Solving the Time-Dependent Schrödinger Equation: Instructor Notes[a]

David G. Robertson[†]

Department of Physics, Otterbein University, Westerville, OH 43081

(Dated: October 10, 2011)

Abstract

Some notes on the simulations are presented, along with solutions to the exercises. Please contact the author if you have any comments or suggestions.

[a] Work supported by the National Science Foundation under grant CCLI DUE 0618252.

[†] Email: drobertson@otterbein.edu

CONTENTS

I. GOALS AND TIME NEEDED

The goals of this module are:

- to provide an opportunity to develop insight into simple quantum systems; and
- to familiarize students with basic concepts involved in the numerical solution of partial differential equations.

In my experience, working through these kinds of exercises is a great way to develop experience and intuition about how quantum mechanics works. The ability to generate solutions to the Schrödinger equation for arbitrary potentials and with realistic initial states (e.g., wavepackets) really helps one develop a feel for how wavefunctions behave. Furthermore, the process of translating a physics problem into a form suitable for implementation on a computer forces the student to understand the problem in a very concrete way. In this case, the full structure of quantum mechanics is engaged. For example, in the exercise on time-dependent perturbations, the student will need to understand how to compute the transition probability from her solution to the Schrödinger equation.

A number of interesting computational issues also arise, including difference approximations to derivatives, stability and accuracy criteria for differencing schemes, and numerical quadrature.

The minimal physics background required includes a basic exposure to wavefunctions and the Schrödinger equation, at the level of a typical sophomore-level course on modern physics. However, more advanced students will be able to do more with the exercises, and the experience can be correspondingly richer for them.

My estimate is that one to two weeks of class time, assuming 3 hours per week in class and some work outside of it, should be sufficient to explore the basic numerical issues and

develop codes to explore most or all of the projects. Once the basic code is in place and has been checked (e.g., by solution of the free particle problem), other potentials and/or initial wavefunctions can be implemented easily.

II. NOTES ON THE SIMULATION PROJECTS

In general I have tried to err on the side of giving the student too little direction rather than too much; this should allow the instructor flexibility to decide what to reveal and what to let the students figure out on their own. Once example is the need to carry out integrals, e.g., to check the wavefunction normalization or compute probabilities. In the last project, for example, the students will need to determine the wave function at some time t and then compute the probability that the system is described by a stationary state $|\psi_n\rangle$ as $P = |\langle\psi_n|\Psi\rangle|^2$, where

$$\langle\psi_n|\Psi\rangle = \int_{-\infty}^{\infty} \psi_n(x)\Psi(x,t)dx \tag{2.1}$$

(recall that the stationary state wavefunctions are real, so that the complex conjugation on the first factor in the integral can be omitted). In general there is no need for sophisticated integration techniques in these exercises; the basic trapezoid method will work just fine. In this case one approximates

$$\int_a^b f(x)dx \approx \frac{\Delta x}{2}\sum_i [f(x_i) + f(x_{i+1})] \tag{2.2}$$

where the sum is over the grid points. Note that each grid point appears twice except for the end points. Thus one can also write

$$\int_a^b f(x)dx \approx \frac{\Delta x}{2}[f(a) + f(b)] + \Delta x{\sum_i}' f(x_i) \tag{2.3}$$

where the prime on the sum indicates that the end points are omitted.

Of course, the computer simulation cannot cover an infinite domain in x, so the model will be restricted to some finite range. The end points on the mesh should be required to have $\Psi = 0$ for all times; this is equivalent to placing infinitely high potential "walls" at these points. So long as the wavefunction stays away from these points, the truncated domain should not affect the results significantly. Alternatively, these may be considered as actual infinite potential walls. My preference is to center the domain around $x = 0$, as this makes the harmonic oscillator more straightforward.

Good visualizations, including animations, will really help make the simulations "speak." Animation may be an issue if you are not working in a Unix/Linux environment. Within such an environment you can always use `gunuplot`, examples of which are shown in the sample programs. In other environments you may need to generate individual frames using a plotting program, and then stack the frames together to produce an animated gif or similar. Please feel free to contact the author if you need advice in this regard.

1. **The Free Particle**

   The first project should be a simulation of a free particle ($V = 0$) with a gaussian initial wavefunction. This problem can be solved exactly, so the correctness (or otherwise) of the code can be checked easily. This exercise will form the basis for all the others.

   In the sample code the domain runs from $x = -25$ fm to $x = +25$ fm. Typical numbers of mesh points used at this stage might be a few hundred to a few thousand, though the student should experiment with different values, comparing to the exact solution. The same is true of the time step, for which typical values might be 0.1 to 0.001 fs. In all cases, the discrete form of the Schrödinger equation will be a good approximation if the change from one grid point to the next, or from one time step to the next, as a fraction of the value of the wavefunction, is small.

   The expectation value of the energy is calculated as

$$\langle H \rangle = \int_{-\infty}^{\infty} \Psi^*(x,t) \hat{H} \psi(x,t) dx \tag{2.4}$$

   where

$$\hat{H} = -\frac{\hbar^2}{2m} \frac{\partial^2}{\partial x^2} + V(x,t) \tag{2.5}$$

   and

$$\frac{\partial^2 \Psi}{\partial x^2} \rightarrow \frac{\Psi_{i+1} - 2\Psi_i + \Psi_{i-1}}{\Delta x^2} \tag{2.6}$$

   in the difference approximation. This quantity should be nicely constant in time, and its value may be compared to the exact result, available in any standard quantum mechanics text. Similarly, the normalization integral

$$\int_{-\infty}^{\infty} \Psi^*(x,t) \psi(x,t) dx = 1 \tag{2.7}$$

   will be constant using the Crank-Nicholson scheme.

Both of these tests will fail for either of the other (non-unitary) approaches to evolution discussed in the Student Manual.

2. **Scattering**

This is the main application that should be pursued once the basic code is in place. It is straightforward to implement steps, barriers, or wells and send gaussian wavepackets against them. Once interesting challenge will be to calculate from the solution the probabilities of reflection and transmission. The student should wait until the reflected and transmitted waves are (mostly) clear of the potential feature (by watching an animation of the process), and then calculate the probability to find the particle on one side or the other, by numerically integrating $|\Psi|^2$.

Note that when the waves reach the endpoints, where the potential is effectively infinite, a lot of ripples will be induced but the total probability of finding the particle on one side will not change; no probability "leaks out" through the endpoints. The rapid oscillation in $|\Psi|^2$ may make the numerical integration less accurate, however.

If time permits, students may experiment with the effect of "softening" the potential barriers or wells, i.e., giving them finite slopes, or even making them completely smooth rising/falling exponentials.

3. **Energy Eigenstates**

This project involves coding at least a few harmonic oscillator stationary states so that (normalized) linear combinations may be selected as the initial state. It may be helpful to choose a nice value for ω_0, for example the value that makes

$$\frac{1}{2}\hbar\omega_0 = 1 \text{ eV} \tag{2.8}$$

Then the energy eigenvalues are simply $E_n = 1, 3, 5, \ldots$ eV.

In addition to showing that stationary states are stationary and $\langle E \rangle$ is constant, there are a variety of calculations that can be pursued if there is sufficient time and interest. The expectation values of p and x can be computed; these oscillate with a particular frequency that can be computed (for a linear combination of two stationary states, say) and compared to the simulation. Another possibility would be to check Ehrenfest's

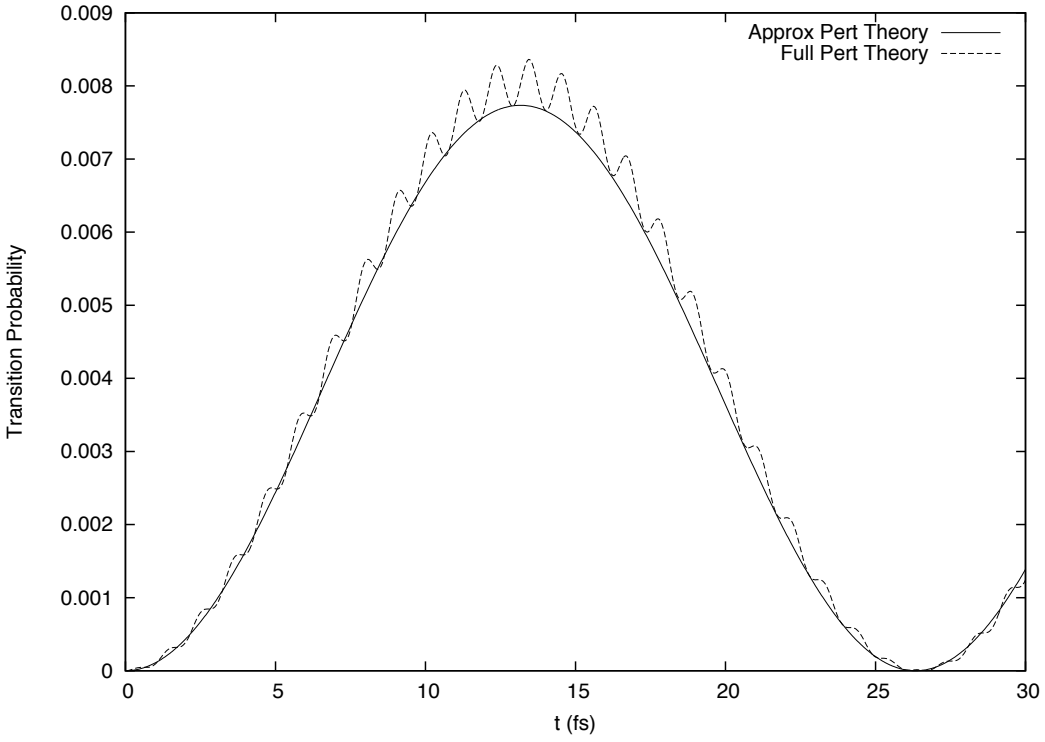

FIG. 1. Comparison of ground to first excited state transition probability for for a sinusoidal driving force applied to a simple harmonic oscillator with $\omega_0 = 3.04$ fs^{-1}, $\omega = 2.8$ fs^{-1}, and $F = 0.1$. The full first-order time-dependent perturbation theory (eq. (2.11)) is shown as a dashed line and the "on resonance" approximation (eq. (2.10)) is the solid line. The full calculation obtained by evolving the Schrödinger equation reproduces the dashed line very well for these parameter values.

theorem for these states,

$$\frac{d\langle p\rangle}{dt} = \langle -\frac{dV}{dx}\rangle \tag{2.9}$$

4. **Periodic Perturbations**

This project will be most meaningful if students have seen time-dependent perturbation theory, so that the origin of the lowest-order formula

$$P(t) = \frac{F^2}{2\hbar m\omega_0}\left(\frac{\sin[(\omega-\omega_0)t/2]}{\omega-\omega_0}\right)^2 \tag{2.10}$$

is familiar. Even without this, however, it can be an instructive exercise since the students must understand how to compute the transition probability.

As discussed in the Student Manual, eq. (2.10) is itself an approximation to the first-order perturbation result, which is valid when $|\omega_0 - \omega| \ll \omega_0 + \omega$. The more complete

formula is

$$P(t) = \frac{F^2}{8\hbar m\omega_0}\left|\frac{e^{i(\omega_0+\omega)t}-1}{\omega_0+\omega}+\frac{e^{i(\omega_0-\omega)t}-1}{\omega_0-\omega}\right|^2 \tag{2.11}$$

Eq. (2.10) is obtained when the first term is neglected compared to the second.

This full perturbation result shows additional structure that will be present in the numerical solution, but that is not reflected in the approximation (2.10). A typical comparison between the two perturbation results is shown in Fig. 1. When the perturbation is small, the "exact" result (i.e., based on numerical solution to the full Schrödinger equation) reproduces the full perturbation result very well.

III. SAMPLE PROGRAMS

Three sample programs, written in C, are distributed with the module. They each take two command-line arguments, the number of spatial grid points and the time step. Other parameters hard-wired via `#define` statements. See comments in the codes for additional details.

These should be compiled as

```
gcc tdse.c -lm
```

and then run as, for example,

```
./a.out 2000 0.001
```

or, for `tdse-gnuplot.c`,

```
./a.out 2000 0.001 | gnuplot
```

1. `tdse.c`

   This is a generic program that implements the evolution scheme for a variety of potentials, including the harmonic oscillator. Gaussian and step-function initial states are available, along with any linear combination of the first five harmonic oscillator stationary states. Routines are provided for computing normalization and overlap integrals, as well as the probability to find the particle in any specified interval.

Command-line arguments are the number of spatial grid points and the time step in fs. The total time to integrate is hard-wired in the code, as well as the domain size (default is 50 nm, centered on the origin).

This code writes separate ASCII output files containing the real and imaginary parts of Ψ and it absolute value. These can then be examined in any plotting program.

2. `tdse-gnuplot.c`

   The same as `tdse.c` but produces output suitable for redirection to `gnuplot`, to produce animations.

3. `tdse-pert.c`

   This code implements the harmonic oscillator with a sinusoidal driving term, as discussed in Project 4. By default it computes the $n = 0$ to $n = 1$ transition probability as a function of time and writes the result to a file. For comparison, the predictions of lowest-order time-dependent perturbation theory are also given (both eqs. (2.10) and (2.11)).