Particle-in-Cell (PIC) Simulations of Plasmas

Marc Swisdak
University of Maryland

Physics 761
28 November 2017
E and B are known on the grid. Particles move freely.
Why Doing Plasma Physics via Computer Simulations Using Particles Makes Good Physical Sense

Inspired by Birdsall & Langdon, Plasma Physics via Computer Simulation

- Debye length $\lambda_D = v_{th}/\omega_{pe} \ll L$; we care about $\lambda \gtrsim \lambda_D$.
- For a meaningful plasma $N_D = n\lambda_D^3 \gg 1$
- But that means

$$\frac{\text{KE (thermal kinetic energy)}}{\text{PE (electrostatic potential energy)}} = N_D^{2/3} \gg 1$$

- $\therefore$ Particles interact collectively, not discretely.
- Grids with $\Delta x \lesssim \lambda_D$ capture the important physics without the unimportant inter-particle effects.
Cartoon Timestep

Update $\mathbf{E}, \mathbf{B}$ at particles

Update $\mathbf{E}, \mathbf{B}$ on grid

Update $\mathbf{x}, \mathbf{v}$ for particles

Update $\mathbf{J}$ on grid
Updating $\mathbf{x}$, $\mathbf{v}$, $\mathbf{J}$, $\mathbf{B}$, and $\mathbf{E}$

- **Field advancement:**
  $$
  \frac{\partial \mathbf{B}}{\partial t} = -c \nabla \times \mathbf{E} \quad \frac{\partial \mathbf{E}}{\partial t} = c \nabla \times \mathbf{B} - 4\pi \mathbf{J}
  $$

- **Particle advancement:**
  $$
  \frac{d\mathbf{x}}{dt} = \mathbf{v} \quad \frac{d(\gamma \mathbf{v})}{dt} = \frac{q}{m} \left( \mathbf{E} + \frac{\mathbf{v}}{c} \times \mathbf{B} \right)
  $$

- **Current density update:**
  $$
  \mathbf{J} = \sum_i q_i \mathbf{v}_i S(\mathbf{X} - \mathbf{x}_i)
  $$

where $S(\mathbf{X} - \mathbf{x})$ is a shape function.
Translating Between Particles and the Grid
Adapted from https://www.particleincell.com/2010/es-pic-method/
Effective Particle Shapes (1D)
Adapted from https://perswww.kuleuven.be/~u0052182/weather/pic.pdf

- Nearest gridpoint
- First-order (cloud-in-cell)
- Quadratic spline
Does PIC Satisfy $\nabla \cdot \mathbf{B} = 0$ and $\nabla \cdot \mathbf{E} = 4\pi \rho$?

Numerically, $\nabla \cdot (\nabla \times \mathbf{E}) = 0$

$$\frac{\partial}{\partial t} (\nabla \cdot \mathbf{B}) = -c \nabla \cdot \nabla \times \mathbf{E} = 0$$

If $\nabla \cdot \mathbf{B} = 0$ at $t = 0$, it remains so (ignoring round-off)

In contrast,

$$\frac{\partial}{\partial t} (\nabla \cdot \mathbf{E}) = c \nabla \cdot \nabla \times \mathbf{B} - 4\pi \nabla \cdot \mathbf{J} = -4\pi \nabla \cdot \mathbf{J}$$

To satisfy Gauss’s Law requires

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \mathbf{J} = 0$$
Unfortunately · · ·
Continuity is not, in general, satisfied

Corrections fall into two broad categories

- “Fix” $E$
- “Fix” $J$

An approach of the first type: Suppose a $\Phi$ exists such that

$$E' = E - \nabla \Phi \quad \text{where} \quad \nabla \cdot E' = 4\pi \rho$$

Find $\Phi$ by solving

$$\nabla^2 \Phi = \nabla \cdot E - 4\pi \rho \equiv b$$

This ($\nabla^2 \Phi = b$) is Poisson’s equation and can be solved many different ways: FFTs, matrix methods, multigrid methods, · · ·
An Alternative: Fluid vs. PIC Simulations

**Fluid (MHD)**
- Advantages:
  - Correct on large scales
  - Computationally fast
- Disadvantages:
  - Wrong at small scales

**Kinetic (PIC)**
- Advantages:
  - ≈ All of the physics
- Disadvantages:
  - Must resolve important scales
  - Computationally painful
Resolution for Explicit PIC

For timestep $\Delta t$, grid spacing $\Delta x$, and velocity $u$ a general constraint is

- CFL (Courant-Friedrichs-Lewy):
  \[
  \frac{u \Delta t}{\Delta x} \leq 1
  \]

For plasmas also need to resolve important physical scales

- $\Delta x < (\lambda_D, \omega_{pe}, \rho_{Le})$
- $\Delta t < (\omega_{pe}, \omega_{ce})$

Not resolving generally leads to numerical instability.
Kinetic Scales

How painful?

- Solar corona: $B = 50$ G, $n = 10^9$ cm$^{-3}$, $L \approx 10^9$ m, $\tau \approx 10^3$ s
  - $d_p \approx 10$ m
  - $\Omega_{pc}^{-1} \approx 2 \times 10^{-6}$ s
  - $\omega_{pi}^{-1} \approx 2 \times 10^{-8}$ s

- Magnetosphere: $B = 2 \times 10^{-4}$ G, $n = 20$ cm$^{-3}$, $L \approx 10^4$ km, $\tau \approx 10^3$ s
  - $d_p \approx 50$ km
  - $\Omega_{pc}^{-1} \approx 0.5$ s
  - $\omega_{pi}^{-1} \approx 2 \times 10^{-4}$ s

- Tokamak: $B = 3 \times 10^4$ G, $n = 2 \times 10^{13}$ cm$^{-3}$, $L \approx 10^2$ cm, $\tau \approx 10^{-2}$ s
  - $d_p \approx 5$ cm
  - $\Omega_{pc}^{-1} \approx 3 \times 10^{-9}$ s
  - $\omega_{pi}^{-1} \approx 2 \times 10^{-10}$ s
Besides real systems being much larger than kinetic scales, nature insists on making the situation worse.

- $m_p/m_e \approx 1836$
- $c/v_A \gg 1$

The resulting separation of scales is computationally challenging. To combat it, artificial values are often used

- $m_p/m_e = 400, 100, 25$
- $c/v_A = 20 - 50$

Potential unwanted side-effects (e.g., $v_{th,e} \rightarrow c$) must be kept in mind.
PIC on Supercomputers

Domain Decomposition

A useful simulation (\( \gtrsim 10^{10} \) particles) needs many cores working in parallel. Communication should be minimized.
Supercomputer Performance

p3d: Weak Scaling on edison

Normalized Time

Number of Processors
Brief Notes on PIC-Related Topics
Accurate Numerical Differentiation
Not PIC-Specific
From the Taylor series

\[ f(x_0 + \Delta x) = f(x) + \Delta x \left. \frac{df}{dx} \right|_{x_0} + \frac{(\Delta x)^2}{2} \left. \frac{d^2f}{dx^2} \right|_{x_0} + \mathcal{O}(\Delta x^3) \]

comes the approximation

\[ \frac{df}{dx} = \frac{f(x + \Delta x) - f(x)}{\Delta x} + \mathcal{O}(\Delta x) \]

Incorporating a variation

\[ f(x_0 - \Delta x) = f(x) - \Delta x \left. \frac{df}{dx} \right|_{x_0} + \frac{(\Delta x)^2}{2} \left. \frac{d^2f}{dx^2} \right|_{x_0} + \mathcal{O}(\Delta x^3) \]

gives something more accurate

\[ \frac{df}{dx} = \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x} + \mathcal{O}(\Delta x^2) \]
Symmetry Reduces Errors and Helps Stability


Basic leapfrog algorithm
The Yee lattice is a popular – but not the only – choice. \( \mathbf{E} \) is known on edges, \( \mathbf{B}/\mathbf{H} \) on faces.

The finite-difference versions of Maxwell’s equations are nice, but bookkeeping is an annoyance.
Explicit Versus Implicit Algorithms

Consider

\[
\frac{\partial u}{\partial t} = D \frac{\partial^2 u}{\partial x^2}
\]

Explicit discretization:

\[
\frac{u_j^{n+1} - u_j^n}{\Delta t} = D \left[ \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} \right]
\]

Implicit discretization:

\[
\frac{u_j^{n+1} - u_j^n}{\Delta t} = D \left[ \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2} \right]
\]

Implicit is typically much more stable but requires much more work to solve.