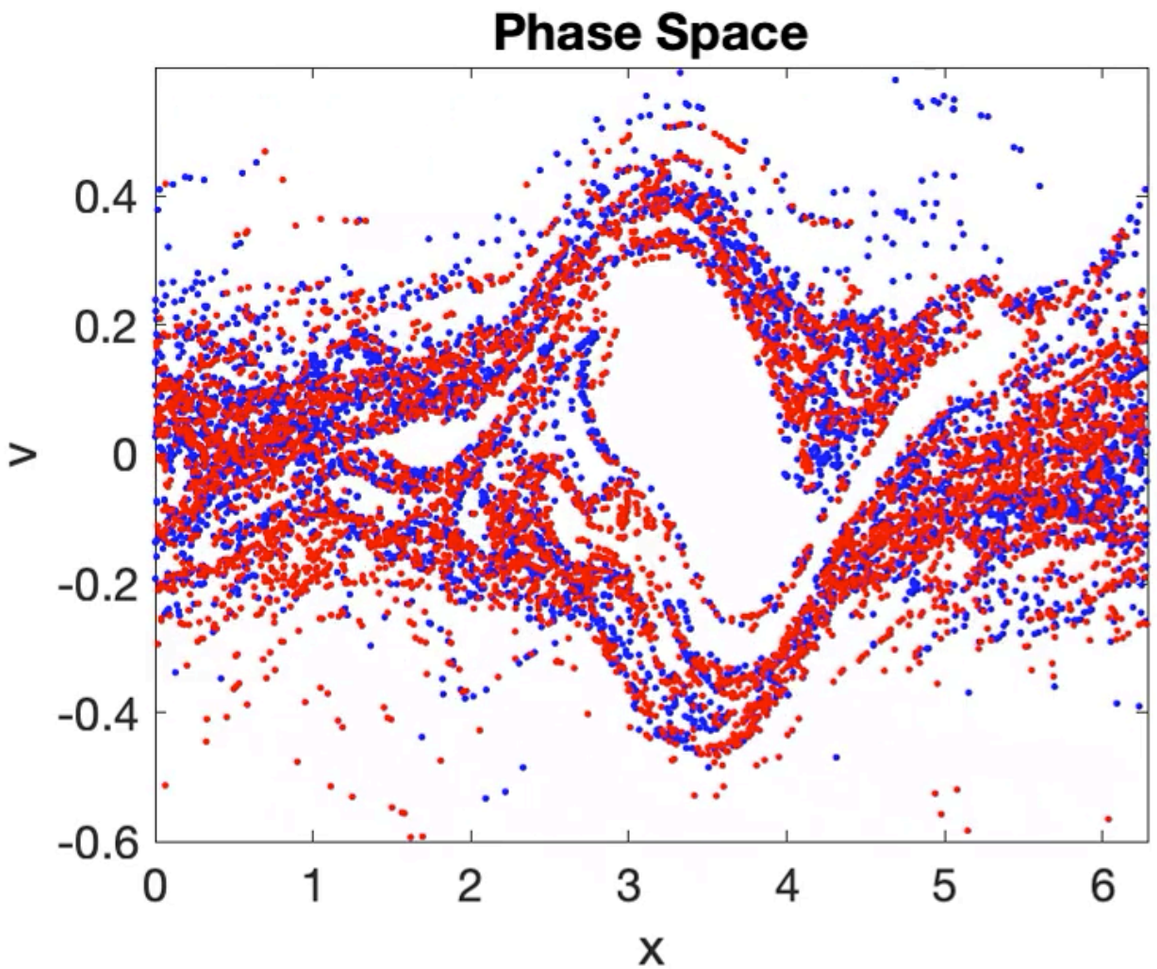
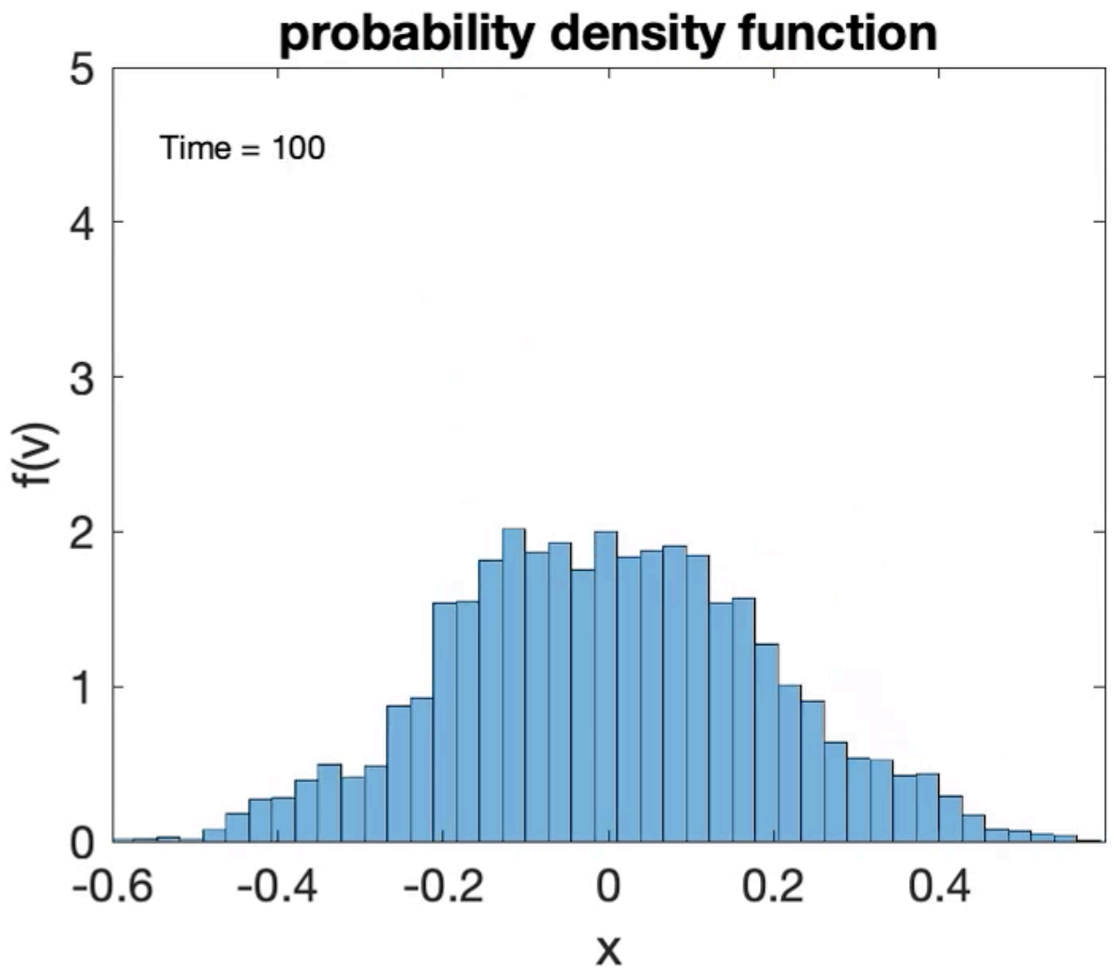
Topic 5

Electrostatic Particle Simulations





Outline

- Review of the 1D1V Vlasov simulation
- Physical Derivations of Particle simulations
- Mathematical Derivations of Particle simulations
  - Equation of Motion
  - Poisson Solving (electrostatic)
- Other considerations
- A simple example - electrostatic PIC in 1D

The Vlasov Equation: 2D linear advection

Recall the linear advection equation: 2-D

$$\frac{\partial}{\partial t}\rho + \frac{\partial}{\partial x}\underbrace{(u_x\rho)}_{F(\rho)} + \frac{\partial}{\partial y}\underbrace{(u_y\rho)}_{G(\rho)} = 0$$

Vlasov equation:

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} + E\frac{\partial f}{\partial v} = 0$$

Now here v and x are independent variables - v is not a function of x, so in the x-direction it's simply an linear advection equation (in configuration space)

The less intuitive part is the second term - eE/m is the acceleration in the velocity space, since eE/m is not a function of v, so the second term is also a linear advection (in velocity space)

$$\frac{\partial f}{\partial t} + \overset{u_x}{v}\frac{\partial f}{\partial x} + \overset{u_y}{E}\frac{\partial f}{\partial v} = 0 \longrightarrow \frac{\partial f}{\partial t} + u_x\frac{\partial f}{\partial x} + u_y\frac{\partial f}{\partial y} = 0$$

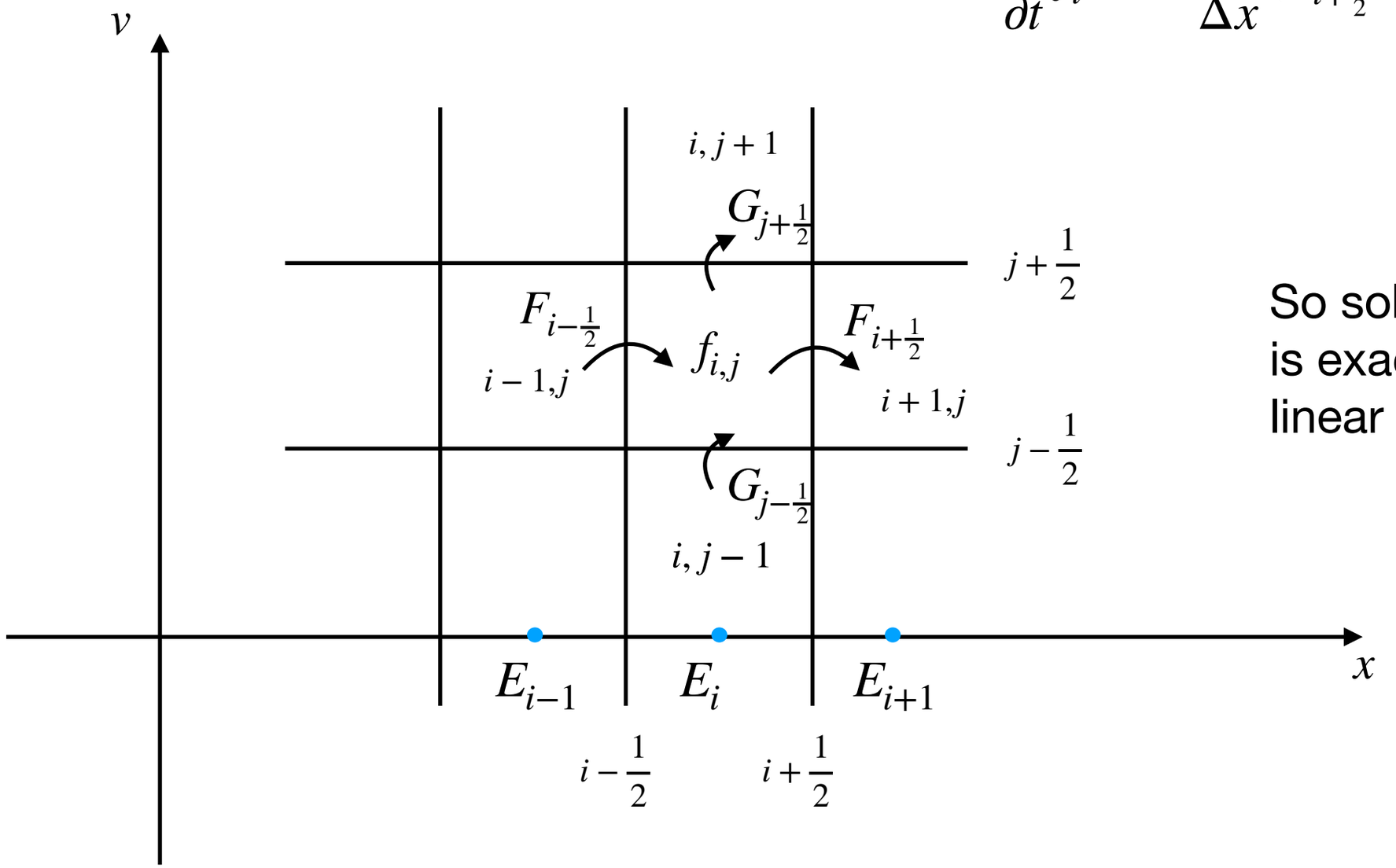
y

The Vlasov Equation: 2D linear advection

Finite Volume Vlasov Solver

$$\frac{\partial f}{\partial t} + \underbrace{v}_{u_x}\frac{\partial f}{\partial x} + \underbrace{E}_{u_y}\frac{\partial f}{\partial v} = 0 \longrightarrow \frac{\partial f}{\partial t} + \frac{\partial}{\partial x}\underbrace{(vf)}_{F(f)} + \frac{\partial}{\partial v}\underbrace{(Ef)}_{G(f)} = 0$$

$$\xrightarrow{\text{FV}} \frac{\partial}{\partial t} f_i = -\frac{1}{\Delta x}(F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}) - \frac{1}{\Delta y}(G_{j+\frac{1}{2}} - G_{j-\frac{1}{2}})$$

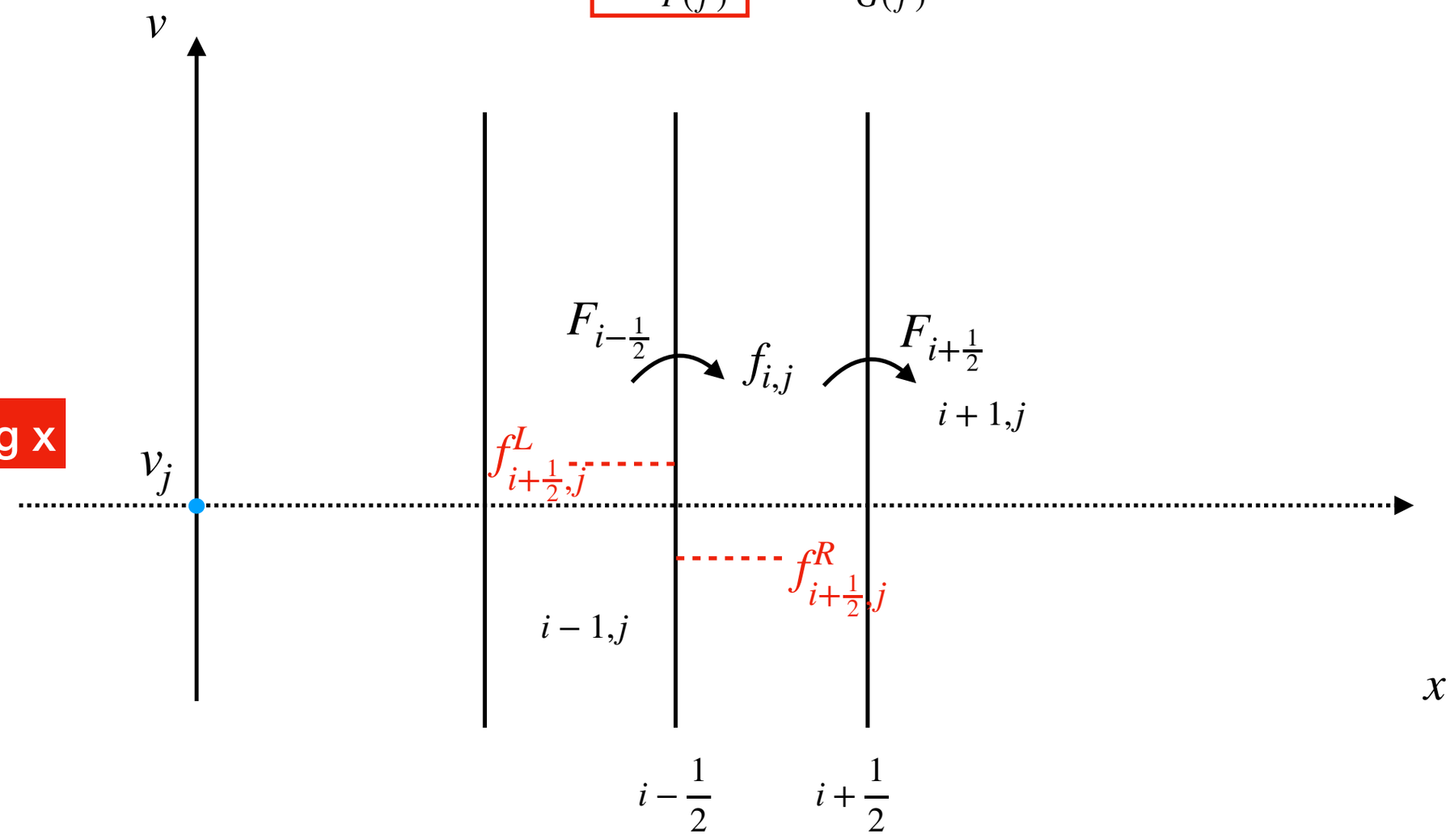


So solving the Vlasov equation is exactly the same as a 2-D linear advection problem!

The Vlasov Equation: 2D linear advection

in x-direction (configuration space)

$$\frac{\partial f}{\partial t} + \underbrace{v}_{u_x}\frac{\partial f}{\partial x} + \underbrace{E}_{u_y}\frac{\partial f}{\partial v} = 0 \longrightarrow \frac{\partial f}{\partial t} + \underbrace{\frac{\partial}{\partial x}(vf)}_{F(f)} + \underbrace{\frac{\partial}{\partial v}(Ef)}_{G(f)} = 0$$



v_j = const along x

Reconstruction in x-dir:

$$f^L_{i+\frac{1}{2},j} \quad f^R_{i+\frac{1}{2},j}$$

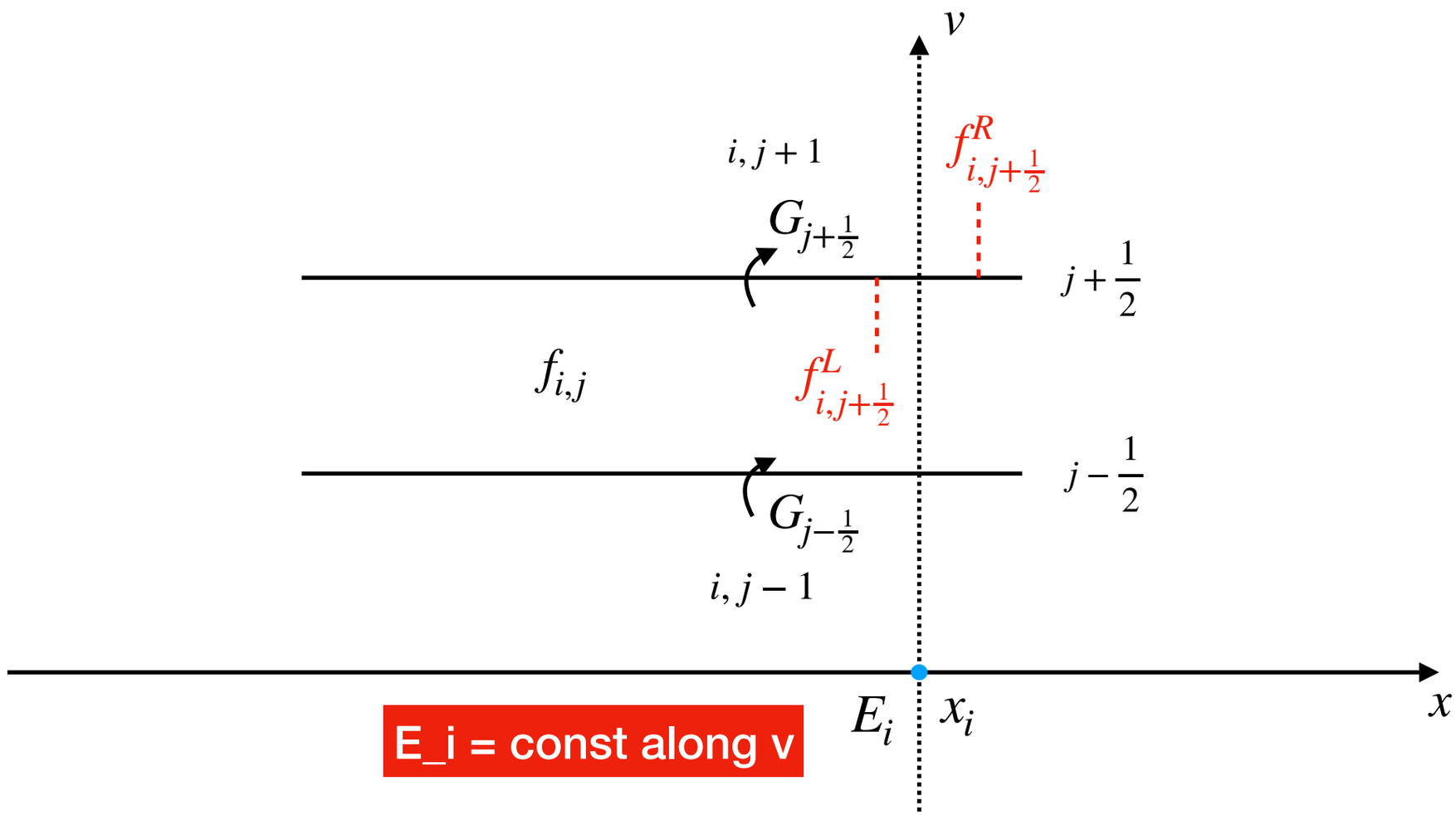
Interface Flux:

$$F_{i+\frac{1}{2}} = \frac{1}{2}(v_j f^L_{i+\frac{1}{2},j} + v_j f^R_{i+\frac{1}{2},j}) - \frac{1}{2}|v_j|(f^L_{i+\frac{1}{2},j} - f^R_{i+\frac{1}{2},j})$$

The Vlasov Equation: 2D linear advection

in v-direction (velocity space)

$$\frac{\partial f}{\partial t} + \underbrace{v}_{u_x}\frac{\partial f}{\partial x} + \underbrace{E}_{u_y}\frac{\partial f}{\partial v} = 0 \longrightarrow \frac{\partial f}{\partial t} + \underbrace{\frac{\partial}{\partial x}(vf)}_{F(f)} + \underbrace{\frac{\partial}{\partial v}(Ef)}_{G(f)} = 0$$



Reconstruction in x-dir:

$$f^L_{i,j+\frac{1}{2}} \quad f^R_{i,j+\frac{1}{2}}$$

Interface Flux:

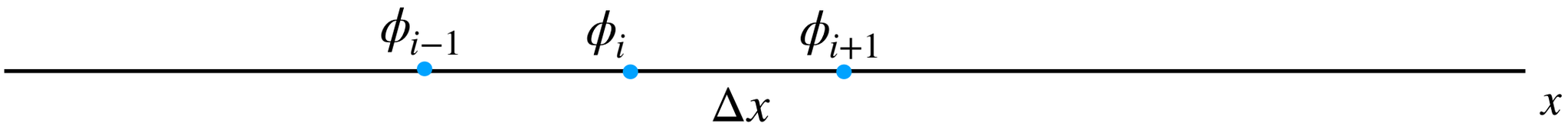
$$G_{i,j+\frac{1}{2}} = \frac{1}{2}(E_i f^L_{i,j+\frac{1}{2}} + E_i f^R_{i,j+\frac{1}{2}}) - \frac{1}{2}|E_i|(f^L_{i,j+\frac{1}{2}} - f^R_{i,j+\frac{1}{2}})$$

The Poisson Equation: 1D Diffusion

The normalized Poisson equation for electric potential: $\dfrac{\partial^2 \phi}{\partial x^2} = -\rho_e$

Use central difference approximation for the LHS:

$$\frac{\partial^2 \phi}{\partial x^2} = \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2} = \rho_{e,i} \longrightarrow \phi_{i+1} - 2\phi_i + \phi_{i-1} = \Delta x^2 \rho_{e,i}$$



So we can write a series of algebra equations:

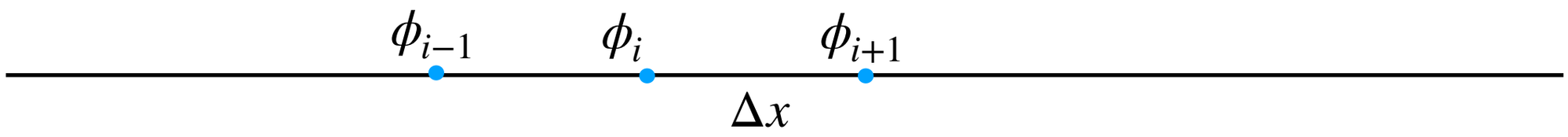
$$\phi_3 - 2\phi_2 + \phi_1 = -\Delta x^2 \rho_{e,2}$$

$$\phi_4 - 2\phi_3 + \phi_2 = -\Delta x^2 \rho_{e,3}$$

...

$$\phi_N - 2\phi_{N-1} + \phi_{N-2} = -\Delta x^2 \rho_{e,N-1}$$

The Poisson Equation: 1D Diffusion



So we can write a series of algebra equations:

$$\phi_3 - 2\phi_2 + \phi_1 = -\Delta x^2 \rho_{e,2}$$

$$\phi_4 - 2\phi_3 + \phi_2 = -\Delta x^2 \rho_{e,3}$$

...

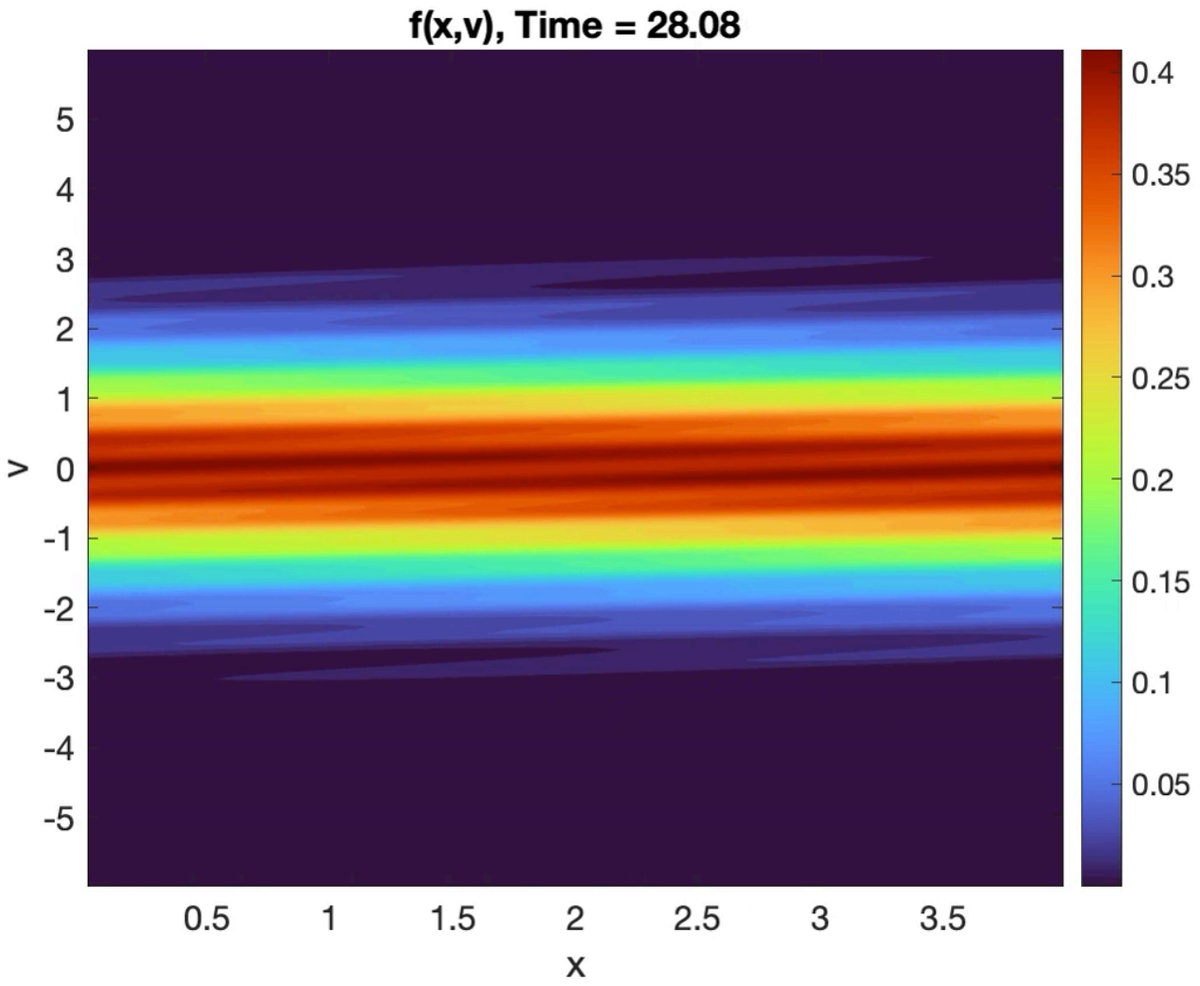
$$\phi_N - 2\phi_{N-1} + \phi_{N-2} = -\Delta x^2 \rho_{e,N-1}$$

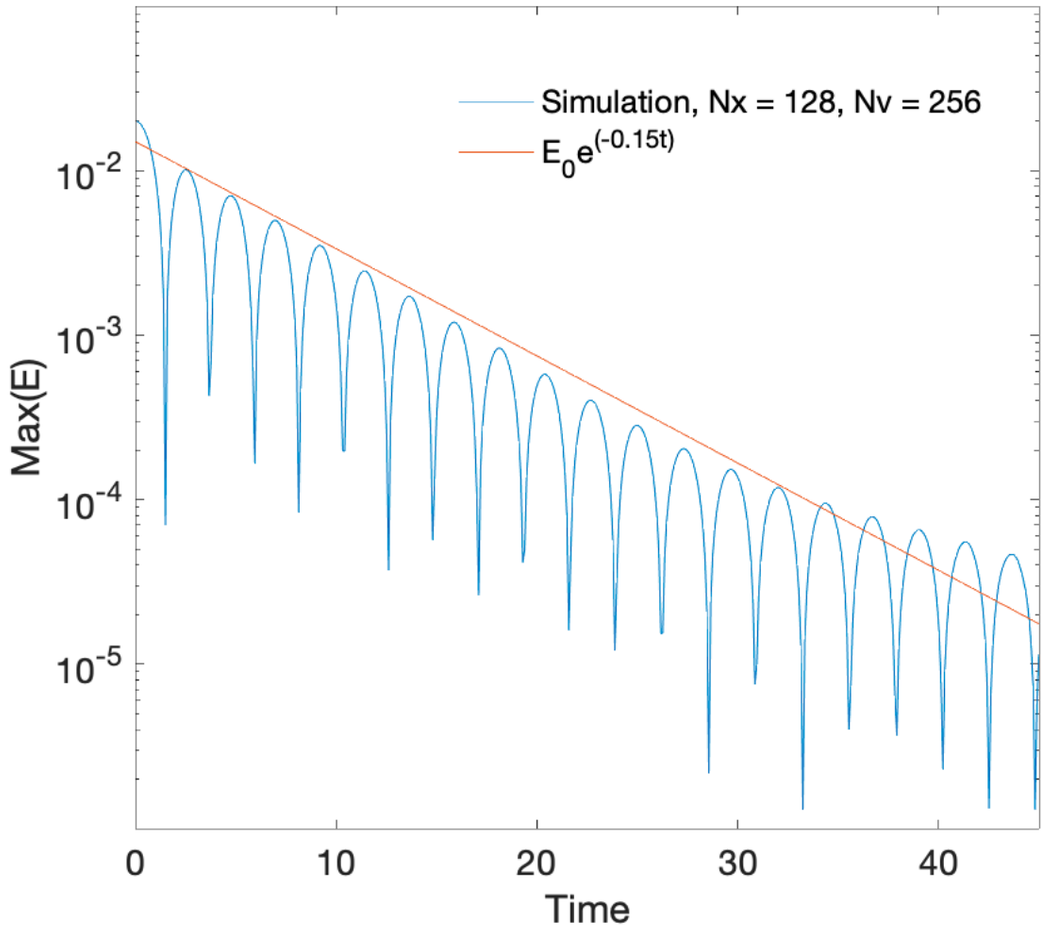
$$\Phi = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_N \end{bmatrix} \quad T = \begin{bmatrix} 2 & -1 & & & & \text{-1} \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ \text{-1} & & & & -1 & 2 \end{bmatrix} \quad b = \begin{bmatrix} \Delta x^2 \rho_{e,1} \\ \Delta x^2 \rho_{e,2} \\ \Delta x^2 \rho_{e,3} \\ \vdots \\ \Delta x^2 \rho_{e,N} \end{bmatrix} \longrightarrow \quad T \cdot \Phi = b$$

Boundary condition

Landau Damping

Simulation results





$$f(v,x)\Big|_{t=0} = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}} \left[1 + A\cos(kx)\right]$$

$$A = 0.05, \;\; k = 0.5$$

```
76      % 3. Initial conditions
77      % Landau Damping (x - configuration space, y - velocity space)
78 -    A = 0.05;
79 -    k = 0.5;
80 -    rho = 1/sqrt(2*pi).*exp(-yc.^2/2).*(1+A.*cos(k.*xc));
```

$$Nx = 128, \;\; Nv = 256$$

```
7       % 1. generate a 2-D cartesian grid
8 -     nx = 128; % # of active cells in x space
9 -     ny = 256; % # of active cells in v space
```

The Equations of Plasma Simulations

In fluid simulations, we use the following equations

$$\frac{\partial}{\partial t}(n_\alpha m_\alpha) + \nabla \cdot (n_\alpha \mathbf{u}_\alpha) = S_\alpha$$

$$\frac{\partial}{\partial t}(n_\alpha \mathbf{u}_\alpha) + \nabla \cdot (n_\alpha \mathbf{u}_\alpha \mathbf{u}_\alpha + \mathbf{P}_\alpha) - \frac{n_\alpha q_\alpha}{m_\alpha}(\mathbf{E} + \mathbf{u}_\alpha \times \mathbf{B}) = \mathbf{R}_\alpha \qquad \textbf{+ Maxwell's equations}$$

$$\frac{\partial \mathscr{E}_\alpha}{\partial t} + \nabla \cdot \mathbf{u}_\alpha \left(\mathscr{E}_\alpha + p_\alpha\right) - \mathbf{u}_\alpha \cdot \mathbf{J} \times \mathbf{B} = 0$$

In Vlasov simulations, we use the following equations

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} + E\frac{\partial f}{\partial v} = 0 \qquad \textbf{+ Maxwell's equations}$$

In particle simulations, we use the following equations

$$\frac{d\mathbf{x}_p}{\partial t} = \mathbf{v}_p \qquad \frac{d\mathbf{v}_p}{\partial t} = \mathbf{E}_p \qquad \textbf{+ Maxwell's equations}$$

The Equations of Particle Simulations

The classical or relativistic description of the natural world is based on describing the interaction of elements of matter via force fields.

In the case of a plasma, the system is composed by charged particles (for example negative electrons and positive ions) interacting via electric and magnetic fields.

If we identify each particle with a label p and their charge with q_p, position with x_p, position with v_p, the force acting on the particles is the combination of the electric and magnetic (Lorentz) force:

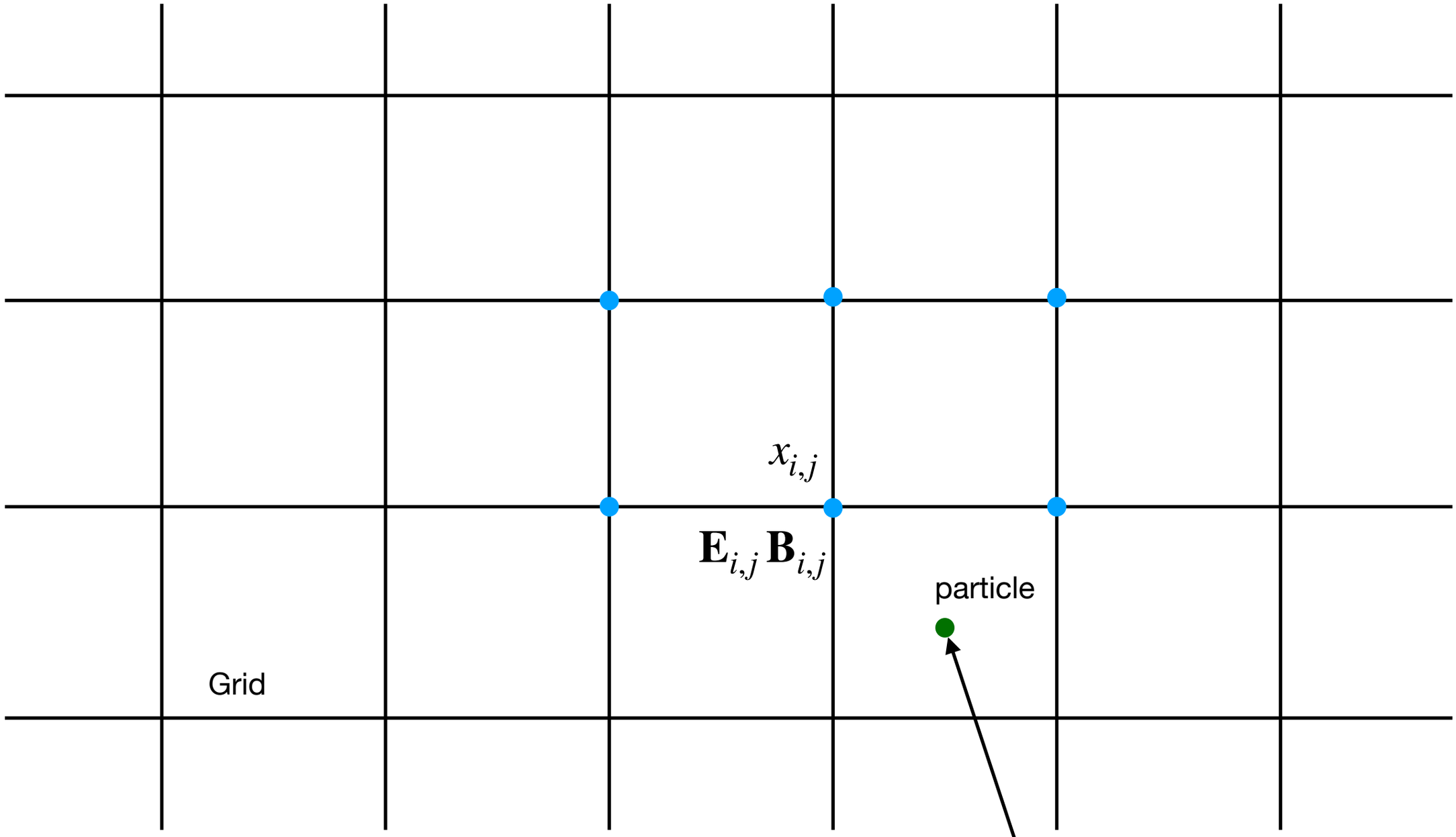
$$\mathbf{F}_p = q_p \mathbf{E}(\mathbf{x}_p) + \mathbf{v}_p \times \mathbf{B}(\mathbf{x}_p)$$

Electric force **Magnetic force**

The electric and magnetic fields are themselves created by the particles in the system and by additional sources outside the system

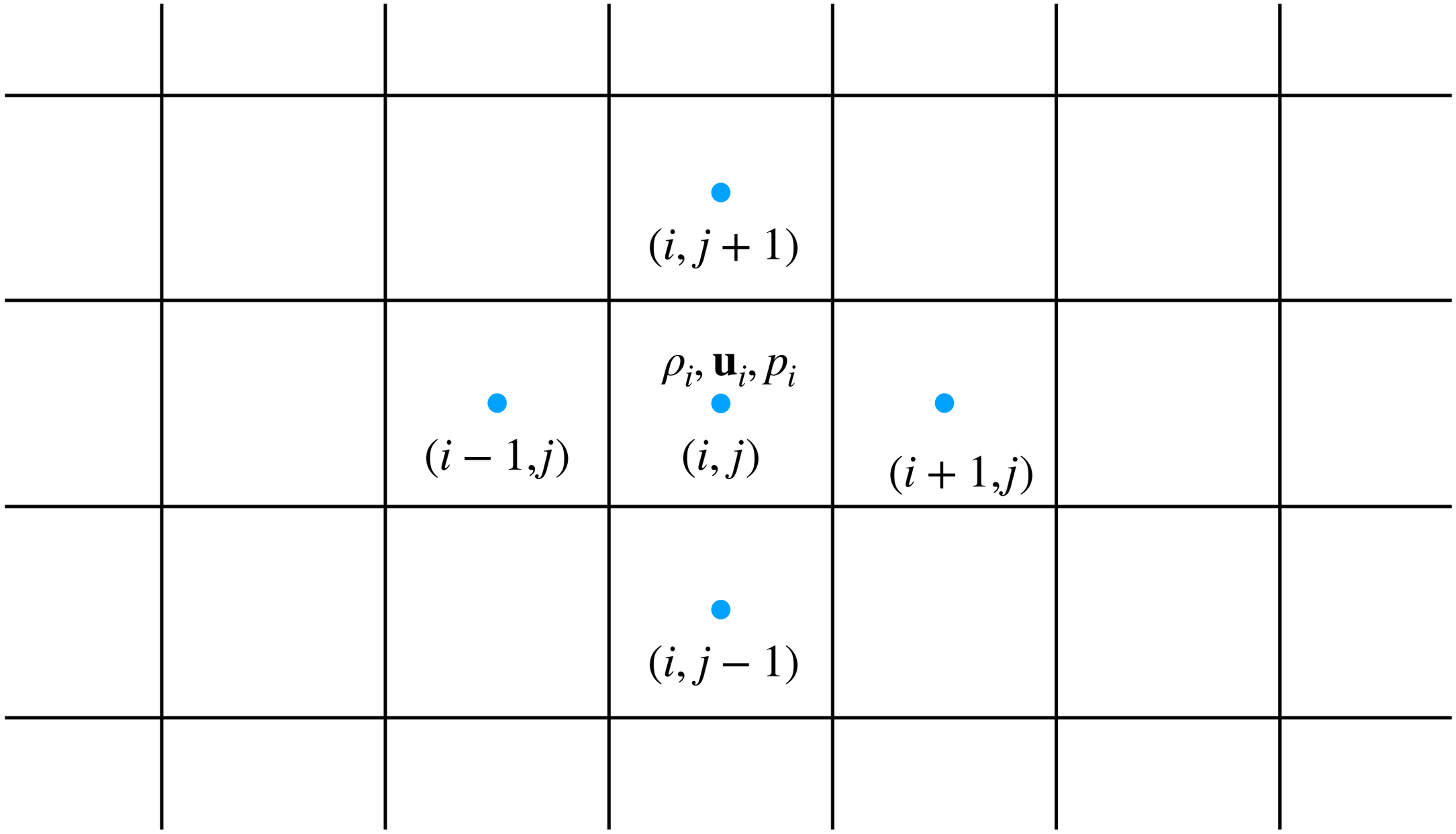
$$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon} \qquad \nabla \cdot \mathbf{B} = 0 \qquad \frac{\partial \mathbf{B}}{\partial \mathbf{t}} + \nabla \times \mathbf{E} = 0 \qquad \frac{\partial \mathbf{E}}{\partial \mathbf{t}} - \nabla \times \mathbf{B} = \mu \mathbf{J}$$

The Equations of Particle Simulations



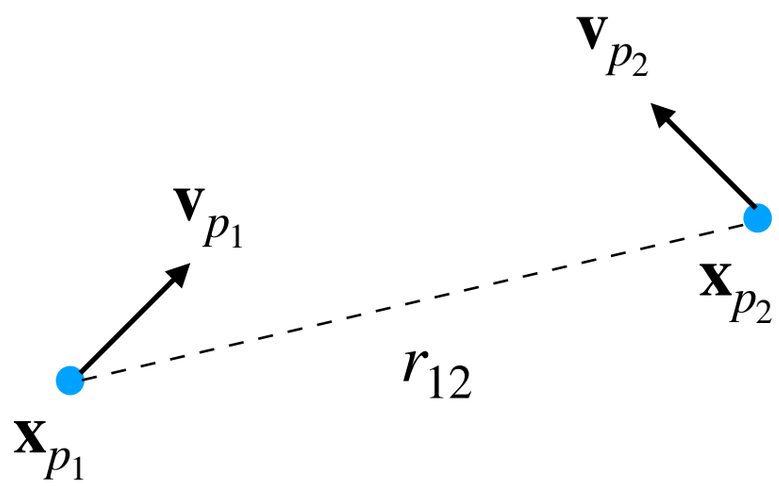
Equations $\dfrac{d\mathbf{x_p}}{dt} = \mathbf{v}_p$ $\mathbf{F}_p = m_p \dfrac{d\mathbf{v_p}}{dt}$

Fluid Simulations



- **Plasma parameters: density, velocity, pressure**
- **Fixed to grid, based on finite-difference or finite volume**
- **Thermal equilibrium**

Particle Simulations (direct)



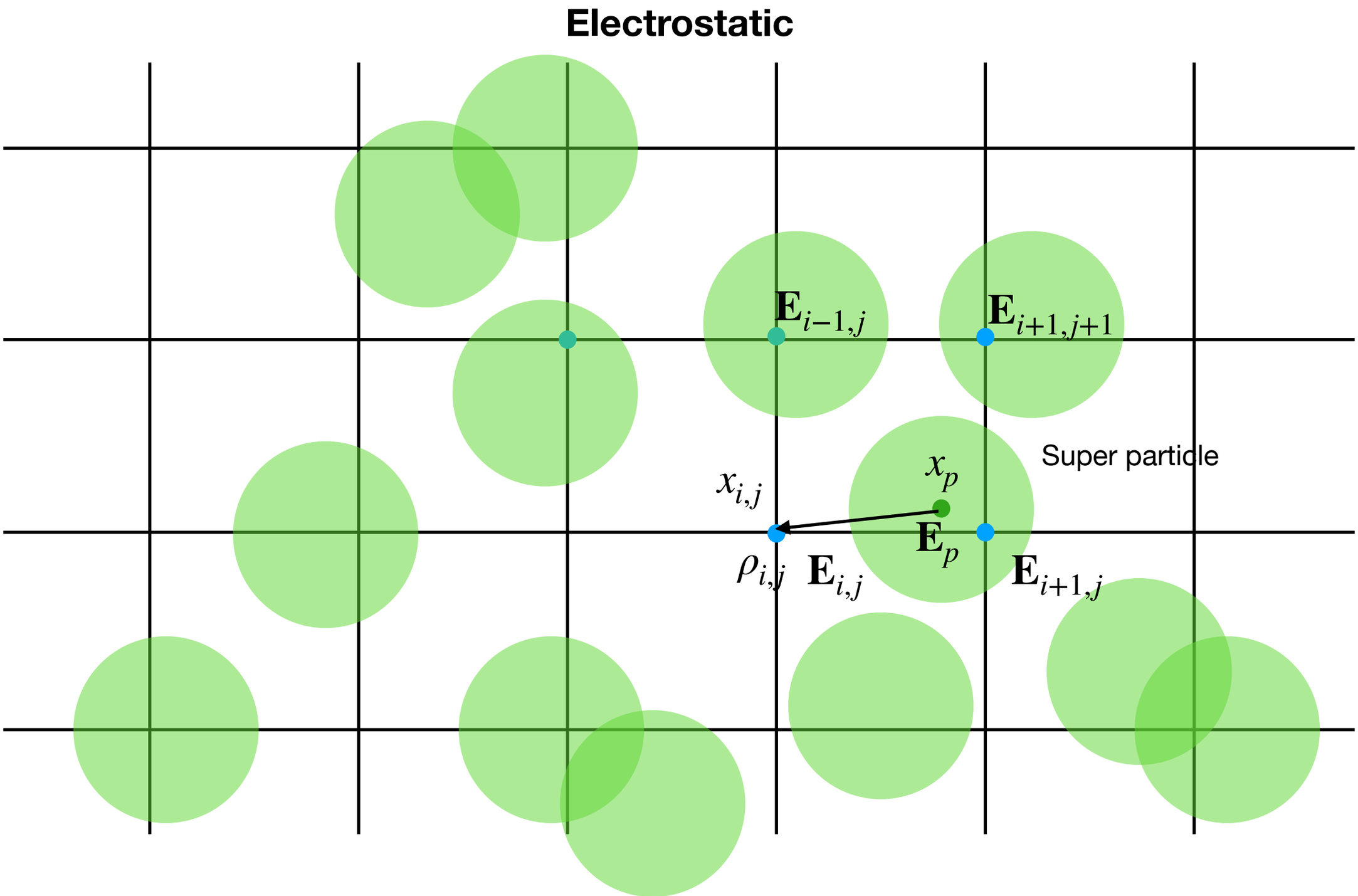
So the force between particles 1 and 2 is calculated as $\mathbf{F}_{12} = \frac{1}{\epsilon_0} \cdot \frac{q_1 q_2}{r_{12}^2} \hat{\mathbf{r}}_{12}$

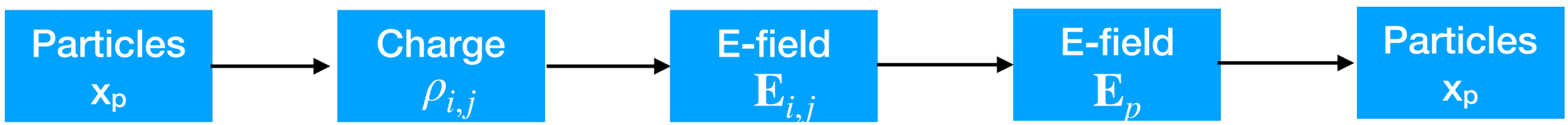
The total force on particle 1 is $\mathbf{F}_1 = \frac{1}{\epsilon_0} \sum_j^N \frac{q_1 q_j}{r_{1j}^2} \hat{\mathbf{r}}_{j2}$

This is the most “realistic” simulation of a plasma system! But it’s not feasible

Operations $\sim N^2$

Particle Simulations (super particles)





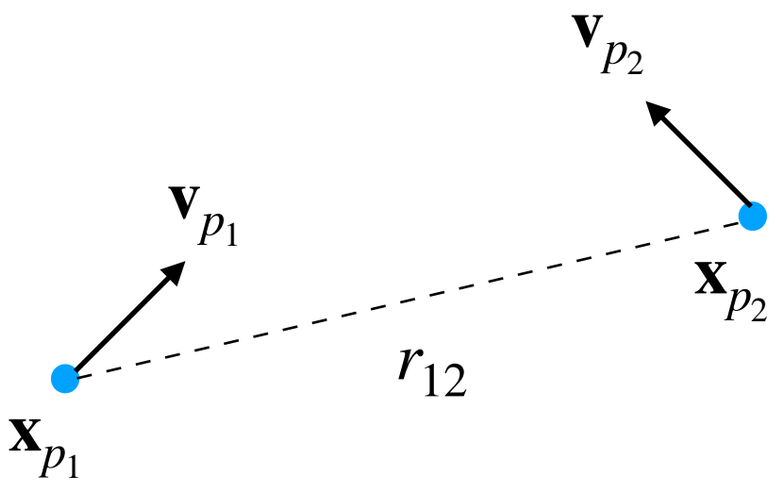
A Paradox

In the equations for PIC, why there's no interactions between "particles"?

$$\frac{d\mathbf{x}_p}{\partial t} = \mathbf{v}_p$$

$$\frac{d\mathbf{v}_p}{\partial t} = \frac{q}{m}\mathbf{E}_p$$

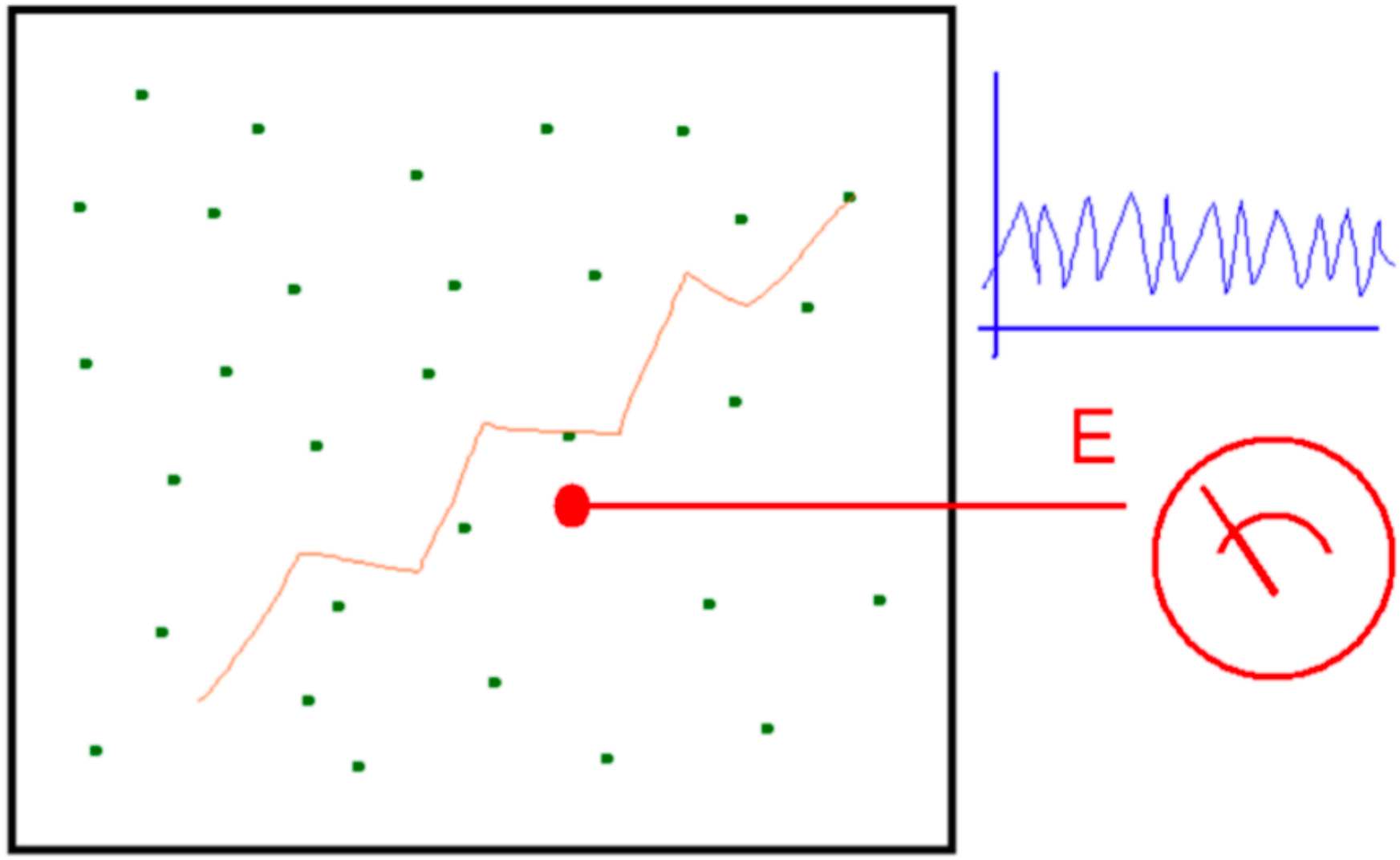
$$-\epsilon_0 \frac{\partial^2 \phi}{\partial x^2} = \rho$$



Where did the Coulomb force go?

Types of Interacting Systems

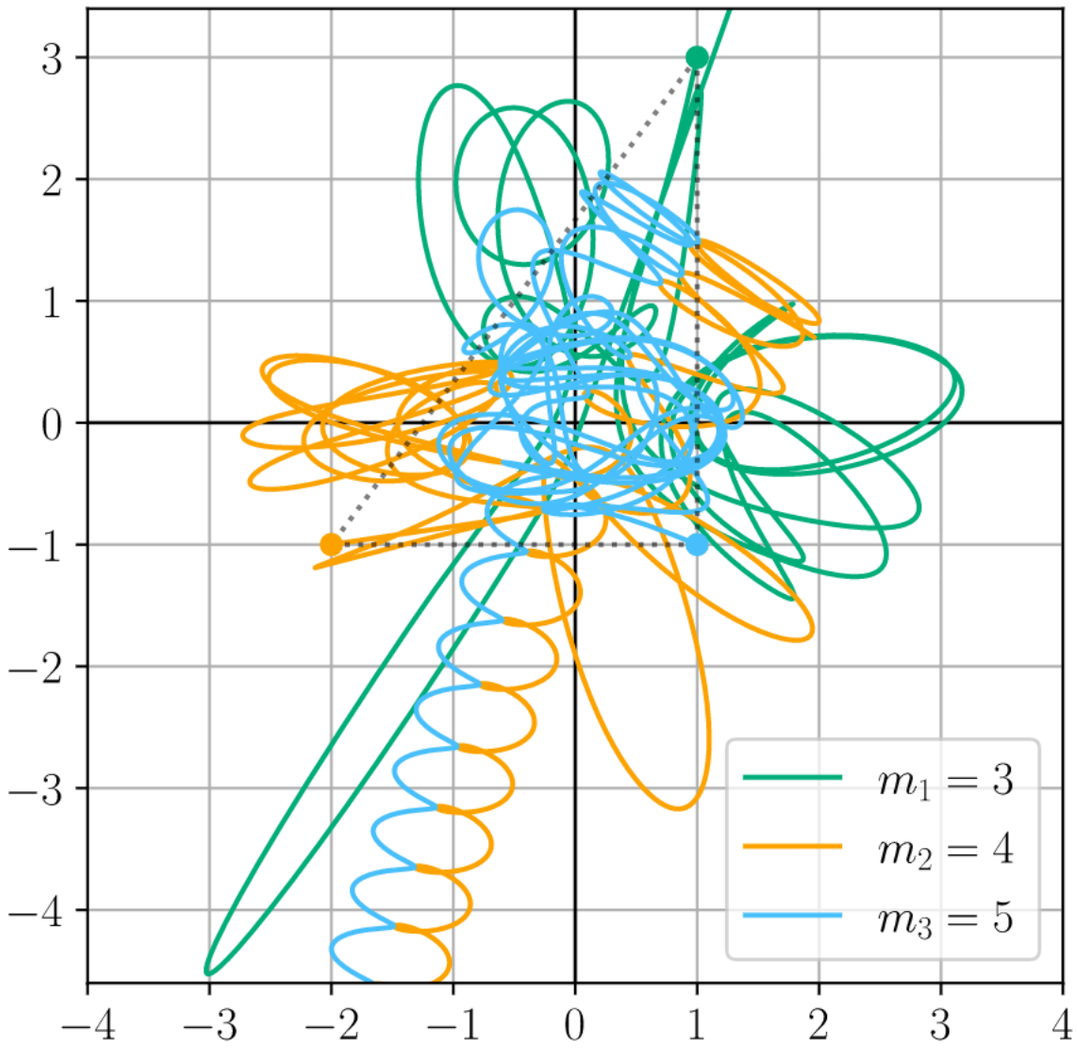
Strongly coupled systems



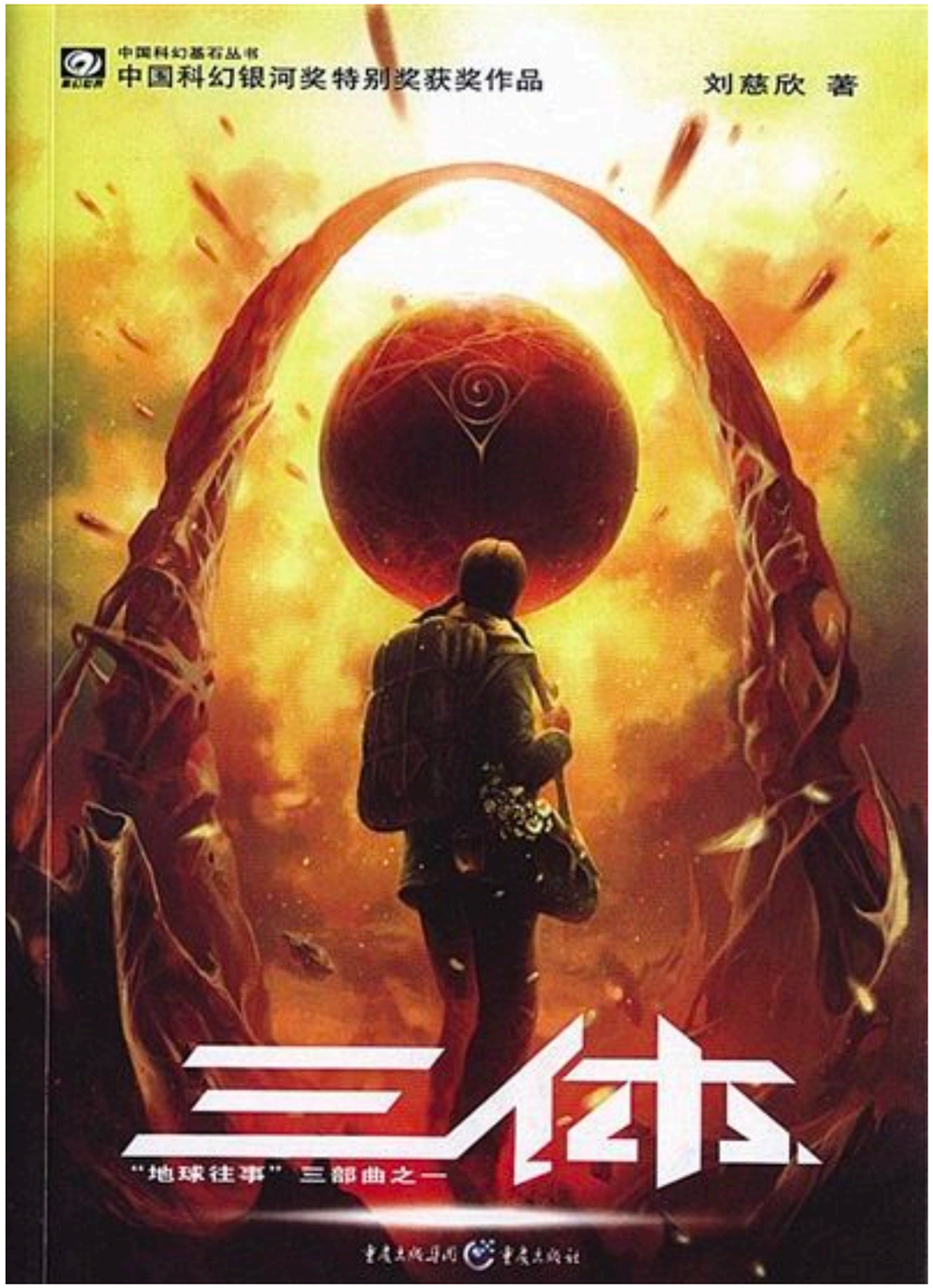
- E field is determined by all the particles in the system
- At a given time, very few particles near the detector
- Individual particle behavior matters to the E measurements

Types of Interacting Systems

An Example of Strongly coupled systems

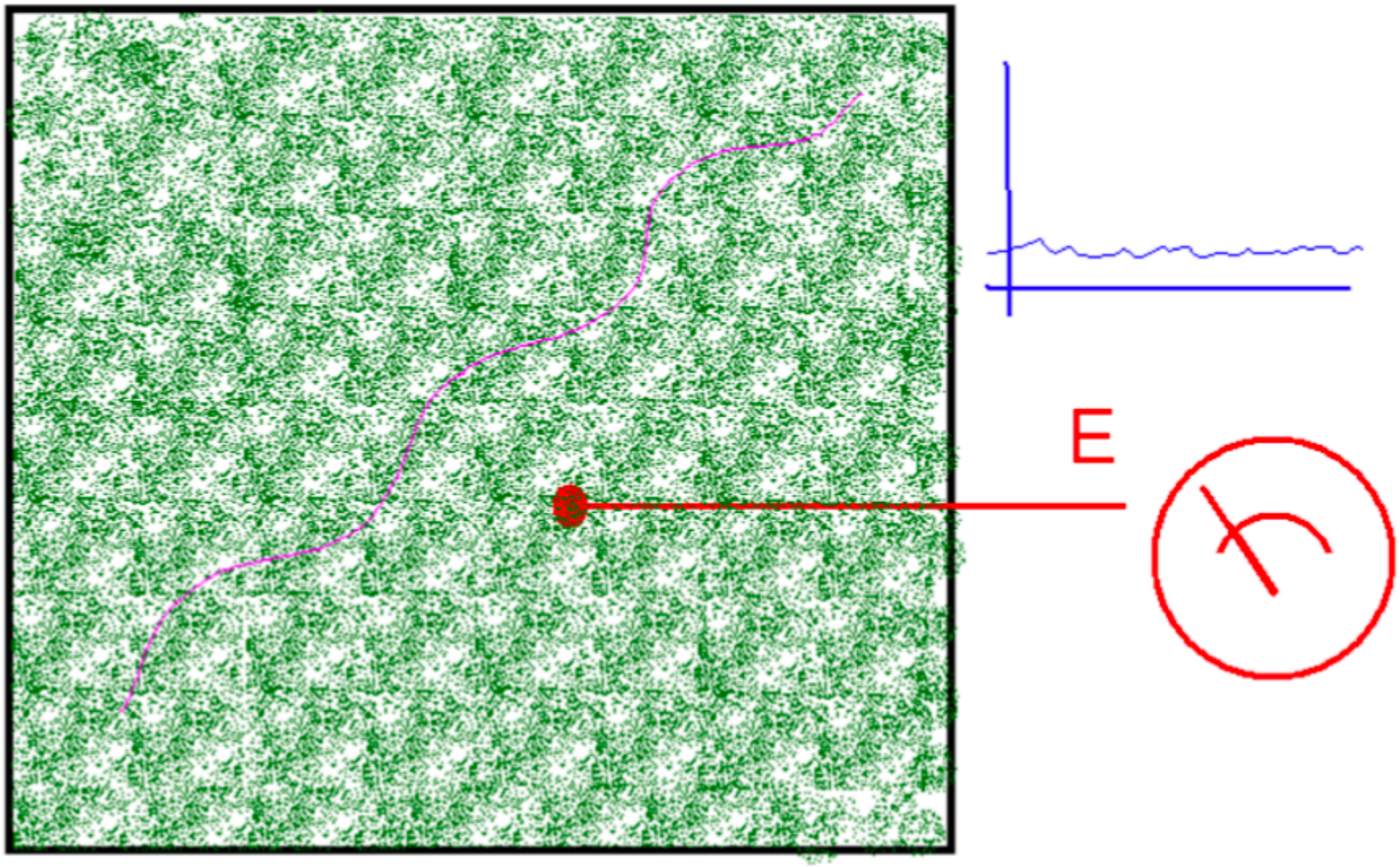


不要回答 x 3



Types of Interacting Systems

Weakly coupled systems



- E field is also determined by all the particles in the system - collective
- At a given time, extremely large number of particles near the detector
- Individual particle behavior does NOT matter to the E measurements

Description of Interacting Systems

The plasma coupling parameter

If we choose the conventional box with side equal to the Debye length, the number of particles present is

$$N_D = n\lambda_D^3$$

volume associated with each particle is n^{-1}

Average interparticle distance $a = n^{-1/3}$ - The particles are distributed randomly and their distances are also random, but on average the interparticle distance is a.

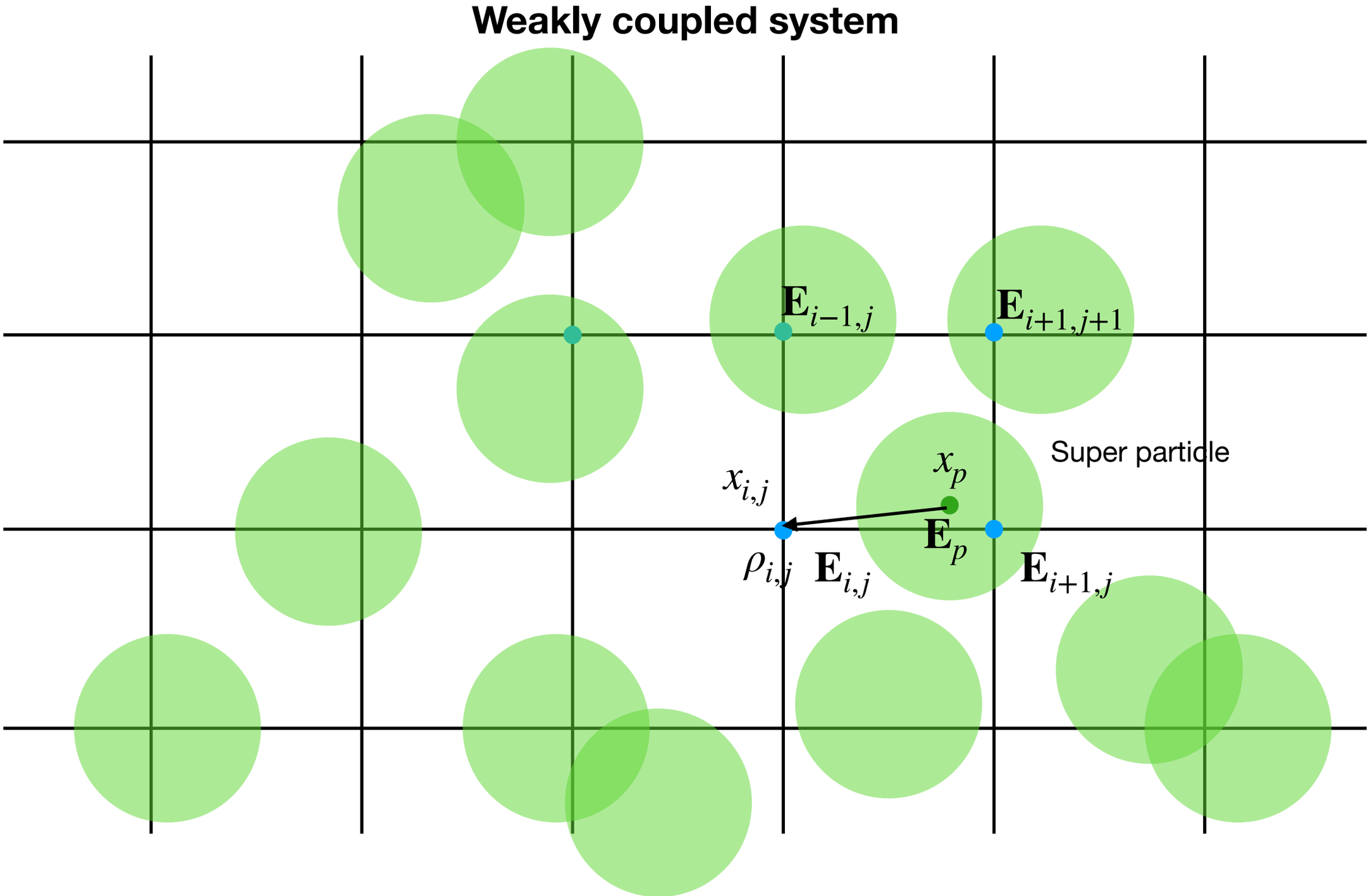
The electrostatic potential energy between two charged particles with distance a is

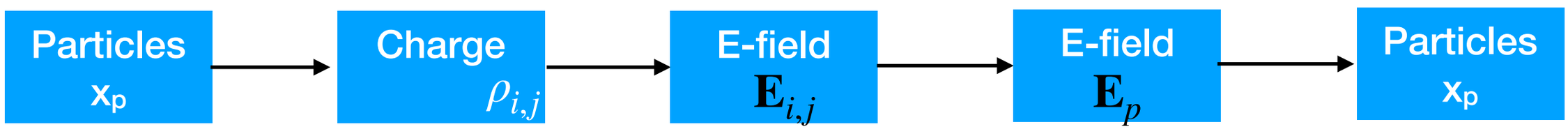
$$E_{pot} = \frac{q^2}{4\pi\epsilon_0 a}$$

So if we compare the thermal energy with the electrostatic potential energy:

$$\frac{E_{th}}{E_{pot}} = \frac{4\pi\epsilon_0 akT}{q^2} \xrightarrow[Debye]{\lambda_D = \sqrt{\epsilon_0 kT/ne^2}} \frac{E_{th}}{E_{pot}} = \frac{4\pi\epsilon_0 akT}{q^2 n^{1/3}} = \boxed{4\pi N_D^{2/3} \equiv \Lambda}$$

The Particle-in-cell (PIC) Method





Mathematical Derivation of the PIC method

the phase space distribution function $f_S(x, v, t)$ for a given species s (electrons or ions), defined as the number density per unit element of the phase space (or the probability of finding a particle in a dx and dv around a certain phase space point (x, v)), is governed by the Vlasov equation:

$$\frac{\partial f_s}{\partial t} + v\frac{\partial f_s}{\partial x} + \frac{q_s E}{m_s}\frac{\partial f_s}{\partial v} = 0$$

The electric field in the electrostatic limit is described by the Poisson's equation for the scalar potential:

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

where the net charge density is computed from the distribution functions as:

$$\rho(x,t) = \sum_s q_s \int f_s(x,t,v)dv$$

Mathematical Derivation of the PIC method

The PIC method can be regarded as a *finite element approach* but with finite elements that are themselves moving and overlapping.

Mathematically, the PIC method is obtained by assuming that the distribution function of each species is given by the superposition of several elements (called computational particles or superparticles):

$$f_s(x, v, t) = \sum_p f_p(x, v, t)$$

The PIC method is based upon assigning to each computational particle a specific functional form for its distribution, a functional form with a number of free parameters whose time evolution will determine the numerical solution of the Vlasov equation:

$$f_p(x, v, t) = N_p S_x(x - x_p(t)) S_v(v - v_p(t))$$

number of physical particles

Shape functions

PIC simulations are basically a discrete form (finite element) of Vlasov simulations

Mathematical Derivation of the PIC method

Shape functions

Properties:

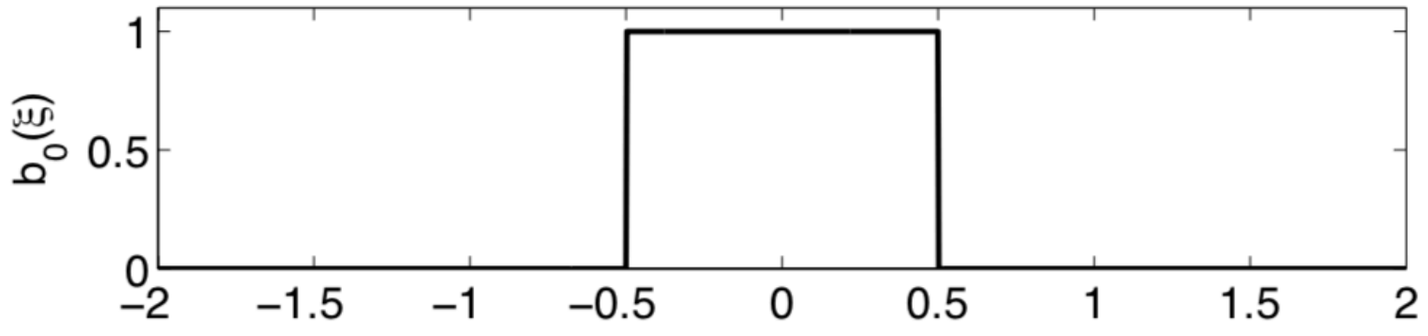
- **Compact - zero outside a small range**
- **Unitary -**

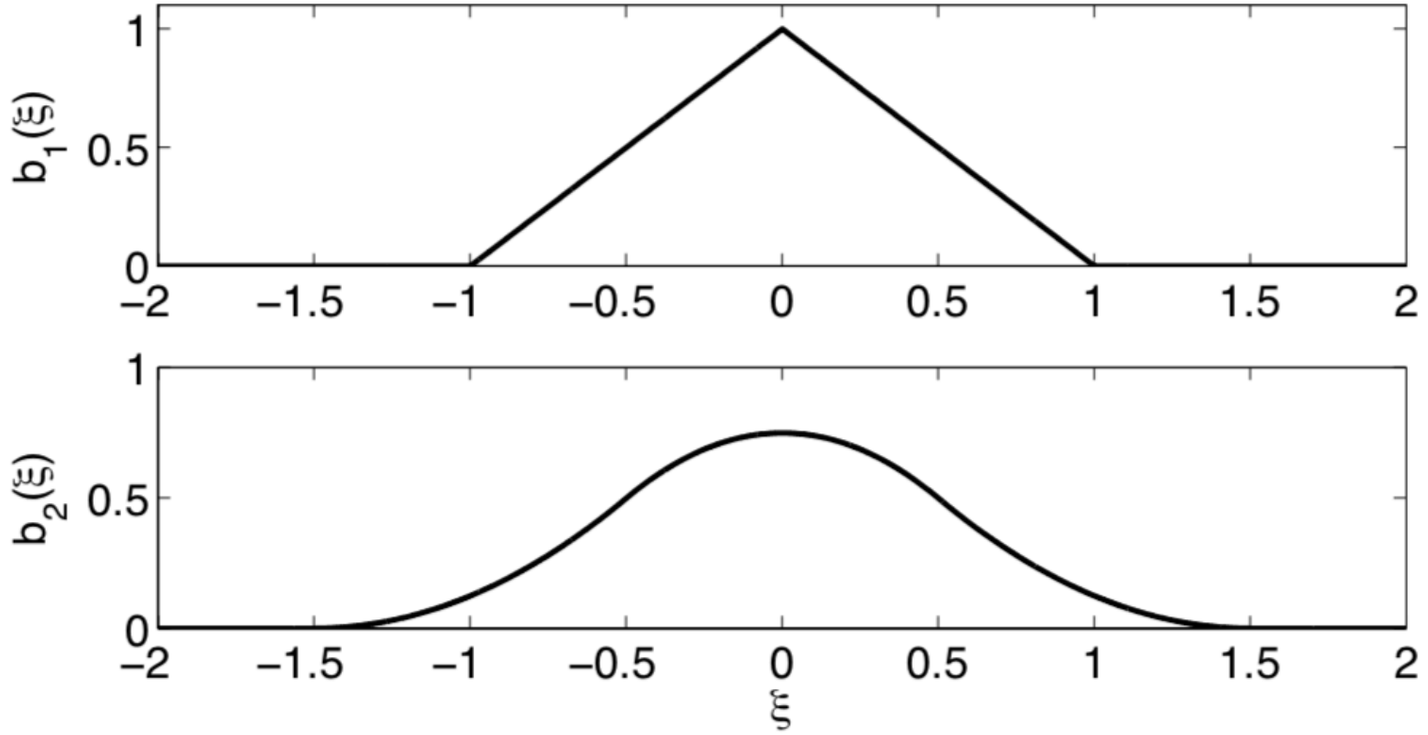
$$\int_{-\infty}^{+\infty} S_x(x - x_p)dx = 1$$

- **Symmetry -**

$$S_x(x - x_p) = S_x(x_p - x)$$

Typical shape functions





$$\xi = x - x_p$$

Mathematical Derivation of the PIC method

From Vlasov to Equation of Motion

Now let's look at how the location $\mathbf{x}_p$ and velocity $\mathbf{v}_p$ of these super particles evolve based on the Vlasov equation

$$\frac{\partial f_p}{\partial t} + v\frac{\partial f_p}{\partial x} + \frac{q_s E}{m_s}\frac{\partial f_p}{\partial v} = 0$$

The **zeroth moment** integral of the Vlasov equation:

Symmetry Compact

$$\frac{\partial \langle f_s \rangle}{\partial t} + \left\langle v\frac{\partial f_s}{\partial x} \right\rangle + \left\langle \frac{q_s E}{m_s}\frac{\partial f_s}{\partial v} \right\rangle = 0$$

N_p

0 0

Recall:

$$\langle \chi \cdot f_s \rangle = \int_{-\infty}^{+\infty} dv \int_{-\infty}^{+\infty} \chi f_s(x, v) dx$$

The zeroth moment integral becomes:

$$\frac{dN_p}{dt} = 0$$

$$f \sim N_p S_x(x - x_p) S_v(v - v_p)$$

The application of the first zeroth order moment leads to the establishment of the conservation of the number of physical particles per computational particle (boring but it's a must).

Mathematical Derivation of the PIC method

From Vlasov to Equation of Motion

The x- moment integral of the Vlasov equation: $\langle x \cdot Vlasov \rangle$

$$\frac{\partial \langle f_s x \rangle}{\partial t} + \left\langle v x \frac{\partial f_s}{\partial x} \right\rangle + \left\langle x \frac{q_s E}{m_s} \frac{\partial f_s}{\partial v} \right\rangle = 0$$

0 Integrate over v

1 2

1 $\langle f_s x \rangle = N_p \underbrace{\int S_v(v - v_p) dv}_{1} \int x S(x - x_p) dx = N_p x_p$

2 $\left\langle v x \frac{\partial f_s}{\partial x} \right\rangle = \int v dv \int x \frac{\partial f_p}{\partial x} dx = \int v [f_p(x = -\infty) - f_p(x = +\infty)] x dv - \int v f dx dv = - \langle f_p v_p \rangle$

So we get:

$$\frac{dx_p}{dt} = v_p$$

$$\langle f_p v_p \rangle = N_p v_p$$

Mathematical Derivation of the PIC method

From Vlasov to Equation of Motion

The v- moment integral of the Vlasov equation: $\langle v \cdot Vlasov \rangle$

$$\frac{\partial \langle f_s v \rangle}{\partial t} + \left\langle v^2 \frac{\partial f_s}{\partial x} \right\rangle + \left\langle v \frac{q_s E}{m_s} \frac{\partial f_s}{\partial v} \right\rangle = 0$$

0 Integrate over x

$\langle f_p v \rangle = N_p v_p$ (1) (2)

(2) $\left\langle v \frac{q_s E}{m_s} \frac{\partial f_s}{\partial v} \right\rangle = \int \frac{q_s E}{m_s} dx \int v \frac{\partial f_s}{\partial v} dv$ Integrate by parts $= -\int \frac{q_s E}{m_s} dx \int f_s dv = -\left\langle \frac{q_s E}{m_s} f_p \right\rangle \equiv -E_p$

The remaining integral defines a new important quantity, the average electric field acting on a computational particle, Ep :

$$E_p = \int S_v(v - v_p) dv \int E(x) S_x(x - x_p) dx = \int E(x) S_x(x - x_p) dx$$

So we get: $\frac{dv_p}{dt} = E_p$ (normalized, q/m = 1)

Mathematical Derivation of the PIC method

Equation of Motion for computational particles

The moment integrals of the Vlasov equation give the following complete set of evolution equations for the parameters defining the functional dependence of the distribution within each "particle":

$$\frac{dN_p}{dt} = 0$$

$$\frac{dx_p}{dt} = v_p$$

$$\frac{dv_p}{dt} = \frac{q_s}{m_s} E_p$$

It is a crucial advantage of the PIC method that its evolution equations resemble the same Newton equation as followed by the regular physical particles!

Naturally, the electric field is itself given by Maxwell's equations which in turn need the charge density (and for complete models also the current density).

$$\rho(x,t) = \sum_p q_s \int f_p(x,t,v) dv = \sum_p q_s N_p S_x(x - x_p)$$

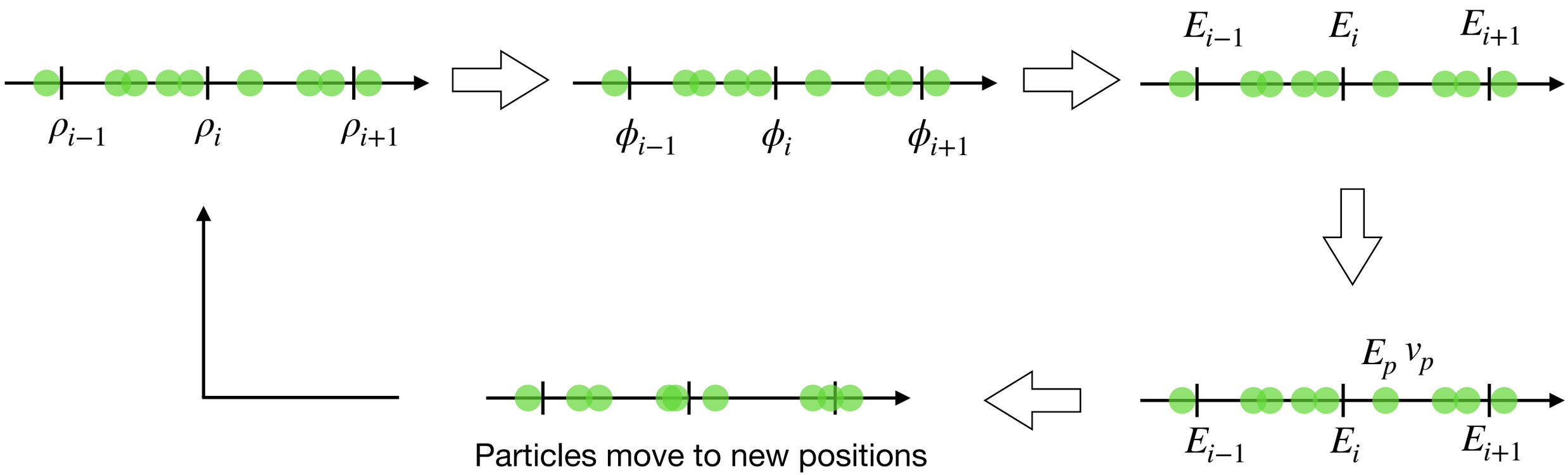
Mathematical Derivation of the PIC method

Field Equations

The solution of the field equations can be done with a wide variety of methods. The majority of the existing PIC methods relies on finite difference or finite volume,

$$\epsilon_0 \frac{\partial^2 \phi}{\partial x^2} = -\rho \longrightarrow \epsilon_0 \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2} = -\rho_i \longrightarrow E_i = \frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x}$$

When E_i is calculated, we can use that to get the E(x_p) where particle is located (x_p):

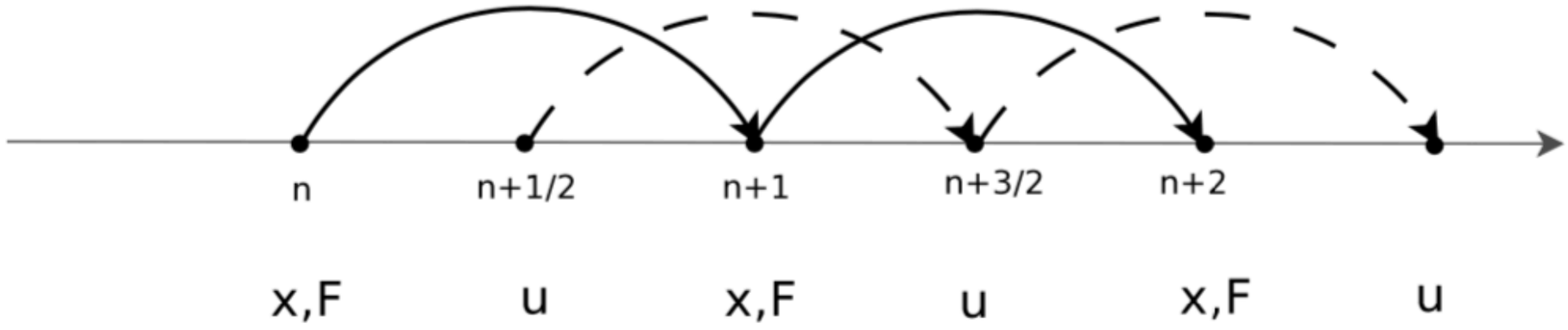


Discretization of the PIC method

The simplest algorithm and by far the most used in the so-called *leap-frog algorithm* based on staggering the time levels of the velocity and position by half time step:

$$x_p(t = n\Delta t) = x_p^n$$

$$v_p(t = n + \frac{1}{2}\Delta t) = v_p^{n+\frac{1}{2}}$$



The numerical scheme is summarized by:

$$x_p^{n+1} = x_p^n + \Delta t v_p^{n+\frac{1}{2}}$$

$$v_p^{n+\frac{3}{2}} = v_p^{n+\frac{1}{2}} + \Delta t \frac{q_s}{m_s} E_p(x_p^{n+1})$$

Half tilmestep

$$\frac{dx_p}{dt} = v_p$$

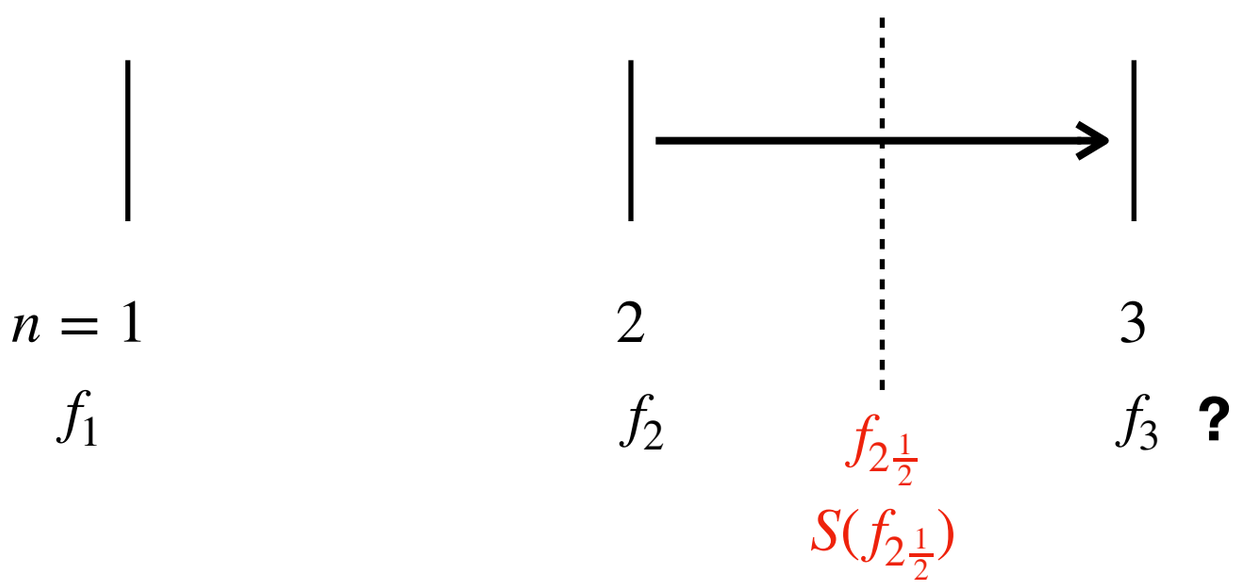
$$\frac{dv_p}{dt} = \frac{q_s}{m_s} E_p$$

Recall - Leapfrog Trapezoidal

There are many choices of getting a viable second order time stepping method, we will use a kind of “predictor - corrector” scheme named “*Leapfrog Trapezoidal*” (LT) method in the following discussions, which uses two sup-steps for the time evolution

Assume that f is known at t1 and t2, corresponding to n=1 and n=2, we want to calculate f at t3 using the differential equation

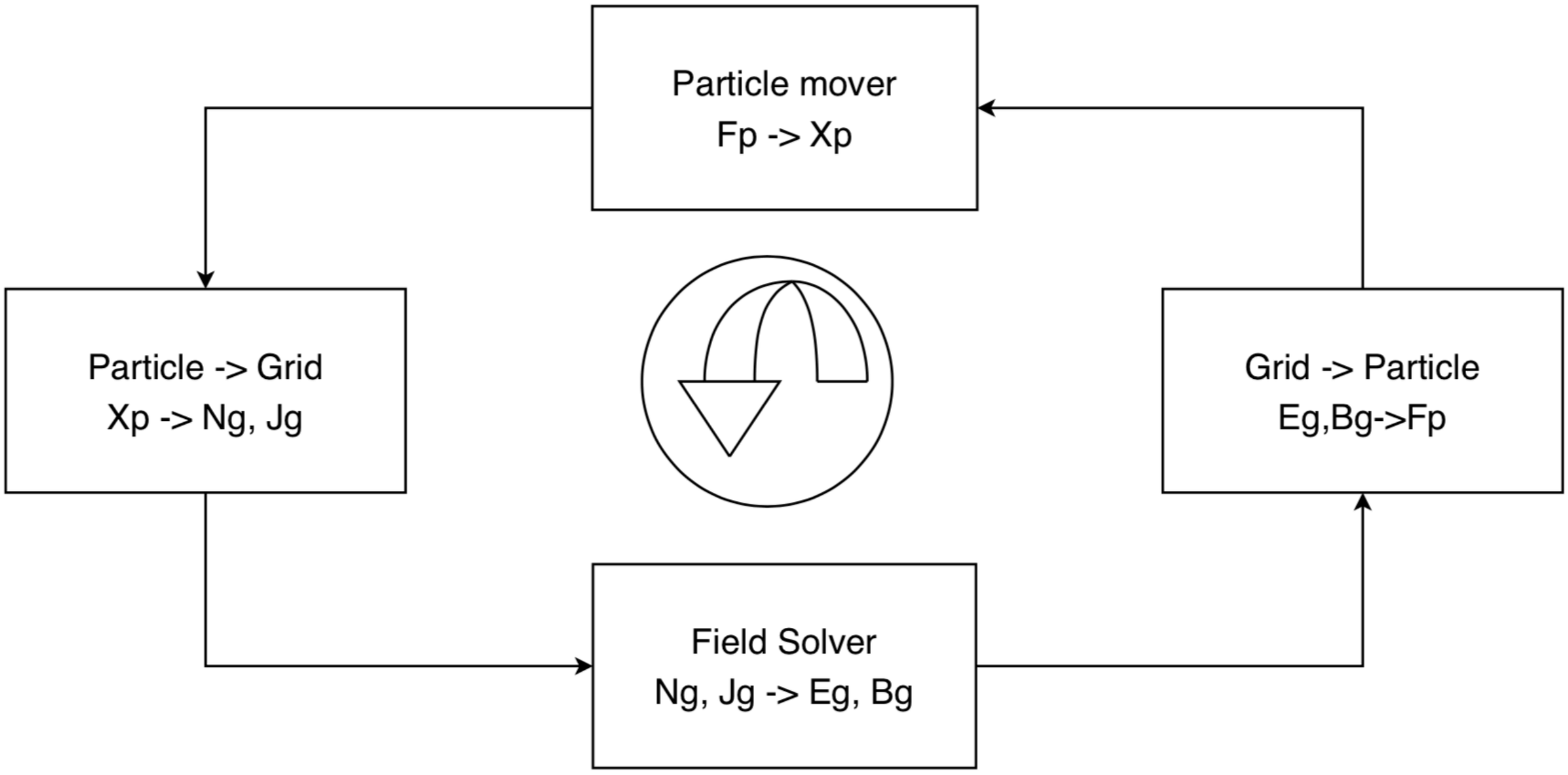
$$\frac{\partial f}{\partial t} = S(f)$$



- Going from f2 to f3 using f2 and S(f2) gives the first order time stepping
- If we know f2.5 and S(f2.5), we get something like a central difference scheme for f3:

$$\frac{f_i^{n+1} - f_i^n}{\Delta t} = S(f_i^{n+\frac{1}{2}}) \quad \Longrightarrow \quad f_i^{n+1} = f_i^n + \Delta t S(f_i^{n+\frac{1}{2}})$$

Flow chart of the PIC method



Summary (algorithm) of the PIC method

Electrostatic, 1D plasmas

1. The plasma is described by a number of computational particles having position xp, velocity vp and each representing a fixed number Np of physical particles

2. The equations of motion for the particles are advanced by one time step using:

$$x_p^{n+1} = x_p^n + \Delta t v_p^{n+\frac{1}{2}}$$

$$v_p^{n+\frac{3}{2}} = v_p^{n+\frac{1}{2}} + \Delta t \frac{q_s}{m_s} E_p(x_p^{n+1})$$

3. The charge density is computed in each cell: $\rho_i = \sum_p q_s S_x(x - x_p)$

4. Solve the Poisson equation for the electric potential: $\epsilon_0 \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2} = -\rho_i$

   Compute the electric field in each cell: $E_i = \frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x}$

5. From the field known in the cells, the field acting on the particles is computed as

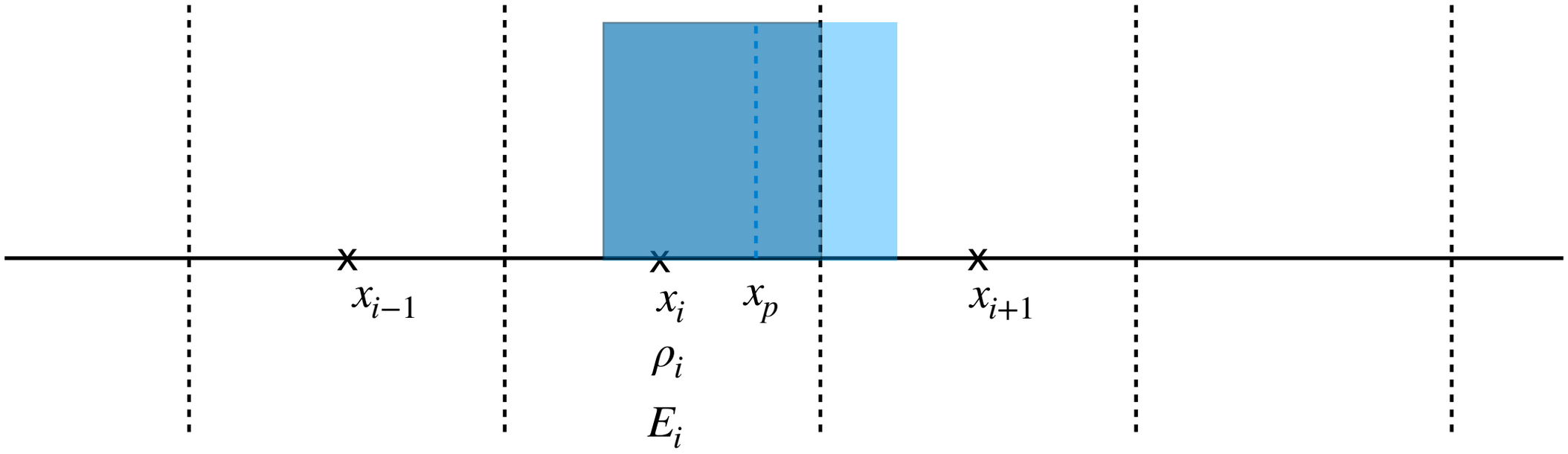
$$E_p = \sum_i E_i S_x(x_i - x_p) dx$$

6. Goto step 2

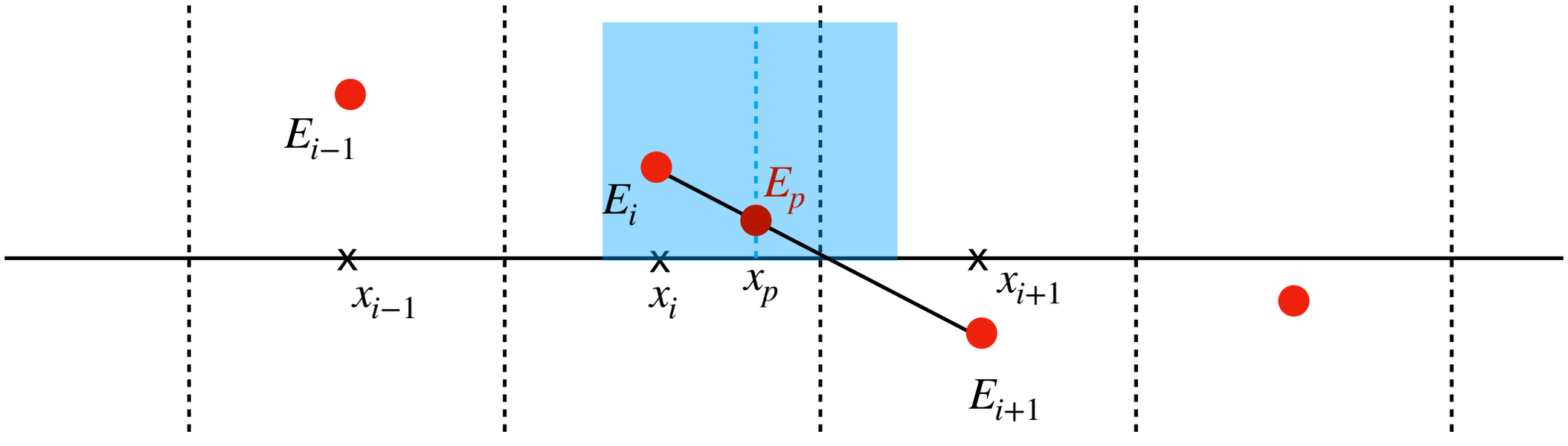
Summary (algorithm) of the PIC method

Electrostatic, 1D plasmas

Distribute charge to grid, calculate E field at grid

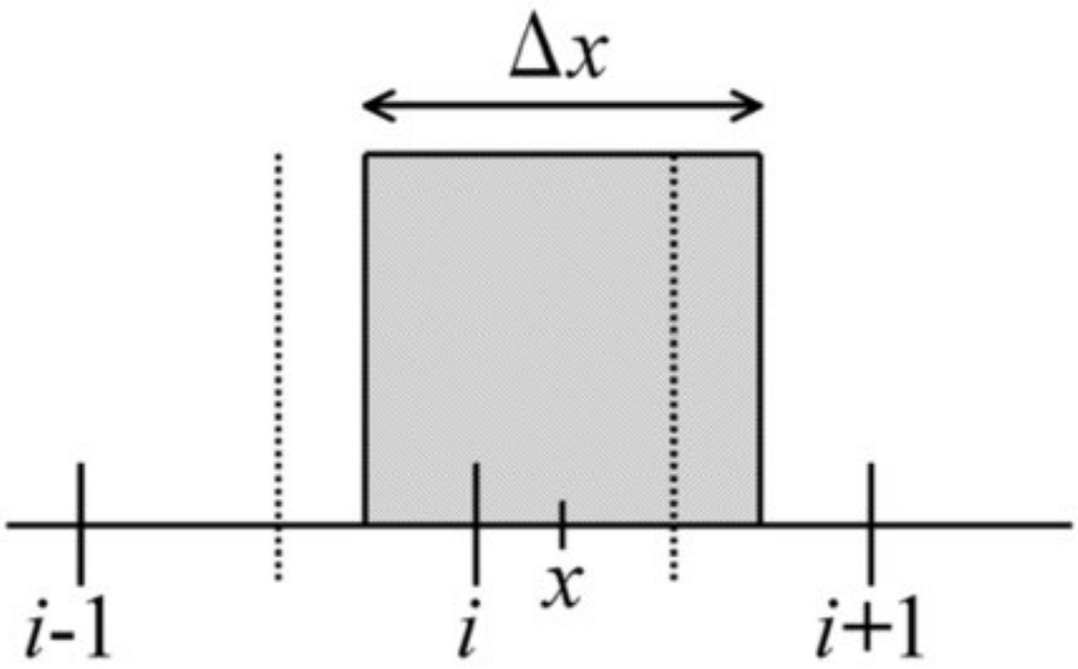


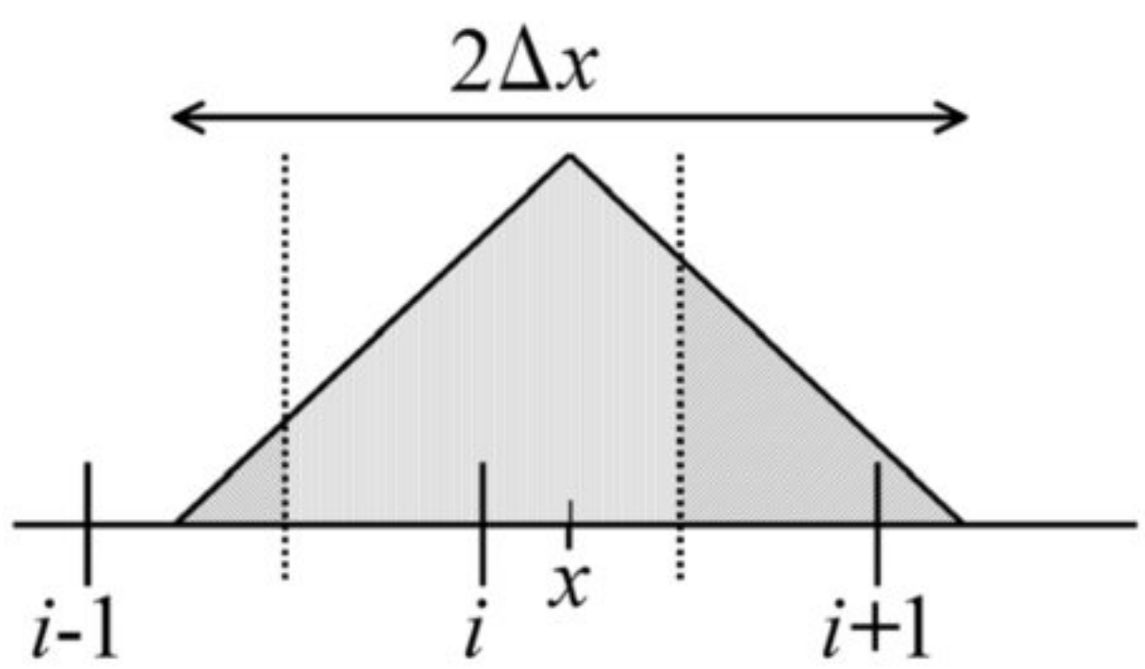
Reconstruct (interpolate) E field at particle location



The Particle-in-cell (PIC) Method

Higher than first order - shape functions





Practical considerations of the PIC method

Stability

The first thing to consider in running a PIC code is a proper choice of the **time step** and of the **grid spacing**.

Recall in the fluid (advection, MHD) simulations, stability requires

$$u\Delta t < \Delta x$$ u is wave speed

In a PIC simulation, the time step needs to resolve both **light-wave propagation** and **Langmuir wave propagation**

$$c\Delta t < \Delta x$$

$$\omega_{pe}\Delta t < 2$$

The grid spacing needs to resolve the electron Debye length to avoid the so-called finite grid instability

$$\Delta x < \kappa\lambda_{De} \qquad \kappa \sim \mathcal{O}(1)$$

Practical considerations of the PIC method

Conservation

The PIC method described before conserves momentum to machine precision. However, energy is not preserved exactly. why?

For explicit PIC codes, based on the algorithm described above, the energy needs to be monitored and the time step and grid spacing need to be chosen sufficiently small as to conserve energy: satisfying the stability constraints is not enough.

In practice usually the time step has to be less than 1/10 of its stability limit and the grid spacing less than 1/3 for energy to be conserved sufficiently.

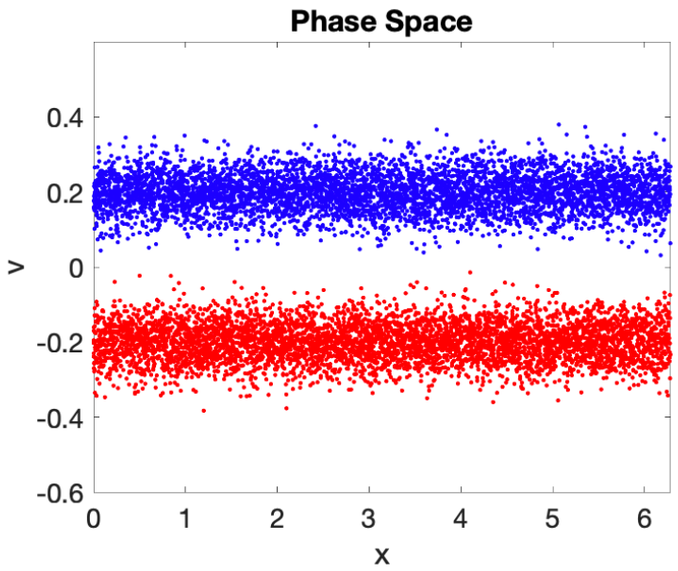
Recall: finite volume method conserves mass, momentum and energy to machine error

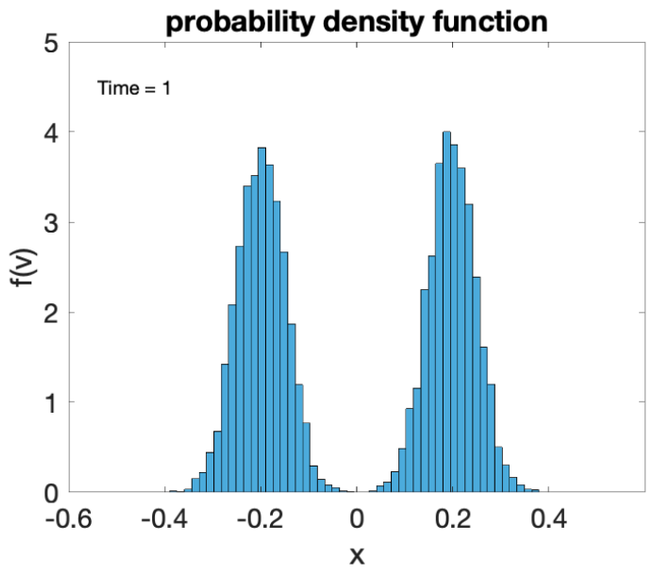
Practical considerations of the PIC method

Initialization - random versus quiet

Random starts are obtained using MonteCarlo methods for distribution sampling. The simplest case is the use of the function RAND of Matlab (or similar in other languages) to generate a normal distribution:

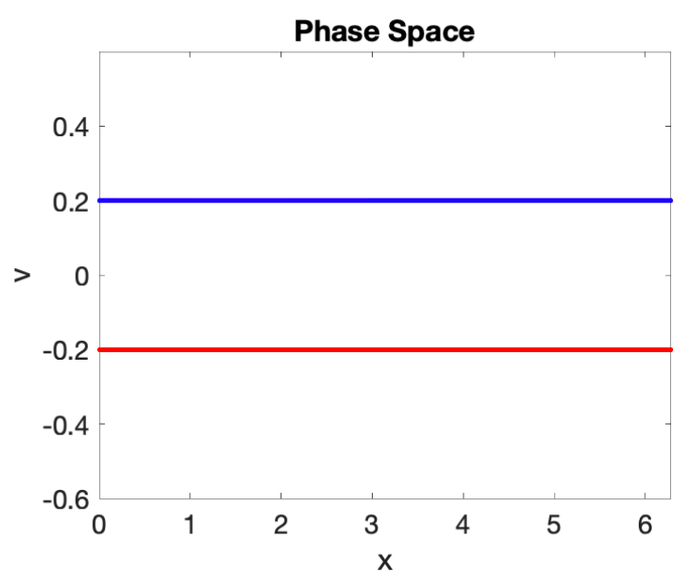
```
V0=0.2;
VT=0.05;
vp(1:2:N-1)=random('normal',V0,VT,[N/2 1]);
vp(2:2:N)=random('normal',-V0,VT,[N/2 1]);
```

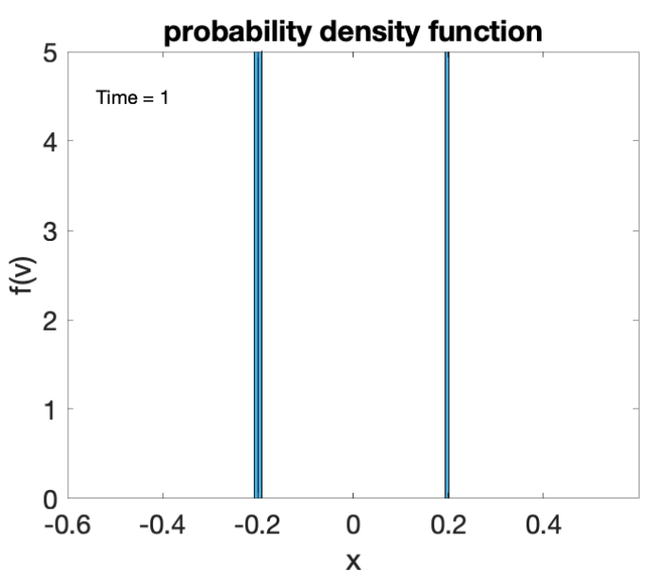




Quiet starts are an attempt to reduce the noise and inde- termination of random starts. The phase space is covered by a pattern of computational particles.

```
V0=0.2;
VT=0.0;
vp(1:2:N-1)=random('normal',V0,VT,[N/2 1]);
vp(2:2:N)=random('normal',-V0,VT,[N/2 1]);
```





Practical considerations of the PIC method

Diagnostics

The great advantage of the PIC method is that it provides quantities very similar to those provided in an actual plasma experiments: the user has both information on the distri- bution of the plasma particles and of the fields. Below we describe some of the most used diagnostics.

Typical *particle diagnostics* are:

- phase space plots (xp vs vp)
- more complex phase space density plots where the phase space is discretized in a grid of cells and the number of particles in each cell is counted to give the distribution function
- Velocity distribution functions (e.g. histogram of particle counts in energy bins)

Typical *field diagnostics* are:

- plots of the fields (e.g. E , ϕ, ϱ, . . .) versus space and/or time
- FFT modes of the field quantities to measure growth rates of specific modes.

Total kinetic energy:

$$E_k = \frac{1}{2}\sum_p m_p v_p^2$$

Total field energy:

$$E_E = \Delta x \sum_c \frac{\epsilon_0 E^2}{2} \equiv \Delta x \sum_c \frac{\rho\varphi}{2}$$

Total momentum:

$$P = \sum_p m_p v_p$$

Once through the PIC_es.m code

Generate a numerical Grid

create x_i

Load Initial Particles (t = 0)

E.g., follow a distribution $f(v,x)\Big|_{t=0} = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}} \left[1 + A\cos(kx)\right]$

Apply B.C.

Fill the passive cells for particles (e.g., periodic)

Start from n=1

Move particles to n+1

$$x_p^{n+1} = x_p^n + \Delta t\, v_p^{n+\frac{1}{2}} \qquad v_p^{n+\frac{3}{2}} = v_p^{n+\frac{1}{2}} + \Delta t \frac{q_s}{m_s} E_p(x_p^{n+1})$$

Calc E field

$$\epsilon_0 \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2} = -\rho_i \qquad E_i = \frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x}$$

Loop

Apply B.C.

Fill the passive cells for particles (e.g., periodic)

T>T_0?

Determine whether reaching the end time (or other criteria)

YES

Done

Once through the PIC_es.m code

```
clear
close all
% parameters and constants
L=2*pi;
DT=.5;
NT=200;NTOUT=25;
NG=64;
N=10000;
WP=1;
QM=-1;
V0=0.2;
VT=0.0;
XP1=1;
V1=0.0;
mode=1;
Q=WP.^2/(QM*N/L);
rho_back=-Q*N/L;
dx=L/NG;
% initial loading for the 2 Stream instability
xp=linspace(0,L-L/N,N)';
vp=VT*randn(N,1);
xp=xp+XP1*(L/N)*sin(2*pi*xp/L*mode);
vp(1:2:N-1)=random('normal',V0,VT,[N/2 1]);
vp(2:2:N)=random('normal',-V0,VT,[N/2 1]);
vp=vp+V1*sin(2*pi*xp/L*mode); % add sin variations
% arrays for the Poisson calc
p=1:N;p=[p p];
un=ones(NG-1,1);
Poisson=spdiags([un -2*un un],[-1 0 1],NG-1,NG-1);
```

Parameters

Initialization

Once through the PIC_es.m code

```
30      % Main computational cycle
31 -    for it=1:1
32          % update xp
33 -        xp=xp+vp*DT;
34          % apply bc on the particle positions
35 -        out=(xp<0); xp(out)=xp(out)+L;
36 -        out=(xp>=L);xp(out)=xp(out)-L;
37          % projection p->g
38 -        g1=floor(xp/dx-.5)+1;g=[g1;g1+1];
39 -        fraz1=1-abs(xp/dx-g1+.5);fraz=[fraz1;1-fraz1];
40          % apply bc on the projection
41 -        out=(g<1);g(out)=g(out)+NG;
42 -        out=(g>NG);g(out)=g(out)-NG;
43 -        mat=sparse(p,g,fraz,N,NG);
44 -        rho=full((Q/dx)*sum(mat))'+rho_back;
45          % computing fields
46 -        Phi=Poisson\(-rho(1:NG-1)*dx^2);Phi=[Phi;0];
47 -        Eg=([Phi(NG); Phi(1:NG-1)]-[Phi(2:NG);Phi(1)])/(2*dx);
48          % projection q->p and update of vp
49 -        vp=vp+mat*QM*Eg*DT;
```

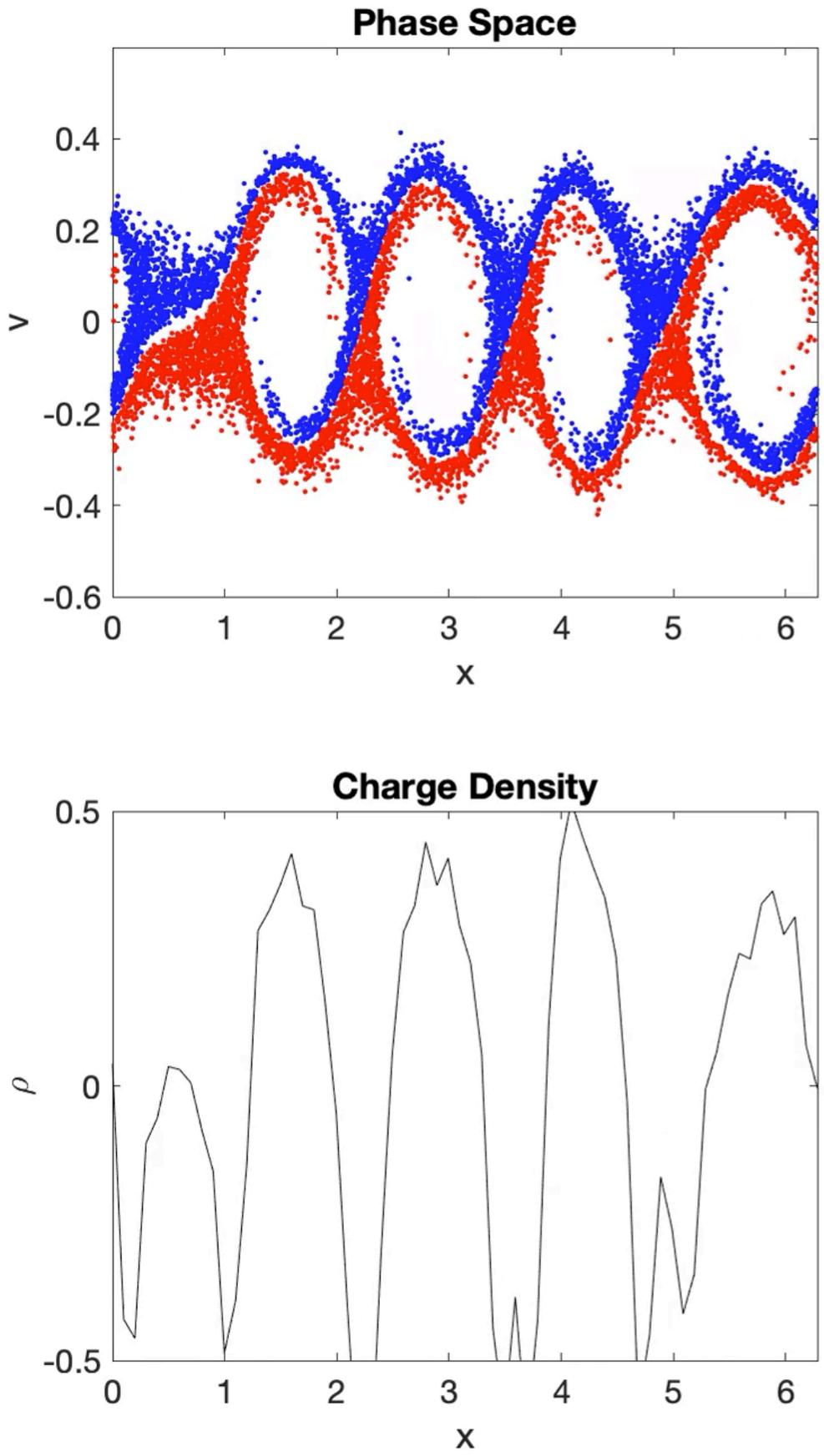
Move particles

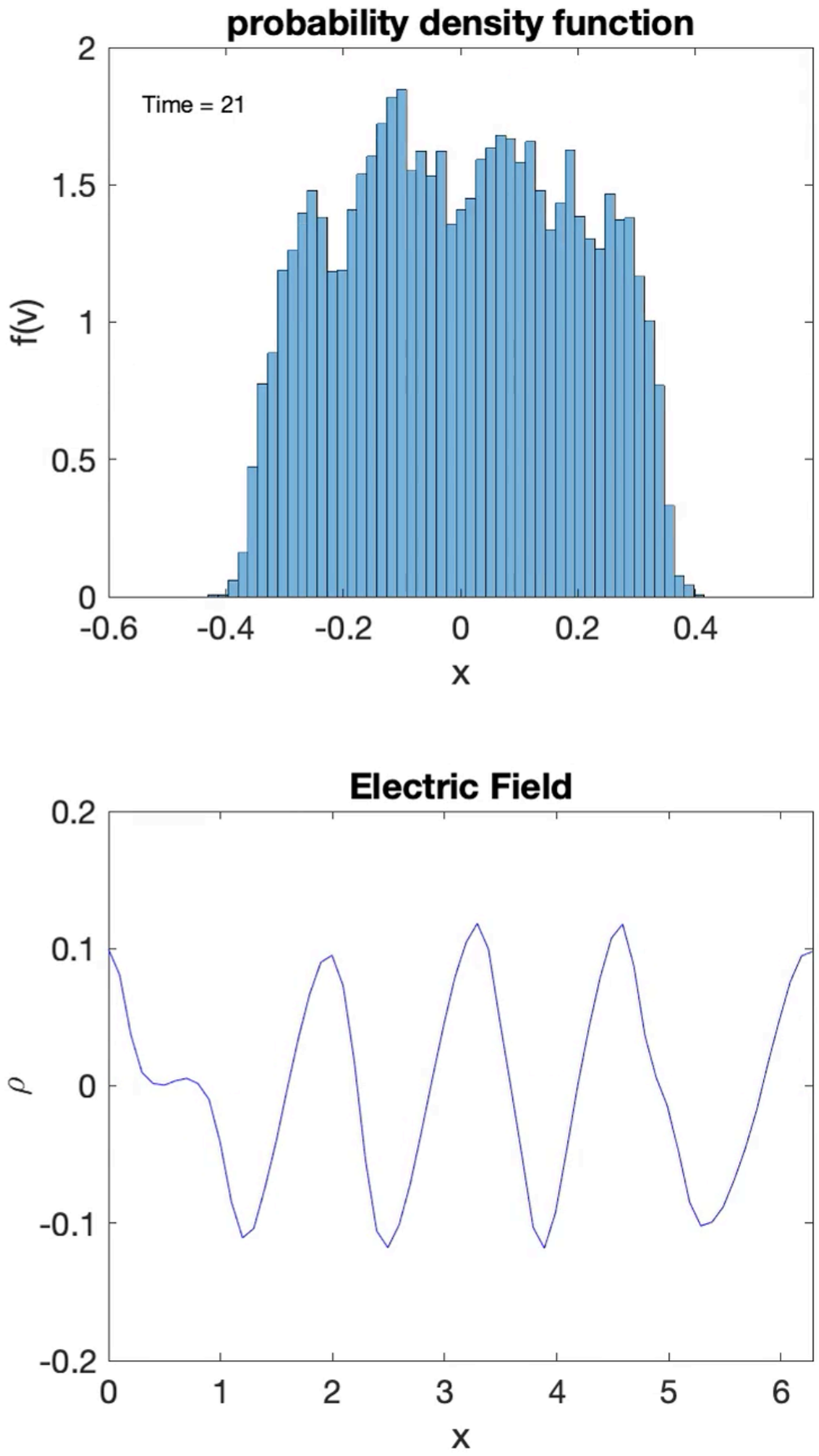
Boundary conditions

Calculate charge density at grid

Calculate E-field at grid

Update velocity


Phase Space
v
x
0.4
0.2
0
-0.2
-0.4
-0.6
0
1
2
3
4
5
6


probability density function
Time = 21
f(v)
x
2
1.5
1
0.5
0
-0.6
-0.4
-0.2
0
0.2
0.4

Charge Density
ρ
x
0.5
0
-0.5
0
1
2
3
4
5
6

Electric Field
ρ
x
0.2
0.1
0
-0.1
-0.2
0
1
2
3
4
5
6

HW assignment

1. Two Stream Instability

   a) Two stream instability is a standard test case for the Vlasov-Poisson system, for which analytic results from the dispersion relation are available for code validation. In all simulations, let's set ε = 0.001 and v ∈ [−10, 10]. Run the Vlasov-Poisson solver with the following initial conditions:

$$f(0,x,v) = \frac{1}{2}\left[\frac{1}{\sqrt{2\pi}} e^{-(v-v_0)^2/2} + \frac{1}{\sqrt{2\pi}} e^{-(v+v_0)^2/2}\right](1+\varepsilon\cos(k\,x))\,, \quad k = 0.2\,,$$

   and with Lx = 2π/k, T = 50, Nx = Nv = 256. For the different stream velocities v_0 given below, plot the square root of the field energy $\int |E|^2 dx$ as a function of time and compare to the analytic growth $e^{\omega i t}$, obtained from the dispersion relation:

$$v_0 = 1.3\ (\omega_i = 0.0011)\,, \quad v_0 = 2.4\ (\omega_i = 0.2258)\,, \quad v_0 = 3.0\ (\omega_i = 0.2845)\,.$$

   b) Now run the 1D electrostatic PIC code with similar initial conditions to simulate the two-stream instability. Compare the simulated growth rate with the analytical solutions used in a). Describe the difference between the two simulations. (hint: for the PIC simulation, you might need to adjust the ε value for a meaningful initial perturbation).

2. Bump-on-tail instability

a) Bump-on-tail instability is another useful test case for kinetic code validation, which has nice and simple analytical solutions to compare with. First setup initial conditions in the Vlasov-Poisson code as:

$$f(0,x,v) = \left[(1-n_b)\frac{1}{\sqrt{2\pi}}e^{-v^2/2} + n_b\frac{1}{\sqrt{2\pi}}e^{-(v-v_b)^2/0.5}\right](1+\varepsilon\cos(k\,x))\,,$$

and with k = 0.3, Lx = 2π/k, T = 60, n_b = 0.1, v_b= 4.5. As before, compare the analytic growth rate $e^{0.198t}$ with the simulation results (total E field energy).

b) Now setup the bump-on-tail instability using the 1D electrostatic PIC code with similar initial conditions in the same simulation domain, compare the simulated growth rate with the analytic rate used in a). Describe the difference between the two simulations.