Lecture 1

Course overview and introduction

D. Dickinson
d.dickinson@york.ac.uk

York Plasma Institute, School of PET, University of York

Semester 1

Overview of computational techniques component

This part of Computational Plasma Physics (CPP) provides a brief overview of concepts relating to numerical methods for solving differential equations. The concepts are not specific to a single language but we will be using Python for the practicals.

Topics to be covered include:

Cover mathematical concepts behind these.
Use modules for completing the maths rather than writing it all ourselves.

- Integrating ODEs
  - Explicit techniques
  - Implicit techniques
  - Using Scipy to integrate ODEs
- Spatial discretisation
  - Finite differencing — Approximate derivatives
  - Spectral methods — Alternative approach to finite differencing.
  - Finite elements
- Particle In Cell (PIC) approaches
- Continuum techniques

- Do reading on PIC

Putting above 2 together to solve more complex problems.

Notes available on the VLE.

Assessment

There are two pieces of assessed work associated with CPP and you should refer to the VLE for more details as and when these are released.

Comptutational techniques: Lectures and practicals

- Portofolio submission (50%) : Small number of programming related assignments issued and submitted through the VLE. Will test ability to use Python to solve practical problems of relevance to plasma physics and connection to related work.

Computational laboratory: Practical application

- Lab book (50%) : Ongoing numerical experiment using a PIC code recorded and assessed through lab book.

  Friday week 11

Overview this lecture

This lecture will look at

- Common plasma models.
- An introduction to different computational techniques
- Some useful definitions

Plasma models : Lorentz+Maxwell

We know the equation of motion for a charged particle from the Lorentz force law:

$$m\frac{d^2\boldsymbol{x}}{dt^2} = q\left[\boldsymbol{E}\left(\boldsymbol{x}\right) + \boldsymbol{v}\times\boldsymbol{B}\left(\boldsymbol{x}\right)\right]$$

with $\boldsymbol{E}$ and $\boldsymbol{B}$ given by Maxwell's equations:

$$\nabla\cdot\boldsymbol{E} = \rho/\epsilon_0 \qquad \nabla\times\boldsymbol{E} = -\frac{\partial\boldsymbol{B}}{\partial t}$$

$$\nabla\times\boldsymbol{B} = \mu_0\boldsymbol{J} + \mu_0\epsilon_0\frac{\partial\boldsymbol{E}}{\partial t} \qquad \nabla\cdot\boldsymbol{B} = 0$$

with ρ the charge density and $\boldsymbol{J}$ the current, which both depend on the particle positions and velocities.

Plasma models : Lorentz+Maxwell

We know the equation of motion for a charged particle from the Lorentz force law:

$$m\frac{d^2\boldsymbol{x}}{dt^2} = q\left[\boldsymbol{E}\left(\boldsymbol{x}\right) + \boldsymbol{v}\times\boldsymbol{B}\left(\boldsymbol{x}\right)\right]$$

with $\boldsymbol{E}$ and $\boldsymbol{B}$ given by Maxwell's equations:

$$\nabla\cdot\boldsymbol{E} = \rho/\epsilon_0 \qquad \nabla\times\boldsymbol{E} = -\frac{\partial\boldsymbol{B}}{\partial t}$$

$$\nabla\times\boldsymbol{B} = \mu_0\boldsymbol{J} + \mu_0\epsilon_0\frac{\partial\boldsymbol{E}}{\partial t} \qquad \nabla\cdot\boldsymbol{B} = 0$$

with ρ the charge density and $\boldsymbol{J}$ the current, which both depend on the particle positions and velocities.

In principle this set of equations completely describes the plasma evolution.

Plasma models : Lorentz+Maxwell

We know the equation of motion for a charged particle from the Lorentz force law:

$$m\frac{d^2\boldsymbol{x}}{dt^2} = q\left[\boldsymbol{E}\left(\boldsymbol{x}\right) + \boldsymbol{v}\times\boldsymbol{B}\left(\boldsymbol{x}\right)\right]$$

with $\boldsymbol{E}$ and $\boldsymbol{B}$ given by Maxwell's equations:

$$\nabla\cdot\boldsymbol{E} = \rho/\epsilon_0 \qquad \nabla\times\boldsymbol{E} = -\frac{\partial\boldsymbol{B}}{\partial t}$$

$$\nabla\times\boldsymbol{B} = \mu_0\boldsymbol{J} + \mu_0\epsilon_0\frac{\partial\boldsymbol{E}}{\partial t} \qquad \nabla\cdot\boldsymbol{B} = 0$$

with ρ the charge density and $\boldsymbol{J}$ the current, which both depend on the particle positions and velocities.

In principle this set of equations completely describes the plasma evolution. $\longrightarrow$ In practice this is impractical![a]

[a] Consider a tokamak with density $\sim 10^{19}\mathrm{m}^{-3}$ and volume $\sim 100m^3$ $\longrightarrow$ need to describe $\sim 10^{21}$ particles, if each requires 1 byte of memory need 10^3 Exabytes ($\sim$ the current total yearly internet traffic).

Too many particles in a plasma to calculate Maxwell's equ's for each

Plasma models : Vlasov

Rather than considering 10^{21} separate particles is there some way we can simplify the book keeping?
$\longrightarrow$ Introduce the distribution function, $f(\boldsymbol{z}, t)$, which describes the number density of particles at a given point in the 6D phase space, $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{v})$.

Plasma models : Vlasov

Rather than considering 10^{21} separate particles is there some way we can simplify the book keeping?
$\longrightarrow$ Introduce the distribution function, $f(\boldsymbol{z}, t)$, which describes the number density of particles at a given point in the 6D phase space, $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{v})$.

Consider the case with a single particle, $f_i(\boldsymbol{z}, t) = \delta(\boldsymbol{z} - \boldsymbol{z}_i[t])$, where $\boldsymbol{z}_i[t]$ is the instantaneous particle position (in phase space) and f_i is the single particle distribution function. Clearly f_i must be constant along the particle trajectory:

$$\frac{\partial f_i}{\partial t} + \frac{dz_j}{dt}\frac{\partial f_i}{\partial z_j} = 0 \tag{1}$$

where the Einstein summation convention is assumed.

Plasma models : Vlasov

Rather than considering 10^{21} separate particles is there some way we can simplify the book keeping?
$\longrightarrow$ Introduce the distribution function, $f(\boldsymbol{z}, t)$, which describes the number density of particles at a given point in the 6D phase space, $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{v})$. Function of phase space

Consider the case with a single particle, $f_i(\boldsymbol{z}, t) = \delta(\boldsymbol{z} - \boldsymbol{z}_i[t])$, where $\boldsymbol{z}_i[t]$ is the instantaneous particle position (in phase space) and f_i is the single particle distribution function. Clearly f_i must be constant along the particle trajectory:

$$\frac{\partial f_i}{\partial t} + \frac{dz_j}{dt}\frac{\partial f_i}{\partial z_j} = 0 \qquad (1)$$

Convective derivative expanded

where the Einstein summation convention is assumed. How a single particle distribution function evolves if we know its position.
-> Sum is all particles, total distribution function

We can note that eqn. 1 holds for each particle, so we define the total distribution function as $f = \sum_i f_i$ and note that eqn. 1 describes its evolution.
However, f is now just a sum of delta functions, we've not really gained anything over the particle approach other than a simplified notation.

Plasma models : Vlasov

We can make progress however by noting that we're interested in the macrostate, and not the microstate. This means we're really interested in the ensemble average of f, $\langle f \rangle$, where we've defined

$$\langle f \rangle = \frac{1}{\Delta z} \int_{\Delta z} f \boldsymbol{dz}$$

with Δz some small phase space volume containing a significant number of particles. Here $\langle f \rangle$ will be a smooth function.

Ensemble averaging eqn. 1 and replace $\boldsymbol{z}$ with $\boldsymbol{x}$ and $\boldsymbol{v}$ leads to

$$\frac{\partial \langle f \rangle}{\partial t} + \boldsymbol{v} \cdot \frac{\partial \langle f \rangle}{\partial \boldsymbol{x}} + \left\langle \boldsymbol{a} \cdot \frac{\partial f}{\partial \boldsymbol{v}} \right\rangle = 0 \tag{2}$$

How our ensemble average changes through time

we can write the last term as

$$\left\langle \boldsymbol{a} \cdot \frac{\partial f}{\partial \boldsymbol{v}} \right\rangle = \langle \boldsymbol{a} \rangle \cdot \frac{\partial \langle f \rangle}{\partial \boldsymbol{v}} - C\left(\langle f \rangle\right)$$

Correction

where $C\left(\langle f \rangle\right)$ is known as the collision operator and accounts for individual particle interactions.

Plasma models : Vlasov

Dropping the ensemble average notation and substituting for $\boldsymbol{a}$ from the Lorentz equation we arrive at

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \frac{\partial f}{\partial \boldsymbol{x}} + \frac{q}{m} \left[\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B} \right] \cdot \frac{\partial f}{\partial \boldsymbol{v}} = C\left(f\right) \tag{3}$$

In the absence of collisions eqn. 3 is known as the Vlasov equation[1].

Describes most of the physics accurately

[1] eqn. 3 is often referred to by many different names including Vlasov, Fokker-Planck, Kinetic, Boltzmann etc.

Plasma models : Vlasov

Dropping the ensemble average notation and substituting for $\boldsymbol{a}$ from the Lorentz equation we arrive at

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \frac{\partial f}{\partial \boldsymbol{x}} + \frac{q}{m} \left[\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B} \right] \cdot \frac{\partial f}{\partial \boldsymbol{v}} = C\left(f\right) \tag{3}$$

In the absence of collisions eqn. 3 is known as the Vlasov equation[1].

Despite all the effort put into obtaining eqn. 3 this hasn't really improved the feasibility of the problem, it's still 6D+time and involves a wide range of spatial and temporal scales. So why did we do it?

[1] eqn. 3 is often referred to by many different names including Vlasov, Fokker-Planck, Kinetic, Boltzmann etc.

Plasma models : Vlasov

Dropping the ensemble average notation and substituting for $\boldsymbol{a}$ from the Lorentz equation we arrive at

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \frac{\partial f}{\partial \boldsymbol{x}} + \frac{q}{m} \left[\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B} \right] \cdot \frac{\partial f}{\partial \boldsymbol{v}} = C\left(f\right) \tag{3}$$

In the absence of collisions eqn. 3 is known as the Vlasov equation[1].

Despite all the effort put into obtaining eqn. 3 this hasn't really improved the feasibility of the problem, it's still 6D+time and involves a wide range of spatial and temporal scales. So why did we do it? $\longrightarrow$ It provides a good starting point for deriving a range of more tractable models.

Can simplify e.g. by not considering areas of plasma we don't need for current problem

[1] eqn. 3 is often referred to by many different names including Vlasov, Fokker-Planck, Kinetic, Boltzmann etc.

Plasma models : Gyrokinetics

Consider the motion of a single particle in the presence of a magnetic field:

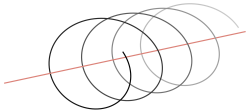

Figure: Cartoon of particle making helical orbit.

- Parallel Motion : constant
- Perpendicular motion : particle gyrates around the field.

Super fast. Nothing else changes over the orbits. Don't care about really fast motion so get rid of it by averaging it out. Think of it like a charged ring. Removes one velocity dimension.

Cyclotron/gyro-frequency is typically $\sim 10^7$ Hz $\gg \omega$, where ω represents the frequencies of interest. As such we can average over this gyro-motion. Somewhat like smearing the particle out into a ring.

This leads to the gyrokinetic equation. By removing this motion we have reduced the problem to 5D+time (3D space+2D velocity).

Plasma models : Gyrokinetics

Consider the motion of a single particle in the presence of a magnetic field:

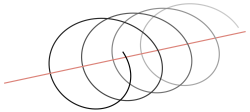

Figure: Cartoon of particle making helical orbit.

- Parallel Motion : constant
- Perpendicular motion : particle gyrates around the field.

Cyclotron/gyro-frequency is typically $\sim 10^7$ Hz $\gg \omega$, where ω represents the frequencies of interest. As such we can average over this gyro-motion. Somewhat like smearing the particle out into a ring.

This leads to the gyrokinetic equation. By removing this motion we have reduced the problem to 5D+time (3D space+2D velocity).

$\longrightarrow$ The problem is now tractable on supercomputers.

Useful if you care about plasma turbulence, but still pretty expensive.

Plasma models : Moment equations

Whilst it's possible to solve the gyrokinetic equation for realistic problems it takes significant computer resource. Need simpler models if we want to run on our desktop in a reasonable time.

Plasma models : Moment equations

Whilst it's possible to solve the gyrokinetic equation for realistic problems it takes significant computer resource. Need simpler models if we want to run on our desktop in a reasonable time.

Collisions cause the distribution function to tend to a Maxwellian in velocity. When f is close to Maxwellian we can parameterise the velocity dependence with the temperature, T. By taking velocity moments of the kinetic equation we integrate out the velocity dependence, leaving a 3D+time problem. For example, defining

$$n\left(\boldsymbol{x},t\right)=\int f\left(\boldsymbol{x},\boldsymbol{v},t\right)\boldsymbol{dv}$$

Bulk flow

$$U\left(\boldsymbol{x},t\right)=\int f\left(\boldsymbol{x},\boldsymbol{v},t\right)\boldsymbol{v}\boldsymbol{dv}$$

etc. we find that the velocity moments of the kinetic equation provide evolution equations for n, U, ...

Plasma models : Moment equations

In practice a lot of different moment based models can arise depending on assumptions made, closure approximations etc. Some common names are Braginskii and MHD (magnetohydrodynamics).

Moment equations, which are 3D+time, tend to be fairly tractable and able to run with modest computing resource.

Further approximations can be made which reduce the computational cost further.

If we take moments of the gyrokinetic equation instead of the kinetic one then we end up with a gyrofluid model.

Plasma models : Summary

A large range of plasma models exist of varying complexity. The computational resources required to treat each model can vary significantly.

Model	Spatial Dim	Velocity Dim	Computing resource
Kinetic	3	3	Supercomputer (simplified problems only)
Gyrokinetic	3	2	Supercomputer
Moment (fluid)	3	0	Desktop

Choosing the most appropriate model for the physics you're studying is a very important consideration.

Plasma models : Summary

A large range of plasma models exist of varying complexity. The computational resources required to treat each model can vary significantly.

Model	Spatial Dim	Velocity Dim	Computing resource
Kinetic	3	3	Supercomputer (simplified problems only)
Gyrokinetic	3	2	Supercomputer
Moment (fluid)	3	0	Desktop

Choosing the most appropriate model for the physics you're studying is a very important consideration.

The basic computational techniques relevant to each model are actually the same.

Computational techniques : Types of equation

Many of the equations we come across have conservative form:

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \nabla f = 0$$

No acceleration - "forceless"

Depends on velocity and how f varies in space

Consider a 1D system

Figure: Sketch of a 1D pulse.

Computational techniques : Types of equation

Many of the equations we come across have conservative form:

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \nabla f = 0$$

How things flow

Consider a 1D system

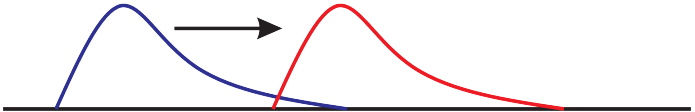

Figure: Sketch showing a 1D pulse which has moved without changing shape.

This is advection, pulse moves with velocity v_x. Such equations are often referred to as hyperbolic and can be related to a wave equation.

Computational techniques : Types of equation

Another common type of form is that of a diffusion equation:

$$\frac{\partial T}{\partial t} = D\nabla^2 T$$

Consider a 1D system

Figure: Sketch showing a peaked curved.

Computational techniques : Types of equation

Another common type of form is that of a diffusion equation: How things spread out

$$\frac{\partial T}{\partial t} = D\nabla^2 T$$

Consider a 1D system

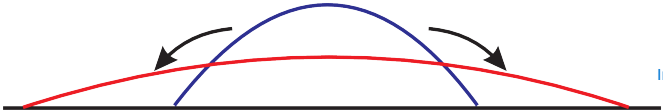

Figure: Sketch showing the peaked curve diffusing.

This is diffusion, the pulse spreads out with diffusion coefficient D. Such equations are often referred to as parabolic and can be related to the heat equation.

Computational techniques : Types of method

Two main approaches can be considered:

Particle Also known as Lagrangian, here we follow discrete "fluid packages" or markers. Consider splitting the domain into lots of small boxes of plasma with specified density, temperature etc. By following these boxes around we can find the plasma properties at a given location at a later time. Tends to suit advection.

Grid Also known as Eulerian, here we discretise space into a grid. Given initial values of density, temperature etc. on a grid point we evolve these parameters in time. Good for diffusion.

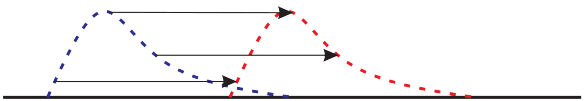

Figure: Sketch showing advection of packages

How do values at points of interest change

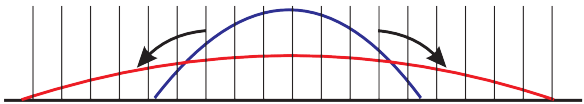

Figure: Sketch showing diffusion on a grid

Very roughly can think of particle approach as tracking the position of given values and grid approach as tracking values at given position.

Computational techniques : Types of method

A common approach is to actually combine both the particle and grid methods to give a particle in cell (PIC) method. Here we use particle methods for some equations/terms and a grid approach for others.

It is important to note that *both* particle and grid methods can be used for *both* kinetic and moment models.

Is a system primarily advection or diffusion, or a complicated mixture of both?

Computational techniques : Types of method

A common approach is to actually combine both the particle and grid methods to give a particle in cell (PIC) method. Here we use particle methods for some equations/terms and a grid approach for others.

It is important to note that *both* particle and grid methods can be used for *both* kinetic and moment models.

We'll come back to these ideas later in the lecture course.

Choosing a method

Several considerations to be made when selecting what method to use:

Efficiency Typically competition between accuracy and speed. Also have to consider how algorithm will parallelise across machines and how it scales with increasing processor count.

Stability If you introduce an error in one step of solution does it shrink, stay the same or grow when you go to the next step? If it grows then method is not stable and solution will quickly be dominated by junk.

Consistency As you reduce the grid spacing and time step of discretised problem the equation you're solving must reduce to the desired model equations.

Convergence As you reduce the grid spacing and time step of discretised problem does solution tend to exact solution of equations you're solving?

Choosing a method

Several considerations to be made when selecting what method to use:

Efficiency Speed vs. accuracy.

Stability Do errors in solution grow?

Consistency Does the numerical model represent correct eqn?

Convergence Does numerical solution represent exact solution?

Choosing a method

Several considerations to be made when selecting what method to use:

Efficiency Speed vs. accuracy.

Stability Do errors in solution grow?

Consistency Does the numerical model represent correct eqn?

Convergence Does numerical solution represent exact solution?

Difference between consistency and convergence somewhat subtle.

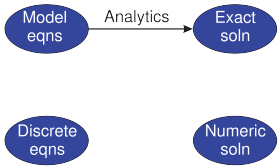

Figure: Picture showing connection between model equations and exact soltution.

- Have model equation, analytic tools could give exact solution.

Choosing a method

Several considerations to be made when selecting what method to use:

- **Efficiency** Speed vs. accuracy.
- **Stability** Do errors in solution grow?
- **Consistency** Does the numerical model represent correct eqn?
- **Convergence** Does numerical solution represent exact solution?

Difference between consistency and convergence somewhat subtle.

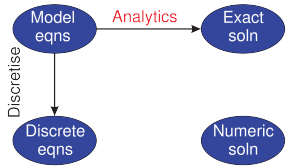

Figure: Picture showing connection between model equations and discrete equations

- Analytic too difficult, so discretise to get numerical model.

Choosing a method

Several considerations to be made when selecting what method to use:

Efficiency	Speed vs. accuracy.
Stability	Do errors in solution grow?
Consistency	Does the numerical model represent correct eqn?
Convergence	Does numerical solution represent exact solution?

Difference between consistency and convergence somewhat subtle.

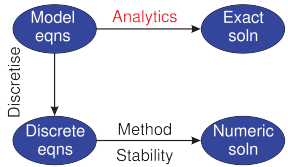

Figure: Picture showing connection between discrete equations and numerical solution

- Using stable method can solve discrete problem to get numerical solution.

Choosing a method

Several considerations to be made when selecting what method to use:

Efficiency	Speed vs. accuracy.
Stability	Do errors in solution grow?
Consistency	Does the numerical model represent correct eqn?
Convergence	Does numerical solution represent exact solution?

Difference between consistency and convergence somewhat subtle.

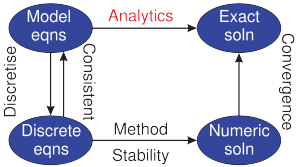

Figure: Picture showing connection between model equations and discrete equations via consistency and the connection between numerical solution and analytic one through convergence.

- If we reduce the grid spacing and time step to zero then the numerical solution should tend to the exact solution (convergence) and the discrete equations should become the model equation (consistent).

Choosing a method

Several considerations to be made when selecting what method to use:

Efficiency	Speed vs. accuracy.
Stability	Do errors in solution grow?
Consistency	Does the numerical model represent correct eqn?
Convergence	Does numerical solution represent exact solution?

Difference between consistency and convergence somewhat subtle.

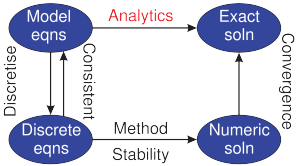

Figure: Picture showing connection between model equations and discrete equations via consistency and the connection between numerical solution and analytic one through convergence.

Lax's equivalence theorem

$\Rightarrow$ For a consistent numerical model then method stability is a necessary and sufficient condition for convergence.