Kinetic Simulations of Collisionless Plasmas

Rahul Kumar

PRINCETON UNIVERSITY
Plasma Physics on Computers

Particle-in-cell (PIC) method

Maxwell's equations

\[ \nabla \cdot \mathbf{B} = 0 \]

\[ \nabla \times \mathbf{B} = \frac{1}{c} \left( 4\pi \mathbf{J} + \frac{\partial \mathbf{E}}{\partial t} \right) \]

\[ \nabla \cdot \mathbf{E} = 4\pi \rho \]

\[ \nabla \times \mathbf{E} = -\frac{1}{c} \frac{\partial \mathbf{B}}{\partial t} \]

Lorentz force

\[ \mathbf{F} = q(\mathbf{E} + \frac{\mathbf{v}}{c} \times \mathbf{B}) \]

1950s-60s - Early development
Dawson, Hockney, Buneman, and others

1970s - Electrostatic PIC, methods develop

1980s-90s - EM PIC method gains ground
Computing Architectures

Marching Forward

Individual Processing Units are not getting any faster
More Processing Units are available

image source: web
Computing Architecture

New programming challenges

>> Processing more data in fewer instructions/cycles (SIMD)

Scalar operation vs. vector operation

>> Independent tasks

CPU

MPI, OpenMP

>> data accessibility

CPU

Cache

RAM

Disk
Particle-in-Cell Code

https://github.com/rahulbgu/PICTOR
PICTOR

Particle-in-Cell Code

https://github.com/rahulbgu/PICTOR

Electromagnetic code
> Second order
> Boris Pusher
> Yee Grid

> Charge conserving current deposition

OpenMP

CPU

SIMD

MPI

RAM

GPU Memory

CPU

GPU (Optional/Exclusive)

MPI
PICTOR

Particle-in-Cell Code

Precision Control
Dynamic Load Balancing
Machine Level Optimization
Cache Usage Optimization
Memory Usage Optimization
OpenMP+SIMD+GPU
MPI + Parallel HDF5

Multiple Species
Test Particles
Particle Tracking
Moving/Fixed Boundaries
Comprehensive Output
Simple/Flexible Setup
Shock, Turbulence, ..
Adopting to SIMD

**Interpolation**

\[
\text{for } (E_i, B_i)
\]

\[
\text{prtl}_E x \; += \; wt \times E_x
\]

\[
\text{prtl}_E x \; = \; \_\text{sum}_\text{ } ( wt \times E_x )
\]

**SIMD**

**Reshape**
Improving Memory Access

Sorting particle data according to their locations significantly improves cache hit.

A simple *Bucket sorting* improves performance significantly both on CPU and GPU.

Sorting keeps the particle data compact.

Strip-mining of particles data (with CPU threads)
Low-level Optimization

32 Threads per Warp

Warp level reduction

Write to memory

Reduces conflicts among concurrent threads/data traffic
Particle-in-Cell Code

PICTOR

https://github.com/rahulbgur/PICTOR

Performance

> CPU 2.4 GHz Xeon
  2D: ~20 M particle/s
  3D: ~8 M particle/s

> GPU Tesla K20X
  3D: ~40-50 M particle/s
Improving GPU memory usage

Strategy: divide available memory in chunks

Particle data

Particle sorting:

1  2  3

4

empty
Exascale Computing

the next frontier

Scalability

Reduce internode communications and global communications

Concurrency

Designing algorithms for very high-concurrency

Performance

Minimize data movement

Input/Output

Asynchronous writing of the output file

Keeping codes simple!

Optimizing total (Computer+Human) time
Prefential Heating and Acceleration of Heavy Ions

Impulsive Solar Flares

Is it due to Cyclotron damping of Alfvénic turbulence?

simulate the turbulence and see how different species behave in the turbu...
Anisotropic Turbulent Cascade

Parallel Velocity
Average Energy/Mass Vs. Time

Heavier ion species (smaller $Q/M$) heat up at faster rate!

\[
\bar{v}_{th} = \bar{v} - \bar{E} \times B_0 \hat{x}/B_0^2
\]

Resonance Condition

\[
\omega \pm k\parallel v\parallel = \omega_c
\]
Perpendicular Heating

Heating is mostly due to the transverse electric field

Velocity space distribution is anisotropic, $T_\perp > T_\parallel$
Acceleration to non-thermal energies

Some particles are accelerated to non-thermal energies.

Heavier ions are accelerated to slightly higher energies.
Acceleration to non-thermal energies

What is so special about the accelerated particles?

Resonance Condition: \[ \omega \pm k_{||} v_{||} = \omega_c \]

Particles that end up in the non-thermal tail vs. all

Accelerated particles had relatively large initial speed to begin with!
Dynamics of The Solar Wind Termination Shock
An Unusual Shock

Most of the dissipated energy did **not** go into heating SWIs.

The downstream was moving faster than expected (low compression).

Why So Different?

Ans: Pickup Ions
No Pickup Ion Data from Voyager 2

Self-consistent plasma simulations can help uncover the missing details!
Shock Simulation

Reflecting Wall

400$c/\omega_{pe}$

$\vec{B}_0$

Expanding Box

Upstream Flow $M_A=5$

Numerical Parameters

$M^+ / M^- = 100$

32 electrons/cell

$V_A/c = 0.01$

$\mathbf{f}(u) \propto \left(\frac{u}{u_c}\right)^{-3/2}$

$u_c \approx V_{sw}$

Maxwellian SWI

25% PUI  75% SWI

Electrons

$t=0.2\Omega_{ci}^{-1}$

$\vec{B}^+/\lambda$
Simulations Vs. Observations

Shock compression ratio is 3.2 (Voyager 2 data suggests 2.5)
SWIs are also hotter than observed

A clear disagreement between numerical simulations and Voyager 2 observations!
A Closer Look At The Data

Most pickup ions were produced when the Solar wind was moving faster!

The wind slowed down

Magnetic field strengthened

Upstream plasma is compressed and heated!


Pickup ions are expected to become more energetic
The gain in energy for PUIs at the shock is larger if they were already more energetic upstream of the shock.
Density Compression

The observed compression was about 2.5

More energetic upstream PUIs lead to lower density compression
Magnetic Field Structure

Day 243 of 2007  Day 244 of 2007

Time (h:min)

Voyager 2 data

Overshoot

PUI overshoot

Undershoot

$X / R_{SWI}^{L}$
Understanding the Energy Partition

Time = $7.2 / \Omega_i$

- **SWI (no PUI)**
- **SWI**
- **PUI**