Spawning of wave-packets in 1D non-adiabatic transitions

Term Thesis

written by
Raoul Bourquin

supervised by
Dr. Vasile Grădinaru
and
Prof. Dr. Ralf Hiptmair

Seminar for Applied Mathematics
ETH Zurich

Spring semester 2011

Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

In this chapter we introduce the fundamental physical and mathematical ideas and structures on which the other chapters build. The central objects here are the time-dependent Schrödinger equation and a non-adiabatic potential.

1.1 The time-dependent Schrödinger equation

Time-dependent problems in quantum physics are governed by the time-dependent Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\left|\varphi\right\rangle = H\left|\varphi\right\rangle \ . \tag{1.1}$$

The Hamiltonian of the system in consideration is given by H, and the function $\varphi\left(x,t\right)$ represents the wavefunction dependent on position x and time t. In d space dimensions this is

$$\begin{aligned}\varphi : \mathbb{R}^d \times \mathbb{R} &\to \mathbb{C} \\ \left(x,t\right) &\mapsto \varphi\left(x,t\right) \ .\end{aligned}$$

There are various mathematical restrictions on what is a valid wavefunction. For example φ has to be square-integrable. Most of these preconditions have little importance for us.

1.2 Semi-classical scaling

We use the semiclassical scaling, where $\varepsilon > 0$ is a real parameter [1]. The equation still keeps its mathematical form

$$i\varepsilon^2\frac{\partial}{\partial t}\left|\psi\right\rangle = H\left|\psi\right\rangle \ . \tag{1.2}$$

It's well known that we get the classical dynamics from the limit $\hbar \to 0$. The same holds of course for the semiclassical parameter ε and for bigger ε we get an increasing amount of quantum effects.

[1] Other authors use ε or even $\hbar$ (without its physical meaning and value) for the quantity we denote by ε^2.

The Hamiltonian operator H is composed of two parts, the kinetic operator T and the potential operator V. Thus we can split H as $H = T + V$ with the following definitions for both operators

$$\begin{aligned} T &:= -\frac{1}{2}\varepsilon^4 \frac{\partial^2}{\partial x^2} \\ V &:= V(x) \,. \end{aligned} \tag{1.3}$$

The mass m which is present in the common definition of the kinetic operator is included in the parameter ε. The potential is a real valued function depending only on space but not on time. This static potential results from the Born-Oppenheimer approximation for the electronic structure problem. For a more detailed theoretical background see for example reference [12]. Assume the potential is given by

$$\begin{aligned} V : \mathbb{R}^d &\to \mathbb{R} \\ x &\mapsto V(x) \end{aligned} \tag{1.4}$$

then this allows us to solve the Schrödinger equation by separation of variables and obtain an analytical result for the time propagation of a quantum state $|\psi(t)\rangle$

$$|\psi(t)\rangle = e^{-\frac{i}{\varepsilon^2}Ht} |\psi(0)\rangle \,. \tag{1.5}$$

The solutions to this time propagation have fine details. A typical wavepacket is highly oscillatory with a wavelength $\mathcal{O}(\varepsilon^2)$ localized in space with $\mathcal{O}(\varepsilon^2)$ and moving with a velocity of $\mathcal{O}(1)$. This tiny structures are a challenge for the algorithms simulating them. We would need a very fine grid and thus a huge bunch of grid nodes.

1.3 Non-adiabatic potentials, avoided crossings

Non-adiabatic potentials are potentials that consist of multiple energy levels. These energy levels may intersect each other, but we are interested in a situation that is called an *avoided crossing*. That is, the two energy surfaces always stay in a minimal distance. We call this distance the energy gap and denote it by δ. A very simple example of such an avoided crossing with only two energy levels is shown in figure 1.1.
Based on the class of physical potential in consideration we have a strict monotone order of the eigenvalues for all x in our space [2]

$$\lambda_0(x) > \lambda_1(x) > \ldots > \lambda_{N-1}(x) \qquad \forall x \,. \tag{1.6}$$

This global consistent order allows us to sort the eigenvalues and the corresponding eigenvectors uniquely in decreasing order.

[2] This is a fairly strong assumption that can be replaced by much weaker formulations more suitable for mathematical analysis. But for our purpose it is sufficient and these details don't really matter.

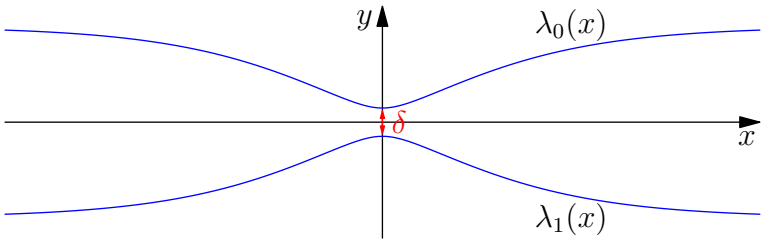


Figure 1.1: Example of an avoided crossing of two energy levels.

For a more elaborate study of the mathematical details and a classification of different types of avoided crossings see reference [8].
We are now interested in what happens with an incoming wavepacket $|\psi\rangle$ while it traverses the narrow part. The magnitude δ of the energy gap plays an important role in this process.

1.4 A vector of states

For the study of avoided crossings of the energy levels we are interested in vector valued states $|\Psi\rangle$. Each component of this vector represents a part of the wavefunction being on the corresponding energy surface.
To describe the dynamics of these states, we need to generalize the Schrödinger equation to a vector valued version. This is not difficult to do, basically the extended equation looks exactly like (1.2) but with the difference that H is a matrix now. Let's write down this in more detail because we will refer to it over and over again.
Assume we deal with N different states hence $|\Psi\rangle$ consists of N components $\varphi_i(x)$. And the Hamiltonian becomes a real valued symmetric $N \times N$ matrix. This gives the following expression for the time-dependent Schrödinger equation

$$i\varepsilon^2 \frac{\partial}{\partial t} \left| \begin{pmatrix} \varphi_0 \\ \vdots \\ \varphi_{N-1} \end{pmatrix} \right\rangle = \begin{pmatrix} & H & \end{pmatrix} \underbrace{\left| \begin{pmatrix} \varphi_0 \\ \vdots \\ \varphi_{N-1} \end{pmatrix} \right\rangle}_{|\Psi\rangle} . \tag{1.7}$$

1.5 The potential

In the case of a non-adiabatic potential with multiple energy levels, the potential V becomes a matrix. We assume that V depends on space x but not on time t, thus it is time-independent.
The matrix representing V is symmetric and with entries $v_{i,j} \equiv v_{j,i} \in \mathbb{R}$. We may write a general unspecified potential as

$$V(x) =: \begin{pmatrix} v_{0,0}(x) & \cdots & v_{0,N-1}(x) \\ \vdots & & \vdots \\ v_{N-1,0}(x) & \cdots & v_{N-1,N-1}(x) \end{pmatrix} \tag{1.8}$$

where each of the matrix entries $v_{i,j}(x)$ is a real valued function

$$\begin{aligned} v_{i,j} : \mathbb{R}^d &\to \mathbb{R} \\ x &\mapsto v_{i,j}(x) \end{aligned}$$

on its own. These functions are assumed to be smooth.

1.5.1 Diagonalization of the potential

We are much more interested in the potential's eigenvalues which are the energy levels of our system. A well known result from linear algebra tells us that symmetric matrices always have only real eigenvalues. Therefore we can diagonalize this matrix and obtain pure real eigenvalues $\lambda_i(x)$ that depend on the space variable x.
The diagonalization itself is performed by orthogonal matrices, the same theorem as above guarantees that we have a full set of orthogonal eigenvectors $\nu_i(x)$ which depend of course on x too.
Given the full set of eigenvalues $\lambda_0(x), \ldots, \lambda_{N-1}(x)$ and the corresponding eigenvectors of $V(x)$ denoted by $\nu_0(x), \ldots, \nu_{N-1}(x)$ the spectral decomposition of the potential's matrix reads

$$\Lambda(x) = M^{-1}(x) V(x) M(x) \tag{1.9}$$

where the matrix Λ is diagonal with the eigenvalues λ_i on its diagonal. The transformation matrix M is orthogonal and contains the eigenvectors as columns

$$M := \begin{pmatrix} \nu_0(x) & , \ldots , & \nu_{N-1}(x) \end{pmatrix} . \tag{1.10}$$

The special case for 2 energy levels

A general potential that only contains two energy levels is given by a symmetric two by two matrix. In this case we can write down the eigenvalues for the potential in closed form. The following formulae are defined in more detail in [11]. Suppose the potential matrix is given by

$$V(x) := \begin{pmatrix} v_1 & v_2 \\ v_2 & -v_1 \end{pmatrix} \tag{1.11}$$

with trace $\mathrm{Tr}(V) = 0$. Then we define a θ as

$$\theta := \frac{1}{2} \arctan\left(\frac{v_2}{v_1}\right) . \tag{1.12}$$

For the numerical computation we have to use the atan2 function to get the signs correct. Finally we can write the two eigenvectors as

$$\nu_0 := \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \quad \nu_1 := \begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix} . \tag{1.13}$$

Obviously they are orthogonal and normed. Remember that if we sort the λ_i we have to change the order of the eigenvectors as well. It's not guaranteed that ν_0 always belongs to λ_0.

1.5.2 Basis transformations of states

For various calculations later on we need to be able to transform states from and to the eigenbasis. This is in principle a trivial process of linear algebra, but lets briefly note the important points.
The transformation from the eigenbasis to the canonical basis will be important when we set up the initial values for a simulation. Assume we have a wavefunction $|\varphi^e\rangle$ given in the eigenbasis. The transformed state in the canonical basis is given by

$$|\varphi^c\rangle = M\,|\varphi^e\rangle \tag{1.14}$$

where M contains the column vectors ν_i.
The opposite transformation becomes important when evaluating observables. Given a state $|\varphi^c\rangle$ in the canonical basis, the image of the transformation into the eigenbasis is

$$|\varphi^e\rangle = M^{\mathrm{T}}\,|\varphi^c\rangle \tag{1.15}$$

where we simplified $M^{-1} = M^{\mathrm{T}}$ for real orthogonal matrices.

Chapter 2

Semiclassical Wavepackets

2.1 Basis functions

In this section we present a particular form of wavepackets defined by G. Hagedorn, see for example [6, 7] and particularly [9].
These wavepackets are a general class of orthonormal basis functions for an $L^2\left(\mathbb{R}^d\right)$ space. For the d dimensional space they are defined as follows. Let $q \in \mathbb{R}^d$ be the position and $p \in \mathbb{R}^d$ the momentum vector of the package. Further there are complex matrices $P, Q \in \mathbb{C}^{d\times d}$ which obey the following important relations

$$\begin{aligned} Q^{\mathrm{T}}P - P^{\mathrm{T}}Q &= 0 \\ Q^{\mathrm{H}}P - P^{\mathrm{H}}Q &= 2i\mathbb{1}\,. \end{aligned} \tag{2.1}$$

With these parameters we can now define the ground state wavefunction ϕ_0 depending on arbitrary but fixed parameters as

$$\begin{aligned} \phi_0\left[P,Q,p,q\right](x) := \left(\pi\varepsilon^2\right)^{-\frac{d}{4}} \det\left(Q\right)^{-\frac{1}{2}} \\ \cdot \exp\left(\frac{i}{2\varepsilon^2}\left\langle (x-q)\,, PQ^{-1}\,(x-q)\right\rangle + \frac{i}{\varepsilon^2}\left\langle p, (x-q)\right\rangle\right) \end{aligned} \tag{2.2}$$

where $x \in \mathbb{R}^d$. Also ε enters this equation with a constant numerical value during all computations[1].
The eigenfunctions of the harmonic oscillator are contained as special cases in the more general formulae for these wavepackets.
For the semiclassical wavepackets we can define and use ladder operators in the same manner as one does for the harmonic oscillator. This analogy builds on the fact that these wavepackets diagonalize the general quadratic Hamiltonian. Before we define these operators, let's restrict the dimension of the position space to one. This way we can avoid some difficulties that have no relevance for us now.

[1] In contrast to some other authors we use the notation of $P := iB$ and $Q := A$ and $q := a$ for the position and $p := \eta$ for the momentum. The motivation for this change are the equations of motion of P and Q that become the classical equations.

2.1.1 Restriction to one space dimension

The restriction to one space dimension where $d = 1$ simplifies things a lot because the vectors p and q and especially the matrices P and Q all reduce to scalar values. Further we don't need to bother with multi-index notation for k.
First we simplify the ground state (2.2) to the one dimensional case

$$\phi_0 \left[P, Q, p, q\right](x) := \left(\pi\varepsilon^2\right)^{-\frac{1}{4}} Q^{-\frac{1}{2}} \exp\left(\frac{i}{2\varepsilon^2} P Q^{-1} \left(x - q\right)^2 + \frac{i}{\varepsilon^2} p \left(x - q\right)\right). \tag{2.3}$$

2.1.2 Ladder operators

Now let's take a closer look at the ladder operators from reference [9]. As mentioned above there exists a lowering operator $\mathcal{L}$ and a raising operator $\mathcal{R}$ for semiclassical wavepackets. We will use these ladder operators later for defining the wavefunctions ϕ_k of higher states $k \geq 1$. The ladder operators are defined as

$$\begin{aligned} \mathcal{R} &= \frac{i}{\sqrt{2\varepsilon^2}} \left(\overline{P}\left(x - q\right) - \overline{Q}\left(y - p\right)\right) \\ \mathcal{L} &= -\frac{i}{\sqrt{2\varepsilon^2}} \left(P\left(x - q\right) - Q\left(y - p\right)\right) \end{aligned} \tag{2.4}$$

where $y := -i\varepsilon^2\nabla$. It exists a lowest state which can not be lowered further by $\mathcal{L}$. This state is the zero state and acts as the bottom of this ladder

$$\mathcal{L}\phi_0 = 0\,. \tag{2.5}$$

On the other hand we can apply the raising operator to the ground state and create ϕ_1

$$\phi_1 = \mathcal{R}\phi_0\,. \tag{2.6}$$

In the same way we can create ϕ_k for arbitrary k by applying $\mathcal{R}$ multiple times. To be more concrete, the following formulae bring the different states into relation

$$\begin{aligned} \phi_{k+1} &= \frac{1}{\sqrt{k+1}} \mathcal{R}\phi_k \\ \phi_{k-1} &= \frac{1}{\sqrt{k}} \mathcal{L}\phi_k\,. \end{aligned} \tag{2.7}$$

To get the state ϕ_k from the given ϕ_0 we have to let $\mathcal{R}$ act k times. Together with the prefactors this yields

$$\phi_k := \frac{1}{\sqrt{k!}} \mathcal{R}^k \phi_0\,. \tag{2.8}$$

Finally we can give an analytical closed form for the function ϕ_k

$$\phi_k\left[P,Q,p,q\right](x) := 2^{-\frac{k}{2}}\left(k!\right)^{-\frac{1}{2}}\left(\pi\varepsilon^2\right)^{-\frac{1}{4}} Q^{-\frac{k+1}{2}}\overline{Q}^{\frac{k}{2}} \cdot H_k\left(\varepsilon^{-1}\left|Q\right|^{-1}(x-q)\right) \cdot \exp\left(\frac{i}{2\varepsilon^2}PQ^{-1}(x-q)^2 + \frac{i}{\varepsilon^2}p(x-q)\right) \quad (2.9)$$

where $H_k(\xi)$ is the Hermite polynomial [2] of degree k. For an arbitrary but fixed $\varepsilon > 0$ and fixed parameters P, Q and given position q and momentum p these functions ϕ_k build a complete basis set $\{\phi_k\}_{k=0}^{\infty}$ of the space L^2. This infinite basis can be truncated at some upper value $K \gg 0$ and we get the set $\{\phi_k\}_{k=0}^{K}$ of functions. This will be used as basis for the semiclassical wavepackets later on.
To end this section, let's emphasize again the close relationship to the eigenfunctions of the harmonic oscillator. Suppose $P = i$, $Q = 1$, choose the origin as position and assume the wavepacket has no momentum, thus $p = q = 0$. Further, assume $\varepsilon = 1$. If we now plug these values in (2.9) we get

$$|\varphi_k\rangle = \frac{H_k(x)\, e^{-\frac{x^2}{2}}}{\pi^{\frac{1}{4}} 2^{\frac{k}{2}}\sqrt{k!}} \quad (2.11)$$

which is exactly the well known expression for the harmonic oscillator.

2.2 Definition of scalar wavepackets

After we have defined a basis set $\{\phi_0, \phi_1, \ldots\}$ for the infinite dimensional Hilbert space of states we can now define the exact form of a state $|\Phi\rangle$. Each state is represented by a linear combination of the basis functions ϕ_k. Of course the single basis functions are valid states too.
Assume we truncate the Hilbert space and use finite many basis functions. Let K be the maximal number of basis functions. Further let $S \in \mathbb{C}$ be a global phase. Any scalar wavepacket can now be written as

$$|\Phi(x)\rangle := e^{\frac{iS}{\varepsilon^2}} \sum_{k=0}^{K-1} c_k \phi_k(x) \quad (2.12)$$

where $c_k \in \mathbb{C}$ are the coefficients of this linear combination. We can collect them in a vector $c := (c_0, \ldots, c_{K-1})^{\mathrm{T}}$. For later reference, we call the set

$$\Pi := \{P, Q, S, p, q\} \quad (2.13)$$

of variables the *Hagedorn parameters* of a scalar wavepacket $|\Phi\rangle$ of form (2.12). This set includes the four parameters P, Q, p and q that come from the basis functions ϕ_k given by equation (2.9) as well as the global phase [3] that enters the above linear combination. These values play an important role in the wavepacket based algorithms discussed in [2].

[2] We use the following definition for the Hermite polynomials

$$H_k(\xi) := (-1)^k e^{\xi^2}\left(\frac{\partial}{\partial\xi}\right)^k e^{-\xi^2} \quad (2.10)$$

[3] Depending on the context, the global phase S may or may not be part of the set denoted by Π.

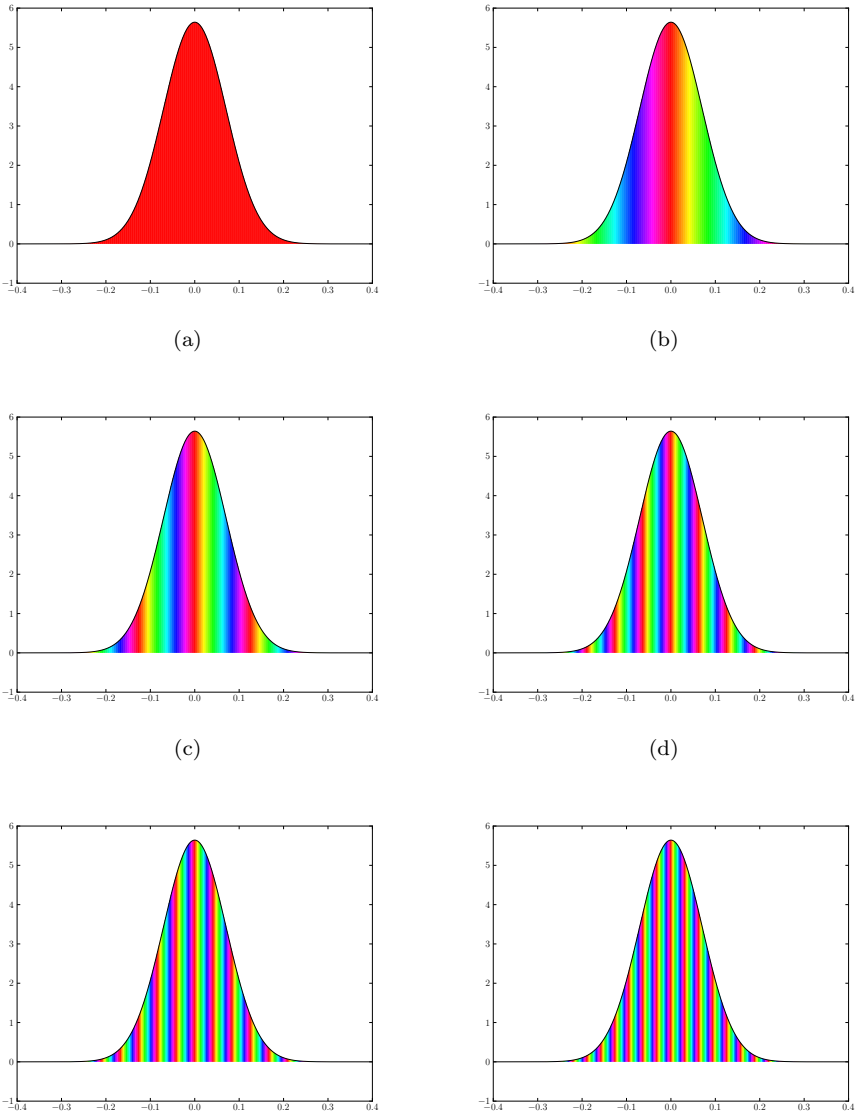


Figure 2.1: Hagedorn wavepackets $|\Psi\rangle$ with increasing momentum. The plots show the absolute value $\langle\Psi \,|\, \Psi\rangle$ and the local phase. The other parameters are $\varepsilon = 0.1$ and $P = i$, $Q = 1$ and $S = 0$. (a) $q = 0.0$ and $p = 0.0$ (b) $q = 0.0$ and $p = 0.25$ (c) $q = 0.0$ and $p = 0.5$ (d) $q = 0.0$ and $p = 1.0$ (e) $q = 0.0$ and $p = 1.5$ (f) $q = 0.0$ and $p = 2.0$

2.3 Homogeneous vector valued wavepackets

For the quantum dynamics with semiclassical wavepackets in the case of the vector valued Schrödinger equation as defined by formula (1.7) we need a wavepacket $|\Psi\rangle$ that is vector valued as well. Such a packet $|\Psi\rangle$ can be build as a vector of multiple scalar semiclassical packets. Assume that the Hamiltonian H in (1.7) is a $N \times N$ matrix, thus there are N energy levels and $|\Psi\rangle$ needs to have N components too. Formally we define

$$|\Psi\rangle := \left| \begin{pmatrix} \Phi_0(x) \\ \vdots \\ \Phi_{N-1}(x) \end{pmatrix} \right\rangle \tag{2.14}$$

where each of the Φ_i is of the form defined in (2.12).
All Φ_i share the same parameters P, Q, global phase S, average momentum q and average position p. We will call a wavepacket that fulfills this condition a *homogeneous wavepacket*. Equivalently we can say the only thing that differs between the Φ_i is the vector of coefficients c. Therefore we add an index i to the notation, c^i stands for the coefficient vector of the component Φ_i. Thus a semiclassical wavepacket suitable for solving (1.7) has the important property that it is fully characterized by a single set Π of parameters P, Q, S, p and q and a vector c^i of coefficients for each component Φ_i.

2.4 Inhomogeneous vector valued wavepackets

For advanced applications we may extend the definition (2.14) of a state $|\Psi\rangle$ and release the main restriction. In contrast to the homogeneous wavepackets of the last section we allow that each component Φ_i has it's very own set of parameters P, Q, S, p and q. We call such a wavepacket an *inhomogeneous wavepacket*. To be able to distinguish the different variables an index i is added also to the parameters Π. Thus a wavepacket is fully characterized by a set Π_i of parameters P_i, Q_i, S_i, p_i and q_i and a vector c^i of coefficients for each component Φ_i.

2.5 Numerical evaluation

2.5.1 Numerical evaluation of basis functions

For the numerical simulation we will need to evaluate the functions $\phi_k(x)$ at some discrete grid nodes x_i. This seems to be a trivial task as we have a closed form expression for ϕ_k given by equation (2.9). Although there is this expression for all k it's a bad idea to use it directly. One critical point in this formula is the factorial. It will soon result in a numerical overflow even for relatively small k. Therefore we need a better approach. An idea is to evaluate the ground state and recursively calculate the higher states based on these values. The essential three term recursion can be obtained as follows. We start with the function for ϕ_0 which we can evaluate numerically without much troubles. It is just a Gaussian exponential. (Note that we omit a factor of $Q^{-\frac{1}{2}}$ for the moment.)

Applying the raising operator $\mathcal{R}$ once results in

$$\phi_1(x) = Q^{-1}\sqrt{\frac{2}{\varepsilon^2}}(x-q)\cdot\phi_0 \tag{2.15}$$

and for the general case we get the following three term recursion by applying $\mathcal{R}$ on ϕ_k and rearranging the terms

$$\phi_{k+1}(x) = \sqrt{\frac{2}{\varepsilon^2}}\frac{1}{\sqrt{k+1}}Q^{-1}(x-q)\,\phi_k(x) - \sqrt{\frac{k}{k+1}}Q^{-1}\overline{Q}\phi_{k-1}(x)\ . \tag{2.16}$$

This is exactly how the calculation is implemented in an efficient and numerically stable way. Because later we will need the values for all ϕ_k from $k=0$ up to a maximum $k_{\max} =: K$, it's not a disadvantage but rather a big benefit that we have to evaluate all previous functions for any ϕ_k.

Algorithm 1 Evaluate basis functions $\phi_k(x)$ of semiclassical wavepackets

Require: A set of grid or quadrature nodes x
Require: A set $\Pi := \{P, Q, p, q\}$ of parameters
  // Base cases
  $\beta_0 := \pi^{-\frac{1}{4}}\varepsilon^{-\frac{1}{2}}\cdot\exp\left(\frac{i}{\varepsilon^2}\left(\frac{1}{2}PQ^{-1}(x-q)^2 + p(x-q)\right)\right)$
  $\beta_1 := Q^{-1}\sqrt{\frac{2}{\varepsilon^2}}(x-q)\cdot\beta_0$
  // Inductive steps
  for $k := 2$ **to** $K-1$ **do**
    $\beta_k := Q^{-1}\sqrt{\frac{2}{\varepsilon^2}}\frac{1}{\sqrt{k}}\cdot(x-q)\cdot\beta_{k-1} - Q^{-1}\overline{Q}\sqrt{\frac{k-1}{k}}\cdot\beta_{k-2}$
  end for
  return $B := (\beta_0, \ldots, \beta_{K-1})^{\mathrm{T}}$

In praxis we do not call algorithm 1 for each grid node $x_i \in \Gamma$ but use vectorization and calculate B for all nodes simultaneously so the returned B is a two-dimensional matrix array.

2.5.2 Numerical evaluation of wavepackets

The numerical evaluation of a wavepacket $|\Psi\rangle$ on given grid nodes x_i is not difficult. We just evaluate all the basis functions ϕ_k and assemble the parts. If we have an homogeneous wavepacket we can do this once and use these values for all N components of $|\Psi\rangle$. Otherwise we have to evaluate the basis functions individually for each component n of an inhomogeneous wavepacket as the basis functions differ by their Hagedorn parameters Π. Then we multiply these values with the coefficients c^n for each component. Finally we have to multiply with the global phase exponential $e^{\left(\frac{iS}{\varepsilon^2}\right)}$. For an inhomogeneous wavepacket we have to keep in mind that each component n has its own phase S_n. The algorithm 2 shows this procedure in the most general form. The outer for loop iterates over all components of $|\Psi\rangle$ while the inner loop is responsible for evaluating the basis functions ϕ_k^n with Π^n given per component n. This part can be implemented efficiently according to algorithm 1.

Algorithm 2 Evaluate a vector valued wavepacket $|\Psi\rangle$ on a set of nodes

Require: A set of grid or quadrature nodes x
Require: An arbitrary (in)homogeneous wavepacket Ψ
  // Iterate over all components of $|\Psi\rangle$
  for $n = 0$ **to** $N - 1$ **do**
    given Π_n as $\{P_n, Q_n, S_n, p_n, q_n\}$
    // Evaluate the basis for component n
    for $k = 0$ **to** $K - 1$ **do**
      $\beta_k := \phi_k \left[P_n, Q_n, p_n, q_n\right](x)$
    end for
    // Calculate the exponential of the phase
    $\pi_n := \exp\left(\frac{iS_n}{\varepsilon^2}\right)$
    // Assemble the component Φ_n
    $\Phi_n := \pi_n \cdot \sum_{k=0}^{K-1} c_k^n \beta_k$
  end for
  return $(\Phi_0, \ldots, \Phi_{N-1})^{\mathrm{T}}$

In the evaluation of the basis functions above we leave out a factor of $\frac{1}{\sqrt{Q}}$ that will cancel with a Q from the quadrature rule in (3.20) later.

Chapter 3

Inner products, integrals and quadrature

3.1 A hierarchy of brakets

In this chapter we will develop and summarize all necessary tools related to inner products of wavepackets. We will follow a top down approach and start with the braket of a full vector valued wavepacket $|\Psi\rangle$ which may be homogeneous or inhomogeneous at the moment. The primes indicate that the bra and the ket can have different parameter sets Π. This is obvious when $|\Psi\rangle$ is an inhomogeneous wavepacket. Also we include an operator F which may be the identity.

$$\begin{aligned} \langle \Psi \,|\, F \,|\, \Psi' \rangle &= \left\langle \begin{pmatrix} \Phi_0 \\ \vdots \\ \Phi_{N-1} \end{pmatrix} \middle| \begin{pmatrix} & \vdots & \\ \dots & F_{r,c} & \end{pmatrix} \middle| \begin{pmatrix} \Phi'_0 \\ \vdots \\ \Phi'_{N-1} \end{pmatrix} \right\rangle \\ &= \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} \langle \Phi_r \,|\, F_{r,c} \,|\, \Phi'_c \rangle \end{aligned} \tag{3.1}$$

where F is a $N \times N$ block matrix consisting of $K \times K$ blocks denoted by $F_{i,j} =: f$. We then consider a single term out of this double sum

$$\begin{aligned} \left\langle \Phi_i \,\middle|\, f \,\middle|\, \Phi'_j \right\rangle &= \left\langle \begin{pmatrix} \phi_0 \\ \vdots \\ \phi_{K-1} \end{pmatrix} \middle| \begin{pmatrix} & \vdots & \\ \dots & f_{k,l} & \end{pmatrix} \middle| \begin{pmatrix} \phi'_0 \\ \vdots \\ \phi'_{K-1} \end{pmatrix} \right\rangle \\ &= \sum_{k=0}^{K-1} \sum_{l=0}^{K-1} \langle \phi_k \,|\, f_{k,l} \,|\, \phi'_l \rangle \end{aligned} \tag{3.2}$$

where ϕ_i are the basis functions from (2.9). Now we pick again a single entry out of the double sum only and find the following integral at the bottom of this hierarchy

$$\langle \phi_k \,|\, f_{k,l} \,|\, \phi'_l \rangle = \int f_{k,l}(x)\, \overline{\phi_k(x)} \phi'_l(x) \; . \tag{3.3}$$

We now want to find an efficient way to calculate this integral for all i, j, k and l or, in other words, set up the matrix F. This integral can almost never be solved analytically thus the integral is approximated by high order quadrature.

3.2 Inner products of basis functions

We speak of inhomogeneous inner products if the part in the bra and the part in the ket of (3.3) have different Hagedorn parameters sets Π. In this case, all formulae become much more complicated.

3.2.1 An analytical ansatz to inner products

The inner product of two basis functions which have different sets of parameters denoted by $\Pi_k := \{P_k, Q_k, S_k, p_k, q_k\}$ and $\Pi_l := \{P_l, Q_l, S_l, p_l, q_l\}$ respectively is written as usual as $\left\langle \phi^k \,\middle|\, \phi^l \right\rangle$. This is the expression we want to evaluate now and we can even write down a closed form solution based on induction and the recursion relation for Hermite polynomials. The expression for the ground states ϕ_0 acts as induction base and is given by

$$\left\langle \phi_0^k \,\middle|\, \phi_0^l \right\rangle = \sqrt{\frac{-2i}{Q_2\overline{P_1} - P_2\overline{Q_1}}} \cdot \exp\left(\frac{i}{2\varepsilon^2} \frac{Q_2\overline{Q_1}\left(p_2 - p_1\right)^2 + P_2\overline{P_1}\left(q_2 - q_1\right)^2}{\left(Q_2\overline{P_1} - P_2\overline{Q_1}\right)} \right.$$
$$\left. - \frac{i}{\varepsilon^2} \frac{\left(q_2 - q_1\right)\left(Q_2\overline{P_1}p_2 - P_2\overline{Q_1}p_1\right)}{\left(Q_2\overline{P_1} - P_2\overline{Q_1}\right)} \right). \quad (3.4)$$

For the inner product of higher level functions ϕ_i the whole thing gets much more complicated

$$\left\langle \phi_k^k \,\middle|\, \phi_l^l \right\rangle = \frac{1}{\sqrt{l!k!}} 2^{-\frac{l+k}{2}} \left\langle \phi_0^k \,\middle|\, \phi_0^l \right\rangle \cdot \left(i\overline{P_1}Q_2 - i\overline{Q_1}P_2\right)^{-\frac{l+k}{2}} \cdot$$
$$\sum_{j=0}^{\min(l,k)} \left(\binom{l}{j} \binom{k}{j} j! 4^j \left(i\overline{Q_2P_1} - i\overline{Q_1P_2}\right)^{\frac{k-j}{2}} \left(iQ_2P_1 - iQ_1P_2\right)^{\frac{l-j}{2}} \right.$$
$$\cdot H_{k-j}\left(\frac{1}{\varepsilon^2} \frac{\overline{P_1}\left(q_1 - q_2\right) + \overline{Q_1}\left(p_1 - p_2\right)}{\sqrt{\overline{Q_2P_1} - \overline{Q_1P_2}}\sqrt{\overline{P_1}Q_2 - \overline{Q_1}P_2}} \right)$$
$$\left. \cdot H_{l-j}\left(-\frac{1}{\varepsilon^2} \frac{-P_2\left(q_1 - q_2\right) + Q_2\left(p_1 - p_2\right)}{\sqrt{Q_2P_1 - Q_1P_2}\sqrt{\overline{P_1}Q_2 - \overline{Q_1}P_2}} \right) \right). \quad (3.5)$$

For the proofs of these formulae see reference [10].
Despite we can evaluate the inner product and have a closed form solution for arbitrary wavefunctions, these formulae are unsuitable for numerical calculation. There are several reasons but for example the factorials and binomial coefficients lead to overflow even for relatively small k and l. Further the sum may be numerically unstable. Thus we need to find a better way to perform these calculations.

3.2.2 Inhomogeneous or mixing quadrature rule

In this section we will develop a quadrature rule to evaluate the brakets in (3.5). First we notice that each ϕ which is given by (2.9) is represented through a mathematical expression of the general form

$$C \cdot P^n(\xi) \cdot \exp(\theta) \tag{3.6}$$

consisting of an arbitrary constant $C \in \mathbb{C}$, a polynomial $P^n(\cdot)$ of degree n and an exponential $\exp(\cdot)$. We try a new ansatz for calculating the inner product. Evaluating the braket $\langle \phi^k \mid \phi^l \rangle$ results in a multiplication of two expressions of the form (3.6). The parts in this expression can be grouped by same type

$$\begin{aligned} \langle \phi^k \mid \phi^l \rangle &= \int_{\mathbb{R}} \overline{C_k P_k^n(\xi_k) \exp(\theta_k)} C_l P_l^n(\xi_l) \exp(\theta_l)\, dx \\ &= \int_{\mathbb{R}} \overline{C_k} C_l P_k^n\left(\overline{\xi_k}\right) P_l^n(\xi_l) \exp\left(\overline{\theta_k}\right) \exp(\theta_l)\, dx\,. \end{aligned} \tag{3.7}$$

Let's take a closer look at the integrand of this expression now. With Gauss Hermite quadrature in mind we are especially interested in the exponential parts. They have a general form like

$$\exp(\theta) = \exp\left(s \cdot (x-m)^2 + \cdots\right). \tag{3.8}$$

We concentrate on the exponentials of (3.7) only. We combine them and distribute the complex conjugate onto the variables affected

$$\begin{aligned} &\overline{\exp\left(\frac{i}{2\varepsilon^2} P_k Q_k^{-1} (x-q_k)^2 + \frac{i}{\varepsilon^2} p_k (x-q_k)\right)} \cdot \exp\left(\frac{i}{2\varepsilon^2} P_l Q_l^{-1} (x-q_l)^2 + \frac{i}{\varepsilon^2} p_l (x-q_l)\right) \\ &= \exp\left(\overline{\frac{i}{2\varepsilon^2} P_k Q_k^{-1} (x-q_k)^2 + \frac{i}{\varepsilon^2} p_k (x-q_k)} + \frac{i}{2\varepsilon^2} P_l Q_l^{-1} (x-q_l)^2 + \frac{i}{\varepsilon^2} p_l (x-q_l)\right) \\ &= \exp\left(-\frac{i}{2\varepsilon^2} \overline{P_k} \overline{Q_k^{-1}} (x-q_k)^2 - \frac{i}{\varepsilon^2} p_k (x-q_k) + \frac{i}{2\varepsilon^2} P_l Q_l^{-1} (x-q_l)^2 + \frac{i}{\varepsilon^2} p_l (x-q_l)\right). \end{aligned}$$

For the sake of readability we define the following variables

$$\begin{aligned} r_k &:= P_k Q_k^{-1} \\ r_l &:= P_l Q_l^{-1}\,. \end{aligned} \tag{3.9}$$

Plugging these into the equation above and expanding the squares we get for the exponent

$$\frac{i}{\varepsilon^2} \left(-\frac{1}{2} \underbrace{\left(\overline{r_k} - r_l\right)}_{\alpha+i\beta} x^2 + \underbrace{\left(\overline{r_k} q_k - r_l q_l\right)}_{\gamma+i\delta} x - \underbrace{\frac{1}{2}\left(\overline{r_k} q_k^2 + r_l q_l^2\right) + \left(p_l - p_k\right) x + p_k q_k - p_l q_l}_{\text{junk}} \right)$$

where we introduced two new complex variables $\alpha + i\beta$ and $\gamma + i\delta$ and sorted out some unimportant expressions. We now carry out the multiplication with respect to the real parts of these complex numbers. This yields

$$-\frac{1}{\varepsilon^2}\left(-\frac{\beta}{2}x^2 + i\,(\cdots) + \delta x - i\,(\cdots) + \cdots\right) \tag{3.10}$$

where the dots indicate more junk. To get back to a form along the lines of (3.8) we have to complete the square, first divide out a factor of $-\frac{\beta}{2}$ and then complete by a factor of $\left(\frac{\delta}{\beta}\right)^2$

$$\begin{aligned} &-\frac{1}{\varepsilon^2}\left(x^2 - \frac{2\delta}{\beta}x + \ldots\right)\left(-\frac{\beta}{2}\right) \\ =&-\frac{1}{\varepsilon^2}\left(\left(-\frac{\beta}{2}\right)\left(x - \frac{\delta}{\beta}\right)^2 + \frac{\delta^2}{2\beta}\right). \end{aligned} \tag{3.11}$$

From this last expression we get the quadrature rule components $s := Q_0$ and $m := q_0$ of expression (3.8) as

$$\begin{aligned} q_0 &:= \frac{\delta}{\beta} = \frac{\Im\left(\overline{r_k}q_k - r_l q_l\right)}{\Im\left(\overline{r_k} - r_l\right)} \\ Q_0 &:= -\frac{\beta}{2} = -\frac{\Im\left(\overline{r_k} - r_l\right)}{2} \\ Q_S &:= \frac{1}{\sqrt{Q_0}}. \end{aligned} \tag{3.12}$$

Now we transform the nodes γ_i according to the weighted position mean q_0 and the parameter Q_S which changes the spread of the nodes. This yields

$$\gamma_i' := q_0 + \varepsilon \cdot Q_S \cdot \gamma_i \tag{3.13}$$

for the mixing quadrature nodes which are located in the space around where the product of ϕ_k and ϕ_l is maximal. A procedure that calculates q_0, Q_S and the adapted quadrature nodes is given by algorithm 3.

Algorithm 3 Mixing two sets Π_r and Π_c of Hagedorn parameters

Require: Two sets Π_r and Π_c of Hagedorn parameters
Require: A quadrature rule (γ_i, ω_i)
  // Apply the mixing formula to the parameters
  $r_r := \frac{P_r}{Q_r}$
  $r_c := \frac{P_c}{Q_c}$
  $q_0 := \frac{\Im(\overline{r_r}q_r - r_c q_c)}{\Im(\overline{r_r} - r_c)}$
  $Q_0 := -\frac{\Im(\overline{r_r} - r_c)}{2}$
  $Q_S := \frac{1}{\sqrt{Q_0}}$
  // And shift the quadrature nodes
  $\gamma' := q_0 + \varepsilon Q_S \gamma$
  return q_0 and Q_S and γ'

We get back to the homogeneous case if we choose the sets Π_k and Π_l of Hagedorn parameters identical.

Note that we get issues if at any time it happens that

$$\Im\left(\frac{\overline{P_k}}{\overline{Q_k}} - \frac{P_l}{Q_l}\right) > 0\,. \tag{3.14}$$

3.2.3 Homogeneous quadrature rule

In this section we reduce the mixing quadrature rule of the last section to the homogeneous case where everything becomes much simpler. We will start from the assumption that the mixing quadrature rule contains the homogeneous one as a special case. (This is not just speculation but can be shown by direct calculation analogous to what we did in the last section.) Suppose for this section that both ϕ_i in (3.3) belong to the same basis and have an identical parameter set Π hence $\Pi_k = \Pi_l$ or at least $P_k = P_l$, $Q_k = Q_l$ and $p_k = p_l$, $q_k = q_l$. With this assumption we simplify the quadrature formulae (3.12) and (3.13) to the homogeneous case.
Let's start with simplification of the Q_0 parameter

$$\begin{aligned} Q_0 &:= -\frac{\Im\left(\overline{r_k} - r_l\right)}{2} = -\frac{1}{2}\Im\left(\overline{r_k} - r_k\right) \\ &= -\frac{1}{2}\left(-\frac{2}{\overline{Q}Q}\right) \\ &= \frac{1}{|Q|^2}\,. \end{aligned} \tag{3.15}$$

The parameter Q_S is then as trivial as

$$Q_S := \frac{1}{\sqrt{Q_0}} = \frac{1}{\sqrt{\frac{1}{|Q|^2}}} = |Q|\,. \tag{3.16}$$

In the last case we have for the weighted position

$$\begin{aligned} q_0 &:= \frac{\Im\left(\overline{r_k}q_k - r_l q_l\right)}{\Im\left(\overline{r_k} - r_l\right)} = \frac{\Im\left(\overline{r_k}q_k - r_k q_k\right)}{\Im\left(\overline{r_k} - r_k\right)} \\ &= \frac{q\Im\left(\overline{r_k} - r_k\right)}{-\frac{2}{\overline{Q}Q}} = \frac{-\frac{2}{\overline{Q}Q}q}{-\frac{2}{\overline{Q}Q}} \\ &= q \end{aligned} \tag{3.17}$$

which should be clear also without computation. For the transformed quadrature nodes we can write

$$\gamma_i' := q + \varepsilon \cdot |Q| \cdot \gamma_i \tag{3.18}$$

which was shown to work in real simulations for homogeneous wavepackets.

3.3 Quadrature rules applied

After we discussed in details the transformation of the quadrature nodes in the last section we now look at the final quadrature rule. It's not really difficult, but it is good to write down all the details at least once.
Assume we have the transformed quadrature nodes γ'_i given by (3.13). We now carry out the quadrature for resolving equation (3.3)

$$\langle \phi_k \mid f \mid \phi'_l \rangle \approx \varepsilon \cdot Q_S \cdot \sum_{r=0}^{R} \overline{\phi_k(\gamma'_r)} \cdot f(\gamma'_r) \cdot \phi'_l(\gamma'_r) \cdot \omega_r \tag{3.19}$$

where the two ϕ in general belong to the different families. But if they really have the same parameter sets Π then we can simplify this formula slightly. The trick is that we omit a prefactor of $\frac{1}{\sqrt{Q}}$ when evaluating $\phi(\gamma'_r)$. Because we have two times this evaluation, both prefactors accumulate to $\frac{1}{|Q|}$. In the homogeneous case we have $Q_S = |Q|$ hence the Q_S outside the sum cancels nicely with this prefactors. And the above formula becomes

$$\langle \phi_k \mid f \mid \phi_l \rangle \approx \varepsilon \cdot \sum_{r=0}^{R} \overline{\phi_k(\gamma'_r)} \cdot f(\gamma'_r) \cdot \phi_l(\gamma'_r) \cdot \omega_r \,. \tag{3.20}$$

Chapter 4

Motivation and procedure for spawning wavepackets

In this chapter we will review the important parts from the paper about tunneling dynamics and spawning [5]. This is necessary because all the further chapters about spawning of wavepackets in the non-adiabatic case build upon and generalize the basic ideas developed there. Also we will work out the basic mathematical procedure for spawning new wavepackets.

4.1 The tunneling problem

In tunneling dynamics we consider a potential shaped like a steep hill. The so called Eckart potential, defined as $V(x) := \frac{v_0}{\cosh^2(ax)}$ where v_0 is the potential energy at the maximum and a is another constant, serves as a good example of such a potential. This potential is shown in figure 4.1 for later reference.
Suppose now that we have a particle coming from one side and moving towards the potential. In the classical world, the particle either crosses over the hill or gets reflected depending on its momentum only. In the quantum world however, the wavefunction always splits up in two parts of which one will be reflected and the other transmitted. These two parts usually have very different amplitudes and will drift apart more and more while their interaction rapidly becomes negligible as they are separated by the potential hill.

4.1.1 Motivation for spawning

For our algorithm based on wavepackets as defined in chapter 2 this split becomes problematical. The reason is that we need very high frequencies to represent the tunneled part of our wavepacket. In most settings the energy of the wavepacket is strictly smaller than the peak value of the potential. This implies that during the propagation of the parameter set Π the parameter q which centers the basis functions ϕ_k in position space gets reflected at the potential. Therefore we get a good basis for representing the reflected part (with only very few basis functions). This basis obviously is unsuitable for representing the tunneled part and indeed we need many basis functions in the high frequency domain. (These problems wouldn't matter if we worked with a infinite basis.)

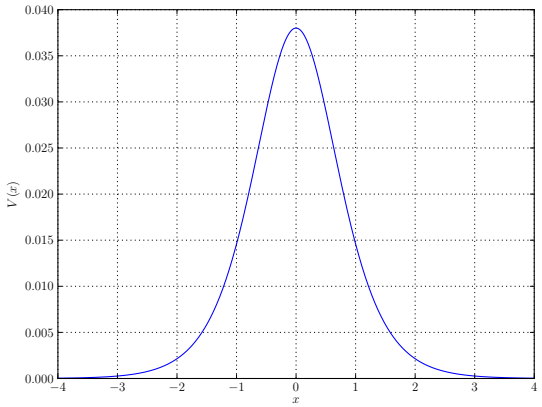


Figure 4.1: The Eckart potential with the parameters set to $v_0 = 0.038008$ and $a = 1.05836$ (same values as in ref.[5]).

In figure 4.2 the output of a tunneling simulation using wavepackets is shown at late time. We clearly see the issue explained above. A basis of size about 20 would be sufficient for representing the reflected part accurately. But for the transmitted part we need up to about 250 basis functions. And the problem gets worse with time (slowly) requiring more and more basis functions. Another important point to notice is that there is a range of coefficients with negligible values in the middle. In the example this range extends roughly from 40 to 110. From this observation it's self-evident that breaking up the packet into two independent parts would be a good idea. This then would also allow us to represent each packet with a much smaller basis. And in the case of tunneling we could even neglect the interaction between the two packets without loosing much information.

4.2 Spawning a new packet

With the motivation from the last section we will now investigate the steps that have to be taken for spawning a new wavepacket. A first step is to find a new, suitable basis. Remember that for fixed ε the set of functions $\{\phi_k\}_{k=0}^{K}$ gives a complete (but truncated) basis of the space L^2 and this is all we need for expanding our wavepacket into a linear combination. Finding this basis is depicted in great detail in section 4.3. Although the mathematics behind this process is mostly trivial, there are plenty of opportunities to make mistakes. Given this new basis we need to find a method for transferring (a part of) the wavepacket to this new basis. We describe two basically different methods in section 4.4.

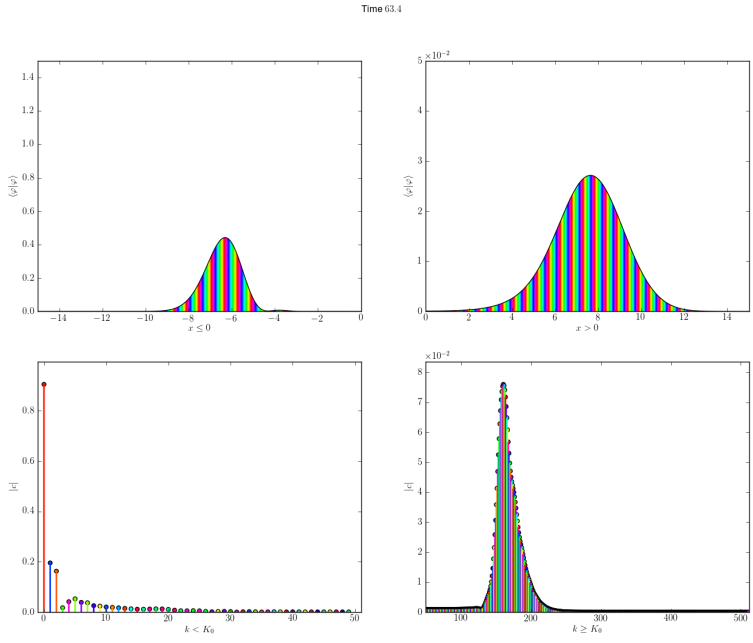


Figure 4.2: This figure shows the output of a tunneling simulation at late time. The upper two panels show $\langle \Phi \,|\, \Phi \rangle$ for the wavepacket $|\Phi\rangle$. The lower two panels show the absolute value $|c_k|$ of the coefficients. The colors represent the complex phase according to the convention from [13]. Please notice the different scales! For the upper panels the x axis is split at 0 and the y axes have a sensible scale. In the lower left panel we have the coefficients for $k < 50$ while the values for $k \geq 50$ are in the right panel where again the y axis is scaled appropriately.

4.3 Parameter estimation

4.3.1 Fragments

In the remainder of this chapter we will use a so called *fragment* $|w\rangle$ as a kind of a placeholder. The definition is very similar to the one of a full scalar wavepacket $|\Phi\rangle$ but more flexible for our purpose of lying out the basic formalism for spawning wavepackets. So we define our fragment as

$$|w\rangle := \sum_{k=\alpha}^{\beta} c_k \phi_k \tag{4.1}$$

with $\alpha, \beta \in \mathbb{N}_0$ and $\beta \geq \alpha$. Basically this is just a handy abbreviation for an arbitrary linear combination of several basis functions. If we demand that $\alpha, \beta \in [0, K-1]$ it becomes clear why we call this a fragment, compared to the full (scalar) wavepacket in (2.12). And in the case of $\alpha = 0$ and $\beta = K-1$ we recover the full packet by

$$|\Phi\rangle = \exp\left(\frac{iS}{\varepsilon^2}\right)|w\rangle \ . \tag{4.2}$$

In principle the fragment could be sparse but this is mathematically equivalent to saying that $\exists k \in [\alpha, \ldots, \beta] : c_k = 0$ and thus we do not care further.
For the moment it does not matter what w precisely is, all we need to know about it is given by equation (4.1). Later w may be a fully normalized wavepacket $\Phi = \sum_{k=0}^{K-1} c_k \phi_k$, only a part of a wavepacket $\Phi' = \sum_{k=\alpha}^{\beta} c_k \phi_k$ with $\alpha \geq 0$ and $\beta \leq K-1$ or a single component $\Phi_i = \sum_{k=0}^{K-1} c_k^i \phi_k^i$ of a homogeneous or inhomogeneous vector valued wavepacket Ψ depending on the context where the spawning technique is to be applied. In the example shown in figure (4.2) motivating this chapter, w would be what is shown in the right column.
As said above the first step is to find a new (in some measure *better*) basis $\{\tilde{\phi}_k\}_{k=0}^{\infty}$ of the space L^2 for representing $|w\rangle$. The basis functions $\tilde{\phi}_k$ are fully characterized by the parameter set $\tilde{\Pi} := \{\tilde{P}, \tilde{Q}, \tilde{p}, \tilde{q}\}$ so all we have to do is estimate these four values. By the way, we denote all quantities in the new bases with a tilde. The new position $\tilde{q}$ and new momentum $\tilde{p}$ are both real numbers and therefore easy. So we will handle these two first to get a feeling for the procedure.

4.3.2 Position $\tilde{q}$ and Momentum $\tilde{p}$

The parameters $\tilde{q}$ and $\tilde{p}$ can be interpreted as the average position and momentum. For this reason we compute expectation values of the position and momentum operators

$$\tilde{q} := \frac{\langle w \,|\, x \,|\, w \rangle}{\langle w \,|\, w \rangle} \tag{4.3}$$

$$\tilde{p} := \frac{\langle w \,|\, y \,|\, w \rangle}{\langle w \,|\, w \rangle} \tag{4.4}$$

where we divide by $\langle w \,|\, w \rangle$ since the fragment is not normalized in general. We can get an explicit expression for position operator x and the momentum

operator y by solving the linear system consisting of the definitions of both ladder operators as shown in (2.4) and get

$$\begin{aligned} x :=& \sqrt{\frac{\varepsilon^2}{2}} \left(Q\mathcal{R} + \overline{Q}\mathcal{L} \right) + q \\ y :=& \sqrt{\frac{\varepsilon^2}{2}} \left(P\mathcal{R} + \overline{P}\mathcal{L} \right) + p \,. \end{aligned} \tag{4.5}$$

Notice the symmetry in the two operators. We can transform x into y by the two replacements $Q \to P$ and $q \to p$. This will halve the work when calculating properties of these operators.
Starting with a fragment $|w\rangle$ and an arbitrary operator O and expanding the linear combination we arrive at a double sum over brakets including basis functions ϕ_k only

$$\begin{aligned} \langle w \,|\, O \,|\, w \rangle :=& \left\langle \sum_{k=\alpha}^{\beta} c_k \phi_k \,\middle|\, O \,\middle|\, \sum_{l=\alpha}^{\beta} c_l \phi_l \right\rangle \\ =& \sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l \, \langle \phi_k \,|\, O \,|\, \phi_l \rangle \ . \end{aligned} \tag{4.6}$$

Now we have to see what happens with these simpler brakets. The full calculation for the position operator x works as follows where we use sesquilinearity and the properties of the ladder operators

$$\begin{aligned} \langle \phi_k \,|\, x \,|\, \phi_l \rangle :=& \left\langle \phi_k \,\middle|\, \sqrt{\frac{\varepsilon^2}{2}} \left(Q\mathcal{R} + \overline{Q}\mathcal{L} \right) + q \,\middle|\, \phi_l \right\rangle \\ =& \left\langle \phi_k \,\middle|\, \sqrt{\frac{\varepsilon^2}{2}} Q\mathcal{R} \,\middle|\, \phi_l \right\rangle + \left\langle \phi_k \,\middle|\, \sqrt{\frac{\varepsilon^2}{2}} \overline{Q}\mathcal{L} \,\middle|\, \phi_l \right\rangle + \langle \phi_k \,|\, q \,|\, \phi_l \rangle \\ =& \sqrt{\frac{\varepsilon^2}{2}} Q \, \langle \phi_k \,|\, \mathcal{R} \,|\, \phi_l \rangle + \sqrt{\frac{\varepsilon^2}{2}} \overline{Q} \, \langle \phi_k \,|\, \mathcal{L} \,|\, \phi_l \rangle + q \, \langle \phi_k \,|\, \phi_l \rangle \\ =& \sqrt{\frac{\varepsilon^2}{2}} Q \left\langle \phi_k \,\middle|\, \sqrt{l+1} \phi_{l+1} \right\rangle + \sqrt{\frac{\varepsilon^2}{2}} \overline{Q} \left\langle \phi_k \,\middle|\, \sqrt{l} \phi_{l-1} \right\rangle + q \, \langle \phi_k \,|\, \phi_l \rangle \end{aligned}$$

and finally we get

$$\langle \phi_k \,|\, x \,|\, \phi_l \rangle = \sqrt{\frac{\varepsilon^2}{2}} \left(Q\sqrt{l+1} \, \langle \phi_k \,|\, \phi_{l+1} \rangle + \overline{Q}\sqrt{l} \, \langle \phi_k \,|\, \phi_{l-1} \rangle \right) + q \, \langle \phi_k \,|\, \phi_l \rangle \ . \tag{4.7}$$

Applying the symmetry mentioned above we get a similar formula for the momentum operator

$$\langle \phi_k \,|\, y \,|\, \phi_l \rangle = \sqrt{\frac{\varepsilon^2}{2}} \left(P\sqrt{l+1} \, \langle \phi_k \,|\, \phi_{l+1} \rangle + \overline{P}\sqrt{l} \, \langle \phi_k \,|\, \phi_{l-1} \rangle \right) + p \, \langle \phi_k \,|\, \phi_l \rangle \ . \tag{4.8}$$

Using the orthonormality of the basis function we can write

$$\langle \phi_k \,|\, x \,|\, \phi_l \rangle = \sqrt{\frac{\varepsilon^2}{2}} \left(Q\sqrt{l+1}\delta_{k,l+1} + \overline{Q}\sqrt{l}\delta_{k,l-1} \right) + q\delta_{k,l} \tag{4.9}$$

$$\langle \phi_k \,|\, y \,|\, \phi_l \rangle = \sqrt{\frac{\varepsilon^2}{2}} \left(P\sqrt{l+1}\delta_{k,l+1} + \overline{P}\sqrt{l}\delta_{k,l-1} \right) + p\delta_{k,l}\,. \tag{4.10}$$

At this point we can substitute back these expressions into the double sum from (4.6) which then gives

$$\begin{aligned}
\langle w \,|\, x \,|\, w \rangle &= \sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l \left(\sqrt{\frac{\varepsilon^2}{2}} \left(Q\sqrt{l+1}\delta_{k,l+1} + \overline{Q}\sqrt{l}\delta_{k,l-1} \right) + q\delta_{k,l} \right) \\
&= \sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l \sqrt{\frac{\varepsilon^2}{2}} Q\sqrt{l+1}\delta_{k,l+1} + \sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l \sqrt{\frac{\varepsilon^2}{2}} \overline{Q}\sqrt{l}\delta_{k,l-1} \\
&\quad + \sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l q\delta_{k,l}\,.
\end{aligned}$$

Because of the Kronecker deltae each of these three double sums can be reduced to a single sum only. For the first one we have $\delta_{k,l+1} = 1 \iff l = k-1$ thus

$$\sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l \sqrt{\frac{\varepsilon^2}{2}} Q\sqrt{l+1}\delta_{k,l+1} = \sqrt{\frac{\varepsilon^2}{2}} Q \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1}\sqrt{k}\,. \tag{4.11}$$

See also figure 4.3 for a better overview over these nasty index transformations. For the second sum the relation is $\delta_{k,l-1} = 1 \iff l = k+1$ and therefore

$$\sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l \sqrt{\frac{\varepsilon^2}{2}} \overline{Q}\sqrt{l}\delta_{k,l-1} = \sqrt{\frac{\varepsilon^2}{2}} \overline{Q} \sum_{k=\alpha}^{\beta-1} \overline{c_k} c_{k+1}\sqrt{k+1}\,. \tag{4.12}$$

The third one is trivial as $\delta_{k,l} = 1 \iff l = k$ and we find

$$\sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l q\delta_{k,l} = q \sum_{k=\alpha}^{\beta} \overline{c_k} c_k\,. \tag{4.13}$$

Notice that this sum is nothing else than the norm of w. We can simplify the expression for $\langle w \,|\, x \,|\, w \rangle$ further. For this we first shift the summation index of the right hand side of (4.12) up by one. (Equivalently we could also shift down the index of (4.11) by one.) Mathematically this means that $k' = k+1$ which gives $k = k'-1$ and we find that

$$\sqrt{\frac{\varepsilon^2}{2}} \overline{Q} \sum_{k=\alpha}^{\beta-1} \overline{c_k} c_{k+1}\sqrt{k+1} = \sqrt{\frac{\varepsilon^2}{2}} \overline{Q} \sum_{k'=\alpha+1}^{\beta} \overline{c_{k'-1}} c_{k'}\sqrt{k'} \tag{4.14}$$

where we wrote the primes once for clarity and drop them from now on. Recognizing now that (4.11) and (4.14) are complex conjugates of each other we can combine them and write for the whole expression

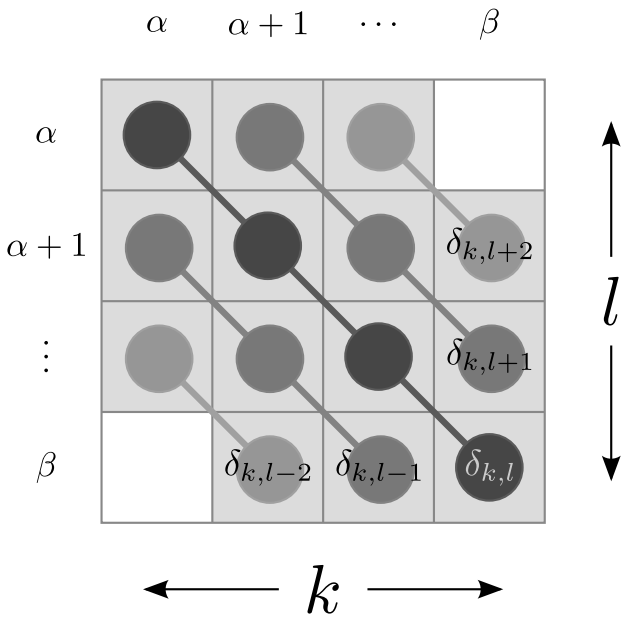


Figure 4.3: Symbolic view on the double sum $\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}$. The diagonal lines show which elements remain in a single sum $\sum_k$ after expanding the corresponding Kronecker delta.

$$\begin{aligned}\langle w \,|\, x \,|\, w\rangle &= \sqrt{\frac{\varepsilon^2}{2}} Q \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1} \sqrt{k} + \sqrt{\frac{\varepsilon^2}{2}} \overline{Q} \sum_{k=\alpha+1}^{\beta} \overline{c_{k-1}} c_k \sqrt{k} + q \sum_{k=\alpha}^{\beta} \overline{c_k} c_k \\ &= \sqrt{2\varepsilon^2} \Re\left(Q \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1} \sqrt{k} \right) + q \sum_{k=\alpha}^{\beta} \overline{c_k} c_k \,. \end{aligned} \tag{4.15}$$

The very same procedure can be applied for the momentum operator y too, but of course we take the shortcut by symmetry and get

$$\langle w \,|\, y \,|\, w\rangle = \sqrt{2\varepsilon^2} \Re\left(P \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1} \sqrt{k} \right) + p \sum_{k=\alpha}^{\beta} \overline{c_k} c_k \,. \tag{4.16}$$

If we now remember that the fragment may not be normalized we find the final formulae for the expected position and momentum of w

$$\tilde{q} := \frac{\langle w \,|\, x \,|\, w\rangle}{\langle w \,|\, w\rangle} = \frac{\sqrt{2\varepsilon^2}}{\sum_{k=\alpha}^{\beta} \overline{c_k} c_k} \Re\left(Q \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1} \sqrt{k} \right) + q \tag{4.17}$$

$$\tilde{p} := \frac{\langle w \,|\, y \,|\, w\rangle}{\langle w \,|\, w\rangle} = \frac{\sqrt{2\varepsilon^2}}{\sum_{k=\alpha}^{\beta} \overline{c_k} c_k} \Re\left(P \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1} \sqrt{k} \right) + p \,. \tag{4.18}$$

4.3.3 Estimating second central moments

In order to estimate the parameters $\tilde{Q}$ and $\tilde{P}$ of a fragment w in a first step we have to compute the following two expectation values

$$\frac{\left\langle w \,\middle|\, (x-\tilde{q})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} \qquad \text{and} \qquad \frac{\left\langle w \,\middle|\, (x-\tilde{p})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} \tag{4.19}$$

which is in principle straight forward but very tedious. The reason why we compute these quantities will become clear later but we essentially try to estimate second central moments. We now show the whole procedure for the first braket and then use the symmetry argument again to get the second one too. We start with expanding the operator $(x-\tilde{q})^2$ and break the expression into parts

$$\begin{aligned} \frac{\left\langle w \,\middle|\, (x-\tilde{q})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} &= \frac{\langle w \,|\, x^2 - 2\tilde{q}x + \tilde{q}^2 \,|\, w \rangle}{\langle w \,|\, w \rangle} \\ &= \frac{\langle w \,|\, x^2 \,|\, w \rangle}{\langle w \,|\, w \rangle} - 2\tilde{q}\frac{\langle w \,|\, x \,|\, w \rangle}{\langle w \,|\, w \rangle} + \tilde{q}^2 \,. \end{aligned} \tag{4.20}$$

In the third term, the norms of w cancel and we rediscover (4.17) in the second term. Thus all that is left over is

$$\frac{\left\langle w \,\middle|\, (x-\tilde{q})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} = \frac{\langle w \,|\, x^2 \,|\, w \rangle}{\langle w \,|\, w \rangle} - \tilde{q}^2 \,. \tag{4.21}$$

Now we will concentrate on the first term and we have to struggle quite a bit to compute it's numerator. As usual we decompose this braket by using the sesquilinearity and write

$$\begin{aligned} \langle w \,|\, x^2 \,|\, w \rangle &= \left\langle \sum_{k=\alpha}^{\beta} c_k \phi_k \,\middle|\, x^2 \,\middle|\, \sum_{l=\alpha}^{\beta} c_l \phi_l \right\rangle \\ &= \sum_{k=\alpha}^{\beta} \sum_{l=\alpha}^{\beta} \overline{c_k} c_l \left\langle \phi_k \,\middle|\, x^2 \,\middle|\, \phi_l \right\rangle \,. \end{aligned} \tag{4.22}$$

This reduced the problem to brakets over basis functions. At that point we need an explicit version of the operator x^2 in terms of raising and lowering operators which can be obtained as follows

$$\begin{aligned} x^2 &= \left(\sqrt{\frac{\varepsilon^2}{2}} \left(Q\mathcal{R} + \overline{Q}\mathcal{L} \right) + q \right) \left(\sqrt{\frac{\varepsilon^2}{2}} \left(Q\mathcal{R} + \overline{Q}\mathcal{L} \right) + q \right) \\ &= \left(\theta Q\mathcal{R} + \theta\overline{Q}\mathcal{L} + q \right) \left(\theta Q\mathcal{R} + \theta\overline{Q}\mathcal{L} + q \right) \\ &= \theta^2 Q\mathcal{R}Q\mathcal{R} + \theta^2 Q\mathcal{R}\overline{Q}\mathcal{L} + \theta Q\mathcal{R}q + \theta^2\overline{Q}\mathcal{L}Q\mathcal{R} + \theta^2\overline{Q}\mathcal{L}\overline{Q}\mathcal{L} + \theta\overline{Q}\mathcal{L}q + \theta qQ\mathcal{R} + \theta q\overline{Q}\mathcal{L} + q^2 \\ &= \theta^2 Q^2\mathcal{R}^2 + \theta^2\overline{Q}^2\mathcal{L}^2 + \theta^2 Q\overline{Q}\mathcal{R}\mathcal{L} + \theta^2\overline{Q}Q\mathcal{L}\mathcal{R} + 2\theta qQ\mathcal{R} + 2\theta q\overline{Q}\mathcal{L} + q^2 \end{aligned} \tag{4.23}$$

where for simplicity we defined $\theta := \sqrt{\frac{\varepsilon^2}{2}}$. We can now use the sesquilinearity of the inner product and split (4.22) once more

$$\begin{aligned}\langle \phi_k \,|\, x^2 \,|\, \phi_l \rangle = &\langle \phi_k \,|\, \theta^2 Q^2 \mathcal{R}^2 \,|\, \phi_l \rangle + \left\langle \phi_k \,\middle|\, \theta^2 \overline{Q}^2 \mathcal{L}^2 \,\middle|\, \phi_l \right\rangle \\ &+ \langle \phi_k \,|\, \theta^2 Q\overline{Q}\mathcal{R}\mathcal{L} \,|\, \phi_l \rangle + \langle \phi_k \,|\, \theta^2 \overline{Q} Q \mathcal{L}\mathcal{R} \,|\, \phi_l \rangle \\ &+ \langle \phi_k \,|\, 2\theta q Q\mathcal{R} \,|\, \phi_l \rangle + \langle \phi_k \,|\, 2\theta q \overline{Q}\mathcal{L} \,|\, \phi_l \rangle + \langle \phi_k \,|\, q^2 \,|\, \phi_l \rangle\end{aligned}$$

resulting in seven parts which we will work out one after the other now. This is a very boring task but necessary to justify the final relatively simple formula.

$$\begin{aligned}\langle \phi_k \,|\, \theta^2 Q^2 \mathcal{R}^2 \,|\, \phi_l \rangle &= \theta^2 Q^2 \langle \phi_k \,|\, \mathcal{R}^2 \,|\, \phi_l \rangle \\ &= \theta^2 Q^2 \left\langle \phi_k \,\middle|\, \mathcal{R} \,\middle|\, \sqrt{l+1}\phi_{l+1} \right\rangle \\ &= \theta^2 Q^2 \sqrt{l+1} \left\langle \phi_k \,\middle|\, \sqrt{l+2}\phi_{l+2} \right\rangle \\ &= \theta^2 Q^2 \sqrt{l+1}\sqrt{l+2} \langle \phi_k \,|\, \phi_{l+2} \rangle = \theta^2 Q^2 \sqrt{l+1}\sqrt{l+2}\delta_{k,l+2}\end{aligned}$$

$$\begin{aligned}\left\langle \phi_k \,\middle|\, \theta^2 \overline{Q}^2 \mathcal{L}^2 \,\middle|\, \phi_l \right\rangle &= \theta^2 \overline{Q}^2 \langle \phi_k \,|\, \mathcal{L}^2 \,|\, \phi_l \rangle \\ &= \theta^2 \overline{Q}^2 \left\langle \phi_k \,\middle|\, \mathcal{L} \,\middle|\, \sqrt{l}\phi_{l-1} \right\rangle \\ &= \theta^2 \overline{Q}^2 \sqrt{l} \left\langle \phi_k \,\middle|\, \sqrt{l-1}\phi_{l-2} \right\rangle \\ &= \theta^2 \overline{Q}^2 \sqrt{l}\sqrt{l-1} \langle \phi_k \,|\, \phi_{l-2} \rangle = \theta^2 \overline{Q}^2 \sqrt{l}\sqrt{l-1}\delta_{k,l-2}\end{aligned}$$

$$\begin{aligned}\langle \phi_k \,|\, \theta^2 Q\overline{Q}\mathcal{R}\mathcal{L} \,|\, \phi_l \rangle &= \theta^2 Q\overline{Q} \langle \phi_k \,|\, \mathcal{R}\mathcal{L} \,|\, \phi_l \rangle \\ &= \theta^2 Q\overline{Q} \left\langle \phi_k \,\middle|\, \mathcal{R} \,\middle|\, \sqrt{l}\phi_{l-1} \right\rangle \\ &= \theta^2 Q\overline{Q}\sqrt{l} \left\langle \phi_k \,\middle|\, \sqrt{l}\phi_l \right\rangle \\ &= \theta^2 Q\overline{Q} l \langle \phi_k \,|\, \phi_l \rangle = \theta^2 Q\overline{Q} l \delta_{k,l}\end{aligned}$$

$$\begin{aligned}\langle \phi_k \,|\, \theta^2 \overline{Q} Q \mathcal{L}\mathcal{R} \,|\, \phi_l \rangle &= \theta^2 \overline{Q} Q \langle \phi_k \,|\, \mathcal{L}\mathcal{R} \,|\, \phi_l \rangle \\ &= \theta^2 \overline{Q} Q \left\langle \phi_k \,\middle|\, \mathcal{R} \,\middle|\, \sqrt{l+1}\phi_{l+1} \right\rangle \\ &= \theta^2 \overline{Q} Q \sqrt{l+1} \left\langle \phi_k \,\middle|\, \sqrt{l+1}\phi_l \right\rangle \\ &= \theta^2 \overline{Q} Q (l+1) \langle \phi_k \,|\, \phi_l \rangle = \theta^2 \overline{Q} Q (l+1) \delta_{k,l}\end{aligned}$$

$$\begin{aligned}\langle \phi_k \,|\, 2\theta q Q\mathcal{R} \,|\, \phi_l \rangle &= 2\theta q Q \langle \phi_k \,|\, \mathcal{R} \,|\, \phi_l \rangle \\ &= 2\theta q Q \left\langle \phi_k \,\middle|\, \sqrt{l+1}\phi_{l+1} \right\rangle \\ &= 2\theta q Q \sqrt{l+1} \langle \phi_k \,|\, \phi_{l+1} \rangle = 2\theta q Q \sqrt{l+1}\delta_{k,l+1}\end{aligned}$$

$$\begin{aligned}\left\langle \phi_k \,\middle|\, 2\theta q\overline{Q}\mathcal{L} \,\middle|\, \phi_l \right\rangle &= 2\theta q\overline{Q} \left\langle \phi_k \,|\, \mathcal{L} \,|\, \phi_l \right\rangle \\ &= 2\theta q\overline{Q} \left\langle \phi_k \,\middle|\, \sqrt{l}\phi_{l-1} \right\rangle \\ &= 2\theta q\overline{Q}\sqrt{l} \left\langle \phi_k \,|\, \phi_{l-1} \right\rangle = 2\theta q\overline{Q}\sqrt{l}\delta_{k,l-1}\end{aligned}$$

$$\left\langle \phi_k \,\middle|\, q^2 \,\middle|\, \phi_l \right\rangle = q^2 \left\langle \phi_k \,|\, \phi_l \right\rangle = q^2\delta_{k,l}$$

Each time we used the properties of the ladder operators and the orthonormality of the basis functions. With all these parts we are ready to take the pieces and rebuild the original expression in bottom-up direction. The formula (4.22) now becomes

$$\begin{aligned}\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l \left\langle \phi_k \,\middle|\, x^2 \,\middle|\, \phi_l \right\rangle = &\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l\theta^2Q^2\sqrt{l+1}\sqrt{l+2}\delta_{k,l+2} + \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l\theta^2\overline{Q}^2\sqrt{l}\sqrt{l-1}\delta_{k,l-2} \\ &+ \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l\theta^2Q\overline{Q}l\delta_{k,l} + \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l\theta^2\overline{Q}Q\left(l+1\right)\delta_{k,l} \\ &+ \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l2\theta qQ\sqrt{l+1}\delta_{k,l+1} + \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l2\theta q\overline{Q}\sqrt{l}\delta_{k,l-1} \\ &+ \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_lq^2\delta_{k,l}\,.\end{aligned} \tag{4.24}$$

Analogously to the last section all double sums collapse due to the Kronecker deltae. The only tricky part is to get the summation limits right. It might help to keep figure 4.3 in mind. Starting with the second off-diagonal terms (sums 1 and 2 in the above expression) we have that $\delta_{k,l+2} = 1 \iff l = k-2$ and thus

$$\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l\theta^2Q^2\sqrt{l+1}\sqrt{l+2}\delta_{k,l+2} = \theta^2Q^2\sum_{k=\alpha+2}^{\beta}\overline{c_k}c_{k-2}\sqrt{k-1}\sqrt{k} \tag{4.25}$$

and $\delta_{k,l-2} = 1 \iff l = k+2$ causes

$$\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l\theta^2\overline{Q}^2\sqrt{l}\sqrt{l-1}\delta_{k,l-2} = \theta^2\overline{Q}^2\sum_{k=\alpha}^{\beta-2}\overline{c_k}c_{k+2}\sqrt{k+1}\sqrt{k+2}\,. \tag{4.26}$$

With the first off-diagonal terms (sums 5 and 6 from above) we do the same. Because $\delta_{k,l+1} = 1 \iff l = k-1$ it holds that

$$\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta}\overline{c_k}c_l2\theta qQ\sqrt{l+1}\delta_{k,l+1} = 2\theta qQ\sum_{k=\alpha+1}^{\beta}\overline{c_k}c_{k-1}\sqrt{k} \tag{4.27}$$

and with $\delta_{k,l-1} = 1 \iff l = k+1$ we get

$$\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta} \overline{c_k} c_l 2\theta q \overline{Q} \sqrt{l} \delta_{k,l-1} = 2\theta q \overline{Q} \sum_{k=\alpha}^{\beta-1} \overline{c_k} c_{k+1} \sqrt{k+1}\,. \tag{4.28}$$

Finally for the diagonal terms (sums 3,4 and 7 from (4.24)) we have trivially $\delta_{k,l} = 1 \iff k = l$ giving

$$\begin{aligned} \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta} \overline{c_k} c_l \theta^2 Q \overline{Q} l \delta_{k,l} &= \theta^2 Q \overline{Q} \sum_{k=\alpha}^{\beta} \overline{c_k} c_k k \\ \sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta} \overline{c_k} c_l \theta^2 \overline{Q} Q \,(l+1)\, \delta_{k,l} &= \theta^2 \overline{Q} Q \sum_{k=\alpha}^{\beta} \overline{c_k} c_k \,(k+1) \end{aligned} \tag{4.29}$$

and

$$\sum_{k=\alpha}^{\beta}\sum_{l=\alpha}^{\beta} \overline{c_k} c_l q^2 \delta_{k,l} = q^2 \sum_{k=\alpha}^{\beta} \overline{c_k} c_k\,. \tag{4.30}$$

The next step to proceed with is shifting the summation indices to a common base. For simplicity we choose to shift down all sums to the starting point α. The sum in (4.27) can be shifted by the index transformation $k' = k-1$ and hence $k = k'+1$ which when applied yields

$$2\theta q Q \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1} \sqrt{k} = 2\theta q Q \sum_{k'=\alpha}^{\beta-1} \overline{c_{k'+1}} c_{k'} \sqrt{k'+1}\,. \tag{4.31}$$

In the same fashion and be the transformation $k' = k-2$ and hence $k = k'+2$ we get for (4.25)

$$\theta^2 Q^2 \sum_{k=\alpha+2}^{\beta} \overline{c_k} c_{k-2} \sqrt{k-1}\sqrt{k} = \theta^2 Q^2 \sum_{k'=\alpha}^{\beta-2} \overline{c_{k'+2}} c_{k'} \sqrt{k'+1}\sqrt{k'+2}\,. \tag{4.32}$$

Now that all summation indices are compatible we can combine some of these seven sums and end up with less terms. The trick is again to recognize complex conjugate pairs. The equations (4.32) and (4.26) are such a pair and we write

$$\begin{aligned} \theta^2 Q^2 \sum_{k=\alpha}^{\beta-2} \overline{c_{k+2}} c_k \sqrt{k+1}\sqrt{k+2} \;+\; \theta^2 \overline{Q}^2 \sum_{k=\alpha}^{\beta-2} \overline{c_k} c_{k+2} \sqrt{k+1}\sqrt{k+2} \\ = 2\theta^2 \Re\left(Q^2 \sum_{k=\alpha}^{\beta-2} \overline{c_{k+2}} c_k \sqrt{k+1}\sqrt{k+2} \right)\,. \end{aligned} \tag{4.33}$$

In the same way we merge (4.31) and (4.28) into a single term

$$2\theta qQ\sum_{k=\alpha}^{\beta-1}\overline{c_{k+1}}c_k\sqrt{k+1}\;+\;2\theta q\overline{Q}\sum_{k=\alpha}^{\beta-1}\overline{c_k}c_{k+1}\sqrt{k+1}$$
$$=4\theta q\Re\left(Q\sum_{k=\alpha}^{\beta-1}\overline{c_{k+1}}c_k\sqrt{k+1}\right). \tag{4.34}$$

Without any trick but simple algebra we can combine the two parts of (4.29) and get

$$\theta^2Q\overline{Q}\sum_{k=\alpha}^{\beta}\overline{c_k}c_k k\;+\;\theta^2\overline{Q}Q\sum_{k=\alpha}^{\beta}\overline{c_k}c_k\,(k+1)=\theta^2Q\overline{Q}\sum_{k=\alpha}^{\beta}\overline{c_k}c_k\,(2k+1)\,. \tag{4.35}$$

At the end of the day we can collect the remaining pieces (4.33), (4.34), (4.35) and (4.30) and rebuild the overall expression for $\left\langle w \,\middle|\, x^2 \,\middle|\, w\right\rangle$ as

$$\begin{aligned}\left\langle w \,\middle|\, x^2 \,\middle|\, w\right\rangle =&2\theta^2\Re\left(Q^2\sum_{k=\alpha}^{\beta-2}\overline{c_{k+2}}c_k\sqrt{k+1}\sqrt{k+2}\right)\\ &+4\theta q\Re\left(Q\sum_{k=\alpha}^{\beta-1}\overline{c_{k+1}}c_k\sqrt{k+1}\right)\\ &+\theta^2Q\overline{Q}\sum_{k=\alpha}^{\beta}\overline{c_k}c_k\,(2k+1)+q^2\sum_{k=\alpha}^{\beta}\overline{c_k}c_k\,.\end{aligned} \tag{4.36}$$

After we now finished the hardest part it is time to return the whole expression (4.21) and we have all parts necessary to rewrite this braket in terms of parameters sets Π, $\tilde{\Pi}$ and coefficients c and ε only. Define for the sake of readability

$$W:=\langle w\,|\,w\rangle=\sum_{k=\alpha}^{\beta}\overline{c_k}c_k\,. \tag{4.37}$$

and denote the first term in (4.36) by a and the third one by c. Using these shortcuts we write

$$\frac{\left\langle w \,\middle|\, x^2 \,\middle|\, w\right\rangle}{\langle w\,|\,w\rangle}-\tilde{q}^2=\frac{a+c}{W}+2q\frac{2\theta}{W}\Re\left(Q\sum_{k=\alpha}^{\beta-1}\overline{c_{k+1}}c_k\sqrt{k+1}\right)+q^2-\tilde{q}^2$$

and insert a decomposition of the zero

$$\begin{aligned}&=\frac{a+c}{W}+2q\frac{2\theta}{W}\Re\left(Q\sum_{k=\alpha}^{\beta-1}\overline{c_{k+1}}c_k\sqrt{k+1}\right)+2q^2-2q^2+q^2-\tilde{q}^2\\ &=\frac{a+c}{W}+2q\left(\frac{2\theta}{W}\Re\left(Q\sum_{k=\alpha}^{\beta-1}\overline{c_{k+1}}c_k\sqrt{k+1}\right)+q\right)-q^2-\tilde{q}^2\end{aligned}$$

where we find that the part inside the big parentheses is just $\tilde{q}$ and therefore we continue like

$$= \frac{a+c}{W} + 2q\tilde{q} - q^2 - \tilde{q}^2$$

and factor the square

$$= \frac{a+c}{W} - (q-\tilde{q})^2 \,.$$

We are left with second off-diagonal terms of a and diagonal terms of c only, the first off-diagonal terms cancel respectively can be put inside the already known value of $\tilde{q}$. To conclude this section we write the final formula

$$\frac{\left\langle w \,\middle|\, (x-\tilde{q})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} = \frac{\varepsilon^2 \Re\left(Q^2 \sum_{k=\alpha}^{\beta-2} \overline{c_{k+2}} c_k \sqrt{k^2+3k+2}\right) + \frac{\varepsilon^2}{2}|Q|^2 \sum_{k=\alpha}^{\beta} \overline{c_k} c_k \,(2k+1)}{\sum_{k=\alpha}^{\beta} \overline{c_k} c_k} - (q-\tilde{q})^2 \,. \tag{4.38}$$

Instead of repeating all this for the other braket $\left\langle w \,\middle|\, (y-p)^2 \,\middle|\, w \right\rangle$ we apply the symmetry transformation here and replace *every* Q and q, not only the ones that originally appear inside the operator y. This then yields

$$\frac{\left\langle w \,\middle|\, (x-\tilde{p})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} = \frac{\varepsilon^2 \Re\left(P^2 \sum_{k=\alpha}^{\beta-2} \overline{c_{k+2}} c_k \sqrt{k^2+3k+2}\right) + \frac{\varepsilon^2}{2}|P|^2 \sum_{k=\alpha}^{\beta} \overline{c_k} c_k \,(2k+1)}{\sum_{k=\alpha}^{\beta} \overline{c_k} c_k} - (p-\tilde{p})^2 \,. \tag{4.39}$$

If we have the case that the estimation of $\tilde{q}$ is perfect then the last square vanishes. This really happens for the very specific special case where only one of the coefficients is non-zero

$$c := \big(c_\alpha = 0, \ldots, c_{\xi-1} = 0, c_\xi = 1, c_{\xi+1} = 0, \ldots, c_\beta = 0\big) = \delta_{k,\xi}$$

where k is the index of a coefficient c_k. For computing the value of $\langle w \,|\, x \,|\, w \rangle$ we insert this Kronecker delta into the formula (4.17)

$$\tilde{q} = \sqrt{2\varepsilon^2}\Re\left(Q \sum_{k=\alpha+1}^{\beta} \overline{\delta_{k,\xi}} \delta_{k-1,\xi} \sqrt{k}\right) + q \,.$$

For a product of two Kronecker deltae it holds that

$$\delta_{\alpha,\beta}\delta_{\gamma,\delta} = 1 \quad \Longleftrightarrow \quad \alpha = \beta \wedge \gamma = \delta \,.$$

Thus we get the unsatisfiable constraint that $k = \xi \wedge k - 1 = \xi$ which crunches the first sum to zero. What remains is

$$\tilde{q} = q\overline{c_\xi} c_\xi = q \,.$$

In the same way we get

$$\tilde{p} = p \,.$$

Note that these relations are *not* true in general when we put no restrictions on the coefficients c of the fragment.

4.3.4 Parameters $\tilde{P}$ and $\tilde{Q}$

For the two parameters $\tilde{P}$ and $\tilde{Q}$ there is no trivial relation as for $\tilde{p}$ and $\tilde{q}$, shown in formula (4.17) and (4.18). Computing the expectation value $\langle w \,|\, (x-\tilde{q})^2 \,|\, w \rangle$ is just the first step on the way to $\tilde{P}$ and $\tilde{Q}$. The next step is given by the relation which can be found at [9, formula 2.24]. In our notation for P and Q this translates to

$$\left\langle \phi_k \,\middle|\, (x-q)^2 \,\middle|\, \phi_k \right\rangle = \frac{\varepsilon^2}{2} |Q|^2 \,(2k+1) \tag{4.40}$$

and in analogy

$$\left\langle \phi_k \,\middle|\, (x-p)^2 \,\middle|\, \phi_k \right\rangle = \frac{\varepsilon^2}{2} |P|^2 \,(2k+1) \;. \tag{4.41}$$

Solving these equations for $|P|^2$ and $|Q|^2$ then gives

$$\begin{aligned} |P|^2 &= \frac{2}{\varepsilon^2 \,(2k+1)} \left\langle \phi_k \,\middle|\, (x-p)^2 \,\middle|\, \phi_k \right\rangle \\ |Q|^2 &= \frac{2}{\varepsilon^2 \,(2k+1)} \left\langle \phi_k \,\middle|\, (x-q)^2 \,\middle|\, \phi_k \right\rangle . \end{aligned} \tag{4.42}$$

We can then substitute the results from (4.38) and (4.39) for the braket into this equation and get and estimate of $|\tilde{Q}|^2$ and $|\tilde{P}|^2$. So this tells us how we can estimate the squared absolute values of $\tilde{P}$ and $\tilde{Q}$ from a fragment w but not how to get the explicit complex values. A solution to this dilemma was attempted in [5, formula 16 and 17] by choosing $\tilde{Q}$ real and $\tilde{Q} > 0$ and then setting

$$\begin{aligned} \tilde{Q} &= \sqrt{|\tilde{Q}|^2} \\ \tilde{P} &= \frac{\sqrt{\tilde{Q}^2 |\tilde{P}|^2 - 1} + i}{\tilde{Q}} \end{aligned} \tag{4.43}$$

but turned out to be incomplete in some situations. So we need to find a more general solution. We know that $\tilde{P}$ and $\tilde{Q}$ are complex variables hence we can decompose them as

$$\begin{aligned} \tilde{Q} &= a + ib \\ \tilde{P} &= c + id \,. \end{aligned} \tag{4.44}$$

Denote the known values of $|\tilde{Q}|^2$ and $|\tilde{P}|^2$ by $v_{\tilde{Q}}$ and $v_{\tilde{P}}$. Of course the compatibility relations (2.1) from chapter 2 still apply. Combining all these parts we end up with the following system of equations

$$\begin{aligned} |\tilde{Q}|^2 &= v_{\tilde{Q}} \\ |\tilde{P}|^2 &= v_{\tilde{P}} \\ \overline{\tilde{Q}}\tilde{P} - \overline{\tilde{P}}\tilde{Q} &= 2i \end{aligned} \tag{4.45}$$

where we can plug in the explicit complex decomposition from above

$$\begin{aligned} a^2 + b^2 &= v_{\tilde{Q}} \\ c^2 + d^2 &= v_{\tilde{P}} \\ ad - bc - 1 &= 0\,. \end{aligned} \tag{4.46}$$

We have only three equations for the four variables a, b, c and $d \in \mathbb{R}$ so there is one degree of freedom in these expressions. In principle we could read the restriction from above and use $b = 0$ as a fourth equation. But even with this restriction the system has four solutions because of the multivalued nature of the square root function

$$\tilde{Q} = -\sqrt{v_{\tilde{Q}}} \qquad \tilde{P} = -\sqrt{v_{\tilde{P}} - \frac{1}{v_{\tilde{Q}}}} - \frac{i}{\sqrt{v_{\tilde{Q}}}} \tag{4.47a}$$

$$\tilde{Q} = \sqrt{v_{\tilde{Q}}} \qquad \tilde{P} = -\sqrt{v_{\tilde{P}} - \frac{1}{v_{\tilde{Q}}}} + \frac{i}{\sqrt{v_{\tilde{Q}}}} \tag{4.47b}$$

$$\tilde{Q} = -\sqrt{v_{\tilde{Q}}} \qquad \tilde{P} = \sqrt{v_{\tilde{P}} - \frac{1}{v_{\tilde{Q}}}} - \frac{i}{\sqrt{v_{\tilde{Q}}}} \tag{4.47c}$$

$$\tilde{Q} = \sqrt{v_{\tilde{Q}}} \qquad \tilde{P} = \sqrt{v_{\tilde{P}} - \frac{1}{v_{\tilde{Q}}}} + \frac{i}{\sqrt{v_{\tilde{Q}}}}\,. \tag{4.47d}$$

Therefore we need a criterion to choose one of these solutions. But let us first go on in the spawning procedure and assume for now that we have a valid value for $\tilde{Q}$ and $\tilde{P}$.

4.3.5 Algorithmic formulation of parameter estimation

The algorithm 4 summarizes the steps shown in the last sections. Given a fragment w together with all necessary information it will compute a new parameter set $\tilde{\Pi}$ and hence a new basis of L^2. This is the last step to take before we can go on and move the fragment w into this new basis.

4.4 Change of basis

The second step in the spawning process is a simple change of basis. Under this term we summarize several methods. Not all of them correspond to what is a change of basis in the sense of linear algebra. We have computed another basis of L^2 in the last sections by finding a new set of parameters $\tilde{\Pi}$. This establishes a new basis $\{\tilde{\phi}_k\}_k$ that we will use for representing the spawned fragment $|\tilde{w}\rangle$. The mathematical formulation of the problem we want to solve in this section can be posed as: *given a fragment w by*

$$w = \sum_{k=\alpha}^{\beta} c_k \phi_k \tag{4.48}$$

Algorithm 4 Parameter estimation for fragments

Require: The parameters Π of the fragment w
Require: The coefficients $\{c_i\}_{i=\alpha}^{\beta}$ of the fragment w
Require: $\alpha \geq 0$ and $\beta \leq K-1$
Require: A value for $k \in \{0, \ldots, \eta - 1\}$
// Compute the squared norm of w
$\|w\|^2 := \sum_{k=\alpha}^{\beta} \overline{c_k} c_k$
// Compute the expectations for position and momentum
$\vartheta_0 := \sum_{k=\alpha+1}^{\beta} \overline{c_k} c_{k-1} \sqrt{k}$
$\tilde{q} := \frac{\sqrt{2\varepsilon^2}}{\|w\|^2} \Re\left(Q\vartheta_0\right) + q$
$\tilde{p} := \frac{\sqrt{2\varepsilon^2}}{\|w\|^2} \Re\left(P\vartheta_0\right) + p$
// Estimate second moments
$\vartheta_1 := \sum_{k=\alpha}^{\beta-2} \overline{c_{k+2}} c_k \sqrt{k^2+3k+2}$
$\vartheta_2 := \sum_{k=\alpha}^{\beta} \overline{c_k} c_k \left(2k+1\right)$
$M_{\tilde{Q}} := \frac{\varepsilon^2}{\|w\|^2} \left(\Re\left(Q^2\vartheta_1\right) + \frac{1}{2}|Q|^2\vartheta_2\right) - \left(q - \tilde{q}\right)^2$
$M_{\tilde{P}} := \frac{\varepsilon^2}{\|w\|^2} \left(\Re\left(P^2\vartheta_1\right) + \frac{1}{2}|P|^2\vartheta_2\right) - \left(p - \tilde{p}\right)^2$
// Plug into (4.42)
$|\tilde{Q}|^2 := \frac{2}{\varepsilon^2(2k+1)} M_{\tilde{Q}}$
$|\tilde{P}|^2 := \frac{2}{\varepsilon^2(2k+1)} M_{\tilde{P}}$
// The set of new parameters
$\tilde{\Pi} := \{\tilde{P}, \tilde{Q}, S, \tilde{p}, \tilde{q}\}$
return $\tilde{\Pi}$

in the old basis, find new coefficients $(d_0, \ldots, d_\eta)$ *such that* w *can be written as a linear combination in the new basis*

$$w \approx \tilde{w} := \sum_{k=0}^{\eta} d_k \tilde{\phi_k} \tag{4.49}$$

with conserving $\|w + \tilde{w}\|$ *as much as possible.* With an infinite basis, i.e. $\eta = \infty$ we could achieve a perfect transformation in all cases and hence an equal sign in the equation above.
Throughout this section we will assume that the original basis is of size K and the set of coefficients of the basis expansion is denoted by $c := \{c_k\}_{k=0}^{K-1}$. For the fragment w we require that $\alpha \geq 0$ and $\beta \leq K - 1$. The new basis can have a different size η which we usually choose much smaller than K and we call the set of new coefficients $d := \{d_k\}_{k=0}^{\eta-1}$.

4.4.1 Lumping procedure

A very simple way to find the coefficients d is what we call the *lumping method*. The method can be explained best if we assume for the moment that we want to spawn a single Gaussian ϕ_0 only. This assumption then gives a first hint that all d_k with $k > 0$ are set to zero and we only have to find a value for d_0. Since there has to be norm conservation of $w + \tilde{w}$ with initially $\|\tilde{w}\| = 0$ and after spawning $\|w\| = 0$ we have to set

$$d_0 := \|w\| . \tag{4.50}$$

But we also have to remove the fragment w and set $c_\alpha, \ldots c_\beta$ to zero. Therefore we can summarize the whole procedure by the following two assignments

$$\begin{aligned} \begin{pmatrix} c_\alpha & \cdots & c_\beta \end{pmatrix} &:= 0 \\ d &:= \begin{pmatrix} d_0 & 0 & \cdots & 0 \end{pmatrix} \end{aligned} \tag{4.51}$$

where $d_0 = \sqrt{\langle w, w \rangle} = \|w\|$ and all other coefficients up to d_η are 0. With this method we get perfect norm conservation but this does not imply perfect approximation or minimal spawning error and $\|w - \tilde{w}\| \neq 0$ in general because w is not a Gaussian in general. When applying this method to tunneling problems, w is a Gaussian at least asymptotically and the method is therefore appropriate. The details can be found in reference [5].
If we a priori know what value k will have or in other words which basis function $\tilde{\phi_k}$ will dominate then we can take this into account and set $d_k = \|w\|$ and $d_i = 0$ for all $i \neq k$. The algorithm 5 shows again the precise details of this method.
A more general version of the lumping method would use the first μ coefficients $d_0, \ldots, d_{\mu-1}$ and hence effectively spawn a linear combination instead of a pure single basis function $\tilde{\phi_k}$ but in that case it is not obvious what values have to be given to these $\{d_i\}_{i=0}^{\mu-1}$.

4.4.2 Basis projection procedure

Another method for the change of basis is called the *basis projection* and is a real projection in the sense of linear algebra. The task now requires projecting w to

Algorithm 5 Lumping method for the change of basis

Require: The size K of the old and η of the new basis
Require: The fragment w with its coefficients $\{c_i\}_{i=\alpha}^{\beta}$
Require: $\alpha \geq 0$ and $\beta \leq K - 1$
Require: A value for $k \in \{0, \ldots, \eta - 1\}$
  // Initialize all new coefficients with zero
  $(d_0, \ldots, d_\eta) := 0$
  // Assign the norm of w to the new coefficient k
  $d_k := \sqrt{\sum_{i=\alpha}^{\beta} c_i^2}$
  // Set the old coefficients to zero
  $(c_\alpha, \ldots, c_\beta) := 0$
  return $d = (d_0, \ldots, d_\eta)$

the new basis specified by the parameter set $\tilde{\Pi}$ and consisting of basis functions $\{\tilde{\phi}_k\}_{k=0}^{\eta-1}$. We can derive the following projection formula for the coefficients d_i of the spawned fragment $\tilde{w}$ by using the normed inner product

$$d_i := \frac{\left\langle w \,\middle|\, \tilde{\phi}_i \right\rangle}{\langle w \,|\, w \rangle} \,. \tag{4.52}$$

The inner product above is essentially an integral, so we can apply our quadrature rule (γ, ω) of order R here and get

$$d_i = \varepsilon \cdot Q_0 \sum_{r=0}^{R-1} \overline{w\left(\gamma_r\right)} \cdot \tilde{\phi}_i\left(\gamma_r\right) \cdot \omega_r \,. \tag{4.53}$$

If we write out the $w\left(\gamma_r\right)$ part as basis expansion we get the following relation which now can be computed directly

$$d_i = \varepsilon \cdot Q_0 \sum_{r=0}^{R-1} \overline{\sum_{k=\alpha}^{\beta} c_k \phi_k\left(\gamma_r\right)} \cdot \tilde{\phi}_i\left(\gamma_r\right) \cdot \omega_r \,. \tag{4.54}$$

We repeat the above quadrature for each i in the range $\{0, \ldots, \mu - 1\}$ where $\mu < \eta$ and hence projecting on the first μ basis functions $\{\tilde{\phi}_k\}_{k=0}^{\mu-1}$ of the spawned basis. From this projection we get the non-zero coefficients $\{d_i\}_{i=0}^{\mu-1}$ of the new fragment $\tilde{w}$ while all higher d_i with $i \geq \mu$ are zero. At this moment we have to remove the part of w and set the coefficients $c_\alpha, \ldots, c_\beta$ to zero. The following two assignments give an overview over both operations

$$\begin{aligned} d &:= \begin{pmatrix} d_0 & \cdots & d_\mu & d_{\mu+1} = 0 & \cdots & d_{\eta-1} = 0 \end{pmatrix} \\ \begin{pmatrix} c_\alpha & \cdots & c_\beta \end{pmatrix} &:= 0 \,. \end{aligned} \tag{4.55}$$

If we want to spawn only a single Gaussian then we can simply set μ to 0 but projecting only onto $\tilde{\phi}_0$ is not the same as lumping with $k = 0$. When η is really small for example say below 8 then we project on a very small basis and might loose much of the norm of w, thus it can happen that $\|\tilde{w}\| \ll \|w\|$. This is the case when w has only minor components in the direction of the first μ new basis

functions. In contrast to this we have perfect norm conservation with lumping. On the other hand, spawning whole linear combinations here arises naturally from the projection while it's a currently unsolved problem with the lumping method. In general the projection method with a sensible size η of the basis will give better results than lumping as in most simulations we will never have pure basis states and this is the only case where lumping is really advantageous.
The only remaining question now is which rule we use for the quadrature in (4.54). There are at least three reasonable options and it's not entirely clear yet which is the best one. Mathematically the mixing quadrature from section 3.2.2 is the only correct choice. The mixing quadrature is considered to be correct because w in the bra and the $\tilde{\phi}_i$ in the ket belong to two different bases with parameters sets Π and $\tilde{\Pi}$ respectively thus we have to mix the two sets as shown in algorithm 3 for a good quadrature. But we may also want to focus the quadrature rule onto the spawned wavepacket $\tilde{w}$. This seems to be also a good idea at first glance because we are interested in the projection of w to the new basis given by $\tilde{\Pi}$. But it turns out that this focusing yields a worse norm conservation than mixing quadrature, compare the panels in figure 4.4 for an explicit example. For this reason we concentrate only on the mixing quadrature. Algorithm 6 shows an algorithmic formulation of basis projection method. The code is inefficient because of the nested loops and the actual implementation uses several optimizations like vectorization and contains no explicit loop.

Algorithm 6 Basis projection method for the change of basis

Require: The size K of the old and η of the new basis
Require: The parameters Π and $\tilde{\Pi}$ defining the bases
Require: The two sets of basis functions $\{\phi_k\}_{k=0}^{K-1}$ and $\{\tilde{\phi}_k\}_{k=0}^{\eta-1}$
Require: The mother fragment w with its coefficients $\{c_k\}_{k=\alpha}^{\beta}$
Require: A mixing quadrature rule of order R with nodes γ and weights ω
Require: A value for $\mu \in \{0, \ldots, \eta - 1\}$
Require: $\alpha \geq 0$ and $\beta \leq K - 1$
  // Compute Q_S from mixing Π and $\tilde{\Pi}$ by the mixing procedure 3
  $Q_S :=$ mix_parameters$(\Pi, \tilde{\Pi})$
  // Initialize all new coefficients with zero
  $(d_0, \ldots, d_{\eta-1}) := 0$
  // Project to the basis of the spawned part and compute new coefficients
  for $i := 0$ **to** μ **do**
    for $r := 0$ **to** $R - 1$ **do**
      $d_i := d_i + \overline{\sum_{k=\alpha}^{\beta} c_k \phi_k(\gamma_r)} \cdot \tilde{\phi}_i(\gamma_r) \cdot \omega_r$
    end for
    $d_i := \varepsilon \cdot Q_S \cdot d_i$
  end for
  // Set the old coefficients to zero
  $(c_\alpha, \ldots, c_\beta) := 0$
  return $d := (d_0, \ldots, d_{\eta-1})$

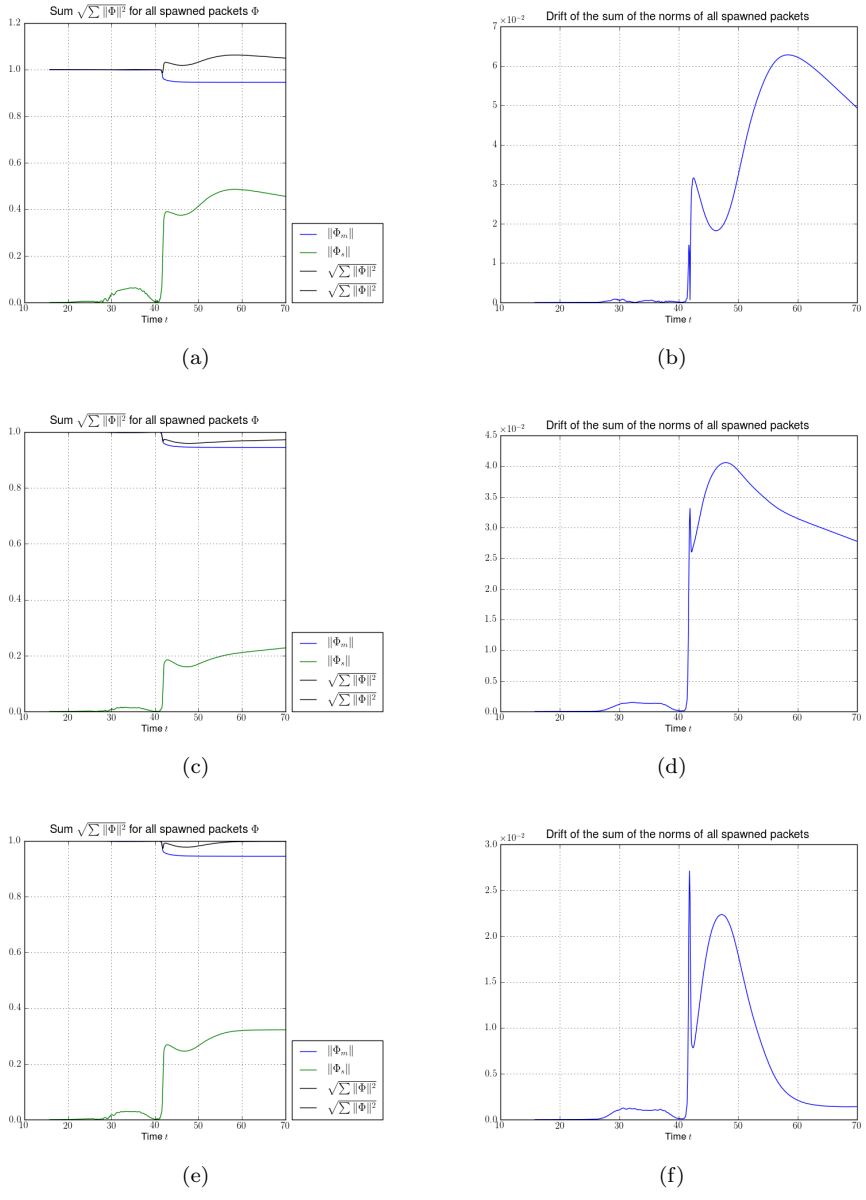


Figure 4.4: The plots show the norms and the norm drift obtained by spawning with the projection method. For the quadrature used inside the projection method there are three possibilities: the quadrature can be homogeneous and focused on either the mother or the child (spawned) wavepacket. Or it can be the fully inhomogeneous quadrature, which is the only correct choice from the mathematical point of view and also yields the least and most rapidly decaying error. (a) Norm, computed with quadrature focused on the mother part. (b) Difference to the theoretical norm, computed with quadrature focused on the mother part. (c) Norm, computed with quadrature focused on the child part. (d) Difference to the theoretical norm, computed with quadrature focused on the child part. (e) Norm, computed with inhomogeneous quadrature. (f) Difference to the theoretical norm, computed with inhomogeneous quadrature.

4.5 Problematic difficulties and open issues

Now we can return to the problem of choosing concrete values for $\tilde{P}$ and $\tilde{Q}$ we deferred at the end of section 4.3.4. After we developed the formalism to really create the spawned fragments $\tilde{w}$ in the last section we can now use this to determine the best choice among the four possibilities of (4.47). The idea is to spawn a fragment using each of these four different parameter pairs resulting in the test fragments $\tilde{w_1}, \tilde{w_2}, \tilde{w_3}$ and $\tilde{w_4}$. And then we can try to maximize the overlap with the original fragment w

$$\arg\max_i \langle \tilde{w_i} \,|\, w \rangle \ . \tag{4.56}$$

This gives us then the best pair of parameters and we can finish the set $\tilde{\Pi}$. The drawback of this method is that while the original parameters $P(t)$ and $Q(t)$ are continuous and smooth functions of time, this maximization procedure can introduce arbitrary jumps. Hence while this method works most of the time and these jumps do not matter for our purposes it's still not the optimal solution. When we combine spawning and propagation later these jumps won't be a problem because we spawn once and the propagate smoothly.

Another severe problem already mentioned above is posed by the k in formula (4.42). In general we do not work with single basis functions but with fragments which are linear combinations thereof. Therefore we need to estimate the parameters for such a linear combination to get a good basis for spawning. We have no way to know a value of k at spawning time and we also have no suitable way to compute it. For the tunneling problem it can be shown [4] that the transmitted part is always a Gaussian (at least asymptotically for large times) and hence we can set $k = 0$ there and spawn a ϕ_0 by lumping. (The procedure still works for the basis projection method, but depends on the fact that we can set $k = 0$ as justified by theory.) More on this in the next chapter.
In the non-adiabatic case k can have any value. Additionally we won't spawn just a single function ϕ_k but a whole linear combination $\sum_i \phi_i$. Under these circumstances it is even more obscure what k should be. A real world example is shown in figure 4.5 where we would like to spawn on the lower level. The fragment there has clearly the shape of a ϕ_2 but setting $k = 2$ can not be justified if we look at the lower right panel.

The last open issue we would like to mention here is that sometimes the relation

$$|\tilde{P}|^2 |\tilde{Q}|^2 \geq 1 \tag{4.57}$$

is violated. This is really bad because this inequality expresses Heisenberg's uncertainty principle which is fundamental in quantum physics and should always be fulfilled, see also [9, remark 2.7]. Violations of this inequality will in turn induce a violation of the compatibility relations (2.1).
The origin of these problems is again formula (4.42). The equation shown there is mathematically only correct for single basis functions ϕ_k and does not provide any solution for linear combinations like our fragments. Because of this it is

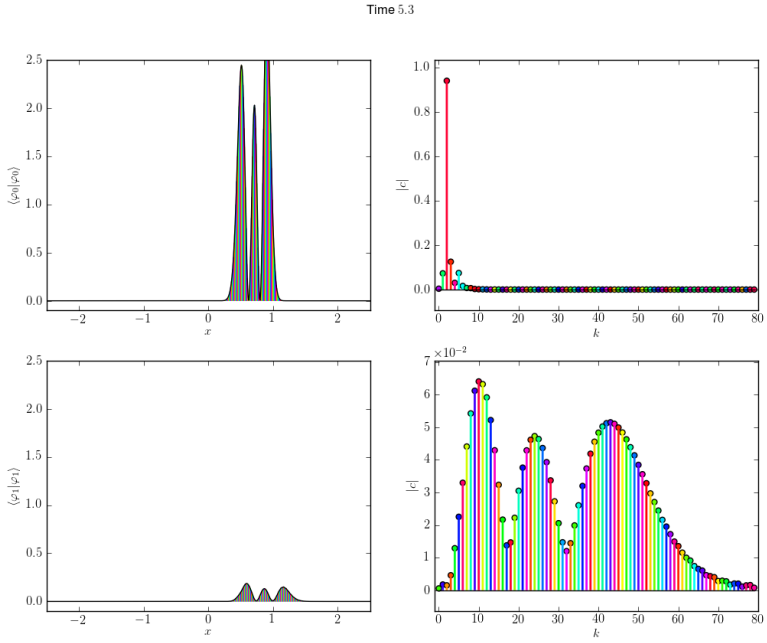


Figure 4.5: If we spawn on the upper level then we can justify to choose $k = 2$ but when we would like to spawn on the lower level, it is by far not obvious how to choose k. One could start guessing and try f.e. the median. In both cases the fragment (left panels) used for estimating parameters has the shape of a ϕ_2.

strictly speaking not allowed to plug in the estimated second moments of w as we did. This discrepancy is what causes the violation of the uncertainty principle. Luckily for us it happened in the simulations only at times where spawning would make no sense anyway. And we should add that it never happened if $k = 0$ but this is no solution either as it will hinder the spawning process taking place at times when the violation is not present.
It's not clear what the best way to proceed from here is. Also it's not clear how important an accurate guess of $\tilde{P}$ and $\tilde{Q}$ really is. For example one can obtain a decent result by performing spawning by doing basis projection and using a large enough target basis.

Chapter 5

Spawning in tunneling problems

In this chapter we will apply the theoretical ideas of spawning developed in the last chapter to the tunneling problem which motivated these ideas in the first place. We will briefly review the original work from the paper [5] and then extend these ideas further. We will explore the advantages of the new projection method over the lumping method for spawning and see how this helps to get better results at finite times. What was done in that paper is a procedure we call now *aposteriori spawning* because we need a full simulation before we can start to analyze the benefits of spawning. While this provides us with insight to the fundamental process, we need to interleave spawning and propagation tightly if we want to improve the simulation algorithms in practice. If done right this reduces the wall time and memory cost drastically. But it is a highly non-trivial task with many pitfalls potentially turning the simulation results into garbage. We will look at all these details in the section on *spawning and propagation*.

5.1 Adapting the theoretical basis

In a first step we adapt the theory from the last chapter to the tunneling problem. This gives simplified formulae for computing the spawned wavepackets. In the tunneling problem our original wavepacket $|\Psi\rangle$ has only a single component Φ_0 and the potential consists of a single real valued function. An example for a typical potential was given in figure 4.1. The component Φ_0 is given according to the definition from (4.1) with a basis of size K. Adapting the notation from [5] we write

$$|\Psi\rangle = u\,(t) =: v\,(t) + w\,(t) \tag{5.1}$$

where we decompose the wavepacket into two parts. These two parts will approximately be the reflected and the transmitted part of the incoming wavefunction. The main point of this whole chapter is to try to replace the transmitted part w by a newly spawned wavepacket $\tilde{w}$. This spawned wavepacket will be independent from the original v and both can be propagated independently[1]. The $w\,(t)$ from

[1] This is not the complete truth. The parameter sets Π and $\tilde{\Pi}$ can be propagated indepen-

above is our fragment and we rewrite the equation in more detail as

$$|\Psi\rangle = \sum_{k=0}^{K_0-1} c_k\phi_k + \underbrace{\sum_{k=K_0}^{K-1} c_k\phi_k}_{=:w} \tag{5.2}$$

where we omitted the global phase which is present for both summands. The first few coefficients $c_0, \ldots, c_{K_0-1}$ are usually not interesting for spawning, they remain at the mother wavepacket and belong to the reflected part. For a picture of the actual situation we refer to figure 4.2 from the last chapter. The value denoted by K_0 is the cutoff where the fragment starts and hence we set $\alpha = K_0$ and $\beta = K - 1$. For the estimated position and momentum we plug in these values in (4.17) and (4.18) and get

$$\tilde{q} := \frac{\langle w \,|\, x \,|\, w\rangle}{\langle w \,|\, w\rangle} = \frac{\sqrt{2\varepsilon^2}}{\sum_{k=K_0}^{K-1} \overline{c_k} c_k} \Re\left(Q \sum_{k=K_0+1}^{K-1} \overline{c_k} c_{k-1}\sqrt{k}\right) + q \tag{5.3}$$

$$\tilde{p} := \frac{\langle w \,|\, y \,|\, w\rangle}{\langle w \,|\, w\rangle} = \frac{\sqrt{2\varepsilon^2}}{\sum_{k=K_0}^{K-1} \overline{c_k} c_k} \Re\left(P \sum_{k=K_0+1}^{K-1} \overline{c_k} c_{k-1}\sqrt{k}\right) + p\,. \tag{5.4}$$

These two formulae are equivalent to formula 14 and 15 from [5]. Notice that our sums stop at $K-1$ inclusive because our basis is of overall size K. For the second moments we apply the same simplifications to (4.38) and (4.39) which then yields

$$\frac{\left\langle w \,\middle|\, (x-\tilde{q})^2 \,\middle|\, w\right\rangle}{\langle w \,|\, w\rangle} = \frac{\varepsilon^2 \Re\left(Q^2 \sum_{k=K_0}^{K-3} \overline{c_{k+2}} c_k \sqrt{k^2+3k+2}\right) + \frac{\varepsilon^2}{2}|Q|^2 \sum_{k=K_0}^{K-1} \overline{c_k} c_k\,(2k+1)}{\sum_{k=K_0}^{K-1} \overline{c_k} c_k} - (q-\tilde{q})^2 \tag{5.5}$$

$$\frac{\left\langle w \,\middle|\, (x-\tilde{p})^2 \,\middle|\, w\right\rangle}{\langle w \,|\, w\rangle} = \frac{\varepsilon^2 \Re\left(P^2 \sum_{k=K_0}^{K-3} \overline{c_{k+2}} c_k \sqrt{k^2+3k+2}\right) + \frac{\varepsilon^2}{2}|P|^2 \sum_{k=K_0}^{K-1} \overline{c_k} c_k\,(2k+1)}{\sum_{k=K_0}^{K-1} \overline{c_k} c_k} - (p-\tilde{p})^2\,. \tag{5.6}$$

Additionally we can set $k = 0$ in (4.42) in this special setup and get

$$\begin{aligned} |P|^2 &= \frac{2}{\varepsilon^2}\left\langle \phi_k \,\middle|\, (x-p)^2 \,\middle|\, \phi_k\right\rangle \\ |Q|^2 &= \frac{2}{\varepsilon^2}\left\langle \phi_k \,\middle|\, (x-q)^2 \,\middle|\, \phi_k\right\rangle\,. \end{aligned} \tag{5.7}$$

The foundation for this step are theoretical results [4] proving that in the tunneling problem the transmitted part w is at least asymptotically always a Gaussian in leading order. Hence we can concentrate on ϕ_0 while estimating

dently of each other by the usual algorithms from [2]. But of course the packets may still interact and thus the coefficients c and $\tilde{c}$ can not be propagated separately. In the case of the tunneling problem this rapidly becomes negligible because the two packets reside on different sides of the potential hill. We will return on this issue in more detail later.

parameters[2]. Plugging in the results from (5.5) and (5.6) here replacing the brakets we end up with two relations for $|\tilde{Q}|^2$ and $|\tilde{P}|^2$ which are equivalent to the ones for $|A|^2$ and $|B|^2$ in the aforementioned paper. This is all we had to do and we are now ready to use the lumping or projection methods directly as defined in the algorithms 5 and 6 for spawning wavepackets in tunneling problems. This also concludes our review of the theory part of reference [5].

5.2 Aposteriori spawning

In this section we will explain the term *aposteriori spawning* in details. This name stems from the fact, that all computations are done after the simulation. It means that first we need a full simulation of the packet $|\Psi\rangle$ being propagated through time. During this simulation nothing related to spawning happens. But afterwards we perform spawning computations and see what would happen if we used spawning to improve the simulation. The whole effort only helps to enhance our understanding of spawning. It does not provide us with a better or faster simulation or more exact results. This can only be the case if we use spawning *during* the simulation.
In an aposteriori spawning simulation we do compute a spawned wavepacket for each timestep of the original simulation. After we spawned the new packet, we measure energies and norms and test how much error we introduced while spawning this new wavepacket. Algorithm 7 gives a more schematic overview on how aposteriori spawning works in general.

Algorithm 7 Aposteriori spawning method (in general)

Require: A time series t consisting of timesteps $\{\tau_0, \ldots, \tau_{\max}\}$
Require: The wavepackets $\Psi(t)$ with parameters Π and coefficients $\{c_k^i\}_{k=0}^{K-1}$
Require: The component $0 \leq \nu \leq N$ of Ψ examined for spawning
Require: Integers $\alpha \geq 0$ and $\beta \leq K-1$
  for all $\tau_i \in t$ **do**
    // Construct the fragment as subset of Φ_ν
    $w := \sum_{k=\alpha}^{\beta} c_k^\nu \phi_k$
    // Estimate the parameters of the fragment by algorithm 4
    $\tilde{\Pi} :=$ **estimate_parameters**(w)
    // Move the fragment w to the new basis by either algorithm 5 or 6
    $\tilde{w} :=$ **change_basis**$(\tilde{\Pi}, w)$
    // Measure norms, energies etc
    measure$(\tilde{w})$
  end for

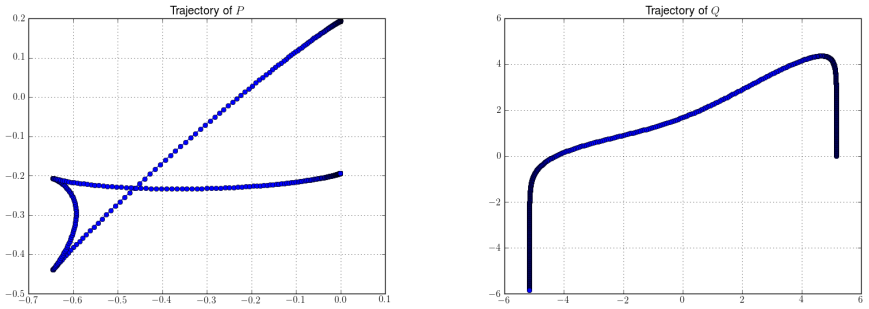
For the tunneling problem this algorithm simplifies a little bit. First our packet $|\Psi\rangle$ consists only of a single component Φ_0 and we set $\nu = 0$. Then we are interested in removing the high frequencies from the original wavepacket and put them into a new packet where the same information on the wavefunction can be (approximately) represented by much less and lower frequencies. Hence we

[2]We exploit this fact only for $\tilde{P}$ and $\tilde{Q}$ and only in the final step there because of the troubles shown at the end of the last chapter. For the second moments we still use the whole w and not only ϕ_0. The same holds for the estimated position and momentum as we can obtain exact results using the complete w easily.

set $\beta = K - 1$ and $\alpha = K_0$ where the correct choice of K_0 poses a new problem. Luckily it turns out that this choice is not critical as long as there is a region around K_0 where the coefficients c_i are very small. In other words, there are two bulges of large coefficients separated by a region with almost zeros and we can choose K_0 arbitrary but well within this region. If we refer to figure 4.2 then this region would range from about 40 up to about 100 and we can choose $K_0 = 50$. But we should keep in mind that for more advanced applications of spawning it may be difficult or even impossible to know a good K_0 a priori. In the aposteriori spawning process this has no importance as we can always look at the coefficients of the original simulation. For the spawning propagation algorithm one could try to use some statistical methods developed for classification problems.

5.3 Basic tunneling simulations

In this section we show the results of three tunneling simulations. First we send a wavepacket $|\Psi\rangle = \phi_0$ from the left. Then we try the same simulation setup with a ϕ_2 and a ϕ_3. All three simulations are very similar in the sense that we use the same initial parameters for the wavepackets with the only exception being the reduced momentum for the later two. The following plots build the reference to which we compare the various spawning approaches from the next section.



(a) (b)

Figure 5.1: The plot shows the trajectories of the parameters P and Q in the complex plane. The simulation parameters are shown in A.1.

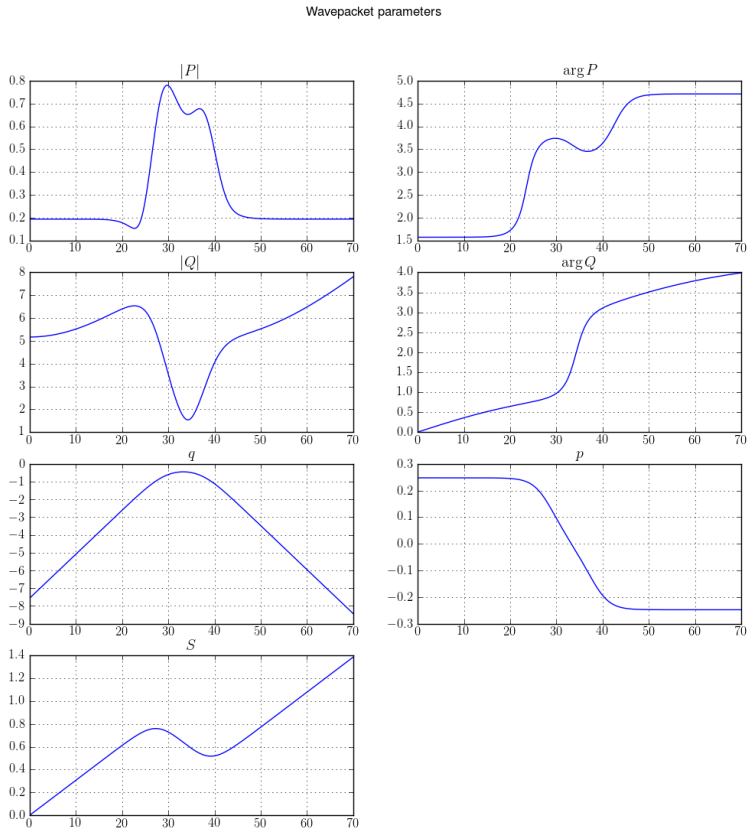


Figure 5.2: The parameter set Π of the simulation with ϕ_0 plotted versus time. The simulation parameters are shown in A.1.

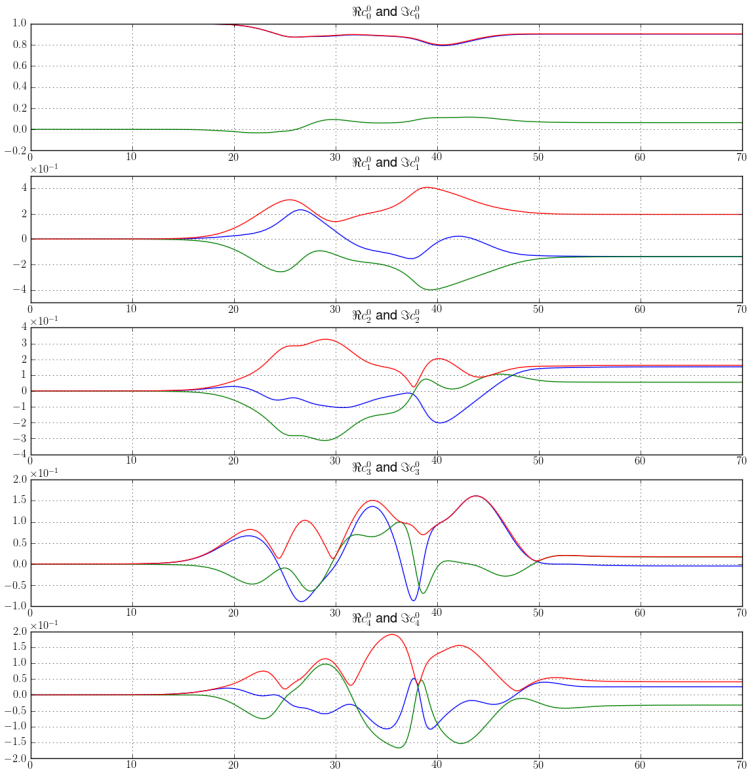


Figure 5.3: The first few coefficients c_i of the simulation with ϕ_0 plotted versus time. The simulation parameters are shown in A.1.

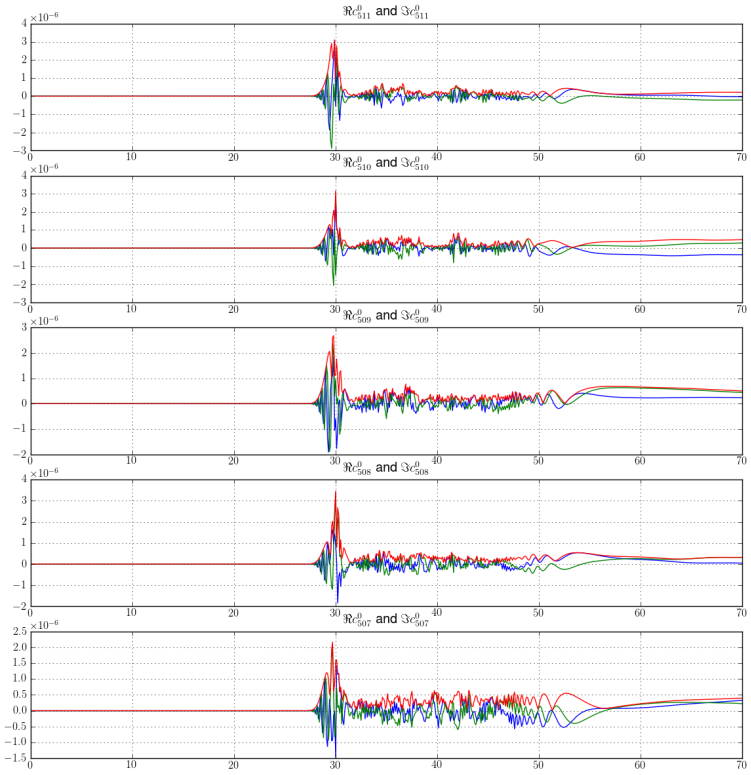


Figure 5.4: The last few coefficients c_i of the simulation with ϕ_0 plotted versus time. We see that their values are really small and thus the loss due to the finite basis is minimal. The simulation parameters are shown in A.1.

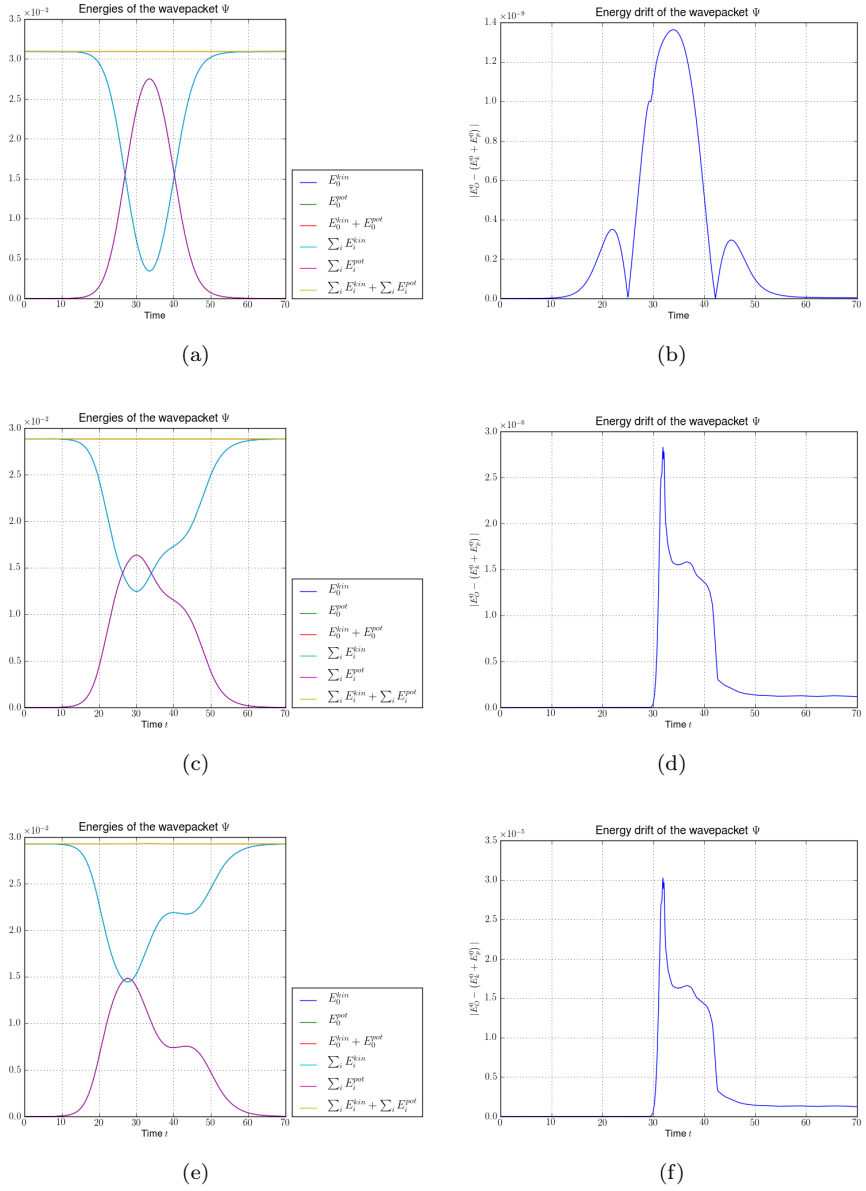
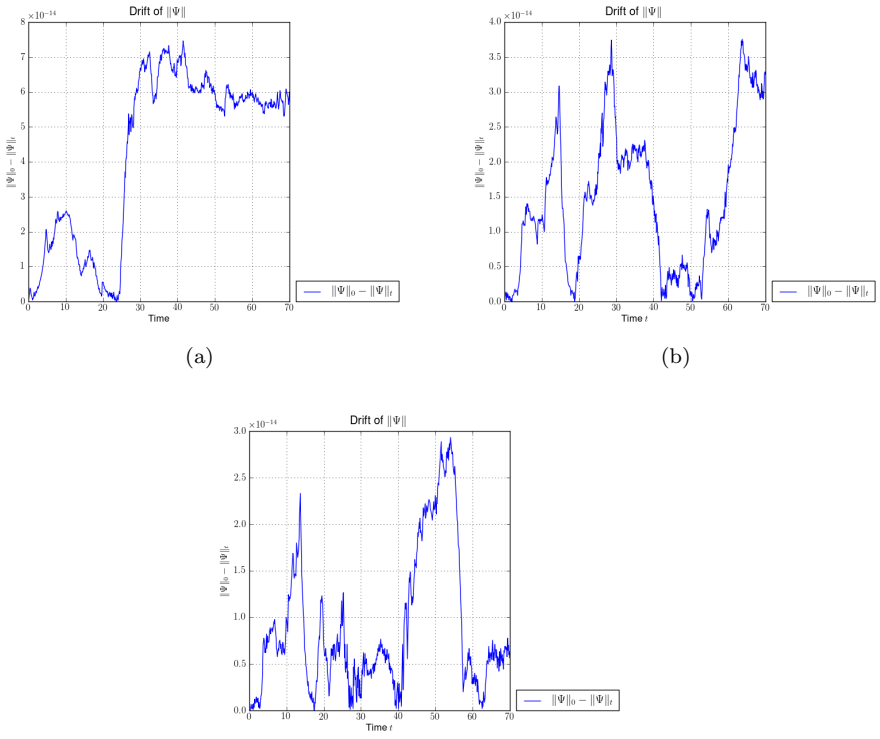


Figure 5.5: This figure shows the energies and energy drift of several tunneling simulations. The simulation parameters are printed in A.1, A.2.1 and A.2.2. (a) Kinetic and potential energy for a wavepacket ϕ_0 tunneling through the Eckart barrier. (b) Energy drift for a wavepacket ϕ_0 tunneling through the Eckart barrier. (c) Kinetic and potential energy for a wavepacket ϕ_2 tunneling through the Eckart barrier. (d) Energy drift for a wavepacket ϕ_2 tunneling through the Eckart barrier. (e) Kinetic and potential energy for a wavepacket ϕ_3 tunneling through the Eckart barrier. (f) Energy drift for a wavepacket ϕ_3 tunneling through the Eckart barrier.



(a) (b)

(c)

Figure 5.6: This figure shows the drift of the norm for several tunneling simulations. We have very good norm conservation. The reason for this is that we used a huge basis with $K = 512$. The simulation parameters are printed in A.1, A.2.1 and A.2.2. (a) Norm drift for a wavepacket ϕ_0 tunneling through the Eckart barrier. (b) Norm drift for a wavepacket ϕ_2 tunneling through the Eckart barrier. (c) Norm drift for a wavepacket ϕ_3 tunneling through the Eckart barrier.

5.4 Spawning using the lumping method

Now we try aposteriori spawning based on the simulation from the last section. In the first examples we use the lumping method which works quite well for large enough times. The spawned packet has the shape of a Gaussian and we set $k = 0$ for lumping. The initial values of $|\Psi\rangle$ are taken to be ϕ_0, ϕ_2 and ϕ_3. We try different values for the parameter K_0 and compare the estimated parameter sets $\tilde{\Pi}$.

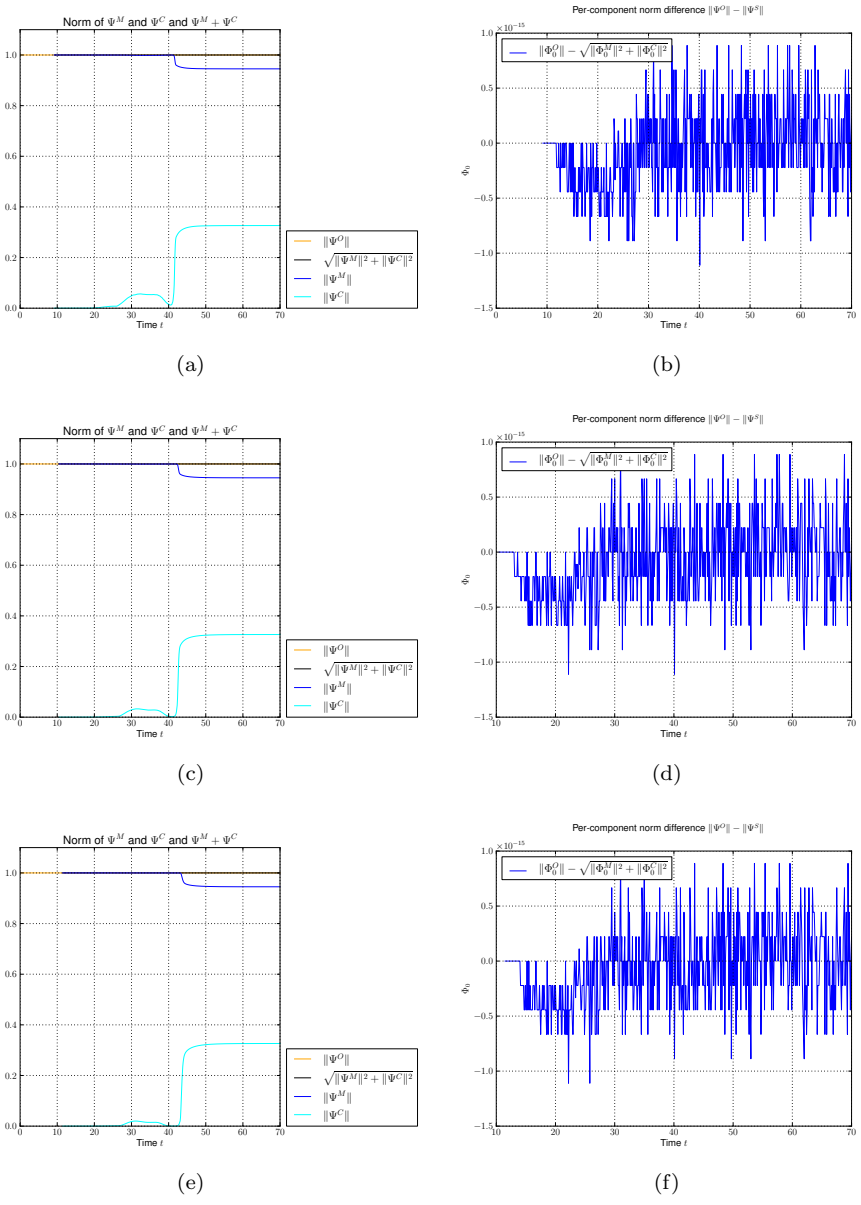


Figure 5.7: Norms and norm drift for an aposteriori spawning simulation based on the tunneling simulation of ϕ_0. We see perfect norm conservation which has to be expected because of the lumping process which is lossless. (a) $\Psi = \phi_0$ and $K_0 = 50$ (b) $\Psi = \phi_0$ and $K_0 = 50$ (c) $\Psi = \phi_0$ and $K_0 = 75$ (d) $\Psi = \phi_0$ and $K_0 = 75$ (e) $\Psi = \phi_0$ and $K_0 = 100$ (f) $\Psi = \phi_0$ and $K_0 = 100$

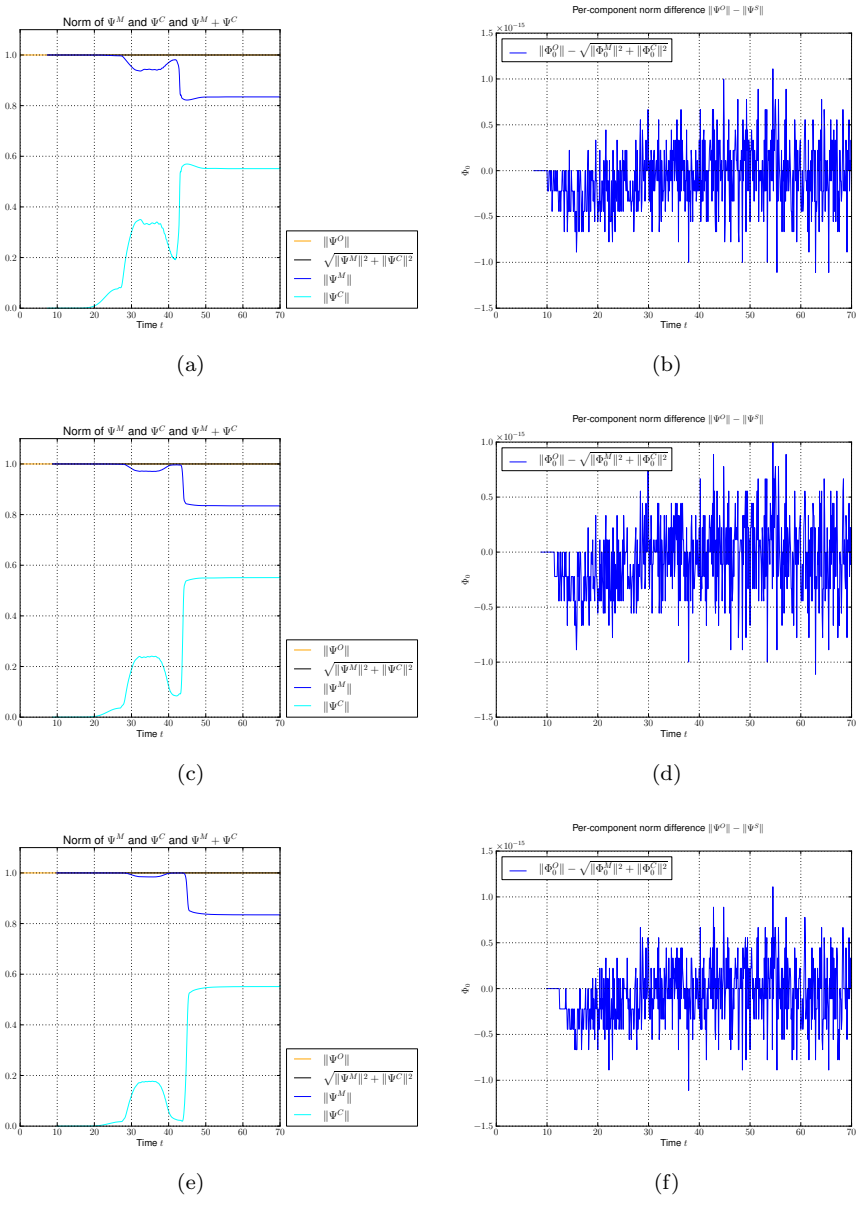


Figure 5.8: Norms and norm drift for an aposteriori spawning simulation based on the tunneling simulation of ϕ_2. We see perfect norm conservation which has to be expected because of the lumping process which is lossless. (a) $\Psi = \phi_2$ and $K_0 = 50$ (b) $\Psi = \phi_2$ and $K_0 = 50$ (c) $\Psi = \phi_2$ and $K_0 = 75$ (d) $\Psi = \phi_2$ and $K_0 = 75$ (e) $\Psi = \phi_2$ and $K_0 = 100$ (f) $\Psi = \phi_2$ and $K_0 = 100$

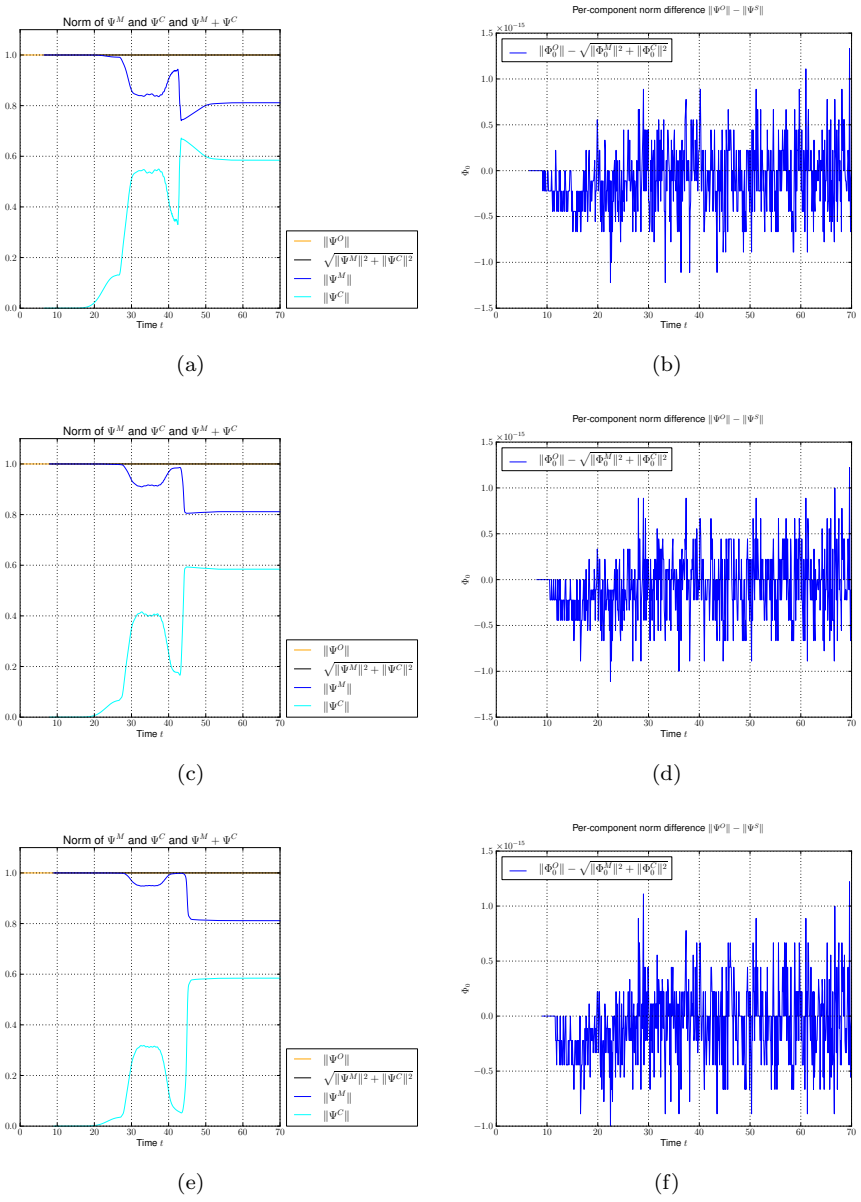
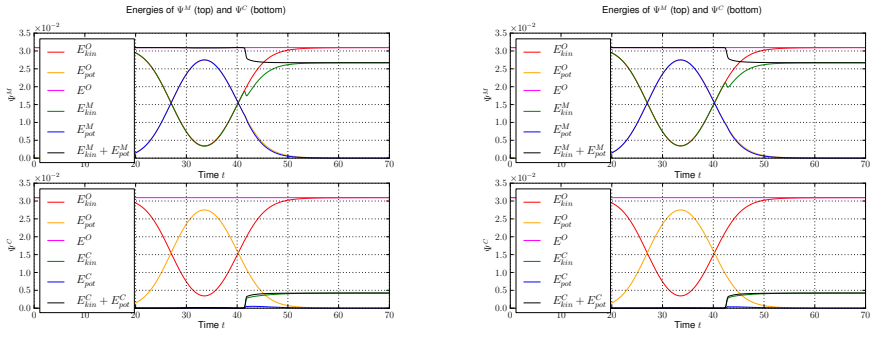
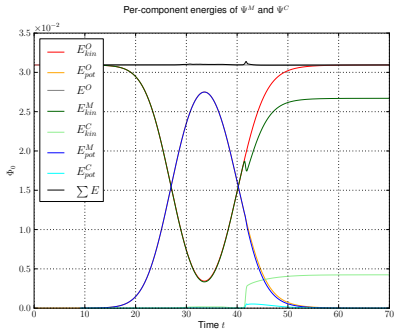


Figure 5.9: Norms and norm drift for an aposteriori spawning simulation based on the tunneling simulation of ϕ_3. We see perfect norm conservation which has to be expected because of the lumping process which is lossless. (a) $\Psi = \phi_3$ and $K_0 = 50$ (b) $\Psi = \phi_3$ and $K_0 = 50$ (c) $\Psi = \phi_3$ and $K_0 = 75$ (d) $\Psi = \phi_3$ and $K_0 = 75$ (e) $\Psi = \phi_3$ and $K_0 = 100$ (f) $\Psi = \phi_3$ and $K_0 = 100$

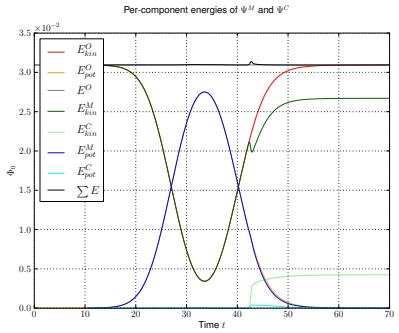


(a)

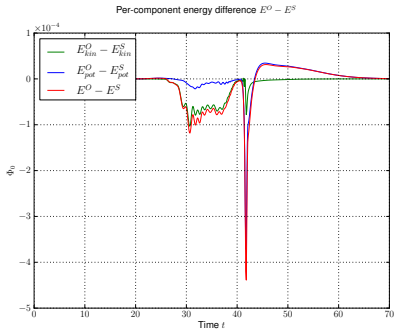
(b)



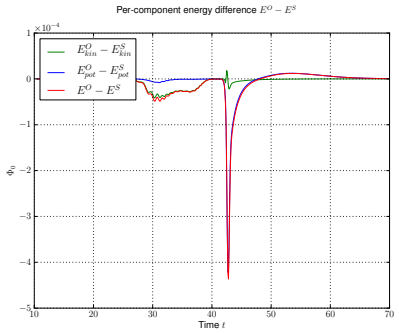
(c)



(d)



(e)



(f)

Figure 5.10: Energies and energy drift for an aposteriori spawning simulation. We try different initial for $|\Psi\rangle$ and several values for K_0. Energy conservation for $\Psi = \phi_0$ is by far good enough but not perfect. (a) $\Psi = \phi_0$ and $K_0 = 50$ (b) $\Psi = \phi_0$ and $K_0 = 75$ (c) $\Psi = \phi_0$ and $K_0 = 50$ (d) $\Psi = \phi_0$ and $K_0 = 75$ (e) $\Psi = \phi_0$ and $K_0 = 50$ (f) $\Psi = \phi_0$ and $K_0 = 75$

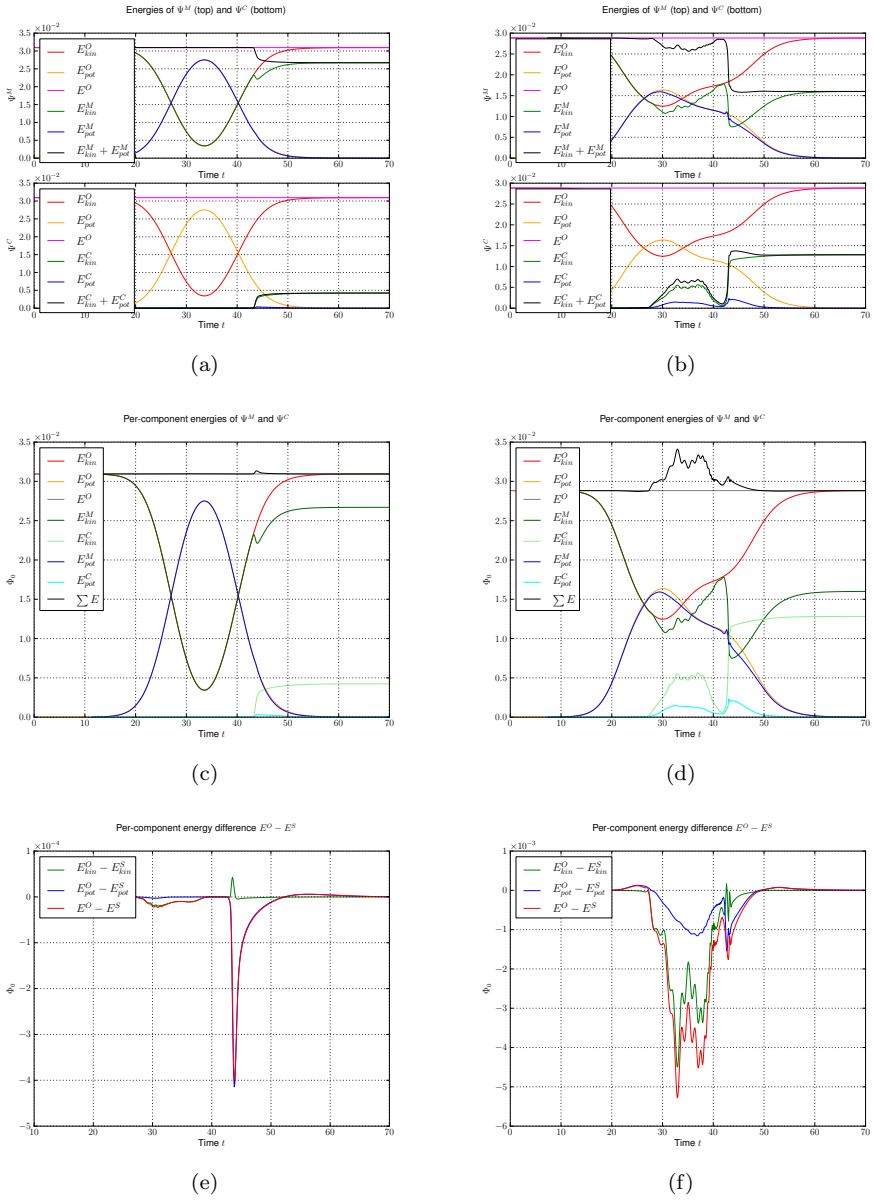
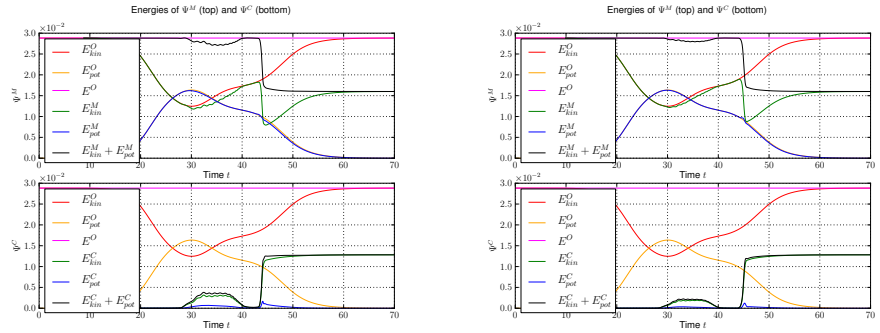
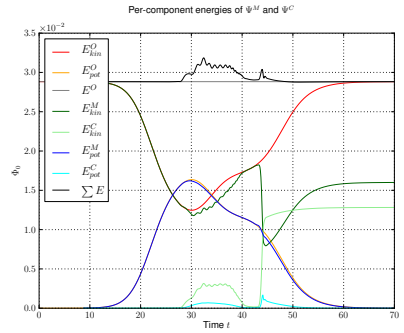


Figure 5.11: Energies and energy drift for an aposteriori spawning simulation. We try different initial for $|\Psi\rangle$ and several values for K_0. Energy conservation for $\Psi = \phi_0$ is by far good enough but not perfect. (a) $\Psi = \phi_0$ and $K_0 = 100$ (b) $\Psi = \phi_2$ and $K_0 = 50$ (c) $\Psi = \phi_0$ and $K_0 = 100$ (d) $\Psi = \phi_2$ and $K_0 = 50$ (e) $\Psi = \phi_0$ and $K_0 = 100$ (f) $\Psi = \phi_2$ and $K_0 = 50$

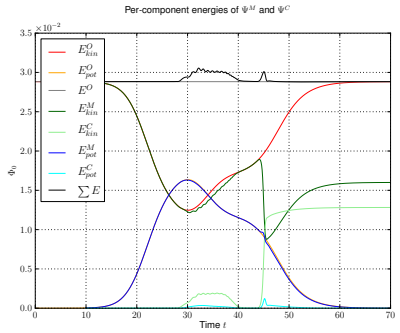


(a)

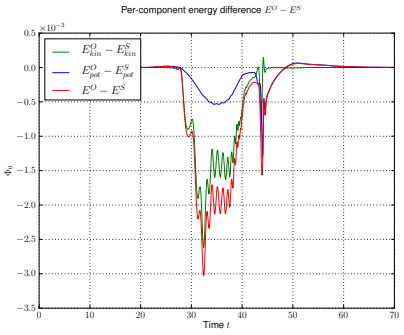
(b)



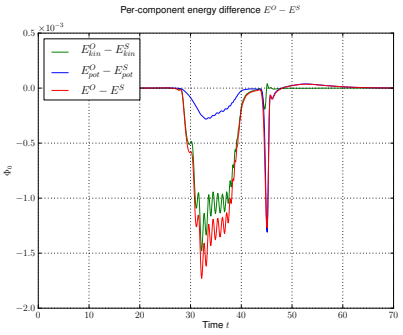
(c)



(d)

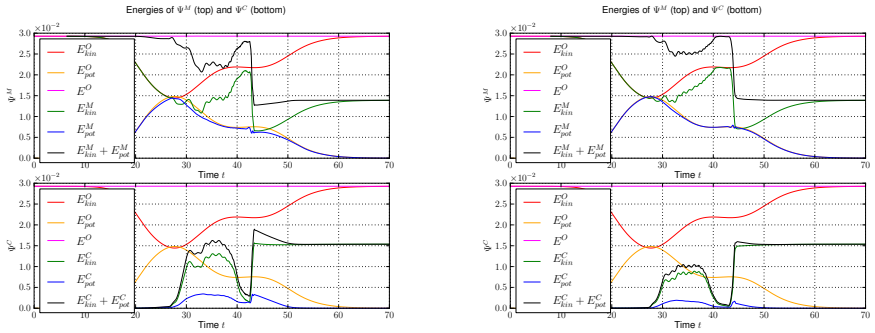


(e)



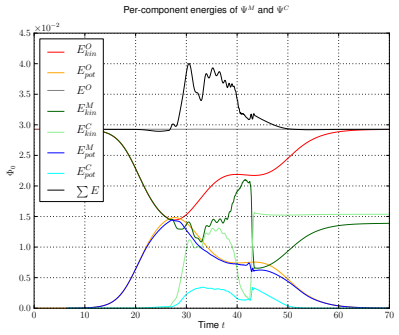
(f)

Figure 5.12: Energies and energy drift for an aposteriori spawning simulation. We try different initial for $|\Psi\rangle$ and several values for K_0. Energy conservation is only fulfilled after the tunneling event happened and tends to get better with higher K_0. (a) $\Psi = \phi_2$ and $K_0 = 75$ (b) $\Psi = \phi_2$ and $K_0 = 100$ (c) $\Psi = \phi_2$ and $K_0 = 75$ (d) $\Psi = \phi_2$ and $K_0 = 100$ (e) $\Psi = \phi_2$ and $K_0 = 75$ (f) $\Psi = \phi_2$ and $K_0 = 100$

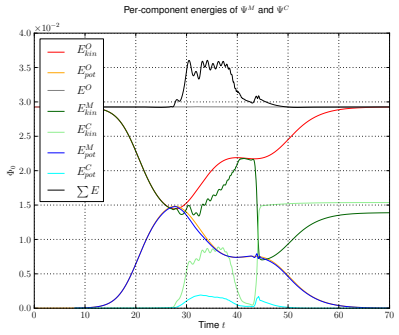


(a)

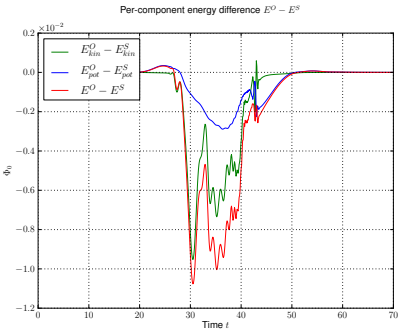
(b)



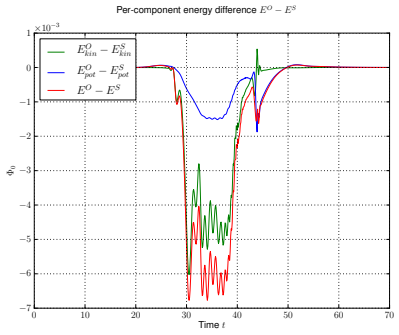
(c)



(d)

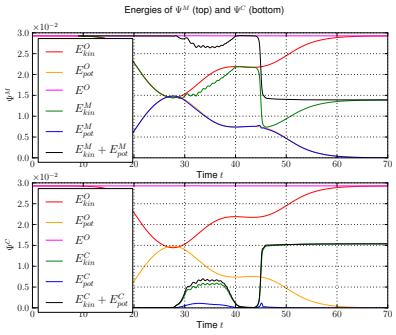


(e)

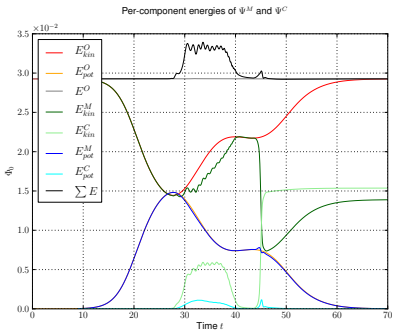


(f)

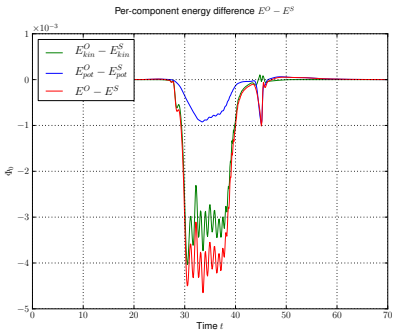
Figure 5.13: Energies and energy drift for an aposteriori spawning simulation. We try different initial for $|\Psi\rangle$ and several values for K_0. Energy conservation is only fulfilled after the tunneling event happened and tends to get better with higher K_0. (a) $\Psi = \phi_3$ and $K_0 = 50$ (b) $\Psi = \phi_3$ and $K_0 = 75$ (c) $\Psi = \phi_3$ and $K_0 = 50$ (d) $\Psi = \phi_3$ and $K_0 = 75$ (e) $\Psi = \phi_3$ and $K_0 = 50$ (f) $\Psi = \phi_3$ and $K_0 = 75$



(a)



(b)



(c)

Figure 5.14: Energies and energy drift for an aposteriori spawning simulation. We try different initial for $|\Psi\rangle$ and several values for K_0. Energy conservation is only fulfilled after the tunneling event happened and tends to get better with higher K_0. (a) $\Psi = \phi_3$ and $K_0 = 100$ (b) $\Psi = \phi_3$ and $K_0 = 100$ (c) $\Psi = \phi_3$ and $K_0 = 100$

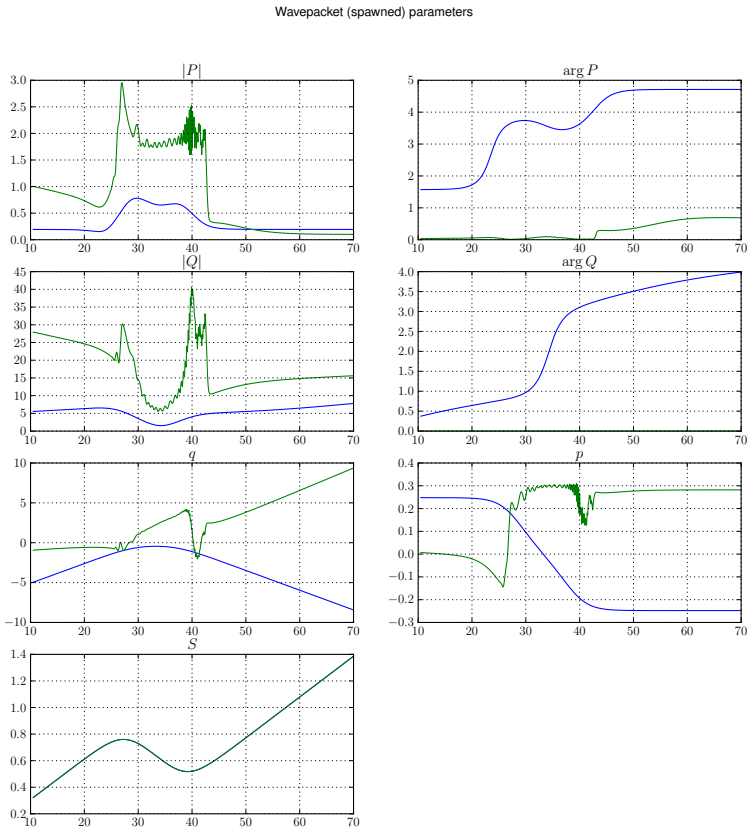


Figure 5.15: The parameter sets Π (blue) and $\tilde{\Pi}$ (green) estimated from the simulation with ϕ_0 plotted versus time. The value of K_0 was set to 75.

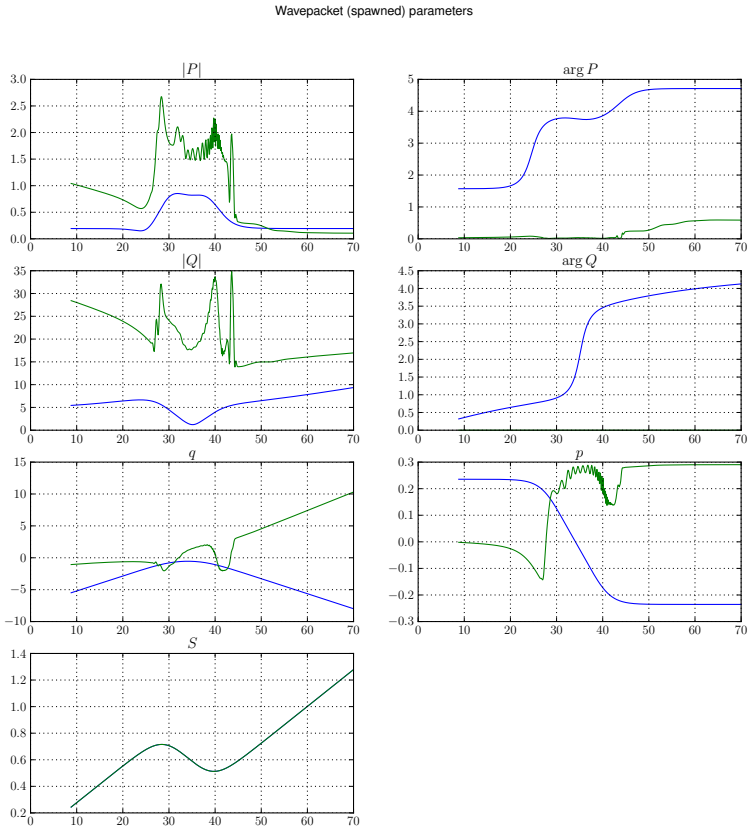


Figure 5.16: The parameter sets Π (blue) and $\tilde{\Pi}$ (green) estimated from the simulation with ϕ_2 plotted versus time. The value of K_0 was set to 75. If we compare this to figure 5.15 we see that the green curves differ much in general but little in the important region for t larger than about 45.

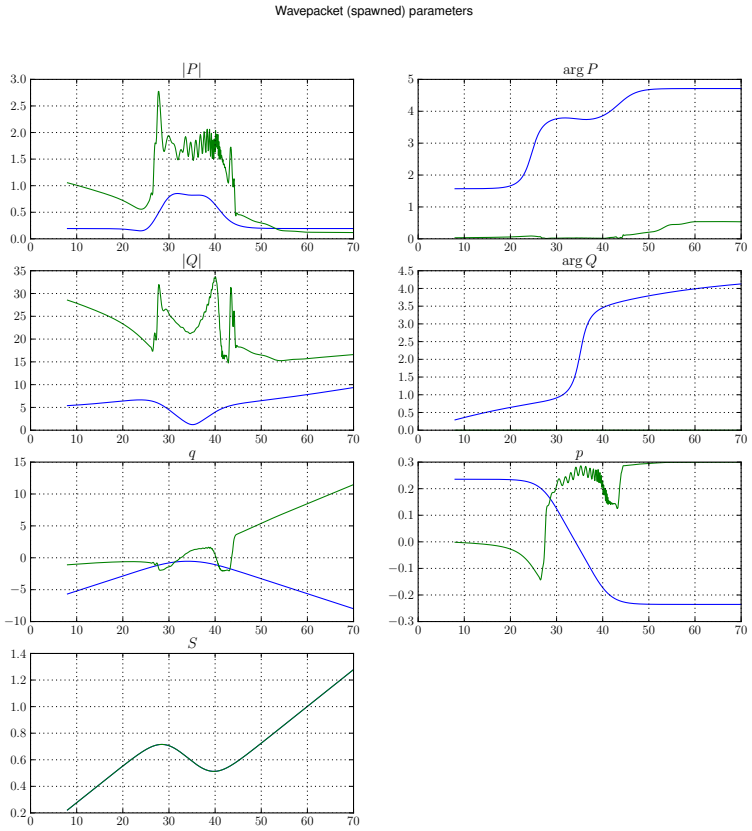
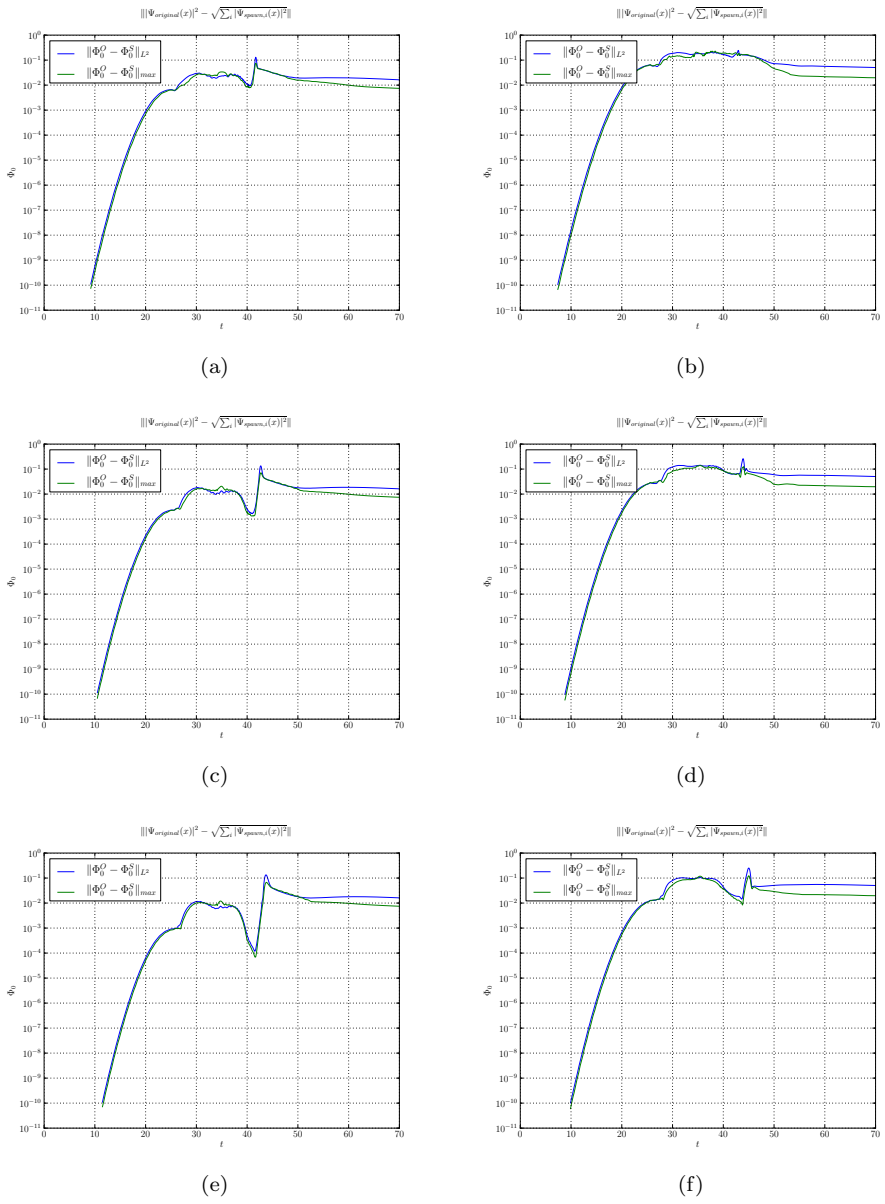
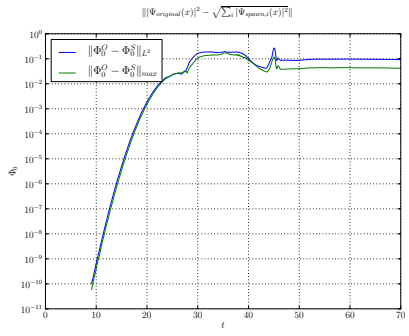


Figure 5.17: The parameter sets Π (blue) and $\tilde{\Pi}$ (green) estimated from the simulation with ϕ_3 plotted versus time. The value of K_0 was set to 75. If we compare this to figure 5.15 we see that the green curves differ much in general but little in the important region for t larger than about 45.

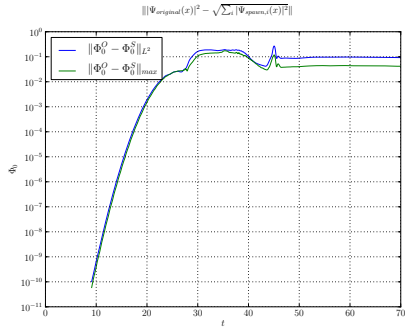


(a) (b) (c) (d) (e) (f)

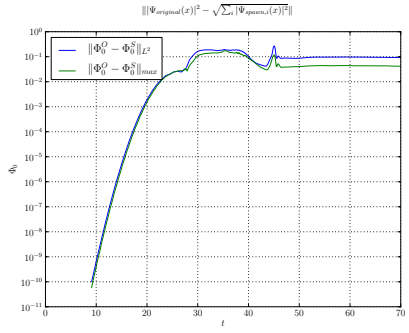
Figure 5.18: The figure shows the spawning error in the L^2 and maximum norm for various initial values ϕ_i and different values of K_0. We see that the choice of K_0 is rather unimportant. (a) $\Psi = \phi_0$ and $K_0 = 50$ (b) $\Psi = \phi_2$ and $K_0 = 50$ (c) $\Psi = \phi_0$ and $K_0 = 75$ (d) $\Psi = \phi_2$ and $K_0 = 75$ (e) $\Psi = \phi_0$ and $K_0 = 100$ (f) $\Psi = \phi_2$ and $K_0 = 100$



(a)



(b)



(c)

Figure 5.19: The figure shows the spawning error in the L^2 and maximum norm for various initial values ϕ_i and different values of K_0. We see that the choice of K_0 is rather unimportant. (a) $\Psi = \phi_3$ and $K_0 = 50$ (b) $\Psi = \phi_3$ and $K_0 = 75$ (c) $\Psi = \phi_3$ and $K_0 = 100$

5.5 Spawning using the projection method

In this section we show several results of the projection method used for the change of basis during the spawning process. All results take the tunneling simulation with a ϕ_0 as the base and set $K_0 = 50$ for parameter estimation. We then try to project on differently many basis functions $\{\tilde{\phi}_k\}_{k=0}^{\mu-1}$. In general we see an increase in the goodness of all properties with larger μ. The results are also better than what we got with the lumping method in the last section. This is exceedingly true for short times right after tunneling happened because at this time the transmitted part has not yet the shape of a Gaussian. This then shows up for larger times and holds only asymptotically. Hence we can improve the results if we drop the assumption that the transmitted part is always a Gaussian.

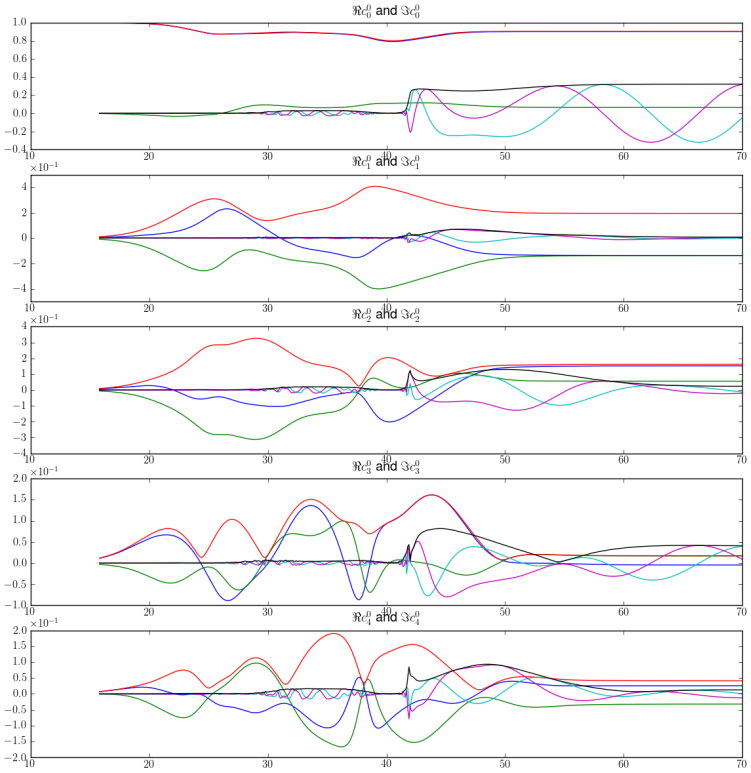


Figure 5.20: The figure shows the first five coefficients c_i of the original and $\tilde{c}_i$ of the spawned packet. The red and black curves are the absolute values, the others are real and imaginary parts. If we project for example only on the first three basis functions, then the black lines for $\tilde{c}_3$ and $\tilde{c}_4$ would be identical zero.

One conclusion of this section is that the lumping method works (and is also very fast to compute) but the projection method is superior in most applications.

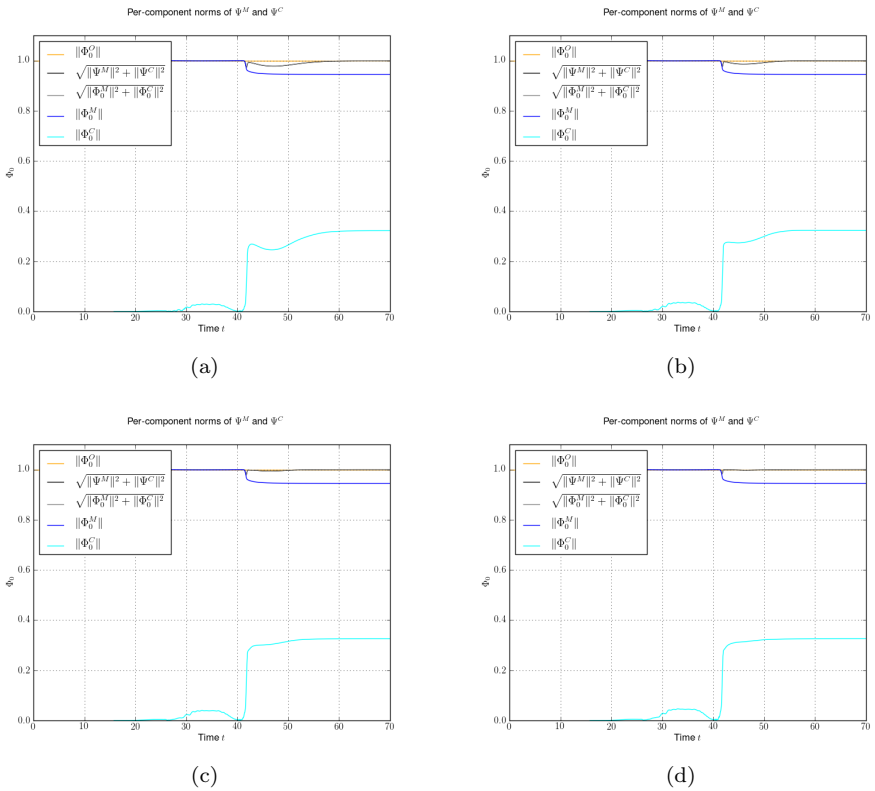


Figure 5.21: The panels show the norms of the original wavepacket (suffix O), the spawned wavepacket (suffix C) and the remainder packet (suffix M). We find increasing norm conservation (black curve is overall energy) during the spawning process for increasing values of the projection spawning parameter μ. (This parameter is the number of basis functions we project onto.) This is a difference to the spawning using lumping where we always have perfect norm conservation. (a) Projection spawning parameter $\mu = 1$. (b) Projection spawning parameter $\mu = 3$. (c) Projection spawning parameter $\mu = 6$. (d) Projection spawning parameter $\mu = 12$.

The reason for this is that we seldom have a pure basis function ϕ_k as the master fragment for which spawning is tried. And even for very slight deviations from the pure form the projection method is better. (Remember that although we theoretically expect a Gaussian after tunneling this result holds only for asymptotically large times. Contrary to that our simulation takes place in short up to moderately long times after the tunneling event.) For example this also happens in simulations where the mother fragment has roughly the shape of a ϕ_2 but with one of its peaks is higher than the others.

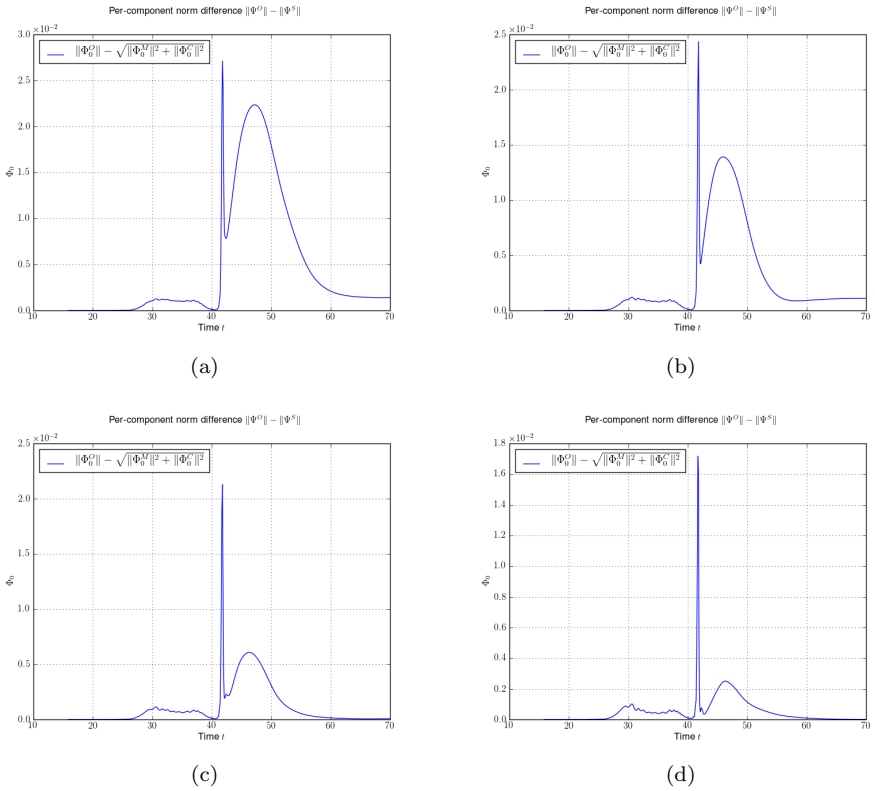


Figure 5.22: The panels show the difference of the norms of the original wavepacket (suffix O), the spawned wavepacket (suffix C) and the remainder packet (suffix M). We find increasing norm conservation for larger values of the projection spawning parameter μ. (This parameter is the number of basis functions we project onto.) This is a difference to the spawning using lumping where we always have perfect norm conservation. (a) Projection spawning parameter $\mu = 1$. (b) Projection spawning parameter $\mu = 3$. (c) Projection spawning parameter $\mu = 6$. (d) Projection spawning parameter $\mu = 12$.

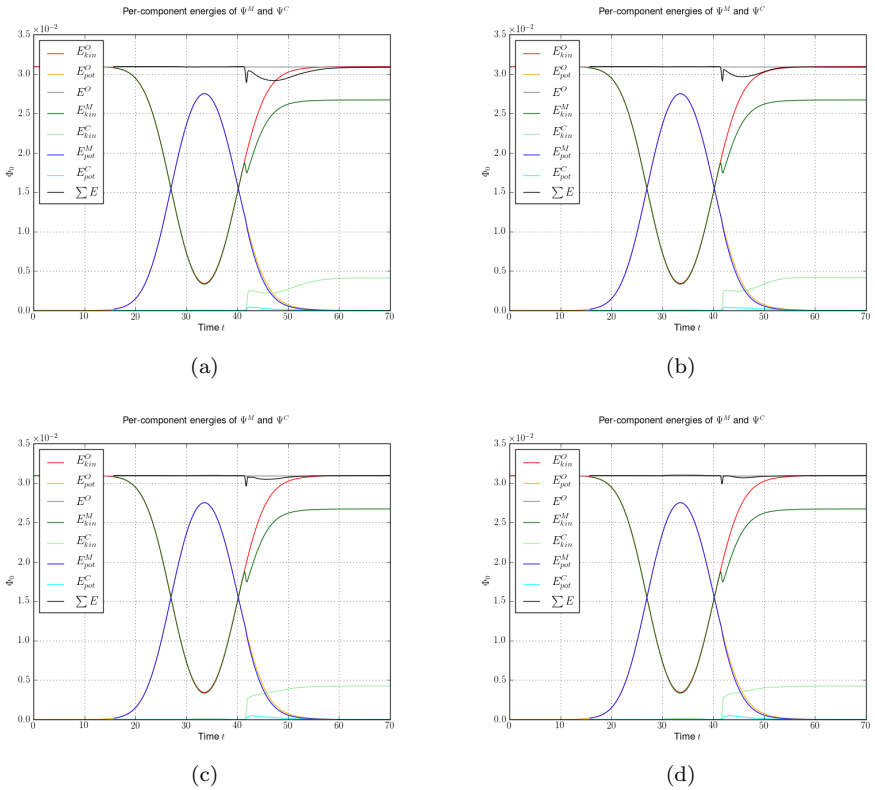


Figure 5.23: The panels show the kinetic and potential energies of the original wavepacket (suffix O), the spawned wavepacket (suffix C) and the remainder packet (suffix M). We find increasing energy conservation (black curve is overall energy) during the spawning process for increasing values of the projection spawning parameter μ. (This parameter is the number of basis functions we project onto.) (a) Projection spawning parameter $\mu = 1$. (b) Projection spawning parameter $\mu = 3$. (c) Projection spawning parameter $\mu = 6$. (d) Projection spawning parameter $\mu = 12$.

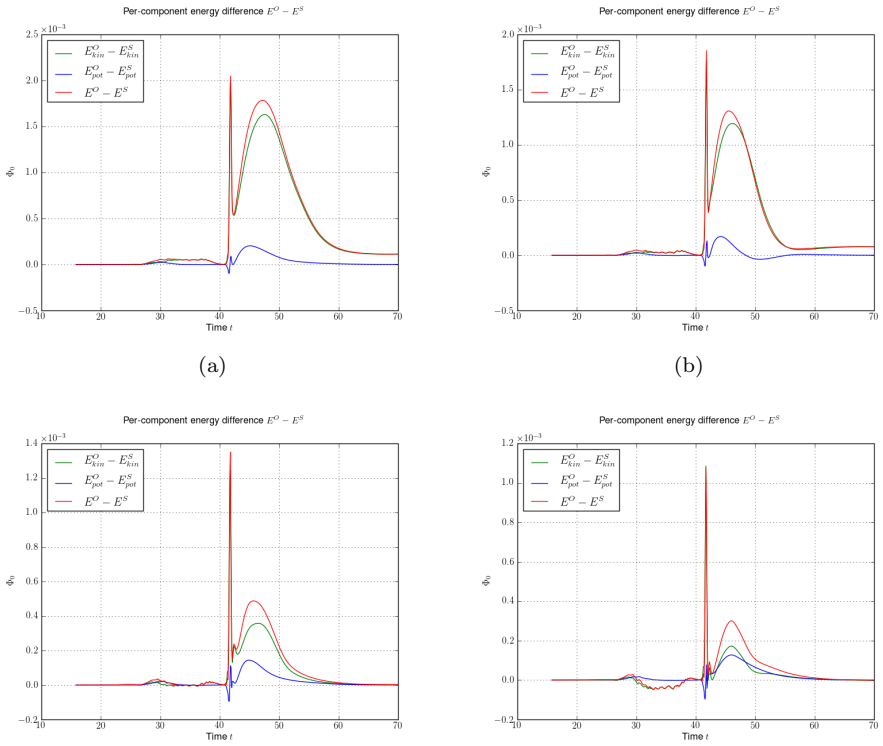
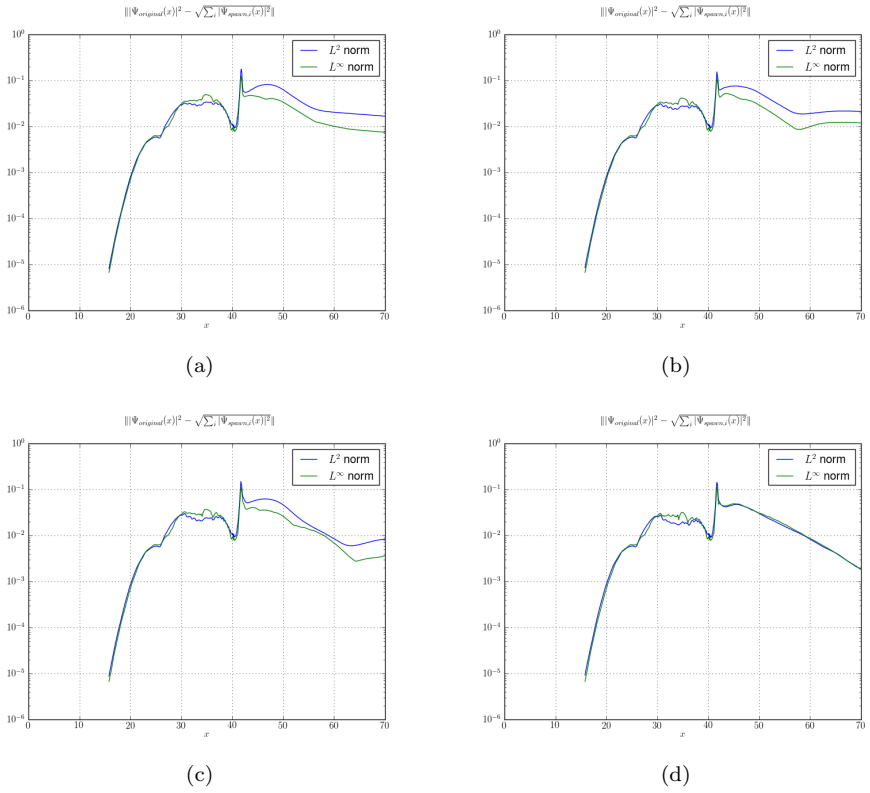


Figure 5.24: The panels show the difference of the kinetic and potential energies between the original wavepacket (suffix O), the spawned wavepacket (suffix C) and the remainder packet (suffix M). We find increasing energy conservation for larger values of the projection spawning parameter μ (This parameter is the number of basis functions we project onto.) (a) Projection spawning parameter $\mu = 1$. (b) Projection spawning parameter $\mu = 3$. (c) Projection spawning parameter $\mu = 6$. (d) Projection spawning parameter $\mu = 12$.



(a) (b)

(c) (d)

Figure 5.25: The panels show the norm of the spawn error measured in L^2 and maximum norm. We see that we can reduce the error with increasing projection spawning parameter μ. (This parameter is the number of basis functions we project onto.) (a) Projection spawning parameter $\mu = 1$. (b) Projection spawning parameter $\mu = 3$. (c) Projection spawning parameter $\mu = 6$. (d) Projection spawning parameter $\mu = 12$.

5.6 Spawning and propagation

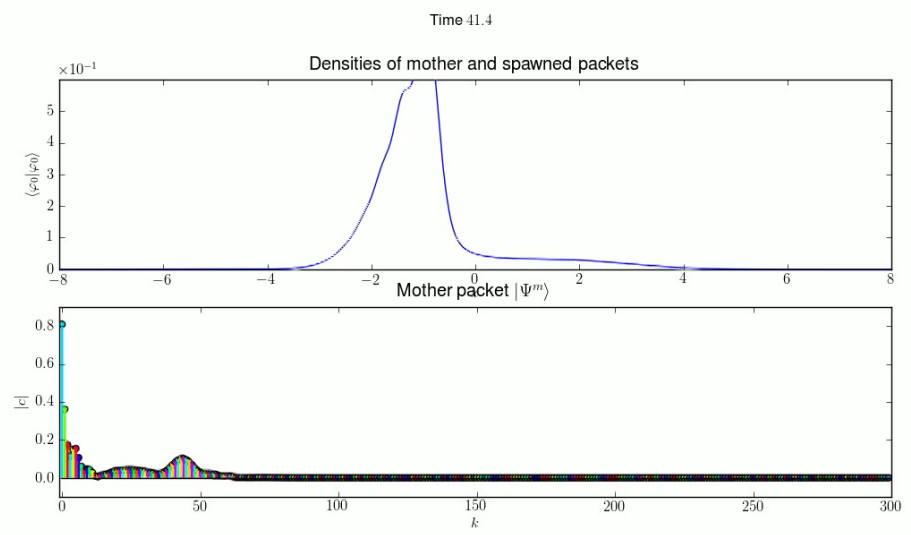
In the last section we looked at many different examples of tunneling simulations. By now we have got an idea how spawning works in practice and saw that it works quite well although there are some difficulties and possibilities for improvement. As we wrote earlier the aposteriori spawning process is only good for gaining insight in the spawning process itself but gives no improvements otherwise. Hence we will investigate in this section how we can take the full advantage of spawning for better and faster simulations.
The key point is that we in principle do the time propagation as usual but now and then check if some condition is fulfilled which then triggers spawning. After a spawning event occurred we proceed with the normal time propagation again but now we have one more packet to propagate. We should stress that spawning events are very rare. And thus there is little risk of ending up with dozens of packets. We call this combined algorithm the *spawning propagation method*.
The criterion for spawning is difficult to choose. There are many possibilities, some of them work well in theory but are problematic to implement and vice versa. The most simple one is probably to introduce a threshold τ and monitor the norms of the mother fragments w which are candidates for spawning. If then the norm $\langle w \,|\, w \rangle$ gets larger than τ this triggers the spawning process once.
We concentrate on this criterion because it is simple both to understand and to implement. But we will see later in the examples that there are some pitfalls.
The algorithm 8 shows the course layout of the concepts described in the paragraph above. It is a simplified version in the sense that we start with a single wavepacket $|\Psi\rangle$ and presume we will fulfill the spawning criterion only once. Hence we will end up with only two wavepackets $|\Psi_0\rangle$ and $|\Psi_1\rangle$. For both of these the criterion will never be true again. To see this also compare to the images 5.26, 5.27 and 5.28. The parts on propagation left out and can be found in the references given, copying over and plugging in these algorithms is obvious. This is exactly the situation in a simple tunneling simulation, thus we can apply this algorithm to such a tunneling problem. For more advanced cases like avoided crossings we will need to extend the algorithm later on in the next chapters.

5.6.1 Spawning propagation using the lumping method

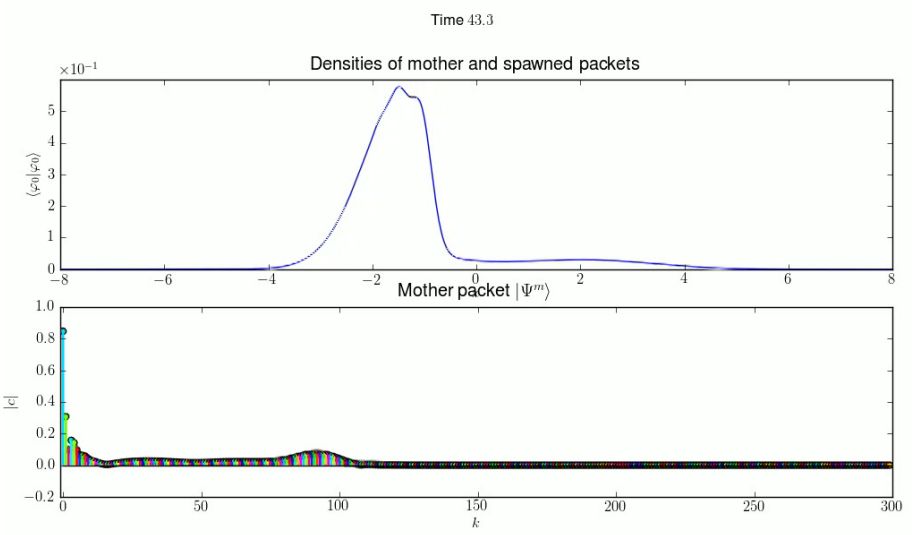
Now we turn back to the tunneling example and see how this works in real simulations. We show several simulations with increasing values of the spawning threshold τ. The simulations used the lumping method for the change to the new basis $\tilde{\Pi}$ thus we have perfect norm conservation. However the energy conservation is not always perfect but it becomes better the higher τ gets and thus the later we spawn. Later we will try the basis projection method too and compare to the lumping method.

Algorithm 8 Spawning propagation method (simplified)

Require: A series t consisting of timesteps $\{\tau_0, \ldots, \tau_{\max}\}$
Require: The potential $V(x)$
Require: The initial wavepacket $|\Psi(t = \tau_o)\rangle$ with its coefficients $\{c_k\}_{k=0}^{K-1}$
Require: The value of $0 \leq K_0 \leq K - 1$
Require: A spawn threshold τ
  for all $\tau_i \in t$ **do**
    // Decompose Ψ according to (5.1) with high-frequency part w of Ψ
    $v := \sum_{k=0}^{K_0-1} c_k \phi_k$
    $w := \sum_{k=K_0}^{K-1} c_k \phi_k$
    // Check the spawning criterion
    if $\langle w \,|\, w \rangle \geq \tau^2$ **then**
      // Perform spawning procedure
      // Estimate the parameters of the fragment by algorithm 4
      $\tilde{\Pi} :=$ **estimate_parameters**(w)
      // Move the fragment w to the new basis by either algorithm 5 or 6
      $\tilde{w} :=$ **change_basis**$(\tilde{\Pi}, w)$
      // Update the reminder (included in algorithm 5 and 6)
      $\tilde{v} :=$ **update_remainder**$(v, \tilde{w})$
      // We have now two new, full wavepackets
      $\Psi_0 := \tilde{v}$ and $\Psi_1 := \tilde{w}$
    end if
    // Time propagation of all Ψ_j using the algorithms from [2] and [1]
    for all $\Psi_j(t = \tau_i)$ **do**
      $\Psi_j(t = \tau_{i+1}) :=$ **time_propagation**$(V, \Psi_j(t = \tau_i))$
    end for
  end for

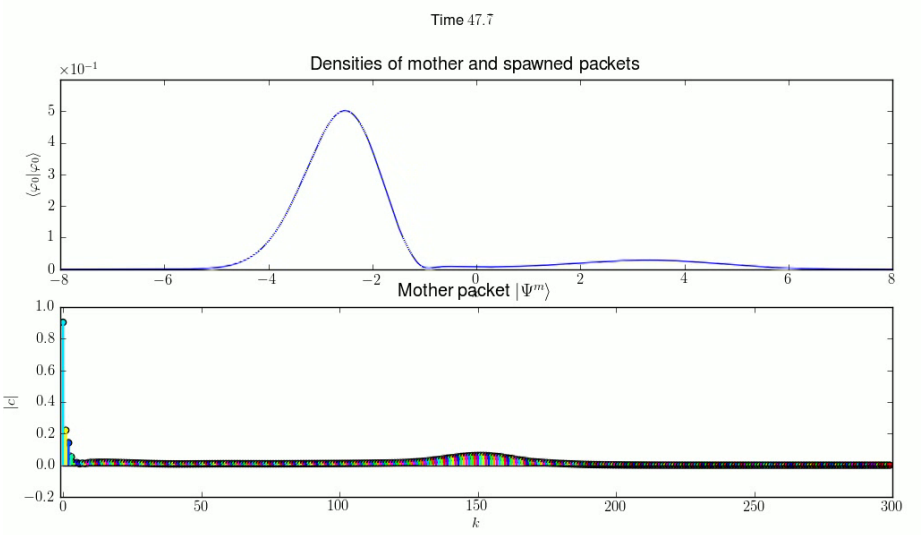


(a)

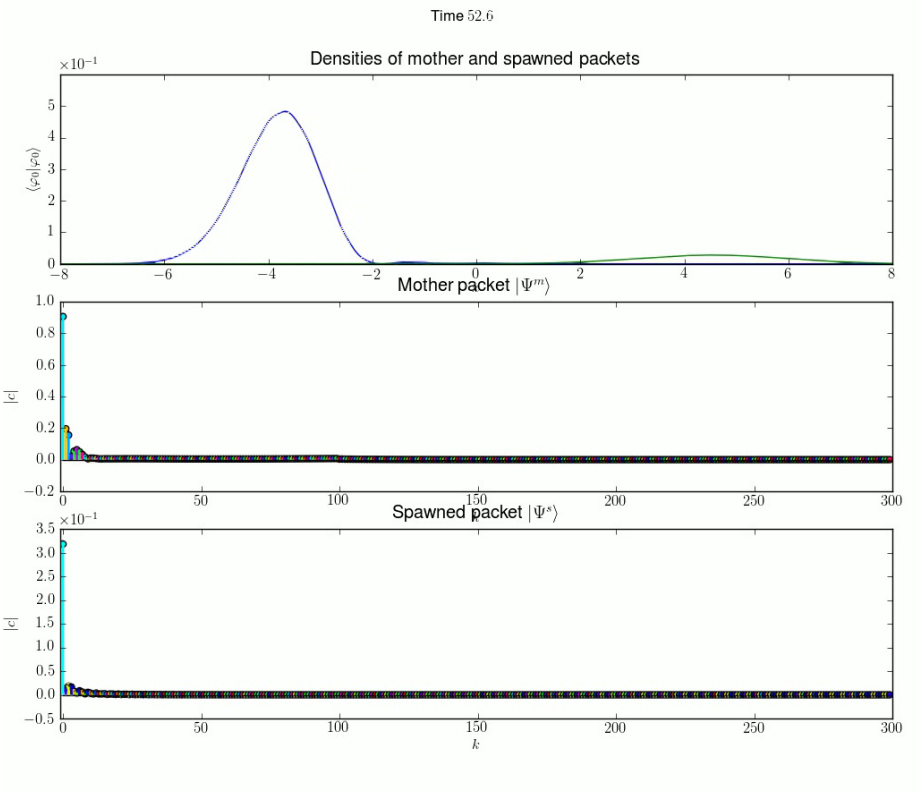


(b)

Figure 5.26: Tunneling simulation at different times before spawning happens. We see the high-frequency bulge in the coefficients.

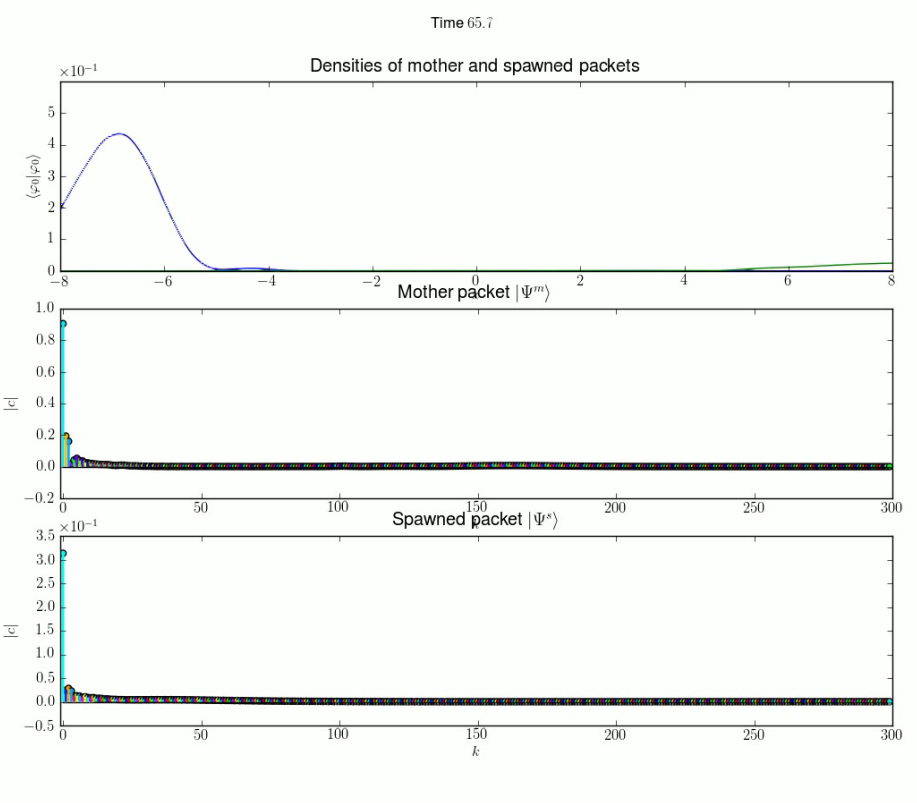


(a)



(b)

Figure 5.27: Tunneling simulation at different times right before spawning and shortly after spawning has happened. We see how the high-frequency bulge in the coefficients is gone and both new packets could be represented with a much smaller basis. About 50 basis functions for each packet should suffice after spawning while we needed at least 200 before spawning.



(a)

Figure 5.28: Tunneling simulation at late time after spawning. Both packets only need a relatively small basis.

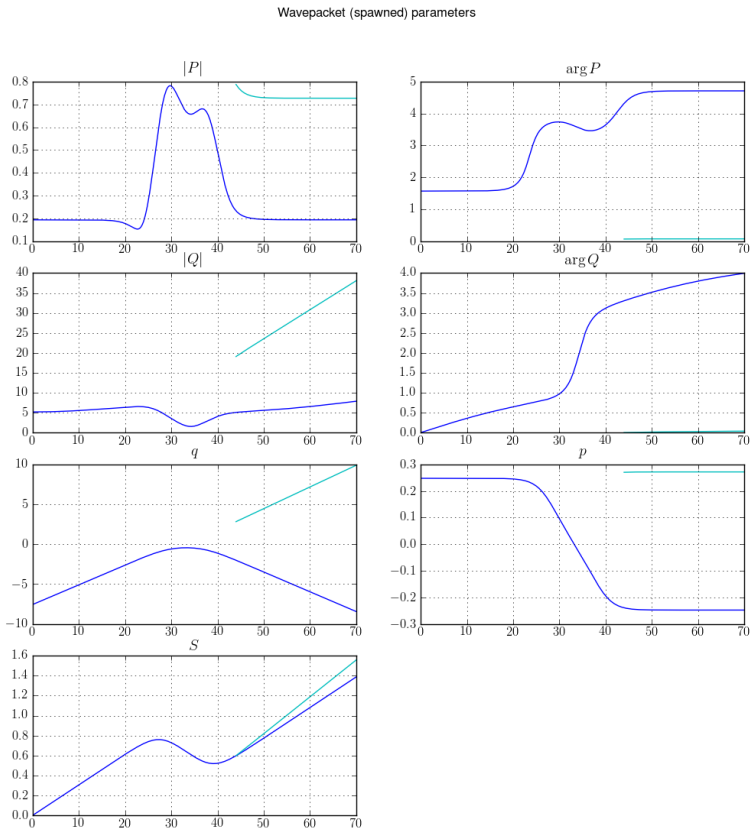
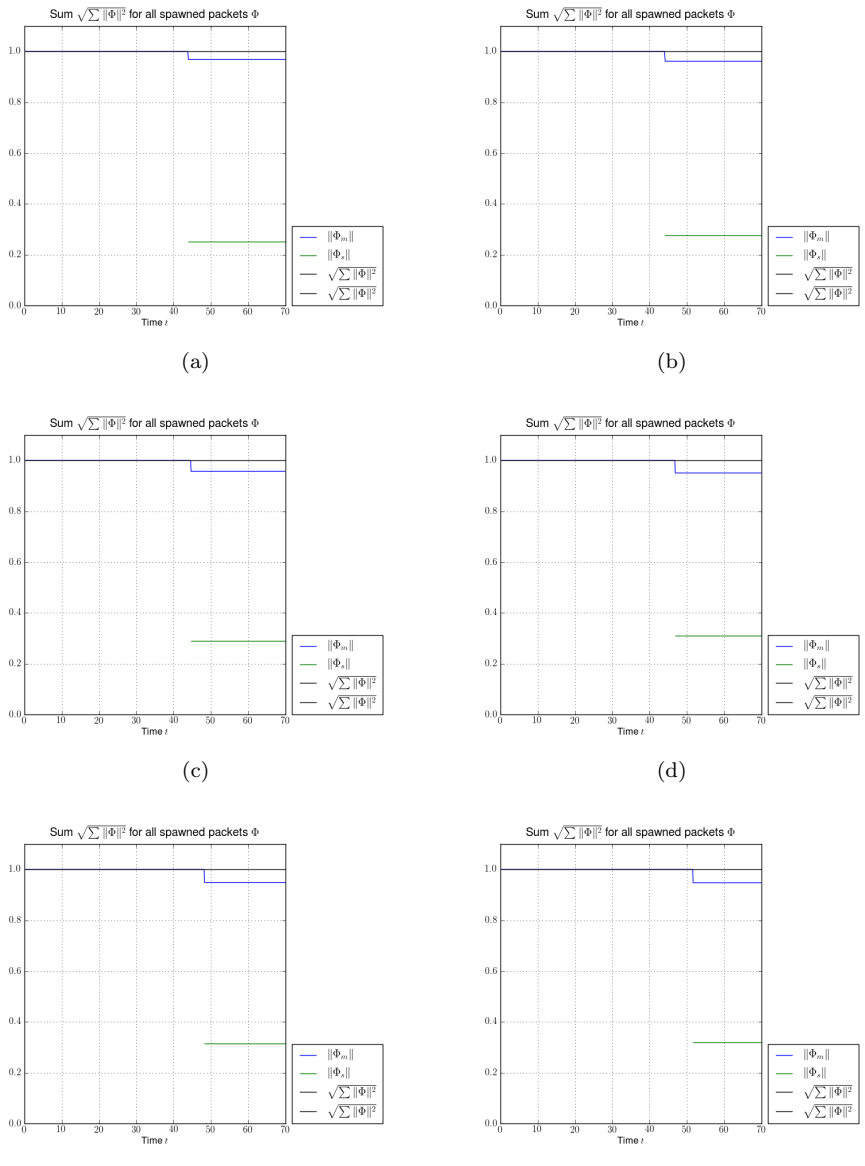


Figure 5.29: This figure shows the original parameter set Π (blue curves) and the estimated parameter set $\tilde{\Pi}$ (cyan curves) during the time propagation. We see that the parameters are estimated once and then smoothly propagated. For the position $\tilde{q}$ and momentum $\tilde{p}$ we can interpret the difference to the original parameters q and p nicely: while q gets reflected at the potential hill $\tilde{q}$ is transmitted. Same for the momentum: for the reflected part p becomes negative again while for the transmitted part $\tilde{p}$ stays positive corresponding to a wavepacket moving to the right. (The spawn threshold τ was 0.25 here but the parameter estimation is independent of this value.)



(a) (b) (c) (d) (e) (f)

Figure 5.30: These panels show the norms of the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the lumping method for the change of basis and thus we have perfect norm conservation. (f) Spawning threshold $\tau = 0.25$. (f) Spawning threshold $\tau = 0.275$. (f) Spawning threshold $\tau = 0.29$. (f) Spawning threshold $\tau = 0.31$. (f) Spawning threshold $\tau = 0.315$. (f) Spawning threshold $\tau = 0.32$.

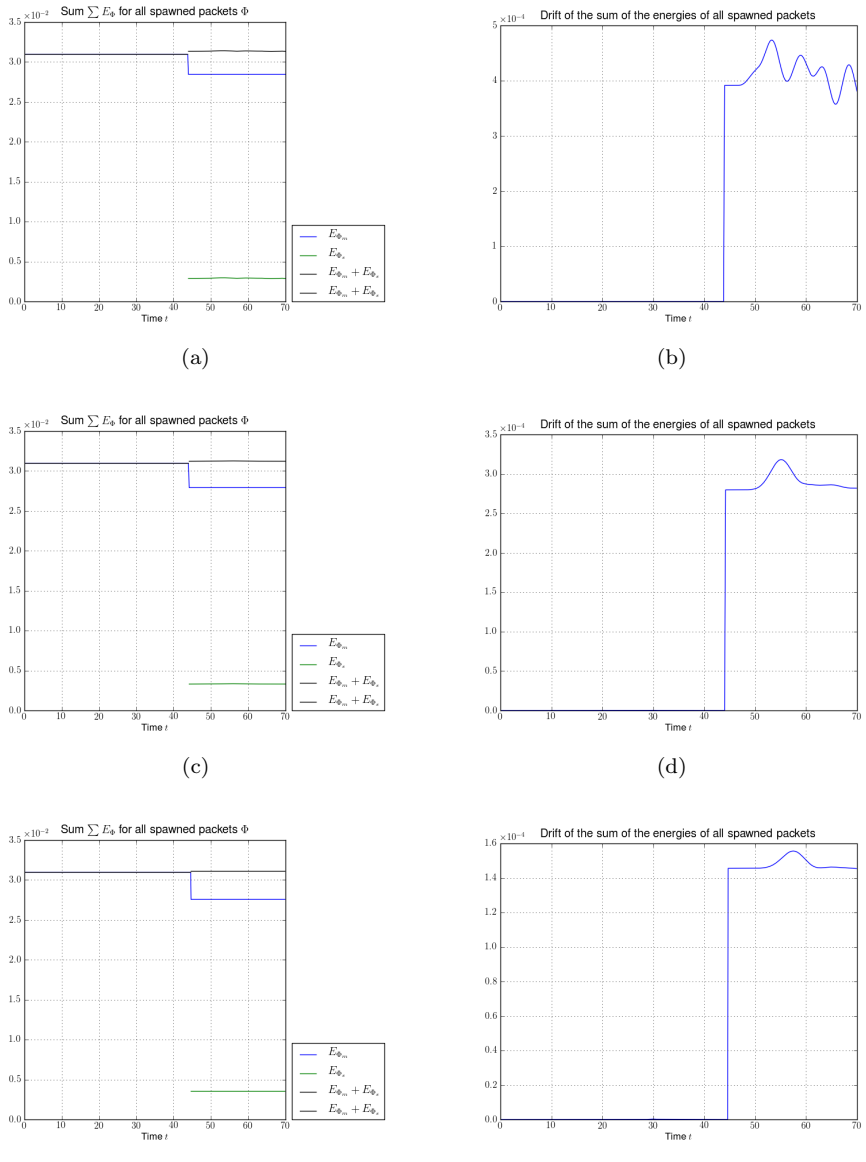
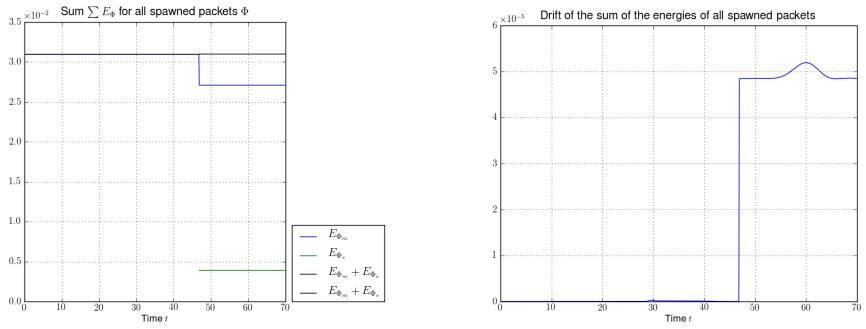
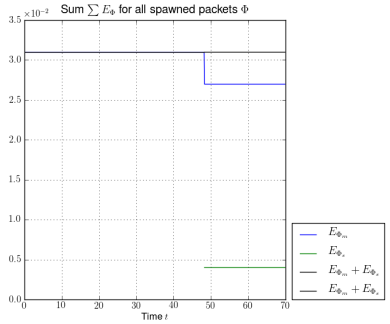


Figure 5.31: These panels show the energies and the error in energy conservation for the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. Notice the scales, and especially how the error in energy conservation drops from order 10^{-4} to order 10^{-6} (in the next panel). (a) Spawning threshold $\tau = 0.25$. (b) Spawning threshold $\tau = 0.25$. (c) Spawning threshold $\tau = 0.275$. (d) Spawning threshold $\tau = 0.275$. (e) Spawning threshold $\tau = 0.29$. (f) Spawning threshold $\tau = 0.29$.

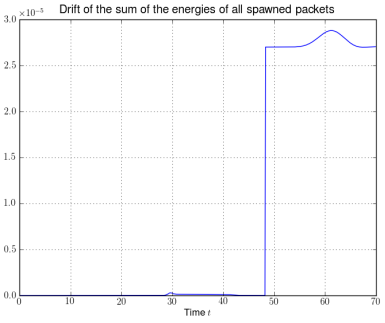


(a)

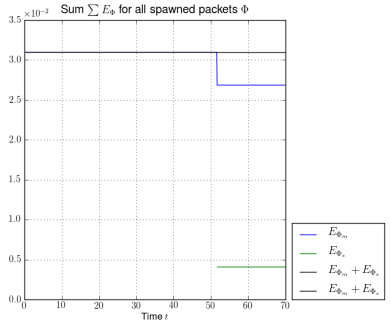
(b)



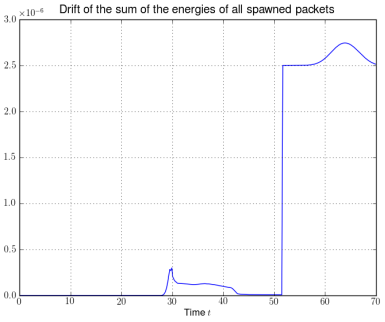
(c)



(d)



(e)



(f)

Figure 5.32: These panels show the energies and the error in energy conservation for the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. Notice the scales, and especially how the error in energy conservation drops from order 10^{-4} (in the last panel) to order 10^{-6}. (a) Spawning threshold $\tau = 0.31$. (b) Spawning threshold $\tau = 0.31$. (c) Spawning threshold $\tau = 0.315$. (d) Spawning threshold $\tau = 0.315$. (e) Spawning threshold $\tau = 0.32$. (f) Spawning threshold $\tau = 0.32$.

5.6.2 Spawning propagation using the projection method

In this section we show some simulation results for spawning propagation. But this time we use the projection method for the change of basis. The results are taken for several different values of μ, the number of basis functions we project onto. In difference to the lumping method and the results in the last section we do not have perfect norm conservation here. The energy conservation is also not perfect but gets better with an increasing value of μ. Contrary to the results from last section where the spawned wavepacket got more energy than it should, it gets less than what is necessary for energy conservation here.

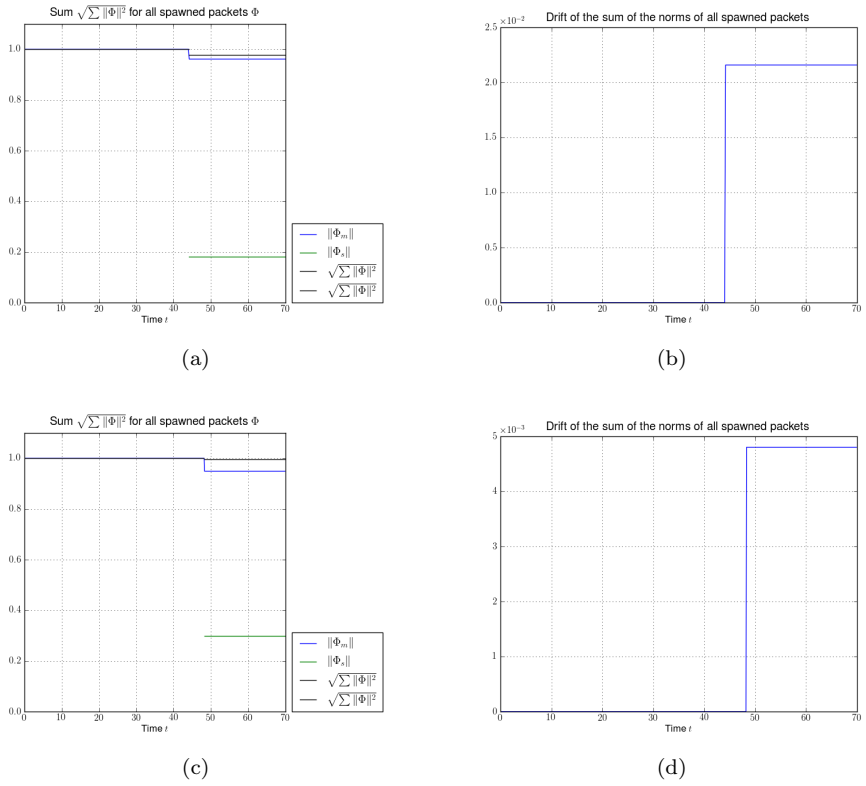
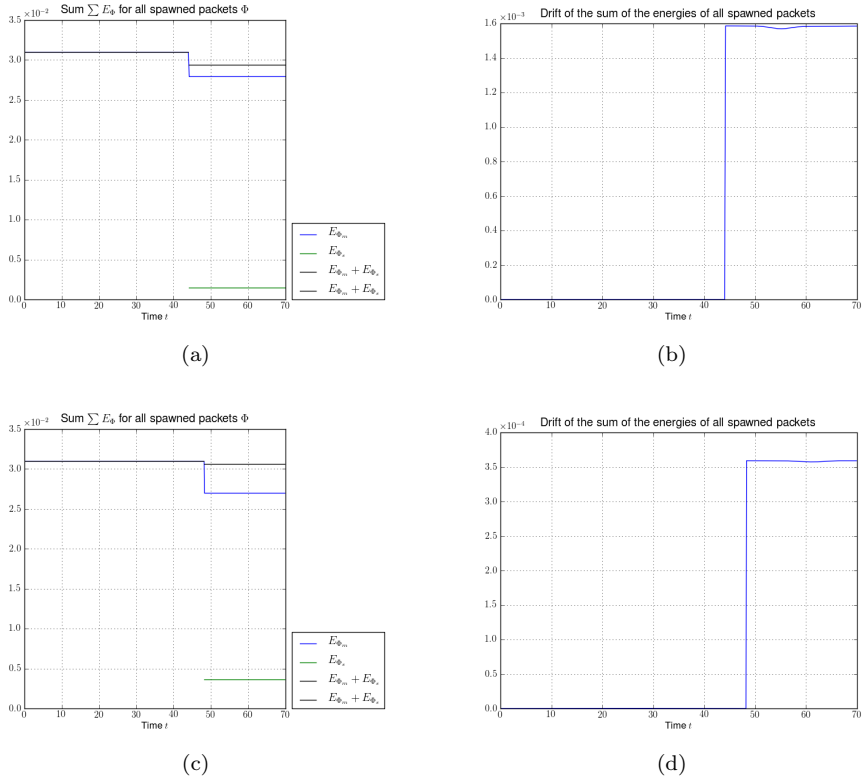


Figure 5.33: These panels show the norms of the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 1. (a) Spawning threshold $\tau = 0.275$. (b) Spawning threshold $\tau = 0.275$. (c) Spawning threshold $\tau = 0.315$. (d) Spawning threshold $\tau = 0.315$.



(a) (b)

(c) (d)

Figure 5.34: These panels show the energies and the error in energy conservation for the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 1. (a) Spawning threshold $\tau = 0.275$. (b) Spawning threshold $\tau = 0.275$. (c) Spawning threshold $\tau = 0.315$. (d) Spawning threshold $\tau = 0.315$.

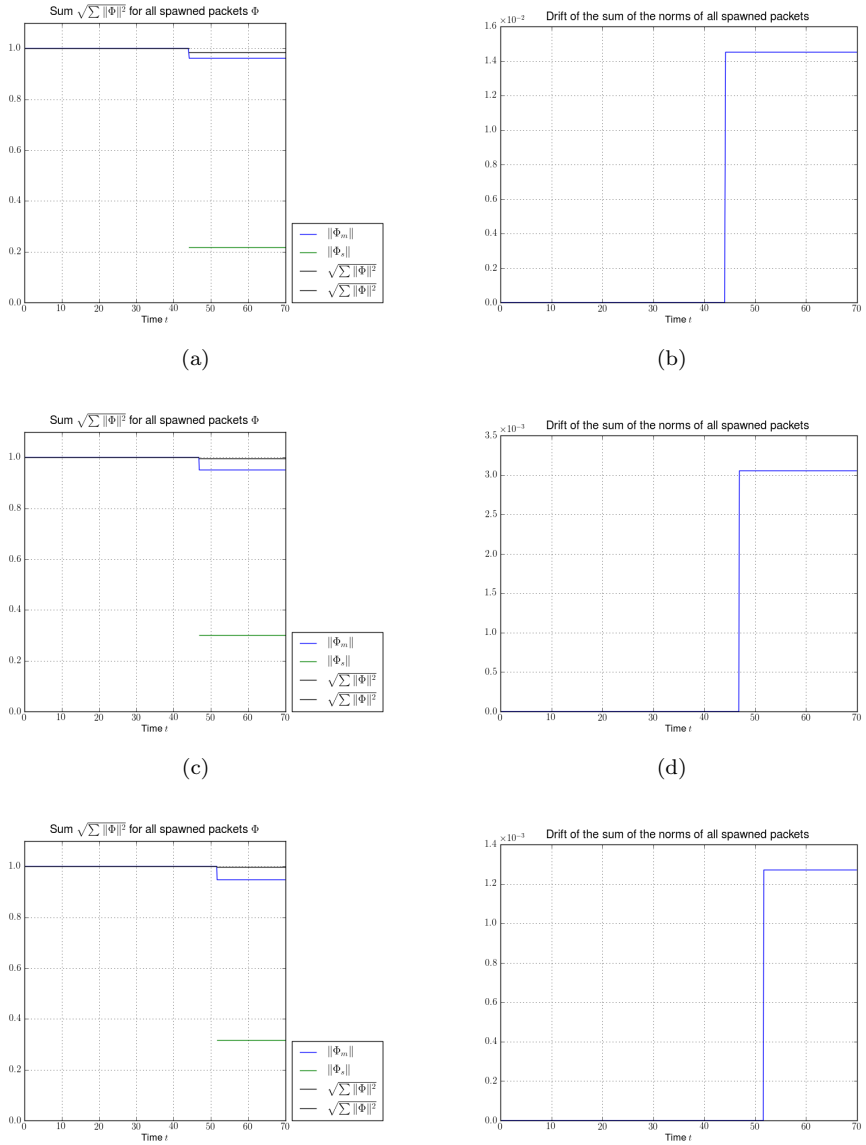
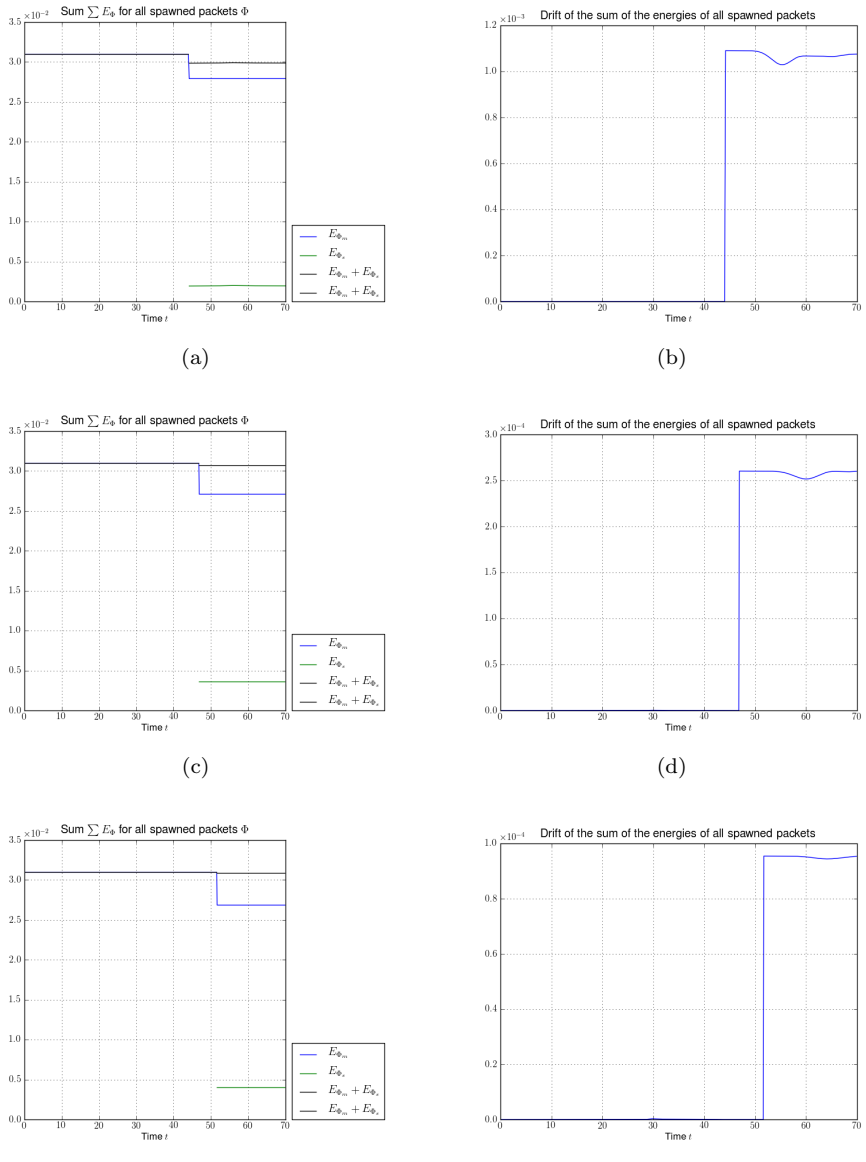


Figure 5.35: These panels show the norms of the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 3. (a) Spawning threshold $\tau = 0.275$. (b) Spawning threshold $\tau = 0.275$. (c) Spawning threshold $\tau = 0.31$. (d) Spawning threshold $\tau = 0.31$. (e) Spawning threshold $\tau = 0.32$. (f) Spawning threshold $\tau = 0.32$.



(a) (b)

(c) (d)

(e) (f)

Figure 5.36: These panels show the energies and the error in energy conservation for the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 3. (a) Spawning threshold $\tau = 0.275$. (b) Spawning threshold $\tau = 0.275$. (c) Spawning threshold $\tau = 0.31$. (d) Spawning threshold $\tau = 0.31$. (e) Spawning threshold $\tau = 0.32$. (f) Spawning threshold $\tau = 0.32$.

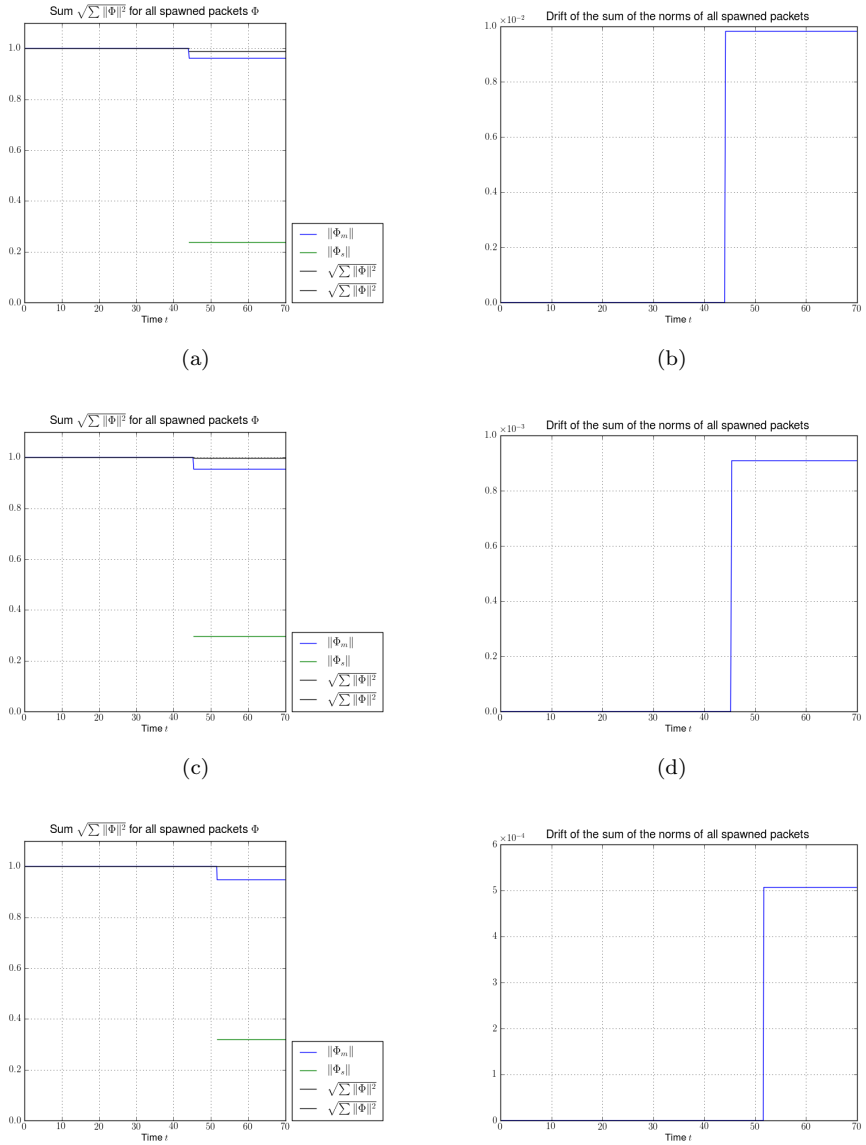
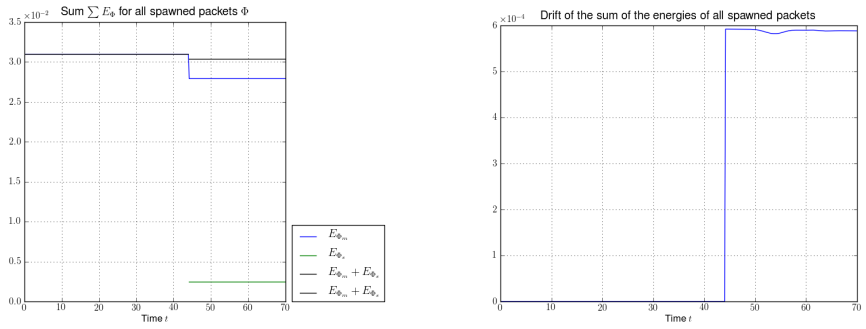
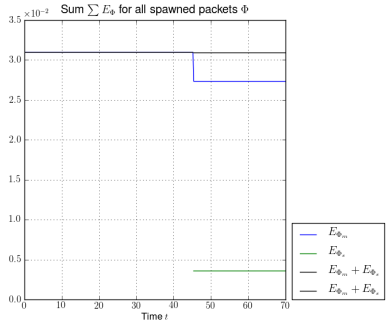


Figure 5.37: These panels show the norms of the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 6. (a) Spawning threshold $\tau = 0.275$. (b) Spawning threshold $\tau = 0.275$. (c) Spawning threshold $\tau = 0.30$. (d) Spawning threshold $\tau = 0.30$. (e) Spawning threshold $\tau = 0.32$. (f) Spawning threshold $\tau = 0.32$.

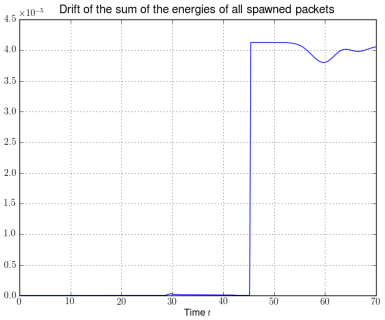


(a)

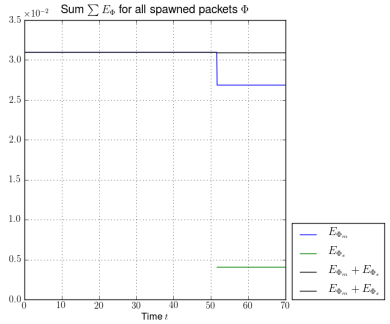
(b)



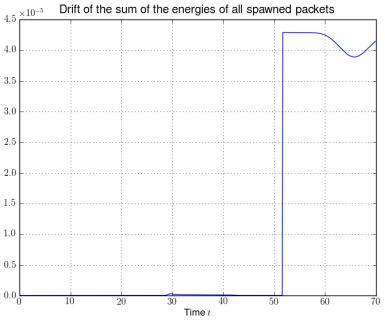
(c)



(d)



(e)



(f)

Figure 5.38: These panels show the energies and the error in energy conservation for the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 6. (a) Spawning threshold $\tau = 0.275$. (b) Spawning threshold $\tau = 0.275$. (c) Spawning threshold $\tau = 0.30$. (d) Spawning threshold $\tau = 0.30$. (e) Spawning threshold $\tau = 0.32$. (f) Spawning threshold $\tau = 0.32$.

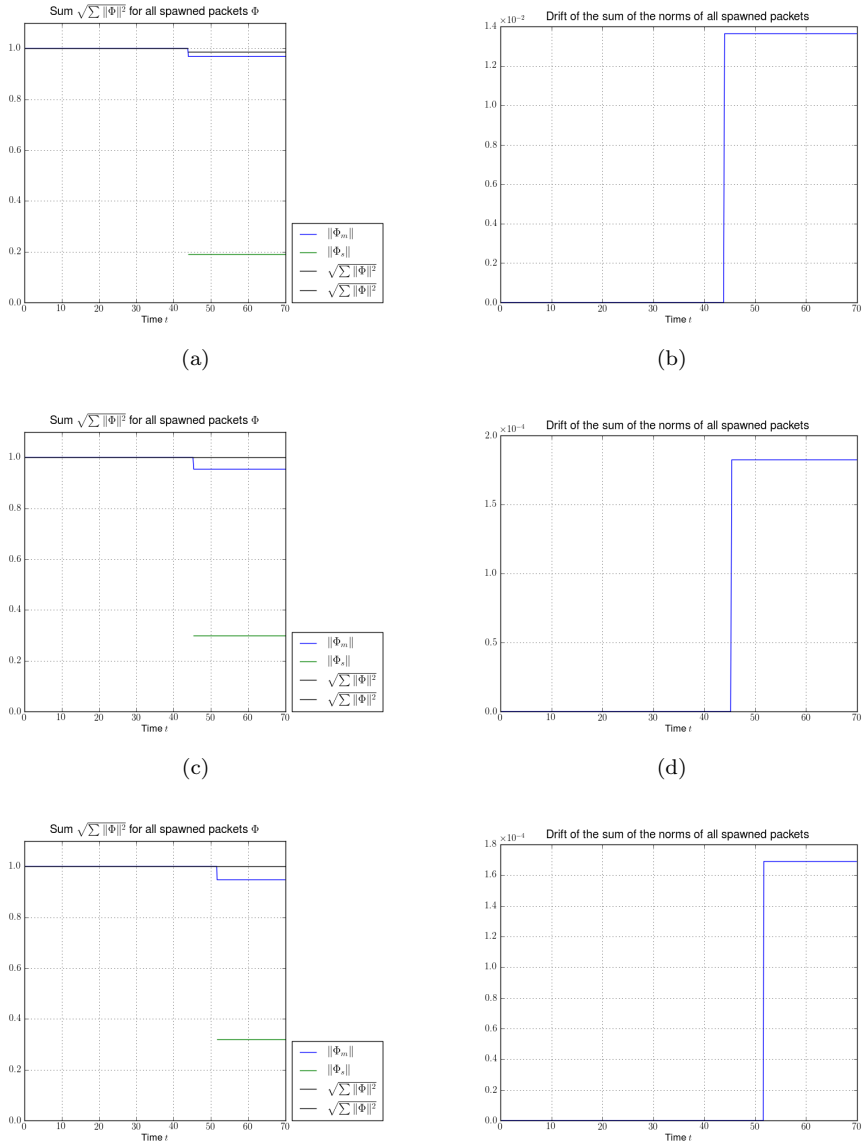
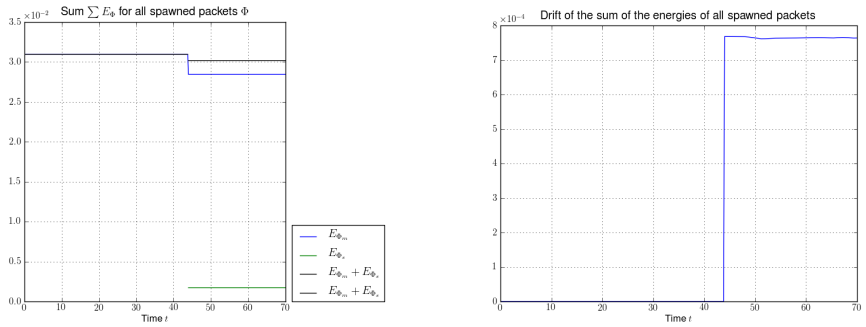
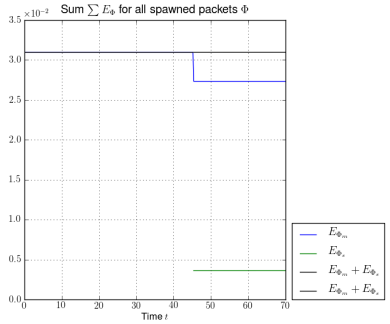


Figure 5.39: These panels show the norms of the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 12. (a) Spawning threshold $\tau = 0.25$. (b) Spawning threshold $\tau = 0.25$. (c) Spawning threshold $\tau = 0.30$. (d) Spawning threshold $\tau = 0.30$. (e) Spawning threshold $\tau = 0.32$. (f) Spawning threshold $\tau = 0.32$.

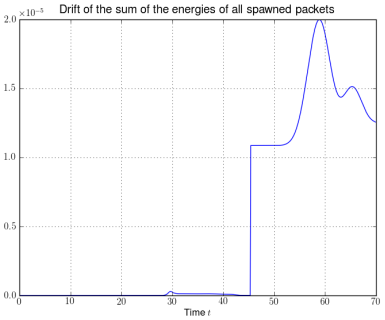


(a)

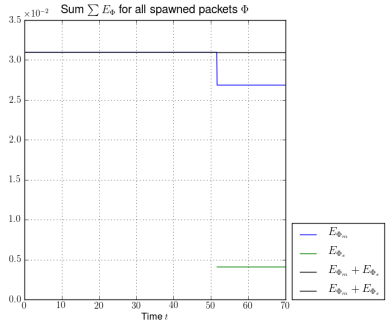
(b)



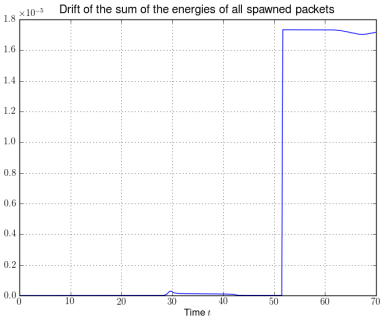
(c)



(d)



(e)



(f)

Figure 5.40: These panels show the energies and the error in energy conservation for the spawned wavepackets during spawning and later propagation for several different threshold parameters τ. The simulations were done with the projection method for the change of basis. The value μ was set to 12. (a) Spawning threshold $\tau = 0.25$. (b) Spawning threshold $\tau = 0.25$. (c) Spawning threshold $\tau = 0.30$. (d) Spawning threshold $\tau = 0.30$. (e) Spawning threshold $\tau = 0.32$. (f) Spawning threshold $\tau = 0.32$.

5.7 Issues and improvements

5.7.1 Other spawning criteria

We have seen in the last section that beside the problems we have with parameter estimation and the two different methods for the change of basis there appears another very important issue. The question is what is the best choice of the spawning parameter τ. The experiments tell us that we have to spawn as late as possible because only then we can be sure that we catch all of the transmitted part. If we look at the spawning case then it is clear the inside some bounds the values of $\langle w \,|\, w \rangle$ is a monotone and bounded function of t with an upper bound $W := \max \langle w \,|\, w \rangle$. (Do not take these mathematical terms with their full rigor here as we use them in a more informal way to explain the actual situation.) Then we have to choose $\tau^2 \in [0, W]$ but we do not know W a priori. In fact in the above simulations we used values known from earlier simulations but this is cheating. And the most serious problem occurs if we accidentally choose $\tau^2 > W$ because spawning will never happen in this case.
An improved criterion for determining the best time to spawn would be to check the derivative of the norm of w

$$\left| \frac{\mathrm{d} \langle w \,|\, w \rangle}{\mathrm{d}t} \right| \leq \zeta \tag{5.8}$$

and we spawn when this value is sufficiently small e.g. smaller than another threshold ζ. In less mathematical terms we spawn when the norm of the transmitted part w does not change too much anymore. To avoid spawning to early when hitting a local maximum we should extend this and require it to hold at least for a minimal time duration Δt larger than a few timesteps. This does not play a role for tunneling simulations as we said $\langle w \,|\, w \rangle$ is monotone but will become important for the avoided crossings in the next chapter.
This criterion frees us from the delicate choice of the threshold τ. But on the other hand it is much more difficult to implement. The value of ζ should pose no further problems if we choose it just small enough.

5.7.2 Adaptive basis size

It would make sense to drop the fixed basis size K and use an adaptive basis size $K(t)$ for both the original and the spawned wavepacket. This would be a good idea as we do need a really big basis size to capture all the high frequencies during the time when the tunneling process takes place only. The packets moving towards or away from the potential hill are essentially Gaussians for which a much smaller basis size would be sufficient. Remember, this was also the most fundamental reason for developing the whole spawning ideas.
We should be aware that the decision when to increase or decrease the basis size is highly non-trivial. If done wrong we will quickly loose the norm conservation.

Chapter 6

Spawning in non-adiabatic crossings

In the following chapter we try to use the ideas on spawning wavepackets in the case of a non-adiabatic potential with several energy levels. First we will look at the theoretical side and show how to use the formulae derived in chapter 4. We will see that this is straight forward and results in formulae with a structure well known from above. Later we recover algorithms similar to the ones from the last chapter. They are not necessarily more general but rather put the focus onto other points.
The second part of this chapter is used to show some simulation results in great detail and demonstrate the capabilities of the spawning algorithms. We will see how the until then theoretical spawning ideas perform in practice. The potential has only two energy levels and one single avoided crossing in the middle. This is probably the simplest example we could use for testing but already here we will discover several interesting facts. The results are mostly good but not without some smaller challenges.
We perform aposteriori spawning only, the reason for this is that the propagation spawning in the non-adiabatic case is considerable more complicated than in the tunneling examples. But an algorithm adapted to the special case of our simulations with only a single avoided crossing is given at the end.
The generalized algorithms for arbitrary potentials with an arbitrary number of energy levels and many crossings will be sketched in the next chapter.

6.1 Adapting the theoretical basis

The theory presented in chapter 4 is general enough that we can use it to describe parameter estimation and the change of basis also in the case of non-adiabatic potentials. In this case we have a potential $V(x)$ with N energy levels denoted by $\lambda_0, \ldots, \lambda_{N-1}$ and the wavepacket $|\Psi\rangle$ has N different components $\Phi_0, \ldots, \Phi_{N-1}$ too. Assume for the moment that all wavepackets are homogeneous ones. (We can drop this requirement at any time but have to be careful with the individual parameter sets Π_i for each component Φ_i.)
The ideas of spawning packets and therewith replacing parts of already existing packets now apply not to subsets of Φ_0 like in the tunneling case but to *whole*

components Φ_i. To clarify this let's make a small example and assume our Ψ is defined as $\Psi := (\Phi_0 = \phi_0, \Phi_1 = 0)$ where we have a ϕ_0 on the first component and nothing on the second. If we propagate this packet through an avoided crossing then there will appear another fragment w on the lower level whose shape depends mainly on the incoming packet and on the gap size δ. So after the crossing our wavepacket looks like $\Psi' = (\Phi_0 = v, \Phi_1 = w)$ where v and w are some linear combinations of basis functions. (We omitted the global phase in this example.) If now the parts v and w propagate at different speed it is a bad idea to keep them glued together in the same wavepacket Ψ'. The reason is similar to what we gave as motivation for developing the spawning ideas in the first place. We will need a basis with more and more high frequency parts although both packets have (for example) the shape of a Gaussian. The idea to split of one of the two parts v or w and put it into a new wavepacket now arises naturally.
To go back from this small example to the general case, we assume that we want to split of the wavefunction represented by the component Φ_ν where $\nu = 1$ above and $\nu \in [0, N-1]$ in general. We call ν the index of the *monitored component*. For now presume the ν is given and we do not have to find it. The fragment for which we perform the parameter estimation is now the whole component Φ_ν and we write $w := \Phi_\nu$ and return to chapter 4 to see what this implies. First of course it tells us the values for α and β, namely $\alpha = 0$ and $\beta = K - 1$ where K is the basis size of Φ_ν. This is already enough to simplify the two formulae (4.17) and (4.18) for position and momentum estimation. By plugging in the values we get

$$\tilde{q} := \frac{\langle w \,|\, x \,|\, w \rangle}{\langle w \,|\, w \rangle} = \frac{\sqrt{2\varepsilon^2}}{\sum_{k=0}^{K-1} \overline{c_k} c_k} \Re \left(Q \sum_{k=1}^{K-1} \overline{c_k} c_{k-1} \sqrt{k} \right) + q \tag{6.1}$$

$$\tilde{p} := \frac{\langle w \,|\, y \,|\, w \rangle}{\langle w \,|\, w \rangle} = \frac{\sqrt{2\varepsilon^2}}{\sum_{k=0}^{K-1} \overline{c_k} c_k} \Re \left(P \sum_{k=1}^{K-1} \overline{c_k} c_{k-1} \sqrt{k} \right) + p \,. \tag{6.2}$$

For the second moments we apply the same simplifications to (4.38) and (4.39) and get

$$\frac{\left\langle w \,\middle|\, (x - \tilde{q})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} = \frac{\varepsilon^2 \Re \left(Q^2 \sum_{k=0}^{K-3} \overline{c_{k+2}} c_k \sqrt{k^2 + 3k + 2} \right) + \frac{\varepsilon^2}{2} |Q|^2 \sum_{k=0}^{K-1} \overline{c_k} c_k \, (2k+1)}{\sum_{k=0}^{K-1} \overline{c_k} c_k} - (q - \tilde{q})^2 \,. \tag{6.3}$$

$$\frac{\left\langle w \,\middle|\, (x - \tilde{p})^2 \,\middle|\, w \right\rangle}{\langle w \,|\, w \rangle} = \frac{\varepsilon^2 \Re \left(P^2 \sum_{k=0}^{K-3} \overline{c_{k+2}} c_k \sqrt{k^2 + 3k + 2} \right) + \frac{\varepsilon^2}{2} |P|^2 \sum_{k=0}^{K-1} \overline{c_k} c_k \, (2k+1)}{\sum_{k=0}^{K-1} \overline{c_k} c_k} - (p - \tilde{p})^2 \,. \tag{6.4}$$

These four equations are all we need to estimate the parameters $\tilde{q}$, $\tilde{p}$, $\tilde{P}$ and $\tilde{Q}$ of our fragment $w = \Phi_\nu$. In contrast to the tunneling case we can in general *not* set $k = 0$ in (4.42) and have to keep the variable k as an input parameter of our algorithm. In the simulations later in this chapter therefore we also show the same simulation setups with different values of k. The remainder of the

overall spawning process works as described in chapter 4 and especially the two algorithms for changing the basis from Π to $\tilde{\Pi}$ stay the same but take as input $w = \Phi_\nu$. Both algorithms are still valid but we will use almost exclusively the basis projection because is is much more flexible and error tolerant and we hardly ever have a pure ϕ_i on the component Φ_ν.

6.2 Basic avoided crossings simulations

In this section we present the simulations that are used as the base for later aposteriori spawning simulations. The potential has a single avoided crossing and is shown in figure 6.1. We will start with different initial wavepackets $|\Psi\rangle$ on the upper level and let the packets move to the right through the crossing. The exact simulation parameters are reprinted in the appendix for reference and reproducibility.

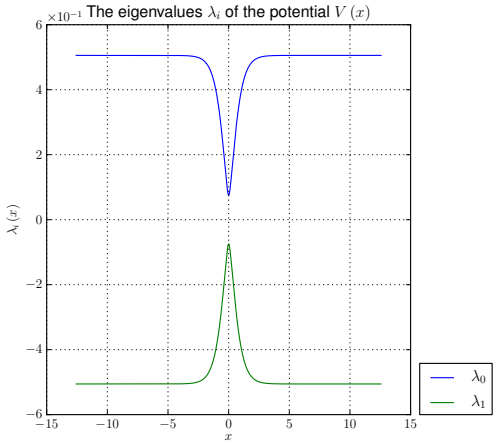


Figure 6.1: A simple potential with two energy levels and a single avoided crossing.

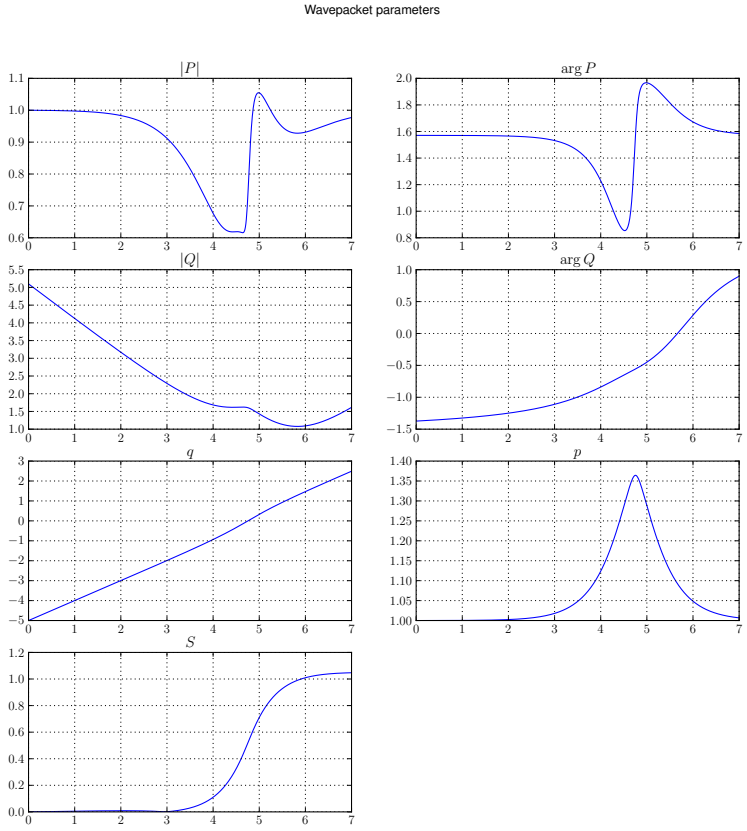
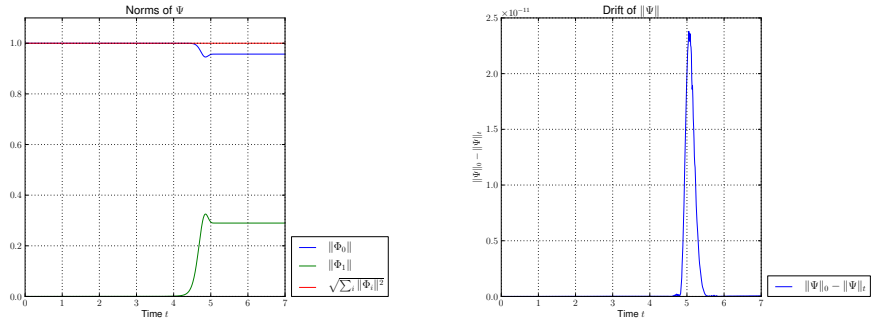
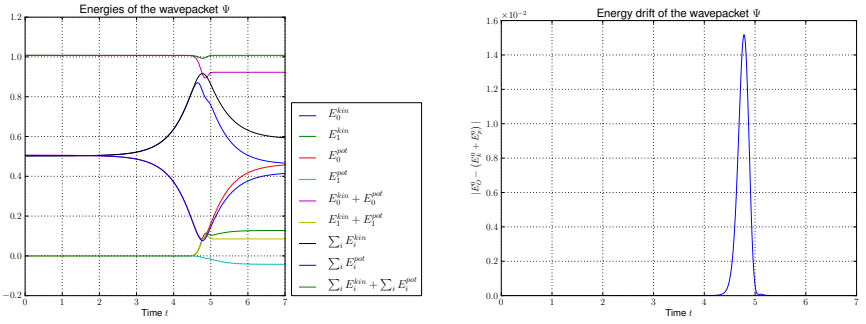


Figure 6.2: The parameter set Π, it is identical to all simulations presented in this section.



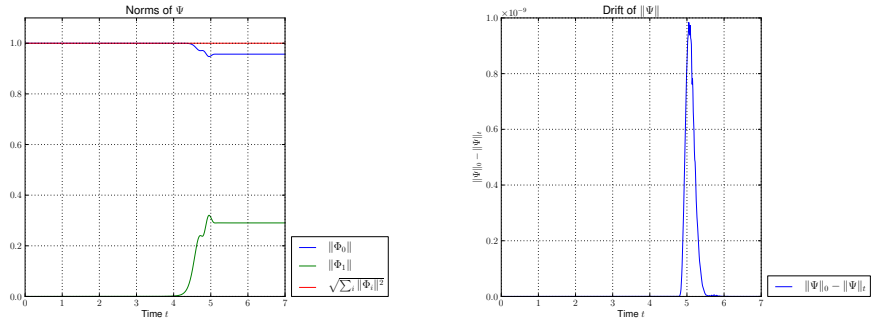
(a) (b)

Figure 6.3: This figure shows the norm of both components Φ_0 and Φ_1 of the wavepacket Ψ as well as the overall norm. The right panel shows the drift of the overall norm, which should be conserved as good as possible. The initial wavepacket $|\Psi(t=0)\rangle = \phi_0$ starts on the upper level. The full set of simulation parameters is printed in A.6.1



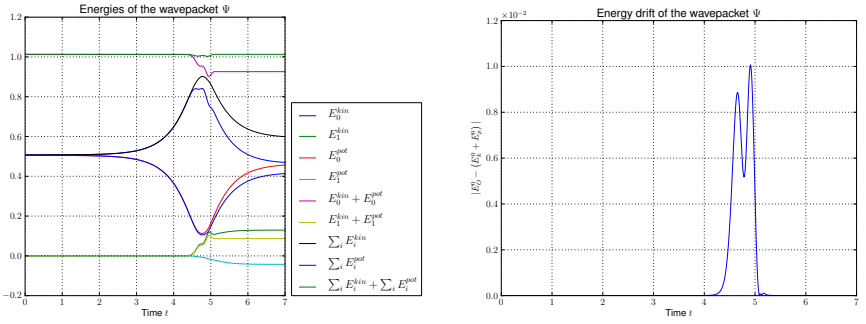
(a) (b)

Figure 6.4: Kinetic and potential energies and the violation of energy conservation for an initial wavepacket ϕ_0 starting on the upper level. The full set of simulation parameters is printed in A.6.1



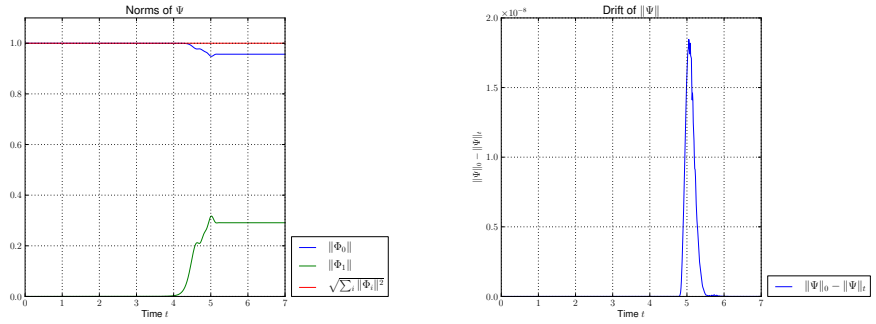
(a) (b)

Figure 6.5: This figure shows the norm of both components Φ_0 and Φ_1 of the wavepacket Ψ as well as the overall norm. The right panel shows the drift of the overall norm, which should be conserved as good as possible. The initial wavepacket $|\Psi(t=0)\rangle = \phi_1$ starts on the upper level. The full set of simulation parameters is printed in A.6.2



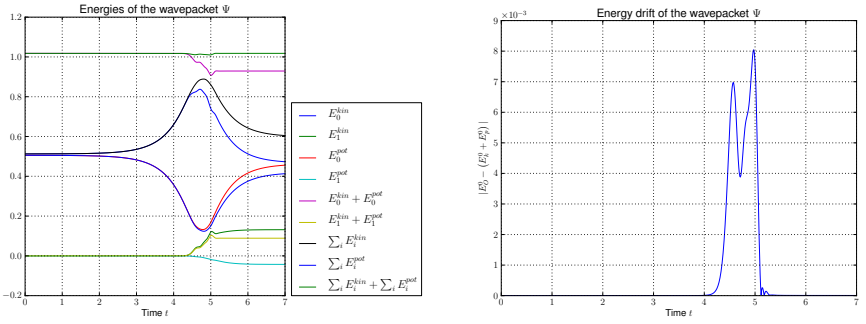
(a) (b)

Figure 6.6: Kinetic and potential energies and the violation of energy conservation for an initial wavepacket ϕ_1 starting on the upper level. The full set of simulation parameters is printed in A.6.2



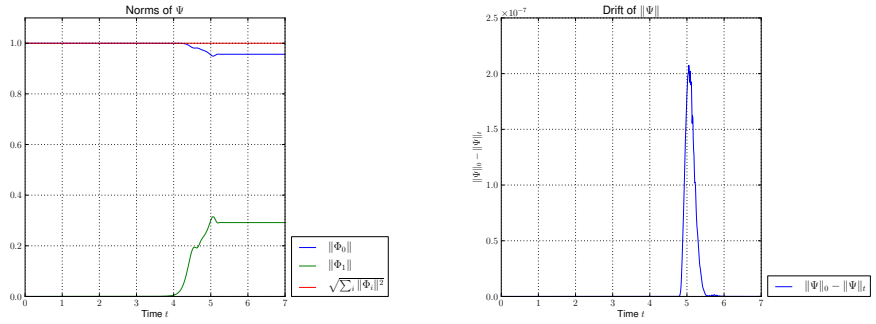
(a) (b)

Figure 6.7: This figure shows the norm of both components Φ_0 and Φ_1 of the wavepacket Ψ as well as the overall norm. The right panel shows the drift of the overall norm, which should be conserved as good as possible. The initial wavepacket $|\Psi(t=0)\rangle = \phi_2$ starts on the upper level. The full set of simulation parameters is printed in A.6.3



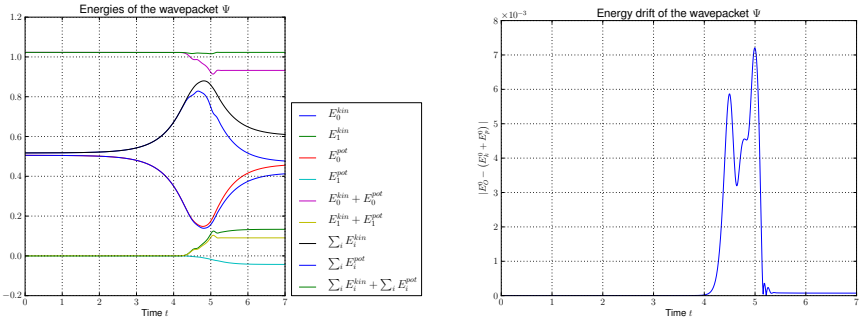
(a) (b)

Figure 6.8: Kinetic and potential energies and the violation of energy conservation for an initial wavepacket ϕ_2 starting on the upper level. The full set of simulation parameters is printed in A.6.3



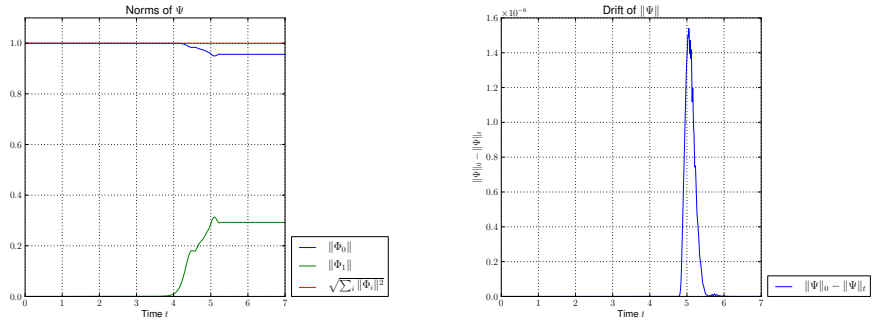
(a) (b)

Figure 6.9: This figure shows the norm of both components Φ_0 and Φ_1 of the wavepacket Ψ as well as the overall norm. The right panel shows the drift of the overall norm, which should be conserved as good as possible. The initial wavepacket $|\Psi(t=0)\rangle = \phi_3$ starts on the upper level. The full set of simulation parameters is printed in A.6.4



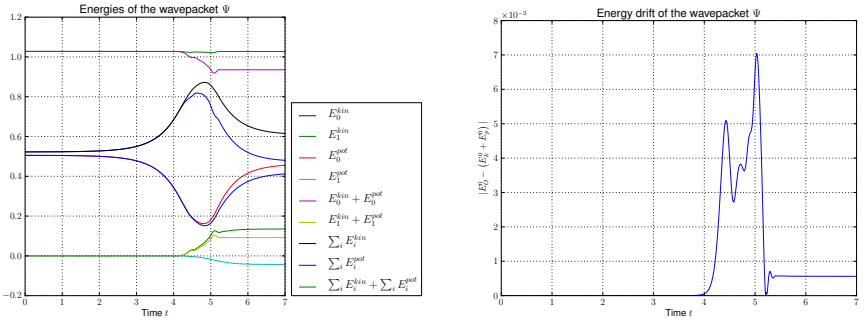
(a) (b)

Figure 6.10: Kinetic and potential energies and the violation of energy conservation for an initial wavepacket ϕ_3 starting on the upper level. The full set of simulation parameters is printed in A.6.4



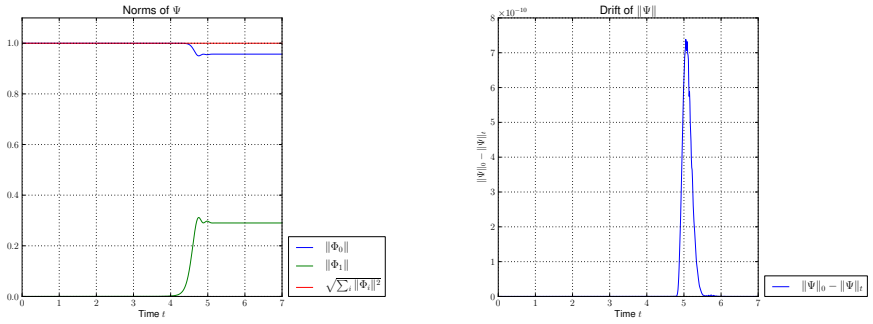
(a) (b)

Figure 6.11: This figure shows the norm of both components Φ_0 and Φ_1 of the wavepacket Ψ as well as the overall norm. The right panel shows the drift of the overall norm, which should be conserved as good as possible. The initial wavepacket $|\Psi(t=0)\rangle = \phi_4$ starts on the upper level. The full set of simulation parameters is printed in A.6.5



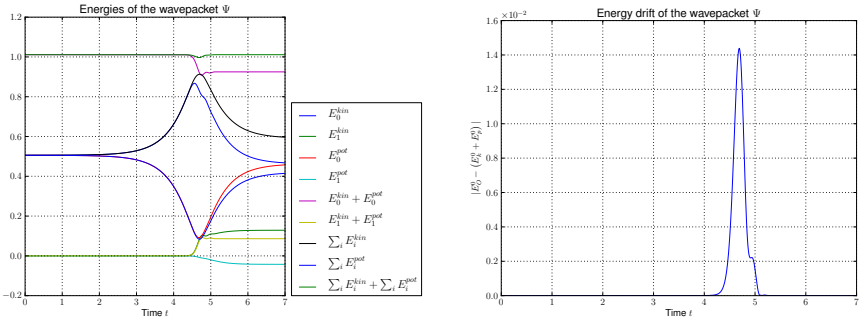
(a) (b)

Figure 6.12: Kinetic and potential energies and the violation of energy conservation for an initial wavepacket ϕ_4 starting on the upper level. The full set of simulation parameters is printed in A.6.5



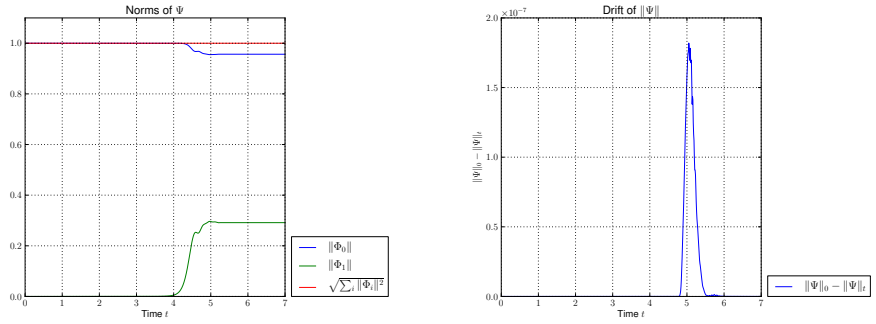
(a) (b)

Figure 6.13: This figure shows the norm of both components Φ_0 and Φ_1 of the wavepacket Ψ as well as the overall norm. The right panel shows the drift of the overall norm, which should be conserved as good as possible. The initial wavepacket $|\Psi(t=0)\rangle = \phi_0 + \phi_1$ starts on the upper level. The full set of simulation parameters is printed in A.6.6



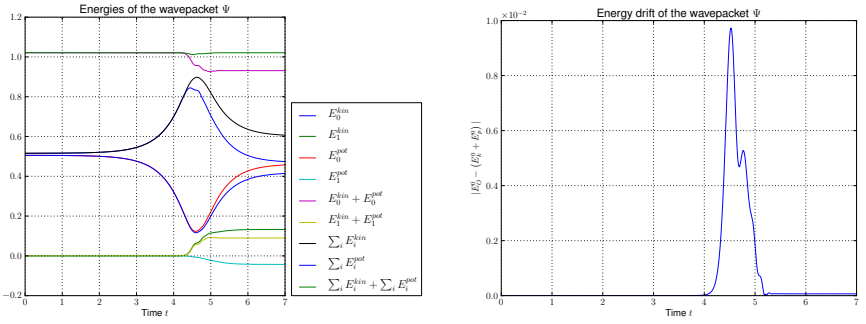
(a) (b)

Figure 6.14: Kinetic and potential energies and the violation of energy conservation for an initial superposition of ϕ_0 and ϕ_1 starting on the upper level. The full set of simulation parameters is printed in A.6.6



(a) (b)

Figure 6.15: This figure shows the norm of both components Φ_0 and Φ_1 of the wavepacket Ψ as well as the overall norm. The right panel shows the drift of the overall norm, which should be conserved as good as possible. The initial wavepacket $|\Psi(t=0)\rangle = \phi_2 + \phi_3$ starts on the upper level. The full set of simulation parameters is printed in A.6.7



(a) (b)

Figure 6.16: Kinetic and potential energies and the violation of energy conservation for an initial superposition of ϕ_2 and ϕ_3 starting on the upper level. The full set of simulation parameters is printed in A.6.7

6.3 Aposteriori spawning for avoided crossings

The aposteriori spawning process for avoided crossings can be described by the general algorithm 7. The only difference to the tunneling case are the input values. First we deal with vector values packets Ψ consisting of N components in the general case. And we concentrate not on a subset of Φ_0 but on a whole Φ_i. Hence the values for α and β must be set to 0 and $K-1$ to obtain the correct fragment w. We set the input value ν to the component we wish to monitor. In the simulation above where we have only two energy levels ($N=2$) and start on the upper one this would be the lower level and thus $\nu = 1$.
The following examples show the aposteriori spawning process based on the simulations from the last section. The gap size δ of the avoided crossing is chosen in a way that if we start which a ϕ_i on the upper level then we end up again with a ϕ_i' on the lower level, for details see reference [1]. We first look at the norms and the energies. The most important observation here is that the results crucially depend on the values of k in the formula (4.42). If we just set $k = 0$ we get wrong norms and wrong energies for higher order ϕ_i. This can be seen in the following figures. In 6.18 and 6.19 the orange curve is the norm of the original part Φ_1 on the lower level. And the cyan curve should match as good as possible. Also in figures 6.20 and 6.21 we observe that for $k = 0$ (left panels) the error in the norm conservation of Φ_1 differs considerably from what we have in the right panels where the norm conservation is fulfilled again after the avoided crossing. For the kinetic and potential energies of Φ_1 we have a similar picture. Figures 6.22 and 6.23 show various energies. The kinetic energy of the original wavepacket's components Φ_0 and Φ_1 is plotted in red, the potential energy in orange. The cyan and light green curves overlap these two only for the compatible choice of k. We should note that in either case we get good results only *after* the packet passed the avoided crossing. This compares to the tunneling case where spawning a new packet makes only sense after tunneling has already occurred. The energy conservation shown in figures 6.24 and 6.25 is violated heavily after the crossing for $k = 0$ but fulfilled quite well for the matching choice of k. In summary we find that the values from the original simulation and the corresponding ones from the aposteriori simulations do not overlap if k does not match to the ϕ_k, sometimes not even for asymptotic large times. So the algorithm does not converge in these cases.
The two figures 6.27 and 6.28 show the spawn error. We see that with the matching choice of k we have a spawn error that is about an order of magnitude smaller than what we get with $k = 0$. Thus it is important to use the correct k in formula (4.42). As an example if we start with a ϕ_3 we must set $k = 3$ for obtaining good results. For sure we could reduce the error further if we choose a larger basis size η but this is not the point here. All comparisons were done with a basis size of $\eta = 16$ and $\mu = \eta$.

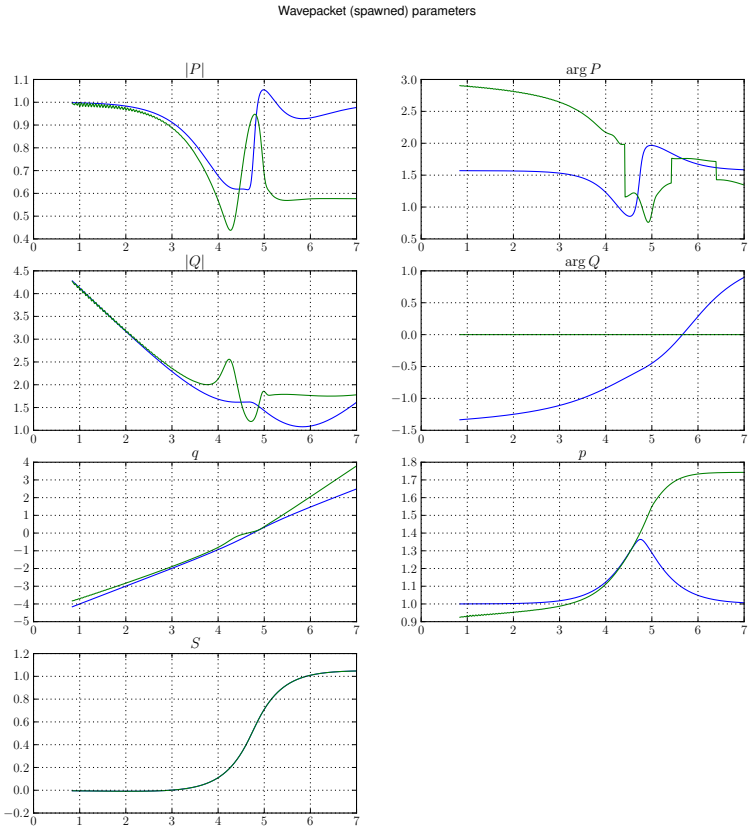


Figure 6.17: The parameter sets Π (blue) and $\tilde{\Pi}$ (green) for the case of $\Psi = \phi_0$ and $k = 0$. We can give a nice interpretation to the parameters $\tilde{q}$ and $\tilde{p}$. For $\tilde{q}$ we see a steeper slope which means that after the crossing the spawned packet on the lower level moves faster. And $\tilde{p}$ continues to increase after the crossing while p becomes smaller again which seems right compared to the shape of the potential. The full set of simulation parameters is printed in A.7.1.

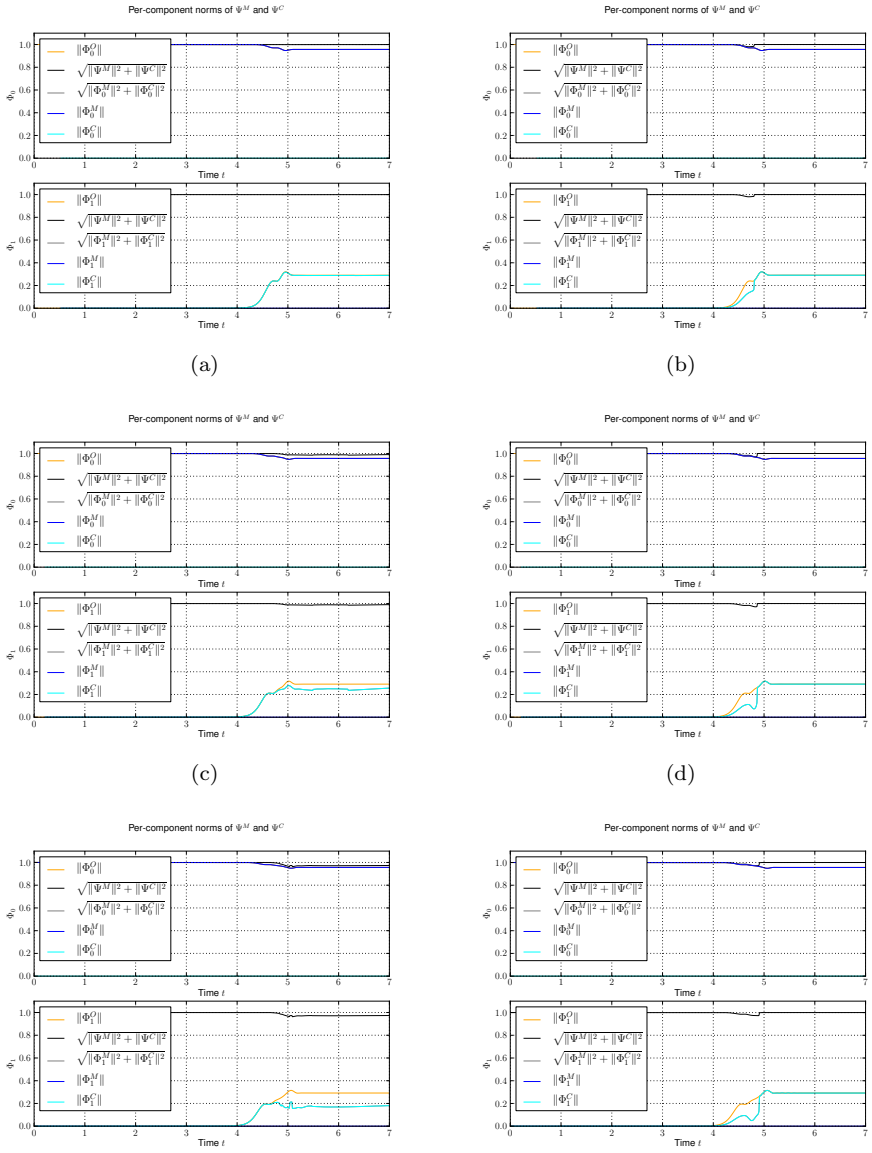
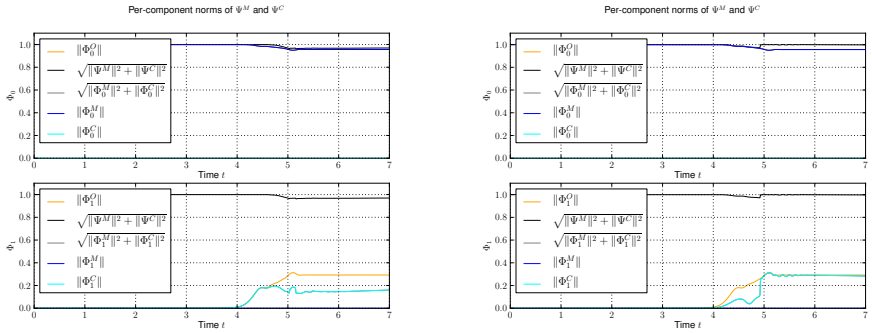
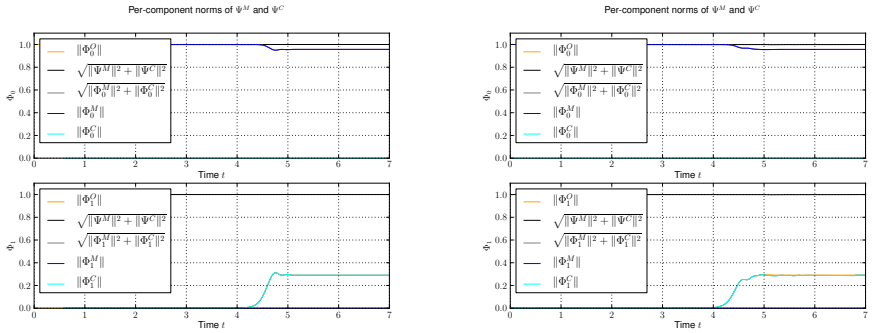


Figure 6.18: This figure shows the component-wise norms for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_1$ and $k = 0$ (b) $\Psi = \phi_1$ and $k = 1$ (c) $\Psi = \phi_2$ and $k = 0$ (d) $\Psi = \phi_2$ and $k = 2$ (e) $\Psi = \phi_3$ and $k = 0$ (f) $\Psi = \phi_3$ and $k = 3$



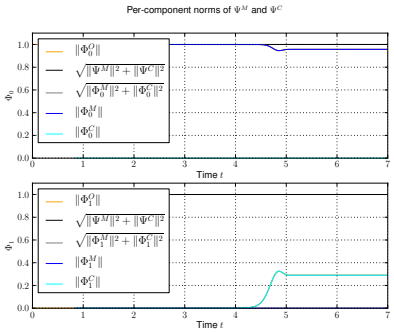
(a)

(b)



(c)

(d)



(e)

Figure 6.19: This figure shows the component-wise norms for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_4$ and $k = 0$ (b) $\Psi = \phi_4$ and $k = 4$ (c) $\Psi = \phi_0 + \phi_1$ and $k = 0$ (d) $\Psi = \phi_2 + \phi_3$ and $k = 0$ (e) $\Psi = \phi_0$ and $k = 0$

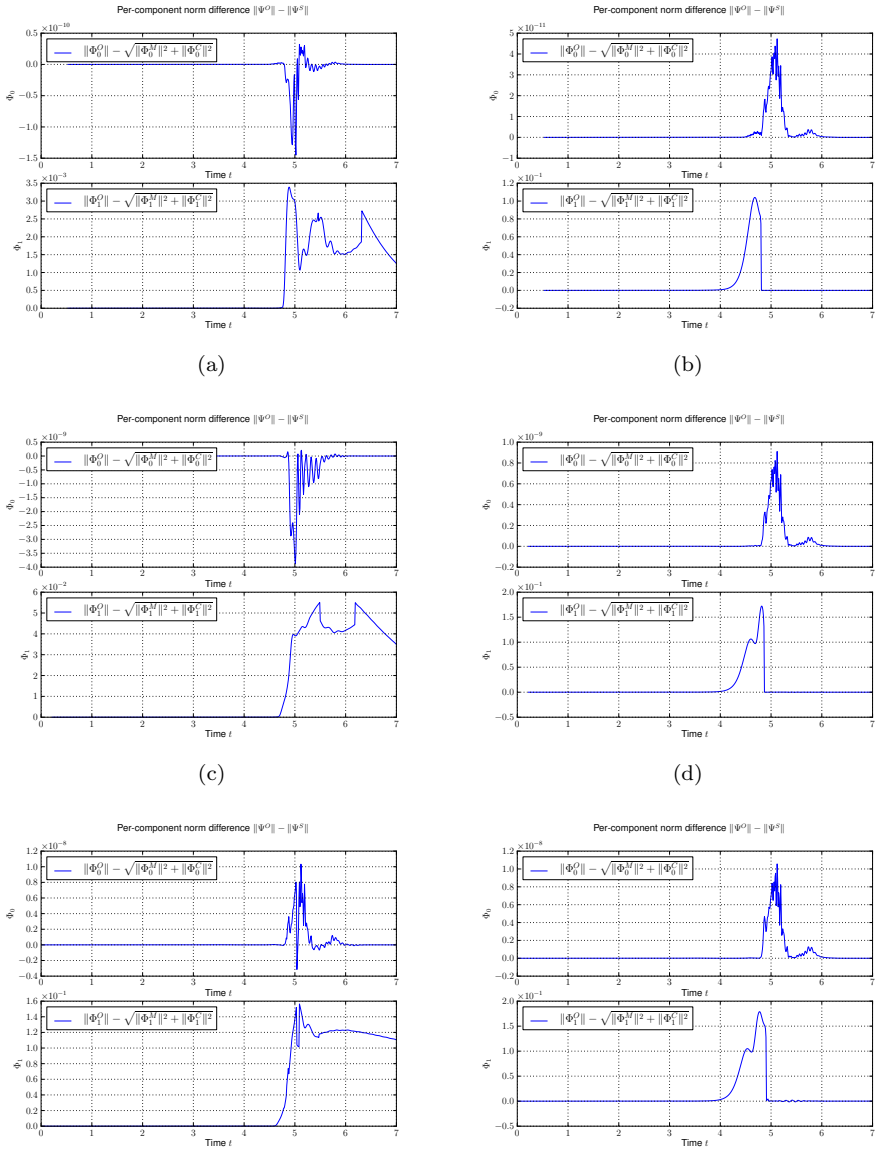
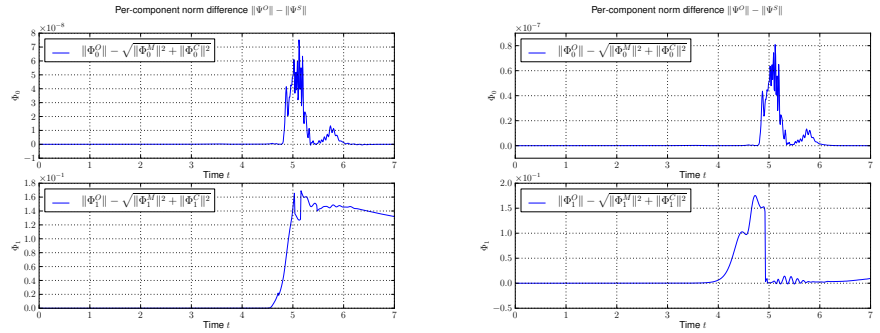
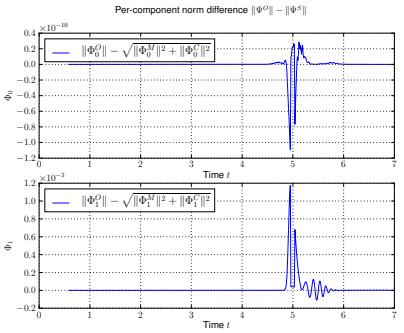


Figure 6.20: This figure shows the component-wise drift in the norm for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_1$ and $k = 0$ (b) $\Psi = \phi_1$ and $k = 1$ (c) $\Psi = \phi_2$ and $k = 0$ (d) $\Psi = \phi_2$ and $k = 2$ (e) $\Psi = \phi_3$ and $k = 0$ (f) $\Psi = \phi_3$ and $k = 3$

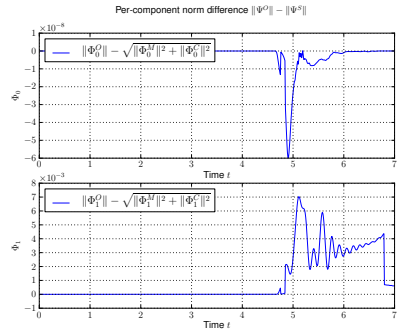


(a)

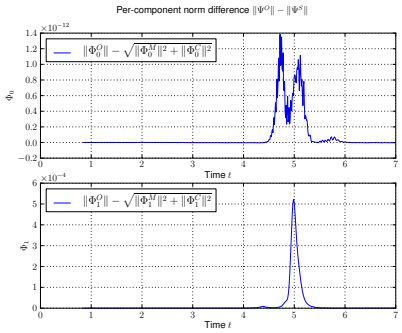
(b)



(c)

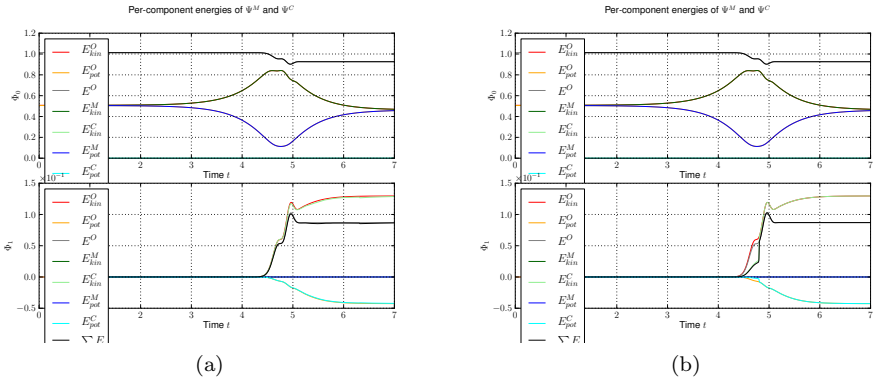


(d)



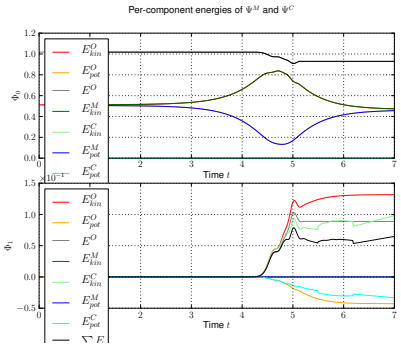
(e)

Figure 6.21: This figure shows the component-wise drift in the norm for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_4$ and $k = 0$ (b) $\Psi = \phi_4$ and $k = 4$ (c) $\Psi = \phi_0 + \phi_1$ and $k = 0$ (d) $\Psi = \phi_2 + \phi_3$ and $k = 0$ (e) $\Psi = \phi_0$ and $k = 0$

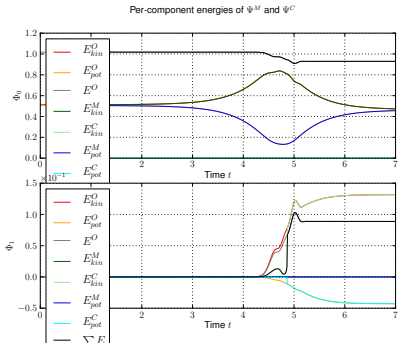


(a)

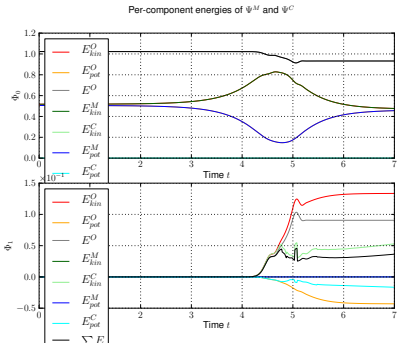
(b)



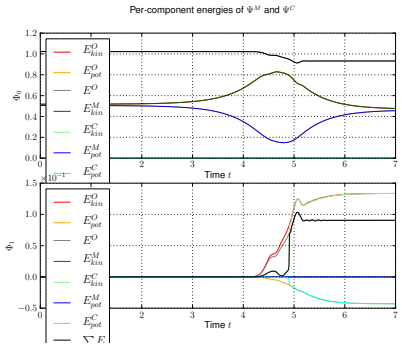
(c)



(d)



(e)



(f)

Figure 6.22: This figure shows the kinetic and potential energies for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_1$ and $k = 0$ (b) $\Psi = \phi_1$ and $k = 1$ (c) $\Psi = \phi_2$ and $k = 0$ (d) $\Psi = \phi_2$ and $k = 2$ (e) $\Psi = \phi_3$ and $k = 0$ (f) $\Psi = \phi_3$ and $k = 3$

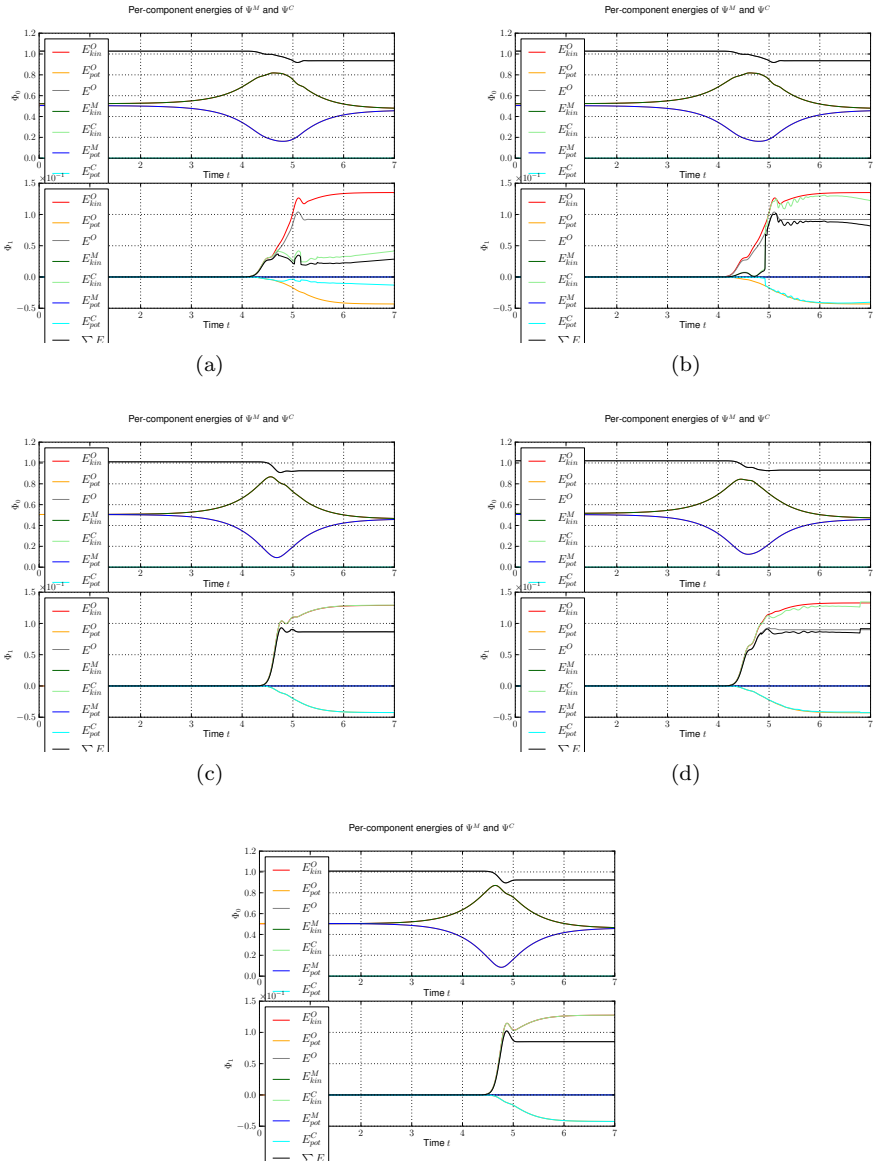
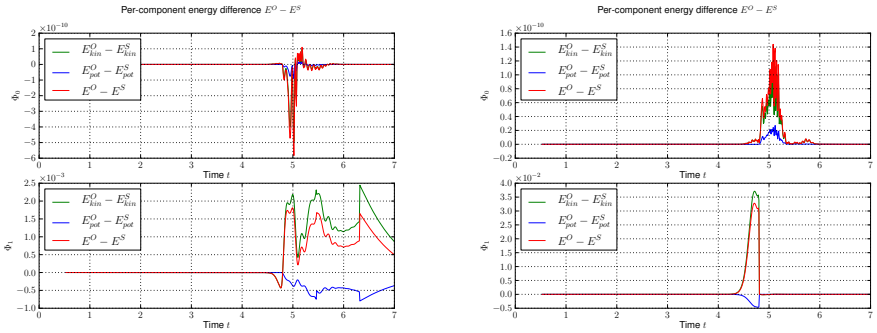
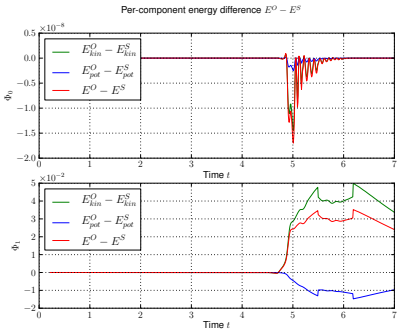


Figure 6.23: This figure shows the kinetic and potential energies for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_4$ and $k = 0$ (b) $\Psi = \phi_4$ and $k = 4$ (c) $\Psi = \phi_0 + \phi_1$ and $k = 0$ (d) $\Psi = \phi_2 + \phi_3$ and $k = 0$ (e) $\Psi = \phi_0$ and $k = 0$

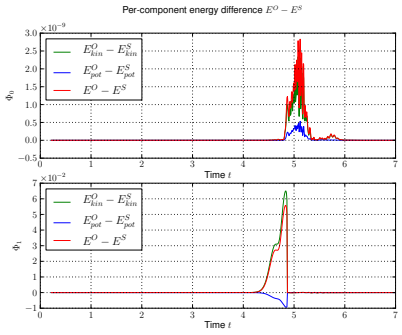


(a)

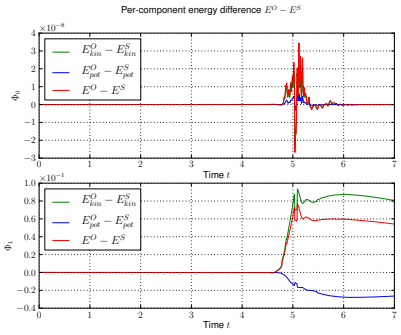
(b)



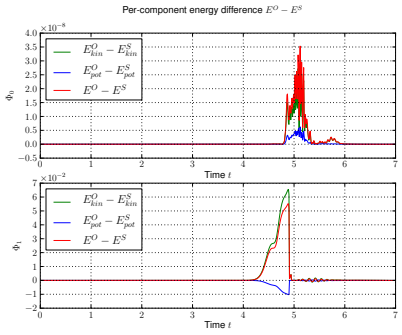
(c)



(d)

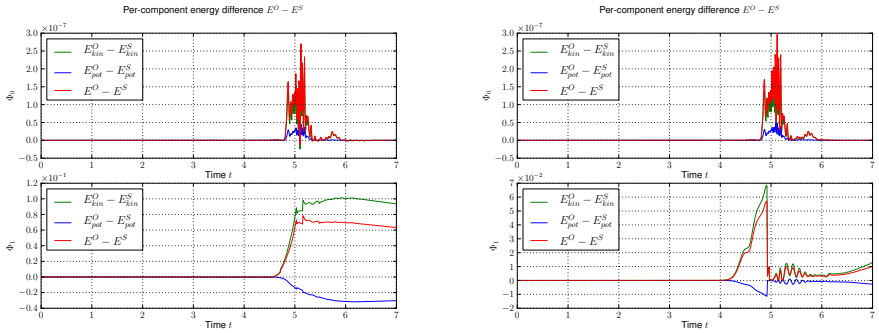


(e)



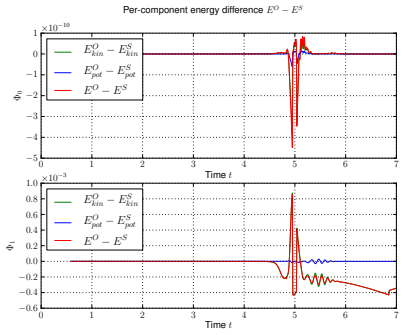
(f)

Figure 6.24: This figure shows the drift in the kinetic, potential and overall energy for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_1$ and $k = 0$ (b) $\Psi = \phi_1$ and $k = 1$ (c) $\Psi = \phi_2$ and $k = 0$ (d) $\Psi = \phi_2$ and $k = 2$ (e) $\Psi = \phi_3$ and $k = 0$ (f) $\Psi = \phi_3$ and $k = 3$

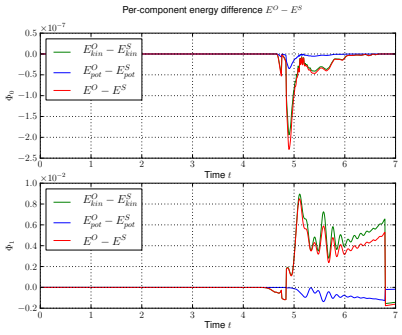


(a)

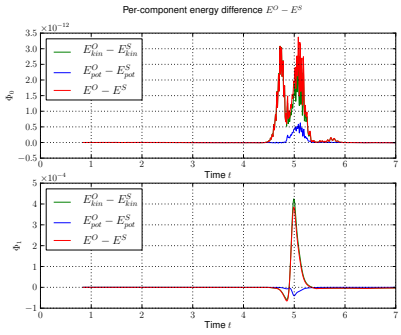
(b)



(c)

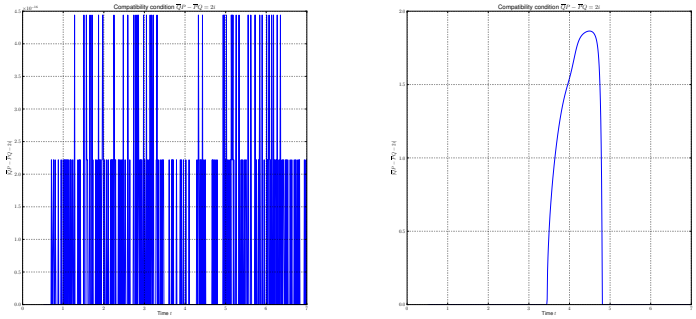


(d)

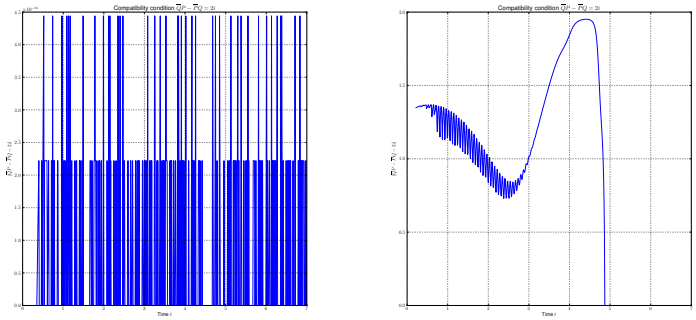


(e)

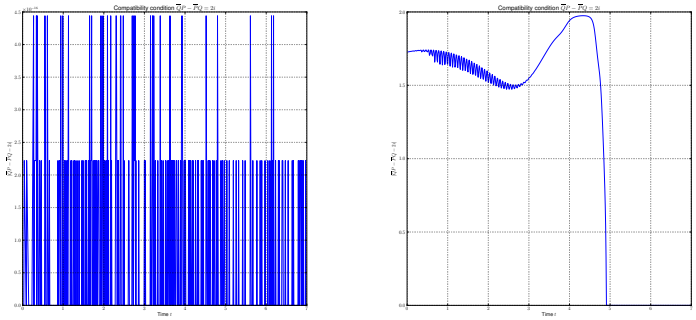
Figure 6.25: This figure shows the drift in the kinetic, potential and overall energy for several non-adiabatic examples. The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_4$ and $k = 0$ (b) $\Psi = \phi_4$ and $k = 4$ (c) $\Psi = \phi_0 + \phi_1$ and $k = 0$ (d) $\Psi = \phi_2 + \phi_3$ and $k = 0$ (e) $\Psi = \phi_0$ and $k = 0$



(a) (b)



(c) (d)



(e) (f)

Figure 6.26: This figure shows that the symplecticity condition from equation (2.1) is violated for the simulations with $k > 0$. The reason for this is not that $k > 0$ but is described in detail at the end of chapter 4. Luckily for us the violations never happened during or after the avoided crossing. The values in the left panels are numerically zero up to machine precision. The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_1$ and $k = 0$ (b) $\Psi = \phi_1$ and $k = 1$ (c) $\Psi = \phi_2$ and $k = 0$ (d) $\Psi = \phi_2$ and $k = 2$ (e) $\Psi = \phi_3$ and $k = 0$ (f) $\Psi = \phi_3$ and $k = 3$

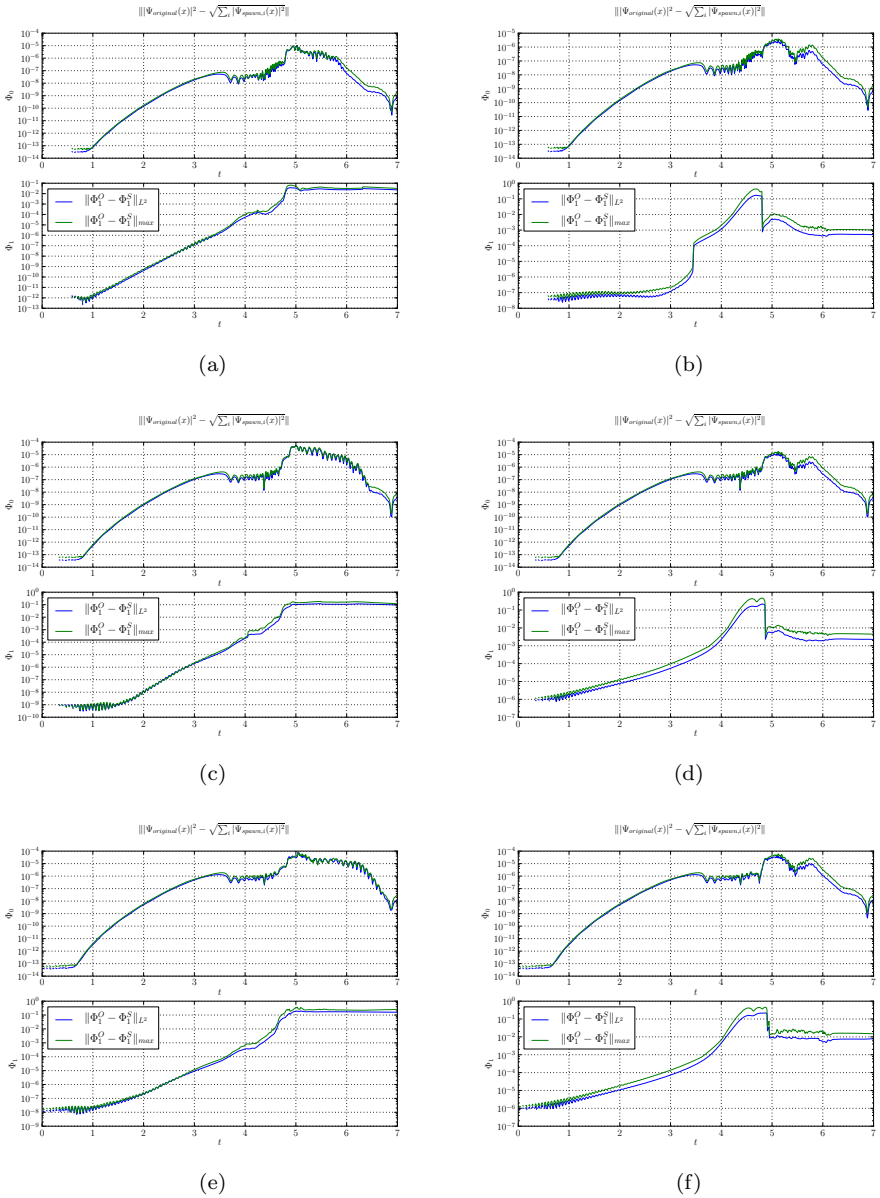
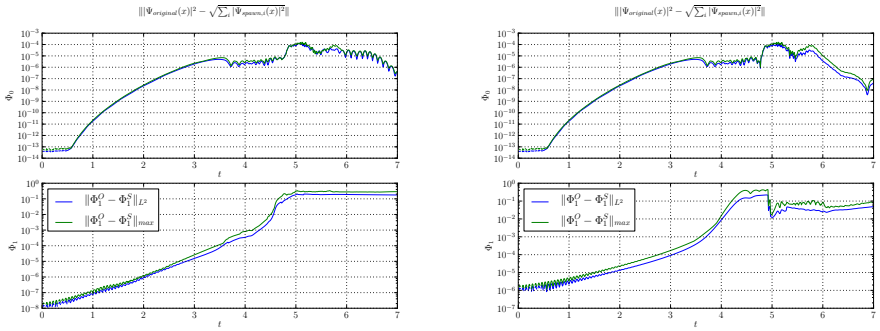
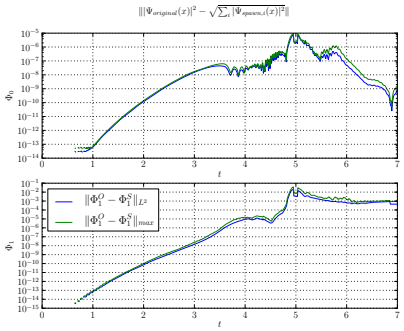


Figure 6.27: This figure shows the spawn error $\|\Psi-(\tilde{v}+\tilde{w})\|$ for both components in L^2 norm (blue) and maximum norm (green). The individual panels are for different wavepackets $\Psi=\phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi=\phi_1$ and $k=0$ (b) $\Psi=\phi_1$ and $k=1$ (c) $\Psi=\phi_2$ and $k=0$ (d) $\Psi=\phi_2$ and $k=2$ (e) $\Psi=\phi_3$ and $k=0$ (f) $\Psi=\phi_3$ and $k=3$

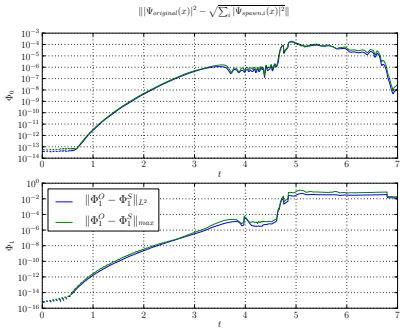


(a)

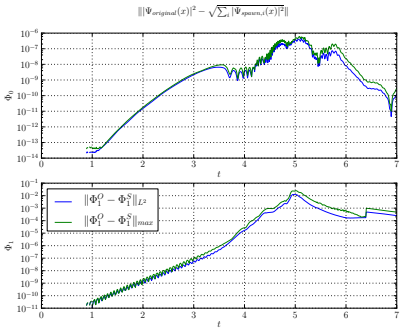
(b)



(c)



(d)



(e)

Figure 6.28: This figure shows the spawn error $\|\Psi-(\tilde{v}+\tilde{w})\|$ for both components in L^2 norm (blue) and maximum norm (green). The individual panels are for different wavepackets $\Psi = \phi_i$ and different values of k where the k is the one from formula (4.42). The full set of simulation parameters is printed in A.7. (a) $\Psi = \phi_4$ and $k = 0$ (b) $\Psi = \phi_4$ and $k = 4$ (c) $\Psi = \phi_0 + \phi_1$ and $k = 0$ (d) $\Psi = \phi_2 + \phi_3$ and $k = 0$ (e) $\Psi = \phi_0$ and $k = 0$

6.4 Spawning and propagation

The combination of spawning and propagation in the non-adiabatic case is similar to what is shown in section 5.6. But the small differences justify the review of the algorithm in the current section. We use the same basic notations from section 6.1. The potential has N energy levels and hence the wavepacket has N components. The wavepacket is usually assumed to be homogeneous but this is not necessary. (In the case of inhomogeneous wavepackets we can forget about the characteristic component χ.)
We start with a single wavepacket $|\Psi\rangle$ as initial value at time τ_0 and assume that the spawning criterion is fulfilled only once during the whole simulation. Then we will end up with two wavepackets $|\Psi_0\rangle$ and $|\Psi_1\rangle$ after spawning. There are basis transformations between the canonical and the eigenbasis involved and we denote quantities in the corresponding basis with a c or e superscript. These transformations albeit expensive are necessary as the propagation takes place in the canonical basis but we have to run the whole spawning procedure in the eigenbasis where the different components are decoupled. The transformation matrix M was defined in section 1.5.1.
With algorithm 9 we conclude this chapter about spawning in the non-adiabatic case.

Algorithm 9 Spawning propagation method (simplified non-adiabatic case)

Require: A series t consisting of timesteps $\{\tau_0, \ldots, \tau_{\max}\}$
Require: The potential $V(x)$
Require: The initial (homogeneous) wavepacket $|\Psi^c(t = \tau_o)\rangle$
Require: The monitored component $0 \leq \nu \leq N - 1$
Require: The value of $0 \leq k \leq K - 1$
Require: A spawn threshold τ
  for all $\tau_i \in t$ **do**
    // Transform the packet to the eigenbasis
    $\Psi^e := M^{-1}\Psi^c$
    // Extract the monitored component Φ_ν of Ψ^e
    $w := \Phi_\nu = \sum_{k=0}^{K-1} c_k^\nu \phi_k$
    // Check the spawning criterion
    if $\langle w \,|\, w \rangle \geq \tau^2$ **then**
      // Perform spawning procedure
      // Estimate the parameters of the fragment by algorithm 4
      $\tilde{\Pi} :=$ **estimate_parameters**(w)
      // Move the fragment w to the new basis by either algorithm 5 or 6
      $\tilde{w} :=$ **change_basis**$(\tilde{\Pi}, w)$
      // Update the reminder (included in algorithm 5 and 6)
      $\tilde{v} :=$ **update_remainder**$(v, \tilde{w})$
      // We have now two new, full wavepackets
      $\Psi_0 := (\Phi_0, \ldots, \Phi_{\nu-1}, \Phi_\nu := \tilde{v}, \Phi_{\nu+1}, \ldots, \Phi_{N-1})$
      $\Psi_1 := (0, \ldots, 0, \Phi_\nu := \tilde{w}, 0, \ldots, 0)$
      // Set the characteristic level χ of Ψ_1 to ν
      // Only used for propagation with homogeneous wavepackets
      $\chi_{\Psi_1} := \nu$
      // Transform the packets back into the canonical basis
      $\Psi_0^c := M\Psi_0^e$ and $\Psi_1^c := M\Psi_1^e$
    end if
    // Time propagation of all Ψ_j using the algorithms from [2] and [1]
    for all $\Psi_j(t = \tau_i)$ **do**
      $\Psi_j(t = \tau_{i+1}) :=$ **time_propagation**$(V, \Psi_j(t = \tau_i))$
    end for
  end for

Chapter 7

General algorithm for spawning and propagation

In this chapter we try to develop a generalized algorithm for combined propagation and spawning of wavepackets. The idea is to extend the algorithm 9 to be applicable to arbitrary non-adiabatic potentials. There are several interesting open questions and we can state some hopefully worthwhile goals for further research.

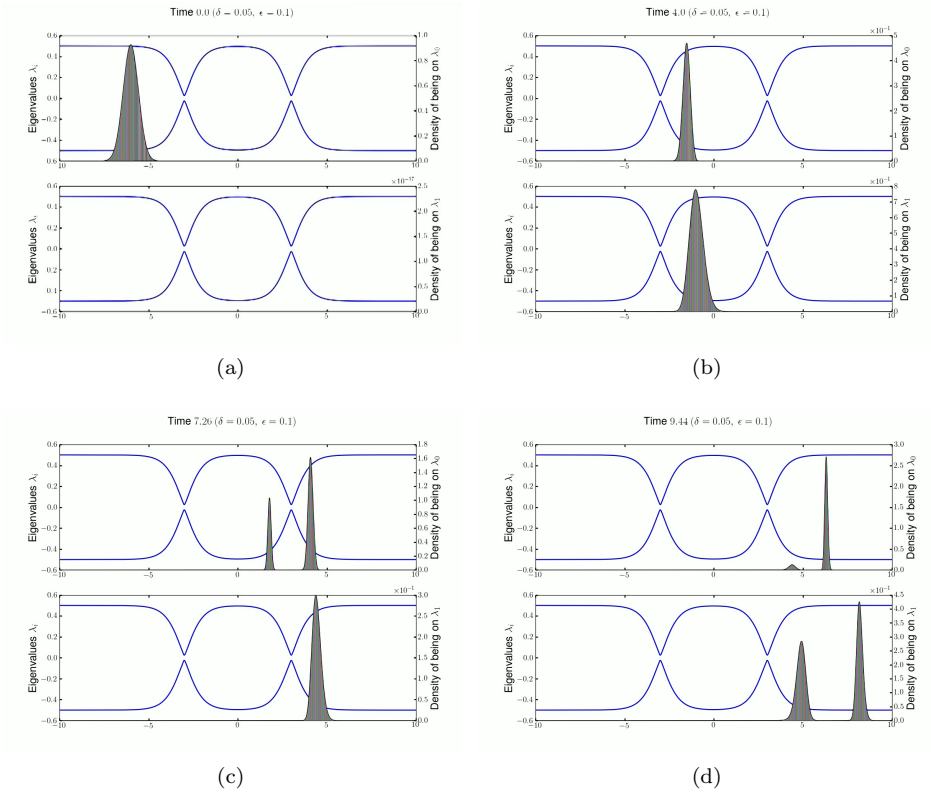
7.1 Motivating example

An example motivating the complicated algorithms developed in this chapter is now examined in much detail. The potential we analyze has two energy levels λ_0 and λ_1 and is given by the following matrix

$$V(x) = \begin{pmatrix} \tanh(\rho+x)\tanh(x-\rho) & \delta \\ \delta & -\tanh(\rho+x)\tanh(x-\rho) \end{pmatrix}$$

where $\rho \in \mathbb{R}$ and δ is the gap size again. In the example we set $\rho = 3$. The potential is shown in the background of figure 7.1. This figure shows four snapshots from a simulation. In the first panel we see the initial configuration where we start with $|\Psi_0\rangle = \phi_0$ on the upper level and to the left of both avoided crossings. The wavepacket has a momentum to the right. When the packet passed the first crossing it will split up into two parts, one on each energy surface. Already now we could ask why we want to keep both parts inside the same mathematical wavepacket. After the last chapters a split and spawn procedure comes into mind immediately. If we now split up the packet Ψ_0 according to $\Psi_0 \to \Psi_1 + \Psi_2$ we end up with two packets after the first crossing which are mathematically independent but still can interact physically. (We also set the characteristic component χ for both packets according to $\chi_{\Psi_1} := 0$ and $\chi_{\Psi_2} := 1$ thus the packet Ψ_1 is the part on the upper level.)
The Ψ_2 on the lower level moves faster than the one on the upper level. And thus the two packets will separate further and further in the long run. For this reason our decision to split and spawn was right. If the potential had only a single crossing we could stop now, but this potential has two crossings and we have to go on with our thought experiment. The packet on the lower level will

enter the region of the second avoided crossing first. We assume the linear part between the two crossings being sufficiently long such that the packets have separated enough to enter the second crossing after each other. This is true for the example shown below. As now Ψ_2 passed the second crossing a part of it will go to the upper level λ_0 again hence the packet splits once more. If we follow this split mathematically and change the representation we have to spawn again and write $\Psi_2 \to \Psi_3 + \Psi_4$ with $\chi_{\Psi_3} := 0$ and $\chi_{\Psi_4} := 1$. At that time we already have three active wavepackets, namely $\{\Psi_1, \Psi_3, \Psi_4\}$ and the first two of these reside on the upper level.
Some time later the back-most packet Ψ_1 on the upper level will finally enter the second crossing. And you guess it, part of this packet will jump to the lower level. Once more we should split and formally write $\Psi_1 \to \Psi_5 + \Psi_6$ where $\chi_{\Psi_5} := 0$ and $\chi_{\Psi_6} := 1$. (These assignments are of course arbitrary and solely change the notation.)



(a) (b)

(c) (d)

Figure 7.1: The potential here has two avoided crossings. We start with a single Gaussian wavepacket on the upper level coming from the left. We see that at each crossing the packet splits up into two new Gaussian packets. This should be motivation enough for a general spawning algorithm. Notice the different scales on the y axis for both energy levels!

We see that already a relatively simple potential exhibits very complex dynamics. We start with a single wavepacket Ψ_0 which has a Gaussian shape and end

up with *four* wavepackets $\{\Psi_3, \Psi_4, \Psi_5, \Psi_6\}$ at the end of the day. The final wavefunction after both crossings is then given by $\Psi_f = \Psi_3 + \Psi_4 + \Psi_5 + \Psi_6$. It should be clear by now that it does not make any sense to keep these four parts inside the same (mathematical) wavepacket $|\Psi\rangle$. What would the optimal choice of the parameter set Π be for this packet? Additionally we would need an exceedingly large basis to represent all parts accurately. (This is by the way the reason why the above simulation was not possible with the wavepacket based algorithms from [2] and [1] for small but interesting gap sizes of $\delta \approx \varepsilon$. The example stems from a simulation done with Strang splitting of the propagation operator.) Just for fun we could briefly imagine a potential matrix like the following one

$$V(x) = \begin{pmatrix} \prod_i \tanh(x-\rho_i) & \delta \\ \delta & -\prod_i \tanh(x-\rho_i) \end{pmatrix}$$

for a whole set $\{\rho_i\}_{i=1}^n$. This potential consists of n avoided crossings in series. And the initial wavepacket $|\Psi_0\rangle$ could in the worst case split up into 2^n packets because we can double the number of packets at each crossing!

7.2 Some notations

For the next section we have to extend our notations around wavepackets with some smaller additions. The (homogeneous) vector valued wavepacket having N components is given by (copied from (2.14))

$$|\Psi\rangle := |(\Phi_0, \ldots, \Phi_{N-1})\rangle \ . \tag{7.1}$$

Now we define the restriction operator which reduces the wavepacket to a single component Φ_j by

$$|\Psi|_j\rangle := |(0, \ldots, 0, \Phi_j, 0, \ldots, 0)\rangle \ . \tag{7.2}$$

Thus the wavepacket Ψ gets reduced to the component that is localized on the energy level λ_j. In principle this is the same as extracting Φ_j from Ψ but with this new definition we keep the vector structure of Ψ.
The definition of the restriction to the complement works alike. If $j \in [0, N-1]$ then the complement $\bar{j}$ is defined as $\bar{j} := [0, N-1] \setminus \{j\}$ and therefore

$$\left|\Psi|_{\bar{j}}\right\rangle := |(\Phi_0, \ldots, \Phi_{j-1}, 0, \Phi_{j+1}, \ldots, \Phi_{N-1})\rangle \ . \tag{7.3}$$

With this operator we can zero out the component Φ_j of Ψ and remove the part localized on the energy level λ_j.

7.3 Wavepacket superpositions and interactions

Presume for this section that our wavefunction φ is represented not by a single homogeneous or inhomogeneous wavepacket Ψ but instead by a linear combination thereof. Formally we write

$$|\varphi\rangle = |\Psi_0\rangle + |\Psi_1\rangle + \ldots + |\Psi_{J-1}\rangle = \sum_{j=0}^{J-1} |\Psi_j\rangle =: |\Upsilon\rangle \tag{7.4}$$

and use this as primary definition for $|\Upsilon\rangle$ which we call a *superposition*. In the motivating example we have as initial condition $|\Upsilon\rangle = \Psi_0$ and after the packets passed both crossings $|\Upsilon\rangle = \Psi_3 + \Psi_4 + \Psi_5 + \Psi_6$. As we already mentioned above the packets Ψ_i could still interact with each other and this is what we try to capture with this new definition. With the spawning operations we split up the initial packet Ψ_0 and put all parts in individual wavepackets Ψ_i with possibly different parameter sets Π_i (do not confuse this with inhomogeneous wavepackets where each component has its own parameter set) and different coefficient vectors. Even if it seems contrary to what we want to achieve we now gather all parts again and put them within $|\Upsilon\rangle$. But this effort is spent to model wavepacket interactions which can become very important in the non-adiabatic case.

7.3.1 Observable computation

Computing observables of a wavefunction is in principle a rather trivial thing. We just write the braket $\langle\varphi\,|\,O\,|\,\varphi\rangle$ with the corresponding operator O and compute the integral. If we want to find the norm of φ then the operator becomes the identity. Let's return to the example from 7.1 and see how we have to compute the norm there. For the initial configuration it is obvious the $\langle\varphi\,|\,\varphi\rangle = \langle\Psi_0\,|\,\Psi_0\rangle$ but for the final configuration where we have four wavepackets we can now use $|\Upsilon\rangle = \Psi_3 + \Psi_4 + \Psi_5 + \Psi_6$ and easily get correct results. Consider the general case

$$\begin{aligned}
\langle\varphi\,|\,\varphi\rangle &:= \langle\Upsilon\,|\,\Upsilon\rangle && (7.5)\\
&= \left\langle \sum_{s=0}^{J-1} \Psi_s \,\middle|\, \sum_{t=0}^{J-1} \Psi_t \right\rangle && (7.6)\\
&= \sum_{s=0}^{J-1}\sum_{t=0}^{J-1} \langle\Psi_s\,|\,\Psi_t\rangle && (7.7)
\end{aligned}$$

and see that the norm of the wavefunction is *not* just the sum of the norms of all contributing wavepackets. (It should be evident from quantum mechanics of course.) This shows that we have to include the off-diagonal terms where $s \neq t$. The brakets $\langle\Psi_s\,|\,\Psi_t\rangle$ can be computed by inhomogeneous quadrature as shown in chapter 3. For the computation of energies we have to put the Hamiltonian operator H inside the braket

$$\begin{aligned}
E &:= \langle\Upsilon\,|\,H\,|\,\Upsilon\rangle = \langle\Upsilon\,|\,T+V\,|\,\Upsilon\rangle && (7.8)\\
&= \sum_{s=0}^{J-1}\sum_{t=0}^{J-1} \langle\Psi_s\,|\,T\,|\,\Psi_t\rangle + \sum_{s=0}^{J-1}\sum_{t=0}^{J-1} \langle\Psi_s\,|\,V\,|\,\Psi_t\rangle\ . && (7.9)
\end{aligned}$$

Apparently we get off-diagonal terms also in the computation of kinetic and potential energies. And these terms where $s \neq t$ are responsible for the interaction

of wavepackets and make the whole thing very complicated. Even if the formulae seem to be simple we have to compute $\mathcal{O}\left(J^2\right)$ pairs in practice. In general we can not neglect these terms otherwise we violate the energy conservation.
However there is some hope. Reconsider the tunneling simulations. There we had, expressed in our extended notation

$$|\Upsilon_{\text{tunnel}}\rangle := |\Psi_{\text{reflected}}\rangle + |\Psi_{\text{transmitted}}\rangle \ . \tag{7.10}$$

Since the two parts are separated by the potential hill their interaction will decay very rapidly and for moderately large times after tunneling occurred we can assume that both packets are physically independent of each other. In that case we can drop the off-diagonal terms and write

$$\begin{aligned} \langle \varphi_{\text{tunnel}} \,|\, \varphi_{\text{tunnel}} \rangle &= \langle \Upsilon_{\text{tunnel}} \,|\, \Upsilon_{\text{tunnel}} \rangle \\ &\approx \langle \Psi_{\text{reflected}} \,|\, \Psi_{\text{reflected}} \rangle + \langle \Psi_{\text{transmitted}} \,|\, \Psi_{\text{transmitted}} \rangle \ . \end{aligned} \tag{7.11}$$

And the same holds for the energies. This is exactly what we did in chapter 5 where we completely neglected the interaction of the two parts. In the non-adiabatic case one can not do a similar approximation. In the general case because wavepackets are highly localized in space one could try to exclude pairs whose packets are far away. A first primitive approach could look like the following inequality

$$|q_s - q_t| > \xi \left(\frac{1}{2} \left\langle \Psi_s \,\middle|\, (x - q_s)^2 \,\middle|\, \Psi_s \right\rangle + \frac{1}{2} \left\langle \Psi_t \,\middle|\, (x - q_t)^2 \,\middle|\, \Psi_t \right\rangle \right) \tag{7.12}$$

with $\xi > 1 \in \mathbb{R}$ is the safety factor. The formula determines if the distance between both packets is (much) larger than their summed position uncertainties. When choosing ξ large enough and this inequality is fulfilled one could probably drop the brakets mixing Ψ_s and Ψ_t. But we leave this question open for more rigorous future research.

7.4 Propagation of multiple wavepackets

The time propagation for superpositions $|\Upsilon\rangle$ of wavepackets also has to be generalized. The algorithm described in [2] can only propagate a single $|\Psi\rangle$. From the four steps in section 3.3 the steps 1, 2 and 4 do not pose any problem. In these steps only the parameter sets Π_j of the wavepackets Ψ_j are involved. We can perform the parameter set propagation for each packet Ψ_j individually and independently of the others. The tricky part is step 3 where we update the coefficients of the wavepackets. In this steps we have to take into account the possible interactions between *all* the Ψ_j in $|\Upsilon\rangle$. It is not clear how to do this yet and we should keep in mind that the solution should not destroy the advantages we obtained from the splitting and spawning procedure. Hence merging all Ψ_j back into a single wavepacket Ψ in not a feasible solution. So this is another open question for future research.

7.5 General spawning and propagation

In this very last section we sketch the outline of the spawning propagation algorithm with all the details from the last chapters included. Even if there are some white spots now we can describe the coarse layout of the procedure. Summarized in one sentence our task is: *Time propagation of a superposition* $|\Upsilon\rangle$ *of wavepackets through an arbitrary non-adiabatic potential* V *with exploiting the advantages of spawning new packets.* This sounds much easier than is really is. We start with a set $\mathcal{W}$ of wavepackets consisting of the J individual Ψ_j from the superposition $|\Upsilon\rangle$

$$\mathcal{W} := \{\Psi_j\}_{j=0}^{J-1} . \tag{7.13}$$

For real simulations most of the time we will start with a single wavepacket Ψ_0 and thus $\mathcal{W} := \{\Psi_0\}$. We then propagate all these packets in $\mathcal{W}$ through time and periodically check if there exists a packet Ψ_j that has a component Φ_l for which the spawning condition $\langle \Phi_l \,|\, \Phi_l \rangle \geq \tau^2$ is fulfilled. If yes then we split off this single component into a new wavepacket $|\Psi_*\rangle$ which we append to the set $\mathcal{W}$. The check of the spawning conditions is very expensive because we have to transform the packets to the eigenbasis. This is necessary because only there the components are decoupled. And decoupled components are a prerequisite for applying the spawning ideas. In the algorithm 10 we do this check in each timestep but needless to say that we can and should do this only every n-th step. Each time we check for spawning every packet Ψ_j could potentially split up into $N-1$ new packets. There may be situations where the number of packets explodes so we should build in a limit on how often we allow spawning and how many wavepackets we maximally propagate. Also if we split too often packets with really small overall norm could appear. At the time when we spawn the packet is guaranteed to have a norm of at least τ but during the propagation the norm may diminish. It would be possible just to remove such small wavepackets from further propagation but this naturally leads to a loss in the norm conservation. We think that with the algorithm presented at the end of this chapter the motivating example above finally should become tractable with wavepacket based ansatz.

Throughout this work we always discussed the splitting of wavepackets. Though for a general algorithm like the procedure in 10 combining the time propagation with spawning ideas it is nearly equally important to have a concept of merging wavepackets. This is just the opposite process of spawning. However there exists no theoretical background on that topic yet. How can we identify situations where merging two packets $\Psi_i + \Psi_j \to \Psi_k$ is appropriate? The parameter set Π_k for the wavepacket Ψ_k could be obtained by formulae similar to the one in algorithm 3 we used for the inhomogeneous quadrature. The coefficients c^i and c^j probably would have to be projected to the new basis Π_k with an algorithm similar to 6 and then summed up. We should set $\mu = \eta$ and project onto all basis functions $\{\phi_i[\Pi_k]\}_{i=0}^{\eta-1}$ of the new basis Π_k because parts of the packets Ψ_i or Ψ_j can possibly end up in the high frequency range. This poses one more open question for future research.

Algorithm 10 Spawning propagation method (general non-adiabatic case)

Require: A series t consisting of timesteps $\{\tau_0, \ldots, \tau_{\max}\}$
Require: The potential $V(x)$
Require: The initial set $\mathcal{W}$ of wavepackets
Require: A spawn threshold τ
  for all $\tau_i \in t$ **do**
    // Check each packet for spawning possibilities
    for all $\Psi_j \in \mathcal{W}$ **do**
      // Transform the packet to the eigenbasis
      $\Psi_j^e := M^{-1}\Psi_j^c$
      // Get the leading component
      $\chi_{\Psi_j^e} :=$ **leading_component**(Ψ_j^e)
      // Check the norms on all other energy levels
      for all $l \in \overline{\chi_{\Psi_j^e}}$ **do**
        // Extract the l-th component Φ_l of Ψ_j^e
        $w := \Psi_j^e|_l$
        // Check the spawning criterion
        if $\langle w \,|\, w \rangle \geq \tau^2$ **then**
          // Perform spawning procedure
          // Estimate the parameters of the fragment by algorithm 4
          $\tilde{\Pi} :=$ **estimate_parameters**(w)
          // Move the fragment w to the new basis by algorithm 5 or 6
          $\tilde{w} :=$ **change_basis**$(\tilde{\Pi}, w)$
          // Update the reminder (included in algorithm 5 and 6)
          $\tilde{v} :=$ **update_remainder**$(v, \tilde{w})$
          // We spawn one new wavepacket
          $\Psi_*[\tilde{\Pi}] := (0, \ldots, 0, \Phi_l := \tilde{w}, 0, \ldots, 0)$
          // And update the wavepacket Ψ_j^e
          $\Psi_j'^{,e} := (\Phi_0, \ldots, \Phi_{l-1}, \Phi_l := \tilde{v}, \Phi_{l+1}, \ldots, \Phi_{N-1})$
          // Set the characteristic component χ of Ψ_*
          $\chi_{\Psi_*} := l$
          // Transform the packets back into the canonical basis
          $\Psi_j'^{,c} := M\Psi_j'^{,e}$ and $\Psi_*^c := M\Psi_*^e$
          // Append the spawned packet and update the set of wavepackets
          $\mathcal{W} := \mathcal{W} \setminus \{\Psi_j^c\} \cup \{\Psi_j'^{,c}, \Psi_*^c\}$
        end if
      end for
    end for
    // Time propagation of all Ψ_j. We write this as a loop what suggests
    // independent handling of all packets in contrary to section 7.4
    // but the computation of interactions is hidden inside the function call.
    for all $\Psi_j\,(t = \tau_i) \in \mathcal{W}$ **do**
      $\Psi_j\,(t = \tau_{i+1}) :=$ **time_propagation**$(V, \mathcal{W})$
    end for
    // Gather all propagated wavepackets
    $\mathcal{W} := \bigcup_j \Psi_j\,(t = \tau_{i+1})$
  end for

Appendix A

Simulation Parameters

In this chapter we reprint the complete simulation configurations for the examples shown in the thesis. All listings are compatible input configuration files for the WaveBlocks simulation code [1] which was used to perform the simulations. The various algorithms presented during the last chapters are all implemented in the latest WaveBlocks code. For more information about this project and the simulation code see reference [3]. Simulation setup and WaveBlocks usage are described in the WaveBlocks manual in many more details than what would fit into this chapter.

A.1 Motivating tunneling example

The tunnelling example shown to motivate the spawn approach. Results of this simulation are shown in figure 4.2.

```
algorithm = "hagedorn"

potential = "eckart"

sigma = 100 * 3.8008 * 10**(-4.0)
a =  1.0/(2*0.52918)

T = 70
dt = 0.005

eps = 0.0234218**0.5

f = 9.0
ngn = 4096

basis_size = 512
leading_component = 0

P = 0.1935842258501978j
Q = 5.1657101481699996
S = 0.0
p = 0.24788547371
q = -7.55890450883
#p = eps**2 * 20 * 0.52918
#q = -4.0/0.52918
```

[1] The simulation code can be found at http://waveblocks.origo.ethz.ch

```
parameters = [ (P, Q, S, p, q) ]
coefficients = [[(0, 1.0)]]

write_nth = 20

matrix_exponential = "arnoldi"
arnoldi_steps = 15
```

The main simulation parameters are exactly the same as in reference [5].

A.2 More tunneling simulations

A simulation starting with a $|\phi_2\rangle$ wavepacket and a little less momentum than the one of $|\phi_0\rangle$.

A.2.1 Starting with a ϕ_2

```
algorithm = "hagedorn"

potential = "eckart"

sigma = 100 * 3.8008 * 10**(-4.0)
a =  1.0/(2*0.52918)

T = 70
dt = 0.005

eps = 0.0234218**0.5

f = 9.0
ngn = 4096

basis_size = 512
leading_component = 0

P = 0.1935842258501978j
Q = 5.1657101481699996
S = 0.0
p = 0.95*0.24788547371
q = -7.55890450883

parameters = [ (P, Q, S, p, q) ]
coefficients = [[(2, 1.0)]]

write_nth = 20

matrix_exponential = "arnoldi"
arnoldi_steps = 15
```

A.2.2 Starting with a ϕ_3

Another simulation this time starting with a $|\phi_3\rangle$ wavepacket.

```
algorithm = "hagedorn"

potential = "eckart"

sigma = 100 * 3.8008 * 10**(-4.0)
a =  1.0/(2*0.52918)
```

```
7
T = 70
9 dt = 0.005

11 eps = 0.0234218**0.5

13 f = 9.0
ngn = 4096
15
basis_size = 512
17 leading_component = 0

19 P = 0.1935842258501978j
Q = 5.1657101481699996
21 S = 0.0
p = 0.95*0.24788547371
23 q = -7.55890450883

25 parameters = [ (P, Q, S, p, q) ]
coefficients = [[(3, 1.0)]]
27
write_nth = 20
29
matrix_exponential = "arnoldi"
31 arnoldi_steps = 15
```

A.3 Parameters for figure 4.5

The parameters used for the simulation shown on figure 4.5 are given below.

```
1 algorithm = "hagedorn"

3 potential = "delta_gap"

5 T = 7
dt = 0.01
7
eps = 0.1
9
delta = 0.75*eps
11
f = 4.0
13 ngn = 4096

15 leading_component = 0
basis_size = 80
17
P = 1.0j
19 Q = 1.0-5.0j
S = 0.0
21
parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
23 coefficients = [[(2,1.0)], [(0,0.0)]]

25 write_nth = 1
```

A.4 Aposteriori spawning for tunneling simulations

A.4.1 Starting with a ϕ_0 and $K_0 = 50$

A simulation starting with a $|\phi_0\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant ↘
··· coefficients [K0, ..., Kmax]
spawn_K0 = 50

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.2 Starting with a ϕ_0 and $K_0 = 75$

A simulation starting with a $|\phi_0\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant ↘
··· coefficients [K0, ..., Kmax]
spawn_K0 = 75

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.3 Starting with a ϕ_0 and $K_0 = 100$

A simulation starting with a $|\phi_0\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant ↘
··· coefficients [K0, ..., Kmax]
spawn_K0 = 100

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.4 Starting with a ϕ_2 and $K_0 = 50$

A simulation starting with a $|\phi_2\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant coefficients [K0, ..., Kmax]
spawn_K0 = 50

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.5 Starting with a ϕ_2 and $K_0 = 75$

A simulation starting with a $|\phi_2\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant coefficients [K0, ..., Kmax]
spawn_K0 = 75

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.6 Starting with a ϕ_2 and $K_0 = 100$

A simulation starting with a $|\phi_2\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant coefficients [K0, ..., Kmax]
spawn_K0 = 100

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.7 Starting with a ϕ_3 and $K_0 = 50$

A simulation starting with a $|\phi_3\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant ↘
··· coefficients [K0, ..., Kmax]
spawn_K0 = 50

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.8 Starting with a ϕ_3 and $K_0 = 75$

A simulation starting with a $|\phi_3\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant ↘
··· coefficients [K0, ..., Kmax]
spawn_K0 = 75

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.9 Starting with a ϕ_3 and $K_0 = 100$

A simulation starting with a $|\phi_3\rangle$ wavepacket on the left of the hill. The change of the basis is done with the lumping method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant ↘
··· coefficients [K0, ..., Kmax]
spawn_K0 = 100

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "lumping"
```

A.4.10 Starting with a ϕ_0, set $K_0 = 50$ and $\mu = 1$

A simulation starting with a $|\phi_3\rangle$ wavepacket on the left of the hill. The change of the basis is done with the projection method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10
```

```
7 # Index of the coefficient defining the intervall of relevant ↘
  ··· coefficients [K0, ..., Kmax]
  spawn_K0 = 50
9
  # Spawn a packet $\phi_order$ by copying over the norm
11 spawn_method = "projection"

13 # Basis size of the spawned packet
  spawn_max_order = 1
```

A.4.11 Starting with a ϕ_0, set $K_0 = 50$ and $\mu = 3$

A simulation starting with a $|\phi_3\rangle$ wavepacket on the left of the hill. The change of the basis is done with the projection method.

```
  # The spawn algorithm
2 algorithm = "spawning_apost"

4 # Threshold for spawning (details see theory)
  spawn_threshold = 1e-10
6
  # Index of the coefficient defining the intervall of relevant ↘
  ··· coefficients [K0, ..., Kmax]
8 spawn_K0 = 50

10 # Spawn a packet $\phi_order$ by copying over the norm
  spawn_method = "projection"
12
  # Basis size of the spawned packet
14 spawn_max_order = 3
```

A.4.12 Starting with a ϕ_0, set $K_0 = 50$ and $\mu = 6$

A simulation starting with a $|\phi_3\rangle$ wavepacket on the left of the hill. The change of the basis is done with the projection method.

```
  # The spawn algorithm
2 algorithm = "spawning_apost"

4 # Threshold for spawning (details see theory)
  spawn_threshold = 1e-10
6
  # Index of the coefficient defining the intervall of relevant ↘
  ··· coefficients [K0, ..., Kmax]
8 spawn_K0 = 50

10 # Spawn a packet $\phi_order$ by copying over the norm
  spawn_method = "projection"
12
  # Basis size of the spawned packet
14 spawn_max_order = 6
```

A.4.13 Starting with a ϕ_0, set $K_0 = 50$ and $\mu = 12$

A simulation starting with a $|\phi_3\rangle$ wavepacket on the left of the hill. The change of the basis is done with the projection method.

```
# The spawn algorithm
algorithm = "spawning_apost"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-10

# Index of the coefficient defining the intervall of relevant coefficients [K0, ..., Kmax]
spawn_K0 = 50

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

# Basis size of the spawned packet
spawn_max_order = 12
```

A.5 Spawning propagation for tunneling simulations

The simulation configurations for the spawning propagation were automatically generated using the following meta configuration files.
Meta configuration file for different thresholds and using the lumping method.

```
# Global parameters that stay the same for all simulations:
GP = {}

GP["algorithm"] = "\"spawning_adiabatic\""
GP["potential"] = "\"eckart\""
GP["eps"] = "0.0234218**0.5"
GP["T"] = 70
GP["dt"] = 0.005
GP["parameters"] = "[ (0.19358422585019 78j, 5.16571014816999 96, 0.0, 0.247885473 71, -7.55890450883) ]"
GP["coefficients"] = [ [(0,1.0)] ]
GP["basis_size"] = 300
GP["leading_component"] = 0
GP["matrix_exponential"] = "\"pade\""
GP["ngn"] = 4096
GP["f"] = 9.0
GP["write_nth"] = 20

# Local parameters that change with each simulation
LP = {}

LP["K0"] = [80, 100, 120]
LP["spawn_threshold"] = [0.2, 0.225, 0.25, 0.275, 0.29, 0.3, 0.31, 0.315, 0.32, 0.325]
```

Meta configuration file for different thresholds and using the projection method.

```
# Global parameters that stay the same for all simulations:
GP = {}

GP["algorithm"] = "\"spawning_adiabatic\""
GP["potential"] = "\"eckart\""
GP["eps"] = "0.0234218**0.5"
GP["T"] = 70
GP["dt"] = 0.005
GP["parameters"] = "[ (0.19358422585019 78j, 5.16571014816999 96, 0.0, 0.247885473 71, -7.55890450883) ]"
```

```
GP["coefficients"] = [ [(0,1.0)] ]
GP["basis_size"] = 300
GP["leading_component"] = 0
GP["matrix_exponential"] = "\"pade\""
GP["ngn"] = 4096
GP["f"] = 9.0
GP["write_nth"] = 20
GP["spawn_normed_gaussian"] = False
GP["K0"] = 100

# Local parameters that change with each simulation
LP = {}

LP["spawn_threshold"] = [0.2, 0.225, 0.25, 0.275, 0.29, 0.3, 0.31, ↘
···0.315, 0.32, 0.325]
LP["spawn_max_order"] = [1, 3, 6, 12]
```

A.6 Basic avoided crossing simulations

A.6.1 Starting with a ϕ_0

A simulation starting with a $|\phi_0\rangle$ wavepacket on the upper level.

```
algorithm = "hagedorn"

potential = "delta_gap"

T = 7
dt = 0.01

eps = 0.1

delta = 0.75*eps

f = 4.0
ngn = 4096

leading_component = 0
basis_size = 80

P = 1.0j
Q = 1.0-5.0j
S = 0.0

parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
coefficients = [[(0,1.0)], [(0,0.0)]]

write_nth = 1
```

A.6.2 Starting with a ϕ_1

A simulation starting with a $|\phi_1\rangle$ wavepacket on the upper level.

```
algorithm = "hagedorn"

potential = "delta_gap"

T = 7
dt = 0.01
```

```
7
eps = 0.1
9
delta = 0.75*eps
11
f = 4.0
13 ngn = 4096

15 leading_component = 0
basis_size = 80
17
P = 1.0j
19 Q = 1.0-5.0j
S = 0.0
21
parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
23 coefficients = [[(1,1.0)], [(0,0.0)]]

25 write_nth = 1
```

A.6.3 Starting with a ϕ_2

A simulation starting with a $|\phi_2\rangle$ wavepacket on the upper level.

```
1 algorithm = "hagedorn"

3 potential = "delta_gap"

5 T = 7
dt = 0.01
7
eps = 0.1
9
delta = 0.75*eps
11
f = 4.0
13 ngn = 4096

15 leading_component = 0
basis_size = 80
17
P = 1.0j
19 Q = 1.0-5.0j
S = 0.0
21
parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
23 coefficients = [[(2,1.0)], [(0,0.0)]]

25 write_nth = 1
```

A.6.4 Starting with a ϕ_3

A simulation starting with a $|\phi_3\rangle$ wavepacket on the upper level.

```
1 algorithm = "hagedorn"

3 potential = "delta_gap"

5 T = 7
dt = 0.01
```

```
7
eps = 0.1
9
delta = 0.75*eps
11
f = 4.0
13 ngn = 4096

15 leading_component = 0
basis_size = 80
17
P = 1.0j
19 Q = 1.0-5.0j
S = 0.0
21
parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
23 coefficients = [[(3,1.0)], [(0,0.0)]]

25 write_nth = 1
```

A.6.5 Starting with a ϕ_4

A simulation starting with a $|\phi_4\rangle$ wavepacket on the upper level.

```
1 algorithm = "hagedorn"

3 potential = "delta_gap"

5 T = 7
dt = 0.01
7
eps = 0.1
9
delta = 0.75*eps
11
f = 4.0
13 ngn = 4096

15 leading_component = 0
basis_size = 80
17
P = 1.0j
19 Q = 1.0-5.0j
S = 0.0
21
parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
23 coefficients = [[(4,1.0)], [(0,0.0)]]

25 write_nth = 1
```

A.6.6 Starting with a linear combination of ϕ_0 and ϕ_1

A simulation starting with a linear combination $|\phi_0 + \phi_1\rangle$ on the upper level.

```
1 algorithm = "hagedorn"

3 potential = "delta_gap"

5 T = 7
dt = 0.01
```

```

eps = 0.1

delta = 0.75*eps

f = 4.0
ngn = 4096

leading_component = 0
basis_size = 80

P = 1.0j
Q = 1.0-5.0j
S = 0.0

parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
coefficients = [[(0,0.6),(1,0.8)], [(0,0.0)]]

write_nth = 1
```

A.6.7 Starting with a linear combination of ϕ_2 and ϕ_3

A simulation starting with a linear combination $|\phi_2 + \phi_3\rangle$ on the upper level.

```
algorithm = "hagedorn"

potential = "delta_gap"

T = 7
dt = 0.01

eps = 0.1

delta = 0.75*eps

f = 4.0
ngn = 4096

leading_component = 0
basis_size = 80

P = 1.0j
Q = 1.0-5.0j
S = 0.0

parameters = [ (P, Q, S, 1.0, -5.0), (P, Q, S, 1.0, -5.0) ]
coefficients = [[(2,0.6),(3,0.8)], [(0,0.0)]]

write_nth = 1
```

A.7 Aposteriori spawning for avoided crossing simulations

A.7.1 Aposteriori spawning with ϕ_0 and $k = 0$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

```

```
# Threshold for spawning (details see theory)
spawn_threshold = 1e-6

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

# The order used in the parameter estimation
spawn_order = 0

# Basis size of the spawned packet
spawn_max_order = 8

# Components for which spawning is tried
spawn_components = [1]
```

A.7.2 Aposteriori spawning with ϕ_1 and $k = 0$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-6

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

# The order used in the parameter estimation
spawn_order = 0

# Basis size of the spawned packet
spawn_max_order = 16

# Components for which spawning is tried
spawn_components = [1]
```

A.7.3 Aposteriori spawning with ϕ_1 and $k = 1$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-6

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

# The order used in the parameter estimation
spawn_order = 1

# Basis size of the spawned packet
spawn_max_order = 16

# Components for which spawning is tried
spawn_components = [1]
```

A.7.4 Aposteriori spawning with ϕ_2 and $k = 0$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-6
```

```
7 # Spawn a packet $\phi_order$ by copying over the norm
  spawn_method = "projection"
9
  # The order used in the parameter estimation
11 spawn_order = 0

13 # Basis size of the spawned packet
  spawn_max_order = 16
15
  # Components for which spawning is tried
17 spawn_components = [1]
```

A.7.5 Aposteriori spawning with ϕ_2 and $k = 2$

```
1 # The spawn algorithm
  algorithm = "spawning_apost_na"
3
  # Threshold for spawning (details see theory)
5 spawn_threshold = 1e-6

7 # Spawn a packet $\phi_order$ by copying over the norm
  spawn_method = "projection"
9
  # The order used in the parameter estimation
11 spawn_order = 2

13 # Basis size of the spawned packet
  spawn_max_order = 16
15
  # Components for which spawning is tried
17 spawn_components = [1]
```

A.7.6 Aposteriori spawning with ϕ_3 and $k = 0$

```
1 # The spawn algorithm
  algorithm = "spawning_apost_na"
3
  # Threshold for spawning (details see theory)
5 spawn_threshold = 1e-6

7 # Spawn a packet $\phi_order$ by copying over the norm
  spawn_method = "projection"
9
  # The order used in the parameter estimation
11 spawn_order = 0

13 # Basis size of the spawned packet
  spawn_max_order = 16
15
  # Components for which spawning is tried
17 spawn_components = [1]
```

A.7.7 Aposteriori spawning with ϕ_3 and $k = 3$

```
1 # The spawn algorithm
  algorithm = "spawning_apost_na"
3
  # Threshold for spawning (details see theory)
5 spawn_threshold = 1e-6

7 # Spawn a packet $\phi_order$ by copying over the norm
```

```
spawn_method = "projection"

# The order used in the parameter estimation
spawn_order = 3

# Basis size of the spawned packet
spawn_max_order = 16

# Components for which spawning is tried
spawn_components = [1]
```

A.7.8 Aposteriori spawning with ϕ_4 and $k = 0$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-6

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

# The order used in the parameter estimation
spawn_order = 0

# Basis size of the spawned packet
spawn_max_order = 16

# Components for which spawning is tried
spawn_components = [1]
```

A.7.9 Aposteriori spawning with ϕ_4 and $k = 4$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-6

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

# The order used in the parameter estimation
spawn_order = 4

# Basis size of the spawned packet
spawn_max_order = 16

# Components for which spawning is tried
spawn_components = [1]
```

A.7.10 Aposteriori spawning with $\phi_0 + \phi_1$ and $k = 0$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-6

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

```

```
# The order used in the parameter estimation
spawn_order = 0

# Basis size of the spawned packet
spawn_max_order = 32

# Components for which spawning is tried
spawn_components = [1]
```

A.7.11 Aposteriori spawning with $\phi_2 + \phi_3$ and $k = 0$

```
# The spawn algorithm
algorithm = "spawning_apost_na"

# Threshold for spawning (details see theory)
spawn_threshold = 1e-6

# Spawn a packet $\phi_order$ by copying over the norm
spawn_method = "projection"

# The order used in the parameter estimation
spawn_order = 0

# Basis size of the spawned packet
spawn_max_order = 64

# Components for which spawning is tried
spawn_components = [1]
```

Bibliography

[1] Raoul Bourquin, Vasile Gradinaru, and George Hagedorn. Non-adiabatic transitions near avoided crossings: theory and numerics. *Journal of Mathematical Chemistry*, pages 1–18, April 2011.

[2] Erwan Faou, Vasile Gradinaru, and Christian Lubich. Computing semiclassical quantum dynamics with hagedorn wavepackets, 2009.

[3] Vasile Gradinaru and Raoul Bourquin. WaveBlocks: Reusable building blocks for simulations with semiclassical wavepackets, 2010, 2011. http://waveblocks.origo.ethz.ch

[4] Vasile Gradinaru, George A Hagedorn, and Alain Joye. Exponentially accurate semiclassical tunneling wavefunctions in one dimension. *Journal of Physics A: Mathematical and Theoretical*, 43(47):474026, 2010.

[5] Vasile Gradinaru, George A. Hagedorn, and Alain Joye. Tunneling dynamics and spawning with adaptive semiclassical wave packets, 2010.

[6] George A. Hagedorn. Semiclassical quantum mechanics I: The $\hbar \to 0$ limit for coherent states. *Communications in Mathematical Physics*, 71(1):77–93, 1980.

[7] George A. Hagedorn. Semiclassical quantum mechanics III: The large order asymptotics and more general states. *Annals of Physics*, 135(1):58–70, 1981.

[8] George A. Hagedorn. Classification and normal forms for avoided crossings of quantum-mechanical energy levels. *Journal of Physics A: Mathematical and General*, 31:369, 1998.

[9] George A. Hagedorn. Raising and lowering operators for semiclassical wave packets. *Annals of Physics*, 269(1):77–104, 1998.

[10] George A. Hagedorn and Sam L. Robinson. Bohr-Sommerfeld quantization rules in the semiclassical limit. *J. Phys. A*, 31(50):10113–10130, 1998.

[11] George A. Hagedorn and Julio H. Toloza. Exponentially accurate semiclassical asymptotics of low-lying eigenvalues for 2×2 matrix schrödinger operators. *Journal of Mathematical Analysis and Applications*, 312(1):300–329, 2005.

[12] S. Teufel. *Adiabatic perturbation theory in quantum dynamics*. Springer Verlag, 2003.

[13] Bernd Thaller. *Visual Quantum Mechanics: Selected Topics with Computer Generated Animations of Quantum Mechanical Phenomena with Cdrom.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 2000.