Electrostatic plasma simulation by Particle-In-Cell method using ANACONDA package

J S Blandón, J P Grisales and H Riascos
Plasma, Laser and Applications Group, Universidad Tecnologica de Pereira, Pereira, Colombia
E-mail: jsblandon@utp.edu.co jpgrisales@utp.edu.co hriascos@utp.edu.co

Abstract. Electrostatic plasma is the most representative and basic case in plasma physics field. One of its main characteristics is its ideal behavior, since it is assumed be in thermal equilibrium state. Through this assumption, it is possible to study various complex phenomena such as plasma oscillations, waves, instabilities or damping. Likewise, computational simulation of this specific plasma is the first step to analyze physics mechanisms on plasmas, which are not at equilibrium state, and hence plasma is not ideal. Particle-In-Cell (PIC) method is widely used because of its precision for this kind of cases. This work, presents PIC method implementation to simulate electrostatic plasma by Python, using ANACONDA packages. The code has been corroborated comparing previous theoretical results for three specific phenomena in cold plasmas: oscillations, Two-Stream instability (TSI) and Landau Damping (LD). Finally, parameters and results are discussed.

1. Introduction
Plasma physics is currently an important discipline since 99% of matter in the universe is plasma [1]. In fact; nuclear fusion, telecommunications, nanotechnology, laser ablation, and others, requires accurate theoretical and experimental developments to explain physical mechanisms that are not fully understood [1]. In this paper, the plasma studied is electrostatic, it means, its behavior is ideal and representative since it allow to explain phenomena such as plasma oscillations, instabilities and damping.

Plasma electrostatic simulation has an important role in the development of plasma physics since is a useful tool to validate theoretical models, which being put into practice, would demand expensive or even not developed technologies. Birdsall and Langdon put the basis for the plasma simulation in their book “Plasma physics via computer simulation” [2]; which provides the necessary background for construction of codes that can represent oscillations, waves, instabilities and Landau damping within plasma. Morse and Nielson in [3] simulate TSI for 1-D, 2-D and 3-D dimensions. However, computational tools have evolved allowing more accurate simulations and efficient use of computational resources like the code develop by Martin n[4] who simulated in C, plasma oscillations, two stream and four stream instabilities in an electrostatic cold plasma.

The present work is a simulation of electrostatic plasma made in python using ANACONDA libraries and is an implementation of the PIC algorithm, a common technique to simulate the motion of charged particles, which consist in the creation of a fixed mesh to calculate electromagnetic fields and a variable mesh to compute velocities and positions, which are updated every step for each particle. The Code was corroborated through analysis of characteristic
phenomena of electrostatic cold plasma such as oscillations, TSI and LD. This was achieved by means of phase space graphics to observe the behavior of such phenomena over time, and dispersion relation (DR) graphic to compare simulated and theoretical data.

2. Theoretical Fundation

Plasma is electrostatic when the magnetic field \( \vec{B} \) is equal to zero, and it is cold when particles temperatures is near to zero \( (T_{i,e} = 0) \) \[6\]. Therefore, the equations used in this model are Gauss law and Lorentz Force:

\[
\nabla \cdot \vec{E} = \frac{\rho}{\epsilon_0} \quad (1)
\]

\[
\frac{d\vec{v}}{dt} = \frac{q\vec{E}}{m} \quad (2)
\]

In equation (1), \( \vec{E} \) corresponds to the electric field, \( \rho \) to the charge density, and \( \epsilon_0 \) to electric permittivity in the vacuum. For equation (2), \( \vec{v} \), \( q \) and \( m \) are the charge, mass, and velocity respectively.

For cold plasma, initial velocities are zero \( (v = 0) \), but for TSI them corresponds to a Maxwell distribution:

\[
f(x,v) = n_0 \left( \frac{1}{\sqrt{2\pi v_{th}^2}} e^{-(v-v_{bh})^2/(2v_{th}^2)} + \frac{1}{\sqrt{2\pi v_{th}^2}} e^{-(v+v_{bh})^2/(2v_{th}^2)} \right) \quad (3)
\]

Where \( v \) is particle velocity, \( vh \) beam’s average velocity, \( v_{th} \) is the thermal velocity, and \( n_0 \) is the particles reference density.

LD effect was also consider. This occurs when there is an energy exchange between a wave and particles in plasma with their velocity approximately equal to phase velocity of the wave.

3. Implementation and simulation: PE1D.py

Implementation of an electrostatic model was made through a python code named Plasma Electrostático 1D (PE1D.py) using ANACONDA packages. It requires a discretization and a normalization of involved equations in order to be included in a computational loop called PIC. Also, an initial setup is needed in order to start the simulation. After its run, PE1D.py shows graphics which allows understanding how the computational system works. Construction of PE1D.py is based on codes: espic.py by Gibbon \[7\], 1D PIC code by Fitzpatrick \[8\] and Electrostatic 1D Particle In Cell in Matlab/Octave of Markidis \[9\].

3.1. PIC Algorithm

PIC algorithm is usually used to simulate particle dynamics. This loop requires a bunch of conditions that the coder must include. Figure 1 shows a diagram of the method.

For electrostatic case, PIC algorithm comprises the following steps:

(i) Movement equation solutions for each particle and actualization of Lagrangian mesh.
(ii) Calculus of plasma charge density through assignation of particles to a Eulerian mesh, across particles were moving.
(iii) Computation of the electric field \( (E_x) \) through the solution of equation 1.
(iv) Electric field \( (E_x) \) is passed to particles making them move. Then return to step 1.

This algorithm, as was said before, it is a loop which requires a previous background such as an initial setup, a discretization and normalization of model equations.
3.2. Initial Setup

Initial setup consist of creation of superparticles that emulates plasma as well as space trough particles move. Therefore, a mobile mesh anchored to particles (Lagrangian) and a fixed mesh (Eulerian) are created [11]. To ensure the number of particles remains constant, periodic boundary conditions were added to the code, as made by Gibbon[7]. To finish initial setup, a first iteration needed by PIC its made.

Code PE1D.py was setting considering different phenomena. For cold plasma oscillations, system was built using 2048 particles, 256 grid, a 4π grid length points, 150 time steps, a \( dt = 0.1 \), with a sinusoidal spacial perturbation of 0.001 of amplitude as was proposed by Birdsall and Langdon [2]. Instead, for TSI time variables still, but 20000 particles, 1024 grid points, 32π grid length, 2.5 average beam velocity, and a sinusoidal velocity perturbation were used. Finally, for LD system setup remains the same, but with 1200 time steps and without perturbation.

3.3. Discretization

For electrostatic model implementation using PIC, equations 1 and 2 must be discretized:

\[
x_{i}^{n+1} = x_{i}^{n} + v_{i}^{n+\frac{1}{2}} \Delta t
\]

\[
v_{i}^{n+\frac{1}{2}} = v_{i}^{n-\frac{1}{2}} + \frac{q_{i}}{m_{i}} E_{i}^{n} \Delta t
\]

Equations 4 and 5 correspond to the Leapfrog Scheme [12]. Density is written in equation 6 using the Cloud In Cell (CIC) weighting scheme describe in [4].

\[
\rho_{j}^{n+1} = \sum_{i} q_{i} \left( 1 - \frac{|x_{i} - x_{j}|}{\Delta x} \right)
\]

The electric field is written using the finite differences method, specifically using central difference form which allows a better approximation of equation 2.

\[
E_{j+\frac{1}{2}}^{n+1} = E_{j-\frac{1}{2}}^{n} + \rho_{j}^{n+1} \Delta x
\]

To assign electric field to particles, it is convenient to use the same weighing scheme as the used in density computation[4].

3.4. Normalization

Normalized units are important in computational physics because saves computational resources since quantities involved do not have many decimals [13]. Also, normalization of physical quantities allows use of dimensionless equations, enabling code transparency [7].
Simulated quantities for PE1D.py are as follows:

\[ \omega_p t \rightarrow t \quad \omega_p x/c \rightarrow x \quad v/c \rightarrow v \]  

\[ eE/m\omega_p c \rightarrow E \quad n_{e,i}/n_0 \rightarrow n_{e,i} \]  

where \( t \) is time, \( \omega_p \) is plasma frequency, \( x \) is position, \( v \) is velocity, \( c \) is light velocity, \( E \) is electric field, \( m \) is mass, and \( n \) is particle density.

According to 8 and 9, the following quantities are considered as unitary: \( q_e, m_e, n_0 \).

3.5. Simulation

PE1D.py satisfies electrostatic model because PIC algorithm ensures that particles responds to the field which is created by themselves, this property is known as collective behavior and it is characteristic of every plasma. Here we considered electric field only.

The parameters of simulation are selected according to conditions proposed by Forslund [14]:

\[ \omega_p \delta t < 2 \]  

\[ \lambda_d \geq \frac{\delta x}{2} \]  

\( \omega_p \) was normalized, and \( \lambda_d \) (Debye length) represents the length of a cell in the mesh. They are related according with the relation deduced by Lindman [15]:

\[ \delta t \leq \frac{2\pi}{\omega_p + \frac{\pi v_{th}}{\delta x}} \]  

Again \( v_{th} = 1 \) for PE1D case. Relation 12 is important because ensure reduction of computational noise of the system which drives to decrease of non physical effects. Moreover, electric field is calculated directly from Gauss equation without computation of electric potential, so this simplifies the model and avoids adding computational noise from an extra step.

Once system is perturbed, the amplitude of it must be small respect to the equilibrium quantities. In TSI case, means perturbed velocity is lesser than average beam velocity (\( v_b \)) [1].

For TSI, theoretical DR where built according to code by Markidis [16]. On the other hand, for simulated DR the two dimensional Fast Fourier Transform algorithm (FFT) was used in spatial and temporal components of electric field [10], [17]. In addition, electric field has a magnitude of number of cells, so this number was conveniently chosen as a power of two, having into account the nature of the FFT.

To visualize PE1D.py conservative behavior it were plot energy in figure 3, considering potential, kinetic, and total energy.

4. Results and Discussion

4.1. Cold plasma oscillations

Cold plasma oscillations were simulated. Figure 2 shows electric field in the Eulerian Mesh. It corresponds to time step 150, and displays an oscillatory behavior as predicted by the cold plasma oscillations theory. Sinusoidal particle distribution is shown in density and hence in electric field generated by the particles, thus collective behavior of plasma is remarked.

Birdsall and Langdon [2] explains that under a sinusoidal perturbation, the system behaves like an harmonic oscillator, showing an exchange between kinetic and potential energy. Figure
Figure 2. Electric field for cold plasma oscillations.

Figure 3. Energies as a function of time for cold plasma oscillations.

Figure 4. Dispersion relation for cold plasma oscillation: theoretical and simulated. The black line corresponds to theoretical DR. Different colors mean different particles densities which oscillates in such frequency. A higher density is symbolized through warm colors such as red, orange and yellow. Instead, lower density is represented by colors such as blue and green. It is clear from the image that most of particles oscillates at $\omega = 0.6$, i.e. near to plasma frequency as expected by theory [1].

3 shows energy system vs time, shows an oscillatory behavior according with the theory predictions.

Figure 4 is the DR and displays the temporal and spatial frequencies one against another for cold plasma oscillations.

4.2. Two-Stream instability

For TSI considerations were take into account: first, the temporal evolution in the phase space to corroborate if it presents its characteristic vorticity [1], and second respective dispersion relation to check the correct simulation of the field. Three test were made with 150 time steps and amplitudes of velocity perturbation of 0, 0.1 and 0.5 for figures 5, 6, 7 respectively.

Figures 5, 6 and 7 are phase spaces plots for TSI with different amplitude perturbation. Those results corresponds to two identical electron flows in a one dimensional plasma. Such case was studied by Fitzpatrick [8]. Results obtained by PE1D.py are according with [8]. Likewise, it is noted a characteristic vorticity of this instability as shows figures 5(b), 6(b) and 7(b) [3]. In figure 5(a), instability appears without external perturbation, it means that it appears only because of particle movement. Clearly is observe that particles suffer a heating because movement as shown by Martin [4]. On the other hand, figures 6(a) and 7(a) instability shows different states of development, which suggest a relation between amplitude perturbation and instability emergence.
Figure 5. Phase space for TS instability in cold plasma: no perturbed.

Figure 6. Phase space for TS instability in cold plasma: 0.1 amplitude of perturbation.

Figure 7. Phase space for TS instability in cold plasma: 0.5 amplitude of perturbation.
Figure 8. Dispersion relation for TS instability in cold plasma: theoretical and simulated without perturbation.

Figure 9. Dispersion relation for TS instability in cold plasma: theoretical and simulated with 0.1 amplitude of perturbation.

Figure 10. Dispersion relation for TS instability in cold plasma: theoretical and simulated with 0.5 amplitude of perturbation.

About DR in figures 8, 9, and 10, the black line on top is plasma frequency and the black curve is theoretical DR [2]. Figure 8 shows that particles oscillate near the first wavelengths in theoretical DR, while in figure 6 oscillations are better distributed along it, possibly as an effect of perturbation. In figure 10 with a perturbation near to average beam velocity, particles oscillates near to larger wavelengths in theoretical DR.

4.3. Landau damping
PE1D.py allows to simulate LD. Results obtained appears after a TSI simulation. As was mentioned in section 3.2, system has the same setup as TSI with time step at 1200 as shown in 11(a). There is a evidence of a particle trapping as Birdsall and Langdon exposed [2]. Figure 11(b) is the $E$ field energy against time. The results obtained in this work are in agreement with [2]. Thus, previous results confirm the validity of electrostatic cold plasma simulation.

5. Conclusions
According to the results, it is conclude that perturbation amplitude accelerates emergence of TS instability. Moreover, perturbation causes that particles oscillating underneath plasma frequency tend to settle according to theoretical DR. Finally, results from different simulated phenomena such as cold plasma oscillations, TS, and LD, had corroborated the validity of PE1D.py as electrostatic unidimensional plasma computational representation.
Figure 11. Phase space (a) with t=1200 and $\vec{E}$ Field Energy (b) showing LD for TS instability.

Acknowledgments
We would like to thanks to Henry Riascos Landázuri, director of the Plasma, Laser and Applications Group (GPLA) and Universidad Tecnológica de Pereira (UTP), for his support to develop this work. Also, we acknowledge the resources given by SENA and Colciencias to do this simulation project.

References
[1] Chen Francis F 1974 Introduction to plasma physics (Springer, New York)
[2] Birdsall C K and Langdon A B 1984 Plasma physics via computer simulation (McGraw-Hill)
[3] Morse R L and Nielson C 1969 Physical Review Letters 23 1087
[4] Martin D 2007 Electrostatic PIC simulation of plasmas in one dimension (Citeseer)
[5] Analytics C 2016 Anaconda software distribution URL https://continuum.io
[6] Guasp J 1976 Elementos de teoría cinética de plasmas (MALLORCA. CONSEJO INSULAR)
[7] Gibbon P 2014 ESPIC.py (CERN)
[8] Fitzpatrick R 2006 Computational physics: An introductory course (University of Texas)
[9] Markidis S 2016 pic.m (PDC Center For High Performance Computing (KTH))
[10] Markidis S and Lapenta G 2011 Journal of Computational Physics 230 7037–7052
[11] Woolfson M M and Pert G J 1999 An introduction to computer simulation (Oxford University Press.)
[12] McMillan S 2013 Computational Physics
[13] Lapenta G 2006 Centrum voor Plasma Astrofysica Katholieke Universiteit Leuven
[14] Forslund D W 1985 Space Science Reviews 42 3–16
[15] Lindman E 1970 Journal of Computational Physics 5 13–22
[16] Markidis S 2014 dispersion.m (PDC Center For High Performance Computing (KTH))
[17] Matsumoto H and Omura Y 1985 Computer simulations of space plasmas