very strongly dependent on the quality of the basis functions. While it is possible to run this algorithm with standard finite element functions, it is advantageous to make use of known solutions to similar problems to generate basis functions which already have more or less the right behavior.

An example is the anharmonic oscillator with Hamiltonian

$$H = H_0 + \lambda q^4 \tag{1.30}$$

$$H_0 = \frac{p^2 + q^2}{2}, \tag{1.31}$$

where the momentum operator is $p = i\hbar\partial_q$. The eigenstates $|n\rangle$ and eigenvalues $\epsilon_n = (n + 1/2)$ of H_0 are known from the introductory quantum mechanics lectures. In real space the eigenstates are given by

$$\phi_n(q) = \frac{1}{\sqrt{2^n n! \sqrt{\pi}}} \exp\left(-\frac{1}{2}q^2\right) H_n(q), \tag{1.32}$$

where the H_n are the Hermite polynomials. Using these eigenstates as a basis set, the operator H_0 becomes diagonal. The position operator is

$$q = \frac{1}{\sqrt{2}}(a^\dagger + a), \tag{1.33}$$

where the raising and lowering operators $a^\dagger$ and a only have the following matrix elements nonzero:

$$\langle n+1|a^\dagger|n\rangle = \langle n|a|n+1\rangle = \sqrt{n+1}. \tag{1.34}$$

The matrix representation of the anharmonic term λq^4 is a banded matrix. After truncation of the basis set to a finite number of states N, a sparse eigensolver (such as the Lanczos algorithm) can be used to calculate the spectrum. In this particular problem, the basis functions are orthonormal and the overlap matrix S is the unit matrix.

1.2.2 The Lanczos Algorithm

Sparse matrices are matrices which have only $O(N)$ non-zero elements. They are very common in scientific calculations, and appear *e.g.* in the discretization of finite difference problems or in the solution of finite element problems (see the chapter on PDEs). Using the properties of sparse matrices becomes important for large N, because the algorithmic scaling for sparse matrices is often vastly better than for dense matrices. The scaling table for typical operations of an $N \times N$ sparse matrix is:

operation	time (full)	time (sparse)	mem (full)	mem (sparse)
storage	-	-	N^2	$O(N)$
matrix-vector	$O(N^2)$	$O(N)$	$O(N^2)$	$O(N)$
matrix-matrix	$O(N^3)$	$O(N)...O(N^2)$	$O(N^2)$	$O(N)...O(N^2)$
full eigensystem	$O(N^3)$	$O(N^2)$	$O(N^2)$	$O(N^2)$
lowest EV	$O(N^2)$	$O(N)$	$O(N^2)$	$O(N)$

An important problem is the determination of some of the eigenvalues and eigenvectors of a system, typically the lowest few eigenvalues. This is usually done by the Lanczos algorithm.

To motivate the Lanczos algorithm we start from the power method for a matrix A: Starting from a random initial vector u_1 we calculate the sequence

$$u_{n+1} = \frac{Au_n}{||Au_n||} \tag{1.35}$$

This series converges to the eigenvector of the largest eigenvalue of the matrix A. The Lanczos method is an optimization of this power method.

Lanczos iterations

The Krylov subspace K_M of a vector u and a matrix A is defined as $K_M = \text{span}\{u, Au, \ldots, A^M u\}$ The Lanczos algorithm builds a basis $\{v_1, v_2, \ldots, v_m\}$ for this 'Krylov' subspace $K_M = \text{span}\{u_1, u_2, \ldots, u_M\}$, which is constructed by

M iterations of the power method 1.35. This is done by the following iterations:

$$\begin{aligned}\beta_{n+1} v_{n+1} &= A v_n - \alpha_n v_n - \beta_n v_{n-1} && (1.36)\\ \alpha_n &= v_n^\dagger A v_n && (1.37)\\ \beta_n &= |v_n^\dagger A v_{n-1}|. && (1.38)\end{aligned}$$

The β_n can be chosen to be real and positive. The method requires the storage of v_n, α_n, and β_n, which are each only vectors of size N. This method therefore requires much less storage than a dense matrix eigensolver, which requires $O(N^2)$ storage.

In the Krylov basis the matrix A is tridiagonal:

$$T^{(n)} := \begin{pmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \beta_2 & \alpha_2 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_n \\ 0 & \cdots & 0 & \beta_n & \alpha_n \end{pmatrix} \qquad (1.39)$$

The eigenvalues $\{\tau_1, \dots, \tau_M\}$ of T are good approximations of the eigenvalues of A. The extreme eigenvalues converge very fast. Thus $M \ll N$ iterations are sufficient to obtain the extreme eigenvalues.

1.2.3 Eigenvectors

The eigenvalues of T can easily be computed, but the will be given in the Krylov basis. To obtain the eigenvectors in the original basis we need to perform a basis transform to the original basis. This can be done iteratively by running the Lanczos iterations a second time.

1.2.4 Roundoff errors and ghosts

By construction, the vectors $\{v_i\}$ are orthogonal and the Lanczos iteration will stop after $N-1$ steps (when all eigenvalues are found).

However, finite numerical precision causes roundoff errors and loss of orthogonality. There are two ways to deal with this: Either the vectors are re-orthogonalized after every step. This requires storing all of the vectors $\{v_i\}$ and is memory intensive. Alternatively we can control the effects of roundoff errors.

The second solution is faster and needs less memory. The main effect of roundoff errors is that the matrix T contains extra spurious eigenvalues, called 'ghosts'. These ghosts are not real eigenvalues of A. However, they converge towards real eigenvalues of A over time and increase their multiplicities.

A simple criterion distinguishes ghosts from real eigenvalues. Ghosts are caused by roundoff errors, and therefore they do not depend on the starting vector. As a consequence of this they are also eigenvalues of the matrix $\tilde{T}$, defined as the matrix T with the first row and column removed:

$$\tilde{T}^{(n)} := \begin{pmatrix} \alpha_2 & \beta_3 & 0 & \cdots & 0 \\ \beta_3 & \alpha_3 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_n \\ 0 & \cdots & 0 & \beta_n & \alpha_n \end{pmatrix} \tag{1.40}$$

And this can be used as a heuristic criterion to distinguish ghosts from real eigenvalues:

- All multiple eigenvalues are real, but their multiplicities might be too large
- All single eigenvalues of T which are *not* eigenvalues of $\tilde{T}$ are also real

You can find numerically stable and efficient implementations of the Lanczos algorithm from `netlib`. The method can be generalized to non-Hermitian matrices. This is the Arnoldi algorithm; a popular implementation is available as `ARPACK` from Rice[1].

1.3 The time-dependent Schrödinger equation

So far we have been studying static equilibrium systems. We will now reintroduce the time dependence to study dynamics in non-stationary quantum systems.

1.3.1 Spectral methods

In a stationary system we can introduce a basis, diagonalize the Hamiltonian, and then obtain the time dependence by time-evolving the eigenvalues.

To calculate the time evolution of an initial state $|\psi(t_0)\rangle$ from time t_0 to time t we first solve the stationary eigenvalue problem

$$H|\phi\rangle = E|\phi\rangle \tag{1.41}$$

and calculate the eigenvectors $|\phi_n\rangle$ and eigenvalues ϵ_n. Next we decompose the initial wave function $|\psi\rangle$ into the eigenbasis:

$$|\psi(t_0)\rangle = \sum_n c_n|\phi_n\rangle. \tag{1.42}$$

Since each of the $|\phi_n\rangle$ is an eigenvector of H, the time evolution $\exp(-i\hbar H(t-t_0))$ is trivial and we obtain

$$|\psi(t)\rangle = \sum_n c_n \exp(-i\hbar\epsilon_n(t-t_0))|\phi_n \tag{1.43}$$

[1] http://www.caam.rice.edu/software/ARPACK/

1.3.2 Direct numerical integration

In a non-stationary system with $H = H(t)$ or if the number of basis states is too large to perform a complete diagonalization of the Hamiltonian we need to perform a direct integration of the Schrödinger equation.

The main difference compared to the integration of the classical wave equation is that the exact quantum mechanical time evolution conserves the normalization

$$\langle\psi(t)|\psi(t)\rangle = \int |\psi(x,t)|^2 dx = 1 \tag{1.44}$$

of the wave function. Ideally we construct an algorithm that also has this property: the approximate time evolution needs to be unitary.

We first approximate the time evolution

$$\psi(x, t+\Delta t) = \exp(-iH\Delta t)\psi(x,t) \tag{1.45}$$

by a forward Euler approximation

$$\exp(-iH\Delta t) \approx 1 - iH\Delta t + O(\Delta t^2) \tag{1.46}$$

This is neither unitary nor stable. The simplest stable unitary integrator can be obtained by using:

$$\begin{aligned}\exp(-iH\Delta t) &= (\exp(iH\Delta t/2))^{-1}\exp(-iH\Delta t/2) \qquad (1.47)\\ &\approx \left(1 + iH\frac{\Delta t}{2}\right)^{-1}\left(1 - iH\frac{\Delta t}{2}\right) + O(\Delta t^3) \qquad (1.48)\end{aligned}$$

This gives an algorithm

$$\psi(x, t+\Delta t) = \left(1 + iH\frac{\Delta t}{2}\right)^{-1}\left(1 - iH\frac{\Delta t}{2}\right)\psi(x,t) \tag{1.49}$$

or equivalently

$$\left(1 + iH\frac{\Delta t}{2}\right)\psi(x, t+\Delta t) = \left(1 - iH\frac{\Delta t}{2}\right)\psi(x,t) \tag{1.50}$$

To solve this, we first introduce a finite basis. We then run a linear solver in each step (rather than inverting the matrix in Eq. 1.49) to solve the equation. In one dimensions, the matrix is often tridiagonal. In higher dimensions the matrix is not longer tridiagonal but still sparse, and specialized algorithms (e.g. the biconjugate gradient method, BiCG) exist. Mathworld has a nice introduction available at `http://mathworld.wolfram.com/topics/Templates.html`.