FYS4150 Term Project 5: Schrödinger's double slit

Stian Aannerud[*]

(Dated: December 15, 2021)

In this project we have made a system for numerically simulating a wave using the Schrödinger equation. We used this to run a series of tests using potentials emulating single, double, and triple slits. From this we successfully managed to recreate interference patterns in the double and triple slit cases. And the simulation was numerically stable and conserved probability with a deviation of only 10^{-14} from the expected 1.

I. INTRODUCTION

One of the most famous experiments in all of physics is the double slit experiment which detected an inherent wave nature in particles as they seemingly caused interference with themselves. Being able to simulate such a system by treating the particle as a wave packet is an interesting computational goal. With Schrödinger's famous thought experiment of a cat in a box, we stuck a double slit in a box and called it Schrödinger's double slit.

In this project we will show how we simulated a wave packet using the Schrödinger equation, and used it to replicate interference patterns. We will show the derivation of a discretized, dimensionless equation on matrix form, and discuss how to solve such a system. Then we describe the process of initializing a wave packet, and running it through a set of walls. We use conservation of probability as a check, and plot everything from snapshots in time to a simulated detector screen.

II. METHODS

The basis of our methods is progressing a system in time using the Schrödinger equation, contained in a box with an environment determined by a static potential. We work in 2+1 dimensions, corresponding to two spatial dimensions x, y and time. We use dimensionless units, and fix the boundaries to $x, y \in [0, 1]$. The system is treated as a grid of points size $M \times M$, but in practice we use only use the inner $(M-2) \times (M-2)$ points as the boundaries are fixed to be a large potential v_0. The boundaries are included as part of the transition to the matrix form of our equation.

A small note on notation used is added in the following section II A. Then we describe how we go from the standard Schrödinger equation to our discretized dimensionless matrix equation in section II B. In section II C we discuss how we chose to solve the equation for each time step, and in section II D we show how we chose the initial system state. Finally in section II E we describe the four experiments we chose to investigate, along with the setups for each.

A. Notation

When we transition from a continuous to a discrete system we use a set of indices i, j, n to indicate position along x, y axes and time t respectively:

$$u(x, y, n) \rightarrow u^n_{i,j} \tag{1}$$

Note the index i is not to be confused with the imaginary value $i = \sqrt{-1}$. The index will always appear as a subscript (unless otherwise mentioned) and the imaginary value as a factor. The index for time n should not be confused with an exponent, and the superscript will always be a time index for discretized vectors and a normal exponent otherwise.

Since we have a set interval $x, y \in [0, 1]$, the position corresponding to the spatial coordinates i, j can be found directly using the system size:

$$x = i/M \tag{2}$$
$$y = j/M \tag{3}$$

Note the indexing always includes boundaries $i, j = 0, 1, ..., M-1$, so the internal system we use for the wave function is indexed $i, j = 1, 2, ..., M-2$.

B. The Schrödinger equation

We evolve our wave packet in time with the time-dependent Schrödinger equation, given in its most general form:

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t} |\Psi\rangle = \hat{H} |\Psi\rangle \tag{4}$$

Where $|\Psi\rangle$ is the quantum state and $\hat{H}$ is the Hamilton operator. Our particle is treated as non-relativistic and we work in two spatial dimensions (plus time). This lets us expand the operator to a kinetic and a potential term:

[*] Code Repository: https://github.com/Stia4/FYS4150/tree/main/Project%205

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t}\Psi = -\frac{\hbar^2}{2m}\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\Psi + V(x,y)\Psi \tag{5}$$

With a complex valued wave function $\Psi = \Psi(x,y,t)$ and time-independent potential acting as the environment of box/slit. The Schrödinger equation in this form is called the wave equation and is how we progress the system in time.

Since the wave function Ψ is complex valued, we extract the actual probability in space with the Born rule:

$$p(x,y;t) = |\Psi(x,y,t)|^2 = \Psi^*(x,y,t)\Psi(x,y,t) \tag{6}$$

Where $p(x,y;t)$ is the probability of finding the particle at position (x,y) for time t. Note that since we will work in a discrete system this probability is actually pure probability, as opposed to a probability density used for continuous systems.

Finally we simplify by removing dimensions, scaling away all constants. Our dimensionless wave equation is then expressed as:

$$i\frac{\partial u}{\partial t} = -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} + v(x,y)u \tag{7}$$

Where we have changed our notation to clarify we are using dimensionless variables. The wave function $\Psi(x,y,t)$ is renamed $u(x,y,t)$, and the potential $V(x,y)$ is renamed $v(x,y)$.

To actually numerically simulate this however we will have to discretize the system. For this we will use the Crank-Nicolson approach, detailed in the following subsection. Then we simplify further by writing the discretized system as a matrix equation to be solved.

1. The Crank-Nicolson approach

Our chosen method of discretization is the Crank-Nicolson approach, which is a second-order implicit method in time. A benefit of being implicit is the stability for any chosen time step, where explicit methods would need to fulfill criteria which would reduce efficiency.

As the derivation of our discrete Schrödinger equation is quite long, it has been moved to Appendix A. The final result is only intended as a middle step, as we in the next subsection wish to express it in a compact matrix form.

2. Matrix form

While we have previously treated the wave equation as a matrix of values corresponding to spatial coordinates, we will now instead write it as a vector. A system of size $(M-2)\times(M-2)$ is now a single vector length $(M-2)^2$, where each $(M-2)$ segment corresponds to a single vertical slice:

$$\vec{u}^n = [(u^n_{1,1}, ..., u^n_{M-2,1}), ..., (u^n_{1,M-2}, ..., u^n_{M-2,M-2})] \tag{8}$$

Note the parentheses are only included to visualize the segments which were previously separate lines, and $\vec{u}^n$ should be treated as a single vector. By re-framing the system in this way we can express the step in time $\vec{u}^n \rightarrow \vec{u}^{n+1}$ as a matrix equation:

$$A\vec{u}^{n+1} = B\vec{u}^n \tag{9}$$

Where the matrices A and B are size $(M-2)^2 \times (M-2)^2$. Their construction is easier illustrated with an example for a size $M = 5$ system:

$$A = \begin{bmatrix} a_0 & -r & 0 & -r & 0 & 0 & 0 & 0 & 0 \\ -r & a_1 & -r & 0 & -r & 0 & 0 & 0 & 0 \\ 0 & -r & a_2 & 0 & 0 & -r & 0 & 0 & 0 \\ -r & 0 & 0 & a_3 & -r & 0 & -r & 0 & 0 \\ 0 & -r & 0 & -r & a_4 & -r & 0 & -r & 0 \\ 0 & 0 & -r & 0 & -r & a_5 & 0 & 0 & -r \\ 0 & 0 & 0 & -r & 0 & 0 & a_6 & -r & 0 \\ 0 & 0 & 0 & 0 & -r & 0 & -r & a_7 & -r \\ 0 & 0 & 0 & 0 & 0 & -r & 0 & -r & a_8 \end{bmatrix}$$

$$B = \begin{bmatrix} b_0 & r & 0 & r & 0 & 0 & 0 & 0 & 0 \\ r & b_1 & r & 0 & r & 0 & 0 & 0 & 0 \\ 0 & r & b_2 & 0 & 0 & r & 0 & 0 & 0 \\ r & 0 & 0 & b_3 & r & 0 & r & 0 & 0 \\ 0 & r & 0 & r & b_4 & r & 0 & r & 0 \\ 0 & 0 & r & 0 & r & b_5 & 0 & 0 & r \\ 0 & 0 & 0 & r & 0 & 0 & b_6 & r & 0 \\ 0 & 0 & 0 & 0 & r & 0 & r & b_7 & r \\ 0 & 0 & 0 & 0 & 0 & r & 0 & r & b_8 \end{bmatrix}$$

We can see that they consist of $M-2 = 3$ sub-matrices along the main diagonal with sizes $(M-2)\times(M-2) = 3\times 3$, and elements $\pm r$ along the $\pm(M-2)$-th diagonals. The value of r is the same as used in the Crank-Nicolson discretization, $r \equiv i\Delta t/2h^2$, and the diagonal vectors $\vec{a}, \vec{b}$ are defined as follows:

$$a_k = 1 + 4r + \frac{i\Delta t}{2}v_{ij} \tag{10}$$

$$b_k = 1 - 4r - \frac{i\Delta t}{2}v_{ij} \tag{11}$$

With k denoting the transformed index from a matrix system to a vector system $k = i + (M-2)j$ (i, j here are indexes). A simplification we can use when creating these matrices is their similar setup and structure, and by close inspection see that $b_k = 2 - a_k$. Therefore we can simply invert the sign of A and add 2 element-wise to the main diagonal to create B.

A challenge with this setup is the size required to store and operate on the matrices. The experiments we wish to run (detailed in section II E) involve a system size $M = 200$, meaning the matrices A, B would be of size $198^2 \times 198^2 \approx 1.54 \cdot 10^9$ elements. Representing each element as double precision floating point numbers, requiring 64 bits each, this gives an expected storage size of ≈ 12.3 gigabytes per matrix. We can circumvent this by using the sparse nature of our matrices, as they have a significant amount of zeros. By only storing only non-zero values (and their corresponding indices) we lower the required space by a large amount, for only a minor sacrifice in computational efficiency.

To actually solve the system we perform a matrix multiplication $B\vec{u}^n = \vec{b}$ to get the system in traditional form $A\vec{u}^{n+1} = \vec{b}$. Then progressing the system simply constitutes solving for the unknown state $\vec{u}^{n+1}$, which we detail and discuss in section II C. Note the right hand side $\vec{b}$ is not the same as the diagonal vector used in construction of the matrix B.

C. Solving

There are many different methods to solve such a system, but we can make them more computationally efficient by taking into account our unique setup. The matrices A and B are sparse, meaning they have a small percentage of non-zero elements, and diagonal, with all elements along five diagonals. Efficient methods of solving our system would then likely be those which benefit from this setup.

The first method we consider is the Jacobi rotation method, also called the Jacobi eigenvalue algorithm. It is an iterative algorithm which eliminates off-diagonal (referring to the main diagonal) elements, called pivots, in succession until all off-diagonals are under a given threshold. This would be an efficient method in our case since we have a small relative amount of non-zero elements, meaning fewer required cycles. One thing to keep in mind however is that the process of elimination also affects other elements, meaning the amount of required cycles would be larger than simply the amount of off-diagonal values. This could also potentially cause the matrix to become dense, which would remove all benefits of the sparse matrices. Due to this uncertainty we decided not to use this method.

Another candidate is lower-upper (LU) decomposition. The goal of this method is factoring the initial matrix (A in our case) into two new matrices L, U, by using the relation $A = LU$. These new matrices are lower and upper tridiagonal respectively, and the relation being multiplicative means the factorization is directly dependent on the number of non-zero elements. This fits our sparse system very well. It is also a direct method of solution, which circumvents the problems encountered for the Jacobi algorithm.

For the reasons listed above, we decided to use an implementation based on LU decomposition. Our system for handling matrices the C++ package *Armadillo* [1] (see code repository for implementation), which includes a wrapper of the *SuperLU* (Supernodal LU) [2] as method for solving Sparse matrix systems. The reasons for this choice were both the aforementioned discussions around the LU decomposition method, as well as the computational efficiency and confident results produced in testing.

D. Initial wave packet

Our initial wave state is a Gaussian wave packet expressed as:

$$u(x, y, t = 0) = e^{-\frac{(x-x_c)^2}{2\sigma_x^2} - \frac{(y-y_c)^2}{2\sigma_y^2} + ip_x(x-x_c) + ip_y(y-y_c)} \tag{12}$$

Where (x_c, y_c) is the centre of the packet, σ_x and σ_y are the standard deviations for x and y axes, and (p_x, p_y) is the momentum of the packet. We also wish to normalize the wave such that the probability of finding the particle in total is equal to 1. Since we work with discrete probabilities, we can just apply the Born rule:

$$p(t = 0) = \sum_{i,j} {u_{i,j}^0}^* u_{i,j}^0 \tag{13}$$

And require that this probability is equal to 1. We do this by dividing the initial wave packet by the square root of initial probability $u_{i_j}^0 / \sqrt{p(t = 0)}$.

E. Experiments

There are four different simulations we wish to perform, corresponding to four different setups of the potential and initial wave packet. Initially we simulate a simple circular packet in an empty box as a test to see the system behaves as expected in simple conditions. Then we introduce a wall with an increasing amount of slits. The single slit case is also just a test to see if the system behaves as expected, while the double and triple slit cases are the ones of interest since we expect to see interference.

One way we check that the simulation is behaving as it should is to see if the probability is conserved. The likelihood of finding the particle in total in the box should be 100%. This quantity should be conserved in time, meaning we can use it to test with. We will plot this as function of time and compare for all cases.

We also wish to plot the likelihood of detecting a particle along a detector wall. For this we assume the position

Parameter	Value
h	0.005
M	200
Δt	$2.5 \cdot 10^{-5}$
(x_c, y_c)	$(0.25, 0.5)$
(p_x, p_y)	$(200, 0)$
σ_x	0.05

TABLE I. Parameters for system size and wave packet which are equal for all simulations. All values are dimensionless.

along x and time to be set, while allowing position y as a variable. This is intended to replicate a real detector to see if we get interference patterns in our simulation. Since we assume x, t are known, the probability is normalized along y.

III. RESULTS AND DISCUSSION

All simulations have a set of parameters in common. Parameters relating to the system and the initial wave packet are listed in TABlE I. In addition to the plots shown in this report, the full simulations can be viewed as animations included in the code repository. For the animations the colorbar has been fixed in $t = 0$ as well as interpolated using a contour plot, so it is only intended for debugging and nice visualization.

The potentials which determine the environment only vary in number of slits. A wall along the y-axis with thickness 0.02 in x is centered at $x = 0.5$. A high potential $v_0 = 10^{10}$ is used to prevent any tunneling. Then we create openings with y-width of 0.05 centered on $y = 0.5$. In the case with multiple openings they are separated by a length 0.05 and distributed symmetrically.

Our initial experiment is the case for an empty box. Here we have used no potential, and the initial wave packet is circular $\sigma_x = \sigma_y = 0.05$. The total run time is 0.008, and a snapshot of the wave packet after time 0.001 is shown in FIG. 1, along with the deviation in total probability over time. The actual snapshot is of little interest, outside of seeing the wave packet is as intended. The deviation in probability is very nice, differing only with 10^{-15} to 10^{-14} from 1. At this small a scale, it is possible this deviation is affected by the lack of precision in double precision floating point numbers. Using an even smaller time step would then not improve the precision by much.

A single slit in the wall was used for the next experiment. This and all following experiments use a wider wave packet, with a vertical standard deviation $\sigma_y = 0.2$. The purpose of this is to spread the wave across a larger portion of the wall. In FIG. 2 we have plotted a snapshot of the wave packet for time 0.002, as well as the corresponding detector wall plot for $x = 0.8$. We can see that a significant amount of the wave is reflected, while a portion moves through the gap and spreads out as expected. The detector agrees with this, and we can see the particle is most likely to be found for $y = 0.5$ while decreasing with distance.

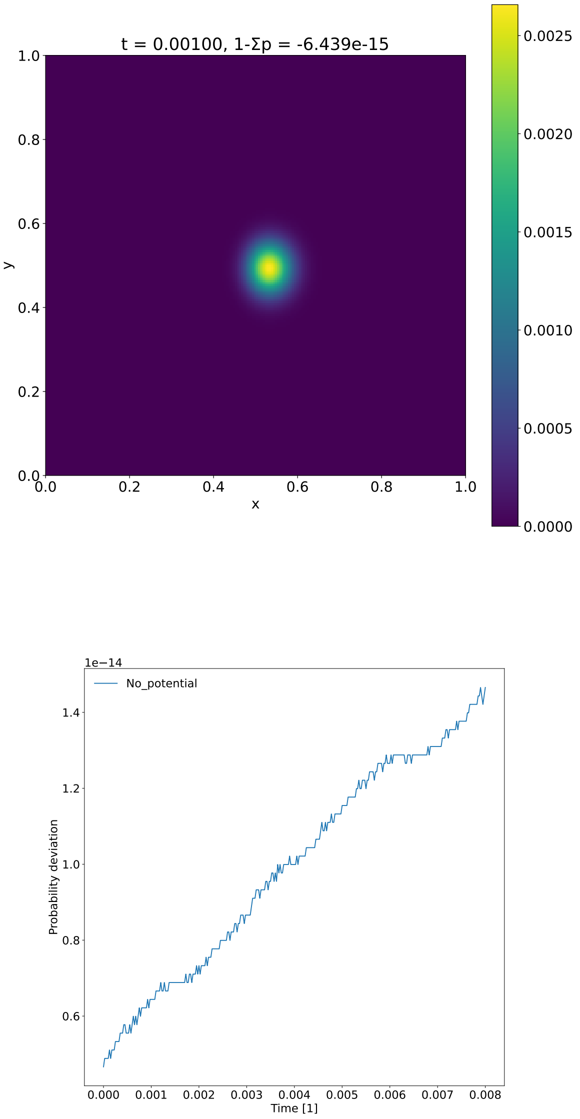

FIG. 1. TOP: Snapshot of wave packet from simulation with no potential at time $t = 0.001$. BOTTOM: Deviation in total probability from 1 as function of time.

Seeing that the simulation is working as intended, we move to the star of the show: The double slit experiment. In FIG. 3 we can see the progression in time as snapshots for times $0.001, 0.002$, and 0.003. Here we can see how the wave starts, enters the two slits, then exits with the characteristic interference pattern. This pattern for $x = 0.8$ is seen more clearly in the top plot in FIG. 5, where three peaks emerge. The deviation in total probability also stays very low just as in the no potential case, as can be seen in FIG. 6, where it is compared with the single

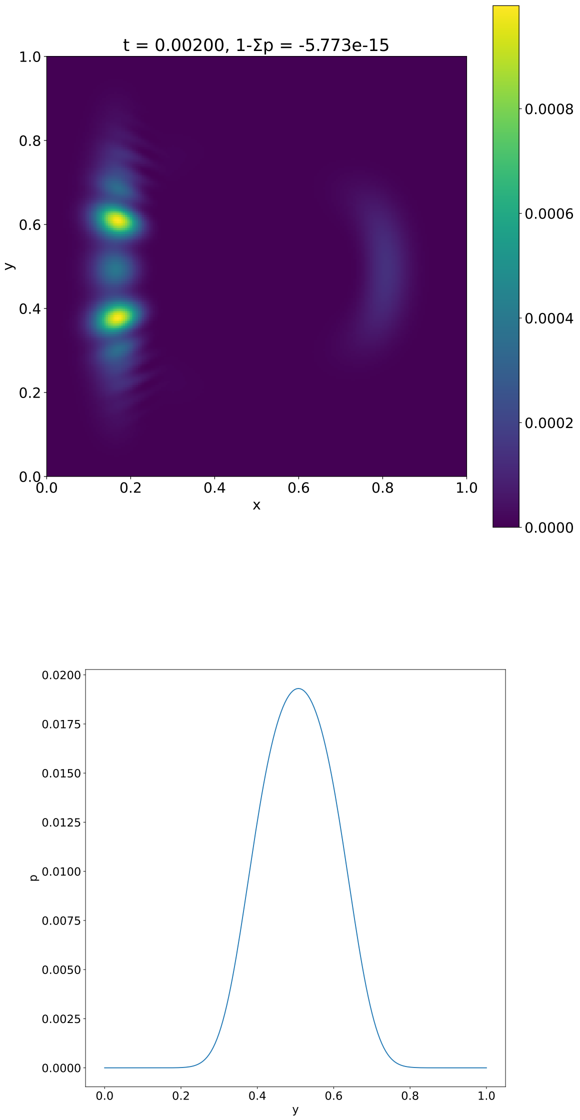

FIG. 2. TOP: Snapshot of wave packet from simulation with a single slit at time $t = 0.002$. BOTTOM: Probability $p(y|x = 0.8; t = 0.002)$ for single slit experiment.

and triple slit experiments.

Finally we do the same procedure with the triple slit experiment. Snapshots from the simulations can be seen in FIG. 4, and we can see that it is very similar to the double slit case. This also applies when looking at the detector screen. In FIG. 5 we in the bottom plot that there are four major peaks as opposed to the three from the double slit case.

As a bonus we also plotted the real and imaginary parts of the wave functions. A comparison of the real and imaginary wave function for the double slit experiment at time $t = 0.001$ can be seen in FIG. 7. We can see that

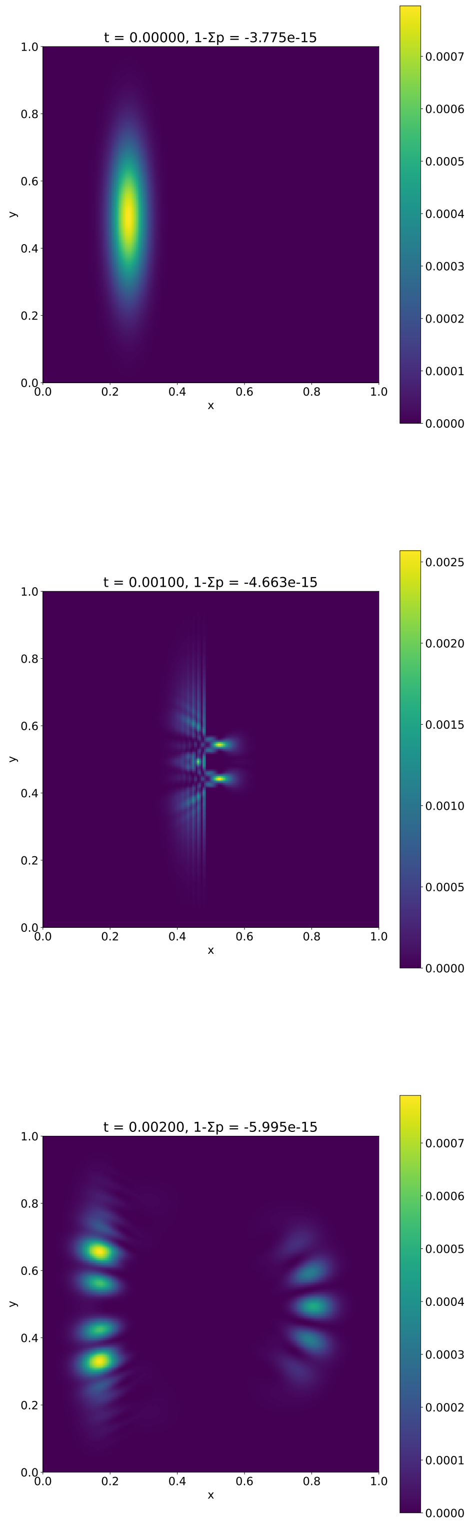

FIG. 3. Time evolution of a double slit simulation. Snapshots progress in time $t = 0.001, 0.002, 0.003$ from top to bottom.

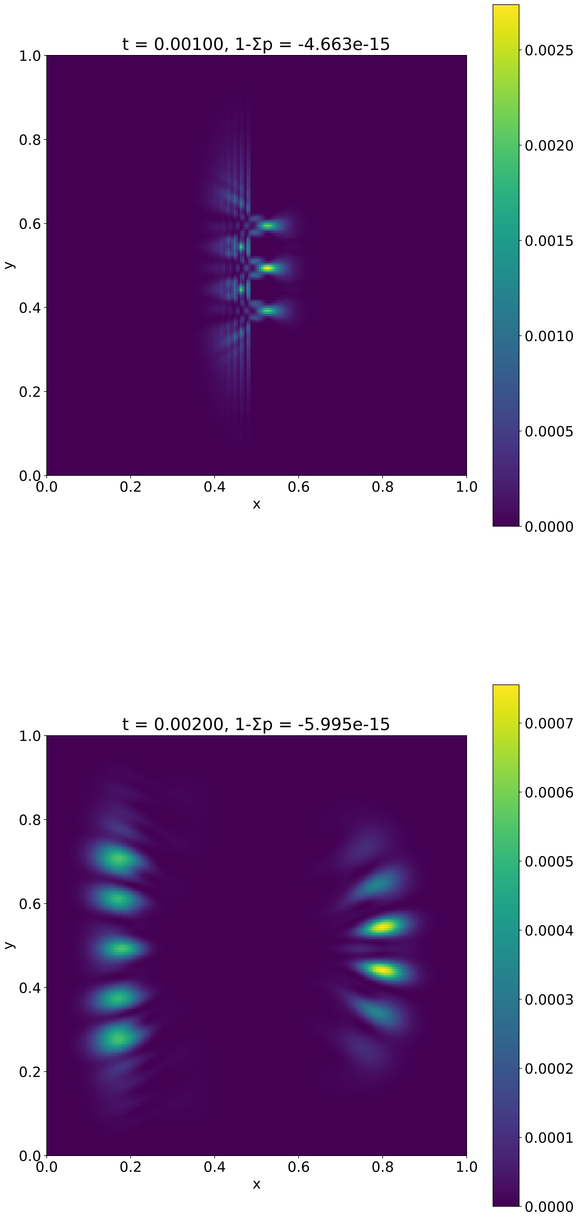

FIG. 4. Snapshots from the triple slit simulation. TOP: Time $t = 0.001$. BOTTOM: Time $t = 0.002$

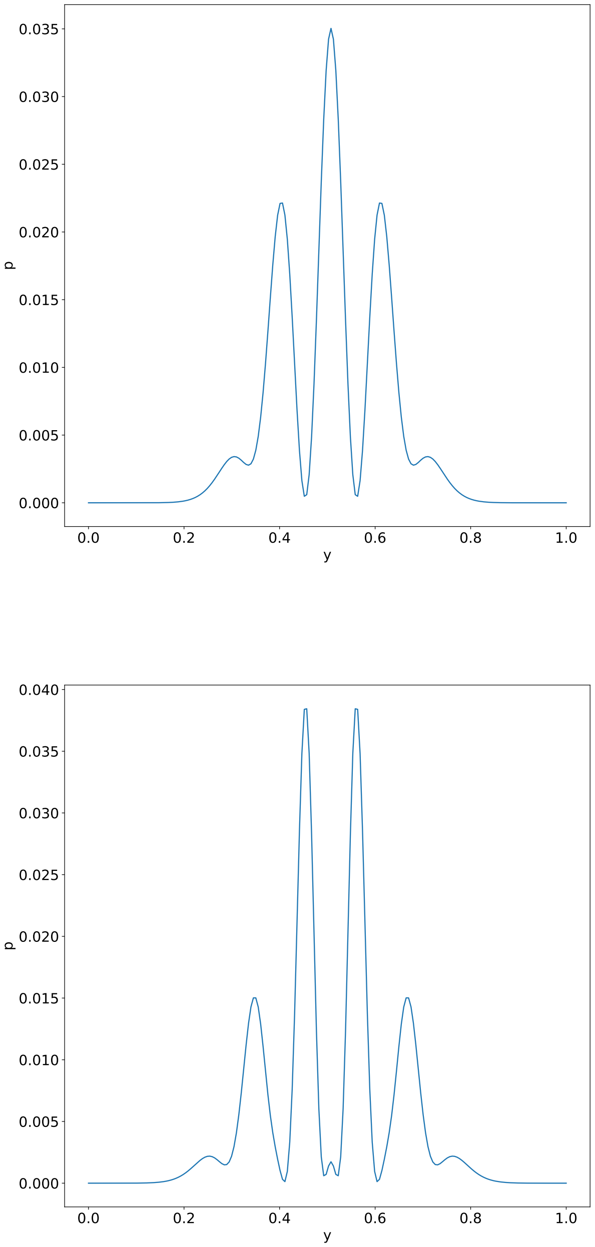

FIG. 5. Comparison of probability observed at detector $p(y|x = 0.8; t = 0.002)$ for two different simulations. TOP: Double slit case. BOTTOM: Triple slit case.

they consist of opposing waves, which together make up the full probability. They are very similar both to each other and the full probability.

IV. CONCLUSION

The simulations were very successful, creating convincing results and with conserved probability. The probability only deviated on scales 10^{-15} to 10^{-14}, which is nearing the limit for a double precision number. Interference patterns were observed for the simulations with double and triple slits.

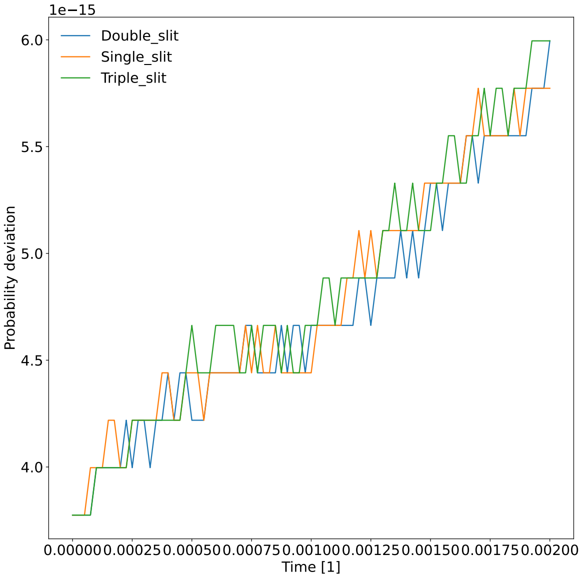

FIG. 6. Comparison of deviation in total probability for all cases with a potential gap.

[1] Conrad Sanderson and Ryan Curtin. Armadillo: a template-based c++ library for linear algebra. *Journal of Open Source Software*, 1(2):26, 2016.

[2] Xiaoye S. Li. An overview of SuperLU: Algorithms, implementation, and user interface. *ACM Transactions on Mathematical Software*, 31(3):302–325, September 2005.

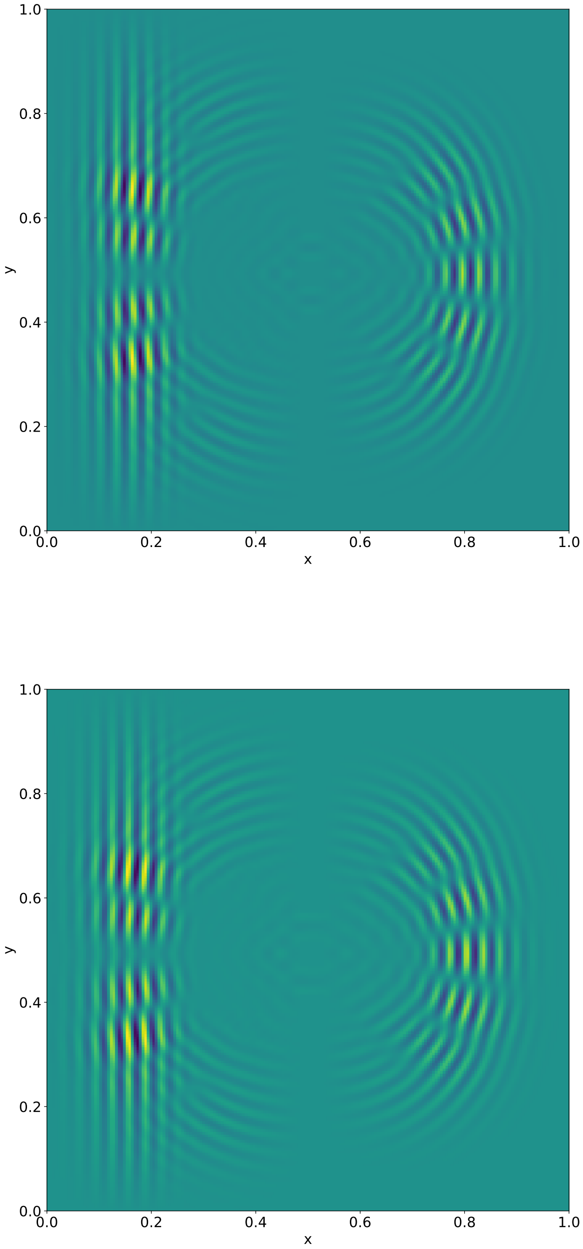

FIG. 7. Comparison of real and imaginary part of the wave function for double slit experiment, time $t = 0.001$. TOP: Real part. BOTTOM: Imaginary part.

Appendix A: Discretizing the Schrödinger equation

We start at the dimensionless Schrödinger equation:

$$i\frac{\partial u}{\partial t} = -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} + v(x,y)u \tag{A1}$$

To discretize we apply the Crank-Nicolson approach in two dimensions:

$$\frac{\partial u}{\partial t} = F(x,y,t) \tag{A2}$$

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} = \frac{1}{2}\left(F_{i,j}^{n+1} + F_{i,j}^{n}\right) \tag{A3}$$

To get the required form (A2) we simply divide our equation (A1) by i, using $i^{-1} = -i$. Note that while we use the letter i for both indexing and imaginary unit, the index only appears as a subscript and the imaginary unit as a factor. We use the central derivative to find the second derivatives:

$$\frac{\partial^2 u_{i,j}^n}{\partial x^2} = \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} \tag{A4}$$

With the same approach for derivatives with respect to y and in time step $n+1$. The right hand side then becomes:

$$F_{i,j}^n = i\left[\frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} + \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2} - v_{i,j}u_{i,j}^n\right] \tag{A5}$$

Which lets us put everything together:

$$\begin{aligned}\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} =& \frac{i}{2}\Bigg[\frac{u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}}{(\Delta x)^2} + \frac{u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}}{(\Delta y)^2} \\ &+ \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} + \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2} \\ &- v_{i,j}u_{i,j}^{n+1} - v_{i,j}u_{i,j}^n\Bigg]\end{aligned} \tag{A6}$$

Here we define equal step size $\Delta x \equiv \Delta y \equiv h$ (not to be confused with the Planck constant, which has been removed). We also multiply by Δt and define $r \equiv i\Delta t/2h^2$, which simplifies to:

$$\begin{aligned}u_{i,j}^{n+1} - u_{i,j}^n =& r\left[u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}\right] + r\left[u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}\right] \\ &+ r\left[u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}\right] + r\left[u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}\right] \\ &- \frac{i\Delta t}{2}v_{i,j}u_{i,j}^{n+1} - \frac{i\Delta t}{2}v_{i,j}u_{i,j}^n\end{aligned} \tag{A7}$$

Then we collect terms for time steps on different sides:

$$\begin{aligned}u_{i,j}^n - r\left[u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}\right] - r\left[u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}\right] + \frac{i\Delta t}{2}v_{i,j}u_{i,j}^{n+1} \\ = u_{i,j}^{n+1} + r\left[u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}\right] + r\left[u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}\right] - \frac{i\Delta t}{2}v_{i,j}u_{i,j}^n\end{aligned} \tag{A8}$$

Which is our Schrödinger equation discretized for the Crank-Nicolson method.