Bachelor Thesis

Numerical Solution of the Time-Dependent 1D-Schrödinger Equation using Absorbing Boundary Conditions

Christoph Wachter

Supervisor: Assoc.-Prof. Dr. Peter Puschnig

University of Graz
Institute of Physics

September, 2017

Abstract

Using computational methods is an efficient way to solve the time-dependent Schrödinger equation. But using such methods, one generally deals with unwanted reflections at the boundaries, that necessarily constrain the computation area. To minimize these reflections absorbing boundary conditions are introduced. We solve the time-dependent Schrödinger equation with the Crank-Nicolson method and absorbing boundary conditions and determine the effectiveness of the absorbing boundary conditions, as well as calculate transmission coefficients for various shapes of potential barriers.

Contents

Chapter 1

Introduction

When first confronted with quantum mechanics, it is often difficult to visualize the physical situation. While in classical mechanics, trajectories for particles can be easily sketched and understood, even a free quantum mechanical particle is difficult to visualize. Additionally, scattering is generally treated using plane waves instead of wave packets. To describe these processes, one commonly uses the *time-independent* Schrödinger equation, which is a stark contrast to the time-dependent nature of scattering processes. For this and other reasons the plane wave approach has been criticized [1]. While solving the time-dependent Schrödinger equation analytically is difficult, and for general potentials, even impossible, numerical solutions are much easier to obtain. Moreover, using computational methods animations of Gaussian wave packets and scattering processes can be created [2] and transmission coefficients for nearly arbitrary potential barriers can be calculated.

But when solving the time-dependent Schrödinger equation numerically a certain problem, aside from the numerical accuracy of the used scheme, arises. The spatial domain has to be restricted by boundaries. These boundaries cause undesirable reflections that disturb the solution. This is especially problematic when simulating a scattering process, as the scattered wave function will reflect back at the potential barrier. While the domain can be enlarged to fix this problem, doing so either reduces accuracy or increases computation time. To solve this problem, many different methods have been developed. These include the use of a negative imaginary potential near the boundaries [3] and so called absorbing boundary conditions [4–6].

In this thesis we use absorbing boundary conditions, specifically those introduced in Ref. [6], together with the Crank-Nicolson scheme [7] to solve the time-dependent Schrödinger equation numerically with Python [8]. We will test the effectiveness of the boundary conditions using a Gaussian wave packet and determine how changing certain parameters affects the boundary conditions. Then we will use the absorbing boundary conditions to calculate the transmission coefficient of a Gaussian for a rectangular barrier, a rectangular double barrier and a smooth double barrier.

Chapter 2

Theory

In this chapter, a brief revision of the necessary quantum mechanics is given, and absorbing boundary conditions for the time-dependant Schrödinger are derived.

2.1 Time-dependent Schrödinger Equation

The time-evolution in a one-dimensional quantum mechanical system is determined by the time-dependent Schrödinger equation (TDSE)

$$i\hbar\frac{\partial}{\partial t}\psi(x,t) = \hat{H}\psi(x,t) \tag{2.1}$$

with the Hamilton operator

$$\hat{H} = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x,t). \tag{2.2}$$

Here, $\hbar$ is the reduced Planck constant, $i = \sqrt{-1}$, m the mass of the particle, V the potential and ψ the wave function. If $\hat{H}$ is not time-dependent, the formal solution of the TDSE is given by

$$\psi(x,t) = \hat{U}(t)\psi(x,0) = e^{-\frac{i\hat{H}t}{\hbar}}\psi(x,0), \tag{2.3}$$

where $\hat{U}(t)$ is called the propagator. Since $\hat{U}$ is a unitary operator[1], the time-evolution operator $\hat{U}$ conserves the norm of the wave function

$$|\psi(x,t)|^2 = |\psi(x,0)|^2. \tag{2.4}$$

Note that the norm squared of the wave function, $|\psi(x,t)|^2$, describes the probability density of the position of the particle.

[1] $\hat{U}\hat{U}^\dagger = \mathbb{1}$

2.2 Absorbing Boundary Conditions

When using computational methods to solve the TDSE, the spatial domain is necessarily restricted by the boundaries of the simulation domain. The boundaries generally cause unwanted reflections, which can disrupt the results of the computer simulation. However, simply enlarging the computation area to alleviate the problem is inefficient. Therefore, here we construct absorbing boundary conditions (ABCs) to minimize reflections at the borders. The derivations in this section follow Ref. [6].

First we consider the elementary solution of the TDSE for particles with definite energy, higher than the potential V:

$$\psi(x,t) = e^{i(kx-\omega t)} \tag{2.5}$$

By inserting the plane wave ansatz (2.5) back into the TSDE, we obtain the dispersion relation

$$\hbar^2 k^2 = 2m(\hbar\omega - V), \tag{2.6}$$

or in terms of ω

$$\omega(k) = \frac{\hbar k^2}{2m} + \frac{V}{\hbar}. \tag{2.7}$$

Comparison of Eq. (2.7) with the TDSE leads to the the following correspondence:

$$k \longleftrightarrow -i\frac{\partial}{\partial x}, \qquad \omega \longleftrightarrow i\frac{\partial}{\partial t} \tag{2.8}$$

Next, we will assume that the potential V is either constant, or a slowly varying function, near the boundaries. We can then calculate the group velocity from the dispersion relation, (2.6):

$$v_g = \frac{\partial\omega(k)}{\partial k} = \frac{\hbar k}{m}. \tag{2.9}$$

A positive group velocity represents a wave travelling to the right, while a negative one represents a wave travelling to the left. If we only want waves leaving the domain at the left boundary, we need to restrict the group velocity to be positive. In mathematical terms this means

$$\frac{\hbar k}{m} = \left|\frac{\hbar k}{m}\right|. \tag{2.10}$$

For the right boundary, k needs to be replaced with $-k$. But this boundary condition is not rational, due to the absolute value function, and therefore cannot be converted to a partial differential equation through (2.8). Thus we need to use an approximation. We choose

$$\frac{\hbar k}{m} \equiv q, \tag{2.11}$$

where q is positive and real. Using the correspondence (2.8) yields the following differential equation:

$$\left(i\frac{\partial}{\partial x}+\frac{mq}{\hbar}\right)\psi=0 \tag{2.12}$$

If this differential equation is satisfied on the boundary, then no waves with group velocity q are reflected, therefore waves with this group velocity are essentially absorbed. Since waves, in general, consist of more than one component with different group velocities, the operator in (2.12) can be generalized by considering more group velocities, which are absorbed. This results in:

$$\prod_{l=1}^{p}\left(i\frac{\partial}{\partial x}+\frac{mq_l}{\hbar}\right)\psi=0. \tag{2.13}$$

The effect of (2.13) at the boundary differs depending on the choice of q_l. If $q_k \neq q_l$ for $k \neq l$, then p different group velocities, of the computed wave solution, are absorbed to the first order. If $q_k = q_l$ for $k \neq l$, then the group velocity q_k is absorbed to pth order.

It should be noted that one can also derive ABCs for the TDSE by directly using the dispersion relation [4,5]. Nonetheless, it can be shown [6] that those ABCs are special instances of Eq. (2.13).

Let us now derive some more specific ABCs from Eq. (2.13), starting with $p = 2$. Here we use (2.8) again

$$\left(\pm k+\frac{mq_1}{\hbar}\right)\left(\pm k+\frac{mq_2}{\hbar}\right)=0. \tag{2.14}$$

In this equation the top sign applies at the right and the bottom sign at the left boundary. If we multiply out (2.14) and use (2.6) to substitute for k^2 we obtain

$$\mp\hbar k=\frac{2}{q_1+q_2}(\hbar\omega-V)+\frac{mq_1q_1}{q_1+q_2}. \tag{2.15}$$

Continuing this process with $p = 3$

$$\left(\pm k+\frac{mq_1}{\hbar}\right)\left(\pm k+\frac{mq_2}{\hbar}\right)\left(\pm k+\frac{mq_3}{\hbar}\right)=0, \tag{2.16}$$

leads to

$$\mp\hbar k=\frac{2mh_1(\hbar\omega-V)+h_3}{2m(\hbar\omega-V)+h_2}, \tag{2.17}$$

where

$$\begin{aligned} h_1 &= m(q_1 + q_2 + q_3), \\ h_2 &= m^2 q_1 q_2 q_3 \left(\frac{1}{q_1} + \frac{1}{q_2} + \frac{1}{q_3} \right), \\ h_3 &= m^3 q_1 q_2 q_3. \end{aligned}$$

Lastly, setting $p = 4$, we obtain the following equation

$$4m^2(\hbar\omega - V)^2 + 2m(\pm g_1 \hbar k + g_2)(\hbar\omega - V) \pm g_3 \hbar k + g_4 = 0, \tag{2.18}$$

where

$$\begin{aligned} g_1 &= m(q_1 + q_2 + q_3 + q_4), \\ g_2 &= m^2(q_1q_2 + q_1q_3 + q_1q_4 + q_2q_3 + q_2q_4 + q_3q_4), \\ g_3 &= m^3 q_1 q_2 q_3 q_4 \left(\frac{1}{q_1} + \frac{1}{q_2} + \frac{1}{q_3} + \frac{1}{q_4} \right), \\ g_4 &= m^4 q_1 q_2 q_3 q_4. \end{aligned}$$

This process can be repeated to derive ABCs of higher order in p.

Chapter 3

Computational Methods

In this chapter, the computational methods for solving the time-dependent Schrödinger equation, as well as the numerical implementation of the ABC derived in Section 2.2 are given. Each section is followed by an implementation of the discussed schemes in Python[1].

3.1 Crank-Nicolson Method

In order to treat the TDSE numerically, we represent $\psi(x,t)$ by its values at a set of grid-points. For the spatial domain we choose $x = x_j = x_0 + j\Delta x$, with $j \in [0, J]$, where x_0 represents the left boundary and Δx is the grid spacing. Similarly, the time domain has the range $t = t_n = n\Delta t$, with $n \in [0, N]$. The values of the wave function at the grid points will be abbreviated by

$$\psi(x_j, t_n) \equiv \psi_j^n. \tag{3.1}$$

After applying this discretization to the time-evolution given by Eq.(2.3), we obtain

$$\psi_j^{n+1} = e^{-i\hat{H}\Delta t}\psi_j^n = \hat{U}(\Delta t)\psi_j^n. \tag{3.2}$$

It is important to note that, since $\hat{U}$ is unitary and preserves the norm of the wave function, any approximations of $\hat{U}(\Delta t)$ must be unitary as well. Therefore simply expanding $\hat{U}$

$$\psi_j^{n+1} = (1 - i\hat{H}\Delta t)\psi_j^n \tag{3.3}$$

does not provide the desired result.

[1]In all the derivations and calculations in Chapters 3 and 4 $\hbar = 1$ and $m = 0.5$, unless stated otherwise.

To derive a unitary approximation, we start by splitting $\hat{U}(\Delta t)$ as follows

$$\psi_j^{n+1} = e^{-\frac{i\hat{H}\Delta t}{2}} e^{-\frac{i\hat{H}\Delta t}{2}} \psi_j^n, \tag{3.4}$$

and multiply the equation from the left by $\hat{U}^\dagger(\frac{\Delta t}{2})$ to obtain

$$e^{\frac{i\hat{H}\Delta t}{2}} \psi_j^{n+1} = e^{-\frac{i\hat{H}\Delta t}{2}} \psi_j^n, \tag{3.5}$$

or by expanding the exponential functions into a Taylor series and cutting off after the second term

$$\Big(1 + \frac{i\hat{H}\Delta t}{2}\Big)\psi_j^{n+1} = \Big(1 - \frac{i\hat{H}\Delta t}{2}\Big)\psi_j^n. \tag{3.6}$$

Thus we have found a unitary method to approximate $\hat{U}$ known as Cayley's form

$$\hat{U}(\Delta t) = e^{-i\hat{H}\Delta t} \simeq \frac{(1 - \frac{i\hat{H}\Delta t}{2})}{(1 + \frac{i\hat{H}\Delta t}{2})}. \tag{3.7}$$

By using the finite-difference representation [7] for the x derivative and $V_j = V(x_j)$, Eq. (3.6) reads

$$\psi_j^{n+1} - \frac{i\Delta t}{2}\left[\frac{\psi_{j+1}^{n+1} - 2\psi_j^{n+1} + \psi_{j-1}^{n+1}}{(\Delta x)^2} - V_j\psi_j^{n+1}\right] = \psi_j^n + \frac{i\Delta t}{2}\left[\frac{\psi_{j+1}^n - 2\psi_j^n + \psi_{j-1}^n}{(\Delta x)^2} - V_j\psi_j^n\right]. \tag{3.8}$$

It should be noted that (3.8) can also be obtained by applying the Crank-Nicolson method to the TDSE (2.1).

Finally, following [9], we introduce the vector $\boldsymbol{\psi}^n = (\psi_0^n, \cdots, \psi_j^n, \cdots, \psi_J^n)$, so we can write (3.8) as a matrix equation:

$$\mathcal{U}_1\boldsymbol{\psi}^{n+1} = \mathcal{U}_2\boldsymbol{\psi}^n. \tag{3.9}$$

Here $\mathcal{U}_1$ and $\mathcal{U}_2$ represent two tridiagonal $(J+1)\times(J+1)$ matrices

$$\mathcal{U}_1 = \begin{pmatrix} \xi_0 & -\alpha & & & \\ -\alpha & \xi_1 & -\alpha & & \\ & \ddots & \ddots & \ddots & \\ & & -\alpha & \xi_{J-1} & -\alpha \\ & & & -\alpha & \xi_J \end{pmatrix}, \quad \mathcal{U}_2 = \begin{pmatrix} \gamma_0 & \alpha & & & \\ \alpha & \gamma_1 & \alpha & & \\ & \ddots & \ddots & \ddots & \\ & & \alpha & \gamma_{J-1} & \alpha \\ & & & \alpha & \gamma_J \end{pmatrix} \tag{3.10}$$

with

$$\alpha = \frac{i\Delta t}{2\Delta x^2}, \ \xi_j = 1 + \frac{i\Delta t}{2}\Big(\frac{2}{\Delta x^2} + V_j\Big), \ \gamma_j = 1 - \frac{i\Delta t}{2}\Big(\frac{2}{\Delta x^2} + V_j\Big). \tag{3.11}$$

Python Implementation

Since the matrices (3.10) are tridiagonal, it is more efficient to only store the non-zero entries. This can be accomplished through the usage of Pythons `scipy.sparse` package. A code example for setting up $\mathcal{U}_1$ is given below[2].

```
o     = np.ones((J),complex)
alp   = (1j)*dt/(2*dx**2)*o                # alpha from (3.11)
xi    = o + 1j*dt/2*(2/(dx**2)*o + V)      # xi from (3.11)
diags = np.array([-1,0,+1])          # positions of the vectors in the matrix
vecs1 = np.array([-alp,xi,-alp])
U1    = scipy.sparse.spdiags(vecs1,diags,J,J) # create tridiagonal sparse matrix
U1    = U1.tocsc() # convert to different sparse format needed for further calculation
```

With an inital wave function ψ^0 to start with, the linear system of equations (3.9) can be solved for each time step using LU-Decompositon.

```
PSI = np.zeros((J,N),complex)  # J times N array to store all solutions
PSI[:,0] = psi0                # psi(x,0)
LU = scipy.sparse.linalg.splu(U1)  # compute LU-decomposition of U1
for n in range(0,N - 1):       # loop over time-steps
    b = U2.dot(PSI[:,n])       # right hand side of eq. (3.9)
    PSI[:,n + 1] = LU.solve(b) # solve system of equations for each time step
```

Alternatively an analytical inversion of $\mathcal{U}_1$ as suggested in [9] can be used to easily implement the Crank-Nicolson method into other computing languages and without making use of packages.

3.2 Implementation of Absorbing Boundary Conditions

The ABCs from Section 2.2 must be discretized before we can implement them into the Crank-Nicholson method.

[2] `1j` represents $i = \sqrt{-1}$.

p=2 ABC

Applying the correspondence relation to Eq. (2.15) we obtain

$$\pm i\frac{\partial \psi}{\partial x} - ic_1\frac{\partial \psi}{\partial t} + (c_1 V - c_2)\psi = 0, \tag{3.12}$$

where

$$c_1 = \frac{2}{q_1 + q_2}, \quad c_2 = \frac{q_1 q_2}{2(q_1 + q_2)}. \tag{3.13}$$

The top sign on the first term refers to the boundary condition applied to the $x_j = x_0$ boundary and the bottom sign refers to the $x_j = x_J$ boundary. This sign convention will be used throughout this section. We will use the following finite-difference discretizations [6] to approximate the derivatives in (3.12):

$$\psi \approx \frac{1}{4}\big(\psi_{j\pm1}^{n+1} + \psi_j^{n+1} + \psi_{j\pm1}^{n} + \psi_j^{n}\big), \tag{3.14}$$

$$\frac{\partial \psi}{\partial x} \approx \pm\frac{1}{2\Delta x}\big(\psi_{j\pm1}^{n+1} - \psi_j^{n+1} + \psi_{j\pm1}^{n} - \psi_j^{n}\big), \tag{3.15}$$

$$\frac{\partial \psi}{\partial t} \approx \frac{1}{2\Delta t}\big(\psi_{j\pm1}^{n+1} + \psi_j^{n+1} - \psi_{j\pm1}^{n} - \psi_j^{n}\big). \tag{3.16}$$

Applying these discretizations to (3.12) results in the following ABC:

$$\zeta_1\psi_{(0,J)}^{n+1} + \zeta_2\psi_{(1,J-1)}^{n+1} = \zeta_3\psi_{(0,J)}^{n} + \zeta_4\psi_{(1,J-1)}^{n}, \tag{3.17}$$

where

$$\begin{aligned}
\zeta_1 &= \left(-\frac{i}{2\Delta x} - \frac{ic_1}{2\Delta t} + \frac{(c_1 V_{(0,J)} - c_2)}{4}\right),\\
\zeta_2 &= \left(\frac{i}{2\Delta x} - \frac{ic_1}{2\Delta t} + \frac{(c_1 V_{(0,J)} - c_2)}{4}\right),\\
\zeta_3 &= \left(\frac{i}{2\Delta x} - \frac{ic_1}{2\Delta t} - \frac{(c_1 V_{(0,J)} - c_2)}{4}\right),\\
\zeta_4 &= \left(-\frac{i}{2\Delta x} - \frac{ic_1}{2\Delta t} - \frac{(c_1 V_{(0,J)} - c_2)}{4}\right).
\end{aligned} \tag{3.18}$$

Implementing Eq. (3.17) into the Crank-Nicolson scheme is straightforward, we simply modify the matrices (3.10) as follows

$$\mathcal{U}_1 = \begin{pmatrix} \zeta_1 & \zeta_2 & & & \\ -\alpha & \xi_1 & -\alpha & & \\ & \ddots & \ddots & \ddots & \\ & & -\alpha & \xi_{J-1} & -\alpha \\ & & & \zeta_2 & \zeta_1 \end{pmatrix}, \quad \mathcal{U}_2 = \begin{pmatrix} \zeta_3 & \zeta_4 & & & \\ \alpha & \gamma_1 & \alpha & & \\ & \ddots & \ddots & \ddots & \\ & & \alpha & \gamma_{J-1} & \alpha \\ & & & \zeta_4 & \zeta_3 \end{pmatrix}. \tag{3.19}$$

p=3 ABC

Repeating this process with Eq. (2.17) leads to

$$\pm i(h_2 - V)\frac{\partial \psi}{\partial x} \mp \frac{\partial^2 \psi}{\partial t \partial x} - ih_1\frac{\partial \psi}{\partial t} - (h_3 - h_1 V)\psi = 0, \tag{3.20}$$

where the h_i are defined in Section 2.2. Eq. (3.20) contains an additional mixed derivative, which we will approximate through [6]:

$$\frac{\partial^2 \psi}{\partial t \partial x} \approx \pm\frac{1}{\Delta t \Delta x}(\psi_{j\pm1}^{n+1} - \psi_j^{n+1} - \psi_{j\pm1}^{n} + \psi_j^n). \tag{3.21}$$

This discretization along with the ones given in (3.14) to (3.16) applied to (3.20) yields

$$\zeta_1^{p=3}\psi_{(0,J)}^{n+1} + \zeta_2^{p=3}\psi_{(1,J-1)}^{n+1} = \zeta_3^{p=3}\psi_{(0,J)}^{n} + \zeta_4^{p=3}\psi_{(1,J-1)}^{n}, \tag{3.22}$$

where

$$\begin{aligned}
\zeta_1^{p=3} &= \left(-\frac{i(h_2 - V_{(0,J)})}{2\Delta x} + \frac{1}{\Delta t \Delta x} - \frac{ih_1}{2\Delta t} - \frac{(h_3 - h_2 V_{(0,J)})}{4}\right),\\
\zeta_2^{p=3} &= \left(\frac{i(h_2 - V_{(0,J)})}{2\Delta x} - \frac{1}{\Delta t \Delta x} - \frac{ih_1}{2\Delta t} + \frac{(h_3 - h_2 V_{(0,J)})}{4}\right),\\
\zeta_3^{p=3} &= \left(\frac{i(h_2 - V_{(0,J)})}{2\Delta x} + \frac{1}{\Delta t \Delta x} - \frac{ih_1}{2\Delta t} + \frac{(h_3 - h_2 V_{(0,J)})}{4}\right),\\
\zeta_4^{p=3} &= \left(-\frac{i(h_2 - V_{(0,J)})}{2\Delta x} - \frac{1}{\Delta t \Delta x} - \frac{ih_1}{2\Delta t} + \frac{(h_3 - h_2 V_{(0,J)})}{4}\right).
\end{aligned} \tag{3.23}$$

p=4 ABC

Rewriting Eq. (2.18) we obtain

$$p_1\frac{\partial^2 \psi}{\partial t^2} \pm p_2\frac{\partial^2 \psi}{\partial t \partial x} + p_3\frac{\partial \psi}{\partial t} \pm p_4\frac{\partial \psi}{\partial x} + p_5\psi = 0, \tag{3.24}$$

where $p_1 = -1$, $p_2 = g_1$, $p_3 = i(g_2 - 2V_{(0,J)})$, $p_4 = i(g_1 V_{(0,J)} - g_3)$, $p_5 = (V_{(0,J)})^2 - g_2 V_{(0,J)} + g_4$ and the g_i are defined in section 2.2. To discretize Eq. (3.24) we use the following approximation for the second order time derivative

$$\frac{\partial^2 \psi}{\partial t^2} \approx \frac{1}{2\Delta t^2}(\psi_{j+1}^{n-1} - 2\psi_{j+1}^{n} + \psi_{j+1}^{n+1} + \psi_j^{n-1} - 2\psi_j^n + \psi_j^{n+1}), \tag{3.25}$$

as well as the discretizations given in (3.14) to (3.16) and (3.21). Using these discretizations in (3.24) yields

$$\begin{aligned}\zeta_1^{p=4}\psi_{(0,J)}^{n+1} + \zeta_2^{p=4}\psi_{(1,J-1)}^{n+1} = \\ \zeta_3^{p=4}\psi_{(0,J)}^{n} + \zeta_4^{p=4}\psi_{(1,J-1)}^{n} + \zeta_5^{p=4}\psi_{(0,J)}^{n-1} + \zeta_6^{p=4}\psi_{(1,J-1)}^{n-1},\end{aligned} \tag{3.26}$$

where

$$\begin{aligned}
\zeta_1^{p=4} &= \left(\frac{p_1}{2\Delta t^2} - \frac{p_2}{\Delta t\Delta x} + \frac{p_3}{2\Delta t} - \frac{p_4}{2\Delta x} + \frac{p_5}{4}\right),\\
\zeta_2^{p=4} &= \left(\frac{p_1}{2\Delta t^2} + \frac{p_2}{\Delta t\Delta x} + \frac{p_3}{2\Delta t} + \frac{p_4}{2\Delta x} + \frac{p_5}{4}\right),\\
\zeta_3^{p=4} &= \left(\frac{p_1}{\Delta t^2} - \frac{p_2}{\Delta t\Delta x} + \frac{p_3}{2\Delta t} + \frac{p_4}{2\Delta x} - \frac{p_5}{4}\right),\\
\zeta_4^{p=4} &= \left(\frac{p_1}{\Delta t^2} + \frac{p_2}{\Delta t\Delta x} + \frac{p_3}{2\Delta t} - \frac{p_4}{2\Delta x} - \frac{p_5}{4}\right),\\
\zeta_5^{p=4} &= \left(-\frac{p_1}{2\Delta t^2}\right),\\
\zeta_6^{p=4} &= \left(-\frac{p_1}{2\Delta t^2}\right).
\end{aligned} \tag{3.27}$$

Note that Eq. (3.26) contains an additional time level, thus Eq. (3.9) must be modified

$$\mathcal{U}_1\boldsymbol{\psi}^{n+1} = \mathcal{U}_2\boldsymbol{\psi}^{n} + \mathcal{Z}\boldsymbol{\psi}^{n-1} \tag{3.28}$$

where

$$\mathcal{Z} = \begin{pmatrix} \zeta_5 & \zeta_6 & & & \\ & 0 & & & \\ & & \ddots & & \\ & & & 0 & \\ & & & \zeta_6 & \zeta_5 \end{pmatrix}. \tag{3.29}$$

Python Implementation

To apply the ABCs, the sparse matrices used for the Crank-Nicolson method must be modified as given in (3.19). This can be done as follows

```
zeta = np.array([0,zeta1,zeta2,zeta3,zeta4])
# array containing zeta values from either (3.18) or (3.23)
o     = np.ones((J),complex)
```

```
alp   = (1j)*dt/(2*dx**2)*o             # alpha from (3.11)
xi    = o + 1j*dt/2*(2/(dx**2)*o + V)   # xi from (3.11)
xi[0]  = zeta[1] ; xi[J-1]   = zeta[1] # insert zeta1 in the diagonal
up     =  -alp   ; dn      = -alp      # off-diagonal elements
up[1]  = zeta[2] ; dn[J-2]= zeta[2]    # insert zeta2
vecs  = np.array([dn,xi,up])
diags = np.array([-1,0,+1])
U1    = sparse.spdiags(vecs,diags,J,J)
U1    = U1.tocsc()
```

The $p = 4$ ABC needs two previous time levels to calculate the subsequent time level, thus we need to alter the scheme discussed in the previous section.

```
LU     = scipy.sparse.linalg.splu(U1)
LW     = scipy.sparse.linalg.splu(W1)
for n in range(0,N - 1):
    if n == 0:
        b = W2.dot(PSI[:,n])
        PSI[:,n+1] = LW.solve(b)
    else:
        b = U2.dot(PSI[:,n]) + Z.dot(PSI[:,n-1])
        PSI[:,n+1] = LU.solve(b)
```

Here `W1,W2` are the standard Crank-Nicolson matrices from Eq. (3.10). The reason for this will be explained in Section 4.1.

Chapter 4

Results

In this chapter, first some general properties of the ABCs are tested and then the ABCs are applied to one dimensional potential scattering in order to determine transmission coefficients.

4.1 Reflection Properties of Absorbing Boundary Conditions

To evaluate the effectiveness of the ABCs, we consider a Gaussian wave packet as our initial condition

$$\psi(x,0) = \sqrt[4]{\frac{1}{\sigma_0^2 \pi}} \exp\left[i k_0 x - \frac{(x-\xi_0)^2}{2\sigma_0^2} \right]. \tag{4.1}$$

This wave packet is centred at ξ_0, has an average momentum of k_0 and a initial width of σ_0. To prevent any disturbances inside the domain the potential V is set to zero. For calculations throughout this section, the parameters listed in Table 4.1 will be used, unless stated otherwise.

Table 4.1

Parameters	J	x_0	x_J	Δt	ξ_0
Values	1024	-10	10	0.0001	0

The grid spacing Δx can be calculated through $\Delta x = (x_J - x_0)/J$ and the other parameters k_0 and σ_0 will be specified in the corresponding figure captions or in the text. For the ABCs we choose $q_l = q_0$, where q_0 is positive and non-zero. If q_0 would be chosen negative, rather than absorbing the wave packet, the boundary condition would increase the norm of the wave packet. A comparison to demonstrate the difference between the Crank-Nicolson method with and without ABCs using (4.1) as initial condition can be seen in Fig. 4.1 below.

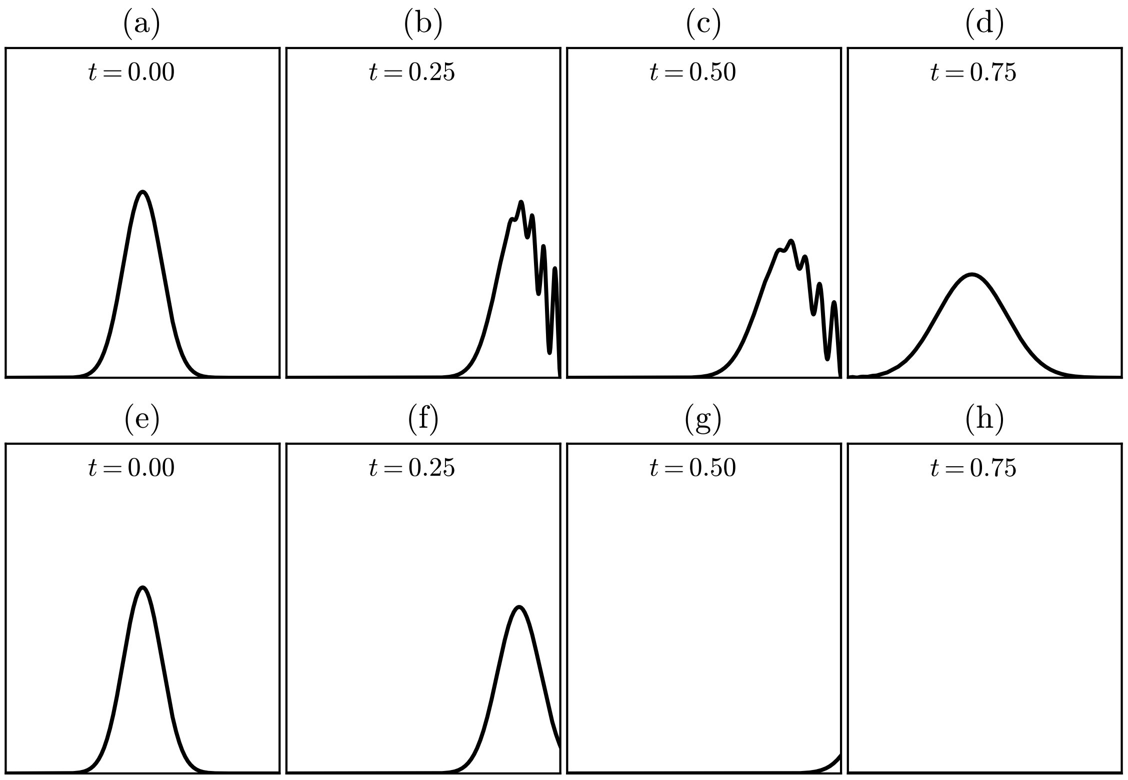


Figure 4.1: Comparison between the standard Crank-Nicolson scheme (a)-(d) and the modified scheme with ABCs (e)-(h). Parameters used: $x_0 = 0$, $x_J = 10$, $\xi_0 = 5$, $k_0 = 7$, $\sigma_0 = 1.0$ and $q_0 = 2k_0$.

We will compare the relative reflective properties of the three different ABCs with respect to each other. To do this we calculate the reflection ratio R at time t_n through

$$R = \frac{\sum_{j=0}^{J} |\psi_j^n|^2 \Delta x}{\sum_{j=0}^{J} |\psi_j^0|^2 \Delta x}. \tag{4.2}$$

In this context, if $R = 1$, the wave packet is completely reflected, e.g. no ABCs are used like in Fig. 4.1 (a)-(d), and if $R = 0$, the wave packet is completely absorbed, representing perfect ABCs. Thus we aim for a reflection ratio as close to zero as possible. According to Eq. (2.13), the following choice of q_0 should give us the best results:

$$q_0 = \frac{\hbar k_0}{m} = 2k_0 \tag{4.3}$$

Nonetheless, we will start our reflection test by calculating R as a function of $q_0/|k_0|$ for several fixed values of k_0. Note that, since the ABCs are not perfect, a small fraction of the wave packet is still reflected. We are interested in the reflection ratio after the wave packet is absorbed at the x_J-boundary, but before the reflected wave packet reaches the x_0-boundary and is absorbed again, further decreasing R. Hence we need to evaluate R at a time t_N, when the reflected wave packet is near the centre of the domain. We calculate t_N through the (average) velocity of the wave packet $v = 2k_0$ and the distance travelled s:

$$t_N = \frac{s}{v} \tag{4.4}$$

We take $s = 24$, giving the ABCs enough time to absorb the wave packet.

Reflection ratios for all three orders of ABCs and for different values of k_0 are shown in Fig. 4.2 on page 17 and Fig. 4.3 on page 18. From these plots we can deduce the following:

- For all three orders of absorbing boundary conditions R has a minimum. We will from now on refer to the q_0 where R is minimal as q_{min}.
- For $k_0 = 5$, q_{min} is close to $2k_0$, except for $p = 4$, as predicted in Eq. (4.3), but for higher values of k_0, q_{min} changes. This change is more drastic for $p = 3$ and $p = 4$.
- For higher values of k_0 the reflection ratios for all three ABCs are about equal at $2k_0$.

The values of q_{min} and the corresponding reflection ratios from Fig. 4.2 and Fig. 4.3 are listed in Table 4.2 on page 17.

As stated above q_{min} shifts with increasing k_0. This can be attributed to numerical accuracy. We consider the average wavelength of a Gaussian wave packet $\lambda_0 = 2\pi/k_0$ and calculate the grid-points per wavelength

$$\frac{\lambda_0}{\Delta x} = \frac{2\pi}{k_0 \Delta x}. \tag{4.5}$$

Increasing k_0 result in less points per wavelength, thus we need to in turn decrease the grid spacing to keep the ratio the same. We repeat the previous simulation with $k_0 = 15$ and $p = 2$, but use different values for J (and therefore Δx). The results are plotted in Fig. 4.4. q_{min} changes with decreasing grid spacing and approaches $2k_0$. Additionally, the minimal reflection ratio decreases as well. Similar results can be obtained for $p = 3$. It is important to note that for Fig. 4.4 we chose $\Delta t = 0.001$, to reduce computation time. This, however, does not influence q_{min} or $R(q_{min})$ in any significant way.

The $p = 4$ ABC has some different properties in comparison to the $p = 2$ and $p = 3$ ABCs. Since the first step in the algorithm is different from the rest, placing the initial wave packet near the boundary decreases the effectiveness of the ABC. Using the Crank-Nicolson matrices (3.10) instead of the already modified matrices (3.19) for the first step improves this somewhat, but does not negate it. Thus $p = 2$ or $p = 3$ ABCs should be used in these situations. Additionally $p = 4$ depends more on the time step Δt. Changing Δt from 0.0001 to 0.001 increases the reflection ratio about an order of magnitude.

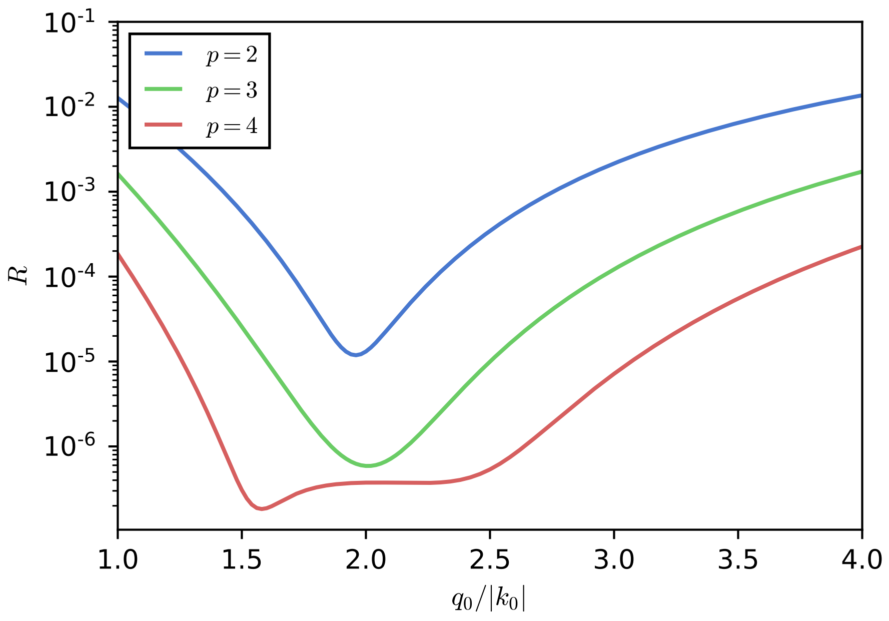
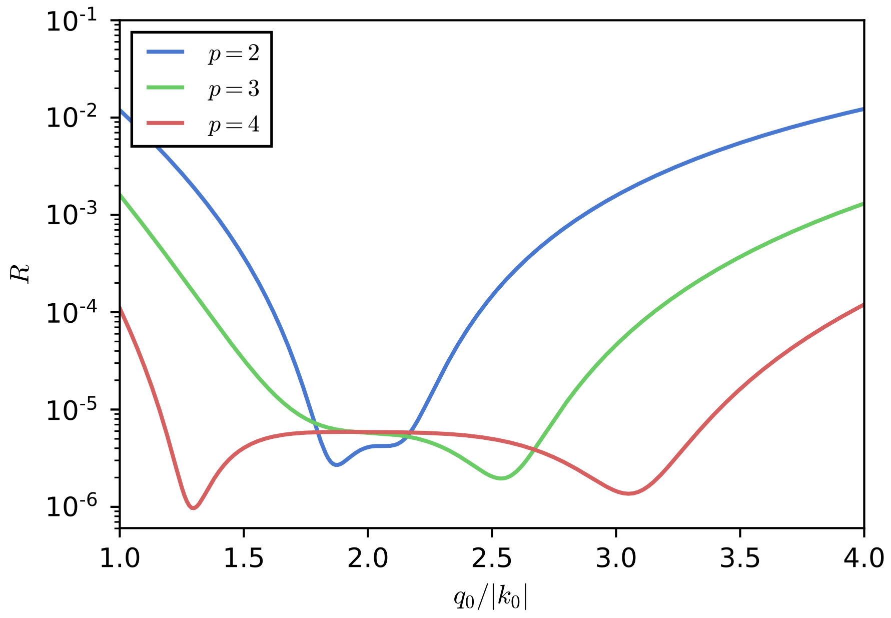


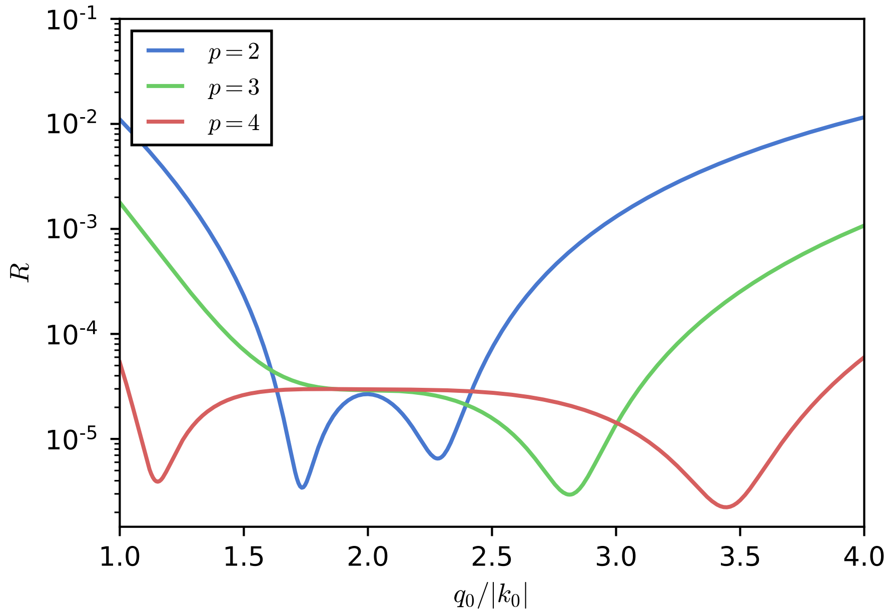
Figure 4.2: Reflection ratio as a function of $q_0/|k_0|$ evaluated at t_N. Parameters used: $k_0 = 5, t_N = 2.4$ and $\sigma_0 = 1.5$. q_0 starts at k_0 and is increased up to $4k_0$ with step size $\Delta q_0 = 0.1$.

Table 4.2

k_0	Order	q_{min}	$R(q_{min})$	$R(2k_0)$
5	$p = 2$	$1.960 \cdot \lvert k_0 \rvert$	$1.185 \cdot 10^{-5}$	$1.312 \cdot 10^{-5}$
	$p = 3$	$2.000 \cdot \lvert k_0 \rvert$	$5.893 \cdot 10^{-7}$	$5.893 \cdot 10^{-7}$
	$p = 4$	$1.580 \cdot \lvert k_0 \rvert$	$1.831 \cdot 10^{-7}$	$3.749 \cdot 10^{-7}$
10	$p = 2$	$1.870 \cdot \lvert k_0 \rvert$	$2.696 \cdot 10^{-6}$	$4.106 \cdot 10^{-6}$
	$p = 3$	$2.540 \cdot \lvert k_0 \rvert$	$1.957 \cdot 10^{-6}$	$5.780 \cdot 10^{-6}$
	$p = 4$	$1.300 \cdot \lvert k_0 \rvert$	$9.698 \cdot 10^{-7}$	$5.884 \cdot 10^{-6}$
15	$p = 2$	$1.733 \cdot \lvert k_0 \rvert$	$3.432 \cdot 10^{-6}$	$2.672 \cdot 10^{-5}$
	$p = 3$	$2.813 \cdot \lvert k_0 \rvert$	$2.949 \cdot 10^{-6}$	$2.915 \cdot 10^{-5}$
	$p = 4$	$3.440 \cdot \lvert k_0 \rvert$	$2.235 \cdot 10^{-6}$	$2.972 \cdot 10^{-5}$



(a) $k_0 = 10, t_N = 1.2$



(b) $k_0 = 15, t_N = 0.8$

Figure 4.3: Reflection ratio as a function of $q_0/|k_0|$ evaluated at t_N. Parameters used: $\sigma_0 = 1.5$. q_0 starts at k_0 and is increased up to $4k_0$ with step size $\Delta q_0 = 0.1$.

.

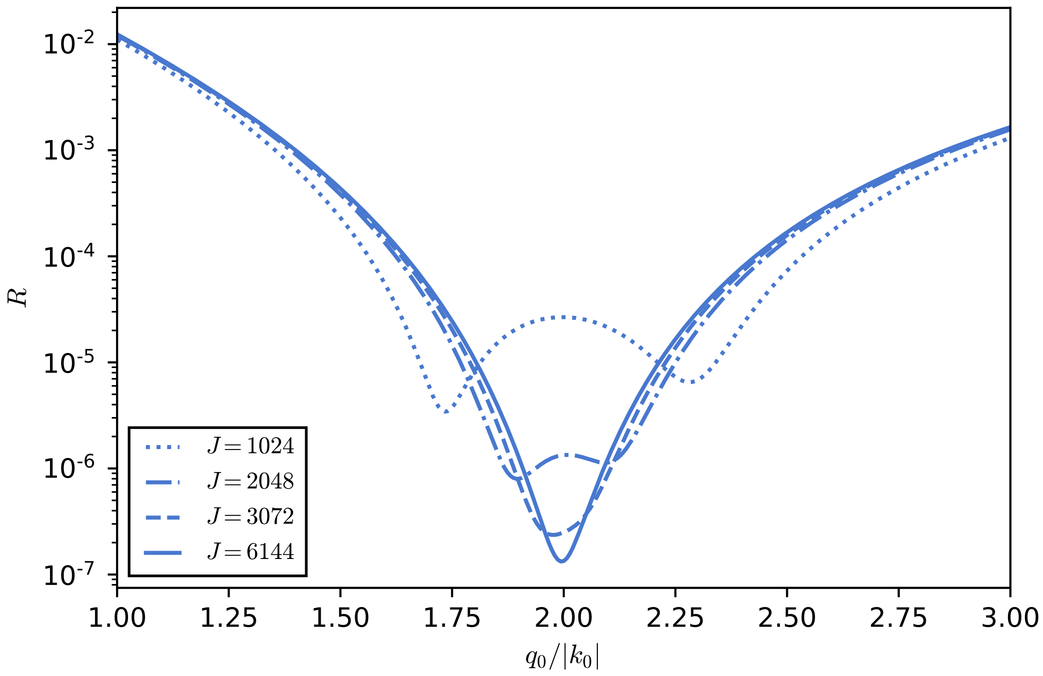


Figure 4.4: Reflection ratio for $p = 2$ as a function of $q_0/|k_0|$ calculated for various choices of J. Parameters used: $k_0 = 15$, $\sigma_0 = 1.5$ and $\Delta t = 0.001$. q_0 starts at k_0 and is increased up to $3k_0$ with step size $\Delta q_0 = 0.1$.

Now we will determine how the initial width σ_0 of the wave packet affects the reflection ratio. To do this, we calculate the reflection ratio as a function of time. Before that, it is important to note that the packet width does influence q_{min}. But this change in q_{min}, at least for the values of σ_0 used in this thesis, is negligible. For the parameters we choose $k_0 = 15$ and $q_0 = 2k_0$. The results of the calculations for $p = 2$ are shown in Fig. 4.5 on page 20.

It is obvious from the plots in Fig. 4.5 that the absorption process takes longer for wider wave packets. Furthermore the reflection ratio is lower for a wider spread. The difference between the reflection ratios of $\sigma_0 = 0.25$ and $\sigma_0 = 1.0$ is multiple orders of magnitude, as can be seen in Table 4.3. To explain this, we consider the momentum uncertainty of a Gaussian wave packet:

$$\Delta k = \frac{1}{\sigma_0\sqrt{2}} \tag{4.6}$$

A wave packet with small momentum uncertainty represents a wave-like object, with $\Delta k = 0$ simply being a plane wave, while a wave packet with large momentum

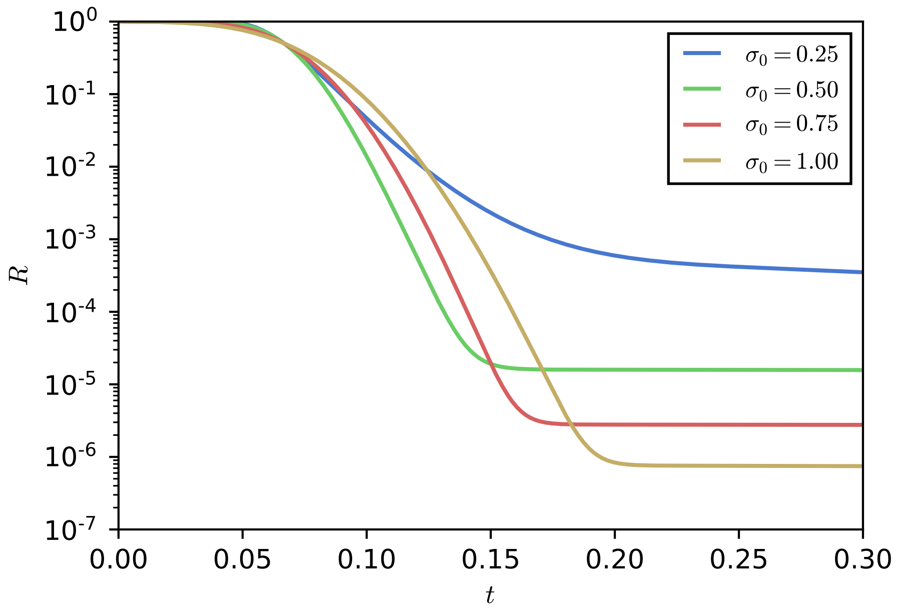


Figure 4.5: Reflection ratio as a function of time calculated for various choices of σ_0. Parameters used: $x_0 = 0$, $x_J = 10$, $J = 3072$, $\xi_0 = 8$, $k_0 = 15$ and $q_0 = 2k_0$.

uncertainty represents a particle-like object. Since the ABCs are derived from the plane wave ansatz (2.5), it is not surprising that the reflection ratio is lower for a higher momentum uncertainty (and therefore lower initial width). Similar results can be obtained for $p = 3$.

Table 4.3

σ_0	t	R
0.25	0.30	$3.507 \cdot 10^{-4}$
0.50	0.30	$1.570 \cdot 10^{-5}$
0.75	0.30	$2.766 \cdot 10^{-6}$
1.00	0.30	$7.461 \cdot 10^{-7}$

4.2 Potential Scattering and Transmission Coefficients

We will now apply the ABCs to quantum mechanical potential scattering and calculate the transmission coefficient for a Gaussian wave packet numerically. Due to the ABCs there will be, for the most part, negligible back reflection at the boundaries. To determine the transmission probability we use the following method

$$\mathcal{T} = \frac{\sum_{n=0}^{N} \mathcal{J}_T(x_{J-2}, t_n)}{\sum_{n=0}^{N} \mathcal{J}_I(x_{J-2}, t_n)}. \tag{4.7}$$

Here $\mathcal{J}_T$ and $\mathcal{J}_I$ are the probability current at the third to last spacial grid point calculated through

$$\mathcal{J}(x_{J-2}, t_n) = \frac{1}{i}\left(\overline{\psi}_{J-2}^{n} \frac{\psi_{J-3}^{n} - \psi_{J-1}^{n}}{2\Delta x} - \psi_{J-2}^{n} \frac{\overline{\psi}_{J-3}^{n} - \overline{\psi}_{J-1}^{n}}{2\Delta x}\right), \tag{4.8}$$

where the bar denotes the complex conjugate. Note that to obtain the initial probability current we perform a calculation with no barrier present, that is the potential set to zero. For the calculations in this section we use the parameters listed in Table 4.4 unless stated otherwise. The average energy of the wave packet is $\langle E \rangle = 100$. For the

Table 4.4

Parameters	J	x_0	x_J	Δt	ξ_0	k_0
Values	3072	-20	10	0.001	-10	10

ABCs we use $p = 3$ and set $q_0 = 2k_0$. To obtain the transmission curves we vary the potential height V_0 from 0 to 150 in integer steps.

Note that it is possible to analytically calculate the transmission coefficient for a Gaussian wave packet through [1]

$$\mathcal{T} = \int_{-\infty}^{\infty} dk P(k) T(k) = \int_{-\infty}^{\infty} dk |\varphi(k, 0)|^2 T(k) \tag{4.9}$$

where $\varphi(k, 0)$ is the initial wave function in momentum space and $T(k)$ is the transmission coefficient of a plane waves through the barrier as a function of momentum. This is of course only possible if $T(k)$ can be calculated for the barrier in question.

4.2.1 Rectangular Barrier

We start with a basic rectangular barrier

$$V_{Barrier}(x, a, w) = \begin{cases} 0, & x < a - \frac{w}{2} \\ V_0, & a - \frac{w}{2} \leq x \leq a + \frac{w}{2} \\ 0, & x > a + \frac{w}{2} \end{cases} \tag{4.10}$$

centred at a and with width w.

For this barrier, the plane wave transmission coefficient reads as follows (for a derivation see for example Ref. [10])

$$T(k) = \begin{cases} \frac{4\rho^2}{(1+\rho^2)^2 \sinh^2(\kappa w) \ + \ 4\rho^2}, & E < V_0 \\ \frac{4|\rho|^2}{(1-|\rho|^2)^2 \sin^2(|\kappa| w) \ + \ 4|\rho|^2}, & E \geq V_0 \end{cases} \tag{4.11}$$

with $k = \sqrt{E}$, $\kappa = \sqrt{V_0 - E}$, $\rho = \frac{\kappa}{k}$ and w the barrier width. We can therefore evaluate Eq. (4.9) to obtain the analytic transmission values and compare them to the numerical results obtained through Eq. (4.7).

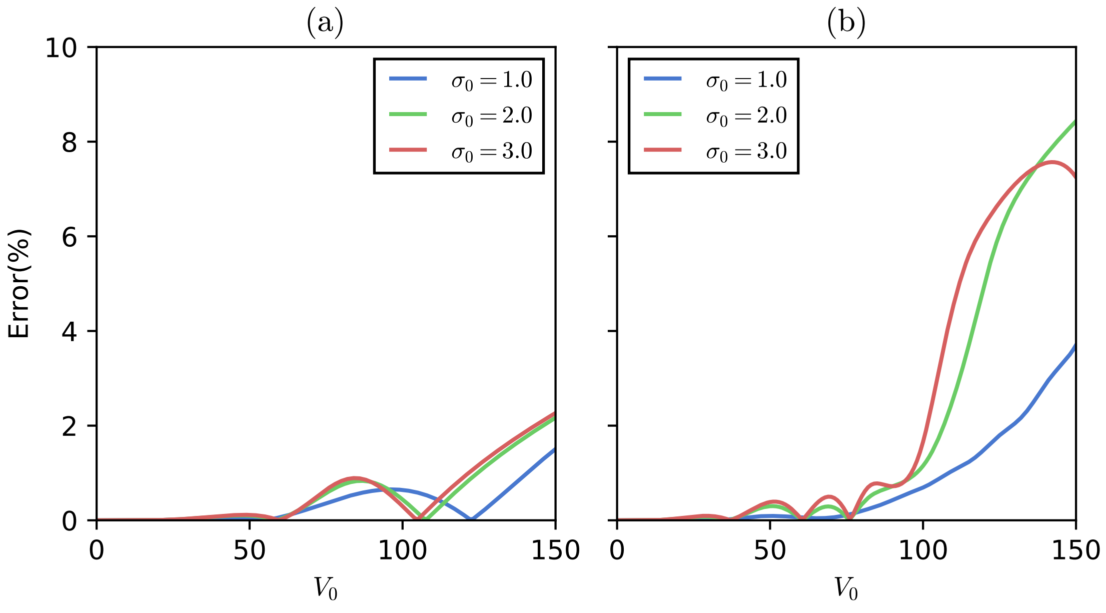


Figure 4.6: (a) Percentage error as a function of potential strength. Parameters used: $t_N = 2.0$, $a = 5$ and $w = 0.5$, (b) same as (a) but $w = 1.0$.

To this end we calculate the percentage error for a barrier with width 0.5 and a barrier with width 1.0. The results are plotted in Fig. 4.6. From this Figure we see that the percentage error is largest as $\langle E \rangle < V_0$. The error is also significantly larger for the wider barrier. This happens since the transmission probability is already low for these potential heights, as in $\mathcal{T} << 0.1$. At this point the leftover wave packet that is reflected at the boundary will start to contribute to the error. Additionally, it should be noted that the transmitted wave packet has, in general, *not* the same velocity as the initial wave packet (for further discussion of the phenomenon see Ref. [11]). This could possibly influence the effectiveness of the ABCs.

In Fig. 4.7 the numerically calculated transmission curves for different initial widths and two different barriers as a function of V_0 are plotted. Here we can see the influence of the momentum uncertainty from Eq. (4.6) on the transmission probability of the wave packet. A Gaussian with high momentum uncertainty has a higher transmission probability when $\langle E\rangle < V_0$. As Δk becomes smaller, the transmission probability approaches the the plane wave values, as should be expected.

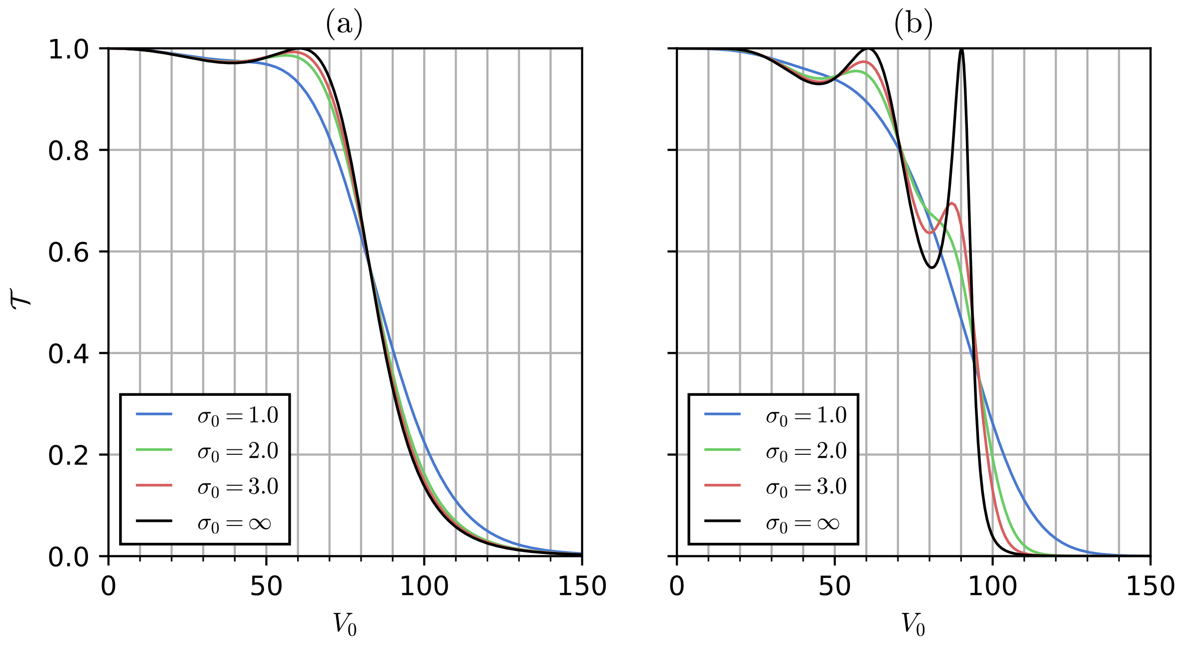


Figure 4.7: Transmission coefficient as a function of potential strength. Parameters used: $t_N = 2.0$, $a = 5$ and (a) $w = 0.5$, (b) $w = 1.0$.

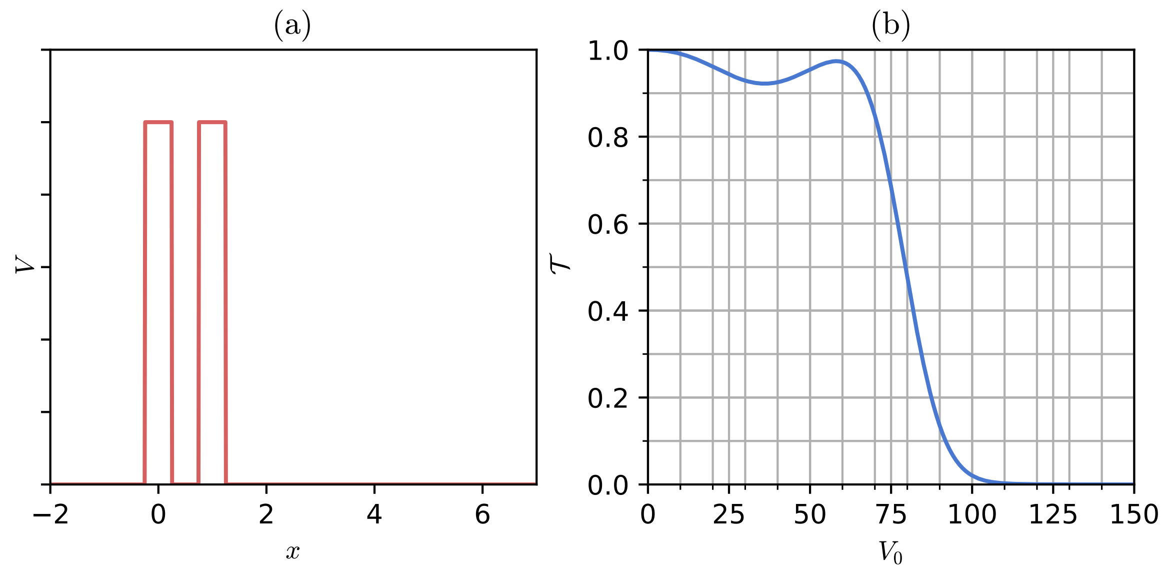
4.2.2 Rectangular Double Barrier

Next we consider a combination of two rectangular barriers with the same height

$$V_{Double} = V_{Barrier}(x, a_1, w_1) + V_{Barrier}(x, a_2, w_2). \tag{4.12}$$

Different double barriers and the corresponding transmission curves are plotted in Fig. 4.8 on page 24 and in Fig. 4.9 on page 25. For these transmission curves we set $t_N = 3.0$ and $\sigma_0 = 2.0$. We note the following:

The transmission curves for two barriers with the same width have the same general shape, as the curve for a single barrier. The transmission probability increases noticeable if the same barriers are placed further from each other.



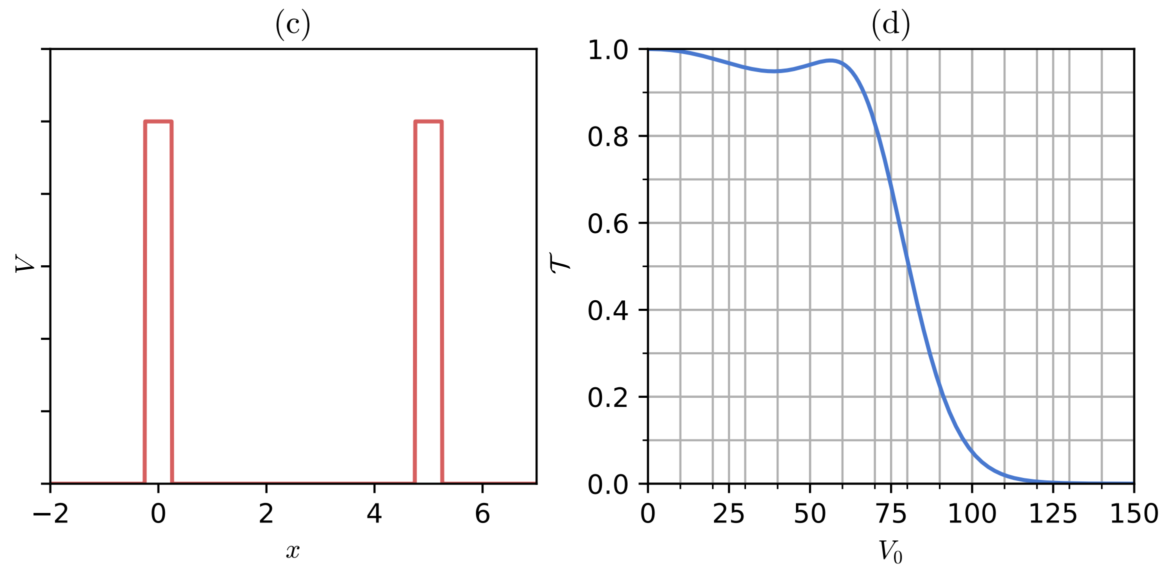


Figure 4.8: Plots of the potential barriers and the transmission probabilities. (a) a_1=0, $a_2 = 1$ and $w_1 = w_2 = 0.5$, (b) transmission probability for the barrier in (a), (c) a_1=0, $a_2 = 5$ and $w_1 = w_2 = 0.5$, (d) transmission probability for the barrier in (c).

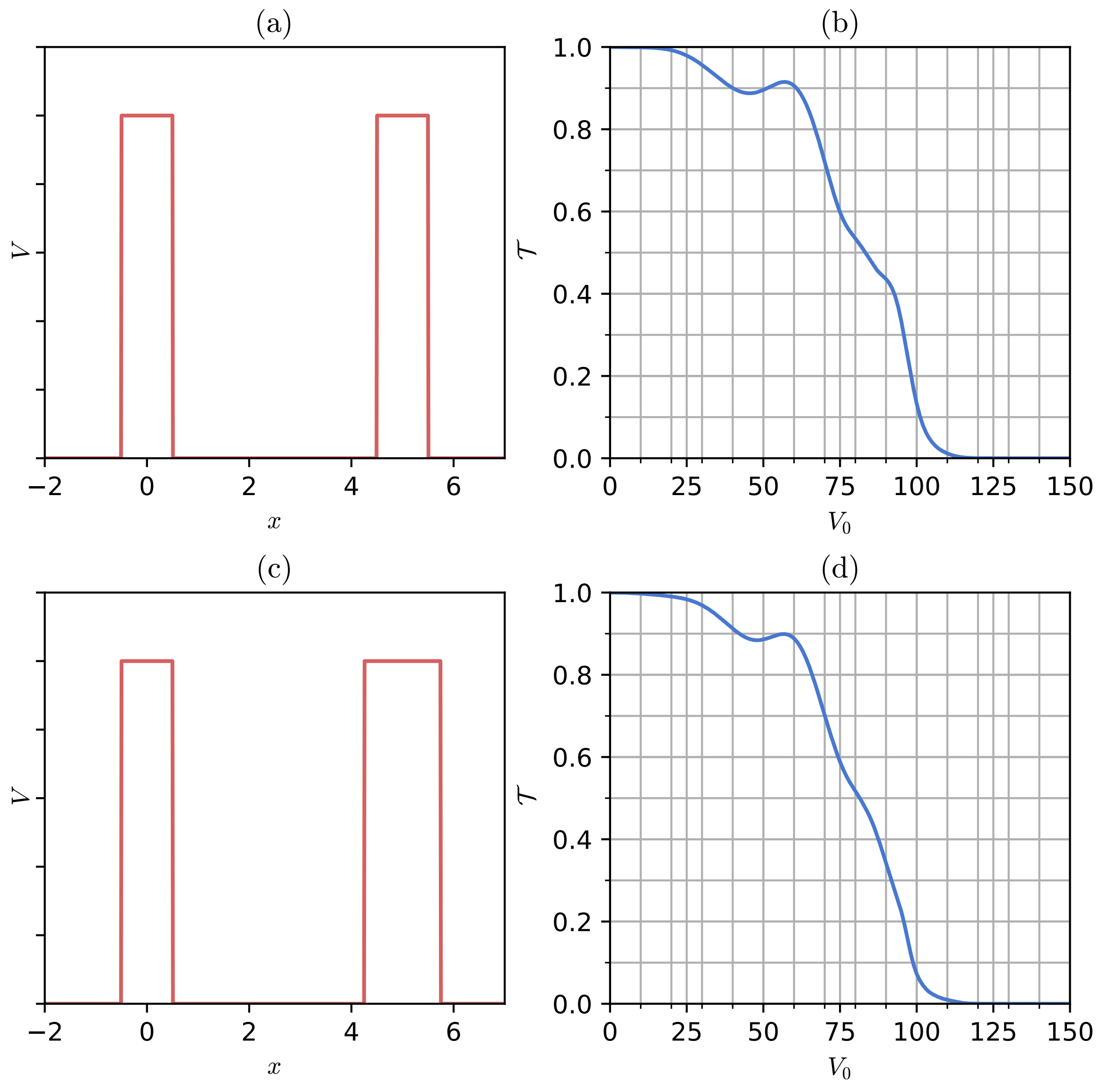


Figure 4.9: Plots of the potential barriers and the transmission probabilities. (a) a_1=0, $a_2 = 5$ and $w_1 = w_2 = 1.0$, (b) transmission probability for the barrier in (a), (c) a_1=0, $a_2 = 5$, $w_1 = 1.0$ and $w_2 = 1.5$, (d) transmission probability for the barrier in (c).

4.2.3 Smooth Double Barrier

Lastly we consider a smooth double barrier, modelled through the following Gaussian function

$$V_{Smooth} = V_0(e^{-(x-a_1)^2/\sigma_1^2} + e^{-(x-a_2)^2/\sigma_2^2}). \tag{4.13}$$

t_N and σ_0 are the same as in subsection 4.2.2. The barriers and the transmission curves are plotted in Fig. 4.10 below and in Fig. 4.11 on page 27.

Compared to the rectangular barriers, the smooth barriers are almost completely transparent up to a certain potential height. Furthermore, the transition from transparent to nearly impenetrable is much sharper, especially for the wider barriers.

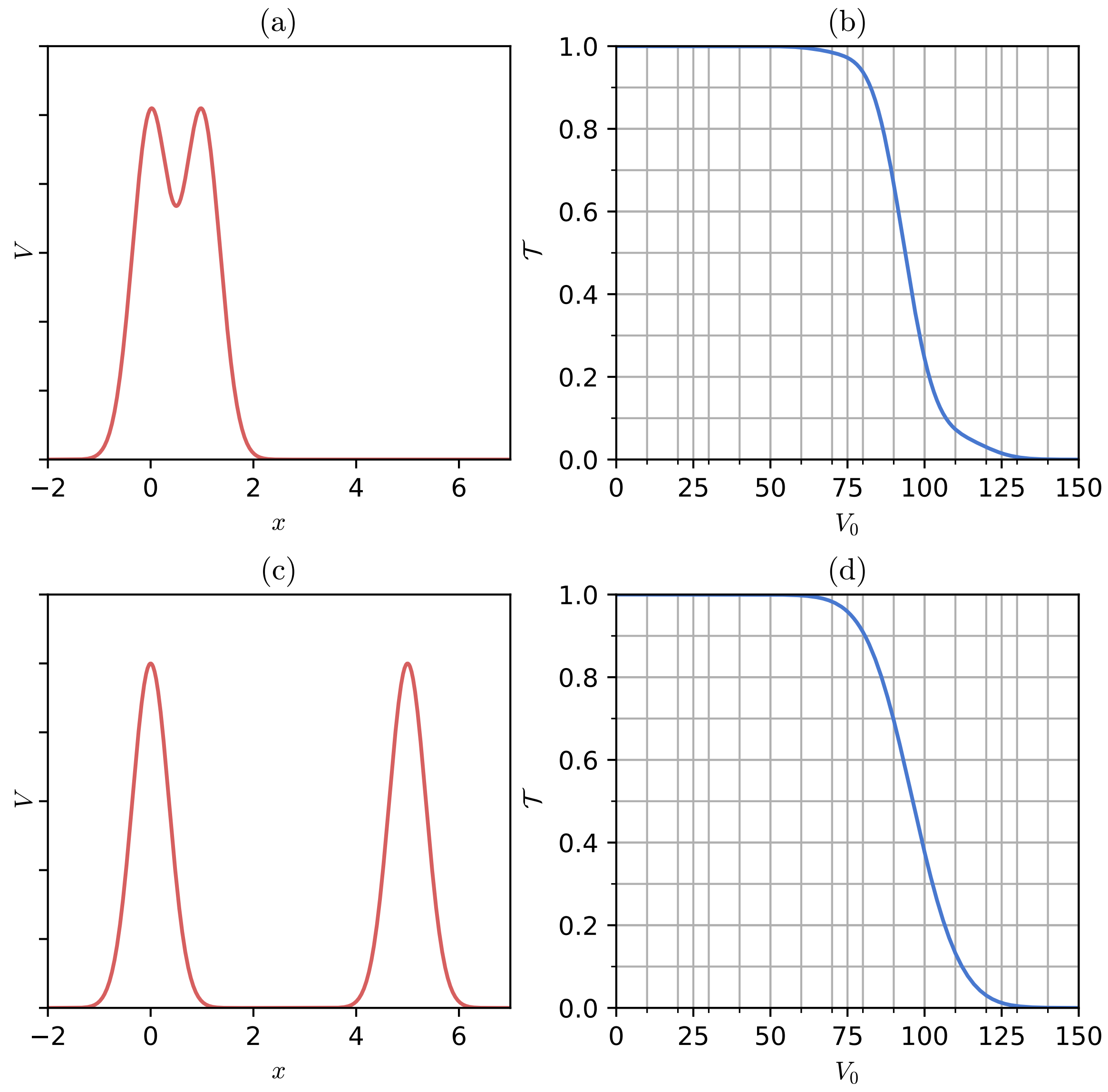


Figure 4.10: Plots of the potential barriers and the transmission probabilities. (a) a_1=0, $a_2 = 1$ and $\sigma_1 = \sigma_2 = 0.5$, (b) transmission probability for the barrier in (a), (c) a_1=0, $a_2 = 5$ and $\sigma_1 = \sigma_2 = 0.5$, (d) transmission probability for the barrier in (c).

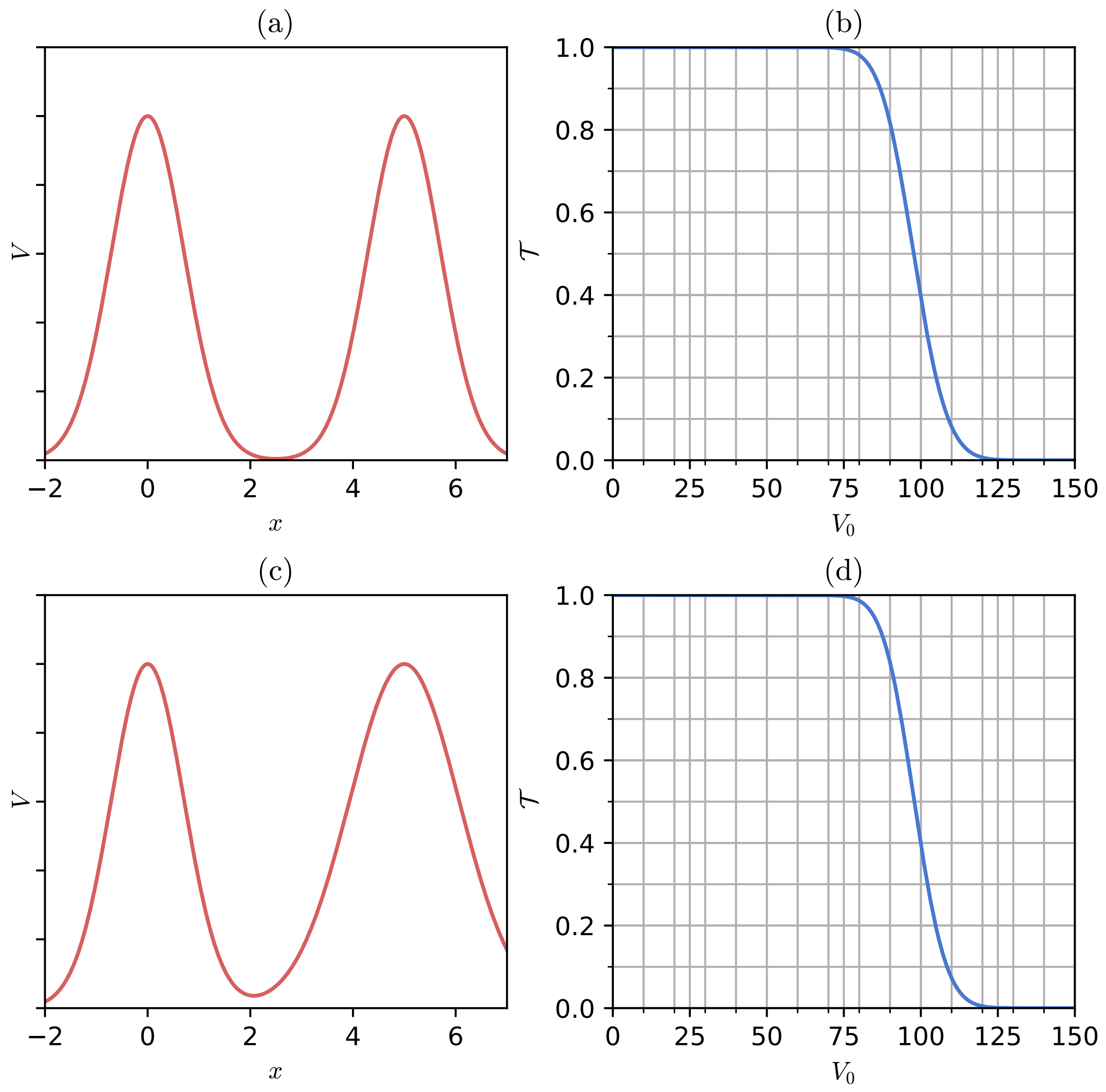


Figure 4.11: Plots of the potential barriers and the transmission probabilities. (a) a_1=0, $a_2 = 5$ and $\sigma_1 = \sigma_2 = 1.0$, (b) transmission probability for the barrier in (a), (c) a_1=0, $a_2 = 5$, $\sigma_1 = 1.0$ and $\sigma_2 = 1.5$, (d) transmission probability for the barrier in (c).

Chapter 5

Conclusion

In this thesis, we discussed a type of absorbing boundary conditions and tested their efficiency with a Gaussian wave packet. From this we conclude that the absorbing boundary conditions used in this thesis are very effective when using wave packets with large initial width ($\sigma_0 > 1$) and small average momentum ($k_0 < 15$). When increasing the momentum, the ratio of grid points per wavelength should be considered, in order to keep the method accurate.

What is more we used the absorbing boundary conditions to calculate the transmission coefficients for various shapes of potential barriers. The results for a rectangular barrier are in good agreement with the analytical values, as long as the transmission coefficient is not to small ($\mathcal{T} << 0.1$).

The methods presented in this thesis can be used to calculate transmission coefficients for other barrier structures, as well as to create animations of Gaussian wave packets in one dimension.

Bibliography

[1] T. Norsen, J. Lande, and S. B. McKagan. How and why to think about scattering in terms of wave packets instead of plane waves. *arXiv:0808.3566 [quant-ph]*, 2008.

[2] A. Goldberg, H. M. Schey, and J. L. Schwartz. Computer-Generated Motion Pictures of One-Dimensional Quantum-Mechanical Transmission and Reflection Phenomena. *Am. J. Phys.*, 35(3):177–186, 1967.

[3] D. Neuhasuer and M. Baer. The time-dependent Schrödinger equation: Application of absorbing boundary conditions. *J. Chem. Phys.*, 90(8):4351–4355, 1989.

[4] J.-P. Kuska. Absorbing boundary conditions for the Schrödinger equation on finite intervals. *Phys. Rev. B*, 46(8):5000–5003, 1992.

[5] T. Shibata. Absorbing boundary conditions for the finite-difference time-domain calculation of the one-dimensional Schrödinger equation. *Phys. Rev. B*, 43(8):6760–6763, 1991.

[6] T. Fevens and H. Jiang. Absorbing Boundary Conditions for the Schrödinger Equation. *SIAM J. Sci. Comput.*, 21(1):255–282, 1999.

[7] P. Puschnig. Computerorientierte Physik. `http://physik.uni-graz.at/~pep/CompOriPhys/CoP.pdf`, 2016.

[8] Python Software Foundation. Python Language Reference, version 2.7. Available at `https://www.python.org/`.

[9] F. L. Dubeibe. SOLVING THE TIME-DEPENDENT SCHRÖDINGER EQUATION WITH ABSORBING BOUNDARY CONDITIONS AND SOURCE TERMS IN MATHEMATICA 6.0. *Int. J. Mod. Phys. C*, 21(11):1391–1406, 2010.

[10] H. G. Evertz and W. von der Linden. Quantenmechanik. `https://itp.tugraz.at/~evertz/QM/qm_2017.pdf`, 2017.

[11] M. H. Bramhall and B. M. Casper. Reflections on a Wave Packet Approach to Quantum Mechanical Barrier Penetration. *Am. J. Phys.*, 38(9):1136–1145, 1970.