MEE4099 Capstone Project
Review 1 tentative report: first draft
Project Title:

Magnetic Mirror Effect in Magnetron Plasma: Modeling of Plasma Parameters

Project ID: 21BTECH10051
Team Members:
18BEM0145 Sashi Kant Shah
18BME2104 Kaushal Timilsina
18BME2109 Hrishav Mishra

Internal Guide:
Professor Sitaram Dash

Contents

1 Introduction

One of the team members- 18BME2104 Kaushal studied the class MEE4005 Surface Engineering taught by Professor Sitaram Dash- our internal guide, during the Fall Semester 2021. Many interesting plasma based surface engineering techniques were studied during the class, one such technique being Magnetron Sputtering. This inspired the study of plasma in this project.

1.1 Plasma

One comes across many definitions of plasma including: fourth state of matter, ionized gas, a non equilibrium state of matter with dynamical characteristics due to electrodynamics, etc. However, it is best to describe plasma with some characteristic parameters, when one attempts to describe a plasma quantitatively. Some quantities that help define a plasma are:

1. Number density, n
   Number density of a plasma describes the number of particles per unit volume. Plasma contains charged particles or ionized species. However, a plasma might at the same time also contain neutral atoms and molecules but also particles of different species- charged and neutral. If multiple species are contained in a plasma system, number densities of each species could be used to describe the system. For example, a plasma may contain electrons, charged ions and neutral atoms and molecules. Mass density ϱ is defined as $\varrho := mn$ and is often used alongside number density, where m is the mass of the species.

2. Ionization, α
   Defined as $\alpha := \frac{n_{charged}}{n_{charged} + n_{neutral}}$, the ionization of a plasma describes the fraction of charged particles, with $\alpha = 1$ meaning that all the particles are charged and $\alpha = 0$ meaning that all the particles are neutral.

3. Temperature, T
   The temperature of a plasma describes the average kinetic energy of the particles in the plasma. When a gas is ionized to form a plasma, the ionization α can depend on the temperature of the plasma.

4. Mean free path, λ_{mfp}
   The gas-like behavior of a plasma is characterized by mean free path of particles much larger than the scale of the plasma. The mean free path is influenced by the temperature of the plasma. The mean free path and the thermal velocity of the particles as described by the temperature, are related by the timescale of collisions as $\lambda_{mfp} := v_{th}\tau$ where τ is the timescale of collisions.

5. Debye Length, λ_D
   In a plasma, electrostatic Coulomb interactions between charged particles compete with random thermal speed of the particles described by the temperature of the

plasma. The Debye sphere is an imaginary sphere around a charged particle, where oppositely charged particles are attracted and in doing so screen the charge of the central particle from the outer plasma so that the electrostatic influence of a particle is limited to the Debye sphere surrounding it. This is why plasma's are often said to be Quasi-neutral as charge screening leads to a neutral behavior electrostatically on a scale much larger than the Debye length. The Debye length is defined as the radius of the Debye sphere.

6. Plasma beta parameter, β
   The beta parameter defined as $\beta := \frac{8\pi nT}{B^2}$ describes the ratio of the thermal and magnetic energies of the plasma, as particles in random thermal motions compete with the Lorentz force.

Many other parameters like the Larmor radius, quantities like frequencies of different waves describing the dynamics; are also important in describing a plasma. However, as we shall see later; particles in the plasma are more important to this project than many interesting parameters describing the plasma.

1.2 Laboratory Plasma

Laboratory plasmas are often characterized by some properties like:

- High ionization fraction
- Sub atmospheric pressure required to sustain ionization
- Temperature range : 1000 - 30000 K

Many surface engineering processes use plasmas to obtain high performance coatings. Some processes that use plasma are:

1. Plasma Immersion Ion Implantation
2. Plasma Enhanced Chemical Vapor Deposition
3. Magnetron Sputtering (PVD)
4. Air Plasma Spray (Spray technique)
5. Plasma Transferred Arc (Hardfacing technique)

1.3 Physical Vapor Deposition

Physical Vapor Deposition(PVD) is a family of surface engineering techniques where thin film coatings are grown on the surface of a specimen, in a vacuum chamber. Particles from a vapor move around on the surface of the specimen as random walkers and eventually get trapped in strained pockets producing nucleation site for the growth of a film. Most PVD techniques fall under either of the two catgories:

1. Evaporation techniques
   Evaporation techniques involve heating the material to to a high temperature when it forms vapor and the particles in the vapor are coated on the substrate. It is for that reason that evaporation techniques are called hot techniques. Semiconductors like Si/Ge, insulators like oxides and metals like Tungsten are often coated on substrates using evaporation techniques.

2. Sputtering techniques
   Sputtering techniques on the other hand are known as cold techniques. Sputtering techniques use a high energy beam to remove material from a target and the removed material is coated on the required substrate.

1.4 Magnetron Sputtering

Magnetron sputtering is a sputtering technique where the sputtered ions form a magnetically confined plasma. The plasma, controlled by magnetic fields, transports the ions to the surface of the substrate forming the coating.

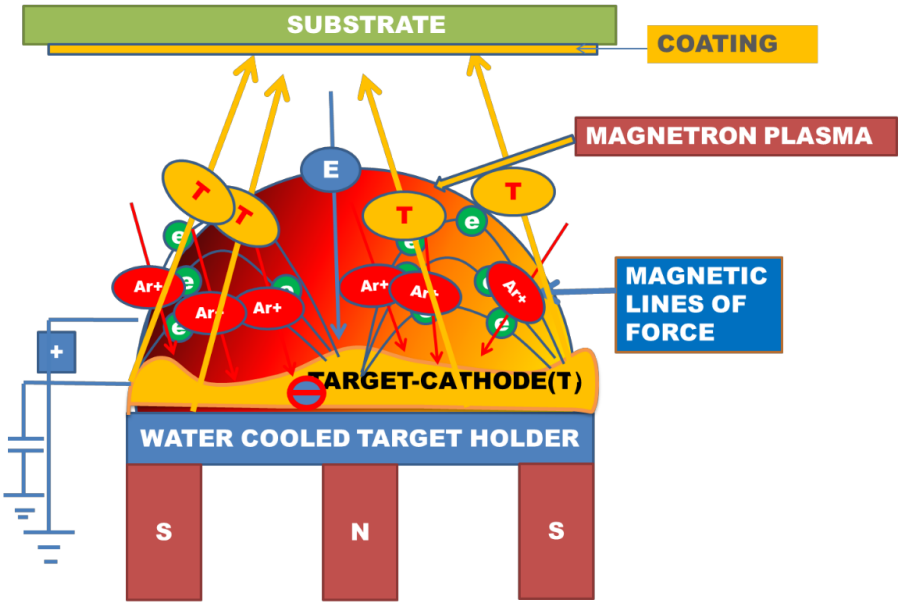

Figure 1: Magnetron Sputtering system. credit: [1]

1.5 Magnetic Mirror

The magnetic mirror effect can first be illustrated with a single particle. For the simple case, it is assumed that there is no electric field acting on the particle, the magnetic field is cylindrically symmetric and the gradient of the magnetic field is only along the axial direction of the cylinder, the z direction. The magnetic moment

$$\mu = \frac{mv_\perp^2}{2B_z}$$

of the particle is an adiabatic invariant of the particle motion and hence is approximately conserved for a magnetic field whose z component B_z does not vary too much . The kinetic energy

$$KE = \frac{1}{2}mv^2$$

of the particle is also approximately conserved. Writing $v_\perp = vsin\theta$,

$$\frac{\mu}{KE} = \frac{sin^2\theta}{B_z}$$

is also a conserved quantity. This means that as B increases (within a small range), $sin^2\theta$ increases as well and so does $|sin\theta|$ which means that $|v_\perp|$ increases and since v is conserved, $v_\parallel$ decreases to zero. When $v_\parallel = 0$, the particle is no longer moving in the z direction, but is moving with velocity v in the plane perpendicular to z direction. The $v_\parallel$ then increases in the opposite direction as the particle moves towards decreasing B because of the Lorentz force. Such a particle is seen as being reflected due to the configuration of the magnetic field and hence such a configuration of the magnetic field is called a magnetic mirror. Particles whose $v_\parallel$ does not decrease to 0 while the value of B is decreasing, escape from the magnetic mirror configuration.

A magnetic mirror configuration is important in a magnetic plasma trap chamber such as that used in Magnetron sputtering. Particles in a plasma have different speeds depending on the initial distribution which is based on parameters like the plasma temperature. The speeds of particles change depending on the electric and magnetic fields. Based on the magnetic mirror effect, one can determine which particle (having certain velocities) can escape the magnetic trap and which of those are reflected. The less the particles escape the magnetic trap, the more of the flux is used in forming coatings and less of the ionized gas is wasted. This is very useful in understanding the required gas supply and rate of deposition.

2 Literature Review

A large part of the literature review for the project consisted of studying basic plasma physics in order to understand how we could formulate the project and proceed ahead.

2.1 Plasma as a system

Various models are used to describe Plasma as a system. Some of the common approaches are:

1. **Single particle description**
   This model is used to describe the motion of a charged particle under the influence of electric and magnetic fields. The particle's motion is described by the Lorentz force

$$\frac{d\boldsymbol{v}}{dt} = \frac{q}{m}\left(\mathbf{E} + \boldsymbol{v} \times \mathbf{B}\right) \tag{1}$$

Describing many particles evolving under the influence of Lorentz force, it is easy to describe a plasma assuming that the mean free path of the particles is much larger than the dimensions of the chamber; meaning that particles hardly ever collide. This is to say that in the simplest case, under this model particles evolve under the influence of electric and magnetic fields but the particles do not produce any electric and magnetic fields of their own or affect the external applied fields and hence also do not interact with other particles.

2. **Kinetic theory**
The kinetic theory describes the plasma as collection of particles whose state (position and velocity) is treated as a random variable with a density function $f(x, y, z, v_x, v_y, v_z, t)$ which describes the number of particles at position (x, y, z) at time t with velocities between v_x and $v_x + dv_x$, v_y and $v_y + dv_y$, v_z and $v_z + dv_z$ in directions x, y and z respectively. For a simpler notation $f(x, y, z, v_x, v_y, v_z, t)$ is denoted as $f(\boldsymbol{x}, \boldsymbol{v}, t)$. The expression

$$\int_{all\ \ v_x} dv_x \int_{all\ \ v_y} dv_y \int_{all\ \ v_z} dv_z f(\boldsymbol{x}, \boldsymbol{v}, t)$$

gives the number of particles at position $\boldsymbol{x}$, at time t. For a simple choice of notation, it if often written as

$$\int_{all\ \ \boldsymbol{v}} d^3v f(\boldsymbol{x}, \boldsymbol{v}, t) \quad or \quad \int_{all\ \ \boldsymbol{v}} d\boldsymbol{v} f(\boldsymbol{x}, \boldsymbol{v}, t)$$

A density function is said to be normalized if

$$\int_{all\ \ \boldsymbol{v}} d\boldsymbol{v} f(\boldsymbol{x}, \boldsymbol{v}, t) = 1$$

Such a density function is also denoted with a hat as $\hat{f}(\boldsymbol{x}, \boldsymbol{v}, t)$.
The average velocity $\bar{v}$ for a density function $\hat{f}(\boldsymbol{x}, \boldsymbol{v}, t)$ is calculated as

$$\int_{all\ \ \boldsymbol{v}} d\boldsymbol{v}\ \ v f(\boldsymbol{x}, \boldsymbol{v}, t)$$

Various other features of the distribution are: the Root Mean Square velocity v_{rms}, the average absolute velocity $|\bar{v}|$, the average velocity in z direction $\bar{v_z}$, etc.
The evolution of the particles is described as the changing of the density function. Boltzmann equation is often used to describe this dynamics:

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \nabla f + \frac{\mathbf{F}}{m} \cdot \partial_{\boldsymbol{v}} f = \left(\frac{\partial f}{\partial t}\right)_c$$

While it may look complicated at first, the left hand side of the equation is actually just a short form for

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x} v_x + \frac{\partial f}{\partial v_x} a_x + \frac{\partial f}{\partial y} v_y + \frac{\partial f}{\partial v_y} a_y + \frac{\partial f}{\partial z} v_z + \frac{\partial f}{\partial v_z} a_z$$

which is just

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial v_x}\frac{dv_x}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial v_y}\frac{dv_y}{dt} + \frac{\partial f}{\partial z}\frac{dz}{dt} + \frac{\partial f}{\partial v_z}\frac{dv_z}{dt}$$

For simplicity, let us look at the 1-dimensional (2-phase dimensional) version of this expression:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial v_x}\frac{dv_x}{dt}$$

If the right hand side of the Boltzmann equation were zero, in 1-dimension (2-phase dimensions), it would read:

$$\frac{df}{dt} = 0$$

This would mean that the distribution function is unchanging (in its own frame). If particles were not interacting and there was no external disturbance, this is exactly what would happen. This is why with the expression on the right hand side set to zero, the Boltzmann equation is said to be collisionless. Adding a non zero expression on the right hand side would account for interactions between the particles or the effect of external disturbances.

3. **Fluid model**
The fluid model is describes the plasma in terms of macroscopic variables like Pressure, Temperature, average velocity, density, flux; like ordinary fluid dynamics. The governing Partial Differential Equations are obtained by taking moments of the Boltzmann equation, or a special case of the Boltzmann equation called the Vlasov equation. Such a procedure gives rise to:

The continuity equation:

$$\frac{\partial \varrho}{\partial t} + \boldsymbol{\nabla} \cdot (\varrho \boldsymbol{v}) = 0$$

and the Cauchy momentum equation:

$$\varrho \left(\frac{\partial}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} \right) \boldsymbol{v} = \mathbf{J} \times \mathbf{B} - \boldsymbol{\nabla} p$$

Higher moments yield other equations for quantities like the entropy. The fluid equations average over the velocity distribution of the particles to obtain the macroscopic variables like Pressure and Temperature, as discussed earlier- by taking moments of the Kinetic equations like the Boltzmann equation or the Vlasov equation.

4. **Magnetohydrodynamics**
Magnetohydrodynamics(MHD) is an extension of the fluid model of describing a plasma. In a first approximation, the magnetic and electric fields acting on the fluid and the currents generated are now solved using Faraday's law:

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

and the Ampere's law:

$$\mu_0 \mathbf{J} = \boldsymbol{\nabla} \times \mathbf{B}$$

The full set of Maxwell's equations can also be coupled with the fluid. MHD description of a plasma also describes dynamic waves such as Alfvén waves and magnetosonic waves. MHD theory can also be used to describe interaction of the plasma with electromagnetic waves, for example those from a LASER source relevant to LASER plasma processes in surface engineering. Ideal MHD studies plasma as a fluid of single species, however, multi-species plasma can also be studied.

5. **The model used in the project**
   In the context of plasma processes like Magnetron Sputtering where particles have a considerably large mean free path, we have decided not to use the fluid model and the MHD descriptions of a plasma. In such processes, it is almost always the case that the strength of the electric and magnetic fields applied by the apparatus is far more stronger than the electric and magnetic fields generated by charged particles in a plasma. This approximation allows us to avoid solving Maxwell's equations and use only the Lorentz force. The model used in the project is mostly similar to the single particle model, where in a collection of particles, each particle evolves under the influence of the electric and magnetic fields set up by the apparatus; governed by the Lorentz force. However, we incorporate a little bit of the Kinetic theory in that we are interested in different initial velocity distributions for particles in the plasma and how the velocity distribution changes over time; as it is important to understand the velocity distribution to characterize the reflection and loss of particles in a magnetic-mirror like trap that may be setup in the plasma chamber.

3 Gaps in Literature

It would take a number of courses on plasma physics to fully understand the current research methods in plasma physics. In the interest of time, we have studied some simple setups; so that we can setup well functioning plasma configuration during the course of the project and study some aspects that we are interested in. The forefront of research in plasma physics concerns complicated devices like Magnetic Confinement Fusion, or Quantum Optic systems. Most research in surface engineering focuses on the properties of coatings obtained and parameters of plasma used; in processes that use plasma. In our project, we would like to set up a simple plasma simulation that can help us study smaller devices like the one available in the School of Mechanical Engineering; where we can control a small flux of particles by tuning the electric and magnetic fields. This section describes, how we construct a simple easy to use, plasma simulation system; which is yet to be functional.

3.1 Particle in Cell Methods

Particle in a cell methods are used to simulate the kinetic theory of plasma. A simple strategy used for particle in cell plasma simulation based on strategies as outlined in the paper [6] and the slides [7] involves the following steps:

1. **Sampling and Initialization**
   The initial positions and velocities of particles in the plasma are sampled from a distribution, or based on some strategy.

2. **Action of fields on the particles**
   The particles move under the influence of electric and magnetic fields as described by the Lorentz force as stated in the equation (1).

3. **Particle deposition**
   In this step, charged particles are deposited on the grid defined by the mesh, and the charge density ϱ_i and the current density $\boldsymbol{j}_i$ generated by the deposited particles is computed. One strategy outlined in the paper [6] defines charge deposition as following. $\boldsymbol{x_i} = (\boldsymbol{i} + 1/2)\,\Delta\boldsymbol{x}$, $\boldsymbol{i} \in \mathbb{Z}^D$ define the grid. A second order deposition can be achieved by:
   $$\varrho_i = \sum_p \left(\frac{q_p}{V_i}\right) \mathbf{W_2}\left(\frac{\boldsymbol{x_i} - \boldsymbol{x_p}}{\Delta x}\right)$$
   where $V_i = \Delta x^D$ is the volume of the cell i and $\mathbf{W_2}\,(\boldsymbol{x})$ is a D-dimensional interpolating function defined in [6]. In simple models, the current density $\boldsymbol{j}_i$ is often not used.

4. **Fields generated by particles**
   In this step, the electric and magnetic fields generated by the charge density and current density are computed. The paper [6] uses Poisson equation to compute the electric field generated by the charge distribution and neglects the magnetic field generated. However, in high performance simulations like that outlined in [7], the full set of Maxwell's equations are used to compute the electric and magnetic fields generated by the particles.

5. **Force on particles**
   In this step, the force on the particles due to the electric and magnetic fields are computed. Most simulations like the one outlined in the paper [6]; because they compute the electric and magnetic fields generated by the deposited particles, are able to describe the interaction of each particle with the electric and magnetic fields generated by other particles in the plasma, and hence capture the particle-particle dynamics.

6. **Action of the force on particles**
   Step 2 is repeated to move the particles under the influence of the electric and magnetic fields.

A flow-chart for the simulation based on a similar strategy outlined in [7] is presented below.

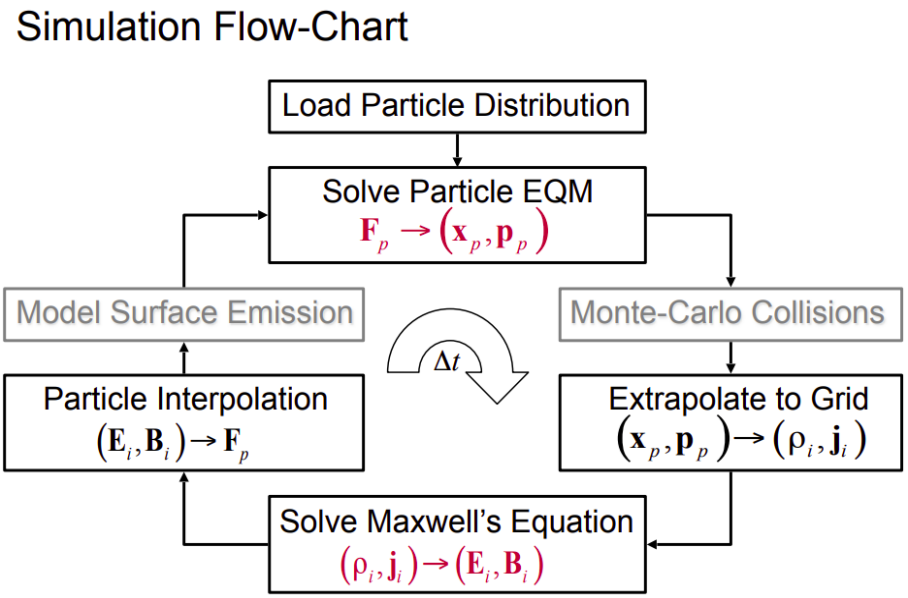

Figure 2: Flowchart for PIC methods. credit: [7]

4 Problem Definition

Our approach is similar to and differs from the general procedure outlined for Particle in Cell methods, in the following ways:

1. **Sampling and Initialization** Similar
   This step is similar to the first step in general PIC methods. We sample the initial positions and velocities of the particles based on some strategies like sampling from the Maxwell-Boltzmann distribution or uniform initialization strategies described in Methodology.

2. **Action of fields on the particles** Similar
   This step is similar to the second step in general PIC methods. We update the positions and particles of the particles according to the Boris Algorithm, based on the Lorentz force; as described in Methodology.

3. **Particle deposition** Skipped
   We skip this step in our approach for reasons described in the next step.

4. **Fields generated** Different
   As discussed earlier; when discussing the model used in the project, in devices such as those used in surface engineering processes, it is often the case that the electric and magnetic fields generated by the apparatus are far more stronger than those

generated by the particles. So it is reasonable to assume that the fields generated by the particles are negligible compared to the fields generated by the apparatus. While general PIC methods compute fields generated by particles in the plasma, we only used fields generated by the apparatus. Since we neglect the fields generated by the particles in the plasma, we can skip depositing particles in the grid; which would be the third step of general outline of PIC methods. This makes our model simpler and easier to work with, evaluate and understand. We define the fields generated by the apparatus or field configurations based on analytic configurations and move the particles in the plasma under their influence. This is later described in methodology.

5. **Force on particles** Skipped
   As we do not compute the electric and magnetic fields generated by the particles, this step is skipped and the action of the electric and magnetic fields created by the apparatus is done directly in step 6.

6. **Action of the fields on particles** Similar
   Similar to the sixth step of general PIC methods, we repeat step 2 to move the particles under the influence of electric and magnetic fields created by the apparatus.

Our approach can be summarized in the following set of steps:
Do for each batch of particles in the plasma stream:

1. **Sampling and Initialization** Kinetic Theory
2. **Definition of fields** Apparatus
   If required use a different field to simulate control of the apparatus, for example: changing the voltage of the electrode; changing the electric field.

   Do for certain number of time steps:

   - **Particle update based on Lorentz force** Single particle dynamics

3. **Remove particles** Surface Engineering process
   Particles are either absorbed to form a coating or exit the plasma chamber.

The details are described later in methodology. The steps of our approach as described help us define our objectives as discussed in the following section.

5 Objectives

The objectives of the project are based on our approach to the problem as described in problem definition:

1. **Single Particle Method**
   To simulate charged particles that evolve under the influence of electric and magnetic fields; as governed by the Lorentz force.

2. **Field Configurations**
   To simulate a few different configurations of electric and magnetic fields- some describing apparatus like coils; some describing analytic expressions for fields and to study the different evolution of particles.

3. **Kinetic Theory**
   To study different initial velocity distributions and how the velocity distribution of particles changes as the particles evolve. Parameters like the Plasma temperature are to be studied under this topic.

4. **Analysis**
   To analyze different batches or collections of particles, subjected to different field configurations.

6 Methodology

6.1 Particle Evolution- single particle dynamics

As discussed earlier, we evolve individual particles in the plasma with the Lorentz force. To achieve this in a simulation, the Lorentz force equations are discretized and then solved using the Boris Algorithm; a standard algorithm for simulating charged particles in electric and magnetic fields.

6.1.1 Lorentz Force

The equations of motion for a charged particle under the influence of Electric and Magnetic fields is described by the Lorentz Force in the S.I. units as

$$\frac{d\boldsymbol{v}}{dt} = \frac{q}{m}\left(\mathbf{E} + \boldsymbol{v} \times \mathbf{B}\right)$$

as discussed earlier in equation (1); along with the expression for the velocity

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v} \tag{2}$$

These equations are discretized to obtain

$$\frac{\boldsymbol{v}_{k+1} - \boldsymbol{v}_k}{\Delta t} = \frac{q}{m}\left[\mathbf{E}_k + \frac{(\boldsymbol{v}_{k+1} + \boldsymbol{v}_k)}{2} \times \mathbf{B}_k\right] \tag{3}$$

from the Lorentz Force equation (1), and

$$\frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{\Delta t} = \boldsymbol{v}_{k+1} \tag{4}$$

from the expression for velocity in equation (2), where the subscript k denotes the k-th time step.

6.1.2 Boris Algorithm

The discretized Lorentz equations may be solved using the Boris Algorithm, as we do in this project. The Boris Algorithm splits equation (6.1.2) into three equations.

$$\frac{\boldsymbol{v}^- - \boldsymbol{v}_k}{(\Delta t/2)} = \frac{q}{m}\mathbf{E}_k \qquad or \qquad \frac{\boldsymbol{v}^- - \boldsymbol{v}_k}{\Delta t} = \frac{1}{2}\frac{q}{m}\mathbf{E}_k \tag{5}$$

which is often called the first half of the electric pulse.

$$\frac{\boldsymbol{v}^+ - \boldsymbol{v}^-}{\Delta t} = \frac{q}{m}\left(\frac{\boldsymbol{v}^+ + \boldsymbol{v}^-}{2}\right)\mathbf{B}_k \tag{6}$$

which is often called rotation by the magnetic field.

$$\frac{\boldsymbol{v}_{k+1} - \boldsymbol{v}^+}{(\Delta t/2)} = \frac{q}{m}\mathbf{E}_k \qquad or \qquad \frac{\boldsymbol{v}_{k+1} - \boldsymbol{v}^+}{\Delta t} = \frac{1}{2}\frac{q}{m}\mathbf{E}_k \tag{7}$$

which is often called the second half of the electric pulse.
Adding equations (5), (6) and (7) gives

$$\frac{\boldsymbol{v}_{k+1} - \boldsymbol{v}_k}{\Delta t} = \frac{q}{m}\left[\mathbf{E}_k + \frac{(\boldsymbol{v}^+ + \boldsymbol{v}^-)}{2} \times \mathbf{B}_k\right]$$

which is almost the discretized Lorentz equation (6.1.2) except that $(\boldsymbol{v}^+ + \boldsymbol{v}^-)$ is substituted for $(\boldsymbol{v}_{k+1} + \boldsymbol{v}_k)$. However, subtracting equation (5) from equation (7) gives $(\boldsymbol{v}^+ + \boldsymbol{v}^-) = (\boldsymbol{v}_{k+1} + \boldsymbol{v}_k)$, giving the discretized Lorentz equation (6.1.2). This means that the Boris algorithm is equivalent to the discretized Lorentz equation.

The equations (5), (6) and (7) can be written slightly different as

$$\begin{aligned} \boldsymbol{v}^- &= \boldsymbol{v}_k + q'\mathbf{E}_k \\ \boldsymbol{v}^+ &= \boldsymbol{v}^- + 2q'\left(\boldsymbol{v}^- \times \mathbf{B}_k\right) \\ \boldsymbol{v}_{k+1} &= \boldsymbol{v}^+ + q'\mathbf{E}_k \\ \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + \Delta t\ \ \boldsymbol{v}_{k+1} \end{aligned} \tag{8}$$

alongwith equation (4) where $q' = \frac{q}{m}\frac{\Delta t}{2}$. These equations are used to update the velocity of the particle under the influence of the Lorentz force under the Boris update strategy.

6.2 Particle Sampling- kinetic theory

As discussed earlier, kinetic theory of plasma uses density function and its evolution to describe a plasma. We pointed out that we will be using some aspects of kinetic theory in our project; in that we will initialize the batches (collections) of particles based on certain distributions. This will allow us to study parameters like the plasma temperature, in our project.

6.2.1 Maxwellian distribution

One important density function often used in the kinetic theory of plasma is the Maxwell-Boltzmann distribution often called the Maxwellian which has the density function

$$\widehat{f_M} := \hat{f}(\boldsymbol{x}, \boldsymbol{v}, t) = \left(\frac{m}{2\pi KT}\right)^{\frac{3}{2}} \exp\left(-\frac{v^2}{v_{th}^2}\right) \tag{9}$$

where

$$v_{th}^2 = \frac{2KT}{m}$$

Some features of the Maxwellian are:

$$v_{rms} = \sqrt{\frac{3KT}{m}}, |\bar{v}| = 2\sqrt{\frac{2KT}{\pi m}}, |\bar{v_z}| = \sqrt{\frac{2KT}{\pi m}}, \bar{v_z} = 0$$

6.2.2 Parabolic distributions

We are also interested in defining other density functions to do the sampling. The simplest density function to try would be to have all particles have the same velocity i.e. a Dirac delta function distribution. Another simple distribution would be a uniform distribution between two velocities, where a particle is equally likely to have any velocity in the range.

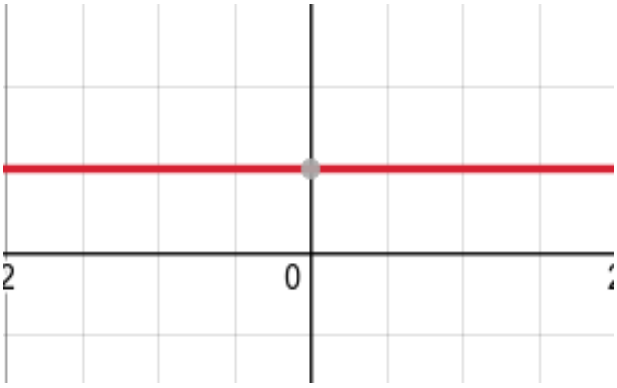

Figure 3: f_0

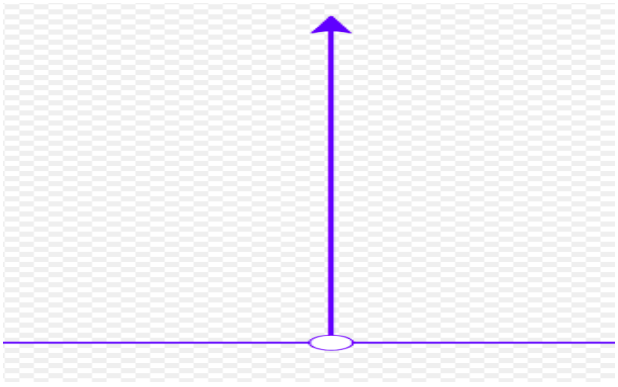

Figure 4: f_δ

Functions that look like f_0 might define he uniform distribution while functions like f_δ might describe the Dirac delta distribution. The figure for f_0 was generated using desmos graphing calculator while the figure for f_δ was snipped from the image in wikipedia for the Dirac delta function. We define initialization strategies based on uniform and Dirac delta distributions in our simulations.

Parabolic functions in a given range also seem like interesting distributions and yet simple to work with. After some trial and error, two functions that seem interesting are:

$$f_1 = \begin{cases} 1 - \frac{v^2}{2v_a^2} & -v_a \leq v \leq v_a \\ 0 & else \end{cases}$$

$$f_2 = \begin{cases} 1 + \frac{v^2}{2v_a^2} & v_a \leq v \leq v_a \\ 0 & else \end{cases}$$

These functions can be normalized as

$$\int_{v_x=-v_a}^{v_x=v_a} dv_x \int_{v_y=-v_a}^{v_y=v_a} dv_y \int_{v_z=-v_a}^{v_z=v_a} dv_z \; c_0 \left(1 - \frac{v_x^2 + v_y^2 + v_z^2}{2v_a^2}\right) = 1 \qquad \text{and}$$

$$\int_{v_x=-v_a}^{v_x=v_a} dv_x \int_{v_y=-v_a}^{v_y=v_a} dv_y \int_{v_z=-v_a}^{v_z=v_a} dv_z \; c_0 \left(1 + \frac{v_x^2 + v_y^2 + v_z^2}{2v_a^2}\right) = 1 \qquad \text{giving}$$

$$\hat{f}_1 = \begin{cases} \frac{1}{4v_a^3}\left(1 - \frac{v^2}{2v_a^2}\right) & -v_a \leq v \leq v_a \\ 0 & else \end{cases}$$

$$\hat{f}_2 = \begin{cases} \frac{1}{12v_a^3}\left(1 + \frac{v^2}{2v_a^2}\right) & v_a \leq v \leq v_a \\ 0 & else \end{cases}$$

The graph of these functions (considering v as a single variable) were generated using desmos graphing calculator. The figures represent $\hat{f}_1$ and $\hat{f}_2$ extrapolated to beyond the defined range.

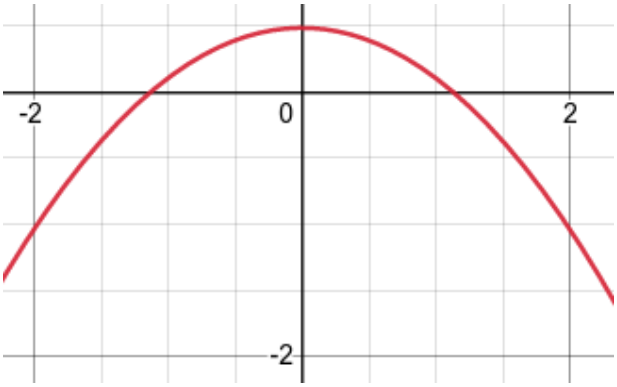

Figure 5: $\hat{f}_1$

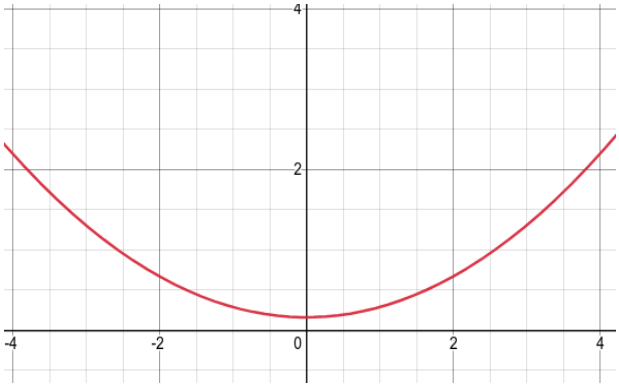

Figure 6: $\hat{f}_2$

In the upper half of the plane, $\hat{f}_1$ looks approximately like a Gaussian distribution which the Maxwellian is for a distribution of velocity (not speed) for a fixed temperature. $\hat{f}_2$ is sort of the opposite of $\hat{f}_1$. But given that the functions are defined in a limited range, the integrals do not diverge.

6.2.3 A justification for Parabolic density functions - Vlasov equation

One good exercise is to check what happens when plugging in $\hat{f}_1$, $\hat{f}_2$ and f_M in the collisionless Vlasov equation. As discussed earlier Kinetic theory of plasma describes the system with a density function. The dynamics of the plasma is described by the changing of the density function. Earlier, Boltzmann equation was discussed as an equation used

to describe this dynamics. Vlasov equation is an instance of the collisionless Boltzmann equation discussed earlier; where it is assumed that the particles do not interact with each other, and the force exerted on the particles is described by the Lorentz force. The Vlasov equation is written as:

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \nabla f + \frac{q}{m} \left(\mathbf{E} + \boldsymbol{v} \times \mathbf{B} \right) \cdot \partial_{\boldsymbol{v}} f = 0$$

For all three of the density functions

$$\frac{\partial f}{\partial t} = 0, \qquad \frac{\partial f}{\partial x} = 0, \qquad \frac{\partial f}{\partial y} = 0, \qquad \frac{\partial f}{\partial z} = 0$$

as all three have no explicit dependence on position or time. The Vlasov equation then becomes

$$\left(\mathbf{E} + \boldsymbol{v} \times \mathbf{B} \right) \cdot \partial_{\boldsymbol{v}} f = 0$$

which is

$$\left(E_x + v_y B_z - v_z B_y \right) \frac{\partial f}{\partial v_x} + \left(E_y + v_z B_x - v_x B_z \right) \frac{\partial f}{\partial v_y} + \left(E_z + v_x B_y - v_y B_x \right) \frac{\partial f}{\partial v_z} = 0$$

For the Maxwellian,

$$\widehat{f_M} := \hat{f}(\boldsymbol{x}, \boldsymbol{v}, t) = \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \exp\left(-\frac{v^2}{v_{th}^2} \right)$$

$$\frac{\partial f_M}{\partial v_x} = \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \exp\left(\frac{-v^2}{v_{th}^2} \right) \left(\frac{-2v_x}{v_{th}^2} \right), \qquad \frac{\partial f_M}{\partial v_y} = \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \exp\left(\frac{-v^2}{v_{th}^2} \right) \left(\frac{-2v_y}{v_{th}^2} \right),$$

$$\frac{\partial f_M}{\partial v_z} = \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \exp\left(\frac{-v^2}{v_{th}^2} \right) \left(\frac{-2v_z}{v_{th}^2} \right)$$

plugging in, the equation becomes

$$\left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \exp\left(\frac{-v^2}{v_{th}^2} \right) \left(\frac{-2}{v_{th}^2} \right) \left[v_x \left(E_x + v_y B_z - v_z B_y \right) + v_y \left(E_y + v_z B_x - v_x B_z \right) + \right.$$

$$\left. v_z \left(E_z + v_x B_y - v_y B_x \right) \right] = 0$$

which gives

$$v_x E_x + v_y Ey + v_z E_z = 0$$

which can also be written as

$$\boldsymbol{v} \cdot \mathbf{E} = 0$$

which says that the particles move perpendicular to the electric field which is characteristic of the Lorentz force. For the parabolic functions,

$$\hat{f}_1 = \begin{cases} \frac{1}{4v_a^3} \left(1 - \frac{v^2}{2v_a^2} \right) & -v_a \leq v \leq v_a \\ 0 & else \end{cases}$$

$$\hat{f}_2 = \begin{cases} \frac{1}{12v_a^3}\left(1+\frac{v^2}{2v_a^2}\right) & v_a \leq v \leq v_a \\ 0 & else \end{cases}$$

$$\frac{\partial \hat{f}_1}{\partial v_x} = \begin{cases} \frac{1}{4v_a^3}\left(\frac{-2v_x}{2v_a^2}\right) & -v_a < v < v_a \\ \text{undefined} & v = -v_a, v = v_a \\ 0 & else \end{cases}$$

$$\frac{\partial \hat{f}_1}{\partial v_y} = \begin{cases} \frac{1}{4v_a^3}\left(\frac{-2v_y}{2v_a^2}\right) & -v_a < v < v_a \\ \text{undefined} & v = -v_a, v = v_a \\ 0 & else \end{cases}$$

$$\frac{\partial \hat{f}_1}{\partial v_z} = \begin{cases} \frac{1}{4v_a^3}\left(\frac{-2v_z}{2v_a^2}\right) & -v_a < v < v_a \\ \text{undefined} & v = -v_a, v = v_a \\ 0 & else \end{cases}$$

$$\frac{\partial \hat{f}_2}{\partial v_x} = \begin{cases} \frac{1}{12v_a^3}\left(\frac{2v_x}{2v_a^2}\right) & -v_a < v < v_a \\ \text{undefined} & v = -v_a, v = v_a \\ 0 & else \end{cases}$$

$$\frac{\partial \hat{f}_2}{\partial v_y} = \begin{cases} \frac{1}{12v_a^3}\left(\frac{2v_y}{2v_a^2}\right) & -v_a < v < v_a \\ \text{undefined} & v = -v_a, v = v_a \\ 0 & else \end{cases}$$

$$\frac{\partial \hat{f}_2}{\partial v_z} = \begin{cases} \frac{1}{12v_a^3}\left(\frac{2v_z}{2v_a^2}\right) & -v_a < v < v_a \\ \text{undefined} & v = -v_a, v = v_a \\ 0 & else \end{cases}$$

In the range $v \in \mathbb{R} \backslash [-v_a, v_a]$ the equation becomes $0 = 0$ which is trivial. The equation becomes undefined for $v = v_a$ and $v = -v_a$ but we can ignore that for now. In the interesting range of $v \in (-v_a, v_a)$, the equation becomes

$$\frac{1}{4v_a^3}\left(\frac{-2}{2v_a^2}\right)\left[v_x\left(E_x + v_y B_z - v_z B_y\right) + v_y\left(E_y + v_z B_x - v_x B_z\right) + v_z\left(E_z + v_x B_y - v_y B_x\right)\right] = 0$$

for $\hat{f}_1$ and

$$\frac{1}{12v_a^3}\left(\frac{2}{2v_a^2}\right)\left[v_x\left(E_x + v_y B_z - v_z B_y\right) + v_y\left(E_y + v_z B_x - v_x B_z\right) + v_z\left(E_z + v_x B_y - v_y B_x\right)\right] = 0$$

for $\hat{f}_2$ which both give the same equation as $\widehat{f_M}$

$$v_x E_x + v_y Ey + v_z E_z = 0 \qquad or \qquad \boldsymbol{v} \cdot \mathbf{E} = 0$$

$\hat{f}_1$ and $\hat{f}_2$ behave similar to $\hat{f_M}$ when plugged into the Vlasov equation.

6.2.4 A justification for Parabolic density functions - Magnetic mirror

Since the quantity $\frac{sin^2\theta}{B_z}$ is conserved in a magnetic mirror, it is a good exercise to calculate the average value or expectation of $sin^2\theta$, $\langle sin^2\theta \rangle = \left\langle \frac{v_\perp^2}{v^2} \right\rangle = \left\langle \frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2} \right\rangle$ For the Maxwellian $\widehat{f_M}$,

$$\left\langle \frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2} \right\rangle = \left(\frac{m}{2\pi KT}\right)^{\frac{3}{2}}$$

$$\int_{v_x=-\infty}^{v_x=\infty} dv_x \int_{v_y=-\infty}^{v_y=\infty} dv_y \int_{v_z=-\infty}^{v_z=\infty} dv_z \left(\frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2}\right) \exp\left(-\frac{v_x^2}{v_{th}^2}\right) \exp\left(-\frac{v_y^2}{v_{th}^2}\right) \exp\left(-\frac{v_z^2}{v_{th}^2}\right)$$

Likewise for the parabolic functions,

$$\left\langle \frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2} \right\rangle = \frac{1}{4v_a^3}$$

$$\int_{v_x=-v_a}^{v_x=v_a} dv_x \int_{v_y=-v_a}^{v_y=v_a} dv_y \int_{v_z=-v_a}^{v_z=v_a} dv_z \left(\frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2}\right) \left(1-\frac{v_x^2}{2v_a^2}-\frac{v_y^2}{2v_a^2}-\frac{v_z^2}{2v_a^2}\right)$$

for $\hat{f}_1$ and

$$\left\langle \frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2} \right\rangle = \frac{1}{12v_a^3}$$

$$\int_{v_x=-v_a}^{v_x=v_a} dv_x \int_{v_y=-v_a}^{v_y=v_a} dv_y \int_{v_z=-v_a}^{v_z=v_a} dv_z \left(\frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2}\right) \left(1+\frac{v_x^2}{2v_a^2}+\frac{v_y^2}{2v_a^2}+\frac{v_z^2}{2v_a^2}\right)$$

for $\hat{f}_2$ which are all very difficult to evaluate because of the denominators.
So an approximation $\left\langle \frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2} \right\rangle = \frac{\langle v_x^2+v_y^2 \rangle}{\langle v_x^2+v_y^2+v_z^2 \rangle} + c_0$ can be done where c_0 is an error term.

For $\widehat{f_M}$, $\langle v_x^2+v_y^2 \rangle = \frac{2KT}{m}$ and $\langle v_x^2+v_y^2+v_z^2 \rangle = \frac{3KT}{m}$ so

$$\langle sin^2\theta \rangle = \left\langle \frac{v_\perp^2}{v^2} \right\rangle = \left\langle \frac{v_x^2+v_y^2}{v_x^2+v_y^2+v_z^2} \right\rangle = \frac{\langle v_x^2+v_y^2 \rangle}{\langle v_x^2+v_y^2+v_z^2 \rangle} + c_0 = \frac{2}{3} + c_0$$

For $\hat{f}_1$, $\langle v_x^2+v_y^2 \rangle = \frac{22}{45}v_a^2$ and $\langle v_x^2+v_y^2+v_z^2 \rangle = \frac{11}{15}v_a^2$ so $\langle sin^2\theta \rangle = \frac{2}{3} + c_0$ and
For $\hat{f}_2$, $\langle v_x^2+v_y^2 \rangle = \frac{98}{135}v_a^2$ and $\langle v_x^2+v_y^2+v_z^2 \rangle = \frac{49}{45}v_a^2$ so $\langle sin^2\theta \rangle = \frac{2}{3} + c_0$
$\hat{f}_1$ and $\hat{f}_2$ behave similar to $\widehat{f_M}$ when plugged into the expression for $\langle sin^2\theta \rangle$.

Since $\hat{f}_1$ and $\hat{f}_2$ behave similar to $\widehat{f_M}$ when plugged into the Vlasov equation and the expression for $\langle sin^2\theta \rangle$, it seems that $\hat{f}_1$ and $\hat{f}_2$ are nice distribution functions to work with.

6.3 Fields

In order to define the plasma chamber and hence the control on the plasma, we are interested in defining a few different electric and magnetic field configurations. We have a few different field configurations in mind. For the electric field, as of now we have 2 field configurations:

1. **Uniform electric field**
   The electric field has the same value everywhere. It is a very simple configuration to think about and use.

2. **Electric field due to an electrode**
   We describe the electrode in the following way: Every particle sees the electrode create an electric field created by a pair of capacitor plate at a distance taken only in the x-y plane. This is to say that every point on the electrode; now assumed to be a line along the z-direction, creates an electric field like that created between a pair of capacitor plates, if there is a particle at the same z-coordinate as that point. So currently we use the expression $\mathbf{E} = V \cdot d\boldsymbol{r}$ where $d\boldsymbol{r}$ is a vector whose z-coordinate is zero.

For the magnetic field, we have 3 configurations as of now:

1. **Uniform magnetic field**
   The uniform magnetic field has the same value everywhere. It is very simple to understand and use.

2. **Magnetic field due to a Helmholtz coil**
   The magnetic field strength due to a Helmholtz coil is given by the expression:

   $$B = \left(\frac{4}{5}\right)^{3/2} \frac{\mu_0 n I}{R}$$

   We may use the axis of the coil as the direction of the field.

3. **Magnetic field due to 2 Helmholtz coils**
   The magnetic field due to two Helmholtz coils is calculated by simply adding the magnetic fields created by the two coils.

6.4 Running different batches

As discussed earlier we would like to describe different batches of particles, so as to model plasma streams; where the particles are sent into the chamber based on different initialization strategies. The particles are then evolved based on the Boris algorithm. We would then remove the particles to simulate particles exiting the chamber, or being absorbed to form coatings as in different surface engineering processes like Magnetron Sputtering.

7 Work carried out so far

The simulation is programmed in jupyter notebook files (.ipynb extensions) that run python and markdown. The notebooks are available in the following Github repository, in the notebooks folder.

GitHub repository:

https://github.com/18BME2104/MagneticMirror

Algorithm Outline

Different aspects of the program are discussed based on the different notebooks (ipynb files) that describe them.

Required functionalities and Files:

1. Constants - **constants.ipynb**
2. Particle - **particle.ipynb**
3. Electric and Magnetic fields - **field.ipynb**
4. Particle initialization - **sampling.ipynb**
5. Updating the particles - **step.ipynb**
6. Batches of updates - **run.ipynb**
7. Plotting - **plot.ipynb**

7.1 Constants - constants.ipynb

The **constants.ipynb** notebook describes some constants useful in the program. Some useful constants are e (electron charge) m_e (electron mass), charges and masses of ions in the plasma, $a.m.u$ (atomic mass unit), N_A (Avogadro's number), ϵ_0 (permittivity of vacuum), μ_0 (permeability of vacuum), K or k_B (Boltzmann's constant), etc.

```
1  class Constants:
2      def __init__(self):
3          self.constants = {
4              #'symbol': ['value', unit', 'digits', '10 ^ power', 'number of
                   digits', 'description'],
5              }
6      def show_constant(self, symbol):
```

7.2 Particle - particle.ipynb

The **particle.ipynb** notebook describes the state of a particle; its position, velocity, mass, charge, name, and optionally acceleration (which is set to 0 as default if it is not required to track the acceleration of a particle) as of now.

Currently the **Boris algorithm** as discussed earlier, is an update strategy defined to update the state of a particle. Other strategies could be defined in new functions in the class. However, Boris algorithm is good enough for us to get started.

```
1  class Particle:
2      #... other things
3
4      def Boris_update(self, afield, argsE, argsB):
5          # Define q_prime
6          q_prime = (self.charge / self.mass) * (dt / 2)
7
8          # Get E and B fields from the afield argument by passing in the
               current position of the particle
9          argsE = V, center
10         E = afield.get_E_field(self.r, V, center)
11         argsB = n, I, R, B_hat, mu_0
12         B = afield.get_B_field(self.r, n, I, R, B_hat, mu_0)
13
14         #Boris velocity update
15         v_minus = self.v + q_prime * E
16         v_plus = v_minus + q_prime * 2 * np.cross(v_minus, B)
17         v_new = v_plus + q_prime * E
18
19         self.v = v_new
20
21
22         #could have also done:
23         #self.v += (2 * q_prime) * (E + np.cross( (self.v + q_prime * E),
               B))
24
25         #update position
26         self.r += v_new * dt
27
28     #... some other things
```

7.3 Electric and Magnetic fields - field.ipynb

Electric and Magnetic field configurations described in **field.ipynb**.

Currently **Uniform Electric field** and the **Radial Electric field** (the field depends on the particle's position) created by an electrode are available to set up electric fields. **Uniform Magnetic field** and Magnetic field created by a **Helmholtz coil** and that by two Helmholtz coils are available.

Other Electric and Magnetic fields can be defined as functions in this class. These fields could be based on modeling of apparatus used to create electric or magnetic fields such as coils, or based on analytic expressions.

```
class Field:
    #... something

    def uniform_E_field(self, E):
        return E

    def radial_E_field(self, r, V, center = [0,0,0]):
        #Get the distance vector of the particle from the electrode
        dr = [(r[0] - center[0]), (r[1] - center[1]), 0]
        #Get the electric field
        E = V * dr
        return E

    def uniform_B_field(self, B):
        return B

    def helmholtz_coil_B_field(self, n, I, R, B_hat, mu_0):
        return ( (4/5)**1.5 * ( (mu_0 * n * I) / (R) ) * B_hat)

    def two_helmholtz_B_field(self, n1, I1, R1, B1_hat, n2, I2, R2, B2_hat
        , mu_0):
        B1 = helmholtz(self, n1, I1, R1, mu_0, B1_hat)
        B2 = helmholtz(self, n2, I2, R2, mu_0, B2_hat)

        #Calculate the resultant of two magnetic fields
        B_hat = B1_hat + B2_hat
        return B_hat

    #...something else
```

7.4 Particle initialization - sampling.ipynb

The initial positions and velocities of the particles play an important role in the evolution of their state under the influence of electric and magnetic fields. The initial distribution of positions and velocities define where the particles start in the setup (or lab apparatus); for example where they are injected into a sputtering chamber through valves, and what velocities they start with; for example what potential they are accelerated through or what parameters were used for the pumps used to pump the particles in.

The tracking of initial distribution is also important in determining how the magnetic mirror effect is observed in the plasma. The Sampler class samples initial positions and velocities for a given number of particles based on some available schemes.

Currently positions can be sampled such that all particles start at the **same position** (like for example if injected through a port), **at the same distance** but randomly distributed in angular positions relative to some point(origin as of now). Velocities can be sampled such

that particles have the **same speed**, **same velocity** or **Maxwellian distributed speeds** or **Maxwellian distributed velocities**.

```python
class Sampler:
    #... something

    def sample_same_given_position(self, r, n):
        positions = []
        for i in range(n):
            positions.append(r)

        return np.array(positions)

    def sample_same_given_distance_all_random_direction(self, d, n):
        positions = []

        for i in range(n):
            positions.append(d * uniform_random_unit_vector())

        return np.array(positions)

    def sample_same_given_velocity_same_direction(self, v, n):
        velocities = []
        for i in range(n):
            velocities.append(v)

        return np.array(velocities)

    def sample_same_given_speed_all_random_direction(self, s, n):
        velocities = []

        for i in range(n):
            velocities.append(s * uniform_random_unit_vector())

        return np.array(velocities)

    def sample_velocity_uniformKE_same_given_direction(self):
        pass

    def sample_velocity_uniformKE_all_random_directions(self, n):
        pass

    def sample_Maxwellian_speed(self, v_median, K, T, m, n):
        alpha = math.sqrt(K * T / m)
        speeds = stats.maxwell.rvs(loc = v_median, scale = alpha, size = n
            )
        return speeds

    def sample_Maxwellian_velocity_same_given_direction(self, v_median, K,
        T, m, v_hat, n):
        speeds = sample_Maxwellian_speed(self, v_median, K, T, m, n)
        velocities = np.outer(speeds, v_hat)

        return np.array(velocities)
```

```
50
51      def sample_Maxwellian_velocity_same_random_direction(self, v_median, K
            , T, m, n):
52          speeds = sample_Maxwellian_speed(self, v_median, K, T, m, n)
53          direction = self.uniform_random_unit_vector()
54          velocities = np.outer(speeds, direction)
55
56          return np.array(velocities)
57
58      def sample_Maxwellian_velocity_all_random_direction(self, v_median, K,
            T, m, n):
59          speeds = sample_Maxwellian_speed(self, v_median, K, T, m, n)
60          velocities = []
61          for i in range(n):
62              velocities.append(speeds[i] * np.array(
                    uniform_random_unit_vector()))
63
64          return np.array(velocities)
65
66      def sample_parabolic_velocity(self):
67          pass
68      #... something else
```

The initial positions and velocities could be passed around in lists but to be able to reuse some generated samples, and avoid sampling all the time, it is convenient to write sampled positions and velocities to files (csv format seems convenient). This functionality is described as:

```
1   class Sampler:
2       #... something
3       def write_to_csv_file(self, ...):
4
5           # write array to csv file
6
7       def write_file_name(self, ...):
8
9           # write the file name to the list of available files
10
11      #... something else
```

7.5 Updating the particles - step.ipynb

The Step class is concerned with **initializing particles**, **defining the fields**, and **updating the states of the particles** (positions and velocities) under the action of electric and magnetic fields.

```
1   class Step:
2
3       # ... something
4
5    ### PARTICELS INTIALIZATION SECTION
```

```
6       def initialize_particles(self, names, q_s, m_s, r_0_s, v_0_s, a_0_s, n
            ):
7
8           for i in range(n):
9               self.particles.append(Particle(names[i], q_s[i], m_s[i], r_0_s
                    [i], v_0_s[i], a_0_s[i] ))
10
11  ### FIELDS INITIALIZATION SECTION
12      def initialize_fields(self):
13          self.fields = Field()
14
15  ### TIME STEP SECTION
16      def update_particles(self, dt, argsE, argsB):
17
18          for particle in self.particles:
19              particle.update(self.fields, dt, argsE, argsB)
20      # ... something else
```

To use sampled positions and velocities that have been saved in csv files, some reading functionality is useful to load these positions and velocities to be used during initialization.

```
1   class Step:
2
3       # ... something
4       def read_r_or_v_file_and_reshape(self, index):
5
6           array_from_file = self.read_r_or_v_file(index)
7           reshaped_array = self.reshaper(array_from_file)
8           return reshaped_array
9
10      # ... something else
```

8 Work to be done

Work on the **constants.ipynb**, **particle.ipynb**, **fields.ipynb** and **sampling.ipynb** has been done to a certain extent. Basic work on step.ipynb has been done. However, we could modify functionalities defined here and add other definitions as work on **run.ipynb** progresses, so that everything works well together.
Other functionalities that could be added are:

1. **Sampling**
   Sampling strategies based on parabolic density functions discussed earlier are yet to be defined. New sampling strategies such as one initializing particles with the same Kinetic energy; as would happen if particles (of possibly different species) were accelerated through the same potential difference, could be defined.

2. **Fields**
   New field configurations could be defined. We could also modify existing field con-

figuration like the Electric field due to an electrode; if we find a different expression to describe it better, or if the current expression does not work well.

However, we do not yet have a plasma simulation running. The major portion of the work to be done to achieve this includes working on the **run.ipynb** notebook. To be able to understand the simulation; to check if works correctly and to understand the plasma behavior, we also need to define functionalities to plot different parameters like particle positions and velocities. These functionalities will be defined in **plot.ipynb** notebook, which can be imported into **ryn.ipynb** and **step.ipynb** notebooks to plot these quantities when the simulation is running, or from saved data.

8.1 Batches of updates - run.ipynb

The Run class is concerned with running the steps defined by the Step class in step.ipynb. This includes creating batches of particles, updating them under the influence of electric and magnetic fields and removing them if they move out of the region of interest or for example are absorbed during a coating process. New batches of particles can be created to model the flow of particles as in a plasma chamber setup. Functionality to change the fields; for example changing the Voltages in electrodes or Currents in the coils, are also defined. This better models the control of plasma supply and electric and magnetic field control apparatus as in a laboratory.

```
1  class Run:
2
3      # ... something
4      def create_particles(self):
5          pass
6
7      def update_particles(self):
8          pass
9
10     def remove_particles(self):
11         # This might include particles moving outside the chamber (the
               Electric and Magnetic fields or
12         # simply positions of interest) or particles being absorbed for
               example in a coating process
13         pass
14
15     def create_fields(self):
16         pass
17
18     def change_fields(self):
19         pass
20
21     #... something else
```

8.2 Plotting - plot.ipynb

The plot class is used to define functionalities to draw particle positions, plot positions and velocities of particles, as the system evolves like a lab apparatus. These functionalities are imported into run.ipynb; and if required other notebooks, to be used when the simulation is running; or optionally from saved data. Work on this notebook is yet to be started.

8.3 Understanding of the plasma

The big question, is that once we have a simulation running, how do we do analysis on the plasma so that the project helps improve our understanding of a plasma system? The sampling aspect of the project, will allow us to understand how different parameters of the plasma like the temperature; when it enters the chamber affect the behavior of the plasma. We can also study for example, the velocity distribution of the particles as they evolve in the chamber. Studying different field configurations; even simple functionalities like being able to change the Voltage of an electrode to change the electric field- like one might with a real apparatus, will help us understand the behavior of plasma under different or changing fields.
Such strategies will allow us to understand how to contain a plasma in chamber; and one such instance would be a magnetron sputtering chamber where one could study the magnetic mirror effect. The magnetic mirror effect in a magnetron sputtering chamber serves as one example of studies we could do on the plasma parameters, studying a plasma in simulation. So our title serves as an example for what we could do with such a project. We have learned through the course of doing the project that more could be achieved with such a system than what we initially planned to focus on.

9 Gantt Chart (Work Plan)

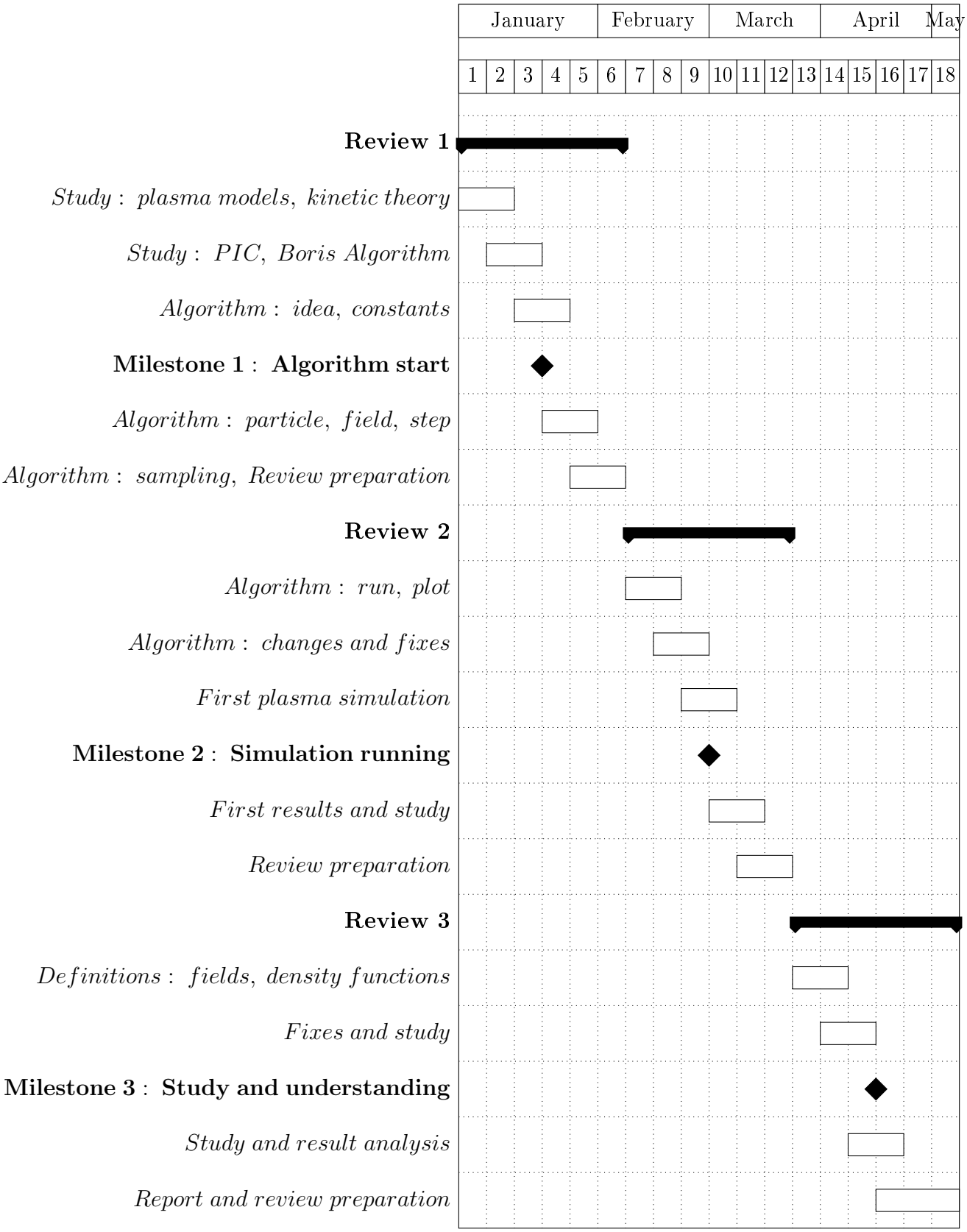

Figure 7: Gantt chart

Progress description:

- **Review 1**

1. Weeks 1-2 **Study: plasma models, kinetic theory**
   We first studied basic plasma physics, and a bit more on the kinetic theory of plasma as that seemed relevant to our project.

2. Weeks 2-3 **Study: PIC, Boris Algorithm**
   We then studied particle in cell methods and then the Boris Algorithm.

3. Weeks 3-4 **Algorithm: idea, constants**
   With some idea of how to proceed with the project, we started working on the algorithm. We started by with work on the **constants.ipynb** notebook, defining the constants that might be needed for the project.

4. Weeks 4-5 **Algorithm: particle, field, step**
   We then worked on the notebook **particle.ipynb** where we defined a particle and how to update it. In the **field.ipynb** file, we worked to define some basic field configurations. Then we worked on **step.ipynb** to define steps of evolutions of the particles in a field configuration. This is the basic work on single particle dynamics.

5. Weeks 5-6 **Algorithm: sampling, Review preparation**
   We then worked on some kinetic theory aspects of the project, defining some sampling strategies based on the Uniform velocity and Maxwellian distribution. We then started preparing for review 1.

- **Review 2**

1. Weeks 7-8 **Algorithm: run, plot**
   We plan to work next on **run.ipynb** to define running of batches of particle evolution and on **plot.ipynb** to define plot functionalities.

2. Weeks 8-9 **Algorithm: changes and fixes**
   We anticipate that there may be errors in the programs that we will need some time to debug.

3. Weeks 9-10 **First plasma simulation**
   We then expect to have the first plasma simulation running.

4. Weeks 10-11 **First results and study**
   Running some plasma simulations, we intend to study the first set of results.

5. Weeks 11-12 **Review preparation**
   We plan to prepare for review 2 based on some preliminary results and understanding.

- **Review 3**

1. Weeks 13-14 **Definitions: fields, density functions**
   We then intend to make some new definitions like sampling based on the parabolic density functions, some new field configurations and possibly some new functionality handling particle states.

2. Weeks 14-15 **Fixes and study**
   We expect to spend some time making final fixes and changes on the algorithm. We then turn to studying the results.

3. Weeks 15-16 **Study and result analysis**
   We intend to spend some time studying the results and trying to understand the plasma model in the simulation.

4. Weeks 16-18 **Report and review preparation**
   We then expect to start preparing for review 3, and preparation of the report and other documents.

10 Milestones in the project phase

Milestone 1 **Algroithm start** Week 3 - **Achieved**
Around week 3 we had some understanding of basic plasma physics and so started working on how to formulate the problem as an algorithm. Before review 1

Milestone 2 **Simulation runnning** Week 9 - **Expected**
We expect to have the simulation running around week 9 of the project. This will allow us to have some preliminary results and understanding of our model, by review 2. Before review 2

Milestone 3 **Study and understanding** Week 15 - **Expected**
We intend to perform study on plasma using our algorithm and improve our understanding of a plasma and how to control and study a plasma system before the end of the project. Before review 3

References

[1] Professor Sitaram Dash. (Fall Semester 2021). *MEE4005 Surface Engineering* (lecture notes). SMEC, VIT Vellore.

[2] Matthew W. Kunz. (November 9, 2020). *Introduction to Plasma Astrophysics* (lecture notes). Princeton Plasma Physics Laboratory.

[3] Chen, F. F. (1984). *Introduction to plasma physics and controlled fusion* (Vol. 1, pp. 8-11). New York: Plenum press.

[4] Na, Yong-Su (2017). *Introduction to nuclear fusion* (Lecture 9 Mirror, lecture slide). Seoul National University Open Courseware.

[5] Föreläsning (2009). *Charged particle motion in magnetic field* (lecture slide). Luleä University of Technology.

[6] Myers, A., Colella, P., & Straalen, B. V. (2017). *A 4th-Order Particle-in-Cell Method with Phase-Space Remapping for the Vlasov–Poisson Equation*. SIAM Journal on Scientific Computing, 39(3), B467-B485.

[7] Anatoly Spitkovsky (2016). *Kinetic plasma simulations* (lecture slide). PiTP 2016 on Computational Plasma Astrophysics. Institute for Advanced Study.

[8] Qin, H., Zhang, S., Xiao, J., & Tang, W. M. (April, 2013). *Why is Boris algorithm so good?*. Princeton Plasma Physics Laboratory, PPPL-4872.