Nonlinear Models for Langmuir Waves in a Collisionless Plasma

by
Margit Iren Ulriksen
Department of Physics
University of Oslo
Norway

Thesis submitted for the degree
Master of Science

January 2010

Abstract

In this study we have performed analytical and numerical analysis of electron density oscillations, known as Langmuir waves, in collisonless plasmas. Such oscillations are important in both experimental and astrophysical plasmas. We have generalized the standard fluid description to include basic features of thermal- and nonlinear- effects. Through the so-called multiple water-bag model we also attempt to include Landau damping in the fluid model. This is advantageous as Landau damping is a kinetic phenomenon, and generalized fluid models are computationally more efficient than the kinetic alternative. For linear Langmuir waves we obtain a good reproduction of Landau damping by the multiple water-bag version of the fluid model. The damping is, however, strongly dependent on the chosen initial conditions for the electron density oscillations. Nonlinear analysis through the inclusion of ponderomotive forces and a special version of the Zakharov model are not as easily solved in the context of multiple water-bags. We also find that Landau damping will be more difficult to extract from this generalized nonlinear model than for the linear waves.

Parts of this thesis were presented as a poster at the 1^{st} Nordic Meeting in Physics in Lyngby, Denmark, the 16-18 June 2009.

Acknowledgements

I would like to thank Professor Hans L. Pécseli for having given me the opportunity to work with the fascinating subject of waves. For being so thoroughly helpful and available for questions. Not even hospitalization kept him from proofreading my thesis.

I also thank Professor Jan Trulsen for his indispensable advice and for being so generous with his time.

I am grateful to every person who has spent time with me and my work throughout my entire education and schooling. My very first teacher, Bodvar Kvitvær, deserves a special thanks for having triggered my interest in natural sciences so early. I find it hard to believe, but apparently most people learn to read before they learn the multiplication table.

My fellow students and good friends the last five years. For discussions, early mornings and late nights. You know who your are.

My parents, for teaching me how important it is to commit oneself to the common good. Runar and Harald, for being exactly what elder brothers should be, for better and for worse.

Yngve. For critical reading, matlab advice and latex support. But most of all for holding my hand.

Contents

Abstract **iii**

Acknowledgements **v**

1 Introduction **1**

1.1 The fluid plasma model 2

1.2 The kinetic plasma model 2

1.3 Modification of the fluid plasma model 3

1.4 Motivation for the present study 4

1.5 Structure of the thesis 4

2 Linear Langmuir waves **7**

2.1 Langmuir waves; High frequency oscillations 7

2.1.1 Dispersion relation 10

2.2 Ion acoustic waves 11

2.3 The multi-component electron fluid model 12

2.3.1 The two-component electron fluid model 15

2.3.2 The dispersion relation of an N-component electron fluid model 16

2.4 Langmuir wave-envelope oscillations 17

3 Landau damping & the water-bag model **19**

3.1 The Vlasov equation 19

3.2 Kinematic description of Landau damping 20

3.2.1 Linearized Vlasov equation and the kinetic dispersion relation 22

3.2.2 Limitations of our analysis 23

3.3 The water-bag model 24

3.3.1 Multiple water-bag models 26

3.3.2 Oscillation amplitudes of the individual components . . 27

4 Nonlinear Langmuir waves 29
4.1 Nonlinearity . . . 29
4.2 Pendulum-Langmuir analogy . . . 30
4.2.1 Nonlinear pendulum . . . 31
4.3 Langmuir waves in a cold plasma revisited . . . 33
4.4 Parametric nonlinearity . . . 40
4.4.1 Weakly nonlinear Langmuir waves with varying ion density . . . 40
4.5 Origin of nonlinearity . . . 44
4.5.1 The ponderomotive force . . . 44
4.5.2 Ion dynamics; The Zakharov model . . . 47
4.6 Generalization of the Zakharov model . . . 51

5 Numerical methods 55
5.1 The finite difference method . . . 55
5.2 Convergence . . . 57
5.2.1 Consistency . . . 57
5.2.2 Stability . . . 58
5.3 Basic concepts of the numerical solutions . . . 60
5.3.1 Scaling and dimensionless variables . . . 60
5.3.2 Boundary conditions . . . 60
5.4 Linear Langmuir waves . . . 61
5.4.1 The single electron fluid model . . . 61
5.4.2 The multi-component electron fluid model . . . 63
5.4.3 Langmuir wave-envelope oscillations in a single electron component plasma . . . 67
5.4.4 Langmuir wave-envelope oscillations in a multi-electron component plasma . . . 71
5.5 Nonlinear Langmuir waves . . . 77
5.5.1 The electrostatic field in a plasma with an inhomogeneous electron density distribution . . . 77
5.5.2 The nonlinear Schrödinger equation . . . 78

6 Results 85
6.1 The basic fluid model . . . 85
6.2 Langmuir waves in multi-component plasmas . . . 87
6.2.1 Dispersion relations . . . 87
6.2.2 Damping . . . 89
6.3 Langmuir-envelopes in single component plasmas . . . 96
6.4 Langmuir-envelopes in multi-component plasmas . . . 101
6.5 Weakly nonlinear Langmuir waves . . . 107

6.6 Weakly nonlinear Langmuir wave-envelopes 109

7 Discussion and Conclusion 121

7.1 Water-bags 121
7.2 Linear Langmuir wave-envelopes 122
7.3 Weakly nonlinear Langmuir waves 123
7.3.1 Generalization of the Zakharov model 124
7.4 Future perspectives 126

A Stability and accuracy analysis 133

A.1 N coupled Langmuir wave equations 133
A.2 N coupled linear Schrödinger equations 135
A.2.1 One sided forward difference in time 135
A.2.2 Leap-Frog 137
A.3 The nonlinear Schrödinger equation 138
A.3.1 One sided forward difference in time 138
A.3.2 Leap-Frog 140

B Source Code 143

B.1 The single electron component fluid
model 144
B.2 The multi-component electron fluid model 146
B.3 Langmuir
envelopes in single electron component plasmas 150
B.4 Langmuir envelopes in multi electron component plasmas . . . 152
B.5 The nonlinear Schrödinger equation 156

Bibliography 160

Chapter 1

Introduction

A plasma is a fully or partially ionized gas compound of electrons and ions. The ionization makes the plasma electrically conductive and therefore long range electromagnetic interactions will dominate the short range inter-atomic or intermolecular forces among a large number of the plasma particles. Collective behavior in this manner is a distinguishing feature of plasmas [1]. For a wide parameter range plasmas can be considered collisionless.

Another characteristic feature of plasmas is the concept of *quasi neutrality*. A quasi neutral plasma has, on large scales, roughly the same amount of free positive and negative charge carriers as negative. This means that the overall electric charge of the plasma will be zero or very small, although on small timescales local or periodic deviations from the quasi neutrality may occur.

In a simple unmagnetized plasma a local perturbation of the charge distribution from quasi neutrality would cause the plasma to oscillate at what is known as the plasma frequency, ω_{pe}. The distance traveled by thermal particles in one corresponding plasma period is known as the Debye length, λ_{De}. These two parameters are fundamental to the very definition of a plasma. Collective behavior can not be established, nor quasi neutrality, for an ensemble of charged particles unless it is many Debye lengths in size and of such a density that a sphere with the Debye length as radius contains many particles [2]. This means that waves are an inherent property of plasmas.

Physically the Debye length is the maximum length over which plasma particles can screen out electric fields. The average number of particles within a Debye sphere is known as *the plasma parameter*, $N_p \equiv n\lambda_{De}^3$ where n is the number density of particles. For an ensemble of particles with N_p somewhat larger than unity a slight change in the distance r between the particles within the Debye sphere would induce a change in the electric field felt by the particles because of the r^{-2} dependence of the Coulomb force. If, on the other hand, the Debye sphere was almost uniformly filled with particles a small change in particle positions would most probably not influence the total electric field. When $N_p >> 1$ we therefore expect individual inter-particle forces to be negligible and collective behavior dominating. This is why the Debye sphere of a plasma, as already mentioned, by definition must contain many particles. It also justifies the assumed collisionless nature of plasmas.

A large variety of wave modes can be found within plasmas, [1, 2], and there are two main models for describing the wave propagation; the fluid model and the kinetic model. The objective of the present thesis is Langmuir waves, one of the most fundamental plasma wave modes. These waves are caused by deviations in the local electron density from the quasi-neutrality equilibrium. We will address them by the fluid plasma model, but aim to include *Landau damping* [3], a concept known from the kinetic model, in the fluid description. Before discussing this any further we introduce some basic concepts associated with the fluid model and the kinetic model.

1.1 The fluid plasma model

The fluid plasma model describes plasma dynamics by considering the particle density, kinetic temperature and flow velocity of a fluid element. Each fluid element is an average over several plasma particles, which means that only macroscopic properties are considered.

One flaw of the fluid model is its incapability of resolving the wave-particle interaction, known as Landau damping, which is expected for particles with thermal velocity close to the phase velocity of a plasma wave, [2, 4].

1.2 The kinetic plasma model

In contrast to the averaged parameters in the fluid model the kinetic model describes the particle velocity distribution function at each point in the plasma. The kinetic model is therefore a microscopic description of a plasma and can resolve microscopic physical properties that are lost in the macroscopic fluid model.

Collisionless plasma dynamics are in the kinetic model described by the Vlasov equation, [5]. This microscopic description has a downside; whereas the fluid model in the full 3D case operates in a 3D phase space ($\boldsymbol{r}$) the kinetic model operates in a 6D phase space ($\boldsymbol{r}, \boldsymbol{v}$), where $\boldsymbol{r}$ and $\boldsymbol{v}$ have components

$\{x, y, z\}$ and $\{v_x, v_y, v_z\}$. The high dimensionality makes a kinetic description of plasma waves much more cumbersome than the fluid alternative.

1.3 Modification of the fluid plasma model

The basic fluid model for Langmuir waves assumes the ions to be an immobile neutralizing background of positive charge. For a linearized analysis, assuming small wave amplitudes, this model is usually adequate. In this limit we can argue that the Langmuir wave frequencies are so high that the large ion inertia makes the ion motion negligible.

Allowing for larger wave amplitudes, and thereby entering the nonlinear regime, it is often necessary to relax the assumption of immobile ions. So called ponderomotive forces resulting from spatially inhomogeneous wave amplitude distributions give rise to spatial inhomogeneities of the bulk electron density. The resulting charge imbalance will give rise to slowly varying electric fields which consequently sets also the ion component into motion.

However, the fluid model as already mentioned, describing only the bulk plasma properties, ignores Landau damping. The single electron fluid model assumes that the electrons are in local thermodynamic equilibrium and follow a Maxwell-Boltzmann velocity distribution. In many cases this assumption is not correct, and the local electron velocity component deviates from a Maxwellian. In such cases, it is often possible to approximate the velocity distribution by a sum of electron components with different temperatures and densities. Each of these components is then assumed to follow its own set of fluid equations, but the dynamics of each component will be coupled to the other components through the collective electric field.

An initial value problem with several electron components will have a very complicated transient time evolution. This is an alternative to the fully kinetic description, where the electron dynamics is described by the Vlasov-equation. We believe that a multi-electron fluid model will incorporate the essentials of the kinetic model when it comes to the space-time evolution of the electric wave field. A generalization of the fluid model in this way will not be very different from the so called "single water-bag" model introduced by DePackh [6], Hohl, Feix and Bertrand [7, 8, 9]. This model keeps the kinetic aspect of a collisionless plasma albeit it has the same complexity as the fluid model. We will solve the resulting set of coupled partial differential equations numerically by the finite difference method.

The Zakharov model [10] for nonlinear electron plasma waves includes ion dynamics in the fluid model. We will systematically generalize the Zakharov model to include the linear Landau damping by using a multi-electron distribution with N coupled differential equations. The essential nonlinear element of the Zakharov model is the perturbation of the bulk plasma density by the ponderomotive forces mentioned before. This effect is readily included also in a multi-electron fluid model, and a generalization of the basic Zakharov model is straight forward, at least in principle.

Our work is intended to illustrate the concepts of a N electron component plasma, it is therefore sufficient to carry out the analysis in one spatial dimension. This will give some convenient simplifications for the numerical modeling.

1.4 Motivation for the present study

Most physical systems are inherently nonlinear in nature [11]. For example when considering a rather simple physical system like a pendulum suspended in a cord, the linear analysis approximating the pendulum dynamics is only valid for a very small portion of the amplitudes the pendulum can undertake [12].

Plasma waves are important for explaining phenomena in various fields such as ionospheric physics, astrophysics and industrial plasmas [2]. However, also for plasmas linear analysis is a valid approximation only for a small regime of oscillations. A variety of nonlinear effects are observed in plasmas [1, 2, 4], and we can not obtain a full understanding of plasma dynamics without entering the nonlinear regime.

The computer resources required for simulating a plasma by a kinetic description are much more demanding than for a fluid description. Multi-component plasma fluid models including Landau damping can therefore be a useful simplifying approach to the fully kinetic alternative. As an example multi water-bag models have been successful in simulating turbulent transport in nearly collisionless fusion plasmas [13]. These simulations are an important tool in the ongoing research for optimizing the energy confinement time in fusion reactors such as tokamaks.

1.5 Structure of the thesis

The aim of the thesis is to generalize the simple fluid analysis of plasmas to include simplified models for kinetic effects and nonlinear effects.

The most general description of a plasma extending the standard fluid models is based on the Vlasov equation. This is, as mentioned before, an equation in terms of three independent variables, space, time and velocity.

It could be desirable to have a model that includes some of the basic important features, such as the linear Landau damping in simpler terms. The present thesis considers the possibility of using a multi-fluid model, based on multi-waterbag distributions. Here we retain space and time as independent variables, but rather than having one more free variable, we look for means of reproducing Landau damping effects by multiple water-bags. These models will only work for a finite time, but might be sufficient for recovering the main features for several practically relevant cases.

The analytical discussions are supported by numerical studies. The writing and implementations of the numerical programs constitute a large part of the work-load of the present thesis.

- Chapter 2
  For completeness, we first present an outline of simple fluid models of Langmuir waves, as found in textbooks. These models are thereafter generalized to include several electron components with individual temperatures and densities.

- Chapter 3
  Kinematic arguments for Landau damping is given. We thereafter illustrate how the multiple water-bag model relates to the kinetic model for Langmuir waves and therefore is expected to include Landau damping. We also show how our generalization of the simple fluid model to multi-electron component plasmas is an equivalent of the water-bag model.

- Chapter 4
  This chapter concerns nonlinear effects for Langmuir waves. The basic nonlinearity addressed here is the nonlinear frequency shift. This is particularly important because it, together with the wave number derivative of the group velocity, determines the modulational stability/instability of a wave through the so-called Lighthill criterion. We will first study the simple cold plasma model with immobile ions. It will be clear from the analysis that we can only obtain a nonlinear frequency shift if we allow mobile ions. The inclusion of ion dynamics leads to a special case of the Zakharov model discussed earlier. Finally we attempt to combine this nonlinear model with the multiple water-bag set of equations obtained for describing Landau damping in a fluid model.

- Chapter 5
  Here we derive finite difference schemes for solving the linear and nonlinear fluid Langmuir wave models numerically.

- Chapter 6
  We represent our numerical results.
- Chapter 7
  A summary of our work is given. We discuss the final results and make conclusive remarks. Future perspectives are also included.

Chapter 2

Linear Langmuir waves

This chapter is an introduction to the physical mechanisms responsible for Langmuir wave propagation in an one-dimensional collisionless plasma. The partial differential equations describing the waves will be derived from the basic fluid model and we shall consider under which conditions electron and ion dynamics, respectively, are relevant.

2.1 Langmuir waves; High frequency oscillations

For one-dimensional high frequency oscillations in a plasma, we assume only electrons to be set in motion. The ions are, because of their inertia, considered immobile for such high frequency oscillations, and therefore their dynamics need not be considered. However, according to the quasi neutrality assumption, the ion density, $n_0 = const$, globally neutralize the electrons.

We assume the electrons to experience an electric field $\boldsymbol{E}(x,t)$ and a magnetic field $\boldsymbol{B}(x,t)$ in the plasma. In a one-dimensional analysis the right hand side of Faraday's law of induction,

$$\nabla \times \boldsymbol{E} = -\frac{\partial}{\partial t}\boldsymbol{B},$$

does not give any contribution and there will therefore be no induction of electric fields. This is also true in two- and three-dimensional analysis if the magnetic field is either zero or constant. For these cases the electric field is termed electrostatic and it can be found from a scalar electric potential field, $\boldsymbol{\phi}$, as $\boldsymbol{E} = -\nabla\boldsymbol{\phi}$. Poisson's equation describes the spatial derivative of the electrostatic field in one-dimension by means of the ion density, n_0, and the electron density, $n = n(x,t)$, as

$$\frac{\partial}{\partial x}E = -\frac{\partial^2}{\partial x^2}\phi = \frac{e}{\epsilon_0}(n_0 - n) \tag{2.1}$$

where e is the electron charge and ϵ_0 the vacuum permittivity.

We aim to describe collisionless plasmas and therefore assume no creation or loss of particles within the plasma. Hence mass is conserved in the plasma and the electrons must obey the continuity equation

$$\frac{\partial}{\partial t}n + \frac{\partial}{\partial x}nu = 0, \tag{2.2}$$

where $u = u(x, t)$ is the electron drift velocity.

Furthermore, the motion of the electrons will be governed by their thermal pressure, $P(x, t)$, and the electrostatic field, $E(x, t)$. This is summarised in Newton's second law on the form

$$mn\left(\frac{\partial}{\partial t}u + u\frac{\partial}{\partial x}u\right) = -\frac{\partial P}{\partial x} + en\frac{\partial \phi}{\partial x}$$

where m is the electron mass. The term $en\partial\phi/\partial x$ is derived from the Lorentz force, $\boldsymbol{F} = q(\boldsymbol{E} + \boldsymbol{u} \times \boldsymbol{B})$, giving the electrostatic force on the electrons. In a one-dimensional analysis we have no contribution from $\boldsymbol{u} \times \boldsymbol{B}$, and in two- and three-dimensional analysis the result is the same if $\boldsymbol{u} \parallel \boldsymbol{B}$.

We are only interested in plasmas with *plasma parameter* $N_p >> 1$. By definition $N_p \equiv n\lambda_{De}^3 \equiv n^{-\frac{1}{2}}\left(\epsilon_0\kappa T/e^2\right)^{3/2}$, hence in the limit $N_p >> 1$ we are dealing with hot and dilute plasmas. An ordinary hot and dilute gas is prone to follow the ideal gas law [14], $P = n\kappa T$ where κ is the Boltzmann constant and T the electron temperature, it therefore seems likely that the electron pressure of a plasma should follow the same law. If we also assume the pressure fluctuations to be adiabatic the pressure will be described by $P = Cn^\gamma$ and the fluctuations by $\nabla P = \gamma\kappa T\nabla n$, where $\gamma = C_p/C_v$ is the ratio between the specific heats at fixed pressure and fixed volume. For a one-dimensional ideal gas $\gamma = 3$. The electron momentum equation is then rewritten as

$$mn\left(\frac{\partial}{\partial t}u + u\frac{\partial}{\partial x}u\right) = -3\kappa T\frac{\partial}{\partial x}n + en\frac{\partial \phi}{\partial x}. \tag{2.3}$$

Together the equations (2.1), (2.2) and (2.3) constitute the fluid model for Langmuir waves in a one-dimensional collisionless electrostatic plasma.

Our analysis concerns plasmas which are initially in a quasi-neutral equilibrium state where the electron velocity, u_0, and the electrostatic potential, ϕ_0, are, for simplicity, both equal to zero and the electron density, n_0, as the ion density, constant. If the plasma deviates slightly from this equilibrium state, we can rewrite the perturbed physical parameters as $\phi = \phi_0 + \tilde{\phi} = \tilde{\phi}$, $u = u_0 + \tilde{u} = \tilde{u}$, $n = n_0 + \tilde{n}$, where the quantities with a tilde denotes small perturbations. The assumption of all perturbations to be small allows us to linearize the equations (2.1)-(2.3), resulting in:

- The Poisson equation
$$\frac{\partial^2 \tilde{\phi}}{\partial x^2} = \frac{e}{\epsilon_0}\tilde{n}.$$

- The electron continuity equation
$$\frac{\partial}{\partial t}\tilde{n} + n_0 \frac{\partial}{\partial x}\tilde{u} = 0.$$

- The electron momentum equation
$$n_0 \frac{\partial}{\partial t}\tilde{u} = -\frac{3\kappa T}{m}\frac{\partial}{\partial x}\tilde{n} + \frac{e n_0}{m}\frac{\partial \tilde{\phi}}{\partial x}.$$

When we assumed the electron pressure to follow the ideal gas law we presupposed the electrons to be Maxwell-Boltzmann distributed. The average thermal velocity of the electrons must therefore be given by $u_{th} \equiv \sqrt{\kappa T/m}$, [14]. Taking the partial time derivative of the linearized continuity equation and introducing the thermal velocity, u_{th}, and the plasma frequency, $\omega_{pe} \equiv \sqrt{e^2 n/\epsilon_0 m}$, we combine the above equations into

$$\frac{\partial^2}{\partial t^2}\tilde{n} - 3u_{th}^2 \frac{\partial^2}{\partial x^2}\tilde{n} + \omega_{pe}^2 \tilde{n} = 0. \tag{2.4}$$

This is a partial differential equation describing the propagation of Langmuir waves in an one-dimensional collisionless plasma.

2.1.1 Dispersion relation

Introducing the Fourier transform of the electron density perturbations $\tilde{n}(x,t)$ in time and space by $\tilde{n}(x,t) = \iint \tilde{n}(k,\omega)e^{-i(\omega t - kx)}d\omega dk$ in (2.4) we obtain

$$\omega^2 = \omega_{pe}^2 + 3u_{th}^2 k^2. \tag{2.5}$$

This is the dispersion relation for high frequency electrostatic waves in a one-dimensional plasma. It is visualized in Figure 2.1.

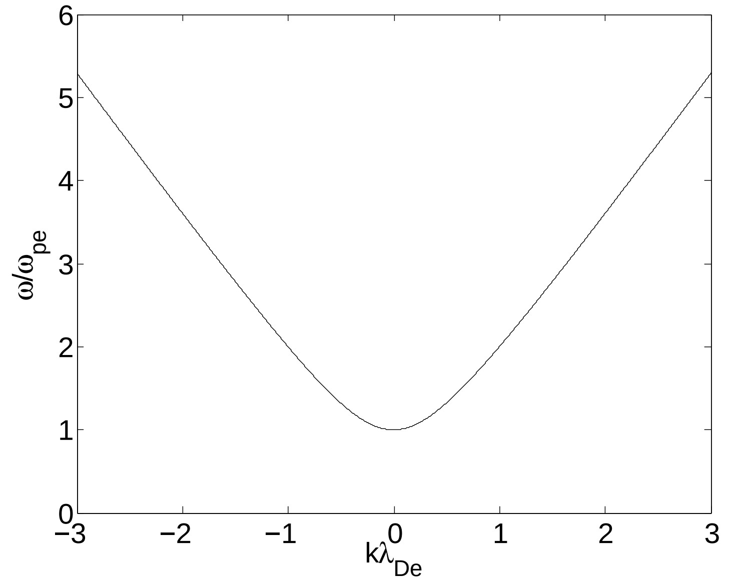


Figure 2.1: The dispersion relation, $\omega(k)/\omega_{pe}$, for high frequency electrostatic waves in a one-dimensional plasma.

In case of a cold plasma, by definition with electron temperature, T, equal to zero, the dispersion relation (2.5) reduces to

$$\omega^2 = \omega_{pe}^2.$$

Thus the plasma frequency is, as mentioned in Chapter 1, the frequency of the non-dispersive rapid electron density oscillations in a cold unmagnetized plasma. We know the electrons thermal velocity, u_{th}, and the distance they travel in one plasma period is found to be the Debye length $\lambda_{De} \equiv u_{th}/\omega_{pe} \equiv \sqrt{\epsilon_0 \kappa T/e^2 n}$.

2.2 Ion acoustic waves

Although we have concluded that the large inertia of ions prevents them from being set in motion by high frequency oscillations, we must assume them to respond to low frequency oscillations. The low frequency ion dynamics are, by the same arguments as for electrons in the high frequency case, described by a continuity equation and a momentum equation. The continuity and momentum equations are respectively

$$\frac{\partial}{\partial t}n_i + \frac{\partial}{\partial x}n_i u_i = 0$$

and

$$Mn_i\left(\frac{\partial}{\partial t}u_i + u_i\frac{\partial}{\partial x}u_i\right) = -\gamma\kappa T_i\nabla n_i + en\frac{\partial\phi}{\partial x},$$

where n_i,u_i, M and T_i represents the ion density, velocity, mass and temperature.

The ion density is changing slowly on the ion time-scale, but electrons adjust almost instantaneously to this change. The electron thermal velocity is thus much larger than the phase velocity of the ion density oscillations and thereby allow the electrons to rapidly equalize their temperature in the entire plasma. This makes it reasonable to assume the electrons are isothermal on the time scale of the ion density oscillations. The electron dynamics will be described by (2.2) and (2.3), but the adiabatic index, γ, in equation (2.3) must, by the isothermal assumption, be set to unity. However, the left hand side of the momentum equation (2.3) is negligible when compared to the right hand side because the low frequency electron inertia effects are much smaller than the thermal effects. In consequence the momentum equation yields an isothermal Boltzmann distribution of the electrons, $n_e = n_0 e^{e\phi/\kappa T_e}$, where n_e is the electron density and T_e the electron temperature. This relation can be inserted directly into the Poisson equation (2.1).

For the Langmuir waves we assumed the plasma to be initially slightly perturbed from quasi neutrality and thereafter linearized the electron fluid equations. We do the same for the low frequency ion dynamics, and after linearization the ion continuity and momentum equations can be combined with the Poisson equation to form

$$\left[\frac{\partial^2}{\partial t^2}\left(1 - \frac{\epsilon_0\kappa T_e}{e^2 n_0}\frac{\partial^2}{\partial x^2}\right) - \frac{\gamma\kappa T_i}{M}\frac{\partial^2}{\partial x^2}\left(1 - \frac{\epsilon_0\kappa T_e}{e^2 n_0}\frac{\partial^2}{\partial x^2}\right) - \frac{\kappa T_e}{M}\frac{\partial^2}{\partial x^2}\right]\frac{e\tilde{\phi}}{\kappa T_e} = 0.$$

This is a partial differential equation describing what is known as ion acoustic waves. $\tilde{\phi}$ is a small deviation from the electrostatic potential, $\phi = 0$, at quasi neutrality. By Fourier transformation and some algebra, [15], the equation can be solved for the dispersion relation, ω, as

$$\omega^2 = \frac{\kappa T_e}{M} \frac{k^2}{1 + (k\lambda_{De})^2} + \frac{\gamma \kappa T_i}{M} k^2.$$

Figure (2.2) is a graphic illustration of this dispersion relation. To obtain the quasi-neutral limit we let $(k\lambda_{De})^2 \to 0$.

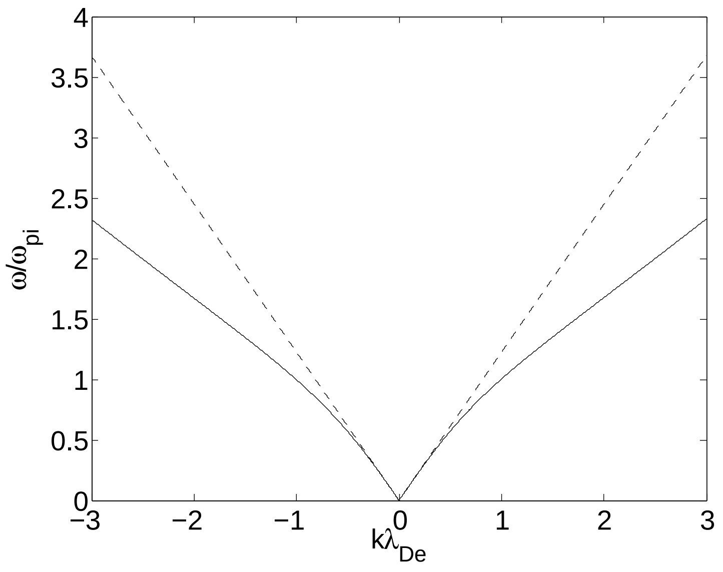


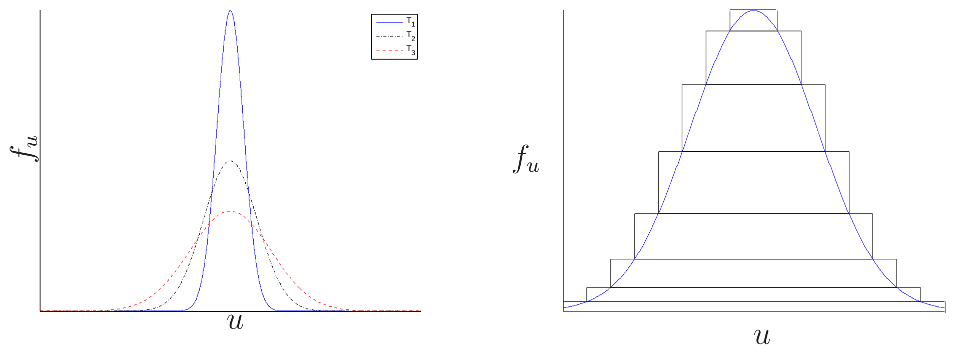
Figure 2.2: The dispersion relation, ω/ω_{pi}, for ion acoustic waves in an electrostatic plasma. Here $\omega_{pi} \equiv \sqrt{e^2 n_0/\epsilon_0 M}$ is the ion plasma frequency, we have chosen $T_e/T_i = 2$ and the dashed line is the quasi-neutral limit.

2.3 The multi-component electron fluid model

Physical plasmas are often not completely Maxwell-Boltzmann distributed, that is they are not in exact thermodynamic equilibrium. This is for example the case in the ionosphere, as observed, among others, by Khotyaintsev et al. [16]. We will approximate such plasmas by a sum of weighted plasma components with different temperatures and densities. The different components will all have their own set of fluid equations, coupled together through the collective electric fields they experience. We return to high frequency oscillations, and all the assumptions from Section (2.1) are valid, i.e. only electron dynamics are considered.

The Maxwell-Boltzmann distribution, $f_u(u) = \frac{n_0}{\sqrt{2\pi\sigma^2}} e^{-u^2/2\sigma^2}$, for the electron velocity, u, in a plasma, takes the form of a normal distribution with variance $\sigma^2 = \kappa T/m$, as illustrated for three different temperatures in Figure 2.3(a). In Figure 2.3(b) we have divided the Maxwellian into blocks, each block corresponding to an electron component. The length of the bottom block is set to be two times 3σ and if we set $r = 1, 2, ..., N$ and $u_r = r\frac{3\sigma}{N}$ each electron component will be defined by the block whose length is set by the velocity interval $[-u_r\,,\,u_r]$. The area of each block is a measure of the fraction of the electron density, $n_{r,0}$, within the block to the overall electron density, n_0, of the plasma. We therefore find the equilibrium density of each electron component as

$$n_{r,0} = n_0 \frac{2u_r}{\sqrt{2\pi\sigma^2}} \left[e^{-\frac{1}{2}\frac{u_{r-1}^2}{\sigma^2}} - e^{-\frac{1}{2}\frac{u_r^2}{\sigma^2}} \right].$$



(a) The Maxwell-Boltzmann electron velocity distribution for three different temperatures. We have $T_1 < T_2 < T_3$.

(b) The Maxwell-Boltzmann electron velocity distribution at temperature T divided into $N = 8$ blocks corresponding to N electron components.

Figure 2.3: Maxwell-Boltzmann distributions for the electron velocity in a plasma.

Each block has its own variance and in practice this is how we introduce the individual temperatures for each electron component. The thermal velocity of each component is given as the root mean square value of the components velocity:

$$u_{th,r}^2 = \frac{\int_0^{u_r} u^2 f_u(u_r)\,du}{\int_0^{u_r} f_u(u_r)\,du} = \frac{1}{3}u_r^2.$$

Each of the N electron components fulfills a continuity equation

$$\frac{\partial}{\partial t}n_r + \frac{\partial}{\partial x}n_r u_r = 0$$

and a momentum equation

$$mn_r\left(\frac{\partial}{\partial t}u_r + u_r\frac{\partial}{\partial x}u_r\right) = -\frac{\partial P_r}{\partial x} + en_r\frac{\partial \phi}{\partial x}$$

where, as in Section 2.1, $\phi(x,t)$ is the electrostatic potential, m is the electron mass, e the electron charge and $P(x,t)$ the electron pressure. Since we are considering electrostatic oscillations the Poisson equation must apply to all the fluid equations:

$$\frac{\partial^2\phi}{\partial x^2} = \frac{e}{\epsilon_0}(n_1 + n_2 + \cdots + n_N - n_0),$$

where n_0 is the ion density.

As before we assume the plasma to initially be in an equilibrium where the electron velocities and the electrostatic potential are equal to zero and the electron density, as the ion density, is constant. If the electron components experience small perturbations from this equilibrium their fluid equations can be linearized in the same manner as in the previous sections. The continuity and momentum equations for each respective electron component can then be combined into one equation and all the resulting equations will be coupled through the Poisson equation, forming the set of equations

$$\frac{\partial^2}{\partial t^2}\tilde{n}_1 - 3u_{th,1}^2\frac{\partial^2}{\partial x^2}\tilde{n}_1 + \omega_{pe,1}^2(\tilde{n}_1 + \tilde{n}_2 + \cdots + \tilde{n}_N) = 0 \tag{2.6}$$

$$\frac{\partial^2}{\partial t^2}\tilde{n}_2 - 3u_{th,2}^2\frac{\partial^2}{\partial x^2}\tilde{n}_2 + \omega_{pe,2}^2(\tilde{n}_1 + \tilde{n}_2 + \cdots + \tilde{n}_N) = 0 \tag{2.7}$$

$$\vdots \tag{2.8}$$

$$\frac{\partial^2}{\partial t^2}\tilde{n}_N - 3u_{th,N}^2\frac{\partial^2}{\partial x^2}\tilde{n}_N + \omega_{pe,N}^2(\tilde{n}_1 + \tilde{n}_2 + \cdots + \tilde{n}_N) = 0, \tag{2.9}$$

where quantities with a tilde over are the small perturbations from quasi-neutrality, and the thermal velocities, $u_{th,r}^2 = \kappa T_r/m$, and the plasma frequencies, $\omega_{pe,r} = \sqrt{e^2 n_r/\epsilon_0 m}$, for each electron component are introduced. This set of coupled equations can be written more compactly as

$$\frac{\partial^2}{\partial t^2}\tilde{n}_r - 3u_{th,r}^2 \frac{\partial^2}{\partial x^2}\tilde{n}_r + \omega_{pe,0,r}^2 \sum_{y=1}^{N} \tilde{n}_y = 0 \quad \text{for } r = 1, 2, ..., N, \tag{2.10}$$

where the thermal velocity of each component is given as $u_{th,r} = r\frac{\sqrt{3}\sigma}{N}$.

2.3.1 The two-component electron fluid model

As an example of a multi electron component problem we will derive the dispersion relation, $\omega(k)$, for Langmuir waves in a plasma with two electron components.

We start by Fourier transforming (2.6) and (2.7) into

$$-\omega^2\tilde{n}_1 + 3u_{th,1}^2 k^2 \tilde{n}_1 + \omega_{pe,1}^2(\tilde{n}_1 + \tilde{n}_2) = 0 \tag{2.11}$$
$$-\omega^2\tilde{n}_2 + 3u_{th,2}^2 k^2 \tilde{n}_2 + \omega_{pe,2}^2(\tilde{n}_1 + \tilde{n}_2) = 0. \tag{2.12}$$

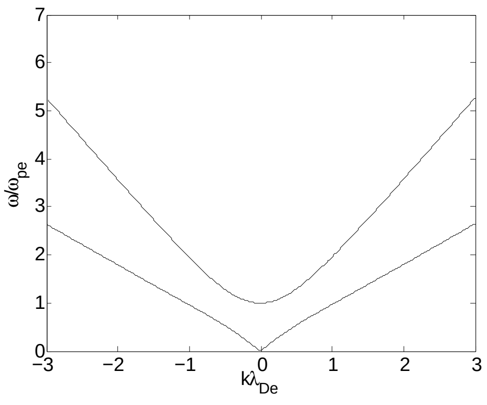
Solving (2.11) for $\tilde{n}_2$ gives

$$\tilde{n}_2 = \left(\frac{\omega^2}{\omega_{pe,1}^2} - \frac{3u_{th,1}^2 k^2}{\omega_{pe,1}^2} - 1\right)\tilde{n}_1$$

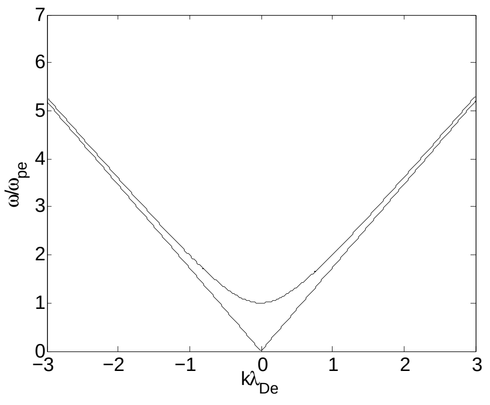
and after inserting it in (2.12) and recognizing $(\omega_{pe,1}^2 + \omega_{pe,2}^2) = \omega_{pe}^2$ we find the dispersion relation

$$\omega^4 - \omega^2(3u_{th,1}^2 k^2 + 3u_{th,2}^2 k^2 + \omega_{pe}^2) + 9u_{th,2}^2 u_{th,1}^2 k^4 + 3u_{th,2}^2 k^2 \omega_{pe,1}^2 + 3u_{th,1}^2 k^2 \omega_{pe,2}^2 = 0. \tag{2.13}$$

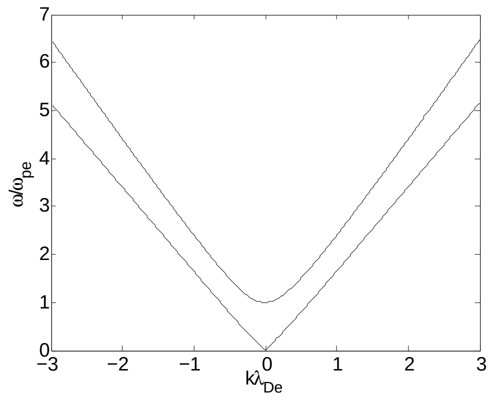
This dispersion relation is shown graphically in Figure 2.4. It is different from the dispersion relation (2.5) for the single electron component description because of the presence of, when compared to Figure (2.2), an acoustic like branch. The result has been verified experimentally [17].



(a) $\frac{u_{th,2}^2}{u_{th,1}^2} = 0.25$ and $\frac{n_2}{n_1} = 0.25$



(b) $\frac{u_{th,2}^2}{u_{th,1}^2} = 0.25$ and $\frac{n_2}{n_1} = 1.00$



(c) $\frac{u_{th,2}^2}{u_{th,1}^2} = 1.5$ and $\frac{n_2}{n_1} = 1.5$

Figure 2.4: The normalized dispersion relation for Langmuir waves in plasmas with two electron components. Shown here for three different set of parameter-values.

2.3.2 The dispersion relation of an N-component electron fluid model

Returning to the N component model, (2.10), a Fourier transformation yields the dispersion relation

$$D(\omega, k) = 1 - \sum_{r=1}^{N} \omega_{pe,r}^2 \frac{1}{\omega^2 - 3u_{th,r}^2 k^2} = 0. \tag{2.14}$$

For any number of components N this equation will have $2N$ solutions $\omega_r = \pm\omega_r(k)$ for $r = 1, ..., N$. As an example we found this relation to be solved for two different values of $\pm\omega$ for each choice of wave-number k when $N = 2$, and similarly we found $\pm\omega$ to be single valued for any k in the single electron component description first presented in this chapter. We thus expect to find a dispersion relation with N different branches for an N component plasma.

It might be interesting to note that we have a simple limit in $k = 0$ where $D(\omega, 0) = 0$ gives $\sum_{r=1}^{N} \omega_{pe,r}^2 = \omega_{pe}^2$, irrespective of the number of electron components. For small k it is then logical to assume a dispersion relation on the form $\omega = \omega_{pe} + 3u_{th}^2 k^2$, similar to the one we found for the single component model. Here u_{th} is an estimate for the total thermal velocity of the electrons.

In the small wave number limit we then have

$$\sum_{r=1}^{N} \frac{\omega_{pe,r}^2}{\omega_{pe}^2 + 3k^2(u_{th}^2 - u_{th,r}^2)} = 1$$

and by the approximation $\frac{1}{1+x} \approx 1 - x$ and some simple algebra we find

$$u_{th}^2 = \sum_{r=1}^{N} \frac{\omega_{pe,r}}{\omega_{pe}} u_{th,r}^2 = \sum_{r=1}^{N} \frac{n_r}{n_0} u_{th,r}^2. \tag{2.15}$$

2.4 Langmuir wave-envelope oscillations

Electron density oscillations in a single electron component plasma can be described as

$$n = n_s(t, x) e^{-i\omega_{pe} t} + \mathit{complex\,conjugate} \tag{2.16}$$

where n_s is a slowly varying amplitude and $e^{-i\omega_{pe} t}$ a rapidly varying phase factor. We shall use this to obtain a partial differential equation describing the evolution of the amplitude part only. There are controversies as to whether this separation in phase factor and amplitude can be obtained at all,

[18], but we will ignore this here and assume this separation to be a priori known.

From (2.16) we obtain

$$\frac{\partial^2}{\partial t^2}n = -\omega_{pe}^2 n_s e^{-i\omega_{pe}t} - 2i\omega_{pe}e^{-i\omega_{pe}t}\frac{\partial}{\partial t}n_s + e^{-i\omega_{pe}t}\frac{\partial^2}{\partial t^2}n_s. \tag{2.17}$$

Assuming $\frac{\partial}{\partial t}n_s << \omega_{pe}n_s$ we let n_s account for a much slower part of the density oscillation than $e^{-i\omega_{pe}t}$ and the last term in the above equation is then so small compared to the others that it can be neglected.

Inserting (2.17) with the last term omitted in the Langmuir wave equation (2.4) we obtain

$$i\frac{\partial}{\partial t}n_s + \frac{3u_{th}^2}{2\omega_{pe}}\frac{\partial^2}{\partial x^2}n_s = 0, \tag{2.18}$$

which is a linear Schrödinger equation describing the slowly varying amplitude part of the electron density oscillation. We expect the time evolution of the slowly varying amplitude to be the wave-envelope of the electron density perturbation described by the full wave equation (2.4).

For the multi-component electron fluid model we can also write the electron density oscillations as a product of a slowly varying amplitude and a rapidly varying phase factor. We do this by assuming each density component to have the total plasma frequency as phase factor, but individual amplitudes. The oscillation for each of the N components is then written as $n_r = n_{r,s}e^{-i\omega_{pe}t} + cc.$ for $r = 1, 2, ..., N$ and following the same arguments as for the single component model we have

$$\frac{\partial^2}{\partial t^2}n \approx -\omega_{pe}^2 n_{r,s}e^{-i\omega_{pe}t} - 2i\omega_{pe}e^{-i\omega_{pe}t}\frac{\partial}{\partial t}n_{r,s}.$$

The set of coupled equations describing the electron density oscillations now become

$$i\frac{\partial}{\partial t}n_{r,s} + \frac{3u_{th,r}^2}{2\omega_{pe}}\frac{\partial^2}{\partial x^2}n_{r,s} + \frac{\omega_{pe}}{2}n_{r,s} - \frac{\omega_{pe,r}^2}{2\omega_{pe}}\sum_{r=1}^{N}n_{r,s} = 0 \tag{2.19}$$

for $r = 1, 2, ..., N$, where $u_{th,r}$ and $\omega_{pe,r}$ are defined in section 2.3. As for the single electron component model we expect these N linear Schrödinger equations to describe the wave-envelope of the full electron density oscillations in the N component plasma.

Chapter 3

Landau damping & the water-bag model

It is necessary to introduce the kinetic model in order to understand Landau damping. The water-bag model will illustrate why a multi electron component plasma fluid model can be expected to include Landau damping in certain limits. In many ways this model is the bridge between the fluid and kinetic descriptions of a collisionless plasma.

3.1 The Vlasov equation

Plasmas are rarely found to be in exact thermodynamic equilibrium, this is especially the case in a collisionless plasma as it does not experience any of the particle collisions that normally tend to bring a fluid into local thermal equilibrium. A kinetic description is therefore often necessary for collisionless plasmas. Such a description includes a model for the space-time evolution of the particle velocity distribution which is generally different from a Maxwellian.

Conventionally a one-dimensional dynamic system is described in terms of its particle distribution function in phase space, $f(x, u, t)$, where x is the spatial coordinate and u is the velocity space coordinate. In the collisionless electrostatic case the dynamics of this particle distribution function is determined by the Vlasov equation:

$$\frac{\partial}{\partial t} f + u \frac{\partial}{\partial x} f + \frac{qE}{m} \frac{\partial}{\partial u} f = 0, \tag{3.1}$$

where m is the mass of the plasma particles, q their electric charge and the electric field $E(x, t)$ can be determined from the Poisson equation. Magnetic forces are ignored in a one dimensional model. The Vlasov equation is a continuity equation in phase space, based on the principle of conservation of probability mass. Liouville's theorem applies to relation (3.1) and the phase-space dynamics is therefore an incompressible laminar flow.

In the literature the Vlasov equation is sometimes referred to as the *collisionless kinetic equation* or the *collisionless Boltzmann equation*.

The fluid model can be derived from the Vlasov equation by averaging the particle density and velocity over the distribution function $f(x, u, t)$, making the fluid description a set of truncated moment equations. This is the reason why the fluid description contains less detailed information than the kinetic model.

3.2 Kinematic description of Landau damping

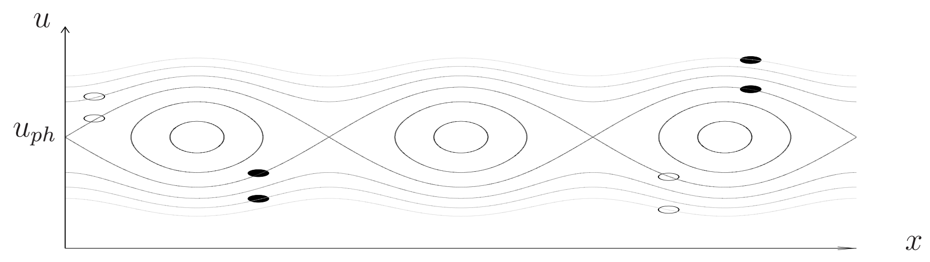
Landau damping is (as the name suggests) a damping mechanism experienced by waves in collisionless plasmas. It was discovered theoretically by Landau in 1946, [3], when he solved the linearized version of the Vlasov equation (3.1) as an initial value problem by introducing an electron density perturbation at time $t = 0$ and assuming the electron distribution function at this instant to follow the Maxwell-Boltzmann distribution. The existence of Landau damping has been verified experimentally [19, 20]. The mathematical derivation is lengthy, requiring the use of contour integration, and is omitted here. We shall instead focus on the physical arguments for Landau damping. Nevertheless, we note that the mathematical description of Landau damping is essentially a dispersion relation with a negative imaginary part which formally introduces the damping.

We consider a collisionless plasma exposed to Langmuir waves with phase velocity u_{ph}. If we let the rest frame of the wave be our frame of reference we will perceive the wave as a stationary electrostatic potential $\phi = \phi_0 \cos kx$ with a characteristic wavelength $\lambda = \frac{2\pi}{k}$, ϕ_0 is here the initial electrostatic amplitude and we have assumed a plane wave.

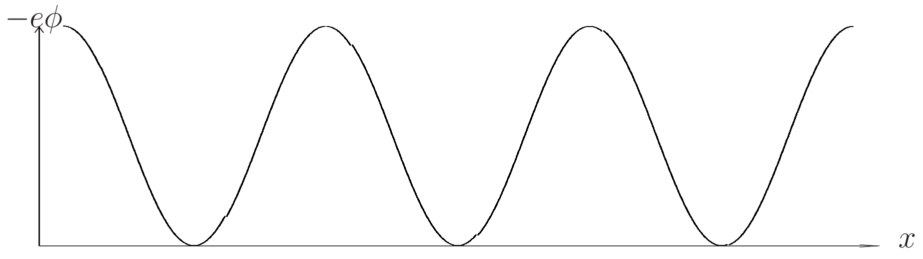
An electron with electric charge $-e$, mass m and velocity u finding itself within the wave, will have total energy

$$W = \frac{1}{2}mu^2 - e\phi_0(\cos kx - 1),$$

which is the sum of its kinetic energy and electric potential energy. The Hamiltonian, H, for this system is a *constant of motion* [21], i.e. $H = W$, and therefore the phase-space trajectories for the particles will be as given in Figure 3.1(a). Figure 3.1(b) is an illustration of the electrostatic potential energy for electrons. Electrons with $W < 0$ do not have enough kinetic energy to climb out of the potential trough of the wave and they therefore have closed trajectories in phase space - they are trapped. Electrons with $W > 0$ is on the other hand free.



(a) Phase-space trajectories.



(b) Electrostatic potential energy for electrons.

Figure 3.1: Phase-space trajectories and electrostatic potential energy for electrons in a collisionless electrostatic one-dimensional plasma as seen in a frame of reference with velocity u_{ph}. Adapted from Chen [22].

The lower half of Figure 3.1(a) corresponds to phase-space trajectories for electrons with velocities lower than the Langmuir wave phase velocity, u_{ph}, whereas the upper-half is for electrons with velocities higher than u_{ph}. Because we are in the rest frame of the wave we will see electrons with velocity $u > u_{ph}$ drifting forwards and electrons with $u < u_{ph}$ drifting backwards, i.e. all electrons in the lower half of Figure 3.1(a) are moving to the left and in the upper half they are going right. Some electrons are indicated by circles in the figure, open circles represent electrons that are gaining kinetic energy and hence losing electrostatic potential energy and closed circles are electrons losing kinetic energy and gaining electrostatic potential energy.

As apparent from Figure 3.2, where the shaded region is the velocity

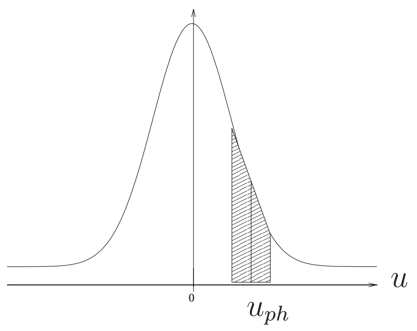


Figure 3.2: The Maxwell-Boltzmann distribution for electrons. Here u_{ph} represents the phase velocity of a Langmuir wave.

spectrum represented by the phase-space trajectories in figure 3.1(a), the presumption of a Maxwellian velocity distribution for the electrons means that there are more particles gaining kinetic energy than losing kinetic energy. The plasma must be energy conserving and we can not explain this antisymmetric energy loss/gain unless the wave is losing energy to the electrons. In consequence the wave must be damped.

Note that the Landau damping is here explained by the detailed structure of the electron velocity distribution. The fluid description normally only consider an average over the same velocity distribution and therefore does not seize the mechanism responsible for Landau damping.

3.2.1 Linearized Vlasov equation and the kinetic dispersion relation

In the quasi neutral equilibrium state the plasma is spatially uniform, the initial electron distribution function, F_0, can therefore be considered a function of the electron velocity, u, only. We assume this equilibrium to be perturbed by small amplitude Langmuir waves. For small amplitude waves the oscillating electrostatic field, E, is also small and leads to a small perturbation $f(x, u, t)$ of the initial electron distribution $F_0(u)$. Linearizing the Vlasov equation we thus obtain

$$\frac{\partial}{\partial t} f + u \frac{\partial}{\partial x} f + \frac{qE}{m} \frac{\partial}{\partial u} F_0 = 0. \tag{3.2}$$

After Fourier transforms of (3.2) with respect to the spatial and temporal variables we find a linear dispersion relation in the form

$$D(\omega, k) = 1 - \left(\frac{\omega_{pe}}{k}\right)^2 \fint_{-\infty}^{\infty} \frac{F_0'(u)}{u - \omega/k} du = 0 \tag{3.3}$$

where the notation $\fint$ is an abbreviation for the velocity integration path defined by Fig. 3.3. This integration path remedies the singularity at $u = \omega/k$ for real ω and k, which appears in case $F_0'(u = \omega/k) \neq 0$. For the case where $F_0'(u = \omega/k) = 0$ no singularity appears, and the dispersion relation becomes simpler.

For the special case where $F_0(u)$ is a Maxwellian, we can express $D(\omega, k)$ in terms of the so–called Z–function [23].

For the case relevant for the present study, where F_0 is the Maxwell-Boltzmann distribution, we have $F_0(u)$ to be a even function, so that $F_0'(u)$ is odd. By this we can rewrite (3.3) as

$$D(\omega, k) = 1 - \left(\frac{\omega_{pe}}{k}\right)^2 \fint_{0}^{\infty} \frac{2uF_0'(u)}{u^2 - (\omega/k)^2} du = 0 \tag{3.4}$$

where now the integration limits are 0 to ∞.

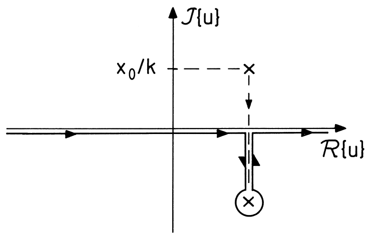


Figure 3.3: Integration path for (3.3) in the complex u–plane. The figure assumes $k > 0$. For $k < 0$, the singularity is *below* the real u–axis, and the figure has to be mirrored with respect to the horizontal axis. In the present figure we introduced the abbreviation $x_0 = \omega_R + i\omega_I$ for a complex frequency used in the inverse Laplace transform in the original work of Landau.

3.2.2 Limitations of our analysis

Near the potential trough of the Langmuir wave, assuming a small amplitude, ϕ_0, for the initial electrostatic perturbation, the equation of motion for an electron is approximated as

$$m\frac{d^2}{dt^2}x = -e\phi_0 k^2 x,$$

which we recognize as the harmonic oscillator, and therefore we know that in this limit the electron motion is characterized by the frequency

$$\omega_B = \sqrt{\frac{e}{m}\phi_0 k^2}.$$

This frequency is the bounce frequency for the trapped electrons. Within a bounce period electrons with velocity slower than the phase velocity of the wave will be accelerated gaining energy from the wave whilst electrons with velocity faster than the phase velocity are decelerated losing energy to the wave.

Our analysis of Landau damping has been in the linear regime. We may only expect to observe linear Landau damping if the bounce period, $\frac{\omega_B}{2\pi}$, of the electrons is much greater than time interval over which we observe the Langmuir wave. This is simply because we in the linear analysis assume the particle distribution function, $f(x,u,t)$, to be undeflected by the wave-particle interactions, but after one bounce period the particle orbits will be significantly altered [24, 25].

In other words the linear analysis is unsuccessful for large ϕ_0 and long timescales, and the full implications of Landau damping can not be fully understood without the inclusion of nonlinear effects.

3.3 The water-bag model

We will now consider a one-dimensional electrostatic collisionless plasma were the initial electron phase space trajectories are as shown in Figure 3.4. Between the two curves v_+ and v_- the electron distribution function is given as $f(x,v,0) = A$ where A is a constant, and for electron velocities, v, larger than v_+ or smaller than v_- the distribution function is equal to zero. This is the so-called water-bag model. Since the water-bag is just a very special choice of distribution, the general statements concerning phase-space dynamics remain true also here, the motion is incompressible in particular. However, the velocity distribution is piecewise constant, i.e. $F_0\prime(u) = 0$ almost everywhere in (3.3) and (3.4) (except exactly at the boundary) and we expect that Landau damping will be absent.

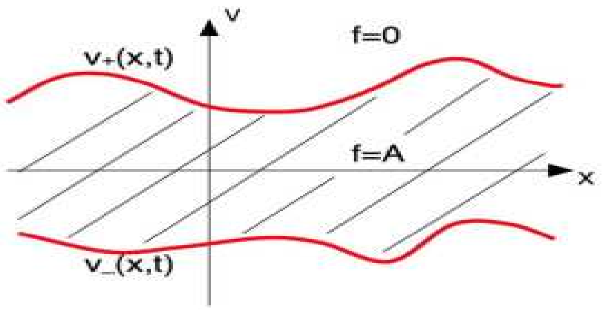


Figure 3.4: The water bag model in phase space. From Morel et al. [13].

In phase space any particle found at one instant at a certain contour will be found at all later times at the same contour [21], so the initial limiting contours in our problem remains limiting for all later times. For that reason it is sufficient to evaluate the two functions $v_+(x,t)$ and $v_-(x,t)$ in order to describe the entire problem. We find the time evolution of v_+ and v_- by substituting the electron distribution function into the Vlasov equation (3.1), [26], yielding the result

$$\frac{\partial}{\partial t}v_\pm + v_\pm\frac{\partial}{\partial x}v_\pm = -\frac{e}{m}E(x,t), \tag{3.5}$$

where $-e$ is the electron charge, m the electron mass and E the electrostatic field described by Poisson's equation,

$$\frac{\partial}{\partial x}E(x,t) = -\frac{e}{\epsilon_0}(n-n_0),$$

where $n = A(v_+ - v_-)$ is the electron density and n_0 the constant ion density.

The average fluid velocity of the plasma is found as $u(x,t) = \frac{1}{2}(v_+ + v_-)$. By subtracting and adding the equations describing the upper contour, v_+, and the lower contour, v_-, we obtain a closed set of equations,

$$\begin{aligned}
&\frac{\partial}{\partial t}n + \frac{\partial}{\partial x}nu = 0\\
&\frac{\partial}{\partial t}u + u\frac{\partial}{\partial x}u = -\frac{1}{mn}\frac{\partial}{\partial x}P + \frac{e}{m}E\\
&P = \frac{mn^3}{12A^2},
\end{aligned}$$

which we recognize as the fluid description, (2.1)-(2.3), introduced in section 2.1. If we let v_+ and v_- be subject to small harmonic perturbations, $w_\pm$, from a homogeneous equilibrium value, $\pm a$, and linearize the equations (3.5) around this equilibrium we obtain the dispersion relation $\omega^2 = \omega_{pe}^2 + k^2 a^2$ [13]. For an adequate choice of the water bag parameters a and A the dispersion relation (2.5) for Langmuir waves in a collisionless electrostatic plasma can then be exactly reproduced, justifying the assertion of the water-bag model as the bridge between the kinetic and fluid descriptions of a plasma.

Even if we now see a link between the fluid and kinetic descriptions the Landau damping is still not reproduced in the single water bag model. This is because the phase velocity, $u_{ph} = \frac{\omega}{k} = \sqrt{a^2 + \omega_{pe}^2/k^2}$, of the perturbations is obviously larger than a and consequently lies in an area of phase space where the electron velocity distribution is equal to zero, i.e. there are no electrons there to interact with the wave. To recover the Landau damping the water bag model has to be generalised into a multiple water bag model.

3.3.1 Multiple water-bag models

Figure 3.5 is an illustration of a three-bag water-bag model where the electron distribution function has three different constant values F_1, F_2 and F_3. On the right hand side of the figure an approximate Maxwell-Boltzmann electron velocity distribution function for the model is shown. We can immediately recognize the box approximation to the Maxwellian as similar to our discussion of N component electron fluid plasmas in Chapter 2.

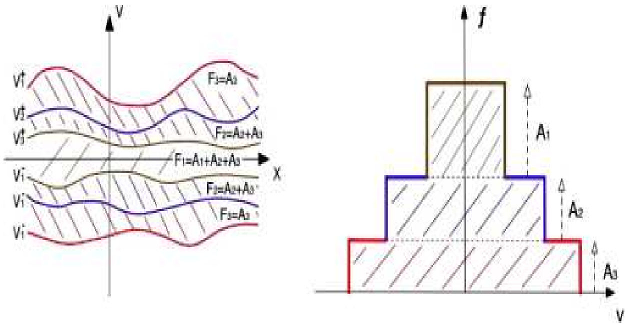


Figure 3.5: Phase space trajectories for a three-component water-bag model and the corresponding Maxwell-Boltzmann distribution. From Morel et al. [13].

The electron density of each water-bag in Figure 3.5 is given by $n_r = A_r(v_r^+ - v_r^-)$ and the average fluid velocity as $u_r = \frac{1}{2}(v_r^+ + v_r^-)$ for $r = 1, 2, 3$. In the same way as we found a set of fluid equations for the two contours of the one-bag model, we find three different sets of fluid equations for this model. Poisson's equation,

$$\frac{\partial}{\partial x}E = \frac{e}{\epsilon_0}\left(\sum_{r=1}^{3} n_r - n_0\right),$$

couples the three different set of equations together. For any number, N, of water-bags the result will be the same, and once again the water-bag model is parallel to the multi-component fluid model.

For the purpose of perturbing each of the contours in a multi water-bag model in the same way as we did for the one-bag case we set $v_r^{\pm}(x,t) = \pm a_r + w_r^{\pm}(x,t)$ for $r = 1, 2, ..., N$, where a_r is the constant equilibrium value of each contour and w_r a small harmonic perturbation from it. Subsequently, after linearizing the equations describing the contour evolution, the dispersion relation

$$D(\omega, k) = 1 - \frac{\omega_{pe}}{n_0}\sum_{r=1}^{N}\frac{2a_r A_r}{\omega^2 - a_r k^2} = 0 \tag{3.6}$$

is found. This dispersion relation is on the same form as the dispersion relation (2.14) we found for multi component electron fluid plasmas, and with an appropriate choice of a_r and A_r they can be made identical. It also resembles the kinetic dispersion relation (3.4). (The derivative, F_0', of the Maxwell-Boltzmann distribution is negative for $u > 0$.)

Some of the harmonic perturbations in an N water-bag model will lie within the bag contours. We can therefore assume that there will be wave-particle interactions. Numerically this has been verified as multiple water-bag models have been found to reproduce Landau damping in the form of a phase mixing processes of individual undamped modes [26, 27, 28].

Because the multi-component fluid model we described in Section 2.3 is an equivalent of the multiple water-bag model we expect to observe Landau damping for the N-component fluid description when $N \to \infty$.

3.3.2 Oscillation amplitudes of the individual components

Fourier transforming the set of coupled wave-equations (2.10), which describe Langmuir wave propagation in an N electron component plasma, we readily obtain

$$\frac{n_r}{\sum_{y=1}^{N} n_y} = \frac{1}{k^2}\omega_{pe,r}^2 \frac{1}{\omega^2/k^2 - 3u_{th,r}^2}. \tag{3.7}$$

We find that the water-bag components with $(\omega/k)^2 \approx 3u_{th,r}^2 = u_r^2$ will have particularly large amplitudes. For these water-bags the boundaries are close to resonance with the wave with phase velocity ω/k. This is the water-bag equivalent of resonant particles.

If we sum both sides of (3.7) by $\sum_{r=1}^{N}$, we recover the dispersion relation (2.14) with $2N$ solutions.

One particular feature of (3.7) deserves to be mentioned explicitly. We can use this relation as a prescription for the initial condition, i.e. the distribution of initial amplitudes for n_r, and insert one of the N dispersion relations for ω, e.g. $\omega = \omega_J(k)$. By this choice we ensure, however, that this and only this dispersion relation will be excited by the initial condition. The result will be one propagating undamped plane wave! This is the water-bag equivalent of the so called "van Kampen-Case modes" discussed in the literature on the linearized Vlasov equation [15]. In particular we note that we can obtain a wave with phase velocity larger than *any* of the u_r (i.e. $\omega^2/k^2 \gg 3u_{th,r}^2$), by using the approximation

$$\frac{n_r}{\sum_{y=1}^{N} n_y} \approx \omega_{pe,r}^2 \frac{1}{\omega^2}.$$

For this case we have all n_r to be positive and weighted according to their fraction of the total density. For more general values of the phase velocity, we note that we will have positive and negative values of n_r when using (3.7).

Chapter 4

Nonlinear Langmuir waves

Our discussion of Langmuir waves have so far been based on linear analysis, here we expand the description to the nonlinear regime. The inherent nonlinearity of electron oscillations will be addressed first. It will be clear from the discussion that it is necessary to include ion dynamics in the analysis if we are to observe any nonlinearities of interest. This finally leads us to the Zakharov model.

4.1 Nonlinearity

One important feature of nonlinear waves is the possibility of a nonlinear frequency shift, $\delta\omega$. This means that the frequency of the finite amplitude wave is different from the one obtained by a linearized model. The nonlinear frequency shift can in general be positive or negative. This nonlinear frequency shift is important for determining the nonlinear stability of a weakly modulated wave train. We have the well know "Lighthill criterion", stating that such a wave train will be unstable (i.e. the modulation will increase with time) provided the following criterion is fulfilled:

$$\delta\omega \; u_g' < 0 \tag{4.1}$$

in terms of the derivative of the group velocity $u_g' \equiv du_g/dk$ with $u_g \equiv d\omega(k)/dk$. The Lighthill criterion (4.1) can be argued from first principles [18]. The nonlinear frequency shift will generally be proportional to the square of the amplitude a of the initial plane wave, and we can write $\delta\omega = \beta a^2$. The parameter β entering here was by Lighthill assumed to be a constant, independent of the wave number k. The conclusion concerning the modulational stability will remain correct also for $\beta = \beta(k)$, where now k is the wave number of the "carrier" wave.

This particular instability is interesting by being purely nonlinear, i.e. the wave will generally be *linearly* stable because terms containing a^2 will be ignored by a linearization. Modulational wave instabilities are observed in nature, for instance for ocean waves, but as we shall see also for Langmuir waves in plasmas. It is a general feature of the modulational instability that the unstable wavelengths of the modulation (i.e. not the carrier wave itself) will become longer and longer as a is reduced, in the limit of $a \to 0$, we recover a modulationally stable plane wave.

4.2 Pendulum-Langmuir analogy

If the wavelength, λ, of a Langmuir wave in a single electron component plasma is much larger than the Debye length, λ_{De}, the last term of the dispersion relation (2.5) becomes much smaller than the first term, that is $\omega_{pe}^2 >> 12\pi^2\omega_{pe}^2\lambda_{De}^2/\lambda^2$. We will only consider long-wavelength Langmuir waves here and thus assume that the oscillation frequency of the wave stays close to the plasma frequency and show little, if any, variation with wavenumber.

There are many examples of physical systems which oscillates with well defined frequencies, the pendulum as illustrated in Figure 4.1 is one of the simplest.

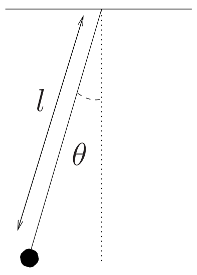


Figure 4.1: A pendulum of length l deflected θ degrees from the vertical.

Langmuir waves and the pendulum are similar in that they share the property of a well defined frequency. We shall therefore concentrate on the simplest system, the pendulum, in order to illustrate some characteristic features of such systems, with particular attention to the nonlinear frequency shift.

4.2.1 Nonlinear pendulum

The basic equation describing the motion of a pendulum like the one in Figure 4.1 is

$$\frac{d^2}{dt^2}\theta + \frac{g}{l}\sin\theta = 0, \tag{4.2}$$

where g is the gravitational acceleration, θ the angular deflection of the pendulum with respect to the vertical and l the length of the pendulum. The gravitational force, $F_g = -\frac{g}{l}\sin\theta$, on the pendulum is derived from the potential, $\Psi = -\frac{g}{l}\cos\theta$, as $F_g = \frac{d\Psi}{d\theta}$.

Assuming small amplitudes only we can linearize the momentum equation by use of the series expansion of $\sin\theta$ and obtain

$$\frac{d^2}{dt^2}\theta + \frac{g}{l}\theta = 0 \tag{4.3}$$

corresponding to an oscillation with frequency $\omega_0 = \sqrt{\frac{g}{l}}$ [12]. The linearization causes us to only consider the series expansion of $\sin\theta$ to first order. Physically we are then approximating the potential well at the minimum of the $\sin\theta$-function by a parabola.

Allowing also terms to next order in θ to be considered we find the nonlinear Duffing equation

$$\frac{d^2}{dt^2}\theta + \frac{g}{l}\theta = \frac{g}{3!l}\theta^3. \tag{4.4}$$

We shall solve this equation in an approximate way by use of the Lindsted-Poincaré method [29], also known as the method of *strained parameters*, following the derivation of Pécseli [15]. The aim of the method is to eliminate terms growing linearly with time from the solution of (4.4). Such terms are denoted secular and must be eliminated because they would cause the pendulum amplitude to increase indefinitely with time and thereby construct an unphysical system. In order to remove secularities we therefore introduce the variables $\tau \equiv t\sqrt{\frac{l}{g}}$ and rewrite the equation (4.4) as

$$\frac{d^2}{d\tau^2}\theta + \theta = \epsilon\theta^3$$

where ϵ is formally a small parameter. We also introduce $\tau \equiv s(1+\epsilon\omega_1+\epsilon^2\omega_2+ ...)$, where ω_1, ω_2 and so on are corrections to the dispersion relation. These corrections originate from the nonlinear effects introduced by the higher order terms in the series expansion of $\sin\theta$. Finally the Duffing then becomes

$$\frac{d^2}{ds^2}\theta + (1+\epsilon\omega_1+\epsilon^2\omega_2+...)^2(\theta+\epsilon\theta^3) = 0. \tag{4.5}$$

Furthermore we let $\theta = \sum_0^\infty \epsilon^n\theta_n(s)$ and find the contributions to (4.5) to like powers of ϵ:

- Zeroth order in ϵ
$$\frac{d^2}{ds^2}\theta_0 + \theta_0 = 0$$
- First order in ϵ
$$\frac{d^2}{ds^2}\theta_1 + \theta_1 = -\theta_o^3 - 2\omega_1\theta_0$$
- Second order in ϵ
$$\frac{d^2}{ds^2}\theta_2 + \theta_2 = -3\theta_o^2\theta_1 - 2\omega_1(\theta_1+\theta_0^3) - (\omega_1^2+2\omega_2)\theta_0$$

The general solution to zeroth order is $\theta_0 = a\cos{(s+\phi)}$, where a and ϕ are integration constants. Inserting θ_0 in the equation to first order we get

$$\begin{aligned}\frac{d^2}{ds^2}\theta_1 + \theta_1 &= -a^3\cos^3{(s+\phi)} - 2\omega_1 a\cos{(s+\phi)}\\ &= -\frac{a^3}{4}\cos 3(s+\phi) - \left(\frac{3}{4}a^2 + 2\omega_1\right)a\cos{(s+\phi)}.\end{aligned}$$

Assuming that this equation has a solution on the form $\theta_1 = A\cos 3(s+\phi) + B\cos{(s+\phi)}$ we find the solution $\theta_1 = \frac{a^3}{32}\cos 3(s+\phi) + \left(\frac{3a^2}{32}+\frac{\omega_1}{4}\right)a\cos{(s+\phi)}$. The term $\frac{\omega_1}{4}a\cos{(s+\phi)}$ will produce secularities. We therefore choose $\omega_1 = -\frac{3a^2}{8}$ to ensure that these secularities are eliminated. Consequently the solution becomes $\theta_1 = \frac{a^3}{32}\cos 3(s+\phi)$.

We now have known expressions for θ_0, θ_1 and ω_1, and inserting them in the equation to second order in ϵ gives

$$\frac{d^2}{ds^2}\theta_2 + \theta_2 = \left(\frac{51}{128}a^3 - 2\omega_2\right)a\cos{(s+\phi)} + \text{terms that do not produce secularities.}$$

In the same way as for ω_1 we shall strain ω_2 to a value that eliminates secular terms, this value must clearly be $\omega_2 = \frac{51a^4}{256}$. We now have the first terms in the series expansion of θ:

$$\theta = \sum_0^\infty \epsilon^n \theta_n(s) = \theta_0 + \epsilon\theta_1 + \epsilon\theta_2 + ... = a\cos(\omega\tau + \phi) + \frac{a^3}{32}\cos 3(\omega\tau + \phi)$$
$$+ \text{ higher order terms}$$

where we have set

$$\omega = \frac{s}{t} = \frac{1}{(1 + \epsilon\omega_1 + \epsilon^2\omega_2 + ...)}.$$

By use of the approximation $\frac{1}{1+x} \approx 1 - x$ we see that

$$\omega \approx (1 - \epsilon\omega_1 - \epsilon^2\omega_2 - ...) \approx 1 + \epsilon\frac{3a^2}{8} - \epsilon^2\frac{51a^4}{256}.$$

Apparently the dispersion relation for the pendulum has a nonlinear frequency shift proportional to a^2. The physical interpretation of this nonlinear frequency shift corresponds to correcting the parabolic potential well mentioned for the linear oscillation, in such a way that it fits $\theta + \epsilon\theta^3$ for any amplitude a.

Although this analysis was conducted for a pendulum we expect the observation of a nonlinear frequency shift to be an important characteristic also for other oscillations with well defined characteristic frequencies, therein Langmuir waves.

The sort of nonlinearity that arises here is frequently referred to as an eigen-nonlinearity because it originates from the very physical system itself and is thus an inherent property.

4.3 Langmuir waves in a cold plasma revisited

In section 2.1 we developed the fluid model for high frequency electrostatic oscillations in an one-dimensional plasma. We argued ions to be immobile because of their large inertia and therefore only considered electron dynamics. For a cold plasma (the electron temperature equal to zero) the electron dynamics are described by the following fluid equations:

- Continuity of the electrons

$$\frac{\partial}{\partial t}n + u\frac{\partial}{\partial x}n = -n\frac{\partial}{\partial x}u \tag{4.6}$$

- The electron momentum equation

$$\frac{\partial}{\partial t}u + u\frac{\partial}{\partial x}u = -\frac{e}{m}E \tag{4.7}$$

- Poisson's equation

$$\frac{\partial E}{\partial x} = \frac{e}{\epsilon_0}(n - n_0) \tag{4.8}$$

As in previous sections $n(x,t)$ is the electron density, $u(x,t)$ the electron drift velocity, $E(x,t)$ the electrostatic potential and ϵ_0 the vacuum permittivity. Assuming only small perturbations of the electron plasma density from the quasi neutrality assumption and linearizing the equations we found cold electrons to oscillate at the plasma frequency, $\omega_{pe} \equiv \sqrt{e^2n/\epsilon_0 m}$. Here we will not linearize the equations but consider the Langmuir wave evolution also for electron density deviations of finite amplitude. In the analysis we use the notation that is standard in the literature.

From Maxwell's equations we have Ampère's law with Maxwell's correction,

$$\nabla \times \boldsymbol{B} = \mu_0 \boldsymbol{J} + \mu_0\epsilon_0\frac{\partial}{\partial t}\boldsymbol{E}, \tag{4.9}$$

where $\boldsymbol{B}$ is the magnetic field, μ_0 the permeability of free space, $\boldsymbol{J}$ the plasma current density and the last term is recognized as the Maxwell displacement current. In our one-dimensional plasma the plasma current density, $J = enu$, must cancel the Maxwell displacement current since there will be no contribution from the left hand side of (4.9). Therefore the relation

$$\epsilon_0\frac{\partial E}{\partial t} = enu$$

is valid. We use it to rewrite Poissson's equation (4.8) by eliminating en/ϵ_0 and find

$$\frac{\partial E}{\partial t} + u\frac{\partial E}{\partial x} = \frac{en_0}{\epsilon_0}u.$$

Combining the rewritten Poisson equation with the electron momentum equation we obtain a closed equation for the electron velocity:

$$\left(\frac{\partial}{\partial t} + u\frac{\partial}{\partial x}\right)^2 u = -\omega_{pe}^2 u. \tag{4.10}$$

The square root of the operator on the left hand side is often referred to as the total time derivative operator D/Dt. In section 4.2 we found a similar equation (4.3) describing the dynamics of the pendulum. If we are only concerned with small electron density perturbations we therefore know that they will, as expected, give harmonic oscillations at the electron plasma frequency ω_{pe}.

So far we have described the electron dynamics in the Eulerian formalism, which means that we stand still at a point (x) and observe the plasma as it pass by us. The Lagrangian formalism on the other hand corresponds to following an individual fluid particle with initial coordinates (x_0, t_0) as it moves through space and time [30]. Dawson, [31], discovered that in order to solve (4.10) without linearization it is advantageous to employ the Lagrangian description. To do so we need the transformation between Eulerian coordinates and Lagrangian coordinates:

$$x = x_0 + \int_0^\tau u(x_0, t')\, dt' \quad \text{and} \quad \tau = t. \tag{4.11}$$

where $u(x_0, t')$ is the electron drift velocity in the Lagrangian frame, it describes the velocity of the specific fluid parcel that was initially at position x_0. Here we have set $t_0 = 0$. From this transformations we after some algebra also have

$$\frac{\partial}{\partial x} = \frac{1}{1 + \int_0^\tau \frac{\partial}{\partial x_0} u(x_0, t')\, dt'} \frac{\partial}{\partial x_0}$$

and

$$\frac{\partial}{\partial t} = \frac{\partial}{\partial \tau} - \frac{u(x_0, \tau)}{1 + \int_0^\tau \frac{\partial}{\partial x_0} u(x_0, t')\, dt'} \frac{\partial}{\partial x_0},$$

These two partial differential operators can be combined into

$$\frac{\partial}{\partial t} + u\frac{\partial}{\partial x} = \frac{\partial}{\partial \tau},$$

which is the time derivative in the Lagrangian frame of reference, and the total time derivative already mentioned.

Relation (4.10) is accordingly expressed in the Lagrangian formalism as

$$\frac{\partial^2}{\partial \tau^2} u(x_0, \tau) = -\omega_{pe}^2 u(x_0, \tau),$$

which we recognize as the harmonic oscillator with solution

$$u(x_0) = U(x_0) \cos \omega_{pe}\tau + \omega_{pe} X(x_0) \sin \omega_{pe}\tau, \tag{4.12}$$

where $U(x_0)$ and $X(x_0)$ is determined by the initial conditions. In the Lagrangian frame all oscillations, both linear and nonlinear, will thus have the plasma frequency as oscillation frequency.

Rewriting also the electron momentum equation (4.7) in the Lagrangian formalism gives

$$\frac{\partial}{\partial \tau} u(x_0, \tau) = -\frac{e}{m} E$$

and after inserting (4.12) we obtain an expression for the electrostatic field:

$$E(x_0, \tau) = \frac{m}{e}\omega_{pe} U(x_0) \sin \omega_{pe}\tau - \frac{m}{e}\omega_{pe}^2 X(x_0) \cos \omega_{pe}\tau.$$

The electron continuity equation (4.6) is rewritten in the Lagrangian formalism as

$$\frac{\partial}{\partial \tau}\left(n(x_0, \tau) \left(1 + \int_0^\tau \frac{\partial}{\partial x_0} u(x_0, t')\, dt \right) \right) = 0$$

from which

$$n(x_0, \tau) = \frac{n(x_0, 0)}{1 + \int_0^\tau \frac{\partial}{\partial x_0} u(x_0, t')\, dt'},$$

and inserting (4.12) we finally find the electron density

$$n(x_0,\tau) = \frac{n(x_0,0)}{1 + \frac{1}{\omega_{pe}}\frac{dU(x_0)}{dx_0}\sin\,\omega_{pe}\tau + \frac{dX(x_0)}{dx_0}(1-\cos\,\omega_{pe}\tau)}.$$

We now have a full description of the Langmuir waves in a cold plasma as seen in the Lagrangian frame of reference. In that frame of reference it is clear that any initial disturbance will oscillate at the electron plasma frequency, ω_{pe}, in a sinusoidal way. We are interested in the behavior of the waves also in the Eulerian frame of reference. It is possible to use the transformation (4.11) to find the Eulerian description, but it requires known expressions for the initial conditions $U(x_0)$ and $X(x_0)$.

Following the example of Davidson and Schram, [32, 33], we set the initial electron density perturbation as a plane wave with wave-number k:

$$\begin{aligned} n_0(x_0,0) &= n_0(1+\Delta\cos kx_0) \\ u(x_0,0) &= 0. \end{aligned}$$

In the Lagrangian frame this initial condition gives

$$u(x_0,\tau) = \frac{\omega_{pe}}{k}\Delta\sin kx_0\sin\omega_{pe}\tau \tag{4.13}$$

$$E(x_0,\tau) = -\frac{m\omega_{pe}^2}{ek}\Delta\sin kx_0\cos\omega_{pe}\tau \tag{4.14}$$

$$n(x_0,\tau) = n_0\frac{1+\Delta\cos kx_0}{1+\Delta\cos kx_0(1-\cos\omega_{pe}\tau)} \tag{4.15}$$

and the Eulerian-Lagrangian coordinate transform (4.11) becomes

$$kx = kx_0 + \alpha(\tau)\sin kx_0 \quad \text{and} \quad \tau = t, \tag{4.16}$$

where $\alpha(\tau) = 2\Delta\sin^2\left(\frac{\omega_{pe}\tau}{2}\right)$. A situation where the electron plasma density is completely depleted in one region, i.e. $n(x_0,\tau)=0$, is synonymous with the electrons being trapped in the potential well associated with the wave and the electron velocity being multi valued. We are not interested in such extreme situations and aim to describe the more general Langmuir wave evolution, therefore it is natural to require $n(x_0,\tau)>0$ for all τ, from which we find $\Delta < \frac{1}{2}$.

If we write

$$\sin kx_0 = \sum_{n=1}^{\infty} a_n(t) \sin nkx$$

where

$$a_n(t) = \frac{k}{\pi} \int_0^{\frac{2\pi}{k}} \sin nkx \sin kx_0 \, dx$$

and use the transformation (4.11) we get

$$a_n(t) = (-1)^{n+1} \frac{2}{n\alpha(t)} J_n[n\alpha(t)]$$

where $J_n[n\alpha(t)]$ is the Bessel function of first kind and order n. We use this to transform the equations (4.13)-(4.15) into the Eulerian frame of reference:

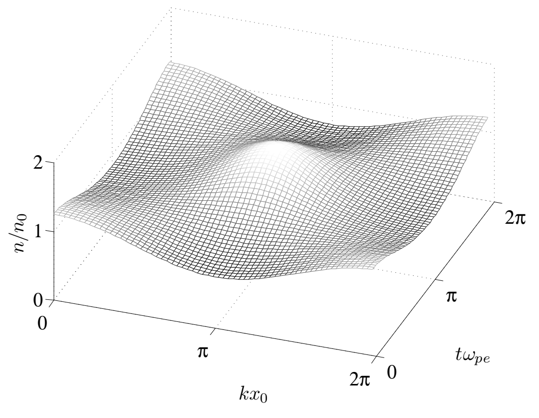
$$u(x,t) = \frac{\omega_{pe}}{k} \Delta \sum_{n=1}^{\infty} (-1)^{n+1} \frac{2}{n\alpha(t)} J_n[n\alpha(t)] \sin nkx \sin \omega_{pe} t \tag{4.17}$$

$$E(x,t) = -\frac{m\omega_{pe}^2}{ek} \Delta \sum_{n=1}^{\infty} (-1)^{n+1} \frac{2}{n\alpha(t)} J_n[n\alpha(t)] \sin nkx \cos \omega_{pe} t \tag{4.18}$$

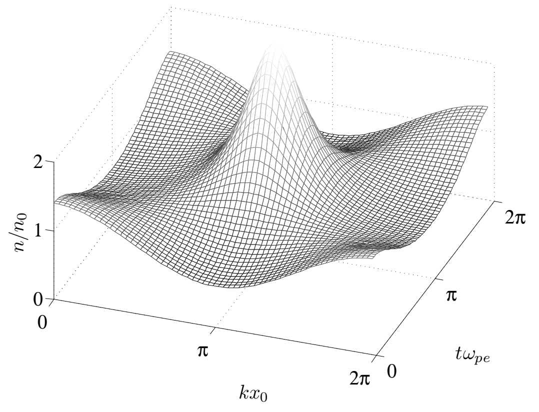
$$n(x,t) = n_0 \left(1 + \frac{2\Delta}{\alpha(t)} \sum_{n=1}^{\infty} (-1)^{n+1} \frac{2}{n\alpha(t)} J_n[n\alpha(t)] \cos nkx \cos \omega_{pe} t \right). \tag{4.19}$$

Figure 4.2 is an illustration of the electron density in the Eulerian frame of reference. It was created with the Lagrangian coordinate x_0 as a parameter for $n(x_0, \tau)$, which was thereafter mapped into the Eulerian frame with the transform (4.16).
From the figure it is clear that after a time interval $T = \frac{2\pi}{\omega_{pe}}$ the electron density is restored to the initial condition. T must then be the oscillation period for the electrons and therefore ω_{pe} the characteristic frequency of the oscillation. This is in accordance with what we would expect from relation (4.19). It is easily realised that $\alpha(t) = 2\Delta \sin^2\left(\frac{\omega_{pe} t}{2}\right)$ will be zero for any t that is a multiple of the period T. In the limit $n\alpha(t) \to 0$ we have, [34],

$$J_n[n\alpha(t)] \approx \frac{[\frac{1}{2}z]^n}{\Gamma(n+1)}$$



(a) Initial electron density perturbation $\frac{n}{n_0} = 1 + 0.25\cos kx_0$.



(b) Initial electron density perturbation $\frac{n}{n_0} = 1 + 0.4\cos kx_0$.

Figure 4.2: Nonlinear evolution of the electron plasma density in cold plasmas with immobile ions. Shown here in Eulerian coordinates for two different initial conditions.

where Γ is the gamma function. In other words, when $\alpha(t)$ approaches zero the Bessel function approaches zero as $\alpha(t)^n$. This means that the Bessel function goes faster to zero than its argument. Returning to the equation (4.19) for the electron density we then see that the initial condition n_0 must be reproduced for any multiple of T, just as in Figure 4.2.

One should note that even if the plasma frequency, ω_{pe}, is the oscillation frequency we observe a clearly nonlinear evolution of the electron density within the time interval $[0, T]$. Figure 4.3 and Figure 4.4 further illustrate this as they show how the electron velocity and the electric field associated with the Langmuir waves exhibit a nonlinear-steepening effect in the Eulerian frame of reference. Nonlinear effects are, however, most clearly visible in the density evolution as seen in Figure 4.2. Although we find clear signatures of nonlinearities for the finite amplitude Langmuir waves in cold plasmas, we also note that these nonlinearities concern only details in the wave-form; the frequency is independent of the wave amplitude. The Lighthill criterion (4.1) predicts that these Langmuir waves are modulationally stable. Modulational instability will be found only if other effects are included.

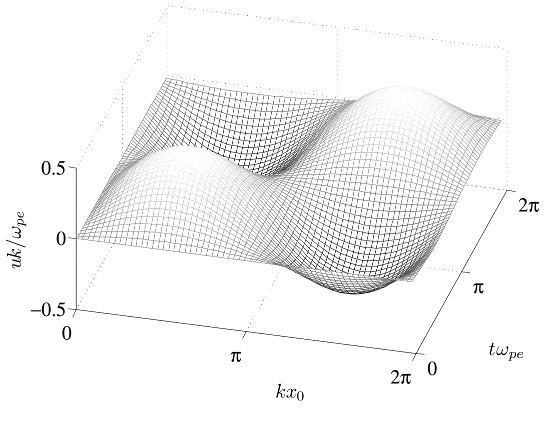
In a warm plasma, where the electron temperature has a finite value, analysis by Coffy [35] and Shivamoggi [4] show that thermal effects will only lead to a small frequency shift from the cold plasma result.

4.4 Parametric nonlinearity

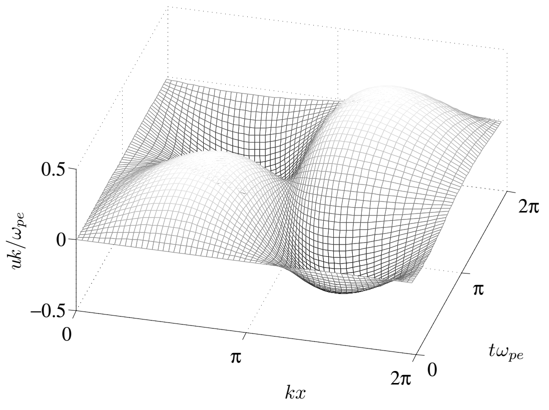
In section 4.2 we found that the pendulum is an inherent nonlinear system. However, by investigating the linearized momentum equation (4.2) we see that if the pendulum length, l, is somehow depending on the amplitude which is a function of time, the equation becomes nonlinear. The nonlinearity is then caused by a parameter in the pendulum system. Nonlinearities of this sort are frequently referred to as parametric. They are known also in electric circuit theory where signals are parametrically amplified for instance by varying the distance between the plates of parallel-plate capacitors [36] and in various engineering structures where vibrations of construction components causes parametric instabilities [37].

4.4.1 Weakly nonlinear Langmuir waves with varying ion density

Parametric instabilities are recognized as a prevailing nonlinear phenomenon in plasmas, not only for the electrostatic waves that we are interested in, but also for linear and nonlinear electromagnetic waves [38]. Here we introduce the self consistent parameter variation through a prescribed spatial variation of the electron density, $n(x) = n_0 + \bar{n}(x)$, where, as in Section 2.1, n_0 is the equilibrium density, while $\bar{n}(x)$ is the new spatial variation. Figure (4.5) is an illustration of the electron distribution in such an inhomogeneous plasma. The electron dynamics will be described by the same fluid model as for linear Langmuir waves in Chapter 2 and the ions are assumed immobile on account of their inertia. However, linearization with the same assumptions as in



(a) Lagrangian coordinates



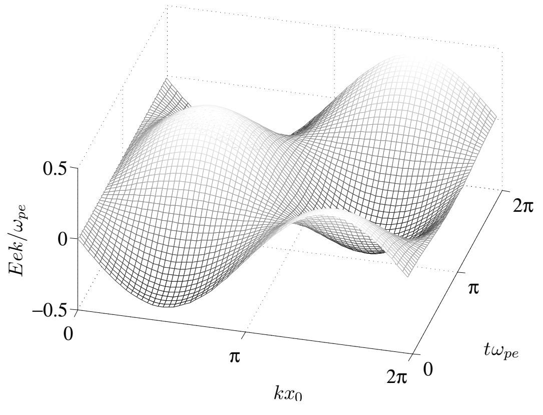
(b) Eulerian coordinates

Figure 4.3: The electron velocity in a cold plasma with immobile ions when the initial electron density perturbation is set to $\frac{n}{n_0} = 1 + 0.4 \cos kx_0$.

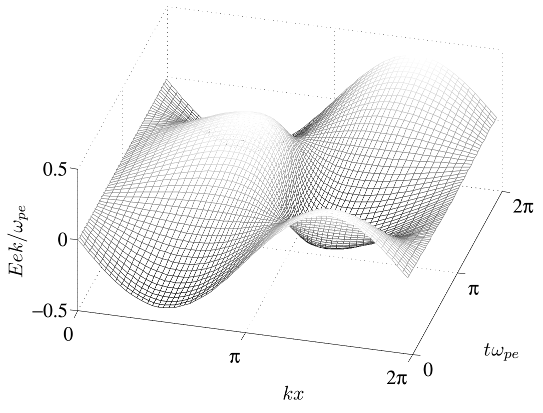
Section 2.1 will lead to a slightly different set of electron fluid equations than we have seen so far:

- The Poisson equation

$$\frac{\partial^2 \tilde{\phi}}{\partial x^2} = \frac{e}{\epsilon_0} \tilde{n}$$



(a) Lagrangian coordinates



(b) Eulerian coordinates

Figure 4.4: The electric field in a cold plasma with immobile ions when the initial electron density perturbation is set to $\frac{n}{n_0} = 1 + 0.4 \cos kx_0$.

- The electron continuity equation

$$\frac{\partial}{\partial t}\tilde{n} + n_0 \frac{\partial}{\partial x}\tilde{u} + \frac{\partial}{\partial x}\bar{n}\tilde{u} = 0$$

- The electron momentum equation

$$\frac{\partial}{\partial t}\tilde{u} = -\frac{3u_{th}^2}{n_0}\frac{\partial}{\partial x}\tilde{n} + \frac{e}{m}\frac{\partial \tilde{\phi}}{\partial x}$$

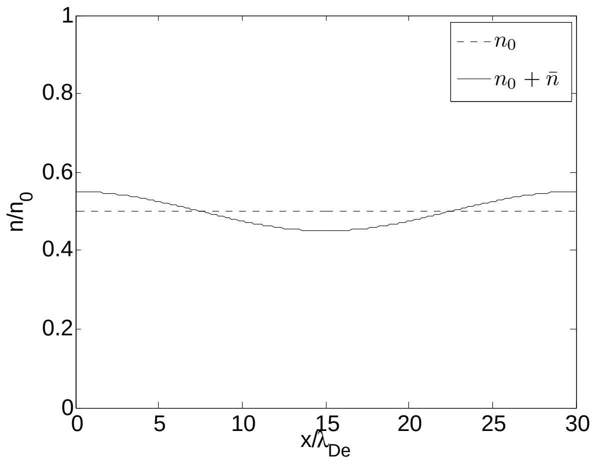


Figure 4.5: The electron density $n = n_0 + \bar{n}(x)$ in a plasma where $n_0 = 0.5$ and $\bar{n}(x) = 0.05\cos\frac{2\pi x}{30}$.

where we have used $u_{th}^2 \equiv \frac{\kappa T}{m}$.

By differentiating the continuity equation with respect to time the three above equations can be combined into

$$\frac{\epsilon_0}{e}\frac{\partial^2}{\partial t^2}\frac{\partial^2}{\partial x^2}\tilde{\phi} - 3u_{th}^2\frac{\partial^2}{\partial x^2}\tilde{n} + \frac{en_0}{m}\frac{\partial^2}{\partial x^2}\tilde{\phi} \\ + \frac{\partial}{\partial x}\left(-\frac{3u_{th}^2}{n_0}\bar{n}\frac{\partial}{\partial x}\tilde{n} + \frac{e\bar{n}}{m}\frac{\partial}{\partial x}\tilde{\phi}\right) = 0. \quad (4.20)$$

The last term is a sum of two nonlinear terms, the first originates from the adiabatic pressure variation and the last from the term $\frac{\partial}{\partial x}nu$ in the continuity equation. Here we have chosen the unperturbed electron velocity as $u_0 = 0$. If this velocity has a finite value the convective term $u\frac{\partial}{\partial x}u$ in the electron momentum equation will also give nonlinear terms in (4.20). Dysthe & Pécseli [39] have shown that nonlinearities arising from adiabatic pressure variations and convection are negligible when compared to the nonlinearity introduced by the continuity equation. The term $-\frac{3u_{th}^2}{n_0}\bar{n}\frac{\partial}{\partial x}\tilde{n}$ is $\bar{n}/n_0$ times the second term in (4.20). Since we assume that $\bar{n}/n_0$ is small in the present analysis and also that electron thermal corrections are small we will ignore the fourth term in (4.20). Cancelling the obsolete nonlinear term and integrating (4.20) over x once we therefore obtain

$$\frac{\partial^2}{\partial t^2}\tilde{E} - 3u_{th}^2\frac{\partial^2}{\partial x^2}\tilde{E} + \omega_{po}^2\tilde{E} + \frac{\bar{n}}{n_0}\omega_{po}^2\tilde{E} = 0, \tag{4.21}$$

which is a weakly nonlinear partial differential equation describing the space-time evolution of the electrostatic field perturbation, $\tilde{E}$, associated with Langmuir waves in an inhomogeneous plasma. The nonlinearity is caused by the externally imposed inhomogeneity in the electron density distribution, i.e. it is a parametric nonlinearity. The result (4.21) has been derived several times in the litterature (e.g. by Dysthe & Pécseli [39]) and is here obtained by more heuristic arguments.

4.5 Origin of nonlinearity

In the previous section the electron density distribution was considered inhomogeneous by some externally imposed modification. Here we shall see how a Langmuir wave travelling with a finite amplitude induce an inhomogeneous electron density distribution and thus effectively modify the refractive index of the plasma. A modification of the plasma density profile in this manner by the plasma wave itself is characterized as an eigen nonlinearity, see section 4.2, and the mechanism is physically identified as the *ponderomotive force*. From the derivation of the pondoromotive force we find that ions will also be set in motion and we finally arrive at the *Zakharov* model.

4.5.1 The ponderomotive force

A spatial variation in the amplitude of the electric field associated with a Langmuir wave causes electrons to experience a Lorentz force whose magnitude varies in different spatial regions. Such spatial variations are expected for any finite amplitude wave. The electron velocity, the electrostatic potential and the electron density will be affected by this and we take them to be sums of a slowly and a rapidly varying part, as respectively $u = \bar{u} + \tilde{u}$, $\phi = \bar{\phi} + \tilde{\phi}$ and $n = \bar{n} + \tilde{n}$, where the quantities with a bar over represents a slow variation and the ones with a tilde a rapid variation. The momentum equation (2.3) describes the electron dynamics, but since we are here mostly interested in low frequency phenomena we take the equation of state to be isothermal and therefore set the adiabatic index, γ, to unity. Averaging the momentum equation over the period for one high frequency electron oscillation

$$mn\left(\overline{\frac{\partial}{\partial t}u} + \overline{u\frac{\partial}{\partial x}u}\right) = \overline{-\kappa T\frac{\partial}{\partial x}n} + \overline{en\frac{\partial \phi}{\partial x}}$$

we obtain

$$m\left(\frac{\partial}{\partial t}\bar{u} + \bar{u}\frac{\partial}{\partial x}\bar{u}\right) + \frac{m}{2}\frac{\partial}{\partial x}\overline{\tilde{u}^2} = e\frac{\partial}{\partial x}\bar{\phi} - \frac{\kappa T}{n_0}\frac{\partial}{\partial x}\bar{n}. \tag{4.22}$$

Because of the small electron inertia we can ignore the two first terms on the left hand side, but the next term needs some special consideration.

High frequency electron oscillations will in their simplest form always follow the plasma frequency. We can say that oscillations at the plasma frequency are lowest order oscillations, whereas to higher order we find small corrections caused by thermal effects (as described in section 2.1) and nonlinearity. In order to approximate the rapid electron oscillations to first order, the governing momentum equation

$$\frac{\partial}{\partial t}\tilde{u} + \tilde{u}\frac{\partial}{\partial x}\tilde{u} = \frac{e}{m}\frac{\partial}{\partial x}\tilde{\phi}$$

is linearized and assuming the electron velocity perturbation to be harmonic, $\tilde{u} = \tilde{u}e^{-i(\omega t + kx)}$, Fourier transformation gives

$$\mid \tilde{u} \mid^2 = \frac{e^2}{m^2\omega^2} \mid \tilde{E} \mid^2 .$$

Finally we set $\omega = \omega_{pe}$ and obtain

$$\mid \tilde{u} \mid^2 = \frac{\epsilon_0}{mn_0} \mid \tilde{E} \mid^2 .$$

When we insert this on the left hand side of (4.22) in the third term, we find that the electron masses cancel, this is why we could not ignore that term earlier. After insertion the averaged momentum equation (4.22) becomes

$$\frac{1}{2}\frac{\epsilon_0}{n_0}\overline{\frac{\partial}{\partial x} \mid \tilde{E} \mid^2} = e\bar{E} - \frac{\kappa T}{n_0}\frac{\partial}{\partial x}\overline{n}. \tag{4.23}$$

The left hand side of this equation is interpreted as the *ponderomotive force.*

Rewriting the slowly varying electric field as $\bar{E} = -\frac{\partial \bar{\phi}}{\partial x}$ we can integrate 4.23 with respect to x, assuming that the fields vanish at $x \to \pm\infty$ so the integration constant is zero. The result is an isothermal Boltzmann distribution of the electron density,

$$n = n_0 e^{\frac{e\bar{\phi}}{\kappa T} - \frac{1}{2}\frac{\epsilon_0}{n_0}\overline{|\tilde{E}|^2}}.$$

The electrons will be attracted by the positive potential, $\bar{\phi}$, and expelled by the ponderomotive potential, $\frac{1}{2}\frac{\epsilon_0}{n_0}\overline{| \tilde{E} |^2}$. As an analogy we know electromagnetic waves transfer momentum to any surface they encounter, causing the surface to experience a force from the wave, an effect known as *radiation pressure* [40]. The effect of the ponderomotive force on the plasma can be seen as a "radiation pressure" from the high amplitude part of the Langmuir wavetrain acting only on the light electrons.

The pondoromotive force is best understood by following an electron as it oscillates in the high frequency electric field, $\tilde{E}$. If only affected by $\tilde{E}$ the electron would oscillate around an effective oscillation center, returning to its original position after every oscillation period. In contrast, when both the high frequency and low frequency fields are present, the electron will "feel" an inhomogeneous electric field acting on it. Consequently the particle, while oscillating, experiences a larger force in areas with high field amplitude compared to areas with lower field amplitude. This results in a net force pulling the electron towards the weak field area. As the electron oscillates it then drifts slowly away from the large field amplitude area, and thereby an inhomogeneous electron density distribution is created.

Index of refraction

The index of refraction in a plasma is given as the ratio of the speed of light in vacuum, c, to the phase velocity, $u_{ph} \equiv \frac{\omega}{k}$, of an electromagnetic wave as it propagates through the plasma [40]. For our Langmuir waves the index becomes

$$\eta \equiv \frac{ck}{\omega} = \frac{c}{\sqrt{3}u_{th}}\sqrt{1 - \frac{\omega_{pe}^2}{\omega^2}}. \tag{4.24}$$

The inhomogeneous electron density set up by the ponderomotive force changes the local plasma frequencies in the various sections of the plasma. In regions where the electron density is enhanced and the local plasma frequency consequently increased we therefore see a decrease in the index of refraction and similarly an increase in areas where the electron density decreases.

Besides the ponderomotive force the index of refraction of a plasma can be changed in a nonlinear manner for instance also by heating of the electrons and a resulting thermal pressure gradient. It was first discovered in the 1930's when people heard the transmission from Radio Luxembourg in the background whilst listening to broadcasters transmitting on completely different frequencies. Radio Luxembourg aimed to broadcast to all of Europe and their high power signal heated the ionospheric electrons. The heating rate was largest when the modulation amplitude of the radio signal was largest and the resulting index of refraction would therefore vary in proportion to the modulation envelope of the signal. Radio signals transmitted with less power travelling through the same ionospheric regions therefore upon reflection acquired a modulation that originally came from the Radio Luxembourg signal and the latter could be heard as a "ghost-signal" on the same frequency [41, 42].

4.5.2 Ion dynamics; The Zakharov model

The charge separation induced in a plasma by an inhomogeneous electron density distribution will cause slowly varying electric fields to set out to restore the plasma to quasi-neutrality, see Figure (4.6).

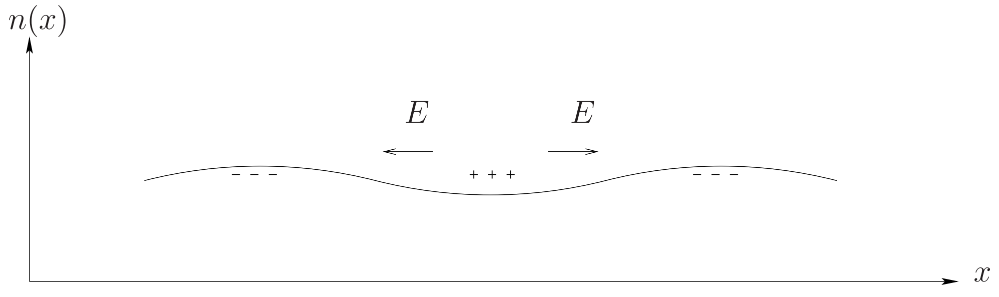


Figure 4.6: An illustration of how a spatially varying electron density distribution causes charge separation in a plasma and therefore induce electric fields.

These fields are an addition to the slowly varying part of the electric field associated with Langmuir waves with finite amplitudes. Since all these fields are low frequency phenomena they can set ions into motion. The resulting ion dynamics is described by the continuity equation

$$\frac{\partial}{\partial t}n + \frac{\partial}{\partial x}nu_i = 0 \tag{4.25}$$

and the momentum equation

$$M\left(\frac{\partial}{\partial t}u_i + u_i\frac{\partial}{\partial x}u_i\right) = e\frac{\partial}{\partial x}\phi$$

where u_i is the ion velocity, M the ion mass and ϕ the electrostatic potential for the slowly varying electric field. On slow timescales the ion density, n, is bounded by the quasi-neutrality assumption to be identical to the electron density, n. We have, for no other reason than simplicity, assumed the ion temperature to be equal to zero.

In the following we assume the ion density, velocity and electrostatic potential to be perturbed from the quasi neutral equilibrium as respectively $n = n_0 + \bar{n} = \bar{n}$, $u_i = u_{i0} + \bar{u_i} = \bar{u_i}$ and $\phi = \phi_0 + \bar{\phi} = \bar{\phi}$ where all the equilibrium values, denoted by $_0$ are set to zero and the parameters with a bar are slowly varying perturbations. After linearization the ion continuity and momentum equations can be combined into.

$$\frac{\partial^2}{\partial t^2}\frac{\bar{n}}{n_0} - \frac{e}{M}\frac{\partial^2}{\partial x^2}\bar{\phi} = 0.$$

Because the ion temperature is set to zero these ion acoustic waves must be propagated solely by the electron pressure $P = \kappa T\frac{\partial}{\partial x}n$, by definition the sound speed of the waves are therefore given as $C_s^2 \equiv \frac{dP}{d\rho} = \frac{\kappa T}{M}$. Using the relation for the ponderomotive force (4.23) the above ion acoustic wave equation is thus rewritten as

$$\frac{\partial^2}{\partial t^2}\frac{\bar{n}}{n_0} - C_s^2\frac{\partial^2}{\partial x^2}\frac{\bar{n}}{n_0} - \frac{\frac{1}{2}\epsilon_0}{Mn_0}\frac{\partial^2}{\partial x^2}\overline{\tilde{E}^2} = 0. \tag{4.26}$$

Remembering that on the time scales considered here the electron and ion densities must, by the quasi neutrality assumption, be roughly identical, it is easily realised that this equation does not only describe the low frequency ion dynamics but also the low frequency electron dynamics (i.e. the bulk slowly varying plasma density).

In section 4.4 we found a relation, (4.21), describing the electric field perturbation, $\tilde{E}$, in a plasma with an inhomogeneous electron distribution. Our interest here is the low frequency dynamics associated with the ponderomotive force and the ion dynamics. Therefore we let $\tilde{E}$ be the product of a slowly varying amplitude and a rapidly varying phase factor as

$$\tilde{E} = \bar{E}(x,t)e^{-i\omega_{pe}t} + complex\, conjugate, \tag{4.27}$$

As already mentioned in section 2.4 this separation in amplitude and phase factor is not without controversy, we do, however, accept it as a valid approximation.

Inserting (4.27) in equation (4.21) we find

$$i\frac{\partial}{\partial t}\bar{E} + \frac{3u_{th}^2}{2\omega_{pe}}\frac{\partial^2}{\partial x^2}\bar{E} - \frac{\omega_{pe}}{2}\frac{\bar{n}}{n_0}\bar{E} = 0, \tag{4.28}$$

where we have made the approximation $\omega_{p0} \approx \omega_{pe}$. The definition (4.27) also yields $\overline{\tilde{E}^2} = \mid \bar{E} \mid^2$. Reconsidering equation (4.26) and assuming the variation in the electron density to be slow enough for the $\frac{\partial^2}{\partial t^2}\frac{\bar{n}}{n_0}$ term to be ignored as compared to the others, we then find

$$C_s^2\frac{\bar{n}}{n_0} \approx -\frac{\epsilon_0}{2n_0 M} \mid \bar{E} \mid^2 .$$

We assume $\bar{n} \to 0$ and $\mid \bar{E} \mid^2 \to 0$ as $x \to \infty$ and therefore set all integration constants equal to zero. Inserting this result in equation (4.28) we finally obtain

$$i\frac{\partial}{\partial t}\bar{E} + \frac{3u_{th}^2}{2\omega_{pe}}\frac{\partial^2}{\partial x^2}\bar{E} + \frac{\omega_{pe}}{4}\frac{\epsilon_0}{n_0\kappa T} \mid \bar{E} \mid^2 \bar{E} = 0, \tag{4.29}$$

which is a nonlinear Schrödinger (NLS) equation, describing the slowly varying electric field caused in part by the ponderomotive force. This equation is also known as a special case of the Zakharov model [10]. One important property of this NLS-equation is that it possesses so called "soliton"-solutions. These are exact solutions to a nonlinear equation: several nonlinear equations have this property, for instance also the Korteweg-de Vries equation that is often used to model shallow water waves in channels [43]. These solutions are important due to a peculiar property: they emerge with unchanged form after a nonlinear interaction with other waveforms, such as other solitons. A full analysis of these type of problems will in general require an inverse scattering transform [43], which is outside the scope of the present thesis. Many nonlinear wave equation have stationary solutions, but these will usually not have the before mentioned soliton property, and they are therefore comparatively less interesting.

By inspection of the NLS-equation we note that the nonlinear frequency shift is negative because the bulk slowly varying plasma density and thus the local plasma frequency decreases when the wave amplitude *increases*, see the third term in (4.29). Recalling the Langmuir wave dispersion relation $\omega^2 = \omega_{pe}^2 + 3u_{th}^2 k^2$ we have the approximation $\omega \approx \omega_{pe} + \frac{3}{2}u_{th}^2 k^2/\omega_{pe}$, giving the group velocity $u_g \approx 3u_{th}^2 k/\omega_{pe}$ and its derivative $u_g' = 3u_{th}^2/\omega_{pe} > 0$. Apart from a numerical factor, this is the coefficient we find to the second term on the left side of (4.29).

If we assume a plane wave solution, $\bar{E} = ae^{-i(\delta\omega t - kx)}$ where a is a constant amplitude, for the slowly varying electrostatic field we find by insertion in (4.29)

$$\delta\omega = \frac{3u_{th}^2}{2\omega_{pe}}k^2 - \frac{\omega_{pe}}{4}\frac{\epsilon_0}{n_0\kappa T}a^2.$$

For small wavenumbers, k, we have the approximation

$$\delta\omega \approx -\frac{\omega_{pe}}{4}\frac{\epsilon_0}{n_0\kappa T}a^2.$$

From this we conclude that the nonlinear frequency shift is not only negative, but also proportional to the square of the amplitude of the electrostatic field perturbation. Furthermore the porportionality has a constant coefficient, exactly as Lighthill assumed for the waves he studied.

Relative nonlinearity

Mathematically the NLS consist of a time derivative, a diffusive term and a term proportional to the energy density (here in one spatial dimension, energy per length), $\frac{W}{L} \propto \mid E \mid^2$, in the electrostatic field E. The ratio of the diffusive term to the last term is found as

$$\frac{3u_{th}^2/2\omega_{pe}}{\omega_{pe}\varepsilon_0/4n_0\kappa T}\frac{\frac{\partial^2}{\partial x^2}E}{\mid E \mid^2 E} \propto \frac{\frac{E}{L^2}}{\frac{WE}{L}} \propto \frac{1}{WL}. \tag{4.30}$$

The above relation suggests that the larger the energy in the electrostatic field is the stronger the nonlinear effect will be and vice versa if the energy is small. If the initial field perturbation is not very energetic we therefore expect the diffusive term to dominate (for given L) so that the time evolution of the perturbation will appear linear.

4.6 Generalization of the Zakharov model

In order to outline the generalization of the single water-bag model (i.e. a simple fluid model) to the multi-water bag model, it may be an advantage to make a summary of the basic results, including a slight generalization.

Weakly nonlinear electron plasma waves, Langmuir waves, in isotropic and homogeneous plasmas are often described by a simplified set of equation, [10], such as

$$\frac{\partial}{\partial x}\left[2i\omega_\star \frac{\partial}{\partial t}\frac{\partial}{\partial x}\phi_s + 3u_{th}^2 \frac{\partial^2}{\partial x^2}\frac{\partial}{\partial x}\phi_s + (\omega_\star^2 - \omega_{pe,r}^2)\frac{\partial}{\partial x}\phi\right] = \omega_{pe}^2 \frac{\partial}{\partial x}\left(\frac{\overline{n}}{n_0}\frac{\partial}{\partial x}\phi_s\right) \quad (4.31)$$

which accounts for the space-time evolution of the Langmuir wave envelope. The low frequency bulk plasma dynamics are governed by

$$\frac{\partial^2}{\partial t^2}\frac{\overline{n}}{n_0} - C_s^2 \frac{\partial^2}{\partial x^2}\frac{\overline{n}}{n_0} = \left(\frac{\omega_{pe}}{\omega_\star}\right)^2 \frac{\varepsilon_0}{2n_0 M}\frac{\partial^2}{\partial x^2}\left|\frac{\partial}{\partial x}\phi\right|^2, \quad (4.32)$$

in standard notations, where M is the ion mass and C_s is the ion acoustic speed. Inclusion of a frequency $\omega_\star \neq \omega_{pe}$ in the form given here reflects a slight freedom in choice of the reference frequency for the fluctuations. In the standard presentations $\omega_\star = \omega_{pe}$ is used, as also before. If we take $\omega_\star = \omega_{pe}$ we find one of the terms in (4.31) disappearing, and we recover the foregoing approximation.

As before the basic idea is to consider in a first approximation oscillations at the cold plasma resonance, and in the next step to consider the nonlinearity and thermal dispersion as small corrections. The important nonlinearity is then the local modification of the cold plasma resonance induced by the wave–nonlinearity. In the basic equations the dominant nonlinearity arises from $\frac{\partial}{\partial x}nu$ in the continuity equation, while e. g. the nonlinear effects from the expression for adiabatic electrons are negligible in comparison.

We recognize the electric field $E = -\frac{\partial}{\partial x}\phi$ in (4.31), which can readily be introduced, but it is an advantage for the moment to retain the electrostatic potential explicitly. In the present one-dimensional presentation, we can also integrate to remove the operator $\frac{\partial}{\partial x}$ which appears on all terms in (4.31). Doing this, we recover the NLS equation (4.29).

We now return to the case of a multi component water-bag model. Here the basic equation becomes

$$\frac{\partial^2}{\partial t^2}n_r - 3u_{th,r}^2 \frac{\partial^2}{\partial x^2}n_r + \frac{e}{m}n_{r,0}\frac{\partial^2}{\partial x^2}\phi = -\frac{e}{m}\frac{\partial}{\partial x}(\overline{n}_r \frac{\partial}{\partial x}\phi), \quad (4.33)$$

with the label of individual water-bags $r = 1, 2, \ldots, N$ as before, where we have introduced the slowly varying bulk plasma density explicitly on the right hand side.

We have also Poisson's equation including the sum over all water-bags

$$\frac{\partial^2}{\partial x^2}\phi = \frac{e}{\varepsilon_0}\sum_{r=1}^{N} n_r. \tag{4.34}$$

If we ignore the right hand side of (4.33) and introduce Poissons equation, we have the wave equation (2.10) studied before.

To study the nonlinear set of equations (4.33), it is advantageous to introduce $\bar{n}$ rather than $\bar{n}_r$ in (4.33) using the relation

$$\bar{n}_r = \bar{n}\frac{n_{r,0}}{T_r}\Big/\sum_k \frac{n_{k,0}}{T_k}. \tag{4.35}$$

This expression is a great simplification. It is obtained by an analysis entirely similar to the one used for deriving the ponderomotive force. For the present case where we do *not* introduce the electron plasma frequency, we find first the ponderomotive force as

$$\frac{1}{2}\frac{e^2}{m\omega_\star^2}|\tilde{E}|^2 = e\phi - \frac{\kappa T_r \bar{n}_r}{n_{r,0}}.$$

We then have

$$\frac{1}{2}\frac{e^2}{m\omega_\star^2}|\tilde{E}|^2 \sum_r^N \frac{n_{r,0}}{\kappa T_r} = e\phi \sum_r^N \frac{n_{r,0}}{\kappa T_r} - \sum_r^N \bar{n}_r \equiv e\phi \sum_r^N \frac{n_{r,0}}{\kappa T_r} - \bar{n}$$

which reproduces (4.35) by use of the previous relation.

The low frequency dynamics of $\bar{n}$ are still governed by (4.32) with the general expression $C_s = \left[\frac{n_0}{M\sum_{r=1}^N (n_{r,0}/\kappa T_r)}\right]^{1/2}$ for the sound speed.

We now repeat the simplifications used to obtain (4.29). With the standard approximation, we rewrite the time–variation of ϕ as $\phi_s(x,t)e^{-i\omega_\star t} + c.c.$ where now $\phi_s(x,t)$ is slowly varying with time, and for the electron density components $n_r = n_{r,s}(x,t)e^{-i\omega_\star t} + c.c.$ where $n_{r,s}(x,t)$ is also slowly varying with time. The equations (4.33) thus reduce to

$$-2i\omega_\star\frac{\partial}{\partial t}n_{r,s} - 3u_{th,r}^2\frac{\partial^2}{\partial x^2}n_{r,s} - \omega_\star^2 n_{r,s} + \frac{e}{m}n_{r,0}\frac{\partial^2}{\partial x^2}\phi_s = \\ -\frac{e}{m}\left(\frac{n_{r,0}}{T_r}\Big/\sum_k \frac{n_{k,0}}{T_k}\right)\frac{\partial}{\partial x}\left(\bar{n}\frac{\partial}{\partial x}\phi_s\right), \tag{4.36}$$

for $r = 1, 2, \ldots, N$ with complex $n_{r,s}$ and ϕ_s, but real $\bar{n}$. The set of equations is completed by (4.32) and (4.34). Together they form a generalized version of the Zakharov model.

Chapter 5

Numerical methods

Several of the partial differential equations introduced in Chapter 2 and 4 are not easily solved analytically, therefore we shall rather find approximate numerical solutions. The main numerical methods available for solving partial differential equations are the finite difference method, spectral methods and the finite element method.

The finite element method is most relevant when it comes to complicated geometries in several space dimensions. Spectral methods are generally very accurate and require little memory and CPU (central processing unit) time but they can not compete with the conceptual simplicity of the finite difference method. Finite difference schemes are easy to generalize for quite complicated problems, and also to implement on a computer. These qualities makes the finite difference method our numerical method of choice.

We start by introducing the finite difference method and discuss the validity of solutions obtained from the method. Thereafter we introduce some general concepts applicable to the numerical solution of all the proceeding problems.

5.1 The finite difference method

The essence of the finite difference method is to replace derivative expressions with approximately equivalent difference quotients. This approximation technique stems from the very definition of the derivative of a smooth function:

$$u'(x) = \lim_{h\to 0} \frac{u(x+h) - u(x)}{h}.$$

Partial derivative	**Finite difference approximation**		**Compact notation**	**Error**
$\frac{\partial u}{\partial x}$	One sided forward	$\frac{u_{j+1}-u_j}{h}$	$[\delta_x^+ u]_i$	$\mathcal{O}(h)$
	One sided backwards	$\frac{u_j-u_{j-1}}{h}$	$[\delta_x^- u]_i$	$\mathcal{O}(h)$
	Two sided	$\frac{u_{j+1}-u_{j-1}}{2h}$	$[\delta_{2x} u]_i$	$\mathcal{O}(h^2)$
$\frac{\partial^2 u}{\partial x^2}$	Centered	$\frac{u_{j+1}-2u_j+u_{j-1}}{h^2}$	$[\delta_x \delta_x u]_i$	$\mathcal{O}(h^2)$

Table 5.1: Partial differential operators and their corresponding finite difference approximations applied to a continuous function $u(x)$.

When we set out to approximate a continuous problem with difference quotients we must first find a discretization of the spatial and temporal domain over which the problem is valid. The above definition of the derivative makes it clear that difference quotients are only valid when the discretization provides grids with sufficiently small spacing.

All our finite difference approximations will be based on the following discretization in time and space:

- **Space**
  A 1D grid on the interval $x \in [0, L]$ consisting of m cells with constant spacing h. The nodes of the grid will be at coordinates $x_j = (hj)$ for $j = 0, 1, 2, ..., m$.

- **Time**
  A time grid with constant spacing Δt for $t \geq 0$. There will be nodes $t_l = (l\Delta t)$ for $l = 0, 1, 2, ..$

Taylor series expansion of smooth functions makes it possible to derive several finite difference approximations to partial derivative operators [44]. In Table 5.1 we list the ones that are to be used in our numerical approaches. The table also give the order of the error in the approximations as found from the Taylor series expansions.

Langtangen, [45], has introduced a most convenient compact notation for difference operators. The notation helps making the discrete equations arising from the finite difference method more intuitive and simplifies the mathematical analysis of their physical properties. We shall use this compact notation throughout our work. In addition to listing partial derivatives and their finite difference counterparts, Table 5.1 also include the compact notational form.

5.2 Convergence

The finite-difference method is based on approximations and the solutions it produces will contain errors. Furthermore, the finite precision arithmetic of computers may cause accumulation of round-off errors and thereby restrict the accuracy of any numerical solution. It is in our interest to find the order of these errors' contribution to the numerical solutions.

Ideally all numerical solutions would be tested against analytical results so that the exact error could be found. As this can of course not always be done it is crucial to ensure that the discrete problem is in fact analogous to the continuous, i.e. the difference between the discrete and continuous problem goes to zero as the grid parameters approaches zero. A numerical scheme with this property is said to be convergent.

It is normally no easy task to directly prove the convergence of a numerical scheme; it requires exact solutions of the discrete equations. Fortunately The Lax Equivalence Theorem [46] provides a means of determining convergence without producing exact solutions.

The Lax Equivalence Theorem :
> Given a well-posed mathematical problem and a consistent finite difference approximation to it, stability is a necessary and sufficient condition for convergence.

According to this theorem it is thus sufficient to show that a scheme is consistent and establish its stability condition in order to ensure it is convergent. Both these concepts, consistency and stability, will be explained in the following and are much easier accessed than the exact error of a numerical scheme.

5.2.1 Consistency

A numerical scheme is said to be consistent if it approximately solves the original partial differential equation when the grid parameters approaches zero [44, 45]. To investigate this we shall consider a partial differential equation $\mathcal{L}(u) = f$ and the corresponding discrete problem $\hat{\mathcal{L}}(\hat{u}) = \hat{f}$, where $\mathcal{L}$ denotes a differential operator, $\hat{\mathcal{L}}$ the finite difference approximation to $\mathcal{L}$, u the continuous solution and $\hat{u}$ the numerical solution.

If we insert the solution to the continuous problem in the discrete problem we expect to find a residual, τ :

$$\tau = \hat{\mathcal{L}}(u) - \hat{f}.$$

This residual is commonly known as the *truncation error* of a numerical scheme and it mirrors how discrete equations are only approximations to continuous problems.

By making Taylor series expansions of u around the grid nodes we can express the truncation error as a function of the grid parameters. The behavior of the truncation error as the grid parameters tend to zero holds valuable information about the consistency of the scheme. If the truncation error of a numerical scheme tend to zero when the grid parameters approaches zero we have that the scheme solves the original partial differential equation in this limit. We will use this as a test to establish consistency for our numerical schemes.

The quality of a numerical approximation is also reflected through the truncation error, in many cases a small truncation error corresponds to a small error [45].

5.2.2 Stability

When solving a time dependent problem we must ensure that the time evolution of the numerical solution is physically plausible. There are several methods for finding the stability criterion of a numerical scheme. Here we shall focus on the traditional von Neumann stability analysis and analysis of the numerical dispersion relations of the problems to be solved.

For both methods the stability criterion will be given as a restriction on the grid parameters. In most cases it is convenient to assemble all the grid and physical parameters in one parameter, often referred to as the Courant number, C, of the scheme. The stability condition can then be expressed as a limiting value for the Courant number. For a scheme discretized in both time and space this means that we can use the chosen grid spacing in the spatial domain to find a good time discretization from the Courant number.

von Neumann stability analysis

The traditional von Neumann stability analysis focus on making the error in the numerical solution bounded in time and space.

If we for example have a partial differential equation, $\mathcal{L}(u) = f$, and a corresponding discrete problem, $\hat{\mathcal{L}}(\hat{u}) = \hat{f}$, we define the error in the numerical solution as $e = u - \hat{u}$. Inserting the error in the partial differential equation we find the error equation, $\mathcal{L}(e) = f$, which can be discretized as $\hat{\mathcal{L}}(\hat{e}) = \hat{f}$.

Assuming the discretized error, following the discretization introduced in Section 5.1, to take the form of a wave train,

$$e_j^l = e^{-i(\tilde{\omega} l \Delta t - kjh))} = \xi^l e^{ikjh},$$

we must require $|\xi| \leq 1$ to ensure that the error is bounded in time. After insertion of the discrete error in the discretized error equation we can use the requirement on ξ to find a stability condition limiting the grid parameters.

Numerical dispersion relations

It is also possible to derive the stability condition for a numerical scheme from the numerical methods ability to reproduce the qualitative properties of the continuous problem. This is done by considering the numerical dispersion relations of the problems to be solved, a method thoroughly presented by Langtangen [45].

As in the previous section we consider a partial differential equation $\mathcal{L}(u) = f$ and the corresponding discrete problem $\hat{\mathcal{L}}(\hat{u}) = \hat{f}$. We assume the solution of the continuous problem, u, to be in the form of a uniform wavetrain

$$u = Ae^{-i(\omega t - kx)}.$$

Considering the same discretization as for the von Neumann analysis the numerical solution

$$u_j^l = Ae^{-i(\tilde{\omega} t_l - kx_j)}$$

must then exhibit the same qualitative properties as the continuous solution if it is to be of any use. Here $\tilde{\omega}$ is the numerical dispersion relation and ω the analytical dispersion relation.

Substituting the two solutions into the partial differential equation and the numerical scheme, respectively, we can solve for the analytical and the numerical dispersion relations. By requiring the numerical dispersion relation to have the same physical properties as the analytical some requirements on the grid parameters are revealed, and these requirements are the stability criteria of the scheme.

An important observation from this analysis of the numerical dispersion relation and the previous von Neumann stability analysis is how comparison of particular solutions to analytical and numerical problems gave general information about the stability of the numerical schemes.

One should also note that these stability analysis are only valid for finite difference schemes for linear partial differential equations. Although an application of the linear stability analysis to nonlinear equations can thus not be justified, it is found to be effective in practice [47, 48]. We will therefore use these stability analysis also for nonlinear problems.

5.3 Basic concepts of the numerical solutions

Some procedures are common for all our numerical solutions, we discuss them briefly here.

5.3.1 Scaling and dimensionless variables

Initial-boundary value problems can, as we have seen in Chapter 2 and 4, depend on several physical parameters. By normalizing the problem and introducing dimensionless variables the number of physical parameters are reduced and thereby the numerical implementation greatly simplified. This is the reason why all the problems in this chapter will be normalized prior to finding numerical schemes for solving them.

In addition to the numerical benefits, normalization allow us to identify the relative size of the various terms in the partial differential equations. This will be helpful for the physical interpretation of the solutions.

5.3.2 Boundary conditions

We have no ambition to reproduce plasmas in their entire complexity, but aim to elucidate the physical processes of linear and finally nonlinear Langmuir wave propagation. Boundary conditions contain and control important physical processes and these processes could potentially obscure the wave phenomena that we are interested in. For all our numerical simulations we therefore choose to focus on a region in the plasma far from any boundaries.

Any plasma must be of spatial extent much larger than the Debye length, λ_{De}, see Chapter 1. To be well within this limit and avoid boundary-effects we shall assume the plasma to be an infinite system in all our applications. This infiniteness is simulated by periodic boundary conditions on a computational domain of length $L\lambda_{De}$ where L is a positive integer.

Usage of periodic boundary conditions means replicating the computational domain throughout space to make an infinite system. In other words we assume any information outside the computational domain to be inferred from the information within the domain. Nevertheless such schemes can only simulate phenomena on the scale of or smaller scales than the computational domain, i.e. for wave phenomena all wavelengths must be within the domain.

This infinite plasma model is a quite good approximation to space plasmas where relevant boundary conditions are mostly at least one earth radius away.

5.4 Linear Langmuir waves

Here we will develop finite difference solvers for the single electron fluid model and the multi-component electron fluid model descriptions of linear Langmuir waves in Chapter 2.

5.4.1 The single electron fluid model

In Section 2.1 we found an equation,

$$\frac{\partial^2}{\partial t^2}n - 3u_{th}^2\frac{\partial^2}{\partial x^2}n + \omega_{pe}^2 n = 0, \tag{5.1}$$

describing the linear Langmuir waves in a plasma where the electrons followed a Maxwell-Boltzmann distribution. Our aim is to solve this equation numerically on a computational domain of L Debye lengths and simulate a infinite system by periodic boundary conditions. We assume the initial electron density perturbation is released at time $t = 0$ and follow some function $f(x)$.

Before finding a finite difference scheme for solving (5.1) we normalize it and therefore introduce the scaled parameters $\bar{n} = \frac{n}{n_0}$, $\bar{x} = \frac{x}{\lambda_{De}}$ and $\bar{t} = t\omega_{pe}$, where, as before, n_0 is the equilibrium electron density, λ_{De} the Debye length, ω_{pe} the plasma frequency and u_{th} the thermal velocity of the electrons. Relation (5.1) and its initial and boundary values will then constitute the initial-boundary value problem

$$\frac{\partial^2}{\partial t^2}n - 3\frac{\partial^2}{\partial x^2}n + n = 0 \quad \text{for } x \in [0, L] \text{ and } t \geq 0 \tag{5.2}$$

$$n(0,t) = n(L,t) \quad \text{for } t \geq 0 \tag{5.3}$$

$$n(x,0) = f(x) \quad \text{for } x \in [0, L] \tag{5.4}$$

$$\frac{\partial}{\partial t}n(x,0) = 0 \quad \text{for } x \in [0, L]. \tag{5.5}$$

In the following we know that all the parameters are scaled, therefore we omit the bar notation explicitly emphasizing scaled parameters.

Discretizing (5.1) by centered differences in space and time as listed in Table 5.1 we obtain

$$n_j^{l+1} = C^2(n_{j+1}^l - 2n_j^l + n_{j-1}^l) + (2 - \Delta t^2)n_j^l - n_j^{l-1} \tag{5.6}$$

valid for $j = 1, 2, ..., m-1$ and $l = 0, 1, ...$, and where the Courant number $C = \sqrt{3}\Delta t/h$ has been introduced.

Problem; the last value, n_j^{l-1}, is outside our grid for the first time step, $l = 0$. This problem is solved by approximating the initial condition (5.5) by a two-sided difference in time as

$$\frac{\partial}{\partial t}n(x,0) \approx \frac{n_j^1 - n_j^{-1}}{2\Delta t} \quad \Rightarrow \quad n_j^1 = n_j^{-1}.$$

Inserting the relation $n_j^1 = n_j^{-1}$ valid for $l = 0$ in the scheme (5.6) we obtain an equation,

$$n_j^{-1} = \frac{C^2}{2}(n_{j+1}^0 - 2n_j^0 + n_{j-1}^0) + \frac{1}{2}(2 - \Delta t^2)n_j^0 \quad \text{for } j = 2, 3, ..., m-1,$$

enabling us to use (5.6) also for the first time step. We could alternatively have eliminated n_j^{-1} by the same technique and thereby obtained a special equation for the first timestep. however, when it comes to the implementation of the scheme we find it most practical to use the fictive value n_j^{-1} so that all timesteps can be computed in the same time loop.

After discretizing also the boundary conditions and the remaining initial condition the initial-boundary value problem (5.2)-(5.5) is transformed into the discretized problem

$$\begin{aligned}
&n_j^0 = f(jh) \quad \text{for } j = 1, 2, ..., m \\
&n_j^{-1} = \frac{C^2}{2}(n_{j+1}^0 - 2n_j^0 + n_{j-1}^0) + \frac{1}{2}(2 - \Delta t^2)n_j^0 \\
&\text{for } j = 2, 3, ..., m-1 \\
&n_0^{-1} = \frac{C^2}{2}(n_1^0 - 2n_0^0 + n_{m-1}^0) + \frac{1}{2}(2 - \Delta t^2)n_0^0 \\
&n_m^{-1} = n_0^{-1} \\
&n_j^{l+1} = C^2(n_{j+1}^l - 2n_j^l + n_{j-1}^l) + (2 - \Delta t^2)n_j^l - n_j^{l-1} \\
&\quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ... \\
&n_0^{l+1} = C^2(n_1^l - 2n_0^l + n_{m-1}^l) + (2 - \Delta t^2)n_0^l - n_0^{l-1} \quad \text{for } l = 0, 1, ... \\
&n_m^{l+1} = n_0^{l+1} \quad \text{for } l = 0, 1, ...
\end{aligned}$$

The truncation error for (5.6) is of order $\mathcal{O}(h^2, \Delta t^2)$ and fulfills the condition for consistency. Stability analysis with regards to the numerical dispersion relation gives the stability criterion $C \leq 2/\sqrt{4-h^2}$. Hence the discrete problem is convergent.

The values of the discrete electron density at each new timestep is computed from know values at lower timesteps. All numerical schemes with this simple structure are characterized as explicit. When compared to their counterpart, implicit schemes, where linear systems must be solved at each timestep, explicit schemes are by far the easiest to implement.

Implementation algorithm

The implementation algorithm for the discretized version of the initial-boundary value problem describing one-dimensional linear Langmuir waves in a single electron component plasma is as follows:

- Define np_j, n_j and nm_j to represent n_j^{l+1}, n_j^l and n_j^{l-1} respectively.
- Set the initial conditions and the artificial value n_j^{-1} and the corresponding periodic boundary conditions.
- While $t \leq tstop$
  - $t \leftarrow t + \Delta t$
  - Update all innerpoints according to the finite difference scheme
  - Set the periodic boundary conditions
  - Initialize for the next time step

$$\begin{aligned} nm_j &= n_j \\ n_j &= np_j \end{aligned}$$

  - Dump the solutions, n_j, to file

5.4.2 The multi-component electron fluid model

Section 2.3 introduced a set of equations,

$$\frac{\partial^2}{\partial t^2} n_r - 3u_{th,r}^2 \frac{\partial^2}{\partial x^2} n_r + \omega_{pe,r}^2 \sum_{r=1}^{N} n_r = 0 \quad \text{for } r = 1, 2, ..., N, \tag{5.7}$$

describing Langmuir wave evolution in a plasma with N electron components at different temperatures and densities. The thermal velocity and density of each component was found as

$$u_{th,r}^2 = r^2\frac{3\sigma^2}{N^2} \quad \text{and} \quad n_{r,0} = n_0\frac{2u_r}{\sqrt{2\pi\sigma^2}}\left[e^{-\frac{1}{2}\frac{u_{r-1}^2}{\sigma^2}} - e^{-\frac{1}{2}\frac{u_r^2}{\sigma^2}}\right],$$

respectively, where $\sigma^2 = \kappa T/m$. We aim to solve this set of equations numerically for the same initial conditions and periodic boundary conditions as we did for the single electron fluid model in the previous section.

In order to normalize (5.7) we introduce the dimensionless quantities $\bar{x} = \frac{x}{\lambda_{De}}$, $\bar{t} = t\omega_{pe}$ and $\bar{n}_{r,0} = \frac{n_{r,0}}{n_0}$, where n_0 is the sum of all the initial electron densities and λ_{De} and ω_{pe} the corresponding Debye length and plasma frequency. The resulting normalized equation becomes

$$\frac{\partial^2}{\partial t^2}n_r - \alpha(r)\frac{\partial^2}{\partial x^2}n_r + \beta(r)\sum_{r=1}^{N} n_r = 0 \quad \text{for } r = 1, 2, ..., N, \tag{5.8}$$

where

$$\beta(r) = \frac{\omega_{pe,r}^2}{\omega_{pe}^2} = \frac{n_{r,0}}{n_0} = \frac{\frac{n_{r,0}}{n_0}}{\sum_{r=1}^N \frac{n_{r,0}}{n_0}} = \frac{\frac{2u_r}{\sqrt{2\pi\sigma^2}}\left[e^{-\frac{1}{2}\frac{u_{r-1}^2}{\sigma^2}} - e^{-\frac{1}{2}\frac{u_r^2}{\sigma^2}}\right]}{\sum_{r=1}^N \frac{2u_r}{\sqrt{2\pi\sigma^2}}\left[e^{-\frac{1}{2}\frac{u_{r-1}^2}{\sigma^2}} - e^{-\frac{1}{2}\frac{u_r^2}{\sigma^2}}\right]}$$
$$= \frac{r\left[e^{-\frac{9}{2}\frac{(r-1)^2}{N^2}} - e^{-\frac{9}{2}\frac{r^2}{N^2}}\right]}{\sum_{r=1}^N r\left[e^{-\frac{9}{2}\frac{(r-1)^2}{N^2}} - e^{-\frac{9}{2}\frac{r^2}{N^2}}\right]}$$

and

$$\alpha(r) = \frac{3u_{th,r}^2}{\lambda_{De}^2\omega_{pe}^2} = \frac{3u_{th,r}^2}{u_{th}^2} = \frac{3u_{th,r}}{\sum_{s=1}^N \frac{n_s}{n_0}u_{th,s}^2} = \frac{r^2\frac{9\sigma^2}{N^2}}{\sum_{s=1}^N s^2\beta(s)\frac{3\sigma^2}{N^2}} = \frac{3r^2}{\sum_{s=1}^N s^2\beta(s)}.$$

Our box approximation to the Maxwellian does not always result in a velocity distribution that fulfills $\sum_{r=1}^N \int f_u(u_r)\,du = 1$. Therefore we have set $\beta(r) = \frac{n_r}{n_0}/\sum_{r=1}^N \frac{n_r}{n_0}$ so that the total unperturbed relative density, $\sum_{r=1}^N \frac{n_r}{n_0}$

or $\sum_{r=1}^{N}\beta(r)$, of the system is equal to unity. In the definition of $\alpha(r)$ we have used the total thermal velocity u_{th}, (2.15), for the electrons.

Finally the initial and boundary conditions will have the normalized form

$$\begin{aligned}
&n_r(0,t) = n_r(L,t) \quad \text{for } t \geq 0 \text{ and } r = 1,2,...,N\\
&n_r(x,0) = f(x) \quad \text{for } x \in [0,L] \text{ and } r = 1,2,...,N\\
&\frac{\partial}{\partial t} n_r(x,0) = 0 \quad \text{for } x \in [0,L] \text{ and } r = 1,2,...,N.
\end{aligned}$$

The finite difference scheme we developed for the single electron component model in the previous section is valid for each of the coupled equations in this multi-component model, although we must find the sum over all the electron densities at each timesteps. Combined with the initial conditions and the periodic boundary conditions the complete discretized initial-boundary value problem becomes

$$\begin{aligned}
n^0_{r,j} &= f(jh) \quad \text{for } j = 0,1,...,m\\
n^{-1}_{r,j} &= \alpha(r)\frac{C^2}{2}(n^0_{r,j+1} - 2n^0_{r,j} + n^0_{r,j-1}) + n^0_{r,j} - \frac{1}{2}\beta(r)\Delta t^2 \sum_{r=1}^{N} n^0_{r,j}\\
&\text{for } j = 1,3,...,m-1\\
n^{-1}_{r,0} &= \alpha(r)\frac{C^2}{2}(n^0_{r,1} - 2n^0_{r,0} + n^0_{r,m-1}) + n^0_{r,0} - \beta(r)\frac{\Delta t^2}{2} \sum_{r=1}^{N} n^0_{r,0}\\
n^{-1}_{r,m} &= n^{-1}_{r,0}\\
n^{l+1}_{r,j} &= \alpha(r)C^2(n^l_{r,j+1} - 2n^l_{r,j} + n^l_{r,j-1}) + 2n^l_{r,j} - n^{l-1}_{r,j} - \beta(r)\Delta t^2 \sum_{r=1}^{N} n^l_{r,j}\\
&\quad \text{for } j = 1,2,...,m-1 \text{ and } l = 0,1,...\\
n^{l+1}_{r,0} &= \alpha(r)C^2(n^l_{r,1} - 2n^l_{r,0} + n^l_{r,m-1}) + 2n^l_{r,0} - n^{l-1}_{r,0} - \beta(r)\Delta t^2 \sum_{r=1}^{N} n^l_{r,0}\\
&\quad \text{for } l = 0,1,...\\
n^{l+1}_{r,m} &= n^{l+1}_{r,0} \quad \text{for } l = 0,1,...
\end{aligned}$$

where $r = 1,2,...,N$. Here the stability criterion from section 5.4.1 becomes $C \leq 2/\sqrt{4\alpha(r) - \beta(r)h^2N}$ for each electron component when $C = \frac{\Delta t}{h}$. The most strict condition on C is for $\alpha(N)$ and $\beta(N)$ and we will use it as the stability criteria for all the components to ensure overall stability and identical timesteps for all computations.

The truncation error is, as for the single electron component model, of order $\mathcal{O}(h^2, \Delta t^2)$.

Implementation algorithm

This is a simple algorithm for implementing the finite difference scheme for linear Langmuir waves in the multi-component electron fluid model:

- Define $np_{r,j}$, $n_{r,j}$, $nm_{r,j}$, sum_n_j and sum_np_j to represent n_j^{l+1}, n_j^l, n_j^{l-1}, $\sum_{r=1}^{N} n_{r,j}^l$ and $\sum_{r=1}^{N} n_{r,j}^{l+1}$ respectively.
- for $r \leq N$
  Find the sum of all the initial electron density components multiplied with their component number:
  $$sum\_product\_n_0 \leftarrow sum\_product\_n_0 + r\left(e^{-\frac{9(r-1)^2}{2}} - e^{-\frac{9r^2}{2}}\right)$$
- for $r \leq N$
  - Find the coefficients $\beta(r)$
    $$\beta(r) = \frac{r\left[e^{\frac{-9(r-1)^2}{2}} - e^{\frac{-9r^2}{2}}\right]}{sum\_product\_n_0}$$
  - Find the sum of all products $r^2\beta(r)$
    $$sum\_product\_\beta = sum\_product\_\beta \leftarrow r^2\beta(r)$$
- for $r \leq N$
  Find the coefficients $\alpha(r)$
  $$\alpha(r) = \frac{3r^2}{sum\_product\_\beta}$$
- for $r \leq N$
  Set the initial conditions and find the sum of all the initial electron density perturbations over all the spatial grid points, $sum\_n_j \leftarrow sum\_n_j + n_j$
- for $r \leq N$
  Set the artificial value $n_{r,j}^{-1}$ and its corresponding boundary conditions

```
• While $t \leq tstop$
    – $t \leftarrow t + \Delta t$
    – for $r \leq N$
        ∗ update all innerpoints according to the finite difference scheme
          and set $sum\_np_j \leftarrow sum\_np_j + np_{r,j}$
        ∗ set the periodic boundary conditions and find the sum over
          all the endpoints, $sum\_np_0 \leftarrow sum\_np_0 + np_{r,0}$
        ∗ Initialize for the next time step
              $nm_{r,j} = n_{r,j}$
              $n_{r,j} = np_{r,j}$
        ∗ Dump the solutions, $n_{r,j}$, to file
    – Initialize the sum over all the $N$ electron density components for
      the next timestep
          $sum\_n_j = sum\_np_j$
          $sum\_np_j = 0$
```

5.4.3 Langmuir wave-envelope oscillations in a single electron component plasma

In section 2.4 we found that the slowly varying amplitude part, $n_s(x,t)$, of the electron density variations associated with Langmuir waves in a single electron component plasma is described by the linear Schrödinger equation

$$i\frac{\partial}{\partial t}n_s + \frac{3u_{th}^2}{2\omega_{pe}}\frac{\partial^2}{\partial x^2}n_s = 0. \tag{5.9}$$

We assume the initial electron density perturbation to follow some function $f(x)$ and aim to solve this initial-value problem for periodic boundary conditions on the same computational domain as we did for the full Langmuir wave solution.

First we normalize the initial-boundary value problem by introducing the dimensionless variables $\bar{x} = \frac{x}{\lambda_{De}}$, $\bar{t} = t\omega_{pe}$ and $\bar{n} = \frac{n_s}{n_0}$ where n_0 is the unperturbed electron density. We obtain

$$i\frac{\partial}{\partial t}n + \frac{3}{2}\frac{\partial^2}{\partial x^2}n = 0 \quad \text{for } x \in [0, L] \text{ and } t \geq 0 \tag{5.10}$$
$$n(0, t) = n(L, t) \quad \text{for } t \geq 0 \tag{5.11}$$
$$n(x, 0) = f(x) \quad \text{for } x \in [0, L], \tag{5.12}$$

where we for notational simplicity have omitted the bar notation and from now on know that all quantities are dimensionless.

The electron density amplitude variation is a complex quantity, $n(x, t) = a(x, t)+ib(x, t)$, and introducing the notation of Herbst, Mitchell and Weideman, [49], we rewrite (5.10) as the real system

$$\frac{\partial}{\partial t}\boldsymbol{n} + \frac{3}{2}A\frac{\partial^2}{\partial x^2}\boldsymbol{n} = 0 \quad \text{for } x \in [0, L] \text{ and } t \geq 0 \tag{5.13}$$

where

$$\boldsymbol{n} = \begin{pmatrix} a \\ b \end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

Discretizing (5.13) by a one sided forward difference in time and a centered difference in space we find the explicit forward Euler scheme

$$\left[\delta_t^+\boldsymbol{n} + \frac{3}{2}A\delta_x\delta_x\boldsymbol{n} = 0\right]_j^l \quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ... \tag{5.14}$$

written here in the compact notation from Table 5.1. Unfortunately a von Neumann stability analysis shows that this scheme will not be stable for any choice of grid parameters h and Δt.

The alternative to the forward difference in time is to use a one sided backward difference in time leading to the backward Euler scheme

$$\left[\delta_t^-\boldsymbol{n} + \frac{3}{2}A\delta_x\delta_x\boldsymbol{n} = 0\right]_j^l \quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ...$$

written out as

$$\boldsymbol{n}_j^l - C(\boldsymbol{n}_{j+1}^l - 2\boldsymbol{n}_j^l + \boldsymbol{n}_{j-1}^l) = \boldsymbol{n}_j^{l-1} \qquad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ...,$$

where $C = \frac{3\Delta t}{2h^2}$ is the Courant number. von Neumann stability analysis show that this scheme is stable for any choice of grid parameters Δt and h, which is a very attractive feature. However, solving for the slowly varying part of the electron density, $\boldsymbol{n}$, with this scheme would require solving a system of linear equations for each timestep. Numerical schemes with this property are termed implicit as they do not yield an explicit updating formula for the value of the unknown at each grid point for each new timestep. Implicit schemes are more difficult to implement than explicit schemes. For some problems they can also be computationally more expensive because of the arithmetic operations connected to solving linear systems.

To avoid solving the implicit scheme we argue that if we use the scheme (5.14) only for the first timestep instabilities will not have time to develop. By doing so we can for all further timesteps use a two sided difference in time and thereby the explicit scheme

$$\left[\delta_{2t}\boldsymbol{n} + \frac{3}{2}A\delta_x\delta_x\boldsymbol{n} = 0\right]_j^l \quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ...$$

Numerical schemes, like this one, that are based on centered time differences over two time intervals are often called *Leap-Frog* schemes. Evaluating the numerical dispersion relation of this Leap-Frog scheme we find the stability condition as $C \leq \frac{1}{4}$. The truncation error will be of order $\mathcal{O}(h^2, \Delta t^2)$ and we have by definition a consistent scheme, and thus also a convergent scheme. The truncation error of (5.14) is of order $\mathcal{O}(h^2, \Delta t)$, the Leap-Frog scheme is therefore most probably more accurate.

All in all this approach leads to the following set of discrete equations approximating the initial-boundary value problem (5.10)-(5.12):

- *Initial conditions*

$$\begin{aligned} a_j^0 &= f_a(hj) \quad \text{for } j = 1, 2, ..., m \\ b_j^0 &= f_b(hj) \quad \text{for } j = 1, 2, ..., m \end{aligned}$$

- *The one-sided forward scheme and the corresponding periodic boundary conditions*

$$
\begin{aligned}
a_j^1 &= -C(b_{j+1}^0 - 2b_j^0 + b_{j-1}^0) + a_j^0 \quad \text{for } 1 < j < m \\
b_j^1 &= C(a_{j+1}^0 - 2a_j^0 + a_{j-1}^0) + b_j^0 \quad \text{for } 1 < j < m \\
a_0^1 &= -C(b_1^0 - 2b_0^0 + b_{m-1}^0) + a_0^0 \\
a_m^1 &= a_0^1 \\
b_0^1 &= C(a_1^0 - 2a_0^0 + a_{m-1}^0) + b_0^0 \\
b_m^1 &= b_0^1
\end{aligned}
$$

- *The Leap-Frog scheme and the corresponding periodic boundary conditions*

$$
\begin{aligned}
a_j^{l+1} &= -2C(b_{j+1}^l - 2b_j^l + b_{j-1}^l) + a_j^{l-1} \quad \text{for } 1 < j < m \text{ and } l > 1 \\
b_j^{l+1} &= 2C(a_{j+1}^l - 2a_j^l + a_{j-1}^l) + b_j^{l-1} \quad \text{for } 1 < j < m \text{ and } l > 1 \\
a_0^{l+1} &= -2C(b_1^l - 2b_0^l + b_{m-1}^l) + a_0^{l-1} \quad \text{for } l > 1 \\
a_m^{l+1} &= a_0^{l+1} \quad \text{for } l > 1 \\
b_0^{l+1} &= 2C(a_1^l - 2a_0^l + a_{m-1}^l) + b_0^{l-1} \quad \text{for } l > 1 \\
b_m^{l+1} &= b_0^{l+1} \quad \text{for } l > 1
\end{aligned}
$$

Implementation algorithm

Here follows a simple implementation algorithm for the finite difference scheme listed above:

- Define ap_j, bp_j, a_j, b_j, am_j and bm_j to represent a_j^{l+1}, b_j^{l+1}, a_j^l, b_j^l, a_j^{l-1} and b_j^{l-1} respectively.
- Set the initial conditions
- $t = t + \Delta t$
- Find the first time step according to the one-sided forward scheme and set the corresponding boundary conditions
- Update the data structures for the next time step

$$
\begin{aligned}
am_j &= a_j \\
a_j &= ap_j \\
bm_j &= b_j \\
b_j &= bp_j
\end{aligned}
$$

- While $t \leq tstop$
  - $t \leftarrow t + \Delta t$
  - Update all innerpoints according to the Leap-Frog scheme
  - Set the periodic boundary conditions
  - Initialize for the next time step

$$\begin{aligned} am_j &= a_j \\ a_j &= ap_j \\ bm_j &= b_j \\ b_j &= bp_j \end{aligned}$$

  - Dump the solutions, a_j and b_j, to file

5.4.4 Langmuir wave-envelope oscillations in a multi-electron component plasma

The low frequency amplitude part of Langmuir waves in a multi-component fluid model are described by a set of coupled linear Schrödinger equations,

$$i\frac{\partial}{\partial t}n_{r,s} + \frac{3u_{th,r}^2}{2\omega_{pe}}\frac{\partial^2}{\partial x^2}n_{r,s} + \frac{\omega_{pe}}{2}n_{r,s} - \frac{\omega_{pe,r}^2}{2\omega_{pe}}\sum_{r=1}^{N} n_{r,s} = 0 \quad \text{for } r = 1, 2, ..., N,$$

where N is the number of electron components, $u_{th,r}^2 = \frac{1}{3}u_r^2$ and $\omega_{pe,r} = \sqrt{e^2 n_{r,0}/\epsilon_0 m}$ for $u_r = r3\sigma/N$ and $n_{r,0} = n_0\frac{2u_r}{\sqrt{2\pi\sigma^2}}\left[e^{-\frac{1}{2}\frac{u_{r-1}^2}{\sigma^2}} - e^{-\frac{1}{2}\frac{u_r^2}{\sigma^2}}\right]$. We want to solve this set of equations numerically for the same inital and boundary conditions as for the low frequency wave-enevlope in a single electron component plasma.

First we introduce the dimensionless parameters $\bar{x} = \frac{x}{\lambda_{De}}$, $\bar{t} = t\omega_{pe}$ and $\bar{n}_r = \frac{n_{r,s}}{n_0}$, and also, a part from a factor $\frac{1}{2}$, the coefficients $\alpha(r)$ and $\beta(r)$ from section 5.4.2, i.e.

$$\beta(r) = \frac{r\left[e^{-\frac{9}{2}\frac{(r-1)^2}{N^2}} - e^{-\frac{9}{2}\frac{r^2}{N^2}}\right]}{2\sum_{r=1}^{N} r\left[e^{-\frac{9}{2}\frac{(r-1)^2}{N^2}} - e^{-\frac{9}{2}\frac{r^2}{N^2}}\right]} \quad \text{and} \quad \alpha(r) = \frac{3r^2}{2\sum_{s=1}^{N} s^2\beta(s)}.$$

From this we obtain the normalized initial-boundary value problem

$$i\frac{\partial}{\partial t}n_r + \alpha(r)\frac{\partial^2}{\partial x^2}n_r + \frac{1}{2}n_r - \beta(r)\sum_{r=1}^{N} n_r = 0 \text{ for } x \in [0,L] \text{ and } t \geq 0 \quad (5.15)$$

$$n_r(0,t) = n_r(L,t) \quad \text{for } t \geq 0 \quad (5.16)$$

$$n_r(x,0) = f_r(x) \quad \text{for } x \in [0,L], \quad (5.17)$$

for $r = 1, 2, ..., N$. Where, as in all the previous examples, the bar notation is omitted for simplicity.

The electron density perturbations can be expressed as $n_r = a_r(x,t) + ib_r(x,t)$ and we rewrite (5.15) as the real system

$$\frac{\partial}{\partial t}\boldsymbol{n_r} + \alpha(r)A\frac{\partial^2}{\partial x^2}\boldsymbol{n_r} + \frac{1}{2}A\boldsymbol{n_r} - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r} = 0 \quad (5.18)$$

for $r = 1, 2, ..., N$, $x \in [0,L]$ and $t \geq 0$, in the same notation as for the single component model, that is

$$\boldsymbol{n_r} = \begin{pmatrix} a_r \\ b_r \end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

To obtain a numerical scheme for solving (5.18) we use a centered finite difference to approximate the spatial derivative, but which approximation to use for the time derivative must be considered more carefully. A one-sided forward finite difference in time will, as for the single-component model, lead to an unstable scheme for all choices of grid parameters, whereas a one-sided backward finite difference gives an implicit scheme. We will therefore follow the same approach as for the single electron component model of Langmuir-envelopes and use the one-sided forward finite time difference giving the scheme

$$[\delta_t^+\boldsymbol{n_r} + \alpha(r)A\delta_x\delta_x\boldsymbol{n_r} + \frac{1}{2}A\boldsymbol{n_r} - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r} = 0]_j^l$$
$$\text{for } r = 1, 2, ..., N,\ 1 < j < m \text{ and } l = 0$$

for the first timestep and a two sided finite time difference resulting in the scheme

$$[\delta_{2t}\boldsymbol{n_r} + \alpha(r)A\delta_x\delta_x\boldsymbol{n_r} + \frac{1}{2}A\boldsymbol{n_r} - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r} = 0]_j^l$$
$$\text{for } r = 1,2,...,N,\ 1 < j < m \text{ and } l > 0$$

for all further timesteps. Whereas the scheme with the one-sided finite difference in time is unstable the Leap-Frog scheme has the stability criterion $C \leq \left(4\alpha(r) + Nh^2\beta(r) - \frac{h^2}{2}\right)^{-1}$ for $C = \Delta t/h^2$, found by evaluating the numerical dispersion relation of the scheme. This stability criteria will be most strict for $r = N$. To ensure equal timesteps and overall stability we therefore apply the stability criterion $C \leq \left(4\alpha(N) + Nh^2\beta(N) - \frac{h^2}{2}\right)^{-1}$ for all N electron components. The truncation error of the Leap-Frog scheme is of order $\mathcal{O}(h^2, \Delta t^2)$, so the scheme is consistent.

Our numerical approximation to the initial-boundary value problem (5.15)-(5.17) is not very different from the one we used for the corresponding single-component problem. However, we are required to sum over all the electron density components at each time level and the set of discrete equations to be solved becomes:

- *Initial conditions*

$$\begin{aligned} a_{r,j}^0 &= f_{r,a}(hj) \quad \text{for } j = 1,2,...,m \\ b_{r,j}^0 &= f_{r,b}(hj) \quad \text{for } j = 1,2,...,m \end{aligned}$$

- *The one-sided forward scheme and the corresponding periodic boundary conditions*

$$a^1_{r,j} = -\alpha(r)C(b^0_{r,j+1} - 2b^0_{r,j} + b^0_{r,j-1}) - \Delta t\frac{1}{2}b_{r,j} + \beta(r)\Delta t\sum_{r=1}^{N} b^0_{r,j} + a^0_{r,j}$$

for $1 < j < m$

$$b^1_{r,j} = \alpha(r)C(a^0_{r,j+1} - 2a^0_{r,j} + a^0_{r,j-1}) + \Delta t\frac{1}{2}a_{r,j} - \beta(r)\Delta t\sum_{r,j}^{N} a^0_{r,j} + b^0_{r,j}$$

for $1 < j < m$

$$a^1_{r,0} = -\alpha(r)C(b^0_{r,1} - 2b^0_{r,0} + b^0_{r,m-1}) - \Delta t\frac{1}{2}b_{r,0} + \beta(r)\Delta t\sum_{r,j}^{N} b^0_{r,0} + a^0_{r,0}$$

$$a^1_{r,m} = a^1_{r,0}$$

$$b^1_{r,0} = \alpha(r)C(a^0_{r,1} - 2a^0_{r,0} + a^0_{r,m-1}) + \Delta t\frac{1}{2}a_{r,0} - \beta\Delta t\sum_{r=1}^{N} a^0_{r,0} + b^0_{r,0}$$

$$b^1_{r,m} = b^1_{r,0}$$

- *The Leap-Frog scheme and the corresponding periodic boundary conditions*

$$a^{l+1}_{r,j} = -2\alpha(r)C(b^l_{r,j+1} - 2b^l_{r,j} + b^l_{r,j-1}) - \Delta t b_{r,j} + 2\beta(r)\Delta t\sum_{r=1}^{N} b^l_{r,j} + a^{l-1}_{r,j}$$

for $1 < j < m$ and $l > 1$

$$b^{l+1}_{r,j} = 2\alpha(r)C(a^l_{r,j+1} - 2a^l_{r,j} + a^l_{r,j-1}) + \Delta t a_{r,j} - 2\beta(r)\Delta t\sum_{r=1}^{N} a^l_{r,j} + b^{l-1}_{r,j}$$

for $1 < j < m$ and $l > 1$

$$a^{l+1}_{r,0} = -2\alpha(r)C(b^l_{r,1} - 2b^l_{r,0} + b^l_{r,m-1}) - \Delta t b_{r,0} + 2\beta(r)\Delta t\sum_{r=1}^{N} b^l_{r,0} + a^{l-1}_{r,0}$$

for $l > 1$

$a^{l+1}_{r,m} = a^{l+1}_{r,0}$ for $l > 1$

$$b^{l+1}_{r,0} = 2\alpha(r)C(a^l_{r,1} - 2a^l_{r,0} + a^l_{r,m-1}) + \Delta t a_{r,0} - 2\beta(r)\Delta t\sum_{r=1}^{N} a^l_{r,0} + b^{l-1}_{r,0}$$

for $l > 1$

$b^{l+1}_{r,m} = b^{l+1}_{r,0}$ for $l > 1$

all valid for $r = 1, 2, ..., N$.

Implementation algorithm

In order to implement the finite difference scheme for the initial-boundary value problem (5.15)-(5.17) we will use the following algorithm:

- Define $ap_{r,j}$, $bp_{r,j}$, $a_{r,j}$, $b_{r,j}$, $am_{r,j}$, $bm_{r,j}$, sum_a_j, sum_b_j, sum_ap_j and sum_bp_j to represent $a_{r,j}^{l+1}$, $b_{r,j}^{l+1}$, $a_{r,j}^{l}$, $b_{r,j}^{l}$, $a_{r,j}^{l-1}$, $b_{r,j}^{l-1}$, $\sum_{r=1}^{N} a_{r,j}^{l}$, $\sum_{r=1}^{N} b_{r,j}^{l}$, $\sum_{r=1}^{N} a_{r,j}^{l+1}$ and $\sum_{r=1}^{N} b_{r,j}^{l+1}$ respectively.
- for $r \leq N$

  Find the sum of all the initial electron density components multiplied with their component number:

$$sum\_product\_n_0 \leftarrow sum\_product\_n_0 + r\left[e^{-\frac{9(r-1)^2}{2}} - e^{-\frac{9r^2}{2}}\right]$$

- for $r \leq N$
  - Find the coefficients $\beta(r)$

$$\beta(r) = \frac{r\left[e^{-\frac{9(r-1)^2}{2}} - e^{-\frac{9r^2}{2}}\right]}{2sum\_product\_n_0}$$

  - Find the sum of all products $r^2\beta(r)$

$$sum\_product\_\beta = sum\_product\_\beta \leftarrow r^2 beta(r)$$

- for $r \leq N$
  Find the coefficients $\alpha(r)$

$$\alpha(r) = \frac{3r^2}{2sum\_product\_\beta}$$

- for $r \leq N$
  Set the initial conditions and find the sum of all the initial perturbations over all the spatial grid points, $sum\_a_j \leftarrow sum\_a_j + a_{r,j}$ and $sum\_b_j \leftarrow sum\_b_j + b_{r,j}$

- $t = t + \Delta t$
- for $r \leq N$

    Find the first time step according to the one-sided forward scheme, set the corresponding boundary conditions and the sum off all the perturbations at this timestep, that is $sum\_ap_j \leftarrow sum\_ap_j + ap_{r,j}$ and $sum\_bp_j \leftarrow sum\_bp_j + bp_{r,j}$

    - Update the data structures for the next timestep

    $$\begin{aligned} am_{r,j} &= a_{r,j} \\ a_{r,j} &= ap_{r,j} \\ bm_{r,j} &= b_{r,j} \\ b_{r,j} &= bp_{r,j} \end{aligned}$$

- Initialize the sums over all the N electron density components for the next timestep

    $$\begin{aligned} sum\_ap_j &= 0 \\ sum\_bp_j &= 0 \end{aligned}$$

- While $t \leq tstop$
    - $t \leftarrow t + \Delta t$
    - for $r \leq N$
        * Update all innerpoints according to the Leap-Frog scheme
        * Set the periodic boundary conditions
        * Initialize for the next time step

        $$\begin{aligned} am_{r,j} &= a_{r,j} \\ a_{r,j} &= ap_{r,j} \\ bm_{r,j} &= b_{r,j} \\ b_{r,j} &= bp_{r,j} \end{aligned}$$

        * Dump the solutions, a_j and b_j, to file
    - Update the sums for the next timestep

    $$\begin{aligned} sum\_ap_j &= 0 \\ sum\_bp_j &= 0 \end{aligned}$$

5.5 Nonlinear Langmuir waves

This section concerns the numerical solutions of the nonlinear partial differential equations, associated with nonlinear Langmuir waves, that we derived in Chapter 4.

5.5.1 The electrostatic field in a plasma with an inhomogeneous electron density distribution

For a plasma with a prescribed inhomogeneous electron density we found in section 4.4 the relation

$$\frac{\partial^2}{\partial t^2}\tilde{E} - 3u_{th}^2\frac{\partial^2}{\partial x^2}\tilde{E} + \omega_{p0}^2\tilde{E} + \frac{\bar{n}}{n_0}\omega_{p0}^2\tilde{E} = 0 \qquad (5.19)$$

describing the electrostatic field perturbation $\tilde{E}$ connected to Langmuir waves. In the absence of Langmuir waves the electron density is given as $n = n_0 + \bar{n}(x)$, and the plasma frequency ω_{p0} relates to the constant part, n_0, of the density. The spatial variation $\bar{n}(x)$ follows some prescribed function of x and (5.19) is mathematically similar to the Langmuir wave equation (2.4) except for the nonlinearity introduced in the last term by $\bar{n}(x)$.

We found a finite difference scheme for solving (2.4) in section 5.4.1 and the same scheme is applicable for the normalized version,

$$\frac{\partial^2}{\partial t^2}E - 3\frac{\partial^2}{\partial x^2}E + \left(1 + \frac{\bar{n}}{n_0}\right)E = 0 \quad \text{for } x \in [0, L] \text{ and } t \geq 0,$$

of (5.19) on a computational domain of L Debye lengths when we assume the following boundary- and initial- conditions:

$$\begin{aligned} &E(0,t) = E(L,t) \quad \text{for } t \geq 0 \\ &E(x,0) = f(x) \quad \text{for } x \in [0,L] \\ &\frac{\partial}{\partial t}E(x,0) = 0 \quad \text{for } x \in [0,L]. \end{aligned}$$

The normalization was executed by introducing the dimensionless parameters $E = \frac{\tilde{E}}{E_0}$, $\bar{x} = \frac{x}{\lambda_{De}}$ and $\bar{t} = t\omega_{p0}$, and thereafter renaming $\bar{x}$ and $\bar{t}$ as x and t.

Here the scheme based on centered differences in time and space,

$$[\delta_t\delta_t E - 3\delta_x\delta_x E + \left(1+\frac{\bar{n}}{n_0}\right)E = 0]_j^l,$$

results in the discretized set of equations

$$\begin{aligned}
&E_j^0 = f(jh) \quad \text{for } j = 0,1,...,m\\
&E_j^{-1} = \frac{C^2}{2}(E_{j+1}^0 - 2E_j^0 + E_{j-1}^0) + \frac{1}{2}\left(2-\Delta t^2\left(1+\frac{\bar{n}}{n_0}\right)\right)E_j^0\\
&\text{for } j = 2,3,...,m-1\\
&E_0^{-1} = \frac{C^2}{2}(E_1^0 - 2E_0^0 + E_{m-1}^0) + \frac{1}{2}\left(2-\Delta t^2\left(1+\frac{\bar{n}}{n_0}\right)\right)E_0^0\\
&E_m^{-1} = E_0^{-1}\\
&E_j^{l+1} = C^2(E_{j+1}^l - 2E_j^l + E_{j-1}^l) + \left(2-\Delta t^2\left(1+\frac{\bar{n}}{n_0}\right)\right)E_j^l - E_j^{l-1}\\
&\quad \text{for } j = 2,3,...,m-1 \text{ and } l = 0,1,...\\
&E_0^{l+1} = C^2(E_1^l - 2E_0^l + E_{m-1}^l) + \left(2-\Delta t^2\left(1+\frac{\bar{n}}{n_0}\right)\right)E_0^l - E_0^{l-1} \quad \text{for } l = 0,1,...\\
&E_m^{l+1} = E_0^{l+1} \quad \text{for } l = 0,1,....
\end{aligned}$$

where $C = \sqrt{3}\Delta t/h$ is the Courant number. By evaluating the numerical dispersion relation of the scheme we find the stability criterion $C \leq 2/\sqrt{4-h^2}$. The truncation error will be of order $\mathcal{O}(h, \Delta t)$.

The implementation algorithm for this discrete problem is identical to the one for Langmuir waves in the single electron component model in section 5.4.1.

5.5.2 The nonlinear Schrödinger equation

The nonlinear Schrödinger equation

$$i\frac{\partial}{\partial t}\bar{E} + \frac{3u_{th}^2}{2\omega_{pe}}\frac{\partial^2}{\partial x^2}\bar{E} + \frac{\omega_{pe}}{4}\frac{\epsilon_0}{n_0\kappa T} \mid \bar{E} \mid^2 \bar{E} = 0,$$

as presented in section 4.5 describes the slowly varying part of the electric field perturbation, $\bar{E}(x,t)$, associated with a Langmuir wave travelling with an inhomogeneous amplitude. We are to solve this equation, as in the previous sections, on a computational domain of length $L\lambda_{De}$ for periodic boundary conditions when $\bar{E}(x,t)$ initially follows a function $f(x)$.

Introducing the scaled quantities $E = \bar{E}/E_0$, $\bar{x} = x/\lambda_{De}$ and $\bar{t} = t\omega_{pe}$ where E_0 is a reference electric field amplitude, λ_{De} the Debye length and ω_{pe} the plasma frequency, and also the dimensionless parameter $\beta = \varepsilon_0 E_0^2/4n_0\kappa T$, we rewrite the Nonlinear Schrödinger equation and the boundary and initial conditions on normalized form as

$$
\begin{aligned}
& i\frac{\partial}{\partial t}E + \frac{3}{2}\frac{\partial^2}{\partial x^2}E + \beta \mid E \mid^2 E = 0 \quad \text{for } x \in [0,L] \text{ and } t \geq 0 && (5.20)\\
& E(0,t) = E(L,t) \quad \text{for } t \geq 0 && (5.21)\\
& E(x,0) = f(x) \quad \text{for } x \in [0,L], && (5.22)
\end{aligned}
$$

renaming $\bar{x}$ as x and $\bar{t}$ as t for notational simplicity.

As any complex quantity, the electric field can be expressed in terms of a real and an imaginary part as $E(x,t) = a(x,t) + ib(x,t)$. In order to solve (5.20) numerically we follow the notation of Herbst, Mitchell and Weideman, [49], and obtain the real system

$$
\frac{\partial}{\partial x}\boldsymbol{E} + \frac{3}{2}A\frac{\partial^2}{\partial x^2}\boldsymbol{E} + \beta\boldsymbol{f}(\boldsymbol{E}) = 0 \quad \text{for } x \in [0,L] \text{ and } t \geq 0 \tag{5.23}
$$

where

$$
\boldsymbol{E} = \begin{pmatrix} a \\ b \end{pmatrix}, \quad A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad \text{and} \quad \boldsymbol{f}(\boldsymbol{E}) = (\boldsymbol{E}^T\boldsymbol{E})A\boldsymbol{E}.
$$

When it comes to developing a finite difference scheme for (5.23) we find, as for the linear Schrödinger equation in section 5.4.3, that discretization by a one sided forward difference in time and a centered difference in space leads to an explicit scheme that is unstable for all grid parameters h and Δt whereas a one sided backward difference in time gives an implicit scheme that is stable for all choices of h and Δt. We avoided the implicit scheme for the solution of the linear Schrödinger equation by using the one sided forward difference in time for the first timestep and thereafter a two sided difference in time for all further timesteps. The same method with the scheme

$$
\left[\delta_t^+\boldsymbol{E} + \frac{3}{2}A\delta_x\delta_x\boldsymbol{E} + \beta\boldsymbol{f}(\boldsymbol{E}) = 0\right]_j^l \quad \text{for } j = 2,3,...,m-1 \text{ and } l = 0,1,... \tag{5.24}
$$

for the first timestep and the Leap-Frog scheme

$$\left[\delta_{2t}\boldsymbol{E} + \frac{3}{2}A\delta_x\delta_x\boldsymbol{E} + \beta\boldsymbol{f}(\boldsymbol{E}) = 0\right]_j^l \quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ...,$$

for all other timesteps will be our approach to solving (5.23) numerically. We thus argue that if we use the unstable scheme (5.24) only for the first timestep there will be no significant accumulation of round off errors.

Evaluating the numerical dispersion relation of the Leap-Frog scheme we find the stability condition as $C \leq 1/(6+h^2\beta)$ when $C = \Delta t/h^2$. The truncation error will be of order $\mathcal{O}(h, \Delta t)$ and fulfills the consistency requirements for finite difference schemes.

All in all this approach leads to the following set of discrete equations approximating the initial-boundary value problem (5.20)-(5.22):

- *Initial conditions*

$$\begin{aligned} a_j^0 &= f_a(hj) \quad \text{for } j = 1, 2, ..., m \\ b_j^0 &= f_b(hj) \quad \text{for } j = 1, 2, ..., m \end{aligned}$$

- *The one sided forward scheme and the corresponding periodic boundary conditions*

$$\begin{aligned} a_j^1 &= -\frac{3}{2}C(b_{j+1}^0 - 2b_j^0 + b_{j-1}^0) - \beta\Delta t((a_j^0)^2 + (b_j^0)^2)b_j^0 + a_j^0 \quad \text{for } 1 < j < m \\ b_j^1 &= \frac{3}{2}C(a_{j+1}^0 - 2a_j^0 + a_{j-1}^0) + \beta\Delta t((a_j^0)^2 + (b_j^0)^2)a_j^0 + b_j^0 \quad \text{for } 1 < j < m \\ a_0^1 &= -\frac{3}{2}C(b_1^0 - 2b_0^0 + b_{m-1}^0) - \beta\Delta t((a_0^0)^2 + (b_0^0)^2)b_0^0 + a_0^0 \\ a_m^1 &= a_0^1 \\ b_0^1 &= \frac{3}{2}C(a_1^0 - 2a_0^0 + a_{m-1}^0) + \beta\Delta t((a_0^0)^2 + (b_0^0)^2)a_0^0 + b_0^0 \\ b_m^1 &= b_0^1 \end{aligned}$$

- *The Leap-Frog scheme and the corresponding periodic boundary conditions*

$$
\begin{aligned}
&a_j^{l+1} = -3C(b_{j+1}^l - 2b_j^l + b_{j-1}^l) - 2\beta\Delta t((a_j^l)^2 + (b_j^l)^2)b_j^l + a_j^{l-1} \\
&\quad \text{for } 1 < j < m \text{ and } l > 1 \\
&b_j^{l+1} = 3C(a_{j+1}^l - 2a_j^l + a_{j-1}^l) + 2\beta\Delta((a_j^l)^2 + (b_j^l)^2)a_j^l + b_j^{l-1} \\
&\quad \text{for } 1 < j < m \text{ and } l > 1 \\
&a_0^{l+1} = -3C(b_1^l - 2b_0^l + b_{m-1}^l) - 2\beta\Delta t((a_0^l)^2 + (b_0^l)^2)b_0^l + a_0^{l-1} \quad \text{for } l > 1 \\
&a_m^{l+1} = a_0^{l+1} \quad \text{for } l > 1 \\
&b_0^{l+1} = 3C(a_1^l - 2a_0^l + a_{m-1}^l) + 2\beta\Delta t((a_0^l)^2 + (b_0^l)^2)a_0^l + b_0^{l-1} \quad \text{for } l > 1 \\
&b_m^{l+1} = b_0^{l+1} \quad \text{for } l > 1
\end{aligned}
$$

Implementation algorithm

Our finite difference approximation to the initial-boundary value problem (5.20)-(5.22) for the nonlinear Schrödinger equation describing the slowly varying part of an electrostatic field with an inhomogeneous amplitude can be implemented numerically by the following simple algorithm:

- Define ap_j, bp_j, a_j, b_j, am_j and bm_j to represent a_j^{l+1}, b_j^{l+1}, a_j^l, b_j^l, a_j^{l-1} and b_j^{l-1} respectively.
- Set the initial conditions
- $t = t + \Delta t$
- Find the first time step according to the one-sided forward scheme and set the corresponding boundary conditions
- Update the data structures for the next time step

$$
\begin{aligned}
am_j &= a_j \\
a_j &= ap_j \\
bm_j &= b_j \\
b_j &= bp_j
\end{aligned}
$$

- While $t \leq tstop$
  - $t \leftarrow t + \Delta t$
  - Update all innerpoints according to the Leap-Frog scheme
  - Set the periodic boundary conditions

- Initialize for the next time step

$$
\begin{aligned}
am_j &= a_j \\
a_j &= ap_j \\
bm_j &= b_j \\
b_j &= bp_j
\end{aligned}
$$

- Dump the solutions, a_j and b_j, to file

Conservation of energy

The energy density, u_W, in electromagnetic waves is given as, [40],

$$u_W = \frac{1}{2}\epsilon_0 E^2.$$

As we have set $E(x,t) = a(x,t) + ib(x,t)$ we must have $u_W \propto (a^2 + b^2)$. We have previously, in section 4.5.2, used W for the total energy of a compact waveform.

Writing out the imaginary and real part of the nonlinear Schrödinger equation (5.20) and multiplying them with a and b, respectively, we obtain

$$
\begin{aligned}
\frac{1}{2}\frac{\partial}{\partial t}a^2 + \frac{3}{2}a\frac{\partial^2}{\partial x^2}b - \frac{1}{4}(a^2+b^2)ab = 0 \\
\frac{1}{2}\frac{\partial}{\partial t}b^2 - \frac{3}{2}b\frac{\partial^2}{\partial x^2}a + \frac{1}{4}(a^2+b^2)ab = 0.
\end{aligned}
$$

These two equations are combined into

$$\frac{1}{2}\frac{\partial}{\partial t}(a^2+b^2) + \frac{3}{2}\left(a\frac{\partial^2}{\partial x^2}b - b\frac{\partial^2}{\partial x^2}a\right) = 0,$$

and integrating over the spatial domain $[0, L]$ we find

$$\frac{\partial}{\partial t}\int_0^L (a^2+b^2)\,dx = 0.$$

This means that the quantity (a^2+b^2) is conserved over the spatial domain for all times, or in other words the principle of conservation of energy applies. It is no surprise that conservation of energy is valid for these Langmuir waves, after all we assumed the plasma to be isothermal for these low frequency oscillations and it is a closed system.

The numerical scheme we have developed to solve the nonlinear Schrödinger equation should give a solution that conserves the energy in the same manner as we have just established. In order to investigate if this is so we will numerically integrate the solutions a_j^l and b_j^l over the spatial domain for several timesteps l. Our chosen numerical integration rule is the trapezoidal rule,

$$\int_0^L ((a_j^l)^2 + (b_j^l)^2)\, dx \approx \frac{h}{2}((a_1^l)^2 + (b_1^l)^2) + \sum_{j=2}^{m-1} h((a_j^l)^2 + (b_j^l)^2) + \frac{h}{2}((a_m^l)^2 + (b_m^l)^2),$$

with an error of order $\mathcal{O}(h^2)$, [45]. We expect the value of this integral to be the same at all timesteps.

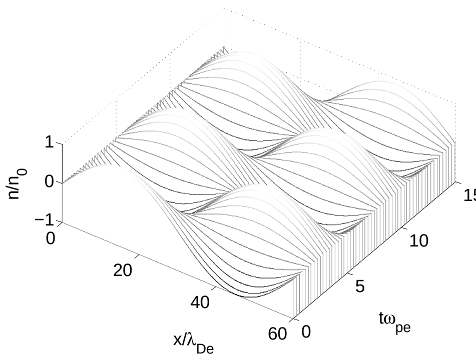
Chapter 6

Results

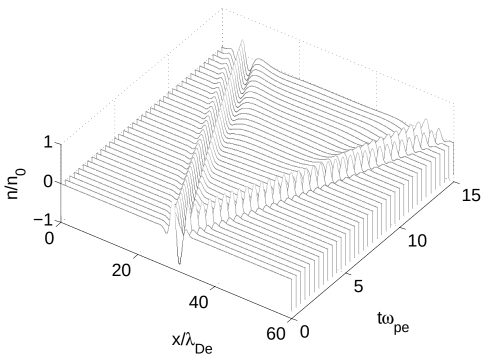
Here we represent numerical solutions for the different Langmuir wave models introduced in Chapters 2 and 4.

6.1 The basic fluid model

For linear Langmuir waves in a single electron component plasma our numerical model from section 5.4.1 gives results as shown in Figure 6.1 for two different initial conditions.



(a) Initial condition $\frac{n(x,0)}{n_0} = \sin\frac{\pi}{30}x$.



(b) Initial condition $\frac{n(x,0)}{n_0} = \sin\left(\frac{\pi}{2}x\right)e^{-\frac{(x-30)^2}{4}}$.

Figure 6.1: Langmuir waves in a one-dimensional single electron component plasma. Shown here for a standing wave and a simple wave-packet as initial conditions.

We have the analytical expression, (2.5), for the dispersion relation for these waves. The numerical solution we find for the electron density perturbations can be Fourier transformed into the frequency domain, thus providing a means for retrieving the dispersion relation. We execute this by first running the solver for several different wave-numbers, k, but otherwise identical initial conditions. Thereafter we Fourier transform all the solutions into the frequency domain by use of the built-in Fast Fourier Transform (FFT) routine in Matlab and find the frequencies corresponding to the different wave-numbers from the resulting spectra. When the frequencies for each wave-number is known the angular frequencies, i.e. the disperison relation, are found without further ado.

In Figure 6.2 we compare the dispersion relation obtained from the numerical results to the analytical dispersion. Our measurement of the angular frequencies through Fourier transforms is limited by the timstep, Δt, we have chosen for the numerical approximation. The uncertainty this represents in Figure 6.2 is however to small to bee seen on the current scale.

As we can see from the figure the numerical and the analytical dispersion are in very good accordance, thus the numerical scheme produces physically plausible solutions.

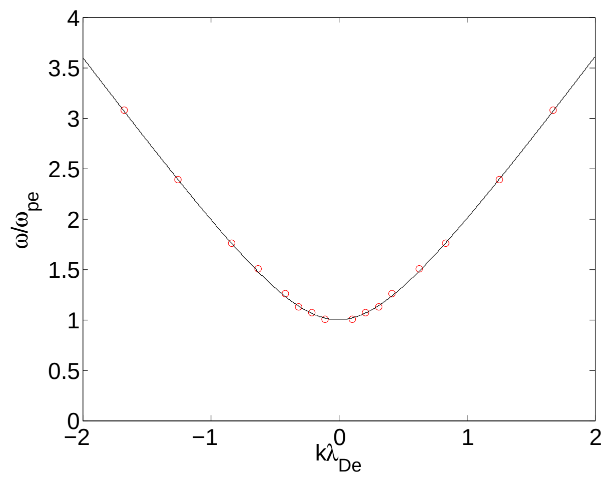


Figure 6.2: The dispersion relation for Langmuir waves in a one-dimensional plasma. The continuous line is the analytical result derived in Chapter 2 and the circles originate from numerical results.

6.2 Langmuir waves in multi-component plasmas

The propagation of Langmuir waves in a multi-component one-dimensional electron fluid model is investigated here by the numerical solver outlined in section 5.4.2. We will see how our numerical model reproduces a know dispersion relation for the two-component model, and for N component models we observe a correspondence between the number of components and the number of dispersion branches. Furthermore, as N increase we find that the waves become damped in time.

6.2.1 Dispersion relations

In the same manner as for the single component model we can use Fourier transformation to find numerical values for the dispersion of an N electron component plasma.

The two electron component dispersion relation

From section 2.3.1 we have an analytical expression, (2.13), for the dispersion relation for electron density perturbations in a two-electron component plasma. We compare this analytical dispersion relation to numerical results for the same density perturbations in Figure 6.3.

It is apparent from the figure that the numerical solution reproduces the analytical dispersion relation. The uncertainty in the measurement of the numerical dispersion results is, as in the previous section, to small to be seen on the present scale. This is a verification of the accuracy of our numerical model and numerical solutions for a greater number of electron components will therefore also be reliable.

N component dispersion relations

Although we do not have explicit analytical expressions for the dispersion relation for Langmuir waves in N electron component plasmas we can find a dispersion plot from the numerical solution in exactly the same way as for the two-component electron plasma. Figure 6.4 are the numerical dispersion results for fluid models with $N = 4$ and $N = 10$.

From these examples of N component dispersion relations we see that the Langmuir waves have the same number of dispersion branches as the number of electron components in the plasma. This is exactly what we predicted analytically with the dispersion relation (2.14).

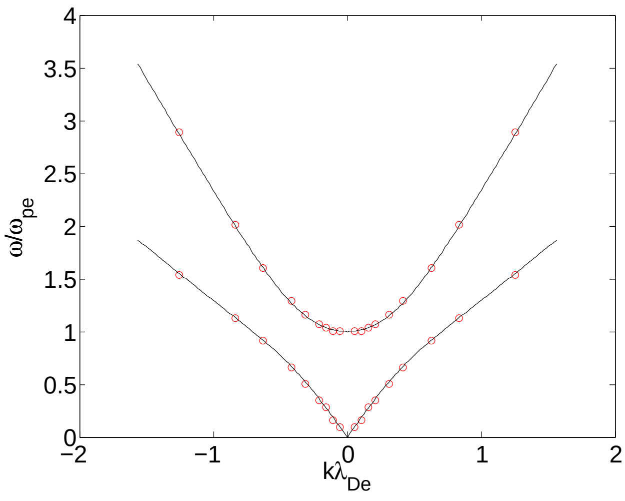
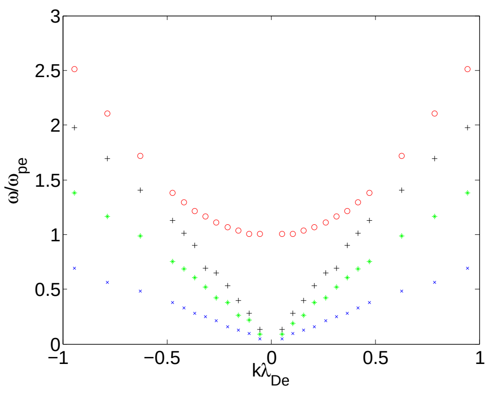
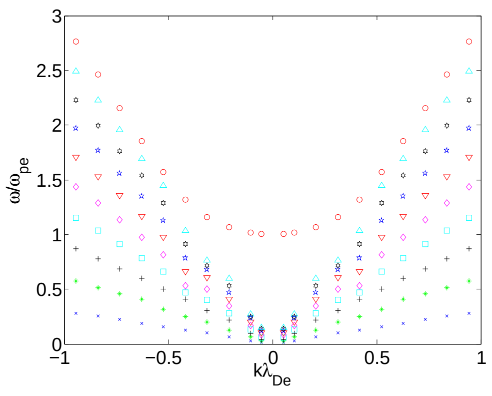


Figure 6.3: The dispersion relation for Langmuir waves in a plasma with two electron components. The continuous lines are the analytical results derived in Chapter 2 and the circles corresponds to numerical results.



(a) N = 4



(b) N = 10

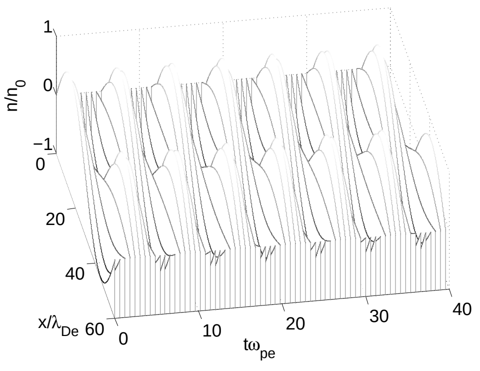
Figure 6.4: Numerical results for the dispersion relations for Langmuir waves in N electron-component plasmas.

Note how the branch with $\omega(k = 0) = \omega_{pe}$ appears for both $N = 2$, 4 and 10. This branch is the same as the one found for single-component plasmas and represents the total plasma frequency. It is excited by the sum of all the different electron density components. For all the N component plasmas we consider here the total electron density is normalized to unity. Therefore any plasma, independent of the number of components, have the same total plasma frequency and we find $\omega(k=0)=\omega_{pe}$ for all of them.

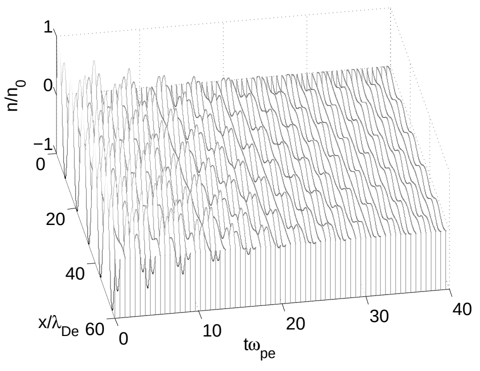
6.2.2 Damping

In Chapter 3 we argued that Landau damping is expected in a multi-component fluid model. We will investigate this here.

Figure 6.5 illustrates Langmuir wave evolution in a plasma with 50 electron components, shown for two different initial conditions. All the 50 components have the same initial amplitude. In Figure 6.5(a) we see the evolution of a initially sinusoidal wave with wave-number $k=\frac{\pi}{60}$ and in Figure 6.5(b) a sinusoidal wave with wave-number $k=\frac{\pi}{6}$. The wave with the highest wave-number is visibly damped whereas the other wave seems to propagate unchanged in the same time interval.



(a) Initial condition $n(x,0)=\sin\frac{\pi}{60}x$.



(b) Initial condition $n(x,0)=\sin\frac{\pi}{6}x$.

Figure 6.5: Langmuir wave evolution for two different initial conditions in a plasma with 50 electron components.

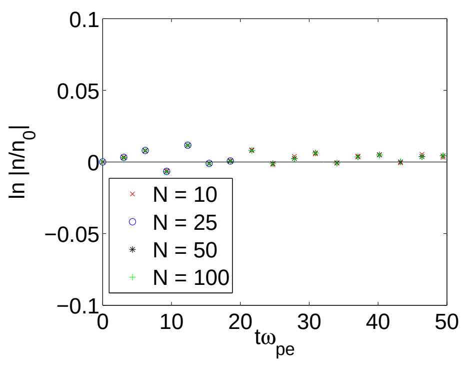
Analytically we have the expression

$$\omega_L=\frac{\pi}{2}\frac{\omega_{pe}^3}{k^2}F_0'\left(\frac{\omega_{pe}}{k}\right)=-\frac{1}{2}\sqrt{\frac{\pi}{2}}\omega_{pe}\left(\frac{\omega_{pe}^3}{k^3\sigma^3}\right)e^{-\frac{\omega_{pe}^2}{k^2\sigma^2}}$$

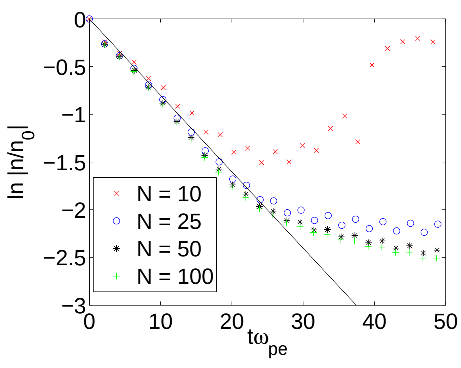
for Landau damping [15]. Here the initial electron density distribution, F_0, is given by the Maxwell-Boltzmann distribution and ω_{pe}/k is an approximation to the phase velocity, u_{ph}, of the wave. We will compare this analytical value for the Landau damping with the damping that we observe numerically for the waves in Figure 6.5.

The phase velocity, u_{ph}, of the Langmuir waves can be estimated by considering the Fourier transformation of the total electron density perturbation in the frequency domain. For an N electron component plasma the transformation, as discussed in the previous section, yields one frequency for each component. We take out the highest frequency and estimate the phase velocity, u_{ph}, from the resulting angular frequency, ω, as $u_{ph} = \omega/k$ where k is the wave-number of the Langmuir wave in question. As seen in Figures 6.3 and 6.4 the dispersion branch with $\omega(k = 0) = \omega_{pe}$, corresponding to Langmuir waves, is always higher than the acoustic branches and it is therefore logical to take out the highest frequency.

In Figure 6.6 we have evaluated the time evolution of the waves in a spatial point where the wave amplitudes are initially at maximum. It is the natural logarithm of the absolute value of the wave crests that are shown as circles and the solid line corresponds to the estimate of the analytical value of Landau damping. The figure also include results for Langmuir waves in plasmas with 10, 25 and 100 electron components. As seen from the figure all the numerical results are in good accordance with the analytical estimate.



(a) Initial condition $n(x,0) = \sin\frac{\pi}{60}x$.



(b) Initial condition $n(x,0) = \sin\frac{\pi}{6}x$.

Figure 6.6: Landau damping in several N electron component plasmas. Shown here for two initially different Langmuir waves. The solid lines corresponds to estimates for the analytical Landau damping whereas numerical results are shown as crosses and circles.

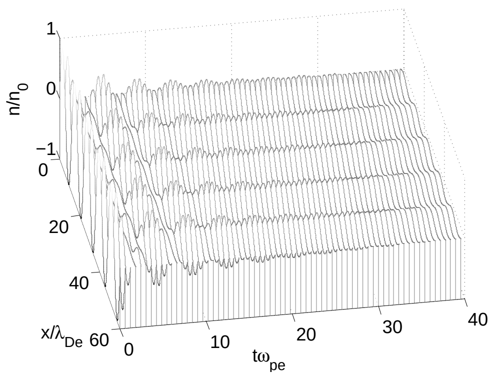
We see in Figure 6.6 that the initial condition $n(x,0) = \sin\frac{\pi}{60}x$ does not result in any noticeable damping at all, whereas an increase in wave-number for the initial electron density perturbation clearly results in a damped wave. Returning to Chapter 3 and Figure 3.2 we find that this is to be expected. Because of the k^{-1} dependence of the phase velocity, $u_{ph} = \omega/k$, an increase in wave-number is a decrease in phase velocity. This means that u_{ph} is shifted to the left in Figure 3.2 and consequently there are more particles loosing energy to the wave than there are at lower wave-numbers. The result is a more efficient damping. For very low wave-numbers (i.e. very large phase velocities) Langmuir waves do not exhibit Landau damping at all. This is consistent with the analytical expression given for the damping rate for the Landau damping.

For the plasma with only 10 electron components we observe a recurrence phenomena in Figure 6.6(b). The plasmas with higher number of electron components also exhibit recurrence, but the higher the number of electron components the longer time it takes before the wave crests reach a minimum and thereafter start to increase towards the initial amplitude. We believe this reflects the efficiency of the phase mixing process of the N undamped modes, as suggested by [26]. It also implies that when $N \to \infty$ the recurrence time approaches infinity and we thus obtain a definitively damped wave.

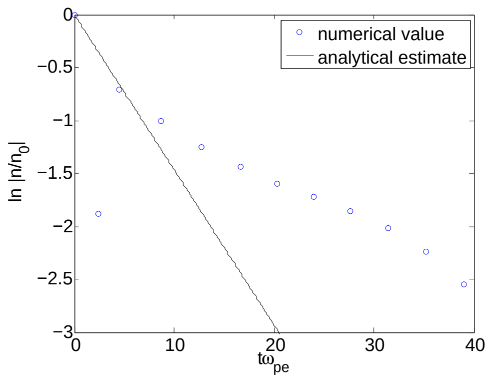
So far we have let the N modes of the plasmas have the same initial amplitude, namely $1/N$. Returning to Figure 2.3(b) we remember that the lowest box in the approximation to the Maxwell-Boltzmann distribution corresponds to electron component number N and the highest box to electron component number 1. In Figure 6.7 we have only initially excited component number one in a plasma with 50 component's in total. The results for the damping of the wave are obtained in the same manner as the results in Figure 6.6. The second numerical point in Figure 6.7(b) is a little "off" from the others. This is because of the decreasing trend seen in Figure 6.8(a) where we have plotted the time evolution of the wave at the spatial point of one of the initial wave crests.

In Figure 6.8(b) we have removed the linear trend from the wave in Figure 6.8(a). The detrending also removes some of the amplitude of the wave, but as we can see in Figure 6.9 the damping of the wave now corresponds much better with the analytical estimate for Landau damping. Although this means that Landau damping is present also in this plasma, we note that signal processing is required to find it. For the plasmas where all electron components were initially equally excited in Figures 6.6(a) and 6.6(b) we saw signatures of Landau damping without performing any sort of signal modification.

All in all, when we compare Figures 6.7(a) and 6.5(b) we see that exciting only the first component results in a wave that first decreases more

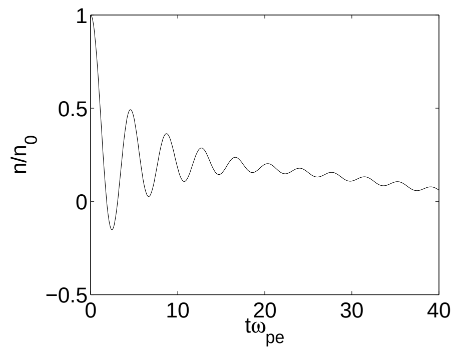


(a) Langmuir wave propagation.

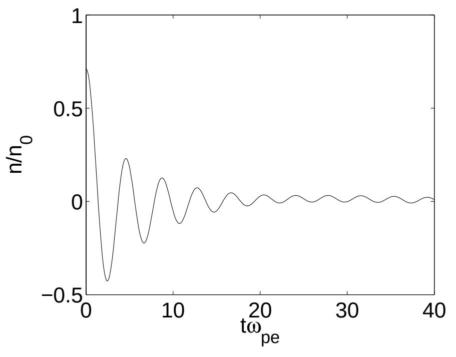


(b) Landau damping.

Figure 6.7: Langmuir wave propagation in a plasma with 50 electron components where the initial condition is set as $n_1(x,0) = \sin\frac{\pi}{6}x$ and $n_r(x,0) = 0$ for $r = 2,3,...,50$.



(a) Langmuir wave propagation as seen from the spatial point $x = 3$.



(b) Langmuir wave propagation as seen from the spatial point $x = 3$ when the linear trend is removed.

Figure 6.8: Langmuir wave propagation in a plasma with 50 electron components where the initial condition is set as $n_1(x,0) = \sin\frac{\pi}{6}x$ and $n_r(x,0) = 0$ for $r = 2,3,...,50$. Here the time evolution of the wave at the spatial point of one of the initial wave crests is shown.

rapidly but quickly reach an almost constant amplitude. From Figure 6.7(b) it is clear that for this new initial condition the damping of the wave is

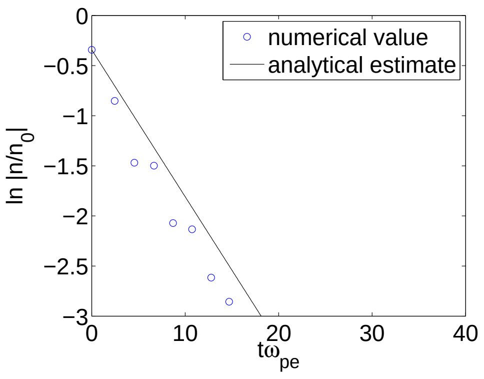
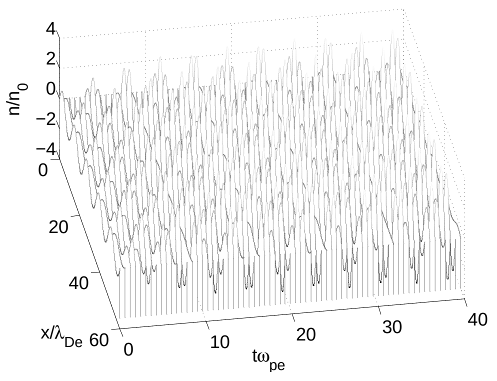


Figure 6.9: Landau damping of the detrended wave in Figure 6.8(b).

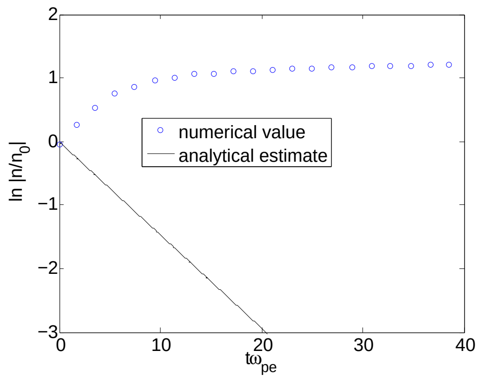
only consistent with Landau damping for approximately 10 plasma periods whereas weighing all the components equally gave a consistent damping for approximately 30 plasma periods. This can be seen in comparison to the difference in the result for plasmas with 10 and 50 electron components, respectively, when all components are given the same initial amplitude. The phase mixing process was less efficient for 10 components and lead to a relatively short recurrence time, as seen in Figure 6.6(b). Although the 49 remaining components will eventually be excited by the electrostatic coupling to the component we initially excite in Figure 6.7, this process takes so long that it is not yet noticeable for the time interval we investigate here. The phase mixing process therefore reproduce Landau damping in a less efficient manner than if all components were given the same initial amplitude.

Furthermore, in Figure 6.10 we have excited component number 50 in a 50 component plasma. As we see in the figure this initial condition gives a wave that is not damped but clearly increases for the first ten plasma periods and thereafter have a more or less constant amplitude. For longer time intervals than shown here we find that the initial condition is reproduced in recurrence intervals of approximately 200 plasma periods.

The amplification of the wave is probably caused by positive interference between the initially excited component and the 49 remaining components which are excited through the electrostatic coupling. Because the increase in amplitude ceases relatively quickly, this is a transient phenomena and not an instability. After $10/\omega_{pe}$ the almost constant amplitude that we observe can be seen in connection to the special initial condition that we discussed in section 3.3.2. In other words we have here chosen the initial perturbation of the electron density so as to give a wave with phase velocity almost identical



(a) Langmuir wave propagation.

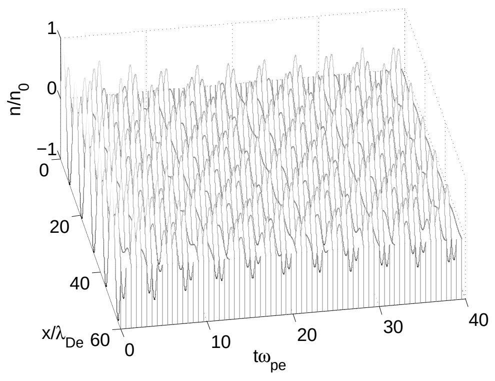


(b) Landau damping.

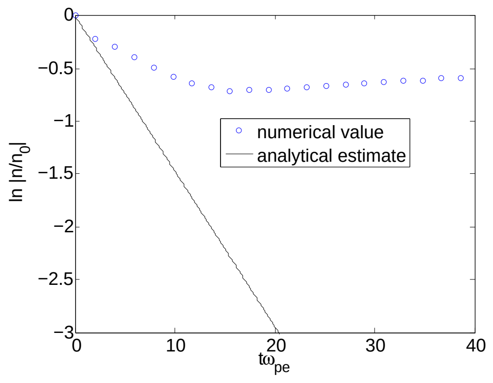
Figure 6.10: Langmuir wave propagation in a plasma with 50 electron components where the initial condition is set as $n_{50}(x,0) = \sin\frac{\pi}{6}x$ and $n_r(x,0) = 0$ for $r = 1, 2, ..., 49$.

to the boundary of the excited water-bag, thus there are no wave-particle interactions and the wave will not be damped.

If we let component number 50 have initial value $n_{50}(x,0) = \frac{1}{6}\sin\frac{\pi}{6}x$ and the other components have the same sinusoidal form but amplitudes $5/294$ so as to make the total amplitude equal to unity, we do not see an increasing amplitude, Figure 6.11. On the contrary we once again see that the wave is damped with a factor not far from the analytical estimate for Landau damping, at least for the first few plasma periods. This means that the phase-mixing process between the undamped modes is again recognizable, albeit not very prominent. Once again we note how the initial distribution of energy between the components is closely related to the efficiency of the phase-mixing process.



(a) Langmuir wave propagation.



(b) Landau damping.

Figure 6.11: Langmuir wave propagation in a plasma with 50 electron components where the initial condition is set as $n_{50}(x,0) = \frac{1}{6}\sin\frac{\pi}{6}x$ and $n_r(x,0) = \frac{5}{294}\sin\frac{\pi}{6}x$ for $r = 1, 2, ..., 49$.

6.3 Langmuir-envelopes in single component plasmas

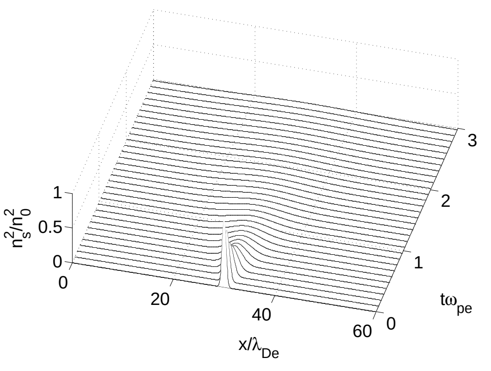
The slowly varying amplitude part of Langmuir waves in a single electron component plasma are described by the linear Schrödinger equation (2.18). For the initial condition $n_s(x,0)/n_0 = e^{-\frac{\pi}{2}x^2}$ we have an exact solution of this equation, [50]:

$$n_s(x,t) = \frac{1}{\sqrt{1+3\pi t}} e^{-\frac{\pi}{2}\frac{x^2}{1+3\pi i t}}. \tag{6.1}$$

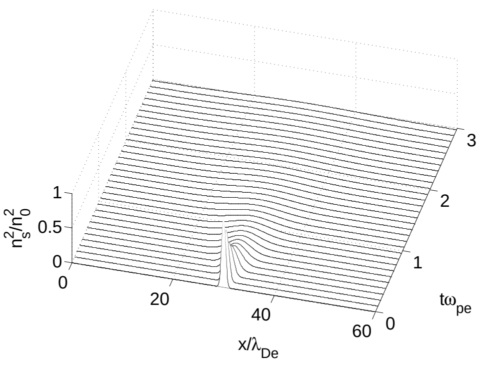
We can use this analytical result to check the accuracy of the finite difference scheme outlined in section 5.4.3 for the linear Schrödinger equation. In Figure 6.12(a) the function (6.1) is shown graphically and in Figure 6.12(b) the numerical solution for the prescribed initial condition is shown. The two figures look very similar. However, when we subtract the numerical result from the analytical in Figure 6.12(c) it becomes clear that there are some discrepancies between the two.

The oscillations visible after $2,5$ plasma periods in Figure 6.12(c) are a numerical artifact caused by the periodic boundary conditions. Utilizing periodic boundary conditions means starting the same initial electron density perturbation in an infinite set of spatial domains identical to the one shown in Figure 6.12(b). After some time these initial perturbations will propagate into the neighbouring domains, creating an interference phenomenon. We can avoid this by choosing a wider computational domain and thus prolonging the interference time. These oscillations are therefore merely a consequence of our choice of length for the computational domain and can be removed so easily that they do not manifest a grave discrepancy between the numerical and analytical solutions.

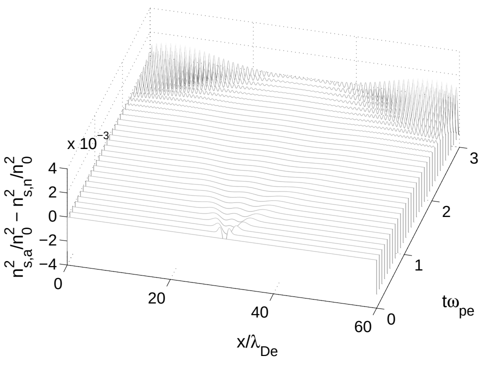
Nevertheless, we see from Figure 6.12(c) that there are differences between the analytical and numerical values, especially for the first plasma periods. The analytical solution is broadening faster than the numerical solution. In particular the first plasma period is the time interval over which the perturbation changes most rapidly. Our numerical solution is limited by our chosen spatial grid, round off errors in the computation and the accuracy of the finite difference approximation. It is exactly at the times when the plasma changes most rapidly that these limitations will be most prononced. We have used grid parameters $h = 0.06$ and $\Delta t = 4.8 \times 10^{-4}$ for the present computation and the truncation error is proportional to the square of these parameters. A smaller value for h therefore consistently gives a better approximation to



(a) Analytical solution from Griffiths [50].



(b) Numerical solution.



(c) The difference between the analytical, $n_{s,a}/n_0$, and the numerical solution, $n_{s,n}/n_0$.

Figure 6.12: The slowly varying part of Langmuir waves in a single electron component plasma as described by the linear Schrödinger equation when the initial condition is set to $n_s(x,0)/n_0 = e^{-\frac{\pi}{2}x^2}$.

the analytical result, but at cost of the computational efficiency. Yet, all the differences between the analytical and numerical solutions are smaller than 2×10^{-3} in Figure 6.12(c), that is less than 2‰ of the initial amplitude, and therefore small enough to be discarded for our purposes.

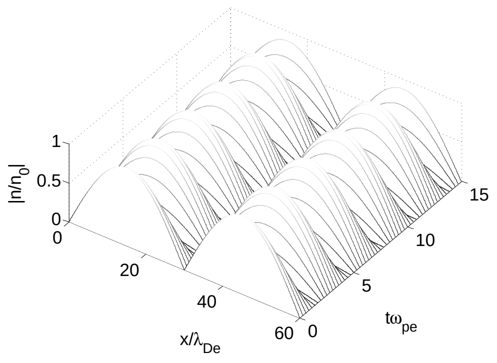
We have established that our numerical scheme for the linear Schrödinger equation almost exactly reproduce a known analytical solution. Thus, we conclude that the method is satisfactorily accurate and expect it to produce reliable solutions also for other initial conditions.

Our expectation is that the linear Schrödinger equation (2.18) will describe the slowly varying amplitude part of a Langmuir wave, i.e. the wave-envelope. We compare two numerical solutions with identical initial and boundary conditions in Figure 6.13, one for the linear Schrödinger equation and one for the full wave equation (2.4). It immediately seems plausible that the result shown in Figure 6.13(b) is the wave-envelope of the result shown in Figure 6.13(a).

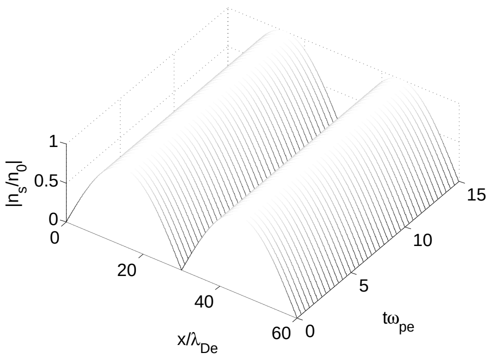
When we say that Figure 6.13 is a comparison of two solutions with identical initial and boundary conditions, we are not entirely sincere. For the numerical solutions of the full wave equation describing Langmuir waves we assumed the partial time derivative, $\frac{\partial n}{\partial t}$, of the electron density to be equal to zero at time $t = 0$. For the solution of the corresponding linear Schrödinger equation we have no such initial condition. However, as $\frac{\partial n_r}{\partial t} = \frac{\partial}{\partial t}\left[n_{r,s}e^{-i\omega_{pe}t} + c.c\right] = 2\frac{\partial}{\partial t}n_s(x,0)$ and we initially take $n_s(x,t)$ to be a slowly varying quantity we assume this to be approximately equal to zero. The initial condition of the full-wave solution is then contained in the Schrödinger solution.

For the Langmuir wave equation solution we see five full periods. In Figure 6.13(c) we show the Langmuir wave propagation as seen at the spatial point of the initial wave crest. The red curve is for the linear Schrödinger solution and the black curve for the full wave solution. As we expected the red curve is the envelope of the black curve. This is a rather convincing result and another verification of the accuracy of our numerical model.

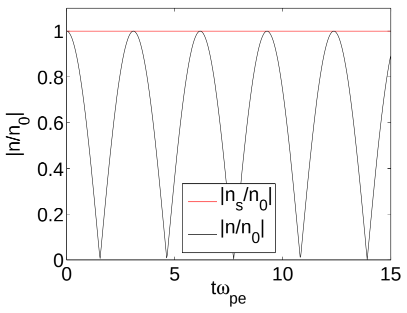
When we derived the linear Schrödinger equation in section 2.4 we assumed $\frac{\partial n_s}{\partial t} << \omega_{pe}n_s$. We have estimated the ratio between these two values at each time step for the wave shown in Figure 6.13(b). The time derivative was found by applying a one sided forward finite difference to the numerical values for the electron density perturbation as seen at the spatial point for the initial wave crest, i.e. at $x/\lambda_{De} = 15$. The result is shown in Figure 6.14. Our assumption is verified by this as the ratio is small at all times.



(a) Solution of the Langmuir wave equation.



(b) Solution of the linear Schrödinger equation.



(c) Langmuir wave propagation as seen at the spatial point $x/\lambda_{De} = 15$ for both the full wave solution, $|n/n_0|$, and the linear Schrödinger solution, $|n_s/n_0|$.

Figure 6.13: Numerical solutions of the Langmuir wave equation and the linear Schrödinger equation, with initial conditions $n(x,0)/n_0 = \sin\frac{\pi}{30}x$ and $n_s(x,0)/n_0 = \sin\frac{\pi}{30}x$, respectively, for the electron density perturbation of a single-electron component plasma. Only absolute values are shown here.

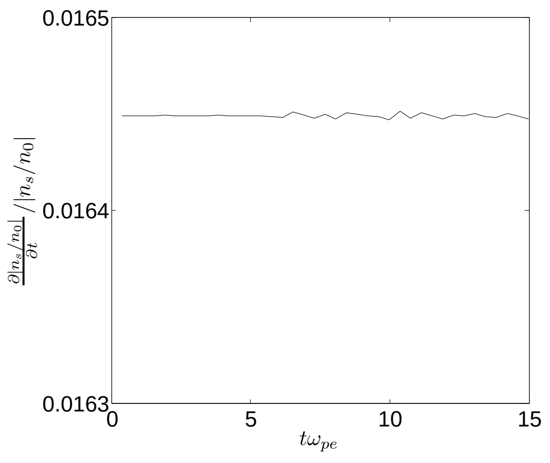
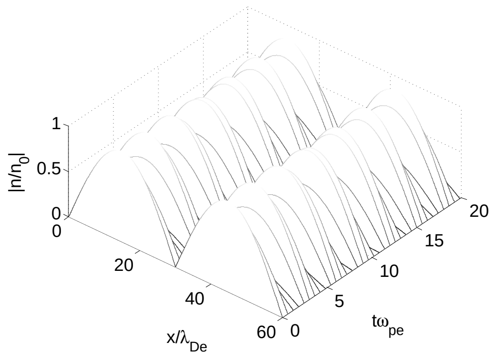


Figure 6.14: The ratio of $\frac{\partial}{\partial t}|n_s/n_0|$ to $\omega_{pe} n_s/n_0$ for the electron density perturbation shown in Figure 6.13(b).

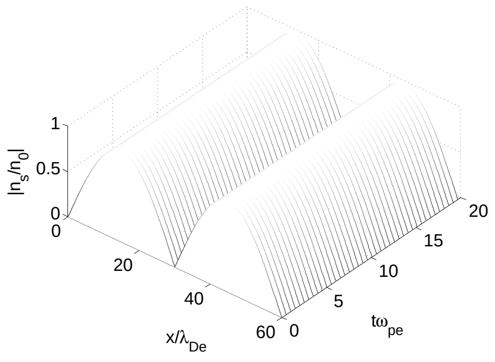
6.4 Langmuir-envelopes in multi-component plasmas

Langmuir wave-envelopes in N electron component plasmas are described by the set of coupled linear Schrödinger equations (2.19), for which we found a finite difference scheme in section 5.4.4. We will investigate solutions for plasmas with $N = 2$ and $N = 25$ here.

Figure 6.15(b) is the numerical solution for the envelope in a plasma with two electron components when both components are given the initial condition $n_s/n_0 = \sin \pi x/30$. The full wave equation solution of the same initial-boundary value problem is shown in Figure 6.15(a).



(a) Total electron density perturbation as described by the Langmuir wave equations.



(b) Total electron density perturbation as described by the linear Schrödinger equations.

Figure 6.15: Numerical solutions of the set of coupled Langmuir wave equations and the set of coupled linear Schrödinger equations in a plasma with two electron components. For both solutions all the components have identical initial conditions $n_r(x, 0)/n_0 = \frac{1}{2} \sin \frac{\pi}{30} x$ and $n_s(x, 0)/n_0 = \frac{1}{2} \sin \frac{\pi}{30} x$, respectively, for $r = 1, 2$. Only absolute values are shown here.

It appears that the linear Schrödinger solution is the wave-envelope of the full wave solution in Figure 6.15. In Figure 6.16 we have plotted the total electron density perturbations as seen at the spatial point of the initial wave crest. The red curve for the linear Schrödinger solution is to a very good approximation the envelope of the black curve representing the full wave solution. This means that the two coupled linear Schrödinger equations do give a good representation of the slowly varying amplitude part of the electron density perturbations. Therefore we expect the assumption $\frac{\partial n_{r,s}}{\partial t} << \omega_{pe} n_{r,s}$ for $r = 1, 2$ in the derivation of the Schrödinger equations to be justified.

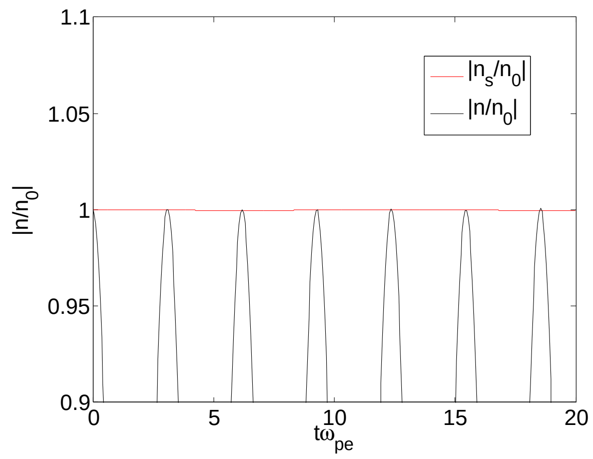


Figure 6.16: Electron density perturbations at the spatial point $x/\lambda_{De} = 45$ for the two solutions shown in Figure 6.15.

We have the time evolution of the ratio $\frac{\partial}{\partial t} n_{r,s}/\omega_{pe} n_{r,s}$ for the two components in Figure 6.17. Both ratios are continually small and we establish the assumption $\frac{\partial}{\partial t} n_{r,s} << \omega_{pe} n_{r,s}$ as correct.

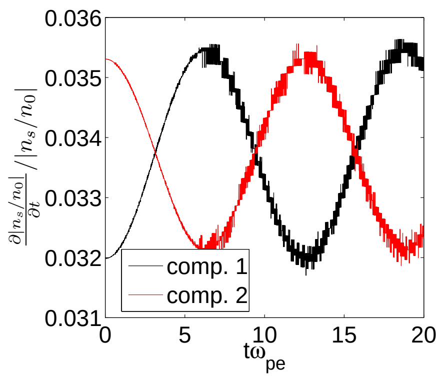
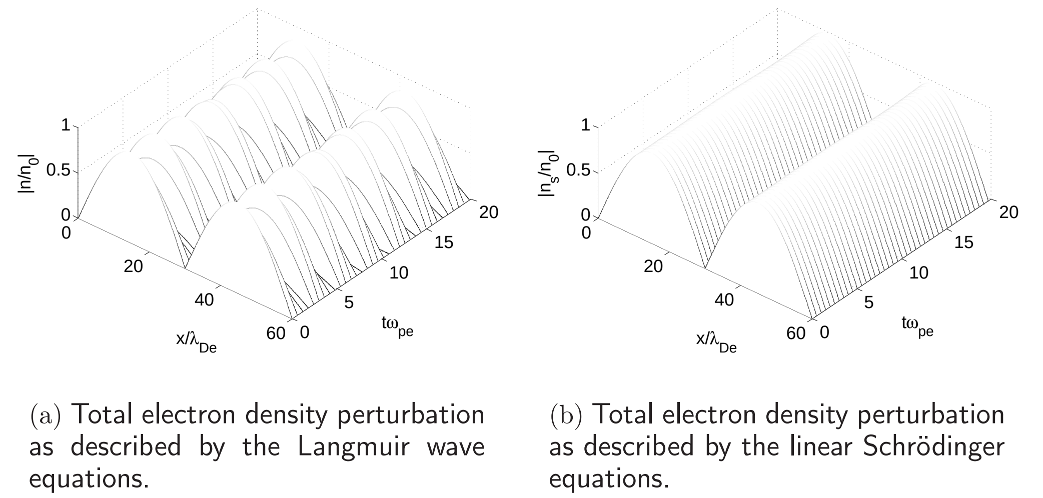


Figure 6.17: The ratio of $\frac{\partial}{\partial t}|n_{r,s}/n_0|$ to $\omega_{pe} n_{r,s}/n_0$ for the density perturbations of the two electron components, $n_{1,s}$ and $n_{2,s}$, of the sum shown in Figure 6.15(b).

Increasing the number of components to 25 the accordance between the solution of the Schrödinger equations and the wave equations also appears to be good. This is shown in Figure 6.18. However, closer inspection reveals that the accordance is not as good as we saw for the two component plasma. In Figure 6.19 we have plotted the electron density perturbations as seen at the spatial point of the initial wave crest for both the solutions in Figure 6.18. The set of coupled Schrödinger equations does not give exactly the wave envelope of the Langmuir wave in a plasma with 25 electron components.



(a) Total electron density perturbation as described by the Langmuir wave equations.

(b) Total electron density perturbation as described by the linear Schrödinger equations.

Figure 6.18: Numerical solutions of the set of coupled Langmuir wave equations and the set of coupled linear Schrödinger equations in a plasma with 25 electron components. For both solutions all the components have identical initial conditions $n_r(x,0)/n_0 = \frac{1}{25}\sin\frac{\pi}{30}x$ and $n_s(x,0)/n_0 = \frac{1}{25}\sin\frac{\pi}{30}x$, respectively, for $r = 1, 2, ..., 25$. Only absolute values are shown here.

The reason for the discrepancy between the wave-envelope of the Langmuir wave and the linear Schrödigner solution in 25 component plasmas can be found in Figure 6.20. Here the ratio $\frac{\partial}{\partial t}n_{r,s}\big/\omega_{pe}n_{r,s}$ for all the 25 components is shown. For several components the ratio is larger than one! Our assumption of a small ratio that explicitly leads to the set of coupled linear Schrödinger equations is therefore not valid and the numerical solution we have does not correctly describe the slowly varying amplitude of Langmuir waves in the plasma. Note, however, that the Schrödinger equations give qualitatively the same result as the full wave solution.

We have not labeled the 25 different components in Figure 6.20. It is component number 23 that has the largest ratio, the ratios thereafter decrease for the components with lower component number and finally component number one has the lowest ratio. For components number 24 and 25 we find that the ratio is decreasing from the value for component number 23 so that

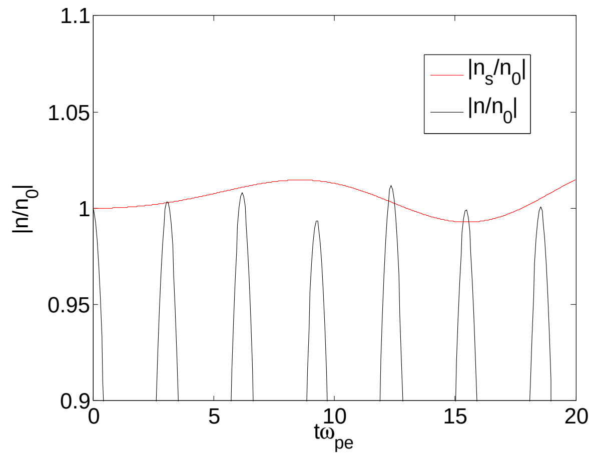


Figure 6.19: Electron density perturbations at the spatial point $x/\lambda_{De} = 15$ for the two solutions of $N = 25$ electron component plasmas shown in Figure 6.18.

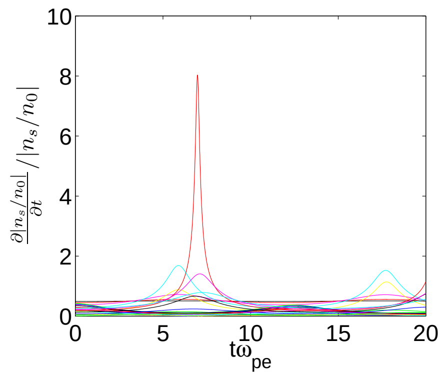
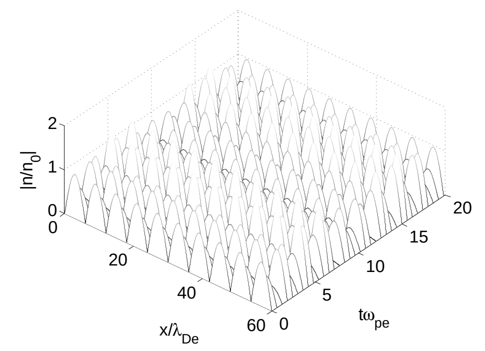


Figure 6.20: The ratio of $\frac{\partial}{\partial t}|n_{r,s}/n_0|$ to $\omega_{pe} n_{r,s}/n_0$ for the density perturbations of the 25 electron components in the sum shown in Figure 6.15(b).

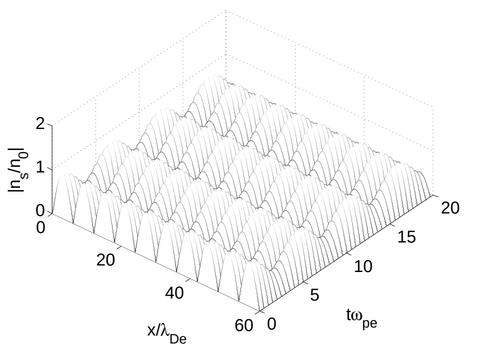
component 23 has a higher ratio than 24 and 24 has a higher ratio than 25.

If we increase the wave-number of the initial sinusoidal electron density perturbation from $\pi/30$ to $\pi/6$ the agreement between the results from the Schrödinger equations and the wave equations becomes evidently poorer for both multi component plasmas. See Figure 6.21. The time evolution of the density perturbations as seen at the position of one of the initial wave crests further illustrates the poor correspondence in figure 6.22.

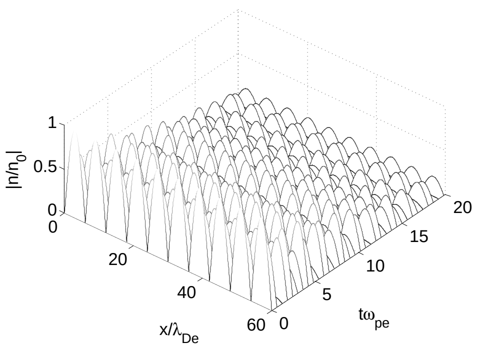
As for the $N = 25$ component plasma with total initial electron density perturbation $n_s/n_0 = \sin \frac{\pi}{30} x$, the assumption we used to derive the set of



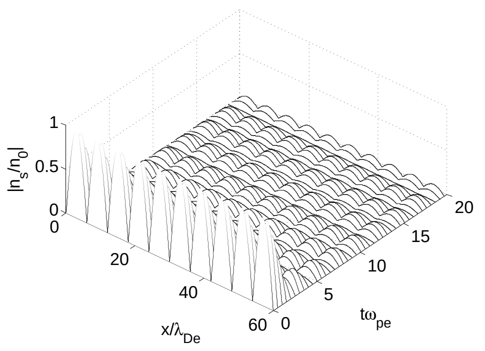
(a) Total electron density perturbation as described by the Langmuir wave equations for $N = 2$.



(b) Total electron density perturbation as described by the linear Schrödinger equations for $N = 2$.



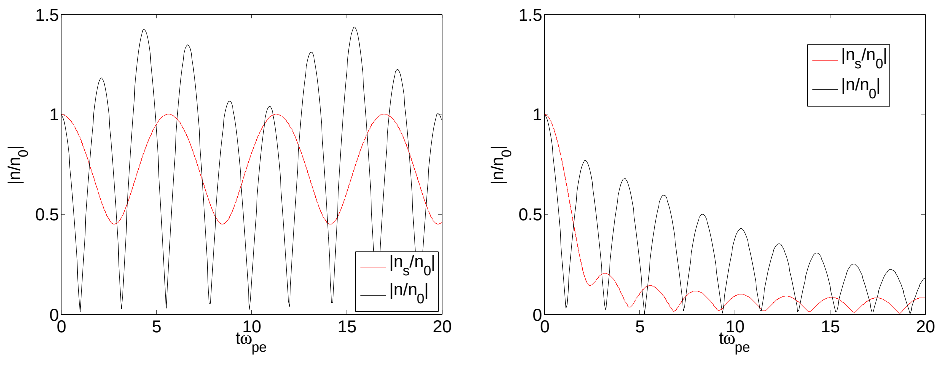
(c) Total electron density perturbation as described by the Langmuir wave equations for $N = 25$.



(d) Total electron density perturbation as described by the linear Schrödinger equations for $N = 25$.

Figure 6.21: Numerical solutions of the set of coupled Langmuir wave equations and the set of coupled linear Schrödinger equations in a plasma with, respectively, $N = 2$ and $N = 25$ electron components. For both type of solutions all the components have identical initial conditions $n_r(x,0)/n_0 = \frac{1}{N}\sin\frac{\pi}{6}x$ and $n_s(x,0)/n_0 = \frac{1}{N}\sin\frac{\pi}{6}x$, respectively, for $r = 1,2,...,N$. Only absolute values are shown here.

coupled Schrödinger equations is not fulfilled for these two plasmas. We see this in Figure 6.23. For the plasma with 25 electron components we have not labeled each component, but as for the two component plasma it is the Nth component which has the largest ratio and the ratios thereafter decrease for each component so that the first component has the lowest ratio. Also for



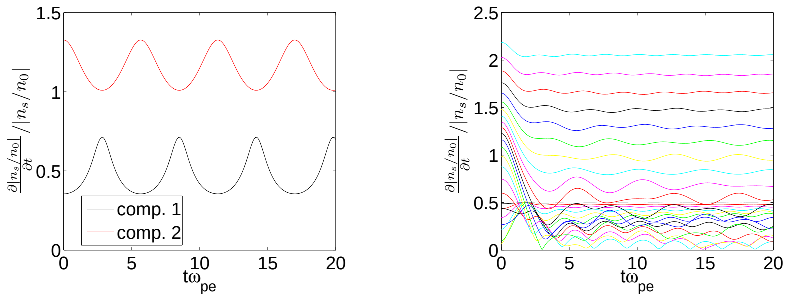
(a) N = 2. (b) N = 25.

Figure 6.22: Electron density perturbations at the spatial point $x/\lambda_{De} = 9$ for the two solutions of two different N electron component plasmas shown in Figure 6.21.

the 25 electron component plasma where the density perturbation was given a higher wavenumber we observed some correlation between high component number and a high ratio between $\frac{\partial}{\partial t}|n_{r,s}/n_0|$ and $\omega_{pe}n_{r,s}/n_0$.

We can relate this correlation to our discussion of the oscillation amplitudes of the individual electron components in a plasma in section 3.3.2. From relation (3.7) describing the dispersion of each electron component we have that the time derivative of the components density perturbation, $\frac{\partial n_r}{\partial t}$, will be larger the closer the water-bag limit, u_r, is to the phase velocity, $u_{ph} = \omega/k$, of the Langmuir wave. When u_{ph} is large the electron component corresponding to the lowest box in the approximation to a Maxwell-Boltzmann distribution is the one with the best fit for $u_r \approx u_{ph}$. As u_r decreases so does the time derivative of the density perturbation of the component. It also makes sense that it is not the highest component that has the highest ratio in the plasma with the higher wavenumber waves, i.e. $k = \pi/6$, as the phase velocity of Langmuir waves must be smaller in this plasma than in the plasmas with $k = \pi/30$.

Furthermore, the dispersion relation for a plasma with $N = 10$ electron components in Figure 6.4(b) illustrates an important concept that is directly connected to the inconsistency we observe for the coupled linear Schrödinger solutions. At wave-number $0.5k\lambda_{De}$ we see that the 10 dispersion branches lie very close to one another, this means that all electron density perturbations propagating with this wave-number will have similar frequencies. Consequently, when we write the electron density perturbations



(a) N = 2. (b) N = 25.

Figure 6.23: The ratio of $\frac{\partial}{\partial t}|n_{r,s}/n_0|$ to $\omega_{pe} n_{r,s}/n_0$ for the density perturbations of the two N electron component plasmas shown in Figure 6.21.

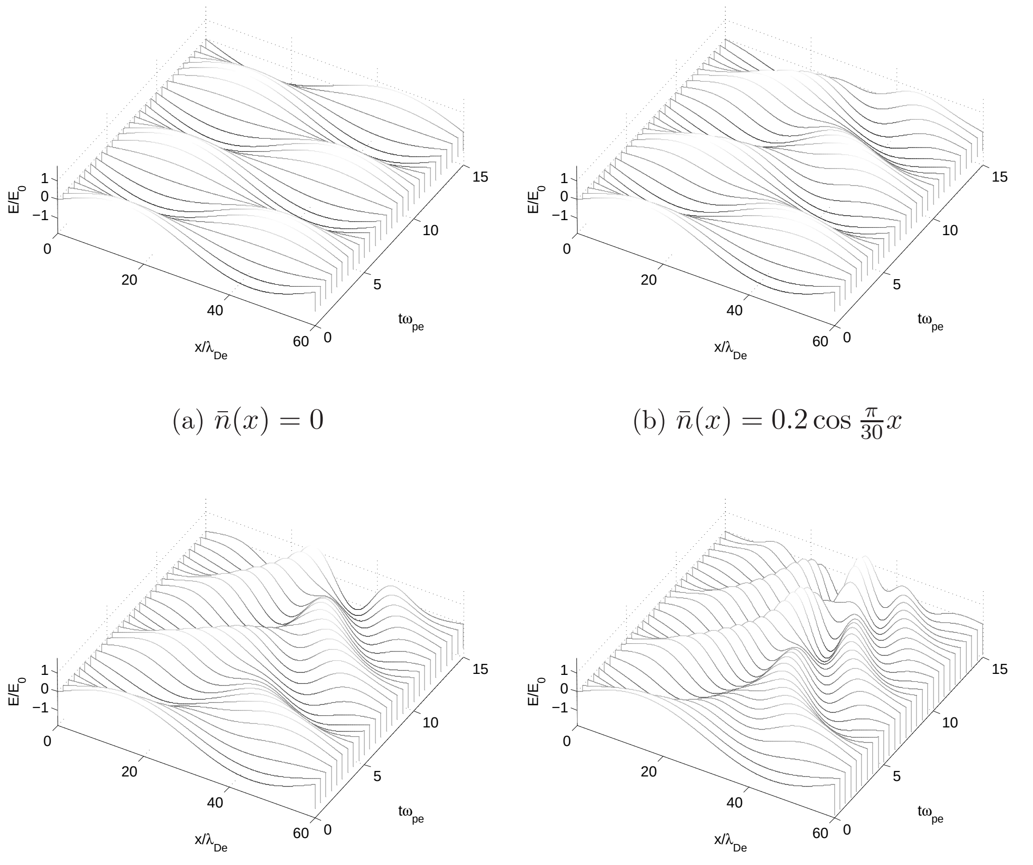
as $n = n_s(t,x)e^{-i\omega_{pe}t} + c.c$ in the belief that we have taken out the high frequency oscillations and only slowly varying amplitudes are left in $n_s(t,x)$, we are wrong because several of the "low" frequencies may be very close to the plasma frequency. We can only imagine that the more electron components there are in a plasma the more frequencies will be close to ω_{pe}. This is consistent with our observation of an increasing discrepancy between the linear Schrödinger solution and the actual Langmuir wave-envelope when the number of components increase and also when the wave-number is increased.

6.5 Weakly nonlinear Langmuir waves

In a plasma with an inhomogeneous electron density distribution the electrostatic field associated with Langmuir waves is described by (4.21). This equation is valid both for cases where $\bar{n}$ is prescribed (i.e. a priori given) or it is induced by the wave itself due to nonlinear effects. We have several numerical solutions to this equation in Figure 6.24.

Figure 6.24(b) shows the most realistic situation where the unperturbed electron density is only slightly dependent on position as $n/n_0 = 1 + 0.2\sin\pi x/30$, resulting in a weakly nonlinear evolution when compared to the homogeneous result in Figure 6.24(a).

A more extreme situation is illustrated in Figure 6.24(d), where the spatial dependence of the initial electron density, $\bar{n}(x) = \cos\frac{\pi}{30}x$, causes the electrons to be completely depleted at the middle of our one-dimensional plasma. As the electrostatic field perturbation propagates we may say that



(a) $\bar{n}(x) = 0$

(b) $\bar{n}(x) = 0.2 \cos \frac{\pi}{30} x$

(c) $\bar{n}(x) = 0.5 \cos \frac{\pi}{30} x$

(d) $\bar{n}(x) = \cos \frac{\pi}{30} x$

Figure 6.24: Electrostatic field perturbation associated with Langmuir waves in plasmas where the unperturbed electron density distribution varies in the spatial domain as $n/n_0 = 1 + \bar{n}(x)$. The initial electrostatic field perturbation is set to $\frac{E(x,0)}{E_0} = \sin \frac{\pi}{30} x$.

the wave "condensates" at these points where the index of refraction (4.24) is maximum and the local phase velocity minimum. This feature is also weakly prominent in the plasmas were the spatial variation of the electron density is smaller, i.e. Figures 6.24(b) and 6.24(c).

6.6 Weakly nonlinear Langmuir wave-envelopes

The time-space evolution of the slowly varying amplitude part of the electrostatic field, $E(x,t)$, associated with weakly nonlinear Langmuir waves is described by the NLS (4.29) for single electron component plasmas. We found a numerical method for solving the normalized form (5.20) of the NLS in section 5.5.2. By setting the nonlinear term in (5.20) to zero (i.e $\beta = 0$) we recover the regular linear Schrödinger equation,

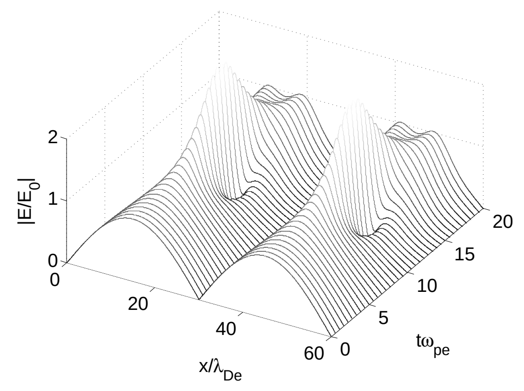
$$i\frac{\partial}{\partial t}E = -\frac{3}{2}\frac{\partial^2}{\partial x^2}E.$$

Physically this is the same as letting the ponderomotive force be equal to zero in equation (4.23). In this limit we are solving the same equation that we solved for the slowly varying amplitude of Langmuir waves described in section 2.4 and the finite difference scheme outlined in section 5.5.2 also becomes identical to the one for low frequency amplitudes. We therefore know, from section 6.3, that the numerical scheme reproduces a known analytical solution and that the usage of the unstable scheme for the first timestep does not compromise the numerical solution. This strengthen our belief that the the numerical method developed in section 5.5.2 produces stable solutions, also for nonlinear situations, that is when the ponderomotive force is present.

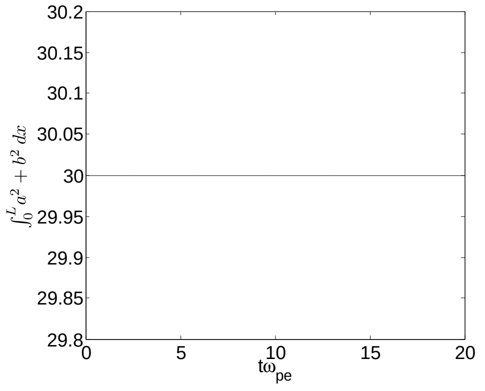
Figure 6.25 shows results for a numerical solution of the NLS (5.20) when the initial electrostatic field perturbation is set to $E/E_0 = \sin(\pi x/30)$. From Figure 6.25(b) we conclude that the energy in the electrostatic field is constant in time and therefore, as expected, conserved. This is a further verification of the physical validity of our numerical approach.

When we derived the NLS we assumed that the time derivative of the amplitude part of the electrostatic field, $\frac{\partial E}{\partial t}$, was much smaller than the product of the electrostatic field and the plasma frequency, $\omega_{pe}E$. We have tested this assumption for the wave in Figure 6.25(a) in Figure 6.26. The ratio is found by the same method as in the previous sections concerning linear Langmuir wave-envelopes. Although the ratio increases for parts of the time interval we study here, it is continuously so small that we can consider our assumption as valid.

If we make the amplitude of the initial electrostatic perturbation three times larger some energy fluctuations become apparent, as seen in Figure 6.27. However these fluctuations are rather small. This initial value is an example of a rather extreme perturbation of the electrostatic field. It is not representative of the general concept of Langmuir waves and therefore not of much interest here. In the following we need not be concerned about the



(a) The electrostatic field.



(b) The energy contained within the electrostatic field.

Figure 6.25: Nonlinear evolution of the electrostatic field associated with Langmuir waves with inhomogeneous amplitudes and the energy contained within it. The initial electrostatic field perturbation is $E(x,0)/E_0 = \sin(\pi x/30)$.

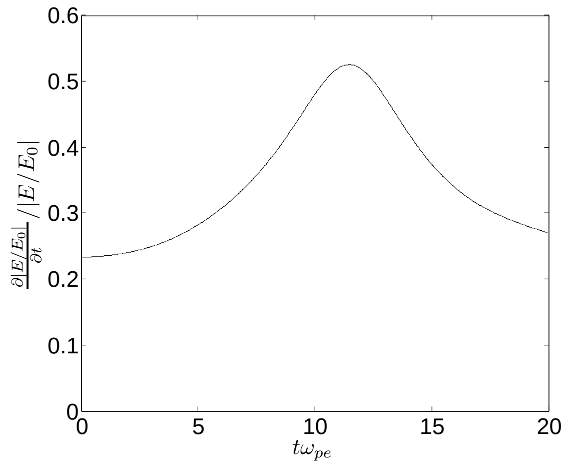
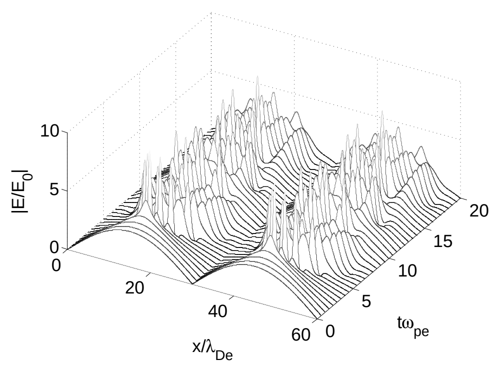


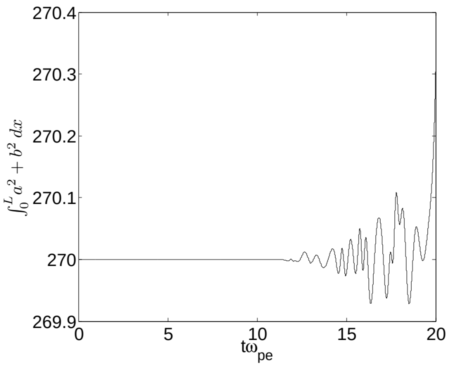
Figure 6.26: The ratio of $\frac{\partial}{\partial t}|E/E_0|$ to E/E_0 for the electrostatic field perturbation associated with Langmuir waves shown in Figure 6.25(a).

uncertainties related to such initial conditions.

We have tested the assumption $\frac{\partial}{\partial t}|E| << \omega_{pe}|E|$ also for the electrostatic field perturbation shown in Figure 6.27(a). It is shown in Figure 6.28. Unfortunately the ratio is larger than one at all times, so our assumption is clearly



(a) The electrostatic field.



(b) The energy contained within the electrostatic field.

Figure 6.27: Nonlinear evolution of the electrostatic field associated with Langmuir waves with inhomogeneous amplitudes and the energy contained within it. The initial electrostatic field perturbation is $E(x,0)/E_0 = 3\sin(\pi x/30)$.

wrong. The field shown in Figure 6.27(a) is therefore not representative for the wave-envelope of a corresponding full-wave solution.This is related to our derivation of the NLS (4.29). The ion acoustic wave equation (4.26) we used to obtain the approximation $\bar{n}/n_0\ |\bar{E}|^2$ is linearized. When we perturb the electrostatic field to the extremities in Figure 6.27(a) the linearized theory is therefore no longer valid, and we can not assume any of our assumptions to be correct for these cases.

In Figure 6.29(a) we have solved the NLS for the same initial condition as for the known analytical solution of the linear Schrödinger equation in Figure 6.12(a). We have also set the ponderomotive force equal to zero and thereby solved the linear Schrödinger equation so as to obtain the same result for the electrostatic perturbation, Figure 6.29(b), as we did in Figure 6.12(b) for the electron density perturbation. The two solutions, one for the nonlinear and one for the linear Schrödinger equation, appear to be identical and as the energy is conserved for both of them they are physically plausible.

By subtracting the linear solution from the nonlinear, Figure 6.30,we see that the two solutions are not exactly identical, but it is clear that the nonlinearity introduced by the ponderomotive force is not very manifested for this initial condition. This is because the energy in the electrostatic field is not very large and thus the ratio (4.30) between the diffusive term and the nonlinear term in the NLS becomes so large that the diffusive term dominates

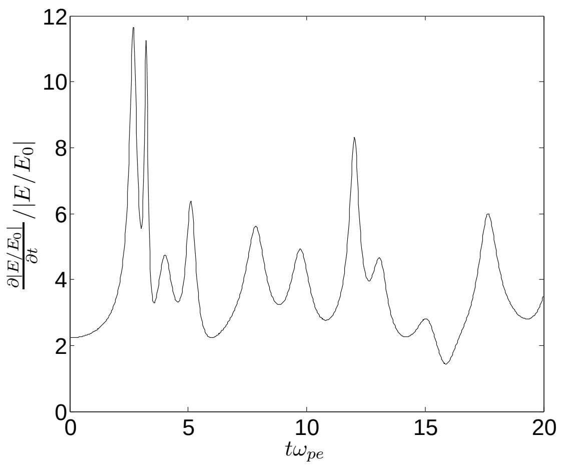
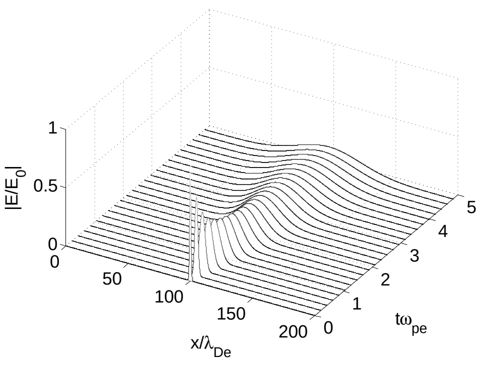


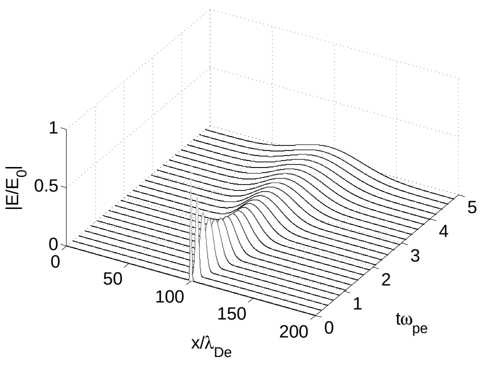
Figure 6.28: The ratio of $\frac{\partial}{\partial t}|E/E_0|$ to E/E_0 for the electrostatic field perturbation associated with Langmuir waves shown in Figure 6.27(a).

the evolution of the electrostatic perturbation. Therefore the propagation of the electrostatic field perturbation in Figure 6.29(a) is almost identical to the linear counterpart.

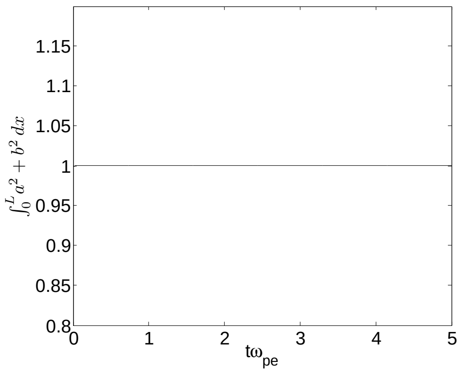
When we increase the energy in the initial perturbation of the electrostatic field we see that the nonlinear solution, as expected, clearly differs from the linear solution, Figures 6.31 and 6.32. The nonlinear field perturbation does not broaden in the spatial domain in the same manner as the linear perturbation.



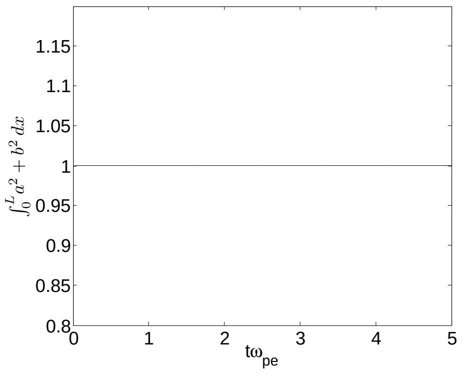
(a) The electrostatic field given by the NLS.



(b) The electrostatic field given by the linear Schrödinger equation.



(c) Energy contained in the electrostatic field given by the NLS.



(d) Energy contained in the electrostatic field given by the linear Schrödinger equation.

Figure 6.29: The electrostatic field perturbation, $|E/E_0|$, associated with Langmuir waves solved with the NLS and the linear Schrödinger equation for the initial perturbation $E/E_0 = e^{-\frac{\pi}{2}x^2}$.

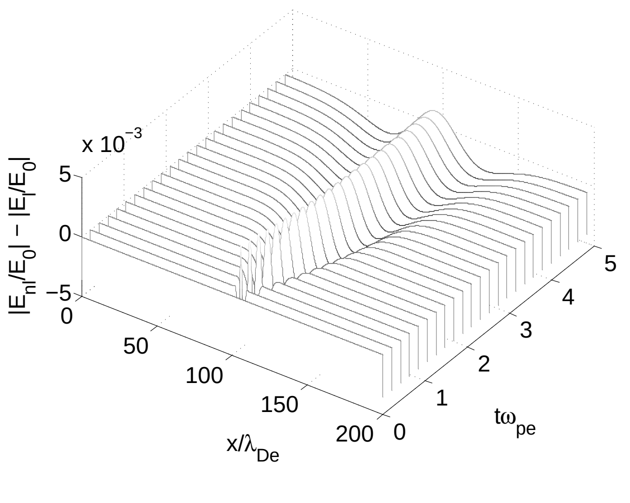
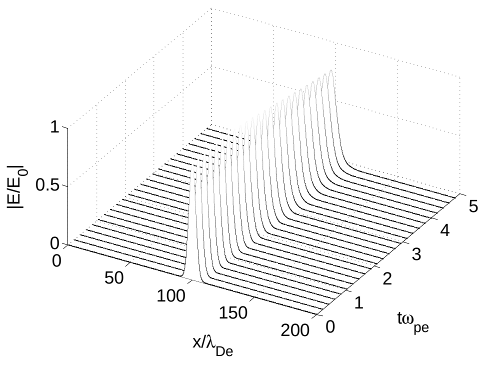
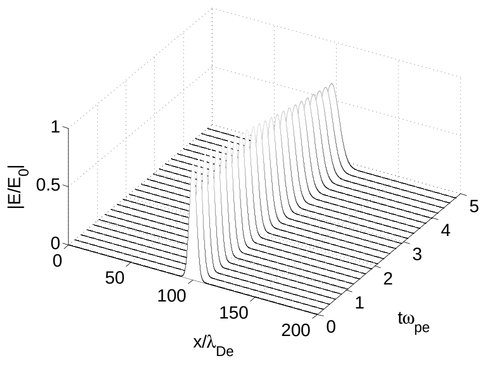


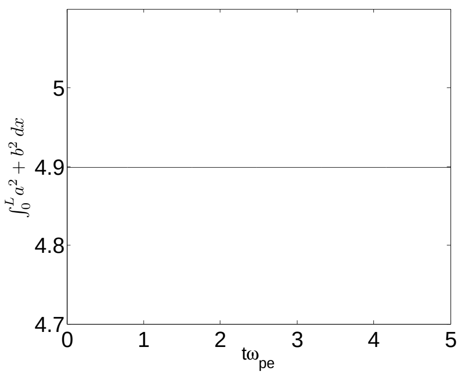
Figure 6.30: The linear result $|E_l/E_0|$ from Figure 6.29(b) subtracted from the non-linear result $|E_{nl}/E_0|$ from Figure 6.29(a).



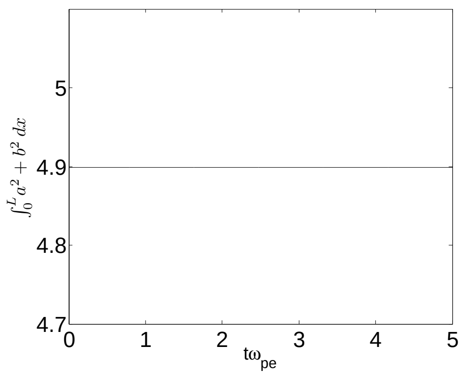
(a) The electrostatic field given by the NLS.



(b) The electrostatic field given by the linear Schrödinger equation.



(c) Energy contained in the electrostatic field given by the NLS.



(d) Energy contained in the electrostatic field given by the linear Schrödinger equation.

Figure 6.31: The electrostatic field perturbation, $|E/E_0|$, associated with Langmuir waves solved with the NLS and the linear Schrödinger equation for the initial perturbation $E/E_0 = e^{-\frac{\pi}{2}\frac{x^2}{24}}$.

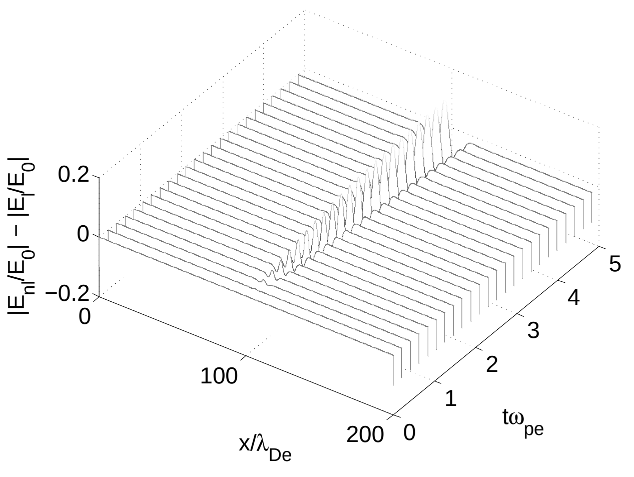
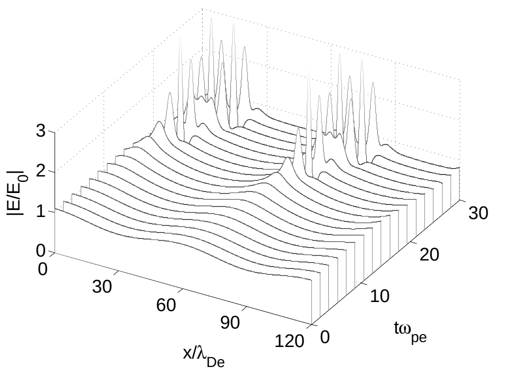


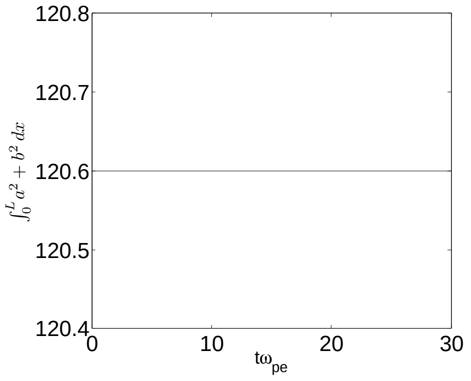
Figure 6.32: The linear result $|E_l/E_0|$ from Figure 6.31(b) subtracted from the non-linear result $|E_{nl}/E_0|$ from Figure 6.31(a).

Modulational instability

In Figure 6.33(a) we have given the initial perturbation of the electrostatic field a slightly modulated perturbation. This modulation increases with time. As we noted in section 4.5.2 the nonlinear frequency shift, $\delta\omega$, for Langmuir waves described by the NLS is negative. We also found the derivative of the group velocity, u_g', for Langmuir waves to be positive. This means that the waves are modulationally unstable by the Lighthill criterion, section 4.1, and therefore the modulation grows with time.



(a) The electrostatic field.



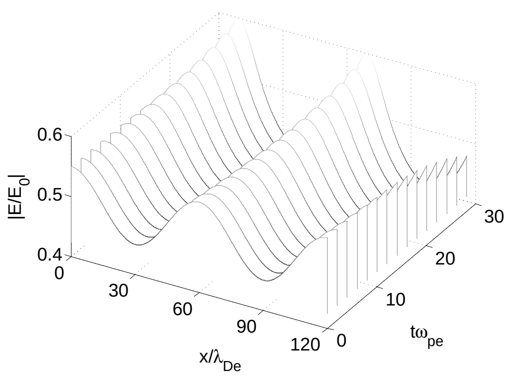
(b) The energy contained within the electrostatic field.

Figure 6.33: The electrostatic field associated with Langmuir waves with inhomogeneous amplitudes and the energy contained within it. The initial electrostatic field perturbation is given a slight modulation as $E(x,0)/E_0 = (1 + 0.1\cos\pi x/10)\,[\cos(\pi x/20) + i\sin(\pi x/20)]$.

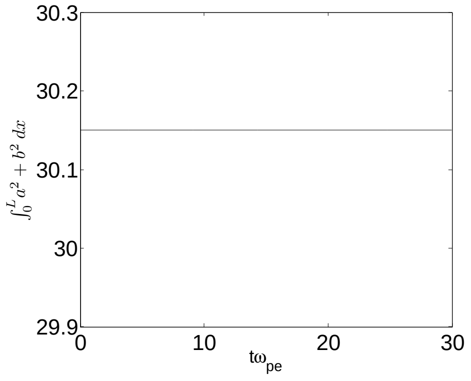
The electrostatic field perturbation shown in Figure 6.34 was initially given half the amplitude of the perturbation in Figure 6.33. Here the growth of the wave modulation is not as pronounced as for the wave with twice the amplitude.

Figure 6.35 is an illustration of a wave that is given an even smaller initial amplitude than the one in Figure 6.34. Here it is not possible to see any growth of the wave modulation, at least not with the naked eye. Note that the Figures 6.33, 6.34 and 6.35 have different scales.

Comparing the results in Figures 6.33, 6.34 and 6.35 we see that reducing the amplitude of the "carrier" wave of the electrostatic perturbation decreases the modulational growth with time. This is, as mentioned in Chapter 4,

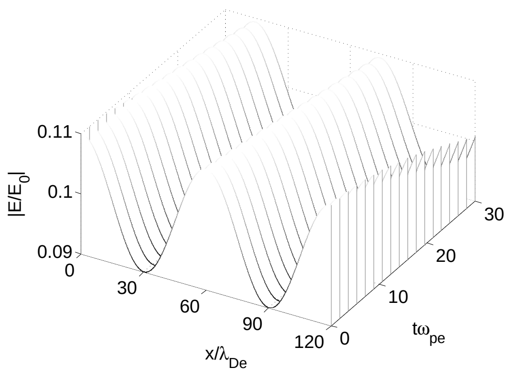


(a) The electrostatic field.

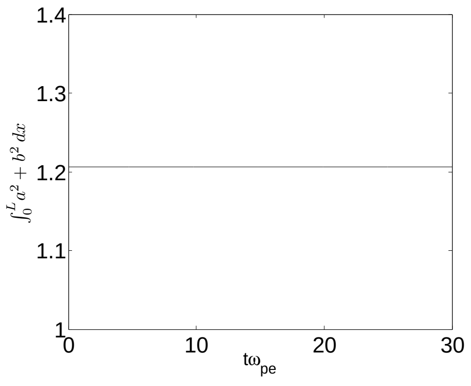


(b) The energy contained within the electrostatic field.

Figure 6.34: The electrostatic field associated with Langmuir waves with inhomogeneous amplitudes and the energy contained within it. The initial electrostatic field perturbation is given a slight modulation as $E(x,0)/E_0 = 0.5(1 + 0.1\cos\pi x/10)\,[\cos(\pi x/20) + i\sin(\pi x/20)]$.



(a) The electrostatic field.



(b) The energy contained within the electrostatic field.

Figure 6.35: The electrostatic field associated with Langmuir waves with inhomogeneous amplitudes and the energy contained within it. The initial electrostatic field perturbation is given a slight modulation as $E(x,0)/E_0 = 0.1(1 + 0.1\cos\pi x/10)\,[\cos(\pi x/20) + i\sin(\pi x/20)]$.

because the unstable wavelengths of the modulation becomes longer and longer as the amplitude of the wave is made shorter.

Chapter 7

Discussion and Conclusion

We studied standard fluid models for Langmuir waves and extended them to include multiple electron components. Our analysis was equivalent to the so-called multiple water-bag model, which includes a simplified model for the kinetic Landau damping. We further generalized the fluid description to include nonlinear effects. Both analytical and numerical results and the methods used to obtain them were presented.

Our nonlinear analysis includes more of the complexity of a plasma than the linear counterpart and the inclusion of several electron components further adds to the complexity, even so it is still the product of several simplifying assumptions. Our study therefore illustrates some of the basic properties of a multi electron component plasma. Here we address the importance of the simplifications on the final results. We also discuss the future research possibilities that points out from our work.

7.1 Water-bags

Our analysis of multi-electron component plasmas was based on one special box approximation to the Maxwell-Boltzmann electron velocity distribution, see section 2.3 and Figure 2.3(b). We could have approximated the Maxwellian by other means, for example by Heaviside step functions as suggested by Rowlands [26] and Navet & Bertrand [27]. Their step functions give equal heights for all the boxes in the approximation, whereas our model ensures that the horizontal difference between all box boundaries is the same.

A slight advantage of our model is that the equal spacing between the limiting velocities, u_r, of the water-bags makes the summation in the dispersion relation (3.6) very close to the integral in the kinetic dispersion relation (3.4). This makes the similarity between the water-bag model and the kinetic model more intuitive.

We found that our water-bag model supports linear Langmuir wave solutions that exhibit damping very close to analytical estimates for linear Landau damping, at least when all the electron components are given identical initial density perturbations. We also point out that the choice of initial conditions is of crucial importance to the water-bag models possibility for reproducing Landau damping. As we saw, some initial conditions can even give undamped plane waves. However, this is, as mentioned, also known from the kinetic description of plasmas and therefore further illustrates how the water-bags relate to the kinetic model.

In addition to the importance of the initial distribution of electron density perturbations for the damping of the waves, we also found that the efficiency of the damping is dependent on the number of water-bags in the plasma. The more water-bags we have the longer time interval over which the damping of the wave corresponds to Landau damping. Damping is induced in the water-bag model through the mixing of the phases of the different oscillations, it is this process that becomes more efficient for an increasing number of water-bags. The essential result is that if we let the number of water bags go to infinity the Langmuir waves propagating in the plasma will be Landau damped.

Navet & Bertrand [27] also found that their water-bag model supports wave damping at the same rate as analytical estimates for Landau damping of linear Langmuir waves. They saw the same correspondence between the efficiency of this damping and the number of water-bags as us. It is however not clear how their model is affected by the choice of initial conditions. Our results therefore extend and generalize their analysis.

7.2 Linear Langmuir wave-envelopes

When we derived the linear Schrödinger equation we factorized the analytical expression to the wave with the electron density, as $n(x,t)e^{-i\omega_{pe}t}$, assuming $n(x,t)$ to be slowly varying with time. Hereby we removed all oscillations at the plasma frequency, ω_{pe}, so that the remaining electron density perturbations would describe the slowly varying amplitude part, or the wave-envelope, of Langmuir waves. In other words we assumed this assumption would sufficiently remove all high frequency oscillations.

For small wave numbers in the single electron component model we found this to be a valid assumption. For large wave numbers, however, it is obvious from the dispersion relation in Figure 6.2 that there will be oscillations at higher frequencies than ω_{pe}.

As we expand the analysis to the water-bag model the assumption of ω_{pe} as the only high frequency of the density oscillations of the individual electron components, still holds for low wave numbers as long as we only consider plasmas with a few electron components. For a higher number of electron components, but still for oscillations at low wave numbers, we find that the assumption is not satisfied for several of the components. We know from their dispersion relation, illustrated by Figure 6.4, that this is because plasmas with many electron components support oscillations at frequencies very close to ω_{pe} even for low wave numbers. The Schrödinger equation consequently no longer correctly gives the envelope of Langmuir waves in the plasma.

A slight increase in the wave number of the Langmuir waves to the values relevant for Landau damping resulted in even fewer components fulfilling the assumption, as seen by comparison of Figures 6.20 and 6.23(b). The set of linear Schrödinger equations describing the N electron component plasma therefore does not fulfill the required assumptions for describing the Langmuir wave-envelope, and they therefore do not reproduce the Landau damping seen for the full wave solutions.

It is plausible that we could obtain some improvement by using a frequency different from ω_{pe} in the factorization mentioned before, i.e. have $n(x,t)e^{-i\omega_\star t}$ with $\omega_\star \neq \omega_{pe}$, but this will imply one more term appearing in the basic equation for $n(x,t)$. The possible choices for $\omega_\star$ are also limited by our aim to describe Langmuir waves, so $\omega_\star$ can not be too different from ω_{pe}. For plasmas with a high number of electron components the $\omega - k$ plane is densely covered by dispersion branches, see Figure 6.4. In particular the highest dispersion branches are very close to each other. In practice this means that it will be difficult to remove all high frequency oscillations even through the factorization $n(x,t)e^{-i\omega_\star t}$. This makes it hard to adequately describe Langmuir wave-envelopes in such plasmas and, for relevant wave numbers, the Landau damping contained in it.

7.3 Weakly nonlinear Langmuir waves

Concerning the nonlinear effects, we studied first the simple cold plasma model with immobile ions by analytical means. This problem was solved exactly in terms of Lagrangian coordinates [32], but we found that this model did not give any nonlinear frequency shift, only the waveform itself was changed by the nonlinear effects, and not its time of recurrence which remained to be the electron plasma period.

To obtain a nonlinear frequency shift it was necessary to allow for mobile ions through the ponderomotive force, thereby allowing for changes in the local plasma density. This gave rise to changes in the local plasma frequency. In particular we found that the weakly nonlinear Langmuir waves became modulationally unstable in this case. The basic equation for these nonlinear Langmuir waves is know in the literature as the Nonlinear Schrödinger (NLS) equation and is a special case of the Zakharov model [10].

The NLS describes the slowly varying part of the electrostatic field associated with the Langmuir waves. To obtain this description we made three fundamental assumptions:

1. High frequency electron oscillations to first order (i.e. excluding thermal and nonlinear corrections) oscillate at the plasma frequency.
2. The time evolution of the slowly varying bulk plasma density is so slow that the time derivative in the driven ion sound equation (4.26) can be neglected.
3. The electrostatic field perturbation is the product of a slowly varying amplitude part, $\bar{E}(x,t)$, and an oscillation at the plasma frequency, $e^{-i\omega_{pe}t}$.

The second assumption is important for the validity of our results. We have not tested it. In future analysis this could be done by solving the two coupled equations (4.21) and (4.26), where the former describes the electrostatic field perturbation associated with Langmuir waves in a plasma where ion dynamics is included and the latter the slowly varying bulk plasma density perturbation in the same plasma. From the arguments for the ponderomotive force it is however a plausible assumption.

The third and last assumption is essentially the same that we used when we derived the linear Schrödinger equation for the wave-envelopes of linear Langmuir waves. We found that for relevant amplitudes this assumption is justified in single electron components plasma, i.e. the simple water-bag model.

7.3.1 Generalization of the Zakharov model

Attempting to combine the nonlinear model with the multiple water-bag set of equations obtained for describing Landau damping in a fluid model, we encountered some difficulties. Implicit in the NLS equation is an assumption of separation of time-scales, i.e. by the assumption of $\frac{\partial}{\partial t}E \ll \omega_{pe}E$. We, as mentioned, found that for a single water bag model, which exactly reproduces the know results from a simple fluid model, this basic assumption is usually fulfilled. However, for a multiple water-bag model we found that we can have even $\frac{\partial}{\partial t}E > \omega_{pe}E$ in long time intervals for a significant number of the constituent water-bag equations. (What we actually demonstrated was an equivalent result, $\frac{\partial}{\partial t}\phi > \omega_{pe}\phi$). Although we presented a complete generalized model corresponding to the NLS-equation, it seems that this basic assumption is violated for those cases we studied here. Even if these generalized model equations can have interest, they do not seem promising as accurate models for generalizing NLS-equations to include Landau damping effects.

We obtained our generalized model for the special case of the Zakharov model by the same arguments as for the generalization of the basic fluid model to multiple electron components. There was, however, one difference. We chose a reference frequency, $\omega_\star$, for the oscillation that was different from the plasma frequency, ω_{pe}. Considering our discussion of reference frequencies for the linear Langmuir envelopes it is possible that this choice of reference frequency makes the model somewhat more likely to reproduce Landau damping qualitatively.

We know from our discussion of linear Langmuir wave-envelopes that the linear Schrödinger equation gives a relatively poor description of these envelopes when we consider Langmuir waves in plasmas with multiple electron components. The nonlinear version can in this sense not be expected to give better approximations. With an appropriate choice of reference frequency it is however our belief that the generalized Zakharov model will show, at least, the basic features of Landau damping.

The generalized Zakharov model (4.36) is not on the same form as the NLS (4.29) we found for the single electron component description. First of all it describes the electron density perturbations of each component rather than the electrostatic field associated with them. Secondly, it is not straight forward to introduce the same quantity in the nonlinear term as we did in the NLS (4.29). This requires the Poisson equation to be solved explicitly. The numerical solution of the generalized Zakharov model is therefore not as easily obtained as for the single electron component Zakharov model.

Finally, there is the possibility of initial large amplitude wave conditions, where Landau damping is insignificant for shorter time intervals. This interval might be well described by a single fluid (i.e. single water-bag) model. However, for larger times, we might have significant density depletions developed by the nonlinearity, and, as seen in e.g. Figure 6.24 we can have a local region of short wavelength Langmuir waves developing. These will experience a locally significant Landau damping, which can be qualitatively described by multiple water-bag models as those studied here.

7.4 Future perspectives

Numerical methods

The numerical methods used in this thesis provided accurate and computationally stable solutions. Despite this one particular feature of the numerical solutions of the linear and nonlinear Schrödinger equations deserves a comment.

We developed explicit finite difference schemes to solve the linear Schrödinger equations. These equations described the wave-envelope of Langmuir waves in N electron component plasmas. In a single electron component plasma the explicit scheme had stability condition $C < 1/4$ for the Courant number $C = 3\Delta t/2h^2$, described by the spatial and temporal grid parameters h and Δt. This was an acceptable condition with regards to the computational efficiency. When we expanded the analysis to a multi electron component plasma the stability condition for the Courant number $C = \Delta t/h^2$ became $C \leq 1/(4\alpha(N)+Nh^2\beta(N)-h^2/2)$ for $r = 1, 2, ..., N$. For $N = 1$ the stability condition for single electron component plasmas is reproduced ($\alpha(1) = 3/2$ and $\beta(1) = 1/2$). Clearly the stability condition becomes more severe when the number of components increase. In this thesis we have not solved more than 25 coupled Schrödinger equations. If we were to solve it for a higher number of components an implicit scheme would, despite the increased complexity of the implementation, be much more computationally efficient. For this purposes the suggested implicit schemes by Fei et. al [51] and Ismail et. al [52] can be implemented. These schemes are developed for coupled *nonlinear* schemes and can therefore also be employed for future solutions of the generalized Zakharov model. The schemes differs from more traditional methods since they do not require iterative solvers, this makes them implementationally favorable and also computationally efficient.

Two component Maxwellians

There exists plasmas in nature whose particle distribution function is a composite of two Maxwell-Boltzmann distributions, i.e. they have one warm and one colder electron component. This is for example often observed in the polar ionosphere, where the energetic component can be precipitating electrons. These cases can be approximated by a two-bag water-bag model. Our work is then directly applicable when the two coupled differential equations of the generalized fluid model are refrased for the appropriate systems. In particular, we may have to include a bulk drift of one of the components. This additional complication has not been included in the present thesis.

Non-thermal distributions

The water-bag formalism is in principle applicable as an approximation to any particle distribution function. We chose the Maxwell-Boltzmann distribution because of our interest in Landau damping.

In future works water-bag models can be implemented for particle distributions that possibly give unstable waves. The concepts of the nonlinear evolution of the instabilities could then be studied. As for the present study, this would be a computationally easier approach than the kinetic alternative.

Langmuir wave decay

We mentioned above that the two coupled equations (4.21) and (4.26) could be solved in order to test the assumption of a negligible time derivative of the slowly varying bulk plasma density in a plasma where ion dynamics are included. These two equations also describe a phenomenon known as "Langmuir wave decay". The decay refers to the fact that the solution of these two equations may give unstable Langmuir waves. For the right choice of parameters the wave can decay into another Langmuir wave and a low frequency ion acoustic wave [33]. It would be interesting to study the nonlinear evolution of the decay as a reference problem as it has already been observed in nature as well as in laboratory plasmas. This problem can be studied also in one spatial dimension.

Two and three spatial dimensions

This thesis is a purely one-dimensional study. The analysis can be extended to two- and three-dimensions. All spatial derivatives will then become vector operators. This means a significant additional book-keeping of variables, velocities as vector variables, but otherwise no conceptually new problems. It is however known that the evolution of nonlinear Langmuir waves differs significantly from the simple one dimensional problem by allowing for so called wave-collapse phenomena, when it is extended to two or three spatial dimensions [10]. It might therefore be worthwhile to make two and three dimensional extensions of the present work, as new phenomena can be found.

Nonlinear Landau damping

Linear Landau damping, as studied here, is only applicable for limited time intervals and moderate amplitude plane waves. For analysis outside these limits nonlinear Landau damping therefore has to be considered. The bouncing of the trapped particles discussed in Chapter 3 will be important for such studies. In the water bag model the equivalent of trapped particles is heat flow [33]. When heat flow is introduced in the water-bag model it no longer reproduces the fluid description of plasmas. Thus it is not favorable to investigate nonlinear Landau damping in a multi-water bag model. In the special field of gyrokinetics there exists, however, successful applications of a generalized version of a two-dimensional water-bag model [13].

"Chaque probléme que j'ai résolu est devenu une règle, qui a servi après, a résoudre d'autres problémes."

Rene Descartes, Discours de la méthode (1637)

Appendix A

Stability and accuracy analysis

In the main text we omitted the calculation of the stability criteria and truncation errors for the various numerical schemes. These are included here.

A.1 N coupled Langmuir wave equations

Stability criterion

We can solve the N coupled Langmuir wave equations (5.8) numerically by the finite difference scheme

$$\left[\delta_t\delta_t\hat{n}_r - \alpha(r)\delta_x\delta_x\hat{n}_r + \beta(r)\sum_{r=1}^{N}\hat{n}_r = 0\right]_j^l$$
$$\text{for } r = 1, 2, ..., N \text{ and } 1 < j < m \text{ and } l \geq 0. \qquad \text{(A.1)}$$

The coefficients $\alpha(r)$ and $\beta(r)$ are given in section 5.4.2. Inserting a solution on the form $\hat{n}_{r,j}^l = e^{-i(\tilde{\omega}l\Delta t - kjh)}$ in the scheme we find

$$-4\sin^2\frac{\tilde{\omega}\Delta t}{2} = C^2\left(4\alpha(r)\sin^2\frac{kh}{2} - Nh^2\beta(r)\right)$$

$$\Rightarrow \quad \tilde{\omega} = \frac{2}{\Delta t}\arcsin\left[\sqrt{C^2\left(4\alpha(r)\sin^2\frac{kh}{2} - Nh^2\beta(r)\right)}\right],$$

where $C = \Delta t/h$ is the Courant number. The Langmuir waves we want to simulate does not exhibit exponential damping or amplification in nature. Therfore we require a real numerical dispersion relation, that is $\tilde{\omega} \in \mathcal{R}$. This gives

$$\left| \sqrt{C^2 \left(4\alpha(r) \sin^2 \frac{kh}{2} - Nh^2\beta(r) \right)} \right| \leq 1$$

from which we find

$$C \leq \frac{1}{\sqrt{\left(4\alpha(r) \sin^2 \frac{kh}{2} - Nh^2\beta(r)\right)}}$$

and in its strictest form

$$C \leq \frac{1}{\sqrt{(4\alpha(N) - Nh^2\beta(N))}}.$$

This is the stability criterion for the above scheme. If we set $N = 1$ the criterion reduce to the one found for Langmuir waves in a single electron component plasmas in section 5.4.1.

Truncation error

The truncation error of the scheme (A.1) is found by inserting the continuous solutions, u_r, to the set of coupled Schrödinger equations (5.8):

$$\tau = [\delta_t\delta_t n_r - \alpha(r)\delta_x\delta_x n_r + \beta(r)\sum_{r=1}^{N} n_r = 0]_j^l$$
$$\text{for } r = 1, 2, ..., N \text{ and } 1 < j < m \text{ and } l \geq 0.$$

We make a Taylor expansion of n_r around the nodes, (x_j, t_l), in the time-space grid to find

$$\tau \approx \left[\frac{\partial^2 n_r}{\partial t^2}\right]_j^l + \frac{\Delta t^2}{12}\left[\frac{\partial^4 n_r}{\partial t^4}\right]_j^l + \mathcal{O}(\Delta t^4)$$
$$- \alpha(r)\left(\left[\frac{\partial^2 n_r}{\partial x^2}\right]_j^l + \frac{h^2}{12}\left[\frac{\partial^4 n_r}{\partial x^4}\right]_j^l + \mathcal{O}(h^4)\right) + \beta(r)\sum_{r=1}^{N} n_{r,j}^l.$$

The terms in the above expressions that together correspond to (5.8) can be cancelled; we know that u_r excatly solve the coupled wave equations and this must be valid also at the grid points (x_j, t_l). The remaining terms are then to lowest order $\mathcal{O}(h^2, \Delta t^2)$ and we see that the truncation error will go to zero when $h \to 0$ and $\Delta t \to 0$. Also here the single electron component result is found by setting $N = 1$.

A.2 N coupled linear Schrödinger equations

A.2.1 One sided forward difference in time

Stability criterion

A set of N coupled linear Schrödinger equations, as given by (5.15), can be solved by the finite difference scheme

$$\left[\delta_t^+ \hat{\boldsymbol{n}}_{\boldsymbol{r}} + \alpha(r) A \delta_x \delta_x \hat{\boldsymbol{n}}_{\boldsymbol{r}} + \frac{1}{2} A \hat{\boldsymbol{n}}_{\boldsymbol{r}} - \beta(r) A \sum_{r=1}^{N} \hat{\boldsymbol{n}}_{\boldsymbol{r}} = 0\right]_j^l \tag{A.2}$$

$$\text{for } r = 1, 2, ..., N,\ 1 < j < m \text{ and } l = 0, \tag{A.3}$$

where

$$\boldsymbol{n}_{\boldsymbol{r}} = \begin{pmatrix} a_r \\ b_r \end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

We define the numerical error as $\boldsymbol{e}_{\boldsymbol{r}} = \boldsymbol{n}_{\boldsymbol{r}} - \hat{\boldsymbol{n}}_{\boldsymbol{r}}$, where $\boldsymbol{n}_{\boldsymbol{r}}$ is the exact solution to the continuous Schrödinger equations. This gives the discretized error equation

$$\left[\delta_t^+ \hat{\boldsymbol{e}}_{\boldsymbol{r}} + \alpha(r) A \delta_x \delta_x \hat{\boldsymbol{e}}_{\boldsymbol{r}} + \frac{1}{2} A \hat{\boldsymbol{e}}_{\boldsymbol{r}} - \beta(r) A \sum_{r=1}^{N} \hat{\boldsymbol{e}}_{\boldsymbol{r}} = 0\right]_j^l$$

$$\text{for } r = 1, 2, ..., N,\ 1 < j < m \text{ and } l = 0.$$

Assuming the error to be on the form

$$\hat{\boldsymbol{e}}_j^l = e^{-i(\tilde{\omega} l \Delta t - kjh)} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = e^{i(kjh)} \boldsymbol{\xi}^l,$$

and inserting it in the discretized error equation, we find

$$\boldsymbol{\xi} = C\left(6\alpha(r)\sin^2\frac{kh}{2}\begin{pmatrix}1\\-1\end{pmatrix} - \frac{h^2}{2}\begin{pmatrix}1\\-1\end{pmatrix} + h^2N\beta(r)\begin{pmatrix}1\\-1\end{pmatrix}\right) + \begin{pmatrix}1\\1\end{pmatrix},$$

where $C = \Delta t/h^2$ is the Courant number. The error must be bounded in time so we set $\boldsymbol{\xi} \leq \mathbf{1}$ and obtain

$$C\left(6\alpha(r)\sin^2\frac{kh}{2}\begin{pmatrix}1\\-1\end{pmatrix} - \frac{h^2}{2}\begin{pmatrix}1\\-1\end{pmatrix} + h^2N\beta(r)\begin{pmatrix}1\\-1\end{pmatrix}\right) \leq \begin{pmatrix}0\\0\end{pmatrix},$$

which effectively means $C \leq 0$. The one sided finite difference in time therefore results in a scheme for the coupled linear Schrödinger equations that is unstable for any choice of grid parameters h and Δt.

For $N = 1$ (5.15) reduces to one linear Schrödinger equation and the stability criteria for the corresponding scheme from section 5.4.3 is reproduced, i.e. it is also unstable.

Truncation error

We insert the continious solution $\boldsymbol{n_r}$ to the set of coupled linear Schrödinger equations in the scheme (A.3) to find the truncation error, τ:

$$\tau = \left[\delta_t^+\boldsymbol{n_r} + \alpha(r)A\delta_x\delta_x\boldsymbol{n_r} + \frac{1}{2}A\boldsymbol{n_r} - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r} = 0\right]_j^l$$

for $r = 1, 2, ..., N$, $1 < j < m$ and $l = 0$.

Taylor expansion of $\boldsymbol{n_r}$ around the grid points (x_j, t_l) yields

$$\tau \approx \left[\frac{\partial\boldsymbol{n_r}}{\partial t}\right]_j^l + \frac{\Delta t}{2}\left[\frac{\partial^2\boldsymbol{n_r}}{\partial t^2}\right]_j^l + \mathcal{O}(\Delta t^2) + \alpha(r)A\left(\left[\frac{\partial^2\boldsymbol{n_r}}{\partial x^2}\right]_j^l + \frac{h^2}{12}\left[\frac{\partial^4\boldsymbol{n_r}}{\partial x^4}\right]_j^l + \mathcal{O}(h^4)\right)$$
$$+ \frac{1}{2}A\boldsymbol{n_r}{}_j^l - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r}{}_j^l.$$

When the terms that corresponds to (5.15) are cancelled as they are exactly solved by $\boldsymbol{n_r}$, we have a remainder of order $\mathcal{O}(h^2, \Delta t)$. The truncation error tends to zero in the limit of $h \to 0$ and $\Delta t \to 0$.

A.2.2 Leap-Frog

Stability criterion

We assume discrete solutions on the form ${\boldsymbol{n_r}}^l_j = \exp\left[-i(\tilde{\omega} l\Delta t - kjh)\right]$ for a set of N coupled linear Schrödinger equations and insert it in the Leap-Frog scheme

$$\left[\delta_{2t}\boldsymbol{n_r} + \alpha(r)A\delta_x\delta_x\boldsymbol{n_r} + \frac{1}{2}A\boldsymbol{n_r} - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r} = 0\right]_j^l \tag{A.4}$$

$$\text{for } r = 1, 2, ..., N,\ 1 < j < m \text{ and } l > 0. \tag{A.5}$$

The result is

$$2i\sin\tilde{\omega} l\Delta t = C\left(8\alpha(r)\sin^2\frac{kh}{2}\begin{pmatrix}1\\-1\end{pmatrix} - h^2\begin{pmatrix}1\\-1\end{pmatrix} + 2Nh^2\beta(r)\begin{pmatrix}1\\-1\end{pmatrix}\right),$$

where the Courant number $C = \Delta t/h^2$ was introduced. We are not interested in solutions of $\boldsymbol{n_r}$ that grow or decrease exponentially with time. Therefore we impose the condition $\tilde{\omega} \in \mathcal{R}$ for the numerical dispersion relation. This gives

$$\left|-iC\left(4\alpha(r)\sin^2\frac{kh}{2}\begin{pmatrix}1\\-1\end{pmatrix} - \frac{h^2}{2}\begin{pmatrix}1\\-1\end{pmatrix} + Nh^2\beta(r)\begin{pmatrix}1\\-1\end{pmatrix}\right)\right| \le \begin{pmatrix}1\\1\end{pmatrix},$$

and we end up with

$$C \le \frac{1}{4\alpha(r)\sin^2\frac{kh}{2} + Nh^2\beta(r) - \frac{h^2}{2}}.$$

From the definitions of $\alpha(r)$ and $\beta(r)$ in section 5.4.4 we known that they are largest for $r = N$. The most strict stability condition we can find for the Leap-Frog scheme for N coupled linear Schrödinger equations is therefore

$$C \le \frac{1}{4\alpha(N) + Nh^2\beta(N) - \frac{h^2}{2}}.$$

For $N = 1$ we recover the stability criterion fond for the Leap-Frog scheme for the linear Shrödinger equation applied to single electron component plasmas in section 5.4.3.

Truncation error

The continuous solution $\boldsymbol{n_r}$ to (5.15) is inserted in the Leap-frog scheme (A.5) to give the truncation error

$$\tau = \left[\delta_{2t}\boldsymbol{n_r} + \alpha(r)A\delta_x\delta_x\boldsymbol{n_r} + \frac{1}{2}A\boldsymbol{n_r} - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r} = 0\right]_j^l$$

$$\text{for } r = 1, 2, ..., N,\ 1 < j < m \text{ and } l = 0.$$

By Taylor expansion of $\boldsymbol{n_r}$ around the grid points (x_j, t_l) we find

$$\tau \approx \left[\frac{\partial \boldsymbol{n_r}}{\partial t}\right]_j^l + \frac{\Delta t^2}{6}\left[\frac{\partial^3 \boldsymbol{n_r}}{\partial t^3}\right]_j^l + \mathcal{O}(\Delta t^4) + \alpha(r)A\left(\left[\frac{\partial^2 \boldsymbol{n_r}}{\partial x^2}\right]_j^l + \frac{h^2}{12}\left[\frac{\partial^4 \boldsymbol{n_r}}{\partial x^4}\right]_j^l + \mathcal{O}(h^4)\right)$$
$$+ \frac{1}{2}A\boldsymbol{n_r}{}_j^l - \beta(r)A\sum_{r=1}^{N}\boldsymbol{n_r}{}_j^l.$$

We cancel the terms that are solved exactly by $\boldsymbol{n_r}$, that is the terms corresponding to (5.15), and the remaining truncation error wil then be of order $\mathcal{O}(\Delta t^2, h^2)$. The error goes to zero when the grid parameters go to zero, i.e. the scheme is consistent. The choice $N = 1$ gives the single electron component result.

A.3 The nonlinear Schrödinger equation

A.3.1 One sided forward difference in time

Stability criterion

The scheme

$$\left[\delta_t^+\hat{\boldsymbol{E}} + \frac{3}{2}A\delta_x\delta_x\hat{\boldsymbol{E}} + \beta\boldsymbol{f}(\hat{\boldsymbol{E}}) = 0\right]_j^l$$
$$\text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ... \quad \text{(A.6)}$$

for the nonlinear Schrödinger equation (5.20) has the discretized error equation

$$\left[\delta_t^+ \boldsymbol{e} + \frac{3}{2}A\delta_x\delta_x\boldsymbol{e} + \beta\boldsymbol{f}(\boldsymbol{e}) = 0\right]_j^l \quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ...$$

when we define the error as $\boldsymbol{e} = \boldsymbol{E} - \hat{\boldsymbol{E}}$, where $\boldsymbol{E}$ is the solution to the NLS and $\hat{\boldsymbol{E}}$ is the solution to the discretized version of the NLS.

We also have

$$\boldsymbol{E} = \begin{pmatrix} a \\ b \end{pmatrix}, \quad A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad \text{and} \quad \boldsymbol{f}(\boldsymbol{E}) = (\boldsymbol{E}^T\boldsymbol{E})A\boldsymbol{E}.$$

Furthermore we assume the discretized error to be on the form

$$\boldsymbol{e}_j^l = e^{-i(\tilde{\omega}l\Delta t - kjh)}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = e^{i(kjh)}\boldsymbol{\xi}^l.$$

Inserted in the error equation this gives

$$\boldsymbol{\xi} = C\left(6\sin^2\frac{kh}{2}\begin{pmatrix} 1 \\ -1 \end{pmatrix} + h^2\beta\begin{pmatrix} 1 \\ -1 \end{pmatrix}\right) + \begin{pmatrix} 1 \\ 1 \end{pmatrix},$$

where we introduced the courant number $C = \Delta t/h^2$. The error should be bounded in time so we require $\boldsymbol{\xi} \leq 1$:

$$\boldsymbol{\xi} \leq 1 \quad \Rightarrow \quad C\left(6\sin^2\frac{kh}{2}\begin{pmatrix} 1 \\ -1 \end{pmatrix} + h^2\beta\begin{pmatrix} 1 \\ -1 \end{pmatrix}\right) + \begin{pmatrix} 1 \\ 1 \end{pmatrix} \leq 1.$$

We then obtain

$$C\left(6\sin^2\frac{kh}{2}\begin{pmatrix} 1 \\ -1 \end{pmatrix} + h^2\beta\begin{pmatrix} 1 \\ -1 \end{pmatrix}\right) \leq 0 \quad \Rightarrow \quad C \leq 0.$$

Therefore the scheme (A.6) will be unstable for all choices of grid parameters Δt and h.

Truncation error

The truncation error for the scheme (A.6) is given as

$$\tau = \left[\delta_t^+ \boldsymbol{E} + \frac{3}{2} A \delta_x \delta_x \boldsymbol{E} + \beta \boldsymbol{f}(\boldsymbol{E}) = 0\right]_j^l$$
$$\text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ...,$$

where $\boldsymbol{E}$ is the solution to the continuous NLS (5.20). By Taylor expansion of $\boldsymbol{E}$ around the space-time point (x_j, t_l) we obtain

$$\tau \approx \left[\frac{\partial \boldsymbol{E}}{\partial t}\right]_j^l + \frac{\Delta t}{2}\left[\frac{\partial^2 \boldsymbol{E}}{\partial t^2}\right]_j^l + \mathcal{O}(\Delta t^2) + \frac{3}{2} A \left(\left[\frac{\partial^2 \boldsymbol{E}}{\partial x^2}\right]_j^l + \frac{h^2}{12}\left[\frac{\partial^4 \boldsymbol{E}}{\partial x^4}\right]_j^l + \mathcal{O}(h^4)\right) + \beta [\boldsymbol{f}(\boldsymbol{E})]_j^l$$

We cancel the NLS from the error, as it will be solved by $\boldsymbol{E}$. The truncation error is therefore of order $\mathcal{O}(h^2, \Delta t)$ and approaches zero when $h \to 0$ and $\Delta t \to 0$.

A.3.2 Leap-Frog

Stability criterion

A two sided difference in time gives the Leap-Frog scheme

$$\left[\delta_{2t} \hat{\boldsymbol{E}} + \frac{3}{2} A \delta_x \delta_x \hat{\boldsymbol{E}} + \beta \boldsymbol{f}(\hat{\boldsymbol{E}}) = 0\right]_j^l \quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ..., \tag{A.7}$$

for the NLS (5.20). Here we have the same matrices and vectors as for the one sided scheme above.

We assume a solution on the form of a wave train $\hat{\boldsymbol{E}}_j^l = e^{-i(\tilde{\omega} l \Delta t - kjh)}$ and insert it in the scheme (A.7) to obtain:

$$2i \sin \tilde{\omega} l \Delta t = C\left(12 \sin^2 \frac{kh}{2}\begin{pmatrix}1\\-1\end{pmatrix} + 2h^2 \beta \begin{pmatrix}1\\-1\end{pmatrix}\right),$$

where $C = \Delta t / h^2$ as before. This leads to

$$\tilde{\omega} = \frac{1}{\Delta t} \arcsin \left[-iC \left(6 \sin^2 \frac{kh}{2} \begin{pmatrix} 1 \\ -1 \end{pmatrix} + h^2 \beta \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right) \right]$$

If the numerical dispersion relation, $\tilde{\omega}$, is imaginary the solution we have prescribed will be exponentially growing or decreasing in time. This is not a feature of the wave phenomena we are interested in solving. Therefore we set

$$\left| -iC \left(6 \sin^2 \frac{kh}{2} \begin{pmatrix} 1 \\ -1 \end{pmatrix} + h^2 \beta \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right) \right| \leq 1.$$

From this we arrive at

$$C \leq \frac{1}{6 \sin^2 \frac{kh}{2} + h^2 \beta}$$

which is most severe when $\sin^2 \frac{kh}{2} = 1$, thus the stability criterion for the Leap-Frog scheme for the NLS (5.20) is

$$C \leq \frac{1}{6 + h^2 \beta}.$$

Truncation error

Finally, we insert the continuous solution to (5.20) in the Leap-Frog scheme to obtain the truncation error

$$\tau = \left[\delta_{2t} \boldsymbol{E} + \frac{3}{2} A \delta_x \delta_x \boldsymbol{E} + \beta \boldsymbol{f}(\boldsymbol{E}) = 0 \right]_j^l \quad \text{for } j = 2, 3, ..., m-1 \text{ and } l = 0, 1, ...$$

The continuous solution, $\boldsymbol{E}$, is Taylor expanded around the grid points (x_j, t_l) to give

$$\tau \approx \left[\frac{\partial \boldsymbol{E}}{\partial t} \right]_j^l + \frac{\Delta t^2}{6} \left[\frac{\partial^3 \boldsymbol{E}}{\partial t^3} \right]_j^l + \mathcal{O}(\Delta t^4) + \alpha(r) A \left(\left[\frac{\partial^2 \boldsymbol{E}}{\partial x^2} \right]_j^l + \frac{h^2}{12} \left[\frac{\partial^4 \boldsymbol{E}}{\partial x^4} \right]_j^l + \mathcal{O}(h^4) \right) + \beta [\boldsymbol{f}(\boldsymbol{E})]_j^l.$$

After the terms that are solved exactly (the NLS), we are left with a truncation error of order $\mathcal{O}(h^2, \Delta t^2)$. This error goes to zero when the grid parameters approach the same limit.

Appendix B

Source Code

Here we list all the C++ implementations of the algorithms described in Chapter 5 for the numerical solutions of linear and nonlinear Langmuir waves in different N component electron plasmas.

For all the implementations a library, *ConfigFile*, facilitating input parameters to be read from file has been imported. The source code for this library is not listed here, but can be found at http://www-personal.umich.edu/ wagnerr/ConfigFile.html.

The various solvers are constructed in the same pattern;

- A makefile which compiles the solver and creates an executable application.
- A file named "main.cpp" which sets up and solves the problem at run time.
- A file cotaining a class that defines all public parameters and functions.
- A file containing the functions.
- A input file for giving the variables for each specific problem.

B.1 The single electron component fluid model

Langmuir.h

```
#include <stdlib.h>
#include <cmath>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <sys/stat.h>
#include <sys/types.h>

using namespace std;

class Langmuir
{
  string dirname;
  string casename;
  int i;          // number of intervals on
      [0,1]
  int m;          // number of points in the
      spatial grid

  double C;       // Courant number
  double tstop;  // The length of the time
      interval
  double nint;   // amplitude of the initial
       disturbance
  int nn;         // number of nodes
  double L;       // Spatial length
  int N;          // How often should the
      result be written to file?

  double h;       //Length of the grid
      intervals


  // For the electric field solution, in
      case of a parametric instability
  double int_n;

  double* np; // n at time level l+1
  double* n;   // n at time level l
  double* nm;  // n at time level l-1

  void setIC ();          // set initial
      conditions
  void timeLoop ();
  void solveAtThisTimeStep();
  void periodicBC();
  void dumpSolution (double t); // write
      the soultion to file
  void updateDataStructures ();
  void xConst (double t);              //
       Write to file the time evolution at
       one spesific x-point

  std::ofstream fout;
  std::ofstream file;

 public:
  void scan ();
  void solveProblem ();

}; // end class Langmuir
```

Langmuir.cpp

```
#include "Langmuir.h"
#include "ConfigFile.h"

void Langmuir:: scan()
{

   //read input from file using
        configuration manager: http://www-
        personal.umich.edu/~wagnerr/
        ConfigFile.html
  ConfigFile config( "langmuir.inp" );
  config.readInto( dirname, "dirname",
      string("Untitled"));
  config.readInto( casename, "casename",
      string("Untitled"));
  i = config.read<int>( "i", 700 );
  config.readInto( C, "C", 0.7 );
  config.readInto( tstop, "tstop", 20.0 );
  config.readInto( nint, "nint", 1.0);
  nn = config.read<int>( "nn", 1);
  config.readInto( L, "L", 30.0);
  config.readInto( int_n, "int_n", 0.005);
  N = config.read<int>( "N", 10);

  m = i+1; // number of points in the
      spatial grid
  np = new double [m]; // n at time level l
      +1
  n = new double [m];    // n at time level
      l
  nm = new double [m];   // n at time level
      l-

  // Initializing the arrays
  for (int j = 0; j <= i; ++j) {
    np[j] = 0;
    n[j] = 0;
    nm[j] = 0;
  } // end for

  } // end scan

void Langmuir:: solveProblem ()
{
timeLoop();
delete [] np;
delete [] n;
delete [] nm;
}

void Langmuir:: timeLoop ()
{

  h = L/(m-1); // Length of the grid
      intervals
  cout << "h␣=␣" << h << endl;

  // write a message in case of an unstable
       solution:
  if (C > 2/(sqrt(4 - h*h))) {
    cout << "NB!␣C␣is␣to␣large,␣the␣
        solution␣will␣be␣unstable.";
    cout << flush;
  } // end if

  double dt = (C*h)/sqrt(3); // time step,
      assume alpha is equal to unity
  double t = 0;

  setIC();         //Set initial conditions
  int step_no = 0; // current step number

   // Open output directory and files
  dirname = dirname + casename;
  int directory = mkdir(dirname.c_str(),
      S_IRWXU | S_IRWXG | S_IROTH |
      S_IXOTH);
  string fName = dirname + "/" + casename +
       ".data";
  fout.open(fName.c_str());
  dumpSolution(t); // dump initial
      displacement
  // Open the file where n(x,t) will be
       stored as a function of time at a
       spescific coordinate x
  string fileName = dirname + "/" +
       casename + ".res";
  file.open(fileName.c_str());

  // dump the inital displacement to file
  xConst(t);

  while (t < tstop) {

    t += dt; // increase time by the time
         step
    step_no++; // increase step number by 1
    solveAtThisTimeStep ();
    periodicBC ();
    updateDataStructures ();
    if (step_no % N == 0) {  // write to
         file every Nth time step:
      dumpSolution(t);
      xConst(t);
    } // end if

    if (step_no % 100 == 0) {  // write a
         message every 100th step:
       cout << "time␣step␣" << step_no << "
            is␣computed,␣and␣time␣" << t <<
            "is␣reached" << "\n";
       cout << flush;               // flush
            forces immediate output
    } // end if

  } // end while

  fout.close();
  file.close(); // close the file for
       storing n(x,t) as a function of time
        at a spescific coordinate x

} // end timeloop

void Langmuir:: setIC ()
{

  int j;  // loop counter over grid points

  // Standing wave
  double initial_const = (nn*2*M_PI*h)/L;
              //help variable
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubation n(x,0)
    n[j] = nint*sin(initial_const*j);
  } // end for

  /*
  //Pulse as an initial condition (wave-
       packet)
  double initial_const = (nn*2*M_PI*h)
       /(2.0*2.0);          //help variable
  for (j = 0; j <= i; j++) {
                         // set the initial
       pertubation n(x,0)
    n[j] = nint*sin(initial_const*j)*exp
         (-(((h*j - 0.5*L)*(h*j - 0.5*L))
         /(2.0*2.0)));
  } // end for
  */

  // Set the help variable nm
  double product_a = 0.5*(2 - (C*C*h*h)
       /3.0);     // help variable
  double product_b = 0.5*C*C;
  for (j = 1; j <= i-1; j++) {
       // set the help variable nm
     nm[j] = product_a*n[j] + product_b*(n[
          j-1] - 2*n[j] + n[j+1]);
  } // end for

  /*
  // Set the help variable nm when an
       inhomogenity is included in the
```

```
       electron density
  double product_b = 0.5*C*C; // help
       variable
   for (j = 1; j <= i-1; j++) {
                        // set the help
        variable nm
     nm[j] = 0.5*(2 - C*C*h*h*(1 + int_n*
          cos((2*M_PI*j*h)/L)))*n[j] +
          product_b*(n[j-1] - 2*n[j] + n[j
          +1]);
   } // end for
   */

   // Periodic boundary conditions
 nm[0] = 0.5*(2 - (C*C*h*h)/3.0)*n[0] +
      0.5*C*C*(n[i-1] - 2*n[0] + n[1]);
  nm[i] = nm[0];


   /*
   // Periodic boundary conditions when an
        inhomogenity is included in the
        electron density
   nm[0] = 0.5*(2 - C*C*h*h*(1 + int_n*cos
        ((2*M_PI*0*h)/L)))*n[0] + 0.5*C*C*(
        n[i-1] - 2*n[0] + n[1]);
   nm[i] = nm[0];
   */

} // end setIC

void Langmuir:: solveAtThisTimeStep ()
{

  int j; // Loop counter over grid points

  // help variables
  double product_a = (2 - (C*C*h*h)/3.0);
  double product_b =  C*C;

  // update inner points according to
       finite diffference scheme:
  for (j = 1; j <= i-1; j++) {
    np[j] = product_a*n[j] - nm[j] +
         product_b*(n[j-1] - 2*n[j] + n[j
         +1]);
  } // end for

  /*
  // update inner points according to
       finite diffference scheme: when an
       inhomogenity is included in the
       electron density
  for (j = 1; j <= i-1; j++) {
    np[j] = (2 - C*C*h*h*(1 + int_n*cos((2*
         M_PI*j*h)/L)))*n[j] - nm[j] +
         product_b*(n[j-1] - 2*n[j] + n[j
         +1]);
  } // end for
  */

} // end solveAtThisTimeStep

void Langmuir:: periodicBC(){

  np[0] = (2 - C*C*h*h/(3.0))*n[0] - nm[0]
       + C*C*(n[i-1] - 2*n[0] + n[1]);
  np[i] = np[0];

  /*
  // When considering the electric field
       and there is an inhomogenity in the
       elctron plasma density
  np[0] = (2 - C*C*h*h*(1 + int_n*cos((2*
       M_PI*0*h)/L)))*n[0] - nm[0] + C*C*(n
       [i-1] - 2*n[0] + n[1]);
  np[i] = np[0];
  */

} // end periodicBC

void Langmuir:: updateDataStructures () {
  for (int j = 0; j <= i; ++j) {
```

```
  nm[j] = n[j];
  n[j] = np[j];
  } // end for
} // end updateDataStructures

void Langmuir:: dumpSolution (double t)
{
  fout << t << "␣" << h << "␣";
  for (int j = 0; j <= i; j++) {
    fout << n[j] << "␣";
  } // end for

  fout << endl;
} // end dumpSoultion

void Langmuir:: xConst (double t) {

// I want to find n(x,t) as a function of a
      specific x
  int p;

  if (nn > 0) {
    p = (3*m)/(4*nn); // Choose the
         amplitude
  } else {
    p = (3*m)/(-4*nn);
  } //end if

  if (t == 0) cout << "The␣choosen␣x␣=␣" <<
       p*h << "\n";

  file << t << "␣" << n[p] << "\n";

} // end xConst
```

main.cpp

```
# include <iostream>
# include <cmath>
#include "Langmuir.h"

using namespace std;

int main (int argc, const char* argv[])
{
  Langmuir problem;
  problem.scan(); problem.solveProblem();
  return 0;  // success
}
```

makefile

```
#makefile

app : main.o Langmuir.o ConfigFile.o
         g++ -o app main.o Langmuir.o
              ConfigFile.o

main.o: main.cpp Langmuir.h
         g++ -c main.cpp

Langmuir.o: Langmuir.cpp Langmuir.h
         g++ -c Langmuir.cpp

ConfigFile.o: ConfigFile.cpp ConfigFile.h
         g++ -c ConfigFile.cpp -pg

clean:
         rm app main.o Langmuir.o ConfigFile
              .o

# end of makefile
```

langmuir.inp

```
dirname = /mn/korona/rp-s1/margitu/Program/
     FD/Results/
 casename = sin_nn5
 i = 1000
 C = 0.4
 tstop = 15
 nint = 1
 nn = 5
 L = 60
 N = 2
```

B.2 The multi-component electron fluid model

MultiLangmuir.h

```
#include <stdlib.h>
#include <cmath>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <sys/stat.h>
#include <sys/types.h>

using namespace std;

class MultiLangmuir
{
  string casename;
  string dirname;
  int i;          // number of intervals on
       [0,1]
  int m;          // number of points in the
       spatial grid

  double C;       // Courant number
  double tstop;  // The length of the time
       interval
  double nint;   // amplitude of the initial
        disturbance
  int nn;         // number of nodes
  double L;       // Spatial length
  int M;          // How often should the
       result be written to file?
  int N;          // Number of electron
       density components

  double h;      //Length of the grid
       intervals
  double dt;      //Time step


  // For the electric field solution, in
       case of a parametric instability
  double int_n;

  double** np; // n at time level l+1
  double** n;    // n at time level l
  double** nm;   // n at time level l-1
  double* sum_n; // The sum of all n at
       time level l
  double* sum_np; // The sum of all n at
       time level l+1

  // Coefficients
  double* alpha;
  double* beta;
  // Help variables
  double* productA;
  double* productB;

  void setIC ();          // set initial
       conditions
  void timeLoop ();
  void solveAtThisTimeStep(int r);
  void periodicBC(int r);
  void dumpSolution (int r, double t); //
       write the solution to file
  void updateDataStructures (int r);
  void xConst(int r, double t); // write
       the solutiona at a spesific x point
       to file

  // Open output files
  vector<ofstream*>fileArray;  // To store
       the timevolution at all points
  vector<ofstream*>fileArrayx; // To store
       the time evolution at one spesific x
       -point

 public:
  void scan ();
  void solveProblem ();

}; // end class MultiLangmuir
```

MultiLangmuir.cpp

```
#include "MultiLangmuir.h"
#include "ConfigFile.h"

void MultiLangmuir:: scan()
{
  //read input from file using
       configuration manager: http://www-
       personal.umich.edu/~wagnerr/
       ConfigFile.html
  ConfigFile config( "multi_langmuir.inp" )
       ;
  config.readInto( dirname, "dirname",
       string("Untitled"));
  config.readInto( casename, "casename",
       string("Untitled"));
  i = config.read<int>( "i", 700 );
  config.readInto( C, "C", 0.7 );
  config.readInto( tstop, "tstop", 20.0 );
  config.readInto( nint, "nint", 1.0);
  nn = config.read<int>("nn", 1);
  config.readInto( L, "L", 30.0);
  config.readInto( int_n, "int_n", 1.0);
  M = config.read<int>( "M", 1);
  N = config.read<int>( "N", 10);

  nint = nint/N;

  m = i+1; // number of points in the
       spatial grid

  // Allocation of matrices and arrays

  np = new double*[N];
  for(register int k=0; k<N; k++)
    np[k] = new double[m];

  n = new double*[N];
  for(register int k=0; k<N; k++)
    n[k] = new double[m];

  nm = new double*[N];
  for(register int k=0; k<N; k++)
    nm[k] = new double[m];

  // The sum over all the N electron
       components
  sum_n = new double[m];
  sum_np = new double[m];

  alpha = new double[N];
  beta = new double[N];

  // Help variables
  productA = new double[N];
  productB = new double[N];

  // The total inital electron
  double sum_n_r = 0;
  for (int r = 0; r < N; r++) {
    sum_n_r = sum_n_r + (r+1)*(exp
         ((-9.0*(((r+1) -1))*((r+1) -1))
         /(2.0*N*N)) - exp((-9.0*(r+1)*(r
         +1))/(2.0*N*N)));
  } // end for

  // Initializing the density matrices and
       the coefficient arrays
  double sum_r_beta = 0; // help variable
       for computing alpha
  double sum_beta = 0;
  for (int r = 0; r < N; r++) {
    for (int j = 0; j <= i; ++j) {
      np[r][j] = 0;
      n[r][j] = 0;
      nm[r][j] = 0;
      sum_n[j] = 0;
      sum_np[j] = 0;
    } // end for
    beta[r] = ((r+1)*(exp((-9.0*(((r+1)-1))
         *((r+1) -1))/(2.0*N*N)) - exp
         ((-9.0*(r+1)*(r+1))/(2.0*N*N))))/
         sum_n_r;
    cout << "beta(" << r << ")␣=␣" << beta[
         r] << endl;
    sum_beta = sum_beta + beta[r];
    sum_r_beta = sum_r_beta + (r+1)*(r+1)*
         beta[r];
    productA[r] = 0;
    productB[r] = 0;
  } // end for

  cout << "sum_beta(r)␣=␣" << sum_beta <<
       endl;
  cout << "sum_r^2beta(r)␣=␣" << sum_r_beta
        << endl;



  for (int r = 0; r < N; r++) {
    alpha[r] = (3.0*(r+1)*(r+1))/(
         sum_r_beta);  // alpha[N-1] is
         always the largest element in this
          array
  } // end for

  // Initializing the sum over all the N
       components
  for (int j = 0; j <= i; ++j) {
    sum_n[j] = 0;
    sum_np[j] = 0;
  }// end for

  } // end scan

void MultiLangmuir:: solveProblem ()
{

  timeLoop();

  // Free memory

  delete [] alpha;
  delete [] beta;
  delete [] productA;
  delete [] productB;

  for (int r = 0; r < N; r++) {

    delete fileArray [r];
    delete fileArrayx [r];
    delete [] np[r];
    np[r] = 0;
    delete [] n[r];
    n[r] = 0;
    delete [] nm[r];
    nm[r] = 0;

  } // end for

  delete [] np;
  delete [] n;
  delete [] nm;
  delete [] sum_n;
  delete [] sum_np;

} // end solveProblem

void MultiLangmuir:: timeLoop ()
{

  h = L/(m-1.0); // Length of the grid
       intervals
  cout << "h␣=␣" << h << endl;

  // write a message in case of an unstable
        solution:
  if (C > 2.0/(sqrt(4.0*alpha[N-1] - h*h*
      beta[N-1]*beta[N-1]*N))) {
    cout << "NB!␣C␣is␣to␣large,␣the␣
         solution␣will␣be␣unstable.";
    cout << flush;
  } // end if

  dt = (C*h); // time step

  // Set the help arrays
  for (int r = 0; r < N; r++) {
    productA[r] = (alpha[r]*dt*dt)/(h*h);
    productB[r] = beta[r]*dt*dt;
  } // end for

  double t = 0;
  cout << "dt␣=␣" << dt << endl;

  setIC();          //Set initial conditions

  int step_no = 0; // current step number

  // Open output directory and files
  dirname = dirname + casename;
  int directory = mkdir(dirname.c_str(),
       S_IRWXU | S_IRWXG | S_IROTH |
       S_IXOTH);
  fileArray.resize(N);
  fileArrayx.resize(N);
  std::string casenames[N];
  std::string casenamesx[N];
  std::string s[N]; // converting int r to
       string s
  for (int r = 0; r < N; r++) {
    std::stringstream out;
    out << (r+1);
    s[r] = out.str();
    casenames[r] = dirname + "/" + casename
         + "_" + s[r] + ".data";
    casenamesx[r] = dirname + "/" +
         casename + "_" + s[r] + ".res";
    fileArray[r] = new ofstream(casenames[r
         ].c_str());
    fileArrayx[r] = new ofstream(casenamesx
         [r].c_str());
    // Dump the initial displacements
    dumpSolution(r, t);
    xConst(r, t);
  } // end for


  while (t < tstop) {

    t += dt; // increase time by the time
         step
    step_no++; // increase step number by 1
    if (step_no % 300 == 0) {  // write a
         message every 200th step:
      cout << "time␣step␣" << step_no <<
           "is␣computed,␣t␣=␣" << t << "\
           n";
      cout << flush;              // flush
           forces immediate output
    } // end if

    for (int r = 0; r < N; r++) {

      solveAtThisTimeStep (r);
      periodicBC (r);
      updateDataStructures (r);

      if (step_no % (10*M) == 0 && t < 40)
           {
      dumpSolution(r, t);
      } // end if

      if (step_no % M == 0) {  // write to
           file every Nth time step:
        xConst(r, t);
      } // end if

    } // end for

    // Update the sum over all the N
         electron components for the next
         timestep
    for (int j = 0; j <= i; ++j) {
    sum_n[j] = sum_np[j];
    sum_np[j] = 0;
    } // end for

  } // end while

} // end timeloop

void MultiLangmuir:: setIC ()
{

  int j;  // loop counter over grid points
  double initial_const[N];

  /*
  //Pulse
   for (int r = 0; r < N; r++) {
     //initial_const[r] = ((r+1)*nn*2*M_PI*
          h)/L;
    for (j = 0; j <= i; j++) {
                         // set the initial
         pertubation n(x,0)
      n[r][j] = nint*exp(-(h*j - L/2)*(h*j
           - L/2));
      sum_n[j] = sum_n[j] + n[r][j];
    } // end for
  } // end for
  */

  // Standing waves
  for (int r = 0; r < N; r++) {
    initial_const[r] = (nn*2*M_PI*h)/L;
              //help variable
    for (j = 0; j <= i; j++) {
                         // set the initial
         pertubation n(x,0)
      n[r][j] = nint*sin(initial_const[r]*j
           );
      sum_n[j] = sum_n[j] + n[r][j];
    } // end for
  } // end for

  /*
  // Different amplitudes for the different
        modes
  for (int r = 0; r < N-1; r++) {
    initial_const[r] = (nn*2*M_PI*h)/L;
              //help variable
    for (j = 0; j <= i; j++) {
                         // set the initial
         pertubation n(x,0)
      n[r][j] = (6.0/(7.0*(N-1)))*nint*sin(
           initial_const[r]*j);
      sum_n[j] = sum_n[j] + n[r][j];
    } // end for
  } // end for

  initial_const[0] = (nn*2*M_PI*h)/L;
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubation n(x,0)
    n[0][j] = 0.5*nint*sin(initial_const
         [0]*j);
    sum_n[j] = sum_n[j] + n[0][j];
  } // end for

  initial_const[N-1] = (nn*2*M_PI*h)/L;
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubation n(x,0)
    n[N-1][j] = (1.0/7.0)*nint*sin(
         initial_const[N-1]*j);
    sum_n[j] = sum_n[j] + n[N-1][j];
  } // end for
  */

  // Set the help variable nm
  for (int r = 0; r < N; r++) {
    for (j = 1; j < i; j++) {
      nm[r][j] = n[r][j] + 0.5*productA[r
           ]*(n[r][j-1] - 2*n[r][j] + n[r][
           j+1]) - 0.5*productB[r]*sum_n[j
           ];
    } // end for

    // Periodic boundary conditions
    nm[r][0] = n[r][0] + 0.5*productA[r]*(n
         [r][i-1] - 2*n[r][0] + n[r][1]) -
         0.5*productB[r]*sum_n[0];
    nm[r][i] = nm[r][0];
   } //end for

} // end setIC

void MultiLangmuir:: solveAtThisTimeStep (
     int r)
{

  int j; // Loop counter over grid points

  // update inner points according to
       finite diffference scheme:
    for (j = 1; j < i; j++) {
      np[r][j] = 2*n[r][j] - nm[r][j] +
           productA[r]*(n[r][j-1] - 2*n[r][
           j] + n[r][j+1]) - productB[r]*
           sum_n[j];
      sum_np[j] = sum_np[j] + np[r][j];
    } // end for

} // end solveAtThisTimeStep

void MultiLangmuir:: periodicBC(int r){

  np[r][0] = 2*n[r][0] - nm[r][0] +
       productA[r]*(n[r][i-1] - 2*n[r][0] +
        n[r][1]) - productB[r]*sum_n[0];
  sum_np[0] = sum_np[0] + np[r][0];
  np[r][i] = np[r][0];

} // end periodicBC

void MultiLangmuir:: updateDataStructures (
     int r) {

  for (int j = 0; j <= i; ++j) {
    nm[r][j] = n[r][j];
    n[r][j] = np[r][j];
  } // end for

} // end updateDataStructures

void MultiLangmuir:: dumpSolution (int r,
     double t)
{

  *fileArray[r] << r << "␣" << t << "␣" <<
       h << "␣";
  for (int j = 0; j <= i; j++) {
    *fileArray[r] << n[r][j] << "␣";
  } // end for
  *fileArray[r] << endl;

} // end dumpSoultion

void MultiLangmuir:: xConst (int r, double
     t) {

// I want to find n(x,t) as a function of a
       specific x
  int p;

  p = (1*m)/(4*nn); // the amplitude

  if (t == 0) cout << "The␣choosen␣x␣=␣" <<
       p*h << endl;

     *fileArrayx[r] << t << "␣" << n[r][p]
          << endl;

} // end xConst
```

main.cpp

```
# include <iostream>
# include <cmath>
#include "MultiLangmuir.h"
using namespace std;

int main (int argc, const char* argv[])
{
  MultiLangmuir problem;
  problem.scan(); problem.solveProblem();
  return 0;  // success
}
```

makefile

```
#makefile

app : main.o MultiLangmuir.o ConfigFile.o
        g++ -o app main.o MultiLangmuir.o
             ConfigFile.o

main.o: main.cpp MultiLangmuir.h
        g++ -c main.cpp

MultiLangmuir.o: MultiLangmuir.cpp
     MultiLangmuir.h
        g++ -c MultiLangmuir.cpp

ConfigFile.o: ConfigFile.cpp ConfigFile.h
        g++ -c ConfigFile.cpp -pg

clean:
        rm app main.o MultiLangmuir.o
             ConfigFile.o

# end of makefile
```

multi_langmuir.inp

```
dirname = /mn/korona/rp-s1/margitu/Program/
     Multi_Componenet_Linear/Results/L_60/
     nn_5/
 casename = multi_25
 i = 1000
 C = 0.1
 tstop = 20
 nint = 1
 nn = 5
 L = 60
M = 10
N = 25
```

B.3 Langmuir envelopes in single electron component plasmas

EnvelopeLangmuir.h

```
#include <stdlib.h>
#include <cmath>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <sys/stat.h>
#include <sys/types.h>

using namespace std;

class EnvelopeLangmuir
{
  string dirname;
  string casename;
  int i;          // number of intervals on
       [0,1]
  int m;          // number of points in the
       spatial grid

  double C;       // Courant number
  double tstop;   // The length of the time
       interval
  double nint;    // amplitude of the initial
        disturbance
  int nn;         // number of nodes
  double L;       // Spatial length
  int M;          // How often should the
       result be written to file?

  double h;          // Length of the grid
       intervals
  double dt;         // Timestep

  double* ap; // a at time level l+1
  double* a;    // a at time level l
  double* am;  // a at time level l-1
  double* bp; // b at time level l+1
  double* b;    // b at time level l
  double* bm;  // b at time level l-1

  void setIC ();                          //
       set initial conditions
  void solveAtTheFirstTimeStep();       //
       Forward Euler scheme
  void timeLoop ();
  void solveAtThisTimeStep();           //
       Leap Frog scheme
  void periodicBC();
  void dumpSolution (double t);         //
       write the soultion to file
  void xConst (double t);
  void updateDataStructures ();

  std::ofstream fouta;
  std::ofstream foutb;
  std::ofstream fileax;
  std::ofstream filebx;

 public:
  void scan ();
  void solveProblem ();

}; // end class EnvelopeLangmuir
```

EnvelopeLangmuir.cpp

```
#include "SlowlyLangmuir.h"
#include "ConfigFile.h"

void SlowlyLangmuir:: scan()
{
  //read input from file using
       configuration manager: http://www-
       personal.umich.edu/~wagnerr/
       ConfigFile.html
  ConfigFile config( "envelope_langmuir.inp
       " );
  config.readInto( dirname, "dirname",
       string("Untitled"));
  config.readInto( casename, "casename",
       string("Untitled"));
  i = config.read<int>( "i", 700 );
  config.readInto( C, "C", 1.0 );
  config.readInto( tstop, "tstop", 20.0);
  config.readInto( nint, "nint", 0.001);
  nn = config.read<int>( "nn", 1);
  config.readInto( L, "L", 30.0);
 M = config.read<int>( "M", 1);

  m = i+1; // number of points in the
       spatial grid
  ap = new double[m]; // a at time level l
       +1
  a = new double[m];    // a at time level l
  am = new double[m];   // a at time level l
       -
  bp = new double[m]; // b at time level l
       +1
  b = new double[m];    // b at time level l
  bm = new double[m];   // b at time level l
       -

  // Initializing the arrays
  for (int j = 0; j <= i; ++j) {
    ap[j] = 0;
    a[j] = 0;
    am[j] = 0;
    bp[j] = 0;
    b[j] = 0;
    bm[j] = 0;
  } // end for

  } // end scan

void EnvelopeLangmuir:: solveProblem ()
{
timeLoop();
delete [] ap;
delete [] a;
delete [] am;
delete [] bp;
delete [] b;
delete [] bm;
}
```

```
void EnvelopeLangmuir:: timeLoop ()
{

  h = L/(m-1); // Length of the grid
      intervals
  cout << "h␣=␣" << h << endl;

  dt = (2.0/3.0)*(C*h*h); // time step,
      assume alpha is equal to unity
  double t = 0;
  cout << "dt␣=␣" << dt << endl;

  // Check that the stability condition is
      withheld
  if ( C > (1.0/4.0) ) {
    cout << "NB!␣C␣is␣to␣large,␣the␣
        solution␣will␣be␣unstable.␣\n";
    cout << flush;
  } // end if

  setIC();        //Set initial conditions
  int step_no = 0; // current timestep
      number

  // Open output directory and files
  int directory = mkdir(dirname.c_str(),
      S_IRWXU | S_IRWXG | S_IROTH |
      S_IXOTH);
  string fName_a = dirname + casename + "_a
      .data";
  string fName_b = dirname + casename + "_b
      .data";
  string fName_ax = dirname + casename + "
      _a.res";
  string fName_bx = dirname + casename + "
      _b.res";
  fouta.open(fName_a.c_str());
  foutb.open(fName_b.c_str());
  fileax.open(fName_ax.c_str());
  filebx.open(fName_bx.c_str());

  // Write the initial displacement
  dumpSolution(t);
  xConst(t);

  // Compute the first timestep, update the
       arrays and write them to file
  t += dt;
  step_no = 1;
  solveAtTheFirstTimeStep ();
  updateDataStructures ();

  while (t < tstop) {

    t += dt; // increase time by one time
        step
    step_no++; // increase step number by 1
    solveAtThisTimeStep ();
    periodicBC ();
    updateDataStructures ();

    // write to file every Mth time step:
    if (step_no % M == 0) {
      dumpSolution(t);
      xConst(t);
    } // end if

    // write a message every 100th step:
    if (step_no % 100 == 0) {
       cout << "time␣step␣number␣" <<
            step_no << "␣is␣computed,␣time␣
            =␣" << t << "\n";
       cout << flush;              // flush
            forces immediate output
    } // end if

  } // end while

  // close all files
  fouta.close();
  foutb.close();
  fileax.close();
  filebx.close();

} // end timeloop

void EnvelopeLangmuir:: setIC ()
{

  int j;  // loop counter over grid points
  double initial_const = (nn*2*M_PI*h)/L;

  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubations a(x,0) and b(x,0)
    a[j] = nint*sin(initial_const*j);
    b[j] = 0*sin(initial_const*j)*exp((-(
        M_PI/2)*(h*j-L/2)*(h*j-L/2))/15);
  } // end for

  /*
  //inital condition from Griffiths page 67
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubations a(x,0) and b(x,0)
    a[j] = exp((-M_PI/2)*(((h*j-L/2)*(h*j-L
        /2))/1.0));
    b[j] = 0;
  } // end for
  */

} // end setIC

void EnvelopeLangmuir::
    solveAtTheFirstTimeStep() // Forward
    Euler scheme
{
   int j; // Loop counter over grid points

  // update inner points according to
       finite diffference scheme:
  for (j = 1; j <= i-1; j++) {
    ap[j] = -C*(b[j+1] - 2*b[j] + b[j-1]) +
          a[j];
    bp[j] = C*(a[j+1] - 2*a[j] + a[j-1]) +
         b[j];
  } // end for

  // Set the periodic boundary conditions
  ap[0] = -C*(b[1] - 2*b[0] + b[i-1]) + a
       [0];
  ap[i] = ap[0];
  bp[0] = C*(a[1] - 2*a[0] + a[i-1]) + b
       [0];
  bp[i] = bp[0];

} // end solveAtTheFirstTimeStep

void EnvelopeLangmuir:: solveAtThisTimeStep
     ()
{
  int j; // Loop counter over grid points

  // update inner points according to
       finite diffference scheme:
  for (j = 1; j <= i-1; j++) {
    ap[j] = -2*C*(b[j+1] - 2*b[j] + b[j-1])
         + am[j];
    bp[j] = 2*C*(a[j+1] - 2*a[j] + a[j-1])
         + bm[j];
  } // end for


} // end solveAtThisTimeStep

void EnvelopeLangmuir:: periodicBC(){


  ap[0] = -2*C*(b[1] - 2*b[0] + b[i-1]) +
      am[0];
  ap[i] = ap[0];

  bp[0] = 2*C*(a[1] - 2*a[0] + a[i-1]) + bm
       [0];
  bp[i] = bp[0];
} // end periodicBC

void EnvelopeLangmuir::
    updateDataStructures () {
  for (int j = 0; j <= i; ++j) {
  am[j] = a[j];
  a[j] = ap[j];
  bm[j] = b[j];
  b[j] = bp[j];
  } // end for
} // end updateDataStructures

void EnvelopeLangmuir:: dumpSolution (
    double t)
{
  fouta << t << "␣" << h << "␣";
  for (int j = 0; j <= i; j++) {
    fouta << a[j] << "␣";
  } // end for
  fouta << endl;

  foutb << t << "␣" << h << "␣";
  for (int j = 0; j <= i; j++) {
    foutb << b[j] << "␣";
  } // end for
  foutb << endl;

} // end dumpSoultion

void EnvelopeLangmuir:: xConst (double t) {

// I want to find n(x,t) as a function of a
     specific x
  int p;

    p = (3*m)/(4*nn); // Choose the
        amplitude

  if (t == 0) cout << "The␣choosen␣x␣=␣" <<
        p*h << "\n";

  fileax << t << "␣" << a[p] << "\n";
  filebx << t << "␣" << b[p] << "\n";

} // end xConst
```

main.cpp

```
#include "EnvelopeLangmuir.h"
using namespace std;

int main (int argc, const char* argv[])
{
  EnvelopeLangmuir problem;
  problem.scan(); problem.solveProblem();
  return 0;  // success
}
```

makefile

```
#makefile

app : main.o EnvelopeLangmuir.o ConfigFile.
    o
        g++ -o app main.o EnvelopeLangmuir.
            o ConfigFile.o

main.o: main.cpp EnvelopeLangmuir.h
        g++ -c main.cpp

EnvelopeLangmuir.o: EnvelopeLangmuir.cpp
     EnvelopeLangmuir.h
        g++ -c EnvelopeLangmuir.cpp

ConfigFile.o: ConfigFile.cpp ConfigFile.h
        g++ -c ConfigFile.cpp -pg

clean:
        rm app main.o EnvelopeLangmuir.o

# end of makefile
```

envelope_langmuir.inp

```
dirname = /mn/korona/rp-s1/margitu/Program/
     Linear_Slowly/Results/
casename = sin_nn5
i = 1000
C = 0.2
tstop = 1
nint = 1
nn = 5
L = 60.0
M = 800
```

B.4 Langmuir envelopes in multi electron component plasmas

EnvelopeMultiLangmuir.h

```
#include <stdlib.h>
#include <cmath>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <sys/stat.h>
#include <sys/types.h>

using namespace std;

class EnvelopeMultiLangmuir
{
  string casename;
  int i;         // number of intervals on
       [0,1]
  int m;         // number of points in the
       spatial grid

  double C;      // Courant number
  double tstop;  // The length of the time
       interval
  double nint;   // amplitude of the initial
        disturbance
  int nn;        // number of nodes
  double L;      // Spatial length
  int M;         // How often should the
       result be written to file?
  int N;         // Number of electron
       density components

  double h;         // Length of the grid
       intervals
  double dt;        // Timestep

  double** ap; // a at time level l+1
  double** a;   // a at time level l
  double** am;  // a at time level l-1
  double** bp; // b at time level l+1
  double** b;   // b at time level l
  double** bm;  // b at time level l-1
  double* sum_a; // The sum of all a at
       time level l
  double* sum_b; // The sum of all b at
       time level l
  double* sum_ap; // The sum of all b at
       time level l+1
  double* sum_bp; // The sum of all b at
       time level l+1

  // Coefficients
  double* alpha;
  double* beta;

  void setIC ();                      //
       set initial conditions
  void solveAtTheFirstTimeStep(int r);
       // Forward Euler scheme
  void timeLoop ();
  void solveAtThisTimeStep(int r);
       // Leap Frog scheme
  void periodicBC(int r);
  void dumpSolution (int r, double t);
              // write the soultion to file
  void updateDataStructures (int r);
  void xConst(int r, double t); // write
       the solutiona at a spesific x point
       to file

  // Open output files
  vector<ofstream*>fileArrayA;  // To store
       the timevolution at all points
  vector<ofstream*>fileArrayB;  // To store
       the timevolution at all points
  vector<ofstream*>fileArrayAx; // To store
       the time evolution at one spesific
       x-point
  vector<ofstream*>fileArrayBx; // To store
       the time evolution at one spesific
       x-point

 public:
  void scan ();
  void solveProblem ();

}; // end class EnvelopeMultiLangmuir
```

EnvelopeMultiLangmuir.cpp

```
#include "EnvelopeMultiLangmuir.h"
#include "ConfigFile.h"

void EnvelopeMultiLangmuir:: scan()
{
  //read input from file using
       configuration manager: http://www-
       personal.umich.edu/~wagnerr/
       ConfigFile.html
  ConfigFile config( "envelope_langmuir.inp
       " );
  config.readInto( casename, "casename",
       string("Untitled"));
  i = config.read<int>( "i", 700 );
  config.readInto( C, "C", 1.0 );
  config.readInto( tstop, "tstop", 20.0);
  config.readInto( nint, "nint", 0.001);
  nn = config.read<int>( "nn", 1);
  config.readInto( L, "L", 30.0);
  M = config.read<int>( "M", 1);
  N = config.read<int>( "N", 10);
  nint = 1.0/(N); // ensures that the
       overall amplitude of the wave is
       equal to 1

m = i+1; // number of points in the
       spatial grid

// Allocation of matrices and arrays

ap = new double*[N];
for(register int k=0; k<N; k++)
  ap[k] = new double[m];

bp = new double*[N];
for(register int k=0; k<N; k++)
  bp[k] = new double[m];

a = new double*[N];
for(register int k=0; k<N; k++)
  a[k] = new double[m];

b = new double*[N];
for(register int k=0; k<N; k++)
  b[k] = new double[m];

am = new double*[N];
for(register int k=0; k<N; k++)
  am[k] = new double[m];

bm = new double*[N];
for(register int k=0; k<N; k++)
  bm[k] = new double[m];

// The sum over all the N electron
      components
sum_a = new double[m];
sum_ap = new double[m];
sum_b = new double[m];
sum_bp = new double[m];

// Coefficients
alpha = new double[N];
beta = new double[N];

// The total inital electron
double sum_n_r = 0;
for (int r = 0; r < N; r++) {
  sum_n_r = sum_n_r + (r+1)*(exp
       ((-9.0*(((r+1) -1))*((r+1) -1))
       /(2.0*N*N)) - exp((-9.0*(r+1)*(r
       +1))/(2.0*N*N)));
} // end for

// Initializing the density matrices and
      the coefficient arrays
double sum_r_beta = 0; // help variable
      for computing alpha
for (int r = 0; r < N; r++) {
  for (int j = 0; j <= i; ++j) {
    ap[r][j] = 0;
    bp[r][j] = 0;
    a[r][j] = 0;
    b[r][j] = 0;
    am[r][j] = 0;
    bm[r][j] = 0;
    sum_a[j] = 0;
    sum_ap[j] = 0;
    sum_b[j] = 0;
    sum_bp[j] = 0;
  } // end for
  beta[r] = ((r+1)*(exp((-9.0*(((r+1)-1))
       *((r+1) -1))/(2.0*N*N)) - exp
       ((-9.0*(r+1)*(r+1))/(2.0*N*N))))
       /(2.0*sum_n_r);
  sum_r_beta = sum_r_beta + (r+1)*(r+1)*
       beta[r];
} // end for

for (int r = 0; r < N; r++) {
  alpha[r] = (3.0*(r+1)*(r+1))/(2.0*
       sum_r_beta); // alpha[N-1] is
       always the largest element in this
        array
  } // end for

 // Initializing the sum over all the N
      components
  for (int j = 0; j <= i; ++j) {
    sum_a[j] = 0;
    sum_ap[j] = 0;
    sum_b[j] = 0;
    sum_bp[j] = 0;
  }// end for

} // end scan

void EnvelopeMultiLangmuir::solveProblem
    ()
{

timeLoop();

//Free memory

 delete [] alpha;
 delete [] beta;

 for (int r = 0; r < N; r++) {

    delete fileArrayA [r];
    delete fileArrayB [r];
    delete fileArrayAx [r];
    delete fileArrayBx [r];

    delete [] ap[r];
    ap[r] = 0;
    delete [] a[r];
    a[r] = 0;
    delete [] am[r];
    am[r] = 0;

    delete [] bp[r];
    bp[r] = 0;
    delete [] b[r];
    b[r] = 0;
    delete [] bm[r];
    bm[r] = 0;

  } // end for

  delete [] ap;
  delete [] a;
  delete [] am;
  delete [] sum_a;
  delete [] sum_ap;

  delete [] bp;
  delete [] b;
  delete [] bm;
  delete [] sum_b;
  delete [] sum_bp;

} // end solveProblem

void EnvelopeMultiLangmuir:: timeLoop ()
{

  h = L/(m-1); // Length of the grid
      intervals
  cout << "h␣=␣" << h << endl;

  dt = (C*h*h); // time step
  double t = 0;

  // Check that the stability condition is
      withheld
  if ( C > (1.0/(4.0*alpha[N-1] + N*h*h*
      beta[N-1] - 0.5*h*h)) ) {
    cout << "NB!␣C␣is␣to␣large,␣the␣
        solution␣will␣be␣unstable.␣\n";
    cout << flush;
    abort();
  } // end if

  setIC();        //Set the initial
      conditions
```

```
int step_no = 0; // current timestep
    number

// Open output directory and files
string dirname = "/mn/korona/rp-s1/
    margitu/Program/Linear_Multi_Slowly/
    Results/" + casename;
int directory = mkdir(dirname.c_str(),
    S_IRWXU | S_IRWXG | S_IROTH |
    S_IXOTH);
fileArrayA.resize(N);
fileArrayAx.resize(N);
fileArrayB.resize(N);
fileArrayBx.resize(N);
std::string casenamesA[N];
std::string casenamesAx[N];
std::string casenamesB[N];
std::string casenamesBx[N];
std::string s[N]; // converting int r to
    string s
for (int r = 0; r < N; r++) {
  std::stringstream out;
  out << (r+1);
  s[r] = out.str();
  casenamesA[r] = dirname + "/" +
      casename + "_" + s[r] + "_a.data";
  casenamesAx[r] = dirname + "/" +
      casename + "_" + s[r] + "_a.res";
  casenamesB[r] = dirname + "/" +
      casename + "_" + s[r] + "_b.data";
  casenamesBx[r] = dirname + "/" +
      casename + "_" + s[r] + "_b.res";
  fileArrayA[r] = new ofstream(casenamesA
      [r].c_str());
  fileArrayAx[r] = new ofstream(
      casenamesAx[r].c_str());
  fileArrayB[r] = new ofstream(casenamesB
      [r].c_str());
  fileArrayBx[r] = new ofstream(
      casenamesBx[r].c_str());
  // Dump the initial displacements
  dumpSolution(r, t);
  xConst(r, t);
} // end for

// Compute the first timestep, update the
      arrays and write them to file
t += dt;
step_no = 1;
for (int r = 0; r < N; r++){
solveAtTheFirstTimeStep (r);
updateDataStructures (r);
if (step_no % M == 0) {
  dumpSolution(r,t);
  } // end if
if (step_no % 100 == 0) {
  xConst(r, t);
  } // end if
} // end for

// Update the sum over all the N electron
      components for the next timestep
for (int j = 0; j <= i; ++j) {
  sum_a[j] = sum_ap[j];
  sum_ap[j] = 0;
  sum_b[j] = sum_bp[j];
  sum_bp[j] = 0;
} // end for

while (t < tstop) {

  t += dt; // increase time by one time
      step
  step_no++; // increase step number by 1

  // write a message every 100th step:
  if (step_no % 100 == 0) {
    cout << "time␣step␣number␣" <<
        step_no << "␣is␣computed,␣time␣
        =␣" << t << "\n";
    cout << flush;           // flush
        forces immediate output
  } // end if
```

```
      for (int r = 0; r < N; r++) {

        solveAtThisTimeStep (r);
        periodicBC (r);
        updateDataStructures (r);

        // write to file every Mth time step
            :
        if (step_no % M == 0) {
          dumpSolution(r,t);
        } // end if
         if (step_no % 100 == 0) {
          xConst(r, t);
        } // end if

      } // end for

      // Update the sum over all the N
           electron components for the next
           timestep
      for (int j = 0; j <= i; ++j) {
        sum_a[j] = sum_ap[j];
        sum_ap[j] = 0;
        sum_b[j] = sum_bp[j];
        sum_bp[j] = 0;
      } // end for

  } // end while

} // end timeloop

void EnvelopeMultiLangmuir:: setIC ()
{

  int j;  // loop counter over grid points
  double initial_const = (nn*2*M_PI*h)/L;

  /*
  //Pulse
   for (int r = 0; r < N; r++) {
     for (j = 0; j <= i; j++) {
                         // set the initial
          pertubations a(x,0) and b(x,0)
       a[r][j] = nint*sin(initial_const*j)*
            exp((-(h*j - L/2)*(h*j - L/2))
            /15);
       b[r][j] = nint*sin(initial_const*j)*
            exp((-(h*j - L/2)*(h*j - L/2))
            /15);
       sum_a[j] = sum_a[j] + a[r][j];
       sum_b[j] = sum_b[j] + b[r][j];
     } // end for
  } // end for
  */

  // Standing waves
  initial_const = (nn*2*M_PI*h)/L;
  for (int r = 0; r < N; r++) {
    for (j = 0; j <= i; j++) {
                         // set the initial
         pertubation n(x,0)
       a[r][j] = nint*sin(initial_const*j);
       sum_a[j] = sum_a[j] + a[r][j];
       b[r][j] = 0;
       sum_b[j] = sum_b[j] + b[r][j];
    } // end for
  } // end for

  /*
  // Only one component larger than one
  // Standing waves
  initial_const = (nn*2*M_PI*h)/L;
  for (j = 0; j <= i; j++) {
                       // set the initial
       pertubation n(x,0)
    a[0][j] = 0.8*nint*sin(initial_const*j)
         ;
    sum_a[j] = sum_a[j] + a[0][j];
    b[0][j] = 0;
    sum_b[j] = sum_b[j] + b[0][j];
  } // end for
```

```
  for (int r = 1; r < N; r++) {
    for (j = 0; j <= i; j++) {
                           // set the initial
         pertubation n(x,0)
      a[r][j] = (0.2/(N-1))*nint*sin(
           initial_const*j);
      sum_a[j] = sum_a[j] + a[r][j];
      b[r][j] = 0;
      sum_b[j] = sum_b[j] + b[r][j];
    } // end for
  } // end for
  */

} // end setIC

void EnvelopeMultiLangmuir::
     solveAtTheFirstTimeStep(int r) //
     Forward Euler scheme
{
   int j; // Loop counter over grid points

  // update inner points according to the
       finite diffference scheme:
  for (j = 1; j <= i-1; j++) {
    ap[r][j] = -C*alpha[r]*(b[r][j+1] - 2*b
         [r][j] + b[r][j-1]) - dt*0.5*b[r][
         j] + beta[r]*dt*sum_b[j] + a[r][j
         ];
    sum_ap[j] = sum_ap[j] + ap[r][j];
    bp[r][j] = C*alpha[r]*(a[r][j+1] - 2*a[
         r][j] + a[r][j-1]) + dt*0.5*a[r][j
         ] - beta[r]*dt*sum_a[j] + b[r][j];
    sum_bp[j] = sum_bp[j] + bp[r][j];
  } // end for

  // Set the periodic boundary conditions
  ap[r][0] = -C*alpha[r]*(b[r][1] - 2*b[r
       ][0] + b[r][i-1]) - dt*0.5*b[r][0] +
        beta[r]*dt*sum_b[0] + a[r][0];
  ap[r][i] = ap[r][0];
  sum_ap[0] = sum_ap[0] + ap[r][0];

  bp[r][0] = C*alpha[r]*(a[r][1] - 2*a[r
       ][0] + a[r][i-1]) + dt*0.5*a[r][0] -
        beta[r]*dt*sum_a[0] + b[r][0];
  bp[r][i] = bp[r][0];
  sum_bp[0] = sum_bp[0] + bp[r][0];

} // end solveAtTheFirstTimeStep

void EnvelopeMultiLangmuir::
     solveAtThisTimeStep (int r)
{
  int j; // Loop counter over grid points

  // update inner points according to
       finite diffference scheme:
  for (j = 1; j <= i-1; j++) {
    ap[r][j] = -2*alpha[r]*C*(b[r][j+1] -
         2*b[r][j] + b[r][j-1]) - dt*b[r][j
         ] + 2*beta[r]*dt*sum_b[j] + am[r][
         j];
    sum_ap[j] = sum_ap[j] + ap[r][j];
    bp[r][j] = 2*alpha[r]*C*(a[r][j+1] - 2*
         a[r][j] + a[r][j-1]) + dt*a[r][j]
         - 2*beta[r]*dt*sum_a[j] + bm[r][j
         ];
    sum_bp[j] = sum_bp[j] + bp[r][j];
  } // end for


} // end solveAtThisTimeStep

void EnvelopeMultiLangmuir:: periodicBC(int
      r){


  ap[r][0] = -2*alpha[r]*C*(b[r][1] - 2*b[r
       ][0] + b[r][i-1]) - dt*b[r][0] + 2*
       beta[r]*dt*sum_b[0] + am[r][0];
  ap[r][i] = ap[r][0];
  sum_ap[0] = sum_ap[0] + ap[r][0];
  bp[r][0] = 2*alpha[r]*C*(a[r][1] - 2*a[r
       ][0] + a[r][i-1]) + dt*a[r][0] - 2*
       beta[r]*dt*sum_a[0] + bm[r][0];
  bp[r][i] = bp[r][0];
  sum_bp[0] = sum_bp[0] + bp[r][0];

} // end periodicBC

void EnvelopeMultiLangmuir::
    updateDataStructures (int r) {
  for (int j = 0; j <= i; ++j) {
  am[r][j] = a[r][j];
  a[r][j] = ap[r][j];
  bm[r][j] = b[r][j];
  b[r][j] = bp[r][j];
  } // end for
} // end updateDataStructures

void EnvelopeMultiLangmuir:: dumpSolution (
    int r, double t)
{

  *fileArrayA[r] << r << "␣" << t << "␣" <<
       h << "␣";
  for (int j = 0; j <= i; j++) {
    *fileArrayA[r] << a[r][j] << "␣";
  } // end for
  *fileArrayA[r] << endl;

  *fileArrayB[r] << r << "␣" << t << "␣" <<
       h << "␣";
  for (int j = 0; j <= i; j++) {
    *fileArrayB[r] << b[r][j] << "␣";
  } // end for
  *fileArrayB[r] << endl;

} // end dumpSoultion

void EnvelopeMultiLangmuir:: xConst (int r,
     double t) {

// I want to find n(x,t) as a function of a
     specific x
  int p;
  p = (3*m)/(4*nn);

  if (t == 0) cout << "The␣choosen␣x␣=␣" <<
       p*h << endl;

     *fileArrayAx[r] << t << "␣" << a[r][p]
          << endl;
     *fileArrayBx[r] << t << "␣" << b[r][p]
          << endl;

} // end xConst
```

main.cpp

```
#include "EnvelopeMultiLangmuir.h"
using namespace std;

int main (int argc, const char* argv[])
{
  EnvelopeMultiLangmuir problem;
  problem.scan(); problem.solveProblem();
  return 0;  // success
}
```

makefile

```
#makefile

app : main.o EnvelopeMultiLangmuir.o
     ConfigFile.o
        g++ -o app main.o
              EnvelopeMultiLangmuir.o
              ConfigFile.o

main.o: main.cpp EnvelopeMultiLangmuir.h
        g++ -c main.cpp

EnvelopeMultiLangmuir.o:
     EnvelopeMultiLangmuir.cpp
     EnvelopeMultiLangmuir.h
        g++ -c EnvelopeMultiLangmuir.cpp

ConfigFile.o: ConfigFile.cpp ConfigFile.h
        g++ -c ConfigFile.cpp -pg

clean:
        rm app main.o EnvelopeMultiLangmuir
             .o

# end of makefile
```

envelope_langmuir.inp

```
casename = multi_25
i = 1000
C = 0.02
tstop = 20
nint = 1
nn = 5
L = 60
M = 7000
N = 25
```

B.5 The nonlinear Schrödinger equation

NLS.h

```
#include <stdlib.h>
#include <cmath>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <sstream>
#include <sys/stat.h>
#include <sys/types.h>

using namespace std;

class NLS
{
  string dirname;
  string casename;
  int i;          // number of intervals on
       [0,L]
  int m;          // number of points in the
       spatial grid

  double C;       // Courant number
  double tstop;   // The length of the time
       interval
  double nint;    // amplitude of the initial
        disturbance
  int nn;         // number of nodes
  double L;       // Spatial length
  int N;          // How often should the
       result be written to file?
  double non_lin;  // Dimensionless
       parameter multiplied by the
       nonlinear term

  double h;          // Length of the grid
       intervals
  double dt;         // Timestep
  double integral; // Holds the current
       energy estimate

  double* ap; // a at time level l+1
  double* a;    // a at time level l
  double* am;  // a at time level l-1
  double* bp; // b at time level l+1
  double* b;    // b at time level l
  double* bm;  // b at time level l-1

  void setIC ();                        //
       set initial conditions
  void solveAtTheFirstTimeStep();       //
       Forward Euler scheme
  void timeLoop ();
  void solveAtThisTimeStep();           //
       Leap Frog scheme
  void periodicBC();
  void dumpSolution (double t);         //
       write the soultion to file
  void updateDataStructures ();
  void dumpEnergy (double t);             //
        Write to file the energy in the
       field
  void energyConservation(); // Numerical
       integration procedure
  void xConst (double t); //spesific x-
       point

  std::ofstream fouta;
  std::ofstream foutb;
  std::ofstream file;
  std::ofstream fileax;
  std::ofstream filebx;

 public:
  void scan ();
  void solveProblem ();

}; // end class NLS
```

NLS.cpp

```
#include "NLS.h"
#include "ConfigFile.h"

void NLS:: scan()
{
  //read input from file using
       configuration manager: http://www-
       personal.umich.edu/~wagnerr/
       ConfigFile.html
  ConfigFile config( "nls.inp" );
  config.readInto( dirname, "dirname",
       string("Untitled"));
  config.readInto( casename, "casename",
       string("Untitled"));
  i = config.read<int>( "i", 700 );
  config.readInto( C, "C", 1.0 );
  config.readInto( tstop, "tstop", 20.0);
  config.readInto( nint, "nint", 0.001);
  nn = config.read<int>( "nn", 1);
  config.readInto( L, "L", 30.0);
  N = config.read<int>( "N", 1);
  config.readInto( non_lin, "non_lin", 0.0)
       ;

  m = i+1; // number of points in the
       spatial grid

  ap = new double[m]; // a at time level l
       +1
  a = new double[m];    // a at time level l
  am = new double[m];   // a at time level l
       -
  bp = new double[m]; // b at time level l
       +1
  b = new double[m];    // b at time level l
  bm = new double[m];   // b at time level l
       -

  // Initializing the arrays
  for (int j = 0; j <= i; ++j) {
    ap[j] = 0;
    a[j] = 0;
    am[j] = 0;
    bp[j] = 0;
    b[j] = 0;
    bm[j] = 0;
  } // end for

  } // end scan

void NLS:: solveProblem ()
{
timeLoop();
delete [] ap;
delete [] a;
delete [] am;
delete [] bp;
delete [] b;
delete [] bm;
}

void NLS:: timeLoop ()
{

  h = L/(m-1); // Length of the grid
       intervals
  cout << "h = " << h << endl;

  dt = (C*h*h); // time step, assume alpha
       is equal to unity
  double t = 0;

  // Check that the stability condition is
       withheld
  if ( C > (4/(24 + h*h)) ) {
    cout << "NB! C is to large, the
         solution will be unstable. \n";
    cout << flush;
  } // end if

  setIC();          //Set initial conditions
  energyConservation(); // Find the inital
       energy
  int step_no = 0; // current timestep
       number

  // Open output directory and files
  int directory = mkdir(dirname.c_str(),
       S_IRWXU | S_IRWXG | S_IROTH |
       S_IXOTH);
  string fName_a = dirname + casename + "_a
       .data";
  string fName_b = dirname + casename + "_b
       .data";
  string fileName = dirname + casename + ".
       res";
  string fName_ax = dirname + casename + "
       _a.res";
  string fName_bx = dirname + casename + "
       _b.res";
  fouta.open(fName_a.c_str());
  foutb.open(fName_b.c_str());
  file.open(fileName.c_str());
  fileax.open(fName_ax.c_str());
  filebx.open(fName_bx.c_str());

  // Write the initial displacement and
       energy to file
  dumpSolution(t);
  xConst(t);
  dumpEnergy(t);


  // Compute the first timestep, update the
       arrays and write them to file
  t += dt;
  step_no = 1;
  solveAtTheFirstTimeStep ();
  updateDataStructures ();

  while (t < tstop) {

    t += dt; // increase time by one time
         step
    step_no++; // increase step number by 1
    solveAtThisTimeStep ();
    periodicBC ();
    updateDataStructures ();

    // write to file every Nth time step:
    if (step_no % N == 0) {
      dumpSolution(t);
    } // end if

    if (step_no % 200 == 0) {
      xConst(t);
    } // end if

    // Find the energy and write it to file
         every 2Nth step
    if (step_no % 2*N == 0) {
      energyConservation();
      dumpEnergy(t);
    } // end if

    // write a message every 100th step:
    if (step_no % 200 == 0) {
       cout << "time␣step␣number␣" <<
            step_no << "␣is␣computed,␣time␣
            =␣" << t << "\n";
       cout << flush;            // flush
            forces immediate output
    } // end if

  } // end while

  // close all files
  fouta.close();
  foutb.close();
  file.close();
  fileax.close();
  filebx.close();

} // end timeloop

void NLS:: setIC ()
{

  int j;  // loop counter over grid points
  double initial_const = (nn*2*M_PI*h)/L;

  /*
  // Standing waves/travelling
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubations a(x,0) and b(x,0)
    a[j] = nint*cos(initial_const*j);
    b[j] = nint*sin(initial_const*j);
  } // end for
  */

  /*
  // Simple wave packet
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubations a(x,0) and b(x,0)
    a[j] = nint*sin(initial_const*j)*exp
         ((-(M_PI/2)*(h*j-L/2)*(h*j-L/2))
         /30);
    b[j] = nint*sin(initial_const*j)*exp
         ((-(M_PI/2)*(h*j-L/2)*(h*j-L/2))
         /30);
  } // end for
  */
  /*
  //inital condition from Griffiths page 67
       """"
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubations a(x,0) and b(x,0)
    a[j] = exp((-M_PI/2)*(((h*j-L/2)*(h*j-L
         /2))/24.0));
    b[j] = 0;
  } // end for
  */

  //Amplitude modulation
  for (j = 0; j <= i; j++) {
                        // set the initial
       pertubations a(x,0) and b(x,0)
    a[j] = (1 + 0.1*cos((M_PI/30)*j*h))*
         nint*cos(initial_const*j);
    b[j] = (1 + 0.1*cos((M_PI/30)*j*h))*
         nint*sin(initial_const*j);
  } // end for

} // end setIC

void NLS:: solveAtTheFirstTimeStep() //
     Forward Euler scheme
{
   int j; // Loop counter over grid points
   double product_b = (non_lin*dt)/4.0;
   double product_a = (3.0*C)/2.0;

  // update inner points according to
       finite diffference scheme:
  for (j = 1; j <= i-1; j++) {
    ap[j] = -product_b*(a[j]*a[j] + b[j]*b[
         j])*b[j] - product_a*(b[j+1] - 2*b
         [j] + b[j-1]) + a[j];
    bp[j] = product_b*(a[j]*a[j] + b[j]*b[j
         ])*a[j] + product_a*(a[j+1] - 2*a[
         j] + a[j-1]) + b[j];
  } // end for

  // Set the periodic boundary conditions
  ap[0] = -product_b*(a[0]*a[0] + b[0]*b
       [0])*b[0] - product_a*(b[1] - 2*b[0]
        + b[i-1]) + a[0];
  ap[i] = ap[0];

  bp[0] = product_b*(a[0]*a[0] + b[0]*b[0])
       *a[0] + product_a*(a[1] - 2*a[0] + a
       [i-1]) + b[0];
  bp[i] = bp[0];

} // end solveAtTheFirstTimeStep

void NLS:: solveAtThisTimeStep ()
{
  int j; // Loop counter over grid points
  double product_b = (dt*non_lin)/2.0;
  double product_a = (3.0*C);

  // update inner points according to
       finite diffference scheme:
  for (j = 1; j <= i-1; j++) {
    ap[j] = -product_b*(a[j]*a[j] + b[j]*b[
         j])*b[j] - product_a*(b[j+1] - 2*b
         [j] + b[j-1]) + am[j];
    bp[j] = product_b*(a[j]*a[j] + b[j]*b[j
         ])*a[j] + product_a*(a[j+1] - 2*a[
         j] + a[j-1]) + bm[j];
  } // end for


} // end solveAtThisTimeStep

void NLS:: periodicBC(){


  ap[0] = - non_lin*0.5*dt*(a[0]*a[0] + b
       [0]*b[0])*b[0] - 3*C*(b[1] - 2*b[0]
       + b[i-1]) + am[0];
  ap[i] = ap[0];
  bp[0] = non_lin*0.5*dt*(a[0]*a[0] + b[0]*
       b[0])*a[0] + 3*C*(a[1] - 2*a[0] + a[
       i-1]) + bm[0];
  bp[i] = bp[0];

} // end periodicBC

void NLS:: updateDataStructures () {
  for (int j = 0; j <= i; ++j) {
  am[j] = a[j];
  a[j] = ap[j];
  bm[j] = b[j];
  b[j] = bp[j];
  } // end for
} // end updateDataStructures

void NLS:: dumpSolution (double t)
{
  fouta << t << "␣" << h << "␣";
  for (int j = 0; j <= i; j++) {
    fouta << a[j] << "␣";
  } // end for
  fouta << endl;

  foutb << t << "␣" << h << "␣";
  for (int j = 0; j <= i; j++) {
    foutb << b[j] << "␣";
  } // end for
  foutb << endl;

} // end dumpSoultion

void NLS:: energyConservation () {

  integral = 0;
  int j; // loop counter

  integral = (h/2)*(a[0]*a[0] + b[0]*b[0] +
        a[i]*a[i] + b[i]*b[i]);

  for (j = 1; j<= i-1; j++) {
    integral = integral + h*(a[j]*a[j] + b[
        j]*b[j]);
  } // end for

} // end energyConservation

void NLS:: dumpEnergy (double t) {

  file << t << "␣" << integral << "\n";

} // end dumpEnergy

void NLS:: xConst (double t) {

  // I want to find n(x,t) as a function of
        a specific x
  int p;

  p = (m)/(4*nn); // Choose the amplitude

  if (t == 0) cout << "The␣choosen␣x␣=␣" <<
        p*h << "\n";

  fileax << t << "␣" << a[p] << "\n";
  filebx << t << "␣" << b[p] << "\n";

} // end xConst
```

main.cpp

```
# include <iostream>
# include <cmath>
#include "NLS.h"
using namespace std;

int main (int argc, const char* argv[])
{
  NLS problem;
  problem.scan(); problem.solveProblem();
  return 0;  // success
}
```

makefile

```
#makefile

app : main.o NLS.o ConfigFile.o
        g++ -o app main.o NLS.o ConfigFile.
              o

main.o: main.cpp NLS.h
        g++ -c main.cpp

NLS.o: NLS.cpp NLS.h
        g++ -c NLS.cpp

ConfigFile.o: ConfigFile.cpp ConfigFile.h
        g++ -c ConfigFile.cpp -pg

clean:
        rm app main.o NLS.o ConfigFile.o

# end of makefile
```

nls.inp

```
dirname = /mn/korona/rp-s1/margitu//Program
      /Slowly_Electric/Results/AM_double_05/
casename = AM_double_05
i = 1000
C = 0.14
tstop = 30
nint = 0.5
nn = 3
L = 120.0
N =  1000
non_lin = 1.0
```

Bibliography

[1] T. J. M. Boyd and J. J. Sanderson, The Physics of Plasmas. Cambridge University Press, 2003.

[2] D. G. Swanson, Plasma Waves. Academic Press, 1989.

[3] L. D. Landau, "On the vibrations of the electronic plasma," Journal of Physics USSR, vol. 10, p. 25, 1946.

[4] B. K. Shivamoggi, Introduction to Nonlinear Fluid-Plasma Waves. No. 8 in Mechanics of Fluids and Transport Processes, Kluwer Academic Publishers, 1988.

[5] A. A. Vlasov, "On the Kinetic Theory of an Assembly of Particles with Collective Interaction," Journal of Physics USSR, vol. 9, pp. 25–40, 1945.

[6] D. C. Depackh, "The Water-bag Model of a Sheet Electron Beamy," International Journal of Electronics, vol. 13, pp. 417–424, 1962.

[7] M. Feix, F. Hohl, and L. Staton, "Nonlinear effects," in Plasmas (G. Kalman and M. R. Feix, eds.), pp. 3–21, Gordon and Breach, 1969.

[8] P. Bertrand and M. R. Feix, "Non linear electron plasma oscillation: the "water bag model"," Physics Letters A, vol. 28, pp. 68–69, 1968.

[9] P. Bertrand, G. Baumann, and M. R. Feix, "Frequency shift of nonlinear electron plasma oscillation," Physics Letters A, vol. 29, pp. 489–490, 1969.

[10] V. E. Zakharov, "Collapse of Langmuir waves," Sov. Phys. -JETP, vol. 39, pp. 285–256, 1972.

[11] D. K. Campbell, "Nonlinear physics: Fresh breather," Nature, vol. 432, pp. 455–456, 2004.

[12] F. M. S. Lima and P. Arun, "An acurate formula for the period of a simple pendulum oscillating beyond the small angle regime," Am. J. Phys., vol. 74, pp. 892–895, 2006.

[13] P. Morel, E. Gravier, N. Besse, A. Ghizzo, and P. Bertrand, "The water bag model and gyrokinetic applications," Communications in Nonlinear Science and Numerical Simulations, vol. 13, pp. 11–17, 2008.

[14] D. V. Schroeder, Thermal Physics. Addison Wesley Longman, 2000.

[15] H. L. Pécseli, "Waves and oscillations in plasmas." Unpublished lecture notes.

[16] Y. Khotyaintsev, N. Ivchenko, K. Stasiewicz, and M. Berthomier, "Electron Energization by Alfvén Waves: Freja and Sounding Rocket Observations," Physica Scripta, vol. T84, pp. 151–153, 1999.

[17] E. Märk, R. Hatakeyama, and N. Sato, "Electron plasma wave in a plasma with non-Maxwellian distribution produced by a large-amplitude wave," Plasma Physics, vol. 20, pp. 415–425, 1978.

[18] H. L. Pécseli, "Solitons and weakly nonlinear waves in plasmas," IEEE Trans. Plasma Sci., vol. PS-13, pp. 53–86, 1985.

[19] J. H. Malmberg and C. B. Wharton, "Collisionless damping of electrostatic plasma waves," Phys. Rev. Lett., vol. 13, pp. 184–186, 1964.

[20] V. Kowalenko, "Landau damping, Stokes phenomenon and the weakly magnetized Maxwellian plasma," Physics Letters A, vol. 337, pp. 408–418, 2005.

[21] J. M. Leinaas, "Classical mechanics and electrodynamics." Unpublished lecture notes.

[22] F. F. Chen, Introduction to plasma physics and controlled fusion. Plenum Press, 2 ed., 1984.

[23] B. D. Fried and S. D. Conte, The Plasma Dispersion Function. Academic Press, 1961.

[24] T. O'Neil, "Collisionless damping of non linear plasma oscillations," Phys. Fluids, vol. 8, pp. 2255–2262, 1965.

[25] W. E. Drummond, "Landau damping," Phys. Plasmas, vol. 11, pp. 552–560, 2004.

[26] G. Rowlands, "Landau damping and the water-bag model," Physics Letters, vol. 30A, pp. 408–409, 1969.

[27] M.Navet and P.Bertrand, "Multiple "water-bag" model and Landau damping," Physics Letters, vol. 34A, pp. 117–118, 1971.

[28] P. Bertrand, M. Gros, and G. Baumann, "Nonlinear plasma oscillations in terms of multiple-water-bag eigenmodes," Phys. Fluids, vol. 19, p. 1183, 1976.

[29] A. H. Nayfeh, Pertubation Methods. John Wiley & Sons, 1973.

[30] B. Gjevik, "Innføring i fluidmekanikk." Unpublished lecture notes.

[31] J. M. Dawson, "Nonlinear Electron Oscillations in a Cold Plasma," Phys. Rev. Lett., vol. 113, p. 383, 1959.

[32] R. W. C. Davidson and P. P. J. M. Schram, "Nonlinear oscillations in a cold plasma," Nucl. Fusion, vol. 8, pp. 183–195, 1968.

[33] R. W. C. Davidson, Methods in Nonlinear Plasma Theory. No. 37 in Pure and Applied Physics, Academic Press, 1972.

[34] M. Abramowitz and I. A. Stegun, Handbook of Mathematical Functions. Dover Publications, 9 ed., 1970.

[35] T. P. Coffey, "Breaking of Large Amplitude Plasma Oscillations," Phys. Fluids, vol. 14, p. 1402, 1971.

[36] K. Iniewski, Wierless technologies: circuits, systems, and devices. CRC Press, 2007.

[37] J. Awrejcewicz, V. A. Krysko, and T. Moldenkova, "Mathematical model of dissipative parametric vibrations of flexible plates with nonhomogenous boundary conditions," Mathematical Problems in Engineering, pp. 1–16, 2006.

[38] K. Nishikawa, C. S. Liu, P. K. Kaw, and W. L. Kruer, "Parametric Instabilities in plasmas," No. 6 in Advances in Plasma physics, pp. 1–269, Interscience, 1976.

[39] K. B. Dysthe and H. L. Pécseli, "Non-linear Langmuir wave modulation in collisionless plasmas," Plasma Physics, vol. 19, pp. 931–943, 1977.

[40] H. D. Young and R. A. Freedman, University Physics. Pearson Addison Wesley, 11 ed., 2004.

[41] A. V. Gurevich, "Nonlinear Phenomena in the Ionosphere," Radiophysics and Quantum Electronics, vol. 19, pp. 595–597, 1976.

[42] K. G. Budden, Propagation of radio waves. Cambridge University Press, 1985.

[43] B. Gjevik, G. K. Pedersen, and K. Trulsen, Forelesninger i bølgeteori. Unipub, 2007.

[44] A. Tveito and R. Winther, Introduction to Partial differential Equations. Springer, 1998.

[45] H. P. Langtangen, Computational Partial Differential Equations. Springer, 2 ed., 2003.

[46] P. D. Lax, "Survey of the stabiblity of linear finite difference equations," Comm. Pure Appl. Math, vol. 9, pp. 267–293, 1956.

[47] R. D. Richtmyer and K. W. Morton, Difference methods for initial-value problems. Interscience, 2 ed., 1967.

[48] B. F. Feng and T. Mitsui, "A finite difference method for the Korteweg-de Vries and the Kadomtsev-Petviashvili equations," Journal of Computational Applied Mathematics, vol. 90, pp. 95–116, 1998.

[49] B. M. Herbst, A. R. Mitchell, and J. A. C. Weideman, "On the Stability of the Nonlinear Schrödinger Equation," Journal of Computational Physics, vol. 60, pp. 263–281, 1985.

[50] D. J. Griffiths, Quantum Mechanics. Pearson Prentice Hall, 2 ed., 2005.

[51] Z. Fei, V. M. Pérez-García, and L. Vázquez, "Numerical Simulation of Nonlinear Schrödinger Systems: A New Conservative Scheme," Applied mathematics and computation, vol. 71, pp. 165–177, 1995.

[52] M. S. Ismail and T. R. Taha, "A linearly implicit conservative scheme for the coupled nonlinear Schrödinger equation," Mathematics and Computers in Simulation, vol. 74, pp. 302–311, 2007.