Lecture 8

Particle in cell methods

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

York Plasma Institute, School of PET, University of York

Semester 1

Overview of course

This course provides a brief overview of concepts relating to numerical methods for solving differential equations.

Topics to be covered include:

- ~~Integrating ODEs~~
  - ~~Explicit techniques~~
  - ~~Implicit techniques~~
  - ~~Using Scipy to integrate ODEs~~
- ~~Spatial discretisation~~
  - ~~Finite differencing~~
  - ~~Spectral methods~~
  - ~~Finite elements~~
- Particle In Cell (PIC) approaches
- Continuum techniques

Notes available on the VLE.

Overview this lecture

This lecture will look at

- An introduction to particle in cell methods

Particle in cell methods

Evaluate derivatives with respect to space, initial value problems, etc

Now that we've looked at how we can integrate equations in time and solve discretised spatial problems we're in a position to combine these and solve more complicated systems of equations similar to the ones we looked at in the first lecture. not just simpler ones like steady state

Today we'll be looking at an approach known as particle in cell (PIC) in the context of a system of charged particles. PIC methods are quite common in plasma physics as they are relatively simple to implement and parallelise well. Unfortunately as we'll see, noise can be a problem in such simulations.

many particles

Particle in cell methods : Particle noise

In a real system we typically have $\sim 10^{20-21}$ particles. This isn't practical however, with a supercomputer we may be able to treat $\sim 10^{9-10}$ and on a standard laptop something like 10^4 particles is more practical.

describe only around 1-10%, and see if we learn anything

As such, we are effectively simulating a random subset of the real particles. This can have consequences for how well we can capture certain physics. Consider calculating the temperature at each location. We can plot the PDF as a function of velocity, first consider the case where we have a large number of particles,

particles at one location (one marker) will experience the same force
- see if we can track just that area (same chunk of phase space - marker)

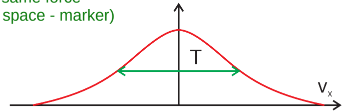

Figure: Sketch of a smooth Gaussian function with well defined temperature.

number of markers determined by how much memory we have available
- more memory, more markers, each with less particles in, better approximation

In a real system we typically have $\sim 10^{20-21}$ particles. This isn't practical however, with a supercomputer we may be able to treat $\sim 10^{9-10}$ and on a standard laptop something like 10^4 particles is more practical.

As such, we are effectively simulating a random subset of the real particles. This can have consequences for how well we can capture certain physics. Consider calculating the temperature at each location. We can plot the PDF as a function of velocity, first consider the case where we have a large number of particles, now if we only have a few particles.

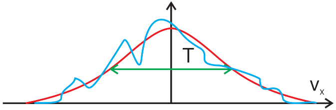

Figure: Sketch of a noisy Gaussian-like function without a well defined temperature.

When we don't have many particles we end up with a very noisy approximation of a smooth function. As we have a statistical sample the noise is Poisson like and signal to noise $\propto \sqrt{N}$, where N is particle count.

doubling markers doesn't reduce noise that much, so better to use loads of markers - need a supercomputer

Noise is one of the main concerns for PIC simulations.

Particle in cell methods : Charged particle problem

The problem of charged particle motion is one of the systems we looked at in the first SciPy practical lecture and we'll revisit it here to illustrate the PIC approach. The motion of a single charged particle is determined by the equations

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}, \quad \frac{d\boldsymbol{v}}{dt} = \frac{q}{m}\boldsymbol{E}$$

force per unit mass

where the electric field comes from Poisson's equation

$$\nabla^2\phi = -\frac{\rho}{\epsilon_0}$$

Can't just use these equations on there own - the markers can move about in space, doesn't live on a grid, is continuous (so need to model as continuous as possible).
Can't just use Coulomb's law, because the force on each marker (electric field) depends on those around it, so calculation would be too large.

Particle in cell methods

Now we can use the equations of motion

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}, \quad \frac{d\boldsymbol{v}}{dt} = \frac{q}{m}\boldsymbol{E}$$

with our time integrator (e.g. solve_ivp) to evolve the system in time.

To do this we need to calculate $\boldsymbol{E}$. A simple approach to this would be to use Coulomb's law to calculate the force on each particle from all the others, as we did in the solve_ivp demonstration lecture. Unfortunately this is quite inefficient; for each of the N particles there are $N-1$ other particles acting on it. So we have to do $N\left(N-1\right)$ calculations to find $\boldsymbol{E}$ for each particle. As we have to repeat this calculation each time we want to know the time derivatives[1] this approach is not feasible even for the case with 10^4 particles.

To solve this problem we'll adopt a PIC approach where we treat the charged particles as markers and solve for $\boldsymbol{E}$ on a discrete grid.

[1] Remember that some techniques like RK4 require multiple evaluations per time step.

Particle in cell methods : Domain

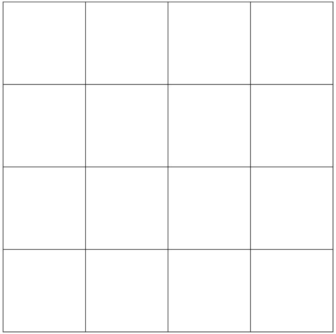

Figure: Sketch of an empty 2D grid.

To implement a PIC algorithm to simulate the charged particle system the first step is to split our domain into discrete cells,

Each square is a cell

Particle in cell methods : Domain

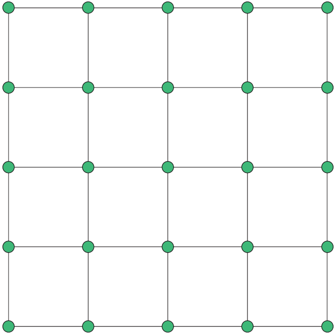

Figure: Sketch of a 2D grid with grid points highlighted to show where we solve for the field.

To implement a PIC algorithm to simulate the charged particle system the first step is to split our domain into discrete cells, we'll solve the fields on cell corners e.g. use Poisson's equation

Particle in cell methods : Domain

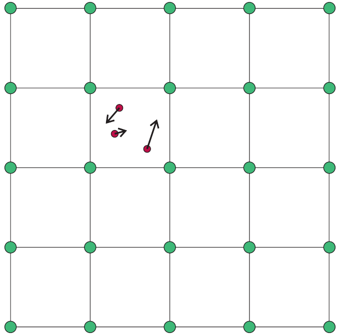

Figure: Sketch of a empty 2D grid with grid points highlighted and markers in between grids to represent particles moving in arbitrary directions.

To implement a PIC algorithm to simulate the charged particle system the first step is to split our domain into discrete cells, we'll solve the fields on cell corners and add particles with specific position and velocity within the cells.
markers

Particle in cell methods : Gather/Scatter approach

To avoid the problems inherent in the simple Coulomb approach to calculating $\boldsymbol{E}$ the PIC approach makes an approximation. We assume that the force on a given particle due to a bunch of particles in a distant cell can be represented by a single charge, which we place on a cell corner. This gives arise to what is referred to as the gather/scatter approach to calculating the field.

Because it doesn't really matter where it is, so choose easiest position.

This consists of three stages:

- Gather : Collect charges onto cell corners.
- Calculate : Solve Poisson's equation on grid.
- Scatter : Use field on grid to find field at particle position.

Calculate the strength that a marker on each corner would have

Find force on each marker, so need way to calculate electric field for each marker.

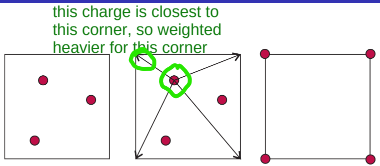

Figure: We can spread the charge from individual particles onto the enclosing cell corners.

markers

Gather Collect charges onto cell corners. For each particle work out distance to its four enclosing corners and contribute to the charge on these corners weighted by the distance. For example a singly charge particle at the exact centre of a cell will contribute 1/4 to each corner, whilst one practically on a corner will contribute most of it's charge to this corner and very little to the rest.

Accumulate charge on the corners

Particle in cell methods : Gather/Scatter approach

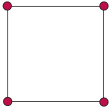

Figure: ρ on grid.

$$\nabla^2\phi = -\frac{\rho}{\epsilon_0}$$

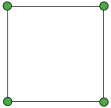

Figure: ϕ on grid.

Gather Collect charges onto cell corners.

Calculate Solve Poisson's equation on grid. We've seen previously how to solve $\epsilon_0\nabla^2\phi = -\rho$ and we simply need to apply these techniques here, where ρ comes from the charge collected on cell corners.

Particle in cell methods : Gather/Scatter approach

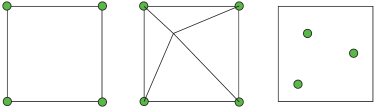

Figure: We can map ϕ on the grid to any location enclosed within the grid.

Gather Collect charges onto cell corners.

Calculate Solve Poisson's equation on grid.

Scatter Interpolate field from cell corners to particle position. Once we know the field at the cell corner we need to estimate its value at the particle location.

For the location that they were originally at, because we want to calculate the force due to the other markers in the cell, so choose one original location and calculate force from corner markers (do this for each original location of marker)

Particle in cell methods : Gather/Scatter approach

Gather Collect charges onto cell corners.

Calculate Solve Poisson's equation on grid.

Scatter Interpolate field from cell corners to particle position.

This approach is typically much more efficient than a direct Coulomb approach and scales much more efficiently.

E-field Coulomb $\sim N_p^2$

E-field grid $\sim N_g^2 \cdot N_g$ ← solution extra cost

↑ no. of grid points

$N_g << N_p$

For PIC we want $\frac{N_p}{N_g} >> 1$

With few markers, sensitive to markers moving between cells - want lots of markers per cell to minimise this

We can summarise the whole PIC algorithm using a simple flow chart.

- We use our time integration techniques to update $\boldsymbol{x}$ and $\boldsymbol{v}$.
- We use our two point boundary value techniques to calculate the field on the grid.

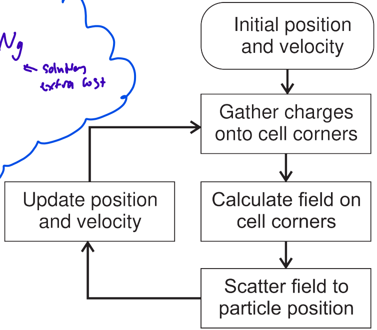

Figure: Flow chart representing PIC loop of gather, solve, scatter, push.