# Dynamics of a Plasma Blob Studied With Particle-In-Cell Simulations

Vigdis Holta


Thesis submitted for the degree of
Master of Science

Department of Physics
Faculty of Mathematics and Natural Sciences

UNIVERSITY OF OSLO

Autumn 2018

# Dynamics of a Plasma Blob Studied With Particle-In-Cell Simulations

Vigdis Holta

Dynamics of a Plasma Blob Studied With Particle-In-Cell Simulations

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

**Abstract**

In this thesis the dynamics of a plasma blob in the scrape-off layer of a magnetic confinement device is investigated on the fundamental, kinetic level through particle simulations. New functionality is added to an existing particle-in-cell code to allow for a non-uniform initial particle density distribution, an inhomogeneous magnetic field, non-periodic particle boundary conditions and tracking of the center of mass of the blob. A parameter scan is conducted to find the effect of initial blob amplitude and ion temperature on the blob propagation. Both electron and ion densities are considered. The results of the particle simulations are compared to those of gyrofluid simulations with and without FLR corrections. A high degree of agreement with previous work is shown, and a disagreement on the direction of poloidal displacement is looked at in detail.

**Acknowledgements**

In addition to what you are about to read, I have learned two things from this thesis: 1) That you should celebrate your victories as they come, before you have time to realize that they were not as victorious as you thought, and 2) That people have a surprising capacity for help and support. This thesis has been the loneliest work of my life, yet I have so many people to thank:

Firstly, thank you to my supervisor Professor Wojciech Miloch and my co-supervisor Professor Odd Erik Garcia for coming up with the topic for my thesis. I want to thank Odd Erik especially for having me in Tromsø twice and sharing your expert knowledge, and Wojciech for valuable discussions, guidance and help in putting together this work.

A huge thanks to Sigvald for writing the PINC code and always meeting my questions, both specific and general, in such a welcoming way, seemingly never tiring of answering and explaining, and to Steffen for sharing PINC testing results and ideas, and always being interested in my progress.

Thank you, Professor Hide Usei and Associate Professor Yohei Miyake, for having me as a visitor at the University of Kobe, and thank you, Diako, Oki and Joakim, for the adventures we had there.

All my office mates through the years deserve a big thank you for putting up with my chattering. Special thanks to Victoria and Audun for kicker tournaments, pep talks and cheering, and to Klaus and Lei, whose warm and including welcome to the Space Physics group made me want to stay on for my thesis.

To my colleagues at Statkraft: Thank you for putting up with my ever-changing schedules, for your interest in my studies, and for opportunities I would not have dreamed of.

Jon Vegard and Arnlaug, you were both part of the reason I made it through the first years: Thank you for collaboration and support through many long semesters at Blindern.

I have had many desperate and disillusioned conversations with both Siri and Ann-Helen, but also some cheerful and optimistic ones. Thank you both for acknowledging the struggles of master life, nobody understood better than you.

Mamma and Pappa, thank you for giving me an interest in natural sciences, for being there to answer all my questions through almost 20 years of education, and for always listening to my rants and offer solutions and support when the need to ventilate and discuss outgrew the need for answers.

Lastly, Oliska: There is no problem too big to be fixed by your soothing purrs and soft cuddles.

To all the rest of my family and friends who have been there for me and supported me: You are all superheros.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Plasma is often called "the fourth state of matter", with the first three being solids, liquids and gases. When a gas is heated to a sufficient temperature, the electrons will have enough energy to tear away from the atom cores, which means the gas becomes *ionized*; ions and electrons exist separately in the gas. While a plasma can be fully ionized, a gas can still be a plasma even if it is only partially ionized and consists of neutral atoms as well as free electrons and ions. In an ionized gas, the electrons and ions can move separately, which allows for electric currents in the gas, for induced magnetic fields, and for external magnetic and electric fields to influence the gas. What is special about the plasma, then, is not only the high temperatures, but its *electromagnetic properties*. These properties will be explored when basic plasma physics is presented in Chapter 2.

Although plasma may be the least known state of matter to most people, it makes up 99% of the visible matter in the universe [28]. On Earth plasma is found in plasma TVs and in fluorescent lamps, and it occurs naturally in lightening. Close to Earth, plasma is responsible for the auroras. What really has an impact on the amount of plasma in the universe are *stars*. These, including our own sun, consist of gases, mostly hot enough to be ionized. Considering that the Sun makes up 99.8% of the mass of our solar system, it does not seem so unreasonable that plasma makes up 99% of the visible mass in the universe.

## 1.1 Plasma Fusion

In the core of stars, energy is created by fusing atoms together, creating heavier atoms and releasing energy. The atoms need sufficiently high kinetic energy to get close enough to fuse, and the high temperature of the plasma can provide this energy.

This is the opposite process of nuclear *fission*, often referred to as nuclear power, where energy is released through the process of splitting heavy atoms into lighter ones. It is the binding energy of the atom that decides if fusion or fission will provide energy. In figure 1.1 the binding energy of different atoms are plotted, showing that up until iron, nuclear fusion will generally provide energy [56]. While nuclear fission has the problems with radioactive waste and being unstable, this is not the case for nuclear fusion. By trying to recreate the conditions within the Sun in controlled environments on Earth, the goal is to provide cheap and clean energy for the future through *plasma fusion*.

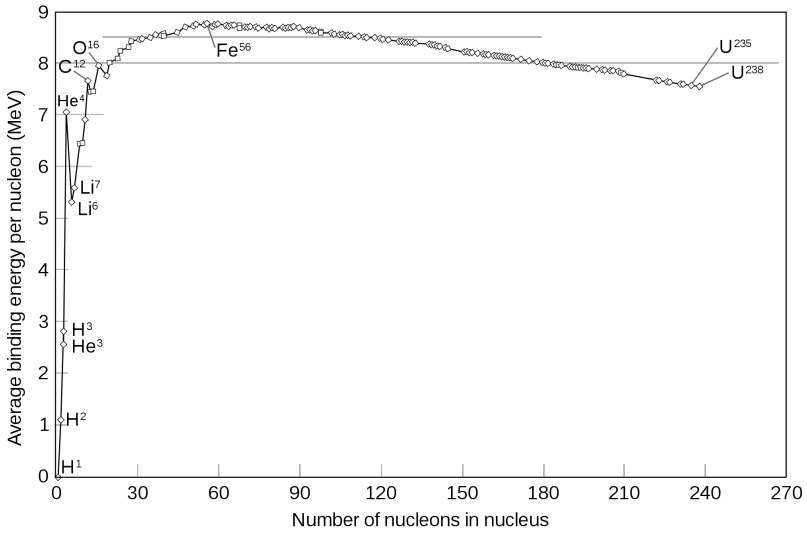


**Figure 1.1:** Binding energy of atoms. The figure shows that fusing atoms up until iron will generally provide energy. (*Numbers calculated for Einstein Online using data from the Atomic Mass Data Center.*)

The plasma in the Sun is held in place by its strong gravitational field. In plasma fusion experiments on Earth, the plasma is confined in a fusion reactor where it is held in place by an external magnetic field, leading the reactor to also be called a magnetic confinement device. The magnetic confinement is possible because of the electromagnetic properties of the plasma. There are several types of fusion reactors; the most successful ones have a toroidal shape, while they use different external magnetic field configurations to confine the plasma, as illustrated in figure 1.2.

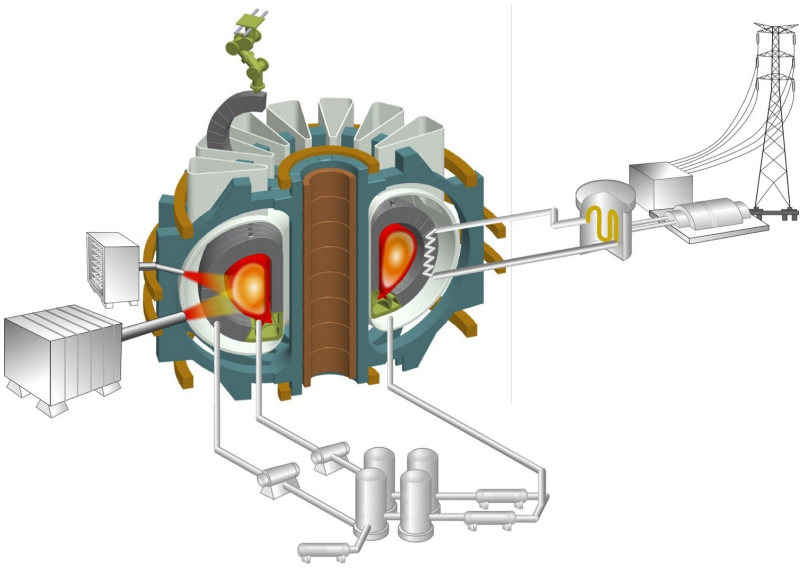

**Figure 1.2:** Schematics of the K-DEMO, South Korean tokamak under development, illustrating the magnetic confinement device as well as the fusion process. The orange-red core shows the plasma in the main chamber. (*South Korea's National Fusion Research Institute*, `www.nfri.re.kr`)

### 1.1.1 The Fusion Process

Although the exact configuration of a magnetic confinement device may vary greatly, the basic principles of how to get energy from plasma fusion remain the same. A gas, often consisting of deuterium ($^2H$) and tritium ($^3H$), is heated until it has enough energy to fuse the nuclei together and create helium. At this point the gas is ionized. Deuterium and tritium are usually chosen for plasma fusion as these enable the highest reactivity at the lowest temperature. When the fusions happens, a helium nucleus and a neutron are created, containing 17.6 MeV of excess energy.

$$^2H + ^3H \rightarrow He + n + 17.6MeV$$

where 3.4 MeV of the new energy is the kinetic energy of the new helium ion, $He$, and the additional 14.1 MeV are carried by the neutron, $n$ [29].

The neutron, as it is neutral, will not be affected by the magnetic field of the confinement device, and is therefore free to travel towards the walls. Here it is absorbed, and the energy is transformed into heat. This energy can then be used to heat water and create steam that can drive electric generators, as illustrated in figure 1.2.

The energy of the helium ion is transferred to other particles in the plasma through collisions. It only takes a few seconds for the helium ion to lose its energy and become *helium ash*. It is then considered an impurity that should be taken out of the device as the helium ash will not be a part of a future fusion process, and new deuterium and tritium particles will be let into the plasma instead, to *refuel* the plasma with new particles that will be ionized [29].

### 1.1.2 The Scrape-Off Layer

Keeping ions away from the walls of the fusion device is crucial. If ions hit the wall, they will break loose neutral atoms from the wall that will pollute the plasma, just like the helium ash is considered pollution. Another problem of breaking atoms from the wall is that it will reduce the life of the device as the walls will be eroded over time [42].

The magnetic fields are configured in such a way as to contain the plasma and keep it in place away from the walls, but some of the ions will still move outwards towards the wall. Because of this there is a *scrape-off layer* between the main chamber and the wall. In the scrape-off layer the plasma is "scraped off" and diverted to an area where the particles can be handled in a controlled way, instead of letting it hit the walls. This is also how the helium ash is taken out of the plasma. In the scrape-off layer the magnetic field lines are almost aligned with the walls. Ionized particles should travel along these field lines, and be lead to the divertor plates. The scrape-off layer and the expected and observed plasma behavior in this region is discussed in greater detail in Chapter 3.

### 1.1.3 The Problem with Blobs

Blobs are structures of plasma with density higher than that of the surrounding background plasma, which arise through instabilities and turbulence in the edge plasma. In the plane perpendicular to the magnetic field, the blobs are circular,

while they are elongated in the parallel plane, giving them a tube shape. As will be explained in more detail in Section 3.5, the blobs give rise to an electric field, which in turn makes the blob travel across the magnetic field lines which should have "scraped" the particles down to the divertors. The blob is then able to pass through the scrape-off layer, where the ionized particles hit the wall, erode it and release impurities.

In contrast to the expected parallel flow in the scrape-off layer, it has been found that the cross-field flow is as large, or larger than the parallel flow [43]. The cross-field particle transport is partially carried by blobs, and their contribution to the contamination of the plasma, as well as their negative effect on the lifetime of the components of the fusion device, make them a problem worth studying and understanding in order to find a way of eliminating them.

## 1.2 Studying Plasma with Numerical Simulations

Running plasma experiments can be extremely expensive and take a lot of time. This is true for space plasma experiments that require rockets or satellites, and it is also true for plasma fusion experiments, that require a fusion device and large amounts of energy. Fortunately, there is another way of studying plasma: through numerical simulations.

Although scientists have often been divided into "experimental" and "theoretical" scientists, it is widely accepted that there is a strength in combining theory with experimental results. As computers have gotten more and more powerful, numerical simulations have gained a place in this relationship. Theories can be tested in numerical simulations, expected results can be found before the experiment is set up. Numerical simulations can be set up to reproduce an experiment, to test different theories and see how they fit the experimental results. Numerical simulations can also be used to find results that may be hard to measure in real life, or to mimic conditions that are hard to access in real life because they are hard to reach (like space) or expensive to go through with (like plasma fusion).

Plasma is mainly described using two different approaches, one being the fluid approach where the plasma is considered at a macroscopic level as a conducting gas, while the other is the fundamental kinetic approach, where the plasma is described through the collective behaviour of the particles and their interaction. These two different views, explained in Section 2.4 and 2.5, give rise to two different kinds of numerical plasma simulations: fluid simulations and particle simulations, respectively.

A full particle simulation is computationally expensive, but it has the advantage of being run by fundamental equations where few or no assumptions are needed. A fluid simulation, on the other hand, is not as computationally demanding, but rely to a greater extent on assumptions and approximations. Chapter 4 touches briefly upon fluid simulations and goes into great detail about particle simulations.

## 1.3 Previous Work and The Goal of This Study

A great number of studies have been conducted to understand the conditions of the scrape-off layer and the dynamics of the plasma blobs [15] [22]. The problem has

been investigated through theoretical studies, observations and measurements from experiments, and numerical simulations. The results of some of these studies are presented in Section 3.5.

Although a large number of fluid simulations have been conducted, there is a lack of studies on the fundamental, kinetic level. In recent years there have been a few papers published on the study of blobs with particle simulations: A three-dimensional particle-in-cell code was developed by Hasegawa and Ishiguro ([34]) and used to simulate the dynamics of plasma blobs of varying sizes [30] [31], and a particle-in-cell code was also used to study the parallel dynamics of a blob by Costea *et al* [11]. However, there are still several topics of blob dynamics which are not yet explored on the kinetic level through particle simulations.

This thesis aims to study the dynamics of a plasma blob in the scrape-off layer using particle simulations to account for the fundamental kinetic effect, investigating macroscopic properties like blob propagation and velocity, and comparing the results to those found in fluid simulations and experiments. Blob generation is outside of the scope of this thesis, which will rather look at seeded blobs and study the dynamics of these. The simulations conducted for this thesis were limited to two dimensions, and the blob dynamics are studied in the cross-field plane perpendicular to the magnetic field, without taking into account the effects of the parallel plane. Chapter 5 contains a detailed explanation of the implementation and simulation setup. The results from the simulations are presented in Chapter 6, and they are discussed and compared to previous work in Chapter 7.

# Chapter 2

# Theoretical Plasma Background

Chapter 1 gave a high-level description of what a plasma is. This chapter looks at how to characterize plasmas and describe their behavior, starting with the motion of single charged particles in electric and magnetic fields, going on to the kinetic description of a large amount of charged particles, and ending with the fluid description of plasma.

## 2.1 Plasma Parameters

Plasma is often described through a characteristic length and a characteristic time. A commonly used time scale is based on the plasma frequency [9]

$$\omega_{pe} = \sqrt{\frac{q_e^2 n_e}{\epsilon_0 m_e}}, \tag{2.1}$$

where $q_e$ is the electron charge, $n_e$ is the electron number density, $m_e$ is the electron mass and $\epsilon_0$ is the permittivity of vacuum. Note that by replacing the electron values with those of ions, an ion plasma frequency can be found. The plasma frequency gives the frequency at which the electrons oscillate against an assumed fixed background of ions, an assumption that is often valid because of the high mass of the ions compared to the electrons [52]. The characteristic time scale of a plasma becomes $\tau_p = \omega_{pe}^{-1}$.

A characteristic length can be the distance traveled by an electron with thermal velocity $v_{th,e} = \sqrt{k_b t_e / m_e}$ (where $t_e$ is the electron temperature and $k_b$ is the Boltzmann constant) within the characteristic time scale $\tau_p$, giving

$$\begin{aligned} \lambda_D &= v_{th,e}\tau_p && (2.2)\\ &= v_{th,e}\omega_{pe}^{-1} && (2.3)\\ &= \sqrt{\frac{k_b t_e}{m_e}}\sqrt{\frac{\epsilon_0 m_e}{q_e^2 n_e}} && (2.4)\\ &= \sqrt{\frac{\epsilon_0 k_b t_e}{q_e^2 n_e}} && (2.5) \end{aligned}$$

This characteristic length scale is known as the electron *Debye length*, and it defines a *shielding distance*. The electric potential of a point charge will be shielded by

particles of opposite charge that have gathered around it, neutralizing the potential at a distance $\lambda_{De}$ away from the particle [26]. Again, by replacing the electron values with those of an ion, the ion Debye length can be defined.

Although these characteristic parameters are commonly used when describing plasmas, they are not the only possible options. In this thesis, to simplify comparison with earlier works, two other characteristic parameters have been used, namely the *ion gyrofrequency*

$$\Omega_i = \frac{q_i B}{m_i} \tag{2.6}$$

and the *ion sound speed gyroradius*

$$\rho_s = \frac{c_s}{\Omega_i} \tag{2.7}$$

where $c_s = \sqrt{k_b t_e / m_i}$ is the acoustic speed. The gyroradius and gyrofrequency are explained in the following section through investigation of the motion of charged particles in electric and magnetic fields.

## 2.2 Single Particle Motion

The force on a charged particle in an electric field $\mathbf{E}$ and magnetic field $\mathbf{B}$ is given by the Lorentz force

$$\mathbf{F} = q\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right), \tag{2.8}$$

which can be inserted into Newton's second law to give

$$m\mathbf{a} = \mathbf{F} \tag{2.9}$$

$$\mathbf{a} = \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}), \tag{2.10}$$

describing the acceleration of the particle. In the absence of a magnetic field, with a constant homogeneous electric field, the last term of (2.10) cancels out, which means that electrons and ions will simply travel on straight paths along the electric field, in opposite directions, creating an electric field.

If there is only a constant homogeneous magnetic field, the $\mathbf{E}$ term of (2.10) disappears, and the only force will be the $\mathbf{v} \times \mathbf{B}$ force. Particles with velocities in the plane perpendicular to $B$, $v_\perp$, will start gyrating in circles with a radius [9]

$$\rho_g = \frac{m v_\perp}{|q| B}. \tag{2.11}$$

This radius is referred to as the *gyroradius* or the *Larmor radius* of the particles. The frequency with which a particle gyrates becomes [9]

$$\Omega_g = \frac{qB}{m}, \tag{2.12}$$

known as the *gyrofrequency* or the *cyclotron frequency*, where the sign of $q$ indicates the direction of gyration. The characteristic frequency in (2.6) is found by inserting the values for an ion, while $\rho_s$ in (2.7) is the gyroradius of an ion travelling with the sound speed, $c_s$.

Because of their opposite charges, electrons and ions will gyrate in opposite directions, and because of the high ion mass compared to the electron mass, the ion gyroradius will be much larger than the electron gyroradius, while the cyclotron frequency of electrons will be higher than the cyclotron frequency of the ions,

$$\rho_e < \rho_i, \qquad \Omega_e > \Omega_i \tag{2.13}$$

The velocity component $v_{||}$ in the plane parallel to $B$ will be unaffected by the magnetic field, and particles with $v_{||} \neq 0$ will travel along the magnetic field lines while they gyrate around them (if $v_\perp \neq 0$).

## 2.3 Particle Drifts

In most cases it is of more interest to consider the *drift* of the particles. The drift can be defined as the change in the average position over a certain period of time. The average position of a particle over one gyro period, $T_g = 2\pi/\Omega_g$ is called the *guiding center*, $\mathbf{R_g}$. The velocity of the guiding center, $\mathbf{u}$, is the particle drift velocity [51].

For the case of an electric field and no magnetic field, $\mathbf{R_g}$ will simply follow the path of the particle. In the case of the magnetic field considered above, the guiding center of a particle with only perpendicular velocity, $v_\perp$, will not change over time, giving $\mathbf{u} = 0$. If, however, the particle velocity also has a parallel component, $v_{||}$, the particle will drift along the magnetic field with the drift velocity

$$\mathbf{u} = \frac{d\mathbf{R_g}}{dt}. \tag{2.14}$$

### 2.3.1 E||B

In the case of a homogeneous electric field parallel to a homogeneous magnetic field, eq. 2.8 will keep both terms. The electric field will give the particles a constant acceleration along the field lines. If there is a perpendicular term to the velocity, the magnetic field will give a gyration in the plane perpendicular to the field lines, while the electric field accelerates the particle in the direction parallel to the field lines, and the result will be a spiral motion in the direction parallel to the fields [51]. In both cases, the guiding center will remain constant in the plane perpendicular to $\mathbf{E}$ and $\mathbf{B}$, while it moves along the field lines. Ions and electrons will both gyrate and drift in opposite directions, giving rise to an electric current.

### 2.3.2 E ⊥ B

In the case of a magnetic and electric field perpendicular to each other, there will be a gyration in the plane perpendicular to $\mathbf{B}$, and as the particle gyrates in the direction of the electric field it will be accelerated further, while it while be decelerated as it travels against the electric field (assuming it is positively charged; the opposite will be true for a negatively charged particle). (2.11) shows that the gyroradius will be larger when $v_\perp$ is increased by the electric field, and smaller after it has been decreased by traveling against the electric field, which is illustrated in figure 2.1a. The total effect of this is an $\mathbf{E} \times \mathbf{B}$ drift that is perpendicular to both $\mathbf{E}$ and $\mathbf{B}$ [9],

$$\mathbf{u} = \frac{\mathbf{E}_\perp \times \mathbf{B}}{B^2}. \tag{2.15}$$

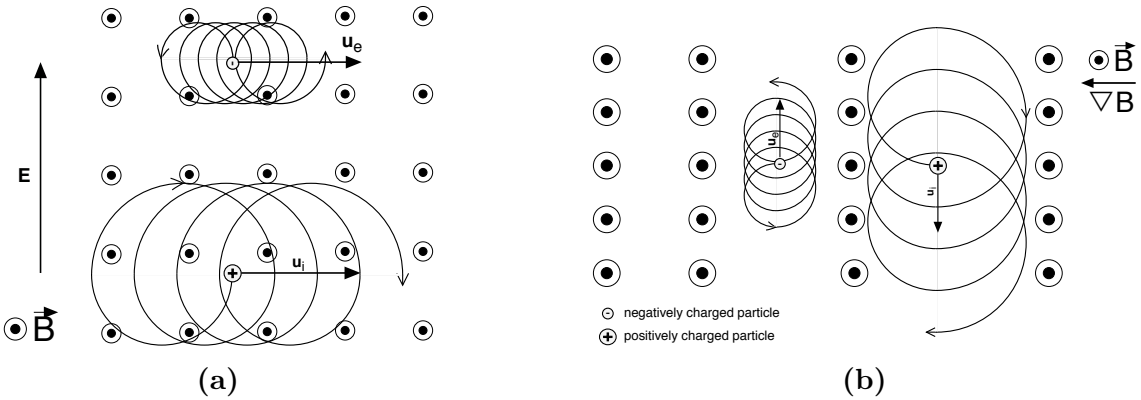


**Figure 2.1:** Illustrations of particle drifts, with the magnetic field coming out of the paper Panel **a** illustrates the drift of electrons and ions when there is an electric field perpendicular to a magnetic field, $\mathbf{E} \perp \mathbf{B}$. In panel **b** there is a gradient in the magnetic field going to the left, and the particles have a $\mathbf{B} \times \nabla B$ drift in opposite directions.

### 2.3.3 $\nabla B \perp \mathbf{B}$

The electric force $q\mathbf{E}$ in (2.8) can be exchanged with other forces, like the gravitational force, which can also affect the motion of the particles. A highly relevant example for this thesis is when there is a gradient in the magnetic field perpendicular to the magnetic field, $\nabla B \perp \mathbf{B}$.

From eq. 2.11 it is clear that the gyroradius of the particle will decrease as the magnetic field strength increases. An illustration of this, and the resulting drifts, can be seen in figure 2.1b.

The drift of the particles in this case are given by [9]

$$\mathbf{u} = \frac{\rho_g v_\perp}{2} \frac{\nabla B \times \mathbf{B}}{B^2}, \tag{2.16}$$

and electrons and ions will drift in opposite directions.

The $\mathbf{E} \times \mathbf{B}$ and $\mathbf{B} \times \nabla B$ drifts were presented here because they are of relevance to this thesis. There is however more to say about particle drifts and orbits, and interested readers are referred to Chapter 2 of *Introduction to Plasma Physics and Controlled Fusion* by Chen [9].

## 2.4 Kinetic Description

When studying plasmas, it is often not the motion of single particles that is of interest, but the behavior of a large number of particles. Kinetic theory considers the collective behavior of the particles on a fundamental level, where the particles are described through their position, $x$, and velocity, $v$, at a time $t$. The distribution function $f(\mathbf{x}, \mathbf{v}, t)$ is defined as the number of particles within a unit in space that have velocity components within a certain range [9]. This is not a probability distribution function, but a function that describes the actual distribution of particles in the *phase space*. The time evolution of this particle distribution is described by the Vlasov equation:

$$\frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial t} + \mathbf{v} \cdot \frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{x}} + \mathbf{F} \cdot \frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{v}} = 0 \tag{2.17}$$

where $\mathbf{F}$ is the Lorentz force from (2.8). Although this equation may look simple, the fields implied by $\mathbf{F}$ must at every time step be self-consistently determined [46], as they are affected by the particles and affect the particles at the same time. Interesting macroscopic properties of the plasma can be found through the *moments* of the Vlasov equation [13]. In Chapter 4 a numerical solution satisfying the first moments of the Vlasov equation is explained, with the zeroth moment being the integration of the full Vlasov equation over the spatial domain and the velocity domain:

$$\int\int \left( \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial t} + \mathbf{v} \cdot \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial \mathbf{x}} + \mathbf{F} \cdot \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial \mathbf{v}} \right) \quad d\mathbf{x}d\mathbf{v}. \tag{2.18}$$

The first spatial moment (or moments, if there are more than one spatial dimensions), is

$$\int\int \mathbf{x} \cdot \left( \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial t} + \mathbf{v} \cdot \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial \mathbf{x}} + \mathbf{F} \cdot \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial \mathbf{v}} \right) \quad d\mathbf{x}d\mathbf{v}, \tag{2.19}$$

and the first velocity moments are similarly given by

$$\int\int \mathbf{v} \cdot \left( \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial t} + \mathbf{v} \cdot \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial \mathbf{x}} + \mathbf{F} \cdot \frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial \mathbf{v}} \right) \quad d\mathbf{x}d\mathbf{v}. \tag{2.20}$$

In high-temperature plasmas, collisions are infrequent because of the long-range Coulomb interactions between the particles [13]. The lack of collisions leads the particles often having a non-Maxwellian velocity distribution [26]. Since no assumption is made about the velocity distribution in $f(\mathbf{x},\mathbf{v},t)$, kinetic theory is valid for any distribution [9]. If there is a need to include collisions in the description, a collision term $\left(\frac{\partial f}{\partial t}\right)_{col}$ can be added to the right hand side of (2.17). If, however, the collisions are frequent enough that Maxwellian velocity distributions are maintained, it is possible to model the plasma using a *fluid description* [26].

## 2.5 Fluid Description

While the kinetic description considered the microscopic properties of the plasma, it can also be considered on the macroscopic lever, as a conducting gas, which can be described through hydrodynamic equations. Assuming that no particles are created or destroyed, plasma follows the continuity equation [25]

$$\frac{\partial \rho_m}{\partial t} + \nabla \cdot (\mathbf{u}\rho_m) = 0 \tag{2.21}$$

where $\rho_m$ is the mass density. As the velocity field $\mathbf{u}$ can have a spatial as well as temporal variation, Newton's second law in continuum mechanics becomes

$$\rho_m \left( \frac{\partial}{\partial t}\mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = \mathbf{F} \tag{2.22}$$

For plasmas, this results in the *Navier-Stokes* equation of motion [25]

$$\rho_m \left( \frac{\partial}{\partial t}\mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \mathbf{J} \times \mathbf{B} + \rho_m \mathbf{g} \tag{2.23}$$

where the force $\mathbf{F}$ is the sum of the pressure force $-\nabla p$, the magnetic force $\mathbf{J} \times \mathbf{B}$, where $\mathbf{J}$ is the current density, and a gravitational force $\rho_m \mathbf{g}$, which can often be neglected.

As the plasma has electromagnetic properties, the laws of Faraday (2.24), Ampere (2.25) and Ohm (2.26) are needed to close the set of equations:

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{2.24}$$

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} \tag{2.25}$$

$$\mathbf{J} = \sigma_p (\mathbf{E} + \mathbf{u} \times \mathbf{B}) \tag{2.26}$$

where $\sigma_p$ is the plasma conductivity and $\mu_0$ is the permeability of free space.

Combining (2.26) with (2.24) gives the magnetic field equation found in (2.27), while inserting Ampere's law (2.25) into the Navier-Stokes equation (2.23) gives (2.29).

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{u} \times \mathbf{B}) + \frac{1}{\mu_0 \sigma_p} \nabla^2 \mathbf{B} \tag{2.27}$$

$$\frac{\partial \rho_m}{\partial t} + \mathbf{u} \cdot \nabla \rho_m = -\rho_m \nabla \cdot \mathbf{u} \tag{2.28}$$

$$\rho_m \left( \frac{\partial}{\partial t} \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \frac{1}{\mu_0} (\nabla \times \mathbf{B}) \times \mathbf{B} \tag{2.29}$$

$$p = P(\rho_m) \tag{2.30}$$

Additionally, an equation of state (2.30) has been introduced into the set of equations. Together, these equations make up the system of magneto-hydrodynamic (MHD) equations [25].

In many cases the MHD equations can be simplified. By assuming an *ideal* conductor where the conductivity $\sigma_p \to \infty$ the magnetic field equation (2.27) becomes:

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{u} \times \mathbf{B}), \tag{2.31}$$

and if *incompressibility*, $\nabla \cdot \mathbf{u} = 0$, can be assumed, the continuity equation (2.28) is simplified to:

$$\frac{\partial \rho_m}{\partial t} + \mathbf{u} \cdot \nabla \rho_m = 0 \tag{2.32}$$

### 2.5.1 Two-Fluid Models

As the plasma is ionized and the electrons and ions can move independently of each other, it is natural to consider them as two different fluids that both follow the continuity equation:

$$\frac{\partial \rho_{m,e}}{\partial t} + \nabla \cdot (\rho_{m,e} \mathbf{u}_e) = 0 \tag{2.33}$$

$$\frac{\partial \rho_{m,i}}{\partial t} + \nabla \cdot (\rho_{m,i} \mathbf{u}_i) = 0 \tag{2.34}$$

The momentum equation (2.29) can be found for both fluids in the same way, and when adding them together, gives [51]

$$\rho_{m,e+i} \frac{\partial \mathbf{u}_{e+i}}{\partial t} = -\nabla (p_e + p_i) + q_e (n_i - n_e) \mathbf{E} + \mathbf{J} \times \mathbf{B} \tag{2.35}$$

where $\mathbf{u}_{e,i} = (\rho_{m,e}\mathbf{u}_e + \rho_{m,i}\mathbf{u}_i)/\rho_{m,e+i}$ is the average velocity of the particles and $\rho_{m,e+i} = \rho_{m,e} + \rho_{m,i}$ is the combined mass density of the electrons and the ions.

Assuming *quasi-neutrality* $n_e \approx n_i$, that the plasma consists of approximately the same number of ions and electrons and is therefore neutral on large scales, the second term on the right hand side of the equations is reduced to zero.

### 2.5.2 Gyrofluid Models

As will become clear in Section 3.5.4, it is sometimes necessary to introduce kinetic effects into the fluid equations, even when only large-scale phenomena are studied. This is true when microscopic events affect the macroscopic properties in a non-negligible way [2]. Specifically, it is of interest to account for the ion dynamics, and to model the effects of gradients and curvatures in the magnetic field, and finite Larmor radius (FLR) effects, which are the effects of the non-zero ion gyroradius [21]. While these effects are inherently a part of the kinetic description, they need to be incorporated into the fluid description. Fluid models that account for kinetic effects are called *gyrofluid* models, named from the fact that they use the gyrokinetic equations to include kinetic effects in fluid models [16].

The quasi neutrality is expressed through the *polarization equation* [32]

$$n_e - \frac{N_i}{1 - \nabla^2 \frac{\rho_i^2}{2}} = \Omega_E, \tag{2.36}$$

where $n_e$ is the electron density and $N_i$ is the ion *gyrocenter density*, which should not be confused with the ion particle density. On the right hand side is the FLR corrected polarization charge [44]. $N_i$ is multiplied by the Padé approximant [16]

$$\Gamma_{1,i}^{\dagger} = \frac{1}{1 - \nabla^2 \frac{\rho^2}{2}} \tag{2.37}$$

which introduces the FLR corrections. In the case of cold ions, $t_i = 0$, the ions will not gyrate and thereby have no gyroradius, $\rho_i = 0$, reducing (2.37) to 1. $N_i\Gamma_{1,i}^{\dagger}$ gives, at any position, the average charge contribution of all ions whose orbit intersect that position [32].

The continuity equation for the gyrocenter density of any specie, $N$, becomes [32]

$$\frac{\partial N}{\partial t} + \nabla \cdot (N[\mathbf{u}_{E\times B} + \mathbf{u}_{B\times\nabla B} + \mathbf{u}_\eta]) = -\nu\nabla^4 N \tag{2.38}$$

where $\mathbf{u}_{E\times B}$ accounts for the $\mathbf{E}\times\mathbf{B}$ drift, $\mathbf{u}_{B\times\nabla B}$ for the $\mathbf{B}\times\nabla B$ drift and $\mathbf{u}_\eta$ for the FLR corrections. On the right-hand side of the equation is a diffusion term.

As can be seen from the equations above, it is the position and velocity of the gyrocenter that is considered, not of the particle itself [21] ([58] [54]). The gyrofluid models have been shown to reproduce to a large extent the behaviour of kinetic models [2].

# Chapter 3

# Plasma in the Scrape-Off Layer

Because of the electromagnetic properties of plasma presented in Chapter 2, it is possible to use magnetic fields to confine plasma. This is the basic principle upon which magnetic confinement devices for plasma fusion are built. There are many different ways to set up a magnetic confinement device, the most notable one being the tokamak. Even within tokamak designs there are a great variation of possible structures. In this chapter a simplified model of a the magnetic fields of a tokamak with divertors is presented.

The scrape-off layer of a magnetic fusion device is the area close to the walls, where the plasma is dominated by instabilities, and it is here that the blob-like structures that are the topic of this thesis are found. Some important properties of the scrape-off layer are explained in this chapter, and the plasma behaviour in this area and ultimately plasma blobs are discussed. Readers who want to learn more about the scrape-off layer are referred to the book *The Plasma Boundary of Magnetic Fusion Devices* by P C Stangeby [57].

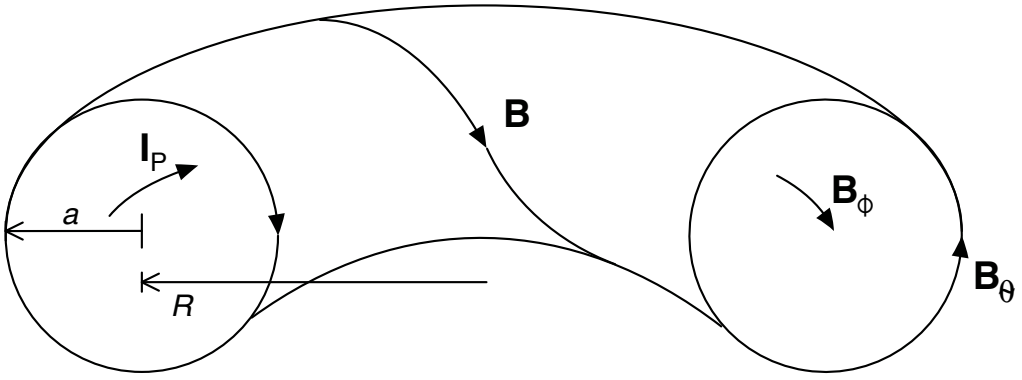


**Figure 3.1:** A simplified illustration of the magnetic field $\mathbf{B} = \mathbf{B}_\phi + \mathbf{B}_\theta$ in a tokamak with major radius $R$ and minor radius $a$. (*Adapted from* [57].)

## 3.1 Magnetic Fields of A Tokamak

The magnetic field of a tokamak consists of two components, $\mathbf{B}_\phi$ in the toroidal direction (parallel to the plasma current, $\mathbf{I}_p$) created by external coils, and $\mathbf{B}_\theta$ in the poloidal direction, caused by the plasma current in the main chamber. This results

in a total magnetic field $\mathbf{B} = \mathbf{B}_\phi + \mathbf{B}_\theta$ that twists around the torus, as shown in figure 3.1. The external magnetic field is usually much stronger than the magnetic field caused by the plasma current, $|\mathbf{B}_\phi| >> |\mathbf{B}_\theta|$, giving the total magnetic field $\mathbf{B}$ a shallow pitch angle, making it almost parallel to the toroidal direction.

### 3.1.1 Magnetic Field Gradient

The magnetic field in the main plasma chamber gets weaker towards the outer walls of the tokamak, which results in a magnetic field strength [8]

$$B = \frac{B_0}{1 + \frac{x}{R}} \tag{3.1}$$

where $R$ is the major radius and $0 \leq x \leq a$ is a point between the middle of the plasma column and the wall, with $a$ being the minor radius of the device, as shown in figure 3.1.

In [8] values for the TBR-1 tokamak are listed as

$$B_0 = 0.5 \text{ T} \tag{3.2}$$
$$R_0 = 0.30 \text{ m} \tag{3.3}$$
$$a = 0.11 \text{ m} \tag{3.4}$$

giving a magnetic field of strength $B(x = 0) = B_0 = 0.5$ T at the center of the plasma column and $B(x = a) \approx 0.73B_0 = 0.37$ T at the wall, which shows a 27% reduction in the magnetic field over a distance of 0.11 m.

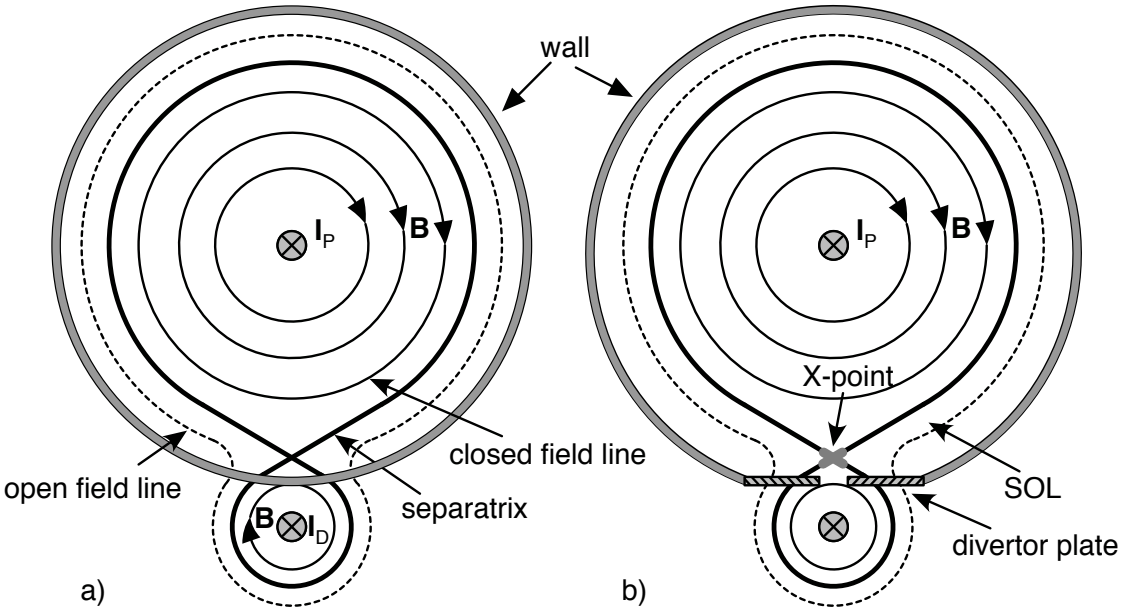


**Figure 3.2:** Illustration of the divertor. Figure **a** shows an illustration of the magnetic field created by the two parallel currents, $\mathbf{I}_p$ and $\mathbf{I}_D$, while figure **b** points out the separatrix, the X-point and the location of the divertor plates. (*Adapted from* [57].)

### 3.1.2 The Divertor

The plasma in the main chamber will mostly flow in the toroidal direction as the plasma current, $\mathbf{I}_p$, illustrates in figure 3.1. However, the plasma will at the same time diffuse outwards towards the wall, creating the need for a way of controlling the plasma that gets close to the wall and is in danger of hitting it [57].

By adding an external current $\mathbf{I}_D$ with the same direction as the plasma current, another magnetic field can be set up outside of the main plasma chamber, as illustrated in figure 3.2a. At a certain point between these two fields, known as the *X-point*, the two magnetic fields will cancel each other out. The magnetic field lines that pass through this point are referred to as the *separatrix*, because they mark the separation between the *closed* and *open* field lines. The open field lines are those in figure 3.2 where the magnetic fields caused by $\mathbf{I}_p$ and $\mathbf{I}_D$ are connected, which pass through the walls of the main chamber. Two solid plates known as the *divertors* cut through these field lines as shown in figure 3.2b, and will work as sinks for the plasma [50].

Particles that cross the separatrix by diffusion will enter into the area of open field lines connected to these solid plates and start moving rapidly along the magnetic field towards the divertors. In the area between the separatrix and the wall, plasma will be "scraped off", resulting in the area being named the scrape-off layer (SOL).

## 3.2 Plasma in the SOL

When plasma crosses the separatrix and enters the SOL, the expectation and aim is that it should travel along the magnetic field towards the divertors. Yet, observations show that most of the particle transport in the SOL is *perpendicular* to the magnetic field, and in the Alcator C-Mod the plasma in the SOL was observed to flow mainly radially towards the walls, and not to the divertors as intended [62].

The observations also implied that the radial plasma transport was too high to be caused by diffusion alone [62], which is supported by later experiments; it was estimated that $\sim 50\%$ of the radial transport in the SOL of the DIII-D tokamak was carried by plasma *objects* with enhanced densities [5]. These objects were found through measurements of turbulence in the SOL as density fluctuations with an amplitude $\Delta n = [0.05, 1]n_0$ [67], where $n_0$ is the background particle density, and they are often referred to as blobs or filaments.

The origin of the structures is not yet described analytically, but through experiments and computer simulations (i.e. [53], [47], [24], [4], [48]) they have been shown to be generated through non-linear saturation of the edge instabilities around the separatrix [15]. The turbulence of the SOL plasma is not completely understood yet either, and in the complex geometry of the SOL there are many factors that should be taken into account when attempting to model it [19].

### 3.2.1 Ion Temperature

While most models of plasma in the SOL assume the ion and electron temperature to be equal, $t_i = t_e$, this is not in agreement with measurements of SOL plasma [1]. In 1998 Uhera *et al* reported ion temperatures that were one order of magnitude

larger than the electron temperatures in the SOL of the JFT-2M tokamak, starting at $t_i = 5t_e$ at the separatrix and increasing towards the wall [61].

More recently, Adámek *et al* have shown $t_i = [1, 1.5]t_e$ in the SOL of the CASTOR tokamak, and report an ion temperature $t_i = [1.5, 3]t_e$ found in the JET tokamak [1]. Measurements from the MAST tokamak are in agreement with these results [18], with experiments estimating an SOL ion temperature $t_i \geq 2t_e$ at the midplane, and $t_i \approx [1, 1.5]t_e$ in the areas close to the divertor. In the ASDEX Upgrade tokamak, the ion temperature was observed to be $t_i = [2, 3]t_e$ in the SOL, and the ion temperature of blobs/filaments at the separatrix was estimated to be $3 - 4$ times larger than the ion temperature of the background plasma [37].

The increased ion temperature in the SOL can be explained by a higher parallel conductivity of electrons than ions, making the electrons experience a stronger cooling by parallel losses to the divertors [37][57].

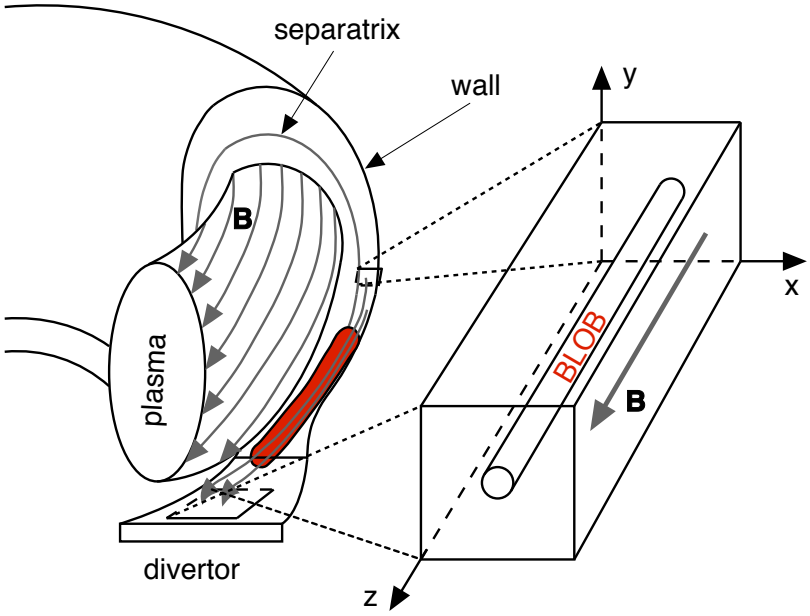


**Figure 3.3:** An illustration of the domain in 3D. The gray arrows indicate the magnetic field lines, while the red cylinder illustrates how the blob is aligned along the magnetic field. The enlarged box to the right shows the domain in Cartesian coordinates, and the domain discussed in this thesis is a slice of this box in the $xy$ plane. (*Adapted from* [31].)

## 3.3 Defining the Relevant Domain

So far, the topic of plasma fusion has been discussed on a large scale: An overview of the full process was given in Chapter 1, and in the current chapter a simplified version of a tokamak has been described. From this point on, however, only a very small domain will be relevant, namely the domain of the blob.

This domain is illustrated in 3D as a box in figure 3.3, with the blob shown as a cylinder along the magnetic field lines. The three-dimensional domain has Cartesian coordinates, with $x$ pointing in the radial direction, $y$ pointing in the poloidal direction and $z$ going in the toroidal direction. To simplify it even more, this thesis will predominantly be concerned with the cross-field plane ($xy$), perpendicular

to **B**, slicing through $z$ at a point far from the divertors, leaving a two-dimensional domain as shown in figure 3.4, where $z$ is pointing out of the paper. The positive $x$ direction will be referred to as *radially outward*, and the negative $x$ direction naturally becomes *radially inward*. Similarly, the positive $y$ direction will be referred to as the *positive poloidal direction*, which makes the negative $y$ direction the *negative poloidal direction*.

## 3.4 Blobs Definition

A blob is a structure of plasma with density higher than the surrounding plasma. They are referred to as blobs, filaments or blob-filaments, because they have a blob-like appearance in the plane perpendicular to **B** and are stretched out like filaments parallel to **B**. As this thesis will mainly consider the perpendicular $xy$ plane, the structures are referred to as blobs. In [15], a blob is defined through three properties:

- It has a single-peaked density distribution with a peak amplitude typically more than 2-3 times higher than the density fluctuations in the background plasma
- It is aligned parallel to the magnetic field and has a "length", $l_b$, along **B** that is much longer than the cross-field size of the blob, $\sigma$: $l_b \gg \sigma$
- There is a dominant $\mathbf{E} \times \mathbf{B}$ velocity component, and a potential and vorticity dipole structure in the direction transverse to the propagation

Images taken of blobs in the SOL of a tokamak can be seen in [45].

## 3.5 Blob Dynamics

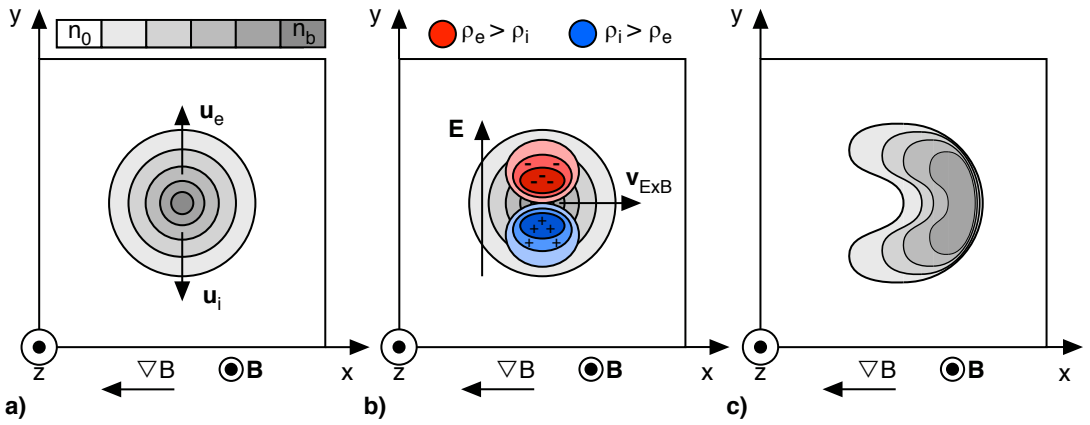


**Figure 3.4:** Illustration of the expected blob dynamics as time progresses. The magnetic field **B** is directed out of the paper, with a gradient $\nabla B$ in the negative $x$ direction. In **a** the $\mathbf{B} \times \nabla B$ force creates an electron drift, $\mathbf{u}_e$, in the positive $y$ direction and an ion drift, $\mathbf{u}_i$, in the negative $y$ direction. This creates a charge separation shown in **b** (where the area of higher electron density is shown in red and the area of higher ion density is shown in blue), giving an electric field **E**, which in turn gives the ions and electron an $\mathbf{E} \times \mathbf{B}$ drift, $v_{E \times B}$, in the positive $x$ direction. In **c** the blob is propagating along $x$, while taking on a "mushroom" shape.

The main characteristics of the blob propagation towards the wall can be explained through the single particle motion and resulting particle drifts explained in Chapter 2. In figure 3.4a a blob is placed in the middle of the domain, with a maximum density amplitude higher than the background density, $n_b > n_0$. There is a magnetic field pointing out of the paper, in the toroidal direction along $z$, with a gradient in the negative radial direction ($-x$). The resulting $\mathbf{B} \times \nabla B$ force leads to an electron drift $\mathbf{u}_e$ in the positive poloidal direction and an ion drift $\mathbf{u}_i$ in the negative poloidal direction, resulting in a charge separation that creates an electric field, shown in figure 3.4b. The electric field leads to an $\mathbf{E} \times \mathbf{B}$ force that makes both particle species drift in the radial direction, along $x$. As the dense center of the blob moves radially with the $\mathbf{E} \times \mathbf{B}$ drift $\mathbf{v}_{E \times B}$, the $\mathbf{B} \times \nabla B$ drifts will keep expanding the blob in the poloidal direction, which results in the mushroom-shaped blob seen in figure 3.4c.

This explanation gives a very simplified, qualitative description of the blob dynamic, which is, of course, in reality more complex than that. There are many factors that affect the blob propagation: the blobs can vary in density amplitude ($n_b$), cross-field size ($\sigma$), parallel length ($l_b$), electron and ion temperature ($t_i, t_e$), placement both in the cross-field plane and parallel to $\mathbf{B}$, and the conditions surrounding the blob may vary as well.

In the following sections care has been taken to convert the coordinates and variables of the referenced papers, and they are presented here in the coordinates and variables introduced in this thesis to avoid confusion. Focus has been kept on the cross-field causes and effect, as that is the relevant domain for this thesis.

### 3.5.1 Theoretical Descriptions of $\mathbf{v}_b$

Several models describing the dynamics of the blob exist, with one of the important properties being the blob velocity ([39] [14] [38] [55]). The theoretical models are based on the MHD equations presented in Chapter 2. Some of the conclusion about the blob velocity, $v_b$, from different theoretical models are presented here. In this section, as in the rest of this thesis, $v_x$ will denote the blob velocity in the radial direction, $v_y$ the blob velocity in the poloidal direction and $v_b = \sqrt{v_x^2 + v_y^2}$ the total blob velocity. The velocities have been derived based on one, two and three dimensional models, but as this thesis is mainly concerned with the perpendicular cross-field plane, most discussions related to the parallel plane are left out of this review.

**From Krasheninnikov (2001) [39]**

The blob continuity equation is derived

$$\frac{\partial n_b}{\partial t} + C_s \frac{\rho_i^2}{2} \frac{l_b}{R} \left( \frac{\partial}{\partial x} \left[ n_b \frac{\partial}{\partial y} \left( \frac{1}{n_0} \frac{\partial n_b}{\partial y} \right) \right] - \frac{\partial}{\partial y} \left[ n_b \frac{\partial}{\partial x} \left( \frac{1}{n_0} \frac{\partial n_b}{\partial y} \right) \right] \right) = 0 \tag{3.5}$$

which, through a suggested separable solution, leads to an expression for the radial blob velocity:

$$v_x = c_s \left( \frac{\rho_i}{\sigma} \right)^2 \frac{l_b}{R} \frac{n_b}{n_0}. \tag{3.6}$$

The blob is given an estimated lifetime $\tau_b \approx l_b / C_s$, denoting the time until the blob dissolves, and the radial distance the blob is expected to propagate becomes

$\Delta x_b \approx v_b \tau_b$. Based on parameters from the DIII-D tokamak, a blob is estimated to travel at $v_b \sim 10^5$ cm/s and propagate $\Delta x_b = 15$ cm, indicating a lifetime

$$\tau_b \sim 6.7 \times 10^{-7} \text{ s.} \tag{3.7}$$

**From D'Ippolito *et al* (2002) [14]**

Another solution for the radial blob velocity is given as

$$v_x = \frac{l_b}{R}\frac{1}{\sigma^2} \tag{3.8}$$

while the poloidal blob velocity $v_y = 0$, meaning that the blob will only propagate radially. This is based on an assumption that there is no electric field in the radial direction. An addition is made to the model and the following blob velocities are given

$$v_x = c_s \frac{l_b}{R}\frac{\rho_s^2}{\sigma^2} \tag{3.9}$$

$$v_y = v_y(t=0) + \frac{q}{B}\frac{\partial \phi}{\partial x} \tag{3.10}$$

Based on these equations, the blob is said to have a constant velocity in the radial direction, and a poloidal velocity that becomes negative for small $t$ and then later turns and becomes positive, with the blob moving back towards $y = y(t = 0)$.

**From Yu and Krasheninnikov (2003) [65]**

It is shown that blobs of a certain size are more stable than others; in a tokamak this size of a stable blob is said to be $\sim 1$ cm, and these can propagate $\sim 10$ cm or more. Large blobs are shown to break apart earlier due to Rayleigh-Taylor instabilities, while smaller blobs get a mushroom shape. A radial blob velocity

$$v_x = c_s \frac{2\rho_s^2}{\sigma^2}\frac{l_b}{R} \tag{3.11}$$

is shown for a tokamak, which is the same as (3.9) given in the appendix of [14]. It is concluded that the ratio between the blob density amplitude and the background density, $n_b/n_0$ does not affect the dynamics of the blob significantly.

**From Krasheninnikov *et al* (2004) [38]**

The plasma blob velocity is found to scale differently near the separatrix than in the far SOL, closer to the wall:

$$\text{close to the separatrix: } v_x \propto \frac{\rho_s}{\sigma} \tag{3.12}$$

$$\text{in the far SOL: } v_x \propto \left(\frac{\rho_s}{\sigma}\right)^2 \tag{3.13}$$

**From Ryutov (2006) [55]**

It is asserted that a blob will experience displacement in the poloidal direction as well as radially, and while the radial displacement will be towards the wall (in the positive $x$ direction), the poloidal displacement can be either in the positive or the negative direction, depending on placement of the blob in the toroidal direction.

In conclusion, there seems to be an agreement about the radial blob velocity scaling like $v_x \propto k(\rho_s/\sigma)$ or $v_x \propto k(\rho_s/\sigma)^2$ where $k$ depends on several factors, with the difference maybe being how far into the SOL the blob is found. Only one of the papers gives the blob density amplitude any significance [39]. The poloidal velocity is not agreed upon, but is said to depend on the toroidal placement of the blob [55], or on time [14].

### 3.5.2 Observations and Measurements

While it is difficult to measure phenomena in the core of the plasma, the edge areas where the SOL is located are easily accessible by measurement instruments like Langmuir probes. It has therefore been possible to gather experimental data on blobs and their dynamics.

#### From Zweben (1984) [66]

Early blob measurements using Langmuir probes showed blobs moving radially inward as well as outward, and with a significant poloidal displacement, mostly in the negative direction, but also in the positive direction. The blob lifetime $\tau_b$ was observed to be around $[3, 6] \times 10^{-6}$s, however the blobs propagated outside of the area measured by the probes, meaning their actual lifetime could be longer. The measurements show a correlation between the blob cross-field size and the lifetime, indicating that larger blobs live longer. There is no correlation shown between the blob amplitude and the velocity, and most blobs seem to propagate at around $[1, 3] \times 10^5$ cm/s.

#### From Endler *et al* (1995) [20]

Plasma structures are detected, and they are observed to be stretched out along the magnetic field, but localized within a few centimeters in the cross-field plane, which is one of the criteria in the description of blob-filaments in Section 3.4. It is shown that warm, high density plasma flows radially outwards and in the positive poloidal direction, while cold, low density plasma flows radially inwards and in the negative poloidal direction.

#### From Grulke *et al* (2006) [27]

The potential distribution of the blob is shown to be of dipole shape in the poloidal direction, with the positive potential being on the negative poloidal side and the negative potential being on the positive poloidal side, consistent with an $\mathbf{E} \times \mathbf{B}$ drift going radially outwards. The radial velocity is typically $v_x \sim 5 \times 10^4$ cm/s. The poloidal propagation is found to be in the negative direction, which is the same direction as the $\mathbf{E} \times \mathbf{B}$ drift of the *background* plasma, and short-time propagation in the positive poloidal direction is also observed, with the mean value being in the opposite direction, around $v_y = -2.5 \times 10^4$ cm/s.

The experimental observations show a more complex image of blobs than the predicted by theory. The blobs are found to move radially inward as well as outward, and in both poloidal directions. Later measurements show that the blob mainly moves radially *outwards* while the blob moves poloidally along the background $\mathbf{E}\times\mathbf{B}$ flow [27], and a connection is found between the temperature and density of the blob and the poloidal and radial propagation [20]. There is a larger degree of agreement about the radial propagation than the poloidal. The blob velocity is reported to be $\sim [10^4, 10^5]$ cm/s, depending on the parameters, which is in agreement with theory [39].

### 3.5.3 Simulation Results

To add to the theory and measurements, numerical simulations have been carried out to investigate blob propagation, many of which have focused on the blob velocity scaling. A small selection of these are presented here.

#### From Garcia *et al* (2005) [23]

It is shown that the radial blob velocity is

$$v_x = c_s\sqrt{\frac{2\sigma}{R}\frac{\Delta\theta}{\Theta}} \tag{3.14}$$

where $\Delta\theta$ is the thermodynamic variable (for example the particle density or the temperature) of the blob and $\Theta$ is the thermodynamic variable of the background plasma. Simulations are run with different values for the Rayleigh number $Ra \propto \frac{\Delta\theta}{\Theta}$, effectively investigating the effect of the initial blob amplitude, $n_b/n_0$.

It is shown that the blob reaches a higher maximum velocity $v_{x,max}$ for higher initial amplitudes, although this dependency is shown to only be true up until a certain amplitude. The density amplitude of the blob is shown to decrease with time as the blob propagates radially outwards.

#### From Kube *et al* (2016) [40]

A two-fluid gyrofluid model is used and two expressions for the maximum radial blob velocity are found:

$$v_{x,max} \approx c_s\mathcal{R}\sqrt{\frac{\sigma}{R}\frac{n_b}{n_0}} \tag{3.15}$$

$$v_{x,max} \approx c_s\frac{\mathcal{P}}{4}\frac{n_b}{n_0} \tag{3.16}$$

where the first is true up to a certain $n_b/\sigma$, after this the second one is the better fit. The critical ratio is found to be

$$\frac{n_b}{n_0}\frac{R}{\sigma} = \left(\frac{4*\mathcal{R}}{\mathcal{P}}\right)^2 \tag{3.17}$$

$$\frac{n_n/n_0}{\sigma} \approx 46R \tag{3.18}$$

when $\mathcal{R} = 0.85$ and $\mathcal{P} = 0.5$, as found through simulations.

The maximum velocity reached increases with increasing blob amplitude, and the time to reach maximum velocity decreases as the blob amplitude is increased. It is also shown that blobs of larger size $\sigma$ reach their maximum velocity later than than smaller blobs, although the maximum velocity reached by larger blobs is higher than that of smaller blobs.

**From Wiesenberger *et al* (2017) [64]**

Here, an attempt to unify the square root velocity scaling (3.15) and the linear velocity scaling (3.16) is made, resulting in a new expression for the maximum radial velocity

$$v_{x,max} = c_s \frac{\mathcal{S}^2}{\mathcal{Q}} \frac{\sigma}{R} \left( \sqrt{1 + \left( \frac{\mathcal{Q}}{\mathcal{R}} \right)^2 \frac{n_b}{n_0} \frac{R}{\sigma}} - 1 \right) \tag{3.19}$$

where $\mathcal{Q} = 0.32$ and $\mathcal{S} = 0.85$ are determined through numerical simulations. As $n_b$ is increased, the dependence will change from $\sigma$ to $n_b$. For a blob size $\sigma = 10^{-2}R$, the velocity scaling shifts from linear to square root around $n_b/n_0 \approx 0.5$ and for a blob size $\sigma = 10^{-2}R$, the transition happens for $n_b/n_0 \approx 0.05$.

The numerical studies show an agreement about the maximum radial blob velocity, which is said to scale with either the blob density or the square root of the blob density, depending on the ratio between the initial blob density and the initial cross-field size of the blob. The radial blob velocity is found in (3.14) to depend on the cross-field size of the blob, but with the opposite relation of what was found by the theoretical studies.

### 3.5.4 FLR Effects

In most gyrofluid simulations of plasma blobs, the ions are assumed to be cold ($t_i = 0$). Ions with no thermal velocity have a gyroradius $\rho_i = 0$. If the ions are warm, however, they will have a gyroradius $\rho_i \neq 0$. This is commonly referred to as a *finite Larmor radius* (FLR) effect, and as the ion temperature of the SOL can be significantly higher than the electron temperature, this is highly relevant for blob simulations. A fluid model taking these kinetic effects into consideration was developed by Dorland and Hammett in 1993 [16], and it was first used for blob simulations in 2008 [35].

In kinetic models, the gyroradius of the ions is inherently a part of the dynamics, and no correctional terms are needed to include FLR effects. As these effects will be a part of the simulations carried out for this project, it is necessary to consider some fluid simulations with FLR corrections to have a complete picture of the expected blob dynamics. Some of the dynamics observed in more recent simulations taking FLR effects into account, published by Madsen *et al* [44], Wiesenberger *et al* [63] and Held *et al* [32], are therefore presented here. On the following page these works will be referred to as Madsen, Wiesenberger and Held, respectively.

### Cold Ions

All three papers present cold ion blobs that travel purely in the radial direction, develop a mushroom shape that is symmetric around $y = y(t = 0)$ and eventually dissolve.

The maximum velocity of the cold ion blobs is shown by Wiesenberger to increase as the initial blob amplitude $n_b$ is increased. By Wiesenberger and Held the cold ion blob is shown to propagate further in the radial direction for higher blob amplitudes. However, the blobs do not reach an equilibrium within the simulations and are still propagating with a constant $v_x$, meaning that depending on the lifetime of the blobs, the high-amplitude blobs could dissolve and be taken over by the lower amplitude blobs.

A blob of cold ions will have a maximum electron density amplitude that decreases significantly already in the early stages of the simulation, according to Madsen.

### Warm Ions

The electron density plots presented by all three papers show that the blob behaves very differently for warm ions. Instead of the symmetric mushroom shape, the blob is shown to keep more of its initial blob-like shape, and it does not diffuse within the timeframe of the simulation. In addition it no longer has a purely radial propagation; there is now a significant displacement in the negative poloidal direction as well.

The total displacement of the warm ion blobs is shown to be larger as $n_b$ is increased by Held, and according to Wiesenberger, a larger cross-field blob size $\sigma$ leads to a larger radial displacement, as for cold ion blobs.

Both Held and Wiesenberger show that the initial radial displacement is larger for warm ion blobs than for cold ion blobs, indicating that the radial propagation by $\mathbf{E} \times \mathbf{B}$ drift starts earlier and accelerates faster when the ions are warm. In the study by Madsen this seems to not be the case: for a blob with a large cross-field size ($\sigma = 20\rho_s$) the initial displacement along $x$ appears to be identical for all ion temperatures, while the warmer ions have a slower initial propagation along $x$ than the colder ones for a blob of small cross-field size ($\sigma = 5\rho_s$). The warm blobs with higher $n_b$ appear to reach a higher maximum radial velocity $v_x$ earlier than those with smaller $n_b$ in the paper published by Wiesenberger, and also decelerate faster, while Held show that they reach a higher $v_x$, but later.

The poloidal displacement is shown by Madsen and Held to be larger in the negative direction as the ion temperature is increased. In both cases, however, the poloidal component of the velocity, $v_y$, is shown to decrease after an initial period of increasing, stop, and change signs, with the blob then propagating in the positive poloidal direction, and it is in some cases found by Madsen to even cross the starting position $y(t = 0)$ and ending up with a net poloidal displacement that is positive. This fits the theoretical description given in (3.10) [14].

The maximum electron density of the warm ion blobs have a slow, linear decrease over time, with the density of blobs of larger $n_b$ decreasing more than those with smaller initial amplitude, according to both Wiesenberger and Madsen.

# Chapter 4

# Numerical Methods

While fluid simulations give a good understanding of the macroscopic properties of plasmas, and efforts have been made to include kinetic effect in gyrofluid models, they are still based on assumptions and approximations. A particle simulation does not depend on assumptions to include the kinetic effects. There are several ways to simulate plasmas on the fundamental, kinetic level, and for this thesis particle-in-cell simulations were chosen as they provide an efficient way to simulate interactions between a high number of particles, which is necessary to study the dense plasma of the SOL.

In this chapter, an overview of the Particle-In-Cell (PIC) method is given, the main ideas are explained, and the important calculations during one timestep are presented. For a full introduction to PIC simulations, the reader is referred to textbooks on the topic, like *Plasma Physics via Computer Simulations* by Birdsall and Langdon [3] and *Computer Simulation Using Particles* by Hockney and Eastwood [33].

## 4.1 Particle-In-Cell Simulations

In particle simulations, the force of each particle on all the other particles is what drives the simulation, and calculating the force of each on all the others would give a nested loop like the one shown below, where `F_ij` is the force on particle `i` from particle `j`, resulting in a force `-F_ij` from particle `i` on particle `j`:

```
for i = 1 to n_p - 1:
    for j = i+1 to n_p:
        F_ij = k_e*q_i*q_j/r_ij^2
        F_i += F_ij
        F_j -= F_ij
```

This results in a number of operations of order $\mathcal{O}(n_p^2)$, where $n_p$ is the number of particles in the simulation [33].

Instead of particle-particle (PP) simulations like the one described above, *particle-mesh* (PM) simulations were suggested in the late 1950s [60]. In PM simulations, the effect of the particles are evaluated on a grid where the forces are calculated and interpolated back to the particles in order to move them.

Hockney and Eastwood show that PM methods have approximately $20n_p + 5N_g\log(N_g)$ operations, with $N_g$ being the number of grid points each dimension,

which in the 1980's gave an improvement in computational time from 1 day (!) for the PP method to 4.5 seconds for the PM method when simulating $n_p = 10^5$ particles in a $(N_g \times N_g \times N_g)$ mesh with $N_g = 32$ grid points [33]. Of course, computers have gotten significantly faster since then, but this is still illustrates the benefit of choosing PM methods over PP methods.

The particle-in-cell (PIC) method is a particle-mesh method where the physical particles are bundled together into superparticles, thus reducing not only the number of interactions by interpolating to a grid, but also the number of computational particles.

### 4.1.1 The Superparticle

If several particles are close to each other in space and have the same velocity at a timestep $t_n$, they will be affected by their surroundings almost identically, and therefore also be close to each other at the next timestep, $t_{n+1} = t_n + \Delta t$. This is taken advantage of in PIC simulations, where physical particles that are close to each other in the phase space described by Vlasov's equation (2.17) are bundled together in particle clouds that will move together through the simulation. These clouds are simulated computational particles and are often called *superparticles*, as one computational particle contains many physical particles. This drastically reduces the number of particles necessary to compute.

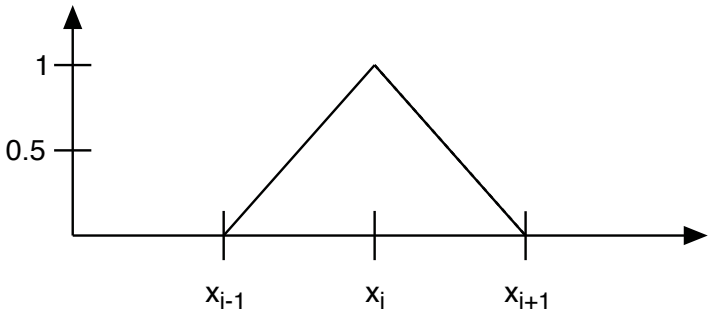


**Figure 4.1:** The spatial shape of the superparticle in one dimension.

The superparticles have a shape in the velocity dimension and in the spatial dimension. The velocity shape is often chosen to be a Dirac's delta, meaning that only particles of identical velocities will be bundled together, securing that they remain close to each other in phase space as time progresses. For the spatial shape a *b-spline* is commonly used, and for PIC simulations, the spline is of first order, shown in figure 4.1.

The superparticle is a superposition of physical particles, and the superparticle distribution in phase space is the sum of the distribution of physical particles in the superparticle:

$$f_s(\mathbf{x}, \mathbf{v}, t) = \sum_p f_p(\mathbf{x}, \mathbf{v}, t). \tag{4.1}$$

With the superparticles taking the place of physical particles, Vlasov's equation (2.17) becomes

$$\frac{\partial f_s(\mathbf{x}, \mathbf{v}, t)}{\partial t} + \mathbf{v}\frac{\partial f_s(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{x}} + \frac{q_s}{m_s}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \frac{\partial f_s(\mathbf{x}, \mathbf{v}, t)}{\partial \mathbf{v}} = 0 \tag{4.2}$$

where both the superparticle mass, $m_s$, and the charge of the superparticle, $q_s$, are summed over the number of physical particles, $p$, contained in the superparticle, keeping the mass-to-charge ratio constant, $m_s/q_s = m_p/q_p$.

### 4.1.2 The Field Grid

In PIC simulations the superparticles are positioned continuously on a discrete grid. The force $\mathbf{F}_p$, velocity $\mathbf{v}_p$, and position $\mathbf{x}_p$, are known for each particle and are therefore subscripted $p$. The field quantities are on the other hand not represented at every single particle; the charge density, $\rho_g$, and the current density, $J_g$, are weighted onto the discrete grid points from the particles values, and from these the electric field, $\mathbf{E}_g$, and the magnetic field, $\mathbf{B}_g$, are found. As these quantities are defined only on the grid points, they are subscripted $g$.

## 4.2 The PIC Cycle

The process of evaluating grid quantities from particle quantities, calculating the electric and magnetic field and weighting the resulting force back to the particles to move them is repeated in every timestep of the PIC simulation. The four main components of a PIC cycle are:

- Moving the particles according to the force on each particle:
  $\mathbf{F}_p \rightarrow \mathbf{v}_p \rightarrow \mathbf{x}_p$

- Weighting the effect of the particles from their positions to the grid to find the charge density and current density at the grid points:
  $\mathbf{x}_p, \mathbf{v}_p \rightarrow \rho_g, \mathbf{J}_g$

- Solving Maxwell's equations to find the electric and magnetic field from the charge and current densities:
  $\rho_g, \mathbf{J}_g \rightarrow \mathbf{E}_g, \mathbf{B}_g$

- Weighting the force created by the electric and magnetic fields onto each particle:
  $\mathbf{E}_g, \mathbf{B}_g \rightarrow \mathbf{F}_p$

These components are shown in figure 4.2, and explained in some detail here, as they create the basis for the PIC code used to conduct the simulations of this thesis.

### 4.2.1 The Particle Mover

It is required that the first moments of the Vlasov equation are exactly satisfied by the shapes chosen for the computational particles. By taking the zero order moment (2.18) and the first order spatial (2.19) and velocity moments (2.20), three equations

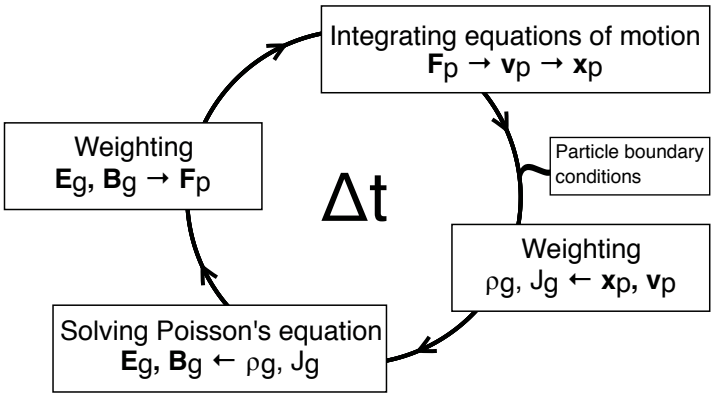


**Figure 4.2:** The four steps that make up one timestep in a PIC simulation.

for the evolution of the plasma are found [41]:

$$\frac{dN_p}{dt} = 0 \tag{4.3}$$

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{v}_p \tag{4.4}$$

$$\frac{d\mathbf{v}_p}{dt} = \frac{q}{m}(\mathbf{E}_p + \mathbf{v}_p \times \mathbf{B}_p) \tag{4.5}$$

Equation 4.3 gives the requirement that the number of physical particles per computational particle must be conserved, while the other two equations are the equations of motion.

To solve the equations of motion it is common in PIC codes to use a *leapfrog discretization* where the position is evaluated at full time steps and the velocity at half time steps [33], giving the following discretized equations to solve in 1D:

$$\frac{x^{t+\Delta t} - x^t}{\Delta t} = v^{t+\Delta t/2} \tag{4.6}$$

$$\frac{v^{t+\Delta t/2} - v^{t-\Delta t/2}}{\Delta t} = \frac{q}{m}\left(E_p + \left(\frac{v^{t+\Delta t/2} - v^{t-\Delta t/2}}{2}\right) \times B_p\right) \tag{4.7}$$

In the case where there is no magnetic field, (4.7) can be solved directly. If there is a magnetic field, (4.7) can be solved implicitly as a linear system. Another common way of solving it is the *Boris method*, where the acceleration from the electric field and the rotation caused by the magnetic field are treated separately by first applying half the acceleration, then the rotation, and lastly the last half of the acceleration. This method is explained in greater detail in Section 5.2.2 as it was implemented in 2D for this project. The Boris method is accurate if the angle of the rotation is small [3].

### 4.2.2 Weighting From Particles To Grid

The charge and current densities are computed as averages over the superparticles:

$$\rho_g(\mathbf{x}, t) = \sum_s q_s \int f_s(\mathbf{x}, \mathbf{v}, t) d\mathbf{v} \tag{4.8}$$

$$\mathbf{J}_g(\mathbf{x}, t) = \sum_s q_s \int \mathbf{v} f_s(\mathbf{x}, \mathbf{v}, t) d\mathbf{v} \tag{4.9}$$

where the integrals depend on the shapes of the superparticles in $\mathbf{x}$ and $\mathbf{v}$.

In the code, the charge density at the grid points is evaluated by weighting the charge of the superparticles from their positions, $\mathbf{x}_p \rightarrow \rho_g$, and their velocities are weighted to find the current density, $\mathbf{v}_p \rightarrow \mathbf{J}_g$. The weighting is done through interpolation from the position to the nearest grid points. For one dimensional 1st order interpolation, the charge density is weighted to the two closest grid points in the following way:

```
for i =0 to n_p:
    x_p = x_p(i)/dx
    x_g = floor(x_p)
    rho(x_g) = rho(x_g) + q(1-(x_p-x_g))
    rho(x_g + 1) = rho(x_g + 1) + q(1-(x_g + 1 - x_p))
                 = rho(x_g + 1) + q(x_p-x_g)
```

where `x_g` is the closest grid point below the particle and `x_g + 1` is the grid point above. This example can easily be extended to two or three dimensions.

### 4.2.3 The Field Solver

Through Maxwell's equations (4.10-4.13), the electric and magnetic field caused by the charge density $\rho_g$ and the current density $\mathbf{J}_g$, can be found.

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{4.10}$$

$$\nabla \times \mathbf{B} = \mu_0 \left( \mathbf{J} + \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} \right) \tag{4.11}$$

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0} \tag{4.12}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{4.13}$$

In an electrostatic approach there is no change (or a very slow change) in the electric and magnetic fields over time, simplifying (4.11) to

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} \tag{4.14}$$

and eliminating (4.10):

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \approx 0 \tag{4.15}$$

This gives an electric field

$$\mathbf{E} = -\nabla \phi, \tag{4.16}$$

which combined with (4.12) gives Poisson's equation:

$$\nabla^2 \phi = -\frac{\rho}{\epsilon_0}. \tag{4.17}$$

Using finite difference, the electric potential can be found through the charge density, $\rho$, which is known at this point, changing Poisson's equation to

$$\frac{\phi^{n+1} - 2\phi^n + \phi^{n-1}}{(\Delta x)^2} = -\frac{\rho}{\epsilon_0}, \tag{4.18}$$

for the 1D case where $n$ is the grid point. This in turn gives the electric field

$$\mathbf{E} = \frac{\phi^{n+1} - \phi^{n-1}}{2\Delta x}. \tag{4.19}$$

(4.18) can be written in matrix form

$$A\phi = -\frac{\rho}{\epsilon_0}(\Delta x)^2, \tag{4.20}$$

creating a linear system that can be solved by any suitable method, including Jacobi, Gauss-Seidel, etc. Other possible solvers includes solvers using fast Fourier transform (FFT) to solve Poisson's equation, like the spectral method. In Chapter 6 of *Computer Simulation Using Particles* [33], an overview of different methods is given, and their strengths and weaknesses are thoroughly explained.

### 4.2.4 Weighting From Grid To Particles

The particles will experience a force from the electric and magnetic field, as given by the Lorentz force

$$\mathbf{F}_p = q(\mathbf{E}_p + \mathbf{v}_p \times \mathbf{B}_p). \tag{4.21}$$

The force is evaluated at every particle position by interpolating $\mathbf{E}_g$ and $\mathbf{B}_g$ back from the grid to the particles. This is done in the same way as the interpolation to the grid shown in Section 4.2.2, adding to the force experienced by each particle. The particles will be accelerated by $\mathbf{F}_p$ in the particle mover, thus concluding one PIC cycle.

## 4.3 Stability Criteria

Through the discretization of the equations to a numerical method and the introduction of a discrete grid, the possibility for numerical instabilities arise. There is a need to decide some stability criteria which should be respected to avoid simulations that are dominated by numerical artifacts rather than actual physics.

### 4.3.1 Time Resolution

The leapfrog discretization (4.6, 4.7) used to find a numerical solution to the equations of motion has an error that vanishes as $\Delta t \to 0$, but becomes significant if $\Delta t$ is too large [3]. This sets a limitation on the size of $\Delta t$, with the stability criteria

$$\Delta t < 0.3\omega_{pe}^{-1} \tag{4.22}$$

suggested by Birdsall and Langdon [3], based on the error when integrating a harmonic oscillator. The *phase error* accumulates over time and after $N$ timesteps it is given by:

$$e_{phase} \approx \frac{N(\omega_{pe}\Delta t)^3}{24} \tag{4.23}$$

### 4.3.2 Space Resolution

The introduction of a finite grid where particles move continuously brings with it a loss of information that allows for numerical instabilities through aliasing. An example of aliasing is shown in figure 4.3 where the grid size has been set to $\Delta x = 7/8\lambda_{De}$. When the values are only known on these grid points, another solution for the Debye length $\lambda'_{De} = 8\Delta x$, meaning that the Debye length has become seven times as large as it initially was. To avoid aliasing, a constraint on the grid spacing is introduced:

$$\Delta x < k\lambda_{De}. \tag{4.24}$$

Here $k$ is a constant that depends on the exact method used, but is often reported as $k \approx \pi$ [41], although it has also been found that $\lambda_{De} \sim \Delta x$ is sufficient [49]. If the Debye length is not resolved by the grid and grows numerically through aliasing, the particle temperature will also increase (see (2.5)). The plasma will experience numerical heating increasing the temperature until $\lambda_{De}$ is resolved. The system can stabilize once the Debye length is large enough for the condition in (4.24) to be resolved, but it can also lead to numerical instabilities throughout the simulation [41].

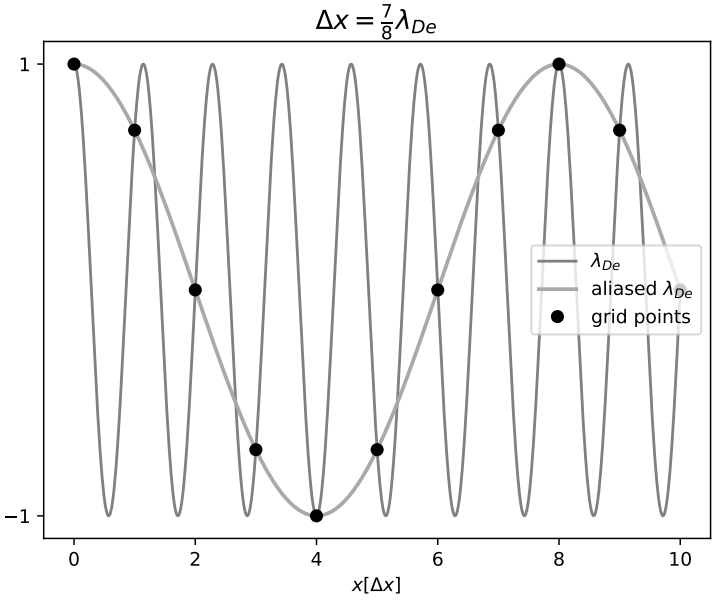


**Figure 4.3:** Example of aliasing occurring when the grid size $\Delta x$ is too small to properly resolve the Debye length $\lambda_{De}$.

### 4.3.3 The CFL Criteria

The Courant-Friedrichs-Lewis (CFL) stability criteria connects $\Delta t$ and $\Delta x$ through the characteristic velocity $c_s$:

$$\frac{\Delta x}{\Delta t} > c_s \tag{4.25}$$

In the case of electromagnetic wave propagation, $c_s$ must be substituted for the speed of light, $c = \sqrt{1/(\mu_0\epsilon_0)}$. The CFL criteria can be interpreted to say that the time step must give the information sufficient time to travel through the cell [59]. It

was shown by Courant, Friedrichs and Lewis that for wave propagation, the solution will not converge if the CFL critera is not satisfied [12].

### 4.3.4 Superparticle Resolution

Although Vlasov is still satisfied for any number of simulation particles, fewer simulation particles will add more numerical noise and increase the graininess of the results. A plasma can only be considered collisionless if the number of particles per Debye cube,

$$N_D = n\lambda_{De}^3, \tag{4.26}$$

is very large, $N_D \gg 1$ [51]. This condition should be resolved for the superparticles as well.

One way to reduce the noise introduced by the superparticles is to let the phase space shape of the superparticles vary from particle to particle, and also to change in time [10]. These variable superparticles are referred to as *blobs*, and have nothing to do with the plasma blobs that are studied in this thesis. For example, the superparticles can be more localized in the phase space in areas where a higher resolution is necessary, and they can be bigger in areas where more noise is acceptable.

# Chapter 5

# Implementation and Setup

In this thesis, a particle-in-cell code was used to simulate the dynamics of a plasma blob in the SOL of a tokamak. New functionality was added by the author to an existing PIC code to allow for the specific conditions needed, and parameters were chosen with stability criteria and simulation speed in mind, while at the same time aiming to stay close to real values. In this chapter the existing PIC code is outlined, implementation of new functionality is documented and the parameters used in simulations are listed and explained.

## 5.1 PINC

PINC is an open-source PIC code developed by Sigvald Marholm at the University of Oslo with contributions from students, among which the multigrid solver written by Gullik Vetvik Killie is one of the largest [36]. The code is written in C, it is parallelized and multidimensional, and extra care was taken to make the code efficient, which makes large scale PIC simulations possible.

PINC is built up around the basic PIC cycle explained in Chapter 4, and does not stand out from other PIC codes in that respect. However, the care that was taken in every part of the code to keep it as efficient as possible without sacrificing precision makes it unique. Some of the noteworthy features of PINC are described here, and readers who want to learn more and get access to the code can contact relevant researchers at The University of Oslo.

### 5.1.1 Subdomains

PINC is fully parallelized, meaning that the grids as well as the particles can be distributed between nodes. The simulation domain can be split into several subdomains, and particles that are distributed on the grid are placed in their respective subdomain. Each subdomain is surrounded by a layer of ghost cells that contain the information of the first layer of grids of the subdomains around it, as illustrated in figure 5.1. This means that a particle on the edge of one subdomain will still be able to experience the forces from the neighbouring domain, and enables particle migration between the domains. The information is exchanged between subdomains by a Message Passing Interface (MPI).

The size of the domain is given through a physical grid size (in meters) and a number of grids points in each dimension. If more than one subdomain is used, the

grid points are distributed between these in the input file. It is worth mentioning here that the grid is non-dimensionalized by the grid size, $dx$. This means that in PINC, each cell is has a length of 1, eliminating several multiplications and divisions throughout the simulation. For example, the area and volume of each cell is 1, and finding the closest grid point $x_g$ to a particle at position $x$ becomes a simple case of

```
x_g = int(x)
```

where no division is necessary. This is speeds up the many interpolations during one PIC cycle.

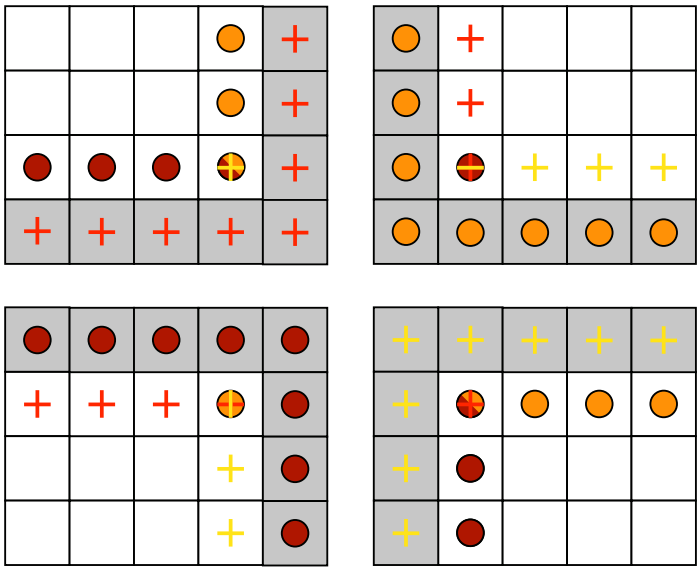

**Figure 5.1:** Illustration of the ghost cells used to exchange information between subdomains. The ghost cells (gray) of one subdomain copy the information of the closest "real" cells (white) from the neighbouring domain. The paired cells are marked with markers of same shape and color.

### 5.1.2 Particle Distribution

The particles are given an initial position, either randomly or uniformly distributed, and an initial velocity. The initial velocity is based on input parameters, where both a drift velocity and a thermal velocity can be set. The thermal velocities distributed to particles are based on a Maxwellian distribution from the thermal velocity given in the input file.

### 5.1.3 The Particle Mover

There are several particle movers to choose from in PINC, including an n-dimensional solver for simulations with no external magnetic field and a 3D Boris solver for a homogeneous external magnetic field. A 2D Boris solver for inhomogeneous external magnetic fields was implemented by the author for this thesis, and is explained in greater detail in section 5.2.2.

### 5.1.4 The Field Solver

There are several options for solving Poisson's equation in PINC. Two iterative solvers, the Jacobi Method and Gauss-Seidel RB, are implemented as options for the multigrid solver. The multigrid solver resolves the field on grids of varying coarseness, with local errors being reduced on the fine grids and distant errors on the coarser grids [36]. This means that a smaller number of iterations is needed to reduce the error of the solution. In addition to the iterative solvers there is a separate Spectral Solver which is based on fast Fourier transform (FFT) [6].

The multigrid solver contains three different choices for boundary conditions: periodic, Dirichlet and von Neumann. The boundary conditions are enforced through the ghost cells, and the periodic boundaries work in the same way as the boundaries between the subdomains. In the case of Dirichlet boundaries, the ghost cells are set to a given value. This can simply be a constant, or it can be a function varying in space and time. With von Neumann boundary conditions, the derivative on the boundary is set, meaning that the ghost cell is set to the value of the cell next to it, $\pm A$, where $A$ is decided by the desired gradient, $\nabla\phi$. Examples of this can be seen in section 5.3.1.

## 5.2 Plasma Fusion in PINC

To be able to simulate the conditions of the SOL, and specifically plasma blobs, in PINC, the development of new functionality was required. The main things missing were

- setting a non-uniform density distribution to seed a blob
- a particle mover for inhomogeneous magnetic fields
- non-periodic particle boundary conditions
- center of mass tracking

These functions were developed for this thesis by the author and incorporated into pre-existing modules where they belonged. They code is found in Appendix A.

### 5.2.1 Plasma Density Distribution

In order to seed a blob, it is necessary to have a non-uniform initial particle distribution. For simplicity this sections describes a 1D case, although the function developed for PINC uses two dimensions.

The enhanced density of the plasma blob is defined as a Gaussian curve, giving a total density

$$n = n_0 + n_b \exp\left(-\frac{(x - x_0)^2}{2\sigma_x^2}\right) \tag{5.1}$$

where $n_0$ is the density of the background plasma, $n_b$ is the peak value of the blob density, $x_0$ defines the center of the blob, and $\sigma_x$ is the size of the blob in the $x$ direction. This function is plotted in figure 5.2, where all the parameters are shown.

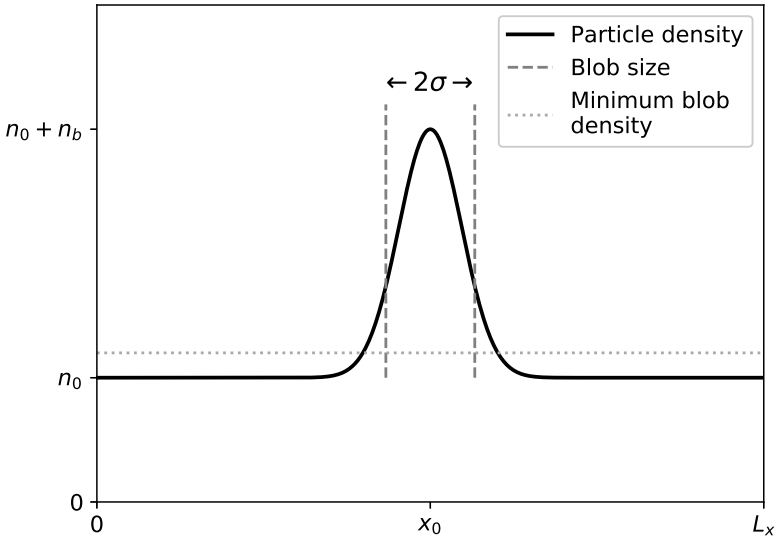


**Figure 5.2:** Particle density distribution for for $n_b = 2n_0$. The diameter of the blob is shown, as well as the definition of blob used for COM tracking.

An average particle density of $n_{sim}$ particles per cell is set, giving a total number of particles $N$. In the initialization, the code iterates through all the particles and places them one at a time. To get a non-uniform distribution, Monte Carlo rejection sampling is used [7]. The idea behind rejection sampling is to draw uniformly distributed values and shape them after a distribution of choice by *rejecting* the particles that are outside of the area under the density distribution curve shown in figure 5.2.

For particle $i$ a random position $x_i$ between 0 and $L$ is drawn. To decide if this position should be rejected or accepted, another random variable, $k_i$, between 0 and $n_0 + n_b$ (the maximum particle density) is drawn and compared to the density distribution function evaluated at $x_i$. If $f(x_i) < k_i$ the position is rejected and a new $x_i$ is drawn. This is repeated until $f(x_i) > k_i$ and the position is accepted. The program then goes on to the next particle and gives this a position. In code this becomes

```
while f(x, nb) < k:
    x = L*random(0,1)
    k = n_max*random(0,1)
pos = x
```

where $f$ could be any function, but in this case is the Gaussian function found in (5.1), `nb` is the ratio $n_b/n_0$ and `n_max` is the maximum possible particle density, $n_0 + n_b$, and `pos` is the accepted position.

For example, a position at the center of the blob will always be accepted as $f(x_0) = n_0 + n_b = k_{max}$, meaning that the position $x = x_0$ will be kept for any $k$, because $k \leq k_{max}$ and $f(x_0) = k_{max}$. At the edges, where $f(x) = n_0$, a position will only be kept if $k < n_0$ is drawn.

As positions are rejected, new ones have to be drawn, which means that the number of positions drawn is greater than the number of particles, which makes it

slower than a uniform particle distribution. However, the seeding is only done once at the beginning of each simulation, and it provides an convenient way of controlling the ratio between the blob density amplitude and the background density, $n_b/n_0$, which is why it was chosen.

### 5.2.2 Inhomogeneous Magnetic Field

A 2D Boris Particle Mover was implemented as suggested in Chapter 4 of *Plasma Physics via Computer Simulations* by Birdsall and Langdon [3] for solving the numerical equations of motion:

$$\frac{\mathbf{x}^{t+\Delta t} - \mathbf{x}^{t}}{\Delta t} = \mathbf{v}^{t+\Delta t/2} \tag{5.2}$$

$$\frac{\mathbf{v}^{t+\Delta t/2} - \mathbf{v}^{t-\Delta t/2}}{\Delta t} = \frac{q}{m}\left(\mathbf{E_p} + \left(\frac{\mathbf{v}^{t+\Delta t/2} - \mathbf{v}^{t-\Delta t/2}}{2}\right) \times \mathbf{B_p}\right) \tag{5.3}$$

In 5.3, the electric field is responsible for the change in the magnitude of the velocity of the particle, while the magnetic field changes the direction. In the *Boris method* this is taken advantage of by splitting up the effect from the electric and the magnetic fields and using rotational parameters to move the particles. If there is an external or internal $\mathbf{B}$ field, (5.2) is used as is while (5.3) is decomposed into three steps: First add half of the acceleration due to $\mathbf{E}$ (5.4), then rotate the particle according to (5.5) for the $x$ component, $v_x$, and (5.6) for the $y$ component, $v_y$, to account for the effect of $\mathbf{B}$, and lastly add the second half of the acceleration due to $\mathbf{E}$ (5.7).

$$\mathbf{v}^{-} = \mathbf{v}^{t-\Delta t/2} + \frac{q}{m}\mathbf{E_p}\frac{\Delta t}{2} \tag{5.4}$$

$$v_x^{+} = cv_x^{-} + sv_y^{-} \tag{5.5}$$

$$v_y^{+} = -sv_x^{-} + cv_y^{-} \tag{5.6}$$

$$\mathbf{v}^{t+\Delta t/2} = \mathbf{v}^{+} + \frac{q}{m}\mathbf{E_p}\frac{\Delta t}{2} \tag{5.7}$$

where $c$ and $s$ are the rotational parameters

$$s \equiv -\sin(\theta) \tag{5.8}$$

$$c \equiv \cos(\theta) \tag{5.9}$$

where $\theta$ is the angle of rotation defined by

$$\tan\left(\frac{\theta}{2}\right) = -\frac{qB}{m}\frac{\Delta t}{2} \tag{5.10}$$

by defining $t \equiv -\tan(\theta/2)$, the equations 5.8 and 5.9 can be redefined to

$$s = \frac{2t}{1+t^2} \tag{5.11}$$

$$c = \frac{1-t^2}{1+t^2} \tag{5.12}$$

The magnetic field to solve for in this case has a gradient in the $x$ direction

$$\mathbf{B} = \{0, 0, B(x)\}, \tag{5.13}$$

meaning that it can either be evaluated at each timestep for each particle $p$ with the position $\mathbf{x}_p$, or it can be treated similarly to the electric field by evaluating the magnetic field on grid points only and then weighting it to the particles when needed. By evaluating the field only on grid points it would only need to be calculated once, although some precision would be lost with the interpolation, and if interpolating is more or less computationally expensive than solving the exact field at the position of each particle depends on the expression for $B(x)$ and the interpolation method.

### 5.2.3 Particle Boundary Conditions

For the electric potential, $\phi$, periodic, Dirichlet and von Neumann boundary conditions were already implemented [36]. Making use of the non-periodic boundary conditions for the potential requires there to be matching boundary conditions for the particles. As only periodic particle boundaries were pre-existing, absorbing and reflecting boundary conditions were implemented for this project.

The PINC code is parallelized, and the simulation domain can be divided into several subdomains for parallel computing. If more than one subdomain is used, the boundaries of the domain will only exist in the lowermost and uppermost subdomain(s); therefore only the lowest subdomain(s) in a dimension is checked for particles crossing the lower boundary of that dimension, and only the highest subdomain(s) is checked for particles crossing the upper boundary of that dimension. This saves having to iterate through all the particles in subdomains where it is impossible for the particle to be outside of the domain boundaries.

#### Reflecting boundaries

Let the lower boundary be placed at $b_0$ and the upper boundary at $b_L$. If a particle crosses the lower boundary between two time steps and gets a position $x < b_0$, it should have been reflected at the boundary. This is done by changing the position to $x_{reflected} = 2b_0 - x$. Similarly, if a particle crosses the upper boundary and gets a position $x > b_L$, it is moved the reflected position $x_{reflected} = 2b_L - x$. In this way the particle is moved back inside the domain if it tries to cross the boundaries.

The velocity is changed in the same way for all boundaries when the particle is reflected, by setting $v_{reflected} = -v$. If the simulation has more than one dimension, only the velocity component in the dimension where the particle hits a boundary is changed; the other components are kept the same.

If reflecting boundaries are used, it can happen that a particle is reflected by a boundary in one dimension and then crosses a boundary in another dimension, if it is close to the corner of the domain. To handle this the particle boundary function should be run twice in a row when one or more boundaries are reflective.

#### Absorbing boundaries

If a boundary $b_L$ is set to be absorbing, a particle with position $x > b_L$ will be deleted from the list of particles. This is done using a function which overwrites the information about the particle that should be deleted with the information about

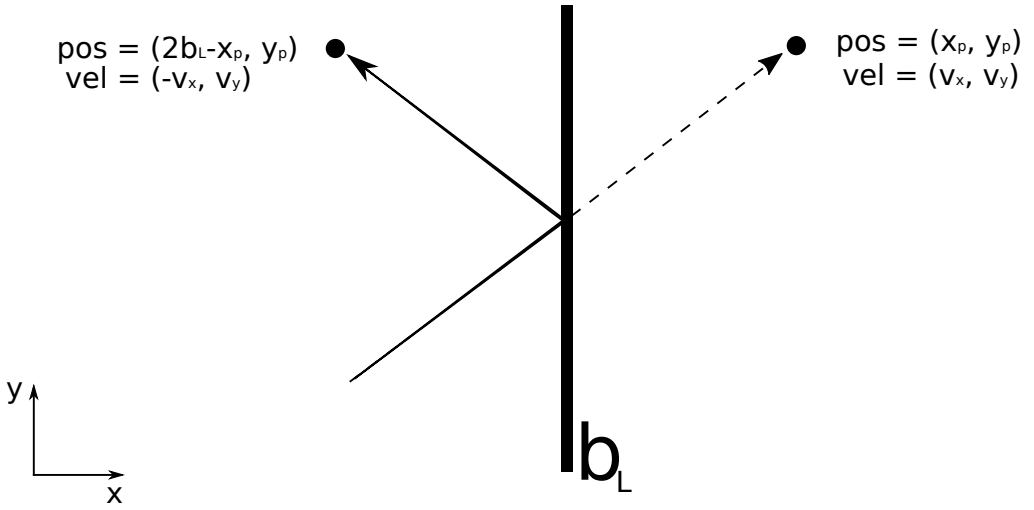


**Figure 5.3:** Example illustration of a particle crossing the upper boundary, $b_L$. Initially it gets the position $(x_p,\ y_p)$, where $x_p > b_L$. This is detected by the particle boundary function and the particle is then moved to the reflected position and the $x$ component of the velocity is changed.

the *last particle in the list* and then removing the last particle. It can be thought of as the last particle being moved to where the deleted particle used to be. This way a "hole" in the list is avoided and the list becomes one element shorter without having to move all the particles.

If the last particle of the list, with particle number $i = N$, is outside of the boundary at the same time step as another particle, with particle number $i = k$, the particle $k$ will be deleted first, and the particle that used to be number $i = N$ will now be number $i = k$. The iteration will move on to $i = k + 1$ and the particle that used to be number $N$ will not be deleted as it should be. For this reason the particle boundary function should be run twice also for absorbing boundaries.

### 5.2.4 Tracking the Blob

To track the blob, it is first necessary to define it. It is common to define the blob as the *electron blob*, and count only the electron density. In [31] the blob is defined as the particles in the areas where the electron density is higher than the background density plus 10% of the initial blob density amplitude,

$$n_e > n_0 + 0.1n_b. \tag{5.14}$$

The additional 10% help filter out some of the variations in the background plasma and makes the location of the blob more accurate. The minimum density line for a blob is shown in figure 5.2.

To see how the blob moves as a whole, the blob *center of mass* (COM) is used. While the blob propagates towards the edges of the domain, the COM of *all* the particles will remain relatively close to the center of the box (depending on the ratio between the background density and the maximum blob density). As it is the COM of the *blob* that is of interest, it is necessary to extract the part of the plasma which contains the blob and find the center of mass of these particles. The COM is therefore considered the electron center of mass where the blob density condition in 5.14 is fulfilled.

The particle density is not usually evaluated in PIC codes, but the charge density, $\rho = qn$, is evaluated at grid points. Since the charge density is a constant different from the particle density, it can be used for the purposes of finding $\mathbf{R}_c$.

In [32] the center of mass is defined as

$$\mathbf{R}_c \equiv \frac{1}{M}\int \mathbf{x}(n_e - n_{e0})d\mathbf{x} \tag{5.15}$$

$$M \equiv \int (n_e - n_{e0})d\mathbf{x} \tag{5.16}$$

where $n_{e0}$ is the initial background electron density and $n_e$ is the electron density at the current timestep.

For this thesis, the center of mass at a given timestep has been defined as

$$\mathbf{R}_c \equiv \frac{1}{M}\int \mathbf{x}(\rho_e(\mathbf{x}) - \rho_{e0})d\mathbf{x} \tag{5.17}$$

$$M \equiv \int (\rho_e(\mathbf{x}) - \rho_{e0})d\mathbf{x} \tag{5.18}$$

$$\tag{5.19}$$

where $\rho_e(\mathbf{x})$ is the electron charge density at position $\mathbf{x}$ and $\rho_{e0} = q(n_0 + 0.1n_b)$. For coding purposes, this becomes

$$\mathbf{R}_c = \frac{\sum_i V_i(\rho_e(\mathbf{x}_i) - \rho_{e0})\mathbf{x_i}}{\sum_i V_i(\rho_e(\mathbf{x}_i) - \rho_{e0})} \tag{5.20}$$

where $i = 0, ..., L$ where $L$ is the number of grid points in the simulation and $V_i$ is the volume of the cell. In PINC the domain is normalized to the cell size, giving a cell volume $V_i = 1$ and a grid point $\mathbf{x_i} = \mathbf{i}$, which further simplifies the expression.

In [32] an additional diagnostic is proposed, where the blob is tracked by the position of the highest electron density, $\mathbf{R}_{max}$. In PINC, the charge density is evaluated on the grid points, not in the center of the cells. To find the cell density, the charge density is interpolated from the closest grid points to the center of the cell. In one dimension, this becomes:

$$\rho_e(i + 0.5) = \frac{\rho_e(i) + \rho_e(i + 1)}{2} \tag{5.21}$$

$\mathbf{R}_{max}$ is found by determining the cell with highest density, $\rho_{e,max}$, and weighting the position inside the cell based on the density at the grid points.

## 5.3 Simulation Setup

The choice of domain and parameters for the simulations carried out for this thesis were largely based on those found in [32], [17] and [31] . The simulation parameters are explained in table 5.1, and figure 5.6 shows an illustration of the seeded blob with some key parameters for the simulation domain.

| Parameter | Description |
|---|---|
| $L_x$, $L_y$ | Length of simulation box in the $x$ and $y$ dimension |
| $\sigma$ | Blob radius (equal in $x$ and $y$) |
| $x_0$, $y_0$ | Location of blob center at $t = 0$ |
| $n_b$ | Initial blob density amplitude |
| $B_0$ | External magnetic field strength at $L_x = 0$ |
| $v_e$,$v_i$ | Electron and ion thermal velocity |
| $t_e$, $t_i$ | Electron and ion temperature |
| $T$ | Number of timesteps in simulation |
| $dt$ | Timestep |
| $N_x, N_y$ | Number of grid points in the $x$ and $y$ dimension |
| $dx$ | Grid length |
| $n_{sim}$ | Number of simulation particles per cell on average |

**Table 5.1:** Description of simulation parameters.

### 5.3.1 Boundary Conditions

A mixed set of boundary conditions were chosen, as seen in [31] . A choice of boundaries needed to be done for the electric potential, $\phi$, as well as for the particles, and these conditions had to match at each boundary. The particle boundary conditions are illustrated in figure 5.4.

In the $y$ direction, periodic boundaries were chosen, meaning that the electric potential at $y = 0$ was set to match the electric potential at $y = L_y$, giving

$$\phi(y = 0) = \phi(y = L_y). \tag{5.22}$$

The particle boundary condition was also set to periodic, and a particle crossing the boundaries $y = 0$ and $y = L_y$ would find itself at the other end of the simulation box.

In the $x$ direction the conditions were different at each end. At $x = 0$ a reflecting boundary was chosen, while the other end, at $x = L_x$, was chosen to be absorbing. Because of the gradient in the magnetic field going in the negative $x$ direction, periodic boundaries were not applicable in this dimension, as a particle crossing the boundaries would have found itself suddenly in a much stronger or weaker magnetic field, depending on which boundary it crossed.

The lower end of $x$ should be an open boundary where particles can cross in and out. This would have required an absorbing particle boundary, as well as a particle source function simulating the particles entering the domain. For simplicity a *reflective* boundary was chosen instead, where the particles passing $x = 0$ were reflected back into the domain. Dirichlet boundary conditions were chosen for the electric potential to match this:

$$\frac{\partial \phi(x = 0)}{\partial x} = 0, \tag{5.23}$$

giving an electric field that is equal at each side of the boundary, meaning that a particle crossing towards $x = -a$ would be going towards a potential equal to the one at $x = a$, which is where it wound find itself after having been reflected back.

At the higher end of $x$ the particles will reach the wall and be absorbed, therefore $x = L_x$ was set to be an absorbing boundary where particles which hit the wall

were taken out of the simulation. The corresponding potential boundary has von Neumann boundary conditions:

$$\phi(x = L_x) = c_n = 0 \tag{5.24}$$

where $c_n = 0$ was chosen for simplicity; any other choice of $c_n$ would have shifted the electric potential at all grid points up/down accordingly.

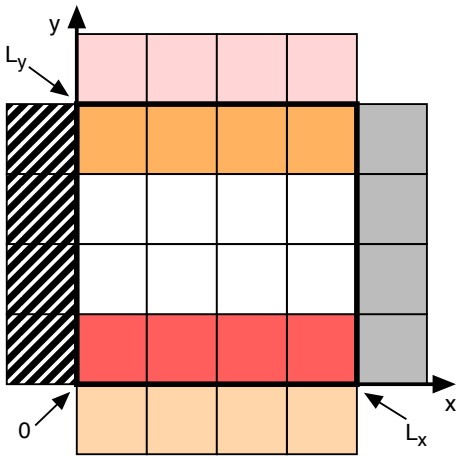


**Figure 5.4:** Illustration of the chosen boundary conditions. A particle entering the light pink ghost cells will be transported to the dark pink cells, similarly a particle entering the light orange ghost cells will be transported to the dark orange cells. A particle entering the gray ghost cells at $x = L_x$ will be absorbed, and a particle crossing the boundary at $x = 0$ will be reflected.

### 5.3.2 The Magnetic Field Gradient

In previous blob simulations using PIC, a larger $\nabla B$ than what is found in real SOLs is used [31]. A similar gradient to this was used for the simulations in this project, with the magnetic field being strongest at $x = 0$ and getting weaker towards $B = L_x$. The magnetic field is given as $\mathbf{B} = \{0, 0, B(x)\}$, where

$$B(x) = \frac{2L_x B_0}{2L_x + x} \tag{5.25}$$

gives the gradient, with the magnetic field being $B_0$ at $x = 0$ and $2/3B_0$ at $x = L_x$, which is a 33% reduction in the field strength over the domain. If the simulation domain spans the entire minor radius shown in figure 3.1 this would be comparable to the gradient in (3.1). The domain in these simulations are however much smaller than that, resulting in this gradient being an exaggeration. In figure 5.5 the theoretical line from (3.1) with the values from the TBR-1 tokamak inserted is plotted together with (5.25) with values used for the simulations in this thesis.

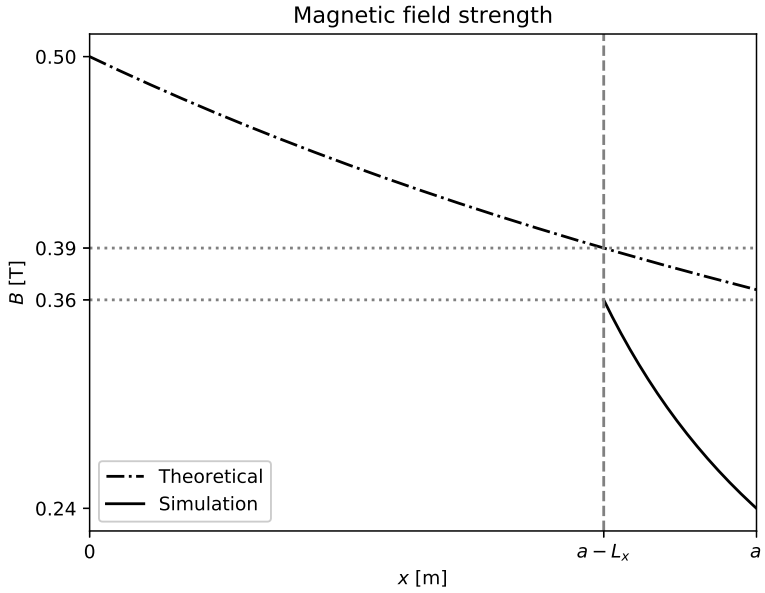


**Figure 5.5:** The theoretical magnetic gradient (dot-dash) plotted together with the gradient used for the simulations in this project, showing that the simulation gradient is exaggerated.

| Parameter | This thesis | M. Held *et al* [32] | L. Easy *et al* [17] |
|---|---|---|---|
| $t_e$ | 25.26 eV | 20 eV | 40 eV |
| $B_0$ | 0.36 T | 2 T | 0.5 T |
| $\rho_s$ | $3.33\times10^{-4}$m | $3.2\times10^{-4}$ m | - |

**Table 5.2:** Comparison of significant parameters between this thesis, [32] and [17].

### 5.3.3 Parameters

For the parameter choices of this thesis, emphasis was kept on choosing values close to those found in real SOLs. In [32], parameters are chosen based on the ASDEX Upgrade (AUG) Tokamak, while [17] lists parameters found in the Mega Ampere Spherical Tokamak (MAST). The magnetic field strength, $B_0$, electron temperature, $t_e$, and the drift scale, $\rho_s$, found in these papers are listed together with the parameters used in the simulations of this thesis in table 5.2. The chosen magnetic field is significantly lower than the magnetic field used in [32], where $B_0 = 2T$, but it is close to the magnetic field found in [17], where $B_0 = 0.5T$ and therefore the choice was made to keep this value for $B_0$. As a stronger magnetic field would give higher ion gyro frequencies and the frequencies must be resolved by the timestep (4.3.1), the magnetic field strength puts a constraint on the timestep.

Many parameters from these fluid simulations are not directly transferable to PIC simulations, and as a guide to complete the list of needed parameters, the ratios found in [31] were helpful. The ion-to-electron mass ratio of $m_i = 100m_e$ suggested there was kept. A comparison between a simulation of reduced mass ratio and the full mass ratio $m_i/m_e = 1836$ is presented in Appendix B, where a reduced ion mass is shown to give a scaling of $\Omega_i^{-1}$, leading to fewer timesteps being needed

to produce the same physics. In addition, the ratios between the basic plasma parameters, $\rho_s = \lambda_{De}$ and $\Omega_i = \omega_{pi} = 0.1\omega_{pe}$, with $\Omega_i$ being the ion gyroradius at $x = 0$, where the magnetic field is strongest, were reused for the simulations of this thesis. An overview of comparable parameters is shown in table 5.3.

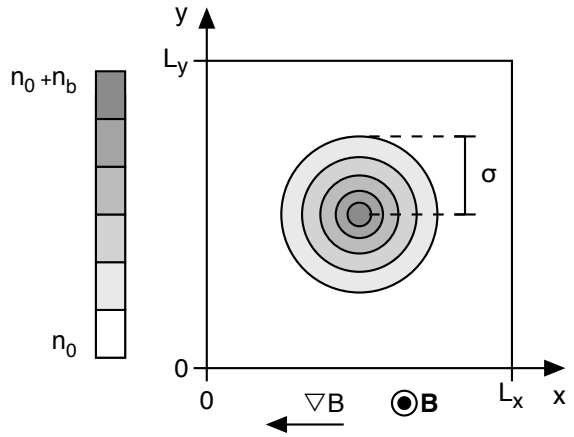


**Figure 5.6:** Illustration of the initialization of the blob in the domain.

The plasma frequencies were found to be

$$\omega_{pe} = 6.324 \times 10^9 \text{ s}^{-1} \tag{5.26}$$

for electrons and

$$\omega_{pi} = 0.1\omega_{pe} \tag{5.27}$$
$$= 6.324 \times 10^8 \text{ s}^{-1} \tag{5.28}$$

for ions, using the ratio $m_i/m_e = 100$. The ion cyclotron frequency for $x = 0$ was found using the ratio $\Omega_i/\omega_{pi} = 1$, giving

$$\Omega_i = 6.324 \times 10^8 \text{ s}^{-1}. \tag{5.29}$$

at $x = 0$. Together with the ion acoustic speed $c_s = \sqrt{k_B t_e/m_i} = 2.108 \times 10^5$ m/s, this gives the characteristic length scale of

$$\rho_s = \frac{c_s}{\Omega_i} \tag{5.30}$$
$$= 3.33 \times 10^{-4} \text{ m}, \tag{5.31}$$

which is very close to the one used in [32].

The blob size in $x$ and $y$, $\sigma = 5\rho_s$ was chosen from one of three values simulated in [32], and is of comparable size to those used in [31]. The blob was initialized in the middle of the domain, at $x_0 = L_x/2$ and $y_0 = L_y/2$.

The size of the domain was set to

$$L_x = L_y \sim 15\sigma, \tag{5.32}$$

as in [17], which is small compared to the $L_x = L_y = 40\sigma$ found in [32], but approximately the same as what is used in [31]. The domain was split into

$$N_x = N_y = 84 \text{ cells}, \tag{5.33}$$

with each cell being $0.9\rho_s$. A timestep of

$$dt = 0.025\omega_{pe}^{-1} \tag{5.34}$$
$$= 2.5 \times 10^{-4}\Omega_i^{-1} \tag{5.35}$$

was used, and the number of timesteps was $T = 50000$ for the short simulations, and $T = 100000$ for the additional long simulations, giving a full simulation time of $\sim 4 \times 10^{-7}$s, which is in order of magnitude of the lifetime of a blob found in (3.7).

| Parameter | This thesis | Hasegawa and Ishiguro [31], run 1 |
|---|---|---|
| $m_i$ | 100 $m_e$ | 100 $m_e$ |
| $\sigma$ | 5 $\rho_s$ | 3.9 $\rho_s$ |
| $\lambda_{De}$ | $\rho_s$ | $\rho_s$ |
| $\omega_{pe}$ | $10\Omega_i$ | $10\Omega_i$ |
| dx | 0.9 $\rho_s$ | 0.976 $\rho_s$ |
| dt | 2.5 $\times 10^{-3}\Omega_i^{-1}$ | $2.44\times 10^{-3}\Omega_i^{-1}$ |
| $L_x$ | $75\rho_s$ | 69 $\rho_s$ |
| $N_x$ | 84 | 64 |

**Table 5.3:** Comparison of computational parameters between this thesis and [31].

### Parameter scan

A parameter scan was carried out to investigate the effect of the ion temperature and the blob amplitude on the COM movement, as in [32]. Three values were picked for the blob amplitude:

$$n_b = \{0.5, 1, 2\}n_0 \tag{5.36}$$

To better resolve the blobs of smaller amplitudes, the number of simulation particles per cell was increased, giving the simulations densities $n_{sim} = \{162, 84, 64\}$ particles per cell on average, ordered from lowest to highest blob amplitude.

The following ion temperatures were used for the parameter scan:

$$t_i = \{0.1, 0.25, 1, 2, 4\}t_e. \tag{5.37}$$

The temperatures were converted to thermal velocities for the two particle species, electron and ions. For the electrons, the thermal velocity was kept at

$$v_{th,e} = \sqrt{\frac{k_b t_e}{m_e}} \tag{5.38}$$
$$v_{th,e} = 2.1 \times 10^6 \text{m/s} \tag{5.39}$$

throughout all simulations. The ion-to-electron temperature ratio with a mass ratio $m_i/m_e = 100$ and an ion temperature $t_i = \alpha t_e$ becomes

$$\frac{v_{th,i}}{v_{th,e}} = \sqrt{\frac{k_b \alpha t_e/100m_e}{k_b t_e/m_e}} \tag{5.40}$$
$$= \sqrt{\frac{\alpha}{100}} \tag{5.41}$$

| Ion temperature $[t_e]$ | Ion thermal velocity [m/s] |
|---|---|
| 0.1 | $6.6 \times 10^4$ |
| 0.25 | $1.1 \times 10^5$ |
| 1.0 | $2.1 \times 10^5$ |
| 2.0 | $3.0 \times 10^5$ |
| 4.0 | $4.2 \times 10^5$ |

**Table 5.4:** List of all simulated ion temperatures and the corresponding ion thermal velocity, assuming an ion-to-electron mass ratio $m_i/m_e = 100$.

If the mass ratio was set to a realistic $m_i'/m_e = 1836$, the velocity ratio would have been $v_{th,i}'/v_{th,e} = \sqrt{\alpha/1836}$, meaning that the simulated ion velocity $v_{th,i}$ is about 4.3 times higher than the ion velocity of ions with a realistic mass, $v_{th,i}'$:

$$\frac{v_{th,i}/v_{th,e}}{v_{th,i}'/v_{th,e}} = \sqrt{\frac{\alpha/100}{\alpha/1836}} \tag{5.42}$$

$$= \sqrt{\frac{1836}{100}} \tag{5.43}$$

$$v_{th,i} \approx 4.3 v_{th,i}'. \tag{5.44}$$

As the gyroradius (2.11) is dependent on the velocity of the particles, this will also be changed through the parameter scan. The gyroradius is also affected by the magnetic field, giving a gyroradius that gets continuously larger as $x$ increases. In table 5.5 the gyroradius of the electrons and the ions (for all different temperatures) is shown at three different locations in the domain.

| | at $x = 0$ | at $x = L_x/2$ | at $x = L_x$ |
|---|---|---|---|
| $\rho_e$ | $3.3 \times 10^{-5}$ m | $4.2 \times 10^{-5}$ m | $5.0 \times 10^{-5}$ m |
| $\rho_i(t_i = 0.1t_e)$ | $1.1 \times 10^{-4}$ m | $1.3 \times 10^{-4}$ m | $1.6 \times 10^{-4}$ m |
| $\rho_i(t_i = 0.25t_e)$ | $1.7 \times 10^{-4}$ m | $2.1 \times 10^{-4}$ m | $2.5 \times 10^{-4}$ m |
| $\rho_i(t_i = t_e)$ | $3.3 \times 10^{-4}$ m | $4.2 \times 10^{-4}$ m | $5.0 \times 10^{-4}$ m |
| $\rho_i(t_i = 2t_e)$ | $4.7 \times 10^{-4}$ m | $5.9 \times 10^{-4}$ m | $7.1 \times 10^{-4}$ m |
| $\rho_i(t_i = 4t_e)$ | $6.7 \times 10^{-4}$ m | $8.3 \times 10^{-4}$ m | $1 \times 10^{-3}$ m |

**Table 5.5:** The electron gyroradius and the different ion gyroradii for different ion temperatures at the left end, middle and right end of the domain.

### 5.3.4 Stability

The stability criteria given in Chapter 4 are evaluated to check the numerical stability of the simulations with the parameters given in the previous section.

#### Timestep

(4.23) evaluates to an accumulated phase error of $6.5 \times 10^{-2}$ radians over $10^5$ timesteps when $dt = 0.025\omega_{pe}$, which should not be problematic. The stability criteria should be evaluated for $\Omega_i$ as well as for $\omega_{pe}$:

$$\Delta t < 0.3\Omega_i \tag{5.45}$$

This is satisfied with $dt = 0.025\omega_{pe} = 0.25\Omega_i$. The accumulated phase error over $10^5$ timesteps becomes $e_{phase} \approx 65$ radians for $\Omega_i$.

### Grid Size

The grid size $dx = 0.9\lambda_{De}$ is clearly within the spatial stability criteria given in (4.24), and all the ion radii in table 5.5 are resolved by the grid size $dx = 3 \times 10^{-4}$ m. The electron gyro radius, however, is not entirely resolved, and the electrons may experience some numerical heating in these simulations.

### CFL Criteria

If a particle in PINC moves more than one cell, $dx$, during one timestep, $dt$, and with that moves outside of a subdomain, it would not be picked up by the neighbouring domain. To avoid losing particles in this way, a constraint

$$v_{max}dt < dx \tag{5.46}$$

is set. As this sets a limit to how fast the particle can travel, the CFL criteria presented in Section (4.3.3) is respected.

### Superparticle

In the simulations with fewest particles, there were 64 particles per cell on average, giving a total of $N_{sim} = 64 \times 84^2 = 4.5 \times 10^5$ simulation particles. The superparticle density is $n_{sim} = 7.2 \times 10^8$ m$^{-2}$, giving $N_D = n_{sim}\lambda_{De}^2 \approx 80$ superparticles per Debye square, which satisfies the criteria $N_D >> 1$.

| Parameter | Value | SI |
|---|---|---|
| $\rho_s$ | - | $3.33 \times 10^{-4}$m |
| $\Omega_i$ | - | $6.324 \times 10^8$s$^{-1}$ |
| $\lambda_{De}$ | $\rho_s$ | $3.33 \times 10^{-4}$m |
| $\omega_{pe}$ | $10\Omega_i$ | $6.324 \times 10^9$s$^{-1}$ |
| dx | 0.9 $\rho_s$ | $3\times10^{-4}$ m |
| $L_x, L_y$ | $75\rho_s$ | $2.5 \times 10^{-2}$m |
| dt | $2.5 \times 10^{-3}\Omega_i^{-1}$ | $3.953 \times 10^{-12}$ s |
| T | $\{125, 250\}\Omega_i^{-1}$ | $\{1.977, 3.953\} \times 10^{-7}$ s |
| $\sigma$ | 5 $\rho_s$ | $1.667 \times 10^{-3}$ m |
| $t_e$ | 25.26 eV | - |
| $t_i$ | $\{0.1, 0.25, 1, 2, 4\}t_e$ | - |
| $n_b$ | $\{0.5, 1, 2\}n_0$ | - |
| $n_{sim}$ | $\{162, 84, 64\}$/cell | - |
| $N_x, N_y$ | 84 cells | - |

**Table 5.6:** Full list of parameters used in the simulations producing the results presented in this thesis.

# Chapter 6

# Results

With the implementations described in Chapter 5 it was possible to simulate plasma blobs using PINC. Several simulations were conducted using the parameters from table 5.6, which are comparable to those used in other blob simulations. As the goal of this thesis was to investigate the dynamics of the blob, the electron center of mass was tracked through all simulations. The parameter scan outlined in the previous chapter made it possible to have a closer look at how the parameters $n_b$ and $t_i$ affect the blob dynamics, and the results of this parameter scan are presented in this chapter. Additionally, some attention is given to the electron and ion densities and how these are affected by the same parameters.

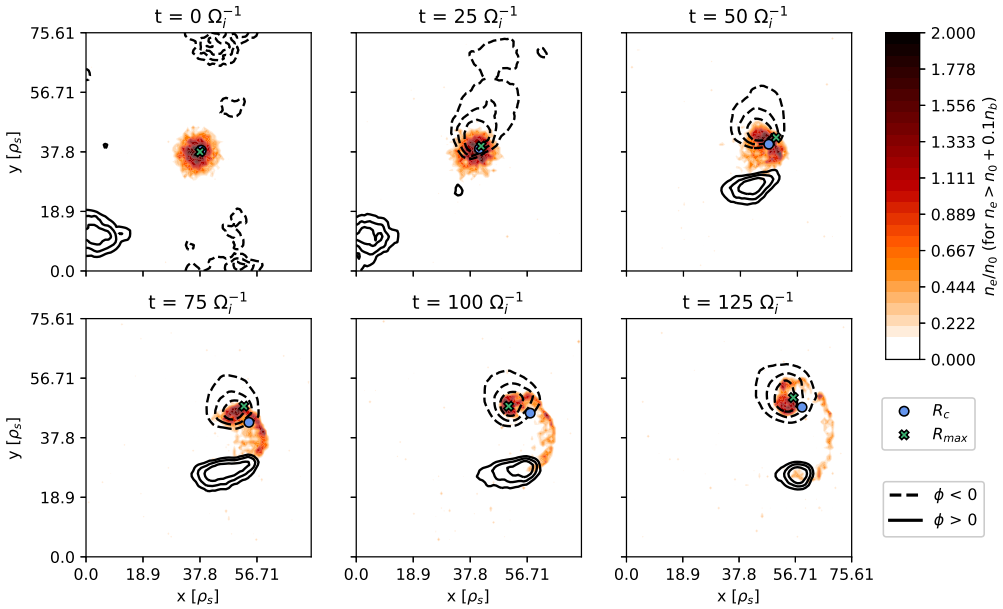


**Figure 6.1:** Time evolution of a blob over 125 $\Omega_i^{-1}$. The plots show the electron particle density normalized by the background density, for densities more than 10% of the initial blob amplitude, $n_b$, above the background density. The blue dot $R_c$ marks the electron center of mass (COM), while the green cross marks the position of the highest electron concentration at that timestep, $R_{max}$. The electric potential is represented by the 40% highest values on the positive (full lines) and negative (dashed lines) side, and the normalized maximum and minimum values are shown in table 6.1.

| $t[\Omega_i^{-1}]$ | $\phi_{min}/\phi_{0,min}$ | $\phi_{max}/\phi_{0,max}$ |
|---|---|---|
| 0 | 1.0 | 1.0 |
| 25 | 5.5 | 1.0 |
| 50 | 9.3 | 1.5 |
| 75 | 11.9 | 1.9 |
| 100 | 12.8 | 2.2 |
| 125 | 13.0 | 2.1 |

**Table 6.1:** Minimum and maximum potential values for $t_i = t_e$.

The results of the blob simulations show an initial charge separation, followed by a propagation in the radial direction, as expected from the blob theory presented in Chapter 3. This is exemplified in figure 6.1 by electron density plots of the blob at different timesteps. The blob shown has an initial blob amplitude $n_b = 2n_0$ and ion temperature $t_i = t_e$, and the electron density is plotted for every $25\Omega_i^{-1}$. The density is normalized by the background density $n_0$, and only densities high enough to count as blob are shown ($n > n_0 + 0.1n_b$, as illustrated in figure 5.2), to give a cleaner view of the blob. In addition the electron center of mass (COM), $R_c$ (5.15), is shown as a blue dot, while the position of the maximum electron density, $R_{max}$, is shown as a green cross. The electric potential is plotted in black contour lines, with full lines for $\phi > 0$ and dashed lines for $\phi < 0$. In both cases only the 40% highest values are shown, to avoid clutter, and the maximum and minimum values at each timestep are shown in table 6.1.

At $t = 25\Omega_i^{-1}$ the electric potential shows that the electrons have started drifting in the positive poloidal direction, and from $t = 50\Omega_i^{-1}$ the charge separation can clearly be seen in the potential lines. This charge separation creates an electric field, and at $t = 50\Omega_i^{-1}$ the $\mathbf{E} \times \mathbf{B}$ drift has started, and the COM can be seen to propagate radially outwards. As the blob propagates, it is being stretched out into a mushroom-like shape. The front of the blob seems to be more or less at the same position in the $y$ dimension throughout the simulation, while the electron COM moves poloidally upwards, indicating a higher electron density in the "upper" part of the mushroom blob, which can also be seen by the density plot itself.

## 6.1 Parameter Scan

The COM position, $R_c$, as shown in figure 6.1, was tracked through the parameter scan of the initial blob density amplitude and the ion temperature. The blob densities $n_b = \{0.5, 1, 2\}n_b$ and the ion temperatures $t_i = \{0.1, 0.25, 1, 2, 4\}t_e$ were investigated. The resulting displacement and velocity of the COM is presented here to show how the blob dynamics are affected by $n_b$ and $t_i$.

From the results it is clear that a lower blob density makes the calculation of the COM position $R_c$ more difficult due to fluctuations in the background plasma, giving a less reliable COM movement. To study the effect of the ion temperatures, the case of $n_b = 2n_0$ has been chosen as these results are least affected by the background noise.

### 6.1.1 Displacement

The displacement of the blob over time is shown in figure 6.2 for all the different parameter combinations. The lines show the movement of the blob over time, and the markers mark the end position after $125\Omega_i^{-1}$. The different ion temperatures are shown in colors that get darker as the temperature gets higher, and the different blob amplitudes are marker by different markers: circles, triangles and squares for $n_b = 0.5n_0$, $n_b = n_0$ and $n_b = 2n_e$ respectively. These identifiers are consistent throughout this chapter.

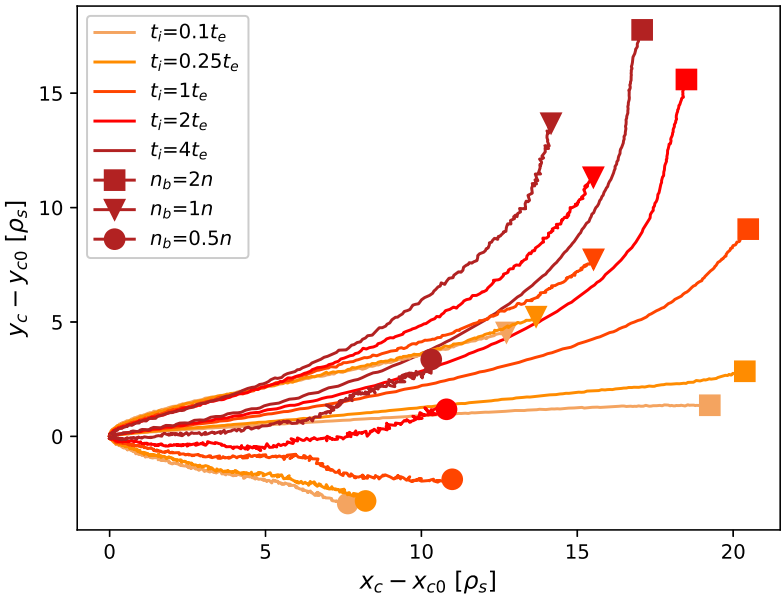


**Figure 6.2:** Full COM movement in the $xy$ plane for all parameter combinations. The lines show the movement of the blob over time, while the marker marks the end position. The displacement is given as the change from the initial position, $R_c - R_0$.

Figure 6.2 shows that a higher blob density makes the blob propagate further, especially in the radial direction, while the ion temperature seems to mostly affect the propagation in the poloidal direction. With low density ($n_b = 0.5n_0$) and colder ions ($t_i = \{0.1, 0.5, 1\}t_e$), the blob propagates slightly downward in the poloidal direction, while all the other parameter combinations give an end position where the $y$ coordinate is higher than at the starting point. The cold, dense blob ($t_i = 0.1t_e$ and $n_b = 2n_0$) appears to be the one that propagates most steadily along the $x$ axis.

As the previous studies of blob propagation presented in Chapter 3 looked at the radial and poloidal displacement separately, the displacement data of this thesis has been split into $x$ and $y$ direction and is plotted against time in figure 6.3, with $x_c - x_{c0} = \Delta x$ being the radial displacement and $y_c - y_{c0} = \Delta y$ the poloidal displacement. The panels show, from left to right, $n_b = 0.5n_0$, $n_b = n_0$ and $n_b = 2n_0$, and the ion temperatures are marked with their respective colors.

In the radial direction the position of the blob at $t = 125\Omega_i^{-1}$ increases with increased initial blob amplitude, while it does not vary much when changing the ion temperature. In the $y$ direction there is also a noticeable difference as the blob

amplitude is increased, but more noteworthy is the difference caused by the different ion temperatures.

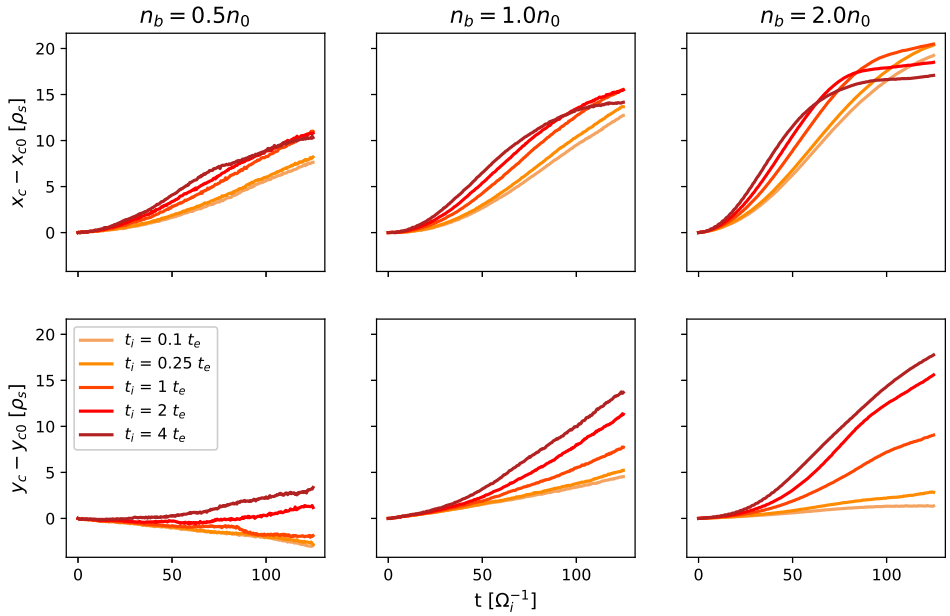


**Figure 6.3:** The blob COM displacement split into $x$ and $y$ direction for the parameter scan. The top row shows the movement in the radial ($x$) direction and the bottom row the movement in the poloidal ($y$) direction, while the columns show results for blobs with initial blob amplitudes $n_b = 0.5n_0$, $n_b = n_0$ and $n_b = 2n_0$ from left to right.

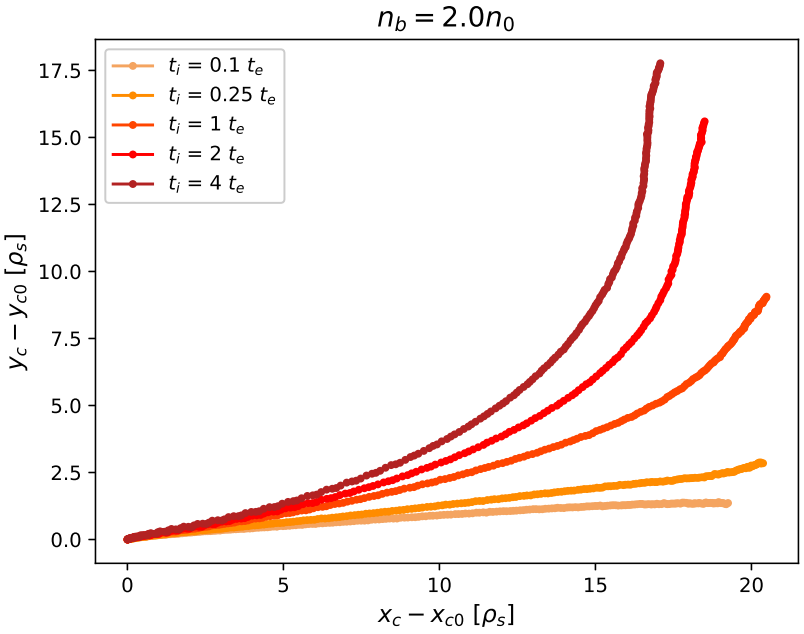


**Figure 6.4:** A closer look at the COM displacement for the case where $n_b = 2n_0$. The lines show the movement of the blob throughout the simulation.

For all the parameter combinations the radial propagation starts around $10\Omega_i^{-1}$, and the blob really hits its stride from $20\Omega_i^{-1}$. In the poloidal direction the propagation starts later, without much movement until $\sim 40\Omega_i^{-1}$. The blobs with warmer ions start moving poloidally a little earlier than the cold ion blobs.

The case of $n_b = 2n_0$ is shown separately in figure 6.4, where the displacement in the cross-field plane is plotted. Although the total distance traveled by the blob increases with increasing ion temperatures, the blob gets furthest in the radial direction for the case of $t_i = 0.25t_e$ and $t_i = t_e$, which both end up with a radial displacement of $\Delta x \approx 20.4\rho_s$. In the poloidal direction the blob displacement increases with increasing ion temperatures. For the warmest ions, $t_i = 4t_e$, the blob actually propagates as far in the poloidal direction as in the radial direction, with the displacement in both directions being $\sim 17\rho_s$.

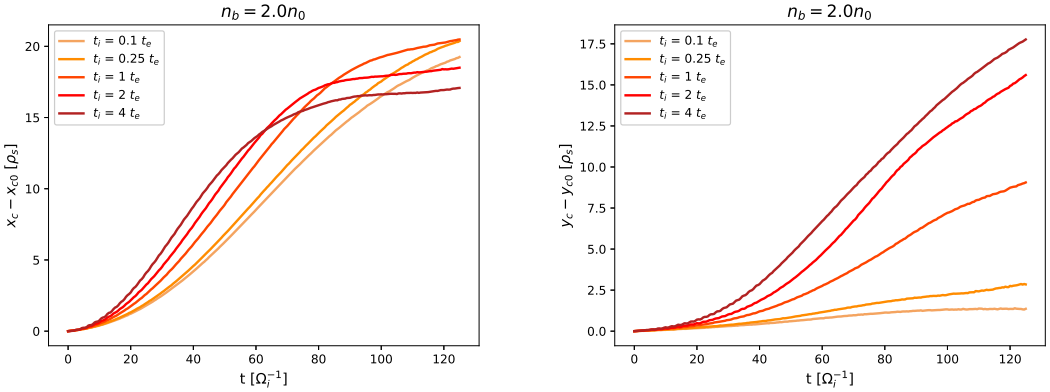


**Figure 6.5:** The COM displacement of the blob with initial amplitude $n_b = 2n_0$ split into $x$ and $y$ components, plotted against time.

In figure 6.5 the same cases are shown with the displacement in each dimension plotted against time. The blobs with warmer ions ($t_i = 4t_e$, $t_i = 2t_e$, and to some extent $t_i = t_e$) reach their maximum radial displacement within the simulation, with the blobs with the highest ion temperatures arriving there earlier. There seems to be little movement in the poloidal plane for the colder ions ($t_i = 0.1t_e$), while the warmer ones have a steady movement along $y$ even at the end of the simulation, indicating that they have not yet reached their end position in this direction.

### 6.1.2 Position at $t = 125\Omega_i$

The positions at the end of the simulations are compared to see how far the blob travels with the different parameter combinations. This is shown for the radial direction in figure 6.6 and for the poloidal direction in figure 6.7.

In the radial direction the position reached at $t = 125\Omega_i^{-1}$ is highest for the simulation with $t_i = t_e$, while the simulations with higher ion temperatures do not reach as far. For the blob with the lowest amplitude, $n_b = 0.5n_0$, the three warm blobs all reach $[10, 11]\rho_s$, while the cold ion blobs stop at a displacement of $\sim 8\rho_s$. For the high density blob with $n_b = 2n_0$, however, the cold ion blobs reach $\sim 20\rho_s$, which is close to the highest end displacement, while the blob with $t_i = 4t_e$ only

reaches a radial displacement of $\sim 17\rho_s$. The average radial position reached by the blobs of different temperatures increases with $\sim 4\rho_s$ when the amplitude is doubled.

It should be kept in mind that figure 6.5 shows that the two blobs with the coldest ions are still propagating in the radial direction at this timestep, while the other three have (more or less) reached their maximum position in that direction.

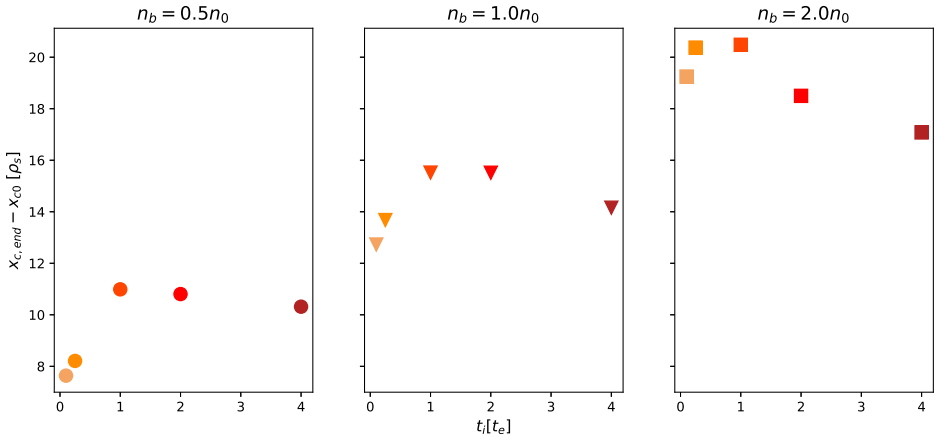


**Figure 6.6:** Total distance traveled by the COM in the radial direction at $t = 125\Omega_i^{-1}$. The frames show values for different blob amplitudes, while the ion temperatures are plotted on the $x$ axis as well as being marked with their respective colors.

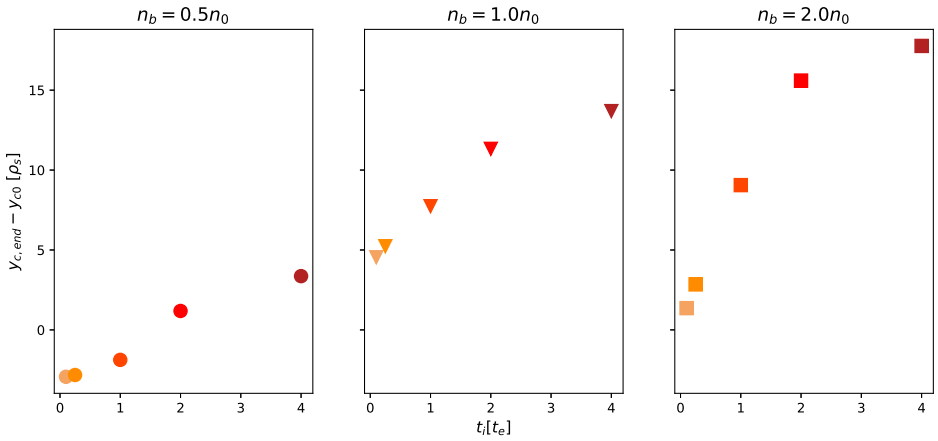


**Figure 6.7:** Total distance traveled by the COM in the poloidal direction at $t = 125\Omega_i^{-1}$ for the different parameter combinations.

All the blobs except the one with $t_i = 0.1t_e$ are shown in figure 6.5 to still be propagating in the poloidal direction at $t = 125\Omega_i^{-1}$, and figure 6.7 is therefore only useful for showing which blob propagates furthest within that timeframe. For the blob with the lowest density amplitude, $n_b = 0.5n_0$, the propagation is negative in the poloidal direction for $t_i = \{0.1, 0.25, 1\}t_e$, while it is positive in all other cases, although it reaches less than $5\rho_s$ even for the blob with $t_i = 4t_e$. This is only $\sim 1/4$ of how far the blob with the same temperature reaches when the initial amplitude is $2n_0$.

To correct for the fact that not all blobs shown in figure 6.6 have reached their maximum radial displacement and find a better estimate for how far the blob propagates depending on temperature, new simulations for a blob with density amplitude $n_b = 2n_0$ and ion temperatures $t_i = \{0.1, 0.2, 1, 2, 4\}t_e$ were carried out for twice as many timesteps. Although the simulations were twice as long, running until $t = 250\Omega_i^{-1}$, measurements of the COM are not reliable much longer than until $t = 150\Omega_i^{-1}$, either because of dissolution of the blob, or because it reached the edges of the simulation box. Closer attention is given to this at the end of this chapter, in Section 6.2.3.

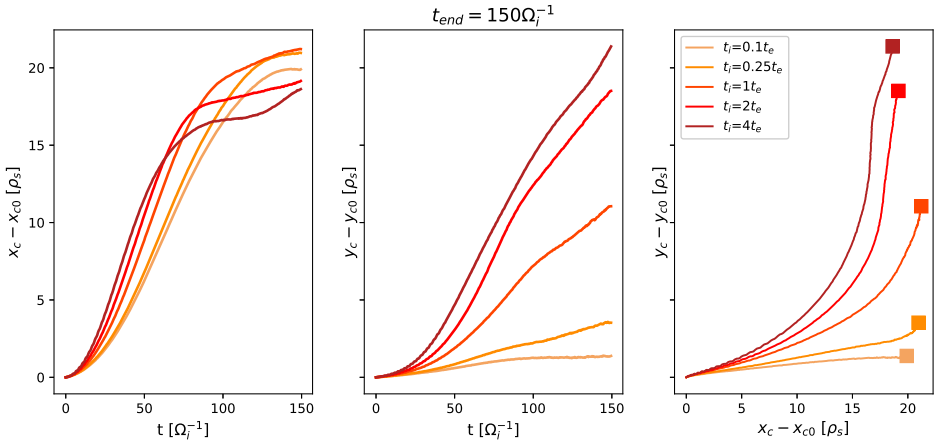


**Figure 6.8:** Blob displacement within $150\Omega_i^{-1}$. The plots show, from left to right, the propagation in the $x$ direction over time, in the $y$ direction over time and in the $xy$ plane, for a blob with $n_b = 2n_0$ and $t_i = \{0.1, 0.25, 1, 2, 4\}t_e$, until $t_{end} = 150\Omega_i^{-1}$.

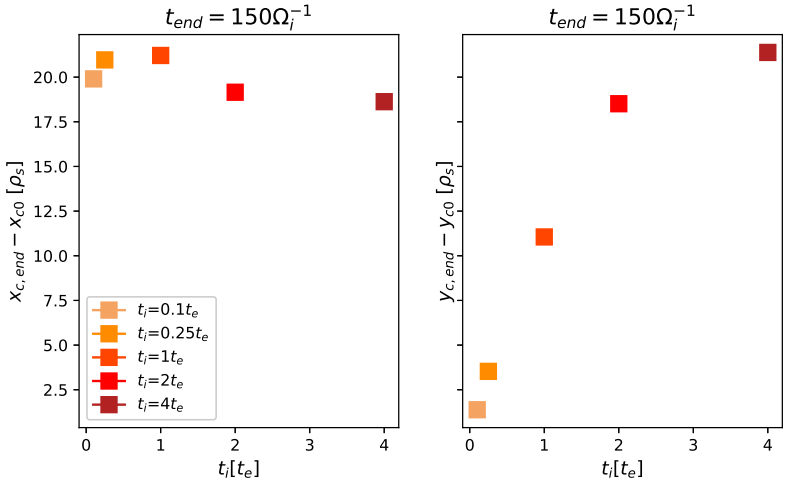


**Figure 6.9:** The total $x$ (left) and $y$ (right) displacement at $t_{end} = 150\Omega_i^{-1}$ of a blob with $n_b = 2n_0$ as a function of ion temperature, $t_i$.

Figure 6.8 shows that at $t = 150\Omega_i^{-1}$ the radial propagation has stopped for the blobs with lower ion temperatures as well as those with higher; the exception being the blob with the warmest ions, which has resumed propagation in the $x$

direction. In the poloidal direction there is still significant movement for the blobs with $t_i = \{1, 2, 4\}t_e$.

In figure 6.9 the positions at $t = 150\Omega_i^{-1}$ are plotted against the ion temperature, and the results are in agreement with those found at $t = 125\Omega_i^{-1}$ in figures 6.6 and 6.7, meaning that the blobs with $t_i = \{0.1, 0.5\}t_e$ did not propagate far in the radial direction between $t = 125\Omega_i^{-1}$ and $t = 150\Omega_i^{-1}$. As the blobs are dissolved, their propagation will stop quite abruptly, and the end position of all blobs in the radial direction seems to be $\sim 20\rho_s$. In the poloidal direction, however, all blobs except that of $t_i = 0.1t_e$ has increased the radial displacement additionally during the last $25\Omega_i^{-1}$.

### 6.1.3 COM Velocity

The velocities were calculated from the displacement to investigate the radial and poloidal blob velocities. The COM velocity, $v_b$, is shown for the entire range of parameters in figure 6.10. To reduce the noise the velocities have been averaged over $10.5\Omega_i^{-1}$. They are still quite noisy for the lower amplitude blobs, and it is hard to draw any conclusions from these plots. The case of $n_b = 0.5n_0$ is presented for completeness, while the case of $n_b = 2n_0$, which was least noisy, will be the focus of the velocity study.

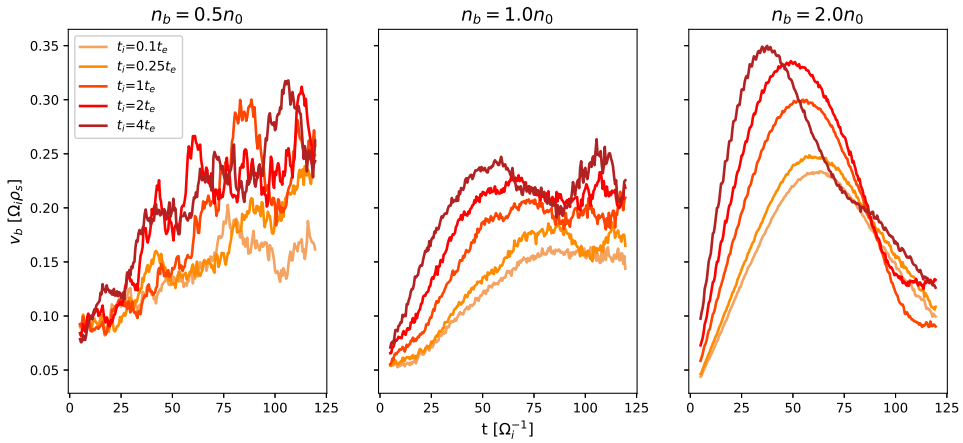


**Figure 6.10:** Velocity $v_b$ of the blob COM for the parameter combinations, averaged over 10.5 $\Omega_i^{-1}$ to reduce noise.

As the blob has been shown to propagate mainly in the radial direction to begin with, and then, to a varying degree, in the poloidal direction, it is again of interest to look at the $x$ and $y$ components separately. This is presented in figure 6.11. The $x$ component of the velocity, $v_{bx}$, reaches higher values than the $y$ component, $v_{by}$, with $v_{bx,max}$ being about 50% higher than $v_{by,max}$ for the blobs with $n_b = 2n_0$. The blob reaches its maximum velocity earlier in the radial direction than the poloidal direction. The blobs with warmer ions also reach their maximum velocity in the radial direction earlier than the colder ones, with about $10\Omega_i^{-1}$ separating the blobs with $n_b = 2n_0$ and $t_i = \{1, 2, 4\}t_e$.

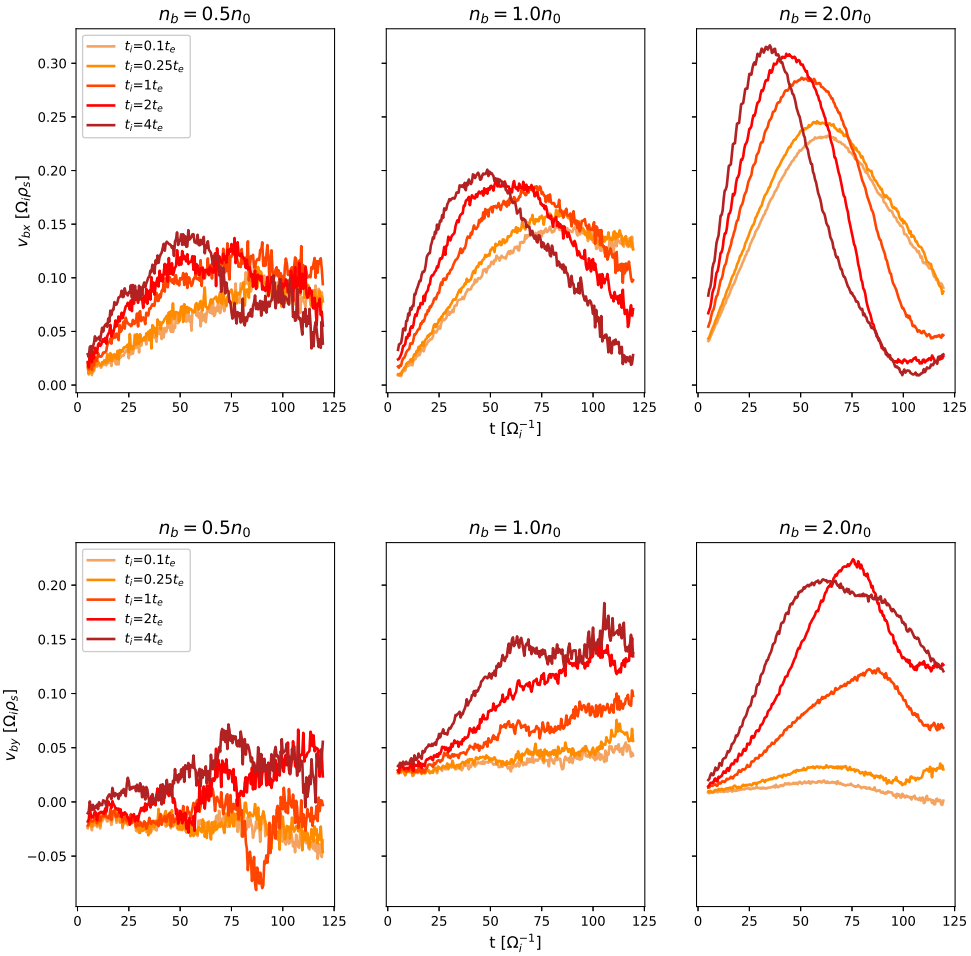


**Figure 6.11:** COM velocity split into $x$ and $y$ components, $v_{bx}$ and $v_{by}$, plotted for different temperatures and initial blob amplitudes.

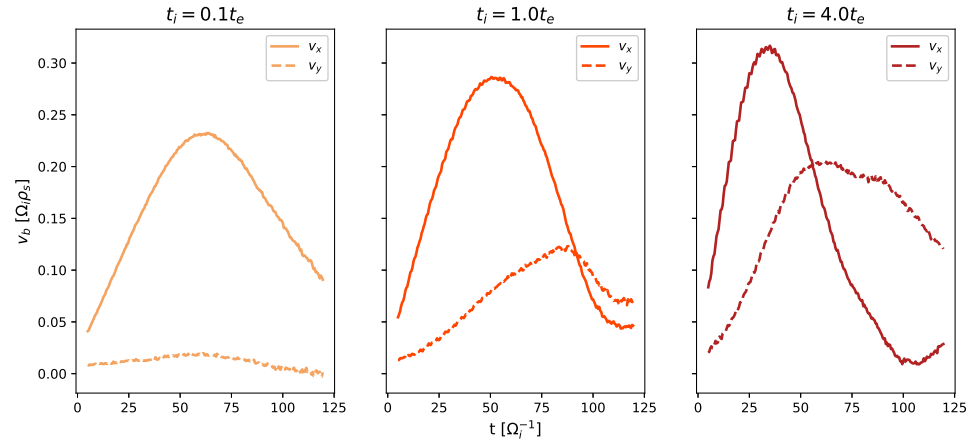


**Figure 6.12:** $x$ and $y$ components of the COM velocity of a blob with amplitude $n_b = 2n_0$ and $t_i = \{0.1, 1, 4\}t_e$ plotted together for comparison.

In figure 6.12 the radial and poloidal components of the velocities are shown together for three different ion temperatures, $t_i = \{0.1, 1, 4\}t_e$. For the case with

the coldest ions the velocity is almost purely in the radial direction. As the ions get warmer, the poloidal component of $v_b$ increases to $\sim 0.12\Omega_i\rho_s$ for $t_i = t_e$ and to $\sim 0.3\Omega_i\rho_s$ for $t_i = 4t_e$, but not until after the radial component has been reduced significantly from its maximum. Actually, the $x$ and $y$ components of $v_b$ seem to be equal when the maximum poloidal velocity is reached. For the warmest ions, the velocity moves over in the poloidal direction earlier, and the maximum of $v_y$ is reached soon after the maximum of $v_x$.

### 6.1.4 Maximum velocity

The maximum velocity of the radial component is plotted against the ion temperature in figure 6.13. The initial blob amplitude increases from the left frame to the right. The blobs with higher density and higher ion temperature reach a higher radial velocity, and it can be seen that the blob density has a higher impact on the velocity in this direction than the ion temperature does.

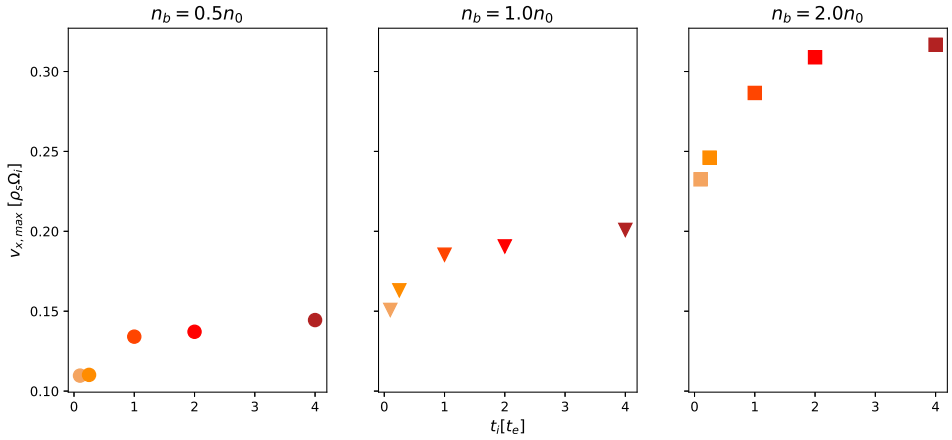


**Figure 6.13:** Maximum COM velocities in the radial direction.

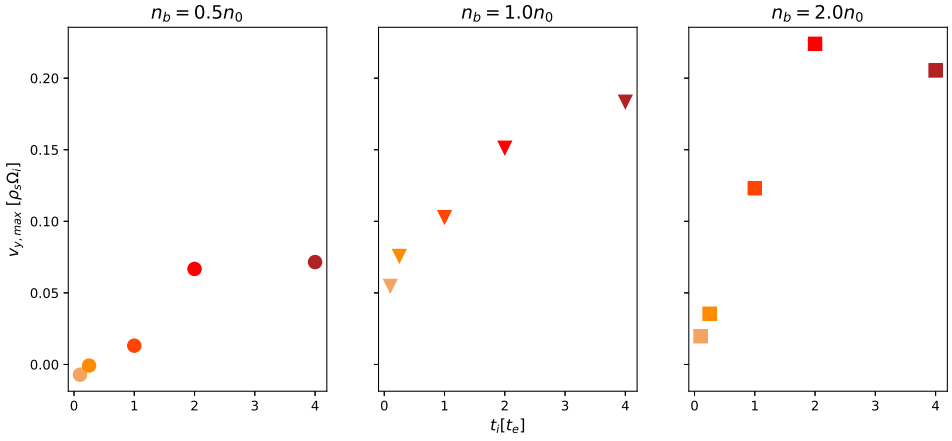


**Figure 6.14:** Maximum COM velocities in the poloidal direction.

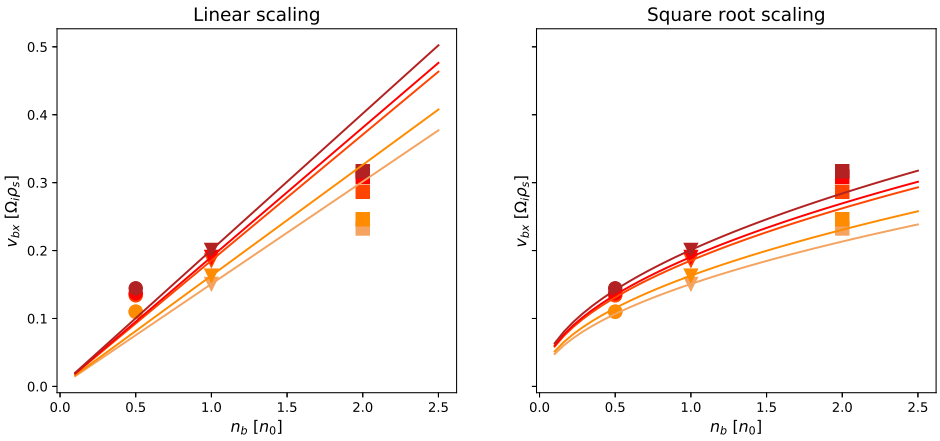


**Figure 6.15:** Maximum radial velocity of a blob with different temperatures plotted against the initial blob amplitude $n_b$. To the left maximum velocity has been fitted with a linear scaling from 3.15, to the right it has been fitted with the square root scaling from 3.16.

In figure 6.14 the same is shown for the poloidal component of the velocity, but with the opposite conclusion. In this dimension the ion temperature has a higher impact than the blob density. For the low density blob, increasing the ion temperature changes $v_{by,max}$ from negative to positive, and for the high density blob, the blob with warm ions reach $v_{by}$ more than five times higher than the cold ion blobs.

In figure 6.15 the maximum radial velocities have been plotted together with the expected scaling from (3.15) and (3.16), respectively. The root mean squared error

$$\text{RMSE} = \sqrt{\frac{\sum_i^{n_v} (v_{act,i} - v_{fit,i})^2}{n_v}} \tag{6.1}$$

is used to decide which fit is better. The results is presented in table 6.2 for each temperature and on average. This shows that the square root scaling is a better fit for the maximum velocity in this case, with an error that is 10% of that of the linear fit.

| | $t_i = 0.1t_e$ | $t_i = 0.25t_e$ | $t_i = 1t_e$ | $t_i = 2t_e$ | $t_i = 4t_e$ | Average |
|---|---|---|---|---|---|---|
| line | $1.98\times10^{-3}$ | $2.41\times10^{-3}$ | $2.93\times10^{-3}$ | $2.32\times10^{-3}$ | $3.06\times10^{-3}$ | $2.54\times10^{-3}$ |
| sqrt | $1.27\times10^{-4}$ | $8.88\times10^{-5}$ | $2.02\times10^{-4}$ | $5.20\times10^{-4}$ | $3.57\times10^{-4}$ | $2.59\times10^{-4}$ |

**Table 6.2:** Root mean squared error for the linear scaling ("line") and square root scaling ("sqrt") of radial maximum velocity by blob size.

### 6.1.5 Average Velocities

The average velocity from $t = 0$ to $t = 100\Omega_i^{-1}$ is plotted against $n_b$ in figure 6.16. From left to right is the radial component, the poloidal component and the total velocity. Again the blob density is shown to be the main impact on the radial component, while the ion temperature impacts the poloidal component.

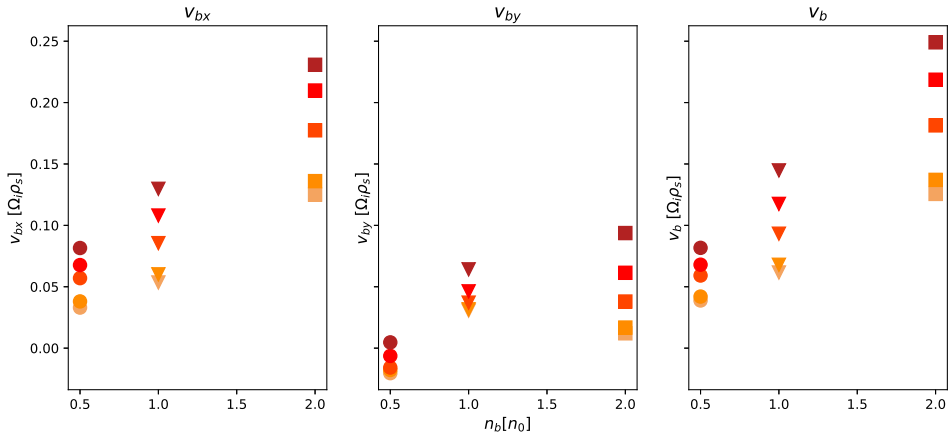


**Figure 6.16:** Average velocities for the radial component, poloidal component and the total velocity of the COM. The temperatures are marked with their respective colors, while the blob densities are shown on the $x$ axis as well as with their respective markers.

A square root scaling of the radial velocity is given in (3.14), which is different from the linear scaling in (3.6). To decide the best fit they are plotted together in figure 6.17. Again the errors are found according to (6.1) and presented in table 6.3, where the square root scaling is shown to on average have an error $\sim 6$ times larger than that of the linear scaling.

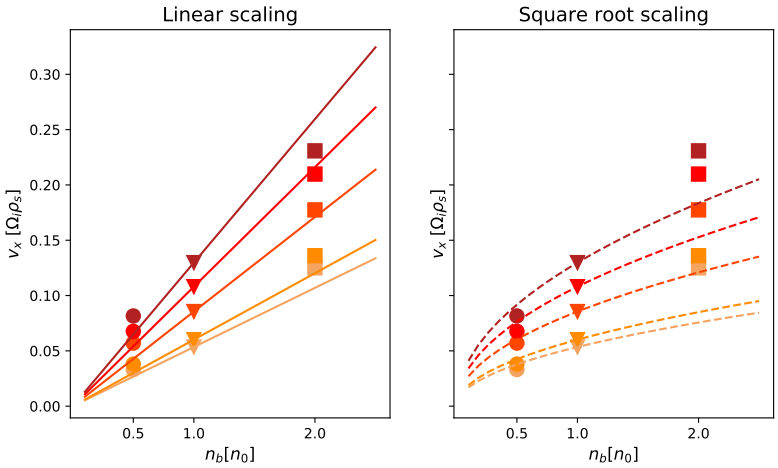


**Figure 6.17:** Average radial COM velocity for different blob amplitudes and ion temperatures. The temperatures are marked with their respective colors, while the blob densities are shown on the x axis as well as with their respective markers. The plots also show a linear scaling (left) and a square root scaling (right) of the velocity.

| | $t_i = 0.1t_e$ | $t_i = 0.25t_e$ | $t_i = 1t_e$ | $t_i = 2t_e$ | $t_i = 4t_e$ | Average |
|---|---|---|---|---|---|---|
| line | $1.23\times10^{-4}$ | $1.05\times10^{-4}$ | $8.14\times10^{-5}$ | $7.49\times10^{-5}$ | $3.68\times10^{-4}$ | $1.50\times10^{-4}$ |
| sqrt | $8.22\times10^{-4}$ | $8.75\times10^{-4}$ | $1.07\times10^{-3}$ | $1.11\times10^{-3}$ | $7.80\times10^{-4}$ | $9.31\times10^{-4}$ |

**Table 6.3:** Root mean squared error for the linear scaling ("line") and square root scaling ("sqrt") of average radial velocity by blob size.

There is a seemingly linear relationship between $v_{bx}$ and $n_b$, which fits the theoretical expression for $v_{bx}$ in equation (3.6). To investigate the effect of the blob amplitude, $n_b$ is set as the variable while the other parameters are kept as a constant $k_t$:

$$v_x = k_t n_b. \tag{6.2}$$

The ion temperature enters the theoretical equation through the ion gyroradius, but for now it is only taken into account by estimating a new constant $k_t$ from the average velocity of a blob with a certain amplitude for each temperature.

In figure 6.18 equation (6.2) is plotted together with the simulation results. In figure 6.18a, $n_b = \{0.5, 1, 2\}$ has been used in (6.2). This seems to overestimate the velocity. Considering that this is the average velocity over $100\Omega_i^{-1}$, this is not so surprising, as the maximum density of the blob does not stay at the initial value throughout the simulation. In figure 6.20 the evolution of maximum density is plotted for electrons and ions, separately and combined, giving an average combined particle density shown in table 6.4. In figure 6.18b, the average combined density of the particles over $100\Omega_i^{-1}$ is used instead of the initial amplitudes $n_b$. This clearly gives a better fit. In figure 6.18c the best linear fit is plotted for each temperature, and the $n_b$s implied by this fit are also shown in table 6.4.

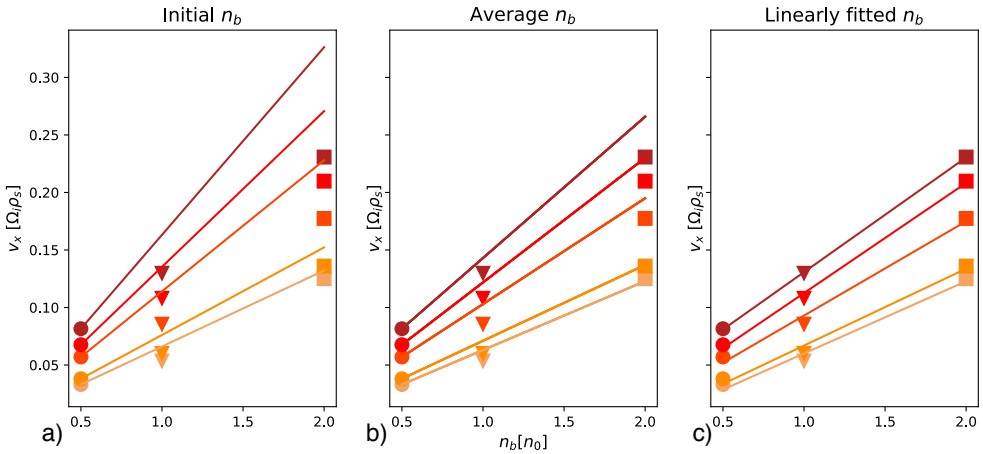


**Figure 6.18:** Average radial velocity of a blob with different temperatures plotted against the initial blob amplitude $n_b$. The lines are fitted with different $n_b$s.

| $t_i/t_e$ | $\overline{n}_{max}/n_0$ | $n_{fit}/n_0$ |
|---|---|---|
| 0.1 | 1.86 | 1.86 |
| 0.25 | 1.80 | 1.76 |
| 1.0 | 1.71 | 1.53 |
| 2.0 | 1.70 | 1.54 |
| 4.0 | 1.63 | 1.41 |

**Table 6.4:** Different estimates for $n_b$ for different ion temperatures when the initial blob amplitude was set to $n_b/n_0 = 2$. $\overline{n}_{max}$ is the measured average maximum particle density over $100\Omega_i^{-1}$ and $n_{fit}$ is the density that is needed for the linearly fitted lines.

Figure 6.18 shows that the ion temperature affects the average blob velocity in two ways: on its own, and also by affecting the particle density of the blob. The

blob velocity is not decided by the electron density alone, but by the total particle density, which shows that the effect of the ions cannot be discounted even when looking at the *electron* center of mass.

The temperature dependence of (3.6) is now considered. From the expression there is a linear dependency on the temperature:

$$v_x \propto \left(\frac{\rho_i}{\sigma}\right)^2 \propto t_i. \tag{6.3}$$

To check the fit of this, the average velocity a blob with initial amplitude $n_b = 2n_0$ and varying ion temperature were plotted for four different velocity scalings:

$$v_x \propto \left(\frac{\rho_i}{\sigma}\right)^2 \qquad v_x \propto \left(\frac{\rho_i}{\sigma}\right) \qquad v_x \propto \left(\frac{\rho_i}{\sigma}\right)^{1/2} \qquad v_x \propto \left(\frac{\rho_i}{\sigma}\right)^{1/3} \tag{6.4}$$

The results are shown in figure 6.19, and the errors of the scalings are shown in table 6.5.

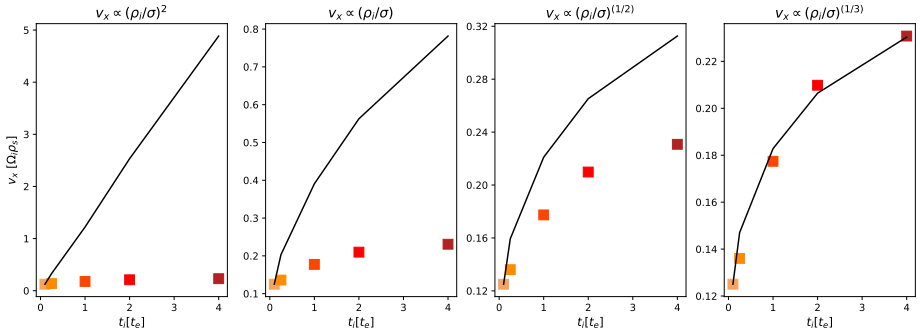


**Figure 6.19:** Average radial velocity of a blob with $n_b = 2n_0$. The temperatures are marked with their respective colors and shown on the $x$ axis. The plots show four different scalings for the velocity.

| | $v_x \perp (\rho_i/\sigma)^2$ | $v_x \perp \rho_i/\sigma$ | $v_x \perp (\rho_i/\sigma)^{1/2}$ | $v_x \perp (\rho_i/\sigma)^{1/3}$ |
|---|---|---|---|---|
| RMSE | 5.64 | $9.56\times10^{-2}$ | $2.44\times10^{-3}$ | $3.23\times10^{-5}$ |

**Table 6.5:** Root mean squared error for the four temperature scalings of radial velocity.

## 6.2 Particle Densities

In addition to providing the average particle density used above, figure 6.20 shows that the maximum ion density is highly affected by the ion temperature. For $t_i = \{1, 2, 4\}t_e$, the ion density drops down to almost the level of the initial ion background density, $n_0$, within $125\Omega_i^{-1}$, which leaves it at the level of fluctuations in the background density. The maximum electron density does not seem to depend on the ion temperature for this time range.

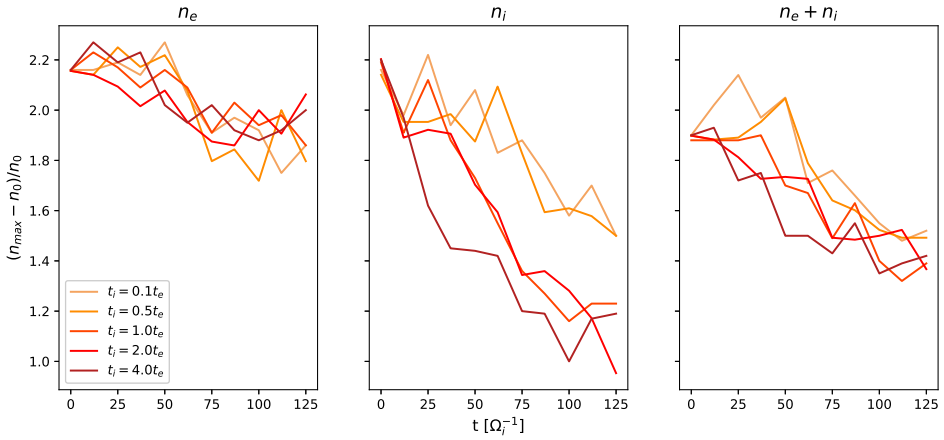


**Figure 6.20:** Evolution of the maximum particle density for electrons and ions separately and combined for different ion temperatures and for an initial blob amplitude $n_b = 2n_0$.

In this section the particle densities and the evolution of these is explored through contour plots for certain timesteps. The two cases on each end of the ion temperature scale are compared ($t_i = 0.1t_e$ and $t_i = 4t_e$), both for a blob with initial density amplitude $n_b = 2n_0$.

### 6.2.1 Cold Ions ($t_i = 0.1t_e$)

The electron density is plotted for six different timesteps in figure 6.21. The expected mushroom shape described in studies presented in Section 3.5 can be seen at $t = 75\Omega_i^{-1}$, and from there the blob continues to expand in both directions along $y$ while propagating radially outwards.

The electron center of mass, $R_c$, is plotted as a blue circle, and the maximum electron density, $R_{max}$ is shown as a green X. While $R_c$ seems to be moving more or less steadily along the $x$ axis, with only a small deviation in the positive poloidal direction, $R_{max}$ is positioned on the upper lobe of the blob, on the expansion in the positive poloidal direction.

| $t[\Omega_i^{-1}]$ | $\phi_{min}/\phi_{0,min}$ | $\phi_{max}/\phi_{0,max}$ |
|---|---|---|
| 0 | 1.0 | 1.0 |
| 25 | 1.4 | 2.3 |
| 50 | 3.3 | 2.2 |
| 75 | 5.0 | 2.7 |
| 100 | 6.0 | 3.2 |
| 125 | 7.0 | 3.5 |

**Table 6.6:** Minimum and maximum potential values for $t_i = 0.1t_e$.

The electric potential is shown in black, with full lines indicating a positive potential and dotted lines indicating a negative potential. At $t = 25\Omega_i^{-1}$ the charge separation is already observable through $\phi$, and at $t = 50\Omega_i^{-1}$ the blob has started moving in the $x$ direction as a consequence of the electric field that has been created, giving the $\mathbf{E} \times \mathbf{B}$ drift.

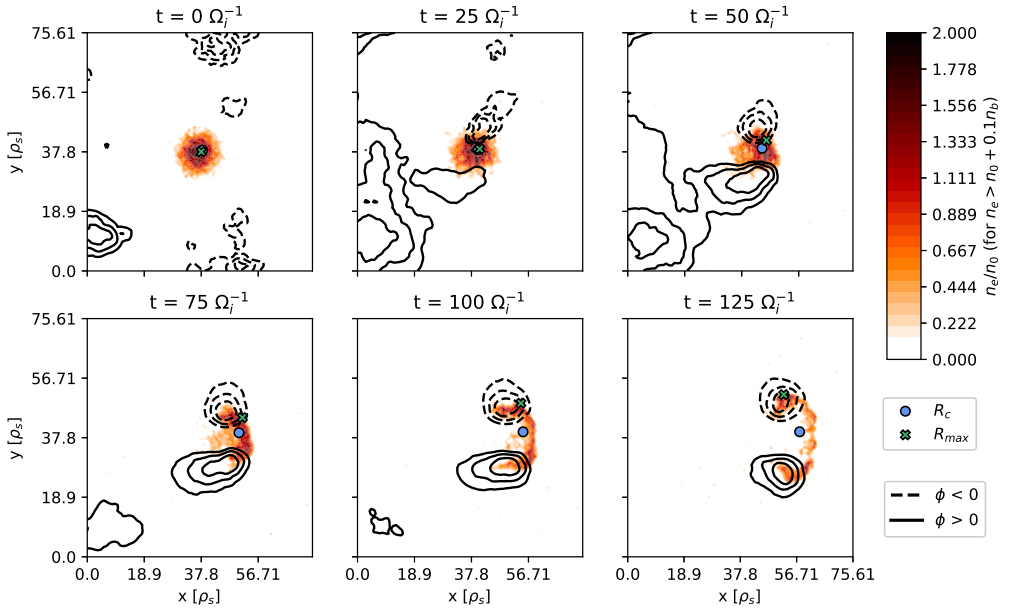


**Figure 6.21:** Electron density for a blob with $n_b = 2n_0$ and $t_i = 0.1t_e$. The electron center of mass (COM), $R_c$ is shown, as well as the position of the maximum electron density, $R_{max}$. The normalized values for the potential is shown in table 6.6.

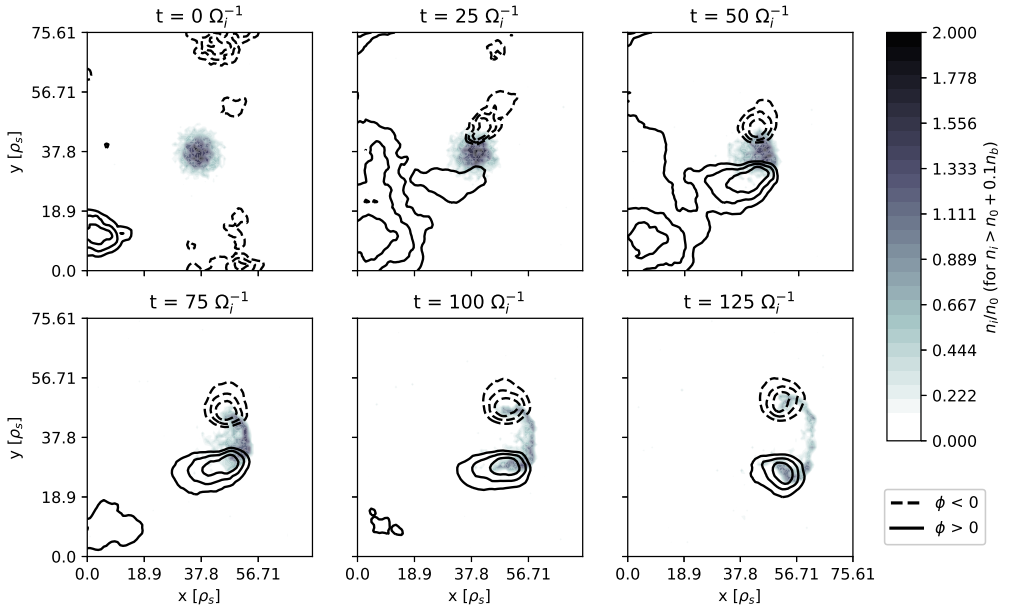


**Figure 6.22:** Ion density for a blob with $n_b = 2n_0$ and $t_i = 0.1t_e$, only showing densities higher than $n_0 + 0.1n_b$.

Similarly, in figure 6.22 the ion density is shown for the same timesteps as in figure 6.21. Although the ions concentrate on the lower lobe of the blob instead of the upper, they largely follow the same motion as the electrons, and the blob clearly consists of ions as well as electrons. At $t = 125\Omega_i^{-1}$ the ion density is low except for in the lower lobe.

In both of these figures the densities have been normalized by the background density, $n_0$, and only densities higher than $n_0 + 0.1n_b = 1.2n_0$ are shown, as this is what has been chosen as the definition of the blob (shown in figure 5.2).

To better study the evolution of the ion density, figure 6.23 shows the full ion density normalized by the highest initial density, $n_0 + n_b$. Again, the ion density can be seen to drop towards the end, and the ion blob is clearly denser in the lower lobe, on the side that has expanded in the negative poloidal direction.

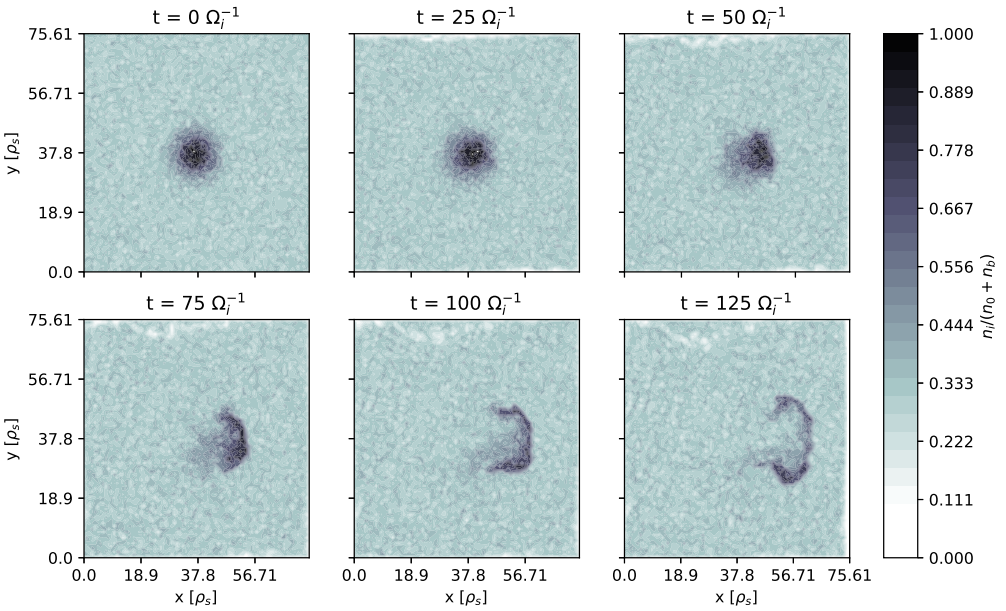


**Figure 6.23:** Full ion density for a blob with $n_b = 2n_0$ and $t_i = 0.1t_e$. The density is normalized by the initial maximum density, $n_0 + n_b$.

### 6.2.2 Warm Ions ($t_i = 4t_e$)

The electron density for the case of high ion temperature ($t_i = 4t_e$) is plotted in figure 6.24. The structure of the plot is the same as in figure 6.21, and the timesteps are the same. The behavior of the blob is significantly different in this case. Instead of the mushroom shape from the previous example, the blob is now closer to a tadpole in appearance, as it looks like a blob with a tail.

As already expected from the position plots, the blob now has a clear movement in the poloidal direction. In this case $R_c$ and $R_{max}$ are located almost in the same spot on the head of the tadpole, and the tail of the tadpole indicates that the blob is spinning while moving upwards.

The electric potential associated with the blob in this case is purely negative. Although a charge separation occurs, there is no concentration of ions strong enough to create a positive potential in this case.

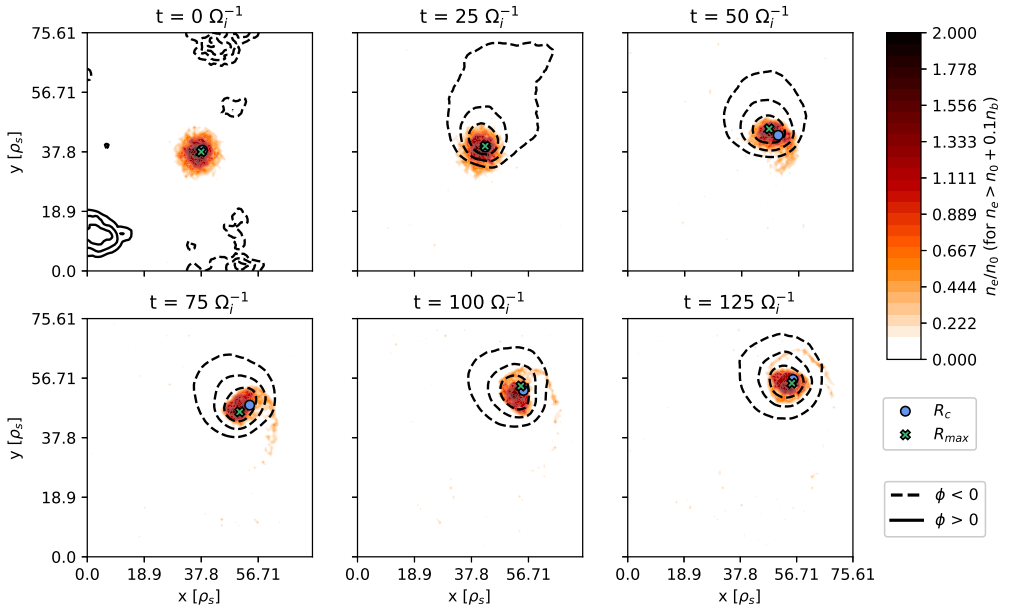


**Figure 6.24:** Electron density for a blob with $n_b = 2n_0$ and $t_i = 4t_e$. The electron center of mass (COM), $R_c$ is shown, as well as the position of the maximum electron density, $R_{max}$. The normalized values for the potential is shown in table 6.7.

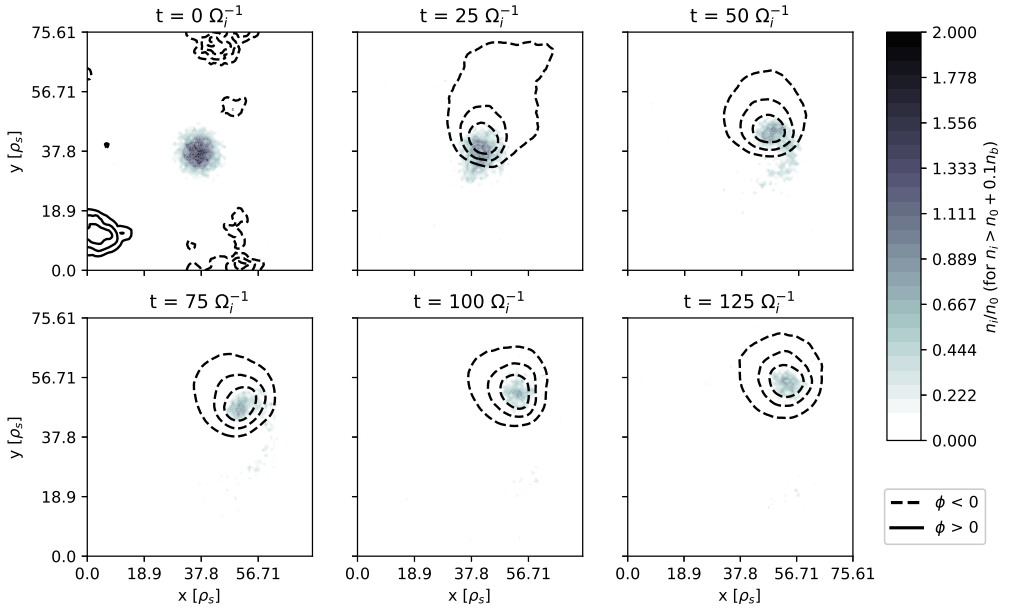


**Figure 6.25:** Ion density for a blob with $n_b = 2n_0$ and $t_i = 4t_e$, only showing densities higher than $n_0 + 0.1n_b$.

The ion density is shown in figure 6.25, and as in the cold ion case, the blob is again seen to consist of both electrons and ions, although the ion density is very low in this case. There is no ion lobe as there was in the case of cold ions, and the densest area of ions is also located at the head of the tadpole, where the electron

density is highest. The strong decrease in the ion density early in the simulation matches the expectations from figure 6.20.

In figure 6.26 the ion density is again plotted in full, normalized by $n_0 + n_b$. Already after $25\Omega_i^{-1}$ the reduction in the ion density is obvious, and by $t = 125\Omega_i^{-1}$ the maximum density is about half of what it started out as. This is in agreement with the second plot in figure 6.20. Because of the low concentration of ions in the blob, the tadpole blob can be considered an *electron blob*. The normalized ion plot also shows the presence of a vague ion "tail", stretching out from the tadpole in the poloidal direction. As this tail is not present in figure 6.25, its density must be less than the minimum density needed to be considered a blob. The short electron tail present in figure 6.24 is located at the same positions as this ion tail.

A white line can be seen at the upper end of the $x$ axis in figure 6.26. This is caused by the absorbing boundary condition, and it shows that there are some boundary effects to be aware of in the simulations with warm ions that are not present to the same degree in the simulation with cold ions.

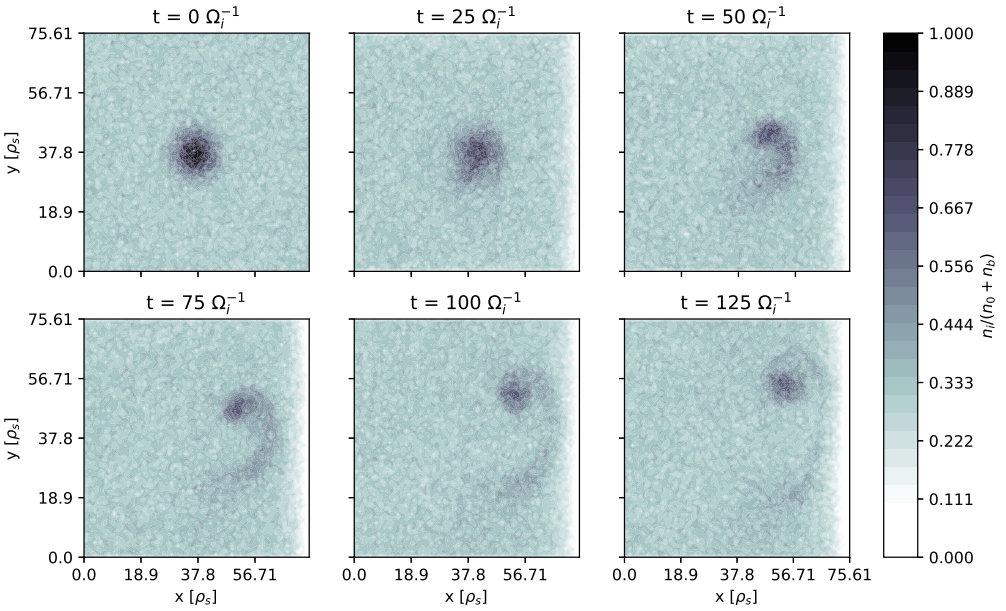


**Figure 6.26:** Full ion density for a blob with $n_b = 2n_0$ and $t_i = 4t_e$. The density is normalized by the initial maximum density, $n_0 + n_b$.

| $t[\Omega_i^{-1}]$ | $\phi_{min}/\phi_{0,min}$ | $\phi_{max}/\phi_{0,max}$ |
|---|---|---|
| 0 | 1.0 | 1.0 |
| 25 | 17.9 | 0.0 |
| 50 | 23.0 | 0.0 |
| 75 | 26.3 | 0.0 |
| 100 | 27.4 | 0.0 |
| 125 | 26.4 | 0.0 |

**Table 6.7:** Minimum and maximum potential values for $t_i = 4t_e$.

### 6.2.3 Dissolving the Blob

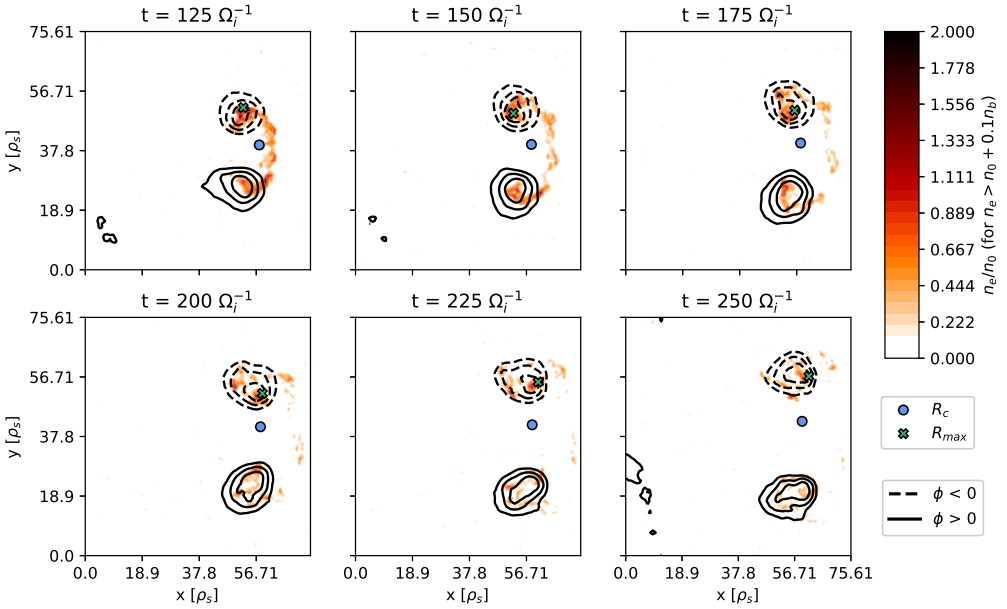


**Figure 6.27:** Electron density from $t = 125\Omega_i^{-1}$ to $t = 250\Omega_i^{-1}$ for $t_i = 0.1t_e$. The mushroom blob dissolves and can no longer be considered a blob.

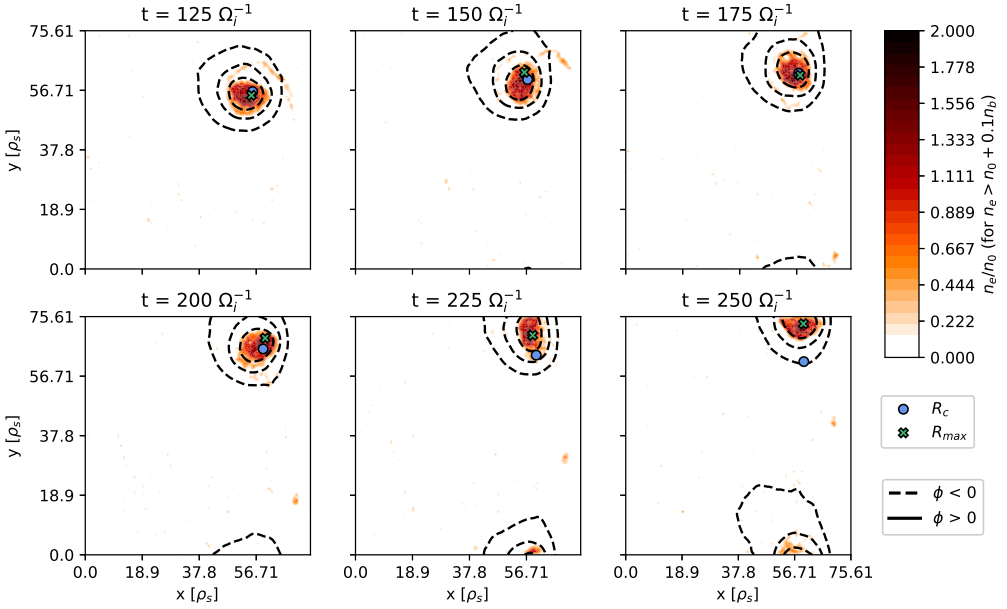


**Figure 6.28:** Electron density from $t = 125\Omega_i^{-1}$ to $t = 250\Omega_i^{-1}$ for $t_i = 4t_e$. The tadpole blob remains dense while it continues to propagate in the $y$ direction.

To observe the fate of the blob, a longer simulation was carried out with $t_{end} = 250\Omega_i^{-1}$. The result for $t_i = 0.1t_e$ is shown in figure 6.27. Although there are still areas left where $n_e > n_0 + 0.1n_b$ even at $250\Omega_i^{-1}$, these are not connected, and the last timestep at which the "blob shape" can be observed properly is at $150\Omega_i^{-1}$; after this the blob is either dissolved or in the process of being dissolved.

In the case of warm ions, $t_i = 4t_e$, it could already be seen in figure 6.24 that the electron blob remained dense. Figure 6.28 shows that this continues to be the case also for the next $125\Omega_i^{-1}$. The tadpole blob continues to move upwards, while it keeps spinning. It appears to lose its tail around $175\Omega_i^{-1}$, when it also, due to the periodic boundary conditions in the $y$ direction, crosses the upper boundary $L_y$ and moves into the box at $y = 0$. As the blob crosses the periodic boundary, the COM tracking is no longer correct.

# Chapter 7

# Discussion

The results of the particle simulations were presented in Chapter 6. In this chapter, the results are discussed, particularly the results of the parameter scan and how the blob velocity is affected by $n_b$ and $t_i$, and the results are compared to those found in previous investigations of blob dynamics.

Lower blob amplitudes made the results noisy because the ratio between the blob density and the variations in the background density became small, making it difficult to distinguish between these and correctly define the blob. Because of this, the effects of varying the ion temperature are mainly discussed in the case where $n_b = 2n_0$, which gave the least noise-affected results.

The expected blob behavior described in section 3.5 and illustrated in figure 3.4 is found in the simulations results, i. e. in figure 6.21. There is a charge separation caused by $\mathbf{B} \times \nabla B$ drift of the ion and electrons in opposite directions, causing an electric field which in turn gives the blob a $\mathbf{E} \times \mathbf{B}$ drift in the radial direction. This description proves to be more aligned with the dynamics of a blob with cold ions, and a slightly different description of a blob with warm ions is needed, and will be presented and discussed in this chapter.

## 7.1 Varying the Initial Blob Density Amplitude

Varying the initial blob density amplitude, $n_b$, is shown to affect the displacement of the blob, as well as the velocity, mainly in the radial direction.

### 7.1.1 Effect on Displacement

The blob is shown in figure 6.2 to have a larger total displacement as the blob density increases, especially noticeable is the increase in radial displacement for the cold ion blobs. The increase in radial displacement is however clear for the other temperatures as well: The positions at $t = 125\Omega_i^{-1}$ are between $5\rho_s$ and $10\rho_s$ for $n_b = 0.5n_0$, between $10\rho_s$ and 15 $\rho_s$ for $n_b = n_0$ and between $15\rho_s$ and $20\rho_s$ for $n_b = 2n_0$.

While the high-density blobs are shown to reach a terminal position in the radial direction within the simulation time, the blob of smaller initial amplitudes are still on the move at the end of the simulations. To compare the effect of the blob amplitude on the radial displacement, the blobs with coldest ions, $t_i = [0.1, 0.25]t_e$, are taken as example, as these move almost exclusively in the radial direction. For

an initial blob amplitude of $n_b = 2n_0$ these blobs were shown to reach their terminal position around $150\Omega_i^{-1}$, when they dissolved, and already before $t = 125\Omega_i^{-1}$ the period of linear displacement can be seen to be over in figure 6.5. The blobs with the same ion temperatures, but lower initial blob density, are still experiencing a linear propagation at this time, as seen in figure 6.3. This means that although they have not reached as far within the same time frame, it is possible that the blobs with an initially smaller density amplitude *live longer* than the blobs of high initial density, and that the total radial displacement of the blob *within the lifetime of the blob* could be the same.

The poloidal displacement is not as much affected by the increased blob amplitude. The strongest effect in this direction is that the blob with $n_b = 2n_0$ has a more stable propagation along $x$ for the cold ions, while for the lowest blob amplitude, $n_b = 0.5n_0$, the blob is shown to propagate in the negative $y$ direction, and the cold ion blob with $n_b = n_0$ has a noticeable displacement in the positive $y$ direction. The effect of noise on this result is uncertain, and the change of number of simulation particles between these simulations could also have affected the results to some degree.

### 7.1.2 The Effect on Velocity

Comparing the blobs of initial amplitude $n_b = n_0$ and $n_b = 2n_0$, it is clear that compared to the blobs with lower amplitude, the blob with the highest initial amplitude

1) has a higher maximum radial velocity,

2) reaches the maximum velocity earlier,

3) maintains the maximum velocity for a shorter amount of time, and

4) starts the deceleration earlier.

The process could get an earlier start because of a higher pressure in the high-density blob, causing a higher rate of diffusion which accelerates the separation of ions and electrons in the early stages of the simulation, explaining point 2).

Because there is a higher total particle density, the ion density in the ion lobe and the electron density in the electron lobe will also be higher, causing a higher difference in charge density. This means the electric field is stronger, and could explain point 1) above, because the $\mathbf{E} \times \mathbf{B}$ force would be stronger and give a higher radial velocity.

A possible explanation for point 3) and 4) is again connected to the density of the lobes. When these are denser, they will experience a stronger $\mathbf{B} \times \nabla B$ drift in opposite directions, possibly tearing the blob apart and dissolving the blob earlier than if it had a lower initial blob density amplitude.

When comparing to the velocity scaling equations presented in Section 3.5, the linear scaling of the radial velocity in (3.6) seems to agree with the average radial velocity found in this thesis, $v_x \propto n_b/n_0$. The agreement is shown in table 6.3 to be better for the blobs with cold ions than for those with warm ions, which could be explained by the decrease in ion density that is more significant for the blobs with warmer ions, an effect which will be discussed in the next section.

The maximum blob velocity was compared to the linear scaling of (3.15) and to the square root scaling of (3.16), and in figure 6.15 the square root scaling has an error that is 10% of that of the linear scaling, as shown in table 6.2. The square root scaling underestimates $v_{x,max}$ for $n_b = 2n_0$ slightly, while the linear fit grossly overestimates these velocities. This matches the expectations from previous work [40] [64], that as the ratio between the blob density and the blob size, $n_b/\sigma$ increases, the maximum radial velocity changes from the linear to the square root scaling, as in (3.19).

## 7.2 Varying the Ion Temperature

The other parameter that was investigated through the parameter scan was the ion temperature, $t_i$, which was varied from $0.1t_e$ (cold) to $4t_e$ (warm). The ion temperature is shown to have an effect on the particle density evolution, and on the propagation in the $xy$ plane, as well as the maximum and average velocities of the blob.

### 7.2.1 Effect on Density

The rapid decrease in maximum ion density shown for warm ions in figure 6.20 can also be seen in figure 6.26, where the ions are visibly less dense than in figure 6.23, which shows the ion density in the blob with cold ions. This decrease in ion density can be explained by the warmer ions having a much higher thermal velocity than the cold ones, making the $\mathbf{B} \times \nabla B$ effect on the ions less significant, and letting the ion blob diffuse to a greater degree than for the cold ions.

| | at $x = 0$ | at $x = L_x/2$ | at $x = L_x$ |
|---|---|---|---|
| $\rho_e/\sigma$ | 0.020 | 0.025 | 0.030 |
| $\rho_i(t_i = 0.1t_e)/\sigma$ | 0.07 | 0.08 | 0.10 |
| $\rho_i(t_i = 0.25t_e)/\sigma$ | 0.10 | 0.13 | 0.15 |
| $\rho_i(t_i = t_e)/\sigma$ | 0.20 | 0.25 | 0.30 |
| $\rho_i(t_i = 2t_e)/\sigma$ | 0.28 | 0.36 | 0.43 |
| $\rho_i(t_i = 4t_e)/\sigma$ | 0.41 | 0.50 | 0.61 |

**Table 7.1:** The ratio between gyroradius and blob size for electrons and for ions of different temperatures at the left end ($x = 0$), middle ($x = L_x/2$) and right end ($x = L_x$) of the domain.

When the thermal velocity is increased it follows from (2.11) that the gyroradius will also grow larger. As the ion temperature is raised, the ion gyroradius gets closer to the size of the blob, $\sigma$. In table 7.1 the gyroradius of ions with different temperatures, as well as the gyroradius of electrons, is shown for different positions on the grid. In the case of $t_i = 4t_e$ the ion gyroradius is approximately half of the cross-field size of the blob.

When the ion gyroradius is no longer small compared to the size of the blob, the ions are able to *gyrate out of the blob*, as illustrated in figure 7.1, where the case of $t_i = 4t_e$ has been chosen. Even an ion in the middle of the blob will quickly end up at the edges of the blob. This explains the rapid decrease in ion density observed

for the blobs with warm ions. The electron gyroradius was not changed between simulations, and as it is small compared to the blob size, the same diffusion is not expected from the electrons, and neither is it observed.

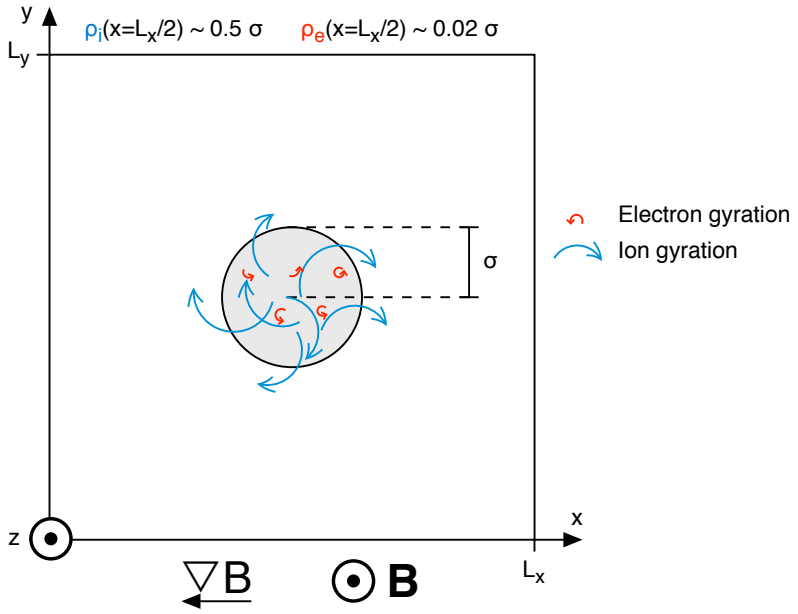


**Figure 7.1:** Illustrations of how ions with a large gyroradius-to-blob size ratio can escape the blob by gyrating out of it, creating a with low ion density.

Additional simulations were conducted to investigate the effect of $\rho_i/\sigma$. The blob cross-field size was doubled from the size of that in the initial simulations to $\sigma = 10\rho_s$, the initial blob amplitude was set to $n_b = 2n_0$ and the ion temperatures $t_i = \{0.1, 0.25, 1, 2, 4\}t_e$ were all tested. The results showed a blob that did not acquire the tadpole shape as much as its counterparts of smaller $\sigma$, and had a smaller poloidal displacement, although still in the positive direction. This strengthens the claim that the loss of ions density in the blob is connected to the ratio between the ion gyroradius and the cross-field size of the blob, $\rho_i/\sigma$.

Figure 6.26 shows that there is a vague ion tail stretching out in the poloidal direction as the blob propagates radially, indicating that there is still a trace of $\mathbf{B} \times \nabla B$ drift in the negative poloidal direction also for the warm ions. The short electron tail of the tadpole blob that can be seen in figure 6.24 is caused by the negative potential from this ion tail. However, the ion tail is not dense enough to act like the ion lobe in the case of cold ions shown in figure 6.22, where the positive electric potential created by the dense ion lob attracts the electrons, giving the electron blob shown in figure 6.21 the mushroom shape. The lack of an ion lobe in the case of warm ions can be explained by the low ion density; the ions diffuse rather than follow the $\mathbf{B} \times \nabla B$ drift to create a lobe.

As already discussed, the large potential difference between the ion and electron lobes can also explain the dissolution of the blob with cold ions. As the blob propagates towards the wall, the ions keep drifting in the negative poloidal and the electrons drift in the positive poloidal direction, while the positive potential in the ion lobe keeps attracting electrons, and the blob is torn apart and dissolves within $250\Omega_i^{-1}$; already at $t = 150\Omega_i^{-1}$ the blob is in several pieces and can hardly be considered a blob anymore.

In the case of $t_i = 4t_e$, as the ions diffuse and no dense ion lobe appears, only a low-density ion tail, there is no tearing drift in the poloidal direction causing the blob to dissolve. This could explain why the warm-ion blob keeps its electron density and tadpole shape even after $250\Omega_i^{-1}$, instead of dissolving like the mushroom blob.

The difference in shape between the blobs of cold and warm ions are in agreement with the results from the gyrofluid simulations with FLR corrections, where the blob was shown to stay dense and keep its shape to a larger degree when the ions were warmer. There is unfortunately no information about the ion density in the previous work, making comparison difficult.

### 7.2.2 Effect on Displacement

Increasing the ion temperature is shown to increase the total displacement of the blob. From the case of $n_b = 2n_0$, the ion temperature is shown to have a small effect on the radial propagation of the blob, with the propagation stopping earlier in this direction for the warm blobs, but with the end position in $x$ at $t = 150\Omega_i^{-1}$ not being affected much by the ion temperature. In the poloidal direction, however, the displacement is significantly increased by increasing the ion temperature, making this the main reason why the total displacement increases with increasing $t_i$. Any effect on the radial displacement seems to come from the velocity shifting from the radial to the poloidal component earlier and to a larger degree as the ion temperatures is raised.

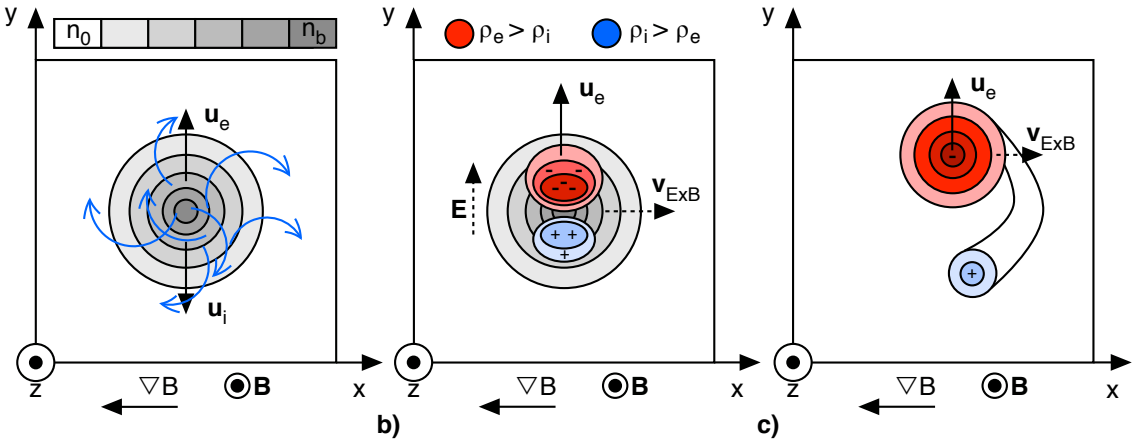


**Figure 7.2:** Illustration of the propagation of a blob with warm ions. Figure **a** shows the ions gyrating out of the blob, leaving a blob with a low ion density. In figure **b** the low density ion lobe is illustrated, and the creation of a weaker electric field leading to a smaller $\mathbf{E} \times \mathbf{B}$ drift, while the $\mathbf{B} \times \nabla B$ drift of the electrons is still significant for the blob propagation. The resulting blob is shown to propagate largely according to the $\mathbf{B} \times \nabla B$ drift of the electrons in figure **c**, as it is now mainly an electron blob, and the ion tail stretching out in the negative poloidal direction is also illustrated here.

The discussed density effects caused by the warm ions can explain the positive poloidal displacement. Because of the significantly reduced ion density in the tadpole blob, it can be expected to behave close to how a pure electron blob would. As electrons according to the $\mathbf{B} \times \nabla B$ force will drift in the *positive* poloidal direction, so will an electron blob, and it follows that a blob consisting largely of electrons would also be expected to drift in this direction. Increasing the ion temperature

decreases the ion density, and so the electron-to-ion density ratio of the blob will be higher as the ion temperature is increased, giving a blob that is to a higher degree moved by the $\mathbf{B} \times \nabla B$ drift of the electrons. The characteristics of the propagation for a blob with warm ions is shown in figure 7.2. This figure can be compared to the propagation shown in figure 3.4, which, as it turns out, shows the dynamics of a blob with cold ions.

### 7.2.3 Effect on Velocity

The ion temperature has a small effect on the maximum velocity in the radial direction, with an apparent distinction between the three blobs with warmest ions and the two with the coldest. The temperature effect on $v_x$ is larger for the denser blobs.

The maximum $x$ component of the velocity is reached earlier as the temperature is increased, indicating that the propagation starts earlier when the blob consists of warmer ions. This could be because the initial charge separation, although not as strong as for colder ions, happens faster due to the large ion gyroradius, thereby starting the $\mathbf{E} \times \mathbf{B}$ drift earlier.

The radial velocity is also seen to decelerate earlier for warmer blobs, leaving the blob to propagate radially for a shorter amount of time when the ions are warmer. As the deceleration of $v_x$ starts, the poloidal velocity of the blob is seen to increase, showing how the velocity of the blob shifts from the $x$ to the $y$ component.

The average velocity is shown to be affected by the ion temperatures in two ways. Firstly, for a blob with a given initial density $n_b$, the average velocity increases as $t_i$ increases. Secondly, the increase in average blob velocity as $n_b$ increases is also affected by the ion temperature through its effect on the blob density.

The theoretical expression in (3.6) predicting a squared scaling with the ion gyroradius-to-blob size ratio implies $v_x \propto t_i$, which can be seen from figure 6.19 and the accompanying errors in table 6.5 to not be the case. A linear scaling with the same ratio would mean a square root scaling with the ion temperature, which is shown to not fit well either. The best fit is shown to be

$$v_x \propto \left(\frac{\rho_i}{\sigma}\right)^{1/3} \tag{7.1}$$

$$\propto t_i^{1/6}. \tag{7.2}$$

## 7.3 Comparisons with Previous Work

There is a large degree of agreement between the results presented in Chapter 6 and those found in previous work presented in Section 3.5. Especially for the blob with the cold ions, which is the one that is most comparable to the gyrofluid simulation without FLR corrections, presented in Section 3.5.4, the agreement is good.

The lifetime of the blob, for example, is found the be $\tau_b \approx 150\Omega_i^{-1} = 2.4 \times 10^{-7}$ s for a blob with $t_i = 0.1t_e$, which is close to what is predicted from earlier work in (3.7) [39]. The average radial velocities found in this thesis are in the range $\sim 0.05\rho_s\Omega_i^{-1}$ to $\sim 0.2\rho_s\Omega_i^{-1}$, which in SI units is $v_x \sim 10^6$ cm/s. This is about one order of magnitude larger than the average velocities found in experiments ([66] [27])

and predicted by theory ([39]) in Section 3.5.1, and could be caused by the increased magnetic gradient or the reduced ion mass, which will both affect the velocities.

The blob consisting of cold ions propagates mainly radially, as expected from the work presented in Section 3.5, and it gets the mushroom shape that has been shown in many previous studies of blob dynamics, and the poloidal division into an ion lobe and an electron lobe, cause a potential difference, is also observed in the results of this thesis.

The simulations where the ion temperature was higher must be compared to the gyrofluid simulations *with* FLR corrections, as presented in Section 3.5.4, and again there is agreement to a large extent. Previous work shows that a blob with warm ions to a larger degree keeps its blob shape and compactness, not dissolving as easily as the cold-ion blob. In electron density plots shown in the three papers with FLR corrections ([32] [44] [63]) the tadpole shape that is found in figure 6.24 of this thesis can even be seen.

The radial as well as the total displacement is found to increase with increasing blob density. This is another result that is in agreement with the FLR-corrected papers [32].

In figure 6.3 it is shown that while the blob is still propagating along the $x$ axis, it has, at any given timestep, reached further for the warmer ions than for the cold ions. This is also in agreement with previous work [32] [63]. However, the same figure shows that the agreement above is true only up until a certain time, when the warm ion blobs are shown to change direction from propagating mainly radially to propagating mainly in the poloidal direction, while the cold ion blobs continue along the $x$ axis and pass the warmer blobs in this dimension.

The change from radial propagation to poloidal propagation is found in all three FLR-corrected gyrofluid simulations reviewed in Section 3.5.4, and the change happens in a similar manner to that presented here. However, the FLR-corrected simulations all show that the poloidal propagation is consistently in the *negative* poloidal direction, with the displacement being larger for higher ion temperatures. This is a mirrored behavior of that found in the simulation results of this thesis, where the blob is shown to propagate in the *positive* poloidal direction, and have a larger positive poloidal displacement as the temperature is increased.

In this thesis it is shown that the ion density of the blob decreases significantly when the ions are warm, and that the decrease in ion density is less pronounced when the ions are colder. This naturally gives a high electron-to-ion density ratio in the blob, leaving a blob that can be compared to an electron blob. As electrons will drift in the *positive* poloidal direction, so will an electron blob, and so will, to a large extent, a blob that consists largely of electrons.

One reason why the blob drifts in the opposite direction in the FLR-corrected gyrofluid simulations could be that the ion gyration is not explicitly modelled. The finite Larmor radius is only accounted for through the averaged effects of the gyration. This way information about the ions is lost, which could lead to an oversight of microscopic properties that end up affecting the macroscopic properties like the COM propagation of the blob.

Observations and measurements presented in Section 3.5.2 seem to find propagation in both poloidal directions, although mostly in the negative poloidal direction. One explanation for this is that the blob follows the background $\mathbf{E} \times \mathbf{B}$ drift, which is in the negative poloidal direction [27]. This drift is not taken into account in

the simulations conducted for this thesis. There is also the possibility that the blob propagates in the negative poloidal direction, but more slowly than the background plasma, which would mean that compared to the background the blob has a velocity in the positive poloidal direction.

## 7.4 Limitations

Two parameters have been exaggerated in these simulations, namely the reduced ion-to-electron mass ratio $m_i/m_e = 100$, the effect of which is looked at in Appendix B, and the increased gradient in the magnetic field, $\nabla B$. In addition, it is possible for the boundary conditions of the domain to introduce numerical effects, especially as the blob moves closer to the edges.

The periodic boundary conditions are shown to give the tadpole blob in figure 6.28 opportunity to keep on propagating as it would in a larger domain, but the COM tracking is no longer correct when the blob crosses the upper boundary at $y = L_y$ and starts moving into the other side of the domain.

### 7.4.1 Effect on Poloidal Propagation

The reduced ion mass leads to an increased thermal ion velocity of $v_{th,i}^{sim} = 4.3 v_{th,i}$ (shown in Chapter 5). This exaggerated ion velocity is likely to enhance the reduction in ion density, giving a blob with a larger electron-to-ion density ratio, which in turn would increase the effect of the $\mathbf{B} \times \nabla B$ drift on the total blob propagation. Although this could cause a stronger effect than what is realistic, it does not explain the consistent increase in displacement in the positive $y$ direction alone, that is shown for blobs of all ion temperatures.

The magnetic gradient is larger than what is usually found in SOLs, which will lead to a stronger $\mathbf{B} \times \nabla B$ drift. This leads to a charge separation happening faster than what is realistic, and an earlier than realistic $\mathbf{E} \times \mathbf{B}$ drift. This is an advantage for the simulation as the number of timesteps needed to observe blob propagation is reduced. In addition to accelerating the propagation process, the strong $\mathbf{B} \times \nabla B$ drift is also likely to tear cold ion blobs apart more quickly, meaning that the end of the propagation, as well as the beginning, is likely to be sped up.

The boundary at $x = L_x$, which acts as the wall, is a Dirichlet boundary where $\phi = 0$ and particles are absorbed. For the case of very fast ions, there is a clear lack of ions in the area close to the wall, creating an electric field in the negative $x$ direction. This could give an $\mathbf{E} \times \mathbf{B}$ drift in the positive $y$ direction of all particles, again enhancing the positive poloidal drift of the tadpole blob. However, the ion plot in figure 6.26 shows that the ions are still affected by the $\mathbf{B} \times \nabla B$ drift in the opposite direction, creating the ion tail, meaning that the electric field is not strong enough to cancel out the negative poloidal drift of the ions. Additionally, the contour lines for the electric potential in figures 6.24 and 6.25 show that the potential is strongest around the blob, not along $L_x$.

# Chapter 8

# Conclusions

The results of this thesis presented in Chapter 6 were shown in Chapter 7 to agree to a large degree with the results from previous work. Even with a reduced ion mass and an enhanced gradient in the magnetic field, the dynamics of the blob proved not only to largely match the scaling predicted in previous work reviewed in Chapter 3, but also to be in range with measured and estimated values like blob lifetime and velocity.

The expected charge separation caused by the cross-field gradient in the magnetic field and the resulting $\mathbf{E} \times \mathbf{B}$ field can clearly be observed, and the blob acquires shapes that are shown in previous work: the mushroom shape when the ions are cold, and the tadpole shape when the ions are warm.

The radial velocity of the blob is shown to scale as

$$v_x \propto t_i^{1/6} n_b \tag{8.1}$$

while the maximum velocity is close to the square root scaling suggested from previous simulations,

$$v_{x,max} \propto \sqrt{\frac{n_b}{n_0}}. \tag{8.2}$$

However, there are also differences, and the most striking one is the mirrored propagation in the poloidal direction. In the gyrofluid simulations with FLR corrections, the blob is shown to propagate increasingly in the negative direction as the ion temperature is raised, while the exact opposite is shown in the results of this thesis, where the blob propagates in the positive poloidal direction.

None of the FLR-corrected simulation papers present the ion density evolution, which is shown in this thesis to decrease rapidly when the ions are warm. If the ion density of the blob is low, it follows naturally that it should follow the $\mathbf{B} \times \nabla B$ drift of the electrons, as the blob is mostly made up of electrons.

In this thesis it is proposed that the rapid decrease of ions in the blob is caused by the large gyroradius of the warm ions compared to the cross-field size of the blob, which lets the ions "escape" the blob. The claim that the ratio between the ion gyroradius and the blob size affects the ion loss in the blob was supported by additional simulations where the cross-field size of the blob was doubled while the same ion temperatures as in the initial simulations were used, giving the same ion gyroradii as before, which showed poloidal propagation to a lesser extent, but still in the positive direction.

As fluid simulations are outside of the scope of this thesis, no attempt to explain the negative poloidal blob propagation of the gyrofluid simulations is made, but it is speculated that only taking the average effects of the gyration of the ions leads to a loss of significant information which affects the blob propagation.

In conclusion, even if fluid simulations can reproduce the blob dynamics to a large degree, and even if attempts at correcting for kinetic effects have proven to be highly successful in many applications, there is still value to be added by investigating the blob dynamics through simulations on the fundamental kinetic level. In the same way that combining theory and experiments can advance the knowledge of a field, fluid and particle simulations can be used to investigate different aspects of blob dynamics, and by playing on their respective strengths together advance the knowledge of plasma blobs.

# Chapter 9

# Suggestions For Further Work

The poloidal drift of the blob can be investigated further by conducting simulations where the ion temperature is high and the ion mass is realistic, to check if the blob dynamics are affected by the small ion-to-electron mass ratio, as is shown for $t_i = 0.25t_e$ in Appendix B. Another improvement that should be tried is to make the simulations box larger without changing the size of the blob, to see if the blob dynamics shown in this thesis were affected by edge effects. This can also enable a longer tracking of the warm ion blob that did not dissolve within the simulations run for this thesis, but rather crossed the periodic boundary at $y = L_y$, making the COM tracking useless after $\sim 150\Omega_i^{-1}$.

Simulations of a blob consisting of only electrons can be tested to find the expected propagation of an electron blob, and the initial ion density of the blob, which in this thesis was the same as the initial electron density in the blob, can be varied between 0 and $n_{b,e}$ to see how the electron-to-ion density ratio of the blob affects the poloidal propagation.

If the blob is said to drift poloidally in the same direction as the background $\mathbf{E} \times \mathbf{B}$ drift [27], it could be of interest to add an initial particle drift in the negative poloidal direction.

In this thesis, the velocity distribution was the same in the blob as outside of it. However, blobs have been shown to have ions that are up to 4 times warmer than those in the background plasma [37]. Having a temperature distribution of the same shape as the particle density distribution, giving the particles in the blob higher velocities than those in the background plasma, could prove interesting and possibly change the results.

To account for parallel effects on the blob propagation, 3D PIC simulations should be conducted. This way the claim that the parallel position of the blob affect the direction of poloidal displacement [55] can also be investigated. One way to reduce the number of superparticles needed and make the simulations less computationally expensive could be to use the "blob" method that was briefly introduced in Section 4.3.

Lastly, if the positive poloidal drift of the blob persists in particle simulations, an attempt to unify the results of this thesis with those of the FLR-corrected gyrofluid simulations should be made, if possible.

# Appendices

# Appendix A

# The Code

The C code for the main functions developed by the author for this thesis is included here. Some smaller changes various places in the code are not included. Explanatory comments from the header files have been added.

## A.1 Gaussian Particle Density Distribution

```
/**
 * @brief     Assign particles to Gaussian distributed
 *            positions
 * @param              ini         Dictionary to input file
 * @param[in,out]      pop         Population of particles
 * @param              grid        Grid in which the particles
 *                                 are to be distributed
 * @param              rngSync     Synchronized random number
 *                                 generator
 * @return             void
 *
 * Using Monte Carlo Rejection Sampling to fit nParticles
 * to the shape defined by pBlobPDF2D. This is the
 * particle density distribution that includes both the
 * background and the blob. The blob is a Gaussian
 * distribution centered at mu (middle of domain) and
 * with a standard deviation sigma = 1, which is the size
 * of the blob.
 *
 */

void pPosBlobMCRS(const dictionary *ini, Population *pop,
   const MpiInfo *mpiInfo, const gsl_rng *rng){
    int nSpecies = pop->nSpecies;
    int nDims = pop->nDims;
    long int *nParticles = iniGetLongIntArr(ini,
       "population:nParticles", nSpecies);
    double nbn0 = iniGetDouble(ini, "blob:ratioBlobBkg");
```

```
int *trueSize = iniGetIntArr(ini, "grid:trueSize",
    nDims);

int *subdomain = mpiInfo->subdomain;
double *posToSubdomain = mpiInfo->posToSubdomain;

int *L = gGetGlobalSize(ini);

// Maximum density:
double maxPDF = 1 + nbn0;

for(int s=0;s<nSpecies;s++){
    long int iStart = pop->iStart[s];
    long int iStop = iStart;
    double *pos = &pop->pos[iStart*nDims];

    for(long int i=0;i<nParticles[s];i++){
        // Generate position for particle i
        double k = maxPDF+1;
        double x = 0;
        double y = 0;

        while(pBlobPDF2D(x,y,nbn0,*L)/maxPDF < k){
            x = L[0]*gsl_rng_uniform_pos(rng);
            y = L[1]*gsl_rng_uniform_pos(rng);
            k = gsl_rng_uniform_pos(rng);
        }
        pos[0] = x;
        pos[1] = y;

        // Count the number of dimensions where the
           particle resides in
        // the range of this node
        int correctRange = 0;
        for(int d=0;d<nDims;d++)
            correctRange += (subdomain[d] ==
                (int)(posToSubdomain[d]*pos[d]));

        // Iterate only if particle resides in this
           sub-domain.
        if(correctRange==nDims){
            pos += nDims;
            iStop++;
        }
    }

    if(iStop>pop->iStart[s+1]){
        int allocated = pop->iStart[s+1]-iStart;
```

```
            int generated = iStop-iStart;
            msg(ERROR, "allocated only %i particles of
                specie %i per node but"
                 "%i generated", allocated, s, generated);
        }

        pop->iStop[s]=iStop;
    }

    pToLocalFrame(pop,mpiInfo);

    free(L);
    free(nParticles);
    free(trueSize);
}

/*****************************
 * DEFINING LOCAL FUNCTIONS
 *****************************/

double pBlobPDF2D(double x, double y, double nbn0, int L){
    double sigma = L/15;
    double mux = L/2;
    double muy = L/2;

    double xNum = -pow((x-mux),2);
    double yNum = -pow((y-muy),2);
    double denom = 2*pow(sigma,2);

    double pdf = 1 + nbn0*exp(xNum/denom + yNum/denom);

    return pdf;

}
```

## A.2 2D Boris Solver For Inhomogeneous Magnetic Field

```
/**
 * @param[in,out]     pop         Population
 * @param             E           Electric field
 * @param             L2          Constant used for the gradient
 * @param             B0          Magnetic field strength
 * @return                        void
 *
 * The 2D Inhomogeneous method (puBorisInh2D1KE) assumes
 * that the particles move in the x-y plane while the
 * magnetic field is along the z axis and changes as:
 *          B(d) = 2*Lx*B0/(2*Lx+x)
 * where x is the dimension of grad B. Set the input as
 * BExt != 0 in the dimension you want to magnetic field
 * to point in, with the value being B0:
 *  - BExt = 0, 0, 0.5
 * This opens the possibility to change the function to
 * work for any dimension.
 *
 * The constant L2 = 2*Lx is calculated outside of this
 * function, and B0 is given in the input file.
 *
*/

void puBorisInh2D1KE(Population *pop, Grid *E, const int
   L2, double B0){
    int nDims = 2;
    int nSpecies = pop->nSpecies;
    double *pos = pop->pos;
    double *vel = pop->vel;
    double *kinEnergy = pop-> kinEnergy;

    // All of this is needed to compute S and T for every
       particle
    double *charge = pop->charge;
    double *mass = pop->mass;
    double halfTimeStep = 0.5;
    int gradDim = 0;
    double L2B0 = (double) L2*B0;

    // This is for puInterpND1:
    int *integer = malloc(nDims*sizeof(*integer));
    double *decimal = malloc(nDims*sizeof(*decimal));
    double *complement = malloc(nDims*sizeof(*complement));
    long int *sizeProd = E->sizeProd;
    double *val = E->val;
```

```
    for(int s=0;s<nSpecies;s++){
        long int pStart = pop->iStart[s]*nDims;
        long int pStop = pop->iStop[s]*nDims;

        gMul(E, pop->charge[s]/pop->mass[s]);
        kinEnergy[s] = 0;
        double factor = halfTimeStep*charge[s]/mass[s];

        for(long int p=pStart;p<pStop;p+=nDims){
            double dv[2], vPlus[2], vMinus[2];
            puInterpND1(dv, &pos[p], val, sizeProd, nDims,
                integer, decimal, complement);

            // B, T and S must be calculated for every
                particle:
            double B = L2B0/(L2+pos[p+gradDim]);
            double T = factor*B;
            double S = 2*T/(1+pow(T,2));

            // Add half the acceleration (becomes vMinus)
            for(int d=0; d<nDims; d++) {vMinus[d] =
                (vel[p+d]+0.5*dv[d]);}

            // Rotate
            double vPrime = vMinus[0] + vMinus[1]*T; //
                Compute vPrime
            vPlus[1] = (vMinus[1] - vPrime*S);   // Compute
                vPlus Y
            vPlus[0] = (vPrime + vPlus[1]*T);    // Compute
                vPlus X

            // Compute Energy
            double velSquared = 0;
            for(int d=0;d<nDims;d++){
                velSquared+=pow(vel[p+d],2);
            }
            kinEnergy[s] += velSquared;

            // Add half the acceleration
            for(int
                d=0;d<nDims;d++){vel[p+d]=vPlus[d]+0.5*dv[d];}
        }
        kinEnergy[s]*=0.5*mass[s];
        // Specie specific renomalization
        gMul(E, pop->mass[s]/pop->charge[s]);
    }
}
```

## A.3 Particle Boundary Conditions

```
/**
 * @brief Absorbing and reflecting particle boundaries
 * @param       pop             Population
 * @param       mpiInfo
 * @param       grid
 * @return      void
 *
 * If the boundary conditions for particles is set to
 * either absorbing or reflecting, this function makes
 * sure they are moved accordingly. Allows mixed
 * boundaries and several dimensions. Input should be
 * given as [x_min, y, min, ..., x_max, ...]. Example:
 * [ABSORBING, PERIODIC, REFLECTING, PERIODIC].
 */

void puParticleBnd(const dictionary *ini, Population *pop,
   const MpiInfo *mpiInfo, Grid *grid){
    int nSpecies = pop->nSpecies;
    int nDims = pop->nDims;
    int *subdomain = mpiInfo->subdomain;
    int *nSubdomains = mpiInfo->nSubdomains;
    int *nGhostLayers = grid->nGhostLayers;
    int rank = grid->rank;
    int *size = grid->size;
    bndType *pBnd = pop->bndP;

    for(int d=0; d<nDims; d++){
        bndType minBnd = pBnd[d+1];
        bndType maxBnd = pBnd[d+1+nDims+1];

        double ld = (double) (nGhostLayers[d+1]);
        double ud = (double)
           (size[d+1]-1-nGhostLayers[d+1+rank]);

        //Lower edge:
        if (minBnd==PERIODIC){
            //PERIODIC, do nothing, will move on to
               periodic handling
        }
        else{
            if(subdomain[d]==0){
                for(int s=0; s<nSpecies;s++){
                    long int iStart = pop->iStart[s];
                    long int iStop = pop->iStop[s];
                    for(long int i=iStart;i<iStop;i++){
                        double *iPos = &pop->pos[i*nDims];
```

```
                                double *iVel = &pop->vel[i*nDims];
                                double dPos = iPos[d];
                                if(dPos < ld){
                                    if(minBnd == REFLECTING){
                                        //REFLECTIVE
                                        *(iPos+d)= 2*ld-dPos;
                                        *(iVel+d)= -*(iVel+d);
                                    }
                                    else if(minBnd == ABSORBING){
                                        //ABSORBING
                                        pCut(pop,s,i*nDims,iPos,iVel);
                                    }
                                }
                            }
                        }
                    }
                }
                // Upper edge
                if(maxBnd==PERIODIC){
                    //PERIODIC, do nothing, will move on to
                       periodic handling
                }
                else{
                    if(subdomain[d]==nSubdomains[d]-1){
                        for(int s=0; s<nSpecies;s++){
                            long int iStart = pop->iStart[s];
                            long int iStop = pop->iStop[s];
                            for(long int i=iStart;i<iStop;i++){
                                double *iPos = &pop->pos[i*nDims];
                                double *iVel = &pop->vel[i*nDims];
                                double dPos = iPos[d];
                                if(dPos > ud){
                                    if(maxBnd == REFLECTING){
                                        //REFLECTIVE
                                        *(iPos+d)= 2*ud-dPos;
                                        *(iVel+d)= -*(iVel+d);
                                    }
                                    else if(maxBnd == ABSORBING){
                                        //ABSORBING
                                        pCut(pop,s,i*nDims,iPos,iVel);
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

## A.4 Blob COM Tracking

```
/**
 * @brief Creates datasets in .xy.h5-file for storing COM
 *        position
 * @param     xy          .xy.h5-identifier
 * @param     pop          Population
 * @return               void
 * Uses xyCreateDataset() to create datasets for the COM
 * position, one for each dimension i. Can be edited to
 * be used for both species:
 *
 *  - /CentreOfMass/specie 0/dim i
 *  - /CentreOfMass/specie 1/dim i
 *
 * pWriteCentreOfMass() can be used to populate these
 * datasets.
 */

void pCreateCentreOfMassDatasets(hid_t xy, Population *pop){

    char name[64];
    int nDims = pop->nDims;

    for(int d=0;d<nDims;d++){

        sprintf(name,"/CentreOfMass/specie 0/dim %i",d);
        xyCreateDataset(xy,name);

        sprintf(name,"/MaxDensityAmplitude/specie 0/dim
           %i",d);
        xyCreateDataset(xy,name);
    }
}


/**
 * @brief Writes COM position to .xy.h5-file
 * @param     xy          .xy.h5-identifier
 * @param     pop          Population
 * @return               void
 *
 * Uses xyWrite() to write COM position stored in
 * Population to .xy.h5 datasets. These datasets must
 * first be created using pCreateCentreOfMassDatasets().
 * For now only specie 0 is tracked.
 *
 */
```

```
void pWriteCentreOfMass(hid_t xy, Population *pop, double
    x){

    char name[64];
    int nDims = pop->nDims;

    for(int d=0;d<nDims;d++){

        sprintf(name,"/CentreOfMass/specie 0/dim %i",d);
        xyWrite(xy,name,x,pop->centreofmass[d],MPI_SUM);

        sprintf(name,"/MaxDensityAmplitude/specie 0/dim
            %i",d);
        xyWrite(xy,name,x,pop->maxElectronAmplitude[d],MPI_SUM);

    }

}

/**
 * @brief
 * @param     pop          Population
 * @param     rho_e        Electron charge density
 * @return              void
 *
 * Calculates the center of mass of the area with
 * density larger than a certain value, which is
 * based on the particle distribution having an initial
 * amplitude. Used for tracking the center of mass of
 * a plasma blob. Currently only tracks the electron
 * COM. The electron charge density is calculated in
 * puDistrND1electrons().
 *
 */

void pCentreOfMass(const Population *pop, Grid *rho_e){

    int nDims = pop->nDims;
    double *val = rho_e->val;
    long int *sizeProd = rho_e->sizeProd;
    int n = pop->nParticles;
    float nbn0 = pop->MaxDensityAmplitude;
    double bkg = n*(1+0.1*nbn0);

    int *size = rho_e->size;

    double totalmass = 0;
    double weightedpos[2] = {0,0};
```

```
    double rho_max = 0;
    double rhomaxpos[2] = {0,0};

    for(int j=0; j<size[1]; j++){
        for(int k=0; k<size[2];k++){
            long int p = j + k*sizeProd[nDims]; //x = i, y
               = j
            long int pj = p + 1;                //x = i +
               1, y = j
            long int pk = p + sizeProd[nDims];  //x = i, y
               = j + 1
            long int pjk = pk + 1;              //x = i +
               1, y = j + 1

            double rho_jk =
               -(val[p]+val[pj]+val[pk]+val[pjk])/4;
            double rho_blob = fmax(rho_jk-bkg, 0);
            totalmass += rho_blob;

            weightedpos[0] += rho_blob*(j+0.5);
            weightedpos[1] += rho_blob*(k+0.5);

            if(rho_jk>rho_max){
                rho_max = rho_jk;

                rhomaxpos[0] = j+0.5;
                rhomaxpos[1] = k+0.5;
            }
        }
    }

    for(int d=0;d<nDims;d++){
        pop->centreofmass[d] = weightedpos[d]/totalmass;
        pop->maxElectronAmplitude[d] = rhomaxpos[d];
    }

}
```

# Appendix B

# Reduced vs Full Ion Mass

For the simulations that lead to the results presented in this thesis, a reduced ion-to-electron mass ratio of $m_i/m_e = 100$ was used. Before settling for this, a simulation with the full ion-to-electron mass ratio was carried through for comparison, as part of an attempt to reproduce [31]. The ion temperature was kept at $t_i = 0.25t_e$ in both simulations, and with $t_e \sim 25eV$ the electron thermal velocity was $v_{th,e} = 2.1 \times 10^6$ m/s, giving an ion thermal velocity $v_{th,i} = 2.5 \times 10^4$ m/s. Both simulations were run for 20000 timesteps, with the time step being $dt = 2.5 \times 10^{-3}\Omega_i^{-1}$. As $\Omega_i^{-1}$ depends on the ion mass (2.12), the simulation time in seconds varied.

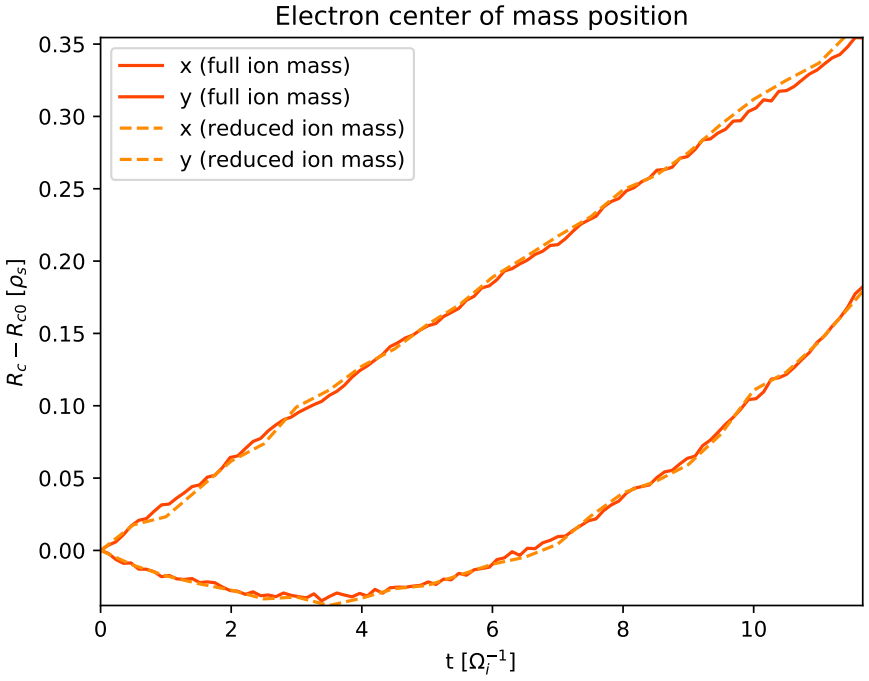


**Figure B.1:** Comparison of the COM displacement of a blob in $x$ and $y$ for the reduced mass ratio $m_i/m_e = 100$ and the full mass ratio $m_i/m_e = 1836$.

In figure B.1 the results of the two simulations are plotted together. The dimensions are scaled based on the respective values for $\Omega_i^{-1}$ and $\rho_s$. The axes are limited to show only the part of the simulation that were within the time frame of the full ion mass simulation.

In conclusion, it seems like reducing the ion mass leads to a scaling of the time, but that the physics, at least in the initial period and for an ion temperature $t_i = 0.25t_e$, is kept intact, meaning that reducing the ion mass is a way to reduced the number of timesteps in the simulation without losing the physics.

# Bibliography


[1] J. Adámek et al. "Simultaneous Measurements of Ion Temperature by Segmented Tunnel and Katsumata Probe". In: *Contributions to Plasma Physics* 48.5 (July 2008), pp. 395–399. ISSN: 08631042, 15213986. DOI: 10.1002/ctpp.200810063.

[2] M. A. Beer and G. W. Hammett. "Toroidal gyrofluid equations for simulations of tokamak turbulence". In: *Physics of Plasmas* 3.11 (Nov. 1996), pp. 4046–4064. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.871538.

[3] Charles K. Birdsall and A. Bruce Langdon. *Plasma physics via computer simulation*. New York: McGraw-Hill, 1985. 479 pp. ISBN: 978-0-07-005371-7.

[4] N. Bisai et al. "Formation of a density blob and its dynamics in the edge and the scrape-off layer of a tokamak plasma". In: *Physics of Plasmas* 12.10 (Oct. 2005), p. 102515. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.2083791.

[5] J. A. Boedo et al. "Transport by intermittency in the boundary of the DIII-D tokamak". In: *Physics of Plasmas* 10.5 (May 2003), pp. 1670–1677. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.1563259.

[6] E. O. Brigham and R. E. Morrow. "The fast Fourier transform". In: *IEEE Spectrum* 4.12 (Dec. 1967), pp. 63–70. ISSN: 0018-9235. DOI: 10.1109/MSPEC.1967.5217220.

[7] J. C. Butcher. "Random Sampling from the Normal Distribution". In: *The Computer Journal* 3.4 (Apr. 1, 1961), pp. 251–253. ISSN: 0010-4620, 1460-2067. DOI: 10.1093/comjnl/3.4.251.

[8] I. L. Caldas et al. "Control of chaotic magnetic fields in tokamaks". In: *Brazilian Journal of Physics* 32.4 (Dec. 2002). ISSN: 0103-9733. DOI: 10.1590/S0103-97332002000500023.

[9] Francis F. Chen. *Introduction to plasma physics and controlled fusion*. 2nd ed. New York: Plenum Press, 1984. 1 p. ISBN: 978-0-306-41332-2.

[10] G.G.M. Coppa et al. "Blob Method for Kinetic Plasma Simulation with Variable-Size Particles". In: *Journal of Computational Physics* 127.2 (Sept. 1996), pp. 268–284. ISSN: 00219991. DOI: 10.1006/jcph.1996.0174.

[11] Stefan Costea et al. "Particle-In-Cell simulation of parallel blob dynamics in near scrape-off-layer plasma of medium-size tokamak". In: 44th EPS Conference on Plasma Physics. June 30, 2017.

[12] R. Courant, K. Friedrichs, and H. Lewy. "On the Partial Difference Equations of Mathematical Physics". In: *IBM Journal of Research and Development* 11.2 (Mar. 1967), pp. 215–234. ISSN: 0018-8646, 0018-8646. DOI: 10.1147/rd.112.0215.

[13] Richard O. Dendy, ed. *Plasma physics: an introductory course*. Repr. OCLC: 833071360. Cambridge: Cambridge Univ. Press, 1996. 513 pp. ISBN: 978-0-521-43309-9.

[14] D. A. D'Ippolito, J. R. Myra, and S. I. Krasheninnikov. "Cross-field blob transport in tokamak scrape-off-layer plasmas". In: *Physics of Plasmas* 9.1 (Jan. 2002), pp. 222–233. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.1426394.

[15] D. A. D'Ippolito, J. R. Myra, and S. J. Zweben. "Convective transport by intermittent blob-filaments: Comparison of theory and experiment". In: *Physics of Plasmas* 18.6 (June 2011), p. 060501. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.3594609.

[16] W. Dorland and G. W. Hammett. "Gyrofluid turbulence models with kinetic effects". In: *Physics of Fluids B: Plasma Physics* 5.3 (Mar. 1993), pp. 812–835. ISSN: 0899-8221. DOI: 10.1063/1.860934.

[17] L. Easy et al. "Three dimensional simulations of plasma filaments in the scrape off layer: A comparison with models of reduced dimensionality". In: *Physics of Plasmas* 21.12 (Dec. 2014), p. 122515. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.4904207.

[18] S Elmore et al. "Upstream and divertor ion temperature measurements on MAST by retarding field energy analyser". In: *Plasma Physics and Controlled Fusion* 54.6 (June 1, 2012), p. 065001. ISSN: 0741-3335, 1361-6587. DOI: 10.1088/0741-3335/54/6/065001.

[19] M. Endler. "Turbulent SOL transport in stellarators and tokamaks". In: *Journal of Nuclear Materials* 266-269 (Mar. 1999), pp. 84–90. ISSN: 00223115. DOI: 10.1016/S0022-3115(98)00659-X.

[20] M Endler et al. "Measurements and modelling of electrostatic fluctuations in the scrape-off layer of ASDEX". In: *Nuclear Fusion* 35.11 (Nov. 1995), pp. 1307–1339. ISSN: 0029-5515. DOI: 10.1088/0029-5515/35/11/I01.

[21] E. A. Frieman and L. Chen. "Nonlinear gyrokinetic equations for low-frequency electromagnetic waves in general plasma equilibria". In: *Physics of Fluids* 25.3 (1982), p. 502. ISSN: 00319171. DOI: 10.1063/1.863762.

[22] O. E. Garcia. "Blob Transport in the Plasma Edge: a Review". In: *Plasma and Fusion Research* 4 (2009), pp. 019–019. ISSN: 1880-6821. DOI: 10.1585/pfr.4.019.

[23] O. E. Garcia et al. "Mechanism and scaling for convection of isolated structures in nonuniformly magnetized plasmas". In: *Physics of Plasmas* 12.9 (Sept. 2005), p. 090701. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.2044487.

[24] O E Garcia et al. "Turbulence simulations of blob formation and radial propagation in toroidally magnetized plasmas". In: *Physica Scripta* T122 (Feb. 2006), pp. 89–103. ISSN: 0031-8949, 1402-4896. DOI: 10.1088/0031-8949/2006/T122/013.

[25] J. P. Goedbloed and Stefaan Poedts. *Principles of magnetohydrodynamics: with applications to laboratory and astrophysical plasmas*. Cambridge, UK ; New York: Cambridge University Press, 2004. 613 pp. ISBN: 978-0-521-62607-1.

[26] R. J. Goldston and P. H. Rutherford. *Introduction to plasma physics*. Bristol, UK ; Philadelphia: Institute of Physics Pub, 1995. 491 pp. ISBN: 978-0-7503-0183-1.

[27] O. Grulke et al. "Radially propagating fluctuation structures in the scrape-off layer of Alcator C-Mod". In: *Physics of Plasmas* 13.1 (Jan. 2006), p. 012306. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.2164991.

[28] Donald A. Gurnett and A. Bhattacharjee. *Introduction to plasma physics: with space and laboratory applications*. OCLC: ocm53993021. Cambridge, UK ; New York: Cambridge University Press, 2005. 452 pp. ISBN: 978-0-521-36483-6.

[29] M. O. Hagler and M. Kristiansen. *An introduction to controlled thermonuclear fusion*. Lexington, Mass: Lexington Books, 1977. 188 pp. ISBN: 978-0-669-99119-2.

[30] Hiroki Hasegawa and Seiji Ishiguro. "Development and Verification of the Three-Dimensional Electrostatic Particle Simulation Code for the Study of Blob and Hole Propagation Dynamics". In: *Plasma and Fusion Research* 12.0 (2017), pp. 1401044–1401044. ISSN: 1880-6821. DOI: 10.1585/pfr.12.1401044.

[31] Hiroki Hasegawa and Seiji Ishiguro. "Study of self-consistent particle flows in a plasma blob with particle-in-cell simulations". In: *Physics of Plasmas* 22.10 (Oct. 2015), p. 102113. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.4933359.

[32] M. Held et al. "The influence of temperature dynamics and dynamic finite ion Larmor radius effects on seeded high amplitude plasma blobs". In: *Nuclear Fusion* 56.12 (Dec. 1, 2016), p. 126005. ISSN: 0029-5515, 1741-4326. DOI: 10.1088/0029-5515/56/12/126005.

[33] Roger W. Hockney and James W. Eastwood. *Computer simulation using particles*. Special student ed. Bristol [England] ; Philadelphia: A. Hilger, 1988. 540 pp. ISBN: 978-0-85274-392-8.

[34] Seiji Ishiguro and Hiroki Hasegwa. "Three-dimensional particle-in-cell simulation of scrape-off-layer plasmas". In: *Journal of Plasma Physics* 72.6 (Dec. 2006), p. 1233. ISSN: 0022-3778, 1469-7807. DOI: 10.1017/S0022377806006003.

[35] D. Jovanović, P. K. Shukla, and F. Pegoraro. "Effects of the parallel electron dynamics and finite ion temperature on the plasma blob propagation in the scrape-off layer". In: *Physics of Plasmas* 15.11 (Nov. 2008), p. 112305. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.3008050.

[36] Gullik Vetvik Killie. "A Parallel Multigrid Poisson Solver for PINC, a new Particle-in-Cell Model". Master thesis. Oslo: University of Oslo, 2016.

[37] M Kočan et al. "Ion temperature fluctuations in the ASDEX Upgrade scrape-off layer". In: *Plasma Physics and Controlled Fusion* 54.8 (Aug. 1, 2012), p. 085009. ISSN: 0741-3335, 1361-6587. DOI: 10.1088/0741-3335/54/8/085009.

[38] S Krasheninnikov, D Ryutov, and G Yu. "Large plasma pressure perturbations and radial convective transport in a tokamak". In: *Journal of Plasma and Fusion Research* 6 (Feb. 4, 2004).

[39] S.I. Krasheninnikov. "On scrape off layer plasma transport". In: *Physics Letters A* 283.5 (May 2001), pp. 368–370. ISSN: 03759601. DOI: 10.1016/S0375-9601(01)00252-3.

[40] R. Kube, O. E. Garcia, and M. Wiesenberger. "Amplitude and size scaling for interchange motions of plasma filaments". In: *Physics of Plasmas* 23.12 (Dec. 2016), p. 122302. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.4971220.

[41] Giovanni Lapenta. "Particle simulations of space weather". In: *Journal of Computational Physics* 231.3 (Feb. 2012), pp. 795–821. ISSN: 00219991. DOI: 10.1016/j.jcp.2011.03.035.

[42] B. Lipschultz et al. "Plasma–surface interaction, scrape-off layer and divertor physics: implications for ITER". In: *Nuclear Fusion* 47.9 (Sept. 2007), pp. 1189–1205. ISSN: 0029-5515, 1741-4326. DOI: 10.1088/0029-5515/47/9/016.

[43] A Loarte et al. "Chapter 4: Power and particle control". In: *Nuclear Fusion* 47.6 (June 2007), S203–S263. ISSN: 0029-5515, 1741-4326. DOI: 10.1088/0029-5515/47/6/S04.

[44] Jens Madsen et al. "The influence of finite Larmor radius effects on the radial interchange motions of plasma filaments". In: *Physics of Plasmas* 18.11 (Nov. 2011), p. 112504. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.3658033.

[45] R.J. Maqueda, D.P. Stotler, and S.J. Zweben. "Intermittency in the scrape-off layer of the National Spherical Torus Experiment during H-mode confinement". In: *Journal of Nuclear Materials* 415.1 (Aug. 2011), S459–S462. ISSN: 00223115. DOI: 10.1016/j.jnucmat.2010.11.002.

[46] David C. Montgomery and Derek A. Tidman. *Plasma Kinetic Theory*. McGraw-Hill Advanced Physics Monograph Series. New York: McGraw-Hill, 1964.

[47] J. R. Myra, D. A. Russell, and D. A. D'Ippolito. "Transport of perpendicular edge momentum by drift-interchange turbulence and blobs". In: *Physics of Plasmas* 15.3 (Mar. 2008), p. 032304. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.2889419.

[48] J.R. Myra et al. "Blob transport models, experiments, and the Accretion Theory of spontaneous rotation". In: 21st IAEA Fusion Energy Conference. International Atomic Energy Agency (IAEA): IAEA, 2007.

[49] Hideo Okuda. "Nonphysical noises and instabilities in plasma simulation due to a spatial grid". In: *Journal of Computational Physics* 10.3 (Dec. 1972), pp. 475–486. ISSN: 00219991. DOI: 10.1016/0021-9991(72)90048-4.

[50] P C Stangeby. "A tutorial on some basic aspects of divertor physics". In: *Plasma Physics and Controlled Fusion* 42.12 (2000), B271. ISSN: 0741-3335.

[51] Hans L. Pécseli. *Waves and oscillations in plasmas*. Series in plasma physics. Boca Raton: CRC Press, Taylor & Francis Group, 2013. 553 pp. ISBN: 978-1-4398-7848-4.

[52] Gerd W. Prolss. *Physics of the Earth's space environment: an introduction*. Berlin ; New York: Springer, 2004. 513 pp. ISBN: 978-3-540-21426-7.

[53] D. A. Russell et al. "Comparison of scrape-off layer turbulence simulations with experiments using a synthetic gas puff imaging diagnostic". In: *Physics of Plasmas* 18.2 (Feb. 2011), p. 022306. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.3553024.

[54] P. H. Rutherford and E. A. Frieman. "Drift Instabilities in General Magnetic Field Configurations". In: *Physics of Fluids* 11.3 (1968), p. 569. ISSN: 00319171. DOI: 10.1063/1.1691954.

[55] D. D. Ryutov. "The dynamics of an isolated plasma filament at the edge of a toroidal device". In: *Physics of Plasmas* 13.12 (Dec. 2006), p. 122307. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.2403092.

[56] J. Kenneth Shultis and Richard E. Faw. *Fundamentals of nuclear science and engineering*. 2nd ed. OCLC: ocn140108889. Boca Raton: CRC Press, 2008. 591 pp. ISBN: 978-1-4200-5135-3.

[57] P. C. Stangeby. *The plasma boundary of magnetic fusion devices*. Plasma physics series. Bristol ; Philadelphia: Institute of Physics Pub, 2000. 717 pp. ISBN: 978-0-7503-0559-4.

[58] J B Taylor and R J Hastie. "Stability of general plasma equilibria - I formal theory". In: *Plasma Physics* 10.5 (Jan. 1, 1968), pp. 479–494. ISSN: 0032-1028. DOI: 10.1088/0032-1028/10/5/301.

[59] F. Trivellato and M. Raciti Castelli. "On the Courant–Friedrichs–Lewy criterion of rotating grids in 2D vertical-axis wind turbine analysis". In: *Renewable Energy* 62 (Feb. 2014), pp. 53–62. ISSN: 09601481. DOI: 10.1016/j.renene.2013.06.022.

[60] D. Tskhakaya et al. "The Particle-In-Cell Method". In: *Contributions to Plasma Physics* 47.8 (Dec. 2007), pp. 563–594. ISSN: 08631042, 15213986. DOI: 10.1002/ctpp.200710072.

[61] K. Uehara et al. "Measurements of ion temperature and flow velocity using symmetric and asymmetric double probes in the boundary plasma of the JFT-2M tokamak". In: *Nuclear Fusion* 38.11 (1998), p. 1665.

[62] M. V. Umansky et al. "Comments on particle and energy balance in the edge plasma of Alcator C-Mod". In: *Physics of Plasmas* 5.9 (Sept. 1998), pp. 3373–3376. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.873051.

[63] M. Wiesenberger, J. Madsen, and A. Kendl. "Radial convection of finite ion temperature, high amplitude plasma blobs". In: *Physics of Plasmas* 21.9 (Sept. 2014), p. 092301. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.4894220.

[64] M. Wiesenberger et al. "Unified transport scaling laws for plasma blobs and depletions". In: *Physics of Plasmas* 24.6 (June 2017), p. 064502. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.4985318.

[65] G. Q. Yu and S. I. Krasheninnikov. "Dynamics of blobs in scrape-off-layer/shadow regions of tokamaks and linear devices". In: *Physics of Plasmas* 10.11 (Nov. 2003), pp. 4413–4418. ISSN: 1070-664X, 1089-7674. DOI: 10.1063/1.1616937.

[66] S. J. Zweben. "Search for coherent structure within tokamak plasma turbulence". In: *Physics of Fluids* 28.3 (1985), p. 974. ISSN: 00319171. DOI: 10.1063/1.865069.

[67] S J Zweben et al. "Edge turbulence measurements in toroidal fusion devices". In: *Plasma Physics and Controlled Fusion* 49.7 (July 1, 2007), S1–S23. ISSN: 0741-3335, 1361-6587. DOI: 10.1088/0741-3335/49/7/S01.