Master Thesis

Fermionic contributions to non-equilibrium processes in quantum field theory

Julius-Maximilians Universität Würzburg

submitted by

Matteo Boser

March 2018

Supervisor: Prof. Dr. Haye Hinrichsen
Second assessor: Prof. Dr. Werner Porod

Zusammenfassung

Ziel dieser Arbeit war die Entwicklung eines Algorithmus zur Lösung der Kadanoff-Baym-Gleichungen (KBE) für fermionische Felder im Ortsraum. Dabei wurde der (1+1)-dimensionale Fall betrachtet. Zusammen mit einem entsprechenden, bereits implementierten Lösungsalgorithmus für skalare Felder wurde damit eine Grundlage zur Beschreibung von inhomogenen, Nicht-Gleichgewichts-Prozessen in der Quantenfeldtheorie geschaffen. Die KBE, wobei es sich um partielle Integro-Differentialgleichungen handelt, werden im Rahmen des 2PI-Formalismus hergeleitet und geben die exakte Zeitentwicklung des vollen Propagators wieder. Als Approximation wurde für die in den KBE auftretende Selbstenergie eine Loop-Entwicklung durchgeführt und Beiträge bis zur 2-Loop-Ordnung berücksichtigt. Für die numerische Lösung der KBE wurde das sogenannte *staggered leapfrog*-Verfahren verwendet, wodurch das Auftreten zusätzlicher, unphysikalischer Moden (*fermion doubling*) vermieden werden konnte. Desweiteren wurde aufgrund der komplizierten Struktur der KBE die Methode des *operator splittings* angewandt. Als abschliessender Test wurde mit Hilfe des entwickelten Algorithmus das Lineare Sigma-Model, welches eine Kopplung zwischen skalaren und fermionischen Feldern beinhaltet, in Bezug auf den Vorgang der (Pre-)Thermalisierung untersucht. Dabei konnnte ein universelles, also von den Details der Anfangsbedingungen unabhänginges Verhalten zu späteren Zeiten beobachtet werden.

Abstract

The goal of this thesis was to develop an algorithm for solving the Kadanoff-Baym-Equations for fermionic fields in position space. Thereby we have considered the (1+1) dimensional case. Combined with a respective, already implemented algorithm for scalar fields, we thereby established a basis for the description of inhomogeneous, non-equilibrium process in quantum field theory. The KBE, which are partial integro-differential equations, are derived in the framework of the 2PI formalism and incorporate the exact time evolution of the full propagator. As an approximation scheme we employed a loop expansion for the self energy appearing in the KBE up to 2-loop order. For solving the KBE numerically we used the so-called *staggered leapfrog* scheme, which avoids the emerge of additional, unphysical modes (*fermion doublers*). Further, due to the complex structure of the KBE we employed an *operator splitting* method. As a conclusive test we used the developed algorithm to investigate the *linear sigma model*, which couples scalar to fermion particles in the context of the process of (pre-)thermalization. Thereby we could observe a late-time behavior that is independent from the details of the initial condition, i.e. *universal*.

Contents

1. Introduction

The Golden Age of cosmology

We are currently in the 'Golden Age' of observational cosmology. The recent detection of gravitational waves by LIGO (*Laser Interferometer Gravitational-Wave Observatory*) in 2015 constitutes a groundbreaking achievement in astronomy [A$^+$16a]. Another example is the incredible high precision to which the CMB (*Cosmic Microwave Background*) - which opens an observational window to the early universe at a time about 380.000 years past the Big Bang - has been measured [A$^+$16b]. One of the most intriguing and promising projects in the near future is eLISA (*Evolved Laser Interferometer Space Antenna*). Encouraged by the recent success of the LISA Pathfinder program, this mission is expected to launch in the early 2030s. In contrast to the ground-based gravitational wave (GW) detectors like LIGO or VIRGO, eLISA will follow an Earth-trailing heliocentric orbit. It is built up by three identical spacecrafts which are arranged in an equilateral triangle and separated by a distance of 2.5 million kilometers. eLISA is not only able to probe the entire sky but will also open a new window in the low-frequency domain for the detection of GWs [A$^+$17].
Besides other aspects eLISA could become of particular interest for people dealing with the sce-

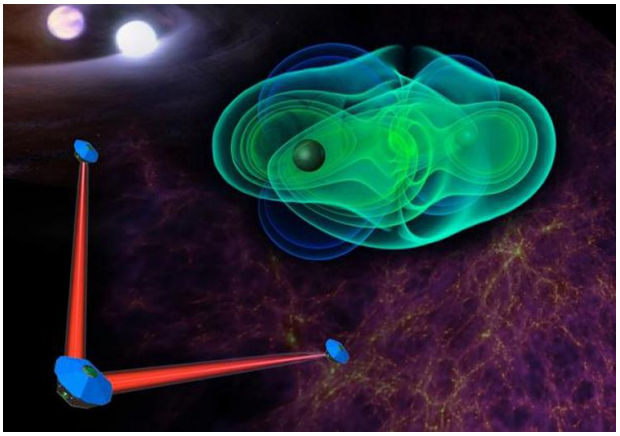


Figure 1.1.: Image taken from `http://www.lisamission.org`

nario of *Electroweak Baryogenesis* (EWBG), which is one of the most popular approaches to explain the matter-antimatter asymmetry observed today. The process of EWBG demands for a first order electroweak phase transition, which features the nucleation of bubbles. These bubbles are rapidly expanding and are therefore expected to collide with each other. These collisions

could function as a source for gravitational waves, which are expected to be within the detection range of eLISA [C$^+$16].

Electroweak Baryogenesis

A long known yet not understood phenomena is the baryon asymmetry that can be observed in the universe today. A common quantity to express the baryon asymmetry is the baryon to photon ratio η. Recent estimations yield [B$^+$13]:

$$\eta = \frac{\eta_b}{\eta_\gamma} = (6.19 \pm 0.14) \times 10^{-10}.$$

where $\eta_{b/\gamma}$ denotes the baryon/photon density respectively.
Any process that wants to describe the emergence of the measured matter-antimatter asymmetry must fulfill the following three conditions postulated by Sakharov back in 1967 [Sak91]:

- Baryon number violation
- C- and CP-violation
- Interactions outside of thermal equilibrium

The first condition is probably the most obvious and self-explanatory. The second is needed, because without C asymmetries in particle number densities every process that violates the Baryon number is of the same width as its C conjugate process. This argument can be pursued to find that in addition to C- also CP-violation is needed to eventually produce an excess of baryons over anti-baryons. For the third argument one can briefly assume we would deal with a system that is in thermal equilibrium. Since the particle and its respective antiparticle are of the same energy, both must obey the Fermi-Dirac-/Bose-Einstein (fermions/scalars) statistics in the same manner. Therefore any excess of either baryons or anti-baryons would be evened out by the respective process. This implies that the baryon-asymmetry originated in a universe out of equilibrium and in addition was preserved from being washed out after the universe reached thermal equilibrium.
One of the most prominent and still popular approaches to explain the observed asymmetry is the so-called *Electroweak Baryogenesis* (EWBG). This scenario starts out with a hot radiation-dominated early universe. When the universe cools down and reaches temperatures $T \lesssim 100\,\mathrm{GeV}$ the electroweak symmetry is spontaneously broken, i.e. the Higgs field develops a nonzero vacuum expectation value [MRM12]. For a successful EWBG it is necessary for the phase transition to be of first order. This implies the nucleation of bubbles, within which the eletroweak symmetry is broken, while the surrounding plasma is still in the symmetric phase. For illustrative purposes one can think of the forming of steam bubbles when water starts boiling.
A region of particular interest is the boundary between the two phases. Here particles from the plasma scatter with the expanding bubbles wall by means of both C and CP-violating processes. This is leading to non-equilibrium conditions in front of the bubble wall. The resulting asymmetries in particle number densities can then diffuse into the symmetric regime, where they can bias so-called *sphaleron processes*. These processes can convert the C-/CP- into B-asymmetries

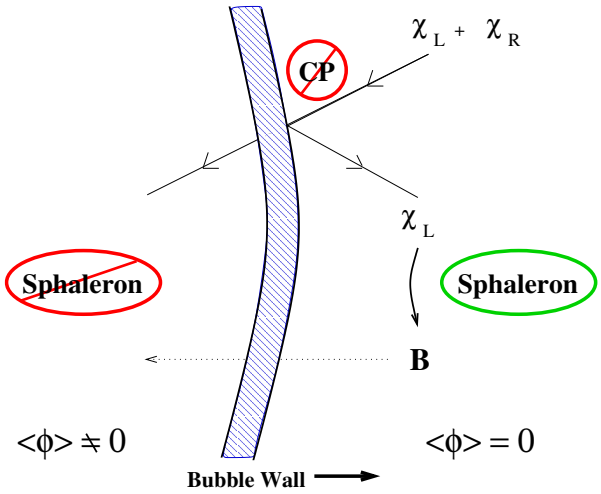


Figure 1.2.: Baryon production in front of the bubble walls. Image taken from [MRM12]

and are therefore B-number violating. Some of the emerging Baryon excess can get picked up by the expanding bubble and - since the sphaleron process is strongly suppressed within the broken phase - gets *frozen in*, i.e. conserved. This procedure is depicted in Fig. 1.2.
As an interim summary we can say that EWBG fulfills the three Sakharov conditions stated above and therefore constitutes a promising scenario. Yet while the Standard Model (SM) can provide the basic ingredients necessary for EWBG, there are obvious shortcomings demanding for new physics beyond the SM.
To give an example for these shortcomings we take a closer look at the nature of the electroweak phase transition. Within the SM alone a first order transition would lead to an upper bound on the Higgs mass of about $m_h \lesssim 95$ GeV [K$^+$96]. This is obviously not consistent with current measurements which yield $m_h \approx 125$ GeV [A$^+$15].
Another crucial point is that even if we would stick with the first order phase transition, the CP-violation within the SM induced by the CKM (*Cabibbo-Kobayashi-Maskawa*) matrix is too suppressed to explain the magnitude of the observed baryon asymmetry [HS95]. A more recent study suggests to cure this shortcoming via a time-varying Yukawa coupling, which results in a baryon asymmetry of the right order [BKS17].
In spite of these ambiguities there are good reasons why EWBG still receives so much attention. First of all, in contrast to other scenarios like *standard thermal leptogenesis* or *Affleck-Dine baryogenesis* EWBG is - due to its low-energy scale - in reach of collider experiments and therefore testable [MRM12]. Secondly - as mentioned earlier - the collision of the expanding bubbles might function as a source of gravitational waves that are potentially detectable by eLISA.

2. Non-equilibrium QFT

For the following brief introduction to non-equilibrium Quantum Field Theory, we primary follow the Lecture Notes of Jurgen Berges [Ber15]. We will work in natural units and therefore set $c = \hbar = k_{\mathrm{B}} = 1$.
To describe scenarios like EWBG and non-equilibrium processes in general one needs a framework that is able to render the dynamics of quantum fields out of equilibrium correctly. It should in particular be able to describe the process of *thermalization*, i.e. the approach of the physical system to thermal equilibrium. To be accurate, since thermal equilibrium itself is invariant under time-reflection it actually can never be reached from a non-equilibrium setup. It can yet be approached to a high degree [Ber15].
To actually set up a scope that is able to meet those requirements, only few ingredients are needed. We start by specifying a (non-equilibrium) density operator ϱ at the initial time t_0.
Since the expectation value of a arbitrary operator $\hat{A}$ is given by

$$\langle \hat{A} \rangle = \mathrm{Tr}\left[\varrho \hat{A}\right], \tag{2.1}$$

one could equivalently presuppose the knowledge of all correlation functions at the initial time.
Under the assumption of a closed system the time-evolution is then completely determined by the respective Hamiltonian $\hat{H}$:

$$\frac{\partial \varrho}{\partial t} = -i\left[\hat{H}, \varrho\right]. \tag{2.2}$$

Since standard pertubative approaches from ordinary QFT break down due to *secular* terms, which are spurious effects growing with time that invalidate the expansion, other approximation techniques have to be employed [Ber15]. The latter should not only be free of these *secular* terms but also obey the principle of *universality*. This means that the late-time dynamics should be universal in the sense that they depend on the initial energy density and other conserved charges only and are therefore insensitive to the details of the initial conditions.
An efficient and powerful approach that meets these requirements is the functional integral method of *n-particle irreducible effective actions*. For this purpose it is important to remember that the system's dynamics given by (2.2) can equivalently be described via a path integral containing the classical action S. This construction will allow us to derive an *effective action* Γ, which is the generating functional for the correlation functions of our theory.
We continue with outlining the basic idea of the so-called *Schwinger-Keldysh*-formalism, which will be employed in the subsequent introduction to the 2PI formalism and constitutes an important ingredient for the derivation of the Kadanoff-Baym-Equations.

For reasons of convenience we will start out with considering a purely scalar field theory and extend our considerations to fermion fields later on.

Schwinger-Keldysh-Contour

To begin, we write down the generating functional for correlation functions:

$$Z[J,R;\varrho_0] = \mathrm{Tr}\{\varrho_0 \mathrm{T}_\mathcal{C} e^{i\left(\int_{x,\mathcal{C}} J(x)\Phi(x) + \frac{1}{2}\int_{xy,\mathcal{C}} \Phi(x)R(x,y)\Phi(y)\right)}\} \tag{2.3}$$

where ϱ_0 denotes the initial density operator, $T_\mathcal{C}$ the time-ordering operator along the contour $\mathcal{C}$ and $\Phi(x)$ a (scalar) Heisenberg field operator. $J(x)$ and $R(x,y)$ are linear and bi-linear source terms respectively. The notation x denotes the 'full' space-time vector, while $\mathbf{x}$ represents the spatial components only.

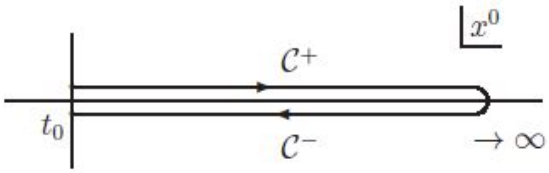


Figure 2.1.: Real-time contour $\mathcal{C}$. Image taken from [Ber15]

As depicted in Fig. 2.1, the time-integration along $\mathcal{C}$ appearing in (2.3) starts at an initial value t_0, advances to some specific moment in the future and eventually returns back to t_0. This choice of the time contour is also referred to as the *In-In-formalism* or, honoring two of its early developers, as the *Schwinger-Keldysh-Contour* (see e.g. [Sch61], [Kel64]). It stands in contrast to the so-called *In-Out-Formalism*, which introduces asymptotic In- and Out-states and is usually employed to describe scattering processes.
We also want to stress that this is not to be confused with an imaginary-time formalism, which is often used for calculations in thermal equilibrium [ZJ00]. The offset of the contour from the real time-axes in Fig. 2.1 is for demonstrative purposes only.
To define non-equilibrium correlation functions we take the functional derivative of (2.3) with respect to the source terms:

$$\left.\frac{\delta Z[J,R;\varrho]}{\delta J(x)}\right|_{J,R=0} = \mathrm{Tr}\{\varrho\Phi(x)\} \equiv \langle\Phi(x)\rangle \equiv \phi(x).$$

This can straightforwardly be extended to higher order correlation functions, e.g. the two point function:

$$\left.\frac{\delta Z[J,R;\varrho]}{\delta J(x)\delta J(y)}\right|_{J,R=0} = \mathrm{Tr}\{\varrho \mathrm{T}_\mathcal{C}\Phi(x)\Phi(y)\} \equiv \langle \mathrm{T}_\mathcal{C}\Phi(x)\Phi(y)\rangle$$

This implies that the correlation functions are defined by the expectation value of time ordered products of Heisenberg operators. Further, since in this way we can obtain any correlation func-

tion we want, the considered quantum system is specified by (2.3) completely.
The time-ordered product of the two Heisenberg field operators can further be re-expressed in terms of *contour Heaviside step functions*:

$$\mathrm{T}_{\mathcal{C}}\Phi(x)\Phi(y) = \Phi(x)\Phi(y)\theta_{\mathcal{C}}(x^0 - y^0) + \Phi(y)\Phi(x)\theta_{\mathcal{C}}(y^0 - x^0),$$

where the latter are defined by:

$$\theta_{\mathcal{C}}(x^0 - y^0) = \begin{cases} \theta(x^0 - y^0) & x^0, y^0 \in \mathcal{C}^+ \\ \theta(y^0 - x^0) & x^0, y^0 \in \mathcal{C}^- \\ 0 & x^0 \in \mathcal{C}^+, y^0 \in \mathcal{C}^- \\ 1 & x^0 \in \mathcal{C}^-, y^0 \in \mathcal{C}^+ \end{cases}$$

where $\theta(x^0 - y^0)$ denotes the standard *Heaviside step function*.
In complete analogy we can define the respective fermion propagator:

$$\Delta_{\alpha\beta}(x, y) \equiv \langle \mathrm{T}_{\mathcal{C}}\Psi_\alpha(x)\bar{\Psi}_\beta(y)\rangle = \langle \Psi_\alpha(x)\bar{\Psi}_\beta(y)\rangle\theta_{\mathcal{C}}(x^0 - y^0) - \langle \bar{\Psi}_\beta(y)\Psi_\alpha(x)\rangle\theta_{\mathcal{C}}(y^0 - x^0)$$

where the minus sign between the two terms is a consequence of the anti-commuting property of the fermionic field operators. Due to notational convenience we will hereafter mostly omit the Dirac indices.
We continue by introducing the spectral function ρ and the statistical function F:

$$\begin{aligned} \rho_{\alpha\beta}(x, y) &= i\langle\{\Psi_\alpha(x), \bar{\Psi}_\beta(y)\}\rangle \\ F_{\alpha\beta}(x, y) &= \frac{1}{2}\langle[\Psi_\alpha(x), \bar{\Psi}_\beta(y)]\rangle. \end{aligned} \tag{2.4}$$

The advantage of expressing the propagator this way is that the two quantities ρ and F allow for a simple physical interpretation. While the spectral function contains information about the spectrum of the system, the statistical function encodes information about occupation numbers.
To find the relation between ρ and F and the full propagator, we split Δ into four distinctive parts: $\Delta^{++}, \Delta^{+-}, \Delta^{-+}$ and Δ^{--}. The superscripts $\pm$ thereby refer to the position of the x^0 and the y^0 coordinate with respect to the time contour $\mathcal{C}$. Using (2.4) we can express the different parts as follows:

$$\begin{aligned} \Delta^{++}(x, y) &= F(x, y) - \frac{i}{2}\rho(x, y)\mathrm{sgn}(x^0 - y^0) \\ \Delta^{+-}(x, y) &= F(x, y) + \frac{i}{2}\rho(x, y) \\ \Delta^{-+}(x, y) &= F(x, y) - \frac{i}{2}\rho(x, y) \\ \Delta^{--}(x, y) &= F(x, y) + \frac{i}{2}\rho(x, y)\mathrm{sgn}(x^0 - y^0) \end{aligned}$$

where $\text{sgn}(x^0 - y^0) = \theta(x^0 - y^0) - \theta(y^0 - x^0)$ denotes the standard sign function.
Since there are only two linear independent parts, the full propagator can be written in terms of ρ and F:

$$\Delta(x,y) = F(x,y) - \frac{i}{2}\rho(x,y)\text{sgn}_{\mathcal{C}}(x^0 - y^0) \tag{2.5}$$

where $\text{sgn}_{\mathcal{C}}(x^0 - y^0) = \theta_{\mathcal{C}}(x^0 - y^0) - \theta_{\mathcal{C}}(y^0 - x^0)$.
For the sake of completeness, we also state the respective expressions for the statistical and spectral function in case of scalars:

$$\begin{aligned} \rho_\phi(x,y) &= i\langle[\Phi(x), \Phi(y)]\rangle \\ F_\phi(x,y) &= \frac{1}{2}\langle\{\Phi(x), \Phi(y)\}\rangle \end{aligned} \tag{2.6}$$

As apparent from (2.6), the statistical and spectral functions swap roles. This means while in case of the fermions the spectral/statistical function is proportional to the anti-commutator/commutator of the field operators, they are proportional to the commutator/anti-commutator in case of the scalars.
In analogy to the fermions we can now write the scalar propagator G as a linear combination of ρ_ϕ and F_ϕ:

$$G(x,y) = F_\phi(x,y) - \frac{i}{2}\rho_\phi(x,y)\text{sgn}_{\mathcal{C}}(x^0 - y^0) \tag{2.7}$$

2.1. 2PI effective action

2.1.1. Initial conditions

In the following we want to give a brief introduction to the 2PI-effective action. We will again start by considering a purely scalar field theory.
Since it will turn out to be convenient, we rewrite our expression for the generating functional (2.3). To be specific, we use that we can represent the trace as a functional integral [Ber15]:

$$Z[J,R;\varrho_0] = \int [\mathrm{d}\varphi_0^+][\mathrm{d}\varphi_0^-]\langle\varphi_0^+|\varrho_0|\varphi_0^-\rangle \int_{\varphi_0^+}^{\varphi_0^-} \mathscr{D}\varphi e^{i\left\{S[\varphi] + \int_{x,\mathcal{C}} J(x)\varphi(x) + \frac{1}{2}\int_{xy,\mathcal{C}} \varphi(x)R(x,y)\varphi(y)\right\}} \tag{2.8}$$

where $\varphi_0^\pm = \varphi^\pm(x^0 = t_0, \mathbf{x})$ with $x^0 \in \mathcal{C}^\pm$ correspond to the field configurations at the initial time t_0 and $S[\varphi]$ denotes the classical action for a real scalar field φ. For the details of this path integral construction we redirect the interested reader to [Ber15].
This representation of the generating functional reflects two central features of non-equilibrium quantum field theory. While the matrix elements of the initial density operator occur as prefactors and reflect the statistical fluctuations connected to the initial density matrix, the functional

integral - containing the classical action $S[\varphi]$ - encodes the quantum fluctuations [Ber15].
To describe the initial conditions of an experimental setup, we have to specify the initial density matrix ϱ_0. Without loss of generality we can express the matrix elements of ϱ_0 as an exponential of a polynomial in the fields [C$^+$94]:

$$\langle\varphi_0^+|\varrho_0|\varphi_0^-\rangle = \exp\left[\alpha^{(0)} + \int \mathrm{d}^3\mathbf{x}\alpha_a^{(1)}(\mathbf{x})\varphi_0^a(\mathbf{x}) + \frac{1}{2}\int \mathrm{d}^3\mathbf{x}\mathrm{d}^3\mathbf{y}\alpha_{ab}^{(2)}(\mathbf{x},\mathbf{y})\varphi_0^a(\mathbf{x})\varphi_0^b(\mathbf{y}) + \ldots\right]. \tag{2.9}$$

where the Latin letters $a, b, ..$ take the values $\{+,-\}$. The first coefficient $\alpha^{(0)}$ plays the role of an irrelevant constant and can therefore be used for normalization.
Since the initial density matrix can only influence the path integral at the initial time t_0, we can further write:

$$\begin{aligned}
&\int \mathrm{d}^3\mathbf{x}\alpha_a^{(1)}(\mathbf{x})\varphi_0^a(\mathbf{x}) \equiv \int_{x,\mathcal{C}} \alpha^{(1)}(x)\varphi(x) \\
&\int \mathrm{d}^3\mathbf{x}\mathrm{d}^3\mathbf{y}\alpha_{ab}^{(2)}(\mathbf{x},\mathbf{y})\varphi_0^a(\mathbf{x})\varphi_0^b(\mathbf{y}) \equiv \int_{x,\mathcal{C}} \alpha^{(2)}(x,y)\varphi(x)\varphi(y) \\
&\vdots
\end{aligned}$$

where the coefficients $\alpha_a^{(1)}, \alpha_{ab}^{(2)}, \ldots$ have to vanish identically for $t \neq t_0$.
In this work, we will restrict ourselves to *Gaussian* initial density matrices. This means we express the matrix elements in (2.9) as a polynomial of degree two, i.e. neglect all terms including higher powers of the initial field configuration φ_0. The parameters $\alpha_a^{(1)}, \alpha_{ab}^{(2)}, \ldots$ can be directly related to the initial values of the 1-point, 2-point,... function [C$^+$94]. This means a Gaussian initial density matrix is equivalent to the initial specification of the 1- and the 2-point function. As outlined in [Ber15], the knowledge of the lowest few correlation functions is often enough to render the initial conditions of an experimental setup sufficiently well, whereas higher correlation functions start to build up at later times. For example, the initial condition for the reheating dynamics after the end of inflation is assumed to be well-described by a Gaussian initial density matrix [Ber15].
Putting everything together and absorbing $\alpha^{(1)}$ and $\alpha^{(2)}$ into the source terms:

$$J(x) \to J(x) + \alpha^{(1)}(x), \qquad R(x,y) \to R(x,y) + \alpha^{(2)}(x,y),$$

we obtain:

$$Z[J,R;\varrho_0^{\text{gauss.}}] \longrightarrow Z[J,R] = \int \mathscr{D}\varphi e^{i\left(S[\varphi] + \int_{x,\mathcal{C}} J(x)\varphi(x) + \int_{xy,\mathcal{C}} \varphi(x)R(x,y)\varphi(y)\right)}$$

For the construction of an analogous path integral for fermion fields one has to introduce so-called *Grassman variables*, which reflect the anti-commuting properties of the fermionic field operators. Details can be found in [Ber15].

2.1.2. Effective action

We move on by defining the generating functional for connected n-point functions $W[J,R] = -i\ln Z[J,R]$. Once more we can obtain the macroscopic field as well as the connected propagator by taking the functional derivative with respect to the source terms:

$$\frac{\delta W[J,R]}{\delta J(x)} = \frac{1}{Z[J,R]}\frac{\delta Z[J,R]}{i\delta J(x)} = \phi(x)$$
$$\frac{\delta W[J,R]}{\delta R(x,y)} = \frac{1}{2}(\phi(x)\phi(y) + G(x,y))$$

Since the generating functional (2.3) can - as well known from thermodynamics - be seen as a generalization of the partition function applying a *Legendre transform* (LT) to the logarithm of $Z[J,R;\varrho]$ must lead to an equivalent description of the physical system.
Performing a LT of $W[J,R]$ with respect to the linear source term $J(x)$ yields the so-called 1-PI (*1-particle irreducible*) effective action:

$$\Gamma^{1\mathrm{PI}}[\phi] = W[J,R] - \int_{x,\mathcal{C}} \frac{\delta W[J,R]}{\delta J(x)} J(x) = W[J,R] - \int_{x,\mathcal{C}} \phi(x)J(x) \tag{2.10}$$

In the language of Feynman diagrams, a contribution is called *1-particle reducible* if by cutting one of the internal propagator lines the diagram falls apart in two separate pieces. Otherwise the diagram is called *1-particle irreducible*. This nomenclature can be extended to *n-particle irreducible* contributions. To illustrate this classification, a simple example is given in Fig. 2.2.

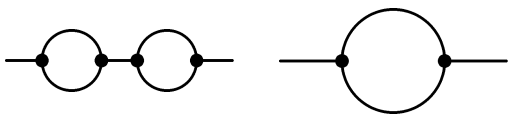


Figure 2.2.: Exemplary diagrams for 1-particle-reducible (left) and 1-particle-irreducible (right) contributions. When cutting the propagator between the two internal vertices in the left diagram, it falls apart in two disconnected pieces. For the right diagram one needs to cut two internal propagators to end up with two separate pieces, i.e. it is 2-particle-reducible.

After performing a subsequent LT on (2.10) with respect to the bi-linear source term $R(x,y)$, we

end up with the 2-PI effective action:

$$\begin{aligned}\Gamma^{\mathrm{2PI}}[\phi, G] &= W[J,R] - \int_{x,\mathcal{C}} \frac{\delta W[J,R]}{\delta J(x)} J(x) - \int_{xy,\mathcal{C}} \frac{\delta W[J,R]}{\delta R(x,y)} R(x,y) \\ &= W[J,R] - \int_{x,\mathcal{C}} \phi(x) J(x) - \int_{xy,\mathcal{C}} \frac{1}{2}(\phi(x)\phi(y) + G(x,y)) R(x,y)\end{aligned} \tag{2.11}$$

By varying this expression with respect to ϕ and G we obtain:

$$\begin{aligned}\frac{\delta \Gamma[\phi,G]}{\delta \phi} &= -J(x) - \int_{y,\mathcal{C}} R(x,y)\phi(y) \\ \frac{\delta \Gamma[\phi,G]}{\delta G} &= -\frac{1}{2} R(x,y)\end{aligned} \tag{2.12}$$

These are the quantum equations of motion for ϕ and G.
The 1-loop contribution can be written as [Ber15]:

$$\Gamma^{\mathrm{2PI}}_{\mathrm{1loop}}[\phi,G] = S[\phi] + \frac{i}{2}\mathrm{Tr}_{\mathcal{C}} \ln G^{-1} + \frac{i}{2}\mathrm{Tr}_{\mathcal{C}}\{(G_0^{-1}(\phi) - G^{-1})G\},$$

where $iG_0^{-1}(x,y;\phi) \equiv \delta^2 S[\phi]/\delta\phi(x)\delta\phi(y)$ denotes the classical inverse propagator.
Now we can express the exact 2-PI effective action as the sum of the 1-loop expression and an additional, higher loop-order term $\Gamma_2[\phi, G]$:

$$\Gamma^{\mathrm{2PI}}[\phi,G] = S[\phi] + \frac{i}{2}\mathrm{Tr}_{\mathcal{C}} \ln G^{-1} + \frac{i}{2}\mathrm{Tr}_{\mathcal{C}} G_0^{-1}(\phi)G + \Gamma_2[\phi,G] + \mathrm{const.}$$

While $\Gamma_2[\phi, G]$ includes all higher order contributions, the term $\frac{i}{2}\mathrm{Tr}_{\mathcal{C}}\{G^{-1}G\}$ can be seen as an irrelevant constant and be used for renormalization. We want to remind the reader that the trace $\mathrm{Tr}_{\mathcal{C}}$ includes not only the spatial integration along the time-contour $\mathcal{C}$ but also the summation over field indices.
Varying this expression with respect to the full propagator G and using the second line of (2.12) yields:

$$-\frac{1}{2}R(x,y) = -\frac{i}{2}G^{-1}(x,y) + \frac{i}{2}G_0^{-1}(x,y;\phi) + \frac{\delta\Gamma_2[\phi,G]}{\delta G} \tag{2.13}$$

$$G^{-1}(x,y) = G_0^{-1}(x,y;\phi) - iR(x,y) - \Pi(x,y;\phi,G) \tag{2.14}$$

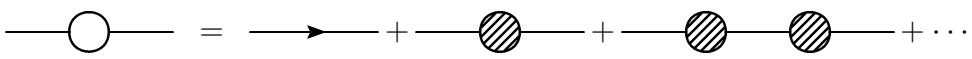
where we have defined the *proper self energy* as:

$$\Pi(x,y;\phi,G) = 2i\frac{\delta\Gamma_2[\phi,G]}{\delta G(x,y)}. \tag{2.15}$$

Inverting (2.14) using the geometric series yields:

$$G = (G_0^{-1} - iR)^{-1} + (G_0^{-1} - iR)^{-1}\Pi(G_0^{-1} - iR)^{-1} + \tag{2.16}$$
$$+ (G_0^{-1} - iR)^{-1}\Pi(G_0^{-1} - iR)^{-1}\Pi(G_0^{-1} - iR)^{-1} + \dots \tag{2.17}$$

where we have omitted the arguments for notational convenience.
The above equation for the full propagator G can also be expressed diagrammatically:



which is simply the *Dyson*-Equation. Thus we conclude that the proper self-energy is simply the sum of all 1-PI contributions. Further, since the self-energy is obtained by taking the functional derivative of $\Gamma^{2\mathrm{PI}}$ with respect to G - what can be interpreted as cutting one propagator line - we can see that all contributions of $\Gamma^{2\mathrm{PI}}$ have to be 2-PI.
In similar fashion one can derive the respective expression for the fermion propagator. With the source term set to zero, we obtain:

$$\Delta^{-1}(x,y) = \Delta_0^{-1}(x,y) - \Sigma(x,y;\Delta) \tag{2.18}$$

where $\Delta_0^{-1}(x,y)$ denotes the free inverse fermion propagator:

$$i\Delta_0^{-1}(x,y) = \delta_{\mathcal{C}}(x-y)[i\not{\partial}_x - m^f]$$

and $\Sigma(x,y)$ the proper self energy for the fermions:

$$\Sigma(x,y) = -i\frac{\delta\Gamma_2[\Delta]}{\delta\Delta(x,y)} \tag{2.19}$$

2.2. Evolution equations for the fermion propagator

In this section we will derive evolution equations, the so-called *Kadanoff-Baym-Equations* (KBE), for the fermionic two-point objects $\rho(x,y)$ and $F(x,y)$.
For this purpose we start from the equation of motion for the time-ordered fermion propagator given by (2.18):

$$\Delta^{-1}(x,y) = \Delta_0^{-1}(x,y) - \Sigma(x,y;\Delta)$$

Convoluting the latter expression with $\Delta(x,y)$ leads to:

$$\begin{aligned} i\delta(x-y) &= i\int_z \Delta^{-1}(x,z)\Delta(z,y) \\ &= \int_z i\Delta_0^{-1}(x,z)\Delta(z,y) - i\int_z \Sigma(x,z;\Delta)\Delta(z,y) \\ &= \int_z \delta_{\mathcal{C}}(x-z)\left[i\partial\!\!\!/_x - m^f\right]\Delta(z,y) - i\int_z \Sigma(x,z;\Delta)\Delta(z,y) \\ &= \left[i\partial\!\!\!/_x - m^f\right]\Delta(x,y) - i\int_z \Sigma(x,z;\Delta)\Delta(z,y) \end{aligned} \tag{2.20}$$

where we have employed the notation $\int_z = \int_{\mathcal{C}} \mathrm{d}z^0 \int \mathrm{d}\mathbf{z}$.
We continue separating the self energy into a local and a non-local part:

$$\Sigma(x,y;\Delta) = -i\Sigma^{\mathrm{local}}(x;\Delta)\delta(x-y) + \bar{\Sigma}(x,y;\Delta) \tag{2.21}$$

The local part $\Sigma^{\mathrm{local}}(x;\Delta)$ can then be absorbed by defining a space-time dependent, effective fermion mass $m^f(x,\Delta) = m^f + \Sigma^{\mathrm{local}}(x;\Delta)$. In the following we will use the notation $\Sigma(x,y;\Delta) \equiv \bar{\Sigma}(x,y;\Delta)$.
Using the decomposition of the fermion propagator in statistical and spectral function and turn our focus to the first term on the r.h.s of (2.20), we see:

$$\begin{aligned} &\left[i\partial\!\!\!/_x - m^f\right]\left(F(x,y) - \frac{i}{2}\rho(x,y)\mathrm{sgn}(x^0-y^0)\right) \\ &= \left[i\partial\!\!\!/_x - m^f\right]F(x,y) - \frac{i}{2}\mathrm{sgn}(x^0-y^0)\left[i\partial\!\!\!/_x - m^f\right]\rho(x,y) - \frac{i}{2}\rho(x,y)i\gamma^0\partial_{x^0}\mathrm{sgn}(x^0-y^0) \\ &= \left[i\partial\!\!\!/_x - m^f\right]F(x,y) - \frac{i}{2}\mathrm{sgn}(x^0-y^0)\left[i\partial\!\!\!/_x - m^f\right]\rho(x,y) + \frac{1}{2}2\rho(x,y)\gamma^0\delta(x^0-y^0) \\ &= \left[i\partial\!\!\!/_x - m^f\right]F(x,y) - \frac{i}{2}\mathrm{sgn}(x^0-y^0)\left[i\partial\!\!\!/_x - m^f\right]\rho(x,y) + i\gamma^0\gamma^0\delta(x-y) \\ &= \left[i\partial\!\!\!/_x - m^f\right]F(x,y) - \frac{i}{2}\mathrm{sgn}(x^0-y^0)\left[i\partial\!\!\!/_x - m^f\right]\rho(x,y) + i\delta(x-y) \end{aligned}$$

This way the δ-function on the l.h.s of (2.20) gets canceled.
Now we will take a closer look at the convolution term on the r.h.s. of (2.20). We therefore start with writing the self energy $\bar{\Sigma}$ as:

$$\bar{\Sigma}(x,y) = \Sigma^F(x,y) - \frac{i}{2}\Sigma^\rho(x,y)\mathrm{sgn}_{\mathcal{C}}(x^0-y^0), \tag{2.22}$$

which is in analogy to the decomposition of the propagator in statistical and spectral function in (2.5). Employing this decomposition yields:

$$\begin{aligned} i\int_z \bar{\Sigma}(x,z;\Delta)\Delta(z,y) = \int_z \Big\{ & i\Sigma^F(x,z)F(z,y) + \frac{1}{2}\Sigma^F(x,z)\rho(z,y)\mathrm{sgn}(z^0-y^0) \\ & + \frac{1}{2}\Sigma^\rho(x,z)F(z,y)\mathrm{sgn}(x^0-z^0) \\ & - \frac{i}{4}\Sigma^\rho(x,z)\rho(z,y)\mathrm{sgn}(x^0-z^0)\mathrm{sgn}(z^0-y^0)\Big\} \end{aligned} \tag{2.23}$$

The first term on the r.h.s. of (2.23) has to vanish because we integrate along a closed time path $\mathcal{C}$. This argument does not hold for the other terms due to the sign functions.
Instead we can split the closed time contour integral for the second term (and in complete analogy for the third term) in the following way:

$$\int_{\mathcal{C}} \mathrm{d}z^0 \mathrm{sgn}(z^0-y^0) = \int_{t_0}^{y^0} \mathrm{d}z^0(-1) + \int_{y0}^{t_0} \mathrm{d}z^0 = -2\int_{t_0}^{y^0} \mathrm{d}z^0,$$

In similar fashion we can also rewrite the time integration for the last term:

$$\int_{\mathcal{C}} \mathrm{d}z^0 \mathrm{sgn}(x^0-z^0)\mathrm{sgn}(z^0-y^0) \to 2\,\mathrm{sgn}(x^0-y^0)\int_{y^0}^{x^0} \mathrm{d}z^0$$

Putting everything together we end up with the following evolution equations for the statistical and the spectral function:

$$\left[i\partial\!\!\!/_x - m^{(f)}(x)\right] F(x,y) = \int_{t_0}^{x^0} dz\Sigma^\rho(x,z)F(z,y) - \int_{t_0}^{y^0} dz\Sigma^F(x,z)\rho(z,y) \tag{2.24}$$

$$\left[i\partial\!\!\!/_x - m^{(f)}(x)\right] \rho(x,y) = \int_{y^0}^{x^0} dz\Sigma^\rho(x,z)\rho(z,y) \tag{2.25}$$

These equations are also called the Kadanoff-Baym-Equations, honoring the early work by L. P. Kadanoff and G. A. Baym [KB62].
The r.h.s of (2.24) and (2.25) are also referred to as *memory integrals*, since they are integrating over the history of the system. We further introduce the abbreviations $M_F(x,y)$ and $M_\rho(x,y)$ for these integrals.
For the sake of completeness, we also state the respective KBE for the scalar case:

$$\left[\Box_x + m_\phi^2(x)\right] F_\phi(x,y) = -\int_{t_0}^{x^0} dz\Pi^\rho(x,z)F_\phi(z,y) + \int_{t_0}^{y^0} dz\Pi^F(x,z)\rho_\phi(z,y) \tag{2.26}$$

$$\left[\Box_x + m_\phi^2(x)\right] \rho_\phi(x,y) = -\int_{y^0}^{x^0} dz\Pi^\rho(x,z)\rho_\phi(z,y) \tag{2.27}$$

We want to stress that so far the given equations are exact. The approximation sets in, when we explicitly insert an expression for the respective self energies.
Further we want to remind the reader that in case of fermion fields we have again omitted the Dirac indices, i.e. the KBE actually exhibit a matrix structure.

3. Numerical Implementation

Since there is no analytic solution for the KBE in reach they have to be treated numerically. This has been done by several authors already, yet under the assumption of spatial homogeneity (see e.g. [Ber02],[BBW04], [BBS03], [LM08]). In other words, the considered systems obey translation invariance and therefore depend on the spatial distance between two points in space only. This setup allows for applying a Fourier Transformation and solving the KBE entirely in momentum space. This and further exploitation of symmetries can simplify the task of solving the KBEs dramatically.
Yet, since it is a long term goal to be able to describe setups comparable to that of the earlier described *Electroweak Baryogenesis*, assumptions like spatial homogeneity are no longer well justified. We are therefore convinced that the development of a framework, which is not limited to the time evolution of homogeneous systems, might allow for a more proper description of inhomogeneous non-equilibrium scenarios.
In the following we want to present the numerical tools and techniques that have been employed for the implementation of the Dirac equation and the Kadanoff-Baym-Equations. Even though so far only the case of (1+1)-dimensions has been implemented and therefore will be considered in the following, the generalization to (2+1)- or (3+1)- dimensions should be straightforward.
Since this work is in particular dedicated to the numerical treatment of fermion fields, we will restrict ourselves to the implementation details of the fermion evolution equations.
We will start out with introducing a discretization scheme for the Dirac equation, which is not only needed in case of the 1-point function, but also appears in the KBE. This will be followed by a brief discussion about the stability and the dispersion relation implied by the employed scheme. In the end of this chapter we will go into details of the algorithm that was implemented to solve the KBE, i.e. the evolution equations for the fermion propagator.

3.1. Discretizing the Dirac equation

3.1.1. Fermion doublers

To introduce the reader to employed discretization scheme, we will start out with the Dirac equation:

$$(i\partial\!\!\!/_x - m)\psi(\mathbf{x}, t) = 0$$

Going to (1+1) space-time dimensions, we can write:

$$i\gamma^0\partial_t \begin{pmatrix} r \\ s \end{pmatrix} = (-i\gamma^1\partial_x + m) \begin{pmatrix} r \\ s \end{pmatrix}. \tag{3.1}$$

where $r = r(\mathbf{x}, t)$ and $s = s(\mathbf{x}, t)$ are the components of the two-component Dirac-Spinor $\psi(\mathbf{x}, t)$, while $m = m(\mathbf{x}, t)$ denotes a space-time dependent mass.
For the γ-matrices we make the following choice that is kept throughout this work:

$$\gamma^0 = \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \gamma^1 = i\sigma_2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

where σ_i are the Pauli-matrices. As the reader might easily verify, γ^0 and γ^1 fulfill the so-called *Clifford Algebra*:

$$\{\gamma^\mu, \gamma^\nu\} = 2\eta_{\mu\nu}$$

where $\eta_{\mu\nu}$ denotes the metric tensor. We thereby use the signature $(+, -)$, i.e.:

$$\eta = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Multiplying (3.1) with γ^0 from the left, we end up with the so-called Schrödinger form:

$$\begin{aligned} i\partial_t \begin{pmatrix} r \\ s \end{pmatrix} &= (-i\partial_x\alpha + \gamma^0 m) \begin{pmatrix} r \\ s \end{pmatrix} \\ \Leftrightarrow \partial_t \begin{pmatrix} r \\ s \end{pmatrix} &= (-\partial_x\alpha - i\gamma^0 m) \begin{pmatrix} r \\ s \end{pmatrix} \end{aligned} \tag{3.2}$$

As one can see from the off-diagonal character of $\alpha = \gamma^0\gamma^1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, we end up with two coupled differential equations.
A first naive approach to discretize the spatial derivative would be the so called *midpoint scheme*. Considering the component r and setting the mass term m set to zero, we obtain:

$$\partial_t r_j^n = -\frac{s_{j+1}^n - s_{j-1}^n}{2\Delta x}. \tag{3.3}$$

It is now instructive to compare the respective dispersion relations for the continuum and the discretized equation:

$$\begin{aligned} &\text{discretized: } \omega = \pm\frac{\sin(k\Delta x)}{\Delta x}, && -\frac{\pi}{\Delta x} < k \leq \frac{\pi}{\Delta x} \\ &\text{continuum: } \omega = \pm k, && -\infty < k < \infty \end{aligned}$$

As apparent from Fig.3.1, in contrast to the continuum case the dispersion relation for the discretized equation is intersected twice by the given energy value $\omega = 0.5$. This occurrence of

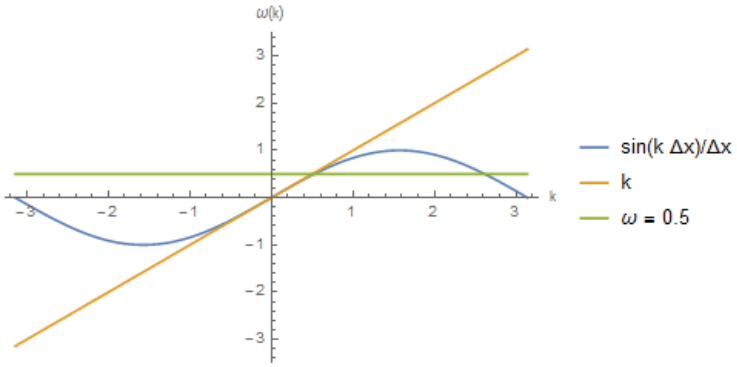


Figure 3.1.: Dispersion relation for the continuum (orange) and the discretized (blue) massless Dirac equation. The lattice spacing Δx was set to 1. In case of the discretized equation there are two intersections/solutions for a given energy $\omega = 0.5$ (green).

additional modes due to this kind of discretization scheme is also referred to as the *fermion doubling problem*. This problem already becomes apparent when taking a closer look back at (3.3). In this case the spatial derivative is calculated via two lattice points which are separated by a distance of $2\Delta x$. This means that the maximum lattice momentum $p = \frac{\pi}{\Delta x}$, i.e. a mode oscillating between each grid point can never be resolved. The latter is rather 'seen' as another zero mode. This applies for all higher modes ($p > \frac{\pi}{2\Delta x}$), meaning they effectively get mapped onto lower modes $p < \frac{\pi}{2\Delta x}$ and therefore lead to the doubling of states (comp. Fig 3.1).
Since this is a problem that has been known for some time, different techniques on how to get rid of these undesired modes have been developed and are available in the literature. One approach suggests to add an additional mass term, the so-called *Wilson term* in the discretized Hamiltonian [MM94]. This term raises the energy of the undesired modes located at one boarder of the Brillouin zone. Since this mass term is chosen to be proportional to the inverse lattice spacing Δx^{-1}, these states get infinitely heavy in the continuum limit and are thereby prevented from being populated [MM94].
Another way to avoid the *doublers* features the introduction of so-called *staggered fermions* [Sus77]. The basic idea is to introduce a second spatial grid, which is shifted by half a lattice-spacing with respect to the initial grid. Now one of the two spinor components is defined to the regular grid while the second lives on the staggered grid. The consequence of putting only one of the two spinor components on each grid effectively leads to a reduction of the degrees of freedom by a factor of two [MM94]. This will be outlined in more detail in section 3.1.3.
In this work we will pursue a modification of the latter approach, i.e. we not only introduce a second staggered grid in space but also in time. We thereby basically follow the outline given in [HPA14].

3.1.2. Staggered Leapfrog

To cure the previously introduced *fermion doubling problem* we discretize (3.2) as follows:

$$\begin{aligned} \frac{r_j^{n+1} - r_j^n}{\Delta t} &= -im_j^{n+\frac{1}{2}} \frac{r_j^{n+1} + r_j^n}{2} - \frac{s_{j+\frac{1}{2}}^{n+1/2} - s_{j-\frac{1}{2}}^{n+1/2}}{\Delta x} \\ \frac{s_{j+\frac{1}{2}}^{n+\frac{1}{2}} - s_{j+\frac{1}{2}}^{n-\frac{1}{2}}}{\Delta t} &= im_{j+\frac{1}{2}}^n \frac{s_{j+\frac{1}{2}}^{n+\frac{1}{2}} + s_{j+\frac{1}{2}}^{n-\frac{1}{2}}}{2} - \frac{r_{j+1}^n - r_j^n}{\Delta x}, \end{aligned} \tag{3.4}$$

where n and j take integer values and indicate the lattice site on the regular grid. This scheme implies that the two components r and s have to live on different sub-grids. While r is defined on 'regular', i.e. integer grid-points, s occupies half-integer grid-points. We want to stress that this shift affects not only the spatial but also the time coordinate. Further both grids fulfill *periodic boundary conditions* (p.b.c.) in space, while they advance straightly in time. This becomes more clear, when taking a closer look to the respective equation for the r component. When r is evolved from time n to $n+1$, the occurring spatial derivative of s is taken at the time lying in between, i.e at $n+1/2$. Figuratively spoken, this can be seen as the time-evolution 'leaping' over the point in time where the spatial derivative is taken. The scheme is sketched out in Fig. 3.2.

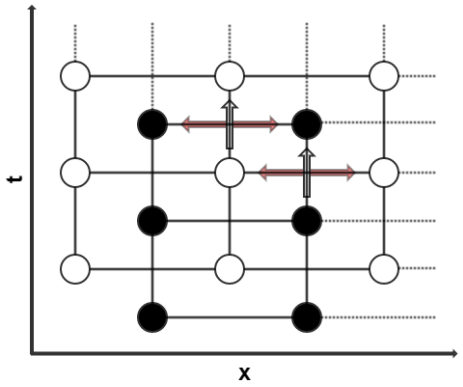


Figure 3.2.: Sketch of the *staggered-leapfrog* scheme in (1+1) dimensions. While the white dots denote points of the integer-valued grid, the blacks dots are shifted by half a step-size in both space and time. The light-red arrow represents the calculation of the spatial derivative, while the white arrow indicates the evolution in time.

As already outlined, the leapfrog scheme demands for a second sub-grid. We therefore introduce a *staggered grid*, which is shifted by $-\frac{1}{2}\Delta t$ in time and $+\frac{1}{2}\Delta x$ in space with respect to the original lattice. In the following we will further consider the case of a constant, i.e. space-time independent mass m.
Returning to our discretization scheme (3.4) we realize that for getting the update routine started we need – apart from the values of r at the time $t = 0$ – the respective values of s at the time $t = -\frac{1}{2}$. Instead of initializing these values, we can use the values of s at the time $t = 0$ and

apply an ordinary *forward Euler* step to evolve s to the time $t = \frac{1}{2}$:

$$\frac{s^{1/2}_{j+\frac{1}{2}} - s^{0}_{j+\frac{1}{2}}}{\Delta t/2} = -im\frac{s^{1/2}_{j+\frac{1}{2}} + s^{0}_{j+\frac{1}{2}}}{2} - \frac{r^{0}_{j+1} - r^{0}_{j}}{\Delta x}$$

This equation allows us to calculate the values of s at time $t = \frac{1}{2}$ without requiring the respective values at $t = -\frac{1}{2}$. After this first step done, the s component lives entirely on the staggered grid. We therefore relabel the indices attached to s in the way that they indicate the lattice sites of the staggered grid: [1]

$$s^{-\frac{1}{2}}_{\frac{1}{2}} \rightarrow s^{0}_{0}.$$

To put it more general:

$$s^{n-\frac{1}{2}}_{j+\frac{1}{2}} \rightarrow s^{n}_{j}.$$

Applying this notation to (3.4), we obtain:

$$\begin{aligned} \frac{r^{n+1}_{j} - r^{n}_{j}}{\Delta t} &= -im\frac{r^{n+1}_{j} + r^{n}_{j}}{2} - \frac{s^{n+1}_{j} - s^{n+1}_{j-1}}{\Delta x} \\ \frac{s^{n+1}_{j} - s^{n}_{j}}{\Delta t} &= im\frac{s^{n+1}_{j} + s^{n}_{j}}{2} - \frac{r^{n}_{j+1} - r^{n}_{j}}{\Delta x} \end{aligned} \tag{3.5}$$

Rearranging the terms leads to:

$$\begin{aligned} r^{n+1}_{j} &= \frac{1}{2 + \Delta t i m}\left[(2 - \Delta t i m) r^{n}_{j} - 2\Delta t \frac{s^{n}_{j} - s^{n}_{j-1}}{\Delta x}\right] \\ s^{n+1}_{j} &= \frac{1}{2 - \Delta t i m}\left[(2 + \Delta t i m) s^{n}_{j} - 2\Delta t \frac{r^{n}_{j+1} - r^{n}_{j}}{\Delta x}\right] \end{aligned} \tag{3.6}$$

Equation (3.6) outlines how the two components r and s can be evolved in time.
The scheme as given by (3.6) provides a second order accuracy[2]in space and time and further guarantees stability for $r \equiv \frac{\Delta t}{\Delta x} \leq 1$. The issue of numerical stability is outlined in more detail in the following section.

3.1.3. Stability and dispersion relation

A crucial property any numerical scheme should embody is of course stability. In the following we will therefore present a brief outline of the *von Neumann stability analysis* of the just introduced *staggered leapfrog* scheme. This will be followed by a short discussion on the induced

[1] This means depending on to which object r or s the indices j and n are attached to they either indicate a lattice side on the regular or the staggered grid.

[2] To be precise we pick up an error of order Δt due to the initializing Euler step.

dispersion relation. We again follow the outline of [HPA14].
Under the assumption of constant coefficients in (3.5) we apply a discrete Fourier transform in space:

$$\begin{aligned}\frac{\tilde{r}^{n+1}-\tilde{r}^{n}}{\Delta t} &= -im\frac{\tilde{r}^{n+1}+\tilde{r}^{n}}{2} - \frac{\tilde{s}^{n+1}\left(1-\exp(ik\Delta x)\right)}{\Delta x}\\ \frac{\tilde{s}^{n+1}-\tilde{s}^{n}}{\Delta t} &= im\frac{\tilde{s}^{n+1}+\tilde{s}^{n}}{2} - \frac{\tilde{r}^{n}\left(\exp(-ik\Delta x)-1\right)}{\Delta x},\end{aligned} \tag{3.7}$$

where $\tilde{r}^n = \tilde{r}^n(k)$ and $\tilde{s}^n = \tilde{s}^n(k)$ are complex, k-dependent numbers. The two coupled equations above can also be written in a matrix notation:

$$\underbrace{\begin{pmatrix}\frac{1}{\Delta t}+\frac{im}{2} & \frac{(1-\exp(ik\Delta x))}{\Delta x}\\ 0 & \frac{1}{\Delta t}-\frac{im}{2}\end{pmatrix}}_{A}\begin{pmatrix}\tilde{r}^{n+1}\\ \tilde{s}^{n+1}\end{pmatrix} = \underbrace{\begin{pmatrix}\frac{1}{\Delta t}-\frac{im}{2} & 0\\ -\frac{(\exp(-ik\Delta x)-1)}{\Delta x} & \frac{1}{\Delta t}+\frac{im}{2}\end{pmatrix}}_{B=A^\dagger}\begin{pmatrix}\tilde{r}^{n}\\ \tilde{s}^{n}\end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix}\tilde{r}^{n+1}\\ \tilde{s}^{n+1}\end{pmatrix} = \underbrace{A^{-1}A^\dagger}_{\equiv L}\begin{pmatrix}\tilde{r}^{n}\\ \tilde{s}^{n}\end{pmatrix} = L\begin{pmatrix}\tilde{r}^{n}\\ \tilde{s}^{n}\end{pmatrix},$$

where the superscript in $A^\dagger$ denotes the Hermitian conjugate of the matrix A. We have further defined the *amplification matrix* L.
Introducing the following abbreviations:

$$a = k\Delta x, \quad b = m\Delta t, \quad r = \frac{\Delta t}{\Delta x}$$

we can rewrite L as:

$$\frac{1}{b+2i}\begin{pmatrix}\frac{8r^2[\cos(a)-1]-(b+2i)^2}{b-2i} & 2ir(\exp(ia)-1)\\ 2ir(1-\exp(-ia)) & -(b-2i)\end{pmatrix}$$

Its eigenvalues are given by:

$$\lambda_\pm = \frac{\mathrm{tr}L}{2} \pm \sqrt{\left(\frac{\mathrm{tr}L}{2}\right)^2 - \mathrm{det}G}$$

while the trace and determinant of L are given by:

$$\begin{aligned}\mathrm{tr}L &= \frac{8\left[1+r^2\cos(a)-r^2\right]-2b^2}{4+b^2}\\ \mathrm{det}L &= 1\end{aligned} \tag{3.8}$$

A further analysis of the eigenvalues and their respective eigenvectors shows that $|\lambda_\pm| = 1$ for $r \leq 1$ and that the scheme grants the *Courant-Friedrichs-Lewy* (CFL)-condition for stability [HPA14]. To obtain the system's dispersion relation, we apply a subsequent discrete Fourier

Transformation with respect to the time to (3.7):

$$Ae^{i\omega\Delta t}\begin{pmatrix}\tilde{\tilde{r}}\\ \tilde{\tilde{s}}\end{pmatrix} = A^*\begin{pmatrix}\tilde{\tilde{r}}\\ \tilde{\tilde{s}}\end{pmatrix}$$
$$\Leftrightarrow e^{i\omega\Delta t}\begin{pmatrix}\tilde{\tilde{r}}\\ \tilde{\tilde{s}}\end{pmatrix} = L\begin{pmatrix}\tilde{\tilde{r}}\\ \tilde{\tilde{s}}\end{pmatrix},$$

where $\tilde{\tilde{r}} = \tilde{\tilde{r}}(\omega, k)$ and $\tilde{\tilde{s}} = \tilde{\tilde{s}}(\omega, k)$. Using our knowledge of the eigenvalues of L we end up with:

$$\omega_\pm = \frac{-i}{\Delta t}\ln(\lambda_\pm) \tag{3.9}$$

As one can see from Fig (3.3) the dispersion relation is satisfied exactly for $r = \frac{\Delta t}{\Delta x} = 1$. For

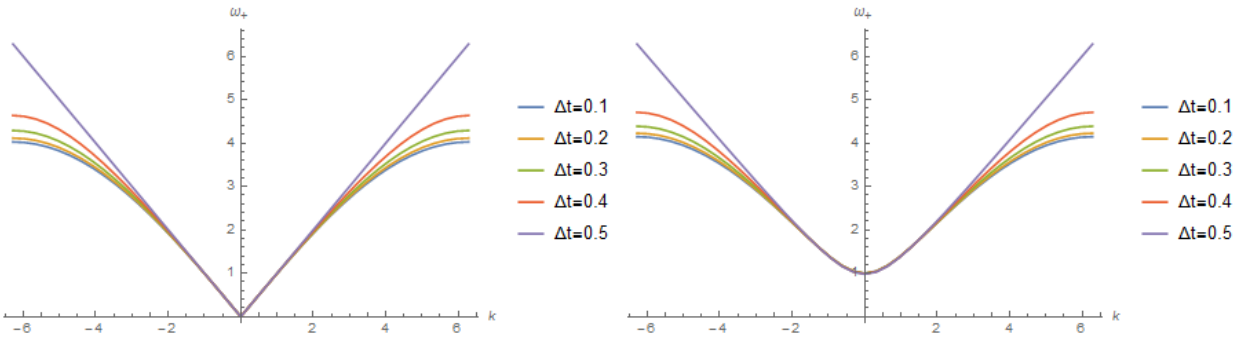


Figure 3.3.: Dispersion relation $\omega_+(k)$ of the staggered leapfrog scheme for different values of Δt, while $\Delta x = 0.5$ is fixed. On the left we have put $m = 0$, on the right $m = 1$. The k-values are in the domain $[-\frac{\pi}{\Delta x}, \frac{\pi}{\Delta x}]$.

$\Delta x > \Delta t$ we get deviations from the exact dispersion relation, which rise with decreasing values of r and growing wave numbers k.
As a check for consistency, we will temporarely consider the massless case. Inserting the explicit expressions of (3.8) into the dispersion relation (3.9) we obtain:

$$\omega_\pm = \frac{-i}{\Delta t}\ln\left[1 + r^2(\cos(a) - 1) \pm \sqrt{[1 + r^2(\cos(a) - 1)]^2 - 1}\right]. \tag{3.10}$$

Choosing $r = 1$ yields

$$\begin{aligned} & \omega_{\pm} = \frac{-i}{\Delta t} \ln\left[\cos(a) \pm \sqrt{\cos^2(a) - 1}\right] \\ \Leftrightarrow \quad & \omega_{\pm} = \frac{-i}{\Delta t} \ln\left[\cos(a) \pm i \sin(a)\right] \\ \Leftrightarrow \quad & \omega_{\pm} = \frac{a}{\Delta t} = \pm k, \end{aligned}$$

i.e. the exact dispersion relation for the continuum.
We further want to stress that even though there are deviations for $r < 1$ from the exact dispersion relation, no additional modes are present within the momentum range $[-\frac{\pi}{\Delta x}, \frac{\pi}{\Delta x}]$. Hence, in contrast to the *midpoint scheme*, the *staggered leapfrog* does not suffer from the appearance of *fermion doublers*.

Why fermion doubling is avoided by the staggered leapfrog scheme

To understand how the *staggered leapfrog* scheme cures the *fermion doubling problem*, it is instructive to follow the outline of [MM94]. Therefore we temporarily consider the massless case, i.e. $m \equiv 0$ and define the following single component fermion field ξ by:

$$\xi_l^k \equiv \begin{cases} r_l^k & \text{if} \quad l = \text{integer} \\ s_l^k & \text{if} \quad l = \text{half-integer} \end{cases}$$

The indices k and l can take integer as well as half-integer values.
Employing the integer-valued indices n and j from above and using (3.4) we can write:

$$\partial_t \xi_j^n = \frac{r_j^{n+1} - r_j^n}{\Delta t} = -\frac{s_{j+\frac{1}{2}}^{n+1/2} - s_{j-\frac{1}{2}}^{n+1/2}}{\Delta x} \overset{\Delta x \to 0}{=} -\partial_x \xi_j^{n+\frac{1}{2}}.$$

This seems to describe right moving waves only. If we now define a second field as:

$$\zeta_l^k = (-1)^{2l} \xi_l^k = e^{\pm 2li\pi} \xi_l^k,$$

and again use the indices n and j, we obtain:

$$\partial_t \zeta_j^n = \frac{r_j^{n+1} - r_j^n}{\Delta t} = -\frac{s_{j+\frac{1}{2}}^{n+1/2} - s_{j-\frac{1}{2}}^{n+1/2}}{\Delta x} \overset{\Delta x \to 0}{=} +\partial_x \zeta_j^{n+\frac{1}{2}},$$

i.e. left moving waves. This means each of two the single component fermion fields describes one Dirac fermion species, i.e. the degrees of freedom have effectively been reduced [MM94].

3.2. 2-point objects

Besides the time-evolution of the 1-point function, we are particularly interested in evolving 2-point objects like the spectral and the statistical function. In the following we will present the basic ingredients that are employed for the numerical integration of the Kadanoff-Baym-Equations. We will start with a short introduction to the *operator splitting* method and continue with the challenge of evolving an object in two time coordinates. In the end we will outline how the *memory integrals* on the r.h.s. of the KBEs are computed.

3.2.1. Operator-Splitting

Due to the complex structure of the KBE including the highly nontrivial *memory integrals*, we make use of a so-called *operator splitting* scheme. The basic idea is to split the full operator dictating the time evolution of the considered system into sub-operators. The latter are then applied consecutively, which allows us to treat each of them individually. This makes us more flexible regarding the used numerical methods and further gives us a better control over the stability of our algorithm in total. We will therefore give a short introduction to this approach and refer the interested reader to [H$^+$10] for a more detailed discussion.
We start with the so-called *Cauchy problem*:

$$\frac{du}{dt} + \mathcal{A}(u) = 0,$$

where $\mathcal{A}$ denotes the full operator.
Under the assumption that $\mathcal{A}$ does not explicitly depend on time, i.e. $\frac{\partial \mathcal{A}}{\partial t} = 0$ its formal solution is given by:

$$u(t) = e^{-t\mathcal{A}} u(t_0)$$

We now assume that we can write $\mathcal{A}$ as a sum of sub-operators $\mathcal{A}_j$:

$$\mathcal{A} = \sum_j \mathcal{A}_j$$

Hence we can - considering discretized time with step size Δt - write the solution as:

$$u(t_{n+1}) = \prod_j e^{-\Delta t \mathcal{A}_j} u(t_n) \tag{3.11}$$

Applied to our case, i.e. the time evolution of the statistical and spectral function given by (2.24) and (2.25) respectively, we define the following two sub-operators:

$$\mathcal{A}_1 = i\gamma^0 M, \qquad \mathcal{A}_2 = \alpha \partial_x + i\gamma^0 m^f$$

where M denotes the respective memory-'operator' (in the sense of: $MF_{\alpha\beta}(x,y) = M_{F_{\alpha\beta}}(x,y)$)). Considering the case of the statistical function $F_{\alpha\beta}(x,y)$ and linearizing the exponential appearing in (3.11) leads to:

$$\begin{aligned} F_{\alpha\beta}(n+1,\mathbf{x},n,\mathbf{y}) &= (1-\Delta t(\alpha\partial_x + im^f\gamma^0))(1-\Delta t i\gamma^0 M)F_{\alpha\beta}(n,\mathbf{x},n,\mathbf{y}) \\ &= (1-\Delta t(\alpha\partial_x + im^f\gamma^0))\underbrace{(F_{\alpha\beta}(n,\mathbf{x},n,\mathbf{y}) - \Delta t i\gamma^0 M_{F_{\alpha\beta}}(n,\mathbf{x},n,\mathbf{y}))}_{\tilde{F}(n+1,\mathbf{x},n,\mathbf{y})} \end{aligned}$$

where $\tilde{F}_{\alpha\beta}(n+1,\mathbf{x},n,\mathbf{y})$ denotes the intermediate result, i.e. what we obtain after the first sub-operator has been applied. The latter is then used as an input for the second sub-operator.
As indicated before due to the complex nature of the *memory integrals* it is quite difficult to estimate the impact of the respective sub-operator on the stability of the total routine. Yet a rule of thumb suggests that the stability of the sub-operator containing the highest number of spatial derivatives can preserve the overall stability of the numerical scheme, even though other involved sub-operators are unstable [P+07]. In our case sub-operator $\mathcal{A}_2$ contains the only spatial derivative and is therefore required to be numerically stable. As discussed in the last section this is guaranteed in case of the *staggered leapfrog* scheme.

3.2.2. Evolving two time coordinates

To quickly remind the reader of the objects we want to evolve in time, we restate the definition of the spectral and the statistical function as given in (2.4):

$$\begin{aligned} \rho_{\alpha\beta}(x,y) &= i\langle\{\Psi_\alpha(x),\bar{\Psi}_\beta(y)\}\rangle \\ F_{\alpha\beta}(x,y) &= \frac{1}{2}\langle[\Psi_\alpha(x),\bar{\Psi}_\beta(y)]\rangle \end{aligned}$$

In (1+1) space-time dimensions the KBE equations can be written as follows:

$$\partial_{x^0}F(x,y) = -\alpha\partial_x F(x,y) - im^f\gamma^0 F(x,y) - i\gamma^0 M_F(x,y) \tag{3.12}$$

$$\partial_{x^0}\rho(x,y) = -\alpha\partial_x \rho(x,y) - im^f\gamma^0 \rho(x,y) - i\gamma^0 M_\rho(x,y) \tag{3.13}$$

where we have omitted the Dirac indices.
The spectral and the statistical function obey the following hermicity properties under the exchange of their arguments [BBS03]:

$$\begin{aligned} (\rho(y,x))^\dagger &= -\gamma^0\rho(x,y)\gamma^0, \\ (F(y,x))^\dagger &= \gamma^0 F(x,y)\gamma^0. \end{aligned} \tag{3.14}$$

The hermitian conjugate of course has to be taken in Dirac space. These relations imply that we only have to calculate the time evolution for times $x^0 \geq y^0$, since the respective values for $x^0 < y^0$ space-time points are always in reach via (3.14).
An aspect that makes dealing with 2-point objects particularly challenging is the presence of

two distinct time coordinates. This means that we need to find a way to evolve our objects in both, x^0- and y^0-direction. While the time evolution along the x^0-direction is directly given by means of the Kadanoff-Baym-Equations, we have to be more careful when addressing the time evolution along the y^0-direction. In the latter case we have to keep in mind that - since the space-time coordinate y is the argument of an adjoint spinor (c.f. (2.4)) - we need to find an 'adjusted' evolution equation in y^0-direction. To derive the latter we take a look at the adjoint of the Kadanoff-Baym-equation (3.12):

$$\begin{aligned}
& \left(\left[i\gamma^0\partial_{x^0} + i\gamma^1\partial_x - m^f\right] F(x,y)\right)^\dagger = M_F(x,y)^\dagger \\
\Leftrightarrow \quad & -i\partial_{x^0}F(x,y)^\dagger\gamma^{0\dagger} - i\partial_x F(x,y)^\dagger\gamma^{1\dagger} - m^f F(x,y)^\dagger = M_F(x,y)^\dagger \\
\Leftrightarrow \quad & \partial_{x^0}F(x,y)^\dagger\gamma^0 - \partial_x F(x,y)^\dagger\gamma^1 - im^f F(x,y)^\dagger = iM_F(x,y)^\dagger \\
\Leftrightarrow \quad & \partial_{x^0}\gamma^0 F(y,x)\gamma^0\gamma^0 - \partial_x\gamma^0 F(y,x)\gamma^0\gamma^1 - im^f\gamma^0 F(y,x)\gamma^0 = iM_F(x,y)^\dagger \\
\Leftrightarrow \quad & \partial_{x^0}F(y,x) - \partial_x F(y,x)\alpha - im^f F(y,x)\gamma^0 = i\gamma^0 M_F(x,y)^\dagger.
\end{aligned}$$

Relabeling the x and y coordinate and treating the spectral function in the same fashion, we eventually get:

$$\partial_{y^0}F(x,y) = \partial_y F(x,y)\alpha + im^f F(x,y)\gamma^0 + i\gamma^0 M_F(y,x)^\dagger \tag{3.15}$$

$$\partial_{y^0}\rho(x,y) = \partial_y \rho(x,y)\alpha + im^f \rho(x,y)\gamma^0 - i\gamma^0 M_\rho(y,x)^\dagger \tag{3.16}$$

Now that we have evolution equations for both time coordinates available, we can pursue finding an effective way of evolving the considered 2-point objects. For this purpose we introduce a new pair of time-coordinates:

$$\tau = \frac{1}{2}(x^0 + y^0), \qquad t_{rel} = (x^0 - y^0)$$

where τ denotes the *absolute time* and t_{rel} the *relative time*.
The time evolution in τ-direction implies - in the picture of the old coordinates x^0 and y^0 - an evolution along the time diagonal. After a new point along the diagonal has been calculated, we can continue the straightforward scheme for the evolution along the x^0-direction. This way we can fill the 'triangle' of space-time points with $x^0 \geq y^0$. Since t_{rel} vanishes for equal times, we omit it in the following lines.
The derivative with respect to τ can be expressed as:

$$\partial_\tau \rho(\tau, \mathbf{x}, \mathbf{y})_{x^0=y^0} = \partial_\tau \rho(x^0, y^0, \mathbf{x}, \mathbf{y})_{x^0=y^0} = \left(\partial_{x^0} + \partial_{y^0}\right)\rho(x^0, y^0, \mathbf{x}, \mathbf{y})_{x^0=y^0}$$

This means that we evolve our system in the absolute time τ as a superposition of the evolution along x^0- and y^0-direction.
For the actual time evolution we use the earlier introduced *operator splitting* scheme. While the first sub-operator $\mathcal{A}_1$ contains the *memory integral*, $\mathcal{A}_2$ involves the Dirac-equation like part. We want to stress that though the latter is now acting on a matrix instead of a spinor we can still make use of the *staggered leapfrog*-scheme introduced in section 3.1.2.

We begin with the sub-operator $\mathcal{A}_1$, which contains the memory operator. Since the *memory integrals* are very complex objects integrating over the system's entire history, they are the main cause for the high CPU-time. They are further the reason why we have to save every space-time point that has been calculated during the time-evolution and are therefore responsible for the high demands for memory. For these reasons we calculate the *memory integrals* at integer space-time points, i.e. for the 'regular' grid only. This allows us to overwrite the staggered grid values at each time step and therefore save a significant amount of memory. Nevertheless, for evolving the objects living on the space-time staggered grid we still need to apply the sub-operator $\mathcal{A}_1$. Therefore we employ an interpolation in space and time to get staggered values of the *memory integrals*:

$$\begin{aligned} M\rho(x^0+\frac{1}{2},\mathbf{x}+\frac{1}{2},y^0,\mathbf{y}) &= M_\rho(x^0+\frac{1}{2},\mathbf{x}+\frac{1}{2},y^0,\mathbf{y})) \\ &= \frac{M_\rho(x^0,\mathbf{x},y^0,\mathbf{y})) + M_\rho(x^0,\mathbf{x}+1,y^0,\mathbf{y}))}{2} \\ &+ \frac{M_\rho(x^0+1,\mathbf{x},y^0,\mathbf{y})) + M_\rho(x^0+1,\mathbf{x}+1,y^0,\mathbf{y}))}{2}. \end{aligned} \tag{3.17}$$

This naive interpolation can of course only be a valid approximation if the value of the *memory integral* does not vary too much within the range of Δt and Δx. For the details of how the *memory integrals* are computed, we refer the reader to section 3.2.3.
Applying the second sub-operator $\mathcal{A}_2$ implies the introduction of a staggered grid. To obtain the respective staggered values, we simply interpolate between two adjacent space points of the original grid:

$$\rho(x^0,\mathbf{x}+\frac{1}{2},y^0,\mathbf{y}) = \frac{\rho(x^0,\mathbf{x},y^0,\mathbf{y}) + \rho(x^0,\mathbf{x}+1,y^0,\mathbf{y})}{2}$$

Now we can employ the earlier introduced *Euler step* to get the offset in time.

$$\rho(x^0,\mathbf{x}+\frac{1}{2},y^0,\mathbf{y}) \xrightarrow{\text{Euler step}} \rho(x^0+\frac{1}{2},\mathbf{x}+\frac{1}{2},y^0,\mathbf{y})$$

With these staggered values at hand, we can essentially follow the update scheme introduced in section 3.1.2. This procedure is employed each time after we have evolved our objects to a new point along the time diagonal to start the evolution in x^0-direction.
In contrast to the example used above to explain the basic idea of the *staggered leapfrog* above, we would like to end up having all quantities evolved to the same space-time. For this purpose we have to introduce staggered as well as non-staggered values for all involved objects. A basic sketch of the whole update procedure can be found in Fig.3.4

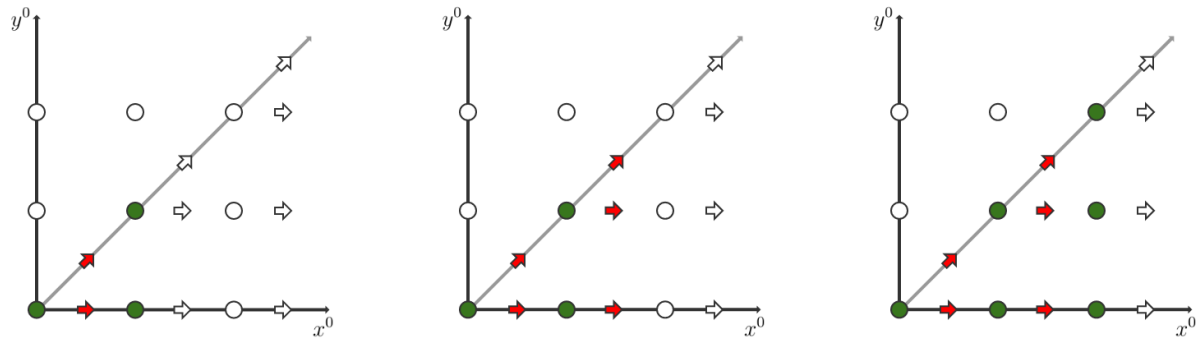


Figure 3.4.: Sketch of the procedure implemented to evolve 2-point objects in time. While the circles represent integer grid values, the arrows depict the objects defined on the staggered grid. In both cases one symbol contains all spatial points $(\mathbf{x}, \mathbf{y})$ at the respective time (x^0, y^0). The color indicates that these space-time values are known or have been calculated. As apparent only segments with $x^0 \geq y^0$ are filled.

3.2.3. Calculation of the memory integrals

As pointed out earlier, the KBE in case of fermions exhibit a matrix structure. To give the reader an impression, we explicitly write down the components of the first integral on the r.h.s. of (2.24) in (1+1) dimensions:

$$\int_{t_0}^{x^0} dz \Sigma^\rho(x,z)F(z,y) = \int_{t_0}^{x^0} dz \begin{pmatrix} \Sigma^\rho_{11}(x,z) & \Sigma^\rho_{12}(x,z) \\ \Sigma^\rho_{21}(x,z) & \Sigma^\rho_{22}(x,z) \end{pmatrix} \begin{pmatrix} F_{11}(z,y) & F_{12}(z,y) \\ F_{21}(z,y) & F_{22}(z,y) \end{pmatrix}$$
$$= \int_{t_0}^{x^0} dz \begin{pmatrix} \Sigma^\rho_{11}(x,z)F_{11}(z,y) + \Sigma^\rho_{12}(x,z)F_{21}(z,y) & \Sigma^\rho_{11}(x,z)F_{12}(z,y) + \Sigma^\rho_{12}(x,z)F_{22}(z,y) \\ \Sigma^\rho_{21}(x,z)F_{11}(z,y) + \Sigma^\rho_{22}(x,z)F_{21}(z,y & \Sigma^\rho_{21}(x,z)F_{12}(z,y) + \Sigma^\rho_{22}(x,z)F_{22}(z,y) \end{pmatrix}$$

Before computing the actual *memory integrals*, a routine is used to calculate the self-energies $\Sigma(x,y)$ appearing in the integrands. With the self-energy values at hand, we can calculate the *memory integrals* at the current time coordinate (x^0, y^0) for all possible pairs of spatial points $(\mathbf{x}, \mathbf{y})$. This also enables us to calculate the respective *staggered memory integrals* via interpolation (c.f. (3.17)).
As outlined before, we calculate and store $\rho(x,y)$ and $F(x,y)$ for space-time points with $x^0 \geq y^0$ only. Depending on the integration domain of the *memory integrals*, values of the spectral and the statistical function with $y^0 > x^0$ occur. To express these objects by the ones we have stored in our memory, we use the hermicity properties ((3.14)) given earlier. To make this more clear, we take a look at an explicit contribution to the *memory integral*, which includes objects with $y^0 > x^0$:

$$\int_{t_0}^{y^0} dz \Sigma_F(x,z)\rho(z,y) \rightarrow \int_{t_0}^{y^0} dz \Sigma_F(x,z)(-\gamma^0\rho(y,z)\gamma^0)^\dagger$$

We now come to the implementation of the actual computation of the *memory integrals*. For this purpose we make use of the so-called *trapezoidal rule*. This is probably the most straightforward numerical approach for computing integrals and can be stated in the following form:

$$\int_a^b f(x)dx = (b-a)\left(\frac{f(a)+f(b)}{2}\right)$$

This means we approximate our function $f(x)$ linearly within the domain of integration. For applying the *trapezoidal rule* to our discretized lattice, we split our integrals into sub-integrals covering the domain between two adjacent grid points. These sub-integrals are then approximated linearly.
To give an example, we consider the spatial integral over the full length of the lattice $l = N\Delta x$

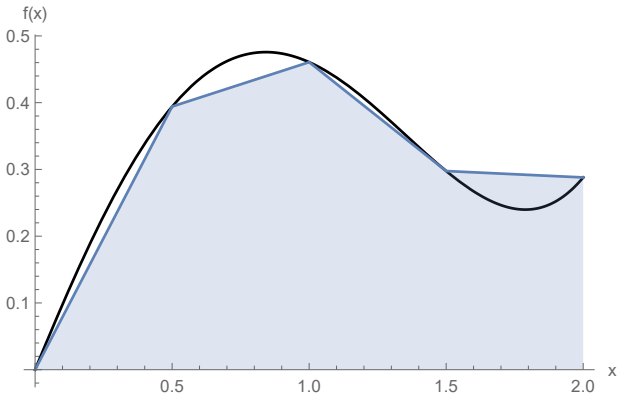


Figure 3.5.: Sketch of the *trapezoidal rule*. The lattice points are separated by units of the lattice spacing $\Delta x = 0.5$. The approximated integral of the function $f(x)$ is shown in light blue.

in (1+1) dimensions. Keeping the p.b.c. in space in mind, we obtain:

$$\begin{aligned}\int_0^{N+1} f(x)dx &= \int_0^{\Delta x} f(x)dx + \int_{\Delta x}^{2\Delta x} f(x)dx + \ldots \int_{N\Delta x}^{(N+1)\Delta x} f(x)dx \\ &= \Delta x\left[\frac{f(0)+f(\Delta x)}{2} + \frac{f(\Delta x)+f(2\Delta x)}{2} + \ldots \frac{f(N\Delta x)+f(0)}{2}\right] \\ &= \Delta x\left[f(0)+f(\Delta x)+\ldots f(N\Delta x)\right]\end{aligned}$$

where we have used that the lattice point (N+1) equals the first lattice point 0 due to the p.b.c.. Since we do not utilize p.b.c. in time, we end up with a slightly different expression:

$$\begin{aligned}\int_0^{N} f(t)dt &= \Delta t\left[\frac{f(0)+f(\Delta t)}{2} + \frac{f(\Delta x)+f(2\Delta x)}{2} + \ldots \frac{f((N-1)\Delta t)+f(N\Delta t)}{2}\right] \\ &= \Delta t\left[\frac{f(0)}{2} + f(\Delta t) + \ldots \frac{f(N\Delta t)}{2}\right]\end{aligned}$$

In contrast to the spatial integration we obtain so-called boundary terms.

4. Results

4.1. Time varying, CP-violating mass

As a first test for our algorithm, we consider the the Dirac equation with a time-varying, CP-violation mass profile. This case has been considered in [Kos15] as a simplified model for the process of electroweak baryogenesis. While the expanding bubble wall (comp. Ch. 1) is modeled by a time varying mass profile, the necessary CP-violation is included due to an imaginary part of the mass. Since the mass-profile does not depend on space, the authors can maintain the assumption of a translation invariant setup. For this given scenario analytic solutions for single mode functions are derived. In the following we will first briefly sketch the derivation of these analytic solutions as given in [Kos15] and then compare them to our numerical results.
It can be easily shown that the ordinary Lagrangian of the free fermion field is no longer hermitian when when a complex mass is used. Therefore the author of [Kos15] used a modified Lagrangian to restore hermiticity. This leads to the following Dirac equation:

$$(i\partial\!\!\!/ - P_{\mathrm{R}} m(x^0) - P_{\mathrm{L}} m^*(x^0))\psi(x) = 0,$$

where $P_{\mathrm{R}} = \frac{1}{2}(\mathbb{1} + \gamma^5)$ and $P_{\mathrm{L}} = \frac{1}{2}(\mathbb{1} - \gamma^5)$ are projection operators. Since in [Kos15] the (3+1) dimensional case was considered, $\psi(x)$ here denotes a four component spinor.
The spinor is then expanded in a helicity basis:

$$\psi(x) = \int \frac{d^3\mathbf{p}}{(2\pi)^3} \sum_{h=\pm} \left[\hat{a}_{\mathbf{p}h} \mu_h(x^0, \mathbf{p}) e^{i\mathbf{px}} + \hat{b}^\dagger_{\mathbf{p}h} \nu_h(x^0, \mathbf{p}) e^{-i\mathbf{px}} \right],$$

where $\mu_h(x^0, \mathbf{p})$ and $\nu_h(x^0, \mathbf{p})$ denote the momentum space spinors while $\hat{a}$ and $\hat{b}$ are the creation and annihilation operators, respectively. Under the assumption of a conserved helicity, the spinors μ and ν can be decomposed in the following way:

$$\mu_h(x^0, \mathbf{p}) = \begin{bmatrix} \eta_h(x^0, \mathbf{p}) \\ \zeta_h(x^0, \mathbf{p}) \end{bmatrix} \otimes \xi_h(\mathbf{p})$$
$$\nu_h(x^0, \mathbf{p}) = \begin{bmatrix} \bar{\eta}_h(x^0, \mathbf{p}) \\ \bar{\zeta}_h(x^0, \mathbf{p}) \end{bmatrix} \otimes \xi_h(\mathbf{p}),$$

where $\xi_h(\mathbf{p})$ denotes the helicity eigen-spinor obeying $\hat{h}\xi_h(\mathbf{p}) = h\xi_h(\mathbf{p})$ with $h = \pm 1$.
Employing a change of basis:

$$\begin{aligned}\phi_{\pm h}(x^0, \mathbf{p}) &= \frac{1}{\sqrt{2}}\left[\eta_h(x^0, \mathbf{p}) \pm \zeta_h(x^0, \mathbf{p})\right], \\ \bar{\phi}_{\pm h}(x^0, \mathbf{p}) &= \frac{1}{\sqrt{2}}\left[\bar{\eta}_h(x^0, \mathbf{p}) \pm \bar{\zeta}_h(x^0, \mathbf{p})\right],\end{aligned} \tag{4.1}$$

one can straightforwardly derive the following evolution equation:

$$i\partial_{x^0}\phi_{\pm h} = \pm m_R\phi_{\pm h} - h\mathbf{p}\phi_{\mp h} \mp im_I\phi_{\mp h}. \tag{4.2}$$

Thereby we have employed the notation $m(t) = m_R(t) + im_I(t)$. The respective equation for $\bar{\phi}_{\pm h}$ has been omitted since it can be obtained by exchanging $h \leftrightarrow -h$ and $\bar{\phi}_{\pm h} \leftrightarrow \phi_{\pm -h}$ in (4.2). We further want to emphasize the two coupled equations for $\phi_{\pm h}$ and $\bar{\phi}_{\pm h}$ are - in case of a vanishing imaginary part m_I - equivalent to the Dirac equation for the two component spinor (3.2) as given in section 3.1 and therefore constitute a good benchmark for us to test the developed update routine for fermionic 1-point functions.
For a mass profile of the form $m(t) = m_1 + m_2\tanh(-\frac{t}{\tau_w})$ the analytic solutions can be written as [Kos15]:

$$\begin{aligned}\phi_+(x^0, \mathbf{p}) &= C_1 \times {}_2F_1(a_+, b_+, c; z) \\ \phi_-(x^0, \mathbf{p}) &= C_2 \times {}_2F_1(a_-, b_-, c; z)\end{aligned} \tag{4.3}$$

where ${}_2F_1$ denotes the hypergeometric function. The parameters $C_{1/2}$, $a_{+/-}$, $b_{+/-}$, c and z depend on the mode $\mathbf{p}$ as well as on the shape of the considered mass profile. The analytic solutions of the mode-functions are then used to construct two points objects, i.e. the so-called *Wightman functions*[1]. Following the outline in [Kos15] we focus on $S^>$:

$$\Delta^{-+}(x^0, y^0)\gamma^0 \equiv iS^>(x^0, y^0)\gamma^0 = \begin{bmatrix}\eta(x^0)\eta^*(y^0) & \eta(x^0)\zeta^*(y^0) \\ \zeta(x^0)\eta^*(y^0) & \zeta(x^0)\zeta^*(y^0).\end{bmatrix} \tag{4.4}$$

Thereby we have chosen a specific helicity ($h = 1$) and therefore dropped the respective index.
After employing absolute (τ) and relative (t_{rel}) time coordinates, (4.4) is subjected to a *Wigner transform*[2]defining the Wigner function $W(\omega, \tau)$:

$$W(\omega, \tau) \equiv iS^>(\omega, \tau)\gamma^0 = i\int_{-\infty}^{\infty} \mathrm{d}t_{rel} e^{i\omega t_{rel} - \Gamma|t_{rel}|} S^>\left(\tau + \frac{t_{rel}}{2}, \tau - \frac{t_{rel}}{2}\right)\gamma^0,$$

where Γ is a constant introduced by the authors to damp non-physical, long-ranged correlations. As a central result the author of [Kos15] observed an interesting behavior in the phase-space

[1]The *Wightman functions* are 2-point correlation functions equivalent to the different parts of the propagator $\Delta^{++}, \Delta^{+-}, \ldots$ as defined in Chapter 2.
[2]The *Wigner transform* reveals information about the phase space structure of the considered system.

structure caused by the changing mass. For early times the Wigner function is determined by the initial condition, which is chosen to consist of the positive frequency ($\omega = \sqrt{\mathbf{p}^2 + m^2}$) solution only. At the late time structure exhibits three distinct peaks. These are interpreted as a particle-antiparticle pair and an additional *coherence* shell located at zero energy. The latter is predicted by the cQPA (*coherent quasiparticle approximation*) and can be interpreted as encoding information about the quantum coherence between particle and antiparticle [Kos15].
To test our algorithm[3]in this context we have chosen a spacial lattice with $N = 40$ points and a lattice spacing $\Delta x = 0.5$. For the momentum $\mathbf{p} = 0.31$ we evolved the mode-functions ϕ_+ and ϕ_- for 400/800 timesteps ($\Delta t = 0.1/0.05$) up to a final time of $t_{final} = 40$. The considered mass profile is plotted in be found in Fig. 4.1, the numerical results for the time evolution of the ϕ_+ mode-function can be found in Fig. 4.2. The respective results for ϕ_- are displayed in the Appendix A. For $\Delta t = 0.05$ we can observe a very good agreement of the numerical results with the analytic solutions. In case of $\Delta t = 0.1$ we can see clear deviations (especially around the mass change) from the analytic solution, yet the qualitative behavior is rendered correctly.

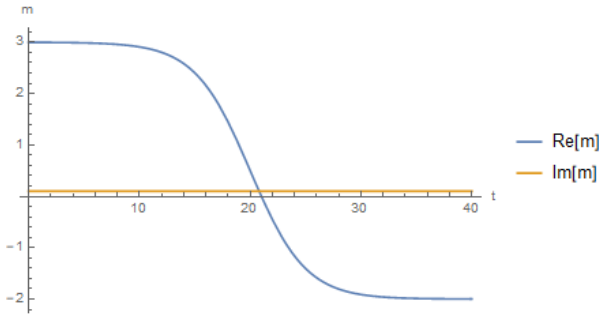


Figure 4.1.: The real (blue) and imaginary (yellow) part of the time dependent mass.

Since our numerical data (in contrast to the analytic solutions in [Kos15]) only covers a finite period in time, we employ a Wigner transform in a box with size L to obtain the desired Wigner function:

$$W(\omega, \tau) = iS^{>}(\omega, \tau)\gamma^0 = i\frac{1}{2L}\int_{-L}^{L} \mathrm{d}t_{rel} e^{i\omega t_{rel}} S^{>}\left(\tau + \frac{t_{rel}}{2}, \tau - \frac{t_{rel}}{2}\right)\gamma^0$$

In Fig. 4.3 we have plotted the absolute value of the 11-component of W. Thereby we have used the data from the $\Delta t = 0.05$ run. One can see that for early times there is one peak around the energy $\omega = \sqrt{\mathbf{p}^2 + m(t=0)^2} \approx 3$. For late times one can observe two peaks around $\omega = \pm\sqrt{\mathbf{p}^2 + m(t=40)^2} \approx 2$ as well as a third peak around $\omega = 0$. This is in agreement with the three shell structure observed in [Kos15].

[3]We want to remind the reader that in the scope of our work only the case of (1+1) space-time dimensions was considered. The core of the algorithm that was developed and employed to solve the Dirac Equation with a complex mass can be found in the Appendix H.

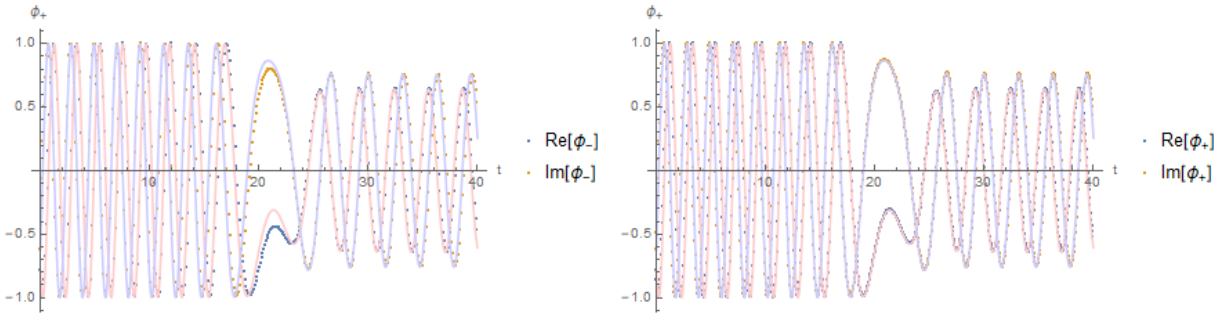


Figure 4.2.: Time evolution of the mode-function ϕ_+ for the momentum $\mathbf{p} = 0.31$ and two different choices of Δt. The analytic solutions are displayed in the light colors. While we clearly see deviations for the $\Delta t = 0.1$ (left) run, the $\Delta t = 0.05$ (right) run agrees very well with the analytic solution.

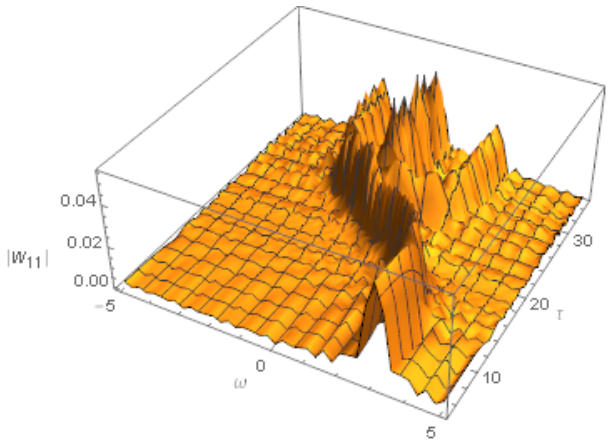


Figure 4.3.: Absolute value of the 11 component of the Wigner function. While we can see one peak around $\omega \approx 3$ in the early times, the three peak structure at late times is apparent.

4.2. (Pre-)Thermalization in the linear sigma model

The main goal of this thesis is to develop an algorithm that is able to solve the Kadanoff-Baym-Equations for fermions entirely in position space. In particular it should be able to render the basic features of the process of (pre-)thermalization. To test our algorithm we will consider homogeneous setups - for which numerical results are available in the literature - as a benchmark. To be more specific we will focus on the the well-known *linear sigma model*, which has been studied extensively in the context of thermalization by several authors (e.g. [BBW04],[BBS03],[LM08])[4].
In the first subsection 4.2.1 we will give a brief introduction to the concept of *prethermalization* as discussed by the authors of [BBW04]. The second subsection 4.2.2 contains details about the considered *linear sigma model*. We will continue with an outline of the employed approximation scheme for the involved self energies and afterwards explain some technical aspects regarding exploited symmetries and the initialization. The third subsection 4.2.3 will embody the presentation of our numerical results.

4.2.1. (Pre-)Thermalization

Probably the most central and fundamental phenomena concerning non-equilibrium physics is the process of thermalization, i.e. the system's internal drive towards thermal equilibrium. Taking a closer look one can further classify the thermalization process into sub-processes with regard to the involved time scales. In other words certain physical quantities reach their thermal equilibrium values on time scales significantly smaller then the time needed for the actual thermalization process. This *prethermalization* was investigated and described by J. Berges et al in [BBW04]. The authors thereby investigated the *linear sigma model* in a spatially homogeneous setup. In Fig. 4.4 we can see a plot of the time evolution of the fermion occupation number for two different (non-equilibrium) initial conditions with the same energy density. All quantities are thereby given in terms of the thermal scalar mass m, which is evaluated in equilibrium.
In the plot above one can clearly identify two distinct time scales:

- the damping time t_{damp}
- the equilibration time t_{eq}

After the significantly shorter damping time $t_{damp} \lessapprox 30\ m^{-1}$ both runs have approached each other to a very high degree and continue evolving in a uniform manner. In other words a major part of information about the initial conditions appears to be lost, even though the system is still far from equilibrium. The actual thermalization process does not take place until $t_{eq} \simeq 95\ m^{-1}$. Another example for a characteristic involved in thermalization given by the authors of [BBW04]is the so-called *Kinetic prethermalization*. As a consequence of dephasing i.e. loss of phase information one can observe an almost conserved equation of state - which is defined as the ratio of pressure over energy density - after a very short time of the order m^{-1}. The *Kinetic*

[4]We want to stress that the just mentioned authors considered the (3+1) dimensional case.

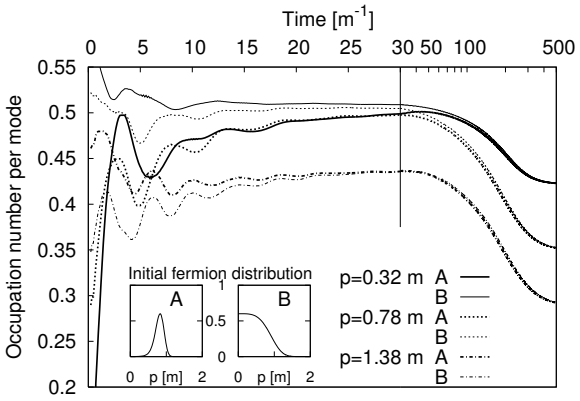


Figure 4.4.: Fermion occupation number $n^{(f)}(t;p)$ for three different momentum modes as a function of time. The evolution is shown for two different initial conditions with *same* energy density. The long-time behavior is shown on a logarithmic scale for $t \geq 30\,m^{-1}$. Image taken from [BBW04]

prethermalization is illustrated in Fig. 4.5, where the equation of state – which is defined as the ratio of pressure over energy density $w = \frac{P}{\epsilon}$ – is plotted as a function of time. The inset in Fig. 4.5 shows the same run for different coupling constants and demonstrates the independence of *Kinetic prethermalization* from the actual scattering driven thermalization process.

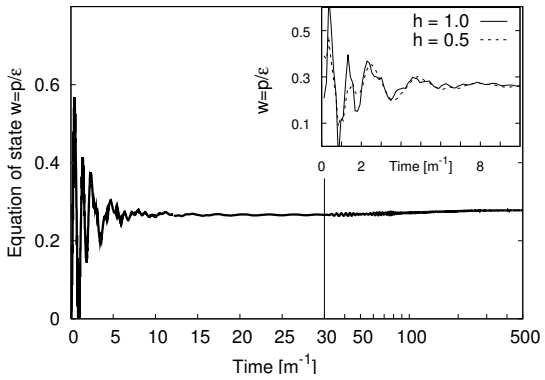


Figure 4.5.: The ratio of pressure over energy density w as a function of time. The inset shows the early stages for two different couplings and demonstrates that the *prethermalization* time is independent of the interaction details. Image taken from [BBW04]

4.2.2. Linear sigma model

The classical action of the *linear sigma model* can be stated in the form:

$$S = \int d^4x \left\{ \bar{\psi} i \partial\!\!\!/ \psi + \frac{1}{2} \left[\partial_\mu \sigma \partial^\mu \sigma + \partial_\mu \pi^a \partial^\mu \pi^a \right] + g \bar{\psi} \left[\sigma + i \gamma_5 \tau^a \pi^a \right] \psi - V(\sigma^2 + \pi^2) \right\}, \quad (4.5)$$

where we have employed the abbreviations $V = \frac{1}{2} m_0^2 \left(\sigma^2 + \pi^2\right) + \frac{\lambda}{4! N_f^2} (\sigma^2 + \pi^2)^2$, and $\pi^2 = \pi^a \pi^a$. The τ^a's are thereby denoting the standard *Pauli matrices* with $a = 1, 2, 3$. The particle content can be interpreted as two fermion flavors ('quarks') coupled via a Yukawa-like interaction g to a scalar field σ ('meson') and a triplet of pseudo scalars π^a ('pions') [BBS03]. Another possible interpretation are two leptons coupled to four scalar fields, where the latter works as a parametrization for the complex Higgs bi-doublet [LM08].
Since there is no mass term for the fermion fields ψ and we further consider the case of vanishing expectation values for the scalar fields, the given action is invariant under $SU(2)_L \times SU(2)_R$ transformations, i.e. exhibits chiral symmetry.

Coupling expansion

The 2-PI-effective action for the model given above can be stated as [LM08]:

$$\Gamma^{\mathrm{2PI}}[G, \Delta] = \frac{i}{2} \mathrm{Tr}_{\mathcal{C}} \ln G^{-1} + \frac{i}{2} \mathrm{Tr}_{\mathcal{C}} G_0^{-1} G - i \mathrm{Tr}_{\mathcal{C}} \ln \Delta^{-1} - \mathrm{Tr}_{\mathcal{C}} \Delta_0^{-1} \Delta + \Gamma_2[G, \Delta] + \mathrm{const}$$

where field-indices and arguments of the propagators have been omitted.
Due to the given chiral symmetry we can - without loss of generality - assume the scalar propagator $G_{ab}(x, y)$ to be diagonal in O(4)-space [BBS03]:

$$G_{ab}(x, y) = G(x, y) \delta_{ab}.$$

In analogy we can assume the fermion propagator $D(x, y)$ to be diagonal in flavor space:

$$\Delta_{ij}(x, y) = \Delta(x, y) \delta_{ij}$$

This can be extended to the respective self-energies in analogous manner.
As outlined before, the derived Kadanoff-Baym-Equations are exact as given in (2.24) and (2.25). Yet for actually solving them numerically one of course needs to apply some sort of approximation. For this purpose we employ a loop expansion, i.e. classify the terms of the effective action with respect to the number of closed loops:

$$\Gamma_2[G, \Delta] = \Gamma_2^{2-\mathrm{loop}}[G, \Delta] + \Gamma_2^{3-\mathrm{loop}}[G, \Delta] + \ldots$$

With the explicit expression for V as given above:

$$V = \frac{1}{2} m_0^2 \left(\sigma^2 + \pi^2\right) + \frac{\lambda}{4! N_f^2} (\sigma^2 + \pi^2)^2, \tag{4.6}$$

we obtain - up to 2 loop order - the following contributions [Ber15, BBS03]:

$$\begin{aligned} \Gamma_2^{2-\text{loop}}[G, \Delta] = &- i g^2 \frac{N_f N_s}{2} \int_{\mathcal{C}} d^4x d^4y \, \text{tr} \left[\Delta(x, y) \Delta(y, x)\right] G(x, y) \\ &- \frac{\lambda}{4! N_f^2} N_s \left(N_s + 2\right) \int_{\mathcal{C}} d^4x G^2(x, x). \end{aligned} \tag{4.7}$$

N_s and N_f denote the number of scalar components and fermion flavors respectively.
We want to stress, that this first non-trivial order already contains scattering as well as off-shell effects [BBS03]. The two contributions on the r.h.s. of (4.7) are depicted diagrammatically in Fig. (4.6).

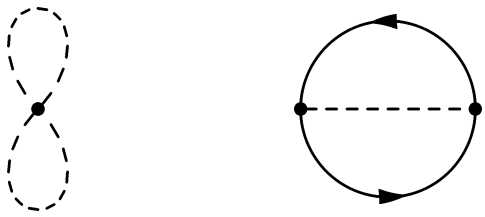


Figure 4.6.: 2-loop contribution to the 2-PI effective action. Solid (dashed) lines represent fermion (scalar) propagators.

To obtain the respective self energies we follow (2.15) and (2.19), i.e. take the functional derivative of the effective action with respect to the scalar and fermion propagator respectively:

$$\text{scalar: } \Pi = \frac{1}{N_s} 2i \frac{\delta \Gamma_2}{\delta G} \tag{4.8}$$

$$\text{fermion: } \Sigma = -\frac{1}{N_f} i \frac{\delta \Gamma_2}{\delta \Delta} \tag{4.9}$$

Starting out with the scalar self energy, we remind the reader that the latter can be decomposed in a local and a non-local part (c.f. (2.21)). The local part can further be absorbed into an effective mass term:

$$m_{eff}^2(x) = m_0^2 + \Pi^{\text{local}}(x; G) \tag{4.10}$$

Applying (4.8) to the local contribution of (4.7) yields:

$$\begin{aligned}\Pi^{\text{local}}(y;G) &= \frac{1}{N_s} 2 \frac{\lambda}{4! N_f^2} N_s \left(N_s + 2\right) 2 \int_{\mathcal{C}} d^4x G(x,x) \underbrace{\frac{\delta G(x,x)}{\delta G(y,y)}}_{\delta(x-y)} \\ &= \lambda \frac{N_s + 2}{6 N_f^2} G(y,y).\end{aligned}$$

Employing the decomposition of the scalar propagator (2.7) and remembering that the scalar spectral function vanishes for equal times, $x^0 = y^0$ leaves us with:

$$\Pi^{\text{local}}(x;G) = \Pi^{\text{local}}(x;F_\phi) = \lambda \frac{N_s + 2}{6 N_f^2} F_\phi(x,x)$$

In analogy we get for the non-local contribution:

$$\begin{aligned}\Pi(w,z) &= \frac{1}{N_s} 2 g^2 \frac{N_f N_s}{2} \int_{\mathcal{C}} d^4x d^4y \, \text{tr}\left[\Delta(x,y)\Delta(y,x)\right] \underbrace{\frac{\delta G(x,y)}{\delta G(w,z)}}_{\delta(x-w,y-z)} \\ &= g^2 N_f \, \text{tr}\left[\Delta(w,z)\Delta(z,w)\right].\end{aligned}$$

Using the decomposition of the full fermion propagator Δ in spectral and statistical function (2.5), we end up with:

$$\begin{aligned}\Pi(x,y) &= g^2 N_f \, \text{tr}\left[\left(F(x,y) - \frac{i}{2}\rho(x,y)\text{sgn}(x^0 - y^0)\right)\left(F(y,x) - \frac{i}{2}\rho(y,x)\text{sgn}(y^0 - x^0)\right)\right] \\ &= g^2 N_f \, \text{tr}\Bigg[F(x,y)F(y,x) + \frac{1}{4}\rho(x,y)\rho(y,x) \underbrace{\text{sgn}^2(x^0 - y^0)}_{\begin{cases}0, & \text{for } x^0 = y^0 \\ 1, & \text{for } x^0 \neq y^0\end{cases}} \\ &\qquad - \frac{i}{2}\left[\rho(x,y)F(y,x) - F(x,y)\rho(y,x)\right]\text{sgn}(x^0 - y^0)\Bigg] \\ &= g^2 N_f \Bigg(\underbrace{\text{tr}\left[F(x,y)F(y,x) + \frac{1}{4}\rho(x,y)\rho(y,x)\text{sgn}^2(x^0 - y^0)\right]}_{\Pi^F} \\ &\qquad - \frac{i}{2}\underbrace{\text{tr}\left[\rho(x,y)F(y,x) - F(x,y)\rho(y,x)\right]}_{\Pi^\rho}\text{sgn}(x^0 - y^0)\Bigg).\end{aligned}$$

In the last line we have indicated the respective decomposition of the self-energy. The two contributions to the scalar self energy are depicted graphically in Fig. (4.7).

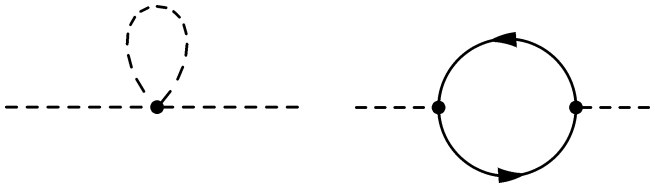


Figure 4.7.: One-loop contribution to the scalar self energy. Solid (dashed) lines represent the fermion (scalar) propagator. Since the left contribution is local, it can be absorbed into an effective mass term.

The fermionic self energy is obtained in the same fashion via (4.9):

$$\begin{aligned}
\Sigma_{\alpha\beta}(w,z) &= -g^2\frac{N_s}{2}\frac{\delta\Gamma_2}{\delta\Delta_{\beta\alpha}(z,w)} \\
&= -g^2\frac{N_s}{2}\int_{\mathcal{C}} d^4x d^4y \frac{\delta(\Delta_{\gamma\delta}(x,y)\Delta_{\delta\gamma}(y,x))}{\delta\Delta_{\beta\alpha}(z,w)} G(x,y) \\
&= -g^2\frac{N_s}{2}\int_{\mathcal{C}} d^4x d^4y \quad \underbrace{\frac{\delta\Delta_{\gamma\delta}(x,y)}{\delta\Delta_{\beta\alpha}(z,w)}}_{\delta_{\gamma\beta}\delta_{\delta\alpha}\delta(x-z)\delta(y-w)} \quad \Delta_{\delta\gamma}(y,x)G(x,y) \\
&\quad + \underbrace{\frac{\delta\Delta_{\delta\gamma}(y,x)}{\delta\Delta_{\beta\alpha}(z,w)}}_{\delta_{\delta\beta}\delta_{\gamma\alpha}\delta(y-z)\delta(x-w)} \quad \Delta_{\gamma\delta}(x,y)G(x,y) \\
&= -g^2\frac{N_s}{2}\Big(\Delta_{\alpha\beta}(w,z)G(z,w) + \Delta_{\alpha\beta}(w,z)\underbrace{G(w,z)}_{=G(z,w)}\Big) \\
&= -g^2 N_s \Delta_{\alpha\beta}(w,z)G(w,z).
\end{aligned}$$

Employing the decomposition of Δ and G, we end up with:

$$\begin{aligned}
\Sigma_{\alpha\beta}(x,y) &= -g^2 N_s\left(F_{\alpha\beta}(x,y) - \frac{i}{2}\rho_{\alpha\beta}(x,y)\mathrm{sgn}(x^0-y^0)\right)\left(F_{\phi}(x,y) - \frac{i}{2}\rho_{\phi}(x,y)\mathrm{sgn}(x^0-y^0)\right) \\
&= -g^2 N_s\Bigg(\underbrace{F_{\alpha\beta}F_{\phi} - \frac{1}{4}\rho_{\alpha\beta}\rho_{\phi}}_{\Sigma^F_{\alpha\beta}} - \frac{i}{2}\underbrace{(F_{\alpha\beta}\rho_{\phi} + \rho_{\alpha\beta}F_{\phi})}_{\Sigma^{\rho}_{\alpha\beta}}\mathrm{sgn}(x^0-y^0)\Bigg).
\end{aligned}$$

In the last line we have omitted the arguments to ease the notation. The fermionic self energy is graphically depicted in Fig. 4.8.
With these 2-loop expressions for the self energies we have all necessary ingredients at hand to calculate the respective *memory integrals* and are therefore ready to solve the KBE employing the numerical implementation as described in Chapter 3.

Figure 4.8.: One-loop contribution to the fermion self energy. Solid (dashed) lines represent the fermion (scalar) propagator.

Lorentz decomposition

It turns out to be useful to decompose the objects $F(x, y)$ and $\rho(x, y)$ into terms of equal transformation properties under Lorentz transformation. We therefore express the spectral function in the following basis [BBS03]:

$$\rho = \rho_S + i\gamma_5\rho_P + \gamma_\mu\rho_V^\mu + \gamma_\mu\gamma_5\rho_A^\mu + \frac{1}{2}\sigma_{\mu\nu}\rho_T^{\mu\nu}$$

This decomposition can of course be applied to the statistical function F in analogous manner. Following the argumentation of [BBS03] one can exploit the symmetries of the model and the initial conditions (i.e. parity, CP-invariance and chiral symmetry), which eventually leads us to:

$$\rho_S = \rho_P = \rho_A^\mu = \rho_T^{\mu\nu} = 0.$$

This means we are left with the vector components ρ_V^μ only. For a more detailed discussion we refer the reader to [BBS03].
In (1+1) dimensions we obtain:

$$\rho = \gamma_\mu\rho_V^\mu = \gamma_0\rho_V^0 - \gamma_1\rho_V^1$$

With our choice of Dirac matrices $\gamma^0 = \sigma_3,\ \gamma^1 = i\sigma_2$ this reduces to:

$$\rho = \begin{pmatrix} \rho_V^0 & -\rho_V^1 \\ \rho_V^1 & -\rho_V^0 \end{pmatrix}.$$

The statistical function F can accordingly be written as:

$$F = \begin{pmatrix} F_V^0 & -F_V^1 \\ F_V^1 & -F_V^0 \end{pmatrix}.$$

In total, this leaves us with four independent quantities. Yet since our algorithm is designed for solving the KBE for more general forms of F and ρ, we evolve all 4 matrix components in time.

Initialization

The 2PI-formalism as introduced in Chapter 2 requires the specification of an initial density matrix. As discussed earlier, we will consider a Gaussian initial density matrix, which is equivalent to the initialization of the lowest two correlation functions.
Since we further consider the case of vanishing field expectation values, i.e. $\langle\phi\rangle = 0$ and $\langle\psi\rangle = 0$ we only have to concern ourselves with the initial specification of the 2-point objects $F_\phi(x,y), \rho_\phi(x,y), F_{\alpha\beta}(x,y)$ and $\rho_{\alpha\beta}(x,y)$.
In case of the fermions the spectral function is completely determined by the anti-commutation relation of the field operators [Ber15]:

$$\{\Psi_\alpha(x), \Psi^\dagger_\beta(y)\}|_{x^0=y^0} = \delta(\mathbf{x}-\mathbf{y})\delta_{\alpha\beta} \tag{4.11}$$

$$\Rightarrow \rho_{\alpha\beta}(x,y)|_{x^0=y^0} = i\gamma^0\delta(\mathbf{x}-\mathbf{y})\delta_{\alpha\beta} \tag{4.12}$$

This leads us to the following initialization for the components of the Lorentz decomposition:

$$\rho^0_V(x,y)|_{x^0=y^0=0} = i\delta(\mathbf{x}-\mathbf{y})^5, \quad \rho^1_V(x,y)|_{x^0=y^0=0} = 0.$$

In case of the scalars, the equal-time commutation relation of two field operators implies [BBS03]:

$$\rho_\phi(x,y)|_{x^0=y^0} = 0, \qquad \partial_{x^0}\rho_\phi(x,y)|_{x^0=y^0} = \delta(\mathbf{x}-\mathbf{y})$$

As outlined in Chapter 2 the statistical function encodes information about the occupation numbers. Accordingly the components of the fermionic statistical function can in momentum space be written in terms of the fermion initial particle number distribution $n^f_0(p)$ [BBS03]:

$$F^0_V(x^0,y^0,p)|_{x^0=y^0=0} = 0, \tag{4.13}$$

$$F^1_V(x^0,y^0,p)|_{x^0=y^0=0} = \frac{1}{2} - n^f_0(p). \tag{4.14}$$

The analogous initialization for the scalar statistical function is of the following form:

$$\begin{aligned} F_\phi(x^0,y^0,p)|_{x^0=y^0=0} &= \frac{1}{\epsilon_0}\left(\frac{1}{2}+n_0(p)\right) \\ \partial_{x^0}F_\phi(x^0,y^0,p)|_{x^0=y^0=0} &= 0 \\ \partial_{x^0}\partial_{y^0}F_\phi(x^0,y^0,p)|_{x^0=y^0=0} &= \epsilon_0\left(\frac{1}{2}+n_0(p)\right) \end{aligned}$$

where $n_0(p)$ denotes the scalar initial particle number distribution and ϵ_0 the initial mode energy[6]. We want to stress that our algorithm is not limited to this kind of initialization. Yet - as outlined before - in the current phase we are mainly interested in testing the developed algorithm, i.e. in

[5] Instead of initializing the Delta-function in position space directly, we have chosen the equivalent way of putting a constant '1' in momentum space. To restore the identity $\int_{-\infty}^{\infty}\delta(\mathbf{x})d\mathbf{x} = 1$, spatial integrals need to be multiplied by factor of $\frac{1}{\Delta x}$.

reproducing results for homogeneous systems available in the literature.
Since the above initialization is in momentum space but our program solves the Kadanoff-Baym Equations in position space, we have to apply a Fourier transform (using the FFTW library [FJ05]) before actually starting the computation. When the desired time evolution has been performed, we can transform the relevant objects back to momentum space or use the position space data directly.

[6]The quasi-particle mode energy is defined via $\epsilon(x^0, p) = \left(\frac{\partial_{x^0} \partial_{y^0} F_\phi(x^0, y^0; p)}{F_\phi(x^0, y^0; p)} \right)^{1/2}_{x^0 = y^0}$. For details see [BBS03].

4.2.3. Numerical results

In the following we will present our numerical results for the *linear sigma model*. This subsection is divided into a part addressing the solution of the 'free' fermion evolution equations, i.e. neglecting the *memory integrals* in the respective KBE. In this case there are analytic solutions at hand, which provide a first benchmark for our algorithm [7]. In the second part we will present our results for solving the full KBEs and thus study the *linear sigma model* in the context of the process of (pre-)thermalization.

Free Equations

As a first test for the developed algorithm we will consider the 'free' evolution equations, i.e. neglect the *memory integrals* in (2.24) and (2.25). Based on this simplification and with the initialization given above, the solution is given by [BBS03]:

$$\begin{aligned} F_V^0(x^0, y^0, p) &= -i\left(\frac{1}{2} - n_0^f(p)\right)\sin[p(x^0 - y^0)] \\ F_V^1(x^0, y^0, p) &= \left(\frac{1}{2} - n_0^f(p)\right)\cos[p(x^0 - y^0)] \\ \rho_V^0(x^0, y^0, p) &= i\cos[p(x^0 - y^0)] \\ \rho_V^1(x^0, y^0, p) &= \sin[p(x^0 - y^0)] \end{aligned} \tag{4.15}$$

These expressions imply that $F(x, y)$ and $\rho(x, y)$ are conserved for equal times $x^0 = y^0$. In case of the statistical function this is reasonable, since for the free evolution we do not expect our occupation numbers to change. The spectral function is fixed by the anti-commutation-relation and therefore is not supposed to change for equal times anyway. For a varying time 'distance' $(x^0 - y^0)$ the components show a momentum-dependent oscillatory behavior.
With an exact expression for the time evolution at hand, this scenario constitutes a good possibility to learn something about the algorithms dependencies on the choice of the lattice parameters like Δx, Δt or the total grid size N.

Dependencies on lattice parameters

As a first test case we consider a system with $N = 10$ and a lattice spacing $\Delta x = 0.5$. The system is evolved to a final time $t_f = 10$, where we have varied the time-lattice spacing Δt. The initial particle distribution was thereby chosen in the form of a Fermi-Dirac-Distribution with vanishing chemical potential, i.e.:

$$n_0^f(p) = n_0^f(E) = \frac{1}{\exp(\frac{E}{T_{\text{init}}}) + 1}. \tag{4.16}$$

[7] The core of the two implemented subroutines corresponding to the two sub-operators $\mathcal{A}_1$ and $\mathcal{A}_2$ as defined in 3.2.1 can be found in the Appendix I.

Thereby T_{init} denotes the initial temperature and was set to 2 for the following runs.
In Fig. (4.9) we have plotted the purely imaginary F_V^0 as well as the real F_V^1 component of the statistical function for different momenta and different values of Δt.
Since $\Delta x = 0.5$ is fixed in these plots, we effectively vary the ratio $r = \frac{\Delta t}{\Delta x}$. This results in a

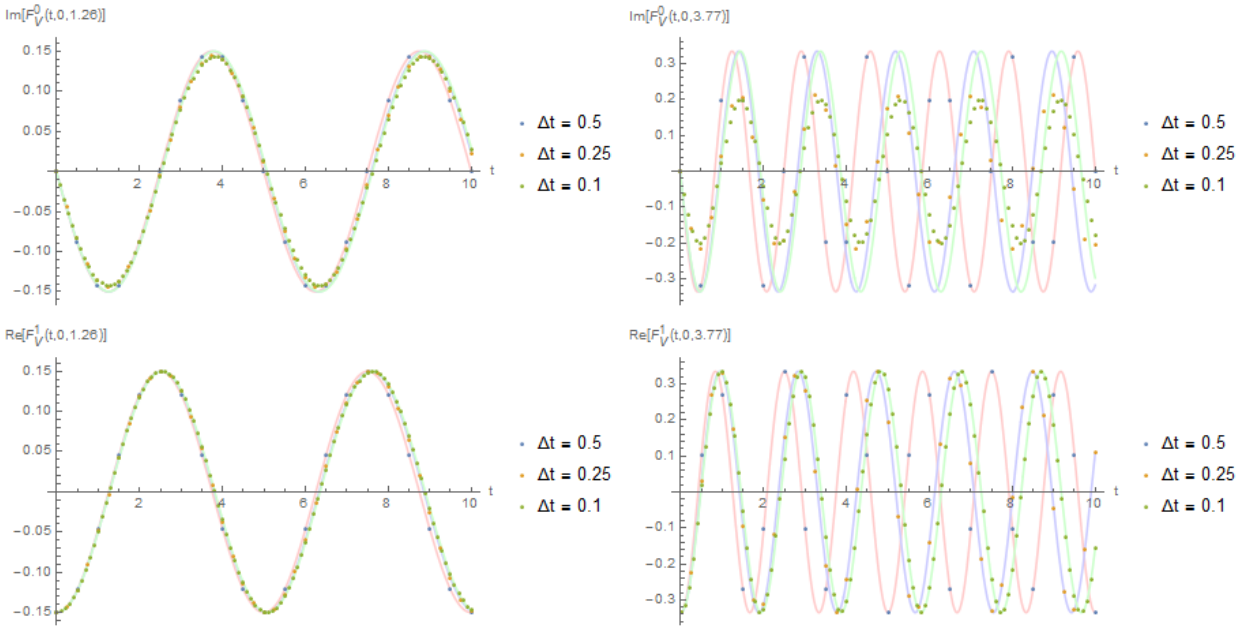


Figure 4.9.: Time evolution of the purely imaginary F_V^0 (top) and the real F_V^1 (bottom) component for two different momenta, a fixed lattice spacing $\Delta x = 0.5$ and varying Δt. The exact solutions are given in light red, blue and green. For $r < 1$ there are deviations in the amplitude for the F_V^0 component. The latter are absent in the case $r = 1$.

change in the dispersion relation (c.f. (3.10)), i.e. the momenta on our lattice are depending on the choice of our lattice parameters. The solid lines in light red, blue and green represent the exact solution given by (4.15), where we have employed the momenta given by (3.10).
While for $r = 1$ we cannot observe any deviations from the exact solutions, in the case of $r < 1$ we get obvious errors in the amplitude for the purely imaginary F_V^0-component. These errors are growing with increasing momenta and for smaller values of r. It is therefore close at hand that these errors are related to the deviations from the exact dispersion relation that was discussed in Chapter 3. In case of the maximal momentum F_V^0 actually drops to zero.
In similar fashion we have plotted the two components of the spectral function in Fig.(4.10). The general behavior is the same, yet in this case the amplitude errors appear in the real-valued ρ_V^1 component.
Analogue plots for $N = 20$ and $\Delta x = 0.5/0.25$ are showing similar patterns and can be found in the Appendix C. We have thereby only plotted F_V^0 and the ρ_V^1 components, i.e. the ones for which the errors in the amplitude have been observed. Increasing the total number of lattice point N did not lead to any observable effects on the results. Choosing a smaller lattice spacing

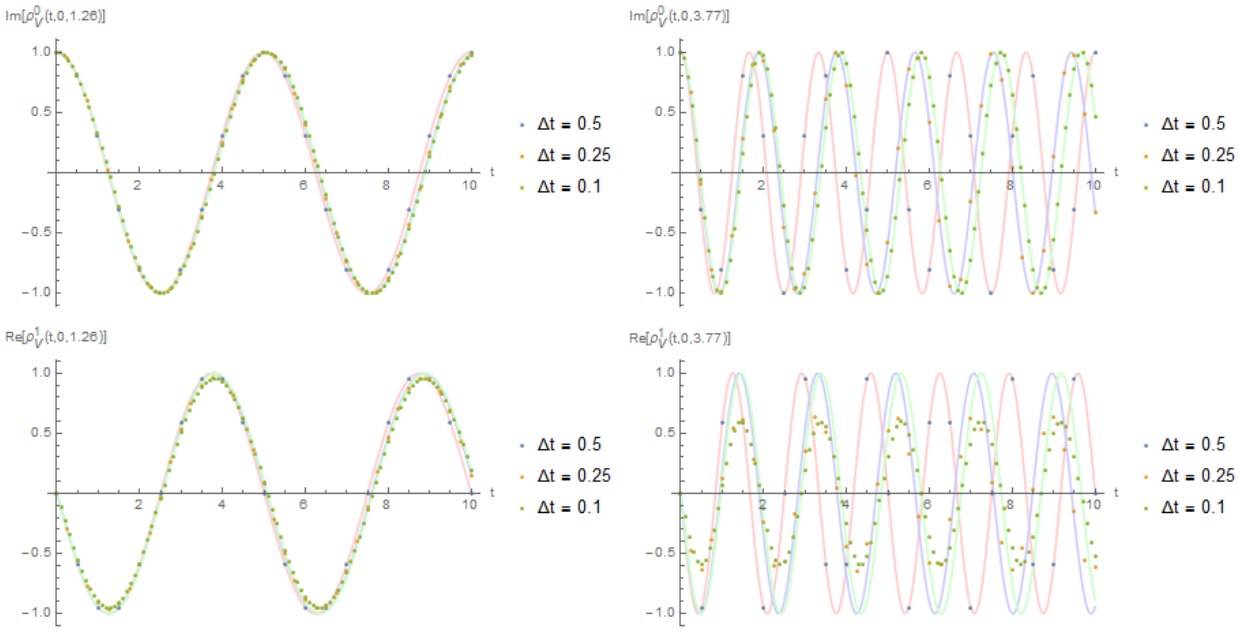


Figure 4.10.: Time evolution of the purely imaginary ρ_V^0 (top) and real ρ_V^1 (bottom) component for two different momenta, a fixed lattice spacing $\Delta x = 0.5$ and varying Δt. The exact solutions are given in light red, blue and green. For $r < 1$, there are visible deviations in the amplitude for the ρ_V^1 component. These are absent for $r = 1$.

$\Delta x = 0.25$ leads to a clear decrease of the amplitude error for $r < 1$. This can be explained by taking a look back at the dispersion relation (3.10) and the plot in Fig. 3.3. We could see that the deviations from the exact dispersion relation start to get large at around $\frac{\pi}{2\Delta x}$, i.e. half the value of the maximum momenta. This means when we consider smaller values of Δx the growth of the deviations sets in later, i.e. for larger values of p.
In summary we can say that our numerical results match the exact solution to a satisfactory degree. The change of the momenta with varying r as predicted by the dispersion relation could be observed. However the error in the amplitude for values $r \neq 1$ should be kept in mind for further investigations.

Prethermalization via memory integrals

In the following we will present our numerical results from solving the 'whole' KBEs, i.e. include the *memory integrals*. This means we explicitly take the interaction between scalar and fermion fields into account, which is encoded in the contributions to the respective self energies. Due to the high demands for memory and CPU the late-time behavior of the system, i.e. the actual way into thermal equilibrium was not in reach with the conventional computers used mainly[8]during this work. Therefore our focus will be on the previously introduced process of *prethermalization*, which occurs on a significantly shorter time scale. We are thereby especially interested in the earlier discussed system's loss of memory regarding the details of its initial condition. Concerning the potential V we will consider three distinct scenarios. Starting with a purely quadratic potential ($\lambda = 0$) as considered in [BBS03], we will continue with a λ of the order of the Yukawa coupling g and in the last case consider a significantly larger value for λ as discussed in [LM08].

Purely quadratic potential

We start with considering a purely quadratic potential V, i.e. set the quartic coupling λ to zero. As outlined by [BBS03] this is sufficient to study the process of thermalization within this model. While the Yukawa coupling g was set to 1, the scalar bare mass m_0 was set to $\sqrt{2}$. The latter was further used to set the scale for all other involved physical quantities, e.g. $t \equiv t[m_0^{-1}],\ p \equiv p[m_0]$. For numerically solving the KBE we have chosen a spatial lattice with $N = 20$ and a lattice constant $\Delta x \equiv \Delta x[m_0^{-1}] = \frac{m_0}{2}$. The stepsize in time was set to $\Delta t \equiv \Delta t[m_0^{-1}] = \frac{m_0}{20}$. Using the conventional computers at our institute we evolved the system up to 300 timesteps, which made us confident to cover the relevant time scale of the *prethermalization* process. To probe the system in this context we have done runs for two different initial conditions, A and B. To allow for a more direct comparison A and B have been chosen in a qualitatively similar fashion as in [BBS03] (see Fig 4.11).
On the l.h.s. of Fig. 4.12 we have plotted the equal-time evolution of $F_V^1(t, t, p)$ for three different

[8]In the final phase of this work we could employ the institute's cluster computer, which has been put into operation recently. Therefore some of the numerical results given in the following have been produced using this cluster. If this is the case, it will be explicitly pointed out in the text.

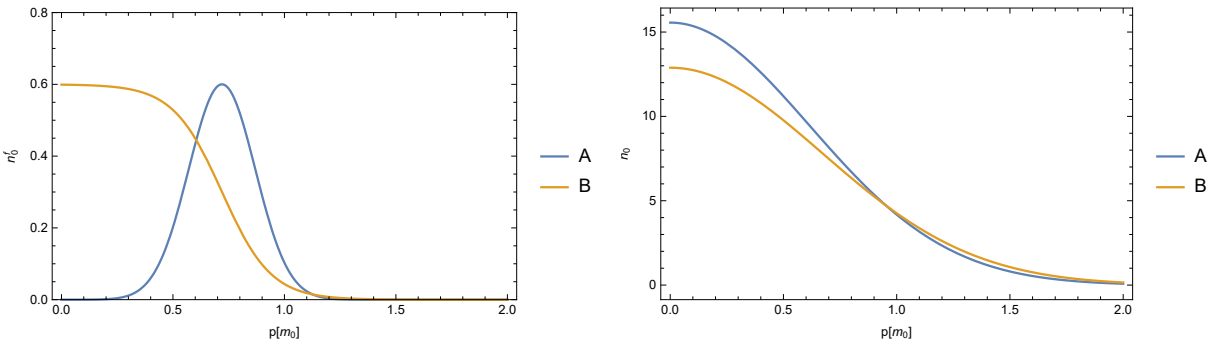


Figure 4.11.: Fermion (left) and scalar (right) particle number distributions for the two initial conditions A and B containing the same average energy density.

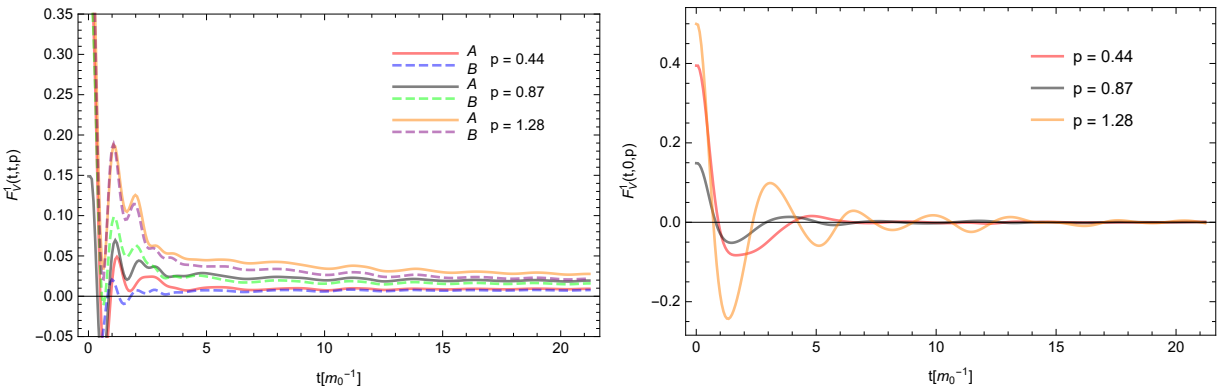


Figure 4.12.: On the left side we have plotted the equal time evolution of $F_V^1(t,t,p)$ for three different momenta and two different initial conditions. One can clearly observe an approach of the two runs. On the right side we see the correlation to the initial time $F_V^1(t,0,p)$, which shows a damped behavior.

momenta p and the two initial conditions. We want to remind the reader that $F_V^1(t,t,p)$ is directly related to the particle number distribution $n^f(p)$ via (4.14). For early times we can observe a rapid decrease of $F_V^1(t,t,p)$ for all three momenta. Since this implies an increase of $n^f(p)$ this can be interpreted as a production of particles. At a time $t \approx 5$ the system continues to evolve in a more smooth and stationary behavior. We further see that the two runs A and B clearly approach each other and continue to evolve in a uniform manner. This clearly indicates the system's loss of memory about the details of the initial conditions, i.e. *universality*. In this context our results are in good agreement with the respective plots given in [BBS03].
The system's loss of memory can also be related to the correlation to the initial time $F_V^1(t,0,p)$, which is plotted on the r.h.s. of Fig. 4.12 for the initial condition A. We can observe a strongly damped behavior, which is consistent with the fast approach of the $F_V^1(t,t,p)$ values for the two runs. The respective plots for the ρ_V^0 component of the spectral function can be found in the Appendix D. While the equal-time evolution is conserved exactly as demanded by the anti-commutation relation 4.11, the correlation to the initial time is damped in a similar manner as the F_V^1 component.
In Fig. 4.13 we can see analogous plots for the scalar statistical function F_ϕ. The equal-time

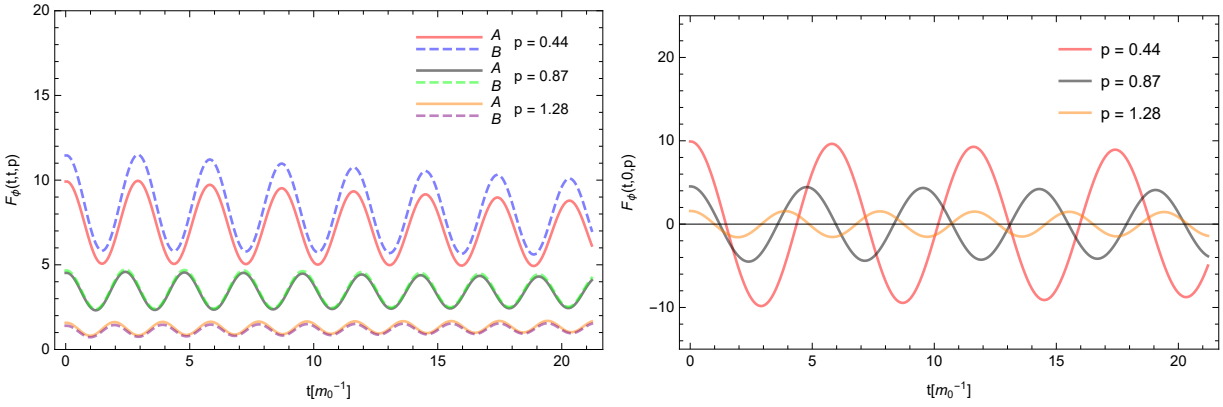


Figure 4.13.: On the left side the equal time evolution of $F_\phi(t,t,p)$ is plotted for three different momenta and the initial conditions A (solid) and B (dashed). We can observe a weakly damped, oscillatory drift motion. The correlation to the initial time $F_\phi(t,0,p)$ on the right side is clearly damped weaker compared to the fermionic case.

evolution on the l.h.s. shows an oscillatory, weakly damped drift motion. The approach of the two initial conditions A and B is clearly slower than in the fermionic case. This is consistent with the less strong damping of the correlation to the initial condition (comp. r.h.s. of Fig 4.13). The overall behavior is once more in good agreement with the results in [BBS03]. The respective plots for the scalar spectral function can again be found in the Appendix D.
To investigate the impact of the Yukawa coupling g in more detail we have repeated run A with a smaller Yukawa coupling $g = 0.8$. In case of the equal-time evolution of $F_V^1(t,t,p)$ (l.h.s. of Fig. 4.14) we see a clear discrepancy for the two choices of the coupling g. In case of the weaker coupling the values of $F_V^1(t,t,p)$ settle on significantly larger values, i.e. on smaller occupation numbers. This can be explained by the fact that the Yukawa coupling determines

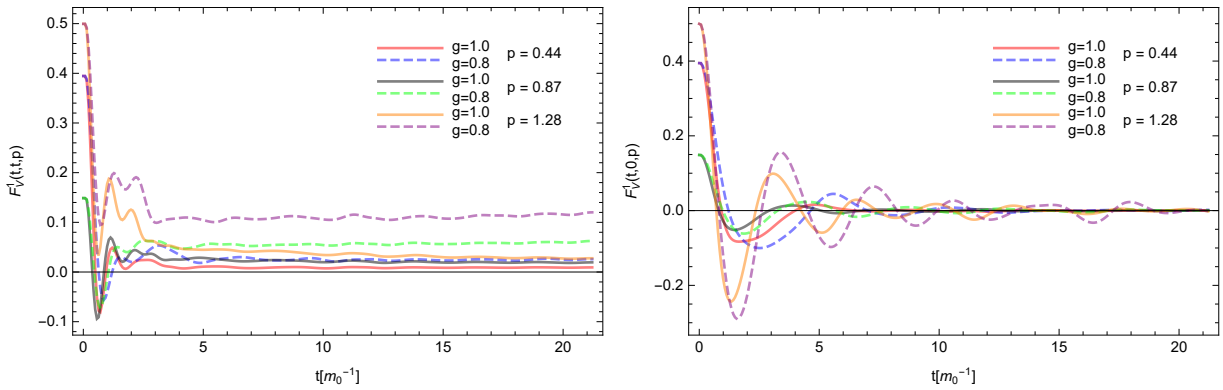


Figure 4.14.: Evolution of the F_V^1 for equal (left) and unequal (right) times for two different values of the Yukawa coupling, $g = 1$ (solid) and $g = 0$ (dashed). As expected decreasing g reduces the damping of the correlation to the initial time.

the interaction strength and therefore the energy exchange between scalar and fermionic fields. For the correlation to the initial time $F_V^1(t, 0, p)$ the decrease of the coupling g leads to a weaker damping (see r.h.s of Fig. 4.14), which appears reasonable.

To probe the system in context of an approach to thermal equilibrium, we have extracted the fermion particle distribution number $n^f(p)$ from $F_V^1(t, t, p)$ via the relation (4.14) and plotted the function $\ln(1/n^f(p) - 1)$ (see l.h.s. of Fig 4.15).[9]The latter should – in case of a Fermi-Dirac distribution – reduce to a straight line with the inverse temperature given by the slope. In our case we observe the emerge of a structure symmetric around $p \approx 1.5$, i.e. close to half the value of the maximal momentum. This is obviously in contradiction to the expectations of a tendency towards a straight line and is further not in accordance with the observations in [BBS03]. In particular the trend towards a symmetric shape suggests that we still encounter some kind of *fermion doubling*. Since in case of the suboperator $\mathcal{A}_2$, which contains the spatial derivative we explicitly avoid *fermion doublers* by employing the *staggered leapfrog* scheme, we presume that these *doublers* are related to the *memory integrals*.

In analogous manner we have plotted $n(\epsilon)$ and $\ln(1/n(\epsilon) + 1)$[10]to investigate the scalars in the context of an approach to thermal equilibrium (see r.h.s. of Fig 4.15). The overall behavior looks rather chaotic, including a non-monotonic behavior of the mode energy ϵ. The latter might be a consequence of an insufficiently high resolution of the dynamics by the discretization employed to calculate ϵ. We further observe negative occupation numbers for early times ($t = 3, 7$), which lead to negative arguments for the logarithm they are therefore excluded from the plots. Because there are no negative values for $n(\epsilon)$ at late times, in stands to reason that these are a consequence of the strong early time dynamics. We can further observe a tendency towards a more smooth and linear shape for late times ($t = 42$). Though this might indicate an approach to thermal

[9]For this run we have employed the cluster computer at our institute. This enabled us to double the number of time steps and reach a final time $t_{final}[m_0^{-1}] = 42$.

[10]To calculate quasi-particle mode energy $\epsilon(x^0, p)$ we have employed the following discretization: $\partial_{x^0}\partial_{y^0} F_\phi(x^0, y^0; p) = \frac{F_\phi(x^0+\Delta t, y^0+\Delta t; p) - F_\phi(x^0+\Delta t, y^0-\Delta t; p) - F_\phi(x^0-\Delta t, y^0+\Delta t; p) + F_\phi(x^0-\Delta t, y^0-\Delta t; p)}{4\Delta t^2}$.

equilibrium, we have to record that the overall behavior clearly differs from the one observed in [BBS03]. To give the reader an additional perspective we have also created non-logarithmic plots for the time evolution of the occupation numbers $n^f(p)$ and $n(\epsilon)$. These can be found in Appendix E.

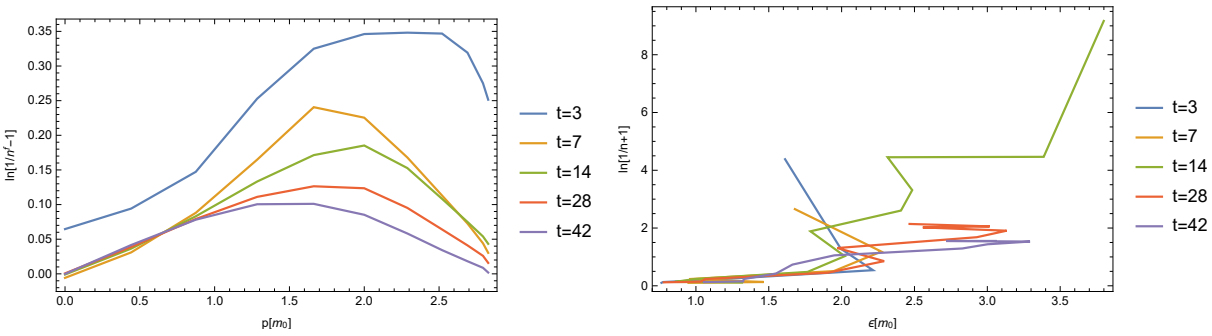


Figure 4.15.: Plot of the fermionic (l.h.s.) and scalar (r.h.s.) particle number distribution for different values of the time t. We have plotted the functions $\ln(1/n^f(p) - 1)$ and $\ln(1/n(\epsilon) + 1)$ which both should – in case of Fermi-Dirac/Bose-Einstein distribution – reduce to a straight line. For the fermions we can observe the emerge ob a symmetric shape, which indicates that our algorithm still suffers from *fermion doublers*.

As indicated before, the driving force behind the system's dynamics are the *memory integrals*, i.e. they play the central role for the (pre-)thermalization process. To give the reader an impression of the nature of these objects, we have plotted some snapshots in time of the *memory integrals* for the run with initial condition A and the Yukawa coupling $g = 1$. To be more specific, we have generated contour plots of the imaginary part of the $[M_F(t, t', r)]_{12}$ component. These can be found in the Appendix F. Each of this plot is for a fixed time t while the spatial distance r and the value of the second time coordinate t' are varied. One can clearly see how the contributions to the *memory integral* are located around $r = 0$ at early times and spread to larger distances with ongoing time. Besides an overall damping with time, one can observe a damping with growing distance $t - t'$ for later times.

To conclude the investigation for the purely quadratic potential, we have probed the algorithm in regard to it's dependence on the choice of the lattice constants Δt and Δx. As a desirable property of any numerical algorithm, we would like to see a convergence when choosing smaller values for the lattice spacing. Therefore we have repeated the upper run (initial condition A, $g = 1$) with reduced lattice spacing ($\Delta t = \frac{m_0}{40}$) while keeping the remaining run parameters constant. Though there are apparent deviations from the run with $\Delta t = \frac{m_0}{20}$, the qualitative behavior is very similar. Decreasing the lattice spacing once more ($\Delta t = \frac{m_0}{80}$) shows only a small effect, i.e. the algorithms converges for $\Delta t \to 0$. In the same fashion we have reduced the spatial lattice spacing from $\frac{m_0}{2}$ to $\frac{m_0}{4}$. In contrast to reducing the lattice spacing in time, this has a very strong impact.[11] Reducing the lattice spacing once more to $\Delta x = \frac{m_0}{8}$ has a smaller, yet still

[11] We want to emphasize that in our case Δx is one order of magnitude larger than Δt, which makes a direct comparison difficult.

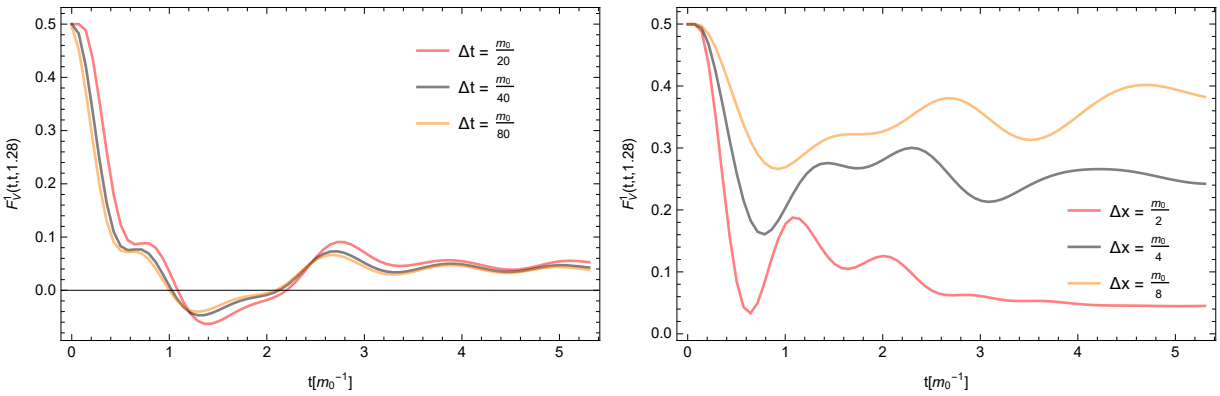


Figure 4.16.: Dependency of F_V^1 on the choice of the lattice spacings Δt and Δx. While we can see a clearly converging behavior for Δt, reducing Δx still has a large impact. We want to emphasize that Δt and Δx are not of the same order of magnitude.

significant effect. On the one hand this indicates a convergence for $\Delta x \to 0$ but on the other hand implies that for reaching the latter, the algorithm demands for significantly smaller values of Δx.
In summary we can on the one hand say that - regarding the early time behavior and the concept of *universality* - our results are in good agreement with the momentum space approach by Berges et al in [BBS03]. On the other hand we see a clearly deviating behavior in case of the time evolution of the particle number distributions $n^f(p)$ and $n(\epsilon)$ compared to the results in [BBS03]. We could further observe that for our choice of Δx the algorithm is still far from convergence.

Quartic scalar self-interaction $\lambda = g = 1$

In this subsection we present our numerical results for $\lambda = g = 1$. This choice has been employed for the results presented in [BBW04]. As discussed in 4.2.2 this term leads to an additional local contribution to the scalar self-energy, i.e. a shift to the scalar effective mass. For the first run we have stuck to the former run parameters, i.e. $N = 20$, $\Delta x = \frac{m_0}{2}$, $\Delta t = \frac{m_0}{20}$ and $g = 1$. Once more we did runs for the two different initial conditions A and B as introduced above. In Fig. 4.17 we have displayed the equal time evolution of $F_V^1(t,t,p)$ and $F_\phi(t,t,p)$. For the fermions we can see an almost identical behavior compared to the runs without the quartic scalar self interaction. The rather small impact on the dynamics of the fermions can be explained by the strongly suppressing $1/4!N^f$ term in (4.6) and the fact that the additional contribution leads to a shift in the scalar effective mass only. In comparison to the results in [BBW04] our values of $F_V^1(t,t,p)$ for $p < 1$ are in good agreement, while $p \sim 1.3$ they settle on significantly smaller values. Also in case of the scalars we observe a very similar behavior compared to the $\lambda = 0$ run. A discrepancy can yet be observed in the envelope of the oscillating equal time evolution $F_\phi(t,t,p)$. Instead of a monotone decrease, one can observe an oscillation in the envelope itself. It stands to reason that this is a consequence of the additional contribution to the scalar effective mass.
Due to the very similar behavior compared to the purely quadratic potential we have omitted

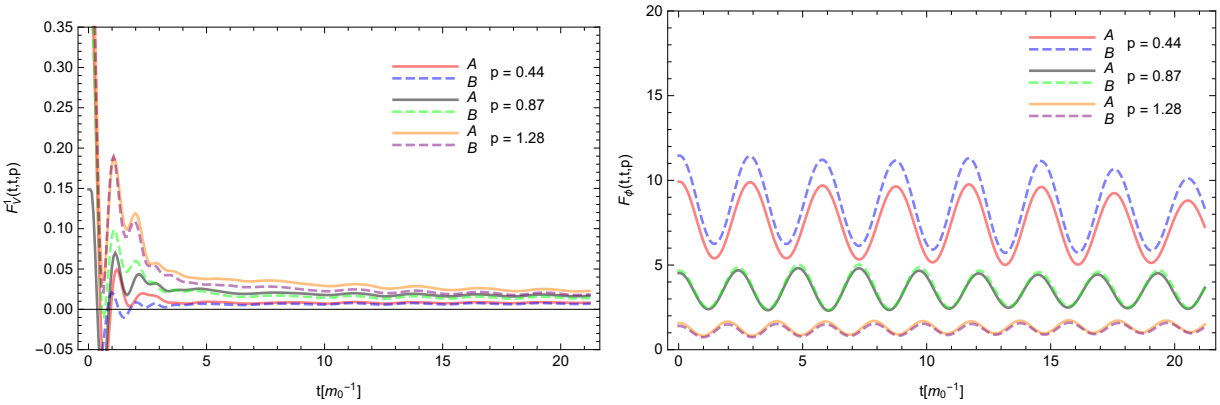


Figure 4.17.: Time evolution of the F_V^1 (l.h.s.) component of the fermion statistical function and the scalar statistical function F_ϕ (r.h.s.). While the impact of the additional quartic scalar self-interaction is clearly visible for the scalars, it is hardly observable in case of the fermions.

further analysis for this scenario.

Quartic scalar self-interaction $\lambda = 24$

As a final setup we have considered the case $\lambda = 24$, i.e. a significantly larger scalar quartic self-coupling. This choice corresponds to the one employed by the authors of [LM08], which for solving the KBE explicitly follow the approach given in [BBS03].[12] The authors of [LM08] observe an approach to thermal equilibrium on significantly shorter time scales. On the one hand this might be a consequence of the larger value for the quartic scalar self-interaction, but on the other hand could also be related to the different initial conditions. In contrast to [BBS03] the scalars modes are not populated strikingly higher than the fermions. To allow for a proper comparison we have modified our initial conditions A and B (see Fig. 4.18) to qualitatively match the ones given in [LM08]. For the following results we have used the institutes cluster computer and – apart from the choice of λ – stuck to the former run parameters.
In Fig. 4.19 we have plotted $F_V^1(t,t,p)$ (l.h.s.) and $F_\phi(t,t,p)$ (r.h.s.). To allow for a more direct comparison with the results in [LM08] we have also employed a logarithmic time axes. In case of the fermions we can see strong movement in the early times that tends to a more smooth and drifting motion for late times. For the scalars we can observe a damped oscillation that also evolves towards a drift-like motion for late times. In this sense our results are in good agreement with those displayed in [LM08]. However neither the transition to an almost completely constant regime nor an approach of the runs two runs A and B to a degree of indistinguishability as observed in [LM08] can be observed within the considered time scale. Both might yet simply be a consequence of not having reached sufficiently late times. A discrepancy in the timescales could be related to the fact that we have considered (1+1) instead of (3+1) dimensions as in

[12]The authors of [LM08] did not consider the factor $4!N_f^2$ appearing in the denominator of (4.6) and therefore effectively considered a significantly larger self coupling λ.

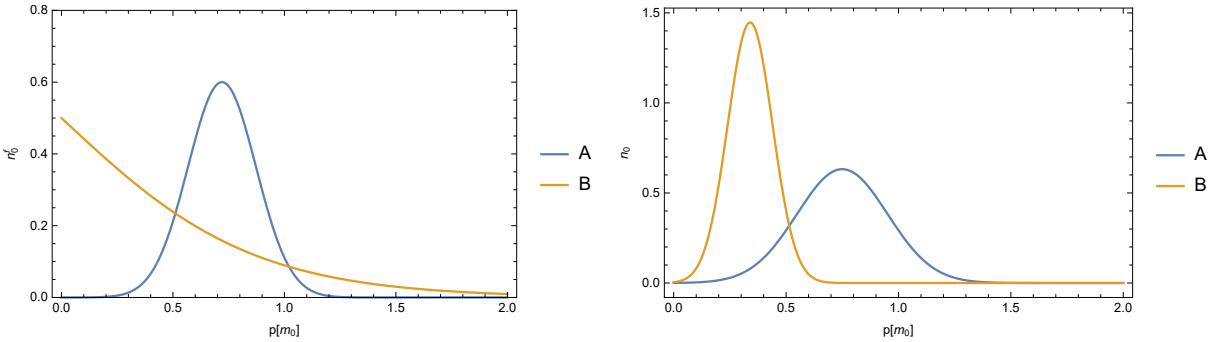


Figure 4.18.: Fermion (l.h.s.) and scalar (r.h.s.) particle number distributions for the two initial conditions A and B. Both contain the same average energy density.

[LM08].
The plots for the correlation to the initial time $F_V^1(t,0,p)$ and $F_\phi(t,0,p)$ can be found in the Appendix G. For both fermions and scalars we see the expected damped behavior. In contrast to scenarios with $\lambda = 0$ and $\lambda = 1$ the damping for fermions and scalars appears to be of a similar magnitude. This might again either be a consequence of the different initial conditions or the effect of the significantly larger self-coupling λ.

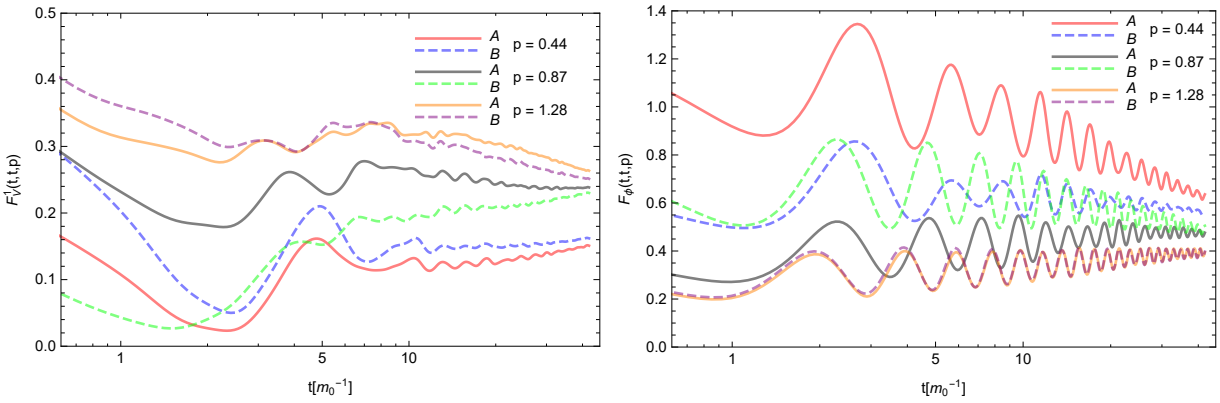


Figure 4.19.: Equal time evolution of $F_V^1(t,t,p)$ (l.h.s.) and $F_\phi(t,t,p)$ (r.h.s.). In both cases we can observe an approach of the two initial conditions A (solid) and B (dashed). The time scale of this approach appears to be similar for fermions and scalars.

Once more we have probed the system in regard to an approach to thermal equilibrium (see Fig. 4.20). In case of the fermions (l.h.s.) we can again observe an evolution towards a symmetric shape, which increases the suspicion that our algorithm still suffers from the existence of *fermion doublers*. For the scalars (r.h.s.) we see – as in the case $\lambda = 0$ – a chaotic early time behavior that becomes more smooth and linear for late times ($t = 28, t = 42$). This might indicate an approach to thermal equilibrium, yet the system is still clearly far from a Bose-Einstein distribution. In this context a direct comparison with the results in [LM08] is not possible, since the authors show plots for late times only. Without the late time behavior in reach, we are therefore not able

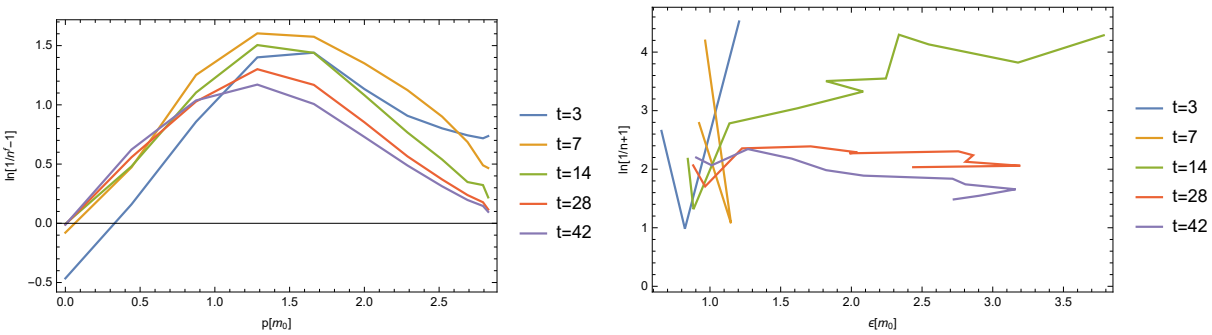


Figure 4.20.: Plot of the functions $\ln(1/n^f(p) - 1)$ (l.h.s.) and $\ln(1/n(\epsilon) + 1)$ (r.h.s.). While in case of the fermions we once more see the emerge of a symmetric structure, the scalars tend towards a more smooth and linear shape for late times ($t = 28, t = 42$), but are yet clearly still far from thermal equilibrium. The missing data points in case of the scalars ($t = 3, 7$) are due to the occurrence of negative occupation numbers at early times.

to judge this conclusively. The non-logarithmic plots for the time evolution of the occupation numbers $n^f(p)$ and $n(\epsilon)$ can be found in the Appendix G.

5. Conclusions and Outlook

Within this work we have developed an algorithm to numerically solve the Kadanoff-Baym Equations for fermions entirely in position space. For reasons of simplicity we have thereby restricted ourselves to the case of (1+1) space-time dimensions. To our knowledge all approaches available in the literature solved the KBE in momentum space, i.e. under the assumption of spatial homogeneity. Our more general framework does not suffer from this restriction but allows for a description of quantum systems out of equilibrium which explicitly violate translation invariance.
To avoid the so-called *fermion doubling problem*, we have employed the *staggered leapfrog* scheme as a discretization. We have further made us of an *operator splitting* scheme to get better control over potential numerical instabilities.
As a test for the developed algorithm we have solved the KBE for the linear sigma model within a homogeneous and highly symmetric setup. The latter has been investigated in (3+1) dimensions by several authors employing a momentum space approach [BBS03] [BBW04] [LM08]. Trying to reproduce these results with our position space approach in (1+1) dimensions, we could observe a good agreement for the early time behavior including the fundamental feature of *universality*. Considering the time evolution of the particle number distribution we yet found clear deviations from the literature. In particular for the fermions we could observe the emerge of an symmetric shape, which indicates that our algorithm still suffers from the existence of *fermion doublers*.
Though we were able to use the cluster computer at the final phase of this work, the late-time behavior and therefore an approach to thermal equilibrium was not in reach within the scope of this work. However, we are confident that extending the usage the cluster will enable us not only to cover the late time behavior but will also allow for the consideration of larger systems. For this purpose we currently work on a way to economize the algorithms demands for memory, i.e. try to find a more flexible and efficient way to deal with the involved *memory integrals*. A promising approach in this context has been suggested by Berges et al. in [Ber15]. They argue that the influence from the early time dynamics on the late time behavior is typically damped. Therefore certain domains of the *memory integrals* can be neglected in further calculations. This implies that some part of the memory can be released and used otherwise.
In the near future we intend to implement a routine for the calculation of the system's total energy, which will allow us to probe our system in the context of energy conservation. It is further planned to implement a routine for solving the KBE for fermions in momentum space. This will provide a possibility to crosscheck the reliability of our position space approach. After a successful reproduction of the results for homogeneous systems in (1+1) dimensions, we want to extend our investigations to (3+1) dimensions and to more general, non-homogeneous setups. As a long-term goal, we would like to use our framework for a proper numerical simulation of

the process of *electroweak baryogenesis*, including the CP-violating scattering of the surrounding plasma with the expanding bubble wall as well as bubble collisions. If the latter leads to the emerge of gravitational waves we could calculate the respective spectrum, which then could be compared to recent data from LIGO.

Appendices

A. Time evolution of the ϕ_- mode-function

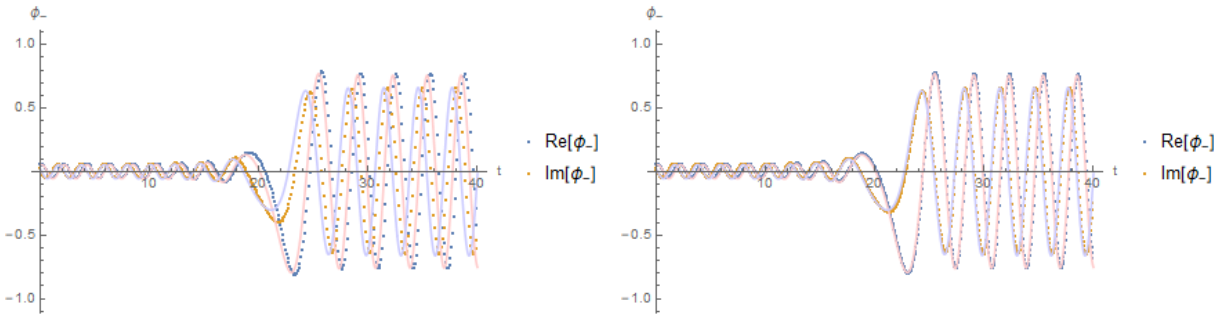


Figure A.1.: Time evolution of the mode-function ϕ_- for the momentum $\mathbf{p} = 0.31$ and two different choices of Δt. The analytic solutions are displayed in the light colors. While we clearly see deviations for the $\Delta t = 0.1$ (left) run, the $\Delta t = 0.05$ (right) run agrees very well with the analytic solution.

B. Momentum discretization for scalar fields

For the scalars we employ the following standard momenta discretization:

$$p^2 \to \sum_{i=1}^{d} \frac{4}{(\Delta x)^2} \sin^2\left(\frac{\Delta x p_i}{2}\right), \qquad p_i = \frac{2\pi n_i}{N\Delta x}$$

This can be motivated by taking a look at the standard discretization of the second derivative appearing in the Klein-Gordon-Equation:

$$\partial_x^2 e^{ipx} \to e^{ipx} \frac{\left[e^{ip\Delta x} + e^{-ip\Delta x} - 2\right]}{(\Delta x)^2} = -e^{ipx} \frac{4\sin^2\left(\frac{\Delta x p_i}{2}\right)}{(\Delta x)^2}$$

In the limit $\Delta x \to 0$ we obtain:

$$\lim_{\Delta x \to 0} p^2 = p_i^2.$$

C. Dependence on lattice parameters

$N = 20, \Delta x = 0.5$

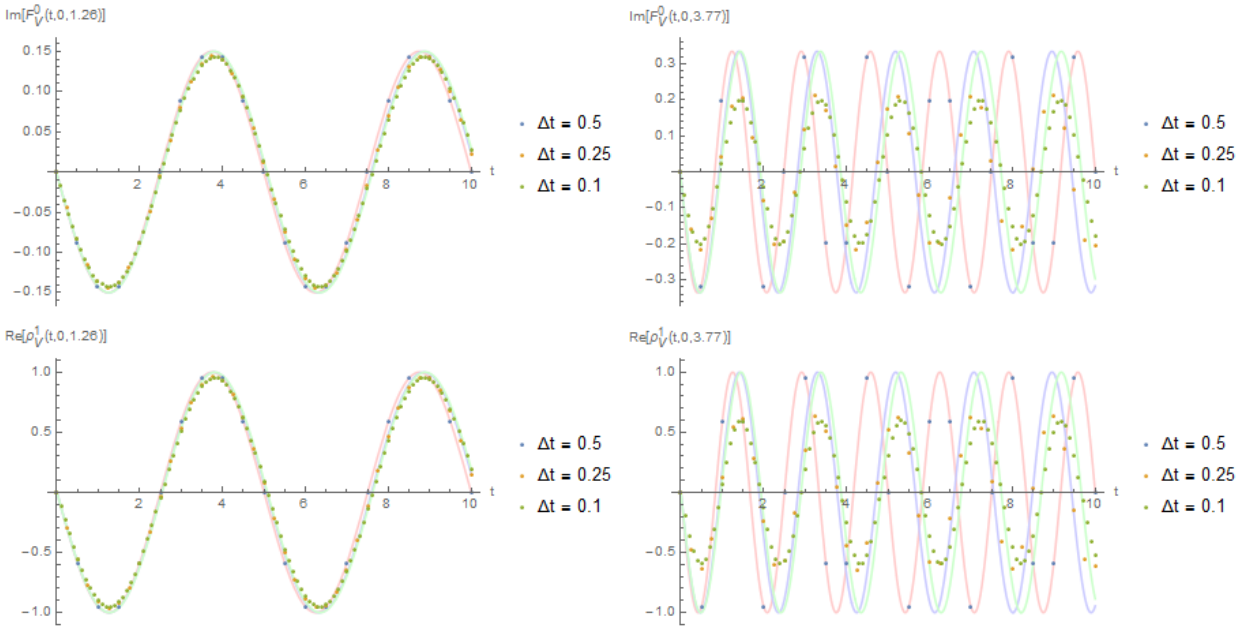


Figure C.1.: Time evolution of the purely imaginary F_V^0 (top) and the real F_V^1 (bottom) component for two different momenta, a fixed lattice spacing $\Delta x = 0.5$ and different values of Δt. The exact solutions are given in light red, blue and green. There are no apparent differences to the run with $N = 10$.

$N = 20, \Delta x = 0.25$

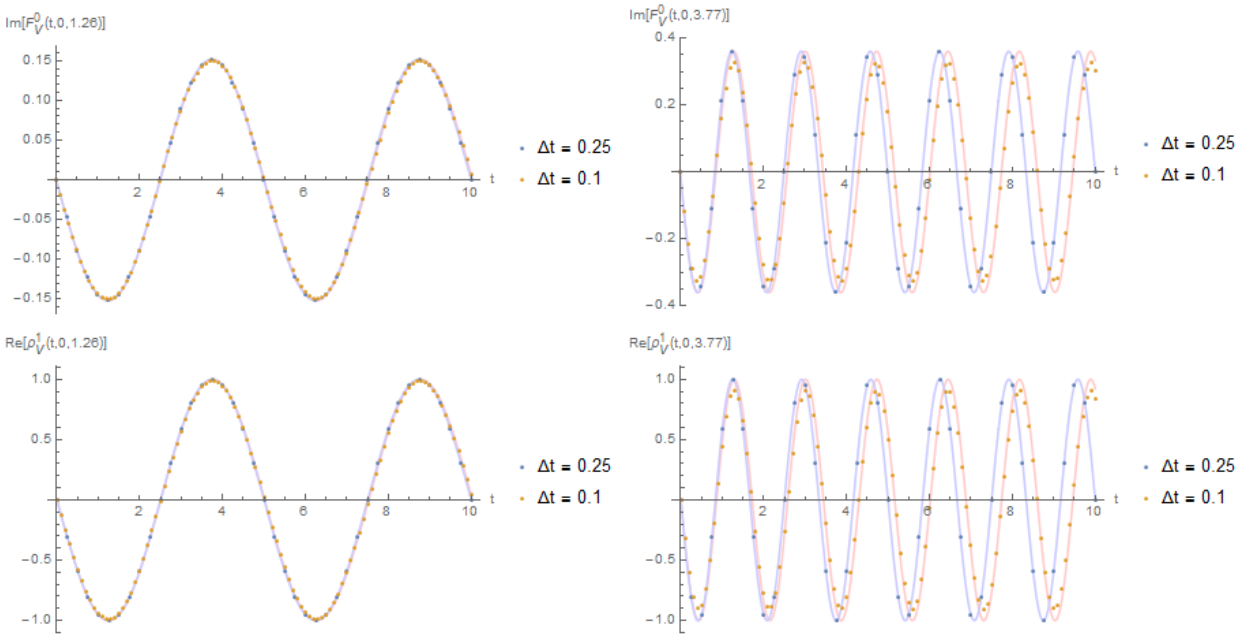


Figure C.2.: Time evolution of the purely imaginary F_V^0 (top) and the real F_V^1 (bottom) component for two different momenta, a fixed lattice spacing $\Delta x = 0.25$ and different values of Δt. The exact solutions are given in light red, blue and green. Compared to the run with $\Delta x = 0.5$ we see less deviations from the exact solution for $r < 1$.

D. Quadratic potential - Spectral function

Fermions:

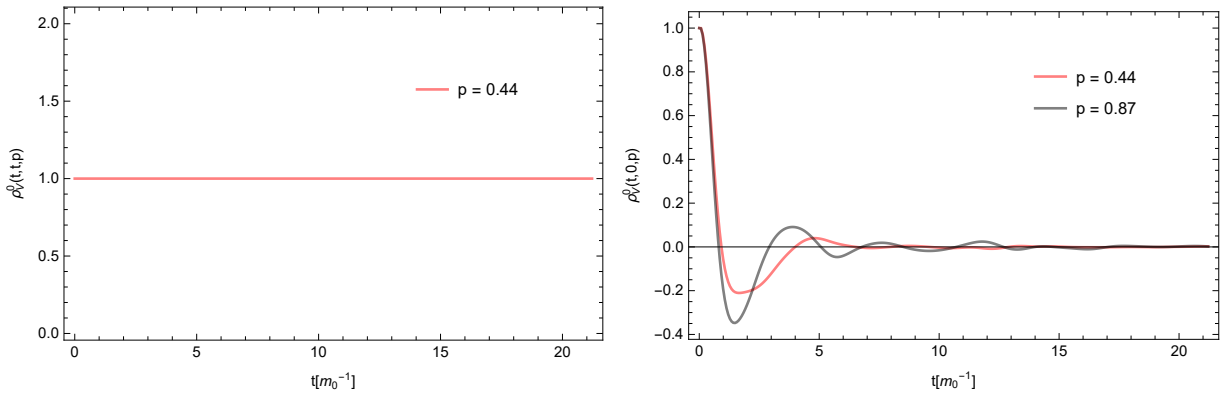


Figure D.1.: Equal and unequal time evolution of ρ^0_V. While for equal times $\rho^0_V(t,t,p)$ is conserved exactly, we see a damped behavior for the correlation to the initial time $\rho^0_V(t,0,p)$.

Scalars:

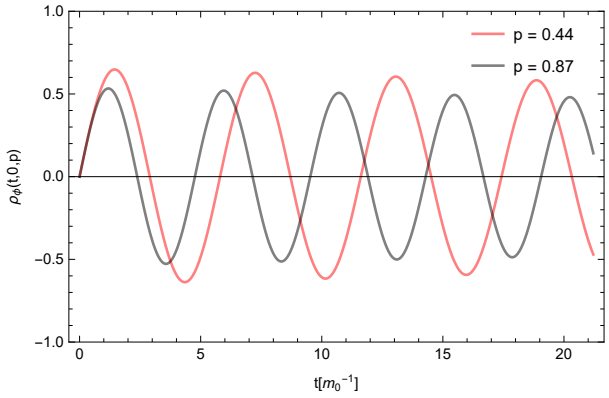


Figure D.2.: Correlation to the initial time of the scalar spectral function $\rho_\phi(t,0,p)$. As for the statistical function we see a damped behavior.

E. Quadratic potential - Time evolution of the particle number distributions $n^f(p)$ and $n(\epsilon)$

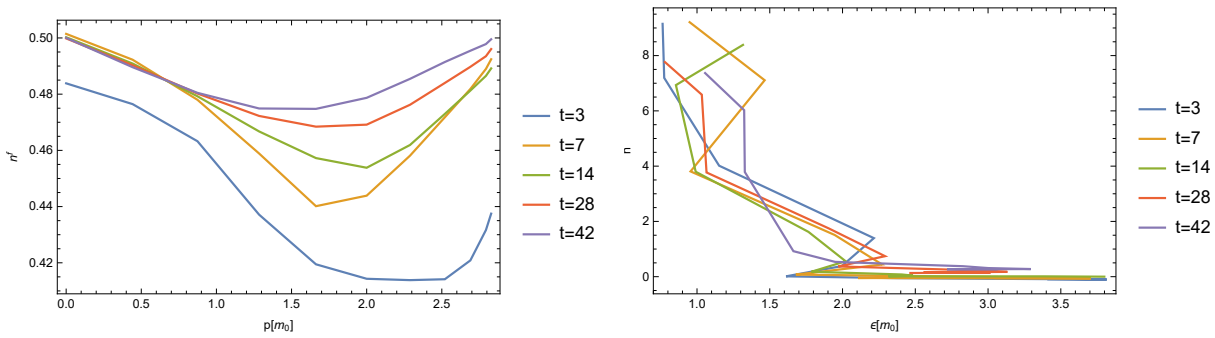


Figure E.1.: Time evolution of the fermionic (l.h.s.) and scalar (r.h.s.) particle number distributions $n^f(p)$ and $n(\epsilon)$.

F. Memory Integrals for quadratic potential

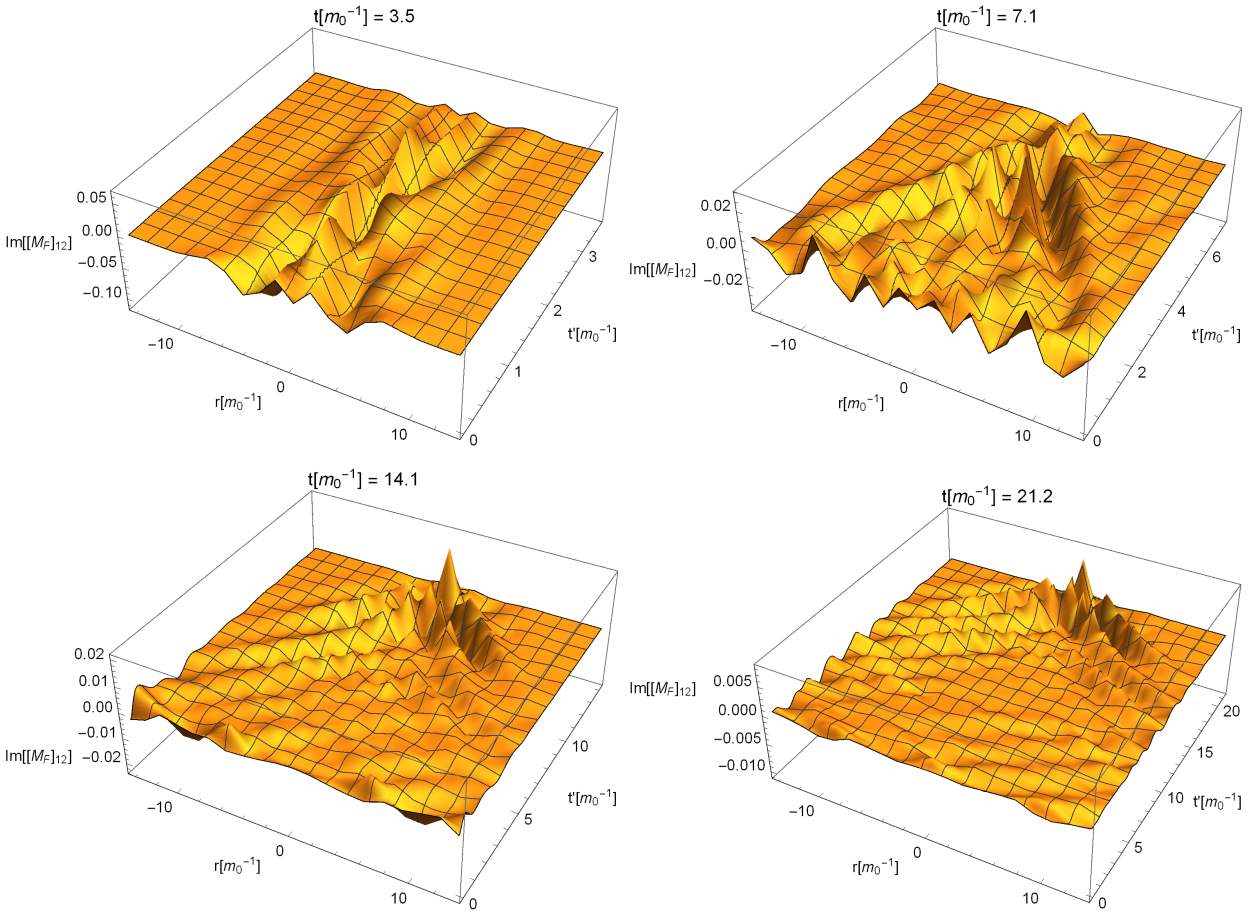


Figure F.1.: Snapshots of the imaginary part of the $[M_F]_{12}$ component of the memory intergral. While the contributions are cumulated near $r = 0$ for early times, they spread out in space with ongoing time. For late times ($t = 14.1, 21.2$) a damping with growing distance $t - t'$ becomes visible.

G. $\lambda = 24$

Correlation to the initial time

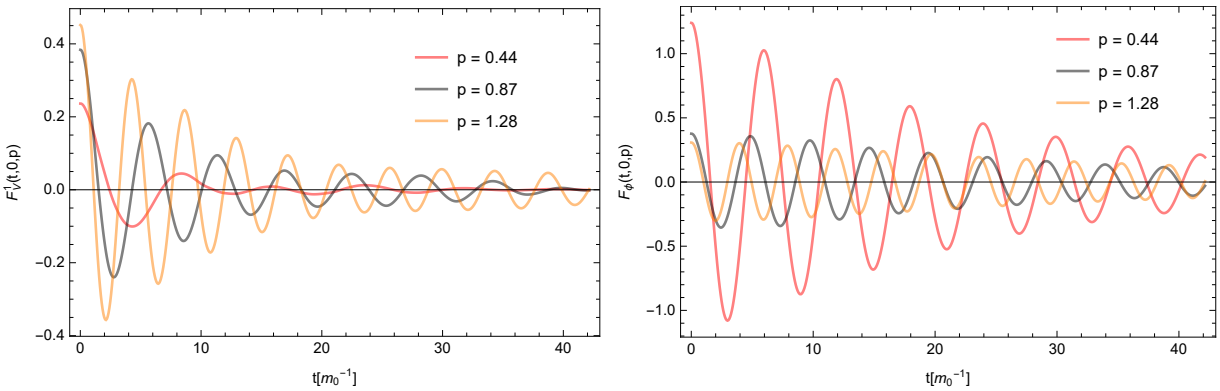


Figure G.1.: Correlation to the initial time of the F_V^1 (l.h.s.) component of the fermion statistical function and the scalar statistical function F_ϕ (r.h.s.). In both cases we see the expected damped behavior.

Time evolution of the particle number distributions $n^f(p)$ and $n(\epsilon)$:

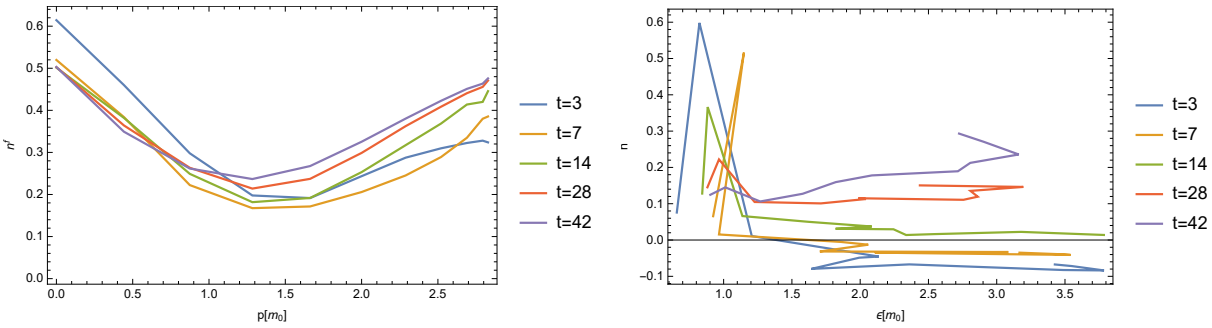


Figure G.2.: Time evolution of the fermionic (l.h.s.) and scalar (r.h.s.) particle number distributions $n^f(p)$ and $n(\epsilon)$.

H. Source code: 1-point function

Core of the update routine for the discretized Dirac Equation with an imaginary mass:

```
//Calculate staggered grid at t=0 via spatial interpolation

if(t==0){
    for (int l=0;l<2;l++){  //loop over spinor components


        for (int j=0;j<timestepsize;j++){ //loop over all spacialpoints at t=0

            //evaluate value of staggered grid
            if(j==(timestepsize-1)){
                newderiv=0.5*(func->val(j+l*compoffset) + func->val(l*compoffset));
            }
            else{
                newderiv  = 0.5*(func->val(j+l*compoffset) + func->val(j+1+l*compoffset));
            }
            //assign value of staggered grid
            func->derivassign(j+l*derivoffset,newderiv);
        }
    }
}

//First step: Evolve staggered grid

for (int l=0;l<2;l++) //loop over spinor components
{
    int m = (l+1)%2; //factor for mixing components

    //loop to cover all relevant space-time points
    for(long long j=0;j<timestepsize;j++)
    {
        //calculate spatial derivative using regular grid
        spatialderiv = (func->val(lattice->Neighbour(tindex+j+m*compoffset,0))
            -func->val(tindex+j+m*compoffset))/deltax;

        if (tindex == 0){//first step to evolve space-staggered grid from t=0 to t=1/2

            //evaluate updated value
            newderiv = (1.0/(4.0+pow(-1.0,l)*imagunit*deltat*mass))
                *((4.0-pow(-1.0,l)*imagunit*deltat*mass)
                *func->derivval(j+l*derivoffset) //value at t=0
                -2.0*deltat*spatialderiv //spatial derivative
                -2.0*deltat*massimag*pow(-1.0,l) //imaginary part of mass
                *(func->val(tindex+j+m*compoffset)));

            //assign updated value
            func->derivassign(j+l*derivoffset, newderiv);
        }
        else{//evolve staggered grid t=n-1/2 to t = n+1/2

            //evaluate updated value
            newderiv = (1.0/(2.0+pow(-1.0,l)*mass*imagunit*deltat))
                *((2.0-pow(-1.0,l)*imagunit*deltat*mass)
                *func->derivval(j+l*derivoffset) //value at t=n-1/2
```

```
                -2.0*deltat*spatialderiv //spatial derivative
                -2.0*deltat*massimag*pow(-1.0,1) //imaginary part of mass
                *(func->val(tindex+j+m*compoffset)));

            //assign updated value
            func->derivassign(j+l*derivoffset, newderiv);
        }
    }

}

//Second step: Evolve regular grid

for (int l=0;l<2;l++) //loop over spinor components
{
    int m = (l+1)%2; //factor for mixing components

    //loop to cover all relevant space-time points
    for(long long j=0;j<timestepsize;j++)
    {
        //calculate spatial derivative using staggered grid
        spatialderiv = (func->derivval(j+m*derivoffset)
            -func->derivval(lattice->Neighbour(j+m*derivoffset,1)))/deltax;

        //evolve regular grid from t = n to t = n+1

        //evaluate updated value
        newval = (1.0/(2.0+pow(-1.0,1)*mass*imagunit*deltat))
            *((2.0-pow(-1.0,1)*imagunit*deltat*mass)
            *func->val(tindex+j+l*compoffset) //value at t=n
            -2.0*deltat*spatialderiv //spatial derivative
            -2.0*deltat*massimag*pow(-1.0,1) //imaginary part of mass
            *(func->derivval(j+m*derivoffset)));

        //assign updated value
        func->assign(tincindex+j+l*compoffset, newval); //assign updated value
    }
}
```

I. Source code: 2-point function

Core of the update routine for the sub-operator $\mathcal{A}_1$:

```
//update of the two point objects with the calculated memory integral

for(int l=0;l<(matrixentries);l++){
    long long spacialindex0=0;
    int row = l/matrixdim;  //row index of the considered matrix element
    int col = l%matrixdim;  //column index of the considered matrix element

    //spatial index of first point of greensfunction
    for(long long k=0;k<spacialpoints;k++){

        //spatial and time index of second point of greensfunction (y0<x0)
        for(long long j=0;j<timestepsize;j++)
        {
         dynindex=j+spacialindex0;

         dynindexxstaggered= dynindex+stepsize;
         if(k==spacialpoints-1){dynindexxstaggered = j;}
         dynindexystaggered = dynindex+1;
         if(j==spacialpoints-1){dynindexystaggered = spacialindex0;}

         gamma0Mstat = 0.0;             // gamma0*M(x,y)_stat
         gamma0Mspec = 0.0;             // gamma0*M(x,y)_spec
         gamma0Mstatstaggered = 0.0;    // gamma0*M(x+1/2,y)_stat for staggered grid
         gamma0Mspecstaggered = 0.0;    // gamma0*M(x+1/2,y)_spec for staggered grid

         //matrixmultiplication for fixed outer indices (row and col)
         for(int m = 0; m<(matrixdim);m++){

            gamma0Mstat += gammas[row*(matrixdim)+m]
                *func->memval(dynindex+(m*matrixdim+col)*dyngreenoffset);
            gamma0Mspec += gammas[row*(matrixdim)+m]
                *func->specmemval(dynindex+(m*matrixdim+col)*dyngreenoffset);

            //spatial interpolation to get staggered memory vals
            //time staggering already done in CalcFermionMem -> memstagval,..
            gamma0Mstatstaggered += 0.5*(gammas[row*(matrixdim)+m]
                *func->memstagval(dynindex+(m*matrixdim+col)*dyngreenoffset)
                +gammas[row*(matrixdim)+m]
                *func->memstagval(dynindexxstaggered+(m*matrixdim+col)*dyngreenoffset));
            gamma0Mspecstaggered += 0.5*(gammas[row*(matrixdim)+m]
                *func->specmemstagval(dynindex+(m*matrixdim+col)*dyngreenoffset)
                +gammas[row*(matrixdim)+m]
                *func->specmemstagval(dynindexxstaggered+(m*matrixdim+col)*dyngreenoffset));
         }

         //evaluate and assign F(n+1,m)tilde, m<n
         newval= func->val(tindex+dynindex+l*greensize)-imagunit*deltat*gamma0Mstat;
         func->assign(tincindex+dynindex+l*greensize,newval);

         //evaluate and assign rho(n+1,m)tilde, m<n
         newval= func->specval(tindex+dynindex+l*greensize)-imagunit*deltat*gamma0Mspec;
         func->specassign(tincindex+dynindex+l*greensize,newval);

```

```
54          //evaluate and assign staggered F(n+1/2,m) -> F(n+3/2,m)tilde , m<n
55          newderiv= func->derivval(dynindex+1*dyngreenoffset)-imagunit*deltat*
    gamma0Mstatstaggered;
56          func->derivassign(dynindex+1*dyngreenoffset,newderiv);
57
58          //evaluate and assign staggered rho(n+1/2,m) -> rho(n+3/2,m)tilde , m<n
59          newderiv= func->specderivval(dynindex+1*dyngreenoffset)-imagunit*deltat*
    gamma0Mspecstaggered;
60          func->specderivassign(dynindex+1*dyngreenoffset,newderiv);
61          }
62
63          //spatial and time index of second point of greensfunction (y0=x0)
64          for(long long j=timestepsize;j<(timestepsize+spacialpoints);j++)
65          {
66              long long dynindex=j+spacialindex0;
67
68              //swap spatial coordinates at equal time
69              long long permdynindex = func->greenspermindex(dynindex+t*func->timestep(0))
70                  %func->timestep(0);
71
72              //dynindex with spatial index of first point increase by one
73              //needed to calculate space-staggered memory integral via interpolation
74              long long dynindexxstaggered = dynindex+stepsize;
75              if(k==spacialpoints-1){dynindexxstaggered = j;}
76
77              //dynindex with spatial index of second point increase by one
78              //needed to calculate space-staggered memory integral via interpolation
79              long long dynindexystaggered = dynindex+1;
80              if(j==spacialpoints-1){dynindexystaggered = spacialindex0;}
81
82              //swap dyindexystaggered to calculate staggered M(y,x)
83              long long permdynindexystaggered = func->greenspermindex(dynindexystaggered
84                  +t*func->timestep(0))%func->timestep(0);
85
86              gamma0Mstat = 0.0;                  //gamma0*M(x,y)_stat
87              gamma0Mspec = 0.0;                  //gamma0*M(x,y)_spec
88              gamma0Mstatdagger = 0.0;            //gamma0*M(y,x)_stat^dagger (y0-evolution)
89              gamma0Mspecdagger = 0.0;            //gamma0*M(y,x)_spec^dagger (y0-evolution)
90              gamma0Mstatstaggered = 0.0;         //gamma0*M(x+1/2,y)_stat for staggered grid
91              gamma0Mspecstaggered = 0.0;         //gamma0*M(x+1/2,y)_spec for staggered grid
92              gamma0Mstatdaggerstaggered = 0.0; //gamma0*M(y+1/2,x)_stat^dagger for
93                                                  //staggered grid (y0-evolution)
94              gamma0Mspecdaggerstaggered = 0.0; //gamma0*M(y+1/2,x)_spec^dagger
95                                                  //for staggered grid (y0-evolution)
96
97              //matrixmultiplication for fixed outer indices (row and col)
98              for(int m = 0; m<(matrixdim);m++){
99
100                 gamma0Mstat += gammas[row*(matrixdim)+m]
101                     *func->memval(dynindex+(m*matrixdim+col)*dyngreenoffset);
102                 gamma0Mspec += gammas[row*(matrixdim)+m]
103                     *func->specmemval(dynindex+(m*matrixdim+col)*dyngreenoffset);
104                 gamma0Mstatdagger += gammas[row*(matrixdim)+m]
105                     *std::conj(func->memval(permdynindex+(col*matrixdim+m)*dyngreenoffset));
106                 gamma0Mspecdagger += gammas[row*(matrixdim)+m]
107                     *std::conj(func->specmemval(permdynindex+(col*matrixdim+m)*dyngreenoffset))
    ;
108
109                 //spatial interpolation to get staggered memory vals
110                 //time staggering already done in CalcFermionMem -> memstagval,..
111                 gamma0Mstatstaggered += 0.5*(gammas[row*(matrixdim)+m]
112                     *func->memstagval(dynindex+(m*matrixdim+col)*dyngreenoffset)
113                     +gammas[row*(matrixdim)+m]*func->memstagval(dynindexxstaggered
114                     +(m*matrixdim+col)*dyngreenoffset));
115                 gamma0Mspecstaggered += 0.5*(gammas[row*(matrixdim)+m]
```

```
                        *func->specmemstagval(dynindex+(m*matrixdim+col)*dyngreenoffset)
                        +gammas[row*(matrixdim)+m]*func->specmemstagval(dynindexxstaggered
                        +(m*matrixdim+col)*dyngreenoffset));
                gamma0Mstatdaggerstaggered += 0.5*(gammas[row*(matrixdim)+m]
                        *std::conj(func->memstagval(permdynindex+(col*matrixdim+m)
                        *dyngreenoffset))+gammas[row*(matrixdim)+m]
                        *std::conj(func->memstagval(permdynindexystaggered
                        +(col*matrixdim+m)*dyngreenoffset)));
                gamma0Mspecdaggerstaggered += 0.5*(gammas[row*(matrixdim)+m]
                        *std::conj(func->specmemstagval(permdynindex+(col*matrixdim+m)
                        *dyngreenoffset))+gammas[row*(matrixdim)+m]
                        *std::conj(func->specmemstagval(permdynindexystaggered
                        +(col*matrixdim+m)*dyngreenoffset)));
            }

            //evaluate and assign F(n+1,n)tilde
            newval= func->val(tindex+dynindex+1*greensize) - imagunit*deltat*gamma0Mstat;
            func->assign(tincindex+dynindex+1*greensize,newval);

            //evaluate and assign rho(n+1,n)tilde
            newval= func->specval(tindex+dynindex+1*greensize) - imagunit*deltat*gamma0Mspec;
            func->specassign(tincindex+dynindex+1*greensize,newval);

            //evaluate and assign F(n+1,n+1)tilde
            newval= func->val(tindex+dynindex+1*greensize) - 1.0*imagunit*deltat
                *(gamma0Mstat-gamma0Mstatdagger);
            func->assign(tincindex+dynindex+1*greensize+spacialpoints,newval);

            //evaluate and assign rho(n+1,n+1)tilde
            newval= func->specval(tindex+dynindex+1*greensize) - 1.0*imagunit*deltat
                *(gamma0Mspec+gamma0Mspecdagger);
            func->specassign(tincindex+dynindex+1*greensize+spacialpoints,newval);

            //evaluate and assign F(n+1/2,n+1/2)tilde (stored one ytimestep ahead)
            newderiv= func->derivval(dynindex+1*dyngreenoffset) - 1.0*imagunit*deltat
                *(gamma0Mstatstaggered-gamma0Mstatdaggerstaggered);
            func->derivassign(dynindex+1*dyngreenoffset+spacialpoints,newderiv);

            //evaluate and assign rho(n+1/2,n+1/2)tilde (stored one ytimestep ahead)
            newderiv= func->specderivval(dynindex+1*dyngreenoffset) - 1.0*imagunit*deltat
                *(gamma0Mspecstaggered+gamma0Mspecdaggerstaggered);
            func->specderivassign(dynindex+1*dyngreenoffset+spacialpoints,newderiv);

            //evaluate and assign F(n+1/2,n)tilde (euler step)
            newderiv= func->val(tindex+dynindex+1*greensize) - imagunit*0.5*deltat
                *(gamma0Mstat);
            func->derivassign(dynindex+1*dyngreenoffset,newderiv);

            //evaluate and assign rho(n+1/2,n)tilde (euler step)
            newderiv= func->specval(tindex+dynindex+1*greensize) - imagunit*0.5*deltat
                *(gamma0Mspec);
            func->specderivassign(dynindex+1*dyngreenoffset,newderiv);

        }

    spacialindex0+=stepsize;

    }

}
```

Core of the update routine for the sub-operator $\mathcal{A}_2$:

```
//First Step: Evolve staggered grid

//loop around matrix components of F/rho - (1+1)D: F11,F12,F21,F22
for (int l=0; l<(matrixentries);l++)
{       spacialindex0=0;

        //spatial index of first point of greensfunction
        for(int k=0; k<(spacialpoints);k++)
        {

            row = l/matrixdim; //row index of the considered matrixelement
            col = l%matrixdim; //column index of the considered matrixelement


            //contributions from the massterm in x0- and y0-direction

            x0_massterm = -imagunit*mass*gammas[row*matrixdim+row]; //i*mass*gamma0*F
                                                                    //with gamma0 diagonal
            y0_massterm =  imagunit*mass*gammas[col*matrixdim+col]; //i*mass*F*gamma0
                                                                    //with gamma0 diagonal

            //Update staggered grid along the diagonal

            //spatial and time index of second point of greensfunction (y0=x0)
            for(long long j=causaltimestep;j<(causaltimestep+ytimestep);j++)
            {
                dynindex=j+spacialindex0;

                //at t==0: euler step to evolve (space) staggered to (1/2,1/2) in time
                if(t==0){
                spatialderiv_stat = 0.0;
                spatialderiv_spec = 0.0;

                //calculation of the matrixproducts ~alpha * delx_F/rho,...  appearing on the
                //r.h.s of KBE in x0- and y0-direction
                for(int r=0;r<matrixdim;r++){
                    spatialderiv_stat += -alpha[row*matrixdim+r]
                        *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
                        *greensize,(r*matrixdim+col),func);    //  -alpha * delx_F
                    spatialderiv_stat += DiracEqSpatialDerivative2(tindex+dynindex
                        +(row*matrixdim+r)*greensize,(row*matrixdim+r),func)
                        *alpha[r*matrixdim+col];               //  +dely_F * alpha
                    spatialderiv_spec += -alpha[row*matrixdim+r]
                        *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
                        *greensize,(r*matrixdim+col),func,0); //  -alpha * delx_rho
                    spatialderiv_spec += DiracEqSpatialDerivative2(tindex+dynindex
                        +(row*matrixdim+r)*greensize,(row*matrixdim+r),func,0)
                        *alpha[r*matrixdim+col];               //  +dely_rho * alpha

                }
                //euler step to evaluate F(1/2,1/2)
                newderiv = (1.0/(4.0-deltat*(x0_massterm+y0_massterm)))
                    *((4.0+deltat*(x0_massterm+y0_massterm))*func->derivval(dynindex
                    +l*dyngreenoffset+3*ytimestep)+2.0*deltat*spatialderiv_stat);

                //assign the value of F(1/2,1/2)
                func->derivassign(dynindex+l*dyngreenoffset+ytimestep,newderiv);

                //euler step to evaluate rho(1/2,1/2)
                newderiv = (1.0/(4.0-deltat*(x0_massterm+y0_massterm)))
                    *((4.0+deltat*(x0_massterm+y0_massterm))*func->specderivval(dynindex
                    +l*dyngreenoffset+3*ytimestep)+2.0*deltat*spatialderiv_spec);

```

```
64                  //assign the value of rho(1/2,1/2)
65                  func->specderivassign(dynindex+l*dyngreenoffset+ytimestep,newderiv);
66
67                  }
68
69                  //at t!=0: evolve staggered from (n-1/2,n-1/2) to (n+1/2,n+1/2)
70                  else if(t!=0){
71
72                  spatialderiv_stat = 0.0;
73                  spatialderiv_spec = 0.0;
74
75                  //calculation of the matrixproducts ~alpha * delx_F/rho,...  appearing on the
76                  //r.h.s of KBE in x0- and y0-direction
77                  for(int r=0;r<matrixdim;r++){
78                      spatialderiv_stat += -alpha[row*matrixdim+r]
79                          *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
80                          *greensize,(r*matrixdim+col),func); //  - alpha * delx_F
81                      spatialderiv_stat += DiracEqSpatialDerivative2(tindex+dynindex
82                          +(row*matrixdim+r)*greensize,(row*matrixdim+r),func)
83                          *alpha[r*matrixdim+col]; //  + dely_F * alpha
84                      spatialderiv_spec += -alpha[row*matrixdim+r]
85                          *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
86                          *greensize,(r*matrixdim+col),func,0); //  - alpha * delx_rho
87                      spatialderiv_spec += DiracEqSpatialDerivative2(tindex+dynindex
88                          +(row*matrixdim+r)*greensize,(row*matrixdim+r),func,0)
89                          *alpha[r*matrixdim+col]; //  + dely_rho * alpha
90
91                  }
92                  //evaluate the value of F(n+1/2,n+1/2)
93                  newderiv = (1.0/(2.0-deltat*(x0_massterm+y0_massterm)))
94                      *((2.0+deltat*(x0_massterm+y0_massterm))*func->derivval(dynindex
95                      +l*dyngreenoffset+splitorder*ytimestep)+2.0*deltat*spatialderiv_stat);
96
97                  //assign the value of F(n+1/2,n+1/2)
98                  func->derivassign(dynindex+l*dyngreenoffset+ytimestep,newderiv);
99
100                 //evaluate the value of rho(n+1/2,n+1/2)
101                 newderiv = (1.0/(2.0-deltat*(x0_massterm+y0_massterm)))
102                     *((2.0+deltat*(x0_massterm+y0_massterm))*func->specderivval(dynindex
103                     +l*dyngreenoffset+splitorder*ytimestep)+2.0*deltat*spatialderiv_spec);
104
105                 //assign the value of rho(n+1/2,n+1/2)
106                 //last two steps are skipped to avoid overwriting
107                 if(t < lattice->getT()-2){
108                 func->specderivassign(dynindex+l*dyngreenoffset+ytimestep,newderiv);
109                 }
110
111             }
112             }
113
114             //Update staggered grid in horizontal (x0) direction
115
116             //spatial and time index of second point of greensfunction (y0<x0)
117             for(long long j=0;j<causaltimestep;j++)
118             {
119
120                 dynindex=j+spacialindex0;
121
122                 spatialderiv_stat = 0.0;
123                 spatialderiv_spec = 0.0;
124
125                 //calculation of the matrixproduct alpha * delx_F/rho appearing on the
126                 //r.h.s of KBE in x0-direction
127                 for(int r=0;r<matrixdim;r++){
128                     spatialderiv_stat += -alpha[row*matrixdim+r]
```

```
                        *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
                        *greensize,(r*matrixdim+col),func);    //  -alpha * delx_F
                    spatialderiv_spec += -alpha[row*matrixdim+r]
                        *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
                        *greensize,(r*matrixdim+col),func,0); //  -alpha * delx_rho
                }

                //evaluate the value of F(n+1/2,m), m<n
                newderiv = (1.0/(2.0-deltat*x0_massterm))
                    *((2.0+deltat*x0_massterm)*func->derivval(dynindex
                    +l*dyngreenoffset)+2.0*deltat*spatialderiv_stat);

                //assign the value of F(n+1/2,m), m<n
                func->derivassign(dynindex+l*dyngreenoffset,newderiv);

                //evaluate the value of rho(n+1/2,m), m<n
                newderiv = (1.0/(2.0-deltat*x0_massterm))
                    *((2.0+deltat*x0_massterm)*func->specderivval(dynindex
                    +l*dyngreenoffset)+2.0*deltat*spatialderiv_spec);

                //assign the value of F(n+1/2,m), m<n
                func->specderivassign(dynindex+l*dyngreenoffset,newderiv);
            }


            //spatial and time index of second point of greensfunction (y0=x0)
            for(long long j=causaltimestep;j<(causaltimestep + ytimestep);j++)
            {
             dynindex=j+spacialindex0;

             spatialderiv_stat = 0.0;
             spatialderiv_spec = 0.0;

             //calculation of the matrixproduct alpha * delx_F/rho appearing on the
             //r.h.s of KBE in x0-direction
             for(int r=0;r<matrixdim;r++){
                 spatialderiv_stat += -alpha[row*matrixdim+r]
                    *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
                    *greensize,(r*matrixdim+col),func);    //  -alpha * delx_F
                 spatialderiv_spec += -alpha[row*matrixdim+r]
                    *DiracEqSpatialDerivative(tindex+dynindex+(r*matrixdim+col)
                    *greensize,(r*matrixdim+col),func,0); //  -alpha * delx_rho
             }
             //euler step to evaluate F(n+1/2,n)
             newderiv = (1.0/(4.0-deltat*x0_massterm))
                *((4.0+deltat*x0_massterm)*func->derivval(dynindex
                +l*dyngreenoffset+2*ytimestep)+2.0*deltat*spatialderiv_stat);

             //assign the value of F(n+1/2,n)
             func->derivassign(dynindex+l*dyngreenoffset,newderiv);

             //euler step to evaluate of rho(n+1/2,n)
             newderiv = (1.0/(4.0-deltat*x0_massterm))
                *((4.0+deltat*x0_massterm)*func->specderivval(dynindex
                +l*dyngreenoffset+2*ytimestep)+2.0*deltat*spatialderiv_spec);

             //assign the value of rho(n+1/2,n)
             func->specderivassign(dynindex+l*dyngreenoffset,newderiv);

            }


         spacialindex0+=stepsize; //increase spatial index of first point of greensfunction

        }
```

```
194
195 }
196
197 //Second Step: Evolve regular grid in horizontal (x0) direction
198
199 //if splitting, memory operator assignes values at tincindex+ytimestep
200 //no memory operator for t=0
201 if(splitorder==1 && t!=0){tindex=tincindex;}
202
203 //loop around matrix components of F/rho - (1+1)D: F11,F12,F21,F22
204 for (int l=0; l<(matrixentries);l++)
205 {       spacialindex0=0;
206
207         //spatial index of first point of greensfunction
208         for(int k=0; k<(spacialpoints);k++)
209         {
210
211             row = l/matrixdim; //row index of the considered matrixelement
212             col = l%matrixdim; //column index of the considered matrixelement
213
214             x0_massterm = -imagunit*mass*gammas[row*matrixdim+row]; //i*mass*gamma0*F
215                                                                     //with gamma0 diagonal
216
217             //spatial and time index of second point of greensfunction (y0<x0)
218             for(long long j=0;j<causaltimestep;j++)
219             {
220                 dynindex=j+spacialindex0;
221
222                 spatialderiv_stat = 0.0;
223                 spatialderiv_spec = 0.0;
224
225                 //calculation of the matrixproduct alpha * delx_F/rho appearing on the
226                 //r.h.s of KBE in x0-direction
227                 for(int r=0;r<matrixdim;r++){
228                     spatialderiv_stat += -alpha[row*matrixdim+r]
229                         *DiracEqSpatialDerivativeDeriv(dynindex+(r*matrixdim+col)
230                         *dyngreenoffset,(r*matrixdim+col),func);    //  -alpha * delx_F
231                     spatialderiv_spec += -alpha[row*matrixdim+r]
232                         *DiracEqSpatialDerivativeDeriv(dynindex+(r*matrixdim+col)
233                         *dyngreenoffset,(r*matrixdim+col),func,0); //  -alpha * delx_rho
234                 }
235
236                 //evaluate the value of F(n+1,m), m<n
237                 newval = (1.0/(2.0-deltat*x0_massterm))
238                     *((2.0+deltat*x0_massterm)*func->val(tindex+dynindex+l*greensize)
239                     +2.0*deltat*spatialderiv_stat);
240
241                 //assign the value of F(n+1,m), m<n
242                 func->assign(tincindex+dynindex+l*greensize,newval);
243
244                 //evaluate the value of rho(n+1,m), m<n
245                 newval = (1.0/(2.0-deltat*x0_massterm))
246                     *((2.0+deltat*x0_massterm)*func->specval(tindex+dynindex+l*greensize)
247                     +2.0*deltat*spatialderiv_spec);
248
249                 //assign the value of rho(n+1,m), m<n
250                 func->specassign(tincindex+dynindex+l*greensize,newval);
251
252
253
254             }
255
256             //spatial and time index of second point of greensfunction (y0=x0)
257             for(long long j=causaltimestep;j<(causaltimestep + ytimestep);j++)
258             {
```

```
                dynindex=j+spacialindex0;

                spatialderiv_stat = 0.0;
                spatialderiv_spec = 0.0;

                //calculation of the matrixproduct alpha * delx_F/rho appearing on the
                //r.h.s of KBE in x0-direction
                for(int r=0;r<matrixdim;r++){
                    spatialderiv_stat += -alpha[row*matrixdim+r]
                        *DiracEqSpatialDerivativeDeriv(dynindex+(r*matrixdim+col)
                        *dyngreenoffset,(r*matrixdim+col),func);    //  -alpha * delx_F
                    spatialderiv_spec += -alpha[row*matrixdim+r]
                        *DiracEqSpatialDerivativeDeriv(dynindex+(r*matrixdim+col)
                        *dyngreenoffset,(r*matrixdim+col),func,0); //  -alpha * delx_rho
                }

                //evaluate the value of F(n+1,n)
                newval = (1.0/(2.0-deltat*x0_massterm))
                    *((2.0+deltat*x0_massterm)*func->val(tindex+dynindex+l*greensize)
                    +2.0*deltat*spatialderiv_stat);

                //assign the value of F(n+1,n)
                func->assign(tincindex+dynindex+l*greensize,newval);

                //evaluate the value of rho(n+1,n)
                newval = (1.0/(2.0-deltat*x0_massterm))
                    *((2.0+deltat*x0_massterm)*func->specval(tindex+dynindex+l*greensize)
                    +2.0*deltat*spatialderiv_spec);

                //assign the value of rho(n+1,n)
                func->specassign(tincindex+dynindex+l*greensize,newval);


            }

            spacialindex0+=stepsize; //increase spatial index of first point of greensfunction

        }

}

//Third Step: Evolve regular grid along diagonal

//if splitting, memory operator assignes values at tincindex+ytimestep
//no memory operator for t=0
if(splitorder==1 && t!=0){tindex=tincindex+ytimestep;}

//loop around matrix components of F/rho - (1+1)D: F11,F12,F21,F22
for (int l=0; l<(matrixentries);l++)
{
    spacialindex0=0;

    //spatial index of first point of greensfunction
    for(int k=0; k<(spacialpoints);k++)
    {

        row = l/matrixdim; //row index of the considered matrixelement
        col = l%matrixdim; //column index of the considered matrixelement

        //contributions from the massterm in x0- and y0-direction

        x0_massterm = -imagunit*mass*gammas[row*matrixdim+row]; //i*mass*gamma0*F
                                                                //with gamma0 diagonal
        y0_massterm =  imagunit*mass*gammas[col*matrixdim+col]; //i*mass*F*gamma0
```

```
                                                        //with gamma0 diagonal

        //spatial and time index of second point of greensfunction (y0=x0)
        for(long long j=causaltimestep;j<causaltimestep+ytimestep;j++)
        {
            dynindex=j+spacialindex0;

            spatialderiv_stat = 0.0;
            spatialderiv_spec = 0.0;

            //calculation of the matrixproducts ~alpha * delx_F/rho,...  appearing on the
            //r.h.s of KBE in x0- and y0-direction
            for(int r=0;r<matrixdim;r++){

                spatialderiv_stat += -alpha[row*matrixdim+r]
                    *DiracEqSpatialDerivativeDeriv(dynindex+(r*matrixdim+col)
                    *dyngreenoffset+ytimestep,(r*matrixdim+col),func);  //-alpha * delx_F

                spatialderiv_stat += DiracEqSpatialDerivative2Deriv(dynindex
                    +(row*matrixdim+r)*dyngreenoffset+ytimestep,(row*matrixdim+r)
                    ,func)*alpha[r*matrixdim+col];                      //dely_F * alpha

                spatialderiv_spec += -alpha[row*matrixdim+r]
                    *DiracEqSpatialDerivativeDeriv(dynindex+(r*matrixdim+col)
                    *dyngreenoffset+ytimestep,(r*matrixdim+col),func,0);//-alpha * delx_rho

                spatialderiv_spec += DiracEqSpatialDerivative2Deriv(dynindex
                    +(row*matrixdim+r)*dyngreenoffset+ytimestep,(row*matrixdim+r)
                    ,func,0)*alpha[r*matrixdim+col];                    //dely_rho * alpha
            }

            //evaluate the value of F(n+1,n+1)
            newval = (1.0/(2.0-deltat*(x0_massterm+y0_massterm)))
                *((2.0+deltat*(x0_massterm+y0_massterm))
                *func->val(tindex+dynindex+l*greensize)
                +2.0*deltat*spatialderiv_stat);

            //assign the value of F(n+1,n+1)
            func->assign(tincindex+dynindex+l*greensize+ytimestep,newval);

            //evaluate the value of rho(n+1,n+1)
            newval = (1.0/(2.0-deltat*(x0_massterm+y0_massterm)))
                *((2.0+deltat*(x0_massterm+y0_massterm))
                *func->specval(tindex+dynindex+l*greensize)
                +2.0*deltat*spatialderiv_spec);

            //assign the value of rho(n+1,n+1)
            func->specassign(tincindex+dynindex+l*greensize+ytimestep,newval);
        }

        spacialindex0+=stepsize; //increase spatial index of first point of greensfunction

    }

}
```

Erklärung der Urheberschaft

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Ort, Datum	Matteo Boser

Bibliography

[A+15] G. Aad et al. Combined Measurement of the Higgs Boson Mass in pp Collisions at $\sqrt{s} = 7$ and 8 TeV with the ATLAS and CMS Experiments. *Phys. Rev. Lett.*, 114:191803, 2015. arXiv:1503.07589v1.

[A+16a] B. P. Abbott et al. Observation of Gravitational Waves from a Binary Black Hole Merger. *Phys. Rev. Lett.*, 116:061102, 2016.

[A+16b] R. Adam et al. Planck 2015 results. I. Overview of products and scientific results. *Astron. Astrophys.*, 594:A1, 2016. arXiv:1502.01582v2.

[A+17] P. Amaro-Seoane et al. Laser Interferometer Space Antenna. 2017. arXiv:1702.00786v3.

[B+13] C. L. Bennett et al. Nine-year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Final Maps and Results. *APJS*, 208:20, 2013. arXiv:1212.5225v3.

[BBS03] J. Berges, S. Borsanyi, and J. Serreau. Thermalization of fermionic quantum fields. *Nucl. Phys.*, B660:51–80, 2003. arXiv:hep-ph/0212404.

[BBW04] J. Berges, S. Borsanyi, and C. Wetterich. Prethermalization. *Phys. Rev. Lett.*, 93:142002, 2004. arXiv:hep-ph/0403234.

[Ber02] J. Berges. Controlled nonperturbative dynamics of quantum fields out-of-equilibrium. *Nucl. Phys.*, A699:847–886, 2002. arXiv:hep-ph/0105311.

[Ber15] J. Berges. Nonequilibrium Quantum Fields: From Cold Atoms to Cosmology. 2015. arXiv:1503.02907.

[BKS17] S. Bruggisser, T. Konstandin, and G. Servant. CP-violation for Electroweak Baryogenesis from Dynamical CKM Matrix. *JCAP*, 1711(11):034, 2017. arXiv:1706.08534.

[C+94] F. Cooper et al. Nonequilibrium quantum fields in the large-N expansion. *Phys. Rev. D*, 50:2848–2869, 1994.

[C+16] C. Caprini et al. Science with the space-based interferometer eLISA. II: Gravitational waves from cosmological phase transitions. *JCAP*, 1604(04):001, 2016. arXiv:1512.06239.

[FJ05] M. Frigo and S. G. Johnson. The Design and Implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005.

[H+10] H. Holden et al. *"Splitting methods for partial differential equations with rough solutions. Analysis and Matlab programs"*. European Mathematical Society Publishing House, 2010.

[HPA14] R. Hammer, W. Pötz, and A. Arnold. A dispersion and norm preserving finite difference scheme with transparent boundary conditions for the Dirac equation in (1+1)D. *J. Comput. Physics*, 256:728–747, 2014.

[HS95] P. Huet and E. Sather. Electroweak baryogenesis and standard model CP violation. *Phys. Rev.*, D51:379–394, 1995. arXiv:hep-ph/9404302.

[K+96] K. Kajantie et al. Is there a hot electroweak phase transition at m(H) larger or equal to m(W)? *Phys. Rev. Lett.*, 77:2887–2890, 1996. arXiv:hep-ph/9605288.

[KB62] L. P. Kadanoff and G. A. Baym. *"Quantum statistical mechanics: Greens's function method in equilibrium and nonequilibrium problems"*. Cambridge University Press, 1962.

[Kel64] L. V. Keldysh. Diagram technique for nonequilibrium processes. *Zh. Eksp. Teor. Fiz.*, 47:1515–1527, 1964. [Sov. Phys. JETP20,1018(1965)].

[Kos15] O. Koskivaara. Exact solutions of a Dirac equation with a varying CP-violating mass profile and coherent quasiparticle approximation. Master's thesis, University of Jyväskylä, 2015.

[LM08] M. Lindner and M. M. Müller. Comparison of Boltzmann kinetics with quantum dynamics for a chiral Yukawa model far from equilibrium. *Phys. Rev. D*, 77:025027, 2008.

[MM94] I. Montvay and G. Münster. *"Quantum Fields on a Lattice"*. Cambridge University Press, 1994.

[MRM12] D. E. Morrissey and M. J. Ramsey-Musolf. Electroweak baryogenesis. *New J. Phys.*, 14:125003, 2012. arXiv:1206.2942.

[P+07] W. H. Press et al. *"Numerical Recipes - The Art of Scientific Computing"*. Cambridge University Press, 3rd edition, 2007.

[Sak91] A. D. Sakharov. Violation of CP invariance, C asymmetry, and baryon asymmetry of the universe. *Soviet Physics Uspekhi*, 34(5):392, 1991.

[Sch61] J. Schwinger. Brownian Motion of a Quantum Oscillator. *Journal of Mathematical Physics*, 2:407–432, 1961.

[Sus77] L. Susskind. Lattice fermions. *Phys. Rev. D*, 16:3031–3039, 1977.

[ZJ00] J. Zinn-Justin. Quantum field theory at finite temperature: An Introduction. 2000. arXiv:hep-ph/0005272.