Solving Time-Dependent Schrödinger Equation for Two Particles

Ender Sahin
endersahin@uchicago.edu

Introduction

The general form of the time-dependent Schrödinger equation (TDSE) is given by

$$i\hbar\frac{\partial}{\partial t}\psi(\mathbf{r},t) = H\psi(\mathbf{r},t)$$

This equation has analytical solutions for only a number of one-particle systems and for none of the two-particle ones. Therefore, it is necessary to employ a numerical solution, which enables us to fully understand the dynamics of two-particle interactions that are usually left out of the Quantum Mechanics textbooks.

For the two-particle non-relativistic one-dimensional system with potential constant in time, the Hamiltonian is given by

$$H = -\frac{\hbar^2}{2m_1}\frac{\partial^2}{\partial x_1^2} - \frac{\hbar^2}{2m_2}\frac{\partial^2}{\partial x_2^2} + V(x_1,x_2)$$

The objective of this project is to solve TDSE with the given Hamiltonian for a variety of configurations (e.g. mass, potential, etc.), visualize the results in a way that gives intuition about the dynamics, and provide a framework that can be easily expanded for more complex problems.

Computational Methods

I have implemented the algorithm laid out in [1] which expands Visscher's staggered-time algorithm [2] for the two-particle setup. This is an explicit method, and is therefore relatively simple and fast comparing to the implicit Crank–Nicolson method. Furthermore, it is stable, second-order accurate in time, and conserves probability.

The main idea behind the algorithm is given below:

1. Discretize independent variables x_1, x_2 and t

$$\psi(x_1,x_2,t) = \psi(x_1 = l\Delta x, x_2 = m\Delta x, t = n\Delta t) \equiv \psi^n_{l,m}$$

2. Calculate spatial derivatives by Taylor expanding ψ with respect to x_1, x_2

$$\frac{\partial^2\psi}{\partial x_1^2} \approx \frac{\psi(x_1+\Delta x,x_2,t) - 2\psi(x_1,x2,t) + \psi(x_1-\Delta x,x_2,t)}{\Delta x^2} + \mathcal{O}(\Delta x^2)$$

3. Calculate temporal derivatives by central difference algorithm [3]

$$\psi(x_1,x_2,(n+1)\Delta t) - \psi(x_1,x_2,(n-1)\Delta t) \approx -2i\Delta t H\psi(x_1,x_2,n\Delta t)$$

4. Separate ψ into real and imaginary parts

$$\psi^n_{l,m} = R^n_{l,m} + i\, I^n_{l,m}$$

5. Solve the pair of coupled equations at staggered times

$$\mathrm{Re}\,\psi(x_1,x_2,t) = R^n_{l,m}, \quad \mathrm{Im}\,\psi(x_1,x_2,t+\Delta t/2) = I^n_{l,m} \Rightarrow$$

$$\rho(x_1,x_2,t) = |R^n_{l,m}|^2 + I^n_{l,m}I^{n-1}_{l,m}, \quad \rho_1(x_1,t) \approx \int_{-\infty}^{+\infty}\rho(x_1,x_2,t)\,dx_2$$

Even though the simulation code of the authors in C is available online [4], I have implemented a version of the algorithm from scratch in Python [5], which has a number of improvements in terms of performance and readability upon the original implementation.

Results

I have investigated two interaction potentials, a square well given by

$$V_{square}(x_1,x_2) = V_0\,\theta(\alpha - |x_1-x_2|)$$

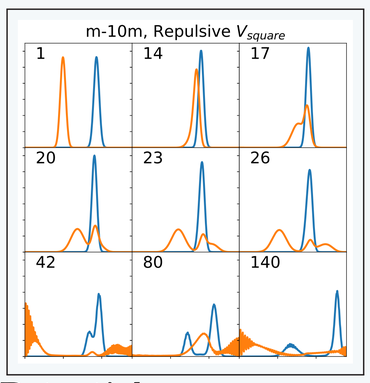
and a Gaussian given by

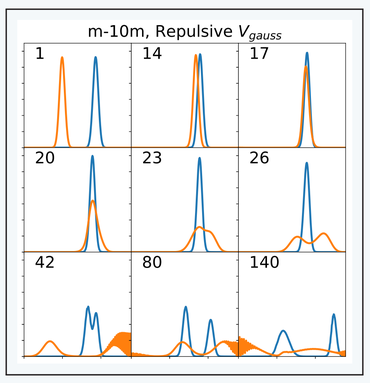
$$V_{gauss}(x_1,x_2) = V_0\,e^{(-|x_1-x_2|^2/2\alpha^2)}$$

where V_0 denote the depth and α the range. By switching the sign of V_0, I have also simulated repulsive and attractive versions. Finally, I have looked into barrier-like (with $m_1 = 10m_2$) and same mass (with $m_1 = m_2$) collisions.

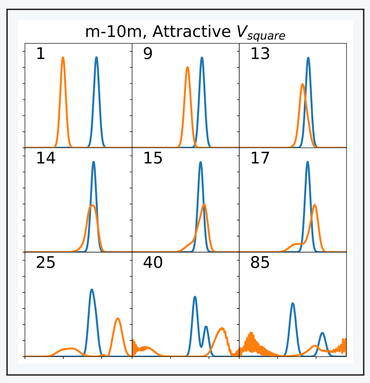
Barrier-like Collisions

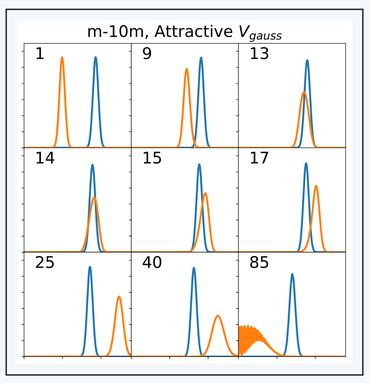
Repulsive Potential





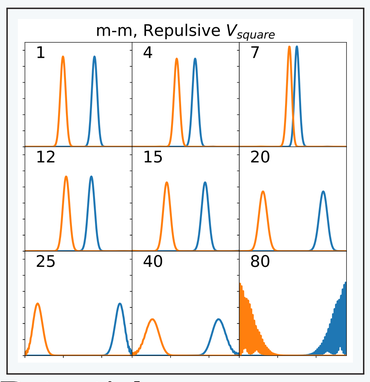
Attractive Potential

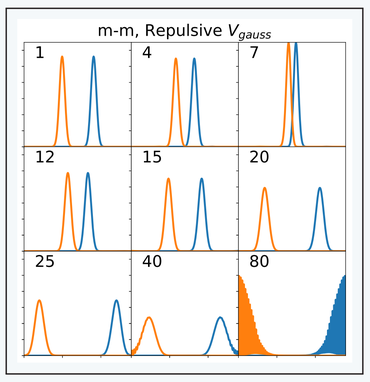




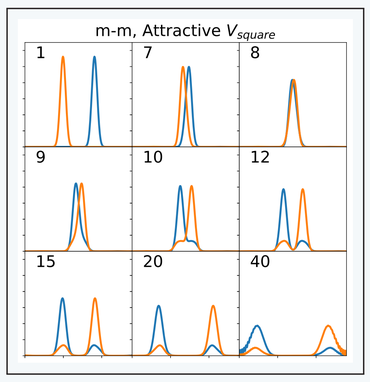
$m - m$ Collisions

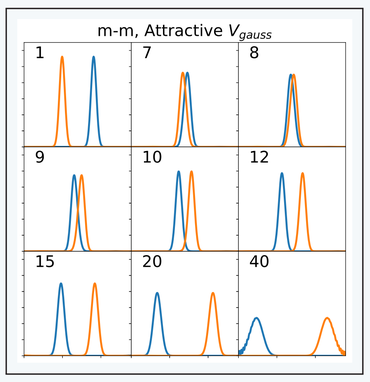
Repulsive Potential





Attractive Potential





References

[1] Maestri, J. J. V., Landau, R. H., and Páez, M. J. (2000). Two-Particle Schrödinger Equation Animations of Wavepacket–Wavepacket Scattering (revised). *American Journal of Physics, 68*(12), 1113–1119.

[2] Visscher, P. B. (1991). A fast explicit algorithm for the time-dependent Schrödinger equation. *Computers in Physics, 5*(6), 596-598.

[3] Askar, A., and Cakmak, A. S. (1978). Explicit Integration Method for the Time-Dependent Schrödinger Equation for Collision Problems. *J. Chem. Phys., 68*, 2794–2798.

[4] Maestri, J. J. V., and Landau, R. H. (1999). Authors' implementation. physics.oregonstate.edu/ landaur/codes/packets|inux.c

[5] Sahin, E. My implementation. https : //github.com/endersahin/phys250 − final − project

Discussion and Future Research

We see that the staggered-time algorithm is easy to implement and relatively fast comparing to other algorithms. Furthermore, it is able to simulate for arbitrary interaction potentials, and even constant background potentials.

Therefore, one take [5] as a starting point and easily expand to 2 or 3 dimensions, 3 or more particles, include spin interactions, support time-varying potentials, etc. Another extension might be letting the user *draw* an arbitrary interaction or background potential and simulating that for different time steps. Finally, all of this can be packed into an applet that accepts inputs from drop-down menus and produces animations of the particles at desired configurations.