



THESIS TITLE

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy by

Author Name

June 2017

Contents

Notations	vii
Preface	ix
Abstract	xi
Acknowledgements	xiii
1 Introduction	1
2 A Chapter	3
3 Particle In cell	5
3.0.1 General Method	7
Calculating the Density	9
Calculating the Potential	10
Calculating the Electric Field	12
The Particle Mover	13
Stability Conditions	14
Consequences of Using a Grid	15
Particle Injection	15
4 A Chapter	21
5 A Chapter	23
6 A Chapter	25
Bibliography	27

Illustrations

List of Figures

List of Tables

Notations

The following notations and abbreviations are found throughout this thesis:

Preface

This thesis is primarily my own work. The sources of other materials are identified.

Abstract

Proofs are given for some results.

Acknowledgements

I would like to thank people.

Chapter 1

Introduction

” Perhaps one of the strongest motivations for pursuing plasma physics is the estimate that 99% of the matter in the universe is in the plasma state.”

Chapter 2

A Chapter

Chapter 3

Particle In cell

Simulations are used to mimic the physical world by solving the appropriate laws of physics and have a wide range of applications across all disciplines of science. Simulations provide a third tool alongside theory and experiments to explain physical phenomena. Deep insight into the physics of plasmas has been gained by the use of computer simulations. In 1964 Landau damping of electrostatic waves was observed in a computational plasma experiment by Dawson [3]. This phenomena had been predicted by theory but had no empirical observations to support it at the time. There have been huge increases in computational power since then and as that power has increased so has the capability of computer simulations to explore the natural world. Today a whole host of plasma simulation codes are used to investigate tokamak plasmas. The application of these codes range from simulating the entire tokamak, providing macroscopic quantities such as plasma density and temperature to tracking individual particles in a small sample of the divertor region to determine their behaviour as they approach the plasma-surface boundary. The simulations can enhance theoretical understanding and provide measurements that would be impossible to make in a physical experiment.

In the ideal case of unlimited computing power and memory, simulation codes would model plasmas by following the trajectories of every particle in the plasma as they move due to self-consistent electric and magnetic fields. Each particle in the system would interact with every other particle via the electric field. The simulations would be advanced in discretised time steps. At each time step the force acting on each particle due to every other particle would be calculated, this force would then go into Newton's equations of motion to supply each particle with a new velocity and position. Interaction forces would then need to be re-calculated and the cycle repeats until the simulation has run for the desired time. Even with the power of modern computers it is not possible to do this. A complete simulation of a tokamak plasma would be required to follow 10^{21} particles for millions of time steps, calculating the force at each time step. Calculating these particle-particle interactions for n particles in the system is of the order n^2 . No computer is capable of handling this many particles. However it is still possible to capture all of the relevant physics without such stringent computing demands.

Various computational techniques have been developed to simulate plasmas and they broadly fall into one of three categories. Particle-In-Cell (PIC) codes which follow the trajectories of individual particles in the plasma, Fluid models that treat the electrons and ions as separate fluids and Hybrid models that use a combination of fluid and PIC techniques to model the plasma [?]. Each method has certain advantages and disadvantages that determine where its use is applicable. These will be discussed below.

Out of the three categories PIC codes are considered to be the most fundamental way to model a plasma. Individual electrons and ions are tracked as they move across a spatial domain responding to self consistent electric and magnetic fields generated by the electric charge of the particles. The individual particles are in fact superparticles that represent many real particles. Rather than particle-particle interactions between every pair of particles in the simulation, superparticles deposit charge at discrete grid points along the domain. Other field quantities such as the electrostatic potential and electric field are then calculated from this charge density. The field is then interpolated back to the particles from the grid points in order to generate a new velocity and position. The discretisation of the field values as well as the use of superparticles allows modelling of the plasma from first principles. The essential physics of a real plasma can be captured with far fewer particles than are present in a real experiment. However in order to reduce statistical noise in the simulations large numbers of superparticles must be followed and there are certain stability criteria that must be met. As a result PIC simulations are compute-intensive, the simulations take a long time to run and this restricts PIC simulations to small regions of plasma.

Fluid codes ease the computational burden by treating the ions and electrons as separate fluids rather than individual particles. The plasma is described by the density, mean velocity and mean energy of the electrons and ions. This method implicitly assumes the particles are in equilibrium with a Maxwellian velocity distribution. The fluid equations are then solved to obtain the macroscopic quantities of the plasma. These equations require closure conditions that must be approximated. These assumptions hinder the accuracy of fluid modelling and limit its applicability. However fluid codes are much less demanding on computer resources and can model large areas of a tokamak.

Hybrid models exist that combine aspects of fluid and PIC codes. A common implementation of this technique is to treat the electrons as a fluid while modelling the ions kinetically. In this case the electrons are a neutralising background with a Boltzmann distribution. This enables the simulation to proceed much faster on ion time scales rather than having to resolve the individual motion of the electrons. This also reduces the number of particles that have to be followed. These models are useful if the user is only interested in the ion dynamics.

It is not possible to simulate a Langmuir probe without capturing the physics of the sheath. The plasma sheath cannot be correctly modelled by fluid codes as the particle distributions inside the sheath are far from being in equilibrium. The exact position of the sheath boundary is not well defined either. PIC codes on the other hand make

no assumptions about the distribution of the particles, they allow for any distribution function in phase space. By applying the fundamental equations the PIC method is able to preserve most of the physics. Therefore to thoroughly investigate the behaviour of Langmuir probes, completely kinetic PIC simulations must be carried out. The rest of this chapter will proceed to describe the general PIC method before moving on to describe each step of the algorithm in more detail.

3.0.1 General Method

It is not possible with modern day computers to simulate a plasma by tracking all 10^{21} particles and calculating all the particle-particle interactions for each pair of particles in the system. The PIC scheme overcomes this problem with the use of superparticles and by introducing a spatial grid as shown in figure 3.1. The superparticles have the same charge to mass ratio as their real particle equivalents and so follow the same trajectories as that of a real particle. For the remainder of this chapter the word particle is synonymous with superparticle. The spatial domain of the simulation is discretised into grid cells of equal or unequal length. Particles are free to take any position within the simulation domain and contribute to the charge density of their neighbouring grid cells. Various algorithms for charge deposition exist and will be discussed below. Once each particle has deposited charge to the grid the resulting charge density can be used to determine the electrostatic potential at each grid point by solving Poisson's equation. The derivative of this potential yields an electric field value at each grid point. These field values are then mapped back on to the particles often using the same algorithm as for the charge deposition. The field feeds into Newton's equations of motion in order to accelerate and move the particles.

For a simulation with n particles the introduction of the grid reduces the amount of calculations required per time step to the order of n rather than n^2 as in the particle-particle scheme. The grid is a mathematical construct that makes it possible to solve differential equations such as Newton's equations of motion and Poisson's equation by converting them into Finite Difference Equations (FDE). It also allows particles to interact with each other via a charge density rather than having to calculate the force between every pair of particles and so greatly reduces the run time of simulations. Splitting a physical, continuous domain up into grid cells does have implications which need to be considered in order to ensure the simulation can still produce physically accurate results. The consequences of introducing a grid on to the domain and how this can still accurately represent a plasma will be discussed.

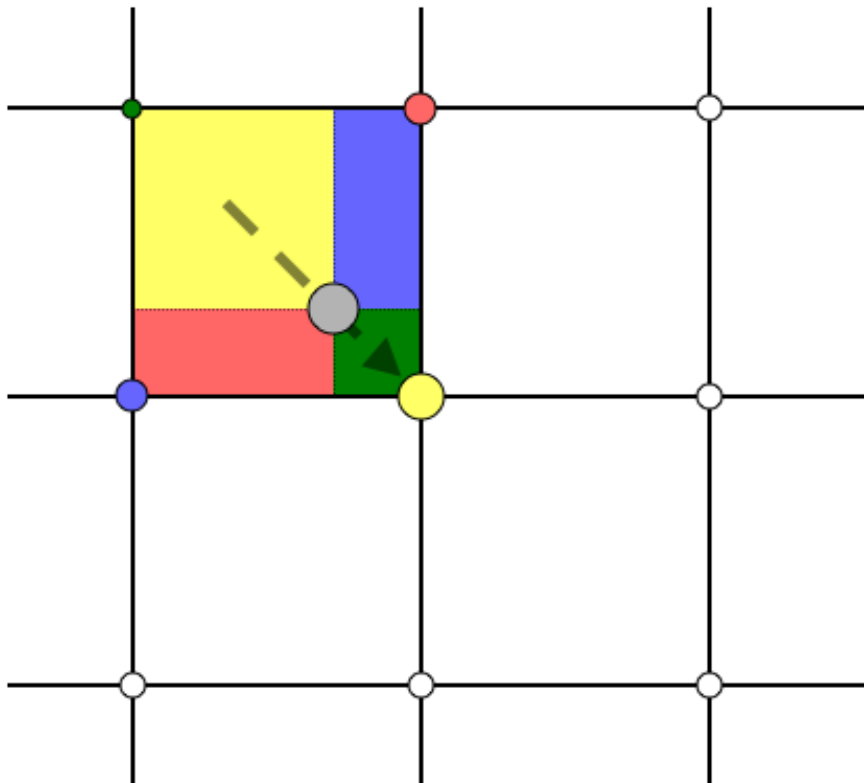
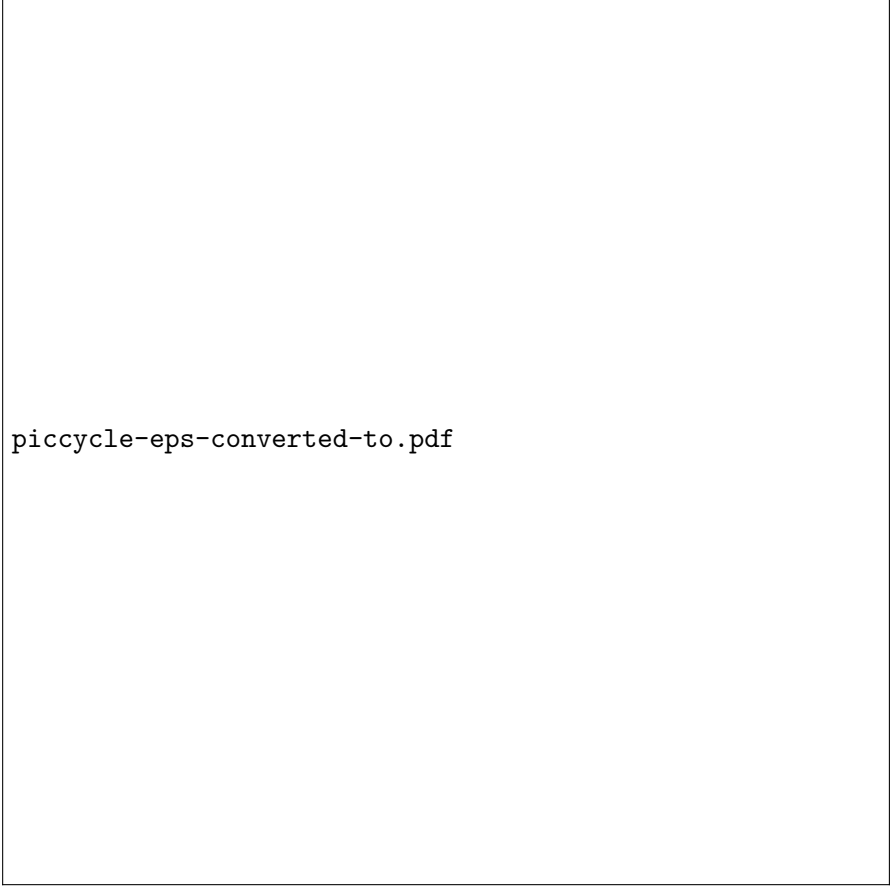


FIGURE 3.1: A representation of a two dimensional grid. The particle (grey circle) moves through the domain and deposits charge on the grid points. The area of the rectangle is proportional to the amount of charge deposited at each grid point.[1]

At the beginning of a PIC simulation the grid is loaded with a particular distribution of particles depending on the plasma parameters to be modelled. It then follows an algorithm as depicted in figure 3.2. Each particle deposits charge to neighbouring grid points, Poisson's equation is solved to obtain a potential, the derivative of this is used to calculate the electric field and the field is mapped back to the particles which are then accelerated and moved. Once the particles have been moved, the PIC algorithm is complete, time is advanced by one time step and the whole cycle restarts by calculation of a new charge density based on the updated particle positions. This cycle will carry on until a certain time has been reached or steady-state has been obtained. The aforementioned steps are essential to any application of the PIC method to plasma simulations. Additional steps can also be added to the cycle based on the requirements of the user. These steps can include Monte Carlo collisions between the particles, absorption and injection of particles and other boundary effects such as sputtering, secondary electron emission and specular reflection. These steps are often added to the end of the PIC cycle once all the original particles in the system have been moved.

Each step of the PIC cycle will now be detailed. For each of these steps highly optimised algorithms have been developed that are suitable for specific users. For clarity the most general methods will be discussed with specific focus on the algorithms used by VORPAL.



piccycle-eps-converted-to.pdf

FIGURE 3.2: A flow chart of the essential steps of the PIC algorithm

Calculating the Density

Weighting is the name given to the calculations which produce charge densities at the grid points from the continuous particle positions. Different weighting schemes have been developed and there is often a trade off between accuracy and computation time. The simplest weighting method is the Nearest Grid Point (NGP) scheme. This is a zero order weighting method in which the entirety of the particles charge is assigned to the grid point which it is closest to. Any particles within half a cell of a grid point are assigned to that grid point. Let the cell width be Δx , x the distance from the grid point and $W(X)$ denote the weighting at grid point X . In the NGP scheme

$$W(x) = \begin{cases} 1, & \text{if } x \leq \frac{\Delta x}{2} \\ 0, & \text{otherwise} \end{cases} \quad (3.1)$$

This is a computationally fast weighting method as it only requires one grid point look up per particle but this comes at the expense of adding noise to the simulation. As a particle moves away from its original grid point and into the region of a new grid point the grid density at the new grid point suddenly jumps up to have a value of one and the grid point it just left falls down to zero. This weighting scheme is not commonly deployed due to the noisy transition as particles move between cells. A less noisy method

is desired. First-order weighting also known as area weighting smooths the density and field fluctuations compared to NGP but is more computationally expensive as it requires more grid point look-ups for each particle. In this method for a 1D simulation each particle contributes charge to its nearest two grid points. The first step calculates the offset of the particle from the closest grid point to its left.

$$offset = x_i - X_j \quad (3.2)$$

where x_i is the particles position and X_j the grid point and $x_i > X_j$ always. The charge assigned to the j^{th} grid point is then

$$q_j = q_c (1 - offset) \quad (3.3)$$

where q_c is the charge of the particle. The charge assigned to the $j+1$ cell is

$$q_{j+1} = q_c (offset) \quad (3.4)$$

This results in a much smoother contribution to the density as the particle moves through the grid. This is the most commonly employed weighting scheme and is the default for VORPAL. Higher order weighting methods do exist such as quadratic and cubic splines, that are second order and third order accurate respectively. These schemes further smooth the non-physical noise at the expense of more computation time by increasing the number of grid points that a particle contributes its charge to. This does lead to problems at the edge of the boundary where there are insufficient neighbouring grid points.

Calculating the Potential

Now the charge density is known at each grid point, Poisson's equation for electrostatics can be solved to obtain the electrostatic potential.

$$\nabla^2 = -\frac{\rho}{\epsilon_0} \quad (3.5)$$

This can be solved numerically on a discretised grid using the Finite Difference Method (FDM). FDM solves a differential equation via the discretisation of its derivatives. The first step is to carry out a Taylor series expansion of the potential, first in the forward direction.

$$\psi(x + \Delta x) = \psi(x) + \Delta x \frac{\partial \psi}{\partial x} + \frac{(\Delta x)^2}{2} \frac{\partial^2 \psi}{\partial x^2} + \dots \quad (3.6)$$

The same procedure can be applied in the backwards direction

$$\psi(x - \Delta x) = \psi(x) - \Delta x \frac{\partial \psi}{\partial x} + \frac{(\Delta x)^2}{2} \frac{\partial^2 \psi}{\partial x^2} + \dots \quad (3.7)$$

These two equations can be combined to give an approximate value for the second derivative of potential. Summing (3.6) and (3.7) and rearranging gives

$$\frac{\partial^2 \psi}{\partial x^2} = \frac{\psi(x + \Delta x) - 2\psi(x) + \psi(x - \Delta x)}{(\Delta x)^2} \quad (3.8)$$

In FDM the solution to the equation is only known at the grid points, the potential is no longer a continuous function. Combining the forward difference and backward difference solution like this is known as central differencing and is second order accurate. For clarity we rewrite (3.8) with labels based on the grid number j .

$$\frac{\partial^2 \psi}{\partial x^2} = \frac{\psi_{j+1} - 2\psi_j + \psi_{j-1}}{(\Delta x)^2} = -\frac{\rho_j}{\epsilon_0} \quad (3.9)$$

The value of ψ at grid point j , (ψ_j), depends on the value of ψ at the two grid points either side of it (ψ_{j-1} and ψ_{j+1}), so the grid points are coupled together. In order to find the value of ψ_j at N different grid points requires the solution of N coupled linear equations. These coupled equations can be expressed in matrix form.

$$\begin{pmatrix} B_1 & C_1 & & & \\ A_2 & B_2 & C_2 & & \\ & A_3 & B_3 & C_3 & \\ & & \ddots & \ddots & \ddots \\ & & & A_N & B_N \end{pmatrix} \begin{pmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_N \end{pmatrix} = \begin{pmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \\ \vdots \\ \rho_N \end{pmatrix} \quad (3.10)$$

Where $A = 1, B = -2$ and $C = 1$. A matrix like this with non-zero elements only on the diagonal and one place either side of it is known as a tri-diagonal matrix. The value of ρ at each grid point is known as it was calculated in the previous step of the PIC algorithm. This matrix equation must now be solved in order to obtain the potential at each grid point. There are various numerical methods to find the solution and they can be divided into two categories: iterative methods and direct methods. Iterative methods begin with an estimate for the solution and use this estimate to get a better estimate. The iterations continue until the error in the estimated solution is below a certain pre-set tolerance. Iterative solves are useful in PIC applications as the potential at the previous time step can be used as the initial estimate for the next time step [4]. Direct methods only need to solve the equation once and do not rely on an initial estimate and so are generally faster in obtaining a solution. They are however harder to implement and more demanding on computer resources. ALSO WHY CAN'T BE USED FOR PARELLEL. Before a solution can be found the boundary conditions must be supplied as the grid points at the edge of the domain only have one neighbouring grid point. Two common choices for boundary conditions exist, Dirichlet boundary conditions where ψ_1 and ψ_N are set to a fixed value or Neumann boundary conditions where the gradient of the potential is fixed at the boundary. The implementation of Dirichlet boundary conditions is simple, B_1 and B_N are set equal to one and ψ_1 and ψ_N are then given

the desired potential boundary values α and β respectively. The first and last matrix equations then read

$$1.\psi_1 + 0.\psi_2 = \alpha \quad (3.11)$$

$$0.\psi_{N-1} + 1.\psi_N = \beta \quad (3.12)$$

Neumann boundary conditions involve fixing the gradient of the potential (i.e. the electric field) at the edge of the domain. This could be implemented by setting $B_1 = -\frac{1}{\Delta x}$, $C_1 = \frac{1}{\Delta x}$ and $\rho_1 = \alpha$. Thus giving the first line in the matrix equation as

$$\frac{\psi_2 - \psi_1}{\Delta x} = \alpha \quad (3.13)$$

Once the boundary conditions have been supplied the tri-diagonal matrix equation can be solved.

Calculating the Electric Field

MENTION WHY YOU DON'T CALCULATE FIELD STRAIGHT AWAY WITHOUT POTENTIAL Once the potential is known at each grid point the electric field is easily found by calculating the gradient of the potential.

$$E = -\nabla\psi \quad (3.14)$$

which in one dimension becomes

$$E = -\frac{\partial\psi}{\partial x} \quad (3.15)$$

This can discretised as before using the central difference method.

$$E_J = -\frac{\psi_{J+1} - \psi_{J-1}}{2\Delta x} \quad (3.16)$$

Central differencing is not applicable at the boundaries due to a lack of neighbouring grid points and so either the forward or backward difference method must be used which is only accurate to first order. This provides the following two conditions to calculate the electric field at the edge of the domain.

$$E_0 = -\frac{\psi_{J+1} - \psi_0}{\Delta x} \quad (3.17)$$

$$E_N = -\frac{\psi_N - \psi_{N-1}}{\Delta x} \quad (3.18)$$

Using a first order equation at the boundaries reduces the accuracy throughout the solution to first order. Fortunately the accuracy can be improved to second order by carrying out a further Taylor expansion

$$\psi(x + 2\Delta x) = \psi(x) + 2\Delta x \frac{\partial\psi}{\partial x} + \frac{(2\Delta x)^2}{2} \frac{\partial^2\psi}{\partial x^2} + \dots \quad (3.19)$$

Combining equations (3.6) , (3.19) and (3.15) gives

$$E_0 = \frac{3\psi_0 + \psi_2 - 4\psi_1}{2\Delta x} \quad (3.20)$$

The exact same method in the backwards direction gives

$$E_N = \frac{4\psi_{N-1} - \psi_{N-2} - 3\psi_N}{2\Delta x} \quad (3.21)$$

The second order boundary conditions restore second order accuracy across the domain.

The Particle Mover

The final step in the PIC cycle is to calculate a new position and velocity for each particle in the simulation based on the forces acting on them. In order to do this Newtons equations of motion must be solved

$$\vec{F} = m \frac{d\vec{v}}{dt} \quad (3.22)$$

$$\vec{v} = \frac{d\vec{x}}{dt} \quad (3.23)$$

For a particle in an electromagnetic field, the force experienced will be given by the Lorentz force

$$\vec{F} = q \left[\vec{E} + \vec{v} \times \vec{B} \right] \quad (3.24)$$

The particles positions and velocities can be found by integrating the differential equations (3.23) and (??) using finite difference methods. VORPAL uses a leap-frog scheme with a Boris advance to push the particles. The leap-frog method involves offsetting the velocity by half a time step from the position. So the velocity of the particles is only known at half integer time steps while the positions are known at integer time steps. This requires the initial velocities of the particles to be moved back half a time step at the beginning of the simulation, a "de-acceleration", in order to have time centred velocities. This just requires calculating the fields as before. This method is known as Leap-frog because in order to calculate new positions requires a leap over the known velocity.

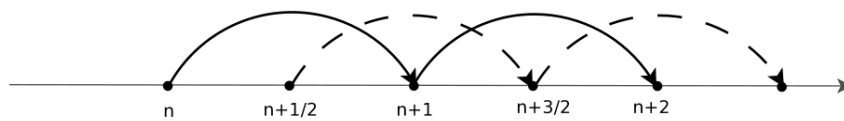


FIGURE 3.3: A graphical representation of the Leap-Frog scheme[2]

Instead of using the velocity at time t to move the particle from t to $t + 1$ like the Euler method, Leap frog uses the velocity at time $t + \frac{1}{2}$ i.e. the average velocity of the particle between those two times. This method is more accurate than Euler's method for the same computational expense. The only extra step involves pushing back the

particles half a time step but this only needs to be carried out once. For this reason Leap-frog is always the preferred choice over Euler and it can be proven to be second order accurate [?]. In discretised form the equations of motion become

$$\frac{x_{t+1} - x_t}{\Delta_t} = v_{t+1/2} \quad (3.25)$$

$$\frac{v_{t+1/2} - v_{t-1/2}}{\Delta_t} = \frac{q}{m} \left[\vec{E}_t + \frac{(v_{t+1/2} + v_{t-1/2})}{2} x B_t \right] \quad (3.26)$$

First equation 3.26 must be solved to get the new velocity of the particle ($v_{t+1/2}$), this is then inserted into equation 3.25 to obtain a new position for the particle (x_{t+1}). This is carried out for every particle in the system.

The most common implementation of the Boris scheme separates the effects of the electric and magnetic fields. Firstly half of the impulse due to the electric field is added to the particle's velocity creating an intermediate variable v^- /

$$v^- = v_{t-1/2} + \frac{q}{m} E_t \frac{\Delta t}{2} \quad (3.27)$$

The magnetic field then acts on v^- to create a second intermediate variable v^+ . The magnetic field only effects the rotation of the velocity vector not the magnitude.

$$\frac{v^+ - v^-}{\Delta t} = \frac{q}{2m} (v^+ + v^-) x B_t \quad (3.28)$$

Finally the second half of the electric impulse is added to v^+ to obtain the new velocity for the particle.

$$v_{t+1/2} = v^+ + \frac{q}{m} E_t \frac{\Delta t}{2} \quad (3.29)$$

The Boris scheme can be used to advance particles for an arbitrarily large number of time steps whilst remaining accurate and is therefore the de facto standard for particle movers [?].

Stability Conditions

PIC simulations utilise FDEs to find solutions to continuous differential equations on a discretised grid. The use of FDEs can provide accurate physical results provided three stability conditions are met. The first places a constraint on the size of each grid cell (Δx).

$$\Delta x < \frac{\lambda_D}{0.3} \quad (3.30)$$

PIC codes can only resolve phenomena that are larger than Δx , anything smaller than this is smoothed over. PIC codes must resolve the Debye length in order to accurately capture the shielding effects. The second constraint is that the plasma frequency must be resolved meaning that

$$\Delta t < \frac{2}{\omega_p} \quad (3.31)$$

The time step is further constrained by the criteria that no particle should be able to travel more than one grid cell in a given time step.

$$\Delta t < \frac{\Delta x}{v_{max}} \quad (3.32)$$

where v_{max} is the speed of the fastest moving particle in the system. For high density, low temperature plasmas $\lambda_D \approx 10^{-6}$ while electron velocities can exceed $10^6 m s^{-1}$. This means thousands of grid cells may be necessary to simulate the $10 mm^2$ tip of a Langmuir probe with time steps as small as $10^{-12} s$. Simulations with these demands can only be carried out on supercomputers.

Consequences of Using a Grid

This gives the particle an effective shape, in this case rectangular. The particle also has a finite size of width Δx . Finite sized particles are a direct result of weighting particles on to the grid. The weighting method determines the effective size and shape of the particle as viewed by grid. Due to the large jumps to grid values caused by the movement of particles from one grid point to the next, the NGP weighting method is a large source of noise in the density and field calculations. The weighting puts the fraction of the clouds charge which is in the J^{th} cell to the X_J grid point and the rest of it goes to the X_{J+1} grid point. This gives the particle a triangular shape, so the particle is effectively a triangular cloud of uniform charge centred at x_i with a width of $2\Delta x$ as it is able to influence a grid point from both sides of it.

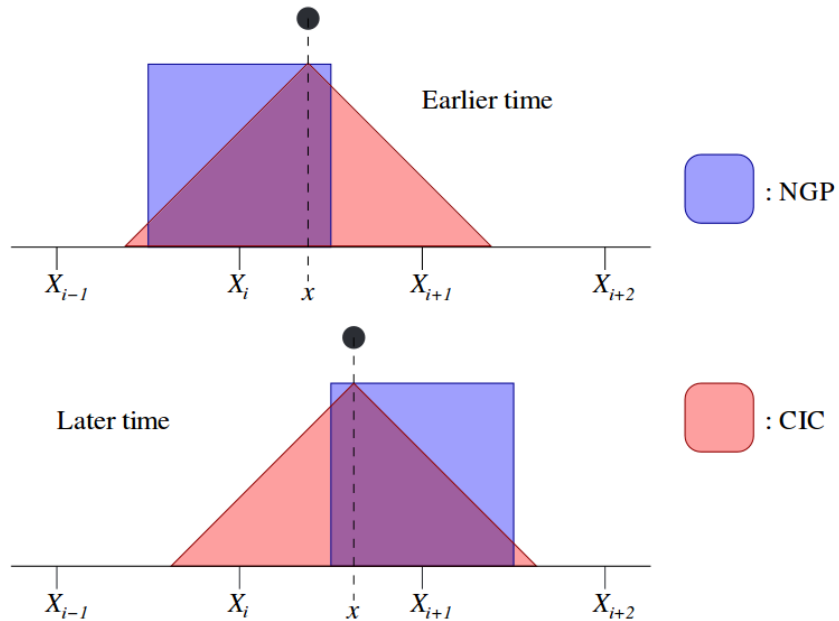


FIGURE 3.4: The effective shape of a particle at position x as seen by the grid.[2]

Particle Injection

In Langmuir probe simulations, particles are lost to the various collecting surfaces and must be replaced so that a constant density plasma can be simulated. It is therefore necessary to include an additional step of particle injection into the PIC algorithm. Particle injection takes place at the end of the PIC cycle once all other particles have been moved. It is desirable to inject particles at a rate that conserves the plasma density specified at the beginning of the simulation however it is not a necessity. The simulation will reach a steady-state density once the particle injection rate is balanced by the outflow rate of particles. The rate required to maintain a constant density plasma can be estimated as

$$R_{constant-density} = v_{th,s} \Delta T * N / \Delta x \quad (3.33)$$

where $v_{th,s}$ is the thermal velocity of the species, ΔT the size of the time step, N the number of particles per cell specified at the beginning of the simulation and Δx the gridspacing.

Langmuir probe theory is based on the assumption of Maxwellian velocity distributions an example of which is shown below. At the beginning of the simulation each species can be given a Maxwellian distribution and there exists various algorithms to implement this e.g. birdsall. The simulation will be required to run for many thousands of time steps in order to reach a steady state solution and conserving this distribution over the time required is not so simple. PIC simulations model a small region of the plasma and use superparticles to sample the velocity distribution. As there are orders of magnitude differences between the number of superparticles followed and the true number of particles in a real plasma, losses of superparticles in a PIC simulation can dramatically change the velocity distribution at very short time scales. The problem is enhanced by the lack of collisions in fusion-relevant PIC simulations. Due to the high densities and low temperatures of a SOL plasma, PIC simulations can only feasibly sample small regions of the plasma. The dimensions of the simulation region are often smaller than any collision mean free paths, so particles are able to move across the whole domain without colliding. Collisions drive particles towards Maxwellian distributions. It is therefore of crucial importance to sample from the correct velocity distribution when injecting particles into the simulation domain. In early simulations, particle loading at the beginning of the simulation and particle injection throughout the simulation both used the same source function. For every particle a random velocity was chosen sampled from a Maxwellian distribution. However it was found in practise that this did not conserve the Maxwellian distribution. With increasing simulation time, the velocity distribution of the particles narrowed, resulting in a 'cooling' of the plasma. Get picture of before and after. Although not anticipated the reasons for this are simple. Particles are initially loaded into the simulation domain with a range of velocities all sampled from a Maxwellian distribution. Provided there are enough superparticles in the domain, the distribution will be sufficiently represented by the finite number of particles. As time

advances particles exit the simulation once they reach an absorbing boundary layer and new particles enter in the injection phase of the PIC cycle. The fastest particles in the simulation are on average the first to leave as they quickly move across the domain to an absorbing surface. However if the injected particles are also sampled from the same Maxwellian distribution then it is far likely that the injected particle will have a velocity close to zero rather than a high velocity in the tail of the distribution. As a result the fastest particles leave the simulation quickly to be replaced by slow moving particles that hang around for a long time. This results in the fastest particles being under-represented in the simulation. The tail disappears, the distribution narrows and the effective temperature of the plasma cools. Ideal probe theory assumes a constant temperature, Maxwellian plasma. Experimental measurements work on this basis too. It is not possible to compare experimental data with simulation results if the the simulation plasma is not at a constant temperature. A source function that conserves temperature as the simulation runs is desired. This source function must replace the particles at a rate proportional to their velocity. The fastest particles are required to be replaced more often while the slow moving particles not so much. Rather than the Maxwellian distribution that peaks at $v = 0$ the new distribution must fall to zero at this point as those particles should never leave the simulation. A method to investigate the shape of the source function was established. COPY REPORT BIT ON GETTING THE RIGHT SOURCE FUNCTION For the purposes of investigating the correct source function to use a 1D simulation was adequate. The simulation domain is shown below.

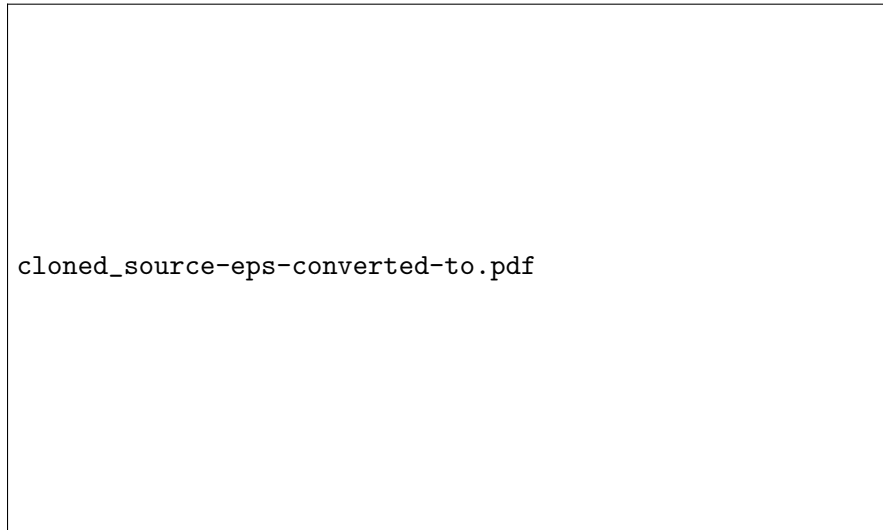


FIGURE 3.5: A source region on the left-hand side replaces particles that are lost to the wall.

The model tracks a length of plasma , hundreds of Debye Length (λ_D) long and follows the motion of all ions and electrons as they move in self-consistent electric fields. The presence of a magnetic field impacts the source function so to begin with only electrostatic simulations were carried out. On one side of the domain is an absorbing surface that represents the probe, any potential can be set to this surface and any

charged particles that hit the surface are deleted from the simulation and their current recorded. On the opposite side of the domain is a source region of plasma. The purpose of this source region is to supply the bulk plasma with new particles to replenish those lost to the sides. At the beginning of the simulation a quasi-neutral plasma with a Maxwellian distribution fills the entire domain. Any particles born into the source region ($0 \leq x \leq LS$) are trapped in the source region for the duration of the simulation. These source particles travel back and forth in the source region and are reflected at the boundaries $x = 0$ and $x = LS$. Both edges of the source are held at the same potential (V_{source}) which fixes the plasma potential. As there is no potential difference between the two sides and no particles can escape the plasma source remains at the temperature and density specified at the beginning of the simulation. The plasma maintains the Maxwellian distribution throughout the simulation as shown in figure 3.7.

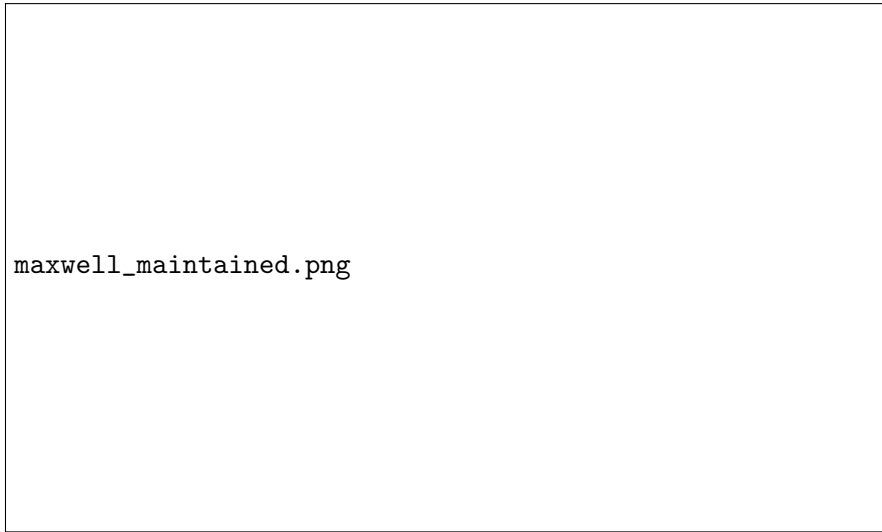


FIGURE 3.6: The velocity distribution of particles in different regions of the plasma at the end of the simulation. The Maxwellian distribution is conserved.

Any source particle striking the $x = LS$ boundary is copied before being reflected. This copy is identical to the original particle, having the same charge, mass, position and velocity but is not reflected at the boundary. Instead the copy travels into the bulk plasma, allowing the source region to replenish the plasma. By looking at the velocity distribution of particles exiting the source it is possible to determine the form of the source function required to sample a Maxwellian plasma. The distribution of particles leaving the source region is equivalent to the distribution of particles that would exit the simulation of a bulk plasma. It is these velocities that must be sampled in order to maintain a constant temperature plasma. The velocity distribution of particles leaving the source is shown below. GET EMMERT PLOT Describe the plot, cite Emmert paper and the source function.

As well as providing a source function this method provided validation for the rate of injection. It was found that the ratio of electrons exiting the source to ions exiting the source was equal to the ratio of their thermal velocities. Now a temperature conserving

source function has been determined there is no need for the source region. Simulating the source region requires tracking many particles, which is feasible in 1D simulations but would take up too much time for higher dimensions. The simulation domain now looks like this. 3.7.

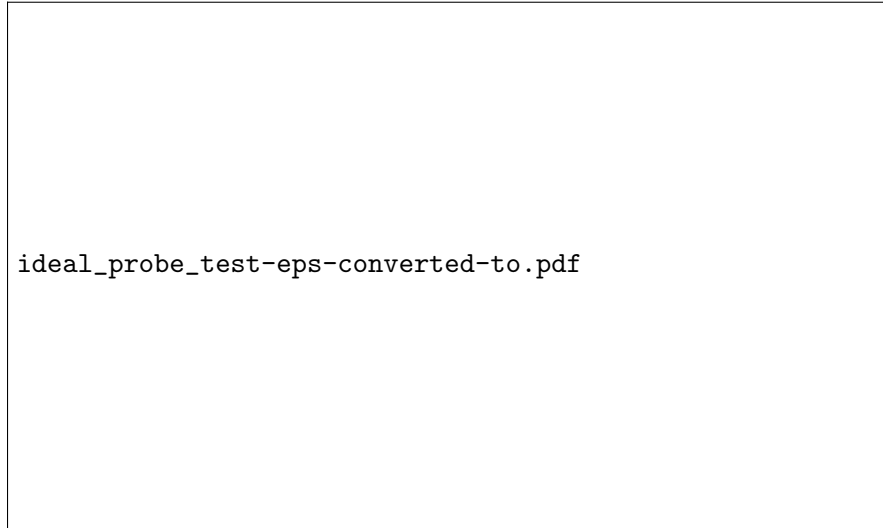


FIGURE 3.7: The velocity distribution of particles in different regions of the plasma at the end of the simulation. The Maxwellian distribution is conserved.

A plasma source is required to replenish particles that are lost to the probe. This source can be held at any potential, I choose 0V, this sets the plasma potential. The system reaches a steady state over a few ion transit times at which point the current drained by the probe surface reaches a constant value and the plasma density stabilises. The domain is shown below.

EFFECTS OF ADDING A B FIELD, YOU WANT TO SPECIFY THE PARALLEL VELOCITY BUT HOW TO DO YOU DO THAT WHEN THE FIELD IS TILTED
 DETAIL TEST CARRIED OUT TO SHOW PARTICLES MOVES AT CORRECT RATE

Chapter 4

A Chapter

Chapter 5

A Chapter

Chapter 6

A Chapter

