Quantum Solver 1D

Documentation for a simple MATLAB application to simulate one-dimensional quantum systems

This application was spun off out of larger research into the implications of simulating and analyzing time-dependent quantum systems. For the simplest cases, this will stay as a one-dimensional application, as higher-dimensional problems require significantly more computational resources, i.e. $O(n^2)$ and $O(n^3)$, compared to the one-dimensional case.

Uses the Crank-Nicolson finite difference method to break up the temporal and spatial domain, and then solves the resulting linear algebra using the Tridiagonal Matrix Algorithm—or whatever MATLAB uses when solving sparse, tridiagonal systems.

If not packaged as a MATLAB App, this program can be run by simply running the MainQuantumSolverUI m-function.

Principle of Operation

At a fundamental level, all this application does is, given an initial wave function and scalar potential, simulates the time-dependent version of Schrodinger's equation. This also includes a basic magnetic field to simulate coupling between spin states; however, there is no magnetic vector potential, as vector potentials don't really make any sense in one dimension.

The application runs the wave function through a big time loop, measuring it as it evolves at every time step, thus creating a time-based version of the wave function. This can be used to get all sorts of interesting information, like the energies of the stationary states (or modes). The application gives you complete control over the initial conditions of the simulation, as well as a few ways to create potentials for whatever problem you'd like to gain some insight into.

Due to boundary conditions required by the numerical analysis method used, the wave function is forced to go to zero at the spatial boundary (just like a particle in a box), so even if there is no potential present, you'll effectively be simulating an infinite potential well.

The general workflow can be summarized in the following steps:

1) Determine the initial conditions of the simulation:
    a) Parameters of the problem domain, e.g. grid points and time step size,
    b) The scalar potential to be analyzed,
    c) The initial wave function to be simulated,
    d) The existence of a magnetic field,
    e) Where you want to sample the wave function at each step,
2) Run the simulation and watch how it evolves,
3) Stare at or otherwise analyze the results,
4) Export results and initial conditions to the workspace for further study.

There are default parameters already programmed in, so as a first run, just navigate to the Simulate tab, hit start, and then jam on some other buttons to see what happens. The default potential is the standard quantum harmonic potential; the results are not surprising!

The Interface

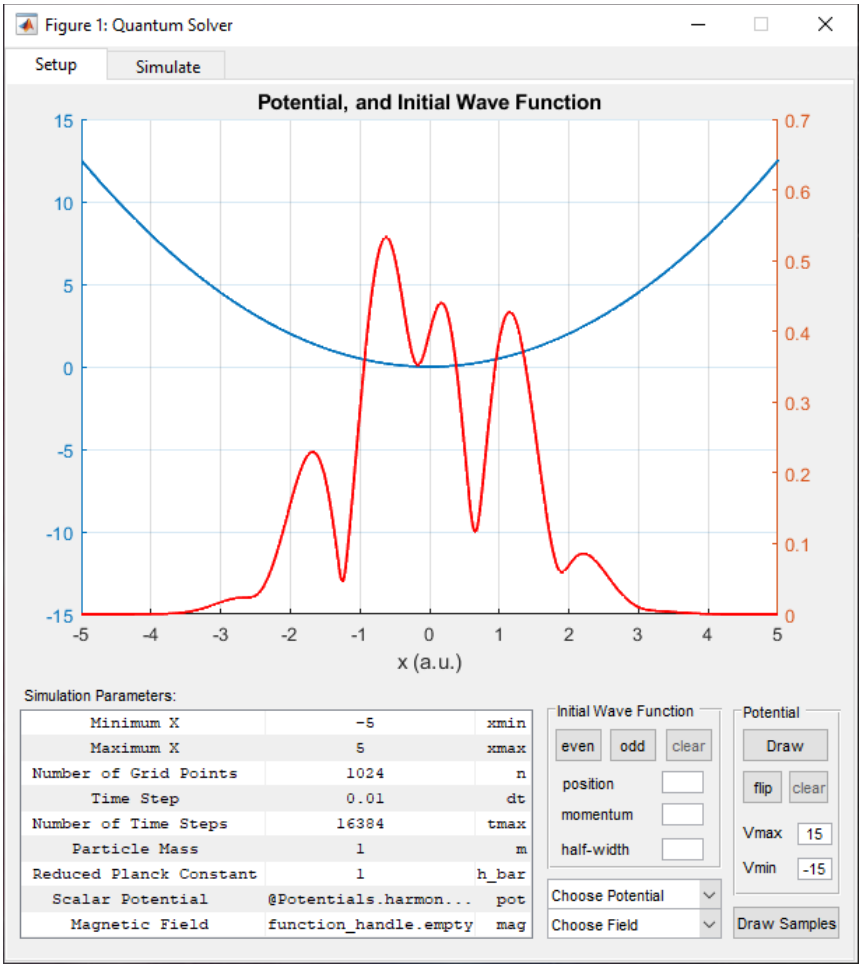
The interface consists of two main tabs:

1) The first to setup the simulation parameters,
2) The second to run the simulation and analyze results.

Things can look overwhelming at first because there's a lot of information to go through, but to get started, you don't really need to do anything other than run through the default parameters over a couple of different potentials to see what happens.

The Setup Tab:

The setup tab is where you set up the initial conditions before you start simulating anything! It's where most of the work is done and it looks like this:



It's not pretty, but for the most part it seems to work. The top two thirds aren't well labeled, since labels on figures seems slow things down. The **blue** line is the potential that you've chosen, and the **red** line is the wave function that you've created to simulate.

The bottom third consists of Simulation Parameters; Initial Wave Function setup; Choosing a Potential; Choosing a Field; Drawing a Potential; and, Drawing the Sample points that the wave function will be sampled at each time step.

Simulation Parameters:

Default simulation parameters are set to be in Hartree atomic units (i.e. m = \hbar = q = k_e = 1), and I don't remember the conversions, but they're on the order of femtoseconds and picometers I think. Because of how the simulation is run,

Minimum X (xmin): The minimum spatial value of your one-dimensional problem domain. Affects the resolution of the Fourier Transform of your space wave function; i.e. in momentum space

Maximum X (xmax): The maximum spatial value of your one-dimensional problem domain. Affects the resolution of the Fourier Transform of your space wave function; i.e. in momentum space.

Number of Grid Points (n): The total number of discrete points between xmin and xmax. Affects the maximum momentum of the Fourier Transform of your space wave function; i.e. in momentum space. Unlike the number of time steps, there are no limitations on the number of grid points, as the wave function is forced to go to zero at the boundary.

Time Step (dt): The time in between simulation steps in the big loop that the wave function is run through. Affects the maximum energy that can be measured by the simulation. The simulation is absolutely stable, so larger time steps will not cause the simulation to blow up; however, aliasing effects can still occur. A warning will be displayed if your time step is too big.

Number of Time Steps (tmax): The total number of time steps to run through in the big simulation loop. The maximum simulation, "time," is then dt * tmax. Affects the resolution of the energy spectrum. Longer absolute simulation times can prevent quantization errors when detecting the energies of stationary states.

> NOTE: Ideally, the total number of time steps needs to be a multiple of two. If not, zero-padding of the Fourier Transform will create sinc-like artifacts to the energy spectrum, complicating analysis. A warning will trigger if tmax is not a multiple of two.

Particle Mass (m): The mass of the particle represented by the wave function. Default is set to one to keep things in Hartree atomic units so that it's easier to compare to theoretical results.

Reduced Planck Constant (h_bar): Planck's constant shows up in quantum mechanics! Default is set to one to keep things in Hartree atomic units so that it's easier to compare to theoretical results.

Electric Potential (pot): The electrical potential that determines the dynamics of the system. Can be determined from the defaults in the Choose Potential popup menu, through an anonymous function typed into the Simulation Parameters table, or, through a function placed in the +Potentials folder.

Magnetic Field (mag): The magnetic field that determines how wave functions are coupled over time between spin up and spin down. If the Magnetic Field is left empty or as the zero field, no coupling will take place. Can be determined from the defaults in the Choose Field popup menu, through an anonymous function typed into the Simulation Parameters table, or, through a function placed in the +Fields folder.

Initial Wave Function

This is the section of the application that allows you to create an initial wave function. If you don't touch this, it will use the default wave function, consisting of one even wave function with a momentum of two, and one odd wave function with a momentum of negative two. Since most simple one-dimensional systems consist of stationary states that are even or odd in nature, we only really need two orthonormal basis functions to stimulate all states as the wave function evolves. In order to stimulate higher order modes, the wave function needs to have either a higher momentum, or be physically located so that the overlap integral between initial wave function and suspected stationary state is maximized.

Even wave functions are normalized Gaussians, and odd wave functions are the first derivative of the same. Blank position, momentum or half-width are the same as entering zero. The half-width of the Gaussian can affect aliasing, as skinnier Gaussians require way more information in momentum-space to create, thus requiring a smaller time step, dt, to properly simulate.

Potential

The, "Potential," section is used to draw custom potentials, i.e. the, "Drawn Potential," in the popup menu. Vmin and Vmax set the minimum and maximum strength of the potential (blue y-ticks on the axis); a potential difference greater than 15 usually reflects most of the wave function.

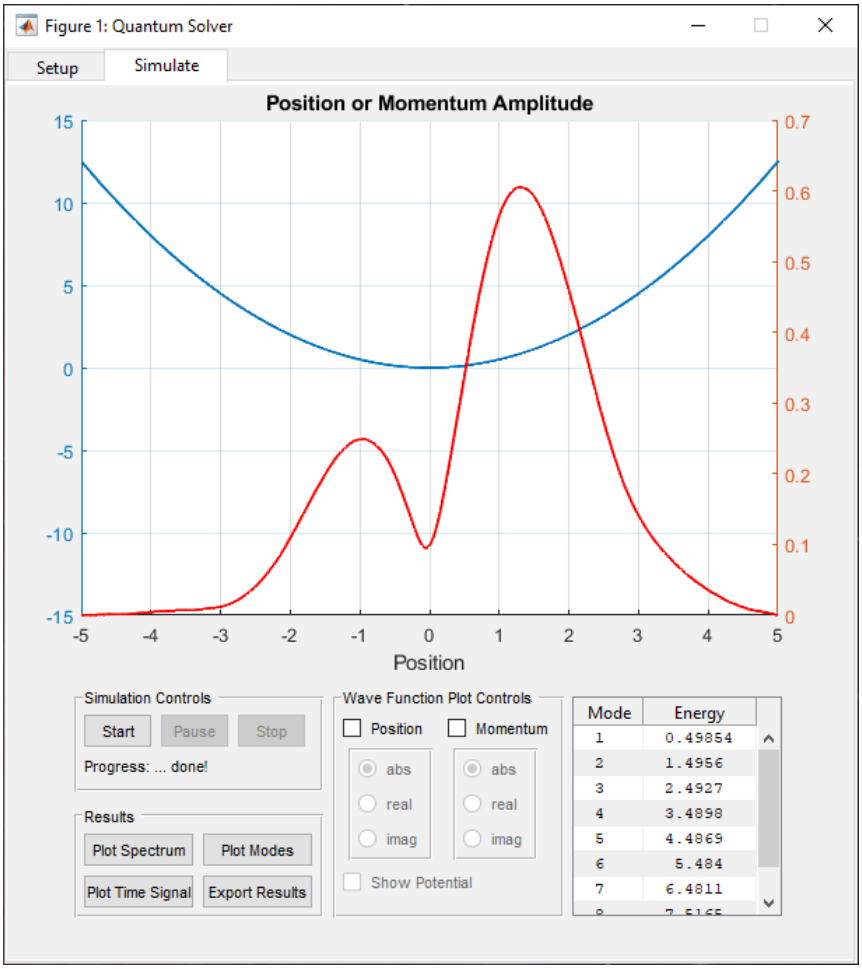
If you toggle the, "Draw," button, you can then click directly on the axis to determine turning points in the potential; but only to carve out potential wells or horizontal gradients. For more complex potentials, you'll have to program them yourself... sorry! The flip button flips the drawn potential vertically, and the clear button resets it.

Draw Samples

Toggling the, "Draw Samples," button allows you to click on the axis to determine at which points along the x-axis the wave function will be sampled as time marches ahead. The default set of samples is a set of points spread out along the axis that I've found captures a decent amount of the stationary states for a range of initial conditions. If you want to better understand what's happening only in higher order modes, for example, you can select sample points farther away from the center of your chosen potential. Or within any area of interest you may have.

The Simulation Tab

The simulation tab doesn't really require a lot of work, as everything was set up in the setup tab (go figure). The only controls here are those for running the simulation; watching the wave function evolve in time—either in position or momentum space—and then viewing or exporting the results. Similar to the setup tab, the blue line is the potential of interest, and the red line is the wave function.



Simulation Controls

These buttons just... start, stop, and pause the simulation! That's it, no magic here. While the simulation is running you can't edit any of the input parameters, but if you have the appropriate boxes checked, the position or momentum wave function will be displayed in the above axis.

Wave Function Plot Controls

This is just a series of checkboxes that allows you to view the evolution the wave function as the simulation progresses. The position checkbox will show you the position wave function, and the momentum checkbox will show you the momentum wave function. You can watch either the absolute value, real part or imaginary part of either wave function. It's fun to watch as these things change, and it's great to get an intuitive understanding of what's going on, especially for simpler potentials.

Mode & Energy Table

Once your simulation is finished running, this table will automatically be populated with the energy levels of the detected modes. Mostly this is here to fill in some blank space.

Results

The results section just gives you a bunch of buttons for plotting the results of the simulation. At the end of a simulation, a QSResults object is created, and can be exported to the base workspace. The results themselves are calculated from:

1) The Time Signal is created by sampling the position wave function at a few points in space, at every moment in time,
2) The Spectrum is calculated by taking the Fourier Transform of the Time Signal,
3) The Modes and their energies are calculated from the local maximums—peaks—of the spectrum.

Functionality for all of this is stored and calculated in the QSResults object, so once you export it you can recreate these graphs or move things around as needed.

Customizing Potentials

Since the whole point of this app is to essentially analyze how wave functions respond in different potentials, there are three ways to customize them beyond the defaults in the popup menu:

1. Drawing it directly onto the axis, and selecting, “Drawn Potential,” in the, “Choose Potential,” popup menu,
2. Typing a function handle into the, “Simulation Parameters,” table directly,
3. Creating your own function in a separate m-file and placing it in the +Potentials folder (package); the app will automatically detect it on startup and add an option in the, “Choose Potential,” popup menu.

Function Handles

When typing a function handle into the, “Simulation Parameters,” table, it has to have the form:

```
@(x) <function of x>
```

Meaning that it has to take in a vector of x values and then return a vector representing the potential. For example, the harmonic potential would take the form:

```
@(x) x.^2/2
```

All you have to do is type something like that right into the, “Simulation Parameters,” table for the, “Scalar Potential”.

Separate M-File Function

Similar to typing up a short function handle, the m-file function needs to take in a vector of x values and then return a vector representing the potential. This means that the function needs to have a signature similar to:

```
function V = custom_potential(x)
    V = <function of x>;
```

For example, the standard electric potential would look like:

```
function V = electric_potential(x)
    V = 1./abs(x);
```

Once you create your custom file, you just have to put it in the folder (package) +Potentials, in the same folder as the main GUI file.

Customizing Magnetic Field

The magnetic field can be customized exactly as the scalar potential above, except there is no functionality to draw magnetic fields. Theoretically, we are only using the magnetic field pointing in the x-direction, coupling spin states using the first Pauli matrix.