

Magnetic Mirror Effect in Magnetron Plasma: Modeling of Plasma Parameters

1 Checks

To verify that our program is working correctly, we perform some checks. We compare results from the program with calculations done by hand; of certain quantities and see if they agree.

1.1 Sampling

As of now we our sampling of particle speeds is based on the Maxwell-Boltzmann distribution, that we have defined previously. We now check if the average speeds of the particles agrees with the calculations done by hand based on the plasma temperature. We recall the Maxwellian density function that we defined earlier.

$$\widehat{f}_M := \hat{f}(\mathbf{x}, \mathbf{v}, t) = \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \exp \left(-\frac{\mathbf{v}^2}{v_{th}^2} \right) \quad (1)$$

where

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

For our convenience, we use the density function with speed instead of velocity which is defined as:

$$\widehat{f}_m := \hat{f}(\mathbf{x}, v, t) = \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} 4\pi v^2 \exp \left(-\frac{v^2}{v_{th}^2} \right) \quad (2)$$

which is obtained by integrating over the solid angle in the velocity variable.

For this density function we calculate the average speed by with the expression:

$$\begin{aligned} \int_{v=-\infty}^{v=\infty} dv \, v \, \widehat{f}_m &= \int_{v=-\infty}^{v=\infty} dv \, v \, \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} 4\pi v^2 \exp \left(-\frac{v^2}{v_{th}^2} \right) \\ &= 4\pi \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \int_{v=-\infty}^{v=\infty} dv \, v^3 \exp \left(-\frac{v^2}{v_{th}^2} \right) \\ &= 4\pi \left(\frac{m}{2\pi KT} \right)^{\frac{3}{2}} \frac{v_{th}^4}{2} = 4\pi \left(\frac{1}{\pi v_{th}^2} \right)^{\frac{3}{2}} \frac{v_{th}^4}{2} \\ &= \frac{2}{\sqrt{\pi}} v_{th} = \frac{2}{\sqrt{\pi}} \sqrt{\frac{2KT}{m}} = \frac{2}{\sqrt{\pi}} \sqrt{\frac{2RT}{M}} = \end{aligned}$$

Task 1

do the calculation here ...

From section 1.1.1 Maxwellian sampling in the **studies.ipynb** notebook, we take the following:

Initialization

```
1 # c1Maxwell means checking the Maxwellian sampling
2 c1Maxwell = Run()
3 # Create 100 particles based on the data available in the files
4 c1Maxwell.create_batch_with_file_initialization('H+', constants.constants['e'][0],
  ↪ constants.constants['m_H'][0] * constants.constants['amu'][0], 100, 100, 'H ions',
  ↪ r_index=0, v_index=1)
```

Inspection

```
1 # Take the 0th batch of particles
2 c1Maxwell_batch = c1Maxwell.batches[0]['H ions']
3 # Take the initial positions and velocities of the particles
4 c1Maxwell_positions = []
5 c1Maxwell_velocities = []
6 for particle in c1Maxwell_batch.particles:
7     c1Maxwell_positions.append(particle.r)
8     c1Maxwell_velocities.append(particle.v)
9 # Let's now look at the velocities
10 c1Maxwell_velocities
11 # We need to check if they are really Maxwellian distributed
12 # Get the speeds
13 c1Maxwell_speeds = np.sqrt( \
14     [ (c1Maxwell_velocities[i][0] ** 2) + (c1Maxwell_velocities[i][1] ** 2) + \
15     (c1Maxwell_velocities[i][2] ** 2) for i in range(len(c1Maxwell_velocities)) ] )
16 c1Maxwell_meanspeed = np.sum(c1Maxwell_speeds) / c1Maxwell_speeds.size
```

We get an average speed of the distribution to be:

14202.764572898674

Task 2

Check if $\frac{2}{\sqrt{\pi}} \sqrt{\frac{2KT}{m}} = \frac{2}{\sqrt{\pi}} \sqrt{\frac{2RT}{M}}$ = is what works out based on what I have used for sampling ...

1.2 Update

We recall the Boris Algorithm, that we use to update the particles in the plasma simulation.

$$\begin{aligned}
\mathbf{v}^- &= \mathbf{v}_k + q' \mathbf{E}_k \\
\mathbf{v}^+ &= \mathbf{v}^- + 2q' (\mathbf{v}^- \times \mathbf{B}_k) \\
\mathbf{v}_{k+1} &= \mathbf{v}^+ + q' \mathbf{E}_k \\
\mathbf{x}_{k+1} &= \mathbf{x}_k + \Delta t \mathbf{v}_{k+1}
\end{aligned} \tag{3}$$

where $q' = \frac{q}{m} \frac{\Delta t}{2}$.

To verify that the simulation works as we expect it to, we perform a calculation and compare it to the output of an algorithm. From section 1.2.1 Boris Update in the **studies.ipynb** notebook, we take the following:

Initialization

```

1  # c2Boris means checking the Boris update
2  c2Boris = Run()
3  #Create 10 particles
4  c2Boris.create_batch_with_file_initialization('H+', constants.constants['e'][0],
    ↪ constants.constants['m_H'][0] * constants.constants['amu'][0], 100, 10, 'H ions',
    ↪ r_index=0, v_index=1)

```

Update input data

```

1
2  c2Boris_index_update = 0 # Update the first batch in this Run instance run_Boris_check
3  c2Boris_particle_track_indices = [i for i in range(10)] # Track all 10 particles
4  c2Boris_dT = 10**(-6) # 1 microseconds
5  c2Boris_stepT = 10**(-7) # 0.1 microseconds time step
6  c2Boris_E0 = 1000 # say 1000 Volts (voltage) per meter (size of chamber)
7  c2Boris_Edirn = [1,0,0] #in the x-direction [1,0,0]
8  c2Boris_B0 = 10 * (10**(-3)) # Meant to say 10 mT
9  c2Boris_Bdirn = [0,1,0] #in the y-direction [0,1,0]
10 c2Boris_argsE = [element * c2Boris_E0 for element in c2Boris_Edirn] # currently the
    ↪ uniform_E_field configuration is used
11 c2Boris_argsB = [element * c2Boris_B0 for element in c2Boris_Bdirn] # currently the
    ↪ uniform_B_field configuration is used

```

Update

```

1 c2Boris_positions_and_velocities =
  ↳ c2Boris.update_batch_with_unchanging_fields(c2Boris_index_update, c2Boris_dT,
  ↳ c2Boris_stepT, c2Boris_argsE, c2Boris_argsB, c2Boris_particle_track_indices)
2
3 #Let's inspect the positions and velocities of the particles at index 1 and 7
4 c2Boris_p1 = c2Boris_positions_and_velocities[1]
5 c2Boris_p7 = c2Boris_positions_and_velocities[7]
6
7 #Let's take the positions and velocities of the particle 1 after the 3rd and the 4th
  ↳ time steps.
8 c2Boris_p134_p = [c2Boris_p1[3][1], c2Boris_p1[4][1]]
9 c2Boris_p134_v = [c2Boris_p1[3][2], c2Boris_p1[4][2]]
10
11 #Similary for particle 7
12 c2Boris_p734_p = [c2Boris_p7[3][1], c2Boris_p7[4][1]]
13 c2Boris_p734_v = [c2Boris_p7[3][2], c2Boris_p7[4][2]]

```

The program gives the following output. For example, we can check the particles 1 and 7 in the update. After the 3rd step of the update, the velocity of particle 1 was:

```
[39771.83801956555, -5029.491038, 4533.343932509505]
```

and after the 4th update, it changed to:

```
[48909.86581773698, -5029.491038, 8798.399243869582]
```

Its position after the 3rd step of the update was:

```
[-0.48985112694809785, -0.0020117964152, 0.00017005183797547688]
```

which was updated after the 4th step of the update to:

```
[-0.4849601403663242, -0.0025147455189999997, 0.001049891762362435]
```

We now check if the same is true, with a calculation of the update done by hand.

$$q' = \frac{q}{m} \frac{\Delta t}{2} = \frac{1.602176634 \times 10^{-19} \text{ C}}{1.008 \text{ a.m.u.} \times 1.6605390666 \times 10^{-27} \text{ kg a.m.u.}^{-1}} \frac{10^{-7} \text{ s}}{2} = 4.78597877761177 \text{ C s kg}^{-1}$$

$$\mathbf{v}^- = \mathbf{v}_3 + q' \mathbf{E} = \begin{bmatrix} 39771.83801956555 \\ -5029.491038 \\ 4533.343932509505 \end{bmatrix} + 4.78597877761177 \begin{bmatrix} 1000 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 44557.81679718 \\ -5029.491038 \\ 4533.34393251 \end{bmatrix}$$

$$\mathbf{v}^+ = \mathbf{v}^- + 2q' (\mathbf{v}^- \times \mathbf{B}) = \begin{bmatrix} 44557.81679718 \\ -5029.491038 \\ 4533.34393251 \end{bmatrix} + 2 \cdot 4.78597877761177 \left(\begin{bmatrix} 44557.81679718 \\ -5029.491038 \\ 4533.34393251 \end{bmatrix} \times \begin{bmatrix} 0.0 \\ 0.01 \\ 0.0 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 44123.88704013 \\ -5029.491038 \\ 8798.39924387 \end{bmatrix}$$

$$\mathbf{v}_4 = \mathbf{v}^+ + q'\mathbf{E} = \begin{bmatrix} 44123.88704013 \\ -5029.491038 \\ 8798.39924387 \end{bmatrix} + 4.78597877761177 \begin{bmatrix} 1000 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 48909.86581774 \\ -5029.491038 \\ 8798.39924387 \end{bmatrix}$$

$$\mathbf{x}_4 = \mathbf{x}_3 + \Delta t \mathbf{v}_4 = \begin{bmatrix} -4.89851127 \times 10^{-01} \\ -2.01179642 \times 10^{-03} \\ 1.70051838 \times 10^{-04} \end{bmatrix} + 10^{-7} \begin{bmatrix} 48909.86581774 \\ -5029.491038 \\ 8798.39924387 \end{bmatrix} = \begin{bmatrix} -0.48496014 \\ -0.00251475 \\ 0.00104989 \end{bmatrix}$$

We see that the values for \mathbf{x}_3 , \mathbf{x}_4 , \mathbf{v}_3 and \mathbf{v}_4 for particle 1 are close (the latter digits differ due to the differing precision of calculations); i.e. the same when done by hand and in the program.

Similarly, for particle 7. After the 3rd step of the update, the velocity of the particle was:

$$[38895.85871094631, -8670.854777, 13914.969423620274]$$

and after the 4th update, it changed to:

$$[47135.881299118584, -8670.854777, 18096.176367366777]$$

Its position after the 3rd step of the update was:

$$[-0.4896669752432719, -0.0034683419108, 0.0038875496903932644]$$

which was updated after the 4th step of the update to:

$$[-0.48495338711336006, -0.0043354273885, 0.0056971673271299424]$$

$$\mathbf{v}^- = \mathbf{v}_3 + q'\mathbf{E} = \begin{bmatrix} 38895.85871095 \\ -8670.854777 \\ 13914.96942362 \end{bmatrix} + 4.78597877761177 \begin{bmatrix} 1000 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 43681.83748856 \\ -8670.854777 \\ 13914.96942362 \end{bmatrix}$$

$$\begin{aligned} \mathbf{v}^+ = \mathbf{v}^- + 2q'(\mathbf{v}^- \times \mathbf{B}) &= \begin{bmatrix} 43681.83748856 \\ -8670.854777 \\ 13914.96942362 \end{bmatrix} + 2 \cdot 4.78597877761177 \left(\begin{bmatrix} 43681.83748856 \\ -8670.854777 \\ 13914.96942362 \end{bmatrix} \times \begin{bmatrix} 0.0 \\ 0.01 \\ 0.0 \end{bmatrix} \right) \\ &= \begin{bmatrix} 42349.90252151 \\ -8670.854777 \\ 18096.17636737 \end{bmatrix} \end{aligned}$$

$$\mathbf{v}_4 = \mathbf{v}^+ + q'\mathbf{E} = \begin{bmatrix} 42349.90252151 \\ -8670.854777 \\ 18096.17636737 \end{bmatrix} + 4.78597877761177 \begin{bmatrix} 1000 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 47135.88129912 \\ -8670.854777 \\ 18096.17636737 \end{bmatrix}$$

$$\mathbf{x}_4 = \mathbf{x}_3 + \Delta t \mathbf{v}_4 = \begin{bmatrix} -0.48966698 \\ -0.00346834 \\ 0.00388755 \end{bmatrix} + 10^{-7} \begin{bmatrix} 47135.88129912 \\ -8670.854777 \\ 18096.17636737 \end{bmatrix} = \begin{bmatrix} -0.48495339 \\ -0.00433543 \\ 0.00569717 \end{bmatrix}$$

We see that the values for \mathbf{x}_3 , \mathbf{x}_4 , \mathbf{v}_3 and \mathbf{v}_4 for particle 7 are close (the latter digits differ due to the differing precision of calculations); i.e. the same when done by hand and in the program.

Based on these calculations, we believe that our particle update works as expected; according to the Boris Algorithm, and this part of the program works correctly. ✓

2 Plasma Stream

- Maxwellian distribution
- maybe Parabolic distribution
- Change Voltage of electrode to change electric field
- Change current in Helmholtz coil to change the magnetic field

while the updates are running. on different batches.

3 The Magnetic Mirror configuration

need parabolic B along z-axis configuration for $\nabla B \parallel B$