Particle In Cell Consulting LLC

2018 Plasma Simulation Courses

Registration is now open for the following online plasma simulation courses:

Fluid Modeling of Plasmas, March 13th to May 22nd, 2018

This new course will teach you how simulate dense plasmas in which the continuum assumption holds. We will cover single and multi-fluid MHD equations as well as hybrid approaches with detailed electron model and some advanced topics like Vlasov solvers. Syllabus:

- Lesson 1: Fluid formulation and numerical methods
- Lesson 2: Single fluid equation
- Lesson 3: Multiple species MHD
- Lesson 4: Hybrid PIC with detailed electron model
- Lesson 5: SPH and Fluid-Particle approaches
- Lesson 6: Vlasov Solvers

Fundamentals of the PIC Method, Fall 2018

This course introduces the Particle in Cell method used for low-density kinetic plasma simulations using a step-by-step approach. We develop 1D, 3D, and 2D (axisymmetric) codes to simulate the plasma sheath, E×B transport, ion flow past a charged sphere, and an cylindrical ion gun. Syllabus:

- Lesson 1: Governing equations
- Lesson 2: 1D sheath
- Lesson 3: 1D Sheath continued
- Lesson 4: Random numbers, velocity sampling, and magnetic confinement
- Lesson 5: 3D flow past a sphere
- Lesson 6: Multiple species, surface interactions, collisions, and data visualization
- Lesson 7: Potential solvers, mesh options, and virtual probes
- Lesson 8: Axially-symmetric flows

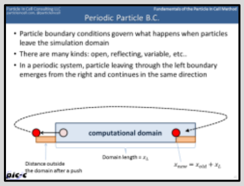
Advanced PIC Techniques (past recordings only): This course covers topics beyond the scope of the intro course. It covers three main concepts: electromagnetic PIC (EM-PIC), Direct Simulation Monte Carlo (DSMC) collision modeling, and finite element PIC (FEM-PIC).

Distributed Computing for Plasma Simulations (past recordings only): In this course we will learn how to develop plasma simulation codes that utilize multiple CPUs and graphic cards to handle larger simulation domains or to run faster. We cover multithreading, distributed computing with MPI, and GPU computing using CUDA.

Instructor: Dr. Lubos Brieda (M.Sc. AE 2005 Virginia Tech, Ph.D. 2012 AE/ME GWU), is the founder and president of Particle In Cell Consulting, LLC, a Los Angeles-based company specializing in providing tools and services for the plasma physics and rarefied gas communities. Dr. Brieda has over 10 years of experience developing PIC codes for a wide range of applications, including electric propulsion, space environment interactions, surface processing, and plasma medicine. His teaching experience includes the position of a Lecturer at the George Washington University.

For more information and to register, visit:
https://www.particleincell.com/courses/
or contact us at info@particleincell.com

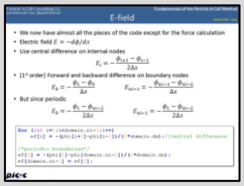
EXAMPLE COURSE SLIDES



Fundamentals of the Particle In Cell Method

Periodic Particle B.C.

- Particle boundary conditions govern what happens when particles leave the simulation domain
- There are many kinds: open, reflecting, variable, etc..
- In a periodic system, particle leaving through the left boundary emerges from the right and continues in the same direction



Fundamentals of the Particle In Cell Method

E-field

- We now have almost all the pieces of the code except for the force calculation
- Electric field $E = -d\phi/dx$
- Use central difference on internal nodes

$$E_i = -\frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x}$$

- (1st order) Forward and backward difference on boundary nodes

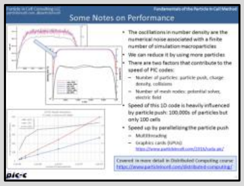
$$E_0 = -\frac{\phi_1 - \phi_0}{\Delta x} \qquad E_{ni-1} = -\frac{\phi_{ni-1} - \phi_{ni-2}}{\Delta x}$$

- But since periodic

$$E_0 = -\frac{\phi_1 - \phi_{ni-2}}{2\Delta x} \qquad E_{ni-1} = -\frac{\phi_1 - \phi_{ni-2}}{2\Delta x}$$

```
for (int i=1;i<domain.ni-1;i++)
    ef[i] = -(phi[i+1]-phi[i-1])/(2*domain.dx);//central difference

/*periodic boundaries*/
ef[0] = -(phi[1]-phi[domain.ni-2])/(2*domain.dx);
ef[domain.ni-1] = ef[0];
```

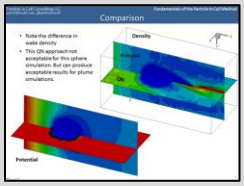


Fundamentals of the Particle In Cell Method

Some Notes on Performance

- The oscillations in number density are the numerical noise associated with a finite number of simulation macroparticles
- We can reduce it by using more particles
- There are two factors that contribute to the speed of PIC codes:
  - Number of particles: particle push, charge density, collisions
  - Number of mesh nodes: potential solver, electric field
- Speed of this 1D code is heavily influenced by particle push: 100,000s of particles but only 100 cells
- Speed up by parallelizing the particle push
  - Multithreading
  - Graphics cards (GPUs)

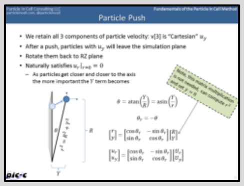
Covered in more detail in Distributed Computing course: https://www.particleincell.com/distributed-computing/



Fundamentals of the Particle In Cell Method

Comparison

- Note the difference in wake density
- This QN approach not acceptable for this sphere simulation. But can produce acceptable results for plume simulations.



Fundamentals of the Particle In Cell Method

Particle Push

- We retain all 3 components of particle velocity: v[3] is "Cartesian" u_y
- After a push, particles with u_y will leave the simulation plane
- Rotate them back to RZ plane
- Naturally satisfies $u_r|_{r=0} = 0$
  - As particles get closer and closer to the axis the more important the Y term becomes

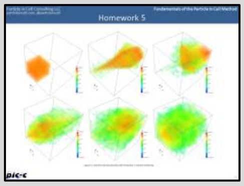
$$\theta = \mathrm{atan}\left(\frac{Y}{R}\right) = \mathrm{asin}\left(\frac{Y}{r}\right)$$

$$\theta_r = -\theta$$

$$\begin{bmatrix} r \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta_r & -\sin\theta_r \\ \sin\theta_r & \cos\theta_r \end{bmatrix} \begin{bmatrix} R \\ Y \end{bmatrix}$$

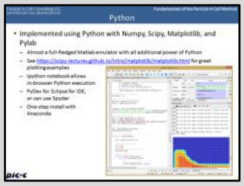
$$\begin{bmatrix} u_r \\ u_y \end{bmatrix} = \begin{bmatrix} \cos\theta_r & -\sin\theta_r \\ \sin\theta_r & \cos\theta_r \end{bmatrix} \begin{bmatrix} U_r \\ U_y \end{bmatrix}$$

Note, this matrix multiplication is not needed, can compute r=Y and set y = 0



Fundamentals of the Particle In Cell Method

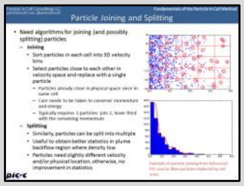
Homework 5



Fundamentals of the Particle In Cell Method

Python

- Implemented using Python with Numpy, Scipy, Matplotlib, and Pylab
  - Almost a full-fledged Matlab emulator with all additional power of Python
  - See https://scipy-lectures.github.io/intro/matplotlib/matplotlib.html for great plotting examples
  - Ipython notebook allows in-browser Python execution
  - PyDev for Eclipse for IDE, or can use Spyder
  - One-step install with Anaconda



Fundamentals of the Particle In Cell Method

Particle Joining and Splitting

- Need algorithms for joining (and possibly splitting) particles
  - Joining
    - Sort particles in each cell into 3D velocity bins
    - Select particles close to each other in velocity space and replace with a single particle
      - Particles already close in physical space since in same cell
      - Care needs to be taken to conserve momentum and energy
      - Typically requires 3 particles: join 2, leave third with the remaining momentum
  - Splitting
    - Similarly, particles can be split into multiple
    - Useful to obtain better statistics in plume backflow region where density low
    - Particles need slightly different velocity and/or physical location, otherwise, no improvement in statistics