Análisis de oscilaciones e inestabilidades de un plasma frío por medio del método Particle-In-Cell

Marcos Antonio Gutierrez Suárez

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ciencias y Humanidades

Trabajo de graduación en modalidad de tesis presentado por
Marcos Antonio Gutierrez Suárez
para optar al grado académico de Licenciado en Física

Guatemala
2020

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ciencias y Humanidades

Trabajo de graduación en modalidad de tesis presentado por
Marcos Antonio Gutierrez Suárez
para optar al grado académico de Licenciado en Física

Guatemala
2020

Vo.Bo.:

(f) _____

MSc. Diego Porres

Tribunal Examinador:

(f) _____

MSc. Diego Porres

(f) _____

MSc. Zaida Urrutia

(f) _____

Ing. Silvio Urizar

Fecha de aprobación: Guatemala, 27 de junio del 2020.

Agradecimientos

Es necesario agradecer a mis padres: Marco y Olga, porque han sido los que me han brindado las herramientas necesarias para los pequeños pasos de la vida. Sin embargo, yo creo que la dedicación de mi tesis (y la dedicatoria que mis padres también deben dar) va para mis abuelos: Marco, Marta, Vicenta y Jerónimo. No solo por haber hecho de mis padres lo que son, también por haber dado ese salto de la vida que nos ha permitido estudiar a todos, por haber hecho la verdadera diferencia. Para finalizar, se la dedico a mis hermanos menores: Marcelo, Esteban y Camila, que somos los que tenemos el peso de los nombres de nuestros abuelos.

Índice

Lista de figuras

Resumen

En este trabajo se construyó el método Particle-in-cell para simulación de un plasma frío electroestático. Esto se logró con el lenguaje `Python` en su versión 3.8.2. Se obtuvo resultados satisfactorios a la hora de estudiar los cuatro fenómenos propuestos: oscilaciones de plasma frío, las inestabilidades Two-stream, Beam-plasma y el amortiguamiento de Landau. El programa construido mostró ser congruente con los resultados obtenidos por otros investigadores. Además, en este trabajo se logró obtener las gráficas de las energías, potencial y cinética, del amortiguamiento de Landau, algo que no se encontró en la bibliografía consultada, con la excepción de la gráfica de energía potencial encontrada en el libro de Birdsall y Langdon[2] que no se muestra tan amigable para el lector.

CAPÍTULO 1

Introducción

El plasma constituye la mayor parte de la materia observable. Fenómenos complejos como las estrellas, la formación de galaxias, las tormentas solares y el comportamiento de la ionosfera se comprenden y modelan a través de la física del plasma[7].

Debido a la relevancia de la física del plasma en distintas aplicaciones de la física, este tipo de trabajos adquieren su valor. Por ejemplo: la física del plasma y sus métodos de simulación están presentes en los problemas de fusión, como es el eje central del libro de Chen[5] y también Donald Lynden-Bell notó que en la dinámica galáctica sucede un fenómeno parecido al amortiguamiento de Landau[16]. Intentando apoyarse sobre todo este respaldo, se trabajó esta tesis pensando en cómo la física computacional, tan avanzada y fundamental en esta era, puede aportar a un campo tan versátil.

Durante las lecturas para entender la física detrás y la construcción matemática de los algoritmos que permiten, hasta cierto punto, entender el comportamiento del plasma, se puede notar que los intentos de simular plasma por computadora han estado sucediendo desde que la teoría fue tomando importancia. Por ejemplo, en el libro de Birdsall y Langdon[2], se exploran todas las consideraciones computacionales para hacer este tipo de simulaciones (consideraciones aún vigentes). Sin embargo, este libro, cuando fue escrito, se programo en el lenguaje `Fortran`[6] y sus simulaciones no trabajan con más que unos pocos miles de partículas.

Existen otros trabajos como el de Martin[17] o Gibbon[18]. En el primero, se exploran las oscilaciones de plasma frío y la inestabilidad Two-stream. Esto se programó en `C++`[20], pero lo hizo con pocas partículas y no exploró uno de los resultados más importantes: el amortiguamiento de Landau. El segundo se programó en `Python`[9] y es más reciente que los otros dos, pero solo abordó las oscilaciones de plasma frío e incluyó un pequeño factor de ajuste térmico. En cambio, en el aspecto teórico de los temas tratados hay una amplia y completa bibliografía. Por lo tanto, pareció pertinente efectuar trabajos computacionales de estos fenómenos del plasma.

En cuanto a la simulación hay dos forma de abordarla: a través de simulación de partículas o trabajando el plasma como un fluido cargado. En este trabajo se utilizó el método Particle-In-Cell, que es un método que trabaja con partículas. Se escogió este método por ser el más directo y por el ser el que permite observar comportamientos e interacciones entre partículas que son omitidos por lo métodos Magnetohidrodinámicos, que son los que conciben al plasma cómo un fluido. Ambos caminos son complementarios, pero tienen herramientas y teorías distintas. La comparación entre métodos y sus profundidades se pueden encontrar en libros como él de Langdon y Birdsall[2] y el de

Hockney [19].

En este trabajo se logró construir un código en `Python` 3.8.2, haciéndolo versátil para explorar las inestabilidad Two-stream, Beam-plasma y el amortiguamiento de Landau bajo distintas condiciones. Se logró comprobar las congruencias de los resultados obtenidos por los distintos autores en la bibliografía citada. Además, el trabajo en sí es un texto funcional para brindar una teoría básica del plasma y las herramientas para hacer simulaciones.

Uno de los análisis hechos en este trabajo es el de lograr observar el comportamiento de las energías, tanto cinética como del campo eléctrico, de los fenómenos en cuestión. Como se ha mencionado, este análisis no se hizo en la bibliografía citada o bien no se presentaron resultados tan notorios como los que realizaremos aquí. Una aclaración que es importante hacer, por el bien de la congruencia con la teoría expuesta, es que las distribuciones utilizadas para las condiciones iniciales de las velocidades son Maxwelianas. Esto se hizo porque es una de las distribuciones que cumplen con las condiciones expuestas en la sección de la teoría del amortiguamiento de Landau (4.87). A pesar de que esas distribuciones podrían ser otras, se hizo esto para que la simulación se mantuviera acorde a la teoría expuesta. Esta distribución se expondrá a lo largo del marco teórico.

CAPÍTULO 2

Justificación

Los estudios de plasma frío, a pesar de no ser un evento posible como tal, son algo que permite simular con mucha mayor facilidad y por lo tanto estudiar el comportamiento de plasmas en equilibrio térmico[2] y de plasmas sin colisiones[8]. Además, el método Particle-in-cell tiene una gran gama de aplicaciones en la física del plasma: esto implica evolución estelar[3], estudios de la ionósfera[7], estado sólido[8], mecánica de fluidos[5], entre otros. Por lo tanto, entender las simulaciones de plasma frío por el método Particle-in-cell brinda muchas herramientas y nuevas líneas de investigación. Sentar las bases en un trabajo como este puede construir las bases para todo un campo en el estudio de fenómenos físicos por computadora, siendo este método, para Guatemala, mucho más viable que los métodos experimentales.

CAPÍTULO 3

Objetivos

3.1. Objetivo general

Construir un código funcional que permita estudiar distintos fenómenos de un plasma frío y a su vez motivar a otras generaciones de estudiantes explorar la física del plasma y más métodos computacionales.También se busca que le método tenga congruencia con la teoría expuesta en el marco teórico y con los antecedentes que hacen referencia a las simulaciones.

3.2. Objetivos específicos

- Programar el código en el lenguaje `Python`, debido a su amplitud en la comunidad científica y versatilidad.
- Analizar los fenómenos principales del plasma desde las facetas presentadas en la mayor parte de la bibliografía: densidad de carga, campo eléctrico y diagrama de fase. A la vez, explorar nuevas formas de análisis.
- Implementar el método de simulación por partículas Particle-in-cell e implementar métodos de perturbación del sistema para explorar sistemas inestables.

CAPÍTULO 4

Marco teórico

4.1. El concepto de temperatura

Antes que nada, es necesario retomar la importancia de la noción de temperatura. Si bien en este trabajo se realizaron simulaciones de plasma frío, siendo este un método de simplificación de los problemas del plasma. Sin embargo, las simulaciones de plasma frío son funcionales porque sus condiciones se asemejan mucho al de un plasma en ***equilibrio térmico***, que es cuando tanto los electrones como los iones están a una misma temperatura. Repasar las nociones de temperatura nos permite, entonces, interpretar mejor los resultados. Además, en las inestabilidades estudiadas en este trabajo, un plasma frío pasa a ser un plasma caliente debido a las interacciones entre las partículas. Esto se explicará en la sección de Discusión y Resultados (Sección 6.2).

El concepto de temperatura se explicará para el caso unidimensional, ya que es el caso en el que se trabajaron las simulaciones. La ampliación a tres dimensiones es sencilla si se parte del caso unidimensional. Esta puede ser consultada en los libros de Bellan[3] y Chen[5].

Empecemos estudiando el fenómeno como un gas. Un ***gas en equilibrio térmico*** es aquel que tiene partículas con distintas velocidades. La distribución más probable de estas partículas es una distribución Maxweliana:

$$f(u) = A \exp\left(-\frac{mu^2}{2KT}\right) \tag{4.1}$$

donde m y u son la masa y velocidad de cada partícula, respectivamente, y K es la constante de Boltzmann:

$$K = 1.380\,649 \times 10^{-23}\,\mathrm{J/K} \tag{4.2}$$

Nótese que el ancho de la distribución está caracterizado por T, algo que llamamos ***temperatura***. Para entender qué es la temperatura, primero estudiemos más a fondo la distribución 4.1. La ***densidad*** está dado por:

$$n = \int_{-\infty}^{\infty} f(u)du \tag{4.3}$$

La constante de normalización A para la distribución de Maxweliana puede ser derivada de la ecuación(4.3) igualándola a 1, para normalizarla. El resultado es:

$$A = \left(\frac{m}{2\pi KT}\right)^{1/2} \tag{4.4}$$

La ***energía cinética promedio*** de las partículas del gas estará dada por:

$$E_{prom} = \frac{\int_{-\infty}^{\infty} \frac{1}{2}mu^2 f(u)du}{\int_{-\infty}^{\infty} f(u)du} \tag{4.5}$$

Definiendo a la ***velocidad térmica*** como $v_t = (2KT/m)^{\frac{1}{2}}$ y realizando la sustitución $y = u/v_t$ en la ecuación (4.1), tenemos:

$$f(u) = A\exp\left(-u^2/v_t^2\right) \tag{4.6}$$

Por lo tanto, la ecuación (4.5) se vuelve:

$$E_{prom} = \frac{\frac{1}{2}Av_t^2 \int_{-\infty}^{\infty} \exp\left(y^2\right)y^2 dy}{Av_t \int_{-\infty}^{\infty} \exp\left(y^2\right)dy} \tag{4.7}$$

Resolviendo el numerador de (4.7) por partes, se elimina un término con el denominador. Luego, resolviendo, se obtiene que:

$$E_{prom} = \frac{\frac{1}{2}mAv_t^3\frac{1}{2}}{Av_t} = \frac{1}{4}mv_t^2 = \frac{1}{2}KT \tag{4.8}$$

Como se puede ver, en este resultado, la energía y la temperatura tienen una relación muy estrecha. Por eso, en la mayoría de ramas de la física-desde mecánica cuántica[7], física del plasma[5], hasta nuclear[5]- estas cantidades se dan en una cantidad denominada ***electrón-volt*** (eV).

4.2. Parámetros fundamentales

Se le llama ***plasma ideal*** a aquel que tiene el mismo número de electrones e iones cuyas masas son, respectivamente, m_e y m_i. Los electrones tienen carga $-e$ y los iones $+e$, donde e es la carga fundamental 1.6×10^{-19} C[8].

Se denota a la ***temperatura cinética*** T_s como:

$$T_s = \frac{1}{3}m_s \langle V_s \rangle^2 \tag{4.9}$$

En esta notación, los subíndices s se refieren a las especies del plasma. Las ***especies*** son el tipo de partícula que está dentro el plasma. En el caso del plasma, las especies pueden ser iones o electrones y T_s es la energía cinética de estas.

Un plasma, a un nivel macroscópico, se debe ver y comportar como un fluido con la diferencia que el primero interactúa con campos electromagnéticos. La ***propiedad de cuasineutralidad*** demanda que, en ausencia de estos campos, la concentración de cargas positivas y negativas sea aproximadamente la misma en pequeñas regiones del espacio. Esta propiedad es importante porque es un diferenciador en la teoría del plasma y podemos expresarla de la siguiente manera:

$$n_i \simeq n_e \equiv n \tag{4.10}$$

Si se asume que los iones y electrones tiene la misma temperatura. Esto no sucedería, por ejemplo, en casos donde alguna de las dos especies se vea acelerada. Como se podrá ver en la Sección 6.2, donde los haces son únicamente de electrones sobre un fondo de iones. Esto significa que son únicos que se modelan en la interacción con el campo electroestático son los electrones mientras que los iones se mantienen fijos[8]. La velocidad térmica se expresa como:

$$Vt_s = \left(\frac{2T}{m_s}\right)^{\frac{1}{2}} \tag{4.11}$$

Operando las ecuaciones (4.10) y (4.11) podemos encontrar una relación entre las velocidades térmicas de las distintas especies:

$$Vt_i \sim \left(\frac{m_e}{m_i}\right)^{\frac{1}{2}} Vt_e$$

4.3. Frecuencia del plasma

La ***frecuencia del plasma*** es la escala temporal más fundamental del plasma y se define como[8]:

$$\omega_p = \left(\frac{ne^2}{\epsilon_0 m}\right)^{\frac{1}{2}} \tag{4.12}$$

Esta ecuación es para plasmas «fríos» ya que se omite el factor térmico. Debido a que cada especie tiene una distinta carga y masa, habrá una frecuencia distinta para cada especie.

Considérese el escenario donde se tiene planos normales respecto al eje x. Si se desplaza una partícula de su estado cuasi-neutral por una distancia δx, se genera una densidad superficial de carga sobre los planos de la forma: $\sigma = en\delta x$. Esta densidad superficial de carga variará dependiendo de la carga de la especie. En este caso, habrá una con signo positivo y otra con signo negativo en direcciones opuestas.

Aplicando la Segunda Ley de Newton a lo largo de δx se obtiene lo siguiente:

$$m\frac{d^2(\delta x)}{dt^2} = eE_x \tag{4.13}$$

Si se revisa la ecuación (10.1) se podrá ver que, en el caso que presentamos, hace falta B. Esto debe a que estamos trabajando el caso electroestático.

Sustituyendo en (4.13) $E = \sigma/\epsilon_0 = -en\delta x/\epsilon_0$ y (4.12), obtenemos que:

$$m\frac{d^2(\delta x)}{dt^2} = -m\omega_p^2\delta x. \tag{4.14}$$

donde la perturbación δx esta es:

$$\delta x = \delta x_0 \cos\omega_p t \tag{4.15}$$

Ese resultado se obtiene trabajando la ecuación diferencial (4.13) con el método del polinomio característico[24] y con la condición inicial $\delta x(0) = \delta x_0$.

Las oscilaciones en el plasma solo pueden ser observables si se hacen sobre periodos τ mayores al periodo del plasma: $\tau_p = 2\pi/\omega_p$[8].

De la misma manera, la medición de una distancia L que sea menor a una distancia típica que recorre una partícula durante un periodo de plasma no será detectada. Esta distancia es $v_t\tau_p$ y se llama ***distancia de Debye***, en honor a Peter Debye y se define como:

$$\lambda_D = \frac{1}{\omega_p}\left(\frac{T}{m}\right)^{\frac{1}{2}} \tag{4.16}$$

Sustituyento en (4.16) la ecuación de la frecuencia del plasma (4.12) obtenemos:

$$\lambda_D = \left(\frac{\epsilon_0 T}{ne^2}\right)^{\frac{1}{2}} \tag{4.17}$$

Estos resultados nos llevan a un resultado importante: un sistema puede ser considerado plasma si cumple con las siguientes relaciones:

$$\frac{\lambda_D}{L} \ll 1, \qquad \frac{\tau_p}{\tau} \ll 1 \tag{4.18}$$

Aquí, L y τ son las escalas de tiempo y distancia que se usan en el proceso de investigación. El trabajo en escalas del plasma y los métodos que utiliza la física del plasma para hacerlo se salen del enfoque de este trabajo, pero en la sección 4.16 del libro de Fitzpatrick[8] se aborda el tema más a detalle, en los casos de plasmas confinados.

4.4. Escudo de Debye

En el anexo 10.3 se expone el ***apantallamiento eléctrico*** en el caso de una partícula. Este fenómeno es importante por tres razones: una es porque explica uno de los retos que tuvo la mecánica de fluidos a la hora de modelar el plasma[5], muestra el porqué de las condiciones para la clasificación del plasma y nos da el salto a esta sección: el fenómeno del apantallamiento explica la validez del escudo de Debye y porqué este se toma en consideración. Debido a que trabajamos con un plasma de electrones sobre un fondo de iones, la densidad fija de los segundos no se puede omitir debido, precisamente, a la existencia de un apantallamiento. Por lo tanto, en la función `electricfield()` que se explica en la sección de metodología y se puede visualizar en el anexo 10.4, se considera la densidad de carga aportada por los iones al sistema. Una forma bastante ilustrativa de ver el escudo

de Debye es pensar en un dieléctrico: la polarización del plasma y la distribución de las partículas cargadas evitan que un campo eléctrico externo penetre el sistema[8].

Si consideramos un plasma lo más cercano al equilibrio térmico, lo suficiente como para que sus dos especies tengan la distribución de Maxwell-Boltzmann:

$$n_s = n_0 \exp\left(\frac{-e_s\phi}{T}\right) \tag{4.19}$$

Donde ϕ es el potencial del plasma y n es la distribución de las partículas de los electrones en el caso electroestático[8], n_0 es la cantidad inicial de una especie s y T es la temperatura de equilibrio. En la ecuación se puede notar que, para satisfacer la Propiedad de cuasi-neutralidad, se necesita que el potencial electroestático sea 0.

Es necesario entender lo que pasaría si ocurriera una perturbación en el potencial electrostático. Supónganse que el potencial $\phi(\mathbf{r})$ es perturbado de la forma $\delta\phi(\mathbf{r})$, en este caso $\mathbf{r}$ es el vector posición de una partícula dentro del plasma. Como consecuencia de una pequeña y localizada densidad de carga $\delta\rho_{ext}$ que se inserta lentamente dentro del plasma, entonces tenemos que la densidad de carga total es después de esto se representa como:

$$\delta\rho = \delta\rho_{ext} + e\left(\delta n_i - \delta n_e\right) \approx \delta\rho_{ext} - 2e^2 n_0 \frac{\delta\phi}{T} \tag{4.20}$$

Esto se obtiene luego de substituir (4.19) y aproximar la suma de exponenciales por una serie de Taylor.

La ecuación de Poisson para la perturbación del potencial es:

$$\nabla^2\phi = -\frac{\delta\rho}{\epsilon_0} = -\left(\frac{\delta\rho_{ext} - 2e^2 n_0 \frac{\delta\phi}{T}}{\epsilon_0}\right) \tag{4.21}$$

Operando, simplificando y sustituyendo la Distancia de Debye (Ecuación (4.17)), se obtiene la ecuación final:

$$\left[\nabla^2 - \frac{2}{\lambda_D^2}\right]\delta\phi = \frac{\delta\rho_{ext}}{\epsilon_0} \tag{4.22}$$

Si la perturbación de la densidad se reduce a solo un punto, entonces:

$$\delta\rho_{ext} = q\delta(r) \qquad \delta\phi(r) = \frac{q}{4\pi\epsilon_0 r}\exp\left(\frac{-\sqrt{2}r}{\lambda_D}\right) \tag{4.23}$$

4.5. Parámetros del plasma

Se define a la ***distancia promedio entre partículas*** como:

$$r_d \approx n^{1/3} \tag{4.24}$$

Este resultado deriva de asumir al plasma como ideal. Para definir la distancia promedio entre partículas se han tomado dos formas de verlo: una es la visión de Wigner-Seitz[11], que lo trabaja como si se tratase de una esfera con densidad $1/n$. El otro enfoque, que es el que mostramos acá, es el resultado de observar el espacio como la longitud del borde de un cubo de densidad $1/n$. Estas aproximaciones se diferencian por un factor de 1.61[11]. Sin estas aproximaciones, encontrar la distancia promedio entre partículas de forma analítica es imposible.

La ***distancia media de aproximación*** [8] se define como la distancia promedio que dos partículas con la misma carga pueden acercarse entre sí. Esto lo escribimos como:

$$r_c = \frac{e^2}{4\pi\epsilon_0 T} \tag{4.25}$$

Balanceando la energía térmica(4.11) con el potencial electrostático repulsivo de un ***par binario*** (la interacción entre dos partículas de la misma carga) se tiene:

$$\frac{1}{2}mV_t^2 = \frac{e^2}{4m\epsilon_0 r_c} \tag{4.26}$$

Es interesante estudiar la proporción r_d/r_c. Si esta proporción es pequeña ($r_d/r_c \ll 1$) significa que las partículas cargadas son dominadas por una influencia electrostática. Esto implica que sus energías cinéticas son menores en comparación respecto a sus interacciones entre potenciales. Este tipo de plasmas se llaman ***fuertemente acoplados***.

Si se da el caso en que la proporción sea grande ($r_d/r_c \gg 1$), significa que las interacciones electrostáticas son raras y muy ocasionales. Estos plasmas son llamados ***débilmente acoplados***.

Todo lo anterior permite definir al ***Parámetro del plasma***:

$$\Lambda = \frac{4\pi}{3} n\lambda_D^3 \tag{4.27}$$

también llamado como ***esfera de Debye***. Combinando las ecuaciones (4.17), (4.25), (4.24) y (4.27) se obtiene:

$$\Lambda = \frac{\lambda_D}{3r_c} = \frac{1}{3\sqrt{4\pi}} \left(\frac{r_d}{r_c}\right)^{\frac{3}{2}} = \frac{4\pi\epsilon_0^{\frac{3}{2}} T^{\frac{3}{2}}}{3e^3 n^{\frac{1}{2}}} \tag{4.28}$$

El parámetro del plasma nos permite, entonces, definir a los plasmas fuertemente y débilmente acoplados: cuando $\Lambda \ll 1$ se tiene un plasma fuertemente acoplado y cuando $\Lambda \gg 1$ se tiene un plasma débilmente acoplado[8].

4.6. Colisiones

Las colisiones dentro de un plasma difieren de las colisiones de un gas, debido a las interacciones de Coulomb entre las partículas cargadas. Ahora, con los resultados de la sección anterior, podemos ver que en plasmas fuertemente acoplados (con una esfera de Debye poco poblada) las colisiones entre partículas no serán tan recurrentes. En cambio, en plasmas débilmente acoplados (con una esfera de Debye densamente poblada) las interacciones entre partículas ocurren con mucha frecuencia[8].

Partiendo de lo anterior para Λ grandes sí se puede hablar de colisiones binarias. Esto nos lleva a definir las ***frecuencias de colisión*** [8]:

$$\nu_s \simeq \sum_{s'} \nu_{ss'} \tag{4.29}$$

La ecuación (4.29) cuenta la cantidad de colisiones que tiene la especie s con las otras especies s'. En concreto, la frecuencia ν mide las trayectorias por las que pasa una partícula bajo un cambio significante del ángulo bajo las interacciones de Coulomb con otras partículas. Esta ν es el inverso del tiempo necesario para que ocurran suficientes colisiones para que ocurran suficientes colisiones y la partículas se desvíe 90°. Por eso, a veces se llama a ν como ***radio de Scattering de*** 90°.

La relativamente pequeña masa del electrón implica que, por unidad de carga iónica y la temperatura de las especies:

$$\nu_e \approx \left(\frac{m_i}{m_e}\right)^{\frac{1}{2}} \nu_i \tag{4.30}$$

Este resultado se verá más claro al final de de esta sección.

Se define la ***media de la trayectoria libre***, o mpf por sus siglas en inglés, como:

$$\lambda_{mpf} = \frac{V_t}{\nu} \tag{4.31}$$

Esta mide la distancia típica que una partícula recorre entre colisiones. Un plasma dominado por colisiones es aquel que cumple con $\lambda_{mpf} \ll L$ donde L está a escalas del observador. Si $\lambda_{mpf} \gg L$ entonces es un plasma sin colisiones.

La magnitud típica de la frecuencia de las colisiones es:

$$\nu \sim \frac{\ln \Lambda}{\Lambda} \omega_p \tag{4.32}$$

Notése que si $\nu \ll \omega_p$ es un plasma débilmente acoplado. Esto implica que las colisiones no interfieren con las oscilaciones del plasma. Por otro lado, cuando $\nu \gg \omega_p$ significa que es un plasma fuertemente acoplado, dominado por interacciones de Coulomb y no tiene cabida en las dinámicas convencionales del plasma.

Uniendo las ecuaciones anteriores se tiene que:

$$\frac{\lambda_{mpf}}{\lambda_D} \sim \frac{\Lambda}{\ln \Lambda} \tag{4.33}$$

donde si $\lambda_{mpf} \gg \lambda_D$ se tiene un plasma débilmente acoplado. Sustituyendo (4.17) y (4.28) en (4.33) se obtiene que la frecuencia de colisiones es :

$$\nu = \frac{3e^4 \ln \Lambda}{4\pi\epsilon_0 m^{\frac{1}{2}}} \frac{n}{T^{\frac{3}{2}}} \tag{4.34}$$

En esta ecuación podemos ver que, a mayor temperatura, el plasma tiende cada vez más a ser uno sin colisiones. En caso contrario, cuando el plasma es más frío, este tiende a tener más colisiones.
‘

4.7. Movimiento en plasmas uniformes

Considérese el caso de una partícula moviéndose en un campo electromagnético uniforme:

$$m\frac{dv}{dt} = e(E + v \times \mathbf{B}) \tag{4.35}$$

Las componentes paralelas al campo magnético tienen la siguiente forma:

$$m\frac{dv_{\|}}{dt} = eE_{\|} \tag{4.36}$$

A continuación se expone el desarrollo para comprender las componentes perpendiculares a **B**.

Como E y B son uniformes:

$$m\frac{d\langle v\rangle_t}{dt} = 0 = e(E + v \times \mathbf{B}) \Rightarrow -E = v \times B$$

Realizamos el producto cruz por la izquierda con **B** en ambos lados de la ecuación, por lo que:

$$\begin{aligned} E \times B &= B \times (v \times B) \\ &= v(B \cdot B) - B(v \cdot B) \end{aligned}$$

donde la segunda igualdad se da gracias a la identidad del triple producto vectorial. Como **v** y **B** son perpendiculares, entonces $(v \cdot B) = 0$:

$$\Rightarrow E \times B = vB^2 \Rightarrow v_d = \tfrac{E\times B}{B^2}$$

Como la velocidad total es $v = v_d + v_{osc}$ donde v_{osc} es la velocidad de oscilación y v_d es el arrastre cruzado de los campos. Para obtener esta expresión se hace lo siguiente:

$$m\tfrac{dv_{\perp}}{dt} = e(E + v_{osc\times B + v_d \times B})$$
$$\Rightarrow m\tfrac{dv_{osc}}{dt} = e(v_{osc} \times B) \Rightarrow m\tfrac{dv_{osc}}{dt} = ev_{osc}B$$

Esto nos lleva a la expresión de velocidad en planos uniformes de una partícula:

$$\Rightarrow v_{osc} = \frac{E \times B}{B^2} + \rho\Omega\left[\sin\left(\Omega + \gamma_0\right)e_1 + \cos\left(\Omega + \gamma_0\right)e_2\right] \tag{4.37}$$

donde Ω es la frecuencia del ciclotrón que se define como qB/m, ρ es el radio de Larmor que se define como $mv_{\perp}$r, γ_0 la girofase y donde e_1 y e_2 son vectores unitarios ortogonales a B.

De esta expresión, utlizando $r\left(t\right) = R\left(t\right) + \rho\left(t\right)$ se puede obtener la posición de una partícula:

$$\rho(t) = \left[-\cos\left(\Omega t\right)e_1 + \sin\left(\Omega t\right)e_2\right] \tag{4.38}$$

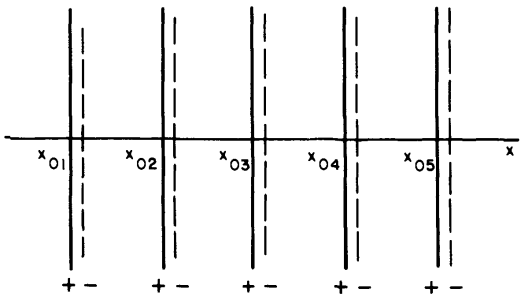


Figura 4.1: Distribuciones de iones y electrones, ignorando el espacio de las mallas, donde x_{0i} son las posiciones de equilibrio de cada grupo de iones y electrones. [2]

$$R\left(t\right)=\left(v_{0||}t+\frac{e}{m}E_{||}\frac{t^{2}}{2}\right)b+v_{d}t \tag{4.39}$$

Donde $v_d = E \times \mathbf{B}/B^2$, R es es el giro-centro que se mueve a través de la espiral y ρ es la componente oscilatoria del movimiento.

4.8. Plasma frío

Un ***plasma frío*** parece ser una contradicción, ya que la mayoría de plasmas tienen una energía térmica que asciende a 1eV ($\approx 10^4 K$)[2]. Pero, haciendo la simplificación $v_t \to 0$ nos permite utilizar condiciones iniciales más simples y explorar la naturaleza del plasma de una manera más sencilla.

Repasando algo expuesto en las secciones anteriores, sabemos que un plasma tiene un movimiento armónico simple alrededor de su punto de equilibrio-los x_0-. Esto se describe de la siguiente forma:

$$\delta x = x - x_0 \quad \text{y} \quad \delta\ddot{x} = -\omega_p^2 \delta x \tag{4.40}$$

La solución de este sistema es:

$$\delta x(t-t_0) = A(t_0)cos\omega_p(t-t_0) + B(t_0)sen\omega_p(t-t_0) \tag{4.41}$$

Este resultado es tan directo que generalmente no levanta muchas preguntas. Pero en el caso de esta tesis, que busca abordar aspectos de la simulación computacional, es necesario plantear algunas.

En primer lugar, no hay ningún problema con tener a los electrones y a los iones moviéndose, ya que estos se mueven a la misma frecuencia $\omega^2 = \omega_p^2 = \omega_e^2 + \omega_i^2$ [2], pero los iones con densidades y velocidades menores a las de los electrones por un factor de $\frac{m_e}{m_i}$ [2]. Si tratamos a los iones como estacionarios, entonces tenemos que $\frac{m_i}{m_e} \Rightarrow \infty$ para oscilaciones de alta frecuencia [2], por la proporción entre las velocidades.

Por el enfoque computacional de este trabajo, se debe decidir qué hacer con los iones. Supóngase que ignoramos el espacio y la distribución entre las mallas—como corresponde en el método en cuestión—y solo nos enfocamos en la distribución de iones y electrones en una dimensión; entonces tendríamos como en la Figura 4.1.

Este es un ***sistema neutro***. Si desplazamos los electrones más de la mitad de el espacio entre iones de la forma $\delta x < \frac{1}{2}(x_{02} - x_{01})$. Al hacer esto, el electrón se vería atraído por su iones correspondiente y este oscilaría alrededor del ion. Sin embargo, esta oscilación no sería de la forma de un movimiento armónico simple [2]. Este problema se soluciona con una distribución uniforme de

los iones en algo llamado ***fondo de iones***. Esto se logra a través del mallado, que es un proceso necesario en los métodos de simulación por partículas. En un sistema periódico, donde la densidad neta no es necesariamente cero en un intervalo, se puede recalibrar la densidad iónica para que que esto se cumpla. Además, se puede colocar un ion por malla, por ejemplo, para omitir el movimiento de los iones o hacer $m_i \gg m_e(\omega_i \ll \omega_e$. Entonces se tiene $\rho_j = \rho_{ion}$ donde el subíndice j representa a las mallas.

Con esto claro, ya se procede a mapear cada uno de los electrones sobre la malla. Estos inician en su posición de equilibrio. Se buscará que la frecuencia de la perturbación tenga la siguiente forma:

$$f_p = \frac{\omega_p}{2\pi} = \frac{q}{2\pi}\sqrt{\frac{N}{L\epsilon_0 m}} \tag{4.42}$$

Donde L es la longitud total de una malla, m la masa y N el número de partículas. Entonces, la oscilación del electrón alrededor de su posición de equilibrio será de una forma sinusoidal:

$$x = x_0 + x_1 cos\left(\frac{2\pi n}{L}x_0\right) \tag{4.43}$$

Donde x_0 es la posición de equilibrio, x_1 es la amplitud de la oscilación y n es un entero. La amplitud, generalmente es mucho más pequeña que la longitud de la malla, esto se debe a que, si esta es muy grande, el movimiento de las partículas puede traslaparse[17]. Estas oscilaciones tendrán un efecto en la densidad de carga—y consecuentemente en el campo eléctrico—que se comportarán de una manera similar a la oscilación de una cuerda atada en sus dos extremos[17]. Esto provocará oscilaciones en tanto la energía potencial como cinética y estas oscilaciones deben oscilar con la frecuencia expuesta anteriormente (4.42) acorde al o expuesto en [17].

4.9. Inestabilidad Two-Stream

La ***inestabilidad Two-stream*** para plasma frío consiste en dos flujos de electrones con velocidades medias iguales pero en sentidos contrarios. Estas velocidades son relativamente diferentes a 0 respecto a un fondo inerte de iones. Los haces de electrones reciben una perturbación sinusoidal. Para entender este fenómeno, reescribamos las ecuaciones de movimiento (4.35):

$$\frac{\partial v_i}{\partial t} = \frac{q}{M}E_1 \tag{4.44}$$

Donde E_1 es el campo eléctrico presente en nuestro sistema.

$$\left[\frac{\partial v_e}{\partial t} + (v_0 \cdot \nabla)\right] = \frac{q}{m}E_1 \tag{4.45}$$

En la ecuación (4.44) se omite el termino $(v_0 \cdot \nabla)$ porque se toma $v_{i0} = 0$. Además, el término mencionado caerá de la ecuación (4.45) porque se asume continuidad para v_0. En esta inestabilidad lo que se busca son ondas planas electroestáticas de la forma, por ser la forma en que se propagan los fenómenos electroestáticos[12]:

$$E_1 = Ee^{i(kx-wt)}\mathbf{x} \tag{4.46}$$

donde k es el número de onda y $\mathbf{x}$ es la dirección de $\mathbf{v}$ y $\mathbf{k}$. Sustituyendo (4.46) en (4.44) y en (4.45) obtenemos lo siguiente:

$$v_i = \frac{iq}{\omega M} E\mathbf{x} \tag{4.47}$$

$$v_e = -\frac{iq}{m}\frac{E}{w - kv_0}\mathbf{x} \tag{4.48}$$

Como las velocidades están sobre el eje x, podemos omitir el vector normal en la notación. La ecuación de continuidad de los iones del sistema es:

$$\frac{\partial \eta_i}{\partial t} + \eta_0 \nabla \cdot \mathbf{v_i} = 0 \tag{4.49}$$

donde

$$\eta_i = \frac{k}{\omega}\eta_0 v_i \approx \frac{ie\eta_0 k}{M\omega^2} E \tag{4.50}$$

Ahora, la ecuación de continuidad de los electrones es:

$$\frac{\partial \eta_e}{\partial t} + \eta_0 \nabla \cdot v_e + (v_0 \cdot \nabla)\eta_e = 0 \tag{4.51}$$

Esto se vuelve:

$$(-i\omega + ikv_0) + ik\eta_0 v_e = 0 \tag{4.52}$$

con la densidad de electrones siendo:

$$\eta_e = \frac{k\eta_0}{\omega - kv_0} v_e \approx \frac{iek\eta_0}{m(\omega - kv_0)} E \tag{4.53}$$

Como las inestabilidades son oscilaciones de plasma de alta frecuencia, estas no afectan el movimiento de los iones; por lo tanto, no se trabaja con una aproximación de plasma, pero sí con la la ecuación de Poisson [5].

$$\epsilon_0 \nabla \cdot E_1 = q(\eta_i - \eta_e) \tag{4.54}$$

Substituyendo (4.50) y (4.53) en (4.54) obtenemos:

$$ik\epsilon_0 E = q(iq\eta_0 kE)\left[\frac{1}{M\omega^2} + \frac{1}{m(\omega - kv_0)^2}\right] \tag{4.55}$$

La relación de dispersión se obtiene dividiendo por $ik\epsilon_0 E$ y usando a 4.12:

$$1 = \omega_p^2 \left[\frac{m/M}{\omega^2}\frac{1}{(\omega - kv_0)^2}\right] \tag{4.56}$$

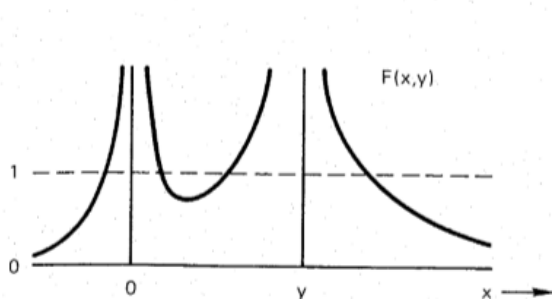


Figura 4.2: Se muestra a $F(x, y)$ en la inestabilidad Two-stream cuando el plasma es estable [5]

Ahora se analizan los posibles resultados para un k real. Si tomamos la ecuación (4.56) y la multiplicamos por el denominador común del lado derecho, tendríamos una ecuación de orden cuatro respecto a ω. Si todas las raíces de esta ecuación son reales, entonces obtendríamos que cada raíz representa una oscilación distinta de la forma (4.46).

En caso de que algunas de las raíces fuesen complejas, tendríamos lo siguiente:

$$\omega_j = \alpha_j + i\gamma_j \tag{4.57}$$

Donde α es la parte real γ la parte imaginaria.

Con esto, ya obtenemos la dependencia temporal:

$$E_1 = Ee^{i(kx-\alpha_j t)}e^{\gamma_j t} \tag{4.58}$$

Si se tiene un γ positivo, significa que se tiene una onda exponencialmente creciente. En caso de ser γ negativo se tiene una oscilación amortiguada.

Como las raíces ω_j vienen en pares conjugados, siempre habrá inestabilidades, a menos que todas las raíces sean reales. Las raíces amortiguadas no son auto-excitadas, por lo que no son de interés [5]. Esto es porque, estamos trabajando en un sistema donde los electrones tienen campos auto-generados, como se podrá ver en la sección de resultados. En ausencia de estos, el sistema no podría llegar a una inestabilidad.

La relación de dispersión (4.56) puede ser analizada sin necesitar resolver la ecuación de cuarto orden. Sean $x = \omega/\omega_p$ y $y = hv_0/\omega_p$. Entonces, podemos reescribir (4.56) de la forma:

$$1 = \frac{m/M}{x^2} + \frac{1}{(x-y)^2} = F(x, y) \tag{4.59}$$

Ahora, se sabe que esta función tendrá singularidades en $x = 0$ y $x = y$. La intersección de esta curva con la línea $F(x, y) = 1$ nos dará los valores de x que satisfacen la relación de dispersión.

En la Figura 4.2 se puede ver que hay cuatro intersecciones sobre el eje de 1. Esto quiere decir que las cuatro soluciones de la ecuación de orden 4 de (4.56) son reales. En cambio, si vemos la Figura 4.3 esta solo tiene dos intersecciones. Esto muestra que con valores de y lo suficientemente pequeños—$\omega_p \gg kv_0$—entonces solo se tiene dos raíces reales. Como estas raíces vienen en pares conjugados, las otras dos raíces deben ser complejas y representan oscilaciones que hacen del plasma inestable. La máxima velocidad de crecimiento está dada por [5]:

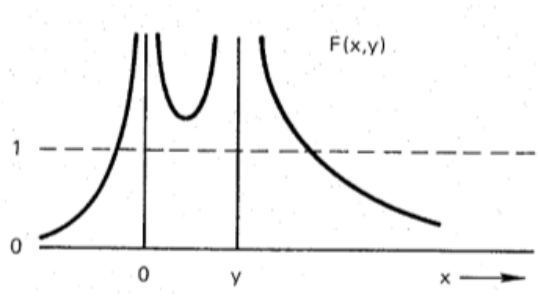


Figura 4.3: Se muestra a $F(x, y)$ en la inestabilidad Two-stream cuando el plasma es inestable [5]

$$\mathrm{Im}\left(\frac{\omega}{\omega_p}\right) = \left(\frac{m}{M}\right)^{\frac{1}{3}} \tag{4.60}$$

Una lectura atenta hará que nos demos cuenta que hay una aparente contradicción en lo que estamos estudiando: para que suceda una inestabilidad se requieren valores muy pequeños de la velocidad. Esto no tiene mucho sentido físico, debido a que la energía cinética es lo que, en primer lugar, mueve la inestabilidad.

La explicación es la siguiente: la frecuencia natural del plasma es ω_p y la frecuencia de oscilación de los iones es $\Omega_p = (m/M)^{\frac{1}{2}}\,\omega_p$. Debido al efecto Doppler de la frecuencia natural en los haces de electrones, estas dos frecuencias pueden coincidir si se dan los valores justos de kv. Además, las densidades de iones y electrones satisfacen la ecuación de Poisson. Todo esto nos podría mostrar energías cinéticas negativas que, a una primera mirada, carecerían de todo sentido. Pero esta energía cinética negativa es la diferencia entre la energía cuando no hay y cuando hay oscilación.

Este resultado nos diría que la energía cinética es menor cuando hay oscilación que cuando no la hay. En un haz sin perturbación, la energía cinética sería $\frac{1}{2}mn_0v_0^2$. En cambio, cuando hay oscilación la energía cinética sería $\frac{1}{2}m(n_0 + n_1)(v_0 + v_1)^2$ y este valor resulta ser menor que el anterior [5]. Esto es menor debido la relación entre n_1 y v_1 que se expone en su ecuación de continuidad(4.50). Gracias a esto, al trabajar las ecuaciones, la energía cinética con oscilaciones resulta ser menor a que no tiene oscilación[5]. Consecuentemente, las oscilaciones de los electrones tienen energía negativa y las oscilaciones de los iones tienen energía positiva, pero ambas oscilaciones se comportan de tal forma que la energía se conserva.

4.10. Inestabilidad Beam-plasma

La ***inestabilidad Beam-plasma*** es consecuencia directa de la inestabilidad Two-stream, salvo con la diferencia en que uno de los haces de electrones tiene una menor densidad que el otro. A lo largo de la bibliografía, pero principalmente en en los libros de Birsdall [2], Fox[10] y Hockney[19] se trabajó con haces donde uno tiene el 10 % de la densidad total y otro el 90 %, pero estas proporciones pueden variar.

Como ya se ha visto, el fundamento teórico detrás de estas inestabilidades se construye alrededor de la ***relación de dispersión***, que es una función que nos relaciona la frecuencia y el número de onda. Entonces, para entender la inestabilidad Beam-plasma se hace un planteamiento igual que en la prueba Two-stream, salvo unos pequeños cambios:

$$(0.10)\frac{\partial v_i}{\partial t} = \frac{q}{M}E_1 \tag{4.61}$$

$$(0.90)\left[\frac{\partial v_e}{\partial t} + (v_0 \cdot \nabla)\right] = \frac{q}{m} E_1 \tag{4.62}$$

Donde ya se hace el ajuste de la distribución del porcentaje de electrones en cada haz. Ahora, sustituyendo en (4.61) la forma de las oscilaciones expuesta en (4.46), se obtiene:

$$v_i = \frac{iq}{0.10\omega M} E\mathbf{x} \tag{4.63}$$

$$v_e = -\frac{iq}{m} \frac{E\mathbf{x}}{0.90(w - kv_0)} \tag{4.64}$$

Con esto se hace el mismo procedimiento para llegar la (4.56).

¿Qué se espera observar en una simulación de Beam-plasma? Al haz con menor densidad de electrones lo podemos tratar como a un rayo y al que tiene mayor densidad como un fondo de electrones. Conforme el campo eléctrico oscila, este hace que la energía del rayo de electrones tenga «pulsaciones». Estas pulsaciones causarán inestabilidades similares a las vistas en la sección de Two-stream. Esto se puede ver en las sección de 6.3 del capítulo Discusión y resultados.

4.11. Amortiguamiento de Landau

El ***amortiguamiento de Landau*** es, quizá, el resultado más importante de la física del plasma. Su importancia yace en el hecho de que fue escubierto en la década de 1920 por Lev Landau, quien fue el primero en notar que los modelos de fluidos no lograban comprender en su totalidad el fenómeno del plasma. Además, el amortiguamiento de Landau se puede aplicar a varios fenómenos de astrofísica. En este trabajo se incluye una deducción más breve utilizando integrales de contorno, pero si se busca consultar una deducción que no utilice este método, se puede consultar la sección 7.6 del libro de Chen [5].

Para empezar, se asume un plasma uniforme con una distribución $f_0(\mathbf{v})$. El campo eléctrico inicial es 0 y denotamos la perturbación en $f(r, v, t)$ como $f_1(r, v, t)$. Entonces:

$$f(x, \mathbf{v}, t) = f_0(\mathbf{v}) + f_1(r, v, t) \tag{4.65}$$

Como la velocidad es una variable independiente, la ***ecuación de Vlasov*** [5] de primer orden para los electrones es:

$$\frac{\partial f_1}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{f_1} - \frac{q}{m} \mathbf{E_1} \cdot \frac{\partial f_0}{\partial \mathbf{v}} = 0 \tag{4.66}$$

Como hemos trabajado en las inestabilidades, se asume que los iones están fijos y también se asume que las ondas electroestáticas son planas. Por lo tanto:

$$f_1 \propto e^{i(kv - \omega t)} \tag{4.67}$$

Utilizando (4.67) en (4.66), obtenemos:

$$-i\omega f_1 + ikv_x f_1 = \frac{q}{m} E_x \frac{\partial f_0}{\partial v_x} \tag{4.68}$$

Despejando a f_1:

$$f_1 = \frac{iqE_x}{m}\frac{\partial f_0/\partial v_x}{\omega - kv_x} \tag{4.69}$$

Utilizando la ecuación de Poisson y substituyendo (4.69) obtenemos:

$$\epsilon_0 \nabla \mathbf{E_1} = ik\epsilon_0 E_x = -qn_1 = -q\int\int\int f_1 d^3v \tag{4.70}$$

Un factor n_0 puede ser factorizando si se reemplaza f_0 por una función normalizada $\hat{f}_0$ [5].

$$1 = \frac{\omega_p}{k}\int_{-\infty}^{\infty} dv_x \int_{-\infty}^{\infty} dv_y \int_{-\infty}^{\infty} \frac{\partial \hat{f}_0(v_x, v_y, v_z)/\partial v_x}{\omega - kv_x} \tag{4.71}$$

Si f_0 es una distribución Maxweliana, las componentes y y z se descartan brevemente debido a que la distribución es factorizable[5]. Entonces, una distribución $\hat{f}_0$ de Maxwell de una sola dimensión se vería de la siguiente forma:

$$\hat{f_m}(v_x) = \left(\frac{m}{2\pi KT}\right)^{1/2} \exp\left(-mv_x^2/2KT\right) \tag{4.72}$$

Como estamos trabajando en una sola dimensión, para simplificar la notación, se puede omitir la escritura de v_x y hacerlo solo como v. Entonces, en esta nueva notación, nuestra relación de dispersión es:

$$1 = \frac{\omega_p}{k}\int_{-\infty}^{\infty} \frac{\partial \hat{f}_0/\partial v}{v - (\omega/k)} dv \tag{4.73}$$

Esta integral no es tan directa como se cree a primera vista. La razón es la singularidad en $v-(\omega/k)$. Se podría pensar que no es necesario considerar esta singularidad ya que ω siempre tendrá una forma compleja debido a las oscilaciones amortiguadas siempre presentes debido a las colisiones internas entre partículas. Sin embargo, esta aproximación no es correcta y en esta interpretación está el mérito de Landau, al ser el primer en abordar propiamente esta ecuación. Landau descubrió que, a pesar de que la singularidad yace en el contorno de la integral, su presencia introduce una modificación en la forma de la relación de dispersión, efectos no predichos en la perspectiva de la teoría de fluidos.

Para seguir los pasos de Landau hay que asumir que tenemos un plasma con una perturbación sinusoidal, como se ha hecho en las otras pruebas. Esto implica que k sea real y, si la perturbación cae o aumenta, ω oscilará, por lo que tendrá un valor complejo. Entonces, esta integral se tiene que abordar como una integral de contorno.

En la Figura 4.4 se tiene dos casos: el caso (a) representa una onda inestable y el caso (b) una onda amortiguada. Generalmente, esta integral se haría en el eje real de v utilizando el ***Teorema del residuo*** [5], el cual dice que:

$$\int_{C_1} G dv + \int_{C_2} G dv = 2\pi i R(\omega/k) \tag{4.74}$$

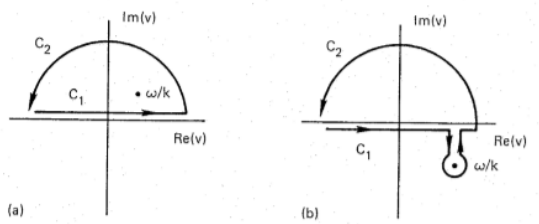


Figura 4.4: Integrales de contorno para el problema de Landau: (a) para $\mathrm{Im}(\omega) > 0$ y (b) para $\mathrm{Im}(\omega) < 0$ [5].

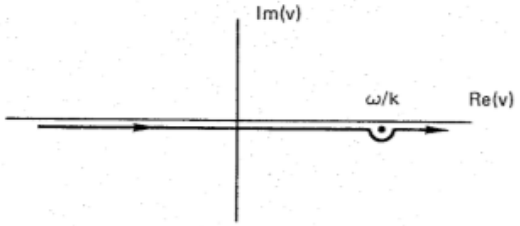


Figura 4.5: Integral de contorno para valores pequeños de $Im(\omega)$ [5].

donde G es el integrante, que por las condiciones de una integral compleja, debe ser continua sobre un intervalo cerrado; C_1 es el camino en el eje x, C_2 es el semicírculo en el infinito y $R(\omega/k)$ es el residuo en ω/k. Desgraciadamente, esto no se cumple para una distribución Maxweliana, debido al término exponencial, el cual crece conforme $v \to \pm i\infty$. Entonces, la contribución de C_2 no puede obviarse. Landau demostró que si se aborda este problema como uno de valores iniciales, la forma de correcta de resolverlo es utilizando la curva C_1 pasando por debajo de la singularidad. Esta integral debe evaluarse numéricamente pero Fried y Conte, en su libro, crearon unas tablas para ahorrarnos ese trabajo [4], aunque sus resultados solo se aplican cuando $\hat{f}_0$ es una distribución Maxweliana.

Sin embargo, un análisis exacto de este problema sigue siendo muy complicado. Para simplificarlo se hacen aproximaciones a una alta velocidad de fase y un amortiguamiento pequeño. En este caso el polo en ω/k se encuentra muy cerca del eje real de v, así como se ve en la Figura 4.5.

Rodeado el polo, se obtiene $2\pi i$ veces el residuo local. Entonces, la ecuación (4.73) se vuelve:

$$1 = \frac{\omega_p^2}{k^2}\left[P\int_{-\infty}^{\infty}\frac{\partial \hat{f}_0/\partial v}{v-(\omega/k)} + i\pi\frac{\partial \hat{f}_0}{\partial v}\Big|_{v=\omega/k}\right] \tag{4.75}$$

donde P es el valor principal de Cauchy, que se explica en el apéndice 10.8. Si la velocidad de fase es lo suficientemente larga, entonces las partes del contorno que serán obviadas a la hora de entrar y salir del polo serán despreciables. Esto se debe a que tanto $\hat{f}_0$ como $\partial\hat{f}_0/\partial v$ son pequeñas en esos puntos. Entonces, la integral de (4.75) se resuelve como una integral por partes:

$$\int_{-\infty}^{\infty}\frac{\partial \hat{f}_0}{\partial v}\frac{dv}{v-v_\phi} = \left[\frac{\hat{f}_0}{v-v_\phi}\right]_{v=-\infty}^{\infty} - \int_{-\infty}^{\infty}\frac{-\hat{f}_0 dv}{(v-v_\phi)^2} = \int_{-\infty}^{\infty}\frac{\hat{f}_0 dv}{(v-v_\phi)^2} \tag{4.76}$$

donde $v_\phi = \omega/k$ es lo mismo que la velocidad de fase. Como este resultado es solo un promedio sobre la distribución, la parte real de de la relación de dispersión se puede escribir como:

$$1 = \frac{\omega^2}{k^2}\overline{(v - v_\phi)^{-2}} \tag{4.77}$$

Como se asumió que $v_\phi \gg v$ se puede expandir $(v - v_\phi)^{-2}$ por series de Taylor:

$$(v - v_\phi)^{-2} = v_\phi^{-2}\left(1 - \frac{v}{v_\phi}\right)^{-2} = v_\phi^{-2}\left(1 + \frac{2v}{v_\phi} + \frac{3v^2}{v_\phi^2} + \frac{4v^3}{v_\phi^3} \cdots\right) \tag{4.78}$$

Los términos impares desaparecen al sacar el promedio[5], entonces:

$$\overline{(v - v_\phi)^{-2}} \approx v_\phi^2\left(1 + \frac{3v^2}{v_\phi^2}\right) \tag{4.79}$$

Ahora, evaluamos en $\hat{f}_0$(4.72), que es una distribución Maxweliana, a $v^2 = v_x^2$. Entonces:

$$\frac{1}{2}mv_x^2 = \frac{1}{2}KT_e \tag{4.80}$$

Substituyendo (4.80) en (4.77) la relación se dispersión se vuelve:

$$\omega^2 = \omega_p^2 + \frac{\omega_p^2}{\omega^2}\frac{3KT_e}{m}k^2 \tag{4.81}$$

Si el ajuste térmico es pequeño, podemos substituir $\omega_p \approx \omega$ en el segundo término. Entonces, tenemos:

$$\omega^2 = \omega_p^2 + \frac{3KT_e}{m}k^2 \tag{4.82}$$

Bueno, esto fue el desarrollo para el término real, ahora vamos a elaborar el término imaginario. Si observamos la ecuación (4.73) y se evalúa el término pequeño, debería ser lo suficientemente preciso desestimar las correcciones térmicas de la parte real y hacer $\omega = \omega_p$ [5]. En las ecuaciones (4.73) y (4.76) se puede observar que el valor principal de la ecuación (4.73) se aproxima a ω^2/k^2. Entonces,(4.73) se vuelve:

$$1 = \frac{\omega_p^2}{\omega^2} + i\pi\frac{\omega_p^2}{k^2}\frac{\partial \hat{f}_0}{\partial v}\Big|_{v=v_\phi} \tag{4.83}$$

Si se observa el desarrollo de las ecuaciones a partir de (4.75) a (4.79), se puede observar que el valor principal de Cauchy se acerca a k^2/w^2. También se puede ser la aproximación $\omega^2 \approx \omega_p^2$ debido a que la corrección térmica ha sido despreciada[5]. Entonces, la ecuación (4.75) se vuelve:

$$\omega_p^2 = \omega^2\left(1 - i\pi\frac{\omega_p^2}{k^2}\left[\frac{\partial \hat{f}_0}{\partial v}\right]_{v=v_\phi}\right) \tag{4.84}$$

Tratando el término imaginario como uno pequeño, podemos eliminar los términos cuadráticos después de una expansión de Taylor:

$$\omega_p \approx \omega \left(1 - i\pi \frac{\omega_p^2}{k^2} \left[\frac{\partial \hat{f}_0}{\partial v} \right]_{v \approx v_\phi} \right) \tag{4.85}$$

Como $\hat{f}_0$ es una distribución Maxweliana unidimensional, tenemos que:

$$\frac{\partial \hat{f}_0}{\partial v} = \frac{2v}{\sqrt{\pi} v_t^3} \exp\left(\frac{-v^2}{v_t^2} \right) \tag{4.86}$$

Se puede aproximar a v como ω/k en el coeficiente, pero se debe mantener la corrección térmica en el exponente[5], tal como se muestra en la ecuación (4.82). Ahora, tomando esto en cuenta y obteniendo $\mathrm{Im}(\omega)$ e $\mathrm{Im}(\omega_p)$ y simplificando, obtenemos:

$$\mathrm{Im}\left(\frac{\omega}{\omega_p} \right) = -0.22\sqrt{\pi} \left(\frac{\omega_p}{k v_t} \right)^3 \exp\left(\frac{-1}{2k^2 \lambda_D^2} \right) \tag{4.87}$$

Como $\mathrm{Im}(\omega)$ es negativo, entonces existente un amortiguamiento de ondas planas que no es causado por colisiones dentro del plasma. Este amortiguamiento es muy pequeño para valores pequeños de $k\lambda_D$. Este efecto está estrechamente relacionado con la perturbación inicial del sistema.

Esta deducción, un poco más simplificada pero respetando el razonamiento que hizo Landau en su momento, ha tenido muchas repercusiones en la física del plasma y en otros campos desde su descubrimiento. No solo por ser uno de los resultados más importantes de los plasmas sin colisiones, también tiene el mérito de ser uno de los grandes trabajos de la matemática aplicada, al ser un fenómeno descubierto primero en la teoría y luego de forma experimental.

Pero, ¿qué nos dice el amortiguamiento de Landau? Lo primero que hay que notar es que $\mathrm{Im}(\omega)$ aparece en el polo, cuando $v \to v_\phi$. Entonces, el amortiguamiento de Landau surge porque hay partículas que tienen, aproximadamente, la misma velocidad de fase. Podemos tener casos en que la partícula va más lento que la onda, por lo que la partícula se ve «empujada» por la onda y ocurre un intercambio de energía: la partícula gana energía y la onda pierde energía. En el caso de que la partícula tenga una velocidad más alta que la onda, sucede que la partícula «jala» a la onda. Entonces, la partícula pierde energía y la onda gana. Una distribución Maxweliana tiene más electrones lentos que rápidos [5]. Por lo tanto, la onda pierde más energía de la que gana y esto causa el amortiguamiento.

4.12. Aspectos importantes de la simulación

En esta sección se explicarán los métodos, tanto computacionales como numéricos, para efectuar una simulación de un plasma por el método Particle-in-Cell. Además, se explica cómo estos métodos se vieron envueltos en las simulaciones efectuadas.

4.12.1. Método Particle-in-Cell

Para modelar plasmas hay dos métodos populares: Particle-in-Cell (PIC) y Magnetohidrodinámica (MHD). El primero, busca resolver las ecuaciones de movimiento para cada partículas del plasma y el segundo busca modelar el plasma como un fluido. Este trabajo se enfoca en el método PIC.

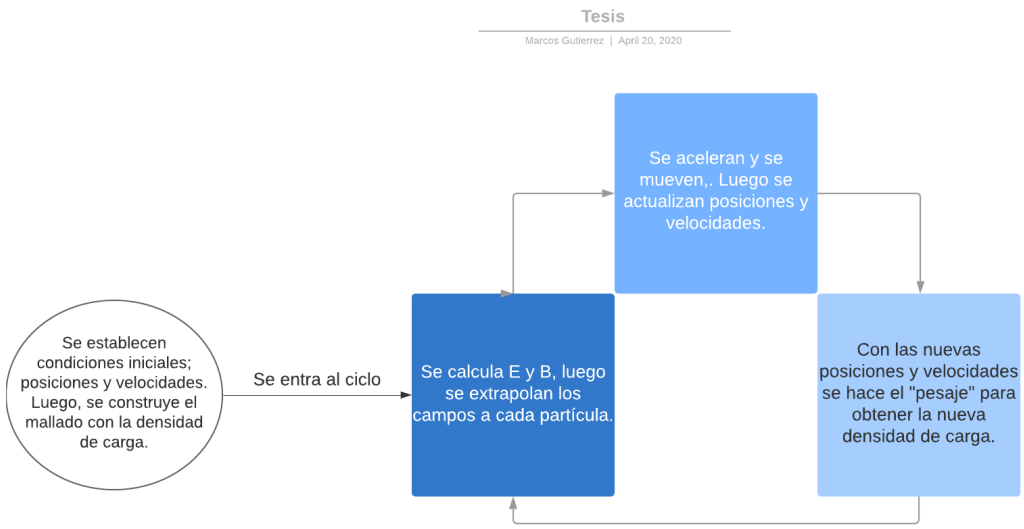


Figura 4.6: Esquema del método Particle-in-Cell (PIC).

Resolver las ecuaciones de movimiento de cada partícula es, computacionalmente, insostenible. Por eso surge el método PIC: este trabaja ciertas condiciones en un ***mallado***, que definir celdas de un tamaño determinado donde el plasma se introduce y se limita bajo condiciones de frontera en los limites de la malla. De esta malla y sus celdas se obtiene, en nuestro esta información es el campo eléctrico y la densidad de carga, y esta se extrapola a las partículas del plasma.

La ecuación de movimiento a resolver es la siguiente:

$$m\mathbf{a} = \mathbf{F} \tag{4.88}$$

Donde $\mathbf{F}$ es una la fuerza de Lorentz (10.1). La discretización de esta ecuación se hace a través del método Leap-frog, el cual veremos en la Sección 4.12.2.

Ya que para obtener el campo eléctrico se necesita la densidad de carga, esta se obtiene a través de una extrapolación de las posiciones de las partículas sobre a la malla. Este proceso se hace través de hace un pesaje de primer orden llamado Cloud-in-Cell. Los detalles de este método y del método Leap-frog se exponen más adelante en esta sección.

Por último, se necesita calcular el campo eléctrico: se resuelve la ecuación de la Ley de Gauss con la la densidad de carga obtenida[17]. Para resolver esto hay varios métodos en toda la bibliografía. En este trabajo se decidió trabajar con una integración numérica con el método del trapecio ya que anteriormente se han obtenido buenos resultados con este metodo como es el caso de Gibbon[18] .

En conclusión, el método se puede resolver mediante el esquema de la Figura 4.6.

4.12.2. Esquema Leap-frog

En [17] se recomienda trabajar la solución de las ecuaciones a través del método Leap-frog. Este es mucho más preciso que el método de Euler y requiere menos poder de computo que los métodos de Runge-Kutta[17]. Para lograr simular un plasma frío en el caso electroestático, lo que se busca conseguir es lo siguiente: a través de de valores del campo eléctrico obtener información de la velocidad,a través del cual se obtiene información sobre la posición de las partículas.

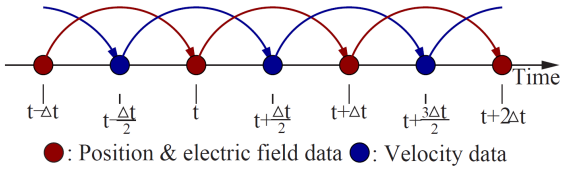


Figura 4.7: Representación del esquema Leap-Frog. La línea roja representa la posición y los datos del campo eléctrico, mientras que la línea azul representa la velocidad. Este diagrama se encuentra en [17].

De la ecuación de la fuerza de Lorentz (10.1) se obtiene las siguientes ecuaciones:

$$\frac{dv}{dt} = \frac{q}{m}E\left(t\right) \qquad \frac{dx}{dt} = v\left(t\right) \tag{4.89}$$

Reescribiendo estas ecuaciones en notación Δ o en pasos de tiempo discretos—debido a que las computadoras no manejan la diferenciación—obtenemos lo siguiente:

$$\frac{\Delta v}{\Delta t} = \frac{v_{n+1} - v_n}{\Delta t} = \frac{q}{m}E\left(t\right) \tag{4.90}$$

$$\frac{\Delta x}{\Delta t} = \frac{x_{n+1} - x_n}{\Delta t} = v\left(t\right) \tag{4.91}$$

Los subíndices n representan los pasos temporales recorridos.

Esto lo podemos reescribir como:

$$v_{n+1} = v_n + \frac{q}{m}E_n\Delta t \tag{4.92}$$

$$x_{n+1} = x_n + v\Delta t \tag{4.93}$$

Estas ecuaciones serían las utilizadas en los métodos de Euler o Runge-Kutta. El método de Leap-frog se caracteriza por «escalonar» las ecuaciones anteriores:

$$v_{n+\frac{1}{2}} = v_{n-\frac{1}{2}} + \frac{q}{m}E_n\Delta t \tag{4.94}$$

$$x_{n+1} = x_n + v_{n+\frac{1}{2}}\Delta t \tag{4.95}$$

Con estas ecuaciones, el método consiste en, a partir de datos conocidos de la velocidad, podemos determinar las nuevas posiciones de las partículas. Este salto de datos conocidos de la velocidad a nuevos datos de la posición se asemeja a un salto de rana, de ahí el nombre del método. En la Figura 4.7 se pueden visualizar los saltos de este método.

Este método, que es más preciso que el método de Euler requiriendo menor poder de computo, presenta el siguiente problema: la velocidad debe estar centrada respecto al tiempo. Cualquier cambio en el Δt establecido, requeriría recalcular los valores iniciales de la velocidad. Para resolver estos

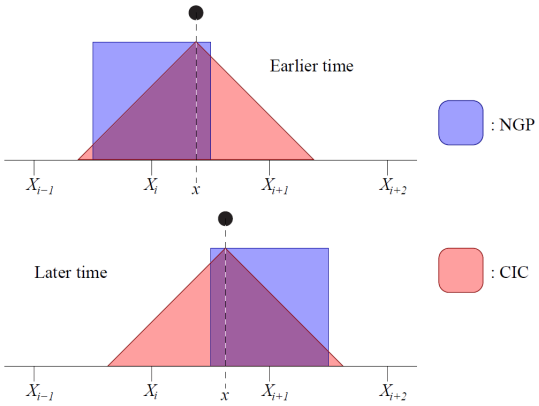


Figura 4.8: Esquema donde se compara gráficamente el método NGP con el CIC. Gráfico de Daniel Martin [17].

problemas, se trabajará con promedios de las velocidades cuando sea necesario. Para el desarrollo futuro, la energía cinética se aproximará de la siguiente forma:

$$T_n = \frac{1}{2} m v_{n-\frac{1}{2}} v_{n+\frac{1}{2}} \tag{4.96}$$

4.12.3. Pesaje de grado 1: Cloud-in-Cell (CIC)

La ***etapa de pesaje*** es cuando se considera el aporte de las partículas a la malla. En este caso, es cómo las partículas en conjunto dotan a la malla de una densidad de carga y luebo se obtiene el campo eléctrico. Si bien se podría calcular los campos de cada carga de manera individual, esto implicaría un excesivo poder de computo que haría el proceso inviable. En cambio, a través del ***método Cloud-in-Cell***, el comportamiento de cada partícula se visualiza como una nube de partículas. Además del método CIC existe uno de grado 0, llamado Near-grid-Point (NGP), que genera muchos errores[2], por lo que no se le consideró en este trabajo.

En concreto, en el método CIC, cada partícula se ve asociada a dos puntos en la malla. La forma que tiene la partícula es triangular con un ancho de $2\Delta x$. La densidad de carga ρ_i es la densidad de carga asociada a una partícula que está más a la izquierda y ρ_{i+1} es la que está más a la derecha (véase la Figura 4.8).

Entonces, tomando a ρ_i como el punto a la izquierda más cercano a la partícula y a ρ_{i+1} como el más cercano a la derecha, hacemos:

La diferencia entre el NGP y el CIC es que en el primero a cada partícula se le asigna un punto en la malla. Este es un método muy torpe para manejar partículas en medio de dos malla. En cambio, el método CIC relaciona a cada partícula con dos puntos en la malla. Esto nos facilita el trabajo unidimensional debido a que las partículas siempre se encuentran entre dos puntos.

$$\rho_{i+1} = \frac{q}{\Delta x}\left[\frac{x - X_i}{\Delta x}\right] \tag{4.97}$$

y

$$\rho_i = \frac{q}{\Delta x}\left[\frac{X_{i+1} - x}{\Delta x}\right] = \frac{q}{\Delta x}\left[\frac{X_{i+1} + \Delta x}{\Delta x}\right] \tag{4.98}$$

Donde x es la posición de la partícula, X_i la posición del i-ésimo nodo de malla y Δx es el espacio entre nodos. Si se hace a $X_0 = 0$, entonces se tiene la siguiente simplificación:

$$\rho_{i+1} = \frac{q}{\Delta x}\left[\frac{x}{\Delta x} + i\right] \quad \text{y} \quad \rho_i = \frac{q}{\Delta x} - \rho_{i+1} \tag{4.99}$$

Después de obtener la densidad de carga y calcular el campo eléctrico, se aplica el campo eléctrico de la malla a la partícula para poder moverla. El campo aplicado a la partícula tiene la siguiente forma:

$$E = \left[1 - \left(\frac{x}{\Delta x} - i\right)\right] E_i + \left[\frac{x}{\Delta x} - i\right] E_{i+1} \tag{4.100}$$

Con estos resultados, ya tenemos los algoritmos necesarios para proceder a las simulaciones.

4.12.4. Diferencia entre el laboratorio y una simulación

Para explicar bien esta diferencia, se apoyará la explicación en un gráfico muy ilustrativo que Birdsall y Langdon brindan en su libro[2]. Este gráfico es la Figura 4.9. La aclaración en esta Figura es importante para las inestabilidades Two-stream y Beam-plasma.

Como se puede observar en la Figura 4.9, la parte (a) es cómo se verían los haces en un laboratorio. Esto sería como si estuviéramos en algún lugar disparando dos haces de plasma uno contra el otro. La parte (b) es cómo lo vemos en la simulación y la parte (c) es cómo se ven las distribuciones iniciales de las velocidades de cada haz (en este caso, ambas Maxwelianas).

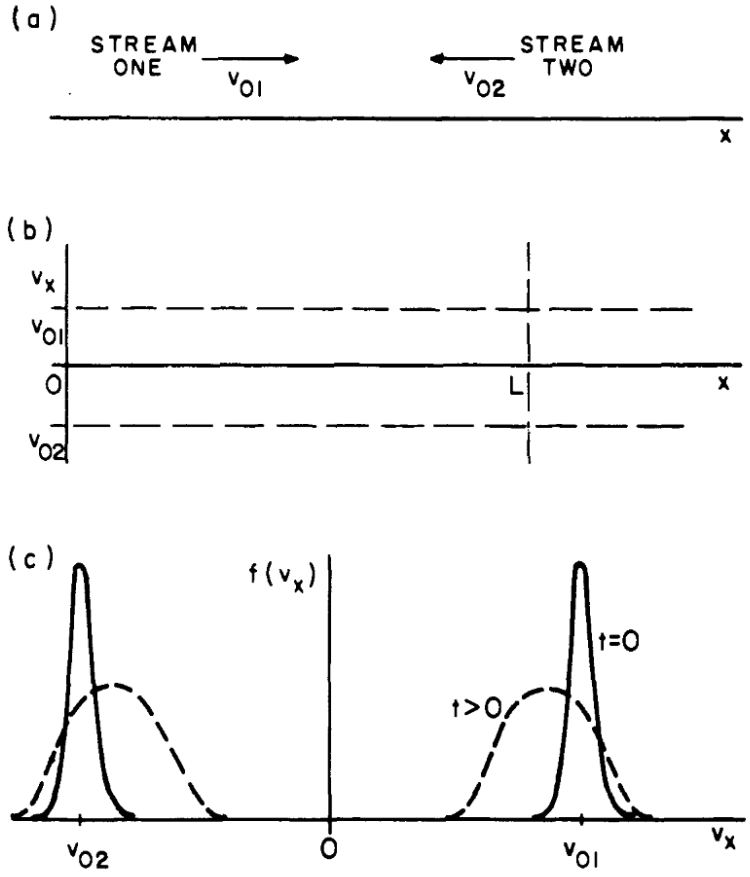


Figura 4.9: Representación visual de cómo se vería un experimento de nos haces en el laboratorio, en una simulación y sus distribuciones de velocidad.

CAPÍTULO 5

Metodología

En este capítulo se expone todo el proceso de programación y otras operaciones necesarias para hacer las simulaciones efectivas. Se explica la naturaleza de los módulos principales: `parametros.py` y `funciones.py`. Junto a estos módulos se encuentran otros tres: `plasmafrio.py`, `beamplasma.py` y `twostream.py`. Estos módulos contienen a las funciones pertinentes durante de la simulación, así como realizar y guardar las gráficas necesarias. No existe un módulo para el amortiguamiento de Landau porque este se obtienen suficientes pasos temporales en las simulaciones de Beam-plasma y Two-stream.

5.1. Especificaciones

Toda la programación necesaria se trabajó con la versión 3.8.2 de `Python`. Tanto la programación como los experimentos de simulación se realizaron en una laptop con Windows 10 instalado más las siguientes propiedades:

- Procesador: Intel Core i5
- RAM: 8GB
- Sistema operativo: 64 bits

5.2. Módulos

Para trabajar con la simulación se elaboraron dos módulos principales: `parametros.py` y `funciones.py`, los cuales exponemos a continuación.

5.2.1. `parametros.py`

En este módulo se incluyeron todas las variables que se utilizan de manera general en los cuatro fenómenos analizados:

- `noParticulas`: número de partículas a simular
- `noMalla`: número de nodos en la malla
- `carga_e`: carga del electrón
- `carga_i`: carga del ion
- `time_step`: número de pasos temporales a simular
- `vh`: velocidad media de los haces
- `coor_malla`: coordenada de cada nodo de la malla
- `x_inicial`: posiciones iniciales de las partículas
- `malla_longitud`: distancia entre nodos de la malla.
- `plasma_inicio`: punto inicial desde donde se empieza a medir la malla
- `plasma_final`: punto donde termina de medirse una malla
- `dx`: paso espacial
- `dt`: paso temporal
- `carga_masa`: relación carga masa de las partículas del plasma
- `rho0`: densidad del fondo de iones
- `x0`: amplitud de pertubación de la posición
- `v0`: amplitud de perturbación de la velocidad
- `ki`: array sobre el que se calcula la energía cinética
- `kdrift`: array sobre el que se calcula el drift(arrastre de energía cinética debido a la integración numérica) de energía
- `upot`: array sobre el que calcula la energía del campo eléctrico
- `totalenergy`: array sobre el que se calcula la energía total
- `x_i`: longitud de una malla
- `espacio_particulas`: espacio entre partícula y partícula
- `carga`: carga total de la distribución de partículas
- `m`: masa de cada un de las partículas

5.2.2. `funciones.py`

En este módulo están todas las funciones utilizadas para abordar los cuatro fenómenos en cuestión.

- `builgrid_pos()`: distribuye la posición inicial de las partículas
- `leapfrog()`: función que efectúa el corrimiento en la posición para que el Método Leapfrog funcione.

- `cf()`: función que se encarga de que las partículas cumplan las condiciones de frontera periódicas dentro de la malla
- `bulgrid_vel()`: función que genera las velocidades iniciales para plasma frío.
- `electricfield()`: función que calcula el campo eléctrico
- `chargevelocity()`: función que actualiza las velocidades de las partículas
- `chargeposition()`: función que actualiza las posiciones de las partículas
- `chargedensity()`: función que calcula las densidades de carga en el mallado
- `Kenergy()`: función que calcula la energía cinética en un tiempo específico
- `Uenergy()`: función que calcula la energía del campo eléctrico en un tiempo específico
- `totalenergy()`: función que calcula la energía total en un tiempo específico
- `drift()`: función que obtiene el drift de la energía cinética
- `builgrid_vel_2bp()`: función que genera las velocidades iniciales de las partículas para la inestabilidad two-stream
- `buildgrid_vel_ibp()`: función que genera las velocidades iniciales para la inestabilidad beam-plasma

5.3. Normalización

Un proceso que es importante a la hora de trabajar con simulación de fenómenos físicos es la ***normalización de unidades***. Esto permite optimizar procesos ya que no se acarrearán constantes con muchos decimales. Esto hace que todas las soluciones se hagan en coordenadas naturales.

En nuestro experimentos normalizamos porque permite optimizar procesos computacionales y se omite el arrastre de constantes. Todo esta programado tomando las siguientes consideraciones:

$$t \Rightarrow \omega_p t \qquad x \Rightarrow \omega_p x/c \qquad v \Rightarrow v/c$$
$$E \Rightarrow qE/m\omega_p c \qquad \eta_{e,i} \Rightarrow \eta_{e,i}/\eta_0$$

Donde la carga q_e, las masas m_e, m_i y la densidad electrónica η_0 valen la unidad.

5.4. Drift de energía

Un elemento que no es posible dejar de lado es el ***drift de energía***. Siempre que se efectúan simulaciones de fenómenos físicos es necesario tomarlo en cuenta debido a las operaciones numéricas que se hacen. Este fenómeno es el aumento gradual de la energía total de un sistema cerrado. Esto, en primera instancia, contradice la ley de la conservación de la energía. Pero esto sucede debido a las integraciones numéricas en un intervalo de tiempo Δt. Este aumento se ve representado por:

$$T = \sum m\mathbf{v}_{real}^2 + \sum m\delta\mathbf{v}^2 \tag{5.1}$$

Donde δv es la posible pequeña perturbación de la velocidad real de la simulación. Este error se puede aminorar con un δt muy pequeño y hacer más suave la curva de integración. Esto es simplemente un error generado por trabajar con soluciones numéricas.

5.5. Ciclo computacional

Las generalidades del metódo PIC se expusieron el la sección 4.12.1. Para ampliar el tema, esta sección se expone el ciclo con las funciones que se utilizan.

5.5.1. Condiciones iniciales

Para entender esta sección se recomendamos dar un breve repaso a la figura Recordando la Figura 4.6.

Primero, se utiliza la función `buildgrid_pos()` para construir la malla y distribuir a las partículas en ella. A este resultado se le aplica la función `leapfrog(x)` para efectuar el corrimiento necesario para hacer efectivo los métodos de integración. Luego, se utiliza una de las funciones `builgrid_vel()`, `buildgrid_vel_2bp()` o `buildgrid_vel_ibp()` para determinar las velocidades iniciales de las partículas en cada uno de los casos pertinentes.

Luego de calcular las condiciones iniciales, se entra al ciclo computacional que efectuará los ciclos que el usuario determine con la variable `time_step`.

5.5.2. Cálculo de la densidad de carga y el campo eléctrico

Ahora, para poder calcular la densidad se carga se utiliza la función `chargedensity(x)`, donde `x` es la posición de la partícula en ese momento. Para calcular el campo se utilizada la función `electricfield(rho)` donde `rho` es la densidad de carga calculada para ese instante.

5.5.3. Actualización de la velocidad y la posición

Para actualizar la velocidad y la posición se utiliza el método Leap-frog expuesto en la sección 4.12.2 mediante la función `leapfrog()`. Para actualizar la velocidad se utiliza la función `chargevelocity(v0,E)` donde `v0` es el valor anteror de la velocidad y `E` es el campo eléctrico calculado para ese instante. Luego, para actualizar la posición se utiliza la función `chargeposition(v,x0)` donde `v` es la velocidad en ese instante y `x0` es la posición anterior de cada una de las partículas. A este resultado se le aplica la función `cf(x)` para evaluar que cada una de las partículas cumpla con las condiciones de frontera periódicas.

Luego de hacer esto, se repite el procedimiento a partir de lo expuesto en la sección 5.5.2.

CAPÍTULO 6

Discusión y resultados

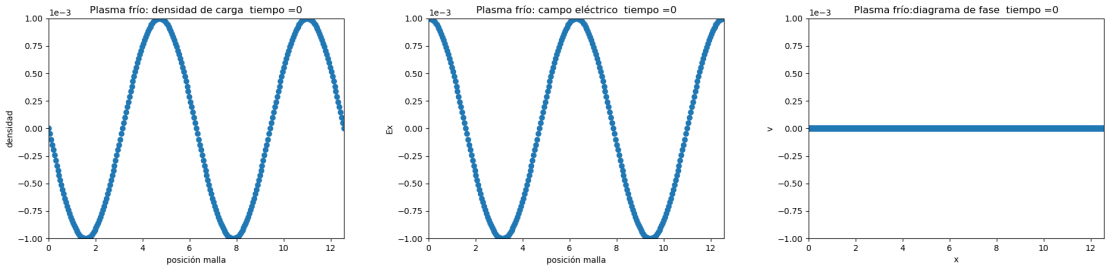
6.1. Plasma frío

Esta simulación se efectuó con $10,000$ partículas, 1000 nodos de malla, una longitud de malla de 4π[17] para que las condiciones de frontera no afectaran la visibilidad de la malla (aunque este número no es necesariamente fijo, solo tiene que cumplir las condiciones vistas en la sección 4.3), un paso temporal de 0.005 y una amplitud de perturbación de 0.001, acorde a lo recomendado en [2].

Para exponer los resultados se mostrarán en los tiempos $t = 0, 1, 1.57, 3, 15$ de la simulación donde se ven los comportamientos tanto de la densidad de carga, el campo eléctrico y el diagrama de fase.

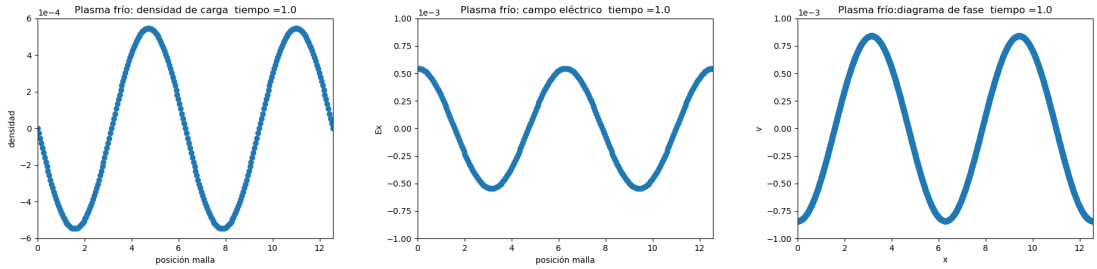
Debido a que estamos trabajando con el caso electroestático, se tiene que lo expuesto en la sección 4.7 se reduce al caso de la ecuación (4.36). Podemos observar en la secuencia de figuras 6.1b,6.2b, 6.3b, 6.4b y 6.5b que el campo eléctrico oscila. Esta oscilación cuadra con los resultados de Daniel Martin [17] y de Birdsdall [2], la cual se debe a los campos autogenerados por el movimiento de las partículas.

Otro resultado interesante en esta simulación, es que en el caso de la ecuación (4.36), los plasmas uniformes cercanos a su posición de equilibrio tienden a tener $E \approx 0$[8]. En este caso, debido a la



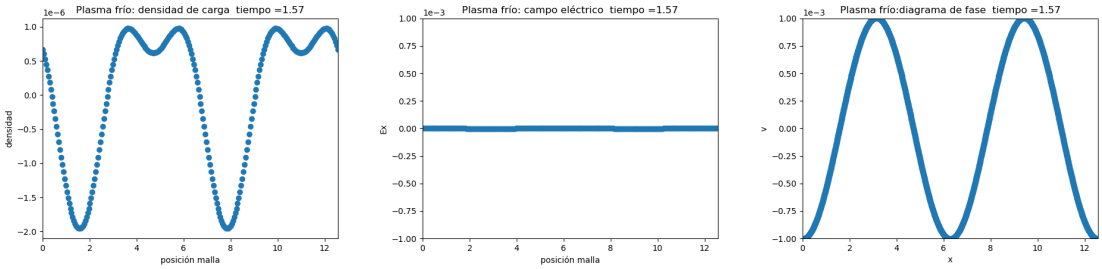
(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.1: Densidad de carga, campo eléctrico y diagrama de fase de un plasma frío en el momento inicial



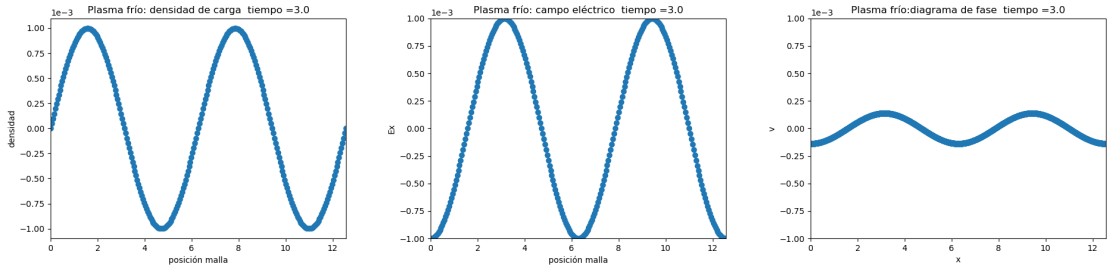
(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.2: Densidad de carga, campo eléctrico y diagrama de fase después de haber transcurrido una unidad de tiempo.



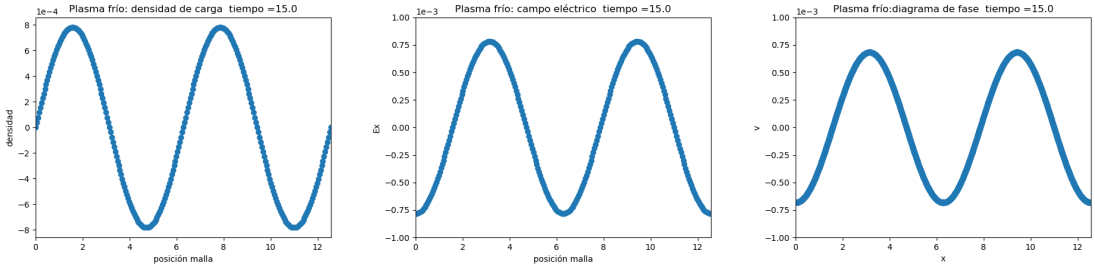
(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.3: Densidad de carga, campo eléctrico y diagrama de fase después de haber transcurrido 1.57 unidades de tiempo.



(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.4: Densidad de carga, campo eléctrico y diagrama de fase después de haber transcurrido tres unidades de tiempo.



(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.5: Densidad de carga, campo eléctrico y diagrama de fase después de haber transcurrido quince unidades de tiempo.

perturbación inicial del plasma, cuando el campo eléctrico tiende a 0 en 6.3b y en el diagrama de fase 6.3c se observa la amplitud completa de la perturbación. Asimismo, se puede observar en las gráficas de densidad de carga 6.3a una oscilación acorde a lo esperado[2], ya que la densidad de carga depende únicamente de la distribución de las partículas en la malla.

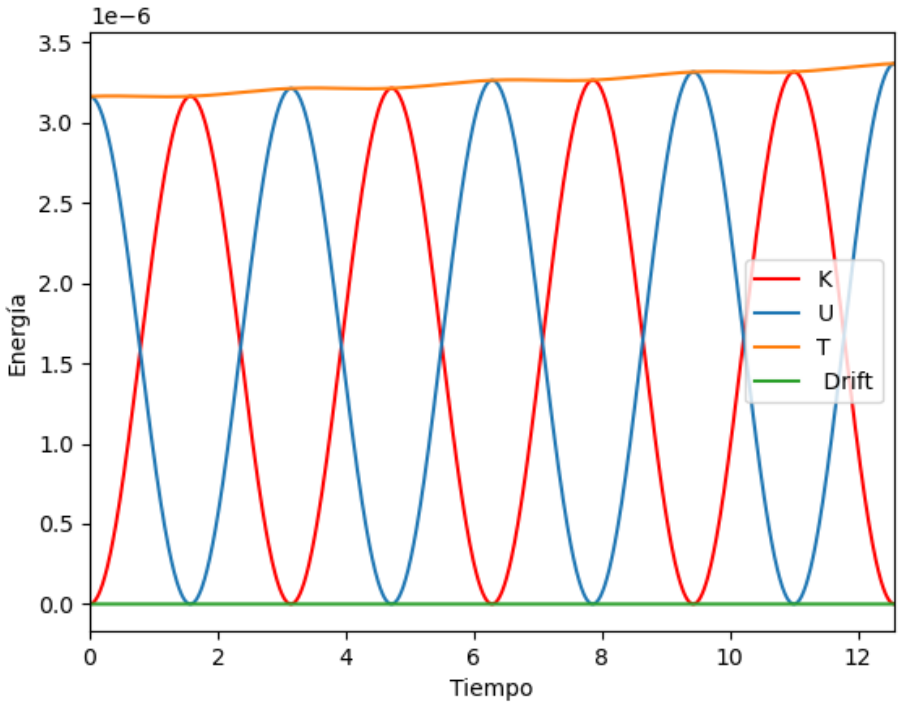


Figura 6.6: Figura de las energías potencial, cinética y total. Además, se gráfica el arrastre energético. Esta gráfica tiene un error $\epsilon \approx 4.28 \exp(-11)$ obtenido por el método de mínimos cuadrados[14]

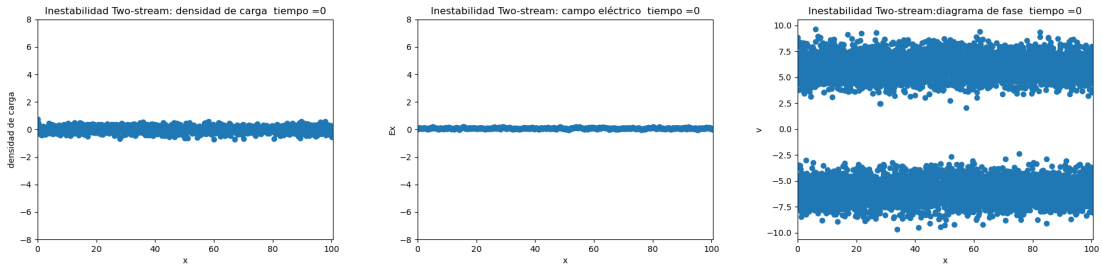
.

En la Figura 6.6 mostramos el intercambio entre la energía potencial—representa en la leyenda U—y la energía cinética—representada por la leyenda K. Además, mostramos una muy leve oscilación en la energía total. Esta puede ser causada por algún leve aumento causado por arrastre de

energía expuesto en la Sección 5.4, aunque este tiende a ser 0 debido al dt tan pequeño.En el último pico de la energía potencial se puede observar un leve aumento respecto a la oscilación anterior. Esto se debe a que, como se está trabajando con la discretización y una integral,se obtiene un error de $\epsilon \approx 4.28 \exp(-11)$ que con cada corrida se va acumulando, por lo tanto, justamente en esa corrida–la número 3000– entre error acumulado se empieza a notar.

6.2. Inestabilidad Two-stream

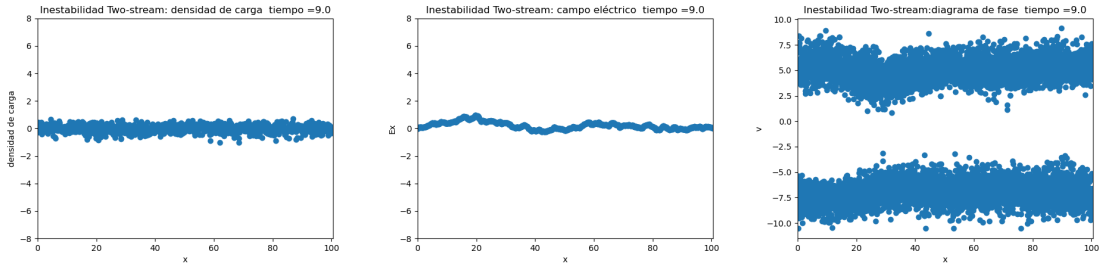
Para esta simulación se utilizaron $10,000$ partículas, 1000 nodos de malla, una longitud de malla de 32π, 200 ciclos temporales y con una velocidad media de los haces de 6. La longitud de la malla es la recomendada por Birsdall [2]. Las otras condiciones se determinaron durante las numerosas corridas del código ya que, a pesar de que todas concordaron con lo expuesto en los trabajos de Martin[17] y Birsdall[2], la escogida fue la que mostró resultados más amigables. Además, se trabajó con distintas amplitudes de perturbación en la posición, 0.0, 0.01 y 0.1, con el fin de observar como esto incide en el desarrollo de la simulación. En esta simulación, se muestran capturas de la inestabilidad en los tiempos $t, 0, 9, 18$



(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

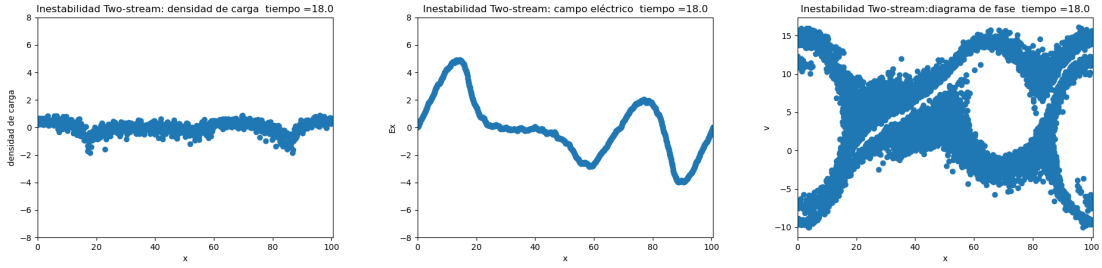
Figura 6.7: Densidad de carga, campo eléctrico y diagrama de fase de una inestabilidad Two-stream con amplitud de perturbación 0.0 en el momento inicial.

En la Figura 6.7 podemos ver el estado inicial del sistema. Como se expuso en la sección 4.9, en este momento, las soluciones de la relación de dispersión (4.56) son reales. Entonces, si se deja el sistema, después de 9 unidades temporales, en la Figura 6.11a se ve que ya se entra a un estado inestable.



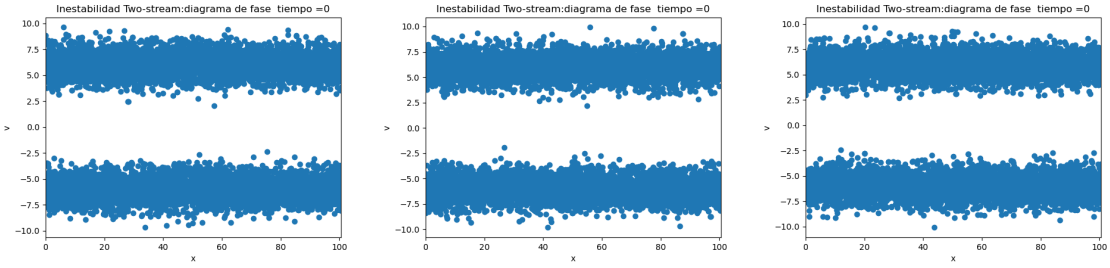
(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.8: Densidad de carga, campo eléctrico y diagrama de fase de una inestabilidad Two-stream con amplitud de perturbación 0.0 después de 9.0 unidades de tiempo



(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.9: Densidad de carga, campo eléctrico y diagrama de fase de una inestabilidad Two-stream con amplitud de perturbación 0.0 después de 18 unidades de tiempo.



(a) Amplitud 0.0. (b) Amplitud 0.01. (c) Amplitud 0.1.

Figura 6.10: Diagramas de face iniciales para inestabilidades Two-stream con amplitudes de perturbación 0.0, 0.01 y 0.1, respectivamente.

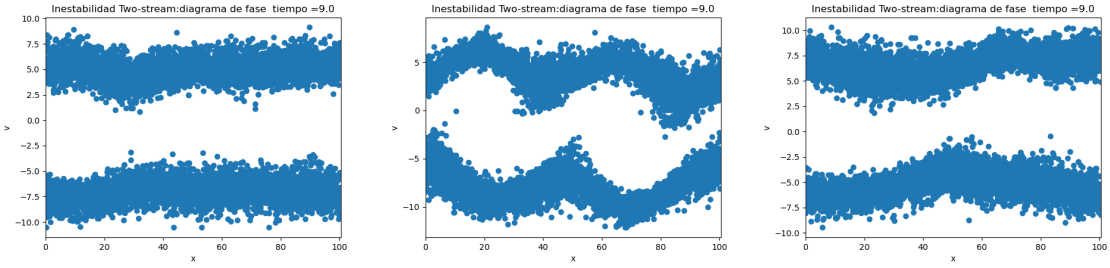
Como se puede ver en 6.8b, después de 9 unidades de tiempo ya ocurre una perturbación en el campo eléctrico. Esto se traduce en una alteración en los haces del plasma, donde se empieza a vislumbrar la aparición de un vórtice en la Figura 6.8c. Este instante donde se empieza a mostrar la inestabilidad se llama ***espacio de transición*** [17].

Después de 18 unidades temporales transcurridas, el sistema se ve como se muestra en la Figura 6.12. Esta figura se puede ver que, conforme la amplitud es mayor, la inestabilidad se acerca más rápido al estado donde hay dos vórtices que buscan alejarse entre sí. Esto sucede porque, al estar trabajando con haces de electrones, estos buscan alejarse mutuamente[17]. Estos vórtices son causados por los electrones que aún están en fase con la onda del campo eléctrico. Los demás, ya están en un estado de estabilidad.

Una tendencia que se puede observar es que cada vórtice busca alejarse lo más posible del otro. Un efecto que se espera de esta inestabilidad es que el plasma se calienta y, por ser haces solamente de electrones, los haces buscan alejarse [17]. El aumento de la temperatura lo podemos observar debido a la extensa dispersión de la velocidad de las partículas.

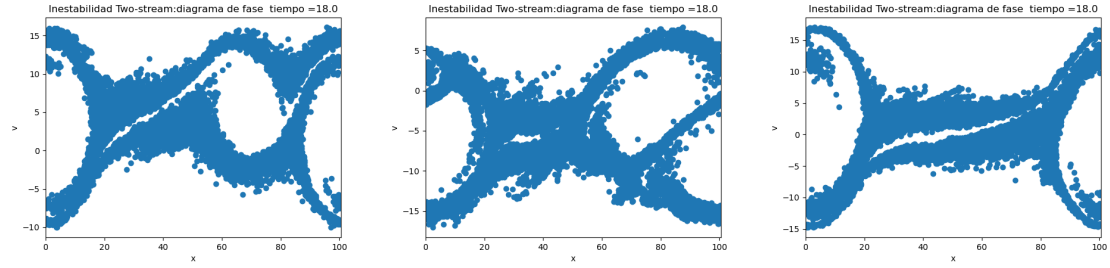
6.3. Inestabilidad Beam-plasma

Para esta simulación se utilizaron $10,000$ partículas, 1000 nodos de malla, una longitud de malla de 32π, un paso temporal de 0.1 y 200 ciclos temporales. La velocidad media fue de 6 y el haz con mayor velocidad fue el que tenía una menor concentración de partículas, tal como se realizó en los trabajos de Fox[10] y Morse[22].



(a) Amplitud 0.0. (b) Amplitud 0.01. (c) Amplitud 0.1.

Figura 6.11: Diagramas de fase después de 9 unidades de tiempo para inestabilidades Two-stream con amplitudes de perturbación 0.0, 0.01 y 0.1, respectivamente.

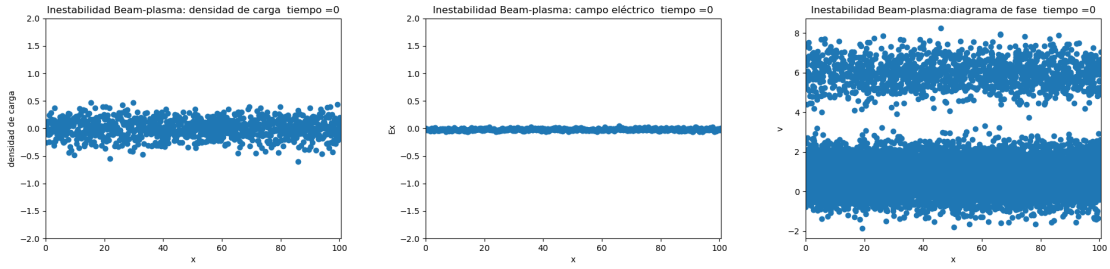


(a) Amplitud 0.0. (b) Amplitud 0.01. (c) Amplitud 0.1.

Figura 6.12: Diagramas de fase después de 18 unidades de tiempo para inestabilidades Two-stream con amplitudes de perturbación 0.0, 0.01 y 0.1, respectivamente.

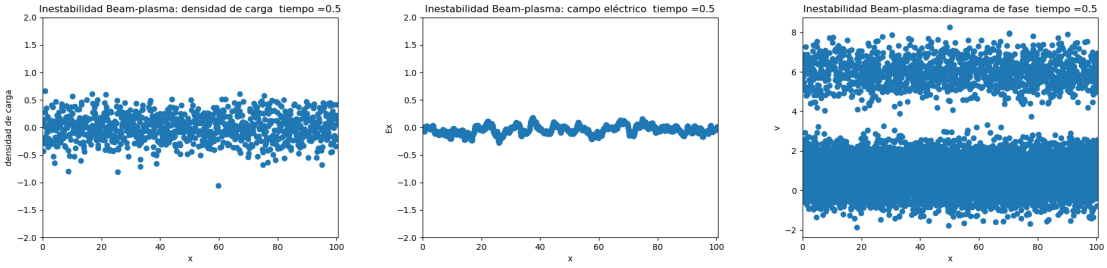
Como se explica en la Sección 4.10, el desarrollo teórico de esta inestabilidad es el mismo que la inestabilidad Two-stream4.9, pero con diferencia en la densidad de los haces. También se tiene la diferencia que la velocidad del haz con mayor cantidad de electrones se distribuye alrededor del equilibrio y la velocidad del otro se distribuye alrededor de la velocidad media del haz. En este caso, siguiendo lo expuesto en libro de Fox[10], se hizo que un haz de electrones tuviera el 10 % de todas las partículas y el otro haz se creó con el 90 %.

Comparando las imágenes 6.7c, 6.13 se puede observar la diferencia entre el diagrama de fase de una inestabilidad Beam-plasma con una Two-stream. En este momento se puede notar la diferencia de densidad entre los haces en el caso de la inestabilidad Beam-plasma.



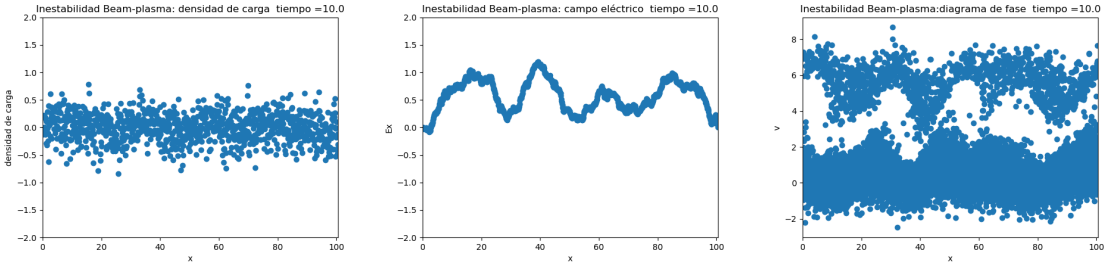
(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.13: Estado inicial de una inestabilidad Beam-plasma con amplitud de perturbación de 0.0.



(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.14: Estado después de 0.5 unidades de tiempo de una inestabilidad Beam-plasma con amplitud de perturbación de 0.0.



(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

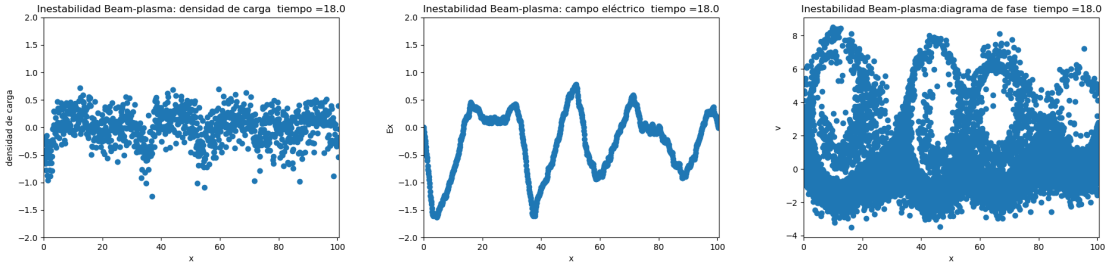
Figura 6.15: Estado después de 10 unidades de tiempo de una inestabilidad Beam-plasma con amplitud de perturbación 0.0.

Si comparamos la Figura 6.14 con su equivalente en Two-stream, la Figura 6.8, se observar que la perturbación en el campo eléctrico del caso Beam-plasma es mucho más fuerte. En el diagrama de fase los vórtices no son tan violentos como en el caso Two-stream, debido a que en este caso el haz con mayor densidad de electrones «absorbe» al haz de menor densidad, a diferencia del caso Two-stream, donde en un vórtice uno domina al otro. La diferencia entre lo visto en la Sección 6.2 reside en que, en el caso Beam-plasma, el haz más grande «dominará» al más pequeño. Esto sucede porque conforme el campo eléctrico va absorbiendo energía del haz de electrones, llega a un punto en que la energía del campo se vuelve lo suficientemente grande como para que los electrones del haz más pequeño se «atrapen» con los del haz más grande y este evento genera vórtices similares a los observados en la sección de la simulación de la inestabilidad Two-stream 6.2. Esto es visible en la Figura 6.15. Además, tal como mencionamos en la discusión de la inestabilidad Two-stream, luego de pasar por esta inestabilidad, el plasma se calienta. Esto se puede ver en la distribución del as velocidades[17].

En la Figura 6.16 se observa el estado final después de 18 unidades de tiempo. En este diagrama de fase se observa como el haz de menor densidad de ve prácticamente «dominado» por el otro. Además, se observa como el campo eléctrico se perturba debido a la densidad de carga cada vez más inestable gracias a la distribución de las partículas.

6.4. Amortiguamiento de Landau

Para observar el amortiguamiento de Landau, podemos abocarnos a dos diagramas basándonos en lo explicado en la Sección 4.87: el intercambio energético entre las partículas y el del campo



(a) Densidad de carga (b) Campo eléctrico (c) Diagrama de fase

Figura 6.16: Estado después de 18 unidades de tiempo de una inestabilidad Beam-plasma con amplitud de perturbación 0.0.

eléctrico. En este caso, se vería el amortiguamiento en la energía potencial. La otra forma de observar el fenómeno en cuestión es a través del diagrama de fase. Esto sería observando un plasma como, después de haber pasado por una inestabilidad, se empiezan a crear regiones estables.

Para efectuar estas pruebas se utilizó una amplitud de perturbación de 0.0, 20,000 partículas, 1024 nodos de malla, 2,000 ciclos temporales y un paso temporal de 0.1. Estas condiciones se establecieron después de haber efectuado numerosas corridas. Se mantuvo una amplitud de 0.0 debido a que en el libro de Birdsdall[2] se afirma que una amplitud más grande puede apresurar la aparición del amortiguamiento. También, el mismo autor menciona, que esto no se puede aminorar con aumentar el número de partículas. Debido a que estas simulaciones son muy susceptibles a las condiciones iniciales al step temporal de integración, se probó con varios valores obteniendo los mejores con los escogidos acá. Además, se muestran los solo resultados obtenidos a través de una inestabilidad Two-stream. Las inestabilidades expuestas en este trabajo, tarde o temprano, llegan al amortiguamiento de Landau[2] [5]. Sin embargo, la inestabilidad Beam-plasma, se tardaba mucho tiempo y debido a la capacidad de la computadora, no se obtenían buenos resultados. De la misma manera como se afirma en en el artículo de Qi[15], una mayor amplitud de perturbación puede llevar a presentar efectos no-lineales, y este tema se va más allá de lo que se presenta en este trabajo.

Con todas las aclaraciones pertinentes hechas, podemos pasar a ver la Figura 6.17. En esta figura se puede ver como después de que la energía potencial llega a un máximo de crecimiento en aproximadamente 95 unidades temporales, la onda empieza a amortiguarse hasta llegar a niveles entre 0 y 1000 después de 200 unidades temporales. Claro, hay ciertos datos atípicos que han surgido inevitablemente en todas las corridas. En esta gráfica, que cuadra con los resultados obtenidos por Birdsdall y Langdon[2](página 126, figura 5-15a). con diferentes parámetros, muestra como la energía del campo eléctrico se amortigua conforme las partículas le «roban» energía a las ondas. La única gráfica obtenida de este fenómeno en la bibliografía consultada se encuentra en la obra de los autores anteriormente citados. En los demás trabajos consultados se encuentran grandes desarrollos teóricos y discusiones, pero no hay otro que haya generado esta gráfica.

En la Figura 6.18 podemos observar como, a partir de 95 unidades temporales, el plasma empieza a presentar cada vez más puntos de estabilidad. Los vórtices, así como en las demás inestabilidades, representan las partículas que están en fase con la onda del campo eléctrico. Esto quiere decir que no están en ningún proceso de intercambio energético con el campo. En la figura se puede ver que, conforme avanza el tiempo, la cantidad de vórtices disminuye y como estos se van alargando hasta desvanecerse. Esto sucede porque, debido a la constante interacción de las partículas con el campo, estas se van acelerando hasta entrar en el rango de velocidad de interacción con la energía potencial. Luego, estas ganan energía gracias a su interacción con el campo, por lo que la energía potencial disminuye y la energía de la partícula aumenta. En la Figura 6.19 se puede ver como la energía cinética aumenta conforme la energía potencial disminuye. Esta figura concuerda con la Figura 6.17 porque, aproximadamente cuando el amortiguamiento comienza, la energía cinética empieza a

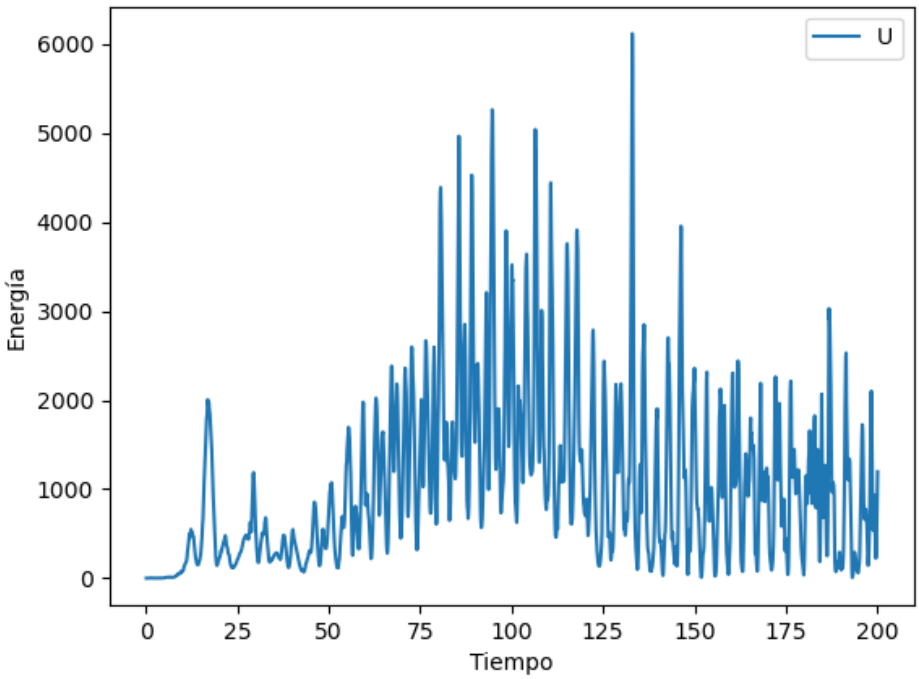
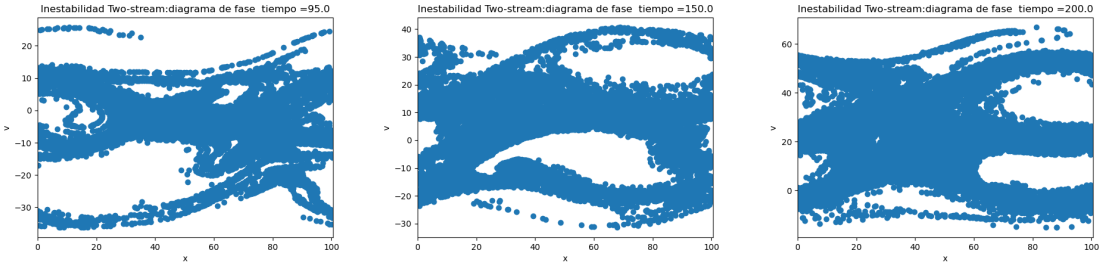


Figura 6.17: Evolución temporal de la energía potencial. Después del máximo de energía es posible ver el amortiguamiento predicho por Landau.



(a) 95 unidades temporales (b) 150 unidades temporales (c) 200 unidades temporales

Figura 6.18: En estos diagramas se puede ver le evolución temporal del diagrama de fase a partir de el máximo de energía potencial

aumentar.

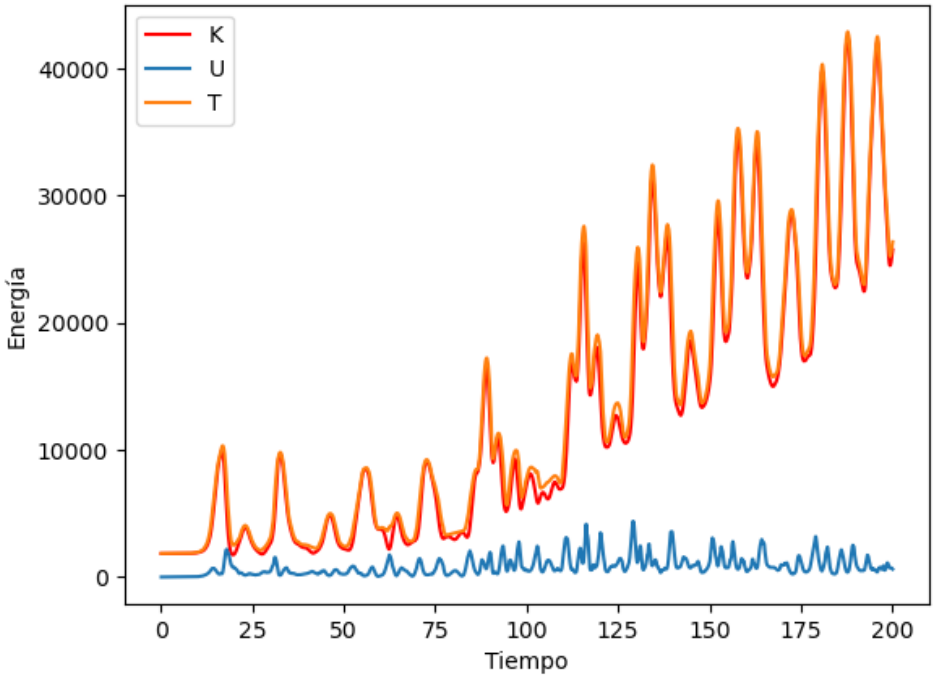


Figura 6.19: Evolución temporal de las energías potencial y cinética. Se puede observar como, conforme la energía potencial disminuye, la cinética aumenta.

6.5. Comparación de los resultados obtenidos con los antecedentes

En cuanto a la teoría fundamental del plasma hay una amplia bibliografía detrás. De la literatura consultada para los fundamentos detrás de los fenómenos estudiados destacan *Fundamentals of Plasma physics* de Paul Bellan [3] y *Plasma physics. An introduction* de Richard Fitzpatrick[8].

Además, un autor que cubre los temas de electromagnetismo aplicado al plasma es David Griffiths en su libro *Introduction to Electrodynamics* [12].

Ahora, todo el fundamento teórico del método Particle-In-Cell se encuentra ampliamente desarrollado en *Computer simulation using particles* de R. Hockney y J.Easwood[19] y en el libro fundamental del tema: *Plasma physics via computer simulation* de A.B. Langdon y C.K. Birdsall [2].

En el caso de las simulaciones, en el trabajo *Electrostatic PIC simulation of plasma in one dimension* de Daniel Martin[17] se encuentran resultados tanto de oscilaciones de plasma frío como de la inestabilidad Two-stream. Sin embargo, hay trabajos más completos en el caso de la inestabilidad Two-stream, como es el caso de *One-, two- and three-dimensional numerical simulation of two-beam plasmas* de W.C. Neslon y R.L. Morse[22]. En el caso de la inestabilidad Beam-plasma solo se encontró el libro *Parallel computing works!* de Fox, Williams y Messina[10]. En este libro se llega a los mismos resultados que los obtenidos en la sección 6.3 pero con otro métodos de computación paralela, utilizando diferentes arquitecturas para sus programas. También se consultó el trabajo de Roger Hess: *Study of Beam-plasma instability by spectroscopic methods*[13] que, utilizando métodos diferentes también llega a resultados congruentes.

En el caso del amortiguamiento de Landau, el mejor desarrollo teórico para el caso lineal (que es el abordado en este trabajo) se encuentra en *Introduction to plasma physics and controlled fusion* de Francis Chen[5]. Además, se tiene resultados de simulación en el trabajo *Simulation of linear and nonlinear Landau damping of lower hybrid waves* de Lei Qi, X.Y. Wang y Y.Lin[15] y en un capítulo del libro de A.B. Langdon C.K. Birdsall[2] anteriormente mencionado. Precisamente en esta sección se encuentra un diferenciador de este trabajo: los resultados con el libro de Lei Qi *et al* son congruentes con los obtenidos, pero desde el enfoque ondas híbridas de baja frecuencia. En el estudio mencionado, se observa el mismo amortiguamiento obtenido en la energía del campo eléctrico[15]. En cambio, con los de A.B. Langdon y C.K. Birdsall se tienen resultados desde el mismo enfoque computacional y distintos parámetros. Sin embargo, en toda la bibliografía solo se presentó un análisis gráfico de las energías durante el amortiguamiento de Landau y este se encuentra en el libro anteriormente mencionado pero con descripción y gráficas y de muy baja calidad. En este trabajo se hizo más énfasis en esta parte.

CAPÍTULO 7

Conclusiones

- Se construyó un código en `Python` 3.8 funcional para analizar oscilaciones de plasma frío, las inestabilidades Two-stream y Beam-plasma y para comprender el amortiguamiento de Landau. Estos resultados fueron congruentes con lo ya explorado en la bibliografía, como ya se ha mencionado en la discusión de los resultados y en los antecedentes.
- Se logró trabajar con una mayor cantidad de partículas,a diferencia de lo expuesto en trabajos como los de Martin[17] y Langdon y Birdsall[2], lo cual permitió observar como la interacción entre partículas y los campos autogenerados se mantiene congruente independientemente del número de partículas.
- En este trabajo se presentó en los resultados de la inestabilidad de Landau las gráficas de la energía que no había sido efectuada en la bibliografía estudiada o bien, como es el caso de la energía potencial en el libro de Langdon[2], la visualización de sus resultados no era tan amigable. Esto permitió observar el fenómeno de una manera más clara y entender como se comporta en los aspectos referentes a la energía del sistema.

CAPÍTULO 8

Recomendaciones

- Para obtener más herramientas en esta área, una recomendación pertinente, para tanto estudiantes como para la carrera de física de la UVG es dedicar más tiempo a métodos computacionales para simular fenómenos físicos.
- Para hacer una continuación de este trabajo, se puede modelar usando los métodos magnetohidrodinámicos[2] mencionados en la introducción de este trabajo.
- Para optimización de los programas y para tener mayor amplitud de herramientas se puede utilizar, como continuación a este trabajo, se puede trabajar con FEniCS[1]. Esta herramienta utilizada métodos de elementos finitos, a diferencia con este trabajo, que todo se hizo con métodos de diferencias finitas.

CAPÍTULO 9

Bibliografía

[1] A. Logg, K.-A. Mardal, G. N. Wells et al: *Automated Solution of Differential Equations by the Finite Element Method*. Springer, 2012.

[2] A.B. Langdon, C.K. Birdsall y: *Plasma physics via computer simulation*. Taylor and Francis, USA, 2ª edición, 2004.

[3] Bellan, Paul: *Fundamentals of plasma physics*. Cambridge University Press, Cambridge, 2006.

[4] Burton D. Fried, Samuel D. Conte: *The plasma dispersion function*. Academic Press, Nueva York, 1ª edición, 1961.

[5] Chen, Francis F.: *Introduction to Plasma physics and controlled fusion*. Springer, USA, Nueva York, 2ª edición, 1984.

[6] Company, The Fortran: *Fortran Language Reference, versión Fortran 77*. Disponible en: www.fortran.com.

[7] Eliezer, Shalom y Yaffa Eliezer: *The fourth state of matter. An introduction to plasma science*. Institute of Physics Publishing, Bristol and Philadelphia, 2ª edición, 2001.

[8] Fitzpatrick, Richard: *Plasma Physics. An Introduction*. CRC Press, USA, Texas, Austin, 1ª edición, 2015.

[9] Foundation, Python Software: *Python Language Reference, versión 3.8*. Disponible en: www.python.org.

[10] Geoffrey Fox, Roy Williams, Paul Messina: *Parallel computing works!* Morgan Kauffmann Publishers, USA, 1ª edición, 1994.

[11] Grifalco, Louis: *Statisticals mechanics of solids*. Oxford University Press, Oxford, 1ª edición, 2003.

[12] Griffiths, David J.: *Introduction to Electrodynamics*. Pearson, USA, 4ª edición, 2013.

[13] Hess, Roger: *Study of a Beam-plasma instability by spectroscopic methods*. University of California, Lawrence Berkeley laboratory, Diciembre 1972. Financiado por el gobierno de los Estados Unidos y su comisión de energía atómica.

[14] Kiusalaas, Jaan: *Numerical Methods in Engineering with Python*. Cambridge University Press, Cambridge, ilustrated edition edición, 2005.

[15] Lei Qi, X.Y. Wang y Y.Lin: *Simulation of linear and nonlinear Landau damping of lower hybrid waves.* API, 20(20), Junio 2013. Publicado en linea.

[16] Lynden-Bell, Donald: *The stability and vibrations of a gas of stars.* Monthly notices of Royal Astronomical Society, 124(4):23–25, Febrero 1962.

[17] Martin, Daniel: *Electrostatic PIC simulation of plasma in one dimension.* School of physics and astronomy, University of Manchester, Enero 2007. Reporte de cuarto de año de la Maestría en Física.

[18] P.Gibbon: *Introduction to plasma physics.* Indico CERN. Plasma Wake Acceleration 2014:https://indico.cern.ch/event/285444/contributions/1636921/, Noviembre 2014. Último acceso: 04/06/2020.

[19] R, Hockney, J Eastwood: *Computer simulation using particles.* Adam Hilger, Bristol, 1ª edición, 1988.

[20] reference, C++: *C++ Language Reference.* Disponible en: www.cppreference.com.

[21] Robert Goldston, Paul Rutheford: *Introduction to Plasma Physics.* Institute of Physics publishing, Bristol, 1995.

[22] W.C. Nielson, R.L. Morse y: *One-, two-, and three-dimensional numerical simulation of two-beam plasmas.* Physical Review Letters, 23(19):1087–1090, Octubre 1969.

[23] Weisstein, Eric W.: *Cauchy Principal Value.* From MathWorld–A Wolfram Web Resource: https://mathworld.wolfram.com/CauchyPrincipalValue.html.

[24] William E. Boyce, Richard C. Dipima: *Elementary Differential Equations and Boundary Value Problems.* Jhon Wiley, Nueva York, 7ª edición, 2001.

CAPÍTULO 10

Anexos

10.1. Plasmas magnetizados

Un ***plasma magnetizado*** es aquel que tiene un campo magnético externo $\mathbf{B}$ lo suficientemente fuerte como para afectar alguna de las condiciones del sistema. Estos plasmas responden de manera extraña a fuerzas fueras paralelas a $\mathbf{B}$. Un plasma magnetizado moviéndose a velocidad media $\mathbf{V}$ tiene un campo eléctrico $\mathbf{E} = -\mathbf{V} \times \mathbf{B}$ que no se ve afectado por el escudo de Debye.

Recordando la fuerza de Lorentz:

$$\mathbf{F} = q(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \tag{10.1}$$

Sabemos que las partículas cargadas responden a la ecuación (10.1) girando en un plano perpendicular al campo magnético $\mathbf{B}$. Estas partículas siguen la trayectoria de un ***radio de Larmor***, también conocido como ***giroradio***. La distancia de estas órbitas de Larmor (o giro-orbitas) son inversamente proporcionales a la magnitud del campo magnético.

El ***radio típico de Larmor*** es:

$$\rho = \frac{v_t}{\Omega} \tag{10.2}$$

donde Ω es la frecuencia del ciclotrón:

$$\Omega = \frac{eB}{m} \tag{10.3}$$

Naturalmente, hay un distinto radio de Larmor para cada especie del plasma. Si las temperaturas son parecidas, la relación entre los radios de Larmor entre electrones y iones es proporcional:

$$\rho_e \sim \left(\frac{m_e}{m_i}\right)^{\frac{1}{2}} \rho_i \tag{10.4}$$

Un plasma se considera magnetizado si la escala L es mucho más grande que el radio de Larmor. En caso contrario, cuando $\rho \gg L$, las partículas cargadas tiene trayectorias rectas. Esto nos lleva a definir el ***parámetro de magnetización***:

$$\delta = \frac{\rho}{L} \tag{10.5}$$

Hay casos de interés cuando los electrones están magnetizados y los iones no. Pero, generalmente, se tratan los casos cuando ambas especies están magnetizadas. Entonces, la expresión general del parámetro de magnetización es la siguiente:

$$\delta_i = \frac{\rho_i}{L} \gg L \tag{10.6}$$

10.2. Valor principal de Cauchy

En este trabajo, el valor principal de Cauchy se presentó en operaciones que involucraban integrales de contorno. Por eso, esta sección solo cubrirá dicho caso.

El ***valor principal de Cauchy*** es un método que permite asignar valores a integrales impropias que, en caso de utilizarse, estas resultarían indefinidas. En el caso de las integrales de contorno tenemos la siguiente situación:

$$f(z) = x + iy \tag{10.7}$$

Donde esta función compleja tiene un polo en el contorno, exactamente como se puede ver en la parte (b) de la Figura 4.4. Este polo es rodeado por un circulo de radio ε y un trozo de camino fuera de camino denominado $L(\varepsilon)$. Siempre que (10.7) sea integrable sobre $L(\varepsilon)$, sin importar el tamaño de ε, entonces, el valor principal de Cauchy es:

$$P\int_L f(z)dz = \lim_{\varepsilon \to 0} \int_{L(\varepsilon)} f(z)dz \tag{10.8}$$

En la bibliografía los valores principales de Cauchy se puede presentar con distintos nombres: valor principal de la integral, parte finita o *partie finie.* En este caso, donde se trabaja con un polo simple, la integración es relativamente sencilla. En los casos de polos con un contorno no tan bien definido, una herramienta muy útil la brinda el lenguaje Wolfram: $Integrate[f, x, a, b, PrincipalValue- > True]$ [23].

10.3. Apantallamiento eléctrico

Para entender el ***apantallamiento el eléctrico***, supongamos el siguiente escenario: Tenemos un plasma de dos especies. iones y electrones. Luego, sumergimos en el plasma un ion como partícula de prueba. Este atrae a los electrones y dispersa a los iones, el resultado de esto es una nube de carga negativa alrededor del ion. Un observador lejos de la partícula de prueba observará los potenciales combinados de tanto la partícula como la nube de electrones que la rodea. Debido a que la nube tiene una polaridad contraria a la partícula de prueba esta «apantallará» el potencial de la partícula [3]. Para calcular el apantallamiento se utiliza la ecuación de Poisson, donde las fuentes son la partícula

de prueba y la nube de electrones.

$$\nabla^2\phi = \frac{1}{\epsilon_0}\left[q_T\delta(\mathbf{r}) + \sum_\sigma n_\sigma(\sigma)q_\sigma\right] \tag{10.9}$$

Donde el término $[q_T\delta(\mathbf{r})$ representa la densidad de carga de la partícula de prueba y $\sum_\sigma n_\sigma(\sigma)q_\sigma$ representa a las partículas que interactúan con la partícula de prueba.Este último término, en ausencia de la partícula de prueba, se anularía debido a que se considera al plasma como neutro. Asumiendo que la partícula de prueba se introduce lentamente, podemos asumir que la respuesta del plasma será como una distribución de Boltzmann(4.2) [3]. También, como la perturbación es infinitesimal, podemos asumir que $q_\sigma\phi << kT_\sigma$ [3]. Entonces, la ecuación (4.2) se vuelve: $n_\sigma \approx n_{\sigma 0}\left(1 - q_\sigma\phi/kT_\sigma\right)$ [3]. Al asumir la neutralidad inicial, $\sum_{\sigma=i,e} n_{\sigma 0}q_\sigma = 0$, esto hace que todos los términos independientes de ϕ se desvanezcan [3]:

$$\nabla^2\phi - \frac{1}{\lambda_D^2}\phi = \frac{q_T}{\epsilon_0}\delta(\mathbf{r}) \tag{10.10}$$

En esta deducción, basada a partir del libro de Bellan[3], el autor asume muestra la distancia de Debye λ_D como una definición sin tomar en cuenta su relación con la frecuencia de un plasma.La ecuación (10.10) se resuelve por el método del polinomio característico[24], dando el resultado:

$$\phi(\mathbf{r}) = \frac{q_T}{4\pi\epsilon_0 r}\exp(-r/\lambda_D) \tag{10.11}$$

En este resultado se puede ver una relación directa con lo obtenido en la sección 4.3. En este caso, si tenemos que $r << \lambda_D$, entonces el potencial se comportará como el de la carga de prueba. En caso contrario, cuando $r >> \lambda_D$ entonces la carga de prueba será completamente apantallada. Como se puede ver, para que el escudo de Debye sea relevante, la segunda condición debe cumplirse.

10.4. Código

A continuación se anexan los códigos utilizados. El código se encuentra en su totalidad en el siguiente repositorio de GitHub: https://github.com/MarcosGutierrez97/Plasma-simulations-thesis. Nótese que el código presentado a continuación es la versión del código a la fecha, el cual puede diferir con el que se encuentre en el repositorio en GitHub debido a actualizaciones y optimizaciones del mismo.

Para usar este código se llama en consola al código que haga referencia a la simulación de interés:

- `plasmafrio.py` = Oscilaciones de plasma frío
- `2streamplasma.py` = Inestabilidad Two-stream
- `beamplasma.py` = Inestabilidad Beam-plasma

En caso de querer cambiar los parámetros, para explorar otras condiciones, esto se hace en `parametros.py`.

Listing 10.1: Parámetros para el ciclo PIC: `parametros.py`

```
import numpy as np
import scipy as sp
###################################
# Parametros para el ciclo del PIC
###################################
```

```
#Parametros principales
noParticulas = 1000 #Numero de particulas
noMalla = 100 #puntos de la malla
carga_e = -1 #electron
carga_i = 1 #ion
time_step = 1500 #para beamplasma #150 para two stream
vh = 6 #velocidad media del haz. Para pruebas 2-stream y Beam Stream


#Parametros de la malla
coor_malla = [float(i) for i in range(noMalla+1)]
campoEx = np.zeros(noMalla + 1)
x_inicial = np.zeros(noParticulas)
malla_longitud = 4*np.pi #Tamano de la malla espacial
plasma_inicio = 0
plasma_final = malla_longitud

#Parametros operacionales
dx = malla_longitud/noMalla
dt = 0.001

#Parametros de las particulas
carga_masa = -1 #relacion carga masa
rho0 = 1 #Densidad del fondo de iones
x0 = 0.001 #perturbacion de amplitud
v0 = 0.0 #perturbacion de velocidad
densidadI = rho0 #densidad ionica


#Parametros energeticos
ki =[0.0 for i in range (time_step + 1)]
kdrift =[0 for i in range (time_step + 1)]
upot =[0.0 for i in range (time_step + 1)]
totalenergy =[0.0 for i in range (time_step + 1)]
x_i = plasma_final - plasma_inicio #Longitud de donde va a cargar la malla
espacio_particulas = x_i / noParticulas
carga = -rho0 * espacio_particulas
m = carga/carga_e #masa
```

Listing 10.2: Funciones para el ciclo PIC: `funciones.py`

```
import parametros as pa
import numpy as np
import scipy as sp
from scipy import integrate
import matplotlib.pyplot as plt
#####################################
# Funciones para el ciclo del PIC
#####################################


#FUNCION PARA POSICIONES INICIALES
def buildgrid_pos(x_0):
    #posiciones:
    plasma_inicio = 0.0
    plasma_final = pa.malla_longitud
    x_i = plasma_final - plasma_inicio #Longitud de donde va a cargar la malla
    espacio_particulas = x_i / pa.noParticulas
    carga = -pa.rho0 * espacio_particulas
    masa = carga/pa.carga_e

    for i in range(pa.noParticulas):
        x_0[i] =  plasma_inicio + espacio_particulas * (i + 0.5)
        x_0[i] += pa.x0 * np.cos(x_0[i])
    return x_0
```

```
def leapfrog(x,v): #Necesaria para que el proceso de Leapfrog sea valido
    for i in range (len(x)):
        x[i] = x[i] + 0.5 * pa.dt * v[i]
    return x

#FUNCION PARA LA VELOCIDAD DE PLASMA FRIO
def buildgrid_vel():
    #velocidades
    #plasma frio
    v_0 = np.zeros(pa.noParticulas)
    v_0[1:pa.noParticulas] = 0.0

    return v_0

#FUNCION PARA LA VELOCIDAD DE LA INESTABILIDAD TWO STREAM PLASMA
def buildgrid_vel_2bp(x):
    velocidad = np.zeros(pa.noParticulas)
    for i in range(pa.noParticulas):
        #Construccion de la distribucion Maxwelliana de la velocidad
        f_velmax = 0.5*(1 + sp.exp(-2*(pa.vh**2)))
        vmin = -5 * pa.vh
        vmax = 5*pa.vh
        v_temp = vmin + (vmax-vmin)*(sp.random.random())
        f_vel = 0.5 * (sp.exp(-(v_temp - pa.vh)*(v_temp - pa.vh) / 2.0) + sp.exp(-(v_temp + pa.vh)*
        x_temp = f_velmax*(sp.random.random())
        while x_temp >f_vel:
            f_velmax = 0.5*(1 + sp.exp(-2*pa.vh*pa.vh))
            vmin = (-5) * pa.vh
            vmax = 5*pa.vh
            v_temp = vmin + (vmax-vmin)*(sp.random.random())
            f_vel = 0.5 * (sp.exp(-(v_temp - pa.vh)*(v_temp - pa.vh) / 2.0) + sp.exp(-(v_temp + pa.
            x_temp = f_velmax*(sp.random.random())
            va_temp = v_temp + pa.v0*np.cos(2*np.pi*x[i]/pa.malla_longitud)
        velocidad[i] = va_temp
    return velocidad
#FUNCION PARA LA VELOCIDAD DE LA INESTABLIDAD BEAM PLASMA
def buildgrid_vel_ibp (x):
    velocidad = []
    v_m = pa.vh
    n_l = pa.noParticulas * 0.9 #90% de las particulas
    n_m = pa.noParticulas * 0.1 #10% de las particulas
    v_s = (v_m * n_m)/(n_l-n_m)

    for i in range (pa.noParticulas):
        f_velmax =0.5*(1 + sp.exp(-2.0*(v_m**2)))
        vmin = -2*v_m
        vmax = 2*v_m
        v_temp = vmin + (vmax - vmin)*(np.random.random())
        f =  (1-(n_m/n_l))*np.exp(-(v_temp - v_s)*(v_temp - v_s)/1.0) + (n_m/n_l)*np.exp(-(v_temp -
        x_temp = f_velmax*(np.random.random())
        while x_temp > f:
            f_velmax =0.5*(1 + sp.exp(-2.0*(v_m**2)))
            vmin = -2*v_m
            vmax = 2*v_m
            v_temp = vmin + (vmax - vmin)*(np.random.random())
            f =  (1-(n_m/n_l))*np.exp(-(v_temp - v_s)*(v_temp - v_s)/1.0) + (n_m/n_l)*np.exp(-(v_ten
            x_temp = f_velmax*(np.random.random())
        velocidad.append(v_temp+ pa.v0*np.cos(2*np.pi*x[i]/pa.malla_longitud) )
    return velocidad

#FUNCION DEL CAMPO ELECTRICO
def electricfield(rho0):
    rho_neto = 1.0 + rho0

    integrante = pa.dx * sp.arange(pa.noMalla + 1)
    Ex = integrate.cumtrapz(rho_neto, integrante, initial=integrante[0])
    E_i = sp.sum(Ex)
```

```
    return Ex


#FUNCION QUE ACUTUALIZA VELOCIDADES
def chargevelocity(x,v, E_malla):

    for k in range (pa.noParticulas):
        xa = x[k]/pa.dx
        j1 = int(xa)
        j2 = j1 + 1
        f2 = xa - j1
        f1 = 1.0 - f2
        ex = f1*E_malla[j1] + f2*E_malla[j2]
        v[k] = v[k] + pa.carga_e*pa.dt*ex
    return v


#FUNCION QUE ACUTALIZA POSICIONES
def chargeposition(v_med, x):
#Condicion necesaria para el metodo de integracion Leap-Frog
    for i in range(pa.noParticulas):
        x[i] = x[i] +  v_med[i] * pa.dt
    return x

#FUNCION QUE VERIFICA LAS CONDICIONES DE FRONTERA
def cf(x_cf):
    for i in range(pa.noParticulas):
        if x_cf[i] < pa.plasma_inicio:
            while x_cf[i] < pa.plasma_inicio:
                x_cf[i] += pa.plasma_final
        elif x_cf[i] > pa.plasma_final:
            while x_cf[i] > pa.plasma_final:
                x_cf[i] -= pa.plasma_final
    return x_cf


#FUNCION QUE CALCULA LA DENSIDAD DE CARGA
def chargedensity(x):
    j1=np.dtype(np.int32)
    j2=np.dtype(np.int32)
    charge_density = np.zeros(pa.noMalla + 1)
    qe = -pa.rho0*((pa.plasma_final-pa.plasma_inicio)/pa.noParticulas)
    re = (qe/pa.dx)
    for k in range (pa.noParticulas):
        xa = x[k]/pa.dx
        j1 = int(xa)
        j2 = j1 + 1
        f2 = xa - j1
        f1 = 1.0 - f2
        charge_density[j1] = charge_density[j1] + re*f1
        charge_density[j2] = charge_density[j2] + re*f2

    charge_density[0] += charge_density[pa.noMalla]
    charge_density[pa.noMalla] = charge_density[0]


    return charge_density
#ENERGIA CINETICA
def Kenergy(v,step):
    v2 = [x**2 for x in v]
    pa.ki[step] =  0.5*pa.m*sum(v2)
    return pa.ki
#ENERGIA POTENCIAL
def Uenergy (Ex,step):
    e2 = [x**2 for x in Ex]
    pa.upot[step] = 0.5*pa.dx*sum(e2)
    return pa.upot
```

```
#ENERGIA TOTAL
def totalenergy (k,u):
    for i in range(pa.time_step):
        pa.totalenergy[i] = k[i] + u[i]
    return pa.totalenergy

#DRIFT DE ENERGIA
def drift(v,step):
    vdrift = sum(v)/pa.noParticulas
    pa.kdrift[step] = 0.5*pa.m*(vdrift**(2))*pa.noParticulas
    return pa.kdrift
```

Listing 10.3: Código para oscilaciones de plasma frío: `plasmafrio.py`

```
import funciones as f
import parametros as pa
import numpy as np
import matplotlib.pyplot as plt

#Ciclo inicial. Este solo se hace una vez
step = 0
xgrid =pa.dx*np.arange(pa.noMalla + 1) #necesario para graficar
posicion1 = f.buildgrid_pos(pa.x_inicial) #Solo se hace una vez
velocidad1 = f.buildgrid_vel() #Solo se hace una vez
posicion1 = f.leapfrog(posicion1,velocidad1)
posicion1 = f.cf(posicion1)
densidad1 = f.chargedensity(posicion1)
E1 = f.electricfield(densidad1)
K1 = f.Kenergy(velocidad1,0)
U1 = f.Uenergy(E1,0)
drift1 = f.drift(velocidad1,0)
T1 = f.totalenergy(K1,U1)



E_acumulado = np.empty((pa.noMalla + 1, pa.time_step + 1))
t = 0
step = 0
p = 0
e = 1
posiciones= [posicion1]
velocidades = [velocidad1]
densidades = [densidad1]
camposE = [E1]
#camposE_particulas = [E_particulas_inicial]
tiempo = [0]
energiacinetica = []
energiapotencial = []
energiatotal = []
DRIFT = []
#densidad de carga
#densidad de carga
path_p = "C:/Users/HP/Documents/graficas_tesis/plasmafrio_resultados/densidad/"
#campo electrico
path_e = "C:/Users/HP/Documents/graficas_tesis/plasmafrio_resultados/campo_electrico/"
#diagrama de fase
path_f = "C:/Users/HP/Documents/graficas_tesis/plasmafrio_resultados/diagrama_de_fase/"
#energias
path_k = "C:/Users/HP/Documents/graficas_tesis/plasmafrio_resultados/energia/"

plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
while  t < pa.time_step*pa.dt:
    if t == 0:
        posicion = posicion1
        posicion = f.cf(posicion)
        velocidad = velocidad1
```

```
        densidad = densidad1
        E = E1
        K = K1
        drift = drift1
        U = U1
        T = T1
        for q in range(1,pa.noMalla + 1):
            E_acumulado[q:step] = E[q]
        print (E)
        print (E_acumulado)
        """
        plt.scatter(posicion, velocidad)
        plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-pa.x0,pa.x0)
        plt.xlabel("x")
        plt.ylabel("v")
        plt.title("Plasma frio:diagrama de fase" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_f + "plasmafrioDF" + str(step) + ".png")
        plt.clf()
        plt.scatter(xgrid, E)
        plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-pa.x0,pa.x0)
        plt.xlabel("posicion malla")
        plt.ylabel("Ex")
        plt.title("Plasma frio: campo electrico" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_e + "plasmafrioE" + str(step) + ".png")
        plt.clf()
        """
        plt.scatter(xgrid, (densidad + 1))
        plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-pa.x0,pa.x0)
        plt.xlabel("posicion malla")
        plt.ylabel("densidad")
        plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
        plt.title("Plasma frio: densidad de carga" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_p + "plasmafrioDC" + str(step) + ".png")
        plt.clf()


    elif t > 0:
        posicion = f.chargeposition( velocidad,posicion)
        posicion = f.cf(posicion)
        velocidad = f.chargevelocity(posicion,velocidad, E)
        densidad = f.chargedensity(posicion)
        E = f.electricfield(densidad)
        for q in range(1,pa.noMalla + 1):
            E_acumulado[q:step] = E[q]
        K = f.Kenergy(velocidad,step)
        U = f.Uenergy(E,step)

        posiciones = np.append(posiciones, posicion)
        velocidades = np.append(velocidades,velocidad)
        densidades = np.append(densidades, densidad)
        camposE = np.append(camposE, E)
        energiacinetica = K
        energiapotencial = U
        DRIFT = drift
        """
        plt.scatter(posicion, velocidad)
        plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-pa.x0,pa.x0)
        plt.xlabel("x")
        plt.ylabel("v")
        plt.title("Plasma frio:diagrama de fase" + "  tiempo =" + str(round(t,2)))
```

```
        plt.savefig(path_f + "plasmafrioDF" + str(step) + ".png")
        plt.clf()
        plt.scatter(xgrid, E)
        plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-pa.x0,pa.x0)
        plt.xlabel("posicion malla")
        plt.ylabel("Ex")
        plt.title("Plasma frio: campo electrico" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_e + "plasmafrioE" + str(step) + ".png")
        plt.clf()
        """
        plt.scatter(xgrid, (densidad + 1))
        plt.ticklabel_format(axis="y", style="sci", scilimits=(0,0))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-pa.x0,pa.x0)
        plt.xlabel("posicion malla")
        plt.ylabel("densidad")
        plt.title("Plasma frio: densidad de carga" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_p + "plasmafrioDC" + str(step) + ".png")
        plt.clf()


    t = t + pa.dt
    step = step + 1
    tiempo.append(t)
for  u in range (len(energiacinetica)):
    energiatotal.append(energiacinetica[u] + energiapotencial[u])

#energias
"""
plt.plot(tiempo,energiacinetica, 'r', label = "K")
plt.plot(tiempo, energiapotencial, label = "U")
plt.plot(tiempo, energiatotal,  label = "T")
plt.plot(tiempo, DRIFT, label = " Drift")
plt.xlabel("Tiempo")
plt.ylabel("Energia")
plt.xlim(0,pa.malla_longitud)
plt.legend()
plt.savefig(path_k + "plasmafrioDC" + str(step) + ".png")
plt.show()
"""
```

Listing 10.4: Código para inestabilidad Two-stream: `2streamplasma.py`

```
Ciclo inicial. Este solo se hace una vez
step = 0
xgrid =pa.dx*np.arange(pa.noMalla + 1) #necesario para graficar
posicion1 = f.buildgrid_pos(pa.x_inicial) #Solo se hace una vez
velocidad1 = f.buildgrid_vel_2bp(posicion1) #Solo se hace una vez
posicion1 = f.leapfrog(posicion1,velocidad1)
posicion1 = f.cf(posicion1)
densidad1 = f.chargedensity(posicion1)
E1 = f.electricfield(densidad1)
K1 = f.Kenergy(velocidad1,0)
U1 = f.Uenergy(E1,0)
T1 = f.totalenergy(K1,U1)

#densidad de carga
path_p = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/densidad_ts/"
#campo electrico
path_e = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/campo_electrico_ts/"
#diagrama de fase
path_f = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/diagrama_de_fase_ts/"
#energias
path_k = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/energia_ts/"
```

```
graf = (np.pi/pa.dt/16)
t = 0
p = 0
e = 1
posiciones= [posicion1]
velocidades = [velocidad1]
densidades = [densidad1]
camposE = [E1]
tiempo = []
energiacinetica = []
energiapotencial = []
energiatotal = []
vmas = []
vmenos = []
xmas = []
xmenos = []



while  t < pa.time_step*pa.dt:
    if t == 0:
        posicion = posicion1
        posicion = f.cf(posicion)
        velocidad = velocidad1
        densidad = densidad1
        E = E1
        K = K1
        U = U1
        T = T1
        """
        plt.scatter(posicion, velocidad)
        plt.xlim(0,pa.malla_longitud)
        plt.xlabel("x")
        plt.ylabel("v")
        plt.title("Inestabilidad Two-stream:diagrama de fase" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_f + "twostreamDF" + str(step) + ".png")
        plt.clf()
        plt.scatter(xgrid, E)
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-8,8)
        plt.xlabel("x")
        plt.ylabel("Ex")
        plt.title("Inestabilidad Two-stream: campo electrico" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_e + "twostreamE" + str(step) + ".png")
        plt.clf()
        plt.scatter(xgrid, (densidad + 1))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-8,8)
        plt.xlabel("x")
        plt.ylabel("densidad de carga")
        plt.title("Inestabilidad Two-stream: densidad de carga" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_p + "twostreamDC" + str(step) + ".png")
        plt.clf()
        """

    elif t > 0:
        posicion = f.chargeposition( velocidad,posicion)
        posicion = f.cf(posicion)
        velocidad = f.chargevelocity(posicion,velocidad, E)
        densidad = f.chargedensity(posicion)
        E = f.electricfield(densidad)
        K = f.Kenergy(velocidad,step)
        U = f.Uenergy(E,step)
        posiciones = np.append(posiciones, posicion)
        velocidades = np.append(velocidades,velocidad)
        densidades = np.append(densidades, densidad)
        camposE = np.append(camposE, E)
```

```
            energiacinetica = K
            energiapotencial = U
            """
            plt.scatter(posicion, velocidad)
            plt.xlim(0,pa.malla_longitud)
            plt.xlabel("x")
            plt.ylabel("v")
            plt.title("Inestabilidad Two-stream:diagrama de fase" + "  tiempo =" + str(round(t,2)))
            plt.savefig(path_f + "twostreamDF" + str(step) + ".png")
            plt.clf()
            plt.scatter(xgrid, E)
            plt.xlim(0,pa.malla_longitud)
            plt.ylim(-8,8)
            plt.xlabel("x")
            plt.ylabel("Ex")
            plt.title("Inestabilidad Two-stream: campo electrico" + "  tiempo =" + str(round(t,2)))
            plt.savefig(path_e + "twostreamE" + str(step) + ".png")
            plt.clf()
            plt.scatter(xgrid, (densidad + 1))
            plt.xlim(0,pa.malla_longitud)
            plt.ylim(-8,8)
            plt.xlabel("x")
            plt.ylabel("densidad de carga")
            plt.title("Inestabilidad Two-stream: densidad de carga" + "  tiempo =" + str(round(t,2)))
            plt.savefig(path_p + "twostreamDC" + str(step) + ".png")
            plt.clf()
            """

        t = t + pa.dt
        step = step + 1
        tiempo.append(t)
    for  u in range (len(energiacinetica)):
        energiatotal.append(energiacinetica[u] + energiapotencial[u])

    #energias
    plt.plot(tiempo,energiacinetica, 'r', label = "K")
    plt.plot(tiempo, energiapotencial, label = "U")
    plt.plot(tiempo, energiatotal,  label = "T")
    plt.xlabel("Tiempo")
    plt.ylabel("Energia")
    #plt.xlim(0,pa.malla_longitud)
    plt.legend()
    plt.savefig(path_k + "energiastwostreamDCAHORA" + str(step) + ".png")
    plt.clf()
    plt.plot(tiempo, energiapotencial, label = "U")
    plt.xlabel("Tiempo")
    plt.ylabel("Energia")
    #plt.xlim(0,pa.malla_longitud)
    plt.legend()
    plt.savefig(path_k + "potencialTwostreamDCAHORA" + str(step) + ".png")

    print ("energia_K_media")
    print (sum(energiacinetica)/len(energiacinetica))
```

Listing 10.5: Código para inestabilidad Beam-plasma: `beamplasma.py`

```
import funciones as f
import parametros as pa
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt


#Ciclo inicial. Este solo se hace una vez
step = 0
xgrid =pa.dx*np.arange(pa.noMalla + 1) #necesario para graficar
```

```
posicion1 = f.buildgrid_pos(pa.x_inicial) #Solo se hace una vez
velocidad1 = f.buildgrid_vel_ibp(posicion1) #Solo se hace una vez
posicion1 = f.leapfrog(posicion1,velocidad1)
posicion1 = f.cf(posicion1)
densidad1 = f.chargedensity(posicion1)
E1 = f.electricfield(densidad1)
K1 = f.Kenergy(velocidad1,0)
U1 = f.Uenergy(E1,0)
T1 = f.totalenergy(K1,U1)

#densidad de carga
path_p = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/densidad_bp/"
#campo electrico
path_e = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/campo_electrico_bp/"
#diagrama de fase
path_f = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/diagrama_de_fase_bp/"
#energias
path_k = "C:/Users/HP/Documents/graficas_tesis/landau_resultados/energia_bp/"
graf = (np.pi/pa.dt/16)
t = 0
p = 0
e = 1
posiciones= [posicion1]
velocidades = [velocidad1]
densidades = [densidad1]
camposE = [E1]
tiempo = []
energiacinetica = []
energiapotencial = []
energiatotal = []
vmas = []
vmenos = []
xmas = []
xmenos = []



while  t < pa.time_step*pa.dt:
    if t == 0:
        posicion = posicion1
        posicion = f.cf(posicion)
        velocidad = velocidad1
        densidad = densidad1
        E = E1
        K = K1
        U = U1
        T = T1
        """
        plt.scatter(posicion, velocidad)
        plt.xlim(0,pa.malla_longitud)
        plt.xlabel("x")
        plt.ylabel("v")
        plt.title("Inestabilidad Beam-plasma:diagrama de fase" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_f + "landauDF" + str(step) + ".png")
        plt.clf()
        plt.scatter(xgrid, E)
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-2,2)
        plt.xlabel("x")
        plt.ylabel("Ex")
        plt.title("Inestabilidad Beam-plasma: campo electrico" + "  tiempo =" + str(round(t,2)))
        plt.savefig(path_e + "landauE" + str(step) + ".png")
        plt.clf()
        plt.scatter(xgrid, (densidad + 1))
        plt.xlim(0,pa.malla_longitud)
        plt.ylim(-2,2)
        plt.xlabel("x")
        plt.ylabel("densidad de carga")
```

```
            plt.title("Inestabilidad Beam-plasma: densidad de carga" + "  tiempo =" + str(round(t,2)))
            plt.savefig(path_p + "landauDC" + str(step) + ".png")
            plt.clf()
            """
        elif t > 0:
            posicion = f.chargeposition( velocidad,posicion)
            posicion = f.cf(posicion)
            velocidad = f.chargevelocity(posicion,velocidad, E)
            densidad = f.chargedensity(posicion)
            E = f.electricfield(densidad)
            K = f.Kenergy(velocidad,step)
            U = f.Uenergy(E,step)
            posiciones = np.append(posiciones, posicion)
            velocidades = np.append(velocidades,velocidad)
            densidades = np.append(densidades, densidad)
            camposE = np.append(camposE, E)
            energiacinetica = K
            energiapotencial = U
            """
            plt.scatter(posicion, velocidad)
            plt.xlim(0,pa.malla_longitud)
            plt.xlabel("x")
            plt.ylabel("v")
            plt.title("Inestabilidad Beam-plasma:diagrama de fase" + "  tiempo =" + str(round(t,2)))
            plt.savefig(path_f + "landauDF" + str(step) + ".png")
            plt.clf()
            plt.scatter(xgrid, E)
            plt.xlim(0,pa.malla_longitud)
            plt.ylim(-2,2)
            plt.xlabel("x")
            plt.ylabel("Ex")
            plt.title("Inestabilidad Beam-plasma: campo electrico" + "  tiempo =" + str(round(t,2)))
            plt.savefig(path_e + "landauE" + str(step) + ".png")
            plt.clf()
            plt.scatter(xgrid, (densidad + 1))
            plt.xlim(0,pa.malla_longitud)
            plt.ylim(-2,2)
            plt.xlabel("x")
            plt.ylabel("densidad de carga")
            plt.title("Inestabilidad Beam-plasma: densidad de carga" + "  tiempo =" + str(round(t,2)))
            plt.savefig(path_p + "landauDC" + str(step) + ".png")
            plt.clf()
            """

        t = t + pa.dt
        step = step + 1
        tiempo.append(t)
    for  u in range (len(energiacinetica)):
        energiatotal.append(energiacinetica[u] + energiapotencial[u])


    plt.plot(tiempo,energiacinetica, 'r', label = "K")
    plt.plot(tiempo, energiapotencial, label = "U")
    plt.plot(tiempo, energiatotal,  label = "T")
    plt.xlabel("Tiempo")
    plt.ylabel("Energia")
    plt.xlim(0,pa.malla_longitud)
    plt.legend()
    plt.savefig(path_k + "energiaslandauDC" + str(step) + ".png")
    plt.clf()
    plt.plot(tiempo, energiapotencial, label = "U")
    plt.xlabel("Tiempo")
    plt.ylabel("Energia")
    #plt.xlim(0,pa.malla_longitud)
    plt.legend()
    plt.savefig(path_k + "potenciallandauDC" + str(step) + ".png")

    print ("energia K media")
```

```
print (sum(energiacinetica)/len(energiacinetica))
```