8. Particle-in-Cell Simulation of Cosmic Plasma

8.1 "*In-Situ*" Observation of Cosmic Plasmas via Computer Simulation

While it is thinkable that our ability to make in situ measurements can perhaps be extended to the nearest stars, most of the universe beyond a few parsecs will be beyond the reach of our spacecraft forever.

From one's unaided view of the clear night sky, it is tantalizing to believe that the physics of the universe can be unfolded from the observable stars, which may be up to kiloparsecs away, or from the fuzzy "nebula" such as the galaxy M31, nearly a megaparsec away. Our experience in unfolding energetic events in our own solar system suggests otherwise.

The inability to make in situ observations places a severe constraint on our ability to understand the universe, even when the full electromagnetic spectrum is available to us. As outlined in Chapter 1, only after satellites monitored our near-earth environment and spacecraft discovered and probed the magnetospheres of the planets, could we begin to get a true picture of the highly-energetic processes occurring everywhere in the solar system. These processes included large-scale magnetic-field-aligned currents and electric fields and their role in the transport of energy over large distances. The magnetospheres of the planets are invisible in the visual octave (400–800 nm) and from earth cannot even be positively identified in the X ray and gamma ray regions, which cover 10 times as many octaves and have more than 1,000 times the bandwidth as the visual octave. Only in the low frequency radio region is there a hint of the presence of quasi-static electric fields which accelerate charged particles in the magnetospheres of the planets.

As the properties of plasma immediately beyond the range of spacecraft are thought not to change, it must be expected that plasma sources of energy and the transport of that energy via field-aligned currents exist at even larger scales than that found in the solar system. How then are we to identify these mechanisms in the distant universe?

The advent of particle-in-cell (PIC) simulation of cosmic plasmas on large computer systems ushered in an era whereby in situ observation in distant or inaccessible plasma regions is possible. While the first simulations were simple, with many physics issues limited by constraints in computer speed and memory, it is now possible to study the full three-dimensional, fully-electromagnetic evolution of magnetized plasma over a very large range of sizes. In addition, PIC simulations have matured enough to contain Monte Carlo collisional scattering and energy loss treatments, conductor surfaces, dielectric regions, space-charge-limited emission from surfaces and regions, and electromagnetic wave launchers. Since a simulation involves the motion of charge or mass particles according to electromagnetic or gravitational forces, all in situ information is available to the simulationist.

If the simulation correctly models the cosmic plasma object under study, replication of observations over the entire electromagnetic spectrum should be expected, to the extent that the model contains sufficient temporal and spatial resolution.

8.2 The History of Electromagnetic Particle-in-Cell Simulation

After the early success of astronomers in rigorously solving the problem of two gravitationally interacting bodies it became quite a disappointment that the notorious "probleme de trois corps" could never be solved by elegant, nineteenth century mathematics.[1] Computations were practical (and respectable) only for the evaluation of a series. Finite difference calculus made its way very slowly during the first few decades of this century. Störmer struggled hard calculating orbits of charged particles in the earth's magnetic field (not even a self-consistent field!) [Störmer 1955].

Strangely, it was a change in physics which brought the next advance: quantum theory changed particle dynamics from ordinary differential equations to partial differential equations, thus putting field and particle dynamics on the same footing. The combination of Schroedinger's equation for electron density with Poisson's equation for the electric potential results in coupled nonlinear partial differential equations. As a first step, taken in the 1920's, one eliminated the angle variables and reduced the problem to two nonlinearly coupled ordinary differential equations in the radial variable.

This meant that an efficient integrating machine or procedure was called for and D. R. Hartree built his first "differential analyzer" from an erector set. It used a continuously variable gear and with this device Hartree could solve mechanically self-consistent problems dealing with atomic wave functions and atomic energy levels.

The "magnetron", a now very familiar microwave generator, had been invented by Boot, Randall, and Sayers in Birmingham. It was of paramount importance to Britain's defense: its high frequencies could not be jammed. The magnetron (Section 1.7.3) is a fine example of "swords into plow-shares." It is replacing man's tradition of many millennia to cook food with incandescent heat.

Initially it was something of a mystery exactly how and why the magnetron worked and the scientific staff at the British Admiralty realized that in order to unravel the workings of the magnetron one would have to solve a self-consistent field problem, namely, that of motion of electrons in the electric field which the electrons themselves produce, in addition to the externally applied electric and magnetic fields.

Hartree was given the problem and he initiated classical particle simulation by integrating, numerically, the orbits of a number of particles in a field which was either revised in accordance with the instantaneous charge density at each step, or only occasionally, in the hope of reaching a steady field by iteration. Both one- and two-dimensional simulations were performed by Hartree and his team: Phyllis Nicolson, Oscar Buneman,; and David Copley [Nash 1991]. "Parallel-processing" was employed by sharing out the several hundred orbits between the three team members.

Operating as human central processing units (CPUs), the team made a number of discoveries including the Crank–Nicolson iteration procedure and the Buneman–Hartree threshold criterion for magnetron operation. The one-dimensional simulations yielded a steady-state but could not account for magnetron operation, or for the observed currents which flow across the magnetic barrier. Only when the technique was taken to two-dimensions did Buneman find an instability, not unlike the Kelvin–Helmholtz in fluid flow. In the transition from one- to two-dimensional simulation, iterative methods were abandoned (in 1944) and Hartree changed to the direct Fourier method (Section 8.4.3). It turned out that only a few harmonics were needed for simulation; the fast-Fourier-transform (FFT) was not yet known. Success came in the numerical observation of the four- and six-wheel spokes of electrons that rotate in the magnetron exciting the high frequencies in the resonators.

The numerical simulation by particles of plasma physics began in the 1950s by Dawson at Princeton and Buneman at Stanford, where various plasma phenomena were identified and studied. It should be mentioned that, in the beginning, it was not at all apparent that the technique developed to study pure electron beam propagation in microwave devices could be applied to the plasma state of matter. Unlike the cold electron beam with charges of all one sign, plasmas often consist of thermal distributions with essentially equal density of charges of opposite sign and greatly different masses. In studying cold electron beams, a few dozen particles sufficed to reproduce the essence of the experiment. However, in laboratory plasmas one has scale lengths greater then the Debye length ($L >> \lambda_D$) and the number of particles in a Debye cube $N_D \equiv n\, \lambda_D^3$ is much greater than one ($N_D >> 1$). For example, the earth's ionosphere has $N_D \approx 10^4$ and the literal simulation of it over its scale length appears unfeasible. However the general character of plasmas can often be found by studying the *collective behavior of collisionless plasmas at wavelengths longer than the Debye length,* $\lambda \gtrsim \lambda_D$. It was found that another characterization of a plasma is that (1) the thermal kinetic energy is much greater than the microscopic potential energy, and (2) the ratio of collision to plasma frequencies is much less than one. Both requirement can be met with rather low values of N_D [Birdsall and Langdon 1985]. Conditions 1 and 2 may be met for finite sized particles called clouds. Clouds occur naturally in simulations which use a spatial grid for interpolation, as well as in simulations which employ spectral methods where the particle profile (usually gaussian) is specified in **k** space.

The term "particle-in-cell" derives from Frank Harlow and his group's work at Los Alamos in the 1950s in investigating the fluid nature of matter at high densities and extreme temperatures. Modern descriptions of the particle-in-cell technique as related to plasma physics are found in the two texts *Computer Simulation Using Particles* [Hockney and Eastwood 1981] and *Plasma Physics via Computer Simulation* [Birdsall and Langdon 1985].

8.3 The Laws of Plasma Physics

The challenge of plasma physics is this: We know with certainty the precise and simple laws of nature that govern the particle and fields in plasmas, yet we are unable to deduce from them how a nontrivial plasma configuration will evolve, nor can we "explain" many of the complicated plasma phenomena which are observed in the laboratory.

Mathematical manipulation of the laws, and intuitive additional assumptions or approximations, have been exploited with only partial success, and often the computer has had to be called upon to "finish the job" in such attempts. The message to be presented here is that one might try to let the computer take us all the way from the basic laws to their macroscopic manifestations. Rather than maximizing intuition and shortcuts which might help the computer get there quicker, let us program the basic laws in their rawest, simplest form and leave all the synthesizing to the computer. Given a big enough computer, this philosophy would justify itself by the demonstration that the computer could not only reproduce all the plasma phenomena which have been observed but also all known theoretical results and, of course, those obtained from more modest simulations on more primitive computers.

In practice, it would be foolish to relinquish theoretical tools, intuition, insight, and past experience entirely in favor of "brute force" computing. Even given a computer that could handle billions of particles with a resolution of one in a thousand for each dimension at reasonable speed, there would still be some "ε" of imperfection to be checked for, and the pressure to get more results per computing dollar would motivate physicists back toward a compromise between traditional theory and highest-power computing.

What is suggested here is a start from the far end: suppose we had that ultracomputer, then how would we do physics with it? And since we haven't got that ultracomputer, what can we do as the next best approximation to that ideal?

We begin our approach by stating the laws of plasma physics in more or less the form which it has been found convenient to program: the equation of motion for the particles with the Lorentz force Eq.(1.5), and the Maxwell–Hertz–Heaviside laws for the electric and magnetic fields Eqs.(1.1)–(1.4).

8.4 Multidimensional Particle-in-Cell Simulation

8.4.1 Sampling Constraints in Multidimensional Particle Codes

The particle-in-cell technique for the analysis of complex phenomena in science has evolved from 1D through $1\frac{1}{2}$D, $1\frac{2}{2}$D, 2D, $2\frac{1}{2}$D, to 3D particle simulations. While at first one has to face certain limitations of an analytic nature, ultimately the limits are set by data management problems the resolution of which depends critically on the available hardware.

A trivial reason for the increasing difficulty of higher dimensional particle simulations is their demand for substantially greater particle numbers. With each added dimension the number of sampling particles has to be multiplied by a certain factor.

This also applies to "half-dimensions." It is customary to denote the inclusion of extra velocity components by referring to them as "half-dimensions." Typically, a pure 1D simulation simulates the plasma as rigid sheet particles, all parallel to the *y*-*z* plane, say, and moving in the *x* direction. It ignores *y* and *z* motions of the planes. A $1\frac{1}{2}$D simulations keeps a record of possible *y* motions, uniform within each plane. Then the *x*-ward Lorentz force in the presence of a *z*-directed magnetic field can be taken into account, as well as the *y*-wards Lorentz force due to *x* motions. In a $1\frac{2}{2}$D simulation, *dz*/*dt* would be recorded as well. In a $2\frac{1}{2}$D simulation, *x*, *y*, *dx*/*dt*, and *dz*/*dt* are tracked (but not *z*); the particles are rigid straight rods whose motion along their axis is taken into account. One-and-a-half dimensional and $1\frac{2}{2}$D simulations have recently found application in space plasma work, namely for simulating the critical ionization phenomenon.

It is reassuring that relatively few samples can often give very good statistics. In many applications the velocity distributions stay close to maxwellian and a modest factor (typically four) in the sample number may suffice to deal with an added half-dimension. One exploits the favorable feature of statistics when initializing thermal velocity distributions: each velocity component is made up as the sum of four random numbers (each uniformly random in a certain interval). The resulting distribution (the "four dice curve," or cubic spline) is almost indistinguishable from the Gaussian.

However, when incrementing by a full dimension, sampling requirements jump dramatically. It is easily checked that the statistical potential energy fluctuations in a granular plasma compete with thermal energies when the particles are spaced on the order of a Debye length apart. Such a plasma would be essentially collision dominated. One is mostly interested in collective effects, since fluid codes are adequate for collision-dominated phenomena. Obviously, one needs several particles per Debye length; again, a modest number suffices. Now, most of the interesting phenomena to be resolved by simulation are on the scale of many Debye lengths (hundreds or thousands). Therefore, the addition of each full dimension calls for an increase of the number of particles by, typically, two orders of magnitude. In 1990 a 1D (electrostatic) simulation can barely be squeezed onto a personal computer, a 2D simulation calls for a minicomputer or workstation, and a 3D simulation needs a supercomputer[2] (Figure 8.1).

8.4.2 Discretization in Time and Space

One-dimensional, $1\frac{1}{2}$D, and $1\frac{2}{2}$D simulations can be done without discretizing in space. The electrical interaction of sheets is independent of distance and one only needs to order the sheets to calculate their accelerations. Even if the sheets are of finite thickness or if they are "soft" (i.e., they have a smooth density profile across), only a few operations per sheet are needed to move them one time step. This is an effort of order N where N is the number of sheets.

However, in all simulations, time must be discretized. By studying the simulation of a simple 1D problem, namely, electrostatic oscillations in a cold plasma, and by Fourier transforming one's numerical procedure in time, one finds that while a time step $\delta t = \omega_p^{-1}$ yields the plasma frequency to 5% accuracy, for $\delta t > 2\ \omega_p^{-1}$ one runs into an instability.

To get over this severe limitation of the speed of simulation in cases where the phenomena of interest are much slower than electrostatic oscillations (typically ion responses), one can either use implicit methods, or one can make one's ions lighter than real ions. Much has been learned from simulations with ion-electron mass ratios as low as 16:1.

The big analytical problems in simulation arise when one advances to two dimensions. Interactions between rods of charge depend on distance, and the many remote rods are as important as the few near ones. For N rods, one has to calculate N^2 interactions and N itself might be typically two orders of magnitude larger than for a 1D simulation.

In order to get back to an effort of order N per step in the particle advance, one tabulates the field over a spatial grid and calculates the self-consistent field from a grid record of the charge and current-densities that each particle contributes.

The permissible coarseness of the grid mesh becomes a critical issue and the problem of integrating the finite difference version (now in both space and time) of the field equation is far from trivial. Fortunately, both these subjects have been advanced to a state of relative completion and are exhaustively covered in two texts.

Very briefly and broadly, one can state that the grid should be fine enough to resolve a Debye length, and that smoothing or filtering of high spatial frequencies should be practiced in order to minimize "aliassing." This is the stroboscopic phenomenon of high frequencies parading as low frequencies (long wavelengths). Many physical instabilities set in preferentially at long wavelengths and can thus be excited numerically through aliassing. Grid effects can often be studied and

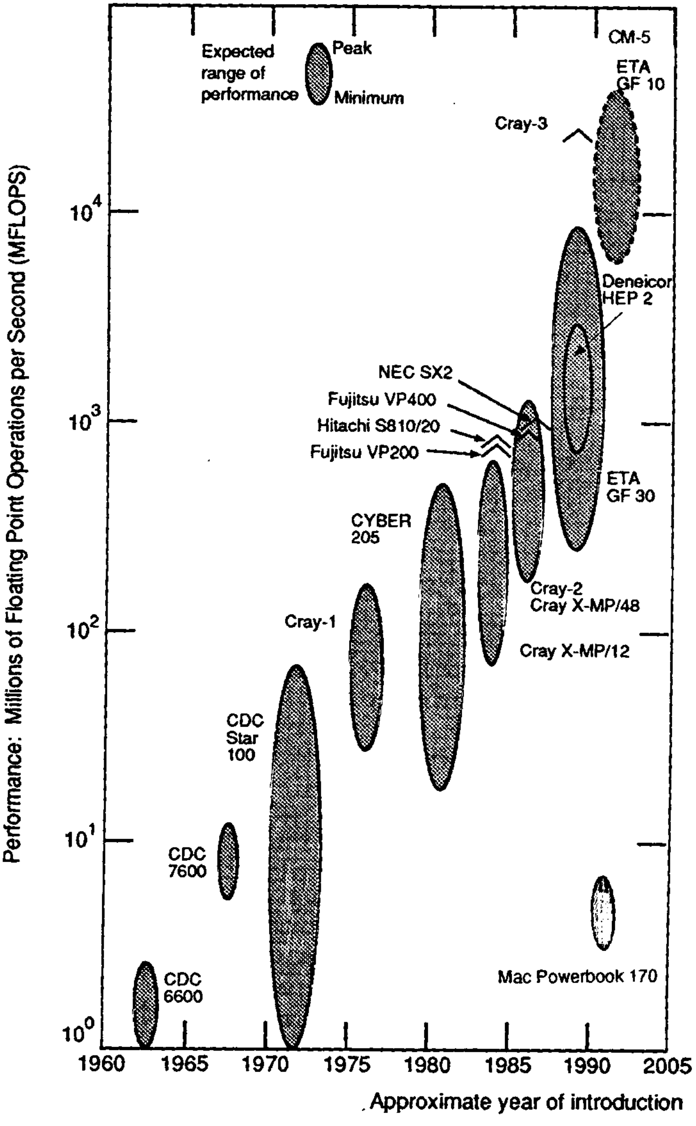

Figure 8.1. Performance ranges of supercomputers, 1960–2005 [adapted from S. Karin and N.P. Smith, *The Supercomputer Era* (Hartcourt, Brace, Jovanovich, Boston 1987)].

checked in 1D where grids are optional. Smoothing can be achieved by particle shaping (i.e., spreading point particles into soft balls with a smooth bell-shaped internal density profile). Likewise, splines and finite-element techniques help.

Regarding the field update from the charge-current record, fast noniterative methods for solving Poisson's equation over an L-by-M mesh have been developed. These include cyclic reduction in rows and columns, or Fourier transforming in one of the two dimensions (say, that of M). This is an effort to the order $LM \log_2 M$. Two-dimensional simulations go back historically to Hartree who initiated the simulation of the pure electron plasma which circulates in the magnetron.

Hartree; also pioneered the time-centered update of the particles from Lorentz's equation Eq.(1.5)

$$\boxed{\frac{d\mathbf{v}}{dt} \pm \frac{e\mathbf{B}}{m} \times \mathbf{v} = \pm \frac{e}{m}\mathbf{E}} \qquad (8.1)$$

using $(\mathbf{v}^{\text{new}} + \mathbf{v}^{\text{old}})/2$ in the second (Lorentz) term and solving the linear equation for $\mathbf{v}^{\text{new}}$ explicitly. No limitation of $\omega_b \delta t = (e B / m)\, \delta t$ arises from this method except that for large values of $\omega_b \delta t$ the phases of the gyromotion are misrepresented. For small $\omega_b \delta t$ one gets the same results as with cycloid fitting (Section 8.5), i.e., joining solutions of the type

$$\mathbf{v}^{\perp} = \mathbf{E} \times \mathbf{B}/B^2 + \text{gyration at frequency } \omega_b \qquad (8.2)$$

for the components of the velocity transverse to **B**. As regards this particle update, there is no significant increase in effort when advancing from $1\frac{1}{2}$D to 2D and 3D.

A further time-step limitation is encountered when one wants to integrate the full electromagnetic equations over the grid. Because the Maxwell–Hertz–Heaviside equations (not including Poissons's) are hyperbolic (i.e., they contain a natural $\partial/\partial t$ or "update" term), they can be solved in an effort which is of the order of magnitude of the number of grid points, LM in the 2D example discussed earlier. Essentially, one solves the wave equation Eq.(B.1). However, this process becomes unstable unless one observes the Courant speed limit $\delta t < \delta x / c$ in 1D, $\delta t < \delta x / c\sqrt{2}$ in 2D, and $\delta t < \delta x / c\sqrt{3}$ in 3D for square and cubic meshes of side δx. In many applications, scales chosen from other considerations are such that c is a large number and this restriction of δt results in a severe slowdown.

8.4.3 Spectral Methods and Interpolation

The Courant condition can be overcome by doing the entire field update in the transform domain. The Maxwell–Hertz–Heaviside laws for the electric and magnetic fields Eqs.(1.1)–(1.4) can be conveniently combined into one equation for the complex field vector $\mathbf{F} = \mathbf{D} + i\mathbf{H}/c$. When Fourier transforming, this equation becomes

$$\boxed{\frac{d\mathbf{F}}{dt} - c\,\mathbf{k} \times \mathbf{F} = -\mathbf{j}} \qquad (8.3)$$

for the spatial harmonic which goes like $\exp\left(i\,\mathbf{k}\cdot\ \mathbf{r}\right)$. This field equation is surprisingly similar to that for the particle velocities Eq.(8.1) and has the corresponding solution for the transverse part of **F**:

$$\begin{aligned}\mathbf{F}^{\perp} &= \mathbf{j}\times\mathbf{k}/k^{2}\ \text{(magnetostatic field)}\\ &+\ \text{circularly polarized wave rotating at angular frequency } ck\end{aligned} \tag{8.4}$$

The time intervals at which one joins successive solutions of this form are dictated by the rate at which **j** changes, not by the magnitude of *ck*.[3]

To Eq.(8.1) we should add an initial condition, namely, Poisson's

$$\boxed{\mathbf{ik}\cdot\ \mathbf{F}_{\mathbf{k}}=\rho_{k}} \tag{8.5}$$

Fourier transforming all fieldlike quantities has many advantages. For instance, the longitudinal part of **F** (which is just **D**) can be obtained from Poisson's equation as $k\,\rho/k^{2}$. Of course, transforming in two dimensions rather than only one (as in the fastest Poisson solvers) makes for an effort of the order $LM(\log_2 L + \log_2 M)$. On the other hand, the ready availability of well-programmed FFTs and the additional benefits of spectral methods make up for this increase in effort.

In the transform domain one can perform the filtering, the particle shaping, an optimization for the spline fitting process, and the truncation of the interaction to be discussed in the section on boundary conditions. One does not have to use any spatial finite difference calculus for the field equations. However, a grid is still necessary since we have only *discrete* numerical Fourier transforms between r space and k space.

This leaves the problem of interpolation in the mesh. By using high-order interpolation, one can greatly reduce aliassing and improve accuracy. Quadratic and cubic splines have been used, but this soon becomes expensive.[4]

Linear interpolation is most commonly used. Interpolation is also needed when the particles contribute their charge and current to the ρ, j arrays. Linear interpolation is then, in 2D, equivalent to "area weighting." For 3D, we have cut down the data look-up (or deposit) effort for linear interpolation by using a tetrahedral mesh. Each particle references only the four nearest mesh-point data. The tetrahedra result from drawing the space diagonals into a cubic mesh and introducing cubic center data. Interpolation of currents must be done twice in each step of each practice, once at its old position and once at its new position, since the current is that due to the movement between the two.

8.5 Techniques for Solution

The crucial equations, Eqs.(8.1) and (8.3), are in the "update" form, ideally suited to computers which are themselves devices whose function it is to update the state of their memory continually, albeit not continuously. If the time interval δt between updates is so chosen that during this interval changes of **E** and **B**, as seen by any particle, can be ignored in Eq.(8.1) and changes of **j** can be

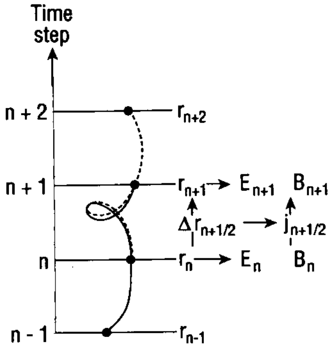
Time step
n + 2
n + 1
n
n - 1
r_{n+2}
$r_{n+1} \rightarrow E_{n+1}$ B_{n+1}
$\Delta r_{n+1/2} \rightarrow j_{n+1/2}$
$r_n \rightarrow E_n$ B_n
r_{n-1}

Figure 8.2. Leap-frogging particles and fields: At time n, cycloids with drifts and gyrations due to $\mathbf{E}_n$ and $\mathbf{B}_n$ are fitted through r_n and r_{n-1}, to be continued through r_{n+1}. Displacements Δr_{n+1} determine currents $\mathbf{j}_{n+1/2}$ from which the transverse fields $\mathbf{E}_n, \mathbf{B}_n$ are advanced to $\mathbf{E}_{n+1}, \mathbf{B}_{n+1}$. The longitudinal $\mathbf{E}_{n+1}$ is obtained from the r_{n+1}. Then, at time $n+1$, new cycloids (dotted) with drifts and gyrations due to $\mathbf{E}_{n+1}$ and $\mathbf{B}_{n+1}$ are fitted through r_{n+1} and r_n, to be continued through r_{n+2}, etc.

ignored in Eq.(8.3), each equation can be solved exactly for the entire interval no matter how long this interval is: The Lorentz equation Eq.(8.1) then produces cycloidal motion in a plane perpendicular to **B**, composed of a drift and a gyration Eq.(8.2). This may be accompanied by free fall parallel to **B**, generated by a parallel electric field component. Given the initial velocity and position, or given the position and displacement during the preceding time interval, the displacement during any subsequent interval, and the new position, can be computed precisely.

8.5.1 Leap-Frogging Particles Against Fields

The average value of the fields **E** and **B** for Eq.(8.1) or the current **j** to be used in Eq.(8.3) is taken to be the actual value at the middle of the time interval. Figure 8.2 shows how the updating from average values proceeds at equal intervals along a time axis. This involves the following

(1) Construct the cycloidal orbit of a particle from t_{n-1} through t_n to t_{n+1} using the known mean values of $\mathbf{E}_n$ and $\mathbf{B}_n$ at t_n (the middle of the interval) and the known particle positions at t_{n-1} and t_n.

(2) This gives the displacement of the particles from t_n to t_{n+1}, and their final positions at t_{n+1}.

(3) The harmonic $\mathbf{j}_k$ of the mean current flowing during the last interval is obtained by summing all the displacements $\delta\mathbf{r}$ with phase factors $\exp i\mathbf{k}\cdot\mathbf{r}$ given by their mean positions, times $q/\delta t$.

(4) The transverse fields $\mathbf{F}_k$ are now advanced by $\mathbf{j}_k$ Eq.(8.3) from t_n to t_{n+1}(Section 8.5.3).

(5) The longitudinal fields $\mathbf{F}_k$ are obtained from ρ_k at t_{n+1} by summing $q\, e^{i\mathbf{k}\cdot\mathbf{r}}$ with the new positions **r** at t_{n+1}.

(6) The process is repeated from t_n through t_{n+1} and then to t_{n+2}.

Each time interval is covered by two cycloids for each particle, one with drift and gyrofrequency as given by the fields at the beginning of the interval, the other as given by the fields at the end of the interval. Both cycloids pass through the same points at the termini (Figure 8.2).

8.5.2 Particle Advance Algorithm

The time interval δt must be smaller than $\sim\omega_p^{-1}$ for proper resolution of electrostatic plasma oscillations, and $\omega_b\ \delta t \leq 1$ to account for synchrotron radiation or for ∇B drifts. The steps in the solution algorithm are given by Hockney [1966] and Buneman [1967]. The interpretation and implementation of this as an electric acceleration followed by a magnetic rotation and another acceleration is due to Boris [1970]. The updating of the particle positions and velocities is done using a time-centered second-order scheme, valid for relativistic particle velocities,

$$\frac{\mathbf{v}_{new}-\mathbf{v}_{old}}{\delta t}=\frac{q}{\gamma m_0}\left[\mathbf{E}+\left(\frac{\mathbf{v}_{new}+\mathbf{v}_{old}}{2}\right)\times\mathbf{B}\right] \tag{8.6}$$

With the scaling[5]

$$\mathbf{E}\leftarrow\frac{q\,\delta t}{2m_0}\mathbf{E},\quad \mathbf{B}\leftarrow\frac{q\,\delta t}{2m_0}\mathbf{B}, \tag{8.7}$$

Eq.(8.6) is solved by the following sequence[6],

$$\gamma_1=\left(1-(\mathrm{v}/c)^2\right)^{-1/2}$$

$$\mathbf{u}_1=\gamma_1\,\mathbf{v}_{old}$$

$$\mathbf{u}_2=\mathbf{u}_1+\mathbf{E} \qquad \text{(first half of electric acceleration)}$$

$$\gamma_2=\left(1+(u_2/c)^2\right)^{+1/2}$$

$$\mathbf{u}_3=\mathbf{u}_2+\left(\frac{2}{\gamma_2^2+B^2}\right)\left(\gamma_2\,\mathbf{u}_2+\mathbf{u}_2\times\mathbf{B}\right)\times\mathbf{B} \qquad \left(\text{rotation through angle arc tan}\,\frac{qB\,\delta t}{2m}\right)$$

$$\mathbf{u}_4=\mathbf{u}_3+\mathbf{E} \qquad \text{(second half acceleration)}$$

$$\mathbf{v}_{new}=\mathbf{u}_4\left(1+(u_4/c)^2\right)^{+1/2}$$

$$\mathbf{x}_{new}=\mathbf{x}_{old}+\delta t\,\mathbf{v}_{new} \tag{8.8}$$

This process is equivalent to rotating the deviation from the gyrocenter drift through the angle $2\,\mathrm{arc\ tan}\left(qB\ \delta t/2m\right)$—the "cycloid fitting"—combined with uninhibited electric acceleration along the magnetic field. It is second-order and time reversible.

Since this algorithm now properly accounts for the effects of relativity, particles are automatically restrained from exceeding the speed of light and need not be artificially braked at c. This limit on the distance traveled by a particle during a single time step plays an important role in particle data management.

8.5.3 Field Advance Algorithm

The advance of the fields through one time step of arbitrary length (subject to $\mathbf{j}_k$ = constant) is mathematically just like that of the particles. According to Eq.(8.4), $\mathbf{F}_k$ consists of a constant component plus a rotating component. The constant part represents the magnetostatic field generated by the currents; the rotating part represents a circularly polarized electromagnetic wave. Again, the advance through any time interval δt is straightforward. The longitudinal (purely electric) component of the field is updated from the record of ρ at the end of the time step using Eq.(8.5).

We note, so far, we have not invoked finite difference calculus either in the space or time domain and, typically, the advance of the fields from Maxwell–Hertz–Heaviside's equations is not restricted by any "Courant condition." However, δt is constrained by the fact that **E** and **B** should not change across the range of the orbit excursions during δt.

Equation (8.3) is used to trace the evolution of the transverse field only. The longitudinal, electrostatic field is constructed "from scratch" at the new time, using the charge density records:

$$\mathbf{F}_{\mathbf{k}}^{\mathrm{e-s}} = i\,\mathbf{k}\,\rho_k / k^2 \tag{8.9}$$

The longitudinal field, then, need not be held over through the particle move phase: it can be generated directly by Fourier transforming the charges accumulated during that phase. The transverse field is calculated as follows. A particular solution is constructed from $\mathbf{j}_k$ using

$$\mathbf{F}_{\mathbf{k}}^{\mathrm{m-s}} = \mathbf{k} \times \mathbf{j}_{\mathrm{k}} / k^2 \tag{8.10}$$

To $\mathbf{F}_{\mathbf{k}}^{\mathrm{m-s}}$ one has to add the rotating "electromagnetic" component

$$\mathbf{F}_{\mathrm{k}}^{\mathrm{e-m}}(\mathrm{new}) = \mathbf{F}_{\mathrm{k}}^{\mathrm{e-m}}(\mathrm{old}) \cos k\ \delta t - (\mathbf{k}/k) \times \mathbf{F}_{\mathrm{k}}^{\mathrm{e-m}}(\mathrm{old}) \sin k\ \delta t \tag{8.11}$$

The new fields are then reconstructed from the updated pieces according to

$$\mathbf{F}_{\mathrm{k}}(\mathrm{new}) = \mathbf{F}_{\mathrm{k}}^{\mathrm{m-s}} + \mathbf{F}_{\mathrm{k}}^{\mathrm{e-m}}(\mathrm{new}) + \mathbf{F}_{\mathrm{k}}^{\mathrm{e-s}} \tag{8.12}$$

and this is kept on record for the next field update.

The field seen by a particle must then be obtained by summation over the entire available spectrum

$$\mathbf{F}(\mathrm{r}) = \mathbf{D}(r) + i\,\mathbf{H}(r)/c = (2\pi)^{-3} \int_{\mathbf{k}} \mathbf{F}_{\mathbf{k}}\, e^{-i\,\mathbf{k}\cdot\mathbf{r}}\, d\mathbf{k}$$

$$= (2\pi)^{-3} \sum_{\mathbf{k}_x} \sum_{\mathbf{k}_y} \sum_{\mathbf{k}_z} \mathbf{F}_{\mathbf{k}}\, e^{-i\,\mathbf{k}\cdot\mathbf{r}} \tag{8.13}$$

To calculate **F**, we must introduce a grid over which field values are generated from the spectrum by FFTs, and we must interpolate the local field from the grid record. Likewise, charge and current harmonics must be built up by interpolation into a grid and subsequent FFTs.

Having avoided spatial grids and spatial finite-difference calculus so far, the introduction of a grid to obtain the electromagnetic fields from the spectrum $\mathbf{F}_k$ leads to difficulties associated with grids: inaccuracies and stroboscopic effects. These problems are reduced using higher-order interpolation methods [Buneman et al. 1980].

8.6 Issues in Simulating Cosmic Phenomena

8.6.1 Boundary Conditions

A major problem in space applications is to simulate free-space conditions outside the computer domain. Complex Fourier methods [with exp $(i\,\mathbf{k}\cdot\ \mathbf{r})$] imply periodic repeats of the computed domain in all dimensions. If the simulation is to represent phenomena in a rather larger plasma, such repeats are acceptable, but for an isolated plasma of limited extent they become unrealistic. This problem can be overcome by keeping a generous empty buffer zone around the domain containing particles and truncating the interaction between charges beyond a certain radius so that the nonphysical repeats introduced by the Fourier method cannot influence the central plasma. This was first applied to gravitational simulations.

The most elusive boundary problem for space plasmas is the radiation condition. To decide what part of the field in the charge-current-free space outside the plasma is outgoing and what is incoming presents no problem in 1D and the incoming part can be suppressed.

In 2D the decision is more difficult. It requires information not only in the source-free boundary layer at any time but also over its past history. It almost seems as if, in principle, the entire past history is needed for the decision. However, Lindman found that a fairly short history (such as three past time steps) of the boundary suffices for an algorithm which will suppress all but 1% of the incoming radiation at all but the shallowest angles of incidence. However, just carrying an absorption layer in an outer envelope seems quite successful. This method, due to Green, simply multiplies the electric and magnetic field by a factor which smoothly approaches zero away from the plasma. A method employing spherical harmonics which should be 100% effective has been reported [Buneman 1986].

8.6.2 Relativity

A reason for keeping the mass coupled with the velocities in the update steps is that under relativistic conditions one really updates momenta rather than velocities. Note, however, that in the $q\mathbf{B}\ \delta t / 2m$ terms one needs $m^{-1} = (m_0^2 + p^2/c^2)^{-1/2}$ where $p = m\mathrm{v}$. During the rotation, this magnitude of the momentum does not change, but in the electric acceleration it does. After the full update of momenta, one must again divide by m in order to get $\mathbf{v} = \delta\,\mathbf{r}/\delta t$. In practice, v rather than the momentum is stored for each particle which means that at the beginning of the update one must calculate $m = m_0(1-\beta^2)^{-1/2}$. Thus, there are three separate calls to a reciprocal square root in the relativistic advance of each particle. As system supplied square roots are time consuming and more accurate than needed for particle pushing, a Padé-type rational first approximation, followed by a Newton iteration, is used instead.

8.6.3 Compression of Time Scales

The number of steps required to simulate a significant epoch in the evolution of a real plasma configuration would be many million, typically, if t has to be of the order ω_p^{-1} or ω_b^{-1}. In order to bring this down to the more acceptable range of several thousand steps, one must compress the time scales. Compressing time scales can be achieved by (1) decreasing the ions' rest mass in relation to the electrons' rest mass, and (2) increasing temperatures so that typical particle velocities get closer to the velocity of light.

For an ion (proton) to electron simulation mass ratio of 16, ion gyrofrequencies $\omega_{ci} = eB/m_i$ will be high by a factor of 1836/16 = 115, ion plasma frequencies $\omega_{pi} = \sqrt{n_i Z^2 e^2 / m_i \varepsilon_0}$, ion thermal velocities $\mathrm{v}_{Ti} = \sqrt{kT_i / m_i}$, the Alfvén velocity $\mathrm{v}_A = \sqrt{B^2 / \mu_0 n_i m_i}$, and the relative velocity in Biot–Savart attraction $\mathrm{v} \approx I_z \sqrt{\mu_0 L / 2\pi \sum m_i}$ will be high by factor of $\sqrt{1836/16} = 10.7$.

The exaggeration of temperatures provides one of several motivations for incorporating relativity into our codes. Note, incidentally, that even a 10-kV electron, a temperature typical of many space plasmas, already moves at 1/5 c.

The exaggeration of "temperatures" of beam or current electrons can also be achieved by exaggerating the external electric field E_z responsible for accelerating the particles. This technique greatly reduces the number of time steps required to study a phenomenon such as Birkeland current formation and interaction in cosmic plasma. Since the current density is proportional to the electric field (i.e., $j_z = I_z / A = n_e e \mathrm{v}_z \sim (n_e e^2 / m_e) E_z t$), both the time required for the pinch condition Eq.(1.9) to be satisfied, and the relative velocity between parallel currents [Eq.(3.49)], are linearly related to E_z.

Of course, when economy necessitates time compression, the time-scales must be "unfolded" upon simulation completion.

8.6.4 Collisions

Just as in real plasmas, there are encounters between particles and these give rise to collisional effects which influence the physics of the model. Since computer models are limited to some 10^6 particles whereas a laboratory plasma may have 10^{18}–10^{20} particles and a galaxy, 10^{65} particles, each particle in the model is a "superparticle" representing many plasma electrons or ions. Thus the forces between model particles are much larger than in a real plasma and the collisional effects are much greater. Fortunately there is a way to reduce the model collisions to rates comparable with real plasmas. This involves the finite-size particle method [Okuda and Birdsall 1970, Langdon and Birdsall 1970].

The 3D codes discussed in this chapter use a gaussian profile for particles. The shaping is done in **k**-space. This is achieved by first building up ρ_k and $\mathbf{j}_k$ for $|\mathbf{k}| < k_{max}$ (truncation of harmonics at maximum k), as if each computer particle were a point and then applying a gaussian filter in **k**-space [Buneman et al. 1980]. The particle shape is then of the form $exp\left(-r^2 k_p^2 / 2\right)$, where the particle profile factor k_p is left as a users option: many simulations have used a profile which keeps the spectrum flat up to a fairly large k and then makes a rapid but smooth slope-off to zero at some desired k_{max}.

A limit to the maximum acceptable radius of the finite-sized particles is set by the collective properties of the plasma. If the effective radius of a gaussian particle is increased much beyond the Debye length, it takes over the role of the Debye length, causing collective effects to be altered.

In simulating a physical system, plasma or gravitating, it is usually sufficient to determine if the system models a collisionless one over the simulation time span. Experimental determination of the effective collisional frequency v_c in 2D models closely follows the empirical law [Hockney and Eastwood 1981]

$$\frac{v_c}{\omega_p / 2\pi} = N_D^{-1}\left[1 + \left(\frac{w}{\lambda_D}\right)^2\right]^{-1}$$

where w is the width of the particle and $N_D = n\ \lambda_D^2$ is the number of particles in a Debye square. In 3D simulations, the reduction of v_c is achieved, without increasing N_D, by "softening the blow" of collisions—making the particles into fuzzy balls. Okuda [1972] has calculated values $v_c / \omega_p / 2\pi \approx 10^{-3}$ for gaussian profile particles for N_D of order unity. This value is consistent with most plasmas, in laboratories or space.

So far we have only considered collisions between particle species that are charged. However, in weakly ionized plasmas where the number of uncharged particles may be hundreds or thousands of times more prevalent, it is often the collision between the massive ions and the massive neutral atoms that cause a redistribution of energy, and concomitant effects, such as plasma heating. Collisions in weakly ionized plasmas have been successfully treated by melding PIC algorithms with MCC—Monte Carlo Collision—algorithms [Kwan, Snell, Mostrom, Mack, and Hughes 1985; Snell, Kwan, Morel, and Witte 1990; Birdsall 1991].

PIC codes involve deterministic classical mechanics which generally move all particles simultaneously using the same time step. The only part left to chance is usually limited to choosing initial velocities and positions and injected velocities. The objective for highly-ionized space plasma is usually seeking collective effects due to self and applied fields. On the other hand, MCC codes are basically probabilistic in nature, seeking mostly collision effects in relatively weak fields. For example, let a given charged particle be known by its kinetic energy W_{kin} and its velocity relative to some target particles. This information produces a collision frequency $v_{coll} = n_{target}\,\sigma(W_{kin})\,v_{relative}$ and a probability that a collision will occur. This information is then used to describe electron collisions with neutrals (elastic scattering, excitation, and ionization) and ion collisions with neutrals (scattering and charge exchange).

The method is to use only the time step of the PIC field solver and mover, δt, and then to collide as many particles as is probable P in that δt separately. The actual fraction of particles in collision is $P = 1 - exp\left(-v_{coll}\ \delta t\right)$. Note that we have slipped into treating our computer particles as single electrons, not as superparticles; the implication is that with a sufficient number of collisions, the resultant scatter in energy and velocity will resemble that of the single particles.

The end result of current efforts at including collisions in PIC codes due to Monte Carlo methods is the change in velocities of the particles. Thus, the only change from a collisionless run is that the particle velocities are varied in a time step. The last task at the end of a time step is to determine the new (scattered) velocity, and new particle velocities if ionization occurs (if ionization and/or recombination processes have been included in the MCC model). Each process is handled separately. Elastic collisions change the velocity angles of the scattered electrons; charge exchanges decrease ion energy and change velocity angles; ionizations do these and create an ion-electron pair, with new velocities. The effect on the neutral gas is not calculated because the lifetimes of the excited atoms are generally less than a time step.

When recombination rates are high, and if the source of energy to the plasma is terminated, gravitational effects must soon be included in the particle kinematics.

8.7 Gravitation

The transition of plasma into stars involves the formation of dusty plasma (Appendix C), the sedimentation of the dust into grains, the formation of stellesimals, and then the collapse into a stellar state. While the above process appears amenable to particle simulation, a crude approximation of proceeding directly from charged particles (actually a cloud of charged particles) to mass particles is made.

The transition of charge particles to mass particles involves the force constant, that is, the ratio of the coulomb electrostatic force between two charges q separated a distance r,

$$F_q(r) = q^2 / 4\pi \varepsilon_0 r^2 \tag{8.14}$$

to the gravitational force between two masses m separated a distance r,

$$F_G(r) = -G m^2 / r^2 \tag{8.15}$$

In the particle algorithm this change is effected by the following:

(1) Changing all particles to a single species.

(2) Limiting the axial extent of the simulation to be of the order of less than the extent or the radial dimension (i.e., about the size of the expected double layer dimension).

(3) Setting the axial velocities to zero.

(4) Setting the charge-to-mass ratio equal to the negative of the square-root of the gravitational constant (times $4\pi\varepsilon_0$).

This last change produces attractive mass particles via the transformation $\varphi_G(r) = \varphi_q(r)$ in the force equation $\mathbf{F} = -\nabla\varphi$, where

$$\varphi_q(r) = q^2 / 4\pi\ \varepsilon_0\ r \tag{8.16}$$

and

$$\varphi_G(r) = -G\,m^2/r \tag{8.17}$$

are the electrostatic and gravitation potentials, respectively.

8.8 Scaling Laws

The scaling of plasma physics on cosmical and laboratory scales generally involves estimates of the diffusion in plasma, inertia forces acting on the currents, the Coriolis force, the gravitational force, the centrifugal force, and the **j**×**B** electromagnetic force [Bostick 1958, Lehnert 1959].

Specification of plasma density, geometry, temperature, magnetic field strength, acceleration field, and dimension set the initial conditions for simulation. The parameters that delineate the physical characteristics of a current-carrying plasma are the electron drift velocity

$$\beta_z = \mathrm{v}_z/c \tag{8.18}$$

the plasma thermal velocity

$$\beta_{th} = \mathrm{v}_{th}/c = \frac{(\lambda_D/\Delta)\,(\omega_p\,\delta t)}{c\,\delta t/\Delta} \tag{8.19}$$

and the thermal/magnetic pressure ratio

$$\beta_p = \frac{n_e k T_e + n_i k T_i}{B^2/2\mu_0} = \frac{[(\lambda_D/\Delta)(\omega_p\,\delta t)]^2\,4(1 + T_i/T_e)}{(c\,\delta t/\Delta)^2\,(\omega_{c0}/\omega_p)^2} \tag{8.20}$$

where n is the plasma density, T is the plasma temperature, k is Boltzmann's constant, and the subscripts e and i denote electron and ion species, respectively. The parameter δt is the simulation time step, Δ is the cell size, and c is the speed of light. All dimensions are normalized to Δ and all times are normalized to δt.

The simulation spatial and temporal dimensions can be changed via the transformation

$$\frac{c\,\delta t}{\Delta} = \frac{c\,\delta t'}{\Delta'} = 1 \tag{8.21}$$

where $\Delta' = \alpha\,\Delta$ and $\delta t' = \alpha\,\delta t$, for the size/time multiplication factor α. The values of n, T, **B**, and **E** remain the same regardless of whether the simulations are scaled to Δ and δt or to Δ' and $\delta t'$.

One immediate consequence of the rescaling is that, while the dimensionless simulation parameters remain untouched, the resolution is reduced, that is, and

$$\varphi_G(r) = -G\,m^2/r \tag{8.17}$$

are the electrostatic and gravitation potentials, respectively.

8.8 Scaling Laws

The scaling of plasma physics on cosmical and laboratory scales generally involves estimates of the diffusion in plasma, inertia forces acting on the currents, the Coriolis force, the gravitational force, the centrifugal force, and the **j**×**B** electromagnetic force [Bostick 1958, Lehnert 1959].

Specification of plasma density, geometry, temperature, magnetic field strength, acceleration field, and dimension set the initial conditions for simulation. The parameters that delineate the physical characteristics of a current-carrying plasma are the electron drift velocity

$$\beta_z = \mathrm{v}_z/c \tag{8.18}$$

the plasma thermal velocity

$$\beta_{th} = \mathrm{v}_{th}/c = \frac{(\lambda_D/\Delta)\,(\omega_p\,\delta t)}{c\,\delta t/\Delta} \tag{8.19}$$

and the thermal/magnetic pressure ratio

$$\beta_p = \frac{n_e k T_e + n_i k T_i}{B^2/2\mu_0} = \frac{[(\lambda_D/\Delta)(\omega_p\,\delta t)]^2\,4(1 + T_i/T_e)}{(c\,\delta t/\Delta)^2\,(\omega_{c0}/\omega_p)^2} \tag{8.20}$$

where n is the plasma density, T is the plasma temperature, k is Boltzmann's constant, and the subscripts e and i denote electron and ion species, respectively. The parameter δt is the simulation time step, Δ is the cell size, and c is the speed of light. All dimensions are normalized to Δ and all times are normalized to δt.

The simulation spatial and temporal dimensions can be changed via the transformation

$$\frac{c\,\delta t}{\Delta} = \frac{c\,\delta t'}{\Delta'} = 1 \tag{8.21}$$

where $\Delta' = \alpha\,\Delta$ and $\delta t' = \alpha\,\delta t$, for the size/time multiplication factor α. The values of n, T, **B**, and **E** remain the same regardless of whether the simulations are scaled to Δ and δt or to Δ' and $\delta t'$.

One immediate consequence of the rescaling is that, while the dimensionless simulation parameters remain untouched, the resolution is reduced, that is,

$$\omega\,\delta t = \omega'\,\delta t' \tag{8.22}$$

where $\omega' = \omega / \alpha$ rad/s is the highest frequency resolvable.

To convert simulation results to dimensional form, it is sufficient to fix the value of one physical quantity (e.g., B_ϕ).

8.9 Data Management

The management of data is a contemporary problem only for 3D simulations. Using a modest factor of 4 in the total particle number for each velocity dimension and a factor 64 for each spatial dimension, one concludes that 3D simulations ought to employ some 2^{24}, that is, 16 million particles, each requiring six data (x, y, z, u, v, w) to be recorded (i.e., 100 million data).

Obviously, there is no hope for doing a plausible 3D simulation without (1) introducing some economies, and (2) storing the particles outside core and calling them in only small batches.

Similarly, the 64^3 linear scale range calls for over 1 million field data: Fields cannot reside in core. "Layering" is therefore practiced in 3D codes. In the tridimensional code TRISTAN a grid of 2×128^3 points (cubic mesh, plus cube centers) is used for recording field data, and particles are kept ordered into 128 layers. At present, the number of particles is 4 1/2 million. This is currently being increased to 50 million. Only two field layers are in core at any time, and four charge current layers (the latter because after moving, particles can have dropped below or risen above their original layers). Triple buffering is used when the particles of each layer are passed through core for processing, in batches of about 5,000. Images of the batches that straddle layer borders are kept in core, for depositing particles that have dropped or risen. Some sorting of the particles is necessary here.

Fields of the next layer above are brought in when a layer of particles has been completed and the lowest of the four charge-current layers is put out to disk. These two moves are accompanied by Fourier transforming within the layer dimensions (e.g., x and y, when layering is in z, to k_x to k_y). Filtering of high harmonics at this stage helps with input/output economy.

When all layers have been processed, the original field record on the disks is converted into a charge-current record, indexed in a hybrid manner, namely according to k_x, k_y, and z.

At this point, field solving can begin, but it will require prior Fourier transforming in z. The $\mathbf{j}, \rho$ data have to be read back into core, but in different order, from scattered disk areas. Again, layering must be used, but now according to k_y (say). TRISTAN uses 16 sectors at this point. Fourier transforming cannot begin until all reading of a sector is complete: Input/output (I/O) cannot be overlapped with computation. A similar bottleneck occurs after the field update when the FFTs in z have to be completed before the new fields of a sector can be written to disk.

It is difficult to assess the cost of waiting for I/O completion; this depends strongly on the operating system. A time step overall (particle plus field update) in TRISTAN takes about 2 m on a CRAY-1.

8.10 Further Developments in Plasma Simulation

As pointed out in Section 8.9, data management problems dominate the subject of 3D plasma simulation using particles-in-cell. In the novel computer architectures, with their high degree of parallelism, data transport becomes an even more important issue. Computing efficiency depends critically on (topological or physical) data proximity in the basic procedure of a problem. "Local" algorithms, such as finite-difference equations, have preference over "global" algorithms, such as Fourier transforms (Note that the calculation of each single Fourier harmonic requires the entire data-base).

With this in mind, new 3D plasma codes have been constructed. In these the particles are advanced just as in Eqs.(8.6)–(8.8), but Maxwell–Hertz–Heaviside's equations are integrated locally over a cubic mesh in the form:

change of **B**-flux through a cell-face = – circulation of **E** around it

change of **D**-flux through a cell-face = circulation of **H** around it – charge flow through it

The **E**- or **D**- data mesh is staggered relative to the **B**- or **H**- data mesh both in space and time.

This method has the advantage that $\nabla \cdot \mathbf{D} = \rho$ needs to be satisfied only at the beginning of a run (where it becomes a triviality of initialization): it is automatically carried forward in time by consistent determination of the charge flow between cells. Thus Poisson's equation does not have to be solved. Poisson's equation is "global": The solution anywhere depends on the data everywhere.

The algorithms for a simplified version of TRISTAN, a fully three-dimensional, fully electromagnetic, and relativistic PIC code, are given in Appendix E.

Notes

[1] The problem of two gravitationally interacting masses was solved when Kepler proved that relative to the center-of-mass the orbits are ellipses or hyperbolae (i.e., "conic sections"). This is exact—and elegant! But this elegance is lost when a third body is added. Only approximate methods exist, all rather ugly. For centuries mathematicians have sought to find an elegant solution to the problem of three gravitationally interacting bodies—without success.

[2] The approximate relationship of supercomputer performance and performance of those in other catagories can be shown proportionately. If the performance of contemporary supercomputers is assigned a value of 100, the values in proportion to supercomputers are: minicomputers 0.1 to 5, workstation 0.1 to 1.0, and personal computers 0.001 to 0.1.

[3] The wavevector $k_{max} = \pi/\delta x$, $\pi\sqrt{2}/\delta x$, or $\pi\sqrt{3}/\delta x$ according to the number of dimensions.

[4] In a 3D, EM code, cubic splines would require each particle to look up 384 data to interpolate the **E** and the **B** that acts on it!

[5] **E** describes half the electric acceleration and the magnitude of **B** is half the magnetic rotation angle during the time step.

[6] The sequence is mathematically concise when $\gamma = 1$. The quantity u is velocity in the sense of momentum per unit restmass. The relativistic γ is obtained from it as the square root of $1 + (u / c)^2$. The equation for u_3 is executed by first dividing **B** by γ_2 and then using "1" in place of γ_2. This accounts for the m rather than m_0 in the angle.