Simulating a wave packet inside a square box using the Crank-Nicolson method

FYS3150

Written by:

Sigurd Kjelsbøl Huse and Odin Johansen

Department of Physics UiO

UiO : University of Oslo

March 8, 2023

Contents

1. Abstract

When simulating a wavepacket inside a square box, where the Schrodinger equation describes the position and time-evolution. By discretizing the system and utilizing the Cranc-Nicolson method to solve the simplified Schrodinger PDE numerically; properties corresponding to a wave appear when representing a particle as a wave packet. We can investigate the diffraction patterns created by this "wave" by introducing a wall with two, or more slits into the box. Utilizing the Cranc-Nicolson method, the sum of probabilities deviates by $1.31 \cdot 10^{-14}$ absolutely from the expected value of one, for a system with no wall after 320 time-steps. While for a system with a wall with two slits, it deviates $1.22 \cdot 10^{-14}$ absolutely from one after 320 time-steps. Doing simulations with two, and three slits; a clear diffraction pattern appears as expected by classical wave theory.

Source code:
https://github.com/SigurdHuse/FYS3150/tree/main/Project5

2. Introduction

Asking good questions is never easy, asking the right question is almost impossible. In 1900 Max Planck started developing a theory, that later became the field known today as quantum physics. One of the fundamental differences between quantum physics and classical physics was the underlying probabilistic nature. In 1927 Clinton Davisson, and Lester Germer showed that electrons had the same wave-properties[4] as photons, which had been proven earlier by Thomas Young[2].

Quantum physics is an incredibly complex and fascinating field of science that seeks to understand the behavior of matter and energy at the smallest scales. Quantum physics has opened up a world of possibilities, from new materials and technologies, to a greater understanding of the fundamental nature of the universe. It has also led to revolutionary advances in computing, communication, and medicine. In short, quantum physics has revolutionized our understanding of the physical world and the possibilities of what can be achieved. Today, the particle-wave duality phenomenon is utilized in many different fields of technology such as in electron microscopy, where the small wavelength associated with electrons is used to view objects much smaller than one can view with visible light[5].

If we wish to understand how such a particle acts, one can solve the shrödinger equation with different potentials. One alternative for doing this, is implementing algorithms for solving partial differential equations (PDEs), in their discretized representation. As the shrödinger equation can be described as a PDE, such algorithms makes it possible to solve it numerically. Developing these kinds of algorithms, the goal is to have them compute the next time step efficiently and with high precision. One algorithm that achieves these goals well, are the analytical Crank-Nicolson method or possibly the numerical eigendecomposition from Jacobi's rotational algorithm. To simplify the calculation of the system's time evolution, a time-independent potential can be assumed. The now static potential can then be modeled into a wall with slits, thus recreating our own version of the double-slit experiment.

Experiments such as the one performed in this paper, are important. As they allow us to gain further insight into the behavior of a wave packet described by the shrödinger equation. The wave-particle duality which we observe that the wave packet possesses is still one of the great mysteries of physics. Meaning that further studies of the phenomenon and a deeper understanding could improve technology that already uses it as well as pave the way for new innovations.

3. Theory

The goal of this paper is to investigate how a wave packet, described by the Schrodinger equation evolves over time, in a system consisting of a square box. Each point in the system consists of a complex number, which represents the current state of that point. For simplicity, all units in the paper are scaled away, meaning all values are unitless.

i. Describing the system

1. The box

A square box, with side length 1[1] in the xy-plane will in this paper be used to represent the system which, for the rest of the paper will be referred to as $u(x, y, t)$ where x, and y are the x- and y-coordinate in the box, and t is the time. To simulate the system, the grid will be discretized with M points in the y-, and x-direction. The number of points along each axis M, is defined by

$$M = \left\lfloor \frac{1}{h} \right\rfloor + 1. \tag{1}$$

Where h is a defined step size. Using this grid model, each point in the box can be represented as (x_i, y_i),

$$x_i = i \cdot h \qquad y_j = j \cdot h \qquad i, j \in [0, 1, \ldots, M-1]. \tag{2}$$

2. Discretizing time

As continuous time is not possible on a computer, the time of the system t is also going to be discretized as

$$t \to t_n = n\Delta t \qquad n = 0, 1, \ldots, N_t. \tag{3}$$

N_t is the number of time steps in the simulation, and Δt is the size of the time step. Using the notation in equations (2, 3), $u(x, y, t) \to u(x_i, y_j, t_n)$ in the discretized representation of the system. Also noting that if the system is simulated for a time T, with a time step Δt, then the number of time steps is $\frac{T}{\Delta t}$.

3. Boundary conditions

To make sure the wave packet stays inside the square box, a Dirichlet boundary condition will be used. Meaning all boundary points are set to 0 at all times

$$u(x = 0, y, t) = 0, \quad u(x = 1, y, t) = 0, \quad u(x, y = 0, t) = 0, \quad u(x, y = 1, t) = 0, \ \forall x, y \in [0, 1], t \geq 0. \tag{4}$$

4. Potential of each grid point

To be able to say something about how the wave packet will move in the box, the potential V of each point also needs to be specified. The potential $V(x, y)$ in the shrödinger equation, is what represents the environment the particle exists in. For the discretized system, V will be a matrix of size $(M-2) \times (M-2)$ as the boundary points are excluded. For all the internal points (x_i, y_j), the corresponding element in V will be $V(i-1, j-1)$, denoted v_{ij}. The internal points, are all points which are not part of the outer edge of the discretized grid. While on the other hand, the boundary points are all the points along the outer edge of the discretized grid.

5. *Finalizing notation*

Finalizing the notation, the current discretized state of the system will be referred to as U^n, for the rest of the paper. Where n is not to be confused with a power, but denotes the current time step. As the system is represented as a square grid, U^n is a $M \times M$ grid with elements u_{ij}^n. Where u_{ij}^n is the grid element (x_i, y_j) at time step n. An example of a discretized grid can be found in figure (1).

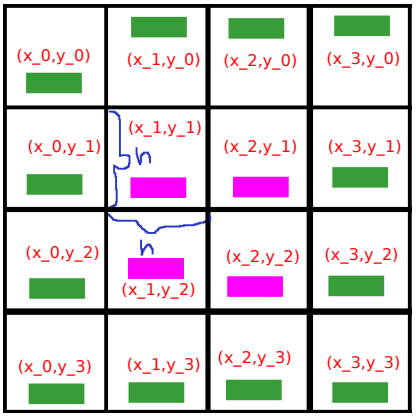

Figure 1. Example of a grid where $h = 1/3$, and thus $M = 4$. The squares with a green rectangle are boundary points, and the squares with a magenta rectangle are internal points.

ii. Discretizing the Schrödinger equation using the Crank-Nicolson method

The idea behind the Crank-Nicolson method, is to combine the Forward, and the Backwards Euler method to get a method, for solving PDEs. Before introducing this method, a closer look at the Schrödinger equation which describes the movement of the wave packet is needed.

1. *The Schrödinger equation*

As stated previously, the movement of the wave packet inside the box, is described by the Schrödinger equation

$$i\hbar \frac{d}{dt}|\mathbf{\Psi}\rangle = \hat{H}|\psi\rangle. \tag{5}$$

Where $\hat{H}$ is some Hamiltonian operator, and $|\mathbf{\Psi}\rangle$ is the quantum state. In this paper, we consider the case of a single relativistic particle in two dimensions. Working in "position space", the quantum state $|\mathbf{\Psi}\rangle$ can be expressed as a complex-valued function $\mathbf{\Psi}(x, y, t)$, referred to as the wave function. With these new considerations, equation (5) becomes the following PDE:

$$i\hbar \frac{\partial}{\partial t}\mathbf{\Psi}(x, y, t) = -\frac{\hbar^2}{2m}\Big(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\Big)\mathbf{\Psi}(x, y, t) + V(x, y, t)\mathbf{\Psi}(x, y, t). \tag{6}$$

With $V(x, y, t)$ being the potential of a grid point (x, y), at time t, and m being the particle mass. For simplicity, all units are scaled away, and only the time-independent potential $V(x, y)$ is considered. Transforming equation (6) into the final PDE which will be studied further in this paper

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

As described in section 3 i 1, $u(x, y, t)$ is the value at point (x, y) at time t in the box, which in equation (7) has the shorthand notation u.

2. Forward, and backwards Euler

Before delving into the Crank-Nicolson method, it will to insightful to look at where it comes from. While also introducing the notation for PDEs, which will be used in this paper. The state of a single grid point in the discretized grid at time step n, is as described in section 3 i 5 denoted u_{ij}^n. The goal of any PDE solver is to determine what will happen at the next time step u_{ij}^{n+1}. To achieve this, it's first necessary to describe how the position u changes as a function of time

$$\frac{\partial u}{\partial t} = F(u, x, y, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial u^2}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial y^2}) = F(\mathbf{z}). \tag{8}$$

For simplicity, the following notation is introduced: $u, x, y, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial u^2}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial y^2} \equiv \mathbf{z}$. Function $F(\mathbf{z})$, is determined by the PDE describing the system. The problem with equation (8), is that it is continuous making it impossible to simulate on a computer without an analytical solution. To avoid finding an analytical solution, as it is often very hard or even impossible, the Forward, and Backwards Euler method becomes relevant. These methods, for discretizing can be used on equation (8) to get an estimate, for u_{ij}^{n+1}. The forward Euler method estimates u_{ij}^{n+1} by

$$\frac{u_{ij}^{n+1} - u_{ij}^n}{\Delta t} = F_{ij}^n(\mathbf{z}) \tag{9}$$

Where Δt is the time step in the simulation, and $F_{ij}^n(\mathbf{z})$ is the function F with coordinates (x_i, y_j), for u at time step n. The partial derivatives in F_{ij}^n are discretized, meaning that the double partial derivatives are represented as

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j^n}}{(\Delta x)^2} + \mathcal{O}((\Delta x)^2), \quad \frac{\partial^2 u}{\partial y^2} = \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2} + \mathcal{O}((\Delta y)^2). \tag{10}$$

The error estimates $\mathcal{O}((\Delta x)^2) and \mathcal{O}((\Delta y)^2)$ in O-notation will be ignored, for the rest of the paper. The values Δx, and Δy are the step sizes in x-and y-direction in the discretization. Similarly Backwards Euler estimates u_{ij}^{n+1} by

$$\frac{u_{ij}^{n+1} - u_{ij}^n}{\Delta t} = F_{ij}^{n+1}(\mathbf{z}). \tag{11}$$

The difference between Forward, and Backwards Euler is that Forward is an explicit method while Backwards is an implicit method. Meaning that Forward Euler gives the next step directly, while Backwards Euler gives an equation to solve, for the next time step.

3. The Crank-Nicolson method

Putting it all together, the Crank-Nicolson method is defined as a linear combination of the Foward, and Backwards Euler method defined in equations (9, 11), meaning that the Crank-Nicolson method estimates u_{ij}^{n+1} by

$$\frac{u_{ij}^{n+1} - u_{ij}^n}{\Delta t} = \frac{1}{2}F_{ij}^{n+1}(\mathbf{z}) + \frac{1}{2}F_{ij}^n(\mathbf{z}). \tag{12}$$

Using equations (10, 12), and noting that $\Delta x = \Delta y = h$, the PDE describing the wave packet in equation (7) is discretized to

$$\begin{aligned} i\frac{u_{ij}^{n+1} - u_{ij}^n}{\Delta t} = &\frac{1}{2}\Big(- \frac{u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}}{h^2} - \frac{u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}}{h^2} + v_{ij}u_{ij}^{n+1}\Big) \\ &+\frac{1}{2}\Big(- \frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}}{h^2} - \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{h^2} + v_{ij}u_{ij}^{n}\Big). \end{aligned} \tag{13}$$

Simplifying equation (13), both sides are multiplied by $-i\Delta t$, and shorthand notation $r \equiv \frac{i\Delta t}{2h^2}$ is introduced. Further, all components with the same time step are moved to the same side

$$\begin{aligned} u_{ij}^{n+1} - r(u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}) - r(u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}) + \frac{i\Delta t}{2} v_{ij} u_{ij}^{n+1} \\ = u_{ij}^{n} + r(u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}) + r(u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}) - \frac{i\Delta t}{2} v_{ij} u_{ij}^{n}. \end{aligned} \tag{14}$$

The expression in equation (14) is the discretized form of equation (7) according to the Crank-Nicolson method. Solving equation (14), it's possible to find the next time step for each internal grid point, which will be studied further in this paper.

iii. Computing the probability of a grid square

To be able to say something about how the probabilities of the wave packet evolve over time, an expression for the probability it has of being at a grid point (x, y), at time step t is needed. This probability $p(x, y|t)$, for point (x, y) at time t is defined by the Born rule as

$$p(x, y|t) = |u(x, y, t)|^2 = u^*(x, y, t) \cdot u(x, y, t). \tag{15}$$

Given that the sum of probabilities in the system is normalized to equal 1. In equation (15), $u^*(x, y, t)$ is the complex conjugate of $u(x, y, t)$.

iv. Diffraction and Particle diffraction

Diffraction is a phenomenon that occurs when waves pass through a small opening or around an obstacle, causing them to bend and spread out. This is due to the properties a wave posses, and results in an interference pattern on the other side. Diffraction can be observed in both light and sound waves, and is an important concept in optics and acoustics.

The diffraction of waves is described by the Huygens-Fresnel Principle, which states that every point along a wavefront can be seen as a source of secondary waves. These secondary waves interfere with each other, creating the diffraction pattern. The angle of diffraction depends on the size of the opening relative to the wavelength of the wave. Diffraction is more pronounced for shorter wavelengths, and can be used to measure the wavelength of a wave.

1. Particle diffraction

Particle diffraction is a quantum mechanical phenomenon and is described by the Schrödinger equation in equation (5). The diffraction of particles occurs when the particles pass through a potential barrier, such as a double slit, and is described by the wave-packet theory. This theory states that the wave-function of the particle is modified by the potential barrier, resulting in a diffraction pattern.

The diffraction of particles can be used to measure the de Broglie wavelength ($\lambda_b = \frac{h}{p}$) of particles, as well as to study the wave-like properties of particles. Particle diffraction can also be used to study the structure of crystals, due to the wave-like interference of the particles with the crystal lattice.

4. Method

As the Crank-Nicolson method is a linear combination of a direct, and implicit method, it is an implicit method. Which makes it necessary to solve a system of equations to obtain the next time-step in the simulation. To achieve this, the system of equations that follows from using the Crank-Nicolson method on all internal grid points is first written in matrix notation. Then one numerical, and one analytical method is discussed, for solving the derived matrix equation.

i. Initializing the potential matrix

However, before discussing methods, for solving a matrix equation on a computer. It is necessary to define some more technical aspects of the simulation. Starting with defining the potential matrix V.

1. No slits

To represent the potential of each grid square in the system, a zero-indexed matrix V of size $(M-2)\times(M-2)$ will be used, with elements v_{ij} defined as $v_{ij} = V(i,j)$. The reason for using a $(M-2)\times(M-2)$ matrix, is that the boundary points are not changed in the simulation, as they are always zero. All elements in V will initially be set to 0, as the potential of the wave packet going there is 0.

2. One, or more slits

To be able to simulate the system with one, or more slits, a wall will be initialized with a potential equal to a defined constant v_0. If the value of v_0 is set high enough and the wall is thick enough, it makes it extremely unlikely that the wave packet can pass the wall, and go to the other side of the box. Making it then possible to make slits by setting elements in the wall to be 0. Causing it to be a significantly greater probability for the wave packet to go to the other side of the wall, by going through the defined slits.

ii. Initializing initial state

Moving on, the initial state of the system at U^0, also needs to be defined. As there is nothing to simulate if the system starts with no wave packet.

1. Definition

To initialize the grid with an initial state the following formula can be used,

$$u(x,y,t=0) = e^{\frac{(x-x_c)^2}{2\sigma_x^2} - \frac{(y-y_c)^2}{2\sigma_y^2} + ip_x(x-x_c) + ip_y(y-y_c)}. \tag{16}$$

Where x_c, and y_c are the coordinates of the center of the initial wave packet, σ_x, and σ_y the initial width of the wave packet in x-and y-direction, p_x, and p_y are the wave packet momenta, and i is the imaginary unit. Remembering that the coordinates of the grid are as defined in equation (2), and using equation (16), U^0 can be initialized with the initial wave packet by specifying: x_c, y_c, σ_x, σ_y, p_x, p_y.

2. *Normalizing sum of probabilities to one*

After filling U^0 we want to normalize the sum of probabilities to be equal to one. This makes it possible to interpret the probability of each grid square, as the probability the wave packet has to be there. To achieve this we compute the sum of probabilities, where each probability is as defined in equation (15), and divide each grid point by the square root of this sum. It is necessary to take the square root, as each probability is the square of the value in each grid point.

iii. Writing the Crank-Nicolson method as a matrix equation

With the rest of the simulation well defined, rewriting the Crank-Nicolson method to matrix notation can commence.

1. *Defining submatricies*

Starting with defining four matrices R, α_i, β_i, and O all with size $(M-2)\times(M-2)$.

$$\alpha_i = \begin{pmatrix} a_{(M-2)i} & -r & 0 & 0 & \dots & 0 & 0 \\ -r & a_{(M-2)i+1} & -r & 0 & \dots & 0 & 0 \\ 0 & -r & a_{(M-2)i+2} & -r & \dots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & a_{(M-2)i+M-4} & -r \\ 0 & 0 & 0 & 0 & \dots & -r & a_{(M-2)i+M-3} \end{pmatrix}^{(M-2)\times(M-2)} \tag{17}$$

$$\beta_i = \begin{pmatrix} b_{(M-2)i} & r & 0 & 0 & \dots & 0 & 0 \\ r & b_{(M-2)i+1} & r & 0 & \dots & 0 & 0 \\ 0 & r & a_{(M-2)i+2} & r & \dots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & b_{(M-2)i+M-4} & r \\ 0 & 0 & 0 & 0 & \dots & r & b_{(M-2)i+M-3} \end{pmatrix}^{(M-2)\times(M-2)} \tag{18}$$

$$R = \begin{pmatrix} r & 0 & 0 & 0 & \dots & 0 \\ 0 & r & 0 & 0 & \dots & 0 \\ 0 & 0 & r & 0 & \dots & 0 \\ 0 & 0 & 0 & r & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & r \end{pmatrix}^{(M-2)\times(M-2)} \tag{19}$$

$$O = \begin{pmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 \end{pmatrix}^{(M-2)\times(M-2)} \tag{20}$$

The values a_k, and b_k are defined as

$$a_k = 1 + 4r + \frac{i\Delta t}{2} v_{ij}. \qquad b_k = 1 - 4r - \frac{i\Delta t}{2} v_{ij}. \tag{21}$$

In equation (21) the i in $\frac{i\Delta t}{2}$, is the imaginary unit, and the index k maps to element v_{ij} as $k = (M-2)\cdot i + j$.

2. *Defining matrices to represent the Crank-Nicolson method*

The Crank-Nicolson method, where each grid point needs to be computed according to equation (14), is a large expression that quickly can become messy when representing it on a computer. For easier notation, and easier representation of the system of equations that needs to be solved to find the next time step, for each internal grid point. Two matrices A, and B both with size $(M-2)^2 \times (M-2)^2$, will be defined to represent the system of equations needed to be solved, to find the next time state U^{n+1}. Using equations (17, 18,19,20)

$$A = \begin{pmatrix} \alpha_0 & -R & O & O & O & \dots & O \\ -R & \alpha_1 & -R & O & O & \dots & O \\ O & -R & \alpha_2 & -R & O & \dots & O \\ O & O & -R & \alpha_3 & -R & \dots & O \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ O & O & O & O & \dots & -R & \alpha_{M-3} \end{pmatrix}^{(M-2)^2 \times (M-2)^2} \tag{22}$$

$$B = \begin{pmatrix} \beta_0 & R & O & O & O & \dots & O \\ R & \beta_1 & R & O & O & \dots & O \\ O & R & \beta_2 & R & O & \dots & O \\ O & O & R & \beta_3 & R & \dots & O \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ O & O & O & O & \dots & R & \beta_{M-3} \end{pmatrix}^{(M-2)^2 \times (M-2)^2} \tag{23}$$

Using the matrices in equations (22, 23), and remembering the boundary conditions in equation (4), equation (14), for all internal points gets rewritten into

$$Au^{n+1} = Bu^n = \vec{b}. \tag{24}$$

With the shorthand notation, $\vec{b} \equiv Bu^n$. The vector u^n is the vector representation of all the internal points in U^n. Solving the system of equations in equation (24), it's possible to go one time step forward in the simulation. The method for solving this, and how to write U^n as the vector u^n, will be discussed in more detail in section 4 iv.

iv. Solving the system of equations from the Crank-Nicolson method

Having written the system of equations, for all internal grid points, in matrix notation. The problem of solving said system, can be discussed in detail. With one analytical, and one numerical method being presented. An analytical approach usually first rewrites the system, before using Gaussian elimination to find an answer. While a numerical approach aims to get as close as possible to the true value, through approximation. With the analytical approach often giving a more precise answer, at the cost of increased computation time. It is also worth noting that the answer from both approaches is actually technically "wrong", due to the limits of floating point representation on a computer, and discretization error. The computer can not with a hundred percent accuracy represent floating numbers, and this machine error causes a "wrong" answer when compared to doing it by hand. This trade-off is however worth it, as equation (24) is not feasible to solve by hand, and the answer's loss in precision due to machine error is negligible.

1. *Writing the system as a vector*

To solve equation (24), we first need to write the system's current state U^n as a vector u^n. The vector u^n is $(M-2)^2$ long, and consists of the column vectors of U^n ignoring the boundary points

$$u^n = (u^n_{1,1}, u^n_{1,2}, \dots u^n_{1,M-2}, u^n_{2,1}, u^n_{2,2}, \dots, u^n_{M-2,M-2}). \tag{25}$$

To write equation (25) more compact, we observe that

$$u^n((M-2)\cdot(i-1)+j-1) = U^n_{i,j}, \quad i,j \in [1,2,\ldots,M-2]. \tag{26}$$

From equations (24, 26) the following system of equation follows

$$Au^{n+1} = \vec{b}. \tag{27}$$

Which is the system of equations which is going to be solved, to find the next time step u^{n+1}.

2. *Solving Crank-Nicolson*

To solve equation (27) on the computer, the "spsolve" command in the armadillo package [1] will be utilized, with the "superlu" parameter. The reason for choosing "spsolve", is that both the A, and B matrix are represented as a sparse matrix as most elements are zero. A sparse matrix does not store zero values, and it is therefore efficient in this case. Doing this it's possible to find u^{n+1}, and thus going one time-step forward in the simulation.

The "spsolve" method with the "superlu" parameter, finds the LU decomposition of the matrix, and then solves the following matrix equation using forward, and backward substitution. This is an analytical approach, meaning the correct result is achieved, assuming it is possible to solve the system of equations in the first place, and the numbers in the equations are of such magnitude that they can be represented accurately on the computer. Although, at the expense of a higher computational time, than when compared to numerical approaches generally. On the flip side, the analytical approach has, as mentioned previously, a higher precision usually. One such approach is doing an eigendecomposition of A using the Jacobi rotation Algorithm.

3. *Discussing one numerical approach*

The idea behind the Jacobi rotation Algorithm is to approximate the eigenvectors, and eigenvalues of a given matrix, by iteratively getting it closer to a similar diagonal matrix. As matrix A is almost already a diagonal matrix, and is symmetrical. We would expect the Jacobi rotation algorithm to converge fairly quickly, giving a good approximation of the eigenvectors, and eigenvalues. Using these approximated eigenvectors, and eigenvalues it's possible to do an eigendecomposition, writing A as

$$A = Q\Lambda Q^{-1} = Q\Lambda Q^T \tag{28}$$

Where Q is a matrix of size $(M-2)^2 \times (M-2)^2$, with the i'th column being the i'th eigenvector. While Λ is a diagonal matrix of size $(M-2)^2 \times (M-2)^2$, with the i'th element along the diagonal being the i'th eigenvector λ_i. The reason $Q^{-1} = Q^T$, is because A is symmetric. This, and the fact that the elements in Λ^{-1} are $1/\lambda_i$, is what makes this numerical approach potentially faster than "spsolve", as Q^{-1}, and Λ^{-1} are fast to compute when compared to finding the inverse of a matrix trough standard methods such as Gaussian elimination. Using equation (28), equation (24) gets rewritten to

$$u^{n+1} = Q\Lambda^{-1}Q^T\vec{b}. \tag{29}$$

The approach in equation (29), for going one time-step forward in the simulation, would most likely be faster than using "spsolve". With one important aspect, of this method, being that the potential is time-independent. This makes it possible to only do one eigendecomposition of A, thus making the computation time significantly lower than if we had to do an eigendecomposition at each time step. For the case of a time-dependent potential, another numerical method, or possibly an analytical method would most likely be preferred. Although it is possible to determine how much faster or slower the Jacobi rotation Algorithm would be, and what the impact on precision would be. Determining them would be beyond the scope of this paper, and could be a starting point, for further research.

v. Simulations

By first filling in the initial wave packet, and using the method described in section 4 iv, the following systems were simulated:

- $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.008$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.05$, $p_y = 0$, $v_0 = 0$, for zero slits
- $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.008$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.10$, $p_y = 0$, $v_0 = 10^{10}$, for two slits
- $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, $v_0 = 10^{10}$, for two slits
- $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, $v_0 = 10^{10}$, for one slit
- $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, $v_0 = 10^{10}$, for three slits

For all simulations with slits, the slit setup was symmetrical around $y = 0.5$, with wall configurations:

- Wall thickness in x-direction: 0.2
- Wall position in x-direction: 0.5
- Length of wall piece separating slits: 0.05
- Slit aperture: 0.05

5. Discussion, and results

We start by looking at the conservation of probability in the system for all time steps. We would theoretically expect the sum of probabilities to equal 1. But, as we are simulating the system on a computer with a discretized grid, a loss of precision is expected. Both from floating point operations, as the machine cannot one hundred percent accurately represent floating values, and discretization errors. For example, as described in section 3 ii 2, the discretization of $\frac{\partial^2 u}{\partial x^2}$ has an error of magnitude $\mathcal{O}((\Delta x)^2)$.

1. *Sum of probabilities absolute deviation from 1*

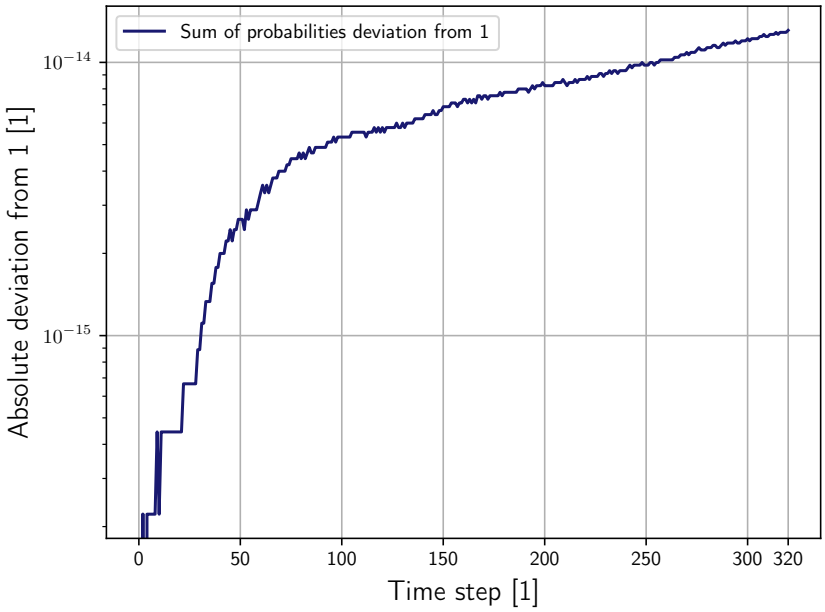

Figure 2. Plot of the absolute value of 1 minus the sum of probabilities in the system as a function of each time step. The system has no wall, and $h = 0.005$, $\Delta t = 2,5 \cdot 10^{-5}$, $T = 0.008$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.05$, and $p_y = 0$.

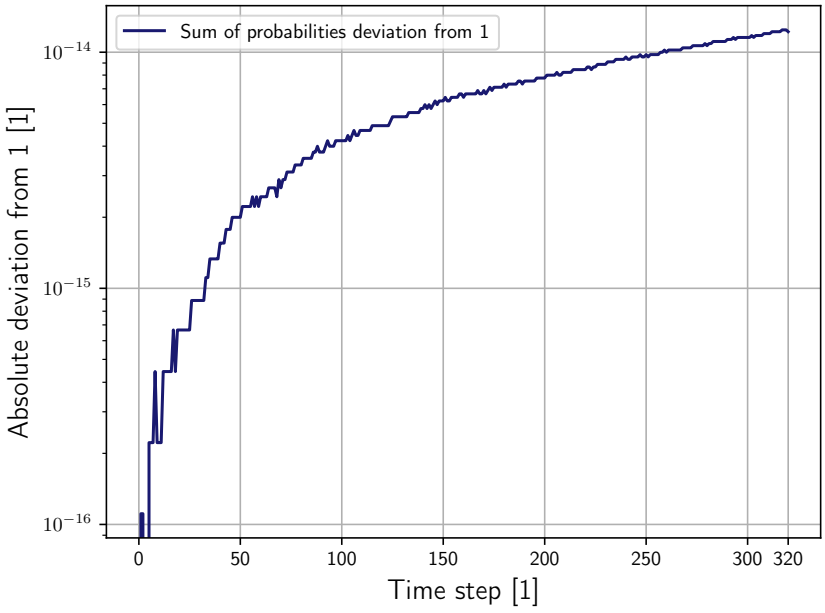

Figure 3. Plot of the absolute value of 1 minus the sum of probabilities in the system as a function of each time step. The system has two slits, and $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.008$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.10$, $p_y = 0$ and $v_0 = 10^{10}$.

The simulations which will be investigated are shown if figures (2, 3). After 320 time steps, in the system with no wall the sum of probabilities deviates $1.31 \cdot 10^{-14}$ absolutely from one. While in the system with two slits the sum of probabilities deviates $1.22 \cdot 10^{-14}$ absolutely from one. It is clear from these values, that we have about the same conservation of probability, for a system with no wall and a system with two slits.

The machine precision for a double, which is the datatype used to store values in this paper, is in c++ approximately $1.0 \cdot 10^{-16}$ [3]. Which when compared to the deviation for no wall, $1.31 \cdot 10^{-14}$, and two slits, $1.22 \cdot 10^{-14}$, indicates that the method in 4 iv, has a high precision. The difference between the machine error, and these values could perhaps be caused by an error in the code, or it could simply be the case that this is the best precision we can hope for when simulating with the method described in section 4 iv. With the latter seeming like the more likely of the two options, considering that the error is almost machine precision.

2. *Time-evolution of probability, real component and imaginary component*

Next, the evolution of each grid point's real, and imaginary component, and probability over time will be investigated, in the case of a two-slit simulation.

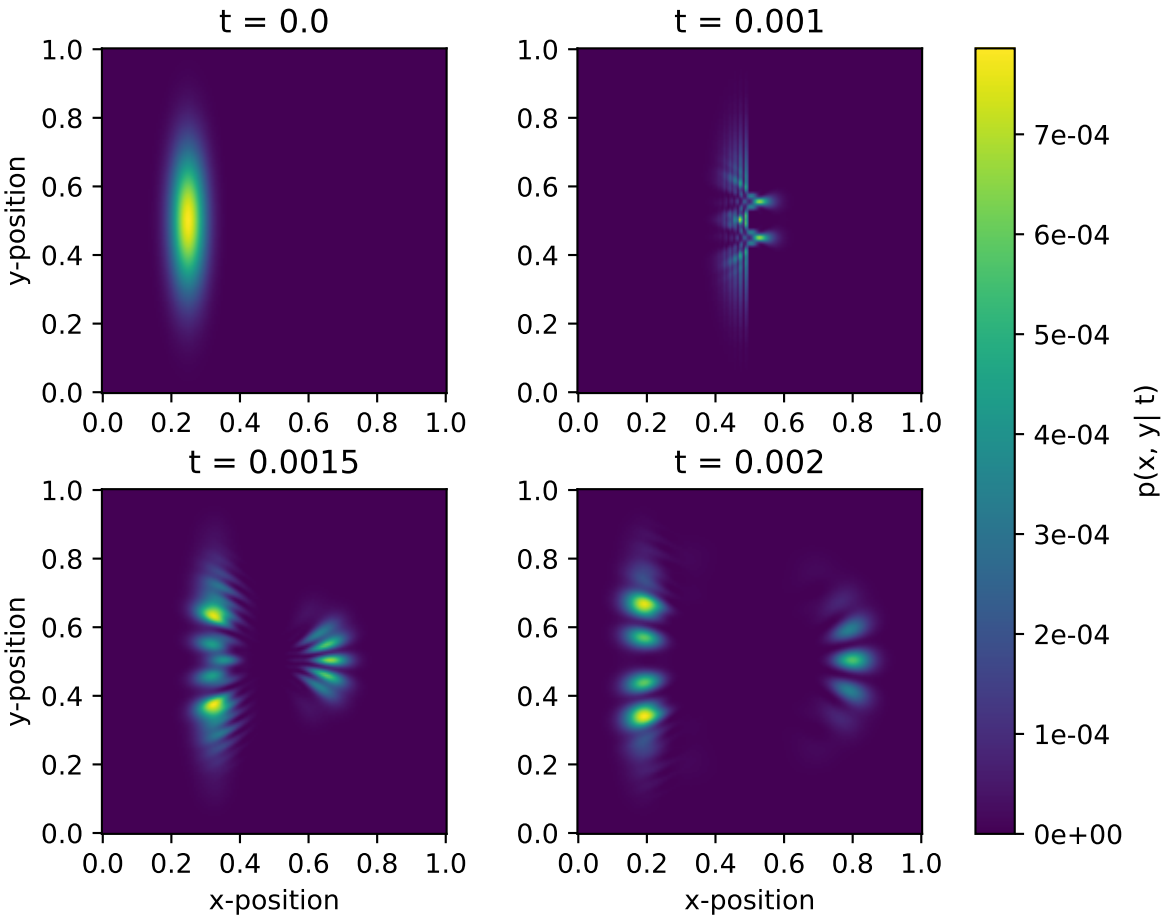

Figure 4. Plot showing the time evolution of the probabilities of a wave packet. Where $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, $v_0 = 10^{10}$, and there are two slits.

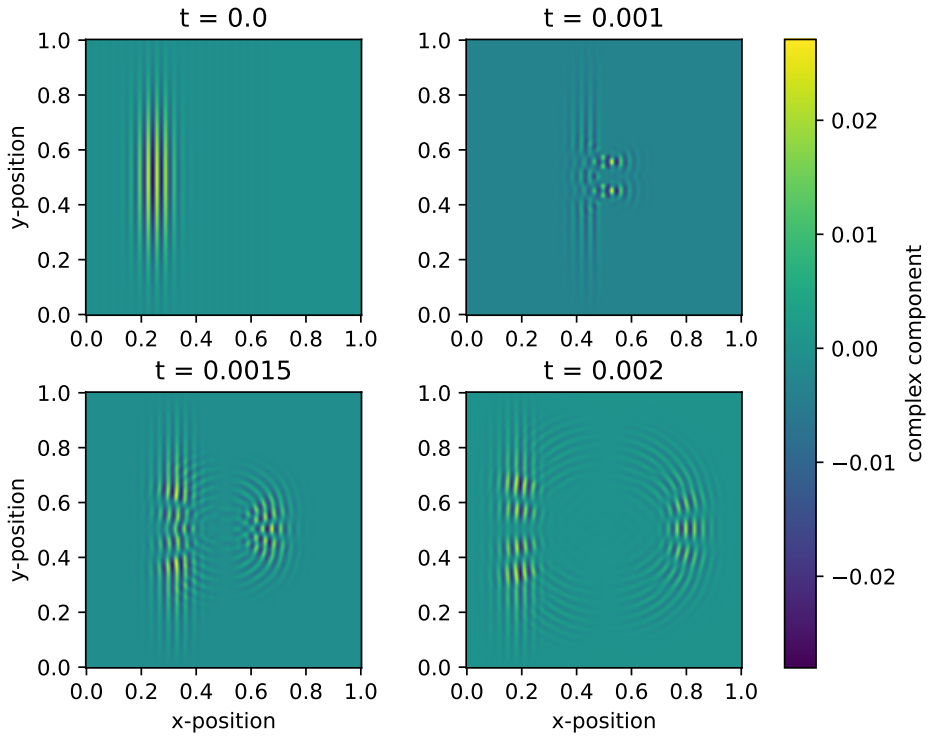

Figure 5. Plot showing the time evolution of the imaginary component of a wave packet. Where $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, $v_0 = 10^{10}$, and there are two slits.

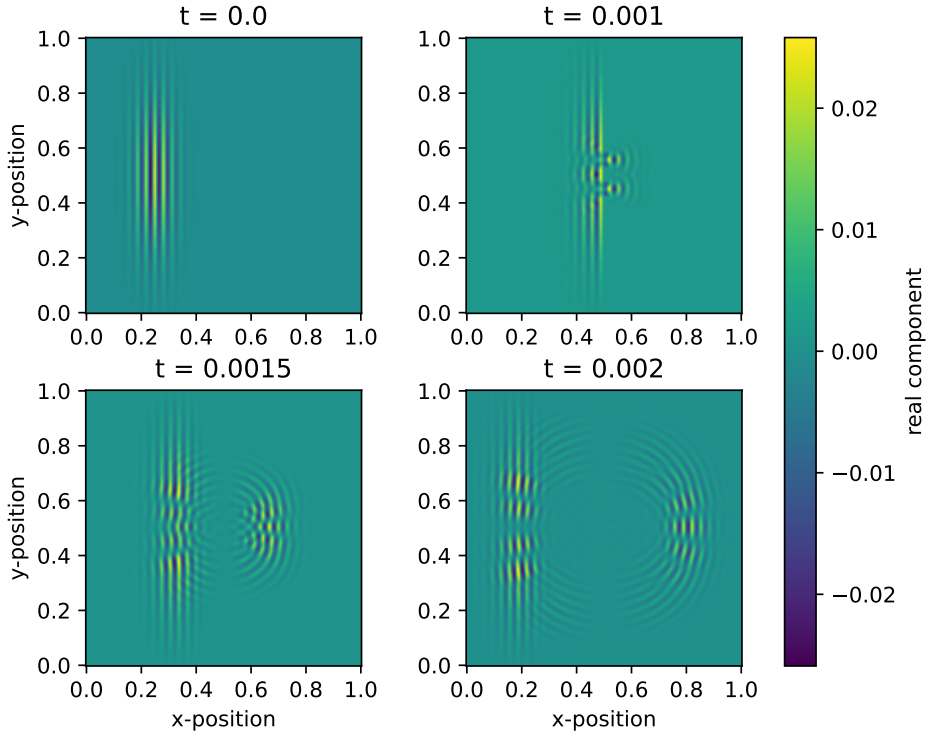

Figure 6. Plot showing the time evolution of the real component of a wave packet at different time steps. Where $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, $v_0 = 10^{10}$, and there are two slits.

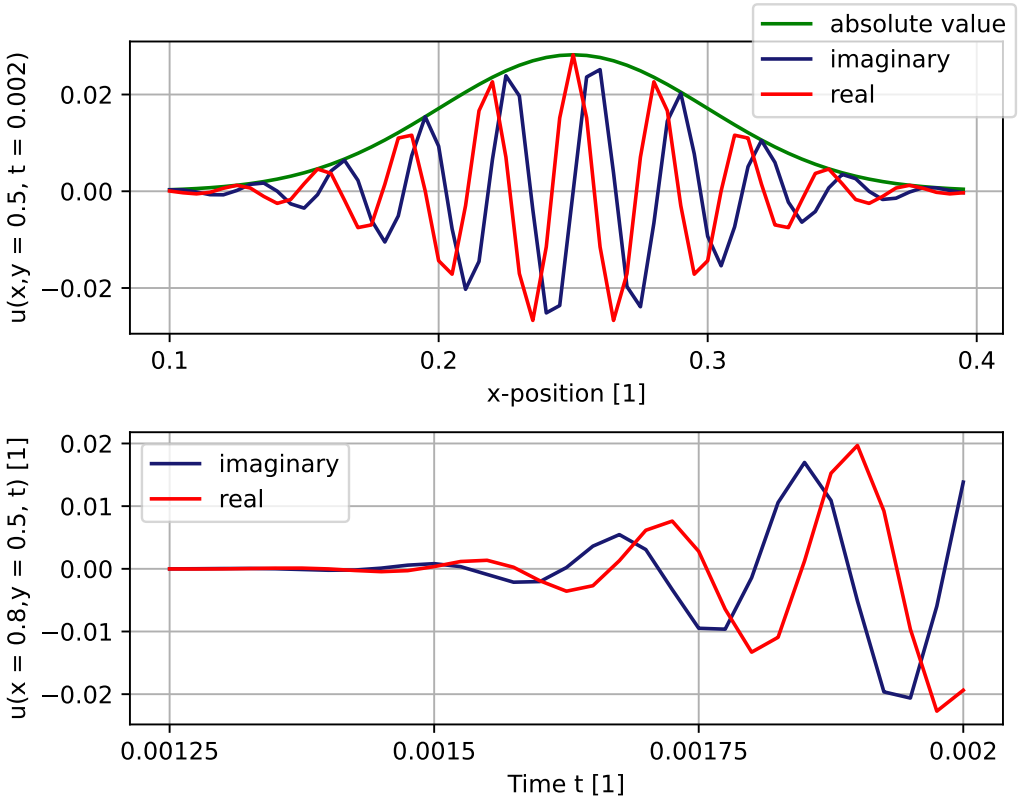

Figure 7. Plot showing the evolution of the real and imaginary part of the grid points along a single slice in the y-direction, and for a single point. The upper plot shows the evolution along $y = 0.5$ at $t = 0.002$. In addition to showing the real and imaginary parts of each grid point along the slice, it also shows the absolute value of each grid point. The bottom plot shows the time-evolution of the real and imaginary parts, for the single grid point $(0.8, 0.5)$. Both plots are from a system with two slits, and $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, $v_0 = 10^{10}$.

From the plots, in figure (4) the wave-like nature of the particle can be observed, as the probability distribution function spreads out as a classical wave. This phenomenon contradicts the Newtonian understanding of particle physics. Which would predict the particle as a solid point, acting similarly to a billiard ball. Knowing this we would expect the results of this experiment to not yield a diffraction pattern, and the wave packet to only pass through the area directly behind the slit it goes through. In reality, what we observe is that the particle produces an interference pattern, spreading out behind the slit. The observation that only one particle, described as a wave packet, yields a diffraction pattern is an example of wave-particle duality.

Observing the real, and complex components of the time evolution in figures (5, 6) the same wave-like properties are observed. With the values spreading out like a wave and creating a diffraction pattern. For an explanation, of why the diffraction pattern occurs, we turn our attention to figure (7). Here an explanation of the diffraction pattern's occurrence can be deduced. Noting that the real and complex values along the slice at $y = 0.5$, show two similar wave patterns, for the real and complex values with a phase shift to the right, for the imaginary component. This discrepancy in phase can also be observed, for a single point although here the imaginary phase shift is to the left for the real component. This causes the probability at some points to be zero as the wave functions cancel each other out, thus creating a diffraction pattern. Observing the absolute value of each grid point along the slice also yields an interesting observation. As we note the resemblance of a wave-like curve, again strengthening our notion that the wave packet has wave-like properties. Seeing as this indicates that,

the wave packet is spread out like a wave in the box.

3. *Time-evolution of probability along a single slice in the x-axis*

Finally, the time evolution of a single slice in the x-axis, at $x = 0.8$, will be investigated. Where the systems to be observed has, one, two and three slits.

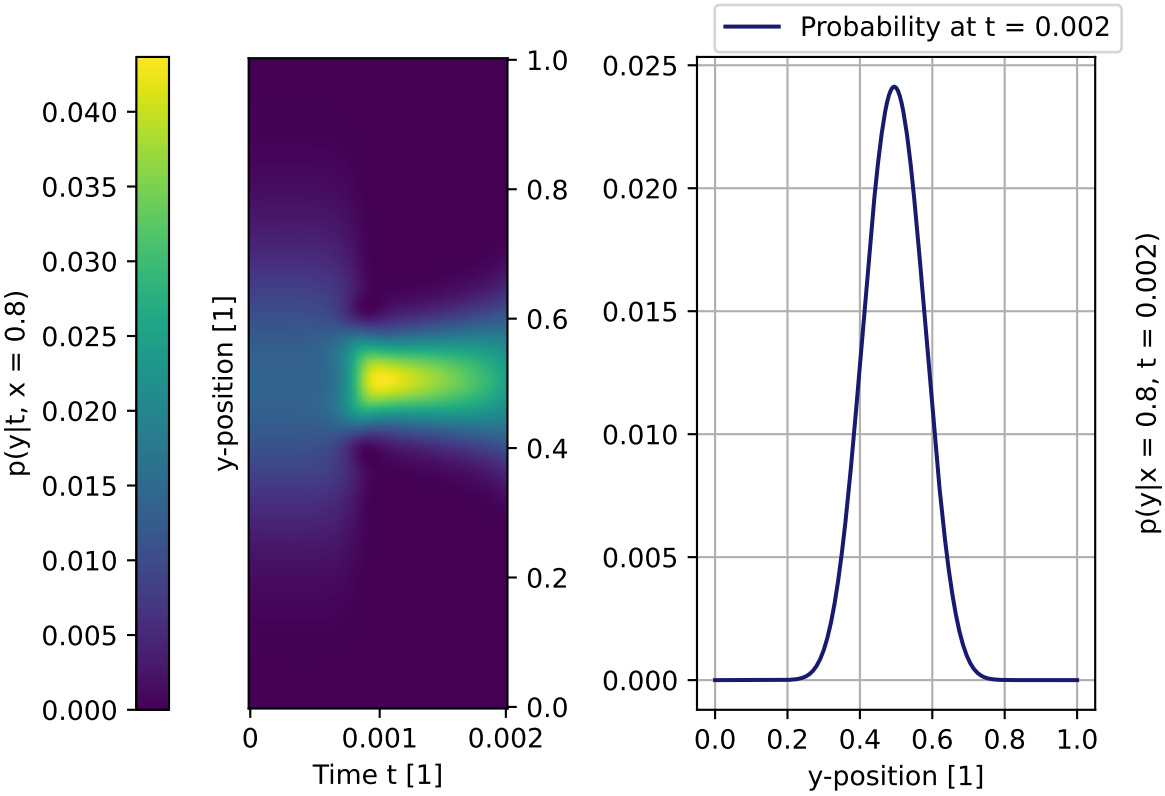

Figure 8. Plot showing the evolution of probabilities at $x = 0.8$ as a function of each time step to the right. The left plot shows the different probabilities as a function of y-position at the last time step $t = 0.002$. To make the plot more readable, the probabilities at each time step were normalized to sum to one. The parameters of the system were: $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, and $v_0 = 10^{10}$ with one slit.

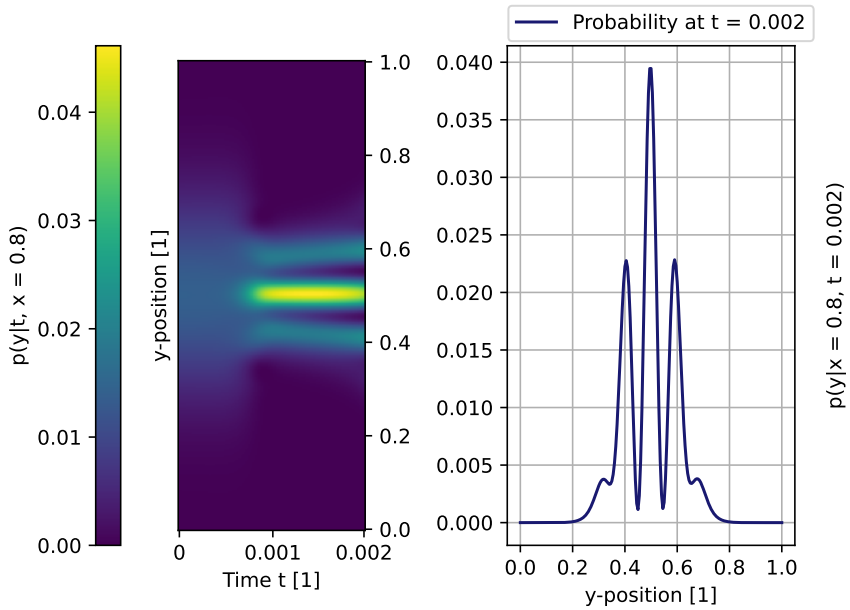

Figure 9. Plot showing the evolution of probabilities at $x = 0.8$ as a function of each time step to the right. The left plot shows the different probabilities as a function of y-position at the last time step $t = 0.002$ To make the plot more readable, the probabilities at each time step were normalized to sum to one. The parameters of the system were: $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, and $v_0 = 10^{10}$ with two slits.

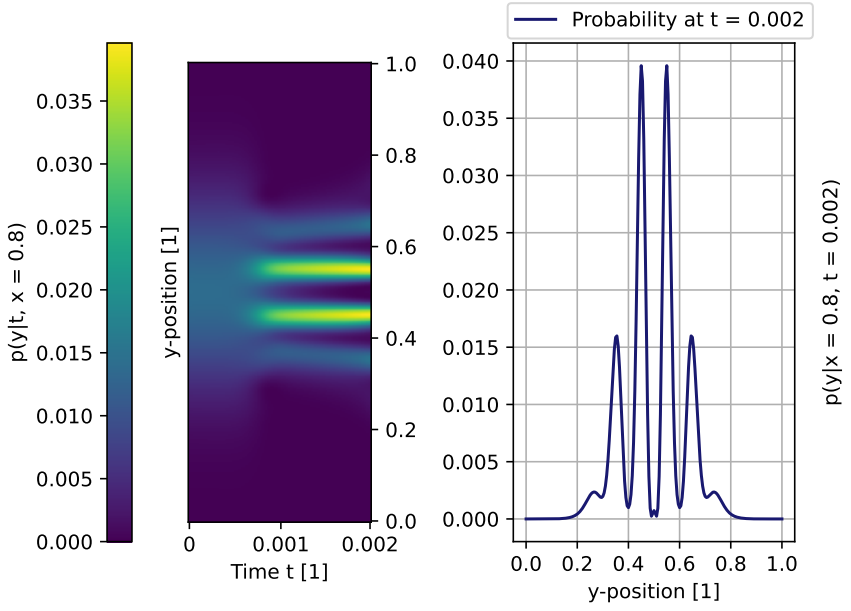

Figure 10. Plot showing the evolution of probabilities at $x = 0.8$ as a function of each time step to the right. The left plot shows the different probabilities as a function of y-position at the last time step $t = 0.002$. To make the plot more readable, the probabilities at each time step were normalized to sum to one. The parameters of the system were: $h = 0.005$, $\Delta t = 2.5 \cdot 10^{-5}$, $T = 0.002$, $x_c = 0.25$, $\sigma_x = 0.05$, $p_x = 200$, $y_c = 0.5$, $\sigma_y = 0.20$, $p_y = 0$, and $v_0 = 10^{10}$ with three slits.

By observation of figure (8) we can clearly observe that sending a particle through one slit gives only one probability peak when $t = 0.002$, and no interference pattern, which is not an unreasonable contradiction towards a classical understanding of particle behavior. In contrast figure (9) which has two slits, shows the particle creating an interference pattern with itself, thus showing the wave-like property of particles. Observing the same phenomenon when adding one more slit in figure (10).

These interference patterns can be described by classical wave theory, and are as expected. Observing the probability functions, for a single slice at $t = 0.002$, shown on the right side of figures (8,9,10). We observe the expected peaks in the interference pattern created by the particles at $t = 0.002$. Seeing as the probability of the particle's movement disperses in the expected number of directions. Each peak indicates one way the probability is going. For a better understanding of this phenomenon, animations of the wave packets movement for the systems in figures (8,9,10), can be found here: One slit, Two slits, and Three slits.

4. Run time

Running all the simulations presented in this paper took: "7 minutes and 28 seconds". Details on hardware and timing method, can be found in the repository for the source code. Although we don't have any values to compare this against, this run time seems fairly high considering the number of time steps and the size of matrices. Improving this run time, could be starting point for further research into more efficient implementations of the Crank-Nicolson method. For example investigating a numerical method for solving equation (27), such as the Jaobi rotation Algorithm, described in section 4 iv 3.

6. Conclusion

The probabilities' absolute deviation from one, for a system with no wall and a system with two slits, as shown in figures (2,3), were acceptable when compared to the machine precision of $1.0 \cdot 10^{-16}$. For the system with no wall it deviates by $1.31 \cdot 10^{-14}$ absolutely from 1 at the end of the simulation, and the system with two slits deviates by $1.22 \cdot 10^{-14}$ absolutely from one at the end of the simulation.

From figures (4, 5, 6) we observe the wavelike property from a particle when sending it through two slits. These results are as expected when comparing them to the famous double-slits experiment [4]. From the same figures, we also observe the time evolution of the particle, and how it disperses at time $t = 0.001$. Which is also an example of the wavelike properties of a particle.

From figures (9, 10) we more clearly see the diffraction patterns, which occur when the system has two or more slits. Which when compared to the system with a single-slit in figure (8). Clearly shows the phenomenon we observe in more detail in figure (7). Of, because the evolution the of real and imaginary part has a wave-like nature with different phases, the real and imaginary part along the slice canceling each other out and causing a diffraction pattern.

To make the results in the paper more precise, the time step Δt or step size h could have been lowered. Although not to much, as a too low time step or step size, would cause a larger error from floating point representation, then it would reduce the error from approximation errors. This would come at the expense of a higher computation time, but most likely yielding a more precise result with a lower deviation in the sum of probability from one. Another point that could have been improved is running more simulations, with varying amounts of slits with different x_c, y_c, σ_x, σ_y, p_x, p_y. To check if the results in the paper are consistent, or if they are a one-off occurrence.

For future research, we could study more in-depth, the wave-like patterns which occur in the real and imaginary

components of each grid point for two slits. Or we could examine the difference in result if we were to numerically solve the system of equations from the Crank-Nicolson method, for example using the method described in section 4 iv 3.

---

[1] Armadillo library for c++. https://arma.sourceforge.net/. Accessed: 2022-11-30.
[2] The bakerian lecture. experiments and calculations relative to physical optics. https://royalsocietypublishing.org/doi/pdf/10.1098/rstl.1804.0001. Accessed: 2022-11-30.
[3] Basic issues in floating point arithmetic and error analysis. https://people.eecs.berkeley.edu/~demmel/cs267/lecture21/lecture21.html. Accessed: 2022-12-11.
[4] Diffraction of electrons by a crystal of nickel. http://hermes.ffn.ub.es/luisnavarro/nuevo_maletin/Davisson_Germer_1927.pdf. Accessed: 2022-11-30.
[5] Umass chan medical school: Electron microscopy. https://www.umassmed.edu/cemf/whatisem/. Accessed: 2022-11-30.