

LA-UR-13-26843

Approved for public release; distribution is unlimited.

Title: Fully implicit particle in cell algorithm for electromagnetic plasma simulations

Author(s): Chacon, Luis

Intended for: NUMKIN13, 2013-09-02 (Munich, Germany)

Issued: 2013-09-03



Disclaimer:

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the Los Alamos National Security, LLC for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Fully implicit particle in cell algorithm for electromagnetic plasma simulations

L. Chacón

Los Alamos National Laboratory
P.O. Box 1663, Los Alamos, NM 87544

Collaborators:

G. Chen, W. Taitano, D. A. Knoll (LANL)
and the CoCoMANS team

NUMKIN 2013

Sept. 2-6, 2013

Garching, Germany

Work funded by the LANL LDRD program

Outline

- Particle-in-cell (PIC) methods for plasmas
- Explicit vs. implicit PIC
- ES implicit PIC:
 - ⇒ Charge and energy conservation
 - ⇒ Moment-based acceleration (NEW)
- Generalization to EM PIC (NEW)
 - ⇒ Review and motivation for Darwin model
 - ⇒ Conservation properties (energy, charge, and canonical momenta)

Introduction

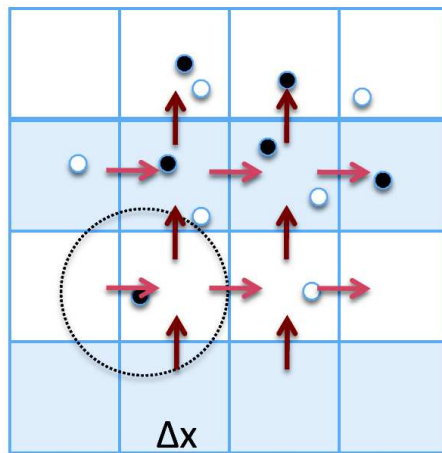
Particle-in-cell (PIC) methods for kinetic plasma simulation

$$\partial_t f + \mathbf{v} \cdot \nabla f + \frac{\mathbf{F}}{m} \cdot \nabla_v f = 0$$

- Lagrangian solution by the **method of characteristics**:

$$f(\mathbf{x}, \mathbf{v}, t) = f_0 \left(\mathbf{x} - \int_0^t dt \mathbf{v}, \mathbf{v} - \frac{1}{m} \int_0^t dt \mathbf{F} \right) ; \mathbf{x}(t=0) = \mathbf{x}_0 ; \mathbf{v}(t=0) = \mathbf{v}_0$$

- PIC approach follows characteristics employing **macroparticles** (volumes in phase space)



$$f(\mathbf{x}, \mathbf{v}, t) = \sum_p \delta(\mathbf{x} - \mathbf{x}_p) \delta(\mathbf{v} - \mathbf{v}_p)$$

$$\dot{\mathbf{x}}_p = \mathbf{v}_p$$

$$\dot{\mathbf{v}}_p = \frac{q_p}{m_p} (\mathbf{E} + \mathbf{v} \times \mathbf{B})$$

$$\partial_t \mathbf{B} + \nabla \times \mathbf{E} = 0$$

$$-\mu_0 \epsilon_0 \partial_t \mathbf{E} + \nabla \times \mathbf{B} = \mu_0 \mathbf{j}$$

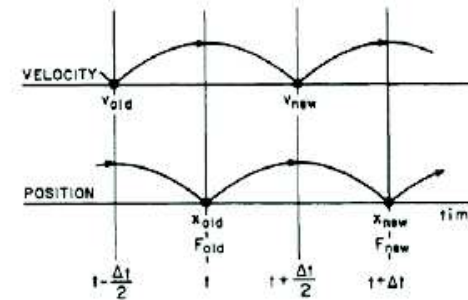
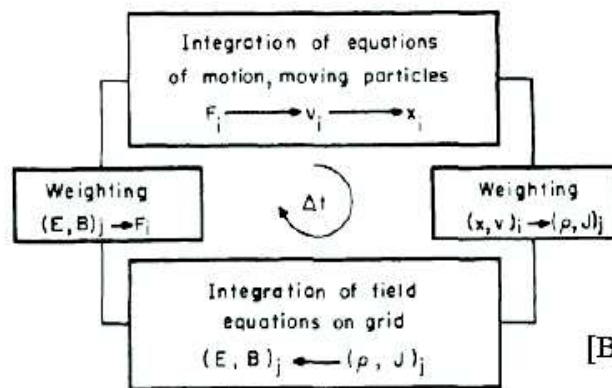
$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \cdot \mathbf{E} = \frac{e(n_i - n_e)}{\epsilon_0}$$

$$\delta(\mathbf{x} - \mathbf{x}_p) \longrightarrow S(\mathbf{x} - \mathbf{x}_p) ; E_p = \sum_i E_i S(x_i - x_p) ; j_i = \sum_p j_p S(x_i - x_p)$$

State-of-the-art *classical* PIC algorithm is explicit

- Classical explicit PIC approach “**leap-frogs**” **particle positions and velocities**, solves for fields after position update:



[Birdsall and Langdon, Plasma physics via computer simulation]

- **Severe performance limitations:**
 - ⇒ $\Delta x < \lambda_{Debye}$ (**finite-grid instability**: enforces a **minimum spatial resolution**)
 - ⇒ $\omega_{pe} \Delta t < 1$ (**CFL-type instability**: enforces a **minimum temporal resolution**)
 - ⇒ **Inefficient** for long-time, large-scale integrations
- In the **presence of strong magnetic fields**, **gyro-averaging** the Vlasov-Maxwell model can significantly ameliorate these limitations, but **there are other issues** (e.g. not asymptotic preserving, required order of expansion to capture some physical effects, treatment of nonlinear terms)

WE FOCUS ON ELECTROSTATIC PIC AS A PROOF OF PRINCIPLE

What about implicit PIC?

- Implicit PIC holds the promise of overcoming the difficulties and inefficiencies of explicit methods for long time-scale simulations
- Exploration of implicit PIC started in the 1980s
 - ⇒ Moment method [Mason, 1981; Brackbill, 1982]
 - ⇒ Direct method [Friedman, Langdon, Cohen, 1981]
- Early approaches used linearized, semi-implicit formulations:
 - ⇒ Lack of nonlinear convergence
 - ⇒ Inconsistencies between particles and moments
 - ⇒ Inaccuracies! → Plasma self-heating/cooling [Cohen, 1989]

Our goal is to explore the viability of a nonlinearly converged, fully implicit PIC algorithm

WHAT IS THE NATURE OF THE RESULTING FULLY-COUPLED ALGEBRAIC SYSTEM?
IS IT PRACTICAL TO INVERT?

Fully implicit PIC

Fully implicit PIC formulation

- A **fully implicit formulation** couples particles and fields non-trivially (integro-differential PDE):

$$\frac{f^{n+1} - f^n}{\Delta t} + \mathbf{v} \cdot \nabla \frac{f^{n+1} + f^n}{2} - \frac{q}{m} \nabla \frac{\Phi^{n+1} + \Phi^n}{2} \cdot \nabla_{\mathbf{v}} \frac{f^{n+1} + f^n}{2} = 0$$
$$\nabla^2 \Phi^{n+1} = \int d\mathbf{v} f^{n+1}(\mathbf{x}, \mathbf{v}, t)$$

- In PIC, f^{n+1} is sampled by a large collection of particles in phase space, $\{\mathbf{x}, \mathbf{v}\}_p^{n+1}$.
 - ⇒ There are N_p particles, each particle requiring $2 \times d$ equations ($d \rightarrow$ dimensions),
 - ⇒ Field requires N_g equations, one per grid point.
- If implemented naively, an **impractically large algebraic system of equations** results:

$$\boxed{\mathbf{G}(\{\mathbf{x}, \mathbf{v}\}_p^{n+1}, \{\Phi^{n+1}\}_g) = 0} \rightarrow \dim(\mathbf{G}) = 2dN_p + N_g \gg N_g$$

- ⇒ No current computing mainframe can afford the **memory requirements**
 - ⇒ **Algorithmic issues are showstoppers** (e.g., how to precondition it?)
- An **alternative strategy** exists: nonlinear elimination (**particle enslavement**)

Particle enslavement (nonlinear elimination)

- Full residual $\mathbf{G}(\{x, v\}_p, \{\Phi\}_g) = 0$ is impractical to implement
- Alternative: nonlinearly eliminate particle quantities so that they are not dependent variables:
 - ⇒ Formally, particle equations of motion are functionals of the electrostatic potential:

$$x_p^{n+1} = x_p[\Phi^{n+1}] ; v_p^{n+1} = v_p[\Phi^{n+1}]$$

$$\mathbf{G}(\mathbf{x}_p^{n+1}, \mathbf{v}_p^{n+1}, \Phi^{n+1}) = \mathbf{G}(\mathbf{x}[\Phi^{n+1}], \mathbf{v}[\Phi^{n+1}], \Phi^{n+1}) = \tilde{\mathbf{G}}(\Phi^{n+1})$$

Nonlinear residual can be *unambiguously* formulated in terms of electrostatic potential only!

- JFNK storage requirements are dramatically decreased, making it tractable:
 - ⇒ Nonlinear solver storage requirements $\propto N_g$, comparable to a fluid simulation
 - ⇒ Particle quantities \Rightarrow auxiliary variables: only a **single copy of particle population** needs to be maintained in memory throughout the nonlinear iteration

Field equation: Vlasov-Poisson vs. Vlasov-Ampere

- **Nonlinear elimination** procedure leads to $\mathbf{G}(\Phi) = 0$ (or $\mathbf{G}(E) = 0$)
- **Two formulations** are possible:

Vlasov-Poisson (VP)	Vlasov-Ampère (VA)
$\partial_t f + v \partial_x f + \frac{qE}{m} \partial_v f = 0$ $\partial_x E = \frac{\rho}{\epsilon_0}$ $E = -\partial_x \Phi$	$\partial_t f + v \partial_x f + \frac{qE}{m} \partial_v f = 0$ $\epsilon_0 \partial_t E + j = \langle j \rangle$
Two systems are equivalent in continuum, but not in the discrete.	
<ul style="list-style-type: none"> ➤ Conventionally used in explicit PIC. ➤ Exact <i>local</i> charge conservation. ➤ Exact <i>global</i> momentum conservation. ➤ Unstable with orbit averaging in implicit context [Cohen and Freis, 1982]. 	<ul style="list-style-type: none"> ➤ Exact <i>local</i> charge conservation. ➤ Exact <i>global</i> energy conservation. ➤ Suitable for orbit averaging. ➤ Can be extended to electromagnetic system.

- We will show, however, that an **equivalent energy-conserving VP formulation** exists.

Energy-conserving (EC) Vlasov-Ampère discretization

- Fully implicit Crank-Nicolson time discretization:

$$\begin{aligned}\varepsilon_0 \frac{E_i^{n+1} - E_i^n}{\Delta t} + \sum_p q_p v_p^{n+1/2} S(x_i - x_p^{n+1/2}) &= 0 \\ \frac{x_p^{n+1} - x_p^n}{\Delta t} &= \frac{v_p^{n+1} + v_p^n}{2} \\ \frac{v_p^{n+1} - v_p^n}{\Delta t} &= \frac{q_p}{m_p} \sum_i \frac{E_i^n + E_i^{n+1}}{2} S(x_i - x_p^{n+1/2})\end{aligned}$$

In time:

centered, 2nd order;
implicit;
unconditionally stable;
non-dissipative.

- C-N enforces energy conservation to numerical round-off:

$$\sum_p \frac{m_p}{2} (v_p^{n+1} + v_p^n) (v_p^{n+1} - v_p^n) = - \sum_i \varepsilon_0 \frac{E_i^{n+1} - E_i^n}{\Delta t} \frac{E_i^{n+1} + E_i^n}{2} \Rightarrow \sum_p \frac{1}{2} m_p v_p^2 + \sum_i \frac{1}{2} \varepsilon_0 E_i^2 = \text{const}$$

- As a result, the formulation does not suffer from finite-grid instabilities (normal mode analysis)
 - ⇒ Unconstrained spatial resolution: $\Delta x \not\leq \lambda_D$!!
- Energy conservation is only realized when particles and fields are nonlinearly converged:
 - ⇒ Requires a tight nonlinear tolerance

Jacobian-Free Newton-Krylov Methods

- After spatial and temporal discretization \Rightarrow a large set of nonlinear equations:

$$\vec{G}(\vec{x}^{n+1}) = \vec{0}$$

- Converging nonlinear couplings requires iteration: Newton-Raphson method:

$$\left. \frac{\partial \vec{G}}{\partial \vec{x}} \right|_k \delta \vec{x}_k = -\vec{G}(\vec{x}_k)$$

- Jacobian linear systems result, which require a linear solver \Rightarrow Krylov subspace methods (GMRES)
 - \Rightarrow Only require matrix-vector products to proceed.
 - \Rightarrow Jacobian-vector product can be computed Jacobian-free (**CRITICAL**: no need to form Jacobian matrix):

$$\left(\frac{\partial \vec{G}}{\partial \vec{x}} \right)_k \vec{y} = J_k \vec{y} = \lim_{\epsilon \rightarrow 0} \frac{\vec{G}(\vec{x}_k + \epsilon \vec{y}) - \vec{G}(\vec{x}_k)}{\epsilon}$$

- \Rightarrow Krylov methods can be easily preconditioned: $P_k^{-1} \sim J_k^{-1}$

$$J_k P_k^{-1} P_k \delta \vec{x} = -\vec{G}_k$$

We will explore suitable preconditioning strategies later in this talk.

Algorithmic implementation details

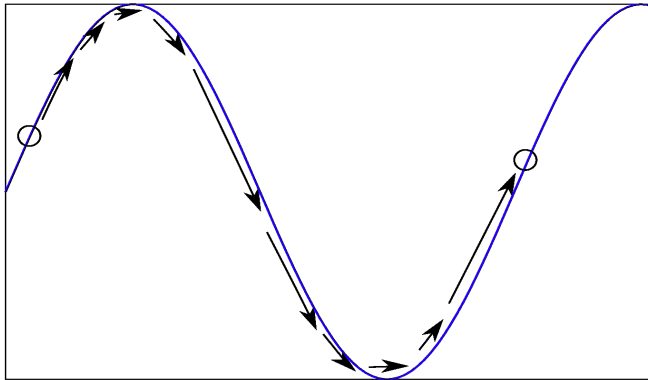
- The **nonlinear residual formulation** $\mathbf{G}(E^{n+1})$ based on Vlasov-Ampere formulation is as follows:
 1. Input E (given by JFNK iterative method)
 2. Move particles (i.e., find $x_p[E]$, $v_p[E]$ by solving equations of motion)
 - (a) Requires inner (local) nonlinear iteration: Picard (not stiff)
 - (b) Can be as complicated as we desire (substepping, adaptivity, etc)
 3. Compute moments (current)
 4. Form Vlasov-Ampere equation residual
 5. return
- Because **particle move is performed within function evaluation**, we have much freedom.
- Rest of the talk will describe **improvements in particle mover** to ensure **long-term accuracy**
 - ⇒ **Particle substepping and orbit averaging** (ensures orbit accuracy and preserves exact energy conservation)
 - ⇒ **Exact charge conservation strategy** (a new charge-conserving particle mover)
 - ⇒ **Orbit adaptivity** (to improve momentum conservation)

Particle orbit substepping

- In applications of interest, **field time-scale (Δt)** and **orbit time-scale ($\Delta \tau$)** can be well separated
 - ⇒ Fields evolve *slowly* (dynamical time scale, Δt)
 - ⇒ Particle orbits may still undergo *rapid change* ($\Delta \tau \ll \Delta t$)
- **Particle orbits need to be resolved** to **avoid large orbit integration errors**

Accurate orbit integration requires particle substepping!

- **Field does not change appreciably:** time-averaged value over long time scale is sufficient



$$\begin{aligned}\frac{x_p^{\nu+1} - x_p^\nu}{\Delta \tau} &= v_p^{\nu+1/2} \\ \frac{v_p^{\nu+1} - v_p^\nu}{\Delta \tau} &= \sum_i \underbrace{\frac{E_i^{n+1} + E_i^n}{2}}_{\text{slow}} S(x_i - x_p^{\nu+1/2})\end{aligned}$$

Energy conservation and orbit averaging

- Particle substepping breaks energy conservation.
- Energy conservation theorem can be recovered by orbit averaging Ampère's law:

$$\epsilon_0 \partial_t E + j = \langle j \rangle \quad , \quad \frac{1}{\Delta t} \int_t^{t+\Delta t} d\tau [\dots] \Rightarrow \epsilon_0 \frac{E^{n+1} - E^n}{\Delta t} + \bar{j} = \langle \bar{j} \rangle$$

- Orbit-averaged current is found as:

$$\bar{j} = \frac{1}{\Delta t} \int_t^{t+\Delta t} d\tau j \approx \frac{1}{\Delta t} \sum_p \sum_{\nu=1}^{N_\nu} q_p v_p S(x - x_p) \Delta \tau^\nu$$

- With these definitions, exact energy conservation is recovered:

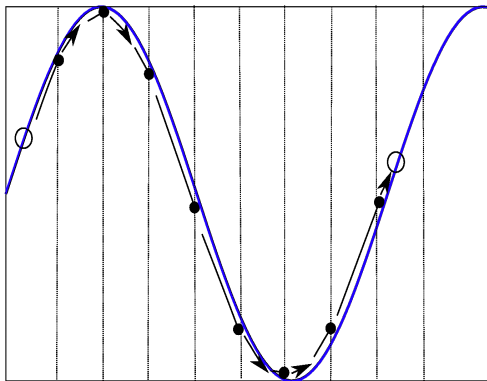
$$\sum_p \sum_\nu \frac{m_p}{2} (v_p^{\nu+1} + v_p^\nu) (v_p^{\nu+1} - v_p^\nu) = - \sum_i \epsilon_0 \frac{E^{n+1} - E^n}{\Delta t} \frac{E_i^{n+1} + E_i^n}{2}$$

$$\Rightarrow \sum_p \frac{1}{2} m_p v_p^2 + \sum_i \frac{1}{2} \epsilon_0 E_i^2 = \text{const.}$$

Exact charge conservation: charge-conserving particle mover

- Local charge conservation (enforced in the continuum by Gauss' law) is violated in discrete Vlasov-Ampère formulation.
- Local charge conservation is essential to ensure long-term accuracy of numerical algorithm
- Exact charge conservation requires a particle mover that satisfies a discrete charge continuity equation, $\partial_t \rho + \nabla \cdot \mathbf{j} = 0$ [Buneman 1968, Morse and Nielson, 1971]
 - ⇒ **Standard strategy** based on current redistribution when particle crosses boundary.
 - ⇒ In our context, current redistribution breaks energy conservation. **Need new strategy.**

Here, charge conservation is enforced by stopping particles at cell boundaries.

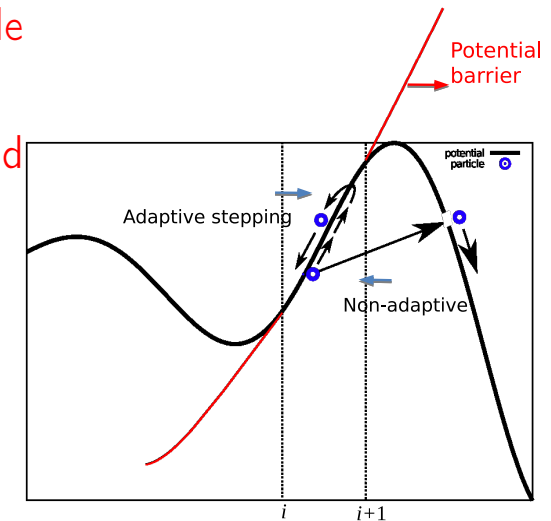


$$\left. \begin{aligned} \rho_{i+\frac{1}{2}} &= \sum_p q_p \frac{S_m(x-x_{i+\frac{1}{2}})}{\Delta x} \\ j_i &= \sum_p q_p v_p \frac{S_{m-1}(x-x_i)}{\Delta x} \\ S'_m(x) &= \frac{S_{m-1}(x+\frac{\Delta x}{2}) - S_{m-1}(x-\frac{\Delta x}{2})}{\Delta x} \end{aligned} \right\} \xRightarrow{(m=1,2)} [\partial_t \rho + \nabla \cdot \mathbf{j} = 0]_{i+\frac{1}{2}}^{n+\frac{1}{2}} = 0$$

Momentum conservation: adaptive orbit integrator

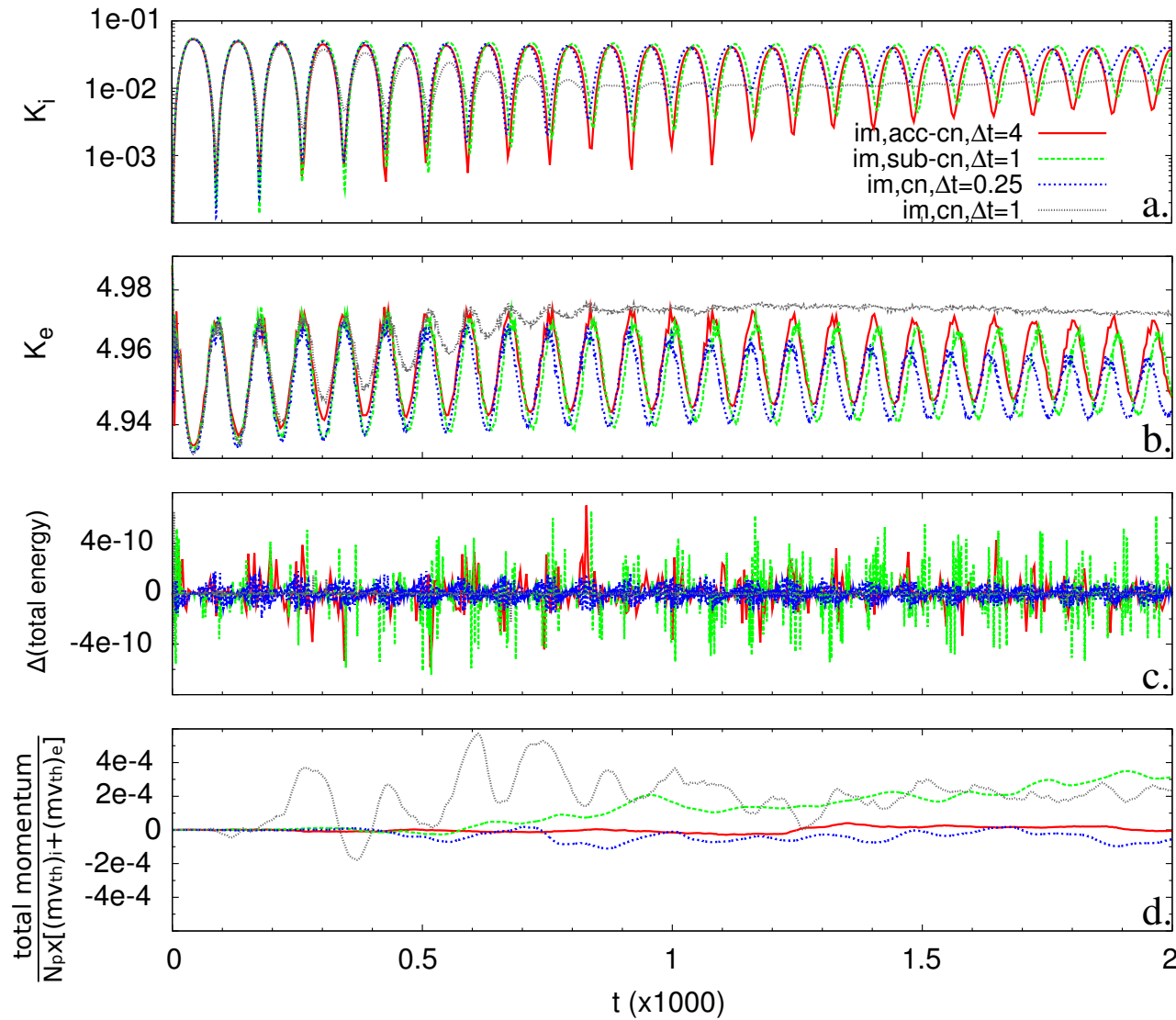
- EC/CC PIC algorithm does not enforce momentum conservation exactly.
 - ⇒ **Controlling error** in momentum conservation is **crucial** for long-term accuracy
- **Orbit integration errors** can significantly affect momentum conservation: **particle tunneling**
- Adaptive orbit integration can be effective in suppressing particle tunneling and thus improve momentum conservation
- **Approach**: find $\Delta\tau$ to control local truncation error. Second order estimator gives:

$$\Delta\tau \leq \sqrt{12\epsilon_r \frac{m_p}{q_p} \left| \frac{dE}{dx} \right|_p^{-1}}$$

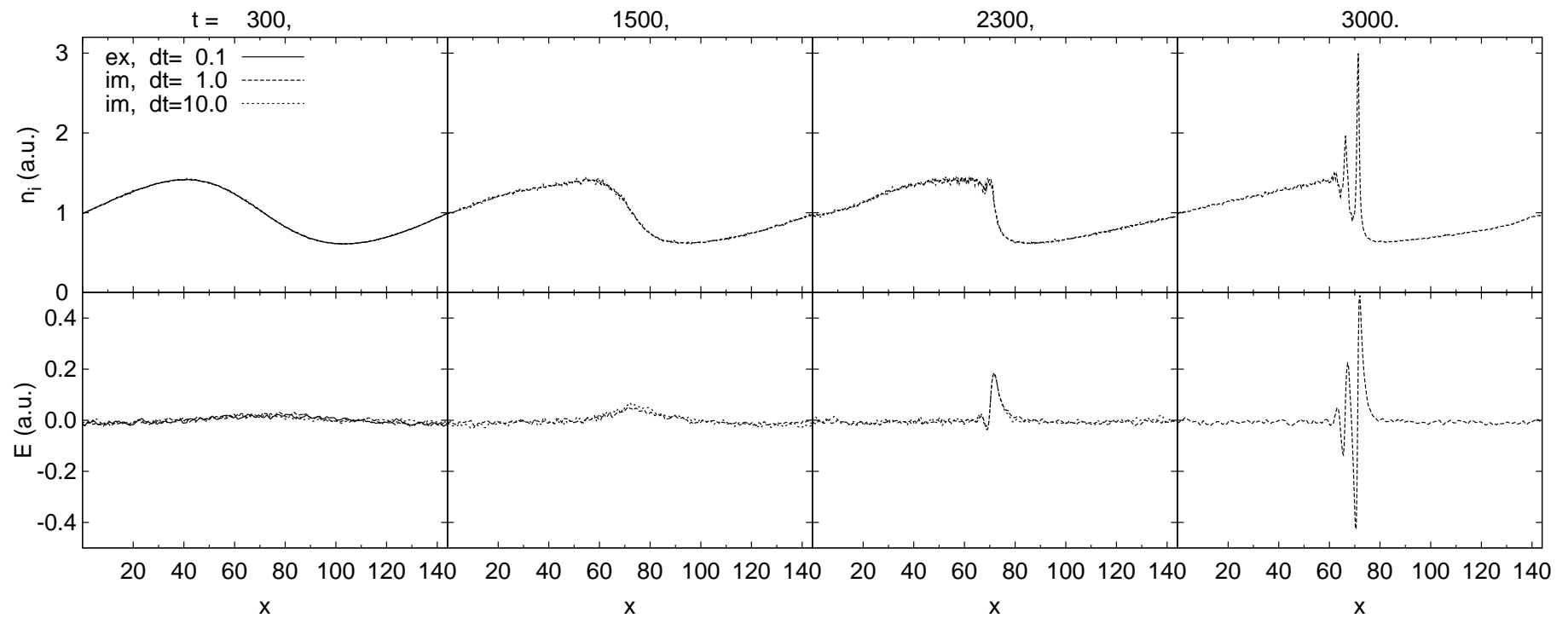


- Electric field gradient is estimated from cell-based gradient:
 $\left. \frac{\partial E}{\partial x} \right|_p \approx \frac{E_{i+1} - E_i}{\Delta x}$. Provides potential barrier!
- Particle is stopped at cell boundaries to ensure charge conservation.

Ion acoustic wave (IAW): accuracy impact of different EC movers



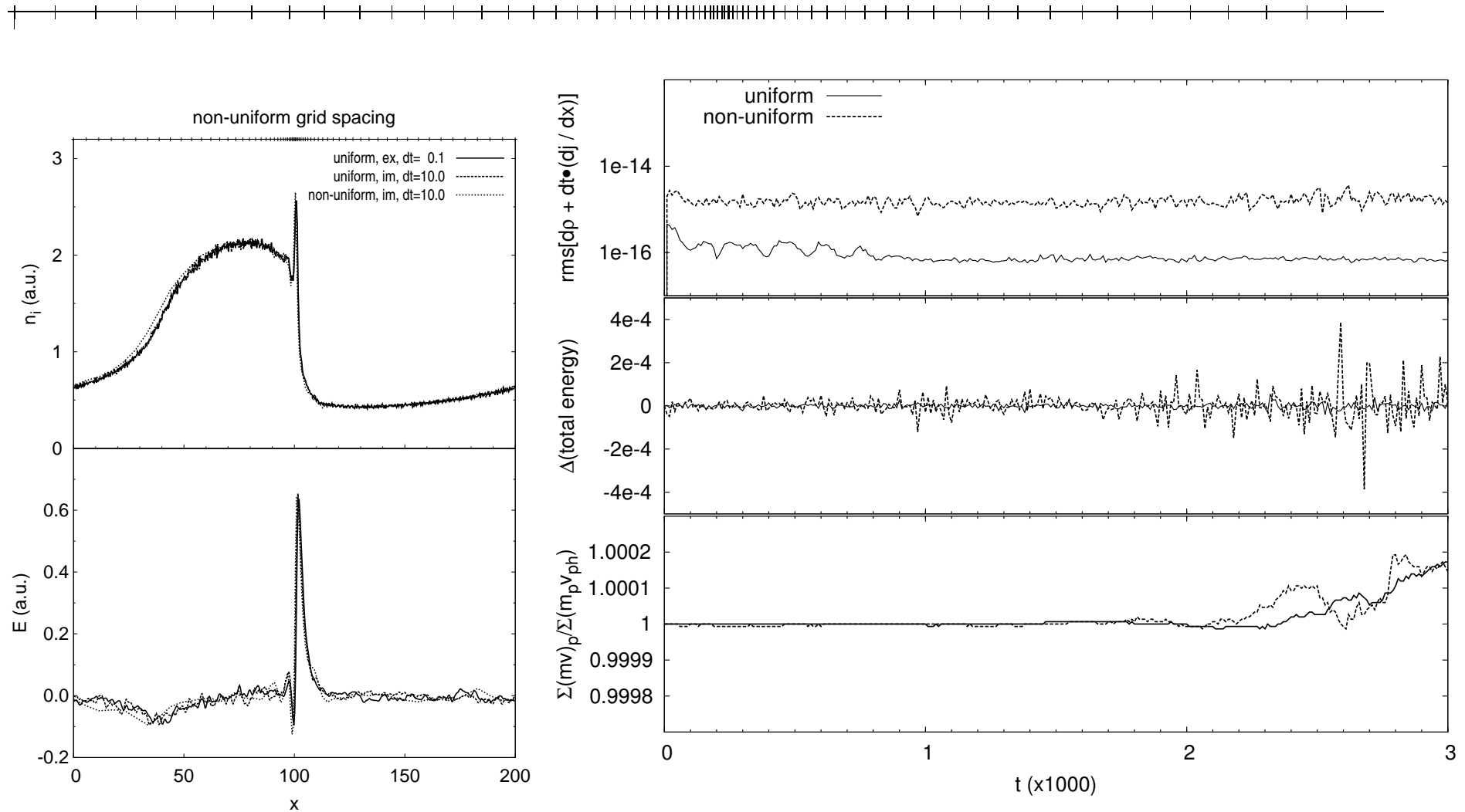
Ion acoustic shock wave



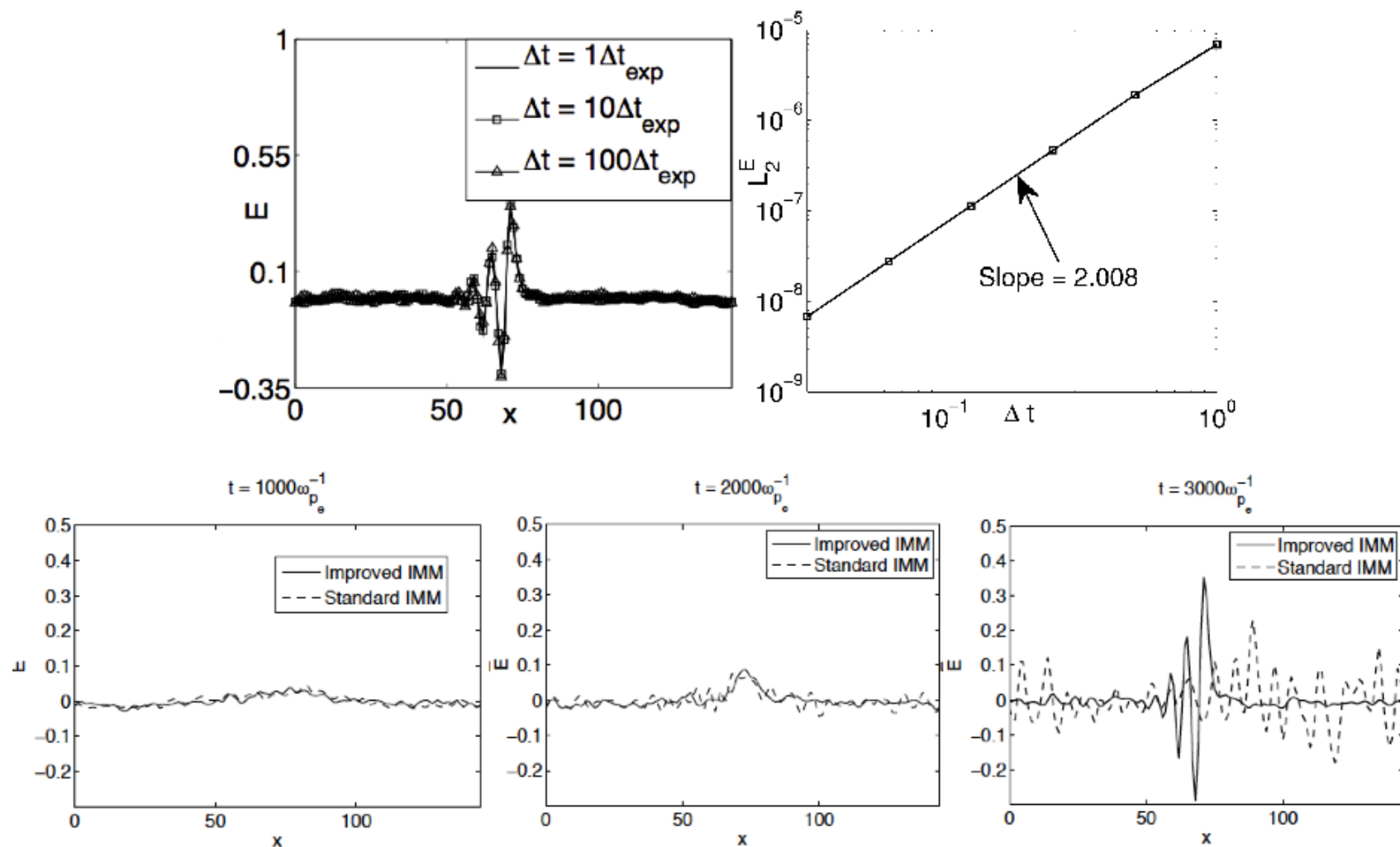
- Propagating IAW with perturbation level $\epsilon = 0.4$, with 4000 particles/cell.
- Realistic mass ratio ($m_i/m_e = 2000$).
- Shock wave length scale \sim Debye length.

Ion acoustic shock wave test

non-uniform grid spacing



Comparison against Implicit Moment Method¹



¹Taitano et al., *SISC*, accepted (2013)

Moment-based acceleration for fully implicit PIC

CPU gain potential of implicit PIC vs. explicit PIC

- Back-of-the-envelope estimate of CPU gain:

$$CPU \sim \left(\frac{T}{\Delta t}\right) \left(\frac{L}{\Delta x}\right)^d n_p C^{solver} ; \quad \frac{C^{imp}}{C^{ex}} \sim N_{FE} \frac{\Delta t_{imp}}{\Delta \tau_{imp}} ; \quad \frac{CPU_{ex}}{CPU_{imp}} \sim \left(\frac{\Delta x_{imp}}{\Delta x_{ex}}\right)^d \frac{\Delta \tau_{imp}}{\Delta t_{ex}} \frac{1}{N_{FE}}$$

- Using reasonable estimates:

$$\Delta \tau_{imp} \sim \min \left[0.1 \frac{\Delta x_{imp}}{v_{th}}, \Delta t_{imp} \right]$$

$$\Delta t_{imp} \sim 0.1 \omega_{pi}^{-1}$$

$$\Delta t_{exp} \sim 0.1 \omega_{pe}^{-1}$$

$$k \Delta x_{imp} \sim 0.2$$

$$\Delta x_{ex} \sim \lambda_D$$

$$\frac{CPU_{ex}}{CPU_{imp}} \sim \frac{1}{(k \lambda_D)^d} \min \left[\frac{1}{k \lambda_D}, \sqrt{\frac{m_i}{m_e}} \right] \frac{1}{N_{FE}}$$

- CPU speedup is:

- ⇒ Independent of time step!
- ⇒ Better for realistic mass ratios!
- ⇒ Limited by solver performance N_{FE} (preconditioning!)

Moment-based acceleration of fully kinetic simulations

- Particle elimination \Rightarrow nonlinear residual is formulated in terms of fields/moments ONLY: $\mathbf{G}(E)$
- Within JFNK, preconditioner ONLY needs to provide field/moment update:

$$\delta E \approx -P^{-1}\mathbf{G}$$

Premise of acceleration: obtain δE from a fluid model using current particle distribution for closure.

- We begin with corresponding fluid nonlinear model:

$$\begin{aligned}\partial_t n_\alpha &= -\nabla \cdot \mathbf{\Gamma}_\alpha \\ m_\alpha \left[\partial_t \mathbf{\Gamma}_\alpha + \nabla \cdot \left(\frac{1}{n_\alpha} \mathbf{\Gamma}_\alpha \mathbf{\Gamma}_\alpha \right) \right] &= q_\alpha n_\alpha \mathbf{E} + \nabla \cdot \left(n_\alpha \left(\frac{\mathbf{\Pi}_\alpha}{n_\alpha} \right)_p \right) \\ \epsilon_0 \partial_t \mathbf{E} &= \sum_\alpha q_\alpha \mathbf{\Gamma}_\alpha\end{aligned}$$

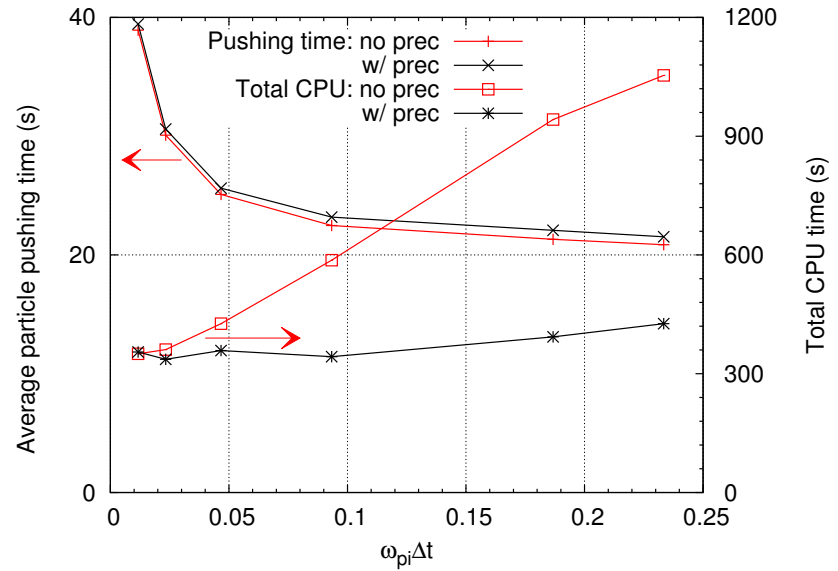
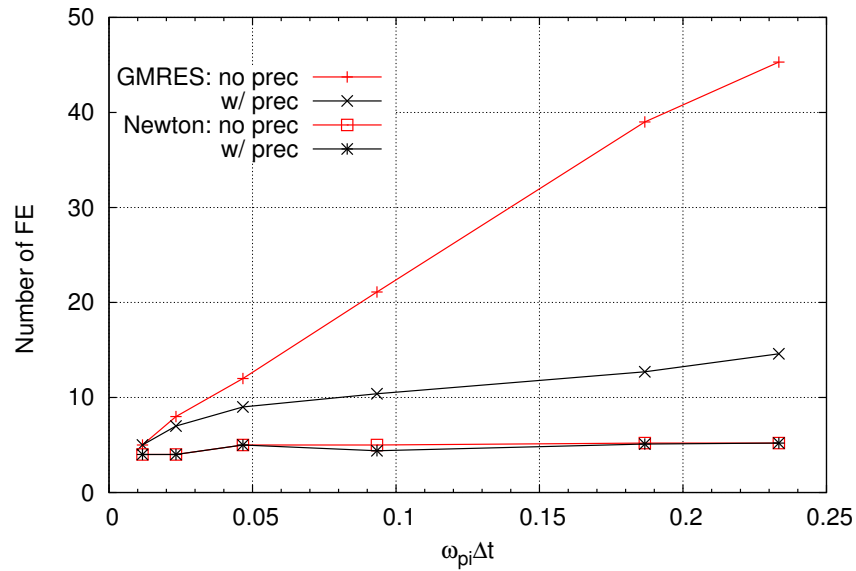
Moment-based acceleration of fully kinetic simulations (cont.)

- We formulate *approximate* linearized fluid equations (neglect linear temperature response):

$$\begin{aligned}\frac{\delta n_\alpha}{\Delta t} &= -\nabla \cdot \delta \mathbf{\Gamma}_\alpha \\ m_\alpha \frac{\delta \mathbf{\Gamma}_\alpha}{\Delta t} &\approx q_\alpha (\delta n_\alpha \mathbf{E} + n_{\alpha,p} \delta \mathbf{E}) + \nabla \cdot \left(\left(\frac{\mathbf{\Pi}_\alpha}{n_\alpha} \right)_p \delta n_\alpha \right) \\ \epsilon_0 \delta \mathbf{E} &= \Delta t \left[\sum_\alpha q_\alpha \delta \mathbf{\Gamma}_\alpha - \mathbf{G}(\mathbf{E}) \right]\end{aligned}$$

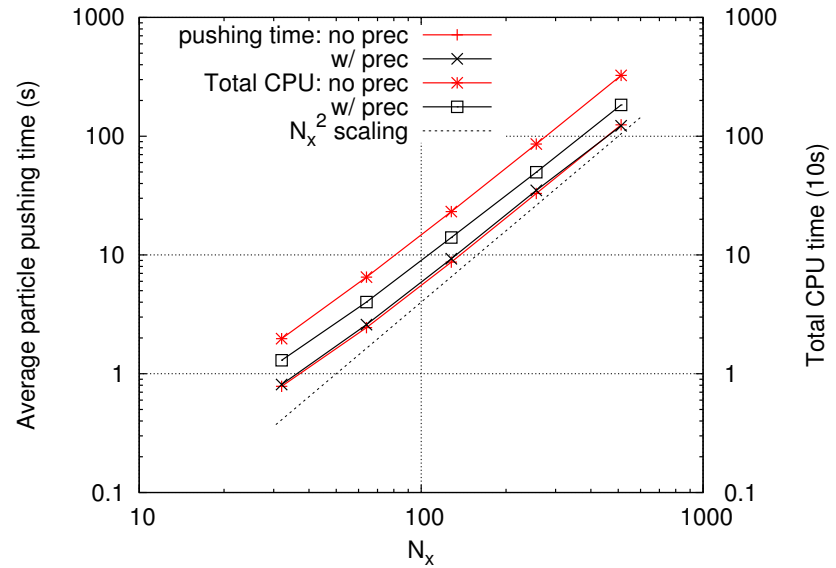
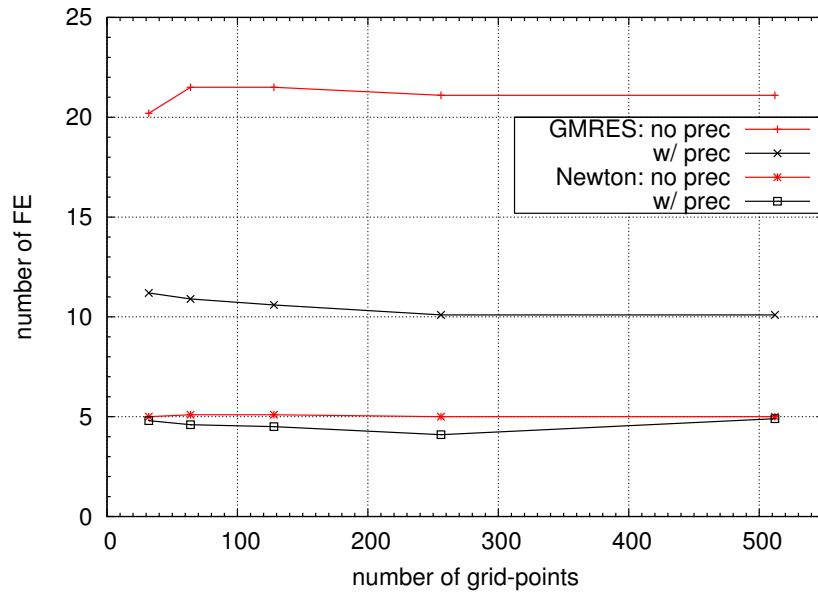
δE can be obtained from Newton state \mathbf{E} , Newton residual $\mathbf{G}(\mathbf{E})$,
and particle closures $\mathbf{\Pi}_{\alpha,p}$ and $n_{\alpha,p}$

Preconditioner performance with Δt



- Number of FE remains constant with Δt (preconditioning)
- Overall CPU time of algorithm is independent of Δt (as predicted!)

Preconditioner performance with N_x



➤ Number of FE independent of N_x (as expected from plasma freq.)

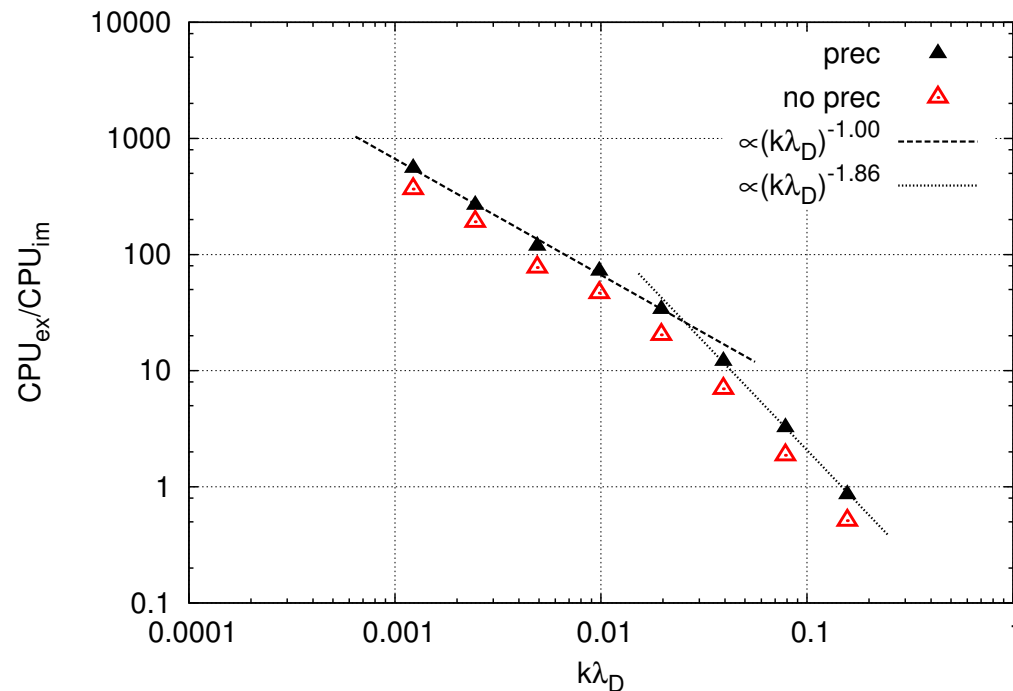
➤ CPU cost grows as N_x^2

⇒ $\propto N_x$ due to particles, and $\propto N_x$ due to crossings

⇒ In multi-D: $CPU \propto N \times N^{1/d}$

Preconditioner performance: CPU scaling

$$\frac{CPU_{ex}}{CPU_{imp}} \sim \frac{1}{(k\lambda_D)^d} \frac{1}{N_{FE}} \min \left[\frac{1}{k\lambda_D}, \sqrt{\frac{m_i}{m_e}} \right]$$



Transition occurs at $k\lambda_D \sim \sqrt{\frac{m_e}{m_i}} \sim 0.025$, as predicted

Electromagnetic PIC:
non-radiative Darwin formulation

Darwin approximation to Maxwell equations: motivation

- To analytically remove light-wave in non-relativistic plasma simulations *while preserving charge separation effects*
- If one keeps light wave with exact energy conservation in non-relativistic setting, one gets enhanced **numerical noise due to numerical Cherenkov radiation**

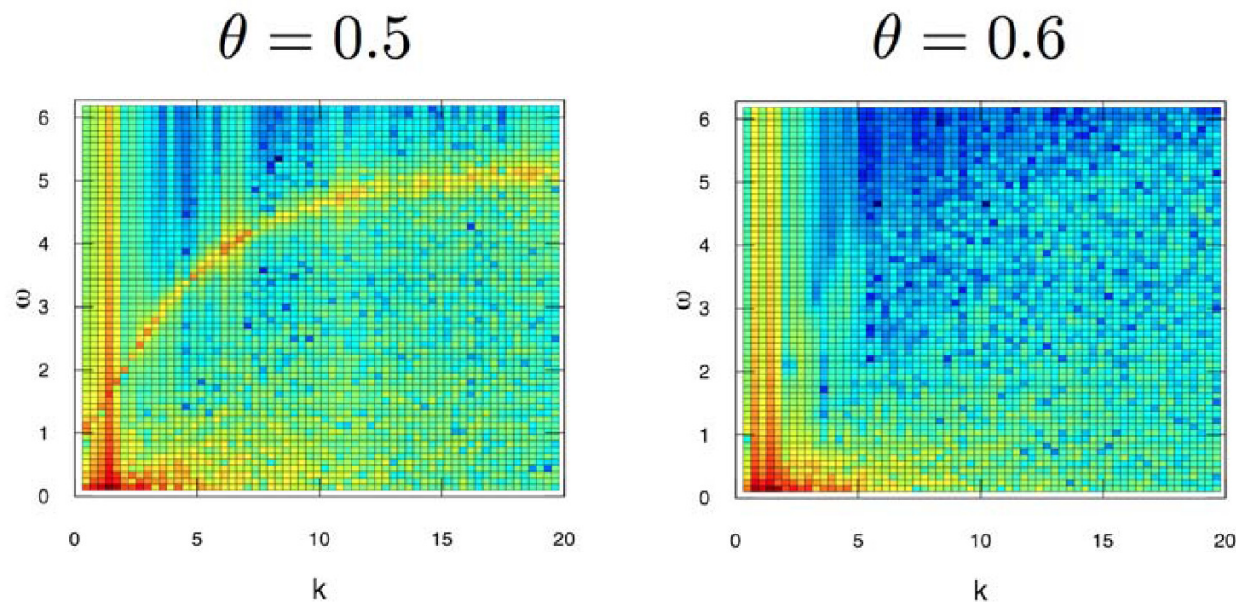


Figure 1: Fourier phase space for exactly energy conserving PIC (left) and dissipative PIC (right) [Markidis and Lapenta, JCP 2011].

Darwin model (potential form)

- We consider potentials ϕ , \mathbf{A} in the Coulomb gauge ($\nabla \cdot \mathbf{A} = 0$) such that:

$$\mathbf{B} = \nabla \times \mathbf{A},$$

$$\mathbf{E} = -\nabla\phi - \partial_t\mathbf{A}.$$

- Darwin model projects out the speed of light without enforcing quasineutrality (i.e., allowing for charge separation effects).

$$\begin{aligned}\nabla^2\chi &= \nabla \cdot \mathbf{j}, \\ -\nabla^2\mathbf{A} &= \mu_0 [\mathbf{j} - \nabla\chi], \\ \chi &= \epsilon_0\partial_t\phi.\end{aligned}$$

- Problem becomes **elliptic** (not hyperbolic): **no issue for implicit algorithm**

- In 1D:

$$\begin{aligned}\epsilon_0\partial_tE_x + j_x &= \langle j_x \rangle, & E_{y,i}^{n+1/2} &= -\frac{A_{y,i}^{n+1} - A_{y,i}^n}{\Delta t}, \\ \frac{1}{\mu_0}\partial_x^2 A_y + j_y &= \langle j_y \rangle, & E_{z,i}^{n+1/2} &= -\frac{A_{z,i}^{n+1} - A_{z,i}^n}{\Delta t}. \\ \frac{1}{\mu_0}\partial_x^2 A_z + j_z &= \langle j_z \rangle.\end{aligned}$$

Implicit EM particle mover

- **Subcycled** particle equations of motion:

$$\frac{x_p^{\nu+1} - x_p^\nu}{\Delta\tau^\nu} = v_x^{\nu+1/2},$$

$$\frac{\mathbf{v}_p^{\nu+1} - \mathbf{v}_p^\nu}{\Delta\tau^\nu} = \frac{q_p}{m_p} \left(\mathbf{E}_p^{\nu+1/2}(x_p^{\nu+1/2}) + \mathbf{v}_p^{\nu+1/2} \times \mathbf{B}_p^{\nu+1/2}(x_p^{\nu+1/2}) \right).$$

- **Implicit Boris update** (analytical inversion):

$$\hat{\mathbf{v}}_p = \mathbf{v}_p^\nu + \alpha \mathbf{E}_p^{\nu+1/2}, \quad \alpha = \frac{q_p \Delta\tau^\nu}{m_p 2}$$

$$\mathbf{v}_p^{\nu+1/2} = \frac{\hat{\mathbf{v}}_p + \alpha \left[\hat{\mathbf{v}}_p \times \mathbf{B}_p^{\nu+1/2} + \alpha (\hat{\mathbf{v}}_p \cdot \mathbf{B}_p^{\nu+1/2}) \mathbf{B}_p^{\nu+1/2} \right]}{1 + (\alpha B_p)^2}.$$

- **Final particle position and velocity** are found from:

$$x_p^{\nu+1} = x_p^\nu + \Delta\tau^\nu v_{x,p}^{\nu+1/2},$$

$$\mathbf{v}_p^{\nu+1} = 2\mathbf{v}_p^{\nu+1/2} - \mathbf{v}_p^\nu.$$

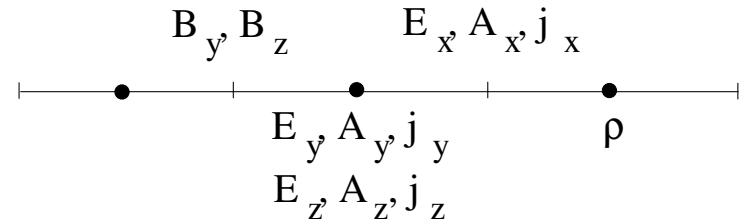
Energy conserving discrete 1D Darwin model

► Field equations:

$$\epsilon_0 \frac{E_{x,i+1/2}^{n+1} - E_{x,i+1/2}^n}{\Delta t} + \bar{j}_{x,i+1/2}^{n+1/2} = \langle j_x \rangle,$$

$$\frac{1}{\mu_0} \partial_x^2 \frac{A_y^{n+1} + A_y^n}{2} \Big|_i + \bar{j}_{y,i}^{n+1/2} = \langle j_y \rangle,$$

$$\frac{1}{\mu_0} \partial_x^2 \frac{A_z^{n+1} + A_z^n}{2} \Big|_i + \bar{j}_{z,i}^{n+1/2} = \langle j_z \rangle$$



► Current gather (with orbit averaging):

$$\bar{j}_{x,i+1/2}^{n+1/2} = \frac{1}{\Delta t \Delta x} \sum_p \sum_v q_p v_{p,x}^{v+1/2} S_m(x_p^{v+1/2} - x_{i+1/2}) \Delta \tau^v,$$

$$\bar{j}_{y,i}^{n+1/2} = \frac{1}{\Delta t \Delta x} \sum_p \sum_v q_p v_{p,y}^{v+1/2} S_l(x_p^{v+1/2} - x_i) \Delta \tau^v,$$

$$\bar{j}_{z,i}^{n+1/2} = \frac{1}{\Delta t \Delta x} \sum_p \sum_v q_p v_{p,z}^{v+1/2} S_l(x_p^{v+1/2} - x_i) \Delta \tau^v,$$

Field scatter to particles (with orbit averaging)

➤ Electric field scatter:

$$E_{x,p}^{v+1/2} = \sum_i \frac{E_{x,i+1/2}^{n+1} + E_{x,i+1/2}^n}{2} S_m(x_p^{v+1/2} - x_{i+1/2}),$$

$$E_{y,p}^{v+1/2} = - \sum_i \frac{A_{y,i}^{n+1} - A_{y,i}^n}{\Delta t} S_l(x_p^{v+1/2} - x_i),$$

$$E_{z,p}^{v+1/2} = - \sum_i \frac{A_{z,i}^{n+1} - A_{z,i}^n}{\Delta t} S_l(x_p^{v+1/2} - x_i).$$

➤ Magnetic field scatter: conservation of canonical momenta in ignorable directions

$$\dot{p}_y = m_p \dot{v}_{p,y} + q_p \dot{A}_{p,y} = 0, \quad \dot{p}_z = m_p \dot{v}_{p,z} + q_p \dot{A}_{p,z} = 0$$

$$B_{y,p}^{v+1/2} = - \sum_i \frac{A_{z,i+1}^{v+1/2} - A_{z,i}^{v+1/2}}{\Delta x} S_{l-1}(x_{i+1/2} - x_p^{v+1/2}) - \boxed{\sum_i \frac{\Delta A_{z,i-1}^v - 2\Delta A_{z,i}^v + \Delta A_{z,i+1}^v}{8} (x_p^{v+1} - x_p^v)},$$

$$B_{z,p}^{v+1/2} = \sum_i \frac{A_{y,i+1}^{v+1/2} - A_{y,i}^{v+1/2}}{\Delta x} S_{l-1}(x_{i+1/2} - x_p^{v+1/2}) + \underbrace{\sum_i \frac{\Delta A_{y,i-1}^v - 2\Delta A_{y,i}^v + \Delta A_{y,i+1}^v}{8} (x_p^{v+1} - x_p^v)}_{\mathcal{O}(\Delta \tau^v \Delta x)^2}.$$

Energy conservation in Darwin

$$\begin{aligned} \frac{K_p^{n+1} - K_p^n}{\Delta t} &= \frac{1}{\Delta t} \sum_p m_p \sum_v \Delta \tau^v \frac{\mathbf{v}_p^{v+1} + \mathbf{v}_p^v}{2} \cdot \frac{\mathbf{v}_p^{v+1} - \mathbf{v}_p^v}{\Delta \tau^v} = \frac{1}{\Delta t} \sum_{p,v} q_p \Delta \tau^v \mathbf{E}_p^{v+1/2} \cdot \mathbf{v}_p^{v+1/2} \\ &= \sum_i \Delta x \left(\frac{E_{x,i+1/2}^{n+1} + E_{x,i+1/2}^n}{2} j_{x,i+1/2}^{n+1/2} + E_{y,i}^{n+1/2} j_{y,i}^{n+1/2} + E_{z,i}^{n+1/2} j_{z,i}^{n+1/2} \right) \end{aligned}$$

$$\sum_i \Delta x \frac{E_{x,i+1/2}^{n+1} + E_{x,i+1/2}^n}{2} j_{x,i+1/2}^{n+1/2} = -\frac{\epsilon_0}{2\Delta t} \sum_i \Delta x \left((E_{x,i+1/2}^{n+1})^2 - (E_{x,i+1/2}^n)^2 \right) = -\frac{W_{\phi x}^{n+1} - W_{\phi x}^n}{\Delta t}$$

$$\sum_i \Delta x E_{y,i}^{n+1/2} j_{y,i}^{n+1/2} = -\frac{1}{2\mu_0 \Delta t} \sum_i \Delta x \left[(\partial_x A_y^{n+1})_{i+1/2}^2 - (\partial_x A_y^n)_{i+1/2}^2 \right] = -\frac{W_{Bz}^{n+1} - W_{Bz}^n}{\Delta t}$$

$$\sum_i \Delta x E_{z,i}^{n+1/2} j_{z,i}^{n+1/2} = -\frac{1}{2\mu_0 \Delta t} \sum_i \Delta x \left[(\partial_x A_z^{n+1})_{i+1/2}^2 - (\partial_x A_z^n)_{i+1/2}^2 \right] = -\frac{W_{By}^{n+1} - W_{By}^n}{\Delta t}$$

$$(K_p + W_{\phi x} + W_{By} + W_{Bz})^{n+1} = (K_p + W_{\phi x} + W_{By} + W_{Bz})^n$$

CPU speedup potential of EM implicit PIC vs. explicit PIC

- Back-of-the-envelope estimate of CPU gain:

$$\frac{CPU_{ex}}{CPU_{imp}} \sim \left(\frac{\Delta x_{imp}}{\Delta x_{ex}} \right)^d \frac{\Delta \tau_{imp}}{\Delta t_{ex}} \frac{1}{N_{FE}}$$

$$\Delta \tau_{imp} \sim 0.1 \min \left[\frac{\Delta x_{imp}}{v_{th,e}}, \omega_{ce}^{-1} \right]$$

$$\Delta t_{exp} \sim \frac{\Delta x_{exp}}{c}$$

$$k \Delta x_{imp} \sim 0.2$$

$$\Delta x_{ex} \sim \lambda_D$$

$$\frac{CPU_{ex}}{CPU_{imp}} \sim \frac{\min[1, kd_e]}{(k\lambda_D)^{d+1}} \frac{1}{N_{FE}}$$

- CPU speedup is:

- ⇒ Independent of time step
- ⇒ Impacted by electron scales

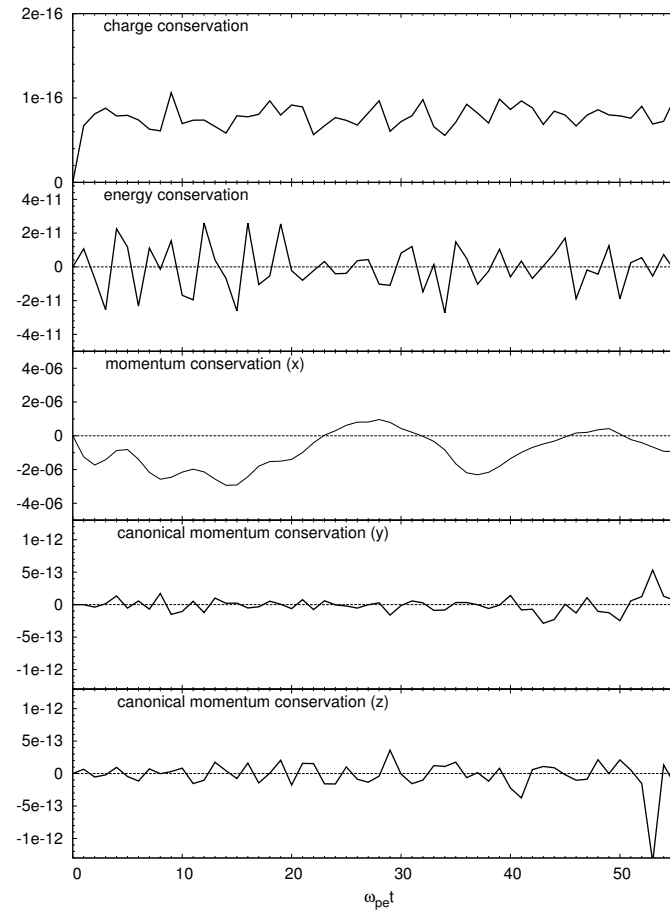
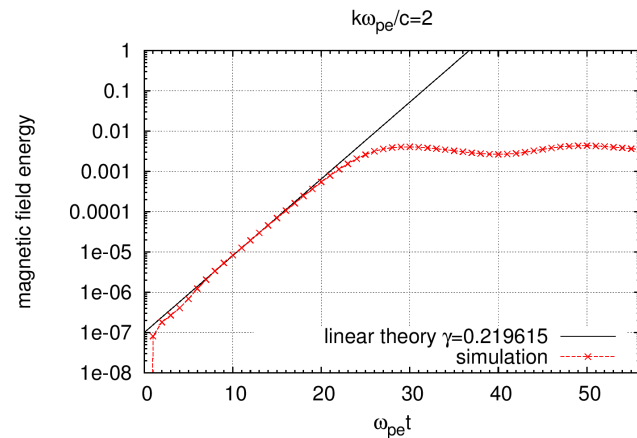
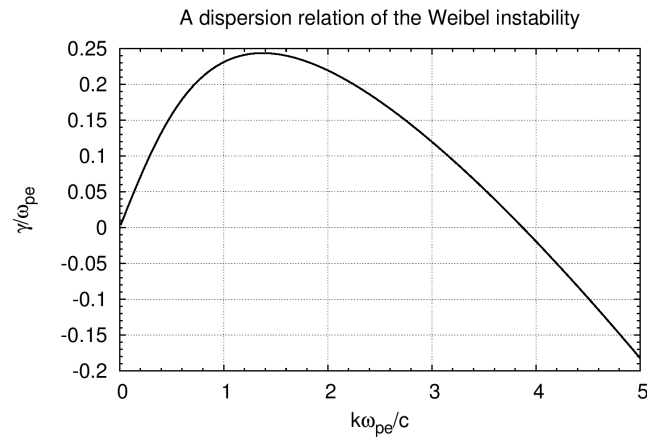
- Again, **key is to minimize** N_{FE} .

- ⇒ We are in the process of developing a moment-based preconditioner.

Verification: Electron Weibel instability

- Isotropic ions, bi-Maxwellian electrons

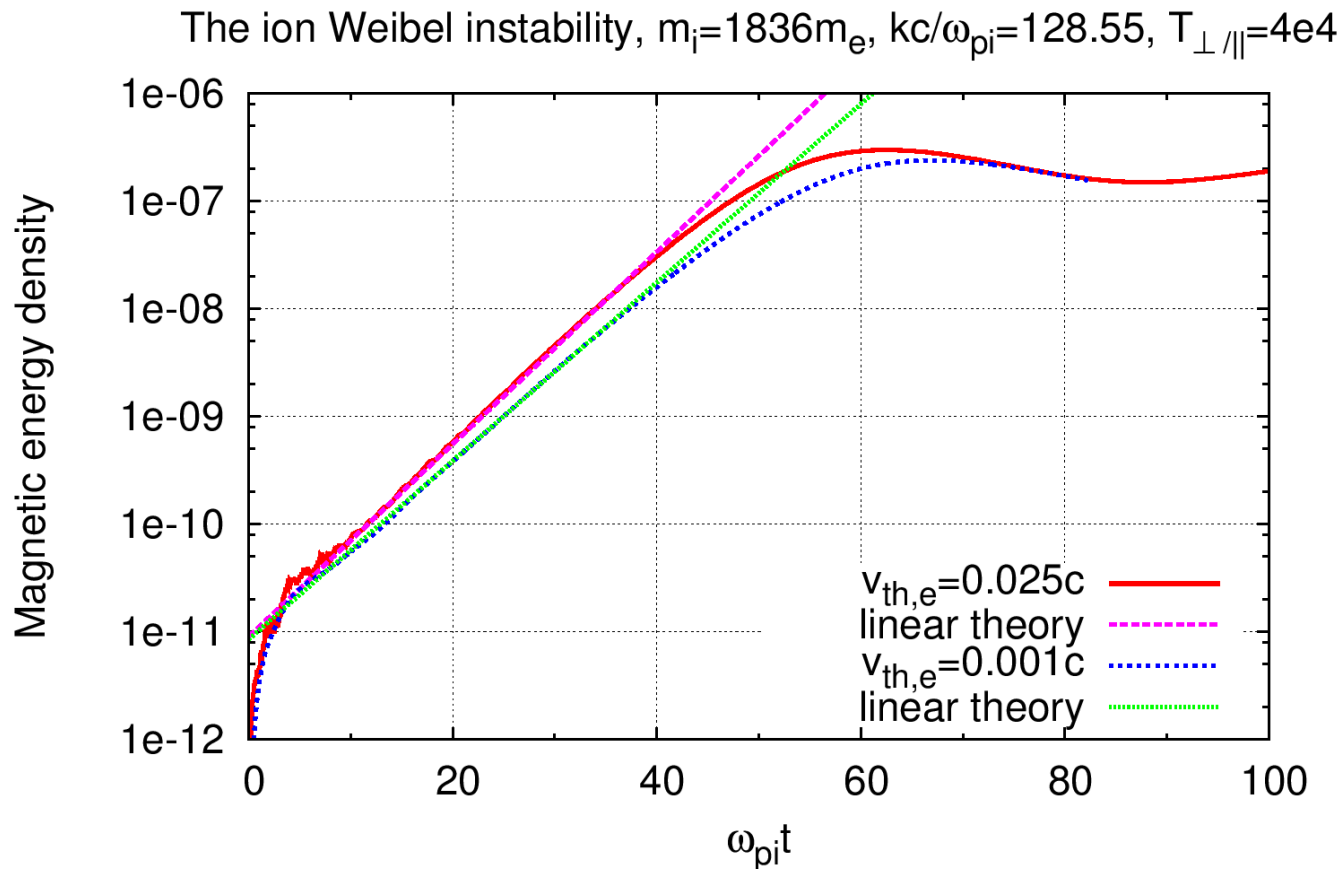
$$m_i/m_e = 1836, T_{e\perp}/T_{e\parallel} = 16, N_{e,i}=128,000, L = 2\pi c/\omega_{pe}, N_g=32.$$



Verification: Ion Weibel instability (large T anisotropy)

- Isotropic electrons, bi-Maxwellian ions

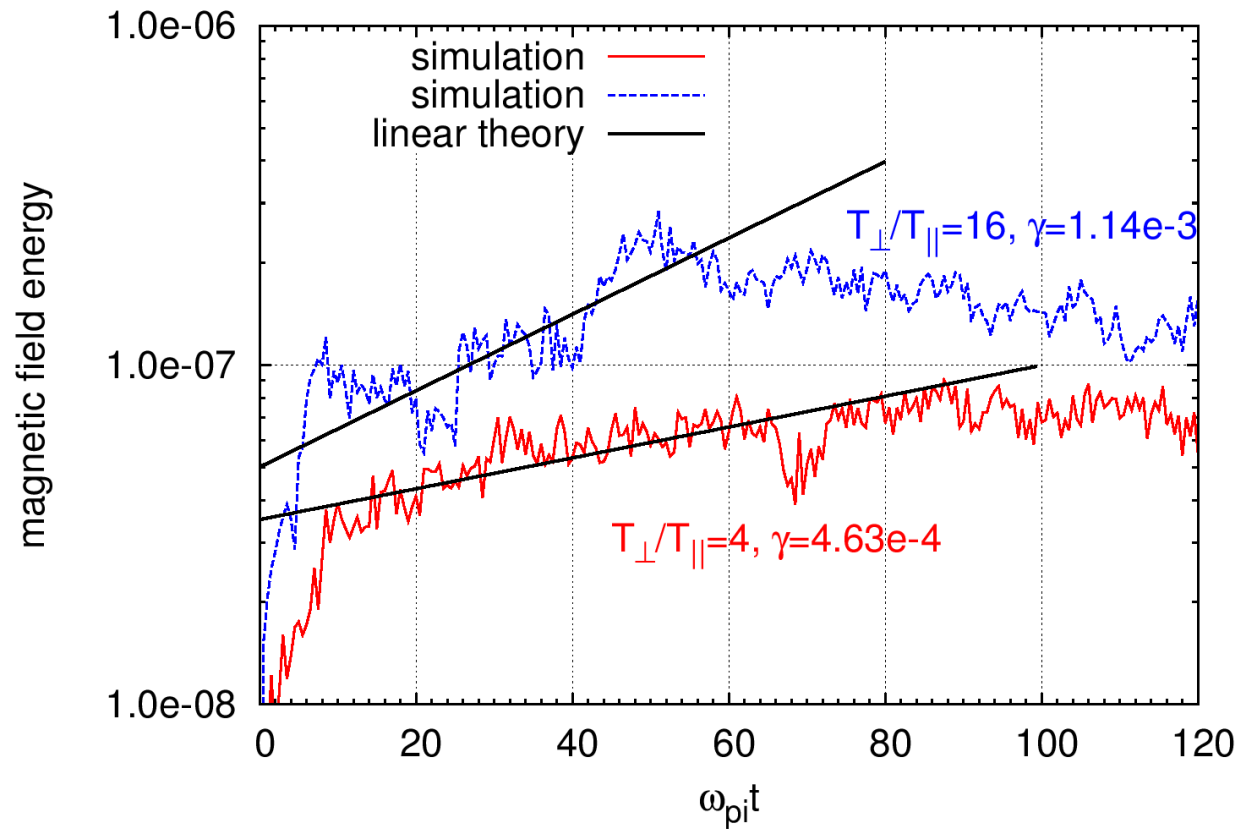
$$m_i/m_e = 1836, T_{i\perp}/T_{i\parallel} = 4 \times 10^4, N_{e,i}=128,000, L = \frac{2}{3}\pi c/\omega_{pe}, N_g=64, \gamma \cong 0.1\omega_{pi}.$$



Verification: Ion Weibel instability (small T anisotropy)

- Isotropic electrons, bi-Maxwellian ions

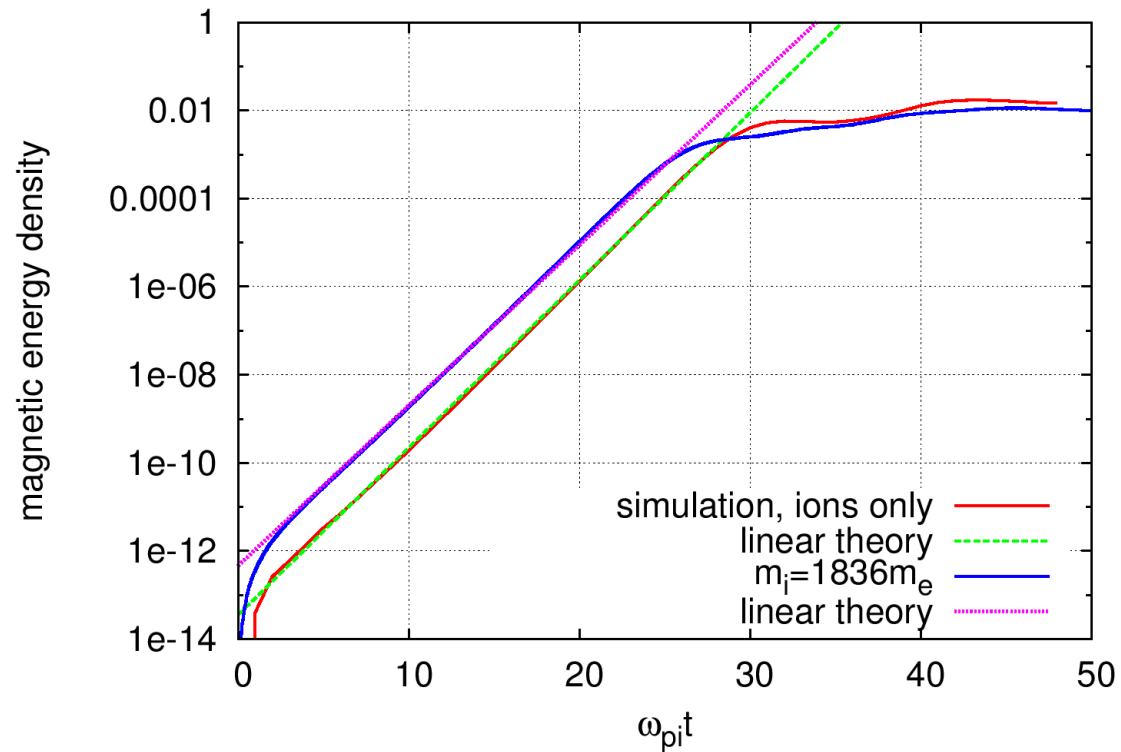
$$m_i/m_e = 128, N_{e,i}=128,000, L = 0.88\pi c/\omega_{pi}, N_g=32$$



Ion cyclotron instability (large T anisotropy)

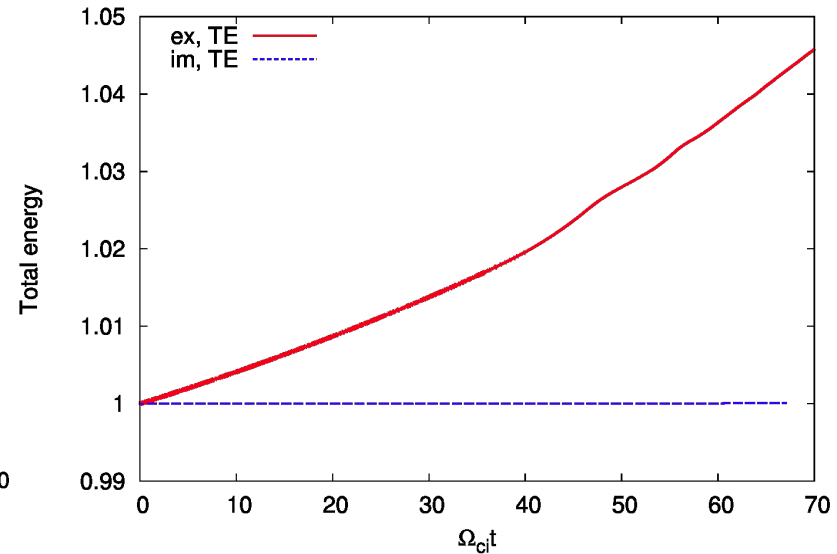
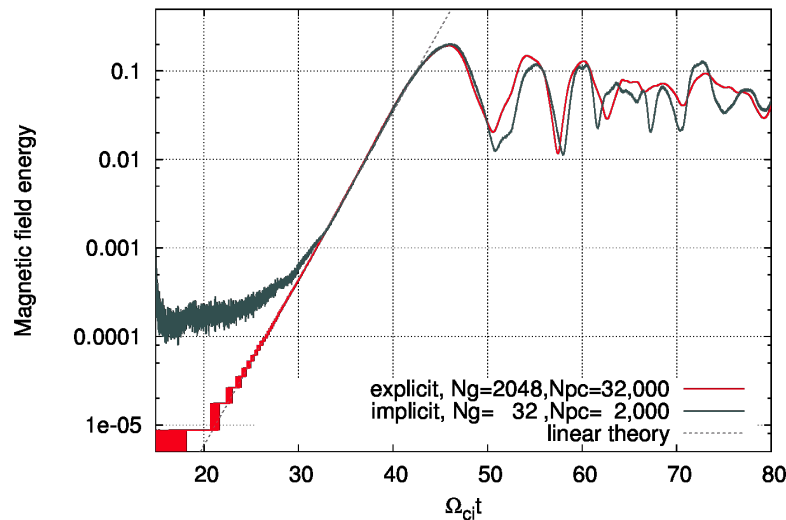
- Isotropic electrons, bi-Maxwellian ions, magnetized ion dynamics ($\beta_{i\perp} = 1$)

$$m_i/m_e = 1836, T_{i\perp}/T_{i\parallel} = 2.5 \times 10^5, N_{e,i}=128,000, L = \pi c/\omega_{pi}, N_g=32, \gamma \approx 0.4\omega_{pi}.$$



Paradigm change: 1D Kinetic Alfvén Wave²

$$m_i/m_e = 1836 ; k\lambda_{De} = 0.003 ; v_{th,e}/c = 0.07$$



- **Explicit PIC:**
 - ⇒ 2048 mesh points, 32,000 pcles/cell (overkill for this problem), **0.05 energy error**
 - ⇒ 500 CPUs × 24 hr, 7×10^6 time steps
- **Implicit PIC:**
 - ⇒ 32 mesh points, 2,000 pcles/cell ($1000 \times$ fewer particles), 10^{-6} **energy error**
 - ⇒ 16 CPUs × 29 hr, 1.3×10^5 time steps, $N_{FE} \sim 30$ ($r_{tol} = 10^{-6}$)
- CPU speedup ~ 26 ($\times 100$ in 2D, $\times 10^4$ in 3D)

²Yin et al., POP 14 (2007)

Summary and conclusions

- We have demonstrated, for the first time, a **fully implicit, fully nonlinear electrostatic PIC formulation** that features:
 - ⇒ **Exact charge conservation** (via a novel particle mover strategy).
 - ⇒ **Exact energy conservation** (no particle self-heating or self-cooling).
 - ⇒ **Adaptive particle orbit integrator** to control errors in momentum conservation.
- The approach has been shown to be **free of CFL and finite-grid numerical instabilities**.
- As a result, the **method is able to take time steps many times larger than explicit**, and resolutions many times coarser.
- Central to our implementation is the **concept of particle enslavement**.
- We have generalized formulation to use **spatial adaptivity via mapped coordinates**.
- The method has **much potential for efficiency gains vs. explicit** in long-time-scale applications, with the CPU speedup scaling as $(k\lambda_D)^{-(d+1)} / N_{FE}$.
 - ⇒ Minimize the number of nonlinear function evaluations N_{FE} for given Δt , $\Delta x \Rightarrow$ preconditioning!
 - ⇒ We have formulated and implemented a **very efficient moment-based preconditioner**.
- We have **ported the algorithm to GPU architectures** (not shown)
- We have **generalized the algorithm to non-radiative electromagnetic regimes** (Darwin model), where, **in addition to charge and energy, we also conserve canonical momenta**.