Technische Universität München
Fakultät für Mathematik
Wissenschaftliches Rechnen

Efficient Algorithms for Semiclassical Quantum Dynamics

David Sattlegger

Vollständiger Abdruck der von der Fakultät für Mathematik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr. Herbert Spohn
Prüfer der Dissertation:	1.	Univ.-Prof. Dr. Caroline Lasser
	2.	Univ.-Prof. Dr. Karl Kunisch (Karl-Franzens-Universität Graz, Österreich)

Die Dissertation wurde am 29.09.2015 bei der Technischen Universität München eingereicht und durch die Fakultät für Mathematik am 07.12.2015 angenommen.

Acknowledgements

First and foremost, I want to thank Caroline Lasser who did not just supervise my thesis but accompanied it all along the way. For this, she has my utmost and everlasting gratitude. I can honestly say that without her guidance and support over the last few years, both on a professional and personal level, I would not have been able to write this thesis.

Furthermore, I want to thank my second supervisor, Karl Kunisch, and his group for cordially welcoming me in Graz; special thanks also to my mentor, Manfred Liebmann, for his personal support, our long and fruitful discussions, and for teaching me basically all I know about parallel programming.

I gratefully acknowledge the funding received from the German Research Foundation (DFG), through the International Research Training Group IGDK1754, as well as the TUM Graduate School and the Faculty for Mathematics at Technische Universität München. They provided me with a certain financial freedom, which allowed me to concentrate on my research and offered opportunities concerning collaboration and participation at workshops and conferences.

Special thanks go to my friends and colleagues from Munich and Graz for enjoying work as well as leisure time with me. In particular, I want to mention Armin, Bao, Behzad, Bertram, and Philip who really made me feel at home while staying in Graz and I want to thank Georg, Ilja, Horst, Jelena, Johannes, Moritz, and especially Stephanie for all the great time we spent in Munich as well as all the members of M3.

Ich möchte auch noch einige ganz persönliche Wortes des Dankes an meine Familie und meine Freunde aussprechen, die mir nicht nur während des Schreibens dieser Dissertation immer zu Seite gestanden sind. Besonders möchte ich natürlich meine Eltern, Brigitte und Peter, und meinen Bruder Sebastian hervorheben. Nicht vergessen möchte ich auch Carina, Lisa, Alex, Jürgen und Thomas. Eure Freundschaft ist für mich unverzichtbar.

Und schließlich geht mein größter Dank an Lena und Susi. Ohne Dich, Lena, hätte ich es sicher nicht geschafft, diese Arbeit zu beenden. Und ohne Dich, Susi, hätte ich es wohl nicht geschafft, sie überhaupt erst zu beginnen.

Abstract

Quantum systems of interest in physics and chemistry have many degrees of freedom. The goal of my thesis is to provide numerical methods capable of simulating the dynamics of such systems. I study the Herman–Kluk propagator, which is capable of providing approximate solutions to the time-dependent semiclassical Schrödinger equation in high dimensions. I derive a discretisation scheme and prove rigorous error estimates. Moreover, this scheme as well as other semiclassical methods are of an intrinsically parallel nature. My contributions are the design and implementation of highly efficient algorithms employing state of the art parallelisation and vectorisation techniques.

Zusammenfassung

Quantensysteme, die in der Chemie und Physik von Interesse sind, weisen gewöhnlich eine große Anzahl an Freiheitsgraden auf. Ziel dieser Dissertation ist die Bereitstellung numerischer Methoden zur Simulation des Verhaltens derartiger Systeme. Zu diesem Zweck untersuche ich den Herman–Kluk Propagator zur Approximation von Lösungen der semiklassischen, zeitabhängigen Schrödingergleichung. Ich entwerfe ein Schema zur Diskretisierung des Herman–Kluk Propagators und zeige rigorose Fehlerabschätzungen. Dieses Schema weist, wie auch andere semiklassische Methoden, von sich aus eine parallele Struktur auf. Darauf aufbauend entwerfe und implementiere ich hochgradig effiziente Algorithmen unter Verwendung der neusten zur Verfügung stehenden Parallelisierungs- und Vektorisierungstechniken.

Table of Contents

Chapter 1

Introduction

The behaviour of things on a very tiny scale is simply different. An atom does not behave like a weight hanging on a spring and oscillating. Nor does it behave like a miniature representation of the solar system with little planets going around in orbits. Nor does it appear to be somewhat like a cloud or fog of some sort surrounding the nucleus. It behaves like nothing you have ever seen before.

– Richard P. Feynman, *The Character of Physical Law* [Fey94]

Motivation

Throughout the history of mankind or at least the history of civilisation, even before science in the modern sense existed, humans were wondering about their environment. They soon realised that 'the behaviour of things' changes in time. There is seemingly continuous and endless movement, such as ocean waves crashing on a beach, or more sudden movement like an apple falling from a tree. The first systematic investigations that we know of have their origins in the writings of various natural philosophers in Ancient Greece. Many concepts we still use today, including the idea of atomism, originate there. Nevertheless, it took almost two millennia from these beginnings to the development of a suitable mathematical formulation, most notably by Galileo Galilei and Isaac Newton. They initiated the study of what

we now call classical mechanics. To paraphrase the above quotation, classical mechanics describe how things behave on scales that we experience in our everyday life. Through the course of the centuries to follow, various experiments suggested that, as Feynman puts it, objects on very small scales are 'simply different'. The establishment of a new theory was inexorable. The scientific 'giants' of the era initiated it on the verge of the twentieth century and it was formulated in satisfactory mathematical terms most prominently by Werner Heisenberg and Erwin Schrödinger. We now call this theory quantum mechanics and it is still the fundamental description of all processes involving 'things on a very tiny scale', such as atoms, molecules, or photons. The principal equations of motion are readily written down, however, the actual computation of solutions to them proves to be extremely difficult. Their numerical solution is nowadays facilitated by the rapid development of ever more powerful computers, but it continues to be a challenge, in particular for systems with many degrees of freedom.

In a very broad sense my thesis aims to provide means to describe and predict the behaviour of what Feynman calls 'things on a very tiny scale' by approximating the full quantum mechanical evolution of a system with semiclassical methods. In other words, an atom still 'does not behave like a weight hanging on a spring and oscillating' but its behaviour can be approximated by studying numerous weights hanging on numerous springs.

Mathematical setting and related research

In essence, quantum mechanics can be described by a single partial differential equation that Erwin Schrödinger [Sch26d] introduced in 1926. In spite of the knowledge of existence and uniqueness of a solution, computation of such a wave function proves difficult since the dimension of the Schrödinger equation is three times the number of particles in the system. As a consequence, we require a model that significantly reduces the number of degrees of freedom as well as efficient approximation schemes. The former is provided by the time-dependent Born-Oppenheimer approximation, the latter is the main subject of this thesis.

The resulting model is described by the time-dependent semiclassical Schrödinger equation

$$\mathrm{i}\varepsilon\frac{d}{dt}\psi(t,x) = H^{\varepsilon}\psi(t,x), \quad \psi(0,\cdot) = \psi_0 \in \mathrm{L}^2(\mathbb{R}^d) \tag{1.1}$$

which expresses the motion of the nuclei in a molecule driven by the potential energy surfaces of the electrons. The operator H^ε is a self-adjoint operator on $\mathrm{L}^2(\mathbb{R}^d)$ called Born-Oppenheimer Hamiltonian. The semiclassical parameter $\varepsilon > 0$ is defined as the square root of the mass ratio of electrons and nuclei.

How can we calculate solutions to Equation (1.1)? First, let me remark that there are big risks involved when using conventional discretisation techniques such as finite differences for Schrödinger calculations in the semiclassical regime. Certain schemes produce completely wrong results under seemingly reasonable discretisation strategies. What is even worse, we get no warning by the scheme, e.g. in form of a blow-up, that something went wrong in the computation. One example is the widely used Crank–Nicolson scheme for temporal discretisation. While it shows some desirable properties, the scheme can violate the gauge invariance of (1.1). Furthermore, even discretisation schemes, that are stable and consistent, require huge computational resources in order to give accurate results for $\varepsilon \ll 1$. For details, let me refer to Jin, Markowich, and Sparber [JMS11] and the references therein.

Hence, totally different analytical and numerical techniques are required when dealing with both $d \gg 1$ and $\varepsilon \ll 1$. I introduce several of these semiclassical methods such as Hagedorn's semiclassical wave packets, Gaussian beam methods, and quasi-classical methods below and elaborate in particular on the Herman–Kluk Propagator. All these methods rely on the fact that for small ε, the solution to (1.1) is closely related to the solution of the corresponding classical mechanical problem.

The current state of the art can be described as follows. On the one hand, we have the works of theoretical chemists who have used semiclassical methods for almost thirty years. They apply these methods with great success and give formal derivations but do not prove them to be well defined, nor do they prove rigorous error estimates. From their point of view, the most important feature of a method is to provide results that are in accordance with and can be verified by actual experiments. From the point of view of a mathematician, the first steps when defining a new object is to prove them to be well defined, then carefully develop an algorithm, and finally establish estimates on its approximation properties. In case of the Herman–Kluk propagator, this has been done only quite recently in the mathematical literature by Swart and Rousse [SR09].

In my work I aim to combine these two points of view and mix them with yet another one, namely the one of a computer scientist. My contributions are as follows.

Based on work in theoretical chemistry as well as mathematical analysis, I design a discretisation scheme and provide a rigorous error analysis thereof. Moreover, I provide a way of implementing this algorithm in a highly efficient manner by systematically taking advantage of its inherently parallel nature.

Scope and outline of this work

This thesis is divided into three parts, one of which establishes the general physical and mathematical framework, another gives a description and an analysis of the discretisation scheme I propose for the Herman–Kluk Propagator, and the third the implementation of the scheme using state of the art techniques from high performance computing.

The parts themselves are further partitioned into chapters. I start with an overview of classical mechanics or rather classical dynamics. It is one of the oldest disciplines in physics and describes the behaviour of physical bodies and their interaction with each other and their environment. We use the concepts I introduce as a base for semiclassical methods, which are described in Chapter 4. The presentation will be brief and concise since this thesis cannot and shall not be a book on classical mechanics, especially since there are already excellent books written on this topic, e.g. by Arnol'd [Arn89].

In the subsequent chapter I give a similar introduction to quantum mechanics starting with the definition of the most important concepts as well as providing a short overview of the historical development in Chapter 3.1. The correspondence between classical and quantum mechanics will be explored in Chapter 3.2. It is followed by a description of quantum molecular dynamics where the time-dependent Born–Oppenheimer approximation is discussed, which gives rise to the semiclassical Schrödinger equation.

Chapter 4 introduces several semiclassical methods, such as Hagedorn's semiclassical wave packets, Gaussian beam methods, and quasi-classical approximations. The latter includes the concept of the Wigner transform, which allows an alternative description of quantum mechanical expectation values and is used to approximate them by means of Egorov's Theorem.

The first chapter of Part II finally gives a formal definition of the Herman–Kluk Propagator and provides details of its most important features. In order to make this definition rigorous we have to introduce the notion of Fourier integral operators, which is the emphasis of Chapter 6. The following chapter introduces 'the' algorithm that I propose for the discretisation of the Herman–Kluk Propagator. Chapter 7.1 describes how the phase space integral is approximated by Monte Carlo and quasi-Monte Carlo techniques, while Chapter 7.3 investigates the temporal discretisation using symplectic integrators in order to solve the underlying system of ordinary differential equations.

Chapters 8 and 9 provide a rigorous error analysis of phase space and time discretisation, respectively.

The third part consists of two chapters, of which the first one serves as a general introduction to parallel programming. Thereupon, Chapter 11 provides details on the highly efficient methods that I designed and implemented and which are based on the algorithm described in Chapter 7 as well as an algorithm based on a quasi-classical approximation introduced in Chapter 4.4. I eventually conclude my work with some final remarks and possible directions for future research.

Let me give some directions to the reader. Taking Higham's [Hig98][Chapter 4.31] advice, I use the pronoun 'I' to refer to myself. 'We', on the other hand, is used in the meaning of 'the reader and I'. As you have probably already realised, I use British English spelling throughout this thesis.

Part I

Dynamics

Chapter 2

Classical dynamics

One of the oldest branches of physics is the field of mechanics[1]. It is concerned with the behaviour of physical bodies and their interaction with each other and their environment. It has its origins in Ancient Greece with the writings of Aristotle and Archimedes. The systematic investigation of general mechanical systems and the development of a suitable mathematical formulation started with the monumental works of Galilei [Gal55] and Newton [New87]. The ideas therein were advanced and refined by the Italian-born mathematician and astronomer Lagrange [Lag88] and the Irish physicist and mathematician Hamilton [Ham34]. They introduced two different formalisms or views on mechanics that still preside over the field today. What we now call Lagrangian formalism is based on variational principles which can be generalised in order to describe quantum field theories. The Hamiltonian approach on the other hand is of a more geometric nature and closely related to quantum mechanics. This chapter is thought as an introduction to the topic of dynamics, the branch of mechanics that is concerned with the motion of bodies under the action of forces. Alternatively, paraphrasing Feynman once more, one could say that this chapter explains the behaviour of 'a weight hanging on a spring and oscillating' or of 'a solar system with little planets going around in orbits'. The presentation here will be concise, let me recommend the book by Arnol'd [Arn89] as a detailed account of the field.

[1] from the Latin term *ars mechanica*, equivalent to the Greek term μηχανικη τεχνη, literally 'the art of inventing or utilising machines', from μηχανη 'tool'

2.1 Lagrangian formalism

The Lagrangian reformulation of mechanics is based on the observation that variational principles govern the fundamental laws of force that were introduced by Newton. Lagrange [Lag88] proposed the configurations of a physical system with d degrees of freedom to be described by 'generalised coordinates' $\{q_1, \ldots, q_d\}$. Mathematically, the set of all possible configurations may be described by a smooth d-dimensional manifold Q. The evolution of the position of a particle is thought to be a curve on Q. Of course, not any curve on this configuration manifold is qualified to be the evolution of a mechanical system. The aim is to find the proper ones. Let us introduce a smooth function $\mathcal{L}$ on the tangent bundle $\mathcal{T}Q$ of Q as well as the following action functional. Together, they characterise the physical system.

Definition 2.1 (Action) *The functional $S : [0, T] \to \mathbb{R}$ that is defined by*

$$S(t) := \int_0^t \mathcal{L}\left(q(\tau), \dot{q}(\tau)\right) \, \mathrm{d}\tau \tag{2.1}$$

is called action integral or simply action along a curve $(q(\cdot), \dot{q}(\cdot))$ on $\mathcal{T}Q$.

Let us consider the variational problem of finding the critical points of the action integral S as a functional of curves on Q. By means of calculus of variations, one finds that the extremal paths must follow the Euler–Lagrange equations

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_k} = \frac{\partial \mathcal{L}}{\partial q_k}. \tag{2.2}$$

for all $k \in \{1 \ldots d\}$ Lagrange proved that these are in fact the curves along which the physical system evolves according to Newton's equations of motion. The Euler–Lagrange equations, however, are invariant under a change of coordinates and can be used to describe more general physical systems than Newton's equations. Let me now introduce the Hamiltonian formulation of classical mechanics which is in fact equivalent to the Lagrangian one as revealed by Theorem 2.6.

2.2 Symplectic geometry

Let us consider a classical physical system with d degrees of freedom. The Hamiltonian description of a mechanical system is formulated on and closely connected

to the geometry of the cotangent bundle $\mathcal{T}^*Q$ of the configuration manifold Q. This cotangent bundle, also called phase space, carries the structure of a symplectic manifold, i.e. a pair of a smooth manifold and a closed, non-degenerate differential two-form Ω.

Proposition 2.2 (Darboux Coordinates) *Let Q be a manifold. Then there exist local coordinates on $(q_1, \dots, q_d, p_1, \dots, p_d)$ on the cotangent bundle T^*Q in which the canonical 2-form is given by $\Omega = dq^k \wedge dp_k$.*

Thus, the canonical 2-form establishes a symplectic structure on T^*Q for any manifold Q. This defines a very profound structure on the cotangent bundle. Functions that preserve this structure are themselves called symplectic.

Definition 2.3 (Symplectic map) *A linear map $A : T^*Q \to T^*Q$ is called symplectic if $\Omega(Aw, Az) = \Omega(w, z)$ for all $w, z \in T^*Q$. A differentiable function $f : T^*Q \to T^*Q$ is called symplectic if its Jacobian matrix is symplectic wherever it is defined.*

Symplectic maps preserve phase space volume, among other things, but the notion of symplecticity has much deeper implications, cf. Theorem 2.4. I now want to leave the setting of arbitrary smooth manifolds in order to introduce Hamiltonian systems only on vector spaces. This is the setting we will need for the remainder of the text and it is much more pleasant to write down the equations of motion without considering local coordinate charts and general differential forms.

2.3 Hamiltonian formalism

Hamilton [Ham34] associates a 'characteristic function' to a dynamical system of attracting and repelling points. This function is now commonly called Hamiltonian function and denoted by $h : \mathbb{R}^{2d} \to \mathbb{R}$. It defines the equations of motion

$$\dot{z} = \mathcal{J} \nabla h(z), \qquad z(0) = z_0 \tag{2.3}$$

which depend on the canonic symplectic form on phase space $\mathbb{R}^{2d}$. This form is represented by the matrix

$$\mathcal{J} = \begin{pmatrix} 0 & \mathrm{I} \\ -\mathrm{I} & 0 \end{pmatrix} \in \mathbb{R}^{2d \times 2d}. \tag{2.4}$$

These equations of motion will form the foundation on which the subsequent considerations are based. For twice continuously differentiable Hamiltonian functions, (2.3) has a unique solution which allows us to define the classical flow

$$\Phi^t : \mathbb{R}^{2d} \to \mathbb{R}^{2d}, \quad z_0 \mapsto \begin{pmatrix} X^t(z_0) \\ \Xi^t(z_0) \end{pmatrix} \tag{2.5}$$

in such a way that $z(t) = \Phi^t(z)$ satisfies (2.3) with initial datum $z(0) = z_0$. This flow is in fact a symplectic transformation.

Theorem 2.4 (Poincaré [Poi99]) *Let h be a twice continuously differentiable function on an open set $h : U \subset \mathbb{R}^{2d} \to \mathbb{R}$. Then the flow Φ^t is a symplectic transformation wherever it is defined.*

But the notion of symplecticity plays an even more fundamental role. It is in fact a characteristic property for Hamiltonian systems. Let us consider an arbitrary system of differential equations $\dot{z} = f(z)$.

Theorem 2.5 *Let $f : U \subset \mathbb{R}^{2d} \to \mathbb{R}$ be continuously differentiable. Then $\dot{z} = f(z)$ is locally Hamiltonian if and only if its flow Φ^t is symplectic for all $z \in U$ and all sufficiently small t.*

A proof of these two theorems is found in the book by Hairer, Lubich, and Wanner [HLW06, Chapter VI.2]. As a consequence, the flow of any mechanical system is a symplectic transformation and any symplectic flow originates from a Hamiltonian system. In this sense, Hamiltonian systems and symplecticity are inseparably intertwined. This should, of course, be reflected when discretising such a system in order to solve it numerically, cf. Chapter 2.4 as well as Chapter 7.3.

The Hamiltonian and the Lagrangian formulation are equivalent to each other. In order to show this, the Hamiltonian function h is viewed as the Legendre transformation of the Lagrangian $\mathcal{L}$ with respect to the second variable, i.e.

$$h(q, p) := \max_{\dot{q}} \{p \cdot \dot{q} - \mathcal{L}(q, \dot{q})\} \tag{2.6}$$

If we additionally demand that for any q, the expression $p = \frac{\partial \mathcal{L}}{\partial \dot{q}}(q, \dot{q})$ defines a continuously differentiable bijection $\dot{q} \mapsto p$ for all $k \in \{1 \dots d\}$, we can prove the following theorem.

Theorem 2.6 *The Euler–Lagrange equations*

$$\frac{d}{dt}\partial_{\dot{q}}\mathcal{L} = \partial_q\mathcal{L} \tag{2.2}$$

are equivalent to Hamilton's equations of motion for $z := (q,p)^T$*, i.e.*

$$\dot{z} = \mathcal{J}\,\nabla h(z). \tag{2.3}$$

where h *is defined by 2.6.*

Proof The assumtion that

$$p_k = \frac{\partial\mathcal{L}}{\partial\dot{q}_k}(q,\dot{q}) \tag{2.7}$$

is a differentiable bijection shows that the Legendre transform (2.6) is equivalent to

$$h(q,p) = p\cdot\dot{q}(p) - \mathcal{L}(q,\dot{q}(p))$$

where $\dot{q}$ is understood as a function of p by inverting (2.7). Moreover, it immediately implies that

$$\begin{aligned}\partial_p h &= \dot{q} + p\,(\partial_p\dot{q}) - \partial_{\dot{q}}\mathcal{L}\,(\partial_p\dot{q}) = \dot{q}\\ \partial_q h &= p\cdot(\partial_q\dot{q}) - \partial_q\mathcal{L} - \partial_{\dot{q}}\mathcal{L}\cdot(\partial_q\dot{q}) = -\partial_q\mathcal{L}.\end{aligned}$$

These two expressions allow us to show the equivalence between the Hamilton's and the Euler–Lagrange equations. Given (2.2), we have

$$\begin{aligned}\partial_p h &= \tfrac{d}{dt}q\\ \partial_q h &= -\partial_q\mathcal{L} = -\tfrac{d}{dt}\partial_{\dot{q}}\mathcal{L} = -\tfrac{d}{dt}p\end{aligned}$$

which are precisely Hamilton's equations of motion. Conversely, if we assume (2.3) we get

$$\partial_q\mathcal{L} = -\partial_q h = \tfrac{d}{dt}p = \tfrac{d}{dt}\partial_{\dot{q}}\mathcal{L}$$

which are again the Euler–Lagrange equations (2.2). □

An immediate consequence of this is the following formula for the action integral.

Corrolary 2.7 *The action integral* (2.1) *along the flow of a Hamiltonian system* (2.3) *can be reformulated as*

$$S(t,z) = \int_0^t \left(\tfrac{\mathrm{d}}{\mathrm{d}\tau}X^\tau(z)\cdot\Xi^\tau(z) - h(\Phi^\tau(z))\right)\,\mathrm{d}\tau \tag{2.8}$$

where the z *variable describes the dependence of* S *on the initial values.*

2.4 Numerical Methods

The governing equations for all classical mechanical systems present themselves as ordinary differential equations. There is a sheer endless variety of numeric integrators at hand. Since we have just seen that nature seems to 'behave in a symplectic way' the method of choice has to reflect that.

Definition 2.8 (Symplectic method) *A numerical one-step method*

$$z_1 = \Psi^\tau(z_0) \tag{2.9}$$

is called symplectic if the map Ψ^τ is symplectic whenever the method is applied to a smooth Hamiltonian system.

The symplectic method I use for the calculation of classical trajectories is based on the Størmer–Verlet scheme which I describe in detail in Chapter 7.3. As I have already stated, symplectic methods have the same geometric properties as flows of Hamiltonian systems. But what about their approximation properties? Hairer, Lubich, and Wanner [HLW06, Ch. IX] give an elaborate explanation based on the concept of backward error analysis. We will take a closer look at this in Chapter 9 since I use this kind of analysis in order to prove the approximation properties of the Herman–Kluk propagator.

Chapter 3

Quantum dynamics

3.1 Fundamental concepts

The word ‘quantum’[1] appears in scientific journals at the beginning of the 20th century. Planck [Pla01] described the concept of quantisation while trying to understand the emission of radiation from heated objects. As a result of his experiments, Planck also proposed the numerical value of h, now known as the Planck constant. He also found a more precise value for the Avogadro number and the unit of electrical charge while using the word ‘quanta’ in the sense of ‘quanta of matter and electricity’. In response to Planck’s work, Einstein [Ein05] postulated that radiation existed in spatially localised packets which he called „Lichtquanta“, quanta of light. Bohr [Boh13] eventually introduced his famous model for the hydrogen atom where electrons can only orbit the nucleus in certain ‘stationary orbits’ without radiating, i.e. ‘a miniature representation of the solar system with little planets going around in orbits’. Even at the time, the Bohr model barely provided a partial theoretical explanation to the experimental results.

Heisenberg and Schrödinger

It required the revolutionary works of Heisenberg [Hei25; Hei27] and Schrödinger [Sch26a; Sch26b; Sch26c; Sch26d] to find a satisfactory mathematical theory of

[1] from the Latin ‘quantus’ meaning ‘how much’

systems at on atomic and even smaller length scales. Their mutually equivalent[2] formulations provide an axiomatic basis for quantum mechanics. In the years to follow, these foundations of quantum mechanics were expanded by numerous authors including exceptional scientists like Born, von Neumann, Dirac, Fermi, Pauli, Hilbert, and many others.

I want to shortly excerpt from one of the Schrödinger's original texts in order to demonstrate how little has changed in the formalism since its original formulation almost a century ago. Schrödinger [Sch26d] realises that the equations of what he calls a re-establishment of mechanics are easier if one assumes that the fundamental quantity is a complex-valued function. This way he finds that the time dependence of a wave function is given by

$$\frac{\partial \psi}{\partial t} = \pm \frac{2\pi \mathrm{i}}{h}\, E\, \psi \tag{3.1}$$

He then eliminates the energy parameter E and obtains the two equations [Sch26d, Eq. (4")],

$$\Delta\psi - \frac{8\pi^2}{h^2} V\psi \mp \frac{4\pi \mathrm{i}}{h} \frac{\partial \psi}{\partial t} = 0. \tag{3.2}$$

They have since become the fundamental law of describing non-relativistic particles in modern science. Schrödinger then postulates that 'the complex wave function ψ satisfies one of these equations'. The other is then solved by its complex conjugate. Furthermore he states that, 'if necessary, the real part of ψ may then be viewed as the real wave function'[3]. Born later establishes a statistical interpretation where $|\psi|^2$ is interpreted as a probability density. I will explain this in more detail after giving some thoughts to the solvability of Schrödinger equations.

Existence of dynamics

So far we have not concerned ourselves with a crucial question. Is there a solution to the Schrödinger equation? And if so, is it unique? We may find very general answers using results from the spectral theory of self-adjoint operators on Hilbert spaces. This field was founded by John von Neumann [Neu32a] in his masterpiece

[2] „Ich habe daher im folgenden einen etwas anderen Weg betreten, der rechnerisch außerordentlich viel einfacher ist und den ich für prinzipiell richtig halte." [Sch26d]

[3] „Wir werden verlangen, daß die komplexe Wellenfunktion ψ einer der beiden Gleichungen genüge. Da alsdann die konjugiert komplexe Funktion $\bar{\psi}$ der anderen Gleichung genügt, wird man als reelle Wellenfunktion (wenn man sie benötigt) den Realteil von ψ ansehen dürfen"

„Mathematische Grundlagen der Quantenmechanik“ [Neu96, reprint]. The following Theorem was originally stated by Stone [Sto32, Thm. A]. Von Neumann [Neu32b] later generalised the statement of the theorem and gave an independent proof.

Theorem 3.1 *If H is a self-adjoint transformation in an abstract Hilbert space $\mathcal{H}$. Then $U(t) := \mathrm{e}^{itH}$, $t \in \mathbb{R}$, is a family of unitary transformations with the group property*

$$U(s+t) = U(s)\,U(t) \tag{3.3}$$

and the continuity property

$$\|U(t)\psi - \psi\| \to 0 \text{ as } t \to 0. \tag{3.4}$$

The expression e^{itH} is understood by means of spectral calculus. As a consequence of Stone's theorem we know that everything is well defined when dealing with a self-adjoint operator. So far we have not discussed if the Hamiltonians which occur in the Schrödinger equation are in fact of such nature. For many physically relevant cases, the Hamiltonian operators in question are indeed self-adjoint including the one for molecules (3.9) which I introduce below. A noteworthy summary on the questions of which Hamiltonians are self-adjoint and how to prove this is given by Reed and Simon [RS75][Chap. X].

Statistical interpretation

Born [Bor26] observes that Heisenberg's quantum mechanics had so far only been used to calculate stationary states. To explain the phenomenon of transitions he favours Schrödinger's formulation of the theory. He solves the Schrödinger equation for a scattering problem and writes the solution as an asymptotic expansion in terms of eigenfunctions of the unperturbed atom. From this he concludes that if one wants to interpret these findings in terms of particles the only possible solution is to interpret the coefficients in the expansion as *probabilities* that an electron is scattered in a certain direction. In a footnote he admits that the probability is actually proportional to the square of the coefficients[4]. This leads to the usual statistical interpretation of quantum mechanics which interprets the square of the absolute value of the wave function, $|\psi|^2$, as a probability density. If the system is

[4] *„Anmerkung bei der Korrektur: Genauere Überlegung zeigt, daß die Wahrscheinlichkeit dem Quadrat der Größe proportional ist.“*

in a state $\psi \in \mathcal{H}$, then the probability that the particle is found in a region Ω is given by

$$\mathbb{P}(x \in \Omega|\psi) = \int_{\Omega} |\psi(x)|^2 \, \mathrm{d}x$$

What is nowadays called Born's rule refers to an even more comprehensive setting.

Definition 3.2 (Born's Rule) *Any self-adjoint operator $\mathcal{A} : D(\mathcal{A}) \subset \mathcal{H} \to \mathcal{H}$ is called an observable. Its average or expected value in the state $\psi \in D(\mathcal{A})$ is given by*

$$\mathbb{E}[\mathcal{A}] = \langle \psi, \mathcal{A}\psi \rangle \, . \tag{3.5}$$

Even more general, one defines the probability $\mathbb{P}(\mathcal{A} \in \Omega|\psi)$ that a result in $\Omega \subset \mathbb{R}$ is found when an observable $\mathcal{A}$ is measured by virtue of the unique spectral measure corresponding to $\mathcal{A}$ that is defined by the spectral theorem.

3.2 Quantisation

Quantisation is a mathematical setting for the correspondence principle between classical and quantum mechanics that was introduced by Bohr [Boh20]. It associates a quantum observable, i.e. a self-adjoint operator on $\mathrm{L}^2(\mathbb{R}^d)$, to every classical observable, i.e. real-valued function on phase space. A natural way of doing this was first introduced by Weyl [Wey27] and is called Weyl quantisation accordingly.

Definition 3.3 (Weyl quantisation) *Let $a \in \mathcal{S}(\mathbb{R}^{2d})$ and $\psi \in \mathcal{S}(\mathbb{R}^d)$ be Schwartz functions. The action of the Weyl quantisation $\mathrm{op}^{\varepsilon}(a)$ of a on the state ψ is given by the absolutely convergent integral*

$$\left(\mathrm{op}^{\varepsilon}(a)\psi\right)(x) = (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^{2d}} a\left(\tfrac{1}{2}(x+y), \xi\right) \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}\xi(x-y)} \psi(y) \, dy \, d\xi. \tag{3.6}$$

3.3 Quantum molecular dynamics

A molecule is an ensemble of N nuclei and L electrons which are bound together by the electrostatic force between them. All electrons carry the same mass m and charge $-\mathrm{e}$.[5] The individual masses of the nuclei are denoted by M_n and their electric charges by $Z_n e$ for all $n \in \{1 \dots N\}$. The Hamiltonian operator describing such a molecule is composed of several terms. Let us denote the coordinates for the electronic degrees of freedom by $y_l \in \mathbb{R}^d$ and the ones for the nuclei by $x_n \in \mathbb{R}^d$. The operators

$$T_{\mathrm{N}} := -\sum_{n=1}^{N} \frac{\hbar^2}{2M_n} \Delta_{x_n} \text{ and } T_{\mathrm{e}} := -\sum_{l=1}^{L} \frac{\hbar^2}{2m} \Delta_{y_l} \tag{3.7}$$

describe the contributions to the kinetic energy of nuclei and electrons respectively. The potential is a sum of the electrostatic repulsion between every pair of nuclei and every pair of electrons and the attraction between nuclei and electrons, i.e.

$$V_{\mathrm{mol}}(x, y) = \sum_{1 \le k < n \le N} \frac{Z_k Z_n \mathrm{e}^2}{\|x_k - x_n\|} + \sum_{1 \le j < l \le N} \frac{\mathrm{e}^2}{\|y_j - y_l\|} - \sum_{n=1}^{N} \sum_{l=1}^{L} \frac{Z_n \mathrm{e}^2}{\|x_n - y_l\|} \tag{3.8}$$

Let us define the molecular Hamiltonian as the sum

$$H_{\mathrm{mol}} = T_{\mathrm{N}} + T_{\mathrm{e}} + V_{\mathrm{mol}} \tag{3.9}$$

of (3.7) and (3.8) respectively. Then, the time-dependent Schrödinger equation

$$\mathrm{i} \frac{d}{dt} \psi(t) = H_{\mathrm{mol}} \psi(t), \quad \psi(0) = \psi_0 \tag{3.10}$$

describes the full quantum-mechanical motion of a molecule if one neglects the spin degrees of freedom and relativistic effects. Kato's Theorem [Kat51, Theorem 1] corroborates that the molecular Hamiltonian H_{mol} is self-adjoint. The spectral theorem (Theorem 3.1) then guarantees the existence and uniqueness of the solution to (3.10).

The time-dependent Born–Oppenheimer approximation

In spite of the knowledge of existence and uniqueness of a solution, the actual computation of it, even for a small-sized molecules, proves to be difficult. Consider

[5] In SI units the elementary charge has a measured value of $\mathrm{e} \approx 1.602176565\ 10^{-19}$ Coulomb and the mass of an electron $m \approx 9.10938215\ 10^{-31}$ kilograms.

e.g. an azane (i.e. ammonia) molecule which consists of one nitrogen and three hydrogen atoms. Counting its 4 nuclei and 10 electrons, the resulting Schrödinger equation is a partial differential equation in 42 variables. As a consequence, efficient approximation schemes alone will not be sufficient. We require a considerable reduction of degrees of freedom, which is provided by the time-dependent Born-Oppenheimer approximation. It has been introduced to the mathematical literature by Hagedorn [Hag80a]. Prior to that, the approximation has already been widely used in the chemical physics literature and is usually attributed to Born and Oppenheimer [BO27] despite the fact that their original paper does not consider time dependent problems at all. Spohn and Teufel [ST01] generalise Hagedorn's analysis from coherent states to arbitrary wave functions.

The underlying physical picture is that the nuclei behave almost classically because of their large mass and that the electrons rapidly adjust to the slow nucleonic motion. This motion is governed by the Born–Oppenheimer Hamiltonian $H^\varepsilon = -\frac{\varepsilon^2}{2}\Delta + V$ and is driven by the potential energy surface V of the electrons. The semiclassical parameter $\varepsilon > 0$ is the square root of the mass ratio of electrons and nuclei and is thus very small. The actual computation of the electronic surfaces, i.e. the solution of the stationary Schrödinger eigenvalue problem for the electrons is one of the central goals in computational quantum chemistry. Here, let us just assume that the problem is solved in a sufficient manner. From now on, we concentrate on the following problem.

Definition 3.4 (The time-dependent semiclassical Schrödinger equation)
Our aim is to find the solution to the time-dependent semiclassical Schrödinger equation

$$\mathrm{i}\varepsilon\frac{d}{dt}\psi(t,x) = H^\varepsilon\psi(t,x) \tag{3.11}$$

for an initial wave function $\psi(0,\cdot) = \psi_0 \in \mathrm{L}^2(\mathbb{R}^d)$ *of norm* $\|\psi_0\| = 1$.

It is this equation that will be the fundamental model for the remainder of this thesis. The fact that $\varepsilon \ll 1$ suggests the analysis through semiclassical methods which are described in the subsequent chapter. The lowest order contributions come from the flow Φ of the classical Hamiltonian function $h(q,p) = \frac{1}{2}p^2 + V(q)$ on nucleonic phase space which is defined such that $H^\varepsilon = \mathrm{op}^\varepsilon(h)$ is the Weyl quantisation of h.

3.4 Numerical methods

Lubich [Lub08] provides a general review of model reduction via variational approximations as well as numerical methods used for time-dependent Schrödinger equations. He describes different methods for space discretisation including Galerkin and collocation approaches using Hermite and Fourier bases. For up to about 10 degrees of freedom these methods can be extended via hyperbolic cross approximations and sparse grids. Several strategies for time-stepping algorithms are presented as well. Schemes based on the Lanczos method or on Chebyshev polynomials are used to approximate the unitary group of a discrete Hamiltonian operator. Splitting methods and their high-order refinements by composition are also presented. Moreover, Lubich [Lub08, Chapter II.3.3] gives an introduction to the multi-configuration time-dependent Hartree (MCTDH) method which was proposed by Meyer, Manthe, and Cederbaum [MMC90]. This method approximates the wave function of the full system by a linear combination of tensor products of wave function for single particles.

Finally, let me mention the split-step Fourier method since I use this particular method to create a reference solution for my numerical experiments. It is based on symmetric Strang operator splitting. Consider an approximation $\psi_n \approx \psi(t_n)$ of the solution to the Schrödinger equation with Hamiltonian $H = T + V$ at time t_n where T and V are the kinetic and potential energy operator respectively. Then the new approximation $\psi_{n+1} \approx \psi(t_{n+1})$ at time $t_{n+1} = t_n + \tau$ is given by

$$\psi_{n+1} = \mathrm{e}^{-\mathrm{i}\frac{\tau}{2}V}\mathrm{e}^{-\mathrm{i}\tau T}\mathrm{e}^{-\mathrm{i}\frac{\tau}{2}V}\psi_n.$$

Lubich [Lub08, Ch. III.3] supplies a detailed account on its structure preserving properties such as unitarity, symplecticity, and time-reversibility as well as error bounds and higher-order methods based on composition strategies. For these reasons, I consider this the method of choice for the solution of a Schrödinger equations in dimensions $d \in \{1, 2, 3\}$ where it is implemented very efficiently using the Fast Fourier Transform.

Chapter 4

Semiclassical methods

There exist numerous approaches for the approximation of solutions to the semiclassical time-dependent Schrödinger equation

$$\mathrm{i}\varepsilon\frac{d}{dt}\psi(t,x) = H^\varepsilon\psi(t,x).$$

Conventional discretisation techniques such as finite differences or the Crank–Nicolson scheme, however, are not the methods of choice. These schemes potentially produce wrong results under seemingly reasonable discretisation strategies and even if they are stable and consistent, they require huge computational resources in order to give accurate results for both, $d \gg 1$ and $\varepsilon \ll 1$. For details, let me again refer to Jin, Markowich, and Sparber [JMS11] and the references therein. They also provide a review on semiclassical techniques such as Gaussian beam methods. Nevertheless they fail to mention Hagedorn's semiclassical wave packets as well as quasi-classical approximations which I present in Chapters 4.2 and 4.4 respectively.

4.1 Gaussian wave packets

An important class of functions in semiclassical calculations are called Gaussian wave packets $\varphi^\varepsilon[z,C,\theta] \in \mathcal{S}(\mathbb{R}^d)$ and they are defined by

$$\varphi^\varepsilon[z,C,\theta](x) := (\pi\varepsilon)^{-d/4}\exp\left(\tfrac{\mathrm{i}}{2\varepsilon}(x-q)^T C(x-q) + \tfrac{\mathrm{i}}{\varepsilon}\,p\cdot(x-q) + \tfrac{\mathrm{i}}{\varepsilon}\,\theta\right). \qquad (4.1)$$

Each individual function is characterised by a set of three parameters, a phase space point $z = (q, p)^T \in \mathbb{R}^{2d}$ around which it is centred, a complex symmetric matrix $C = C^T \in \mathbb{C}^{d\times d}$ with positive definite imaginary part which determines its width, and a global phase $\theta \in \mathbb{C}$. The phase θ is chosen properly with respect to the width matrix C such that

$$\|\varphi^\varepsilon[z, C, \theta]\|^2 = \int_{\mathbb{R}^d} |\varphi^\varepsilon[z, C, \theta](x)|^2 \, \mathrm{d}x = 1.$$

A general Gaussian wave packet can be propagated unitarily by evolving its centre along the flow of the classical Hamiltonian system

$$\dot{z} = \mathcal{J}\nabla h(z) \tag{2.3}$$

which is supplemented by a Riccati equation for the width matrix C and an ordinary differential equation for θ in order to ensure the correct phase and normalisation. Then, one finds for every $T > 0$ a constant $c \geq 0$ such that for all $\varepsilon > 0$

$$\sup_{t\in[0,T]} \|\varphi_0^\varepsilon[z_t, C_t, \xi_t] - \mathrm{U}_t^\varepsilon \varphi_0^\varepsilon[z_0, C_0, \xi_0]\| \leq c\sqrt{\varepsilon}.$$

In particular, if the potential function V is a polynomial of degree ≤ 2, then $c = 0$, and the Gaussian wave packet approximation is exact. Over decades, general Gaussian wave packets have been used as a flexible tool in chemical physics, see e.g. Heller [Hel76] or Littlejohn [Lit86]. They have also been considered more recently for the systematic construction of numerical integrators by Faou and Lubich [FL06].

4.2 Hagedorn's semiclassical wave packets

Hagedorn [Hag80b; Hag98] notices that, when considering a general Gaussian wave packet (4.1), important features and conclusions are obtained by factorising the matrix C into two matrices with special properties. A key result in this context is the following lemma.

Lemma 4.1 (Hagedorn [Hag98, Section 3]) *Any complex symmetric matrix $C \in \mathbb{C}^{d\times d}$ with positive definite imaginary part can be represented as a product $C = PQ^{-1}$, where $P, Q \in \mathbb{C}^{d\times d}$ are invertible and satisfy*

$$\begin{aligned} Q^T P - P^T Q &= 0, \\ Q^* P - P^* Q &= 2\mathrm{i}\,\mathrm{I}. \end{aligned} \tag{4.2}$$

Conversely, for any two complex matrices $P, Q \in \mathbb{C}^{d\times d}$ that satisfy (4.2)*, the product $C := PQ^{-1}$ is complex symmetric with positive definite imaginary part $\operatorname{Im} C = (QQ^*)^{-1}$.*

Note that I use the a notation that is also favoured by Lubich [Lub08, Chapter V.1]. The matrices Q and P in Lemma 4.1 correspond to the matrices A and $\mathrm{i}B$ in Hagedorn's original paper. Furthermore, Lubich observes that the conditions in (4.2) are equivalent to the matrix

$$\begin{pmatrix} \operatorname{Re} Q & \operatorname{Im} Q \\ \operatorname{Re} P & \operatorname{Im} P \end{pmatrix}$$

being symplectic. In order to facilitate notation, let me introduce the rectangular matrix

$$Z = \begin{pmatrix} Q \\ P \end{pmatrix} \in \mathbb{C}^{2d\times d}$$

for matrices Q and P satisfying (4.2). This motivates the definition of the wave packet

$$\varphi_0^\varepsilon[z, Z](x) := (\pi\varepsilon)^{-d/4} \det(Q)^{-1/2} \exp\left(\tfrac{\mathrm{i}}{2\varepsilon}(x-q)\cdot PQ^{-1}(x-q) + \tfrac{\mathrm{i}}{\varepsilon}p\cdot(x-q)\right)$$

which is parametrised by the matrix Z and a phase space point $z = (q,p)^T \in \mathbb{R}^{2d}$. The correct normalisation, i.e.

$$\|\varphi_0^\varepsilon[z, Z]\| = 1$$

is a consequence of (4.2). Taking $\varphi_0^\varepsilon[z, Z]$ as a starting point allows an elegant construction of an orthonormal basis

$$\{\varphi_k^\varepsilon[z, Z], k \in \mathbb{N}^d\}$$

of $\mathrm{L}^2(\mathbb{R}^d)$. The basis functions

$$\varphi_k^\varepsilon[z, Z] = \frac{1}{\sqrt{k!}} A^\dagger[z, Z]^k \varphi_0^\varepsilon[z, Z]$$

are constructed by the iterated application of the raising operator

$$A^\dagger[z, Z] = \frac{\mathrm{i}}{\sqrt{2\varepsilon}} Z^* J(\hat{z} - z), \text{ where } (\hat{z}\psi)(x) = \begin{pmatrix} x\psi(x) \\ -\mathrm{i}\varepsilon\nabla\psi(x) \end{pmatrix}.$$

For the unitary propagation of these semiclassical wave packets one complements the Hamiltonian equations (2.3) by a rectangular version of its variational equation, i.e.

$$\dot{Z}(t) = \mathcal{J}\nabla^2 h(z(t))Z(t) \tag{4.3}$$

and the action integral $S(t)$ defined by (2.8). This generalises the Gaussian wave packet approximation discussed in the previous section as follows.

Theorem 4.2 (Hagedorn [Hag98, Theorem 2.9]) *For all $k \in \mathbb{N}^d$ and all $T > 0$ there exists a constant $c \geq 0$ such that for all $\varepsilon > 0$*

$$\sup_{t\in[0,T]} \left\| \mathrm{U}_t^\varepsilon \varphi_k^\varepsilon[z(0), Z(0)] - \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(t)} \varphi_k^\varepsilon[z(t), Z(t)] \right\| \leq c\sqrt{\varepsilon}.$$

Furthermore, Hagedorn [Hag98, Theorem 2.5] shows that if the potential V is a quadratic polynomial, then $c = 0$, which means that in this case $\mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(t)}\varphi_k^\varepsilon[z(t), Z(t)]$ is the exact solution to the Schrödinger equation. Using this property for harmonic Hamiltonians Faou, Gradinaru, and Lubich [FGL09] and Gradinaru and Hagedorn [GH14] develop Galerkin methods with time-splitting for a convergent discretisation based on the unitary evolution of Hagedorn's semiclassical wave packets.

4.3 Gaussian beam methods

A complementary type of semiclassical approximations is built for initial data that is less localised in position space than semiclassical wave packets. Wentzel–Kramers–Brillouin (WKB) wave functions

$$w_0^\varepsilon(x) = \alpha_0(x)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}\sigma_0(x)}, \quad x \in \mathbb{R}^d,$$

are defined by a complex-valued amplitude function $\alpha_0 \in C^\infty(\mathbb{R}^d)$ and a real-valued phase function $\sigma_0 \in C^\infty(\mathbb{R}^d)$. A first order Gaussian beam approximation of the unitary Schrödinger dynamics carries WKB initial data by continuously superimposing general Gaussian wave packets according to

$$b^\varepsilon(t) = (2\pi\varepsilon)^{-d/2} \int_{\Lambda_0} \alpha(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}\sigma(t,z)}\, \varphi^\varepsilon[\Phi^t(z), C(t,z), \sigma(t,z)]\, \mathrm{d}z. \tag{4.4}$$

The centres of the initial Gaussians are chosen from the set

$$\Lambda_0 = \{(x, \nabla\sigma_0(x)) \mid x \in \operatorname{supp}(a_0)\},$$

while the propagation of the beam parameters $\alpha(t,z) \in \mathbb{C}$, $\sigma(t,z) \in \mathbb{R}$, and $C(t,z) \in \mathbb{C}^{2d\times d}$ is determined by a system of coupled ordinary differential equations driven by the classical Hamiltonian flow Φ^t. Its building blocks resemble the variational equation and the equation for the action integral. Zhen [Zhe14, Theorem 5.1] proves that for all $T > 0$ there exists a constant $c \geq 0$ such that for all $\varepsilon > 0$

$$\sup_{t\in[0,T]} \|\mathrm{U}_t^\varepsilon w_0^\varepsilon - b_t^\varepsilon\| \leq c\,\varepsilon.$$

Higher order Gaussian beam approximations with $\mathcal{O}(\varepsilon^{N/2})$ accuracy, $N \in \mathbb{N}$, have been developed as well [LRT13]. The discretisation of the continuous Gaussian beam superposition (4.4) and its higher order versions have been addressed by grid based numerical quadrature. Thus, the numerical applications have been restricted to systems in dimension $d = 1$ and $d = 2$.

4.4 Quasi-classical approximations

The methods we have discussed so far all provide approximations to the full time-evolved wave function $\psi(t,\cdot) = \mathrm{U}_t^\varepsilon\psi_0$. Practical applications often need derived quadratic quantities. This is particularly the case in physics and chemistry since expectation values

$$\mathbb{E}_{\mathrm{qm}}[\mathcal{A}](t) = \langle\psi(t), \mathcal{A}\psi(t)\rangle$$

for a given self-adjoint operator $\mathcal{A}$ are the quantum mechanical quantities that can be actually measured in experiments. Typical observables are ε-scaled pseudodifferential operators and can be expressed as the Weyl quantisation $\mathcal{A} = \mathrm{op}^\varepsilon(a)$ of a smooth phase space function $a : \mathbb{R}^{2d} \to \mathbb{R}$. Chapter 6 will provide the definition of such objects and show them to be well-defined. The most popular quasi-classical approximation [Mil74; TW04; LR10] takes advantage of the properties of the Wigner transform.

The Wigner Transform

Wigner [Wig32] made an attempt to express quantum mechanical expectation values in the same form as the averages in classical statistical mechanics. As a consequence he introduced the following transformation.

Definition 4.3 *For any $\psi \in \mathrm{L}(\mathbb{R}^d)$ and $z = (q,p) \in \mathbb{R}^{2d}$ let us define*

$$W\psi(z) := (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^d} \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} p\cdot y}\, \overline{\psi\left(q + y/2\right)}\, \psi\left(q - y/2\right)\, dy.$$

The function $W\psi : \mathbb{R}^{2d} \to \mathbb{R}$ is called Wigner transform of ψ.

One of the best-known properties of the Wigner transform is that it has features similar to those of a probability distribution.

Proposition 4.4 *Assume that $\psi \in \mathrm{L}^1(\mathbb{R}^d) \cap \mathrm{L}^2(\mathbb{R}^d)$ and denote by $\mathcal{F}$ the ε-scaled Fourier transform. Then*

$$\int_{\mathbb{R}^d} W\psi(z)\,\mathrm{d}p = |\psi(q)|^2 \quad \text{and} \quad \int_{\mathbb{R}^d} W\psi(z)\,\mathrm{d}q = |\mathcal{F}\psi(p)|^2 \tag{4.5}$$

and hence

$$\int_{\mathbb{R}^{2d}} W\psi(z)\,\mathrm{d}z = \|\psi\|_{\mathrm{L}^2(\mathbb{R}^d)} = \|\mathcal{F}\psi\|_{\mathrm{L}^2(\mathbb{R}^d)}\,. \tag{4.6}$$

A proof of this result, amongst many others, can be found in the enlightening book by Gosson [Gos06, Ch. 6.4]. It follows immediately from (4.6) that $\int_{\mathbb{R}^{2d}} W\psi(z)\,\mathrm{d}z = 1$ if $\|\psi\| = 1$. Therefore the Wigner transform of a normalised wave function is sometimes called a quasi-probability density. It would be a probability density if in addition $W\psi \geq 0$ was true, which is not the case for any wave functions ψ. In fact, the Wigner quasi-probability density $W\psi$ is a true density if and only if ψ is the exponential of a quadratic polynomial, i.e. a Gaussian wave packet. This was shown by Hudson [Hud74] for one-dimensional systems and generalised by Soto and Claverie [SC83] for systems with an arbitrary number of degrees of freedom.

Example 4.5 *Consider a general Gaussian wave packet* (4.1) *centred at $z_0 \in \mathbb{R}^{2d}$ with width matrix $C = \mathrm{iI}$ and global phase $\theta = 0$, i.e.*

$$\varphi(x) := \varphi^\varepsilon[z_0, \mathrm{I}, 0](x) = (\pi\varepsilon)^{-d/4} \exp\left(-\tfrac{1}{2\varepsilon}\,|x - q_0|^2 + \tfrac{\mathrm{i}}{\varepsilon}\, p_0 \cdot (x - q_0)\right).$$

Its Wigner transform is then given by

$$\begin{aligned} W\varphi(z) &= (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^d} \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} p\cdot y}\, \overline{\varphi\left(q + y/2\right)}\, \varphi\left(q - y/2\right)\, \mathrm{d}y \\ &= (\pi\varepsilon)^{-d/2}\, \mathrm{e}^{-\frac{1}{\varepsilon}|q-q_0|^2}\, (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^d} \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}(p-p_0)\cdot y - \frac{1}{4\varepsilon}|y|^2}\, \mathrm{d}y \\ &= (\pi\varepsilon)^{-d/2}\, \mathrm{e}^{-\frac{1}{\varepsilon}|q-q_0|^2}\, (\pi\varepsilon)^{-d/2}\, \mathrm{e}^{-\frac{1}{\varepsilon}|p-p_0|^2}. \end{aligned}$$

This expression shows that $W\varphi > 0$ and makes it easy to check that the two formulae in (4.5) *are fulfilled.*

There is a fundamental relationship between the Wigner transform and the Weyl pseudo-differential calculus which we will see in Chapter 6. Let me just state a result due to Moyal [Moy49]. It permits us to express the expectation value of a Weyl quantised operator $\mathcal{A} = \mathrm{op}^\varepsilon(a)$ (cf. Definition 3.3) in terms of the Wigner transform and the Weyl symbol.

Proposition 4.6 *Let $\mathcal{A} = \mathrm{op}^\varepsilon(a)$ and $\psi \in \mathrm{L}^2(\mathbb{R}^d)$ be a wave function of norm $\|\psi\| = 1$. Then the expectation value of $\mathcal{A}$ in the state ψ may be expressed by the formula*

$$\langle \psi, \mathcal{A}\psi \rangle = \int_{\mathbb{R}^{2d}} a(z) W\psi(z)\, \mathrm{d}z. \tag{4.7}$$

This suggests that the expectation value of $\mathcal{A}$ is obtained by 'averaging' the classical observable a with respect to $W\psi$.

Egorov's Theorem

An idea on how to calculate an approximation to the time evolution of expectation values of a Weyl quantised operator may be derived from formula (4.7). What if, instead of considering the full quantum evolution, we just evolved the classical observable a along the Hamiltonian flow $\Phi^t : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$? Let us define this quasi-classical expectation value by

$$\mathbb{E}_{\mathrm{qcl}}[a](t) = \int_{\mathbb{R}^{2d}} (a \circ \Phi^t)(z)\, W(\psi_0)(z)\, \mathrm{d}z. \tag{4.8}$$

Is this an adequate approximation of the quantum expectation value $\mathbb{E}_{\mathrm{qm}}[\mathcal{A}](t)$? An answer is provided by Egorov [Ego69] through the following theorem.

Theorem 4.7 *For all $T > 0$ there exists a constant $c \geq 0$ such that for all $\varepsilon > 0$*

$$\sup_{t\in[0,T]} |\mathbb{E}_{\mathrm{qm}}[\mathcal{A}](t) - \mathbb{E}_{\mathrm{qcl}}[a](t)| \leq c\,\varepsilon^2. \tag{4.9}$$

The constant c depends on the observable $\mathcal{A}$ and derivatives of the flow Φ^t, but is uniformly bounded for all normalised initial data with $\|\psi_0\| = 1$.

Robert [Rob87] supplies a rigorous and more modern statement and proof of this result. This theorem permits the calculation of expectation values up to an error of order ε^2 by only knowing the dynamics of the corresponding classical equations of motion. Methods based on this approach are also called linearised semiclassical initial value representation (LSC-IVR).

Remark 4.8 As for the Hagedorn wave packets, it holds that $c = 0$ if the potential V is a quadratic polynomial, i.e. the time evolution is exact in this case.

In Part III we will take a closer look on a specific quasi-classical approximation. Lasser and Röblitz [LR10] introduce a discretisation scheme based on (4.8). Due to its inherently parallel structure I use it as a starting point to design an efficient method for high dimensional problems. Details are presented in Chapter 11.4.

Part II

Discretising the Herman–Kluk Propagator

Chapter 5

The Herman–Kluk propagator

Herman and Kluk [HK84] observe that 'if wave functions become delocalised after relatively short periods of time, a single wave packet cannot accurately approximate a quantum system'. For this reason they propose to use a set of localised packets instead and provide a formal justification of this method that we now call Herman–Kluk propagator. A similar heuristic justification is proposed by Heller [Hel81] for the Frozen Gaussian approximation.

This chapter provides an intuitive introduction to the Herman–Kluk propagator, a formal definition and an idea why it could be a useful approximation of the unitary time evolution group. Chapter 6 provides a rigorous definition based on the theory of pseudo-differential and Fourier integral operators and results on the approximation properties of a class of operators which contains the Herman–Kluk propagator as a prominent example.

5.1 Gaussian wave packets and the FBI Transform

As introduced in Chapter 4.1, Gaussian wave packets are an important tool in semi-classical analysis. Let me remind you of the definition of a general Gaussian wave packet $\varphi^\varepsilon[z, \mathrm{I}, 0]$ in (4.1). It is characterised by a centre point $z = (q, p) \in \mathbb{R}^{2d}$, a complex symmetric width matrix $C \in \mathbb{C}^{d\times d}$ with positive definite imaginary part and a global phase $\theta \in \mathbb{C}$. Since we will study the special case of $C = \mathrm{I}$ and $\theta = 0$ most of the time, I would like to introduce the following notation.

Definition 5.1 *Let us define a Gaussian wave packet $g_z^\varepsilon \in \mathcal{S}(\mathbb{R}^d)$ centred at $z = (q,p)^T \in \mathbb{R}^{2d}$ by*

$$g_z^\varepsilon(x) := \varphi^\varepsilon[z, \mathrm{I}, 0](x) = (\pi\varepsilon)^{-\frac{d}{4}} \exp\left(-\frac{1}{2\varepsilon}\,|x-q|^2 + \frac{\mathrm{i}}{\varepsilon}\,p\cdot(x-q)\right). \tag{5.1}$$

Again, $\mathcal{S}(\mathbb{R}^d)$ denotes the space of Schwartz functions over $\mathbb{R}^d$.

Iagolnitzer [Iag75] uses these wave packets to develop a generalisation of the Fourier transform.

Definition 5.2 *For all $\psi \in \mathcal{S}(\mathbb{R}^d)$ and $z = (q,p)^T \in \mathbb{R}^{2d}$ let us define $T^\varepsilon\psi \in \mathcal{S}(\mathbb{R}^{2d})$ by*

$$(T^\varepsilon\psi)(z) := (2\pi\varepsilon)^{-d/2}\,\langle g_z^\varepsilon, \psi\rangle = (2\pi\varepsilon)^{-d/2}\,(\pi\varepsilon)^{-\frac{d}{4}}\,\mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}p\cdot q}\int\limits_{\mathbb{R}^d} \mathrm{e}^{-\frac{\mathrm{i}}{\varepsilon}p\cdot x}\,\psi(x)\,\mathrm{e}^{-\frac{1}{2\varepsilon}|x-q|^2}\,\mathrm{d}x$$

This introduces a mapping

$$T^\varepsilon : \mathcal{S}(\mathbb{R}^d) \longrightarrow \mathcal{S}(\mathbb{R}^{2d}), \quad \psi \mapsto T^\varepsilon\psi$$

which is called Fourier–Bros–Iagolnitzer (FBI) transform.

Martinez [Mar02, Chapter 3.1] rigorously proves this transformation to be well-defined. Furthermore, he shows that T^ε maps $\mathrm{L}^2(\mathbb{R}^d)$ isometrically into $\mathrm{L}^2(\mathbb{R}^{2d})$ and that for all $\psi \in \mathrm{L}^2(\mathbb{R}^d)$ the convenient formula

$$\psi = (2\pi\varepsilon)^{-d}\int\limits_{\mathbb{R}^{2d}} g_z^\varepsilon\,\langle g_z^\varepsilon, \psi\rangle\,\mathrm{d}z \tag{5.2}$$

holds. From this we get the formal expression

$$\mathrm{U}_t^\varepsilon\psi = (2\pi\varepsilon)^{-d}\int\limits_{\mathbb{R}^{2d}} (\mathrm{U}_t^\varepsilon g_z^\varepsilon)\,\langle g_z^\varepsilon, \psi\rangle\,\mathrm{d}z.$$

which motivates to approximation of the unitary propagator U_t^ε by the approximation of the unitary propagation of Gaussian wave packets, which should be easier and require less effort. In the chemical literature such methods are known as Initial Value Representations (IVR), see [TW04]. From a mathematical point of view they are Fourier Integral Operators with complex valued phase functions which are

defined in Chapter 6. Heller [Hel81] institutes 'a very simple semiclassical approximation'

$$U_t^\varepsilon g_z^\varepsilon \;\approx\; \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(t,z)} g^\varepsilon_{\Phi^t(z)}.$$

which is called Frozen Gaussian approximation. It evolves the wave packet's centre according to the classical flow Φ^t of the Hamiltonian system $\dot{z} = \mathcal{J}\nabla h(z)$ (2.3) with initial datum $z(0) = z$. The phase of the wave packet changes according to the action integral (2.8) along the classical trajectory as described in Chapter 4.1 while its width stays 'frozen'.

5.2 Definition of the Herman–Kluk propagator

The approximate propagator established by Herman and Kluk in [HK84] is more sophisticated as it accounts for the changes of the width of a wave packet. Let me recall that we aim to find an approximation of the solution to the time-dependent semiclassical Schrödinger equation

$$\mathrm{i}\varepsilon\frac{d}{dt}\psi(t,x) = H^\varepsilon\psi(t,x) \tag{3.11}$$

for $\varepsilon \ll 1$. The Hamiltonian operator $H^\varepsilon = \mathrm{op}^\varepsilon(h)$ is the Weyl quantisation of the classical Hamiltonian function $h(q,p) = \frac{1}{2}p^2 + V(q)$ on phase space. The definition of the Herman–Kluk propagator requires quantities that are derived from the classical flow (2.5) of this particular Hamiltonian function.

Definition 5.3 *For any initial wave function $\psi_0 \in \mathrm{L}^2(\mathbb{R}^d)$ and time $t \in [0,T]$ the Herman–Kluk propagator is defined by*

$$\mathcal{I}_t^\varepsilon\psi_0 = (2\pi\varepsilon)^{-d}\int\limits_{\mathbb{R}^{2d}} u(t,z)\,\mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(t,z)} g^\varepsilon_{\Phi^t(z)}\;\langle g_z^\varepsilon,\psi_0\rangle\;\mathrm{d}z. \tag{5.3}$$

As mentioned, its time evolution depends on the classical flow $\Phi^t = (X^t,\Xi^t)$ of the Hamiltonian system (2.3) *and its corresponding action* (2.8)*, i.e.*

$$S(t,z) = \int\limits_0^t \left(\tfrac{\mathrm{d}}{\mathrm{d}\tau}X^\tau(z)\cdot\Xi^\tau(z) - h(\Phi^\tau(z))\right)\mathrm{d}\tau.$$

The quantity $u(t,z)$ is called Herman–Kluk prefactor. It incorporates the components of the Jacobian matrix of the flow

$$(D\Phi^t)(z) = \begin{pmatrix} \partial_q X^t(z) & \partial_p X^t(z) \\ \partial_q \Xi^t(z) & \partial_p \Xi^t(z)\end{pmatrix} \in \mathbb{R}^{2d\times 2d}$$

and is defined by

$$u(t,z) := \sqrt{2^{-d} \det \left(\partial_q X^t(z) - \mathrm{i}\partial_p X^t(z) + \mathrm{i}\partial_q \Xi^t(z) + \partial_p \Xi^t(z) \right)} \tag{5.4}$$

for all $z = (q,p)^T \in \mathbb{R}^{2d}$.

Rousse and Swart [RS08] prove the well-posedness of a general class of operators which includes the Herman–Kluk propagator $\mathcal{I}_t^\varepsilon$. I present some of their results in Chapter 6.1 after introducing a precise definition of Fourier Integral Operators. Furthermore, Swart and Rousse [SR09, Theorem 2] provide a theorem on the approximation properties of such operators.

Theorem 5.4 (Swart and Rousse [SR09, Theorem 2]) *Let* U_t^ε *be the unitary time evolution of* (3.10) *with subquadratic potential* V*. The Herman–Kluk propagator* $\mathcal{I}_t^\varepsilon$ *satisfies*

$$\sup_{t\in[0,T]} \| \mathrm{U}_t^\varepsilon - \mathcal{I}_t^\varepsilon \|_{\mathrm{L}^2(\mathbb{R}^d)\to\mathrm{L}^2(\mathbb{R}^d)} \leq c\, \varepsilon,$$

where $T > 0$ *is a fixed time and* $c > 0$ *is independent of* ε.

This theoretical approximation estimate calls for a numerically stable and efficient algorithm to actually compute the Herman–Kluk propagator in practice. This is one of the contributions of the present thesis. Chapter 7 describes a discretisation scheme for the Herman–Kluk propagator. In the process of proving an error estimate for this scheme, cf. Theorem 9.1, I also review the idea of a proof of Theorem 5.4.

Remark 5.5 As an intermediate result of the original proof, [SR09, Lemma 2.] shows that for any time $t \in [0,T]$ the Herman–Kluk prefactor $z \mapsto u(t,z)$ is a smooth function and is uniformly bounded away from 0.

Remark 5.6 Note that for $t = 0$ the Herman–Kluk propagator reduces to the FBI inversion formula (5.2), i.e. $\mathcal{I}_0^\varepsilon \psi_0 = \psi_0$ for all $\psi_0 \in \mathrm{L}^2(\mathbb{R}^d)$, since $\Phi^0(z) = z$, $u(0,z) = 1$, and $S(0,z) = 0$.

5.3 The Herman–Kluk propagator in momentum space

The Fourier transform of the wave function is of interest in many situations as well, e.g. when calculating the expectation values of the momentum operator $\psi \mapsto \mathrm{i}\varepsilon\nabla\psi$

or the kinetic energy operator $\psi \mapsto -\frac{\varepsilon^2}{2}\Delta\psi$. Since in general we want to avoid the evaluation of the Herman–Kluk wave function on a uniform grid, using the Fast Fourier Transform is not an option. There is, however, a way to calculate the Herman–Kluk propagator and its Fourier transform simultaneously by considering the following argument. For all $\xi \in \mathbb{R}^d$ let

$$(\mathcal{F}\psi)(\xi) = (2\pi\varepsilon)^{-d/2} \int_{\mathbb{R}^d} \mathrm{e}^{-\frac{\mathrm{i}}{\varepsilon}x\cdot\xi}\, \psi(x)\, \mathrm{d}x \tag{5.5}$$

be the ε-scaled Fourier transform. Then,

$$\begin{aligned}\mathcal{F}\left(\mathcal{I}_t^\varepsilon \psi_0\right) &= (2\pi\varepsilon)^{-d}\ \mathcal{F}\left(\int_{\mathbb{R}^{2d}} u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(t,z)} g^\varepsilon_{\Phi^t(z)}\, \langle g^\varepsilon_z, \psi_0\rangle\ \mathrm{d}z\right)\\ &= (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^{2d}} u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(t,z)} \left(\mathcal{F} g^\varepsilon_{\Phi^t(z)}\right)\, \langle g^\varepsilon_z, \psi_0\rangle\ \mathrm{d}z.\end{aligned}$$

This is made rigorous by Fubini's theorem. Once one manages to calculate the Herman–Kluk propagator, it is sufficient to know the Fourier transform of a single Gaussian wave packet, i.e.

$$\mathcal{F} g^\varepsilon_{(q,p)} = \mathrm{e}^{-\frac{\mathrm{i}}{\varepsilon}p\cdot q}\, g^\varepsilon_{(p,-q)},$$

in order to calculate its Fourier transform. In numerical simulations these calculations are performed simultaneously without substantial additional effort.

5.4 Comparison to further semiclassical methods

Let us take another look at the Herman–Kluk prefactor (5.4), i.e.

$$u(t,z) = \sqrt{2^{-d} \det\left(\partial_q X^t(z) + \partial_p \Xi^t(z) + \mathrm{i}\partial_q \Xi^t(z) - \mathrm{i}\partial_p X^t(z)\right)}.$$

As mentioned above, it incorporates the components of the Jacobian matrix

$$(D\Phi^t)(z) = \begin{pmatrix} \partial_q X^t(z) & \partial_p X^t(z) \\ \partial_q \Xi^t(z) & \partial_p \Xi^t(z) \end{pmatrix} \in \mathbb{R}^{2d\times 2d}$$

of the Hamiltonian flow Φ^t. The time evolution of $D\Phi^t$ is determined by the variational equation

$$\dot{W}(t) = \mathcal{J}\,\nabla^2 h(\Phi^t)\, W(t), \quad W(0) = \mathrm{I} \tag{5.6}$$

that corresponds to the Hamiltonian system $\dot{z} = \mathcal{J}\nabla h(z)$. In Chapter 7.3 I explain in detail how to solve (5.6) numerically. Here, I just want to make the following observation. In addition to the classical flow, the Herman–Kluk propagator requires only the solution to the variational equation. This is a feature that is shared with Hagedorn's semiclassical wave packets as well as Gaussian beam methods, cf. Chapters 4.2 and 4.3 respectively. Thus, the reason why these semiclassical methods are suitable approximations to quantum mechanics is encoded in a variational equation similar to (5.6).

In terms of computational effort for the time discretisation of the ordinary differential equations, all three methods are equivalent. The advantage of the Herman–Kluk propagator with respect to Hagedorn's semiclassical wave packets is simply the higher order of approximation, $\mathcal{O}(\varepsilon)$ for Herman–Kluk versus $\mathcal{O}(\sqrt{\varepsilon})$ for the wave packets. With respect to Gaussian beams the superiority lies in the discretisation of the phase space integral. An efficient quadrature rule for the integral in (5.3) is more readily available than for the one in (4.4), particularly in high dimensions, which is illustrated in Chapter 8.

Chapter 6

Fourier Integral Operators

The Herman–Kluk propagator is originally introduced by Herman and Kluk [HK84] using a formal derivation based on the path integral formulation of quantum mechanics. The previous chapter provides a different but still heuristic definition. In this chapter I want to place this intuitive definition on a rigorous foundation by viewing it in the context of Fourier integral operators. Nevertheless, I do not go into too much detail since I want to keep the focus on the development of an efficient discretisation scheme.

6.1 Well-posedness of the Herman–Kluk Propagator

Rousse and Swart [RS08] consider a general class of Fourier integral operators by viewing a symplectic transformation ϕ and a smooth function v as parameters of the operator

$$\mathcal{I}^\varepsilon[\phi; v]\psi = (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^{2d}} v(z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(z)} g^\varepsilon_{\phi(z)}\ \langle g^\varepsilon_z, \psi\rangle\ \mathrm{d}z.$$

The Herman–Kluk propagator is included in this class by choosing $\phi = \Phi^t$ and v equal to the Herman–Kluk factor u. In order to formulate a general result on the well-definedness of such operators we need to introduce the notion of symbols and symbol classes.

Definition 6.1 (Symbol class) *Let $m \in \mathbb{R}$ and $d \in \mathbb{N}$ and $\alpha \in \mathbb{N}^d$ a multi-index. A function $a \in \mathcal{C}^\infty(\mathbb{R}^d, \mathbb{C})$ is called a symbol of class $S[m; d]$ if the expressions*

$$M_k^m(a) := \max_{|\alpha|=k} \sup_{z \in \mathbb{R}^d} \left| \left(1 + |z|^2\right)^{-m/2} \partial_z^\alpha a(z) \right| < \infty$$

for all $k \in \mathbb{N}$.

This allows me to quote the following theorem which provides the continuity of the aforementioned class of Fourier integral operators on L^2.

Theorem 6.2 (Swart and Rousse [SR09, Theorem 1]) *Let $v \in S[0; 2d]$ be a symbol, then $\mathcal{I}^\varepsilon[\phi; v]$ can be extended in a unique way to a linear bounded operator $L^2(\mathbb{R}^d; \mathbb{C}) \to L^2(\mathbb{R}^d; \mathbb{C})$ such that*

$$\|\mathcal{I}^\varepsilon[\phi; v]\|_{L^2 \to L^2} \leq 2^{-d/2} \|v\|_{L^\infty} .$$

There is a corresponding result on Weyl-quantised pseudo-differential operators, the well-known Calderón-Vaillancourt Theorem. Let me refer to Martinez [Mar02, Theorem 2.8.1] for the statement and the proof thereof.

6.2 Composition of PDOs and time-derivatives with FIOs

The proof of Theorem 9.1, my main result on the approximation properties of a discrete version of the Herman–Kluk propagator, requires information about the composition of pseudo-differential operators and time derivatives with Fourier integral operators. Let me formulate two lemmata which are special cases of two propositions by [SR09]. In order to do so, let us now consider $\Phi^t = (X^t, \Xi^t)$ to be an arbitrary family of symplectic maps on $T^*\mathbb{R}^d$, not necessarily the flow of the underlying Hamiltonian system. Its Jacobian $D\Phi^t \in S[0; 2d]$ is required to be point-wise continuously differentiable with respect to time and to fulfil

$$\sup_{t \in [0,T]} M_k^0\left(D\Phi^t\right) < \infty, \text{ and } \sup_{t \in [0,T]} M_k^0\left(D\Phi^t\right) < \infty.$$

I facilitate the presentation by using a notation similar to the one of Kay [Kay06] which I find more convenient. In the following calculations I treat $z := q + \mathrm{i}p \in \mathbb{C}^d$

as a complex variable. Note that this does not change the symplectic structure of the system. Furthermore, let me define the short hand notations

$$\partial_z := \partial_q - \mathrm{i}\partial_p \text{ and } Z^t := X^t + \mathrm{i}\partial_p \Xi^t \tag{6.1}$$

for the complex version of a derivative that already reflects the symplectic structure and the complex version of Φ^t.

Lemma 6.3 (Swart and Rousse [SR09, Prop. 2]) *Let $h \in S[m_h; 2d]$ be polynomial in p and $u \in S[m_u; 2d]$. Then the action of the Weyl quantised Hamiltonian operator $H^\varepsilon = \mathrm{op}^\varepsilon(h)$ on $\mathcal{I}^\varepsilon(\Phi^t; u)$ is given by*

$$\mathrm{H}^\varepsilon\left(\mathcal{I}^\varepsilon(\Phi^t; u)\right) = \mathcal{I}^\varepsilon(\Phi^t; w_0 + \varepsilon w_1) + \varepsilon^2\, \mathcal{I}^\varepsilon(\Phi^t; w_2^\varepsilon)$$

which is understood as an equation of operators from $\mathcal{S}(\mathbb{R}^d)$ to $\mathcal{S}(\mathbb{R}^d)$. The functions $w_0, w_1 \in S[m_u + m_h; 2d]$ are given by

$$\begin{aligned} w_0 &= u\ (h \circ \Phi)\,, \\ w_1 &= -\partial_z \cdot \Big(u \left(\partial_z Z^t\right)^{-1} (\partial_z h \circ \Phi)\Big) + u\, \tfrac{1}{2}\mathrm{tr}\left(\left(\partial_z Z^t\right)^{-1} \partial_z \left(\partial_z h \circ \Phi\right)\right). \end{aligned}$$

The function $w_2^\varepsilon \in S[(m_h, m_u + m_h); (d, 2d)]$ is provided by a linear partial differential operator L_2^ε which depends on derivatives of order 3 and 4 of h acting on u, i.e. $w_{N+1}^\varepsilon(x, \cdot) = L_2^\varepsilon u$.

Lemma 6.4 (Swart and Rousse [SR09, Prop. 3]) *Let $u \in \mathcal{C}^1(\mathbb{R}, S[m; 2d])$ be a family of time-dependent symbols with $(\frac{d}{dt}u)(t, \cdot) \in S[m; 2d]$. Then,*

$$\mathrm{i}\,\varepsilon \tfrac{d}{dt}\left(\mathcal{I}^\varepsilon(\Phi^t; u)\right) = \mathcal{I}^\varepsilon(\Phi^t; v_0 + \varepsilon v_1)$$

where $v_0, v_1 \in \mathcal{C}\left(\mathbb{R}, S[m; 2d]\right)$ are explicitly given by

$$\begin{aligned} v_0(t,z) &= u(t,z)\ \left(-\tfrac{d}{dt}S(t,z) + \left(\tfrac{d}{dt}X^t z\right) \cdot \Xi^t z\right), \\ v_1(t,z) &= \mathrm{i}\,\tfrac{d}{dt}u(t,z) - \mathrm{i}\,\partial_z \cdot \Big(u(t,z)\left(\partial_z Z^t z\right)^{-1}\left(\tfrac{d}{dt}Z^t z\right)\Big). \end{aligned}$$

In addition, I need yet another lemma, due to Hagedorn [Hag98, Lemma 2.8], that turns an asymptotic solution into an approximate one.

Lemma 6.5 *Let $U^\varepsilon_{N+1}(t)$ be an asymptotic propagator of order $N+1$, i.e.*

$$\left(\mathrm{i}\,\varepsilon\tfrac{d}{dt} - \mathcal{H}^\varepsilon\right) U^\varepsilon_{N+1}(t)\psi = R^\varepsilon_{N+1}(t)\psi, \quad U^\varepsilon_{N+1}(0) = \mathrm{Id}$$

for all $\psi \in \mathcal{S}(\mathbb{R}^d)$ and

$$\left\|R^\varepsilon_{N+1}\right\|_{\mathrm{L}^2\to\mathrm{L}^2} \le r(t)\,\varepsilon^{N+1}.$$

Then U^ε_{N+1} approximates U^ε_t in the sense that

$$\left\|\mathrm{U}^\varepsilon_t - U^\varepsilon_{N+1}\right\|_{\mathrm{L}^2\to\mathrm{L}^2} \le \varepsilon^N \int_0^t r(\tau)\,\mathrm{d}\tau.$$

Note that by passing from an asymptotic to an approximate solution we lose one power in ε.

6.3 Higher order approximations

As mentioned above, the Herman–Kluk propagator is an approximation of order one in ε to the unitary propagator. It would of course be desirable to construct higher order approximations. Swart and Rousse [SR09, Theorem 2] show that, in principle, this is possible. They give a rigorous proof by expanding the symbol of a general Fourier integral operator in powers of ε. The coefficients of the expansion are determined by ordinary differential equations involving ever higher derivatives of the potential and the classical flow.

Hochman and Kay [HK06] pursue a formal but more constructive approach. They introduce a corrected propagator that differs from the Herman–Kluk formula only by replacing the prefactor $u(t,z)$ by $u(t,z)\sum_{n=0}^N v_n(t,z)\varepsilon^n$. They find the equation of the lowest-order correction to be

$$\mathrm{i}\frac{d}{dt}v_1 = (1-V_2)\left(\frac{5}{8}b_2^2 - \frac{1}{4}b_3\right) + V_3\left(\frac{5}{12}b_2c_1 - \frac{1}{6}c_2\right) - \frac{1}{8}V_4c_1^2.$$

The functions V_k are given by the k-th derivative of the potential along the classical flow,

$$V_k(t,z) := \left(\partial_q^k V\right)(\Phi^t(z)).$$

The expressions for b_k and c_k involve derivatives of the classical flow and are given by

$$b_k(t,z) := \frac{\partial_z^k Z^t(z)}{\left(\partial_z Z^t(z)\right)^k} = \frac{\left(\partial_q - \mathrm{i}\partial_p\right)^k \left(X^t(z) + \mathrm{i}\Xi^t(z)\right)}{\left(\left(\partial_q - \mathrm{i}\partial_p\right)\left(X^t(z) + \mathrm{i}\Xi^t(z)\right)\right)^k}$$

and

$$c_k(t,z) := \frac{\partial_z^k X^t(z)}{\left(\partial_z Z^t(z)\right)^k} = \frac{\left(\partial_q - \mathrm{i}\partial_p\right)^k X^t(z)}{\left(\left(\partial_q - \mathrm{i}\partial_p\right)\left(X^t(z) + \mathrm{i}\Xi^t(z)\right)\right)^k}$$

respectively. The zeroth order term in the expansion is of course $v_0 = 1$ so that for $N = 0$ the Herman–Kluk propagator is recovered. The corresponding equations for higher-order corrections are 'more lengthy and involve sums and products of b_k, c_k, and V_k with larger k'. Nevertheless, this approach seems a promising starting point for further investigation. In fact, a similar investigation has already been conducted by Gaim and Lasser [GL14] for the method that is portrayed in Section 4.4. They derive an expression for a correction term to the symbol in order to achieve an approximation of $\mathcal{O}(\varepsilon^4)$ [GL14, Theorem 2.1].

Chapter 7

The algorithm

So far we have concentrated on the theoretic properties of the Herman–Kluk propagator. I will now present 'the' algorithm that I propose for actual computations. In order to do so let me recall the definition of the Herman–Kluk propagator (5.3), i.e.

$$\mathcal{I}_t^\varepsilon \psi_0 = (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^{2d}} u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z)} g^\varepsilon_{\Phi^t(z)} \, \langle g^\varepsilon_z, \psi_0 \rangle \, \mathrm{d}z$$

for $\psi_0 \in \mathcal{S}(\mathbb{R}^d)$ and $t \in [0,T]$. The evaluation of the wave function at just one instant $t \in \mathbb{R}$ and one point $x \in \mathbb{R}^d$ in state space requires two nested integrations. The outer one is an integral over the phase space $\mathbb{R}^{2d}$, the inner one the scalar product on the state space $\mathbb{R}^d$ and has to be evaluated for each phase space point. Both integrals have potentially highly oscillatory integrands. The time evolution of the wave function requires knowledge of the classical flow $\Phi^t(z)$, the classical action $S(t,z)$, and the Herman–Kluk factor $u(t,z)$ for all times $t \in \mathbb{R}$ as well as every phase space point $z \in \mathbb{R}^{2d}$. In this chapter I describe discretisation strategies for phase space and time. It also contains a schematic description of the resulting algorithm. Chapters 10 and 11 present the parallelisation and efficient implementation of the algorithm.

7.1 Phase space discretisation

Grid based quadrature methods are not a suitable choice for this problem since they would require N^{3d} points if we used N quadrature nodes in each direction. For high dimensional integrands Monte Carlo and quasi-Monte Carlo techniques have become the methods of choice. Their shortcoming of having a low order of accuracy is of little consequence here since the total error is already dominated by the asymptotic error of order ε, see Theorem 5.4. In order to apply these methods let me specify the class of initial wave functions to choose from by means of the following assumption.

Assumption 7.1 *Let $\psi_0 \in \mathcal{S}(\mathbb{R}^d)$ such that for all $z \in \mathbb{R}^{2d}$ there is a multiplicative decomposition*

$$(2\pi\varepsilon)^{-d} \langle g_z^\varepsilon, \psi_0 \rangle =: r_0^\varepsilon(z) \cdot \mu_0^\varepsilon(z), \tag{7.1}$$

where $\mu_0^\varepsilon \in \mathcal{S}(\mathbb{R}^{2d})$ is a probability distribution on $\mathbb{R}^{2d}$ and the complex-valued function $r_0^\varepsilon \in C^\infty(\mathbb{R}^{2d}) \cap L^1(\,\mathrm{d}\mu_0^\varepsilon)$ is growing at most polynomially for $z \to \infty$.

This sounds like a huge limitation at first but it is not. The assumption still allows a variety of initial wave functions, especially the ones that are commonly used in semiclassical calculations, including Hermite functions as well as Hagedorn's semiclassical wave packets. Lasser and Troppmann [LT14] provide analytic formulae for their respective FBI transforms. In addition, linear combinations of functions fulfilling the assumption are also admissible since the Herman–Kluk propagator is linear with respect to the initial wave function. In particular, one may approximate an arbitrary wave function with a linear combination of Hermite functions and use this as initial data. By means of the decomposition formula (7.1) the Herman–Kluk propagator can be reinterpreted as a weighted integration over phase space,

$$\mathcal{I}_t^\varepsilon \psi_0 = \int_{\mathbb{R}^{2d}} r_0^\varepsilon(z)\, u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z)}\, g^\varepsilon_{\Phi^t(z)}\, \mathrm{d}\mu_0^\varepsilon(z). \tag{7.2}$$

Note that r_0 and μ_0^ε depend on the initial wave function ψ_0 only. We will use Monte Carlo or quasi-Monte Carlo quadrature to approximate (7.2). In both cases the approximate wave function is given by

$$\psi_M(t) = \frac{1}{M} \sum_{m=1}^{M} r_0(z_m)\, u(t,z_m)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z_m)}\, g^\varepsilon_{\Phi^t(z_m)} \tag{7.3}$$

where $z_1, \dots, z_M \in \mathbb{R}^{2d}$ are sampled from μ_0^ε. I will present rigorous error estimates in Chapter 8. For now, let me give the following example.

Example 7.2 *A common choice as initial wave function in semiclassical calculations is a Gaussian wave packet* (5.1) *centred at a phase space point* $z_0 = (q_0, p_0) \in \mathbb{R}^{2d}$*, i.e.*

$$\psi_0(x) = g^\varepsilon_{z_0}(x) = (\pi\varepsilon)^{-d/4} \exp\left(-\tfrac{1}{2\varepsilon}\ |x - q_0|^2 + \tfrac{\mathrm{i}}{\varepsilon}\ p_0 \cdot (x - q_0)\right).$$

In this case, the explicit formula for the scalar product that occurs in the FBI transform can be calculated by completing the square in the exponent.

$$\begin{aligned}
\langle g^\varepsilon_z, \psi_0\rangle &= \int_{\mathbb{R}^d} \overline{g^\varepsilon_z(x)}\, g^\varepsilon_{z_0}(x)\,\mathrm{d}x \\
&= (\pi\varepsilon)^{-d/2} \int_{\mathbb{R}^d} \exp\left(-\tfrac{1}{2\varepsilon}\left(|x - q_0|^2 + |x - q|^2\right) + \tfrac{\mathrm{i}}{\varepsilon}\left(p_0 \cdot (x - q_0) - p \cdot (x - q)\right)\right) \mathrm{d}x \\
&= \exp\left(-\tfrac{1}{4\varepsilon}\ |z - z_0|^2 + \tfrac{\mathrm{i}}{2\varepsilon}\ (p + p_0) \cdot (q - q_0)\right).
\end{aligned}$$

Thus we can rewrite the Herman–Kluk propagator as

$$\mathcal{I}^\varepsilon_t \psi_0 = (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^{2d}} u(t, z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z) + \frac{\mathrm{i}}{2\varepsilon}(p+p_0)\cdot(q-q_0)}\, g^\varepsilon_{\Phi^t(z)}\, \mathrm{e}^{-\frac{1}{4\varepsilon}|z-z_0|^2}\, \mathrm{d}z.$$

This translates to (7.2) *with the probability density*

$$\mu_0^\varepsilon(z) = (4\pi\varepsilon)^{-d} \mathrm{e}^{-\frac{1}{4\varepsilon}|z-z_0|^2}$$

on $\mathbb{R}^{2d}$ *and*

$$r_0(z) = 2^d \mathrm{e}^{\frac{\mathrm{i}}{2\varepsilon}(p+p_0)\cdot(q-q_0)}.$$

For the corresponding explicit formulae for Hermite and Hagedorn functions let me refer once more to Lasser and Troppmann [LT14].

7.2 Calculation of expectation values

As mentioned before, one of the great advantages of the Herman–Kluk propagator is the ability to compute the full wave function including its phase. On the other hand, expectation values

$$\mathbb{E}_{\mathrm{qm}}[\mathcal{A}](t) = \langle \psi(t), \mathcal{A}\psi(t)\rangle \tag{7.4}$$

for a given self-adjoint operator $\mathcal{A}$ are often the quantities of interest in physics and chemistry. These observables include position or momentum as well as kinetic or potential energy. Formula (7.4) suggests that in order to calculate expectation values we would have to perform yet another numerical quadrature. The computational effort would thus grow quadratically in the number of quadrature points. Fortunately, there is a way to compute expectation values without actually evaluating the wave function using the same idea that led to the interpretation of the Herman–Kluk propagator as a weighted integration over phase space (7.2), i.e.

$$\mathcal{I}_t^\varepsilon \psi_0 = \int\limits_{\mathbb{R}^{2d}} r_0^\varepsilon(z)\, u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z)}\, g^\varepsilon_{\Phi^t(z)}\, \mathrm{d}\mu_0^\varepsilon(z).$$

Let μ_0^ε and r_0^ε be defined according to Assumption 7.1 and let us abbreviate the integrand by

$$f^\varepsilon(t,z) := r_0^\varepsilon(z)\, u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z)}. \tag{7.5}$$

If we rewrite the expectation value of an observable with respect to the Herman–Kluk wave function, we get

$$\begin{aligned}
\langle \mathcal{I}_t^\varepsilon \psi_0, \mathcal{A}\, \mathcal{I}_t^\varepsilon \psi_0 \rangle &= \int\limits_{\mathbb{R}^d} \overline{\mathcal{I}_t^\varepsilon \psi_0(x)}\, (\mathcal{A}\, \mathcal{I}_t^\varepsilon \psi_0(x))\, \mathrm{d}x \\
&= \int\limits_{\mathbb{R}^{2d}} \int\limits_{\mathbb{R}^{2d}} \overline{f(t,w)}\, f(t,z)\, \left\langle g^\varepsilon_{\Phi^t(w)}, \mathcal{A}\, g^\varepsilon_{\Phi^t(z)} \right\rangle \mathrm{d}\mu_0^\varepsilon(w)\, \mathrm{d}\mu_0^\varepsilon(z) \\
&= \int\limits_{\mathbb{R}^{4d}} \overline{f(t,w)}\, f(t,z)\, \left\langle g^\varepsilon_{\Phi^t(w)}, \mathcal{A}\, g^\varepsilon_{\Phi^t(z)} \right\rangle \mathrm{d}\,(\mu_0^\varepsilon \otimes \mu_0^\varepsilon)\,(w,z).
\end{aligned}$$

Thus we may interpret the expectation value as a weighted integral on $\mathbb{R}^{4d}$ with respect to the product measure $\mu_0^\varepsilon \otimes \mu_0^\varepsilon$ instead of two separate integrations on $\mathbb{R}^{2d}$. Consider a sequence of (Monte Carlo or quasi-Monte Carlo) quadrature points

$$(w_1, z_1), \ldots, (w_M, z_M) \in \mathbb{R}^{4d}$$

that are sampled from $\mu_0^\varepsilon \otimes \mu_0^\varepsilon$. Then

$$A_M(t) := \frac{1}{M} \sum_{m=1}^{M} \overline{f(t,w)}\, f(t,z)\, \left\langle g^\varepsilon_{\Phi^t(w^m)}, \mathcal{A}\, g^\varepsilon_{\Phi^t(z^m)} \right\rangle \tag{7.6}$$

is an approximation to (7.4). Note that the computational effort grows linearly in the number of quadrature points albeit on a space of twice the dimension. In addition, the inner product of two Gaussian wave packets with distinct centres

$$\left\langle g^\varepsilon_{\Phi^t(w)}, \mathcal{A}\, g^\varepsilon_{\Phi^t(z)} \right\rangle$$

is easier to handle than the one with arbitrary wave functions. There are even analytic expressions for several observables including position, momentum, and kinetic energy operators, as well as all polynomial potentials and the torsional potential. Some examples are presented in Appendix A.

7.3 Time discretisation

In addition to the classical flow $\Phi^t(z)$, the Herman–Kluk propagated wave function at time $t \in \mathbb{R}$ requires the knowledge of the classical action

$$S(t,z) = \int_0^t \left(\tfrac{\mathrm{d}}{\mathrm{d}\tau} X^\tau(z) \cdot \Xi^\tau(z) - h(\Phi^\tau(z)) \right) \mathrm{d}\tau \tag{2.8}$$

as well as the Herman–Kluk factor

$$u(t,z) = \sqrt{2^{-d} \det \left(\partial_q X^t(z) - \mathrm{i}\partial_p X^t(z) + \mathrm{i}\partial_q \Xi^t(z) + \partial_p \Xi^t(z) \right)} \tag{5.4}$$

at each phase space point $z \in \mathbb{R}^{2d}$. Let me remind you that in order to compute $u(t,z)$ we need to solve the variational equation

$$\dot{W}(t) = \mathcal{J}\, \nabla^2 h(\Phi^t)\, W(t), \quad W(0) = \mathrm{I} \tag{5.6}$$

that corresponds to the Hamiltonian system (2.3). Again, $W(t) = D_z \Phi^t$ is the derivative of the flow with respect to the initial values and $\nabla^2 h$ is the Hessian of the Hamiltonian function. This equation inherits the symplectic structure of (2.3), similar to the one for the matrices in Hagrdorn's wave packets in Chapter 4.2. It is therefore favourable to solve (5.6) with the same symplectic method as (2.3). The following idea lets us treat the action $S(t,z)$ in a similar manner. It is just the application of the fundamental theorem of calculus on the action integral (2.8).

Proposition 7.3 *Consider a Hamiltonian function $h(q,p) = T(p) + V(q)$ that separates in a kinetic part $T(p)$ only depending on the variable $p \in \mathbb{R}^d$ and a potential $V(q)$ only depending on $q \in \mathbb{R}^d$. Then the classical action may be seen as solution to the initial value problem*

$$\dot{S}(t,z) = T(\Xi^t(z)) - V(X^t(z)), \quad S(0,z) = 0 \tag{7.7}$$

for all $z = (q,p) \in \mathbb{R}^{2d}$.

In order to solve (7.7) by the same symplectic method as (2.3) and (5.6), let us artificially spilt it into two equations, defining S_q and S_p by

$$\begin{pmatrix} \dot{S}_q(t,z) \\ \dot{S}_p(t,z) \end{pmatrix} = \begin{pmatrix} T(\Xi^t(z)) \\ -V(X^t(z)) \end{pmatrix}, \quad \begin{pmatrix} S_q(0,z) \\ S_p(0,z) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{7.8}$$

Furthermore, we need a way to convert the matrix-valued variational equation into a vector-valued form.

Definition 7.4 (Vectorisation of a matrix) *For any matrix $A \in \mathbb{R}^{m\times n}$ let*

$$\vec{\text{vec}}\,(A) := (A_{1,1}, \dots A_{m,1}, A_{1,2}, \dots A_{m,2}, \dots\dots A_{1,n}, \dots A_{m,n})^T \in \mathbb{R}^{mn}$$

denote the linear transformation that converts the matrix into a column vector.

This allows us to the define the following system of ordinary differential equations.

Proposition 7.5 *Consider a separable Hamiltonian function $h(q,p) = T(p) + V(q)$. For any $z = (q,p) \in \mathbb{R}^{2d}$ let us define the vectors $\mathcal{Z}^t(z) \in \mathbb{R}^{2d+4d^2+2}$ and $F\left(\mathcal{Z}^t(z)\right) \in \mathbb{R}^{2d+4d^2+2}$ by*

$$\mathcal{Z}^t(z) := \begin{pmatrix} X^t(z) \\ \Xi^t(z) \\ \vec{\text{vec}}\left(\partial_q X^t(z)\right) \\ \vec{\text{vec}}\left(\partial_p X^t(z)\right) \\ \vec{\text{vec}}\left(\partial_q \Xi^t(z)\right) \\ \vec{\text{vec}}\left(\partial_p \Xi^t(z)\right) \\ S_q(t,z) \\ S_p(t,z) \end{pmatrix} \text{ and } F\left(\mathcal{Z}^t(z)\right) := \begin{pmatrix} \partial_p T(\Xi^t(z)) \\ -\partial_q V(X^t(z)) \\ \vec{\text{vec}}\left(\partial_p^2 h(\Phi^t(z)) \cdot \partial_q \Xi^t(z)\right) \\ \vec{\text{vec}}\left(\partial_p^2 h(\Phi^t(z)) \cdot \partial_p \Xi^t(z)\right) \\ \vec{\text{vec}}\left(-\partial_q^2 h(\Phi^t(z)) \cdot \partial_q X^t(z)\right) \\ \vec{\text{vec}}\left(-\partial_q^2 h(\Phi^t(z)) \cdot \partial_p X^t(z)\right) \\ T(\Xi^t(z)) \\ -V(X^t(z)) \end{pmatrix} \tag{7.9}$$

respectively. Then the classical flow of (2.3)*, and the solutions to the corresponding variational equation* (5.6)*, as well as the classical action* (7.8) *are simultaneously found by solving*

$$\dot{\mathcal{Z}}^t(z) = F\left(\mathcal{Z}^t(z)\right) \tag{7.10}$$

with initial values

$$\begin{pmatrix} X^0(z) \\ \Xi^0(z) \\ \vec{\mathrm{vec}}\left(\partial_q X^0(z)\right) \\ \vec{\mathrm{vec}}\left(\partial_p X^0(z)\right) \\ \vec{\mathrm{vec}}\left(\partial_q \Xi^0(z)\right) \\ \vec{\mathrm{vec}}\left(\partial_p \Xi^0(z)\right) \\ S_q(0,z) \\ S_p(0,z) \end{pmatrix} = \begin{pmatrix} q \\ p \\ \vec{\mathrm{vec}}\,(\mathrm{I}) \\ 0 \\ 0 \\ \vec{\mathrm{vec}}\,(\mathrm{I}) \\ 0 \\ 0 \end{pmatrix}. \tag{7.11}$$

This is the foundation for all numerical calculations in Part III of this thesis.

Symplectic integrator

As I point out in Chapter 2.2, the dynamics of a Hamiltonian system are inextricably linked to the notion of symplecticity. In order to preserve the symplectic structure of the classical system

$$\dot{z} = \mathcal{J}\,\nabla h(z) \tag{2.3}$$

it is crucial to use a suitable numerical integrator. I cannot stress this fact enough. Any non-symplectic method inevitably changes the nature of dynamics, even if it is a good approximation in terms of ordinary differential equations. Mere energy conservation is not sufficient, the method has to be symplectic in the sense of Definition 2.8.

My method of choice is based on the Størmer–Verlet scheme which is introduced by Størmer [Stø07] for numerical computations of the trajectories of charged particles in order to describe the aurora borealis. Verlet [Ver67] proposes this method for computations of thermodynamical properties of molecules. It has since become a widely used integration scheme in molecular dynamics. For a separable system of the form $h(q,p) = T(p) + V(q)$, the Størmer–Verlet method admits an explicit one-step formulation $\Psi^t : z_n \mapsto z_{n+1}$ which is useful for actual computations. A single step of the method is given by

$$\begin{aligned} p_{n+1/2} &= p_n - \tfrac{\tau}{2}\,\nabla V(q_n) \\ q_{n+1} &= q_n + \tau\,\nabla T(p_{n+1/2}) \\ p_{n+1} &= p_{n+1/2} - \tfrac{\tau}{2}\,\nabla V(q_{n+1}). \end{aligned}$$

Hairer, Lubich, and Wanner [HLW06] investigate this scheme in great detail and show that it is a symplectic and symmetric method of order two. The order of the scheme can be further increased by applying a composition strategy. Kahan and Li [KL97] provide composition constants which are custom-built for the task.

Complex square root

Another issue arises from the fact that we have to evaluate a complex square root in order to compute the Herman–Kluk factor

$$u(t,z) = \sqrt{2^{-d} \det \left(\partial_q X^t(z) - \mathrm{i} \partial_p X^t(z) + \mathrm{i} \partial_q \Xi^t(z) + \partial_p \Xi^t(z) \right)}.$$

for $t \in [0,T]$ and $z \in \mathbb{R}^{2d}$. The function u is continuous with respect to t and therefore we need find a way to reflect this in our calculations. In order to do so, let us introduce the following notion. Let $\gamma : [0,T] \to \mathbb{C} \setminus \{0\}$ be a path in $\mathbb{C}$ not passing through the origin. A continuous map $\theta : [0,T] \to \mathbb{R}$ such that

$$\gamma(t) = |\gamma(t)| e^{\mathrm{i}\theta(t)}$$

is called continuous choice of argument along γ. Let me refer to Stewart and Tall [ST83, Theorem 7.1] for a proof of the existence of such a map.

Proposition 7.6 *For any $\gamma : [0,T] \to \mathbb{C} \setminus \{0\}$ there exists a continuous choice of argument. Moreover, any further continuous choice of argument from this by a constant integer multiple of 2π.*

Since u is uniformly bounded away from 0, this makes it possible to define a continuous complex square root by

$$\sqrt{u(t,z)} := \sqrt{|u(t,z)|} \, \exp\left(\frac{\mathrm{i}}{2} \theta(t) \right)$$

where θ is a continuous choice of argument along u. Thus, in addition to the system (7.10) of ordinary differential equations which evolves the matrices $\partial_q X^t$, $\partial_p X^t$, $\partial_q \Xi^t$, and $\partial_p \Xi^t$ in time, the time-stepping algorithm also has to calculate the absolute value and a continuous argument for

$$\det \left(\partial_q X^t(z) - \mathrm{i} \partial_p X^t(z) + \mathrm{i} \partial_q \Xi^t(z) + \partial_p \Xi^t(z) \right).$$

This allows us to evaluate $u(t,z)$ whenever we need it. It also eliminates additional error sources that may arise from numerically checking the continuity of the square root.

7.4 Schematic description of the algorithm

Our goal is to calculate either a wave function, more precisely the solution to the Schrödinger equation, or expectation values of operators along this solution. The two tasks require different sampling points but may use the same time-stepping algorithm.

1. a) Sample $z_1, \ldots, z_M \in \mathbb{R}^{2d}$ from μ_0^ε;

   or

   b) Sample $(w_1, z_1), \ldots, (w_M, z_M) \in \mathbb{R}^{4d}$ from $\mu_0^\varepsilon \otimes \mu_0^\varepsilon$;

2. Allocate an array $Z \in \mathbb{R}^{(2d+4d^2+2)\times M}$ containing the sampling points and the corresponding initial values (7.11);

3. Evolve the columns of Z according to (7.10) by means of the high-order, symplectic, and symmetric numerical integration method described above to some desired time t;

4. Compute the Herman–Kluk factor with a continuous phase;

5. a) Calculate the approximate wave function by (7.3), i.e.

$$\psi_M(t) = \frac{1}{M} \sum_{m=1}^{M} r_0(z_m)\, u(t, z_m)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t, z_m)}\, g^\varepsilon_{\Phi^t(z_m)}; \tag{7.12}$$

   or

   b) Calculate expectation values by (7.6), i.e.

$$A_M(t) := \frac{1}{M} \sum_{m=1}^{M} \overline{f(t, w)}\, f(t, z) \left\langle g^\varepsilon_{\Phi^t(w^m)}, \mathcal{A}\, g^\varepsilon_{\Phi^t(z^m)} \right\rangle;$$

Because of the parallel nature of Step 3, this algorithm can be implemented in a highly efficient manner, which is the subject matter of Part III.

Chapter 8

Phase Space Discretisation

This chapter contains a rigorous error analysis of the phase space discretisation that I introduce in Chapter 7.1. Let me recollect the crucial ideas. We want to discretise the phase space integral of the Herman–Kluk Propagator (5.3). Monte Carlo and quasi-Monte Carlo methods are well suited for the quadrature over phase space since we want to solve this for as many degrees of freedom as possible. In order to apply these quadrature methods, only specific initial wave functions are admissible. This permits to interpret the Herman–Kluk Propagator as a weighted integral over phase space,

$$\mathcal{I}_t^\varepsilon \psi_0 = \int_{\mathbb{R}^{2d}} r_0^\varepsilon(z)\, u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z)}\, g_{\Phi^t(z)}^\varepsilon \,\mathrm{d}\mu_0^\varepsilon(z), \tag{7.2}$$

where μ_0^ε and r_0^ε are defined in accordance with Assumption 7.1. In order to facilitate notation let us denote the integrand

$$f_t^\varepsilon(z) := r_0^\varepsilon(z)\, u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon} S(t,z)} g_{\Phi^t(z)}^\varepsilon \in \mathcal{S}(\mathbb{R}^d) \tag{8.1}$$

for all $z \in \mathbb{R}^{2d}$ and $t \in [0,T]$.

8.1 Error Estimate for Monte Carlo quadrature

I want to introduce the concept of Monte Carlo quadrature methods directly applied to the problem at hand. The presentation is based on the book by Krommer and

Ueberhuber [KU98, Chapter 6.4.] which I recommend as a general introduction to the topic of numerical integration. In essence, Monte Carlo methods are based on the computation of a deterministic object by means of stochastic quantities. In our case, let us interpret the integrand f_t^ε as a random variable with values in the Hilbert space $\mathrm{L}^2(\mathbb{R}^d)$ which is distributed according to the probability measure μ_0^ε. Then the phase space integral is its expected value, i.e.

$$\mathcal{I}_t^\varepsilon \psi_0 = \int_{\mathbb{R}^{2d}} f_t^\varepsilon(z)\,\mathrm{d}\mu_0^\varepsilon(z) = \mathbb{E}[f_t^\varepsilon]. \tag{8.2}$$

A fundamental technique in statistics is to estimate an expected value using a sample mean. By taking M independent samples $z_1, \ldots, z_M \in \mathbb{R}^{2d}$ of the probability distribution μ_0^ε we define the Monte Carlo estimator

$$\psi_M^\varepsilon(t) := \frac{1}{M} \sum_{m=1}^{M} f_t^\varepsilon(z_m) \in \mathrm{L}^2(\mathbb{R}^d). \tag{8.3}$$

which is simply a linear combination of classically evolved Gaussian wave packets. From a stochastic point of view we regard the samples $z_1, \ldots, z_M \in \mathbb{R}^{2d}$ as realisations of a sequence of independent, identically distributed random variables. By the strong law of large numbers, the Monte Carlo estimator converges almost surely to the expected value. However, this does not provide us with any information about the error for a specific sample of size M.

A measure for the accuracy of the Monte Carlo method is the expected mean squared error. It is defined as the expected value of the square of the error between the estimator and what is estimated, i.e.

$$\mathbb{E}\left[\left\|\psi_M^\varepsilon(t) - \mathcal{I}_t^\varepsilon \psi_0\right\|^2\right]. \tag{8.4}$$

The following proposition provides an estimate for the mean squared error which shows an accuracy of $\mathcal{O}(M^{-1/2})$ with respect to the number of sample points. This is the typical convergence rate of the error in integration formulas based on Monte Carlo methods and, most importantly, it does not depend on the dimension d.

Proposition 8.1 *Let the initial wave function $\psi_0 \in \mathcal{S}(\mathbb{R}^d)$ satisfy Assumption 7.1 and let $\psi_M^\varepsilon(t)$ be the Monte Carlo estimator defined in* (8.3)*. Then, the mean squared error is given by*

$$\mathbb{E}\left[\left\|\psi_M^\varepsilon(t) - \mathcal{I}_t^\varepsilon \psi_0\right\|^2\right] = \frac{\mathbb{V}(f_\varepsilon^t)}{M},$$

where $\mathbb{V}[f_t^\varepsilon]$ *is the variance of* f_t^ε*. It satisfies*

$$\mathbb{V}[f_t^\varepsilon] \leq 3 \int_{\mathbb{R}^{2d}} |u(t,z) r_0^\varepsilon(z)|^2 \, \mathrm{d}\mu_0^\varepsilon(z) + \|\mathcal{I}_t^\varepsilon \psi_0\|^2$$

for all $t \in [0,T]$ *and* $\varepsilon > 0$*.*

Proof We observe that

$$\mathbb{E}[\psi_M^\varepsilon(t)] = \frac{1}{M} \sum_{M=1}^{M} \mathbb{E}[f_t^\varepsilon] = \mathcal{I}_t^\varepsilon \psi_0. \tag{8.5}$$

Since the samples are independent and identically distributed, we get

$$\mathbb{E}\Big[\|\psi_M^\varepsilon(t) - \mathcal{I}_t^\varepsilon \psi_0\|^2\Big] = \mathbb{V}[\psi_M^\varepsilon(t)] = \frac{1}{M^2} \sum_{m=1}^{M} \mathbb{V}[f_t^\varepsilon] = \frac{\mathbb{V}[f_t^\varepsilon]}{M},$$

Moreover, by definition of $\mathbb{V}$,

$$\begin{aligned}\mathbb{V}[f_t^\varepsilon] &= \mathbb{E}\Big[\|f_t^\varepsilon - \mathcal{I}_t^\varepsilon \psi_0\|^2\Big] = \int_{\mathbb{R}^{2d}} \|f_t^\varepsilon(z) - \mathcal{I}_t^\varepsilon \psi_0\|^2 \, \mathrm{d}\mu_0^\varepsilon(z) \\ &= \int_{\mathbb{R}^{2d}} \|f_t^\varepsilon(z)\|^2 \, \mathrm{d}\mu_0^\varepsilon(z) - 2 \int_{\mathbb{R}^{2d}} \Re\langle f_t^\varepsilon(z), \mathcal{I}_t^\varepsilon \psi_0 \rangle \, \mathrm{d}\mu_0(z) + \|\mathcal{I}_t^\varepsilon \psi_0\|^2 .\end{aligned}$$

By writing

$$\int_{\mathbb{R}^{2d}} \langle f_t^\varepsilon(z), \mathcal{I}_t^\varepsilon \psi_0 \rangle \, \mathrm{d}\mu_0^\varepsilon(z) = \int_{\mathbb{R}^{4d}} \langle f_t^\varepsilon(z), f_t^\varepsilon(w) \rangle \, \mathrm{d}(\mu_0^\varepsilon \otimes \mu_0^\varepsilon)(w,z)$$

and estimating

$$|\langle f_t^\varepsilon(z), f_t^\varepsilon(w) \rangle| \leq \|f_t^\varepsilon(z)\| \, \|f_t^\varepsilon(w)\| \leq \frac{1}{2}(\|f_t^\varepsilon(z)\|^2 + \|f_t^\varepsilon(w)\|^2)$$

we find that

$$\left| \int_{\mathbb{R}^{2d}} \langle f_t^\varepsilon(z), \mathcal{I}_t^\varepsilon \psi_0 \rangle \, \mathrm{d}\mu_0^\varepsilon(z) \right| \leq \int_{\mathbb{R}^{2d}} \|f_t^\varepsilon(z)\|^2 \, \mathrm{d}\mu_0^\varepsilon(z).$$

Since $\|f_t^\varepsilon(z)\| = |u(t,z) r_0^\varepsilon(z)|$, this concludes the estimate

$$\mathbb{V}[f_t^\varepsilon] \leq 3 \int_{\mathbb{R}^{2d}} |u(t,z) r_0^\varepsilon(z)|^2 \, \mathrm{d}\mu_0^\varepsilon(z) + \|\mathcal{I}_t^\varepsilon \psi_0\|^2 . \qquad \square$$

Note that the final estimate of Proposition 8.1,

$$\mathbb{V}[f_t^\varepsilon] \leq 3 \int_{\mathbb{R}^{2d}} |u(t,z) r_0^\varepsilon(z)|^2 \,\mathrm{d}\mu_0^\varepsilon(z) + \|\mathcal{I}_t^\varepsilon \psi_0\|^2 ,$$

is dominated by its first summand, since Theorem 5.4 provides

$$\|\mathcal{I}_t^\varepsilon \psi_0\| = \|\mathrm{U}_t^\varepsilon \psi_0\| + \mathcal{O}(\varepsilon) = \|\psi_0\| + \mathcal{O}(\varepsilon).$$

In case of our previous example we may even calculate the initial variance $\mathbb{V}[f_0^\varepsilon]$ analytically and observe ε-independence as well as convergence to one as $d \to \infty$.

Example 8.2 *For the initial mean squared error of the sampling of a single Gaussian wave packet $\psi_0 = g_{z_0}^\varepsilon$, there is an analytic expression for the variance. We have*

$$f_0^\varepsilon(z) = r_0^\varepsilon(z) g_z^\varepsilon = 2^d \,\mathrm{e}^{\frac{\mathrm{i}}{2\varepsilon}(p+p_0)\cdot(q-q_0)} g_z^\varepsilon \quad \textit{and} \quad \mathbb{E}[f_0^\varepsilon] = g_{z_0}^\varepsilon$$

so that

$$\mathbb{V}[f_0^\varepsilon] = \int_{\mathbb{R}^{2d}} \int_{\mathbb{R}^d} \left| r_0^\varepsilon(z) g_z^\varepsilon(x) - g_{z_0}^\varepsilon(x) \right|^2 \,\mathrm{d}x \,\mathrm{d}\mu_0^\varepsilon(z) = 1 - 4^{-d}.$$

Figure 8.1 shows the sampling error for the initial wave function with respect to the number of Monte Carlo quadrature points M.

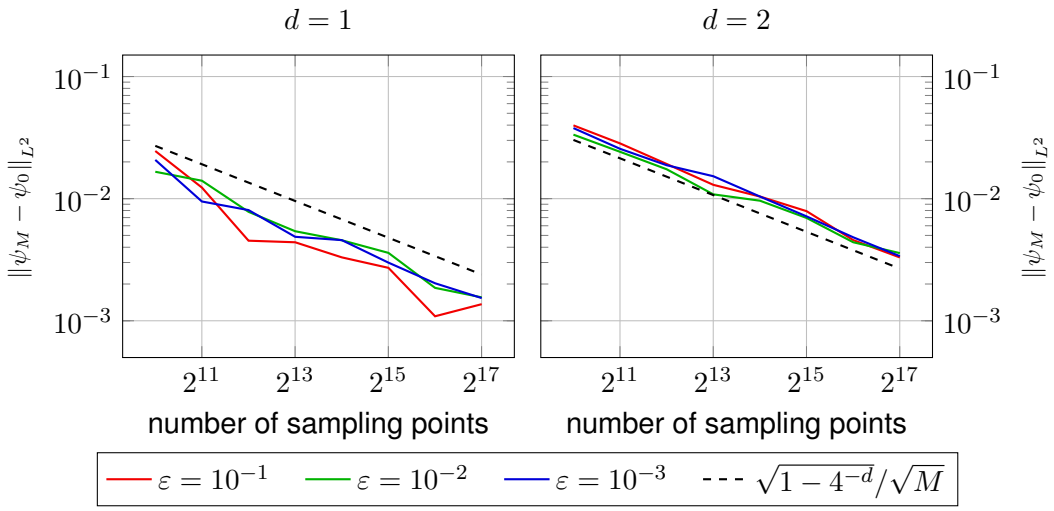

Figure 8.1: Initial sampling error with respect to the number of Monte Carlo quadrature points M.

Each wave function is produced by averaging over 10 independent samples. The two pictures show the error for one and two space dimensions respectively. Note that the error shows no dependence on the value of ε.

8.2 Error Estimate for quasi-Monte Carlo quadrature

Quasi-Monte Carlo quadrature rules are number-theoretic integration formulas that define well-chosen deterministic quadrature points in order to cover the integration domain more uniformly. Such sequences of points are called low discrepancy sequences. I will not go into details on that point and take for granted that such sequences exist, since I only use quasi-Monte Carlo integration as a tool. Finding such sequences by means of number-theoretic considerations is a vast field of study itself. Niederreiter [Nie92] provides an excellent introduction to the topic in general. I will only introduce the important notions, such as the following.

Definition 8.3 (Discrepancy function) *Let μ be a probability measure on $\mathbb{R}^d$. For any $x_1, \dots, x_M \in \mathbb{R}^d$ and all $x \in \mathbb{R}^d$ let us define the discrepancy function of μ by*

$$\mathcal{D}_M(x_1, \dots, x_M; x) := \frac{1}{M} \sum_{m=1}^{M} \chi_{]-\infty,x]}(x_m) - \mu(]-\infty, x]).$$

The discrepancy function quantifies the deviation of the empirical distribution for the d-dimensional rectangular interval $]-\infty, y]$ which is defined by

$$]-\infty, y] :=]-\infty, y_1] \times \cdots \times]-\infty, y_d] \subset \mathbb{R}^d$$

for every $y = (y_1, \dots, y_d)^T \in \mathbb{R}^d$. If the measure μ is the product of one-dimensional probability measures and the inverses of the one-dimensional cumulative distribution functions are accessible, then the well-established low discrepancy sets for the uniform measure on the unit cube $[0,1]^d$ allow the construction of points $x_1, \dots, x_M \in \mathbb{R}^d$ with

$$\sup_{x \in \mathbb{R}^d} |\mathcal{D}_M(x_1, \dots, x_M; x)| = \mathcal{O}\left(\frac{(\log M)^{d-1}}{M}\right),$$

see [AD15, Theorem 4]. The following lemma illuminates why the discrepancy function is a crucial quantity for this type of quadrature.

Lemma 8.4 *Let $f \in \mathcal{S}(\mathbb{R}^d)$ and μ a probability measure on $\mathbb{R}^d$ such that $f \in L^1(\,\mathrm{d}\mu)$. Then, for all $x_1, \ldots, x_M \in \mathbb{R}^d$,*

$$\begin{aligned}&\frac{1}{M}\sum_{m=1}^{M} f(x_m) - \int_{\mathbb{R}^d} f(x)\,\mathrm{d}\mu(x)\\ &= (-1)^d \int_{\mathbb{R}^d} \partial^{1:d} f(y) \left(\frac{1}{M}\sum_{m=1}^{M} \chi_{]-\infty,y]}(x_m) - \mu(]-\infty,y]) \right) \mathrm{d}y.\end{aligned}$$

The proof adjusts the argumentation of the proof of the Koksma–Hlawka inequality by Aistleitner and Dick [AD15, Theorem 1] from the integration of functions of bounded variation on the unit cube to the integration of Schwartz functions on unbounded domains.

Proof For any $x \in \mathbb{R}^d$ we have

$$\begin{aligned} f(x) &= -\int_{x_1}^{\infty} \partial_1 f(y_1, x_2, \ldots, x_n)\,\mathrm{d}y_1\\ &= (-1)^d \int_{x_1}^{\infty} \cdots \int_{x_d}^{\infty} \partial^{1:d} f(y_1, \ldots, y_d)\,\mathrm{d}y_d \cdots \mathrm{d}y_1\\ &= (-1)^d \int_{[x,\infty[} \partial^{1:d} f(y)\,\mathrm{d}y. \end{aligned}$$

This implies for the arithmetic mean

$$\begin{aligned} \frac{1}{M}\sum_{m=1}^{M} f(x_m) &= \frac{(-1)^d}{M} \sum_{m=1}^{M} \int_{\mathbb{R}^d} \chi_{[x_m,\infty[}(y)\ \partial^{1:d} f(y)\,\mathrm{d}y\\ &= (-1)^d \int_{\mathbb{R}^d} \partial^{1:d} f(y) \frac{1}{M}\sum_{m=1}^{M} \chi_{]-\infty,y]}(x_m)\,\mathrm{d}y \end{aligned}$$

and for the integral

$$\int_{\mathbb{R}^d} f(x)\,\mathrm{d}\mu(x) = (-1)^d \int_{\mathbb{R}^d} \partial^{1:d} f(y)\mu(]-\infty,y])\,\mathrm{d}y,$$

where the last equation also uses Fubini's theorem. □

Let us now apply this result on the phase space discretisation of the Herman–Kluk propagator. Let us recall Equations (8.1) and (8.3), i.e.

$$f_t^\varepsilon(z) = r_0^\varepsilon(z)\, u(t,z)\, \mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}S(t,z)} g^\varepsilon_{\Phi^t(z)} \in \mathcal{S}(\mathbb{R}^d)$$

and

$$\psi_M^\varepsilon(t) := \frac{1}{M}\sum_{m=1}^{M} f_t^\varepsilon(z_m) \in \mathrm{L}^2(\mathbb{R}^d), \tag{8.6}$$

respectively, for $z \in \mathbb{R}^{2d}$ and $t \in [0,T]$. Then we obtain the following weak convergence result.

Proposition 8.5 *Let $\psi_0 \in \mathcal{S}(\mathbb{R}^d)$ and $r_0^\varepsilon, \mu_0^\varepsilon$ be defined according to Assumption 7.1 and consider $\psi_M^\varepsilon(t)$ as defined in* (8.3)*. Furthermore, let $\mathcal{D}_M^\varepsilon(z_1,\ldots,z_M;z)$ be the discrepancy function of μ_0^ε. Then,*

$$\psi_M^\varepsilon(t) - \mathcal{I}_t^\varepsilon\psi_0 = \int_{\mathbb{R}^{2d}} \partial_z^{1:2d} f_t^\varepsilon(z)\, \mathcal{D}_M^\varepsilon(z_1,\ldots,z_M;z)\,\mathrm{d}z. \tag{8.7}$$

In particular,

$$\lim_{M\to\infty} \sup_{z\in\mathbb{R}^{2d}} |\mathcal{D}_M^\varepsilon(z_1,\ldots,z_M;z)| = 0$$

implies

$$\lim_{M\to\infty} \langle \phi, \psi_M^\varepsilon(t) - \mathcal{I}_t^\varepsilon\psi_0\rangle = 0 \tag{8.8}$$

for all test functions $\phi \in \mathcal{S}(\mathbb{R}^d)$.

Proof For all $\phi \in \mathcal{S}(\mathbb{R}^d)$ the mapping $z \mapsto \langle \phi, f_t^\varepsilon(z)\rangle$ defines a Schwartz function on $\mathbb{R}^{2d}$. Therefore we apply Lemma 8.4 and obtain

$$\langle \phi, \psi_M^\varepsilon(t) - \mathcal{I}_t^\varepsilon\psi_0\rangle = \int_{\mathbb{R}^{2d}} \partial_z^{1:2d}\langle \phi, f_t^\varepsilon(z)\rangle\, \mathcal{D}_M^\varepsilon(z_1,\ldots,z_M;z)\,\mathrm{d}z,$$

which means

$$\psi_M^\varepsilon(t) - \mathcal{I}_t^\varepsilon\psi_0 = \int_{\mathbb{R}^{2d}} \partial_z^{1:2d} f_t^\varepsilon(z)\, \mathcal{D}_M^\varepsilon(z_1,\ldots,z_M;z)\,\mathrm{d}z.$$

Moreover,

$$|\langle \phi, \psi_M(t) - \mathcal{I}_t^\varepsilon\psi_0\rangle| \leq \sup_{z\in\mathbb{R}^{2d}} |\mathcal{D}_M^\varepsilon(z_1,\ldots,z_M;z)| \int_{\mathbb{R}^{2d}} \left|\partial_z^{1:2d}\langle \phi, f_t^\varepsilon(z)\rangle\right| \mathrm{d}z,$$

so that

$$\lim_{M\to\infty} \sup_{z\in\mathbb{R}^{2d}} |\mathcal{D}_M^\varepsilon(z_1,\dots,z_M;z)| = 0$$

implies (8.8). □

Even though we have proven weak convergence, we notice that the mixed derivative of the integrand $f_t^\varepsilon(z)$ depends unfavourably on various parameters as the following example illustrates.

Example 8.6 *Let us examine the mixed derivative of the initial integrand $f_0^\varepsilon(z)$ for a Gaussian wave packet $\psi_0 = g_0^\varepsilon$ centred in the origin $z_0 = 0$. A straightforward calculation shows that*

$$\partial^{1:2d} f_0^\varepsilon(z) = f_0^\varepsilon(z) \prod_{j=1}^{d} \frac{\mathrm{i}}{2\varepsilon^2} \left((x - q_j - \mathrm{i}p_j)(x - \tfrac{1}{2}q_j)\right)$$

and

$$\left\|\partial^{1:2d} f_0^\varepsilon(z)\right\|^2 = \varepsilon^{-4d} \prod_{j=1}^{d} \left(\tfrac{1}{8}\varepsilon(q_j^2 + 6\varepsilon) + \tfrac{1}{4}p_j^2(q_j^2 + 2\varepsilon)\right).$$

Hence, the norm of the mixed derivative of f_0^ε has a factor ε^{-2d} in front of a polynomial in z.

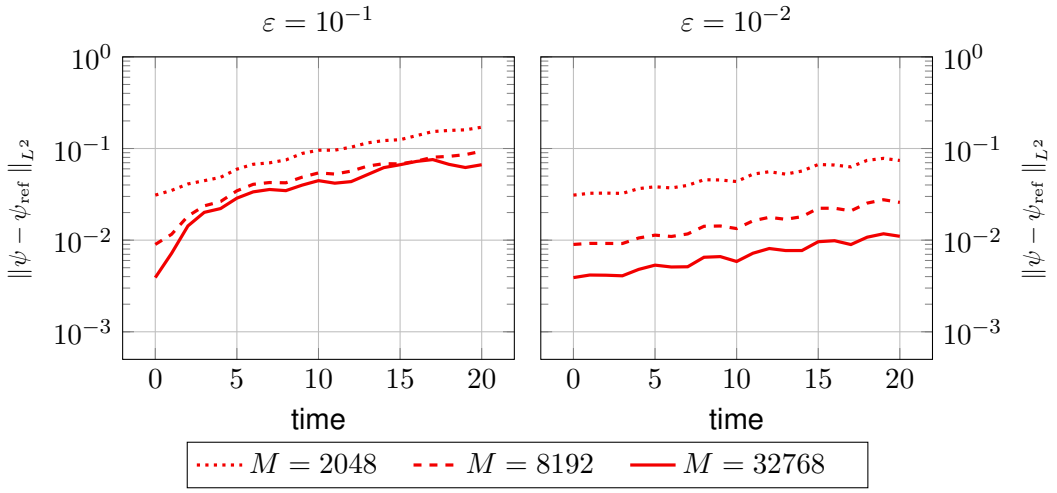

Figure 8.2: Propagation of the error between the Herman-Kluk and the reference solution for the torsional potential in the L^2 norm.

Our numerical experiments confirm that the smaller ε and the larger the dimension d, the more quadrature points are required. However, it seems that beneficial cancellations in the key equation (8.7) *allow for a much smaller M than expected. Figure 8.2 shows the error between the reference solution and the Herman–Kluk wave function in the L^2 norm. The total error is a combination of the asymptotic error of order ϵ and the quadrature error which depends on the number M of quasi-Monte Carlo points.*

Chapter 9

Time Discretisation

As a result of Theorem 5.4 we are aware that the Herman–Kluk propagator $\mathcal{I}_t^\varepsilon$ approximates the unitary time evolution U_t^ε of the semiclassical Schrödinger equation with a Weyl quantised Hamiltonian operator $H^\varepsilon = \mathrm{op}^\varepsilon(h)$. More precisely, for any $T > 0$ there exists a $c > 0$ such that

$$\sup_{t\in[0,T]} \|\mathrm{U}_t^\varepsilon - \mathcal{I}_t^\varepsilon\|_{\mathrm{L}^2(\mathbb{R}^d)\to\mathrm{L}^2(\mathbb{R}^d)} \leq c\,\varepsilon.$$

In this chapter I prove a similar theorem for the time-discretisation of the Herman–Kluk propagator. Consider the numerical solution to the system (7.10) of ordinary differential equations by means of the method that is described in Section 7.3. In the spirit of backward error analysis of symplectic methods I interpret these solutions as the exact solutions to a modified system of equations, which I explain in Chapter 9.1. For now, let me formulate the main result of this chapter.

Theorem 9.1 *Let $\tilde{\Phi}^t$, $\tilde{S}$ and $\tilde{u}$ be calculated from the numerical solutions to* (7.10) *by a symmetric and symplectic method of order $\gamma \in \mathbb{N}$ and let*

$$(\tilde{\mathcal{I}}_t^\varepsilon\psi)(x) := (2\pi\varepsilon)^{-d} \int_{\mathbb{R}^{2d}} \tilde{u}(t,z)\,\mathrm{e}^{\frac{\mathrm{i}}{\varepsilon}\tilde{S}(t,z)}\,\langle g_z^\varepsilon,\psi\rangle\, g^\varepsilon_{\tilde{\Phi}^t(z)}\,\mathrm{d}z. \tag{9.1}$$

be the corresponding Herman–Kluk propagator. Then $\tilde{\mathcal{I}}_t^\varepsilon$ is an approximation to the exact propagator U_t^ε, meaning that for any $t > 0$ there exist $c_0, c_1, c_2 > 0$ such that for all $\varepsilon > 0$

$$\left\|\tilde{\mathcal{I}}_t^\varepsilon - \mathrm{U}_t^\varepsilon\right\| \leq \frac{\tau^\gamma}{\varepsilon}\,c_0 + \tau^\gamma\,c_1 + c_2\,\varepsilon + \mathcal{O}(\tau^{\gamma+2})$$

where $\tau > 0$ is the time step size of the symplectic method. The constants c_0, c_1, and c_2 depend on the final time T but neither on τ nor on ε.

This theorem allows us to achieve an overall error of order $\mathcal{O}(\varepsilon)$ for a sufficiently small time step $\tau > 0$. The proof consist of three main steps. First of all it requires a backward error analysis of symplectic methods, which I introduce by reviewing the main ideas of Hairer, Lubich, and Wanner [HLW06] on this topic. As a second step, I make use of the results on the composition of pseudo-differential operators and time derivatives with Fourier integral operators that are stated in Chapter 6.2 to find that $\tilde{\mathcal{I}}_t^\varepsilon$ is indeed an asymptotic propagator. Finally, I apply Lemma 6.5 to turn this asymptotic propagator into an approximate one.

9.1 Backward error analysis for symplectic methods

The fundamental idea of backward error analysis is to interpret the errors in the output of an algorithm as additional errors in the input. This dates back to Wilkinson [Wil60] for problems in numerical linear algebra and was eventually also applied to ordinary differential equation. Hairer, Lubich, and Wanner [HLW06, Chapter IX] provide a systematic study of the concept.

Consider a numerical method Ψ^t to solve an ordinary differential equation $\dot{z} = F(z)$. As mentioned, the idea of backward error analysis is to look for a modified equation

$$\dot{\tilde{z}} = \tilde{F}(\tilde{z}) \tag{9.2}$$

such that the exact flow of this modified equation is equal to the numerical solution of the original by Ψ^t. Let us now consider a Hamiltonian system

$$\dot{z} = \mathcal{J}\,\nabla h(z), \quad z(0) = z_0$$

and a one-step method described by $z_{n+1} = \Psi^\tau(z_n)$ with step size $\tau > 0$. Furthermore, let Ψ^τ allow for a Taylor series expansion $\Psi^\tau = \sum_{k\in\mathbb{N}} \tau^k D_k(y)$ with smooth D_k, $k \in \mathbb{N}$. Then we have the following result which is a combination of [HLW06, Ch. IX, Theorem 1.2] and [HLW06, Ch. IX, Theorem 3.1].

Lemma 9.2 *Suppose that the method Ψ^τ is symplectic and of order $\gamma \in \mathbb{N}$, i.e.*

$$\Psi^\tau(y) = \Phi^\tau(y) + \tau^{\gamma+1} D_{\gamma+1}(y) + \mathcal{O}(\tau^{\gamma+2})$$

where Φ^τ denotes the exact flow of $\dot{z} = \mathcal{J}\nabla h(z)$. Then the modified equation (9.2) *is again a Hamiltonian system. More precisely, there exist smooth functions $h_k : \mathbb{R}^{2d} \longrightarrow \mathbb{R}$ for $k \in \{2, 3, \dots\}$, such that*

$$\dot{\tilde{z}} = \mathcal{J}\nabla h(\tilde{z}) + \tau^\gamma \mathcal{J}\nabla h_{\gamma+1}(\tilde{z}) + \tau^{\gamma+1}\mathcal{J}\nabla h_{\gamma+2}(\tilde{z}) + \dots$$

with $\tilde{z}(0) = z_0$ and $\mathcal{J}\nabla h_{\gamma+1}(z) = D_{\gamma+1}(z)$.

When using a symmetric method such as the one proposed in Chapter 7.3 there are additional advantages. Not only is the order of the method even but also the following statement holds, cf. [HLW06, Ch. IX, Theorem 2.2].

Lemma 9.3 *The coefficient functions of the modified equation of a symmetric method Ψ^τ satisfy $F_k(y) = 0$ whenever k is even, so that (9.2) has an expansion in even powers of τ.*

These two lemmata allow us to treat the numerical solutions as the exact flow of the modified Hamiltonian system given by $\tilde{h}$. Let us hence assume that Ψ^τ is a symmetric, symplectic method of order $\gamma \in \mathbb{N}$. Then

$$\Psi^\tau(y) = \Phi^\tau(y) + \mathcal{O}(\tau^{\gamma+2})$$

and h and $\tilde{h}$ are related by

$$\tilde{h}(z) = h(z) + \tau^\gamma h_{\gamma+1}(z) + \mathcal{O}(\tau^{\gamma+2}). \tag{9.3}$$

9.2 Error estimate for the time discretisation

We are now prepared for the proof of Theorem 9.1. Let $\tilde{\Phi}^t$, $\tilde{S}$ and $\tilde{u}$ be calculated from the numerical solutions to (7.10) by a symmetric, symplectic method of order $\gamma \in \mathbb{N}$ and let the corresponding Herman–Kluk propagator $\tilde{\mathcal{I}}_t^\varepsilon$ be defined by 9.1. In addition to the error of the symplectic method we need information about the systematic error that is made in the construction of asymptotic approximations which is provided by Lemmata 6.3 and 6.4. The following proof uses the complex notation

$$z := q + \mathrm{i}p, \partial_z := \partial_q - \mathrm{i}\partial_p, \text{ and } Z^t := X^t + \mathrm{i}\partial_p \Xi^t \tag{9.4}$$

introduced in Chapter 6.2.

Proof The action of the Hamiltonian on the Herman–Kluk propagator is provided by Lemma 6.3, i.e.

$$\mathrm{H}^{\varepsilon}\left(\mathcal{I}^{\varepsilon}(\tilde{\Phi}^t;\tilde{u})\right) = \mathcal{I}^{\varepsilon}(\tilde{\Phi}^t;\tilde{w}_0+\varepsilon\tilde{w}_1) + \mathcal{I}^{\varepsilon}(\tilde{\Phi}^t;\tilde{w}_2^{\varepsilon}) \tag{9.5}$$

where

$$\begin{aligned}
\tilde{w}_0 &= \tilde{u}\,\left(h\circ\tilde{\Phi}^t\right),\\
\tilde{w}_1 &= -\partial_z\cdot\left(\tilde{u}\left(\partial_z\tilde{Z}^t\right)^{-1}\left(\partial_z h\circ\tilde{\Phi}^t\right)\right) + \tfrac{1}{2}\,\tilde{u}\,\mathrm{tr}\left(\left(\partial_z\tilde{Z}^t\right)^{-1}\partial_z\left(\partial_z h\circ\tilde{\Phi}^t\right)\right).
\end{aligned}$$

Lemma 6.4 provides the formula

$$\mathrm{i}\,\varepsilon\tfrac{d}{dt}\left(\mathcal{I}^{\varepsilon}(\Phi^t;u)\right) = \mathcal{I}^{\varepsilon}(\Phi^t;v_0+\varepsilon v_1) \tag{9.6}$$

for the application of a time derivative on the propagator where

$$\begin{aligned}
\tilde{v}_0 &= \tilde{u}\,\left(-\tfrac{d}{dt}\tilde{S} + \left(\tfrac{d}{dt}\tilde{X}^t\right)\cdot\tilde{\Xi}^t\right),\\
\tilde{v}_1 &= \mathrm{i}\,\tfrac{d}{dt}\tilde{u} - \mathrm{i}\,\partial_z\cdot\left(\left(\tilde{u}\,\partial_z\tilde{Z}^t\right)^{-1}\left(\tfrac{d}{dt}\tilde{Z}^t\right)\right).
\end{aligned}$$

Combining (9.5) and (9.6) we get

$$\left(\mathrm{i}\,\varepsilon\tfrac{d}{dt} - \mathrm{H}^{\varepsilon}\right)\mathcal{I}^{\varepsilon}(\tilde{\Phi}^t;\tilde{u}) = \mathcal{I}^{\varepsilon}(\tilde{\Phi}^t;(\tilde{v}_0-\tilde{w}_0)+\varepsilon(\tilde{v}_1-\tilde{w}_1)) + \mathcal{R}^{\varepsilon}\,\varepsilon^2.$$

This means that we have to examine the expressions $\tilde{v}_0-\tilde{w}_0$ and $\tilde{v}_1-\tilde{w}_1$. By definition of $\tilde{S}$ we have

$$\tilde{h}\circ\tilde{\Phi}^t = -\tfrac{d}{dt}\tilde{S} + \left(\tfrac{d}{dt}\tilde{X}^t\right)\cdot\tilde{\Xi}^t$$

and therefore

$$\tilde{v}_0-\tilde{w}_0 = \tilde{u}\left(\tilde{h}\circ\tilde{\Phi}^t - h\circ\tilde{\Phi}^t\right).$$

The expression for equation for $\tilde{v}_1-\tilde{w}_1$ is given by

$$\begin{aligned}
\tilde{v}_1-\tilde{w}_1 = \mathrm{i}\,\tfrac{d}{dt}\tilde{u} - \mathrm{i}\,\partial_z\cdot&\left(\left(\tilde{u}\,\partial_z\tilde{Z}^t\right)^{-1}\left(\tfrac{d}{dt}\tilde{Z}^t\right)\right)\\
&+\partial_z\cdot\left(\tilde{u}\left(\partial_z\tilde{Z}^t\right)^{-1}\left(\partial_z h\circ\tilde{\Phi}^t\right)\right)\\
&-\tfrac{1}{2}\,\tilde{u}\,\mathrm{tr}\left(\left(\partial_z\tilde{Z}^t\right)^{-1}\partial_z\left(\partial_z h\circ\tilde{\Phi}^t\right)\right).
\end{aligned}$$

By definition, $\tilde{u}$ fulfils

$$\tfrac{d}{dt}\tilde{u} = \tfrac{1}{2}\,\tilde{u}\,\mathrm{tr}\left(\left(\partial_z \tilde{Z}^t\right)^{-1} \tfrac{d}{dt}\left(\partial_z \tilde{Z}^t\right)\right)$$

and we get

$$\begin{aligned}\tilde{v}_1 - \tilde{w}_1 = \tfrac{1}{2}\,\tilde{u}\,\mathrm{tr}\left(\left(\partial_z \tilde{Z}^t\right)^{-1}\left(\mathrm{i}\,\tfrac{d}{dt}\left(\partial_z \tilde{Z}^t\right) - \partial_z\left(\partial_z h \circ \tilde{\Phi}^t\right)\right)\right)\\ + \partial_z \cdot \left(\tilde{u}\left(\partial_z \tilde{Z}^t\right)^{-1}\left(\left(\partial_z h \circ \tilde{\Phi}^t\right) - \mathrm{i}\,\tfrac{d}{dt}\tilde{Z}^t\right)\right).\end{aligned}$$

Let us now represent the Hamiltonian equations of motion in the terminology of (9.4), i.e.

$$\mathrm{i}\,\tfrac{d}{dt}\tilde{Z} = \mathrm{i}\,\tfrac{d}{dt}\tilde{X}^t - \tfrac{d}{dt}\Xi^t = \mathrm{i}\,\partial_p \tilde{h} \circ \tilde{\Phi}^t + \partial_q \tilde{h} \circ \tilde{\Phi}^t = \partial_z \tilde{h} \circ \tilde{\Phi}^t.$$

Together with the definition of $\tilde{\Phi}^t$ as the exact flow of $\tilde{h}$ this gives us

$$\begin{aligned}\tilde{v}_1 - \tilde{w}_1 = \tfrac{1}{2}\,\tilde{u}\,\mathrm{tr}\left(\left(\partial_z \tilde{Z}^t\right)^{-1}\partial_z\left(\partial_z \tilde{h} \circ \tilde{\Phi}^t - \partial_z h \circ \tilde{\Phi}^t\right)\right)\\ + \partial_z \cdot \left(\tilde{u}\left(\partial_z \tilde{Z}^t\right)^{-1}\left(\partial_z h \circ \tilde{\Phi}^t - \partial_z \tilde{h} \circ \tilde{\Phi}^t\right)\right).\end{aligned}$$

Finally, we can use the backward error analysis which was introduced in the previous chapter. Equation 9.3 gives us

$$\left(\tilde{h} - h\right) \circ \tilde{\Phi}^t = \tau^\gamma \left(h_{\gamma+1} \circ \tilde{\Phi}^t\right) + \mathcal{O}(\tau^{\gamma+2}).$$

Introducing

$\tilde{u}_0 := \tilde{u}\left(h_{\gamma+1} \circ \tilde{\Phi}^t\right)$ and

$$\tilde{u}_1 := \tfrac{1}{2}\,\tilde{u}\,\mathrm{tr}\left(\left(\partial_z \tilde{Z}^t\right)^{-1}\partial_z\left(\partial_z h_{\gamma+1} \circ \tilde{\Phi}^t\right)\right) + \partial_z \cdot \left(\tilde{u}\left(\partial_z \tilde{Z}^t\right)^{-1}\left(\partial_z h_{\gamma+1} \circ \tilde{\Phi}^t\right)\right)$$

$$\left(\frac{1}{2}\,\tilde{u}\,\mathrm{tr}\left(\tilde{Z}^{-1}\partial_z\left(\partial_z h_{\gamma+1} \circ \tilde{\Phi}^t\right)\right) + \mathrm{div}_z\left(\left(\partial_z h_{\gamma+1} \circ \tilde{\Phi}^t\right)^* \tilde{Z}^{-1}\tilde{u}\right)\right)$$

we obtain

$$\begin{aligned}\tilde{v}_0 - \tilde{w}_0 = \tilde{u}_0\,\tau^\gamma + \mathcal{O}(\tau^{\gamma+2}),\\ \tilde{v}_1 - \tilde{w}_1 = \tilde{u}_0\,\tau^\gamma + \mathcal{O}(\tau^{\gamma+2})\end{aligned}$$

and

$$\left(\mathrm{i}\varepsilon\tfrac{d}{dt} - \mathrm{H}^\varepsilon\right)\mathcal{I}^\varepsilon(\tilde{\Phi}^t;\tilde{u}) = \tau^\gamma\mathcal{I}^\varepsilon(\tilde{\Phi}^t;\tilde{u}_0) + \tau^\gamma\varepsilon\mathcal{I}^\varepsilon(\tilde{\Phi}^t;\tilde{u}_1) + \mathcal{I}^\varepsilon(\Phi^t;w_2^\varepsilon)\,\varepsilon^2 + \mathcal{O}(\tau^{\gamma+2}).$$

Theorem 6.2 guarantees that $\mathcal{I}^\varepsilon(\tilde{\Phi}^t;\tilde{u}_0)$, $\mathcal{I}^\varepsilon(\tilde{\Phi}^t;\tilde{u}_1)$ and $\mathcal{I}^\varepsilon(\Phi^t;w_2^\varepsilon)$ are bounded operators. This allows us to apply Lemma 6.5 and we finally arrive at

$$\left\|\tilde{\mathcal{I}}_t^\varepsilon - \mathrm{U}_t^\varepsilon\right\| \leq \frac{\tau^\gamma}{\varepsilon}\, c_0 + \tau^\gamma\, c_1 + c_2\, \varepsilon + \mathcal{O}(\tau^{\gamma+2})$$

where

$$c_0 = \int_0^t \|\mathcal{I}^\varepsilon(\tilde{\Phi}^\tau;\tilde{u}_0)\|\,\mathrm{d}\tau, \quad c_1 = \int_0^t \|\mathcal{I}^\varepsilon(\tilde{\Phi}^\tau;\tilde{u}_1)\|\,\mathrm{d}\tau, \text{ and } c_2 = \int_0^t \|\mathcal{I}^\varepsilon(\Phi^t;w_2^\varepsilon)\|. \quad \square$$

Remark 9.4 A proof of Theorem 5.4 is included in the proof I just gave for Theorem 9.1 by considering the case $\tilde{\Phi}^t = \Phi^t$.

Numerical validation

Now I want to illustrate Theorem 9.1 by means of two examples. The first one is the quantum mechanical harmonic oscillator which is one of the few systems for which an analytic solution is known explicitly. The second one is a system with a torsional potential in two dimensions.

Example 9.5 (The harmonic oscillator) *As a proof of concept let us restrict ourselves to one spacial dimension where a grid based approach is still feasible. This allows us to demonstrate the robustness of my algorithm even for large times, in this case $t \in [0, 100]$. Let us consider the harmonic oscillator potential $V(x) = x^2/2$ and a Gaussian wave packet*

$$\psi_0(x) = g^\varepsilon_{(q_0,p_0)} = (\pi\varepsilon)^{-1/4} \exp\left(-\frac{1}{2\varepsilon}(x-q_0)^2 + \frac{\mathrm{i}}{\varepsilon}\, p_0(x-q_0)\right).$$

as initial wave function. Let $q(t)$, $p(t)$ and $S(t)$ be the position, momentum and action of the classical harmonic oscillator respectively, i.e.

$$\begin{aligned} q(t) &= \quad q_0\cos(t) + p_0\sin(t) \\ p(t) &= -q_0\sin(t) + p_0\cos(t) \\ S(t) &= \tfrac{1}{2}\sin(t)\left(\left(p_0^2 - q_0^2\right)\cos(t) - 2p_0q_0\sin(t)\right). \end{aligned}$$

Then the analytic solution to the quantum mechanical problem is then given by

$$\psi_a(t,x) = (\pi\varepsilon)^{-1/4} \exp\left(\frac{\mathrm{i}}{\varepsilon}\, S(t) - \frac{\mathrm{i}}{2}\, t - \frac{1}{2\varepsilon}(x-q(t))^2 + \frac{\mathrm{i}}{\varepsilon}\, p(t)(x-q(t))\right).$$

For the numerical calculations I use an equidistant grid in classical phase space $\mathbb{R}^2$ *with grid size* $\delta_q = \delta_p = 0.1$ *and another equidistant grid in the wave function's position space with grid size* $\delta_x = 2^{-8}\pi$ *on the interval* $[-\pi, \pi]$*. The time is discretised in equal steps of size* $\tau = 0.05$ *from* $t = 0$ *to the final time* $t = 100$*. The initial position and momentum are* $q_0 = 1$ *and* $p_0 = 0$*. Figure 9.1 shows the error between the Herman–Kluk and the analytic solution in the* L^2*-norm for different values of the semiclassical parameter* $\varepsilon \in \{10^{-1}, 10^{-2}, 10^{-3}\}$*. It underlines that the Herman–Kluk propagator is exact for quadratic potentials and that my algorithm preserves this feature over long times. Furthermore, the overall error is proportional to* τ^γ/ε *as predicted by Theorem 9.1.*

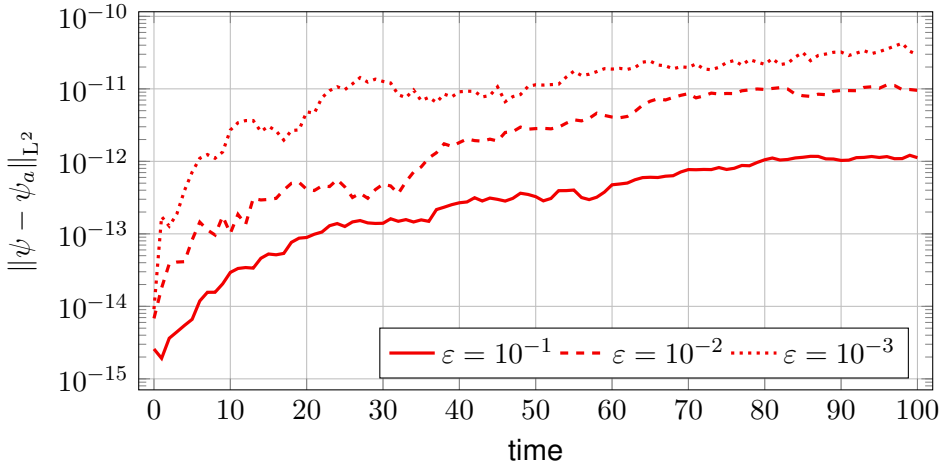

Figure 9.1: Evolution of the deviation between the Herman-Kluk and the analytic solution for the harmonic oscillator potential and different values of ε. The error stays close to the initial sampling error even over long times.

Example 9.6 (The torsional potential in two dimensions)
Intramolecular rotations are often modelled by a torsional potential of the form

$$V(x) = \sum_{k=1}^{d} (1 - \cos(x_k)).$$

In two dimensions the Herman–Kluk wave function can still be evaluated on an equidistant grid and thus be compared to a reference solution calculated by means of a split-step Fourier method. Again, I take a Gaussian wave packet

$$\psi_0(x) = (\pi\varepsilon)^{-1/2} \exp\left(-\frac{1}{2\varepsilon}|x - q_0|^2 + \frac{\mathrm{i}}{\varepsilon}\, p_0 \cdot (x - q_0)\right) \tag{9.7}$$

as initial wave function with $q_0 = (1,0)^T$ *and* $p_0 = (0,0)^T$*. Figure 9.2 shows the behaviour of the error*

$$\|\psi(T) - \psi_{ref}(T)\|_{L^2}$$

between the Herman–Kluk solution and the reference solution at the final time $T = 20$ *with respect to the length of one time step. The number of quasi-Monte Carlo points in phase space remains fixed. This is done for two different values of the semiclassical parameter, namely* $\varepsilon = 10^{-1}$ *and* $\varepsilon = 10^{-2}$ *to show that the overall error is dominated by* ε *if the length of a time step becomes sufficiently small. The calculations are performed by the original Størmer-Verlet scheme as time integrator, i.e.* $\gamma = 2$*, as well as a composition method of order* $\gamma = 4$*. The error behaves exactly as predicted by Theorem 9.1.*

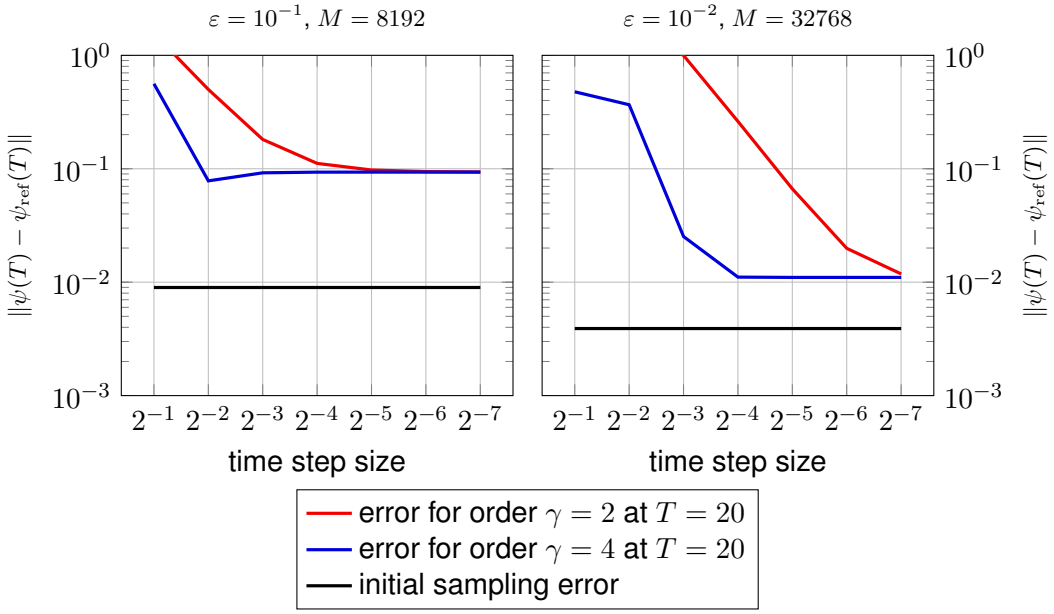

Figure 9.2: Dependence of the error between the Herman-Kluk and the reference solution on the length of one time step.

Part III

High Performance Computing

Chapter

10

Parallel programming

Parallel programming and the design of efficient parallel programs is a well established concept in scientific computing. Nevertheless, it used to be a niche within the entire field until in recent years, the hardware development advanced toward multicore architectures. The reasons for this is of fundamental nature. Placing an ever larger number of transistors on a single chip leads to thermal problems as well as restrictions on the clock speed. And thus, even desktop computers nowadays are small parallel systems of their own. Software developers need to restructure existing non-parallel code in order to take advantage of the new computing capabilities. In particular, one cannot expect to automatically improve the efficiency of software products by just increasing the number of computing cores.

The most important step in parallel programming is the design of a parallel algorithm for a given problem. First, the problem is composed into several tasks which can be computed simultaneously. Those tasks are then assigned to processes or *threads* which are delegated to physical computation units for execution.

Here, we can see one of the main advantages of the algorithm presented in Chapter 7. Let us recall the basic structure. The calculation of the Herman–Kluk propagator requires a large number of sample points and, for each point, a system of ordinary differential equations has to be solved numerically. The points and the solutions to the equations are completely independent of each other. There is no need to devise elaborate parallelisation strategies, the algorithm is inherently parallel.

Chapter 11 covers the details of the implementation and gives a performance analysis. The present chapter serves as an introduction to the concepts of parallel computing. It is based on the book by Rauber and Rünger [RR10] which I want to recommend as an introductory text on the topic of parallel programming.

10.1 OpenMP

OpenMP is a common standard for parallel programming. It was designed in 1997 and is maintained by the OpenMP Architecture Review Board[1]. Its mission is 'to standardise directive-based multi-language high-level parallelism that is performant, productive and portable'. The current version 4.0 was published in 2013 [Ope13]. OpenMP is not an independent programming language. Rather, it provides a collection of compiler directives, library routines, and environmental variables. The compiler directives are used to extend C and C++ as well as other languages with single program, multiple data constructs. This gives the programmer a simple and flexible interface for the development of parallel applications on platforms ranging from embedded systems and accelerator devices to multicore and shared-memory systems. In OpenMP, parallelism is controlled by compiler directives. For C and C++, OpenMP directives are specified with the `#pragma` directive of the C and C++ standards. In the implementation that I presented in the next chapter, only a single directive is required in front of the main loop over all the sampling points.

```
#pragma omp parallel for
    for (T m = 0; m < M; m++)
    {
    (...)
    }
```

Note that the parallel and the sequential code only differ in a single line. Moreover, the sequential code is still accessible by setting the value of the environment variable `OMP_NUM_THREADS`. This is done by means of the command

```
export OMP_NUM_THREADS = n
```

where n is the number of threads one wants to use. For $n = 1$, for example, the program is executed sequentially. In order to translate the source code into

[1] http://openmp.org/

multithreaded code, a single additional compiles option `-fopenmp` is set to compile the program. I present a numerical experiment that reflects the performance of the parallel Herman–Kluk method with respect to the number of OpenMP threads in Chapter 11.3.

10.2 Message-Passing Interface

Let us consider an abstract model of a parallel computer with no global memory access, data exchange between the different processors must be performed explicitly by passing messages. To guarantee portability of programs, no assumptions on the topology of the interconnection network is made. Instead, it is assumed that each processor can send a message to any other processor. In order to make program design easier, it is usually assumed that each of the processor executes the same program (single program, multiple data). The processor executing a message-passing program can exchange local data by calling communication functions. These are provided by a communication library, the most popular is called Message-Passing Interface (MPI). The MPI standard defines the syntax and semantics of library routines for standard communication patterns. It is defined and maintained by the MPI Forum[2]. The current standard, MPI-3.1, was approved by the MPI Forum in 2015 [MPI15].

In the case of my algorithm only a few MPI commands are required since the processes do not need to share any data except for the initial setup of the phase space points, Code Listing 10.1 displays some examples.

The MPI function `MPI_Init()` is called before any other MPI function to initialise the MPI runtime system. The second and third call return the rank of the calling process in the variable `rank` and the total number of processes in variable `size` respectively. Other functions the algorithm requires are `MPI_Send()` in order to distribute the sampling points and `MPI_Reduce()` to accumulate the final sum over all sampling points. By using `#ifdef` directives whenever inserting an MPI function, the identical source code is used for MPI as well as OpenMP and sequential execution. Similar to OpenMP, one activates MPI by means of a compiler flag, in this case `-D_MPI_`. This combines the simplicity of not having to maintain multiple versions of the code with the capability of using various parallel programming techniques.

[2] http://www.mpi-forum.org/

```
#ifdef _MPI_
#include <mpi.h>
#define omp_get_wtime MPI_Wtime
#endif

(...)

#ifdef _MPI_
int rank, size;
MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &size);
#endif

(...)
```

Code Listing 10.1: Some basic MPI commands are included in C++ code.

10.3 Graphics processing units

Another trend in parallel computing is the use of graphics processing units (GPUs). The GPU architecture provides hundreds of specialised processing cores that can perform computations in parallel. It employs a single instruction, multiple thread model as a parallelisation paradigm. The vectorisation is performed by the compiler, using essentially the sequential code with a few GPU specific extensions. In order for this model to work efficiently, the data layout of the simulation code needs adaptation. Often, this is the most important aspect of optimising code for graphics processors. Apart from that, the computational kernels for the GPU accelerated simulation code are compiled from the same source code as the CPU version. The main modification necessary for the GPU kernel is the replacement of the outer `for` loop, that iterates over the sample points, with the computation of a unique thread id, cf. Code Listing 10.2. The GPU implementation of the algorithm uses CUDA[3] which is a parallel computing platform provided by NVIDIA. It provides 'a comprehensive development environment for C and C++ developers building GPU-accelerated applications'. Note that, again, the use of CUDA is managed by an `#ifdef` directive in order to eliminate the necessity of maintaining multiple versions of the same code.

[3] https://developer.nvidia.com/cuda-toolkit

```
#ifdef CUDA
    T m = threadIdx.x + blockIdx.x * blockDim.x;
#else
#pragma omp parallel for
    for (T m = 0; m < M; m++)
#endif
    {
    (...)
    }
```

Code Listing 10.2: The `for` loop, that iterates over the sample points, is replaced by a unique thread id for the GPU kernel.

10.4 Vectorisation

Modern CPUs include vector extensions, such as Streaming SIMD Extensions (SSE) and Advanced Vector Extensions (AVX and AVX-512). They extend the instruction set of the processors to increase parallelism and throughput in floating point calculations. Basically, the processor simultaneously performs the same operation on a vector of multiple (up to 8) floating point numbers instead of single one. Although these vector operations offer high theoretical speed-up, it is tedious task to maintain separate versions of the simulation code for each extension.

One of the results from my cooperation with Manfred Liebmann [LS15] is the development of an explicit vectorisation library in C++. It hides the underlying complexities and makes the vector processing capabilities more easily accessible. One of our goals is to keep the syntax for the instantiation of the vector data types as simple as possible and close to the built-in `double` data type. As a consequence we avoid a heavy template approach in contrast to e.g. Kretz and Lindenstruth [KL12]. Our vectorisation library defines classes for the vector data types `double2`, `double4`, and `double8`. Some of the member functions that encapsulate the vector extension intrinsics in the C++ class for the `double2` data type are given in Code Listing 10.3 as an example. These two functions define how to convert the intrinsic `__m128d` and `double` datatypes to `double2` respectively.

```
class double2
{
    __m128d v;
    double2()
    {
    }
    double2(__m128d &rhs)
    {
        v = rhs;
    }
    double2(double rhs)
    {
        v = _mm_set1_pd(rhs);
    }
    (...)
};
```

Code Listing 10.3: Some member functions that encapsulate the vector extension intrinsics for the `double2` data type.

Beyond that, our library also provides all elementary arithmetic operators $+, -, *, /$ by overloading the standard built-in operators as well as common functions, such as `pow`, `exp`, `sqrt`, `sin`, `cos`. The summation and multiplication of `double2` as well as the square root are just redefinitions of the intrinsic vector functions, cf. Code Listings 10.4 and 10.5. Functions such as the inverse tangent in Code Listing 10.6 are vectorised explicitly.

```
    double2& operator+=(const double2& rhs)
    {
        v = _mm_add_pd(v, rhs.v);
        return *this;
    }
    double2& operator*=(const double2& rhs)
    {
        v = _mm_mul_pd(v, rhs.v);
        return *this;
    }
```

Code Listing 10.4: The summation and multiplication for data of type `double2`.

```
double2& sqrt()
{
    v = _mm_sqrt_pd(v);
    return *this;
}
```

Code Listing 10.5: The square root function for data of type `double2`.

```
double2& atan2(const double2& rhs)
{
    double *a = (double*)&v;
    double *b = (double*)&rhs;
    double a0 = std::atan2(a[0], b[0]);
    double a1 = std::atan2(a[1], b[1]);
    v = _mm_set_pd(a0, a1);
    return *this;
}
```

Code Listing 10.6: The inverse tangent function for data of type `double2`

```
inline double2 operator+(double2 lhs, const double2& rhs)
{
    lhs += rhs;
    return lhs;
}
inline double2 operator*(double2 lhs, const double2& rhs)
{
    lhs *= rhs;
    return lhs;
}
inline double2 sqrt(double2 lhs)
{
    lhs.sqrt();
    return lhs;
}
```

Code Listing 10.7: Operators and functions for `v = sqrt(x) * y + 2.0 * z` to become valid code for the `double2` data type.

Up to this point, the arithmetic operators and functions are overloaded to act directly on the instances of the vector data types as member functions, for example `v.sqrt()` or `v *= 2.0`. In order to allow a 'more natural' way to write code, we de-

fine global operators. The strategy is to implement the same syntax for arithmetic operations for vector data types as well as built-in data types. This makes it possible to use identical code independent of the data type. The expression `v = sqrt(x) * y + 2.0 * z`, for example, is valid code for all vector data types. Code Listing 10.7 shows the operators and functions in C++ which are necessary in order to achieve that goal.

Let me stress once more that the use of this vectorisation library allows the use of the same source code for all vector data types. This is implemented by using the C++ template mechanism as explained in Chapter 11.1. We thus have no additional effort for the vectorised calculations with respect to the implementation with standard floating point calculations but gain a substantial increase in performance. Chapter 11.4 includes a performance comparison between the `double` data type and the vectorised versions with `double2` as well as `double4`. The speed-up is close to optimal.

Chapter

11

Implementation and performance

This chapter presents a toolbox of highly efficient methods which I implemented using the concepts of parallel computing that we have seen in the previous chapter. Let me stress that the toolbox is designed in such a way that it is no longer necessary to maintain multiple versions of the code for the different parallelisation and vectorisation strategies. In order to switch between either sequential execution or parallel is simply controlled by compiler flags as described above. Let me now present the layout of the toolbox which implements the algorithm I introduce in Chapter 7. Remember that the core task of the time stepping algorithm is to solve a system of ordinary differential equations (7.10),

$$\dot{\mathcal{Z}}^t(z) = F\left(\mathcal{Z}^t(z)\right),$$

which comprises the equations for the classical flow (2.3), the corresponding variational equation (5.6), and the classical action (7.8). Written out in full detail the system looks as follows.

$$\begin{pmatrix} \frac{d}{dt}X^t(z) \\ \frac{d}{dt}\Xi^t(z) \\ \frac{d}{dt}\vec{\mathrm{vec}}\left(\partial_q X^t(z)\right) \\ \frac{d}{dt}\vec{\mathrm{vec}}\left(\partial_p X^t(z)\right) \\ \frac{d}{dt}\vec{\mathrm{vec}}\left(\partial_q \Xi^t(z)\right) \\ \frac{d}{dt}\vec{\mathrm{vec}}\left(\partial_p \Xi^t(z)\right) \\ \frac{d}{dt}S_q(t,z) \\ \frac{d}{dt}S_p(t,z) \end{pmatrix} = \begin{pmatrix} \partial_p T(\Xi^t(z)) \\ -\partial_q V(X^t(z)) \\ \vec{\mathrm{vec}}\left(\partial_p^2 h(\Phi^t(z)) \cdot \partial_q \Xi^t(z)\right) \\ \vec{\mathrm{vec}}\left(\partial_p^2 h(\Phi^t(z)) \cdot \partial_p \Xi^t(z)\right) \\ \vec{\mathrm{vec}}\left(-\partial_q^2 h(\Phi^t(z)) \cdot \partial_q X^t(z)\right) \\ \vec{\mathrm{vec}}\left(-\partial_q^2 h(\Phi^t(z)) \cdot \partial_p X^t(z)\right) \\ T(\Xi^t(z)) \\ -V(X^t(z)) \end{pmatrix} \tag{11.1}$$

It is a system with d degrees of freedom requires the solution of $4d^2+2d+2$ ordinary differential equations and has to be solved for a huge number of different initial values $\{z_1, \ldots, z_M\} \subset \mathbb{R}^{2d}$ where M is the number of quadrature points in the phase space integral. The initial values are given by

$$\begin{pmatrix} X^0(z) \\ \Xi^0(z) \\ \vec{\mathrm{vec}}\left(\partial_q X^0(z)\right) \\ \vec{\mathrm{vec}}\left(\partial_p X^0(z)\right) \\ \vec{\mathrm{vec}}\left(\partial_q \Xi^0(z)\right) \\ \vec{\mathrm{vec}}\left(\partial_p \Xi^0(z)\right) \\ S_q(0,z) \\ S_p(0,z) \end{pmatrix} = \begin{pmatrix} q \\ p \\ \vec{\mathrm{vec}}\left(\mathrm{I}\right) \\ 0 \\ 0 \\ \vec{\mathrm{vec}}\left(\mathrm{I}\right) \\ 0 \\ 0 \end{pmatrix}. \tag{11.2}$$

Before explaining the layout, implementation, and performance of the algorithm in detail, let me introduce the following example.

Example 11.1 (Henon–Heiles potential)
The d-dimensional Henon–Heiles potential is defined by

$$V(x) = \frac{1}{2}\sum_{k=1}^{d} x_k^2 + \sigma \sum_{k=1}^{d-1}\left(x_k x_{k+1}^2 - \frac{1}{3}x_k^3\right) + \beta \sum_{k=1}^{d-1}\left(x_k^2 + x_{k+1}^2\right)^2 \tag{11.3}$$

where $\sigma > 0$ and $\beta > 0$ are two positive constants that control the strength of interaction and confinement respectively.

This particular model has become a standard example for testing semiclassical simulations in mathematics and chemistry, see e.g. [LR10], [FGL09], and [NM02], as well as many others. I therefore use this example in order to compare my methods to the ones in the literature.

11.1 Layout of the algorithm

Templates are a feature of the C++ programming language that allows functions and classes to work with many different data types without being rewritten for each one. Templates are of great utility, in particular when combined with the vectorisation library presented in Chapter 10.4. Everything starts with a main class called

herman_kluk that depends on two template parameters T and V, cf. Code Listing 11.1. The first one is the data type that is used for loops and is usually chosen to be the integer data type int while the latter data type is the one in which all floating point operations are performed and is thus either chosen to be double or one of the vectorised types double2 and double4.

```
template<class T, class V> class herman_kluk
{
(...)
public:
  herman_kluk(const T num_points, const T composition_order,
    const V semiclassical_parameter, const T dimension)
    {
    Z = (V*)_mm_malloc((6 * D * D + 2 * D + 7) * M * sizeof(V), 4096);
    V *pZ = Z;
    (...)
    }
  (...)
}
```

Code Listing 11.1: Definition of the main class herman_kluk that depends on two template parameters.

In order to create an instance of the class herman_kluk one needs to specify the number of quadrature points, the composition order γ of the ode solver, the semiclassical parameter ε as well as the dimension d of the problem. Basically, the only task that the main class actually performs is to allocate enough memory to store the solution to (7.10) for each initial individual sampling point. The actual computing kernels are defined in subclasses which all inherit the template parameters T and V. The first step is to arrange the problems in terms of the dimension their degrees of freedom, cf. Figure 11.1. Each class herman_kluk_*d only contains two important functions, one to set the parameters for the initial wave function and the second to evaluate the Herman–Kluk wave function at given point. All the actual computation kernels are included in yet another set of subclasses. Figure 11.2 shows this second partition into subclasses according to the type of potential that is considered. This is the basic layout for what I call the SemiClassical Toolbox. Code Listing 11.2, shows how an object named hk of class henon_6d is created. By means of its member functions we can then evolve hk in time or calculate the

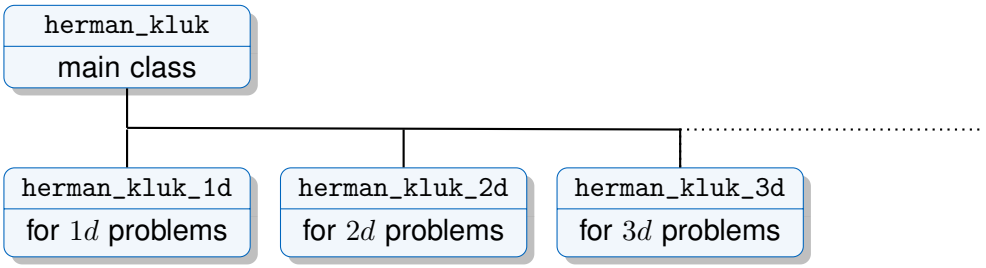

Figure 11.1: The main class, `herman_kluk`, is divided into different subclasses, `herman_kluk_*d`, one for each dimension.

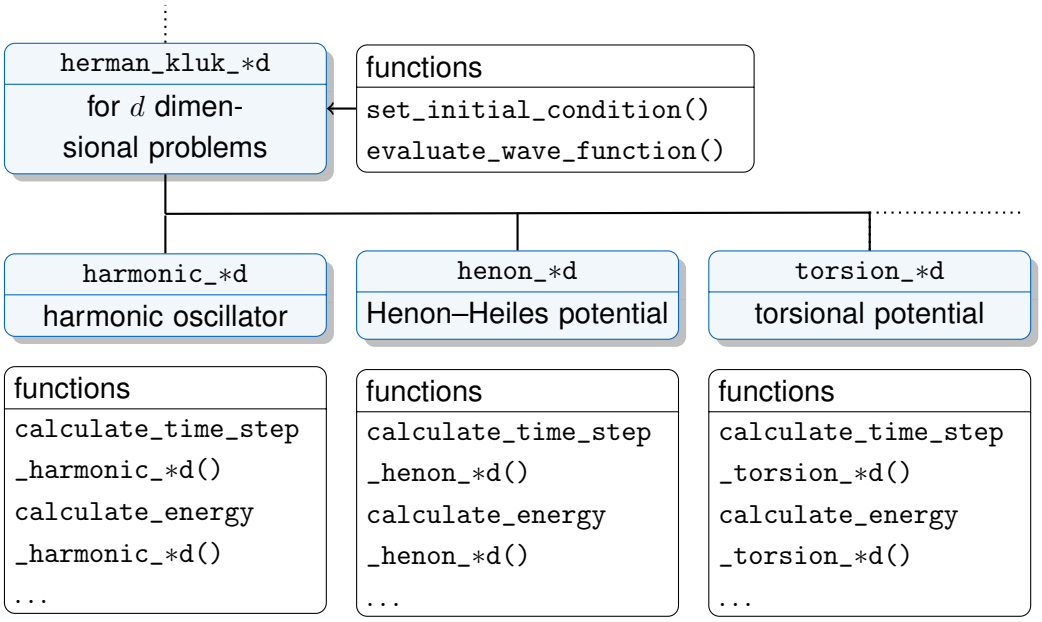

Figure 11.2: The dimension specific class `herman_kluk_*d` is divided into different subclasses according to the underlying problem that is considered. `herman_kluk_*d` as well as all of its subclasses have their own member functions for specific tasks.

energy expectation values of the system or evaluate its wave function. As you can see, the whole layout of the toolbox is in principle readily expandable to arbitrary systems in arbitrary dimensions. In practise, however, it requires a lot of effort to actually implement the classes and their respective member functions, in particular in high dimensions. The following chapter provides a solution to this predicament.

```
int main(int argc, char * argv[])
{
    int M =  128 * 1024;     // number of sampling points
    const int gamma = 8;     // composition_order
    double epsilon = 0.01;     // semiclassical parameter

    henon_6d<int, double> hk(M, gamma, epsilon);

}
```

Code Listing 11.2: Example for the creation of an object of class `henon_6d` which is capable of simulation the Herman–Kluk wave function for a Henon–Heiles potential in six dimensions.

11.2 Automated code generation

Let me redirect your attention to the system (11.1) of ordinary differential equations. For problems with many degrees of freedom and complicated potentials, it is a already a tedious task to calculate the gradient and the Hessian of the potential. Implementing the right hand side of (11.1) is even worse since we additionally have to perform four matrix-matrix multiplications. On the one hand we definitely want to avoid two nested for-loops in order to do so is. On the other hand, writing every single line of code by hand is almost not feasible. For $d = 12$, for example, we have $4d^2 + 2d + 2 = 602$ and even a simple Størmer–Verlet scheme would already require almost 2000 lines of code. And that is just the effort for one specific problem in one specific dimension. Not to mention the rate of typing errors for such an enterprise.

I resolve these issues by a rather elementary idea: In essence, the task of creating the code for the right hand side of (11.1) is a long series of string operations that has to be performed without creative thinking but with highest precision. That sounds like a task better suited for a computer than for a human being. In this spirit I developed an automated code generator in the computer algebra system Mathematica. It creates code of very high complexity by only a couple of well-placed commands. In addition, the generator is implemented in such a way that the creation of the code can be done by the same commands for arbitrary dimensions and potentials. With a sequence of further string operations, the Mathematica program is capable of generating the complete header files for the C++ classes ready for compilation.

Let me illustrate the modus operandi of the generator by creating the C++ source code for the time stepping scheme for the Henon–Heiles potential in three dimensions. It is given by

$$V(x) = \frac{1}{2}\sum_{k=1}^{3} x_k^2 + \sigma \sum_{k=1}^{2} \left(x_k x_{k+1}^2 - \frac{1}{3} x_k^3 \right) + \beta \sum_{k=1}^{2} \left(x_k^2 + x_{k+1}^2 \right)^2$$

where $\sigma > 0$ and $\beta > 0$, cf. Example 11.1. This specific example is already complex enough to understand the efficiency of the automatic code generation while it is still manageable enough as not to fill page upon page with code. As a first step we choose the dimension $\mathrm{d} = 3$ and initialise the variables we need. The $2\,\mathrm{d}$ components of the vector $z = (q, p) \in \mathbb{R}^{2d}$ and the $4\,\mathrm{d}^2$ entries of the matrices $\partial_q X^t, \partial_p X^t, \partial_q \Xi, \partial_p \Xi^t \in \mathbb{R}^{d \times d}$ are initialised as depicted in Code Listing 11.3. As a

```
d = 3;

q = Symbol["q" <> ToString[#]] & /@ Range[d];
p = Symbol["p" <> ToString[#]] & /@ Range[d];

Xq = Array[Symbol["Xq"<>ToString[#1]<>"x"<>ToString[#2]]&,{d, d}];
Xp = Array[Symbol["Xp"<>ToString[#1]<>"x"<>ToString[#2]]&,{d, d}];
Xiq = Array[Symbol["Xiq"<>ToString[#1]<>"x"<>ToString[#2]]&,{d, d}];
Xip = Array[Symbol["Xip"<>ToString[#1]<>"x"<>ToString[#2]]&,{d, d}];
```

Code Listing 11.3: Initialisation of the variables that are required for automatic code generation.

second step let us specify the potential, symbolically calculate its gradient and Hessian, and simplify the resulting expressions, cf. Code Listing 11.4. Furthermore, the matrix-matrix multiplication of the Hessian and W are also computed symbolically and the results are simplified. Let us again take a closer look at the system of equations (11.1) for which we want to implement one step of the Størmer–Verlet time stepping scheme. Code Listing 11.5 contains the Mathematica commands for the variables p, $\partial_q \Xi$, and S_p, the code for the rest of the variables is produced in the same way. The resulting C++ code is printed in Code Listing 11.6. As I have already mentioned above, the Mathematica commands are accompanied by a series of string operations in order to generate the complete header files for the C++ classes ready for compilation. This includes all the member functions, e.g. for a time

```
V = Sum[0.5 q[[k]]^2, {k, 1, d}] +
   a Sum[q[[k]] q[[k + 1]]^2 - q[[k]]^3/3, {k, 1, d - 1}] +
   b  Sum[(q[[k]]^2 + q[[k + 1]]^2)^2, {k, 1, d - 1}] // FullSimplify;
GradV = D[V, {q}] // FullSimplify;
HessV = D[V, {q, 2}] // FullSimplify;

HessVXq = HessV.Xq //FullSimplify;
HessVXp = HessV.Xp //FullSimplify;
```

Code Listing 11.4: Calculate of the gradient and the Hessian of the potential and the simplification thereof for automatic code generation.

```
ToString[
Array[ToString[p[[#1]]]<>"+="<>
ToString[CForm[t NGradV[[#1]]
//.{Power[x_,2]->HoldForm[x*x],Power[x_,3]->HoldForm[x*x*x]}]]
<>";"&,{d,1}]
]<>
ToString[
Array[ToString[Xiq[[#1,#2]]]<>"+="<>
ToString[CForm[t NHessVXq[[#1,#2]]]
//.{Power[x_,2]->HoldForm[x*x]}]<>";\t"&,{d,d}]
]<>
"Sp␣+=␣"<>
ToString[
CForm[t NV]
//.{Power[x_,2]->HoldForm[x*x],Power[x_,3]->HoldForm[x*x*x]}
];
```

Code Listing 11.5: Commands to produce the code for one step of the Størmer–Verlet time stepping scheme in the variables p, $\partial_q \Xi$, and S_p.

step or for the calculation of expectation values, cf. Figure 11.2. Code Listing 11.7 shows a condensed version of the final code which is ready for compilation.

```
p1 += 4. * t * (-0.25 * s * (q1 * q1) + b * (q1 * q1 * q1)
        + 0.25 * s * (q2 * q2) + q1 * (0.25 + b * (q2 * q2)));
p2 += t * (-s * (q2 * q2) + 8. * b * (q2 * q2 * q2)
        + s * (q3 * q3) + q2 * (1. + 2. * s * q1
        + 4. * b * (q1 * q1) + 4. * b * (q3 * q3)));
p3 += q3 * t * (1. + 2. * s * q2 + 4. * b * (q2 * q2 + q3 * q3));
Xiq1x1 += t * (2. * (s + 4. * b * q1) * q2 * Xq2x1
        + Xq1x1 * (1. - 2. * s * q1 + 12. * b * (q1 * q1)
        + 4. * b * (q2 * q2)));
Xiq1x2 += t * (2. * (s + 4. * b * q1) * q2 * Xq2x2
        + Xq1x2 * (1. - 2. * s * q1 + 12. * b * (q1 * q1)
        + 4. * b * (q2 * q2)));
Xiq1x3 += t * (2. * (s + 4. * b * q1) * q2 * Xq2x3
        + Xq1x3 * (1. - 2. * s * q1 + 12. * b * (q1 * q1)
        + 4. * b * (q2 * q2)));
Xiq2x1 += t * (2. * (s + 4. * b * q1) * q2 * Xq1x1
        + 2. * (s + 4. * b * q2) * q3 * Xq3x1
        + Xq2x1 * (1. + 2. * s * q1 - 2. * s * q2
        + 4. * b * (q1 * q1)
        + 24. * b * (q2 * q2) + 4. * b * (q3 * q3)));
Xiq2x2 += t * (2. * (s + 4. * b * q1) * q2 * Xq1x2
        + 2. * (s + 4. * b * q2) * q3 * Xq3x2
        + Xq2x2 * (1. + 2. * s * q1 - 2. * s * q2
        + 4. * b * (q1 * q1) + 24. * b * (q2 * q2)
        + 4. * b * (q3 * q3)));
Xiq2x3 += t * (2. * (s + 4. * b * q1) * q2 * Xq1x3
        + 2. * (s + 4. * b * q2) * q3 * Xq3x3
        + Xq2x3 * (1. + 2. * s * q1 - 2. * s * q2
        + 4. * b * (q1 * q1)
        + 24. * b * (q2 * q2) + 4. * b * (q3 * q3)));
Xiq3x1 += t * (2. * (s + 4. * b * q2) * q3 * Xq2x1
        + Xq3x1 * (1. + 2. * s * q2
        + 4. * b * (q2 * q2 + 3. * (q3 * q3))));
Xiq3x2 += t * (2. * (s + 4. * b * q2) * q3 * Xq2x2
        + Xq3x2 * (1. + 2. * s * q2
        + 4. * b * (q2 * q2 + 3. * (q3 * q3))));
Xiq3x3 += t * (2. * (s + 4. * b * q2) * q3 * Xq2x3
        + Xq3x3 * (1. + 2. * s * q2
        + 4. * b * (q2 * q2 + 3. * (q3 * q3))));
Sp += t * (0.5 * (q1 * q1) + 0.5 * (q2 * q2) + 0.5 * (q3 * q3)
        - 0.3333333333333333 * s * (q1 * q1 * q1 - 3. * q1 * (q2 * q2)
        + q2 * q2 * q2 - 3. * q2 * (q3 * q3))
        + b * ((q1 * q1 + q2 * q2) * (q1 * q1 + q2 * q2)
        + (q2 * q2 + q3 * q3) * (q2 * q2 + q3 * q3)));
```

Code Listing 11.6: C++ code for one time step of the Størmer–Verlet time stepping scheme in the variables p, $\partial_q \Xi$, and S_p that was created automatically by the commands in Code Listing 11.5.

```
template<class T, class V>class henon_confined_3d :
public herman_kluk_3d<T,V>
{
private:
    const V sigma;
    const V beta;
public:
    henon_confined_3d(const T num_points, const T composition_order,
    const V epsilon, const V coupling_constant,
    const V confinement_strength)
    : herman_kluk_3d<T,V>(num_points, composition_order, epsilon),
    sigma(coupling_constant),
    beta(confinement_strength)
    {
    }
    virtual ~henon_confined_3d()
    {
    }

    void operator()(const V dt, const T N)
    { (...) }

   virtual void calculate_energy( (...) )
   { (...) }

   virtual void calculate_initial_energy( (...) )
   { (...) }
};
```

Code Listing 11.7: Final header file for the definition of the class `henon_confined_3d` that was entirely created by a series of Mathematica commands.

11.3 Parallel Herman–Kluk method

Allow me to illustrate the performance of the parallel Herman–Kluk method on an example. Let us again consider the Henon–Heiles potential (11.3) in six dimensions with parameters $\sigma = \frac{1}{\sqrt{80}}$ and $\beta = \frac{\sigma^2}{16}$,

$$V(x) = \frac{1}{2}\sum_{k=1}^{6} x_k^2 + \sigma \sum_{k=1}^{5} \left(x_k x_{k+1}^2 - \frac{1}{3} x_k^3 \right) + \beta \sum_{k=1}^{5} \left(x_k^2 + x_{k+1}^2 \right)^2 . \tag{11.4}$$

In order to compare my results to those of Lasser and Röblitz [LR10] as well as Faou, Gradinaru, and Lubich [FGL09], I choose the semiclassical parameter $\varepsilon = 0.01$ and the initial wave function to be a Gaussian wave packet with $q_k = 2$ and $p_k = 0$ for all $k \in \{1, \ldots, d\}$. The numerical experiments concerning high performance were performed on the Mephisto compute cluster[1].

OpenMP

In the first experiment we observe how the methods scales with the number of OpenMP threads. As explained in Chapter 10.1 this number is controlled by setting the value of the environment variable `OMP_NUM_THREADS`. Table 11.3 catalogues the execution times for the time-evolution of the parallel Herman–Kluk method for 200 time steps of size 0.01. I use a composition Størmer–Verlet method of order 8 for

Intel Xeon CPU E3-1235 @ 3.20GHz		
threads	**execution time**	**speed-up**
1	170.968 s	
2	88.515 s	1.932
4	48.225 s	3.545
8	38.790 s	4.408
16	39.398 s	4.340

Intel Xeon CPU E5-2650 @ 2.00GHz		
threads	**execution time**	**speed-up**
1	286.697 s	
2	143.569 s	1.997
4	72.406 s	3.960
8	36.217 s	7.916
16	18.362 s	15.614

Table 11.3: Dependence on the number of OpenMP threads on two different CPUs

[1] cf. https://hpc-wiki.uni-graz.at/Wiki-Seiten/Mephisto.aspx for more information about the cluster as well as its hardware configuration.

the solution of the underlying system (11.1). This system is solved for 2^{19} distinct initial values in parallel which means that altogether over 80 million ordinary differential equations are solved. I conducted this experiment on two separate hardware configurations. The first one is an ordinary desktop computer holding a 1 x Intel Xeon CPU E3-1235 @ 3.20GHz with 4 cores and 16 GB of memory, the second one is a single compute node of the aforementioned Mephisto cluster containing a 2 x Intel Xeon Eight-Core CPU E5-2650 @ 2.00GHz with 16 cores and 256GB of memory. The methods scales perfectly with the number of threads no matter on which hardware configuration. I illustrat this fact in Figure 11.4.

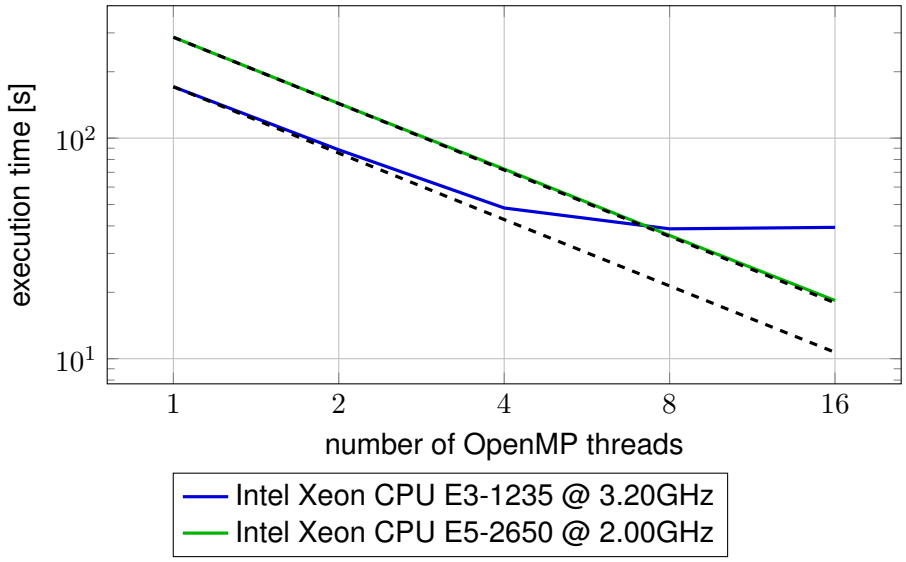

Figure 11.4: Visualisation of the dependence on the number of OpenMP threads on two different CPUs

The dashed lines describe a linear speed-up of the execution time with respect to the number of threads. As you can see, the measured times suggest that my method scales perfectly. However, it does not make sense to increase the number of threads above the number of physical compute cores of the processor. At 1, 2, and 4 threads the Intel Xeon CPU E3-1235 is faster than the Intel Xeon CPU E5-2650, simply because of the higher clock rate. But then the larger number of cores becomes the more important factor. This is a striking demonstration of the advantages of parallel computing.

Vectorisation Library

In order to test the performance of the vectorisation library introduced in Chapter 10.4, I conducted the same experiment as above for several different setups where the number of MPI threads varies from 1 to 16 and different vector data types are used. Table 11.5 shows the resulting execution times and Figure 11.6 visualises them. Note that each time the speed-up scales perfectly with the number of threads. The data type `double2` gives an almost optimal speed-up of a factor two and `double4` is about three times faster than the non-vectorised code.

no. of MPI threads	**data type** `double` **(64 bit)**	**speed-up**
1	881.14 s	
2	441.31 s	2.00
4	221.23 s	3.98
8	111.65 s	7.89
16	54.78 s	16.09
no. of MPI threads	**data type** `double2` **(128 bit)**	**speed-up**
1	449.78 s	1.96
2	226.34 s	3.89
4	113.61 s	7.76
8	57.38 s	15.36
16	28.05 s	31.41
no. of MPI threads	**data type** `double4` **(256 bit)**	**speed-up**
1	289.34 s	3.05
2	145.16 s	6.07
4	72.58 s	12.14
8	36.25 s	24.31
16	18.00 s	48.95

Table 11.5: Execution times with respect to the number of MPI threads as well as the specific vector data type.

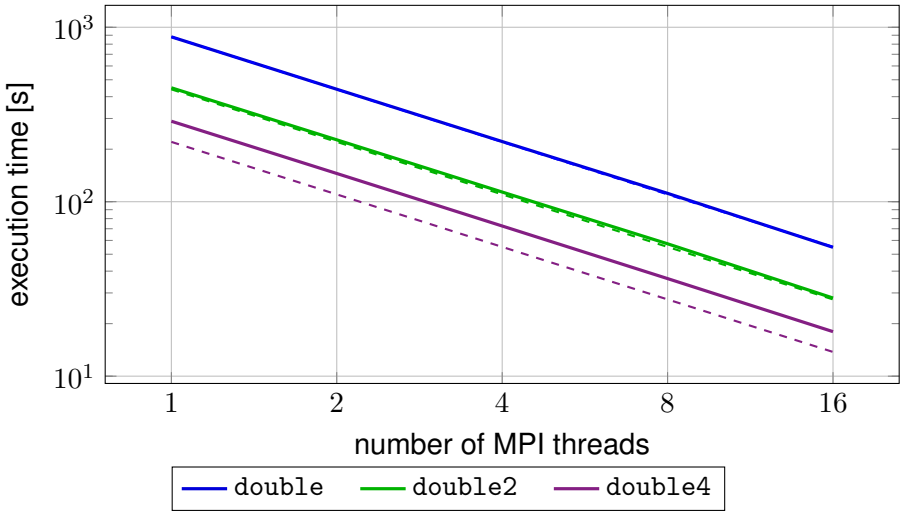

Figure 11.6: Visualisation of the almost perfect speed-up achieved by vectorisation.

11.4 Parallel Egorov method

Chapter 4.4 presents a way to calculate an approximation to the time evolution of the expectation value

$$\mathbb{E}_{\mathrm{qm}}[\mathcal{A}](t) = \langle \psi(t), \mathcal{A}\psi(t) \rangle$$

of a Weyl quantised operator $\mathcal{A} = \mathrm{op}^{\varepsilon}(a)$. The basic idea is to define a quasi-classical expectation value by

$$\mathbb{E}_{\mathrm{cl}}[a](t) = \int_{\mathbb{R}^{2d}} (a \circ \Phi^t)(z)\, W(\psi_0)(z)\, \mathrm{d}z.$$

by means of the Wigner transform $W\psi_0$ of the initial wave function ψ_0. Theorem 4.7 then guarantees us that the error between $\mathbb{E}_{\mathrm{qm}}$ and $\mathbb{E}_{\mathrm{qcl}}$ is of order $\mathcal{O}(\varepsilon^2)$. To formulate this more rigorously, for any $T > 0$ there exists $c > 0$ such that

$$\sup_{t \in [0,T]} |\mathbb{E}_{\mathrm{qm}}[\mathcal{A}](t) - \mathbb{E}_{\mathrm{qcl}}[a](t)| \leq c\, \varepsilon^2.$$

The constant c depends on the observable $\mathcal{A}$ and derivatives of the classical flow Φ^t, but is uniformly bounded for all normalised initial data with $\|\psi_0\| = 1$. I use the algorithm proposed by Lasser and Röblitz [LR10] as a starting point to create a highly parallelised method capable of efficiently computing expectation values for systems with literally hundreds of degrees of freedom. The key idea in [LR10]

is to use the fact that the Wigner function $W\psi_0$ is a quasi-probability distribution as described by Proposition 4.4. If the initial datum is a Gaussian wave packet $\psi_0 = g^\varepsilon$ we already know that $W\psi_0$ is the density function of a multivariate normal distribution, see Example 4.5. In this case we just draw M independent samples $\{z_1, \dots, z_M\}$ from this initial distribution and use Monte Carlo or quasi-Monte Carlo techniques to approximate the time evolution of the expectation values by

$$\int_{\mathbb{R}^{2d}} (a \circ \Phi^t)(z)\, W(\psi_0)(z)\, \mathrm{d}z \approx \frac{1}{M} \sum_{m=1}^{M} (a \circ \Phi^t)(z_m).$$

This is almost the same approach as the one I use to discretise the phase space integral of the Herman–Kluk propagator in Chapter 8. The discretisation is done in two steps, the first is the interpretation of the method as a weighted integral over phase space and the sampling of the initial distribution. The second is the classical transport of the involved quantities. In this case, all that is required is the classical flow Φ^t and therefore we only need to solve

$$\begin{pmatrix} \frac{d}{dt} X^t(z) \\ \frac{d}{dt} \Xi^t(z) \end{pmatrix} = \begin{pmatrix} \partial_p T(\Xi^t(z)) \\ -\partial_q V(X^t(z)) \end{pmatrix}, \quad \begin{pmatrix} X^0(z) \\ \Xi^0(z) \end{pmatrix} = \begin{pmatrix} q \\ p \end{pmatrix} \tag{11.5}$$

for all $z = (q,p)^T \in \{z_1 \dots z_M\}$ instead of the full system (11.1). This allows me to use almost the same basic programming design for the implementation of the Herman–Kluk and the Egorov method. The previous chapter demonstrates how well the parallel Herman–Kluk method performs with respect to the number of threads and to vectorisation techniques. Here, I want to present how well the parallel Egorov method performs when implemented on a graphics processing unit. For reasons of reproducibility I chose the same confined Henon–Heiles potential as Keller and Lasser [KL14]. It is again of the general form (11.3) with constants $\sigma = 1.8436$ and $\beta = 0.4$, i.e.

$$V(x) = \frac{1}{2} \sum_{k=1}^{d} x_k^2 + 1.8436 \sum_{k=1}^{d-1} \left(x_k x_{k+1}^2 - \frac{1}{3} x_k^3 \right) + 0.4 \sum_{k=1}^{d-1} \left(x_k^2 + x_{k+1}^2 \right)^2.$$

The semiclassical parameter has a value of $\varepsilon = 0.0037$ and the initial wave function is a Gaussian wave packet

$$\psi_0(x) = (\pi\varepsilon)^{-d/4} \exp\left(-\frac{1}{2\varepsilon} |x-q|^2 + \frac{\mathrm{i}}{\varepsilon}\, p \cdot (x-q) \right) \tag{11.6}$$

centred at position $q_k = 0.1215$ and momentum $p_k = 0$ for all $k \in \{1, \dots, d\}$. The parallel Egorov method is employed to simulate the system for an increasing number of degrees of freedom. For this simulation, 2^{18} Monte Carlo points and a composition Størmer–Verlet method of order 8 with a time step of 1 femtosecond are

used in order to transport them up to a final time of $100\,\mathrm{fs}$ for each individual calculation. Figure 11.7 shows the time-evolution of the potential energy for 16 up to 512 dimensions. Keller and Lasser [KL14] perform a similar experiment starting from

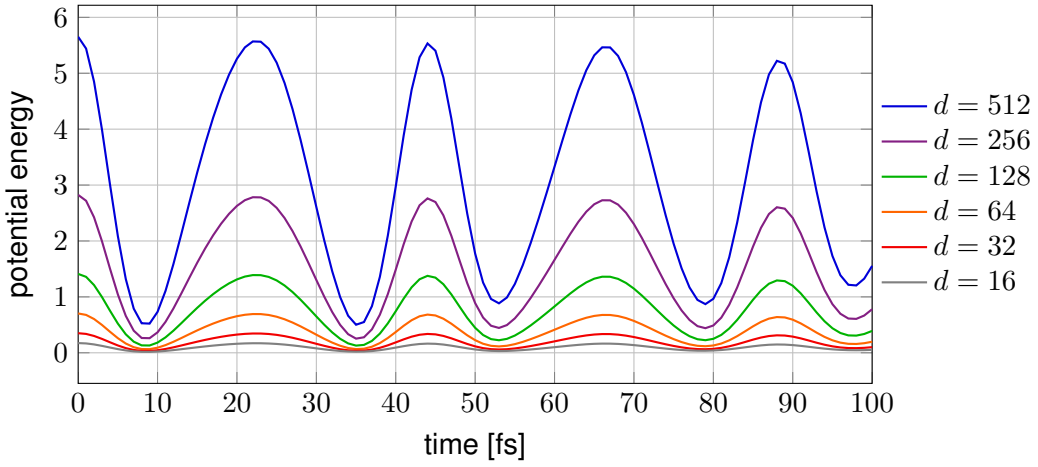

Figure 11.7: Evolution of the expectation values of the potential energy for various dimensions.

2 and going to 32 degrees of freedom. They use 2^{12} sampling points and report an execution time of $19.8\mathrm{s}$ for $d = 32$. Table 11.8 lists the respective execution times of my algorithm. For the 32 dimensional problem the GPU code on a Nvidia Tesla K20 is over 140 times faster compared to a single CPU core and about 9 times faster compared to a dual socket eight core Intel Xeon E5-2650 @ 2.00GHz compute node. For higher dimensional systems the GPU performance advantage drops to a factor 3 compared to the dual socket compute node. The reason for the performance drop is the exhaustion of the 255 registers per thread. Another interesting aspect is reflected in the compilation time needed for the Henon–Heiles simulation which increases dramatically with the number of degrees of freedom of the underlying simulation. This is a byproduct of the fact that the code was automatically generated. The automatic code generator which I introduce in Chapter 11.2 produces very explicit instructions as illustrated by Code Listing 11.6. This is very efficient for actual computations since it does not involve any complicated construct but it requires an enormous effort from the compiler in order to translate it. For CPU and GPU code I used the GNU compiler g++ (version $4.8.2$) and the Nvidia's nvcc compiler (version 7.0) respectively. The times are listed in Table 11.9.

dim	2 × Intel Xeon E5-2650	1 × Nvidia Tesla K20	speed-up
16	4.260 s	0.641 s	6.646
32	10.386 s	1.180 s	8.802
64	24.793 s	4.412 s	5.619
128	109.393 s	29.266 s	3.738
256	219.183 s	71.980 s	3.045
512	454.614 s	155.915 s	2.916

Table 11.8: Execution times for the Egorov method for the Henon–Heiles potential.

dim	nvcc	g++
16	9.037 s	2.113 s
32	9.916 s	3.181 s
64	11.527 s	5.887 s
128	20.539 s	20.193 s
256	35.137 s	71.920 s
512	133.548 s	352.200 s

Table 11.9: Compilation times for Nvidia nvcc 7.0 and GNU g++ 4.8.2

Epilogue

Chapter 12

Conclusion

Let me conclude this dissertation with some short remarks. It achieves the development of a scheme for the numerical calculation of the Herman–Kluk propagator. I prove that this scheme behaves as desired under phase space and time discretisation. In order to do so I require techniques from various mathematical fields, most prominently from functional analysis and operator theory. Moreover, in order to solve the underlying classical equations of motion, a system of ordinary differential equations, I introduce the concept of symplectic manifolds and vector spaces and examine schemes that preserve these structures. This includes the notion of backward error analysis, a well-established concept from numerical analysis. Since complex-valued functions occur in the calculation I even have to include techniques from complex analysis. Additionally, Some number theory is needed for the construction of quasi-Monte Carlo points. Combining all these ideas with state-of-the-art parallel programming strategies I finally create highly efficient, stable tools to calculate approximate solutions to the semiclassical Schrödinger equation for a large number of degrees of freedom.

I am certain that these tools are entitled to a prominent spot in a quantum mechanic's toolbox.

Suggestions for further research

The next two logical steps would be the derivation of higher order corrections to these methods as well as the extension to explicitly time dependent Hamiltonians. Swart and Rousse [SR09] and Kay [Kay06] show that it is in principle possible to achieve both goals.

This would also allow the treatment of optimal control problems with objective functionals such as

$$J(\psi, \alpha) = \frac{1}{2} \langle \psi(T, \cdot), \mathcal{A}\psi(T, \cdot) \rangle + \frac{\gamma}{2} \langle \alpha, \alpha \rangle , \tag{12.1}$$

where $\mathcal{A}$ is a bounded linear operator on L^2, which is assumed to be essentially self-adjoint and hence a physical observable. Several different cost terms are discussed in mathematical and chemical literature. The optimality conditions for the objective functional (12.1) together with the time dependent Schrödinger equation

$$\mathrm{i}\varepsilon\partial_t\psi = -\frac{\varepsilon^2}{2}\Delta\psi + V(x)\psi - \alpha(t)\mu(x)\psi, \quad x \in \mathbb{R}^d, t \in \mathbb{R} \tag{12.2}$$

are given as follows.

$$\begin{aligned}
&\mathrm{i}\,\varepsilon\,\partial_t\psi = H\psi - \alpha\mu\psi, \quad \psi(0,\cdot) = \psi_0 \\
&\mathrm{i}\,\varepsilon\,\partial_t\chi = H\chi - \alpha\mu\chi, \quad \chi(T,\cdot) = -\mathrm{i}\mathcal{A}\psi(T,\cdot) \\
&\alpha = -\frac{1}{\gamma}\langle\chi, \mu\,\psi\rangle \\
&\lambda = -\mathrm{i}\chi(0,\cdot)
\end{aligned}$$

A direct solution by means of the Herman–Kluk propagator proves to be problematic since the wave function has to be evaluated at the final time T. It remains unclear on how to do this while still obeying Assumption 7.1. A different ansatz would be to approximate (12.1) with the help of Egorov's Theorem and then start an investigation on this approximated objective functional by means of optimal control for ordinary differential equations. I believe that this is an idea worth considering.

Appendix A

Expectation values for Gaussian wave packages

In Chapter 7.2 I describe a way to calculate an approximation to the expectation value of a multiplication operator. The process involves evaluating integrals of the form

$$\int_{\mathbb{R}^d} \overline{g^{\varepsilon}_{z^{(1)}}(x)}\, V(x) g^{\varepsilon}_{z^{(2)}}(x)\, dx$$

where $z^{(1)} = (q^{(1)}, p^{(1)}) \in \mathbb{R}^{2d}$ and $z^{(2)} = (q^{(2)}, p^{(2)}) \in \mathbb{R}^{2d}$ are elements of phase space, cf. Equation (7.2). As mentioned above, there are several cases in which this integral may be computed analytically. Let us give some examples.

Example A.1 *Let us first consider the case $\mathcal{A}_1 = \mathrm{I}$. This means that we have to compute the scalar product of two Gaussian wave packages with the same width parameter but possibly different centres, $z^{(1)} = (q^{(1)}, p^{(1)}), z^{(2)} = (q^{(2)}, p^{(2)}) \in \mathbb{R}^{2d}$.*

$$\begin{aligned}
&\left\langle g^{\varepsilon}_{z^{(1)}}, g^{\varepsilon}_{z^{(2)}} \right\rangle = \int_{\mathbb{R}^d} \overline{g^{\varepsilon}_{z^{(1)}}(x)}\, g^{\varepsilon}_{z^{(2)}}(x)\, dx \\
&= (\pi\varepsilon)^{-\frac{d}{2}} \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\varepsilon}\left(\left|x-q^{(1)}\right|^2 + \left|x-q^{(2)}\right|^2\right)\right)\dots \\
&\qquad \exp\left(-\frac{\mathrm{i}}{\varepsilon}\, p^{(1)} \cdot \left(x-q^{(1)}\right) + \frac{\mathrm{i}}{\varepsilon}\, p^{(2)} \cdot \left(x-q^{(2)}\right)\right) dx \\
&= (\pi\varepsilon)^{-\frac{d}{2}} \int_{\mathbb{R}^d} \exp\left(-\frac{1}{\varepsilon}\left|x - \frac{1}{2}q^{(1)} + \frac{1}{2}q^{(2)} + \frac{\mathrm{i}}{2}\left(p^{(1)} - p^{(2)}\right)\right|^2\right)\, dx\ \dots \\
&\qquad \exp\left(\frac{1}{4\varepsilon}\left|q^{(1)} + q^{(2)} + \mathrm{i}\left(p^{(1)} - p^{(2)}\right)\right|^2 - \frac{1}{2\varepsilon}\left(\left|q^{(1)}\right|^2 + \left|q^{(2)}\right|^2\right)\right)\dots \\
&\qquad \exp\left(\frac{\mathrm{i}}{\varepsilon}\left(p^{(1)} \cdot q^{(1)} - p^{(2)} \cdot q^{(2)}\right)\right) \\
&= \exp\left(-\frac{1}{4\varepsilon}\left|z^{(1)} - z^{(2)}\right|^2 - \frac{\mathrm{i}}{2\varepsilon}\left(p^{(1)} + p^{(2)}\right) \cdot \left(q^{(1)} - q^{(2)}\right)\right).
\end{aligned}$$

Example A.2 (harmonic oscillator) *Consider $\mathcal{A}_2 = |x|^2 / 2$ to be the potential energy of the harmonic oscillator. By partial integration one obtains*

$$\begin{aligned}
\left\langle g^{\varepsilon}_{z^{(1)}}, \mathcal{A}_2\, g^{\varepsilon}_{z^{(2)}} \right\rangle &= \int_{\mathbb{R}^d} \overline{g^{\varepsilon}_{z^{(1)}}(x)} \sum_{k=1}^{d} \frac{x_k^2}{2}\, g^{\varepsilon}_{z^{(2)}}(x)\, dx \\
&= \sum_{k=1}^{d} \frac{1}{8}\left(2\varepsilon - \left((p_k^{(1)} - p_k^{(2)}) + \mathrm{i}\,(q_k^{(1)} + q_k^{(2)})\right)^2\right) \left\langle g^{\varepsilon}_{z^{(1)}}, g^{\varepsilon}_{z^{(2)}} \right\rangle.
\end{aligned}$$

Example A.3 (kinetic energy) *Using an ε-scaled version of the Fourier transform allows us to calculate the above integral for the kinetic energy operator $\mathcal{A}_3 = -\frac{\varepsilon^2}{2}\Delta$. The result is*

$$\begin{aligned}
\left\langle g^{\varepsilon}_{z^{(1)}}, \mathcal{A}_3\, g^{\varepsilon}_{z^{(2)}} \right\rangle &= \int_{\mathbb{R}^d} \overline{g^{\varepsilon}_{z^{(1)}}(x)} \left(-\frac{\varepsilon^2}{2}\Delta\right) g^{\varepsilon}_{z^{(2)}}(x)\, dx \\
&= \sum_{k=1}^{d} \frac{1}{8}\left(2\varepsilon - \left((p_k^{(1)} + p_k^{(2)}) + \mathrm{i}\,(q_k^{(1)} - q_k^{(2)})\right)^2\right) \left\langle g^{\varepsilon}_{z^{(1)}}, g^{\varepsilon}_{z^{(2)}} \right\rangle.
\end{aligned}$$

Remark A.4 Note that, by repeated application of the techniques used in Examples A.2 and A.3, analytic expressions may be found for any observable that is a poynomial of position and momentum operator. This includes the Henon-Heiles potential.

We need not restrict ourselves to polynomial observables. The above integral may also be calculated analytically for trigonometric potentials.

Example A.5 (torsional potential) *Consider $\mathcal{A}_4$ to be the torsional potential. In a similar manner as in the previous examples one obtains the expression*

$$\begin{aligned}\left\langle g^{\varepsilon}_{z^{(1)}}, \mathcal{A}_4\, g^{\varepsilon}_{z^{(2)}} \right\rangle &= \int\limits_{\mathbb{R}^d} \overline{g^{\varepsilon}_{z^{(1)}}(x)} \sum_{k=1}^{d} (1-\cos(x_k))\; g^{\varepsilon}_{z^{(2)}}(x)\, dx \\ &= \sum_{k=1}^{d} \left(1 - \mathrm{e}^{-\frac{\varepsilon}{4}} \cosh\left(\frac{(p_k^{(1)} - p_k^{(2)})}{2} + \frac{\mathrm{i}(q_k^{(1)} + q_k^{(2)})}{2}\right)\right) \left\langle g^{\varepsilon}_{z^{(1)}}, g^{\varepsilon}_{z^{(2)}} \right\rangle .\end{aligned}$$

Figures and Tables

Code Listings

Bibliography

[AD15] Christoph Aistleitner and Josef Dick. 'Functions of bounded variation, signed measures, and a general Koksma-Hlawka inequality.' *Acta Arith.* 167.2 (2015), pp. 143–171. DOI: 10.4064/aa167-2-4.

[Arn89] Vladimir I. Arnol'd. *Mathematical Methods of Classical Mechanics*. Vol. 60. Graduate Texts in Mathematics. Springer New York, 1989. ISBN: 978-1-4757-2063-1. DOI: 10.1007/978-1-4757-2063-1.

[BO27] Max Born and Robert Oppenheimer. 'Zur Quantentheorie der Molekeln'. *Annalen der Physik* 389.20 (1927), pp. 457–484. DOI: 10.1002/andp.19273892002.

[Boh13] Niels Bohr. 'I. On the constitution of atoms and molecules'. *Philosophical Magazine Series 6* 26.151 (1913), pp. 1–25. DOI: 10.1080/14786441308634955.

[Boh20] Niels Bohr. 'Über die Serienspektra der Elemente'. *Zeitschrift für Physik* 2.5 (1920), pp. 423–469. DOI: 10.1007/BF01329978.

[Bor26] Max Born. 'Zur Quantenmechanik der Stoßvorgänge'. *Zeitschrift für Physik* 37.12 (1926), pp. 863–867. DOI: 10.1007/BF01397477.

[Ego69] Yurii V. Egorov. 'The canonical transformations of pseudodifferential operators'. *Uspekhi Mat. Nauk* 24.5 (149) (1969), pp. 235–236.

[Ein05] Albert Einstein. 'Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt'. *Annalen der Physik* 322.6 (1905), pp. 132–148. DOI: 10.1002/andp.19053220607.

[Fey94] Richard P. Feynman. *The Character of Physical Law*. Modern Library, New York, 1994, pp. xx + 167. ISBN: 0-679-60127-9.

[FGL09] Erwan Faou, Vasile Gradinaru, and Christian Lubich. 'Computing semiclassical quantum dynamics with Hagedorn wavepackets.' *SIAM J. Sci. Comput.* 31.4 (2009), pp. 3027–3041. DOI: 10.1137/080729724.

[FL06] Erwan Faou and Christian Lubich. 'A Poisson Integrator for Gaussian Wavepacket Dynamics'. *Computing and Visualization in Science* 9.2 (2006), pp. 45–55. DOI: 10.1007/s00791-006-0019-8.

[Gal55] Galileo Galilei. *Discorsi e dimostrazioni matematiche*. 1655.

[GH14] Vasile Gradinaru and George A. Hagedorn. 'Convergence of a semiclassical wavepacket based time-splitting for the Schrödinger equation'. *Numer. Math.* 126.1 (2014), pp. 53–73. DOI: 10.1007/s00211-013-0560-6.

[GL14] Wolfgang Gaim and Caroline Lasser. 'Corrections to Wigner type phase space methods'. *Nonlinearity* 27.12 (2014), p. 2951. DOI: 10.1088/0951-7715/27/12/2951.

[Gos06] Maurice de Gosson. *Symplectic geometry and quantum mechanics*. Vol. 166. Operator Theory: Advances and Applications. Birkhäuser Basel, 2006. ISBN: 978-3-7643-7575-1. DOI: 10.1007/3-7643-7575-2.

[Hag80a] George A. Hagedorn. 'A time dependent Born-Oppenheimer approximation'. *Communications in Mathematical Physics* 77.1 (1980), pp. 1–19. DOI: 10.1007/BF01205036.

[Hag80b] George A. Hagedorn. 'Semiclassical quantum mechanics'. *Comm. Math. Phys.* 71.1 (1980), pp. 77–93. DOI: 10.1007/BF01230088.

[Hag98] George A. Hagedorn. 'Raising and Lowering Operators for Semiclassical Wave Packets'. *Ann. Physics* 269.1 (1998), pp. 77–104. DOI: 10.1006/aphy.1998.5843.

[Ham34] William R. Hamilton. 'On a General Method in Dynamics; by which the Study of the Motions of all free Systems of attracting or repelling Points is reduced to the Search and Differentiation of one central Relation, or characteristic Function'. *Philosophical Transactions of the Royal Society* (1834), pp. 247–308.

[Hei25] Werner Heisenberg. 'Über quantentheoretische Umdeutung kinematischer und mechanischer Beziehungen.' *Zeitschrift für Physik* 33.1 (1925), pp. 879–893. DOI: 10.1007/BF01328377.

[Hei27] Werner Heisenberg. 'Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik'. *Zeitschrift für Physik* 43.3-4 (1927), pp. 172–198. DOI: 10.1007/BF01397280.

[Hel76] Eric J. Heller. 'Time dependent variational approach to semiclassical dynamics'. *J. Chem. Phys.* 64.1 (1976), pp. 63–73. DOI: 10.1063/1.431911.

[Hel81] Eric J. Heller. 'Frozen Gaussians: A very simple semiclassical approximation'. *J. Chem. Phys.* 75.6 (1981), pp. 2923–2931. DOI: 10.1063/1.442382.

[Hig98] Nicholas J. Higham. *Handbook of Writing for the Mathematical Sciences*. Second. Society for Industrial and Applied Mathematics, 1998. DOI: 10.1137/1.9780898719550. eprint: http://epubs.siam.org/doi/pdf/10.1137/1.9780898719550.

[HK06] Gili Hochman and Kenneth G. Kay. 'Semiclassical corrections to the Herman-Kluk propagator'. *Phys. Rev. A* 73 (6 2006), p. 064102. DOI: 10.1103/PhysRevA.73.064102.

[HK84] Michael F. Herman and Edward Kluk. 'A semiclassical justification for the use of non-spreading wavepackets in dynamics calculations'. *Chem. Phys.* 91.1 (1984), pp. 27–34. DOI: 10.1016/0301-0104(84)80039-7.

[HLW06] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration. Structure-preserving algorithms for ordinary differential equations*. Second edition. Springer Series in Computational Mathematics 31. Springer, Berlin, 2006. DOI: 10.1007/3-540-30666-8.

[Hud74] Richard L. Hudson. 'When is the Wigner quasi-probability density non-negative?' *Reports on Mathematical Physics* 6.2 (1974), pp. 249 –252. DOI: 10.1016/0034-4877(74)90007-X.

[Iag75] Daniel Iagolnitzer. 'Appendix Microlocal essential support of a distribution and decomposition theorems, An introduction'. In: *Hyperfunctions and Theoretical Physics*. Vol. 449. Lecture Notes in Math. Springer, Berlin, 1975, pp. 121–132. ISBN: 978-3-540-07151-8. DOI: 10.1007/BFb0062919.

[JMS11] Shi Jin, Peter Markowich, and Christof Sparber. 'Mathematical and computational methods for semiclassical Schrödinger equations'. *Acta Numerica* 20 (May 2011), pp. 121–209. DOI: 10.1017/S0962492911000031.

[Kat51] Tosio Kato. 'Fundamental properties of hamiltonian operators of Schrödinger type.' *Trans. Am. Math. Soc.* 70 (1951), pp. 195–211. DOI: 10.2307/1990366.

[Kay06] Kenneth G. Kay. 'The Herman–Kluk approximation: Derivation and semiclassical corrections'. *Chemical Physics* 322.1-2 (2006), pp. 3–12. DOI: http://dx.doi.org/10.1016/j.chemphys.2005.06.019.

[KL12] M. Kretz and V. Lindenstruth. 'Vc: A C++ Library for Explicit Vectorization'. *Softw. Pract. Exper.* 42.11 (2012), pp. 1409–1430. DOI: 10.1002/spe.1149.

[KL14] Johannes Keller and Caroline Lasser. 'Quasi-classical description of molecular dynamics based on Egorov's theorem'. *The Journal of Chemical Physics* 141.5, 054104 (2014). DOI: 10.1063/1.4891517.

[KL97] William Kahan and Ren-Cang Li. 'Composition constants for raising the orders of unconventional schemes for ordinary differential equations.' *Math. Comput.* 66.219 (1997), pp. 1089–1099. DOI: 10.1090/S0025-5718-97-00873-9.

[KU98] Arnold R. Krommer and Christoph W. Ueberhuber. *Computational Integration*. Society for Industrial and Applied Mathematics, 1998. ISBN: 978-0-89871-374-9. DOI: 10.1137/1.9781611971460.

[Lag88] Joseph-Louis Lagrange. *Méchanique analitique*. La Veuve Desaint, Paris, 1788, pp. XII + 512.

[Lit86] Robert G. Littlejohn. 'The semiclassical evolution of wave packets'. *Phys. Rep.* 138.4–5 (1986), pp. 193 –291. DOI: 10.1016/0370-1573(86)90103-1.

[LR10] Caroline Lasser and Susanna Röblitz. 'Computing expectation values for molecular quantum dynamics.' *SIAM J. Sci. Comput.* 32.3 (2010), pp. 1465–1483. DOI: 10.1137/090770461.

[LRT13] Hailiang Liu, Olof Runborg, and Nicolay M. Tanushev. 'Error estimates for Gaussian beam superpositions.' *Math. Comput.* 82.282 (2013), pp. 919–952. DOI: 10.1090/S0025-5718-2012-02656-1.

[LS15] Manfred Liebmann and David Sattlegger. 'Parallel algorithms for semiclassical quantum dynamics'. *Preprint No. IGDK-2015-25* (2015).

[LT14] Caroline Lasser and Stephanie Troppmann. 'Hagedorn wavepackets in time-frequency and phase space.' *J. Fourier Anal. Appl.* 20.4 (2014), pp. 679–714. DOI: 10.1007/s00041-014-9330-9.

[Lub08] Christian Lubich. *From Quantum to Classical Molecular Dynamics: Reduced Models and Numerical Analysis*. Zurich lectures in advanced mathematics. European Mathematical Society, Zürich, 2008, pp. ix + 144. ISBN: 978-3-03719-067-8.

[Mar02] André Martinez. *An Introduction to Semiclassical and Microlocal Analysis*. Universitext. Springer New York, 2002, pp. viii + 190. ISBN: 978-1-4757-4495-8. DOI: 10.1007/978-1-4757-4495-8.

[Mil74] William H. Miller. 'Quantum mechanical transition state theory and a new semiclassical model for reaction rate constants'. *J. Chem. Phys.* 61.5 (1974), pp. 1823–1834. DOI: 10.1063/1.1682181.

[MMC90] H.-D. Meyer, U. Manthe, and L. S. Cederbaum. 'The multi-configurational time-dependent Hartree approach'. *Chemical Physics Letters* 165.1 (1990), pp. 73 –78. DOI: http://dx.doi.org/10.1016/0009-2614(90)87014-I.

[Moy49] José Enrique Moyal. 'Quantum mechanics as a statistical theory'. *Mathematical Proceedings of the Cambridge Philosophical Society* 45 (01 1949), pp. 99–124. DOI: 10.1017/S0305004100000487.

[MPI15] Message Passing Interface Forum MPI. *A Message-Passing Interface Standard Version 3.1*. 2015. URL: http://www.mpi-forum.org/docs/.

[Neu32a] John von Neumann. *Mathematische Grundlagen der Quantenmechanik*. (Die Grundlehren der Mathematischen Wissenschaften in Einzeldarstellungen, Bd. XXXVIII). Berlin, J. Springer, 1932, p. 262.

[Neu32b] John von Neumann. 'Über Einen Satz von Herrn M. H. Stone'. *Annals of Mathematics*. Second Series 33.3 (1932), pp. 567–573. DOI: 10.2307/1968535.

[Neu96] John von Neumann. *Mathematische Grundlagen der Quantenmechanik*. 2. Aufl. Springer Berlin Heidelberg, 1996, pp. x + 262. ISBN: 978-3-642-64828-5. DOI: 10.1007/978-3-642-61409-5.

[New87] Isaac Newton. *Philosophiæ Naturalis Principia Mathematica*. Jussu Societatis Regiae ac typis Josephi Streater, prostant venales apud Sam. Smith, 1687.

[Nie92] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, 1992. ISBN: 978-0-89871-295-7. DOI: 10.1137/1.9781611970081.

[NM02] M. Nest and H.-D. Meyer. 'Benchmark calculations on high-dimensional Henon–Heiles potentials with the multi-configuration time dependent Hartree (MCTDH) method'. *The Journal of Chemical Physics* 117.23 (2002), pp. 10499–10505. DOI: 10.1063/1.1521129.

[Ope13] Architecture Review Board OpenMP. *Application Program Interface Version 4.0*. 2013. URL: http://openmp.org/wp/openmp-specifications/.

[Pla01] Max Planck. 'Über die Elementarquanta der Materie und der Elektricität'. *Annalen der Physik* 309.3 (1901), pp. 564–566. DOI: 10.1002/andp.19013090311.

[Poi99] Henry Poincaré. *Les méthodes nouvelles de la mécanique céleste, Tome III*. Gauthier-Villars et fils, Paris, 1899.

[Rob87] Didier Robert. *Autour de l'approximation semi-classique*. Vol. 68. Progress in Mathematics. Birkhäuser Boston, 1987, pp. x + 329. ISBN: 0-8176-3354-5.

[RR10] Thomas Rauber and Gudula Rünger. *Parallel Programming*. Springer Berlin Heidelberg, 2010, pp. XIII + 516. ISBN: 978-3-642-04818-0. DOI: 10.1007/978-3-642-04818-0.

[RS08] Vidian Rousse and Torben Swart. 'Global L^2-Boundedness Theorems for Semiclassical Fourier Integral Operators with Complex Phase'. *arXiv e-prints* (2008). arXiv: 0710.4200v3.

[RS75] Michael Reed and Barry Simon. *Fourier analysis, Self-Adjointness*. Methods of Modern Mathematical Physics Vol. 2. Academic Press, 1975. ISBN: 978-0125850025.

[SC83] Francisco Soto and Pierre Claverie. 'When is the Wigner function of multidimensional systems nonnegative?' *Journal of Mathematical Physics* 24.1 (1983), pp. 97–100. DOI: 10.1063/1.525607.

[Sch26a] Erwin Schrödinger. 'Quantisierung als Eigenwertproblem. I. (Erste Mitteilung.)' *Annalen der Physik* 384.4 (1926), pp. 361–376. DOI: 10.1002/andp.19263840404.

[Sch26b] Erwin Schrödinger. 'Quantisierung als Eigenwertproblem. II. (Zweite Mitteilung.)' *Annalen der Physik* 384.6 (1926), pp. 489–527. DOI: 10.1002/andp.19263840602.

[Sch26c] Erwin Schrödinger. ‘Quantisierung als Eigenwertproblem. III. (Dritte Mitteilung.)’ *Annalen der Physik* 385.13 (1926), pp. 437–490. DOI: 10.1002/andp.19263851302.

[Sch26d] Erwin Schrödinger. ‘Quantisierung als Eigenwertproblem. IV. (Vierte Mitteilung.)’ *Annalen der Physik* 386.18 (1926), pp. 109–139. DOI: 10.1002/andp.19263861802.

[SR09] Torben Swart and Vidian Rousse. ‘A Mathematical Justification for the Herman-Kluk Propagator’. *Comm. Math. Phys.* 286.2 (2009), pp. 725–750. DOI: 10.1007/s00220-008-0681-4.

[ST01] Herbert Spohn and Stefan Teufel. ‘Adiabatic Decoupling and Time-Dependent Born–Oppenheimer Theory’. *Communications in Mathematical Physics* 224.1 (2001), pp. 113–132. DOI: 10.1007/s002200100535.

[ST83] Ian Stewart and David Tall. *Complex Analysis. The hitchhiker’s guide to the plane*. Cambridge University Press, 1983, pp. VIII + 290. ISBN: 0-521-24513-3. DOI: 10.1017/CBO9781139171632.

[Sto32] Marshall H. Stone. ‘On One-Parameter Unitary Groups in Hilbert Space’. *Annals of Mathematics*. Second Series 33.3 (1932), pp. 643–648. DOI: 10.2307/1968538.

[Stø07] Carl Størmer. ‘Sur les trajectoires des corpuscules électriques dans l’espace sous l’action du magnétisme terrestre, avec application aux aurores boréales.’ *Arch. phys. et nat.* 24.4 (1907), pp. 5–8, 113–158, 221–247.

[TW04] Michael Thoss and Haobin Wang. ‘Semiclassical description of molecular dynamics based on initial-value representation methods’. *Annu. Rev. Phys. Chem.* 55.1 (2004), pp. 299–332. DOI: 10.1146/annurev.physchem.55.091602.094429.

[Ver67] Loup Verlet. ‘Computer “Experiments” on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules’. *Phys. Rev.* 159 (1 1967), pp. 98–103. DOI: 10.1103/PhysRev.159.98.

[Wey27] Hermann Weyl. ‘Quantenmechanik und Gruppentheorie’. *Zeitschrift für Physik* 46.1-2 (1927), pp. 1–46. DOI: 10.1007/BF02055756.

[Wig32] Eugene Wigner. ‘On the Quantum Correction For Thermodynamic Equilibrium’. *Phys. Rev.* 40 (5 1932), pp. 749–759. DOI: 10.1103/PhysRev.40.749.

[Wil60] James H. Wilkinson. ‘Error analysis of floating-point computation’. *Numerische Mathematik* 2.1 (1960), pp. 319–340. DOI: 10.1007/BF01386233.

[Zhe14] Chunxiong Zhen. ‘Optimal error estimates for first-order Gaussian beam approximations to the Schrödinger equation.’ *SIAM J. Numer. Anal.* 52.6 (2014), pp. 2905–2930. DOI: 10.1137/130935720.