Numerical Solution of the Schrödinger Equation

Håkon Taskén[a], Paul Thrane[a]

[a]*Institute for Physics, Norwegian University of Science and Technology, N-7491 Trondheim, Norway.*

Abstract

We have written a program that integrates the Schrödinger equation numerically, thereby finding how a system evolves given a time independent potential and an initial state. The solution works both for one and two dimensional systems.

1. The program

The program is composed of two major parts, schrodingerFD and plotSchrodinger.py. schrodingerFD is implemented using C++, and is used to make the initial state, determine the time evolution of the system by integrating the Schrödinger equation using finite differences, as well as saving these results in certain files. plotSchrodinger.py is implemented in python, and is used to make an animated movie of the progress of the system[2]. In addition to these programs, there are several .txt files that are needed to run a simulation; also, several shell scripts have been made to automate the process. For information on how to run a simulation, see README.md[1]. All of these files should be included with this document, but can also be found at `https://github.com/task123/schrodingerFD`.

2. Results

The program works well for the different conditions tested, and can easily be expanded to include new potentials and initial conditions. To view the results of a few simulations, a couple of output .mp4 files have been placed in the subdirectory simulationMovies. For the specific situations described in problem 4 and 5 in [1], the probability of reflection and transmission was calculated for varying potential height and width, respectively. The results are shown in figure 1 and 2.

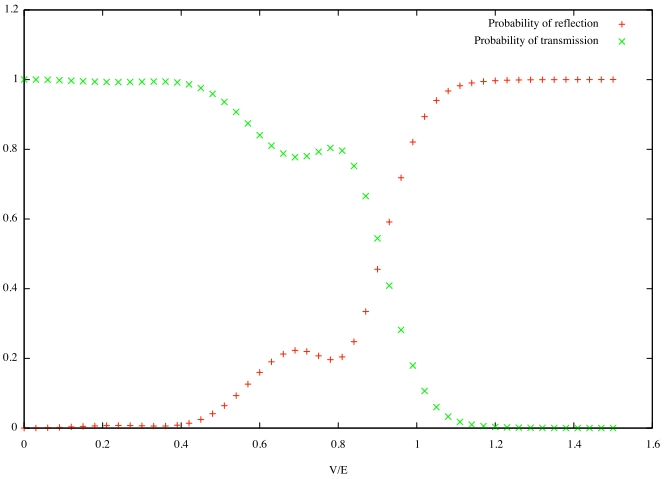

Figure 1: Probability of transmission and reflection for a wave packet traveling towards a square potential in one dimension, as described in problem 4 in [1].

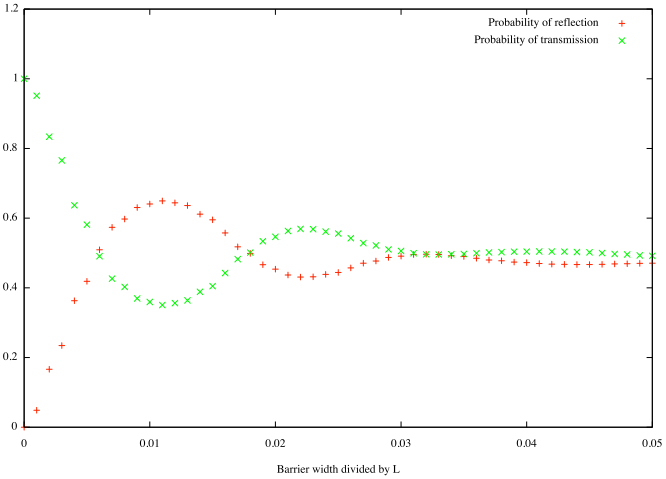

Figure 2: Probability of transmission and reflection for a wave packet traveling towards a square potential in one dimension, as described in problem 5 in [1]. Here, L is the lengt of the system.

References

[1] Institute for Physics, NTNU. Numerical exercise FY2045 Quantum Mechanics, Fall 2015. Available from `http://amokk.phys.ntnu.no/files/FY2045_2015/exercise15/numerics.pdf`

[2] Jake Vanderplas. Matplotlib Animation Tutorial, August 18, 2012. Available from `https://jakevdp.github.io/blog/2012/08/18/matplotlib-animation-tutorial/`

[1] Note that to run plotSchrodinger.py as included, ffmpeg must be installed. An alternative is to modify plotSchrodinger.py as described in the code.