ADVANCED TOPICS IN COMPUTATIONAL PHYSICS
One dimensional plasma model

Filipe Cruz (n.87317), Óscar Amaro (n.87343)[1]

[1] *Contacts: filipe.d.cruz@tecnico.ulisboa.pt, oscar.amaro@tecnico.ulisboa.pt*

MEFT 2017 - Departamento de Física, Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal

October 2020

We implement the one-dimensional model of a plasma originally developed by John Dawson. We explain the specifics of our code and comparisons with physical theory.

I. INTRODUCTION

In order to study complicated plasma systems, one frequently develops a simplified model of the system in study, to be able to understand some of the properties of the plasma in that system. One of these models is the electrostatic one-dimensional model of a plasma proposed by John Dawson [1], which considers a large number of charge sheets embedded in a uniform neutralizing background. Here we consider the sheets to be perpendicular to the a single direction (in this case the x axis) where they are allowed to move.

In this work, we show the results of the implementation of the one-species model of a one-dimensional plasma model based on the Dawson's model [2] for numerical simulations, using multiple MATLAB scripts. It was used the MATLAB version R2020a for our simulations.

II. EQUATIONS

As described in [2], if a certain sheet labeled with i is displaced by x_i from its equilibrium condition, and if no collision between the sheets happens, then the equation of motion for the sheet will be $\ddot{x_i} = -\omega_p^2 x_i$, where ω_p is the plasma frequency, meaning it is equal to an harmonic oscillator. The solution is simply:

$$x_i(t) = x_i(0)\cos(\omega_p t) + (\dot{x_i}(0)/\omega_p)\sin(\omega_p t) \quad (1)$$

$$\dot{x_i}(t) = \dot{x_i}(0)\cos(\omega_p t) - \omega_p x_i(0)\sin(\omega_p t) \quad (2)$$

When two particles i and j a cross at t, we can calculate the collision time t_{c1} by interpolating linearly the positions in $t - dt$ and t:

$$\Delta t_{c1} = \Delta t \frac{x_j(t-\Delta t) - x_i(t-\Delta)}{x_j(t-\Delta) - x_i(t-\Delta) + x_i(t) - x_j(t)} \quad (3)$$

III. MODEL IMPLEMENTATION

A. Variables and initiation

For our simulation, we considered $\omega_p = 1.0$ for numerical simplifications. To define the system described by each of our simulations, multiple parameters need to be defined. These parameters are:

- `boxL` - the dimension of the box (in machine units);
- `nparts` - the number of particles/sheets;
- `tmax` - the maximum simulation time (units $1/\omega_p$);
- `dt` - the time step Δt (units $1/\omega_p$);
- `ndump` - the exportation frequency of the simulation data for diagnostics.

For our work, it was considered multiple values for each parameter. However the typical value for `boxL` is $\sim 40.$, `tmax` $\sim 50.$, `nparts`~ 2000 and `dt` is carefully chosen to be $\ll 1/\omega_p = 1$ and $\ll \delta/v$, where δ is the distance between sheets in the equilibrium condition and v is the typical velocity of the simulation (drag velocity or thermal velocity).

To execute the numerical operations for each time step, multiple vectors were creating, including:

- `vi0` and `xi0` - initial velocity and displacement from the position equilibrium for all particles;
- `vi` and `xi` - velocity and position for current time step;
- `xieq` - equilibrium positions.

The units of the velocities are the units of length times ω_p. The initial velocities are initiated with a specific velocity distribution while the initial positions are the equilibrium positions, so that all particles are distanced by dx by their closest neighbours (including the boundary particles, since boundaries are periodic).

The position vectors are kept ordered and each vector index represents the same particle.

B. Particle Push

To push particles in a new time step, when there are no collisions, it was possible to use the Dawson method described in [2], by using the equations 4 and 5 obtained from equations 1 and 2:

$$\dot{x_i}(t+\Delta t) = \dot{x_i}(t)\cos(\omega_p \Delta t) - \omega_p X_i(t)\sin(\omega_p \Delta t) \quad (4)$$

$$x_i(t+\Delta t) = x_i(t) + \dot{x_i}(t)\sin(\omega_p \Delta t) - X_i(t)(1-\cos(\omega_p \Delta t)) \quad (5)$$

where the values for each time steps are calculated from the values of the previous iteration. The strategy used for the particle push in our code, however, does not require to use the values from the last iteration, because it uses the equations 1 and 2, using instead the initial conditions. Using equations 1 and 2 rather than 4 and 5 is less time and access memory efficient, but leads to results with better resolution since is less dependent of the parameter `dt` and avoids calculating X_i.

When two particles suffers a collision, the equations used are no longer valid, so the initial conditions used in the equations must be updated to the values after the collision for that two particles, and the time t used, must be the difference between the time and the last collision time (registered in the vector `ti0`).

C. Collisions

The previous equations do not consider collisions between particles. Because the position vector are kept sorted, a collision occurs for the particle i and its neighbour particle $i+1$ if:

$$x_i(t) > x_{i+1}(t)$$

From the Dawson numerical model [2], the collision are considered by calculating the collision time t_{c1} from equation 3 with $j = i+1$, and use that time to calculate the particles values in $t + \Delta t_{c1}$ by equations 4 and 5. It then calculates the new collision time t_{c2} using the values from t and $t + \Delta t_{c1}$, and then calculate the values from the time t_{c2}. To calculate the corrected particle values of $t + \Delta t$ from t_{c2}, the Dawson model then adds an acceleration term $\pm\omega_p^2\delta$ to the equations, meaning a term $\pm\omega_p^2\delta t$ to the velocity and $\pm\omega_p^2\delta t^2/2$.

Our approach is similar, with the exception that, instead of calculating the time collision t_{c2} with the values of t and t_{c1}, we used the values t_{c1} and $t - dt$ as represented in figure 1, and instead of adding a term $\pm\omega_p^2\delta$, we simply swapped the values of the particles.

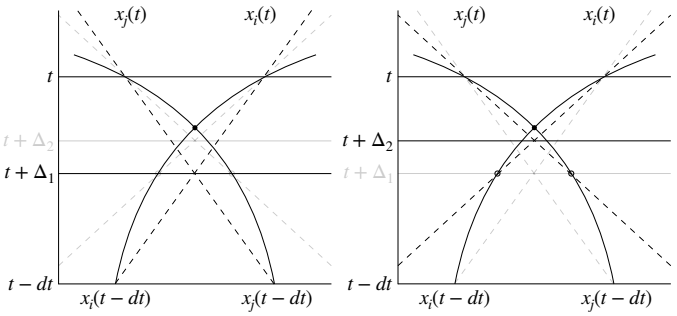

FIG. 1: Representation of our method for the calculation of the crossing time. (left) Calculation of t_{c1}. (right) Calculation of t_{c2}.

D. Periodic Boundaries

Our simulation considered boundary particles. Because we are assuming that each particle can only collide with only one other particle and because the physics of the system doesn't change if a particle passes the boundary of the system and no collision happens, we only need to execute the the collision between the two boundary particles to apply boundary conditions. This collision will occur if:

$$x_{last}(t) - x_{first}(t) > \texttt{boxL} \quad (6)$$

After adding a cycle to execute the collisions between intermediate particles, we only then need to apply the same procedure for the collisions where $i = 1$ (first particle) and $j =$ `Nparts` (last particle).

For most of the diagnostics, we only

E. Energy conservation

As we have seen previously, the method to get the time of collision depends on the precision dt. In particular, dx/vth is the typical time between collisions in a waterbag with vth, so for $dt < 0.1dx/vth$, the energy changes about $\sim 10^{-7}$ in relative terms, but for $dt \sim dx/vth$, the energy starts to deviate significantly from the initial value. Throughout the simulation the total energy is calculated and the maximum relative error is saved and plotted.

The parameters used in the simulations were `nparts` $= 1000$, `vth` $= -0.3$, `boxL` $= 40$, `tmax` $= 10$. The maximum relative error in energy seems to grow exponentially with dt for this specific setup, which means it there is a significant increase in accuracy with a modest increase in computation time.

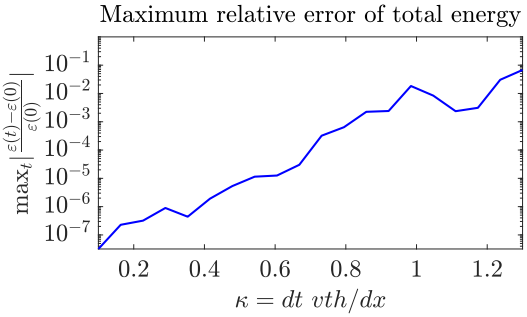

FIG. 2: Relative error in energy as a function of the time step.

IV. VELOCITY DISTRIBUTION

For this work, it was considered three different initiations of the particle thermal velocities: (i) cold plasma, with the thermal velocities equal to zero; (ii) *waterbag* function distribution obtained and (iii) Gaussian function distribution. The initiation for the last two cases is represented in figure 3.

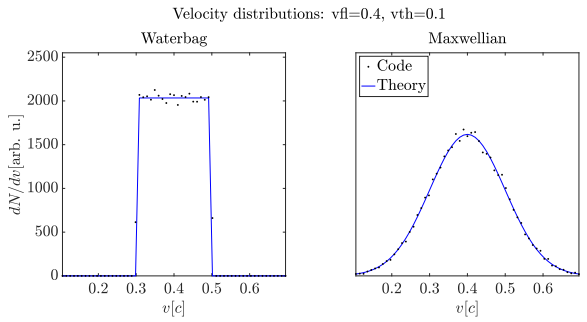

FIG. 3: Velocity distribution: (left) *Waterbag* distribution from -vth to vth. (right) Gaussian distribution.

From theory, we expect that for a system initiated with a *waterbag* distribution, will evolve itself to a gaussian distribution. This can be seen in figure 4.

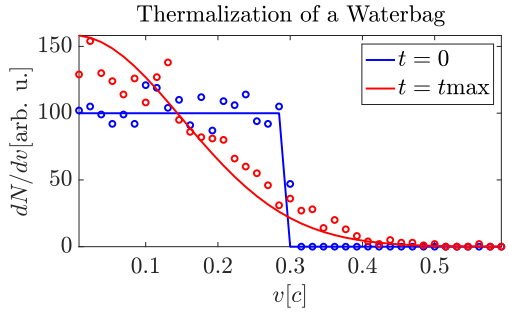

FIG. 4: Thermalize

To obtain these results, we ran a simulation with parameters: `nparts`= 2000, `boxL` = 10, `vth` = 0.3 (waterbag), `tmax` = 12000 ω_p^{-1}, `dt` = 0.4∗`dx`/`vth`. To improve the matching of the simulation results with theory one needs to increase both the number of particles and the duration of the simulation.

V. WAKEFIELD

A. Drag on a Fast Sheet

If we inject a fast sheet in a cold plasma, as it collides with the stationary particles it will decrease in velocity according to:

$$\frac{dv}{dt} = -\frac{\delta}{2} \tag{7}$$

The drag is thus independent of the sheet velocity, and in the process the sheet excites a plasma oscillation in its wake.

We keep track of the "fast sheet" as it looses energy by saving its index in the array of plasma particles. In figure 5 we show the time evolution of the sheet velocity for different initial conditions.

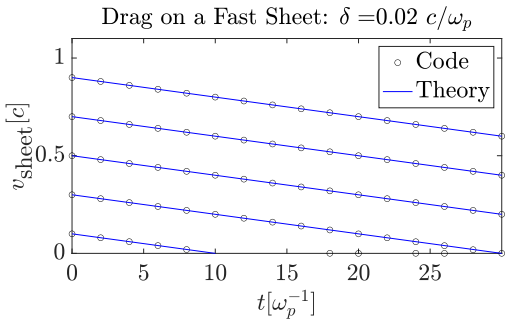

FIG. 5: Time evolution of velocity decay of a fast sheet. The slope of the curves is constant as it only depends on the linear plasma density.

The results are consistent with theory.

B. Electric field

In the previous subsection we saw the evolution of a self-consistent system. Now we impose that the fast sheet always has a velocity v, which means we are constantly adding more energy to the system. This case is studied in [3]. Linearizing the continuity, momentum and Gauss equations, we obtain:

$$\begin{cases} \partial_t v + E = 0 \\ \partial_x v + \partial_t n = 0 \\ \partial_x E + n + \delta(x - v_b t) = 0 \end{cases} \tag{8}$$

This set of equations reduces to:

$$\partial_{tt} n + n + \delta(x - v_b t) = 0 \tag{9}$$

This equation has the solution $n(x,t) = n_0 \sin(k_p x - t)\delta(x - v_b t)$, where here δ is the Dirac-delta. This functional dependence on the coordinate is shared with the electric field and the velocity, albeit with different phases. This can be seen in figure 6.

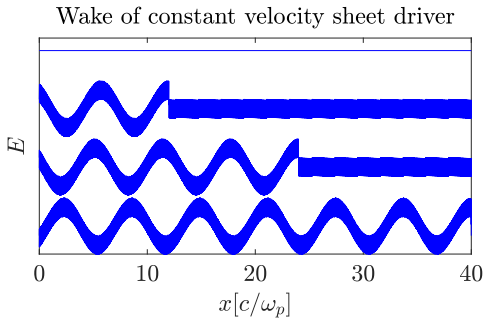

FIG. 6: Electric field in the wake of the driver at different times. The y axis is not shown to suggest the time evolution of the wake. The fast and small oscillations of the field artificially increase the thickness of the curves here presented.

C. Wavelength

Next we obtain the wavelength of the wake as a function of the phase velocity of the driver sheet.

To obtain this parameter of the wake, we chose a simulation duration so that the driver sheet would be at the edge of the box. This means that, except for the driver, the phase space of the system will have a structure $v = \sin(k_p x)$ where $k_p = 2\pi/\lambda_p$. Theoretically we expect $\lambda_p = 2\pi v_\phi$. To obtain the wavelength numerically, we fitted the phase space to a sin function. For each phase velocity we plot the corresponding wavelength in figure 7. The results are according to theory.

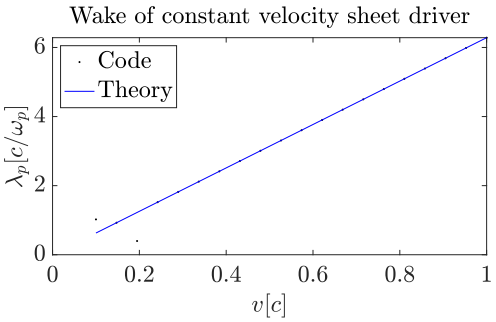

FIG. 7: Wavelength vs theory.

VI. COLD WAVEBREAKING

Wavebreaking means a large portion of the particles in the plasma get accelerated by the wave. As this is a qualitative description, a quantitative way to measure wavebreaking becomes necessary. Our solution depends on the number of collisions in the system.

At each time step we observe the number of collisions and for each phase velocity we calculate its average.

We observed that when there is a transition to wavebreaking, the number of collisions changes abruptly, as can be seen in figure 8.

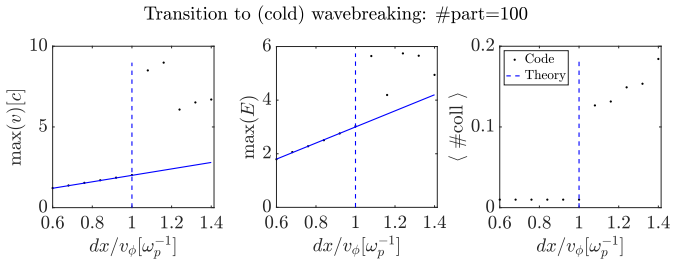

FIG. 8: Cold wavebreaking. Left: maximum velocity. Center: maximum field amplitude. Right: average number of collisions. The dashed line represents the expected transition to wavebreaking.

We found that $\max(v) = \delta$ and $\max(E) = 1.5\delta$. While the first scaling law is simple to explain, we found no reasonable explanation for the proportionality constant in the second. In any case, we then conclude that the electric field at wavebreaking is $1.5 v_\phi$. We confirmed that the phase spaces of electric field beyond this value present a much more chaotic structure than outside the wavebreaking regime.

This method could then be used to track the parameters (phase velocity, thermal velocity) that induce wavebreaking. However, lack of time kept us from comparing the theoretical expression $E_{WB} = \left(1 - \frac{8}{3}\beta^{1/4} + 2\beta^{1/2} - \frac{\beta}{3}\right)^{1/2}$ with simulations.

VII. CONCLUSIONS

With the electrostatic plasma sheet model developed by Dawson, many of the phenomena present in more complex approaches such as the Vlasov equation can be recovered. As a consequence, the computational implementation of the method emulates the same physics but at a lower cost.

[1] J. Dawson, The Physics of Fluids **5**, 445 (1962), https://aip.scitation.org/doi/pdf/10.1063/1.1706638, URL https://aip.scitation.org/doi/abs/10.1063/1.1706638.

[2] J. Dawson, Academic Press Inc. (1970).

[3] S. Wilks, T. Katsouleas, J. M. Dawson, P. Chen, and J. J. Su, IEEE Transactions on Plasma Science **15**, 210 (1987).