Entwicklung eines interaktiven Programms zur Simulation und Veranschaulichung von quantenmechanischen Wellenfunktionen

Development of an interactive application for simulation and visualization of quantum-mechanical wave functions

von
Patrick Richter

Bachelor-Arbeit im Studiengang Physik
Universität Hamburg
1. Institut für Theoretische Physik

2020

1. Gutachter: Prof. Dr. Robin Santra
2. Gutachterin: Prof. Dr. Daniela Pfannkuche

Hiermit bestätige ich, dass die vorliegende Arbeit von mir selbständig verfasst wurde und ich keine anderen als die angegebenen Hilfsmittel - insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen - verwendet habe und die Arbeit von mir vorher nicht einem anderen Prüfungsverfahren eingereicht wurde. Diese Version entspricht der eingereichten schriftliche Fassung bis auf einige Fehlerkorrekturen. Ich bin damit einverstanden, dass die Bachelorarbeit veröffentlicht wird.

Hamburg, 25. Juni 2020

Zusammenfassung

Die Beschreibung von Teilchen durch quantenmechanische Wellenfunktionen ist ein spannender Teilaspekt der modernen Physik, der jedoch für viele nicht besonders intuitiv ist. In dieser Arbeit wird die Entwicklung und Handhabung einer interaktiven Webanwendung beschrieben welche beim Verständnis dieses Konzeptes eine Hilfestellung sein kann.

Das Programm simuliert die zeitliche Entwicklung von eindimensionalen Teilchenwellen in der Ortsdarstellung mit Hilfe der Schrödingergleichung und stellt diese graphisch dar. Als numerisches Verfahren für die Zeit wird dabei das Runge-Kutta-Verfahren vierter Ordnung verwendet, während die Finite-Differenzen-Methode als numerisches Verfahren für den Ort genutzt wird.

Die Anwendung bietet viele unterschiedliche Werkzeuge für die Darstellung und Manipulation der quantenmechanischen Wellenfunktionen. Ein besonderer Schwerpunkt liegt dabei darin dem Nutzer die Möglichkeit zur geben, selbst ein zeitabhängiges Potential zu definieren, über das er auch bei laufender Simulation mit der Wellenfunktion interagieren kann.

Für die Überprüfung des Fehlers der numerischen Berechnungen wird die Übereinstimmung der Simulationsergebnisse mit der analytischen Lösung der Schrödingergleichung untersucht. Das wird am Beispiel des unendlich hohen Potentialkastens durchgeführt.

Abstract

The description of particles by quantum-mechanical wave functions is a fascinating aspect of modern physics, that is not very intuitive to many people. This work describes the development and use of an interactive web-application that is helpful for the understanding of that concept.

The program simulates the time dependent change of one-dimensional particle waves in a representation of the position by using the Schrödinger equation. Therefore, the Runge-Kutta method of fourth order is used in combination with finite differencing.

The application offers many different tools for the visualization and manipulation of quantum-mechanical wave functions. An special focus is the possibility of adding a user-defined, time-dependent potential, which can be changed interactively while the simulation is running.

For the verification of the numerical method, the outcome of the simulation is compared with the analytic solution. This is done with the example of the infinite potential box.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die Quantenmechanik ist ein faszinierender Teilbereich der Physik, welcher einerseits elementar für das Verständnis vieler physikalischer Prozesse ist, aber andererseits für interessierte Menschen nicht immer leicht zugänglich ist [1]. Das liegt daran, dass viele Konzepte für jemanden, der sich bisher lediglich mit der klassischer Physik befasst hat, möglicherweise nicht besonders intuitiv sind.

Ein gutes Beispiel für so ein Konzept ist der Welle-Teilchen-Dualismus. Dieser besagt, dass physikalische Teilchen sich bei genauerer Betrachtung nicht an einem Ort lokalisieren lassen, sondern Welleneigenschaften aufweisen. Eine Erkenntnis der Quantenmechanik ist, dass man solche Teilchenwellen im einfachen Fall als eine komplexwertige Funktion in Abhängigkeit des Ortes und der Zeit beschreiben kann. [2] Diese Funktion $\Psi(x,t)$ kann nicht direkt im Experiment gemessen werden. Dafür beschreibt das Betragsquadrat der Funktion die Aufenthaltswahrscheinlichkeit des Teilchens bei einer Messung.

Die zeitliche Entwicklung einer solchen Wellenfunktion kann im nichtrelativistischen Fall durch die Schrödingergleichung beschrieben werden [2]:

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = -\frac{\hbar^2}{2m}\Delta\Psi(x,t) + V(x,t)\Psi(x,t) \tag{1.1}$$

Dabei ist m die Masse des Teilchens und $V(x,t)$ das Potential, dem das Teilchen durch die jeweilige Wechselwirkung ausgesetzt ist. $\hbar$ ist eine Naturkonstante, die man als das reduzierte Plancksche Wirkumsquantum bezeichnet.

Die Beschreibung von Teilchen durch eine Wellenfunktion, deren Zeitentwicklung durch die Schrödingergleichung gegeben ist, kann die Ergebnisse vieler physikalischer Messungen vorhersagen, die durch die klassische Physik nicht korrekt beschrieben werden.

Wer es aber gewohnt ist, sich ein Elektron als lokalisiertes Teilchen vorzustellen, dem fällt es eventuell schwer zu verstehen, wie ein solches Elementarteilchen durch eine Welle beschrieben werden soll. Der Vorteil an einem klassischen Teilchen ist, dass es leicht möglich ist, ein Äquivalent aus dem Alltag dazu zu finden. Man kann es sich als ein greifbares Objekt vorstellen, das sich durch den Raum bewegt.

Bei einer Teilchenwelle ist das nicht so ohne Weiteres möglich. Man hat zwar möglicherweise eine exakte mathematische Beschreibung der Welle und deren zeitlicher Entwicklung, aber kann sich unter Umständen keine bildliche Vorstellung davon machen, wie der Vorgang abläuft. Dabei ist es in vielen Situationen hilfreich, wenn man sich einen physikalischen Vorgang visuell vorstellen kann. Dadurch gewinnt man einen zusätzlichen Blickwinkel auf ein physikalisches Problem und findet möglicherweise Lösungsansätze, die einem durch alleinige Betrachtung der mathematischen Formel nicht zugänglich gewesen wären. Wenn man ein intuitives Verständnis und eine visuelle Vorstellung davon erlangen will, was eine Teilchenwelle auszeichnet, dann ist es hilfreich ein makroskopisches

Äquivalent zu haben, mit dem man interagieren und dessen Verhalten man beobachten kann. Eine große Hilfestellung kann hier ein interaktives Simulationsprogramm leisten mit dem man eigenständig experimentieren kann [3].

Es gibt bereits viele gute Programme, die hilfreich sein können, um quantenmechanische Phänomene nachzuvollziehen und vorherzusagen [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]. Jedes davon hat eine eigene Herangehensweise und weist unterschiedliche Vor- und Nachteile auf. Ein Ansatz, der dabei häufig zum Einsatz kommt, um die Eigenschaften von Teilchenwellen zu zeigen, ist die Berechnung der exakten Lösung der Schrödingergleichung durch die Superposition von Eigenzuständen zu einem bekannten, zeitunabhängigen Potential. Nicht so häufig verwendet wird der Ansatz, dem Nutzer eine freie Wahl des Anfangszustands und des zeitabhängigen Potentials zu ermöglichen. Das kann jedoch hilfreich sein, wenn man mit ungewöhnlicheren quantenmechanischen Szenarien experimentieren will.

1.1 Das Ziel der Arbeit

Das Ziel dieser Arbeit ist es deswegen, ein interaktives Programm zur Simulation und Veranschaulichung von eindimensionalen, quantenmechanischen Wellenfunktionen zu entwickeln, das dem Nutzer viele unterschiedliche Werkzeuge zur Manipulation der Wellenfunktion zur Verfügung stellt.

Die zeitliche Änderung einer Teilchenwelle soll numerisch berechnet werden. Die berechneten Änderungen sollen auf dem Bildschirm als flüssige Animation dargestellt werden. Der Nutzer soll die Möglichkeit haben, den Einfluss unterschiedlicher zeitabhängiger Potentiale auf die Entwicklung der Teilchenwelle zu studieren. Der Schwerpunkt der Arbeit liegt darauf, das Programm möglichst interaktiv zu gestalten. Die Manipulation der Teilchenwelle durch eine Änderung des Potentials soll bei laufender Berechnung und Animation möglich sein. Auf diese Weise sollen Studierende eine intuitive Vorstellung davon gewinnen können, was eine quantenmechanische Teilchenwellenfunktion ist und wie sie sich zeitlich entwickelt. Damit das Programm leicht genutzt werden kann soll es als Webanwendung über das Internet nutzbar sein.

1.2 Eine Übersicht zu den Kapiteln

- In Kapitel 2 werden zunächst einige Überlegungen zusammengetragen, welche für die Umsetzung des Programms relevant sind.
- Kapitel 3 beschreibt, welche Strategie für die Entwicklung angewandt wird. Dabei werden sowohl physikalische als auch technische Aspekte beleuchtet.
- Kapitel 4 dokumentiert die Eigenschaften des fertigen Programms und erklärt den Umgang damit.
- In Kabitel 5 wird ein Maß für den Fehler der numerischen Simulation definiert, mit dessen Hilfe das verwendete numerische Verfahren überprüft wird.
- Kapitel 6 beschreibt zwei einfache Anwendungsbeispiele des Programms und zählt einige Beispiele für weitere Einsatzmöglichkeiten auf.
- Kapitel 7 befasst sich mit Verbesserungsvorschlägen an dem Programm und möglichen, zukünftigen Weiterentwicklungen.

- Kapitel 8 enthält eine knappe Zusammenfassung der Ergebnisse dieser Arbeit.

1.3 Die verwendeten Einheiten

In dieser Arbeit sind alle Angaben zu physikalischen Einheiten wie dem Ort, der Zeit, der Energie und der Teilchenmasse in atomaren Einheiten [16]. Das betrifft auch die Parameter des Simulationsprogramms. Damit nimmt sowohl das reduzierte Plancksche Wirkungsquantum $\hbar$, als auch die Elementarladung e den Wert 1 a.u. an. Zur besseren Einordnung wird das Plancksche Wirkungsquantum dennoch in Gleichungen verwendet. Auf das Anhängen des Kürzel a.u. (Für Atomic Units) wird in den meisten Fällen jedoch verzichtet.

Kapitel 2

Überlegungen zur Umsetzung des Programms

Damit das Programm sein Ziel erfüllen kann, als Hilfe für das Verständnis von quantenmechanischen Teilchenwellenfunktionen genutzt zu werden, sollte es einige Voraussetzungen erfüllen. Manche davon sind physikalischer Natur, während andere die technische Umsetzung betreffen. In diesem Abschnitt sollen die Erwartungen an das Programm formuliert und unterschiedliche Ansätze für die Umsetzung diskutiert werden.

2.1 Überlegungen zur physikalischen Umsetzung

Die zentrale Grundvoraussetzung ist, dass die Simulation physikalisch sinnvolle Ergebnisse liefert, die auch in einem sinnvollen Kontext dargestellt werden. Zusätzlich soll die gewählte Funktionalität flexibel eingesetzt werden können, um unterschiedliche Experimente nachzuvollziehen und dabei möglichst intuitiv sein. Folgende Anforderungen an die physikalische Umsetzung sollten erfüllt sein:

- Das Programm soll genutzt werden können, um die zeitliche Änderung quantenmechanischer Wellenfunktionen zu simulieren.
- Die Teilchenwelle soll so dargestellt werden, dass sie sich vollständig anhand der Abbildung charakterisieren lässt.
- Die Darstellung der Teilchenwelle soll intuitiv sein.
- Das verwendete Iterationsverfahren soll numerisch stabil sein.
- Die Ergebnisse der numerischen Berechnung sollen eine gute Übereinstimmung mit den erwarteten analytischen Lösungen aufweisen. Bei optimierter räumlicher und zeitlicher Auflösung soll das Ergebnis möglichst gegen die erwartete Lösung konvergieren.
- Parameter wie die Masse des Teilchens, die räumliche Auflösung der Wellenfunktion und die zeitliche Auflösung der numerischen Berechnung sollen vom Nutzer festgelegt werden können.

2.1.1 Das betrachtete Szenario

Zu Beginn des Entwicklungsprozesses sollte geklärt werden, welches physikalische Szenario von dem Programm simuliert wird. Vorgegeben ist bereits, dass die Wellenfunktion eines

Teilchens mit einer Masse größer als 0 behandelt werden soll. In Frage kommt also zum Beispiel das Elektron, welches als Elementarteilchen keine zusätzliche Komplexität mit sich bringt.

Es ist aber auch denkbar, mit dem Programm das Verhalten von Protonen, Neutronen oder sogar ganzen Atomen oder kleinen Molekülen zu simulieren. Beachtet werden sollte dabei, dass bei komplexeren Teilchen auch zusätzliche Effekte wie Wechselwirkungen innerhalb des Teilchens auftreten, die von der einfachen Schrödingergleichung nicht mehr korrekt beschrieben werden können. Auch muss bei schwereren Teilchen mit kleineren quantenmechanischen Wellenlängen gerechnet werden, was es schwerer macht, die von der Schrödingergleichnung beschriebenen Effekte durch Experimente zu beobachten.

Wichtig ist, dass die betrachteten Teilchen nichtrelativistische Geschwindigkeiten haben, weil die Schrödingergleichung nur in diesem Fall geeignet ist, um physikalische Vorgänge in guter Näherung zu beschreiben.

Das zeitabhängige Potential, welches zur Berechnung der Schrödingergleichung verwendet wird, lässt sich in der Praxis durch Kraftfelder realisieren, mit denen das gewählte Teilchen wechselwirkt. Ist es geladen, so ist die Verwendung eines elektromagnetischen Feldes naheliegend. Auch andere Wechselwirkungen, wie die durch das Gravitationsfeld, kommen in Frage.

Da das Simulationsprogramm die Schrödingergleichung in nur einer Dimension berechnet, während sich reale Teilchen im dreidimensionalen Raum bewegen, muss das betrachtete Szenario entsprechend präpariert sein, um ein eindimensionales Verhalten aufzuweisen. Beispielsweise kann man die Bewegungsfreiheit der betrachteten Teilchen auf eine Dimension einschränken.

2.1.2 Die verwendete Basis

Eine wichtige Frage, die vor der Implementierung des Programms geklärt sein sollte, ist, in welcher Darstellung die Teilchenwelle behandelt werden soll. Dabei geht es nicht nur um die Darstellung auf dem Bildschirm, sondern auch darum wie das Programm die Teilchenwelle intern handhabt. Da quantenmechanische Wellenfunktionen Vektoren in einem unendlich-dimensionalen Vektorraum sind, kann man unterschiedliche Basen verwenden, um sie darzustellen.

Drei Darstellungen, die häufig zum Einsatz kommen sind die Darstellung im Ortsraum, im Impulsraum und im Energieraum. Jede dieser Darstellungen hat unterschiedliche Vor- und Nachteile. Die Darstellung im Impulsraum bietet bei der Berechnung eines Teilchens im Kastenpotential keinen Mehrwert, weil es hier keine Impulserhaltung gibt.

Die Darstellung im Energieraum ist besonders geeignet, wenn man die exakte Lösung der zeitabhängigen Schrödingergleichung bei einem statischen Potential bestimmen will, bei dem die Eigenwerte des zugehörigen Hamilton-Operators bekannt sind. Für die Berechnung der zeitlichen Änderung einer Wellenfunktion in einem unendlich hohen Potentialtopf, welcher im Inneren ein konstantes, zeitlich unverändertes Potential hat, lässt sich dieser Ansatz gut verwenden.

Kommt es jedoch zu einer Änderung des Potential innerhalb des Kastens, so ändern sich damit auch die Eigenvektoren des Energieoperators. Bei einem komplizierten, zeitabhängigen Potential ist diese Darstellung also ungeeignet. In diesem Fall ist die Behandlung im Ortsraum sinnvoll. Weil hier das Potential auf einfache Weise in dem Hamilton-Operator einfließt, muss dieser bei einer Potentialänderung nicht aufwändig neu berechnet werden.

2.1.3 Das Iterationsverfahren

Zur numerischen Lösung der Schrödingergleichung bieten sich verschiedene mathematische Verfahren an. [17] Jedes davon unterscheidet sich im Konvergenzverhalten sowie der numerischen Stabilität und beansprucht dabei unterschiedlich viel Rechenleistung des Computers. Unterschieden wird dabei zwischen numerischen Verfahren für den Raum und für die Zeit. Bei der Darstellung der Wellenfunktion durch diskrete Funktionswerte im Ortsraum wird ein numerisches Verfahren zur Berechnung der zweiten Ortsableitung benötigt.

Ein besonders leicht zu realisierendes Beispiel ist die Berechnung mittels der Finite-Differenzen-Methode [18] [19] . Auch für die Durchführung der Zeitentwicklung in diskreten Zeitschritten muss ein numerisches Verfahren genutzt werden. Ein Beispiel hierfür ist das explizite Runge-Kutta-Verfahren [20]. Es lässt sich für verschiedene Ordnungen anwenden.

2.2 Überlegungen zur technischen Umsetzung

Bei der technischen Umsetzung des Programms sollten einige Dinge beachtet werden, welche den Umgang mit der Anwendung angenehmer gestalten. Diese zu berücksichtigen ist wichtig, wenn man erreichen will, dass das Programm produktiv zur Untersuchung von physikalischen Problemen eingesetzt werden kann:

- Das Programm sollte unkompliziert und ohne aufwändige Installation zugänglich sein.

- Das Programm sollte auf einer möglichst großen Anzahl von unterschiedlichen elektronischen Geräten funktionieren und keine speziellen Betriebssysteme voraussetzen.

- Die verwendete Implementierung sollte effizient genug sein, um die Simulation in angemessener Geschwindigkeit durchzuführen. Die Anzeige auf dem Bildschirm sollte zeitliche Änderungen flüssig darstellen.

- Das Programm sollte die Möglichkeit bieten die durchgeführten Experimente zu reproduzieren. Deswegen sollte es möglich sein den aktuellen Zustand des Programms abzuspeichern und zu einem späteren Zeitpunkt neu zu laden.

2.2.1 Programmiersprache

Bevor mit der Arbeit an dem Programmcode begonnen werden kann, müssen eine oder mehrere Programmiersprachen ausgewählt werden. Die Wahl der Programmiersprache hängt von mehreren Faktoren ab. Manche Programmiersprachen wie Python oder Javascript eignen sich für hohe Abstraktion und bieten eine Vielzahl von Tools an, welche die Arbeit an komplexen Programmabläufen vereinfachen können. Dafür unterliegen sie meist deutlich in der Ausführungsgeschwindigkeit, wenn man sie mit systemnahen Programmiersprachen wie C oder Fortran vergleicht. Außerdem sind manche Programmiersprachen auf bestimmte Ausführungsumgebungen zugeschnitten. Beispielsweise wird die Programmierung von Webanwendungen standardmäßig in Javascript vorgenommen.

2.2.2 Implementierung als klassisches Programm

Eine Option ist es das Simulationsprogramm als klassische Anwendung zu implementieren, die direkt von einem Betriebssystem wie Windows oder Linux ausgeführt werden kann. Diese Implementierung ermöglicht meist einen sehr effizienten Zugriff auf die Hardware des genutzten Geräts. Dadurch kann eine gute Rechenleistung und auch eine hohe Framerate bei der graphischen Darstellung erreicht werden. Der Nachteil ist, dass der Nutzer das Programm vor der Anwendung erst herunterladen und eventuell installieren muss. Außerdem ist es nicht trivial das Programm so zu gestalten, dass es auf verschiedener Hardware mit unterschiedlichen Betriebssystemen fehlerfrei läuft.

2.2.3 Implementierung als Webanwendung

Eine Alternative ist es das Programm als Webanwendung zu gestalten. Das hat den Vorteil, dass es grundsätzlich möglich ist, die Anwendung auf jedem Gerät mit einem modernen Webbrowser auszuführen. Die Software muss vor der Ausführung nicht erst vom Nutzer installiert werden sondern kann direkt per Eingabe der URL über das Internet genutzt werden.

Der Nachteil ist, dass es aufwändig ist die Webanwendung so zu gestalten, dass die physikalischen Berechnungen in angemessener Geschwindigkeit durchgeführt werden können. Die klassische Methode ein Programm als Webanwendung auszuführen, sieht dafür die Programmiersprache Javascript vor. Diese ist allerdings in der Regel deutlich langsamer in der Ausführung als vergleichbare hardwarenahe Implementierungen in anderen Programmiersprachen. Für rechenintensive Aufgaben wie die numerische Berechnung der zeitabhängigen Schrödingergleichung eines Wellenpakets ist sie deshalb ungeeignet.

2.2.4 Implementierung als Server-seitige Webanwendung

Eine Möglichkeit, die Performance der Webanwendung zu steigern, ist es die rechenintensiven Aufgaben auf einen Server zu verlagern. Ein Server ist ein Computer, der Anfragen eines technischen Gerätes über das Internet entgegennehmen und beantworten kann. Wenn das Programm als Webanwendung implementiert wird, wird dafür mindestens ein Server benötigt, der dem Internetbrowser die benötigten Inhalte der Webanwendung zur Verfügung stellt.

Ein solcher Server könnte zusätzlich auch die Berechnungen der zeitabhängigen Schrödingergleichung verwendet werden. Das kann sinnvoll sein weil ein guter Server eine viel größere Rechenleistung erreichen kann als ein gewöhnlicher PC. Gerade wenn es um aufwändige Berechnungen geht (Beispielsweise im mehrdimensionalen Fall) bietet sich dieser Ansatz an. [4]

2.2.5 Implementierung als Client-seitige Webanwendung

Die andere Option besteht darin, die Berechnung komplett auf Seite des Clients durchzuführen. Als Client wird in diesem Fall das Gerät bezeichnet, welches die Webanwendung im Browser anzeigt. Ein Vorteil ist, dass die Ergebnisse der Berechnung so nicht über das Internet übertragen werden müssen. Außerdem wird hier kein Server mit guter Rechenleistung benötigt und die Zahl der Nutzer, die das Programm gleichzeitig nutzen können ohne dass zusätzliche Server eingesetzt werden müssen, ist um ein Vielfaches höher.

Kapitel 3

Strategie

Um quantenmechanische Wellenfunktionen zu veranschaulichen wird ein Programm entwickelt, das diese in einer Dimension numerisch simuliert. Die Wellenfunktion wird dabei von dem Programm intern in der Ortsdarstellung gehandhabt. Die zeitliche Entwicklung der Wellenfunktion wird mit Hilfe der Finite-Differenzen-Methode [18] als numerischem Verfahren für den Ort und dem Runge-Kutta-Verfahren [20] als numerischem Verfahren für die Zeit berechnet.

Der Nutzer hat die Möglichkeit das Potential welches für die Berechnung der zeitabhängigen Schrödingergleichung verwendet wird mit der Computermaus innerhalb des Intervalls $[0, L]$ einzuzeichnen. Dabei ist es möglich, das Ergebnis der numerischen Berechnung auf unterschiedliche Weisen zu visualisieren:

Das Potential, der Imaginärteil, der Realteil, das Betragsquadrat der Wellenfunktion sowie eine dreidimensionale Projektionsdarstellung, die den Real- und Imaginärteil gemeinsam zeigt, können jeweils in Abhängigkeit des Ortes abgebildet werden. Die Zuordnung erfolgt durch eine individuelle Farbgebung. Diese Vielseitigkeit der Darstellung kann hilfreich für das Verständnis der Wellenfunktion sein [21].

Um die Interaktivität zu erhöhen, wird der aktuelle Fortschritt der numerischen Berechnung in Echtzeit auf dem Bildschirm angezeigt. Es ist möglich das Potential währen der laufenden Berechnung zu verändern, sodass der Benutzer über das Potential mit der Teilchenwelle interagieren kann.

Das Programm stellt verschiedene Werkzeuge zur Manipulation und Visualisierung der Teilchenwelle bereit. Parameter wie die Länge des Intervalls, die Teilchenmasse und auch die numerische Orts- und Zeitauflösung δ_x und δ_t können über ein Eingabefeld verändert werden. Der Zustand des Programms lässt sich als Datei abspeichern und aus ihr auch wieder laden.

Um das Programm für Interessierte leicht zugänglich zu machen ist es als Client-seitige Webanwendung im Internet unter einer URL aufrufbar und kann von modernen Browsern ausgeführt werden.

In diesem Kapitel wird erläutert auf welche Weise die oben beschriebenen Eigenschaften umgesetzt wurden. Zunächst wird die Strategie für die physikalische und anschließend für die technische Umsetzung beschrieben.

3.1 Strategie für die physikalische Umsetzung

Das Programm behandelt die Wellenfunktion als numerisches Objekt und verwendet mit dem expliziten Runge-Kutta-Verfahren vierter Ordnung [20] für die Zeitentwicklung und mit der Finite-Differenzen-Methode [18] für die Berechnung des Laplace-Operators

zwei numerische Verfahren. In diesem Abschnitt soll die Definition des numerischen Wellenvektors und des zeitabhängigen Potentials beschrieben und das Iterationsverfahren für die Berechnung der zeitlichen Änderung erläutert werden.

3.1.1 Die Wellenfunktion

Die numerische Teilchenwelle soll von dem Programm in der Ortsdarstellung behandelt werden. Sie wird also als komplexe Funktion gehandhabt, die auf einem Intervall der Länge L definiert ist und deren Betragsquadrat die Aufenthaltswahrscheinlichkeitsdichte des Teilchens angibt. Außerhalb dieses Intervalls nimmt die Funktion und somit auch die Aufenthaltswahrscheinlichkeitsdichte des Teilchens per Definition den Wert 0 an.

Um die Wellenfunktion mit endlichen Ressourcen an Arbeitsspeicher darstellen zu können, werden nur die Funktionswerte p_n an N_r verschiedenen diskreten Orten x_n betrachtet, welche im gleichen Abstand über das Intervall der Länge L verteilt sind. Die Distanz zwischen zwei numerischen Raumpunkten ist $\delta_x = L/(N_r - 1)$ da der erste und letzte Raumpunkt (x_1 und x_{N_r}) sich jeweils am Rand des Raumintervalls befinden. Weil die Wellenfunktion stetig sein soll und außerhalb des Intervalls den Wert 0 annimmt, müssen die Funktionswerte p_0 und p_{N_r} in guter Näherung ebenfalls den Wert 0 haben.

3.1.2 Das Potential

Das Potential soll analog zu der Wellenfunktion aus Abschnitt 3.1.1 an $(N_P)_r$ diskreten Orten $(x_P)_n$ durch die reellen Werte $(p_P)_n$ auf dem Intervall der Länge L definiert sein. Die Zahlen N_r und $(N_P)_r$ müssen nicht notwendigerweise identisch sein. Damit das Potential auf L kontinuierlich definiert ist wird es auf den Abschnitten zwischen zwei benachbarten Definitionspunkten $(x_P)_n$ und $(x_P)_{n+1}$ durch eine Gerade definiert welche diese verbindet:

$$[V(x)]_{x\in[(x_P)_n,(x_P)_{n+1}]} = (p_P)_n + [(p_P)_{n+1} - (p_P)_n]\frac{x - (x_P)_n}{(x_P)_{n+1} - (x_P)_n} \tag{3.1}$$

Das Potential soll sich im Verlauf der Zeit verändern können. Dazu sei es zu $(N_P)_t$ diskreten Zeitpunkten $(t_P)_m$ auf die oben beschriebene Weise im Raum definiert, zu denen es die Werte $V_m(x)$ annimmt. Die Zeitintervalle $(t_P)_{m+1} - (t_P)_m$ zwischen diesen V_m müssen anders als die Ortsintervalle zwischen den $(x_P)_n$ nicht äquidistant sein.

Für die Definition des Potentials auf dem Zeitintervall zwischen zwei Zeitpunkten $(t_P)_m$ und $(t_P)_{m+1}$ gibt es zwei unterschiedliche Modi: Das Potential kann entweder auf dem gesamten Zeitintervall den Zustand $V_m(x)$ annehmen und erst bei $(t_P)_{m+1}$ unstetig zu $V_{m+1}(x)$ wechseln, oder linear ineinander übergehen:

$$[V(x,t)]_{t\in[(t_P)_m,(t_P)_{m+1}]} = V_m(x) + [V_{m+1}(x) - V_m(x)]\frac{t - (t_P)_m}{(t_P)_{m+1} - (t_P)_m} \tag{3.2}$$

Für alle Zeitintervalle kann individuell gesetzt werden, welcher der beiden Modi angewandt wird.

3.1.3 Berechnung der zeitlichen Änderung

Die zeitliche Änderung des numerischen Wellenvektors soll mit Hilfe des expliziten Runge-Kutta-Verfahrens berechnet werden. Um das Runge-Kutta-Verfahren anwenden zu können muss die zeitliche Änderung der Wellenfunktion Ψ in Abhängigkeit von Ψ und der Zeit t bekannt sein.

$$\frac{\partial}{\partial t}\Psi(x,t) = f(\Psi(x,t),t) \tag{3.3}$$

Die zeitliche Änderung des analytischen Wellenvektors ist durch die Schrödingergleichung (Gleichung 1.1) gegeben. Um die Änderung des numerischen Vektors zu definieren muss eine Methode zur Berechnung des Laplace-Operators Δ eingeführt werden. In diesem Fall kommt hier die Finite-Differenzen-Methode für die Bestimmung der räumlichen Ableitung zum Einsatz.

3.1.4 Die Finite-Differenzen-Methode

Um die Ortsableitung am Punkt $x_n+\delta_x/2$ zu bestimmen, werden die beiden benachbarten numerischen Raumpunkte hinzugezogen. Seien p_n die diskreten Funktionswerte des Wellenvektors am Zeitpunkt t und x_n die dazugehörigen Orte. Dann nimmt man für $n \in \{1, \ldots, N_r - 1\}$ an:

$$\left[\frac{\partial}{\partial x}\Psi(x,t)\right]_{x=x_n+\frac{\delta_x}{2}} = \frac{p_{n+1} - p_n}{\delta_x} \tag{3.4}$$

Außerdem nimmt man für $n \in \{2, \ldots, N_r - 1\}$ an:

$$\begin{aligned} &\left[\Delta\Psi(x,t)\right]_{x=x_n} \\ =&\frac{\left[\frac{\partial}{\partial x}\Psi(x,t)\right]_{x=x_n+\frac{\delta_x}{2}} - \left[\frac{\partial}{\partial x}\Psi(x,t)\right]_{x=x_n-\frac{\delta_x}{2}}}{\delta_x} \\ =&\frac{p_{n-1} + p_{n+1} - 2p_n}{(\delta_x)^2} \end{aligned} \tag{3.5}$$

Durch Einsetzen von (3.5) in die Schrödingergleichung (1.1) erhält man nun die zeitliche Änderung aller p_n mit $n \in \{2, \ldots, N_r - 1\}$

$$\frac{\partial}{\partial t}p_n = ip_n\left(\frac{\hbar}{2m}\frac{p_{n-1} + p_{n+1} - 2p_n}{(\delta_x)^2} - \frac{1}{\hbar}V(x_n,t)\right) \tag{3.6}$$

Für p_1 und p_{N_r} wird die zeitliche Änderung auf 0 gesetzt, da der Funktionswert hier konstant 0 sein muss. Damit ist die, in Gleichung (3.3) gezeigte, zeitliche Änderung f des numerischen Vektors Ψ definiert.

3.1.5 Das explizite Runge-Kutta-Verfahren

Das explizite Runge-Kutta-Verfahren schreibt nun vor, wie man die Änderung von Ψ in diskreten Zeitschritten numerisch approximieren kann. Dazu wird ein Zeitintervall $[t, t+\delta_t]$ gewählt, für das der Wandel des Vektors Ψ_t hin zum Vektor $\Psi_{t+\delta_t}$ berechnet werden soll. Der Algorithmus ist durch die folgenden beiden Gleichungen gegeben:

$$\Psi_{t+\delta_t} = \Psi_t + \delta_t \sum_{j=1}^{R} b_j \Omega_j \tag{3.7}$$

$$\Omega_j = f(\Psi_t + \delta_t \sum_{l=1}^{j-1} a_{jl}\Omega_l, t + c_j\delta_t) \tag{3.8}$$

Der Parameter R steht für die Ordnung des Runge-Kutta-Verfahrens. Die Koeffizienten b_j, c_j und a_{jl} sind charakteristisch für das verwendete Runge-Kutta-Verfahren und können

in einer Tabelle aufgelistet werden, die sich am Butcher-Schema [20] orientiert:

$$\begin{array}{ccccc|c} & & & & & c_1 \\ a_{2,1} & & & & & c_2 \\ a_{3,1} & a_{3,2} & & & & c_3 \\ \vdots & \vdots & \ddots & & & \vdots \\ a_{R,1} & a_{R,2} & \dots & a_{R,R-1} & & c_R \\ \hline b_1 & b_2 & \dots & b_{R-1} & b_R & \end{array} , \tag{3.9}$$

Dieser Satz an Koeffizienten wird vom Programm als eine eindimensionale Liste gespeichert, welche die Werte aus der Tabelle 3.9 von links nach rechts Zeile für Zeile einordnet.

$$[c_1, a_{2,1}, c_2, a_{3,1}, a_{3,2}, c_3, \dots, a_{R,1}, a_{R,2} \dots, a_{R,R-1}, c_R, b_1, b_2, \dots, b_R - 1, b_R] \tag{3.10}$$

Als Voreinstellung werden beim Neustart des Programms die Koeffizienten des klassische Runge-Kutta-Verfahren vierter Ordnung geladen [22].

$$\begin{array}{cccc|c} & & & & 0 \\ \frac{1}{2} & & & & \frac{1}{2} \\ 0 & \frac{1}{2} & & & \frac{1}{2} \\ 0 & 0 & 1 & & 1 \\ \hline \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} & \end{array} \tag{3.11}$$

3.1.6 Berechnung der Norm

Während die zeitliche Änderung des numerischen Vektors Ψ berechnet wird soll fortlaufend auch die Norm von Ψ ermittelt und angezeigt werden. Weicht sie wesentlich von dem erwarteten Wert 1 ab, so ist das ein guter Indikator dafür, dass die numerische und analytische Lösung nicht mehr übereinstimmen. Die Norm einer eindimensionalen Wellenfunktion wird als Integral über das Betragsquadrat der komplexen Funktionswerte aufgefasst.

$$\|\Psi\|^2 = \int |\Psi(x)|^2 dx \tag{3.12}$$

Dieses Integral wird numerisch nach einem Verfahren erster Ordnung berechnet. Weil dieses auch an anderer Stelle zum Einsatz kommt, soll es hier allgemein über die Funktion $f(\Psi(x), x)$ formuliert werden, die in diesem Fall durch $|\Psi(x)|^2$ gegeben ist. Man betrachte zunächst die Funktionswerte von f an den Orten x_n:

$$f_n = f(p_n, x_n) \tag{3.13}$$

Das Integral wird über eine Summe der Werte von f_n für $n \in [1, \dots, N_r]$ berechnet. Diese werden dabei mit δ_x multipliziert da sie links und rechts von sich jeweils ein Intervall der Länge $\delta_x/2$ liegen haben, auf welchen in dieser Näherung f als konstant angenommen wird. Weil die Werte f_1 und f_{N_r} jeweils am Rand des Intervalls liegen werden sie zusätzlich durch zwei geteilt.

$$\int_{\text{num}} f dx := \delta_x \left(\sum_{i=2}^{N_r - 1} f(p_i, x_i, t) + \frac{f(p_1, x_1, t)}{2} + \frac{f(p_{N_r}, x_{N_r}, t)}{2} \right) \tag{3.14}$$

3.2 Strategie für die technische Umsetzung

Die Umsetzung des Programms als ausführbare Software geschieht durch die Kombination mehrerer unterschiedlicher Technologien. Da das Programm im Browser laufen soll, wird die visuelle Oberfläche des Programms als HTML-Dokument gestaltet. Der Javascript-Code, welcher die Benutzerinteraktion entgegen nimmt und für die Steuerung der numerischen Berechnungen zuständig ist, wird in dieses Eingebunden. Das numerische Verfahren selbst wird in der Programmiersprache C geschrieben. In diesem Abschnitt soll eine grobe Übersicht darüber gegeben werden wie das Programm technisch gesehen aufgebaut ist.

3.2.1 Gestaltung der Benutzeroberfläche als HTML-Dokument

Die Benutzeroberfläche ist als HTML-Dokument gestaltet. Darin sind alle Graphischen Elemente beschrieben, die der Nutzer sehen kann. Die Zeichenfläche, auf der die Wellenfunktion und das Potential visualisiert werden ist hier als `Canvas`-Element definiert und so gesetzt, dass es den gesamten Hintergrund des Browser-Fensters ausfüllt. Als zweites wichtiges Element ist an dieser Stelle ein Fenster definiert, in dem die Steuerungsparameter des Programms über viele unterschiedliche Buttons und Eingabefelder eingestellt werden können. Es befindet sich im Vordergrund der Zeichenfläche, sodass es diese verdeckt, kann aber ausgeblendet werden.

3.2.2 Implementierung der Programmsteuerung in Javascript

Der Programmcode, welcher für die Übergeordnete Steuerung aller Aktivitäten des Programms zuständig ist, wird in der Programmiersprache Javascript verfasst und ist direkt in der HTML-Datei eingebunden. Er wird zu verschiedenen Anlässen ausgeführt wie beispielsweise bei einem Neustart der Anwendung oder bei einer Benutzerinteraktion. In ihm ist auch die Hauptschleife enthalten, welche die Animation und Berechnung der Wellenfunktion abwechselnd, fortlaufend durchgeführt.

Da Javascript Multithreading (das parallele ausführen von Code) nicht direkt unterstützt, wird sämtlicher Javascript- und C-Code von einem Thread ausgeführt. Damit die Hauptschleife also den übrigen Javascript-Code nicht blockiert besteht sie aus einer Funktion, die nach erfolgreicher Ausführung fortlaufend ihren nächsten Durchlauf über den '`window.requestAnimationFrame`' Befehl anfordert.

Auch die Darstellung der Plots auf dem `Canvas`-Element sind in Javascript implementiert. Sie setzen sich aus vielen einzelnen geraden Linien zusammen, die jeweils für alle Intervalle zwischen zwei senkrechten Pixelreihen separat gezeichnet werden. Neben der Hauptschleife gibt es ein Vielzahl von Javascript-Funktionen, die jeweils für spezielle Benutzerinteraktionen aufgerufen werden und eine Änderung der Wellenfunktion, des Potentials oder der Darstellung bewirken. Ein Beispiel hierfür ist das Zeichnen des Potentials mit der Maus. Alle rechenintensiven Vorgänge wie die Berechnung der zeitlichen Änderung des Vektors werden werden von C-Code ausgeführt, der über Javascript aufgerufen wird.

3.2.3 Die Einbindung des C-Codes in die Webanwendung

Um den C-Code über den Browser laufen lassen zu können, wird die Webassembler-Schnittstelle genutzt. Sie ermöglicht es, Maschinencode im Webassembler-Format direkt im Browser auszuführen. Über die Webassembler-API kann das Javascript-Programm auf

die deutlich schneller laufenden Webassembler-Funktionen zugreifen. Für das Umwandeln des C-Codes in das Webassembler-Format wird das open-source-Werkzeug "Emscripten" verwendet. Es übernimmt außerdem einen Teil der Interaktion zwischen dem Javascript-Code und dem Webassembler-Code.

Der C-Code ist in zwei separaten Dateien abgespeichert und wird von Emscripten über eine Makefile-Datei in das Webassembler-Format konvertiert. Eine der beiden Dateien ist eine Header-Datei welche Struct-Definitionen und Deklarationen von Funktionen enthält. Sie hat die Dateiendung ".h". Die andere enthält die Definitionen der Funktionen und somit den eigentlichen Programmcode. Sie trägt die Dateiendung ".c". Alle Funktionen, auf die von der Javascript-Umgebung zugegriffen werden soll, werden mit dem Schlüsselwort "`EMSCRIPTEN_KEEPALIVE`" versehen. Das verhindert, dass sie bei der Codeoptimierung des Compilers entfernt werden.

3.2.4 Die Implementierung der Simulationsberechnung in C

C ist eine sehr maschinennahe Programmiersprache, die jedoch nicht objektorientiert ist. Für das Simulationsprogramm ist es allerdings für die Übersichtlichkeit an mehreren Stellen von Vorteil, wenn Daten und Funktionen zu logischen Objekten zusammengefasst werden. Ein numerischer Wellenvektor ist ein Beispiel für solch ein logisches Objekt.

Eine weitere Methode den Code zu abstrahieren und flexibler zu gestalten (die von C standardmäßig nicht unterstützt wird) ist die Verwendung von Schnittstellen. Sie ermöglichen, dass eine Funktion ein logisches Objekt verwendet und auf zugehörige Funktionen zugreift ohne dass die Implementierung des Objektes vorgegeben sein muss. Um von den Vorteilen beider Konzepte zu profitieren, werden bei der Umsetzung des Programms folgende Strategien eingesetzt:

Datenzugriff über Funktionszeiger

Daten, die logisch zusammengehören, werden in einem Struct gemeinsam abgespeichert. Solche Structs werden, wenn sie über einen längeren Zeitraum bestehen sollen, in einem Speicherbereich abgelegt, der zuvor auf dem Heap (Bereich im Arbeitsspeicher eines Programms für langlebige Daten) reserviert wurde. Um diese logischen Objekte an beliebige Funktionen innerhalb des Programmcodes weiterreichen zu können, ohne dass der gesamte Inhalt an einen anderen Speicherort kopiert werden muss, kommen Zeiger zum Einsatz.

Ein Zeiger ist eine Zahl, die an einen Ort im Arbeitsspeicher zeigt, an dem sich die dazugehörigen Daten befinden. Wichtig bei dieser Vorgehensweise ist, dass ein reservierter Speicherbereich wieder freigegeben wird, sobald man die zugehörigen Daten nicht mehr benötigt. Andernfalls kann es passieren, dass das Programm bei zunehmender Laufzeit immer mehr Speicherplatz benötigt bis es irgendwann abstürzt.

Der Einsatz von Makros

Um eine vielseitigere Verwendung einzelner Code-Abschnitte zu ermöglichen, soll Programmcode den Zugriff auf Funktionen haben können, deren Implementierung erst bei der Ausführung bekannt ist. C unterstützt zwar standardmäßig Funktionszeiger, das sind Zahlen, die auf eine Stelle im Arbeitsspeicher des Programms zeigen an der sich eine ausführbare Funktion befindet, allerdings verfügen diese Funktionen nicht über eigenen Arbeitsspeicher. Um diese Einschränkung zu umgehen werden als Hilfsmittel Macros eingesetzt über die sich die Funktionalität von Schnittstellen realisieren lässt.

Makros sind Schlüsselworte, die vom Compiler automatisch durch eigenen Code ersetzt werden. Sie ermöglichen es, die Syntax des Programmcodes selbst zu definieren, was die Übersichtlichkeit steigern kann.

Die Aufgabe der Makros ist es in diesem Fall die Definition von solchen Structs zu erleichtern welche einen Funktionszeiger mit einem Zeiger auf einen beliebigen Dateninhalt kombinieren. Zusätzlich implementieren sie einen Mechanismus, der es ermöglicht diese Structs als Funktion mit einem zuvor festgelegten Satz an Parametern aufzurufen. Intern wird dann der Funktionszeiger aufgerufen und erhält als zusätzlichen Parameter den Zeiger auf den Dateninhalt.

Diese Methode kommt zum Beispiel zum Einsatz um das Runge-Kutta-Verfahren unabhängig von der Funktion zu implementieren, welche die Zeitableitung des Vektors, wie in 3.3 gezeigt, berechnet. Dadurch ist es bei zukünftigen Erweiterungen leicht möglich den selben Code an anderer Stelle wiederzuverwenden wie beispielsweise bei der Berechnung von zweidimensionalen Teilchenwellen.

3.2.5 Die Verwendung von Javascript-Klassen

Die durch C-Code realisierten Funktionen können wie bereits erwähnt mit Hilfe von Emscripten von dem Javascrip-Code aus verwendet werden. Auch Zeiger auf Struct-Instanzen können von Javascript genutzt werden. Der Umgang mit beidem ist jedoch kompliziert und kann schnell unübersichtlich werden. Um die Verwendung zu erleichtern wird die Handhabung von Zeigern und der Zugriff auf C-Funktionen in Javascript-Klassen verlagert. Deren Instanzen können anschließend wie native Javascript-Objekte behandelt werden. Die folgende Liste nennt einige der wichtige Javascript-Klassen, die in dem Programm zum Einsatz kommen.

- `f64_SCHROED_BERECHNEN`: Diese Klasse fasst den gesamten Prozess der Berechnung der Zeitentwicklung der Wellenfunktion in sich zusammen und beinhaltet dabei einige weitere hier gelistete Klassen.

- `cf64_VEK`: Diese Klasse stellt, wie in 3.1.1 beschrieben, einen eindimensionalen numerischen Vektor in der Ortsdarstellung dar.

- `cf64_DOPPEL_VEK`: Diese Klasse fasst zwei `cf64_VEK` Instanzen zu einem Objekt zusammen. Sie wird benötigt weil in jeder Iteration der Zeitentwicklungsberechnung auf Grundlage des einen Vektors berechnet und in den Anderen geschrieben wird. Die Rolle der beiden Vektoren wechselt in jeder Iteration.

- `f64_SCHROED_WANDEL`: Diese Klasse berechnet die Zeitableitung des Vektors wie in 3.3 beschrieben.

- `f64_RUNGEKUTTA`: Diese Klasse kann ein beliebiges explizites Runge-Kutta-Verfahren darstellen.

- `f64_WANDEL_POT`: Diese Klasse enthält den Gesamten Zeitverlauf des Potentials der, wie in 3.1.2 beschrieben, in Form von einzelnen Keyframes abgespeichert sind. Diese sind untereinander wie eine Kette angeordnet und verweisen aufeinander mit Hilfe von Zeigern. Damit ein schnellerer Zugriff auf den Keyframe der aktuellen Zeit erfolgt wird separat auf den zuletzt verwendeten Keyframe verwiesen.

3.2.6 Gleitkommazahlen

Für die Berechnungen der Wellenfunktion wird neben den ganzzahligen `int` der Datentyp `double` verwendet. Er ist eine 64-Bit Gleitkommazahl und deswegen sinnvoll, weil er eine vergleichsweise hohe Präzision mit einem großen Wertebereich kombiniert und die zugehörigen Rechenoperationen von der Hardware der meisten Geräte unterstützt werden. Außerdem verwendet Javascript diesen Datentyp standardmäßig für alle Zahlen.

Damit aber untersucht werden kann, welche Rolle die Präzision des verwendeten Datentyps auf die numerische Berechnung hat, soll das gesamte Programm auch mit dem Datentyp `float` ausgeführt werden können, der nur eine 32-Bit Gleitkommazahl ist. Um beide Programmversionen parallel zu entwickeln wird darauf geachtet, dass der Programmcode durch eine einfache Modifizierung, zwischen beiden Versionen wechseln kann. Dabei muss man darauf achten, dass für Vektoren aus float- und double-Werten unterschiedlich viel Speicherplatz reserviert werden muss.

Kapitel 4

Das Programm

Das Programm ist als Webanwendung über den Browser aufrufbar. Zum Zeitpunkt der Veröffentlichung dieser Arbeit ist dies unter dem folgenden Link möglich:

https://quantsimulant.de/versionen/bachelorarbeit

Das Programm sollte über einen Computer aufgerufen werden, welcher über Maus und Tastatur verfügt. Empfohlen wird der Browser Firefox, unter dem die Anwendung getestet wurde. Da das Programm einige Schnittstellen nutzt, die noch nicht lange unterstützt werden, sollte man eine aktuelle Browser-Version verwenden. Beim ersten Aufruf soll ein Fenster mit Parametern im Vordergrund erscheinen welches in Abbildung 4.1 zu sehen ist. Bei einigen Browser-Konfigurationen kann es vorkommen, dass das Parameterfenster zu groß abgebildet wird um vollständig auf dem Bildschirm sichtbar zu sein. In diesem Fall wird empfohlen über die im Browser integrierte Zoom-Funktion das gesamte Fenster sichtbar zu machen. ('Strg'+'-' in Firefox). Im Hintergrund des Fensters wird ein blauer Bogen gemeinsam mit einem waagrechten schwarzen Strich angezeigt. Die blaue Kurve stellt den Realteil der Wellenfunktion im Grundzustand dar, während die schwarze Kurve das Potential anzeigt, das zu Beginn konstant den Wert 0 annimmt. Um die Kurven besser betrachten zu können kann mit der Taste 'P' das Parameterfenster ein- und ausgeblendet werden.

4.1 Die Simulation der Teilchenwelle

Über das Parameterfenster kann man Einstellung und Funktionsweise des Programms verändern. Die Simulation kann über den Button 'Start' gestartet und beendet werden. Alternativ ist das auch durch eine Betätigung der Leertaste möglich. Sobald man die Simulation startet, wird die zeitliche Änderung der Schödingergleichung auf dem Bildschirm angezeigt. Die aktuelle Zeit wird über ein entsprechend beschriftetes Eingabefeld in atomaren Einheiten angegeben und beginnt vom Startwert 0 aus hochzuzählen. Wenn man die Simulation startet ohne zuvor Änderungen am Potential vorgenommen zu haben, ist kaum eine Bewegung erkennbar. Das liegt daran, dass sich das Teilchen zu Beginn im Grundzustand mit niedriger Energie befindet. Dadurch beginnt es mit sehr langsamer Winkelgeschwindigkeit in den komplexen Zahlen zu rotieren weshalb sich der angezeigte Realteil der Wellenfunktion nur sehr langsam ändert.

Mit Hilfe der Maus ist es möglich das in schwarzer Farbe dargestellte Potential neu zu zeichnen. Das lässt sich sowohl bei angehaltener als auch bei laufender Simulation durchführen. Wenn die Berechnung der zeitlichen Änderung bei einem mit der Maus geänderten Potential erfolgt, wird sie dadurch beeinflusst. Auf dem Bildschirm ist nun

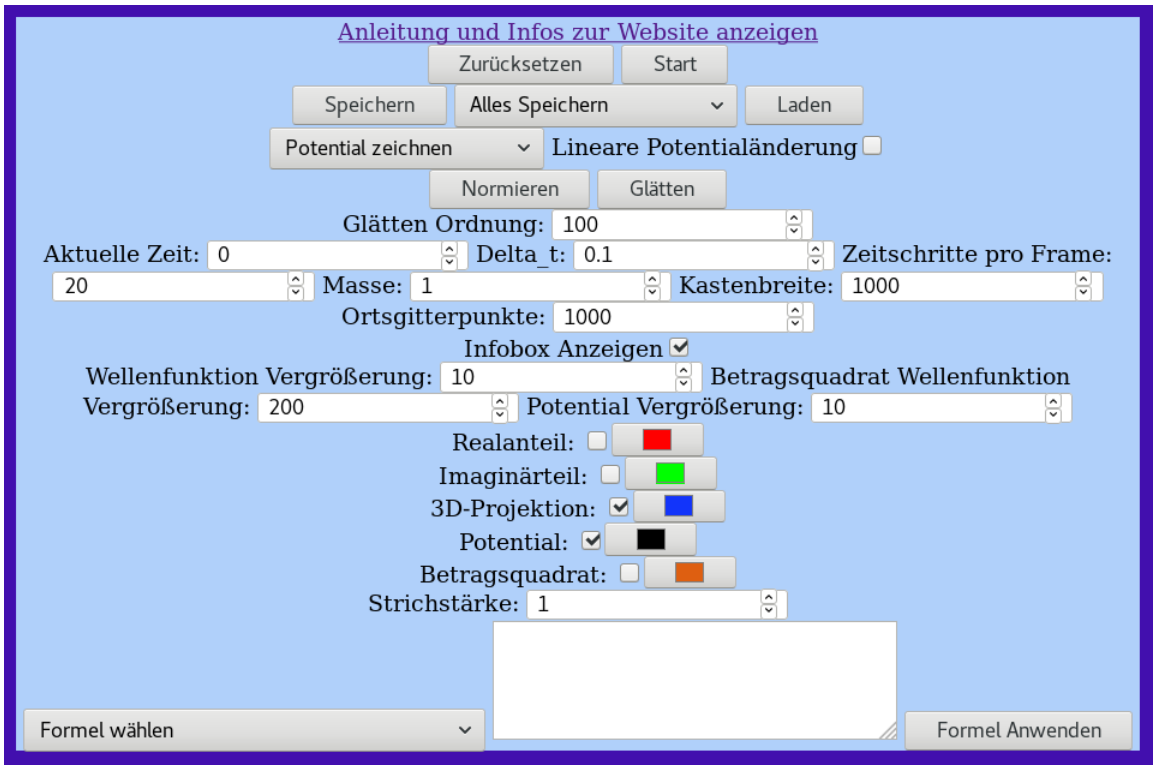


Abbildung 4.1: Ansicht des Parameterfensters in seinem voreingestellten Zustand

eine Änderung des Realteils der Wellenfunktion als flüssige Animation erkennbar, die von der Form des gezeichneten Potentials abhängt. Die aktuelle Zeit wird sowohl im Parameterfenster als auch in einem Infotext am oberen linken Bildschirmrand angezeigt, lässt sich jedoch nur über das Parameterfenster setzen.

Die Simulation kann jederzeit über die Leertaste angehalten werden. In diesem Fall stoppt die Zeit und der aktuelle Zustand des Realteils kann als statisches Bild betrachtet werden.

Über den Button 'Zurücksetzen' kann man alle Änderungen an der Wellenfunktion und dem Potential verwerfen und zu dem Startzustand des Programms zurückkehren. Das Potential hat dann wieder den Wert 0 und die Wellenfunktion nimmt den Grundzustand an.

4.2 Die Norm des Wellenvektors

In dem Infofeld am oberen, linken Rand wird die Norm der aktuellen Wellenfunktion angezeigt. Sie wird bei jeder Neuberechnung des numerischen Vektors laufend aktualisiert. Im Normalbetrieb ist zu beobachten, dass diese bei laufender Simulation leicht abfällt aber nicht deutlich vom Wert 1 abweicht. Um den Vektor wieder zu normieren kann man einen entsprechend beschrifteten Button im Parameterfenster anklicken oder die Taste 'n' betätigen.

4.3 Setzen der physikalischen Parameter

Die Berechnung der Wellenfunktion erfolgt über das in Abschnitt 3.1.3 beschriebene Runge-Kutta-Verfahren. Die dabei verwendeten Werte der Länge des Kastens L, der Anzahl der numerischen Raumpunkte des Vektors N_r, die Masse des Teilchens m und

das Zeitintervall zwischen zwei numerischen Zeitpunkten δ_t lassen sich in atomaren Einheiten über das Parameterfenster durch entsprechend beschriftete Eingabefelder angeben. Außerdem kann man die Anzahl der numerischen Zeitschritte N_f auswählen, die für einen einzelnen Frame (Also einem der Bilder aus denen die Animation besteht) berechnet werden. Die Werte für den Abstand zwischen zwei numerischen Raumpunkten δ_x und der Zeit pro Frame T_f ergeben sich indirekt aus:

$$\delta_x = \frac{L}{N_r - 1} \tag{4.1}$$

$$T_f = \delta_t N_f \tag{4.2}$$

Wenn man die Einstellungen dieser Felder ändert, werden sie automatisch auf die Wellenfunktion angewendet, wobei sie ihre vorherige Form beibehält, sofern das die Wahl der Ortsauflösung erlaubt. Eine Änderung der Kastenlänge staucht und streckt die bisherige Wellenfunktion, sodass sie weiterhin den gesamten Kasten ausfüllt. Weil die Funktionswerte dabei nicht geändert werden, verliert die Funktion bei diesem Vorgang ihre Norm. Außerdem wird bei einer Änderung von L das bisherige Potential nicht mitgestreckt, weshalb es sich im Verhältnis zur Wellenfunktion verschiebt. Auch der Bereich auf dem das Potential gezeichnet werden kann bleibt unverändert, was dazu führen kann, dass es auf der rechten Seite keinen anderen Wert als 0 annimmt.

Um eine fehlerfreie Verwendung des Programms bei geänderter Kastenlänge zu gewährleisten, wird deshalb empfohlen nach einer solchen Änderung den Button "Zurücksetzen" zu betätigen. Dadurch wird die Wellenfunktion in den Grundzustand und das Potential auf 0 gesetzt. Die Länge des Kastens und die anderen gewählten Parameter werden jedoch beibehalten und das Programm sollte fehlerfrei funktionieren.

Man wird feststellen, dass bei manchen Einstellungen der hier beschriebenen Parameter die Wellenfunktion bei laufender Simulation nicht stabil berechnet wird sondern ein immer stärker werdendes Rauschen ausbildet und kurz darauf nicht mehr angezeigt wird. Die angezeigte Norm wird dabei immer größer und nimmt schließlich den Wert `NaN` (Not a Number) an. Nach einem Neustart des Programms kann man das beispielsweise durch Einstellen von δ_t auf 2 erreichen. Dieses Verhalten entsteht durch eine Instabilität der numerischen Berechnung, die in Kapitel 5 genauer untersucht wird. Auch in diesem Fall kann man das Programm anschließend zurücksetzen, um zum Anfangszustand zurückzukehren.

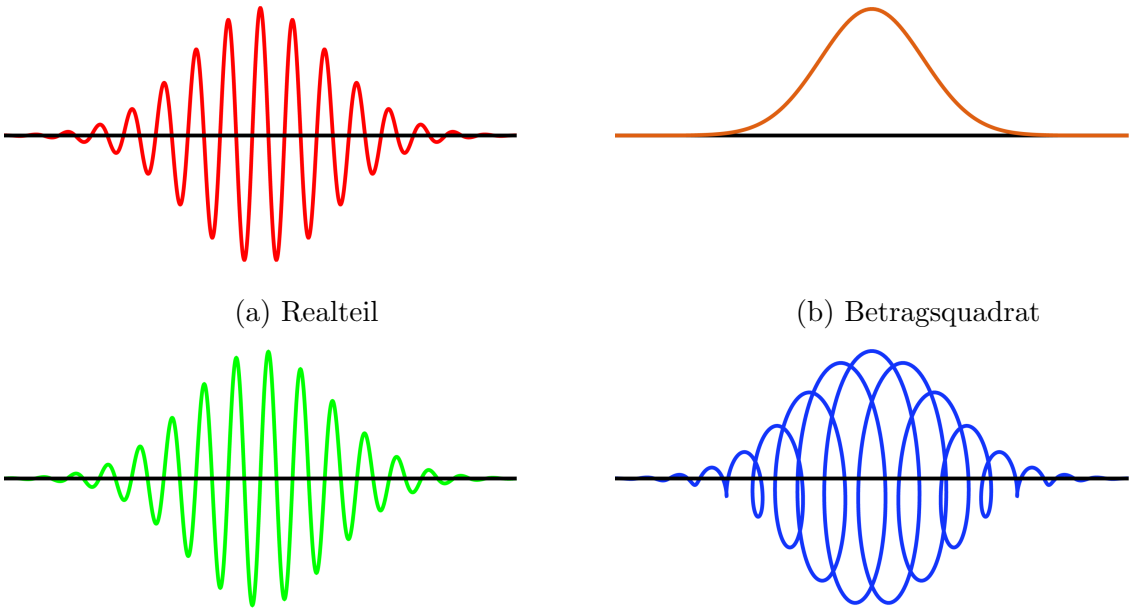
4.4 Verschiedene Darstellungen

Die Darstellung der Wellenfunktion und des Potentials lässt sich durch das Parameterfenster anpassen. Da das Potential als reelle Funktion in einer Dimension problemlos dargestellt werden kann gibt es hierfür nur einen Anzeigemodus in dem sie als einfacher Graph abgebildet ist. Die Wellenfunktion hingegen kann als komplexe Funktion in Abhängigkeit einer reellen x Koordinate nicht so einfach in zwei Dimensionen dargestellt werden. Für sie stehen vier verschiedene Darstellungsmodi zur Verfügung. Jeder plottet die komplexe Wellenfunktion auf eine andere Weise.

Über das Parameterfenster kann sowohl für jede dieser vier Darstellungsarten als auch für das Potential über eine entsprechend beschriftete Checkbox gewählt werden, ob der dazugehörige Plot angezeigt wird. Außerdem lässt sich über eine daneben platzierte Farbauswahl die Farbe des Plots festlegen. Die Strichstärke lässt sich für alle vier Plots

des Wellenvektors und dem des Potentials gemeinsam über einen Stellregler einstellen. Die vier Darstellungsmodi der Wellenfunktion sind:

- Der Realteil der Wellenfunktion wird in Abhängigkeit des Ortes aufgetragen.
- Der Imaginärteil der Wellenfunktion wird in Abhängigkeit des Ortes aufgetragen.
- Das Betragsquadrat der Wellenfunktion wird in Abhängigkeit des Ortes aufgetragen. Das entspricht der Aufenthaltswahrscheinlichkeit des Teilchens.
- Eine Parallelprojektion der dreidimensionalen Komplexen Wellenfunktion auf den zweidimensionalen Bildschirm wird angezeigt. Dieser Modus soll im Folgenden genauer beschrieben werden:



(a) Realteil (b) Betragsquadrat

(c) Imaginärteil (d) 3D-Ansicht

Abbildung 4.2: Ein Vergleich der Vier Darstellungsmodi der Wellenfunktion am Beispiel eines Gaußschen Wellenpakets

4.5 Der 3D-Modus

Der 3D-Modus eignet sich gut um eine intuitive Vorstellung davon zu erlangen, wie sich der komplexe Funktionswert der Wellenfunktion in Abhängigkeit des Ortes ändert. Dargestellt ist eine Parallelprojektion eines dreidimensionalen Kurvenverlaufs welcher die komplexen Funktionswerte als Punkte in zwei Dimensionen auffasst, die wiederum in Abhängigkeit des Ortes als dritte Dimension zu einer Kurve zusammengefügt sind. Das ist so realisiert, dass jedes Tripel $x, \mathrm{Re}(\Psi(x)), \mathrm{Im}(\Psi(x))$ aus der X-Position und den davon abhängigen Real- und Imaginärteilen der Wellenfunktion ein Punkt $P = (P_x, P_y)$ auf dem zweidimensionalen Bildschirm zugeordnet wird. All diese Punkte werden dann in der Reihenfolge der X-Werte durch Linien miteinander verbunden. Konkret erfolgt diese Zuordnung auf folgende Weise:

$$P_x = x + AK\,\mathrm{Im}(\Psi(x)) \tag{4.3}$$

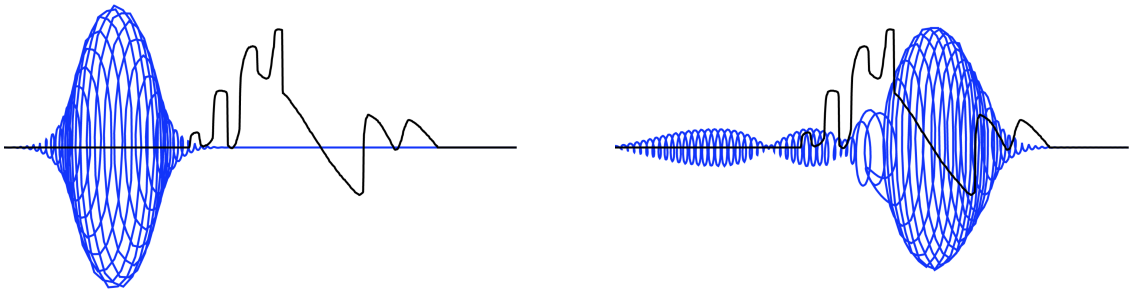
$$P_y = A\,\mathrm{Re}(\Psi(x)) \tag{4.4}$$

A ist dabei der Skalierungskoeffizient der die Amplitude der dargestellten Wellenfunktion regelt, und K gibt die perspektivische Verkippung an.

Bei einem Neustart ist das Programm so voreingestellt, dass das Potential zusammen mit der 3D-Projektion der Wellenfunktion abgebildet wird. Weil die Perspektivische Verkippung jedoch auf den Wert 0 gesetzt wird ist der dargestellte Kurvenverlauf der 3D-Ansicht nicht von dem des Realteils zu unterscheiden. Über die Tasten 'A' und 'D' kann der die Verkippung innerhalb des Intervalls $[-0.5, 0.5]$ verschoben werden. Wenn man bei angehaltener Simulation die 3D-Projektion der Wellenfunktion in verschiedene Richtungen verkippt, kann man eine gute Vorstellung ihrer räumlichen Struktur gewinnen.

Eine weitere Navigationsmöglichkeit, die diesen dreidimensionalen Eindruck ergänzen kann, ist die Rotation der Wellenfunktion in der komplexen Zahlenebene. Diese kann über die Tasten 'W' und 'S' durchgeführt werden. Weil sich eine Rotation der komplexen Phase eines quantenmechanischen Zustands in diesem Szenario nicht auf seine messbaren Eigenschaften auswirkt, nimmt man dadurch keinen Eingriff in die Verhaltensweise des Teilchens vor.

Weil das menschliche Gehirn auf natürliche Weise darauf spezialisiert ist die Struktur von dreidimensionalen Objekten zu erfassen welche im Raum gedreht werden, ist eine Navigation mit diesen vier Tasten gut geeignet um die Eigenschaften eines komplexen Wellenvektors zu untersuchen. Beachtet werden sollte jedoch, dass es in dieser Darstellung zwei verschiedene räumliche Interpretationen des Szenarios gibt die beide gleichwertig sind. Wechselt man von einem dieser Szenarien in das andere entspricht das einer Spiegelung der in den Bildschirm hinein verlaufenden Z-Achse. Verwirrend kann das werden, wenn man beide Interpretationen gleichzeitig auf verschiedene Teilbereiche des Vektors anwendet. Als Hilfestellung kann man eine dieser Interpretationen auszeichnen indem man definiert dass die Tasten 'W','A','S' und 'D' den perspektivisch vorderen Teil der Wellenfunktion wie bei der Anordnung auf der Tastatur nach oben, links, unten und rechts verschieben. In dieser Interpretation sind Funktionswerte mit positivem Imaginärteil weiter vom Betrachter entfernt als solche mit negativem.



(a) Vorher (b) Nachher

Abbildung 4.3: Ein Gaußsches Wellenpaket in der 3D-Darstellung vor und nach der Wechselwirkung mit einem Potential, dass vom Nutzer gezeichnet wurde.

4.6 Orientierung im Koordinatensystem

Das Programm stellt keine Achsenbeschriftung zur Verfügung. Jedoch ist es möglich, Informationen über die Abmessungen des dargestellten Koordinatensystems zu erhalten indem man den Mauszeiger über den Bildschirm bewegt. Das Informationsfeld in der

oberen linken Ecke stellt neben der aktuellen Zeit und der Norm der Wellenfunktion auch die X- und Y-Koordinaten des Mauszeigers dar. Da Die Plots des Potentials und des Betragsquadrates jeweils unterschiedliche Einheiten und somit auch eine von den Restlichen Plots abweichende Skalierung der Y-Achse besitzen gibt es drei verschiedene, entsprechend beschriftete Angaben für die Y-Koordinate des Mauszeigers, die nur angezeigt werden wenn der dazugehörige Plot sichtbar ist.

Wenn man die Wellenfunktion stark lokalisiert, kann es passieren, dass die auf 1 normierte Wellenfunktion an manchen Orten Werte annimmt, die so groß oder klein sind, dass sie nicht mehr auf dem Bildschirm dargestellt werden können. Dieser Fall kann zum Beispiel bei einer kleinen Wahl von L oder auch bei einem stark fokussierenden Potential eintreten. Andersherum kann es bei einer großflächigen Verteilung der Aufenthaltswahrscheinlichkeit vorkommen, dass die Funktionswerte so klein werden, dass sie kaum noch von 0 unterscheidbar sind. Das selbe gilt für die Graphen des Betragsquadrats und des Potentials.

Um die Graphen dennoch mit maximalem Detailreichtum abbilden zu können, kann man im Parameterfenster den Wert des Eingabefeldes mit der Aufschrift "Wellenfunktion Vergrößerung" anpassen. Wenn dieser den Wert F annimmt, dann ist der dargestellte Wertebereich des dazugehörigen Plots $[-\frac{1}{F}, \frac{1}{F}]$. Weil es drei verschiedene Y-Achsen-Skalierungen gibt (die des Potentials, des Betragsquadrats und der restlichen Plots) gibt es zu jeder auch einen eigenen Stellregler des dargestellten Abschnitts.

4.7 Glätten der Wellenfunktion

Ein praktisches Werkzeug um eine komplizierte Wellenfunktion mit hochfrequenten Anteilen zu vereinfachen ist die Glätten-Funktion. Über einen Button mit dieser Aufschrift kann man hohe Frequenzen in der Wellenfunktion entfernen.

Das ist zum Beispiel dann praktisch, wenn man während der Simulation viele Änderungen am Potential vorgenommen hat, was der Wellenfunktion Anteile mit sehr hoher Energie verliehen haben, die man nun beseitigen möchte. Das Werkzeug zerlegt die Wellenfunktion innerhalb des Kastens in eine Fourier-Reihe [23] und setzt sie dann aus dieser wieder zusammen. Dabei berücksichtigt es nur die Fourier-Koeffizienten bis zur einer bestimmten Ordnung Q. Diese kann man über ein Eingabefeld mit der Bezeichnung 'Glätten Ordnung' festlegen.

Wählt man den Wert klein so bleiben bei einer Glättung nur sehr grobe Strukturen erhalten. Bei einer hohen Ordnung Q sind weiterhin viele feinere Details in der Wellenfunktion sichtbar. Weil bei diesem Vorgang Anteile des Wellenvektors fallen gelassen werden, bleibt die Norm nicht erhalten, sondern wird kleiner. Aus diesem Grund wird der Wellenvektor nach der Glättung automatisch neu normiert.

4.8 Unterschiedliche Werkzeuge zur Manipulation mit der Maus

Neben dem einfachen Zeichnen des Potentials mit der Maus gibt es noch weitere Möglichkeiten mit Hilfe des Mauszeigers auf die Graphen einzuwirken. Um diese Nutzen zu können kann man in einer Auswahlbox im Parameterfenster den Stifttyp ändern. Standardmäßig ausgewählt ist die bereits beschriebene Funktion das Potential zu zeichnen. Wählt man die Option 'Potential Linie zeichnen' so kann man gerade Linien in das Potential zeichnen, welche die beiden Punkte verbinden an denen man dem Mauszeiger

gedrückt und losgelassen hat. Dieses Werkzeug ist zum Beispiel für das Zeichnen von Potentialtöpfen praktisch.

Ein Tool mit deutlich vielseitigeren Einsatzmöglichkeiten wird durch die Auswahlmöglichkeit 'Wellenfunktion zeichnen' bereitgestellt. Ist dieses ausgewählt, so kann man direkt den Realteil der Wellenfunktion zeichnen. Es wird empfohlen die Simulation dafür anzuhalten, wobei es grundsätzlich auch bei laufender Berechnung der Zeitentwicklung möglich ist. Weil beim Zeichnen scharfe Kanten entstehen können, welche dem Wellenvektor bei Anwendung der numerischen Schrödingergleichung hohe Frequenzen verleihen, die möglicherweise unerwünscht sind wird nach dem Loslassen der Maustaste die bereits beschriebene 'Glätten'-Funktion auf den Vektor angewandt. Wenn man nicht will, dass diese Kanten nach dem Zeichnen entfernt werden kann man das Eingabefeld 'Glätten Ordnung' auf denselben Wert stellen wie die Anzahl der Raumgitterpunkte. Auf diese Weise werden hohe Frequenzen nicht entfernt und man kann die Auswirkungen scharfer Kanten auf die numerische Berechnung beobachten.

4.9 Die Anwendung benutzerdefinierter Formeln

Das Programm bietet eine sehr vielseitige Möglichkeit den Vektor und das Potential über mathematische Formeln zu definieren oder zu transformieren. Dazu kann das große Textfeld am unteren Rand des Parameterfensters genutzt werden. Die Syntax, welche man dabei verwenden muss, ist die eines Javascript-Programms.

Um das Potential neu zu setzen, muss man der Variable 'V' einen neuen Wert zuweisen. Dieser kann von Variablen, wie zum Beispiel dem Ort 'X', der Masse 'M' und der Kastenlänge 'L' abhängen. Auch die Definition eines neuen Wellenvektors ist möglich. Weil Javascript standardmäßig keine komplexen Zahlen unterstützt muss man den Real und Imaginärteil separat durch reelle Zahlen setzen. Um den Vektor neu zu definieren muss man der Variable des Realteils 'R' und Imaginärteils 'I' der Wellenfunktion einen Wert geben. Dabei sollte man darauf achten, dass beide Anteile an den Rändern des Kastens den Wert 0 annehmen. Neben den bereits genannten Variablen kann über 'alt_V', 'alt_R' und 'alt_I' auch auf den vorherigen Wert der Wellenfunktion und des Potentials zugegriffen werden.

Über die Auswahl neben dem Textfeld kann eine der Beispielformeln geladen werden. Diese kann man dann gegebenenfalls auch anpassen. Eine einfache Formel zur Definition des Potentials eines Harmonischen Oszillators wäre also gegeben durch:

```
V = 0.000001 * (X - L / 2) * (X - L / 2);
```

Neben '+', '-', '*' und '/' stehen auch weitere mathematische Operatoren wie Sinus, Cosinus, die Exponentialfunktion und der Betrag über die Javascript-Schnittstelle zur Verfügung und können auf Folgende weise realisiert werden: '`Math.sin()`', '`Math.cos()`', '`Math.exp(,)`' und '`Math.abs()`'. Jede Zuweisung einer Variable muss mit einem Semikolon ';' beendet werden. In diesem Formelfeld kann ein Großteil der Funktionalität der Javascript-Programmiersprache verwendet werden auch Bedingungen oder Programmschleifen sind verfügbar.

Wenn die gewünschte Formel in das Textfeld eingetragen ist kann über den Button 'Formel Anwenden' der Code ausgeführt werden. Wenn eine oder mehrere der Variablen 'V', 'R', 'I' in der Formel gesetzt werden wird das Potential und die Wellenfunktion entsprechend verändert. Geschieht das nicht bleiben sie unverändert. Die automatische Aktualisierung des Potentials oder der Wellenfunktion durch eine zeitabhängige Formel ist nicht implementiert.

4.10 Navigation durch die Zeit

Wie bereits erwähnt, kann die Simulation über die Leertaste angehalten werden. Für eine präzisere Navigation durch die Zeit kann anschließend die Taste 'l' verwendet werden um die Simulation für einzelne Frames fortzusetzen. Man kann die Taste auch gedrückt halten, um die Simulation für mehrere Frames nach vorne laufen zulassen. Analog kann die Taste 'j' genutzt werden wobei das Runge-Kutta-Verfahren hier nicht für die Zeitspanne δ_t sondern für $-\delta_t$ angewendet wird. Damit berechnet man also aus dem aktuellen einen früheren Wellenfunktionszustand. Die Tasten 'l' und 'j' können also genutzt werden um in der Zeit vor und zurück zu navigieren. Auch die angezeigte Zeit wird entsprechend aktualisiert.

4.11 Das zeitabhängige Potential

Eine praktische Eigenschaft des Programms ist, dass es sich den zeitlichen Verlauf des Potentials merkt. Das geschieht über eine Speicherung seines Zustandes bei allen Keyframes an denen der Nutzer auf es eingewirkt hat. Die Speicherung geschieht sowohl für Potentialänderungen bei laufender als auch bei angehaltener Simulation.

Hat der Nutzer also im zeitlichen Verlauf Änderungen am Potential vorgenommen und navigiert nun mit Hilfe der Taste 'j' oder durch eine Einstellung des Wertes δ_t auf einen negativen Wert rückwärts durch die Zeit so kann er beobachten wie die am Potential vorgenommenen Änderungen in umgekehrter Reihenfolge ablaufen. Im Optimalfall sollte die Wellenfunktion dadurch wieder in ihren ursprünglichen Zustand zurückkehren. Beachtet werden sollte, dass bei jeder Erzeugung eines Keyframes die neu definierte Potentialform für den Zeitraum bis zum zeitlich darauf folgenden Keyframe gilt. (Dies gilt nicht für den linearen Modus. Dazu später mehr.) Wenn man die Simulation also erst rückwärts laufen lässt, dass Potential ändert, und die Simulation anschließend wieder vorwärts laufen lässt, wird man nicht zum Ursprünglichen Zustand zurückkehren weil weiterhin das neu gesetzte Potential gültig ist. Für alle Zeitpunkte vor dem ersten und nach dem letzten Keyframe ist das Potential weiterhin durch deren Werte definiert.

Vorsichtig sollte man auch sein wenn man das Potential in einem Zeitraum ändert in dem bereits Keyframes gesetzt sind. Wenn man es an der exakt selben Stelle eines bereits bestehenden Keyframes abändert überschreibt man diesen. Ansonsten sind die alten Keyframes weiterhin gültig.

Um den Änderungsverlauf des Potentials besser bearbeiten und nachvollziehen zu können kann man mit den Tasten 'i' und 'k' zum nächsten bzw. vorherigen Keyframe springen. Dabei wird jedoch nicht die Zeitentwicklung des Teilchens berechnet. Mit der Taste 'x' kann der aktuelle Keyframe entfernt werden. Bei einem Neustart des Programms ist das Potential durch einen einzigen Keyframe beim Zeitpunkt 0 definiert. Zu diesem Zustand kann auch jederzeit mit Hilfe des 'Zurücksetzen'-Buttons zurückgekehrt werden. Durch das Setzen des Hakens in der Checkbox 'Lineare Potentialänderung' im Parameterfenster, kann man zum linearen Modus für die Änderung des Potentials wechseln, der in Gleichung 3.2 beschrieben wird. Dieser kommt nun bei allen Keyframes zum Einsatz, welche neu gesetzt werden während die bereits existierenden weiterhin ihren Modus beibehalten. Für jedes Zeitintervall zwischen zwei Keyframes wird der Modus angewandt, der dem späteren Keyframe zugeordnet ist. Mit dem linearen Modus kann man auf einfache Weise auch Potentiale erzeugen, die sehr langsam ihre Form in eine Andere verändern. Das kann hilfreich sein, wenn man den Zustand bei einer Potentialänderung möglichst wenig anregen will.

Ein Nachteil an der Speicherung des Potentials ist, dass das Programm während der Simulation für jede Änderung am Potential Speicherplatz belegt. Wenn man davon viel Gebrauch macht ohne das Potential zwischenzeitlich zurückzusetzen ist der vorhandene Speicherplatz irgendwann aufgebraucht und es können keine neuen Keyframes mehr gesetzt werden. In diesem Fall wird ein Neustart durch Zurücksetzen empfohlen.

4.12 Speichern und Laden

Das Programm implementiert eine Möglichkeit die aktuelle Sitzung als eine binäre Datei abzuspeichern und zu einem späteren Zeitpunkt wieder zu laden. Beides lässt sich über einen entsprechend beschrifteten Button durchführen. Über ein Auswahlfeld kann man sich entscheiden, ob man die gesamte Sitzung, nur den Vektor oder nur das Potential in der Datei abspeichern will. Lädt man eine Datei, die nur eine der beiden Komponenten enthält so wird nur diese aus der Datei geladen und überschrieben, während die andere in ihrem vorherigen Zustand beibehalten wird. Das kann man ausnutzen um den Vektor und das Potential aus verschiedenen Projekten zusammenzubringen.

Das Potential wird immer als vollständiger zeitlicher Verlauf abgespeichert. Das Abspeichern einzelner Keyframes und das Kombinieren der Keyframes aus verschiedenen Projekten ist nicht implementiert. Bei der Wahl der Option 'Alles Speichern' wird der Vektor, das Potential und die aktuelle Zeit in eine gemeinsamen Datei geschrieben. Die meisten Einstellungen im Parameterfenster werden jedoch nicht abgespeichert.

Kapitel 5

Code-Validierung

Wenn man die numerische Berechnung startet, beginnt die Wellenfunktion sich zu verändern. Je nach Wahl der Parameter, ist das Ergebnis aber unterschiedlich überzeugend. Manche Einstellungen für δ_t und δ_x führen dazu, dass bereits nach wenigen Iterationen ein zunehmendes Auseinanderdriften der Funktionswerte benachbarter Gitterpunkte beginnt. Der Betrag der Wellenfunktion beginnt exponentiell zu wachsen und nach kurzer Zeit wird die numerische Funktion nicht mehr angezeigt, weil einige Funktionswerte so stark steigen, dass die 64-Bit Gleitkommazahlen sie nicht mehr darstellen können. Ein frühes Stadium einer solchen Divergenz ist in Abb. 5.1 gezeigt. Dieses Verhalten der Wellenfunktion ist eindeutig nicht physikalisch korrekt. In diesem Abschnitt soll untersucht werden, wie sehr die vom Programm berechneten Lösungen mit den analytisch korrekten Lösungen übereinstimmen und welchen Einfluss die Wahl der Parameter darauf hat.

5.1 Das Teilchen im unendlich hohen Kastenpotential

Um die numerischen Ergebnisse zu überprüfen, muss zunächst ein Potential gewählt werden, für das die analytische Lösung bekannt ist. Beispiele hierfür sind der harmonische Oszillator und das Colomb-Potential. Auch der unendlich hohe Potentialtopf kommt dafür in Frage [24]. Letzterer ist besonders geeignet, weil es die Voraussetzung des Programms an das Potential ist, dass dieses nur auf einem endlichen Intervall endliche Werte annimmt. Im Folgenden wird also das Konvergenzverhalten der numerischen Berechnung im unendlich hohen Kastenpotential untersucht. Man wähle das Potential

$$V_{\mathrm{K}}(x) = \begin{cases} \infty & x < 0 \\ 0 & x \in [0, L] \\ \infty & x > L \end{cases} \tag{5.1}$$

Die analytischen Lösung zu der zugehörigen Schrödingergleichung sind alle Superpositionen der Eigenvektoren

$$\Psi_k(x) := \begin{cases} 0 & x < 0 \\ C \sin(\frac{k\pi x}{L}) & x \in [0, L] \\ 0 & x > L \end{cases} \tag{5.2}$$

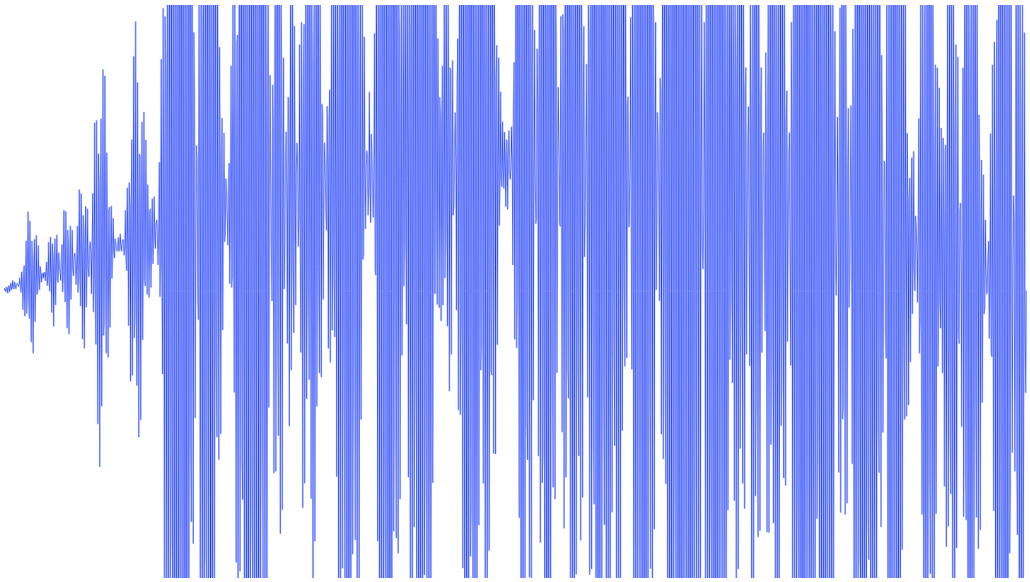


Abbildung 5.1: Ein Ausschnitt des Realteils einer im Grundzustand präparierten Wellenfunktion nach einer instabilen numerischen Berechnung der zeitabhängigen Schrödingergleichung ($L = 1000$, $N_r = 1000$, $\delta_t = 2$)

C ist dabei eine Konstante, die zur Normierung der Funktion auf den Wert 1 benötigt wird.

$$|C| = \sqrt{\frac{2}{L}} \tag{5.3}$$

Die Funktionen Ψ_k, die an den Punkten $x = 0$ und $x = L$ den Wert 0 annehmen und damit stetig jedoch nicht stetig differenzierbar sind, erfüllen auf dem Intervall $[0, L]$ die Eigenwertgleichung des Energieoperators:

$$H\Psi_k(x) = -\frac{\hbar^2}{2m}\Delta\Psi_k(x) = \frac{1}{2m}\left(\frac{\hbar k\pi}{L}\right)^2 \Psi_k(x) \tag{5.4}$$

Die Zeitentwicklung ist damit gegeben durch:

$$\Psi_k(x,t) = e^{-it\frac{\hbar}{2m}(\frac{k\pi}{L})^2}\Psi_k(x) \tag{5.5}$$

Es soll nun der Vektor $\Psi(x,t)$ aus einer Superposition der Ψ_k bis hin zur Ordnung Q gebildet werden, anhand dessen ein Vergleich der analytischen und numerischen Lösung vorgenommen werden kann.

$$\Psi(x,t) = \sum_{k=1}^{Q} c_k \Psi_k(x,t) \tag{5.6}$$

Durch die komplexen Koeffizienten c_k ist er eindeutig definiert. Sie müssen so gewählt sein, dass die Norm von Ψ den Wert 1 annimmt:

$$\sum_{k=1}^{Q} |c_k|^2 = 1 \tag{5.7}$$

Es wird nun der numerische Startvektor Ψ_0 gebildet indem Ψ zum Zeitpunkt $t = 0$ ausgewertet wird. Der numerische Vektor setzt sich wie in Abschnitt 3.1.1 beschrieben

aus N_r komplexen Zahlen p_n an den Orten x_n zusammen und soll nun über den Zeitraum T in $N_t - 1$ Iterationen geändert werden. In jeder Iteration wird das in Abschnitt 3.1.3 beschriebene Runge-Kutta-Verfahren jeweils für den Zeitraum von $\delta_t = 1/(N_t - 1)$ angewandt. Die komplexen Zahlen p_n hängen also von der Zeit t ab, welche, beginnend bei 0, um diskrete Zeitintervalle der Länge δ_t erhöht werden.

5.2 Der Fehler Θ

In diesem Abschnitt soll die Größe Θ definiert werden, welche ein Maß für die Abweichung der numerischen von der analytischen Lösung sein soll.

Zunächst kann dafür zu jedem $p_n(t)$ die Abweichung $D_n(t)$ zu dem, von der analytischen Lösung geforderten, Funktionswert berechnet werden:

$$D_n(t) = |p_n(t) - \Psi(x_n, t)| \tag{5.8}$$

Die in Gleichung 3.14 beschriebene numerische Integrationsmethode erster Ordnung soll nun genutzt werden um diese Abweichung über den gesamten Vektor zu integrieren:

$$\theta(t) = \delta_x \left(\sum_{n=2}^{N_r - 1} D_n(t) + \frac{D_1(t)}{2} + \frac{D_{N_r}(t)}{2} \right) \tag{5.9}$$

Der Wert θ kann nur bedingt als Maß dafür genutzt werden, wie gut die Übereinstimmung zwischen der analytischen und numerischen Lösung ist weil er von der Normierung der Wellenfunktion abhängt. An folgendem Gedankenexperiment wird deutlich, inwiefern das eine Rolle spielt:

Streckt man sowohl die numerischen als auch die analytische Wellenfunktion so, dass sie in einen Kasten von doppelter Länge $2L$ passen, so müssen die Funktionswerte und damit auch die Abweichungen $D(x)$, mit $1/\sqrt{2}$ multipliziert werden, damit die Normierung erhalten bleibt. Weil δ_x jetzt doppelt so groß ist, ändert sich θ folgendermaßen: (Das Integralzeichen steht hier für die numerische Integration)

$$\theta_{\text{neu}} = \int_0^{2L} \frac{D(\frac{x}{2})}{\sqrt{2}} dx = \sqrt{2} \int_0^{L} D(x) dx = \sqrt{2}\theta \tag{5.10}$$

Würde man θ als ein Maß für den Fehler der numerischen Wellenfunktion verwenden würde sich dieser verändern, wenn man die X-Achse anders skaliert. Um die Abhängigkeit von der Skalierung zu eliminieren wird θ durch das numerische Integral über den Betrag der analytischen Wellenfunktion Ψ geteilt. Das Ergebnis Θ wird als Maß für den Fehler der numerischen Lösung verwendet.

$$\Theta(t) = \frac{\theta(t)}{\delta_x \left(\sum_{n=2}^{N_r - 1} |\Psi(x_n, t)| + \frac{|\Psi(x_1, t)|}{2} + \frac{|\Psi(x_{N_r}, t)|}{2} \right)} \tag{5.11}$$

Θ sagt also aus in welchem Verhältnis das Integral über die Abweichung von numerischer und analytischer Lösung zu dem Integral über den Betrag der analytischen Wellenfunktion steht. Ist Θ wesentlich kleiner als 1 (aber per Definition natürlich niemals negativ), dann bedeutet das, dass der Betrag der Differenz zwischen numerischer und analytischer Lösung durchschnittlich eine viel kleinere Größenordnung hat als der Betrag der dazugehörigen analytischen Funktionswerte. Die numerische Lösung zeigt also eine gute Übereinstimmung mit dem physikalisch erwarteten Zustand des Wellenvektors. Je kleiner

Θ ist desto besser ist diese Übereinstimmung. Hat Θ den Wert 0 so stimmen die p_n exakt mit den erwarteten Funktionswerten der analytischen Lösung an den Orten x_n überein.

Wenn Θ nahe bei 1 liegt bedeutet das, dass die Abweichung des numerischen Vektors von dem analytischen Vektor ungefähr so groß ist wie der mittlerer Betrag des Letzteren. Es lässt sich also keine Übereinstimmung der beiden Vektoren zeigen. Nimmt Θ Werte an, die deutlich größer als 1 sind, so bedeutet das, dass die durchschnittliche Abweichung des numerischen von dem analytischen Vektor viel größer ist als die mittleren Beträge seiner Funktionswerte. Es ist dann also anzunehmen, dass die Normierung des numerischen Vektors in diesem Fall verletzt ist und dass dessen Zustand nicht mehr geeignet ist um eine physikalische Teilchenwelle zu beschreiben.

5.3 Erzeugung des Startvektors

Die Koeffizienten des Startvektors c_n sollen zufällig gewählt werden, damit die Konvergenz der numerischen Berechnung an einem Beispielvektor, welcher eine nicht-triviale zeitliche Entwicklung vollzieht, überprüft werden kann. Der Koeffizient der höchsten Ordnung c_Q soll einen überdurchschnittlich großen Betrag haben, damit der Startvektor klar als ein Vektor der Ordnung Q identifizierbar ist.

Die Erzeugung des Startvektors soll außerdem reproduzierbar sein. Um das zu erreichen, wird eine einfache Formel verwendet, um die realen und imaginären Anteile von c_n für $n \in \{1 \dots Q-1\}$ auf einen pseudo-zufälligen Wert zwischen -1 und 1 zu setzen. c_Q wird auf 1 gesetzt. Der so erzeugte Vektor wird anschließend normiert.

5.4 Die Wahl der Parameter

Bevor die Berechnungen gestartet werden, wird eine Wahl für die Parameter der Kastenlänge L, der Teilchenmasse m, der Ordnung Q sowie für die Zahl der Raumpunkte N_r, der Zeitpunkte N_t und des Zeitintervalls T, über das die Simulation stattfinden soll, getroffen werden. Diese Wahl soll sich für die numerische Berechnung eignen und als Orientierung für eine Variation der Parameter dienen. Das Zeitintervall T wird auf den Wert 1000 gesetzt. Die Masse des Teilchens wird auf 1 festgelegt, was der Masse eines Elektrons entspricht. Die Ordnung des höchsten Eigenwertes, welcher im Startvektor enthalten ist, wird auf 50 festgelegt. Bei einer Bildschirmbreite von etwa 2000 Pixeln sind so einzelne Wellenberge deutlich erkennbar und der Vektor hat dennoch genug Freiheitsgrade, um über eine gewisse Komplexität zu verfügen.

Mit der Wahl des Raumintervalls wird die Bewegung eines Startvektors innerhalb des Zeitintervalls nun eindeutig festgelegt. Sie sollte so getroffen werden, dass sich das Wellenpaket deutlich sichtbar verändert aber nicht zu viele Schwingungen durchläuft, weil sonst für die Darstellung der zeitlichen Entwicklung mehr numerische Zeitschritte benötigt werden, was den Rechenaufwand steigert. Das Raumintervall erhält einen Wert von 1000. Damit vollzieht der Eigenvektor mit Ordnung 50 innerhalb des betrachteten Zeitraums fast 2 Umdrehungen in der komplexen Zahlenebene.

Die Anzahl der numerischen Raum- und Zeitschritte werden jeweils auf 1000 festgelegt. Damit erhält sowohl δ_x als auch δ_t den Wert 1. Die dadurch entstehende Anzahl an Rechenschritten kann von einem Computer noch innerhalb einer Sekunde berechnet werden und die resultierende numerische Teilchenwelle divergiert nicht und weicht auch nicht deutlich sichtbar von der analytischen Lösung ab.

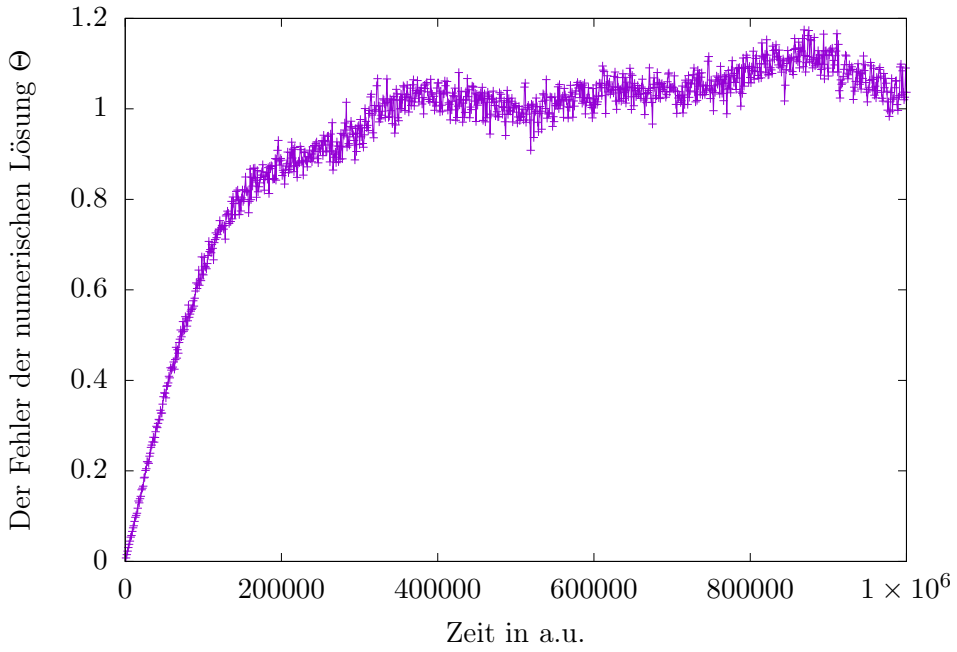


Abbildung 5.2: Entwicklung des Fehlers Θ im Verlauf der Zeit bei der nicht divergierenden numerischen Berechnung einer Wellenfunktion mit den Parametern $L = 1000$, $\delta_t = 1$, $\delta_x = 1$ und $Q = 50$

5.5 Durchführung der Codevalidierung

Weil der Fehler der numerischen Wellenfunktion in Abhängigkeit unterschiedlicher Parameter untersucht werden soll, wird ein Verfahren entwickelt um diesen automatisiert zu berechnen. Dazu wird eine leicht abgewandelte Version des Hauptprogramms verwendet, welche die Simulation der Teilchenwelle über ein gegebenes Zeitintervall ausgehend von dem pseudo-zufälligen Startvektor für verschieden gesetzte Parameter wiederholt. Parallel dazu berechnet diese Programmversion die analytische Lösung und vergleicht sie mit dem numerischen Vektor. Die Ergebnisse werden dabei in eine Datei geschrieben, und anschließend von dem Programm 'Gnuplot' gelesen und graphisch dargestellt.

5.6 Die Zeitentwicklung im numerisch stabilen Fall

Für die Parameter $L = 1000$, $\delta_t = 1$, $\delta_x = 1$ und $Q = 50$ wird die Entwicklung der Abweichung Θ im Verlauf des Zeitintervalls 10^6 betrachtet. Die Entwicklung von Θ wird in Abb. 5.2 dargestellt.

Es ist erkennbar, dass die Abweichung im Verlauf der Zeit anfangs in guter Näherung linear zunimmt. Etwa ab dem Wert $\Theta = 0.7$ wird die Steigung flacher und beginnt dabei zunehmend stärkere Unregelmäßigkeiten auszubilden. Der Kurvenverlauf pendelt sich anschließend unter weiteren Schwankungen etwa um den Wert 1 herum ein. Sobald Θ zum ersten mal Werte nahe der 1 annimmt ist kaum noch eine Übereinstimmung zwischen der numerischen und analytischen Wellenfunktion feststellbar. Der weitere Kurvenverlauf ist nicht mehr von Bedeutung weil er die Abweichung zweier komplett unterschiedlicher Wellenfunktionen voneinander beschreibt. In Abbildung 5.3 ist das Quadrat der Norm des gleichen numerischen Vektors im Verlauf des selben Zeitraumes dargestellt.

Die Abbildung zeigt eine lineare Abnahme der quadratischen Norm. Beginnend mit

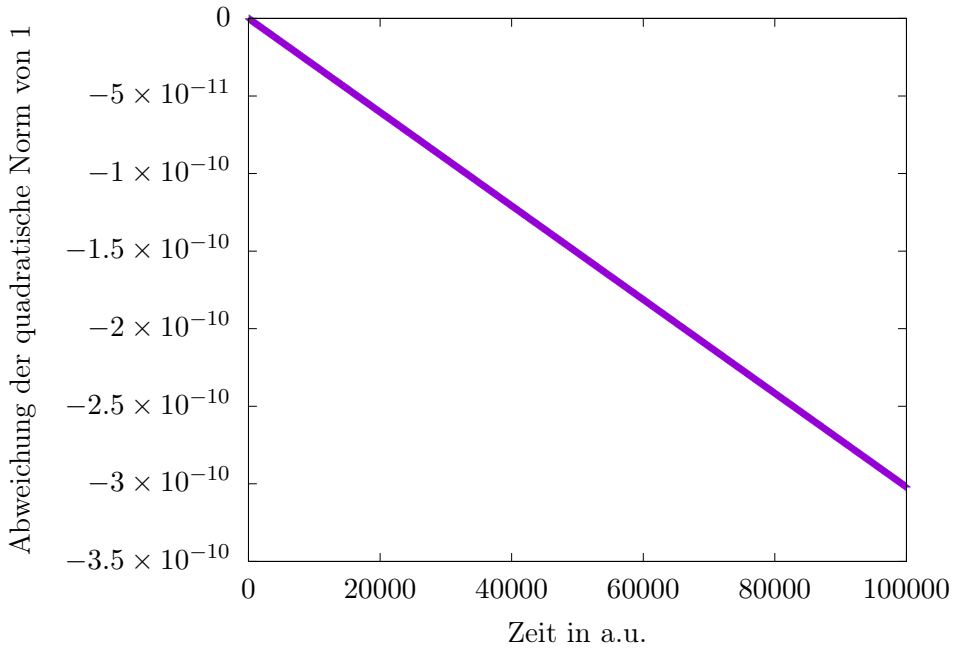


Abbildung 5.3: Entwicklung der quadratischen Norm im Verlauf der Zeit bei der nicht divergierenden numerischen Berechnung einer Wellenfunktion mit den Parametern $L = 1000$, $\delta_t = 1$, $\delta_x = 1$ und $Q = 50$

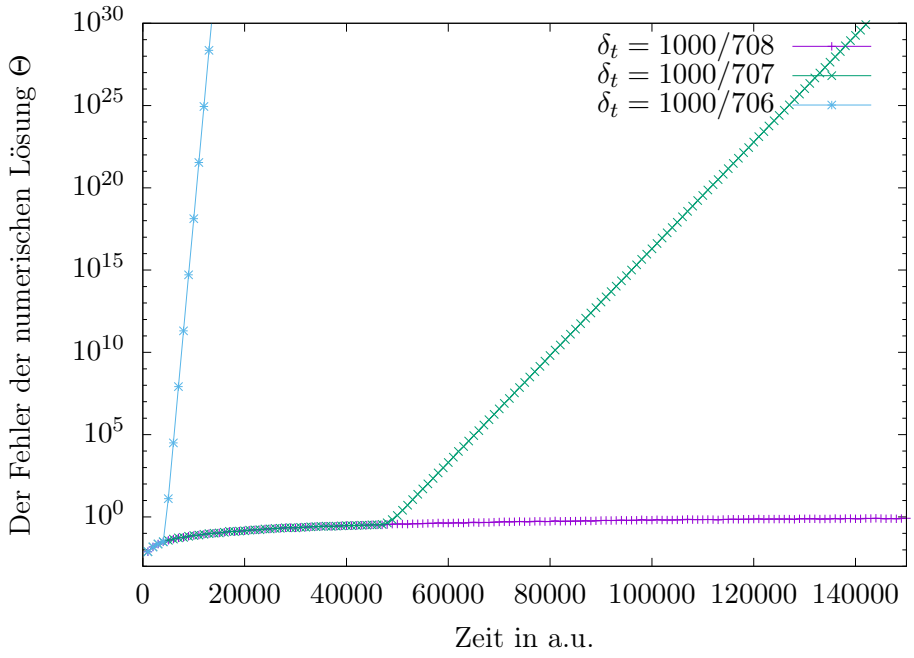
dem Wert 1 fällt sie innerhalb des Zeitraums etwa um 3×10^{-10} ab. Die Änderung ist zwar gemessen an der Größenordnung des Betrags sehr gering. Allerdings ist der Verlust der Norm klar zu erkennen. Die Norm einer numerischen Wellenfunktion sollte optimalerweise konstant den Wert 1 annehmen.

5.7 Die Zeitentwicklung im numerisch instabilen Fall

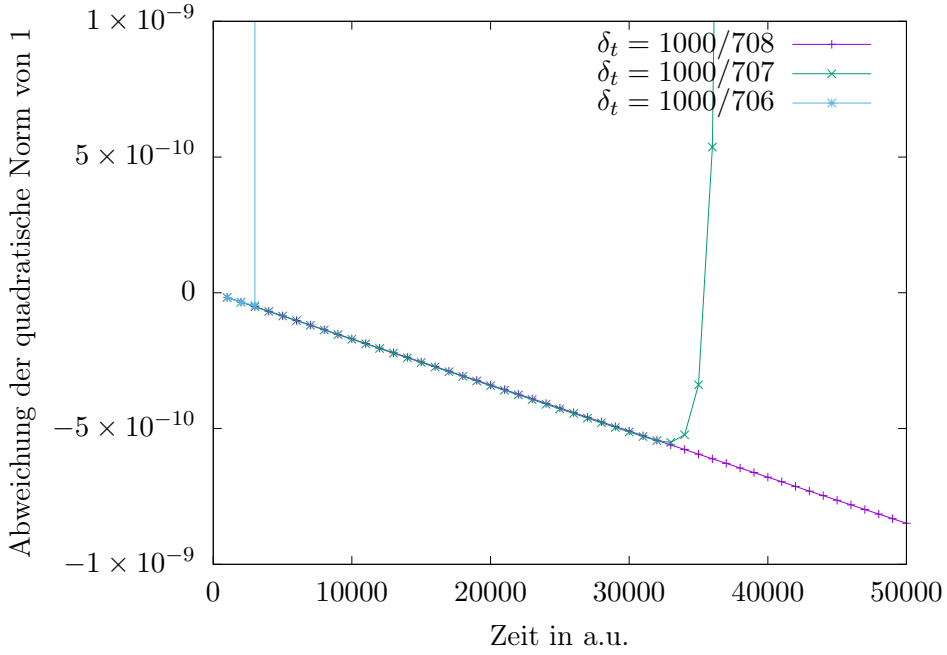
Als nächstes wird der zeitliche Verlauf des Fehlers eines numerischen Vektors im Divergenzfall untersucht. Dazu wird für die Einstellung der Parameter $L = 1000$, $\delta_x = 1$, $m = 1$ und $Q = 50$ untersucht, für welche δ_t die Wellenfunktion divergiert. Für ein Zeitintervall von 1000 wird beobachtet, dass die numerische Wellenfunktion bei großen N_t gut mit der analytischen Lösung übereinstimmt und erst bei $N_t < 704$ also einem $\delta_t > \frac{1000}{703}$ ein deutliches Zacken-Muster ausbildet welches dazu führt, dass die Wellenfunktion divergiert.

Wird das Zeitintervall verlängert, so fällt auf, dass auch kleinere δ_t noch zu einer Divergenz der Wellenfunktion führen können. In Abb. 5.4a ist der Fehler Θ für verschiedene δ_t im Verlauf der Zeit abgebildet. Die Skalierung der Y-Achse ist dabei logarithmisch.

Zu sehen sind drei Kurven, welche nur gering in ihrem δ_t voneinander abweichen, aber ein komplett unterschiedliches Divergenzverhalten aufweisen. Der Graph mit dem kleinsten $\delta_t = \frac{1000}{708}$ steigt zu Beginn erst stark und dann schwächer an und nähert sich dabei den Wert 1 an. Über die gesamte betrachtete Zeitspanne von etwa 140000 steigt der Verlauf nie wesentlich über dem Wert 1 an. Die Annäherung an den Wert 1 verläuft relativ schnell sodass die Kurve bereits nach dem ersten Drittel des Zeitverlaufs so stark abgeflacht ist, dass kaum noch eine Steigung erkennbar ist. Dieses Verhalten zeigt, dass die Übereinstimmung der numerischen mit der analytischen Lösung schon früh



(a) Entwicklung des Fehlers Θ



(b) Entwicklung der Abweichung der quadratischen Norm von 1

Abbildung 5.4: Die Zeitentwicklung von divergierenden, numerischen Wellenfunktionen im Vergleich

in der betrachteten Zeitspanne sehr schlecht ist. Dass der Fehler jedoch nie wesentlich größer als 1 ist zeigt aber, dass der numerische Vektor innerhalb dieses Intervalls kein Divergenzverhalten aufweist. Auch der visuelle Eindruck des Real- und Imaginärteils der Wellenfunktion zeigt keine Auffälligkeiten. Die Kurve mit dem etwas größerem $\delta_t = \frac{1000}{707}$ zeigt über eine lange Zeitspanne von etwa 50000 eine gute Übereinstimmung mit der ersten Kurve. Etwa ab dieser Zeit macht sie jedoch einen Knick und beginnt in der logarithmischen Darstellung linear zu steigen.

Dieses exponentielle Ansteigen des Fehlers zeigt, dass der Betrag der Differenz von den Funktionswerten der numerischen und analytischen Lösung eine viel höhere Größenordnung erreicht als der Betrag der analytischen Funktionswerte selbst. Das deutet auf die Divergenz des numerischen Vektors hin. Die Kurve mit dem größten $\delta_t = \frac{1000}{706}$ weicht schon nach kurzer Zeit deutlich von der analytischen Lösung ab und hat ab diesem Punkt in der logarithmischen Darstellung ein deutlich stärkeres lineares Wachstum. Die Divergenz verläuft also stärker exponentiell als bei $\delta_t = \frac{1000}{707}$. Abb. 5.4b zeigt die Entwicklung der quadratischen Norm der selben numerischen Vektoren.

Alle drei Kurven liegen zur Beginn der Simulation auf einer schwach abfallenden Geraden welche den Verlust der Norm andeutet, der auch schon in Abschnitt 5.6 beobachtet wurde. Die beiden Kurven mit größerem δ_t verlassen diesen Verlauf jedoch ab einem Zeitpunkt, welcher mit dem Beginn des exponentiellen Anstiegs des Fehlers übereinstimmt.

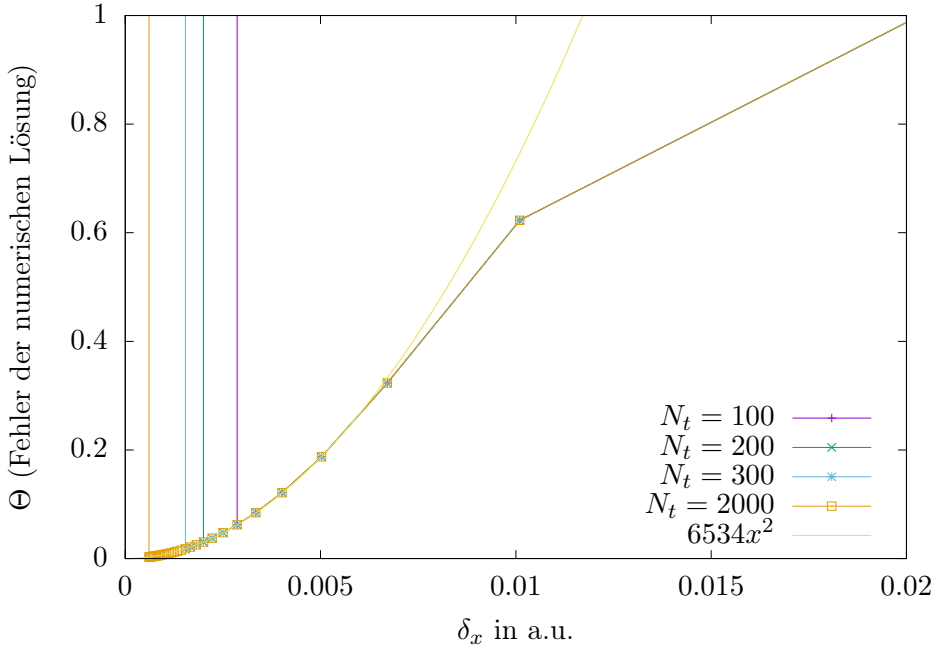
5.8 Der Fehler der numerischen Wellenfunktion in Abhängigkeit von δ_x und δ_t

Eine wichtige Frage ist, welche Auswirkung auf den Fehler der numerischen Lösung beobachtet werden kann, wenn man die Zahl der zeitlichen und räumlichen Gitterpunkte und somit δ_x und δ_t variiert. Hierfür wird Θ für $N_r, N_t \in \{50, 100, 150, \ldots, 1950, 2000\}$ bestimmt. Weiterhin verwenden wir: $L = 1000$, $T = 1000$, $Q = 50$ und $m = 1$. Die Auswertung von Θ erfolgt jeweils am Ende des Zeitintervalls T. In Abb. 5.5a und 5.5b ist der Fehler Θ in Abhängigkeit von δ_x jeweils für ausgewählte N_t abgebildet. Während Abb. 5.5a eine lineare Achsenskalierung hat, sind bei Abb. 5.5b die beiden Achsen jeweils in logarithmischer Darstellung.

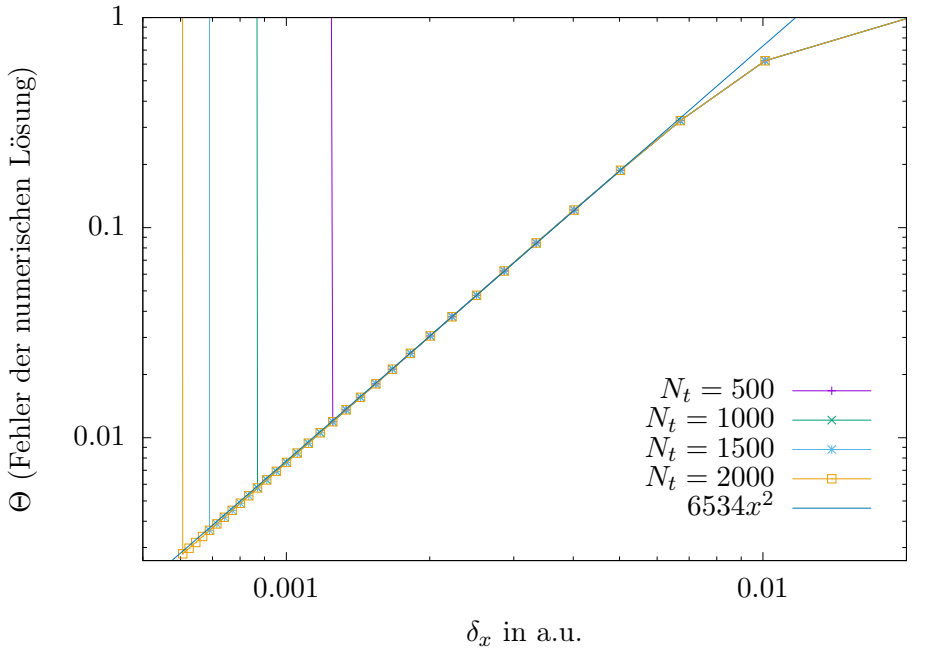
In beiden Darstellungen bilden die Graphen für unterschiedliche N_t einen gemeinsamen Strang, auf dem sie fast ununterscheidbar übereinander liegen. Man sieht dabei senkrechte Linien, die diesen Strang verlassen und nach oben hin aus dem Bild verschwinden. Diese gehören zu Plots, welche für höhere Werte von δ_x auf dem Strang liegen und für kleinere Werte divergieren. Für die Werte auf dem gemeinsamen Strang gibt es bei der logarithmischen Achsenwahl einen linearen Zusammenhang zwischen der Ortsauflösung und dem Fehler. Das bedeutet dass man für kleine δ_x folgende Abschätzung machen kann:

$$\Theta \approx B(\delta_x)^A \tag{5.12}$$

Um den Wert von A und B in einem konkreten Fall zu berechnen wird eine Simulationsreihe mit $N_t = 10001$ und $N_x \in \{10, 20, \ldots, 3790, 3800\}$ durchgeführt. Da der Fehler für $N_x > 3760$ divergiert und bei niedrigen N_x deutlich sichtbar von der einfachen Approximationsformel abweicht, werden davon die Simulationswerte mit $N_x \in \{1010, 1020, \ldots, 3750, 3760\}$ verwendet, um A und B mit Hilfe des Programms Gnuplot zu berechnen. Die errechneten Werte sind $A = 1.99954 \pm 0.00042\%$ und



(a) Lineare Achsenskalierung



(b) Logarithmische Achsenskalierung

Abbildung 5.5: Der Fehler in Abhängigkeit von δ_x

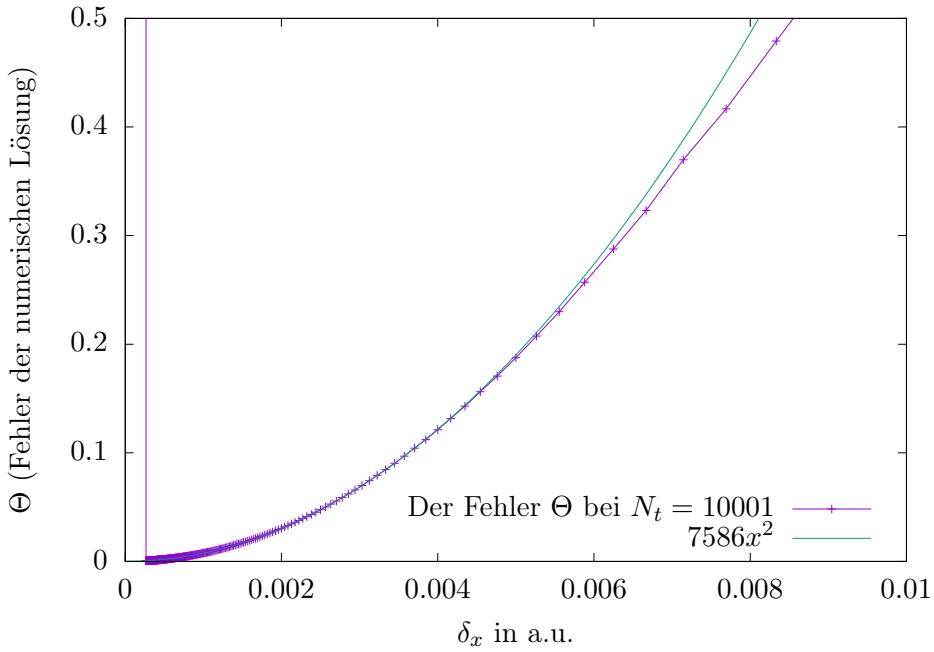


Abbildung 5.6: Der Fehler in Abhängigkeit von δ_x (Vergrößerter Ausschnitt)

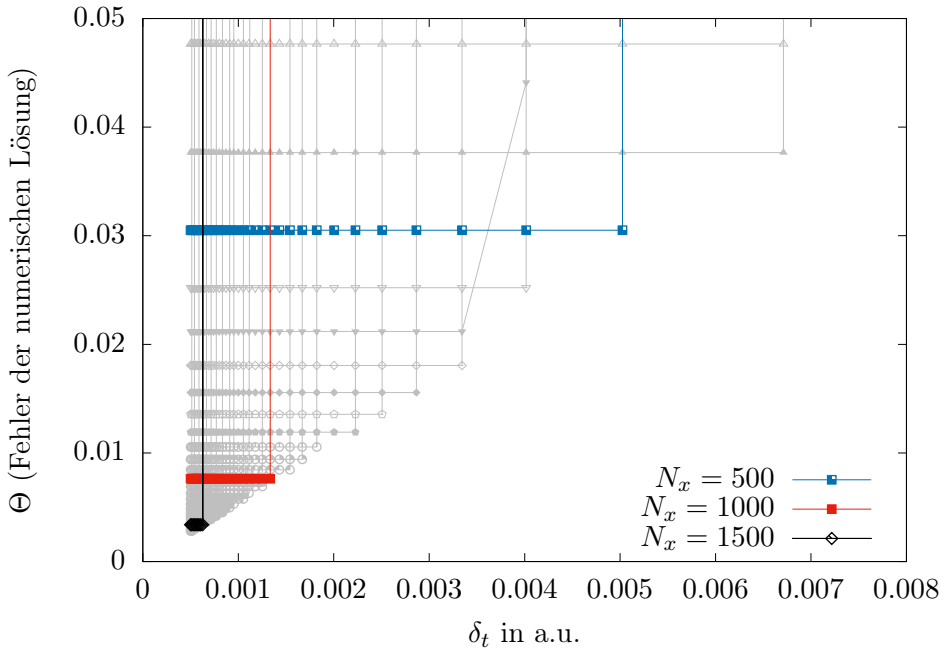
$B = 7585.98 \pm 0.0060\%$. Abb. 5.6 zeigt den Vergleich der Simulationsergebnisse mit der Näherungsformel.

Zu sehen ist eine gute Übereinstimmung bei $\delta_x < 0.005$ und eine zunehmende Abweichung der Simulationswerte gegenüber der Näherungsformel bei steigendem δ_x. Außerdem ist die Divergenz der Wellenfunktion für die zu klein gewählten δ_x als senkrecht aufsteigende Linie am linken Rand der Abbildung zu erkennen. Diese Divergenz soll im Folgenden genauer untersucht werden. Dazu wird der selbe Datensatz aus Abb. 5.5a und 5.5b mit einer anderen Achsenwahl abgebildet. Dargestellt in Abb. 5.7a und 5.7b ist nun der Fehler Θ in Abhängigkeit von δ_t für $N_r \in 50, 100, \ldots, 1950, 2000$. Jeweils vier ausgewählte Kurven sind farbig dargestellt und beschriftet während die übrigen grau gehalten sind um die Übersichtlichkeit zu erhöhen. Abb. 5.7a hat lineare Achsenbeschriftungen, während die Achsen von Abb. 5.7b in logarithmischer Darstellung gehalten sind.

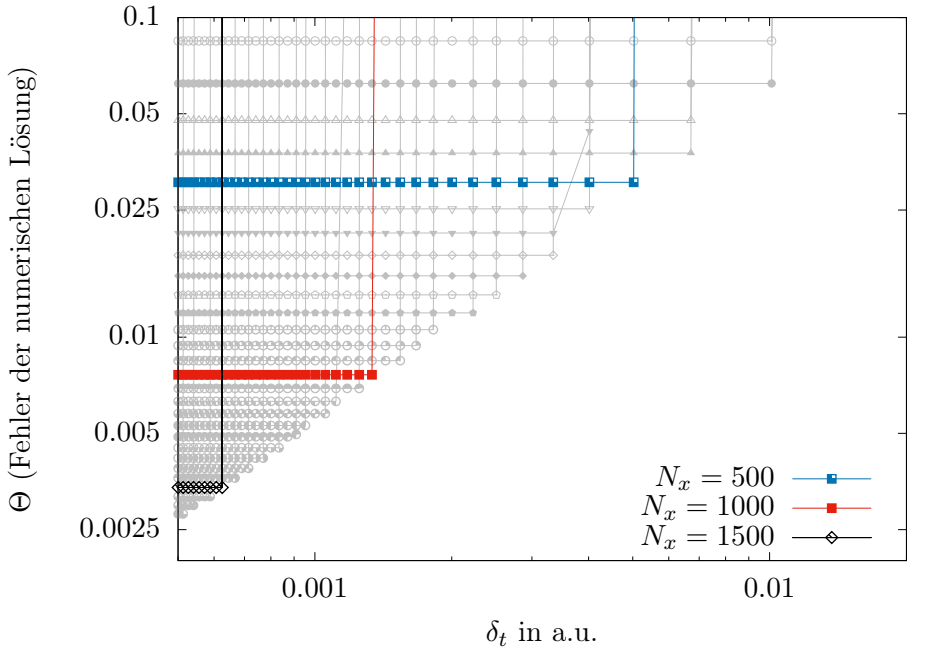
Man sieht, dass die einzelnen Plots in beiden Abbildungen am linken Bildrand bei kleinem δ_t beginnen und nahezu horizontal nach rechts verlaufen. Sie sind übereinander angeordnet und haben dabei von oben nach unten gelesen die Reihenfolge der dazugehörigen N_x. Wenn δ_t einen kritischen Wert überschreitet, divergiert der Wert des Fehlers Θ, was an einer senkrecht aufsteigenden Linie zu erkennen ist, mit der jeder der abgebildeten Graphen auf der rechten Seite endet.

Dieser kritische Wert ist bei den einzelnen Funktionen unterschiedlich. Das Minimum des Fehlers Θ aus allen betrachteten Graphen für ein gegebenes δt verhält sich dazu in etwa linear. Das führt dazu, dass die Gesamtmenge der Graphen eine Fläche in der oberen linken Ecke der Abbildungen abdeckt, die in der logarithmischen aber auch in der normalen Darstellung durch eine gerade Linie begrenzt wird, die diagonal durch das Bild verläuft.

Die beiden Abbildungen zeigen, dass eine Änderung von δ_t bei festem N_x im Konvergenzfall einen vergleichsweise geringen Effekt auf den Fehler der numerischen Lösung hat. Jedoch lässt sich N_x bei einem kleineren δ_t auf einen höheren Wert setzen, ohne dass die Wellenfunktion divergiert. Das wiederum ermöglicht es, den Fehler Θ um



(a) Lineare Achsenskalierung



(b) Logarithmischer Achsenskalierung

Abbildung 5.7: Der Fehler in Abhängigkeit von δ_t

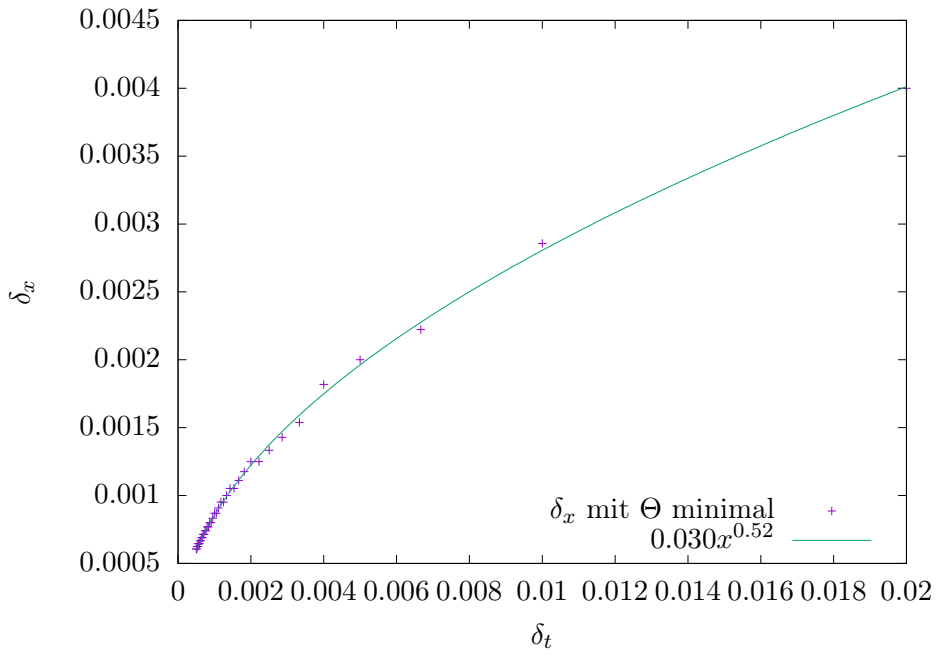


Abbildung 5.8: Das δ_x mit dem kleinstem Fehler in Abhängigkeit von δ_t

eine viel höhere Größenordnung zu senken.

Um diesen Effekt zu untersuchen, werden aus dem Datensatz die Werte von N_x gesammelt, bei denen Θ bei einem festen N_t den kleinsten Wert annimmt. In Abb. 5.8 sind die zu diesen Werten zugehörigen δ_x in Abhängigkeit von δt dargestellt. Der Kurvenverlauf ähnelt dem einer Wurzelfunktion. Durch einen Fit soll überprüft werden, wie gut er durch die folgende Approximationsformel beschrieben werden kann.

$$\delta_x \approx B(\delta_t)^A \tag{5.13}$$

Folgende Werte werden für A und B berechnet: $A = 0.5157 \pm 0.4985\%$, $B = 0.03016 \pm 1.433\%$. Die daraus folgende Funktion ist zusammen mit den Simulationswerten in Abb. 5.8 eingezeichnet und gibt den Gesamtverlauf gut wieder. Besonders für große δ_t sind aber Abweichungen der einzelnen Werte gut zu erkennen was sich auch in dem asymptotischen Standardfehler von A und B widerspiegelt, welcher höher ausfällt als bei der vorherigen Approximation durch Gleichung 5.12. Ein Grund dafür ist, dass die Ermittlung des δ_x mit dem kleinsten Fehler in diesem Fall sehr ungenau ist, weil N_x bei der Suche danach in 50er-Schritten erhöht wird.

In einer letzten Untersuchung des Konvergenzverhaltens des numerisch berechneten Vektors in Abhängigkeit der Zeitauflösung soll nun der direkte Einfluss einer Änderung von δ_t bei konstantem N_x genauer untersucht werden. In Abb. 5.5a und 5.5b war bereits erkennbar, dass dieser bei konvergenten Vektoren vergleichsweise gering ist. Abb. 5.9 zeigt den Verlauf des Fehlers Θ in Abhängigkeit von δ_t für $N_x = 1000$ bei vergrößertem Y-Achsenabschnitt.

Zu sehen ist, dass der Fehler bei zunehmenden δ_t erst sehr schwach und dann immer stärker ansteigt. Etwa bei $\delta_t = 0.00139$ macht der Graph einen Knick und steigt nahezu senkrecht an, weil die Wellenfunktion für größere δ_t divergiert. Der gesamte gezeigte Kurvenverlauf umfasst lediglich eine Änderung des Fehlers in der Größenordnung 10^{-9}. Sie ist damit etwa um die Größenordnung 10^{-6} kleiner als der Wert von Θ selbst. Es ist deutlich zu sehen, dass der Fehler bei $\lim_{\delta_t \to 0}$ nicht gegen 0 konvergiert. Über einen Fit soll geprüft werden ob der Verlauf des Fehlers durch den folgenden Ansatz beschrieben

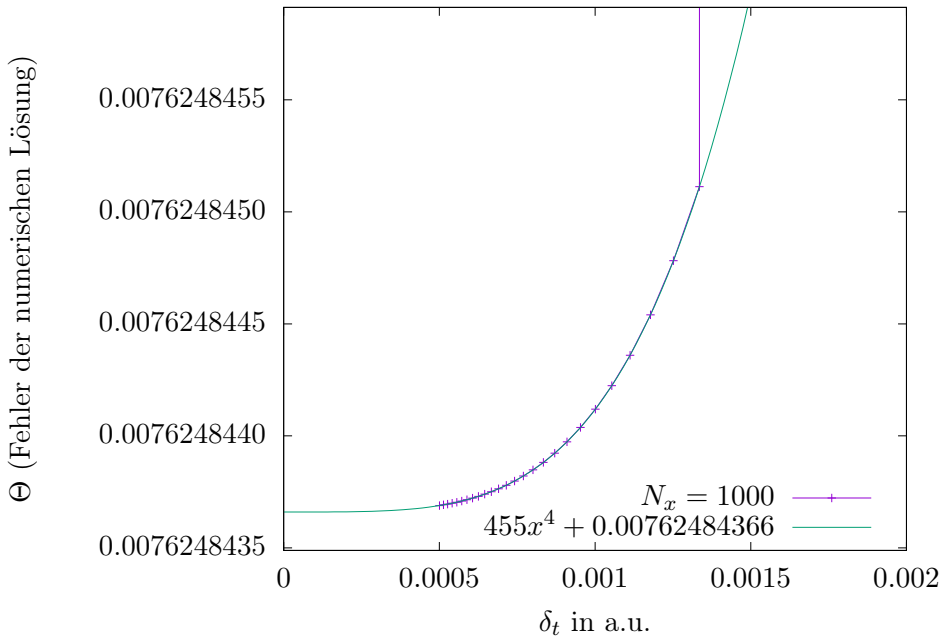


Abbildung 5.9: Der Fehler in Abhängigkeit von δ_t (Stark vergrößerter Ausschnitt)

werden kann.

$$\Theta \approx B(\delta_t)^A + C \tag{5.14}$$

Die berechneten Werte für A, B und C sind: $A = 3.99939 \pm 0.028\%$, $B = 455.1 \pm 0.76\%$ und $C = 0.00762484366043 \pm 8.44 \times 10^{-10}\%$ Damit hängt der Fehler Θ etwa mit der Potenz 4 von δ_t ab. Die resultierende Kurve ist in 5.9 eingezeichnet und zeigt eine gute Übereinstimmung für die konvergenten Simulationswerte.

5.9 Die Abweichung der analytischen von der numerischen Lösung in Abhängigkeit der Ordnung

Es wird nun untersucht, welchen Einfluss die Ordnung Q des Startvektors auf die Übereinstimmung der numerischen und analytischen Lösung hat. Es ist plausibel, dass höhere Ordnungen schlechter durch das numerische Verfahren berechnet werden können, weil ein einzelner Wellenberg dann durch weniger Gitterpunkte dargestellt wird. Diese Untersuchung soll eine Abschätzung liefern, welche Mindestanzahl an Gitterpunkten zur Darstellung eines Wellenberg verwendet werden sollte. Für die Parameter $L = 1000$, $T = 1000$, $m = 1$, $\delta_t = 1$ und $\delta_x = 1$ wird die Abweichung Θ in Abhängigkeit der Ordnung des Startvektors bestimmt. Abbildung 5.10 stellt sowohl den Fehler Θ als auch die quadratische Norm des numerisch berechneten Vektors in Abhängigkeit der Ordnung des Startvektors in einem Koordinatensystem dar.

Zu sehen ist, dass der Fehler bei einer Ordnung kleiner als 50 sehr gering ist und kaum ansteigt. Von der Ordnung 50 bis hin zur Ordnung 150 steigt der Fehler stark an bis er fast den Wert 1 erreicht. Ab diesem Punkt ist der Fehler so groß, dass kaum noch ein Zusammenhang zwischen analytischer und numerischer Lösung erkennbar ist.

Etwa bei der Ordnung 350 steigt der Wert Θ mit circa 1.15 auf ein Maximum und fällt dann wieder leicht ab, wobei er stets höher als 1 bleibt. Die quadratische Norm nimmt für niedrige Ordnungen stabil den Wert 1 an und beginnt ab der Ordnung 300 zu sinken. Etwa bei Erreichen der Ordnung 700 hat sie sich halbiert und fällt dabei zunehmend

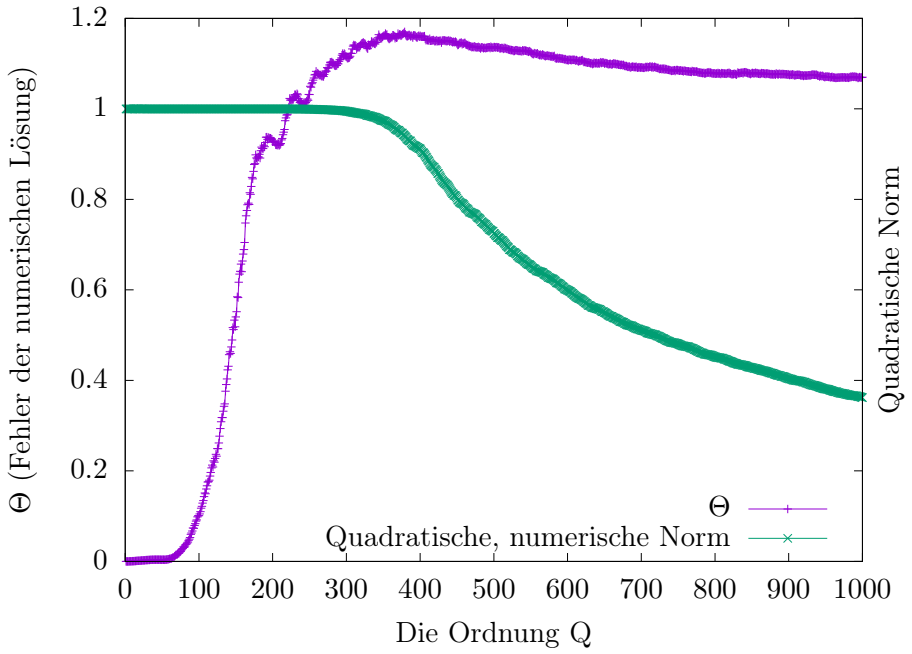


Abbildung 5.10: Der Fehler und die quadratische Norm in Abhängigkeit der Ordnung Q

langsamer ab. Abb. 5.11 zeigt einen vergrößerten Ausschnitt von Abb. 5.10, welcher die kleinen δ_t mit gutem Konvergenzverhalten beinhaltet. Man kann sehen, dass bis hin zur Ordnung 60 ein sehr kleines $\Theta < 0.01$ zu beobachten ist. Um ein gutes Simulationsergebnis zu erzielen, sollte jeder einzelne Wellenberg hier also mindestens durch 15 Raumpunkte dargestellt sein.

Abb. 5.12 bildet den Fehler Θ in Abhängigkeit von δ_x für verschiedene Ordnungen in doppelt logarithmischer Darstellung ab. Man kann sehen, dass auch für hohe Ordnungen ein geringer Fehler erreicht werden kann, wenn die Ortsauflösung groß fein gewählt ist.

5.10 Zusammenfassung

Für den numerischen Vektor konnte bei $\lim_{\delta_t,\delta_x\to 0}$ eine Konvergenz hin zu der analytischen Lösung gezeigt werden. Das zeigt, dass das verwendete Iterationsverfahren bei einer klugen Wahl dieser Parameter geeignet ist, um den Zeitverlauf einer Teilchenwellenfunktion numerisch zu approximieren. Der Fehler steigt dabei etwa proportional zu $(\delta_x)^2$ und zu $(\delta_t)^4$. Das ist plausibel, weil das Runge-Kutta-Verfahren vierter Ordnung für die Zeitentwicklung angewandt wurde, während mit der Finite-Differenzen-Methode ein viel einfacheres Verfahren für die Bestimmung der zweiten Ableitung des Ortes verwendet wurde.

Vektoren, die Schwingungen von höherer Ordnung enthalten, benötigen eine bessere Ortsauflösung, um den Fehler weiterhin gering zu halten.

Die Änderung des Fehlers nach δ_t hat bei den untersuchten Parametern im numerisch stabilen Fall eine deutlich kleinere Größenordnung als die Änderung nach δ_x und fällt daher kaum ins Gewicht.

Wenn δ_t jedoch einen Maximalwert überschreitet, der ungefähr proportional zu $(\delta_x)^2$ ist dann wird die numerische Berechnung instabil und der Vektor divergiert im Verlauf der Zeit. In der 3D-Ansicht der Wellenfunktion wird sichtbar, dass diese Divergenz durch eine Ausbildung von Zacken eingeläutet wird, die in der komplexen Ebene senkrecht zum

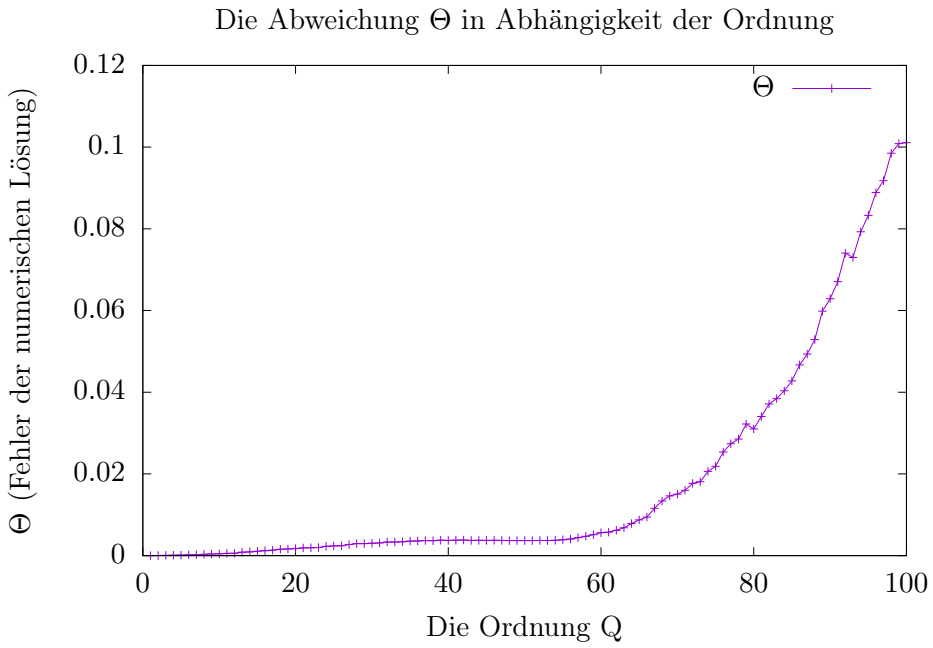


Abbildung 5.11: Der Fehler in Abhängigkeit der Ordnung Q (Vergrößerter Ausschnitt für kleine Q)

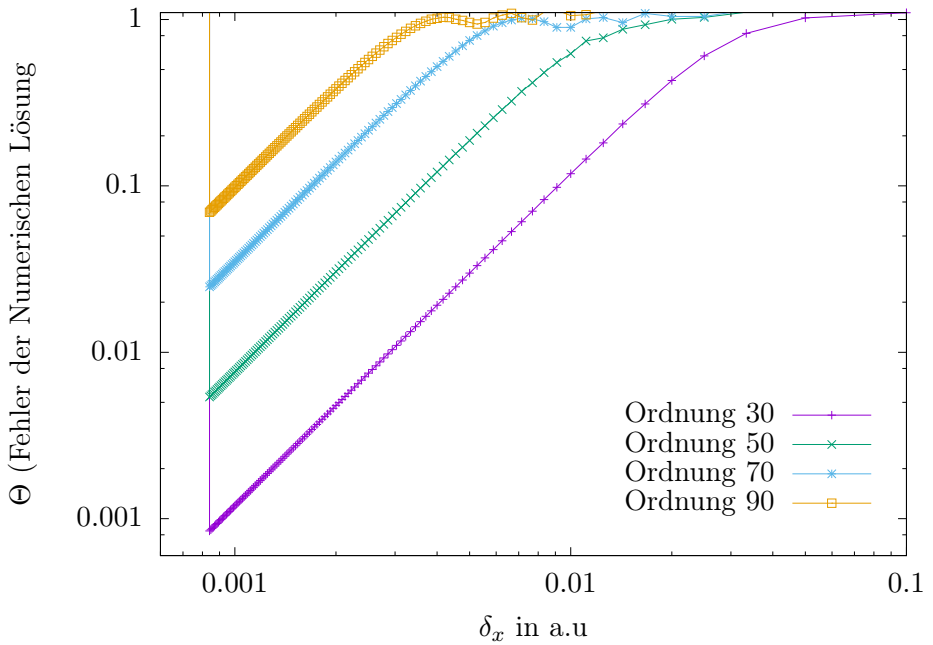


Abbildung 5.12: Der Fehler in Abhängigkeit von δ_x für verschiedene Ordnungen

jeweiligen Funktionswert ausgerichtet sind.

5.11 Deutung des Divergenzverhaltens

Um eine Erklärung für dieses Divergenzverhalten zu finden wurde auch untersucht, ob die unzureichende Präzision der verwendeten Gleitkommazahlen eine Rolle dabei spielen könnte. Dafür wurde das Divergenzverhalten sowohl mit 64-Bit als auch mit 32-Bit-Gleitkommazahlen untersucht. Abgesehen von einer leichten Verzögerung der Divergenz um niemals viel mehr als eine Iteration des Runge-Kutta-Verfahrens konnte hier jedoch kein Effekt beobachtet werden.

Der Grund für die Divergenz könnte darin liegen, dass bei einem kleinen δ_x in Kombination mit einem großen δ_t die zweite Ortsableitung, welche jeweils aus drei sehr nahe zusammenliegenden Funktionswerten errechnet wird, sehr stark Einfluss auf die zeitliche Änderung des Vektors nimmt. Dadurch könnte es sein, dass kleine Unregelmäßigkeiten in der Wellenfunktion, die als hochfrequente Schwankungen mit Wellenlängen in der Größenordnung δ_x auftreten, durch das Runge-Kutta-Verfahren in jeder Iteration vergrößert werden.

Ein interessanter Effekt, der gerade für schnell verlaufende Divergenzen auffällt ist, dass die Divergenzmerkmale der Wellenfunktion auf der linken Hälfte des Bildschirms mit leichter zeitlicher Verzögerung auftreten können. Auch in Abb. 5.1 ist das zu erkennen.

Eine mögliche Deutung wäre, dass beim Präparieren der Wellenfunktion im Grundzustand mit Hilfe der Sinus-Funktion durch die fehlende Präzision der Gleitkommazahlen kleine Unregelmäßigkeiten in die Wellenfunktion eingearbeitet werden. Diese würden dann erst durch das numerisch instabile Runge-Kutta-Verfahren vergrößert werden. Weil auf der linken Bildschirmhälfte mit kleineren X-Werten gearbeitet wird, könnte es sein, dass die Gleitkommazahlen an dieser Stelle mit einer höheren Präzision arbeiten als auf der rechten Seite und die Unregelmäßigkeiten deswegen kleiner ausfallen.

Kapitel 6

Physikalische Anwendung

Das Simulationsprogramm stellt viele verschiedene Werkzeuge zur Manipulation und Visualisierung von eindimensionalen Teilchenwellen bereit. Diese können dazu verwendet werden, eine Vielzahl von quantenmechanischen Versuchen zu simulieren und das jeweilige Verhalten der Teilchenwelle nachzuvollziehen. In diesem Kapitel wird das anhand von zwei einfachen Beispielen vorgeführt. Außerdem werden einige weitere Versuche aufgezählt, die sich mit Hilfe des Programms nachstellen lassen.

6.1 Die Moden im unendlich hohen Kastenpotential

Mit Hilfe der Anwendung können einige grundlegende Vorgänge der Quantenmechanik auf interaktive und spielerische Weise untersucht werden. Beispielsweise kann man eine Mode des Kastenpotentials als Startvektor festlegen. Dazu wählt man in der Formelauswahl die Beispielformel "Moden des unendlich hohen Kastenpotentials" und ändert die Variable n auf die gewünschte Ordnung der Mode. In diesem Beispiel wird n auf 20 gesetzt. Folgender Text müsste nun im Formelfeld zu sehen sein:

```
var n = 20;
R = Math.sin(X * n *Math.PI / L) * Math.sqrt(2 / L);
I = 0;
```

Betätigt man den Button "Formel anwenden" und wählt für die visuelle Darstellung das Betragsquadrat so werden 20 Wellenberge auf dem Bildschirm sichtbar, welche die erhöhte Aufenthaltswahrscheinlichkeitsdichte des Teilchens angeben. Lässt man die Simulation laufen so wird man keine Änderung an dem Betragsquadrat feststellen. Das liegt daran, dass es sich bei der Wellenfunktion um einen Eigenvektor des Energieoperators handelt. Stellt man über das Parameterfenster die Visualisierung für den Real- und Imaginärteil ein so kann man eine Rotation der Funktionswerte in der komplexen Zahlenebene feststellen.

Man kann nun untersuchen welche Auswirkung ein Anheben beziehungsweise Absenken des Potentials hat indem man die Simulation pausiert, das Potential mit Hilfe des Formelfelds neu setzt, und die numerische Berechnung anschließend wieder startet. Um das Potential konstant auf einen neuen Wert zu setzen genügt es v neu zuzuweisen und die Formel dann anzuwenden. Um das Potential auf 0.05 zu setzen schreibt man also:

```
V = 0.05;
```

Man wird feststellen das die Höhe des Potentials im linearen Zusammenhang mit der Winkelgeschwindigkeit steht mit der der Vektor in der komplexen Ebene rotiert. Aus der Schrödingergleichung 1.1 für den unendlich hohen Potentialtopf der Höhe A folgt für

$\Psi(x,t) = \exp(-i\omega t)\sin(n\pi x/L)$ innerhalb des Intervalls $[0, L]$:

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = \omega\hbar\Psi \tag{6.1}$$

$$= -\frac{\hbar^2}{2m}\Delta\Psi + A\Psi(x,t) = (\frac{(\hbar n\pi)^2}{2mL^2} + A)\Psi \tag{6.2}$$

$$\implies \omega = \frac{\hbar(n\pi)^2}{2mL^2} + \frac{A}{\hbar} \tag{6.3}$$

6.2 Der Einfluss eines linear abfallenden Potentials auf ein Wellenpaket

Bei angehaltener Simulation kann man nach Auswahl der Option "Wellenfunktion zeichnen" ein nach belieben geformtes Wellenpaket auf den Bildschirm malen. Wenn man nun die Wirkung eines räumlich abfallenden Potentials $V(x,t) = Ax$ auf das Wellenpaket beobachtet, wird man feststellen, dass die komplexen Werte der Wellenfunktion an verschiedenen Stellen mit unterschiedlichen Winkelgeschwindigkeiten in den komplexen Zahlen rotierten. Ein solches Potential kann man beispielsweise über folgende Formel erzeugen:

```
V = -0.0001 * x;
```

Ist das Wellenpaket breit genug, und die räumliche Krümmung somit klein genug, so kann der Term $\frac{\hbar^2}{2m}\Delta$ in der Schrödingergleichung vernachlässigt werden. Vereinfacht ausgedrückt gilt dann:

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = V(x,t)\Psi(x,t) = Ax\Psi(x,t) \tag{6.4}$$

Die Winkelgeschwindigkeit der Rotation der Funktionswerte in der komplexen Zahlenebene steigt also linear mit x an. Das bedeutet, dass das Wellenpaket damit beginnt eine spiralförmige Struktur auszubilden deren Verdrehung immer stärker wird.

$$\Psi(x,t) = e^{-iAtx/\hbar}\Psi_0(x) \tag{6.5}$$

Das aber ist äquivalent zu einer Verschiebung im Impulsraum. Was man auch daran beobachten kann, dass das Wellenpaket beginnt sich in Richtung des abfallenden Potentials zu bewegen.

6.3 Weitere Vorschläge für Experimente

In diesem Abschnitt werden einige weitere quantenmechanische Phänomene gesammelt, die man mit Hilfe des Programms studieren kann. Zu manchen Beispielen in dieser Liste sind voreingestellte Formeln in der Formelwahl des Programms verfügbar.

- Es können Gaußsche Wellenpakete erzeugt werden, die man auch mit einem Impuls ausstatten kann.
- Durch die Untersuchung der Expansionsgeschwindigkeit von stärker und schwächer lokalisierten Gaußschen Wellenpaketen kann ein Verständnis von der Orts-Impuls-Unschärfe gewonnen werden.

- Es kann die Teilreflexion eines Wellenpakets an einer Potentialbarriere untersucht werden.
- Mit Hilfe der Reflexion an Potentialbarrieren kann auch konstruktive und destruktive Selbstinterferenz realisiert werden.
- Die Eigenvektoren des harmonischen Oszillators können erstellt und ihr Verhalten beobachtet werden.
- Die Schwingung eines Gaußschen Wellenpakets im harmonischen Oszillator kann beobachtet werden.
- Die Wellenfunktion kann so präpariert werden, dass sie sich aus einer Gruppe von Gaußschen Wellenpaketen zusammensetzt die in der Bildschirmmitte eine Gitterstruktur ausbilden. Wenn jetzt die Simulation gestartet wird bilden sich, ähnlich wie bei der Elektronen-Beugung am Gitter Maxima und Minima durch Interferenz aus.
- Die Manipulation von Wellenfunktionen durch ein zeitabhängiges Potential kann untersucht werden.

Kapitel 7

Mögliche Weiterentwicklungen

Die Entwicklung der Simulationssoftware ist ein Prozess, bei dem man an vielen Stellen abwägen muss in welchen Bereichen man die Funktionalität des Programms erweitert. Jede Erweiterung benötigt Zeit und erhöht unter Umständen die Komplexität. Manchmal ist es sinnvoll auf zusätzliche Performance und Features zu verzichten, wenn sie den Zeitplan des Projektes sprengen.

Das bedeutet, dass die fertige Webanwendung zur Simulation von Teilchenwellen in vielen Punkten ausbaufähig ist. Einige der möglichen Weiterentwicklungen betreffen die Umsetzung bereits implementierter Konzepte. Andere betreffen Erweiterungen, welche eine zusätzliche Hilfe für das Verständnis von quantenmechanische Teilchenwellenfunktionen sein könnten. In diesem Abschnitt werden Kritikpunkte an der bestehenden Anwendung gesammelt und Ideen zusammengetragen, welche Verbesserungen und Erweiterungen des Programms möglich sind.

7.1 Mögliche Verbesserung der numerischen Berechnung

Wie die Untersuchungen in Kapitel 5 gezeigt haben, ist das Iterationsverfahren geeignet, um das Verhalten von Teilchenwellen mit großen Wellenlängen bei konstantem Potential über kürzere Zeiträume zu simulieren. Dabei ist aufgefallen, dass eine Erhöhung der Zeitauflösung im numerisch stabilen Fall kaum einen Beitrag zu einem besseren Simulationsergebnis leistet. Dennoch gibt es eine Mindestanforderung an die Zeitauflösung, welche eingehalten werden muss, um eine Divergenz des Wellenvektors zu verhindern.

Es stellt sich also die Frage, ob es sinnvoll ist ein Runge-Kutta-Verfahren vierter Ordnung als numerisches Verfahren für die Zeit mit der Finite-Differenzen-Methode als numerisches Verfahren für den Ort zu kombinieren. Damit ist die Methode zur Handhabung der Zeitableitung deutlich präziser als die, zur Berechnung der Ortsableitung. Es wäre also möglich, dass eine bessere numerische Lösung bei niedriger Anzahl an Rechenoperationen erreicht werden kann, wenn man ein Verfahren mit höherer Ordnung für die Berechnung der Ortsableitung verwendet.

Nicht untersucht wurde in dieser Arbeit, welche Anforderungen an das Potential gegeben sein müssen, damit die Ergebnisse der numerischen Berechnung mit der analytischen Lösung gut übereinstimmen. Auch das wäre ein Aspekt, der im Zusammenhang mit dem verwendeten numerischen Verfahren untersucht werden könnte. Grundsätzlich besteht auch die Möglichkeit, die Anzahl und Position der numerischen Raumpunkte dynamisch an die jeweilige Wellenfunktion anzupassen. Wenn eine

Teilchenwelle in manchen Raumbereichen lokal eine stärkere Krümmung aufweist, könnte die Dichte an numerischen Raumpunkten dort erhöht werden.

7.2 Mögliche Verbesserung der Performance

Die Steigerung der Performance der Anwendung bringt viele Vorteile. Durch eine höhere Zahl an Rechenoperationen pro Sekunde kann die Präzision der Berechnung weiter erhöht werden, ohne dass es Einbußen bei der Simulationsgeschwindigkeit gibt. Außerdem können dann komplexere physikalische Systeme flüssig simuliert werden, mit denen ansonsten keine Echtzeitinteraktion möglich wäre. Die Strategie, alle rechenintensiven mathematischen Berechnungen von Webassembler-Code ausführen zu lassen, welcher aus C-Code generiert wird, sorgt für eine deutliche Verbesserung der Performance gegenüber reinem Javascript-Code. Allerdings gibt es noch weitere Möglichkeiten die Leistung der Anwendung zu verbessern:

Die verwendete Architektur des Programms bewirkt, dass alle Rechenoperationen der Reihe nach hintereinander ausgeführt werden. Es ist aber möglich manche Berechnungen zeitgleich durchzuführen. Zum einen kann so die Berechnung der Schrödingergleichung in einen vom restlichen Programm unabhängigen Rechenprozess ausgelagert werden. Zum anderen kann auch die Anwendung des Runge-Kutta-Verfahrens auf einzelne Abschnitte des Wellenvektors parallel zueinander durchgeführt werden.

Eine Möglichkeit das zu erreichen ist die Verwendung mehrerer Threads. Ein Thread ist ein unabhängiger Rechenprozess innerhalb eines Programms. Die Javascript-Umgebung und Webassembler-Umgebung haben entsprechende Schnittstellen, um Anwendungen mit mehreren Threads zu unterstützen. Auch Emscripten unterstützt den Zugriff auf diese durch den C-Code. Allerdings waren diese Schnittstellen zum Zeitpunkt der Planung des Programms noch in einem frühen Entwicklungsstadium. Der Firefox-Browser hatte einige Features wieder aus seiner Standardversion entfernt nachdem Sicherheitsprobleme mit diesen Schnittstellen bekannt geworden waren. Es ist aber denkbar, dass diese Technologien in naher Zukunft in allen modernen Browsern verfügbar sind.

Eine zweite, deutlich effizientere Methode die Berechnungen zu parallelisieren wäre es diese auf die Grafikkarte auszulagern. Auch hierfür gibt es mit WebGL eine Schnittstelle für Webanwendungen.

Neben der fehlenden Parallelisierung gibt es in dem bestehenden Programm noch eine zweite Schwachstelle, die sich negativ auf die Performance auswirken könnte. Die Plots der Wellenfunktionen setzen sich aus einzelnen geraden Linien zusammen, die vom Javascript-Code nacheinander über die Befehle `beginPath`, `lineTo` und `stroke` des `CanvasRenderingContext2D` Elementes aufgerufen werden. Eine zusätzliche Steigerung der Geschwindigkeit könnte man also möglicherweise erreichen wenn man die graphischen Pixeldaten der Plots direkt durch Webassembler-Code erzeugt.

7.3 Mögliche Erweiterungen

An dieser Stelle sollen einige sinnvolle Erweiterungen aufgelistet werden, welche die Einsatzmöglichkeiten und den Funktionsumfang der Anwendung noch weiter verbessern würden:

- Das Programm ist in seinem jetzigen Zustand nicht dafür ausgelegt auf Smartphones oder Tablets zu laufen weil es eine Maus und eine Tastatur benötigt. Eine

Bedienungsmöglichkeit über den Touchscreen würde die mobile Einsatzfähigkeit verbessern.

- Das Programm bietet nur die Möglichkeit der Berechnung von eindimensionalen Teilchenwellen. Grundsätzlich würde die Leistungsfähigkeit von modernen Computern auch ausreichen um zweidimensionale Teilchenwellen mit niedriger Auflösung interaktiv zu simulieren. So könnten auch noch weitere Konzepte wie der quantenmechanische Drehimpuls untersucht werden. Eine einfache und nicht interaktive Demonstration der Berechnung einer zweidimensionalen Teilchenwelle mit einer Auflösung von 500×500 Raumgitterpunkten kann abgerufen werden auf:

  quantsimulant.de/demos/2d_schroed_demo/

- Es wäre auch möglich zwei wechselwirkenden Teilchen in einer Dimension zu simulieren. Der Rechenaufwand wäre vergleichbar mit der Simulation eines Teilchens im zweidimensionalen Fall.

- Eine weitere, nützliche Eigenschaft des Programms wäre es das aktuelle Energiespektrum der Wellenfunktion zu berechnen und anzuzeigen.

- In einer zukünftigen Version des Programms könnte man das Teilchen mit einem Spin ausstatten.

- Die Messung als zentraler Teilaspekt der Quantenmechanik wird in der hier beschriebenen Programmversion nicht behandelt. Eine einfache Möglichkeit dieses Konzept zu integrieren wäre es die Messung des Spins zu ermöglichen. Dabei ließe sich der anschließende Kollaps der Wellenfunktion leicht implementieren. Über gezielte Wechselwirkungen zwischen den beiden Spin-Zuständen vor der Messung könnte man dabei auch indirekt Rückschlüsse über andere Eigenschaften der Wellenfunktion wie dem Ort des Teilchens ziehen.

Kapitel 8

Zusammenfassung

Das Ziel dieser Arbeit war die Entwicklung eines Programms, dass durch die interaktive Simulation der Schrödingergleichung beim Verständnis von Teilchenwellenfunktionen hilfreich sein kann. Es ist gelungen, dies in Form einer vielseitig einsetzbaren Webanwendung zu realisieren. Die Eigenschaften dieser Anwendung wurden detailliert beschrieben und einige Verwendungsmöglichkeiten anhand von einfachen Beispielen vorgeführt.

Für den Fall des unendlich hohen Kastenpotentials konnte beobachtet werden, dass die über einen festen Zeitraum numerisch berechnete Wellenfunktion bei einer Erhöhung der Orts- und Zeitauflösung gegen die analytische Lösung konvergiert. Für einige Einstellungen der Simulationsparameter, bei denen die Ortsauflösung zu fein und die Zeitauflösung zu grob gewählt wurde, konnte eine numerische Instabilität beobachtet werden, die zu einer Divergenz des numerischen Vektors führt.

Das entwickelte Programm kann beim Studium von Teilchenwellen dazu eingesetzt werden den gelernten Stoff nachzuvollziehen oder auch dazu dienen Vorhersagen für neue quantenmechanische Experimente zu machen.

Kapitel 9

Danksagung

Danken möchte ich allen, die mich bei diesem Projekt unterstützt haben:

Vielen Dank an Prof. Dr. Robin Santra dafür, dass er mir so viel Freiheit bei der Themenwahl meiner Bachelorarbeit gelassen und mich zu dieser Arbeit ermutigt hat.

Vielen Dank an Prof. Dr. Daria Gorelova, dafür dass sie sich immer Zeit für mich genommen und mir in vielen Fragen weitergeholfen hat.

Vielen Dank an Michael Obermeyer für die Hilfe mit der Latex-Formatierung meiner Bachelorarbeit. Und auch an alle anderen, aktuellen und ehemaligen Mitglieder der CFEL-DESY Theory Division, die bei der Entstehung dieser Bachelorarbeit stets für eine freundliche Arbeitsatmosphäre gesorgt haben.

Mein herzlicher Dank gilt außerdem allen aus meinem Freundeskreis und meiner Familie, die sich in den letzten Wochen meine Arbeit durchgelesen und ihr wertvolles Feedback mit mir geteilt haben. Durch eure Hilfe konnte ich viele Fehler aus dieser Arbeit vertreiben.

Vielen Dank an: Dominik, Hannes, Josina, Caro, Claas, Remy und Martin

Literaturverzeichnis

[1] Chandralekha Singh and Emily Marshman. Review of student difficulties in upper-level quantum mechanics. *Phys. Rev. ST Phys. Educ. Res.*, 11:020117, Sep 2015.

[2] Hans Lüth. *Teilchen-Welle-Dualismus*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[3] Noah S. Podolefsky, Katherine K. Perkins, and Wendy K. Adams. Factors promoting engaged exploration with computer simulations. *Phys. Rev. ST Phys. Educ. Res.*, 6:020117, Oct 2010.

[4] Edgar Figueiras, David Olivieri, Angel Paredes Galan, and Humberto Michinel. Qmwebjs-an open source software tool to visualize and share time-evolving three-dimensional wavefunctions. *Mathematics*, 8, 03 2020.

[5] S. B. McKagan, K. K. Perkins, M. Dubson, C. Malley, S. Reid, R. LeMaster, and C. E. Wieman. Developing and researching phet simulations for teaching quantum mechanics. *American Journal of Physics*, 76(4):406–417, 2008.

[6] Mario Belloni and Wolfgang Christian. Physlets for quantum mechanics. *Computing in Science & Engineering*, 5:90 – 97, 02 2003.

[7] Ryan Sayer, Alexandru Maries, and Chandralekha Singh. Quantum interactive learning tutorial on the double-slit experiment to improve student understanding of quantum mechanics. *Phys. Rev. Phys. Educ. Res.*, 13:010123, May 2017.

[8] Shaeema Zaman Ahmed, Jesper Hasseriis Mohr Jensen, Carrie Ann Weidner, Jens Jakob Sørensen, Marcel Mudrich, and Jacob Friis Sherson. Quantum composer: A programmable quantum visualization and simulation tool for education and research, 2020. arXiv:2006.07263.

[9] Nicholas F. Polizzi and David N. Beratan. Open-access, interactive explorations for teaching and learning quantum dynamics. *Journal of Chemical Education*, 92(12):2161–2164, 2015.

[10] Patricia L. Lang and Marcy Hamby Towns. Visualization of wavefunctions using mathematica. *Journal of Chemical Education*, 75(4):506, 1998.

[11] Peter Junglas. Interaktive simulationsprogramme zur lösung der schrödingergleichung. *Wismarer Frege-Reihe*, 3:13–18, 01 2009.

[12] Chandralekha Singh. Interactive learning tutorials on quantum mechanics. *American Journal of Physics*, 76(4):400–405, 2008.

[13] Antje Kohnle, Charles Baily, Anna Campbell, Natalia Korolkova, and Mark J. Paetkau. Enhancing student learning of two-level quantum systems with interactive simulations. *American Journal of Physics*, 83(6):560–566, 2015.

[14] Mads Kock Pedersen, Birk Skyum, Robert Heck, Romain Müller, Mark Bason, Andreas Lieberoth, and Jacob F. Sherson. Virtual learning environment for interactive engagement with advanced quantum mechanics. *Phys. Rev. Phys. Educ. Res.*, 12:013102, Apr 2016.

[15] Markus Unkel. Numerische lösungen der zeitabhängigen schrödinger-gleichung mit interaktiver visualisierung, 2016. Bachelorarbeit, Rheinische Friedrich-Wilhelms-Universität Bonn.

[16] H. SHULL and G. G. HALL. Atomic units. *Nature*, 184(4698):1559–1560, Nov 1959.

[17] H. Gharibnejad, B.I. Schneider, M. Leadingham, and H.J. Schmale. A comparison of numerical approaches to the solution of the time-dependent schrödinger equation in one dimension. *Computer Physics Communications*, 252:106808, 2020.

[18] Larsson Stig and Thomée Vidar. *Partial Differential Equations with Numerical Methods*. Springer-Verlag Berlin Heidelberg, 2003.

[19] Ricardo Becerril, F. Guzmán, A Rendón-Romero, and Susana Valdez. Solving the time-dependent schrödinger equation using finite difference methods. *Revista mexicana de física E*, 54:120–132, 12 2008.

[20] Peter Deuflhard and Folkmar Bornemann. *Gewöhnliche Differentialgleichungen*. De Gruyter, Berlin, Boston, 2008.

[21] Shaaron Ainsworth. *The Educational Value of Multiple-representations when Learning Complex Scientific Concepts*, pages 191–208. 01 2008.

[22] Endre Süli and David F. Mayers. *An Introduction to Numerical Analysis*. Cambridge University Press, 2003.

[23] H. S. CARSLAW. *Introduction to the theory of Fourier's series and integrals*. London, Macmillan and Co., 1921.

[24] Franz Schwabl. *Quantenmechanik*, chapter 3.4, pages 64–66. Springer Berlin Heidelberg, 1988.