Numerical Thermalization Timescales of Electrostatic Particle-in-Cell Simulations

Sierra Jubin

Princeton Plasma Physics Laboratory

Princeton University

Outline

- Background
  - Defining numerical thermalization
  - Numerical thermalization vs. numerical heating
  - Importance in multidimensional PIC
- Numerical thermalization = numerical collisions
- Thermalization timescales of PIC plasmas
  - 2D PIC thermalization
  - 3D PIC thermalization
  - 1D PIC thermalization and complications (MCC)
- Mitigation strategies
- Summary

What is numerical thermalization?

- It relaxes the EVDF towards a Maxwellian distribution
- It is often called numerical collisions or noise
- Cause: Inherent granularity of PIC simulations, decreases with increasing particles-per-cell (ppc)

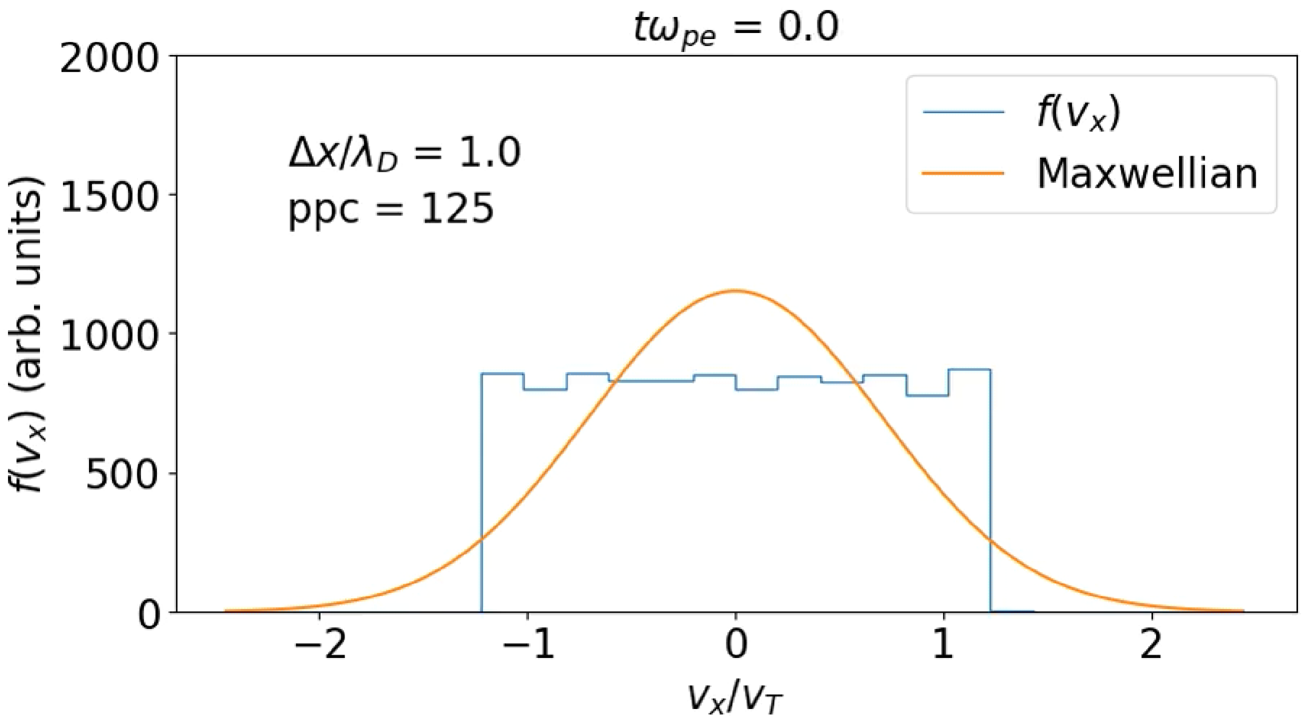


Fig: EVDF evolution from initial waterbag in 3D explicit energy-conserving PIC simulation of a homogeneous plasma.
NO COLLISIONS ADDED

Numerical thermalization is **NOT** heating

Fig: EVDF evolution from initial waterbag in 2D EC-PIC simulation: comparison to Maxwellian

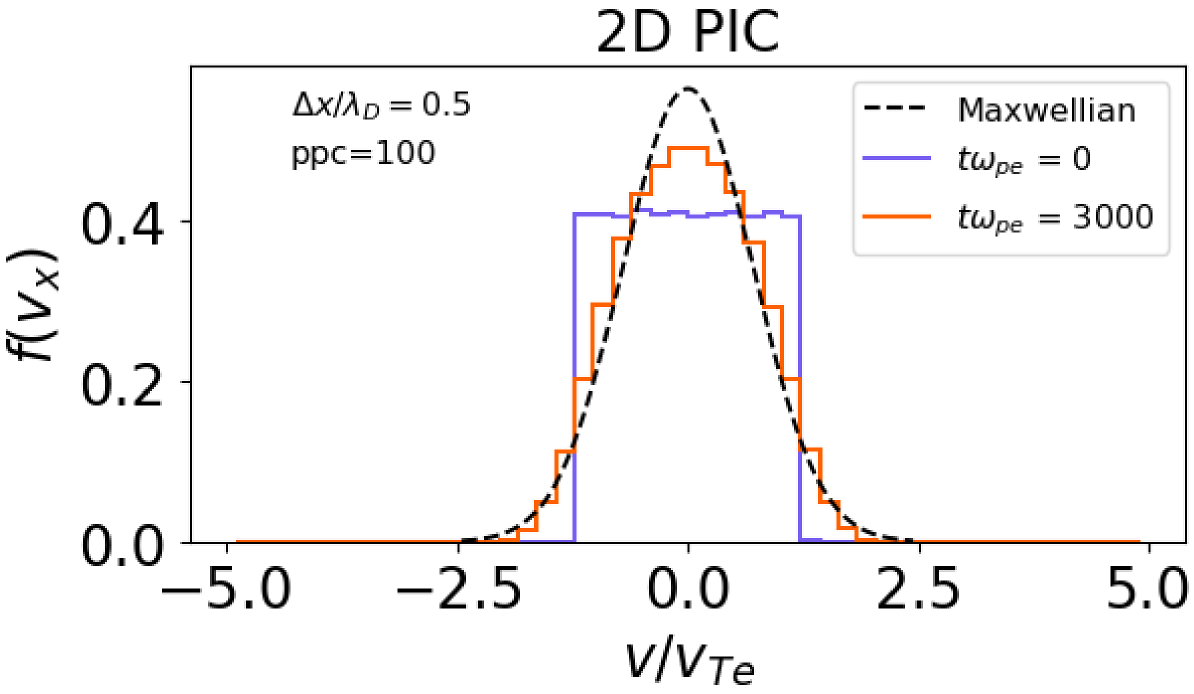
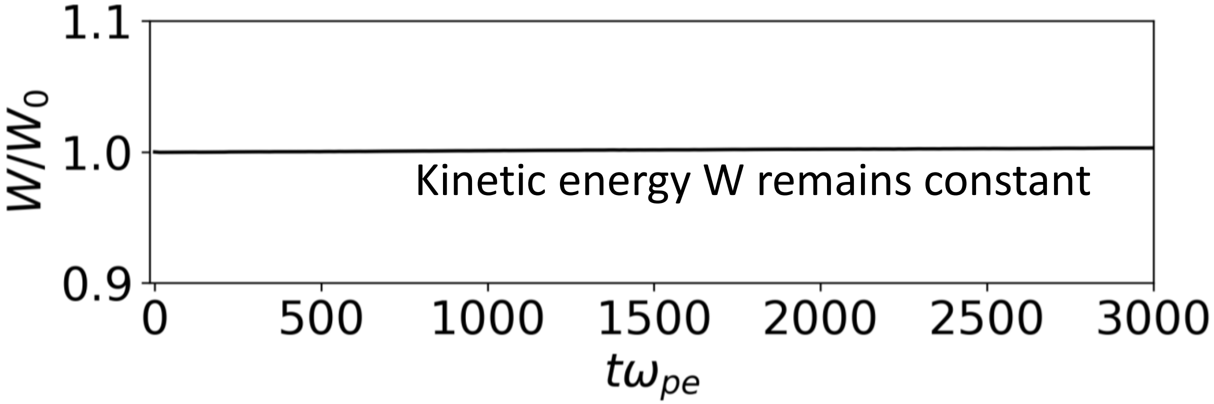


Fig: Evolution of the kinetic energy of the electrons in the simulation shown on the left

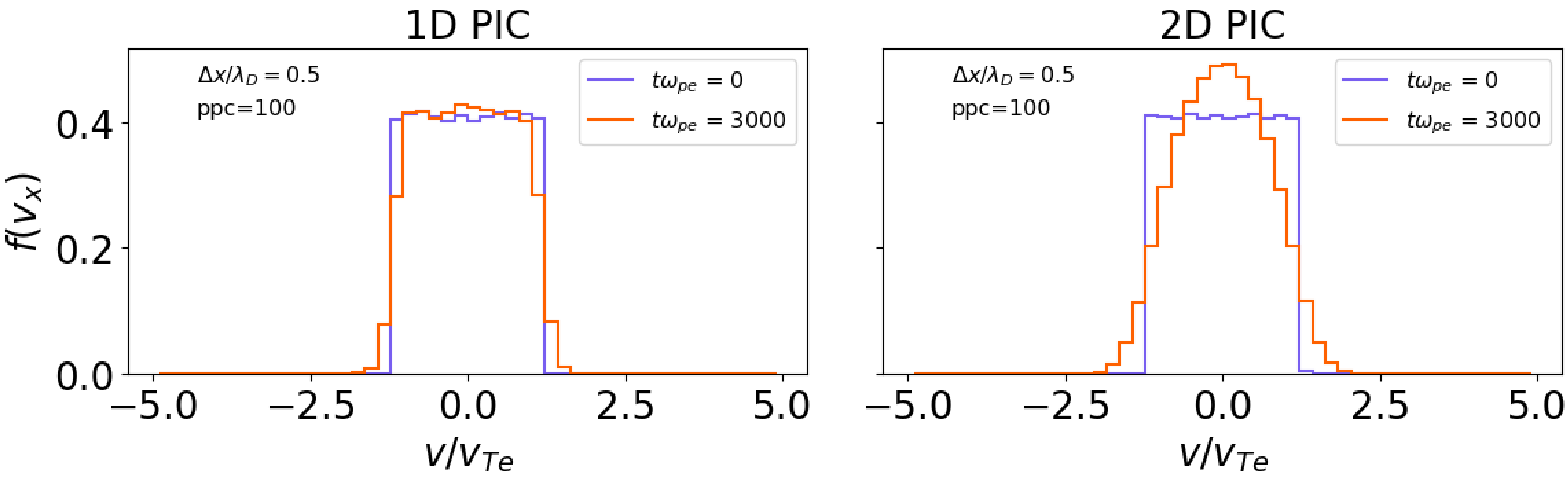


Thermalization often happens much faster than heating timescales

EVDF can evolve inconsistent with Vlasov eqn. while energy is conserved

Numerical thermalization is often faster in multidimensional PIC simulations

Fig: EVDF evolution from initial waterbag in 1D and 2D PIC simulations with identical $\Delta x/\lambda_D$ and ppc



By $t\omega_{pe} = 3000$, the EVDF in the 2D simulation has evolved significantly.

Thermalization is a departure from Vlasov eqn. representing macroparticle collisions

$$\frac{df_\alpha}{dt} = \underbrace{\left[\frac{\partial}{\partial t} + \boldsymbol{v}\cdot\nabla + \boldsymbol{a}\cdot\nabla_v\right] f_\alpha(t,\boldsymbol{x})}_{\substack{\text{Vlasov physics, fluid in phase space} \\ 0^{\text{th}}\text{ order in } 1/N_D}} = \underbrace{\left(\frac{\partial f_\alpha}{\partial t}\right)_c}_{\substack{\text{Effects due to inherent “graininess” of the plasma: collisions} \\ 1^{\text{st}}\text{ order and higher in } 1/N_D}}$$

Key parameter N_D: number of macroparticles per Debye length/square/cube

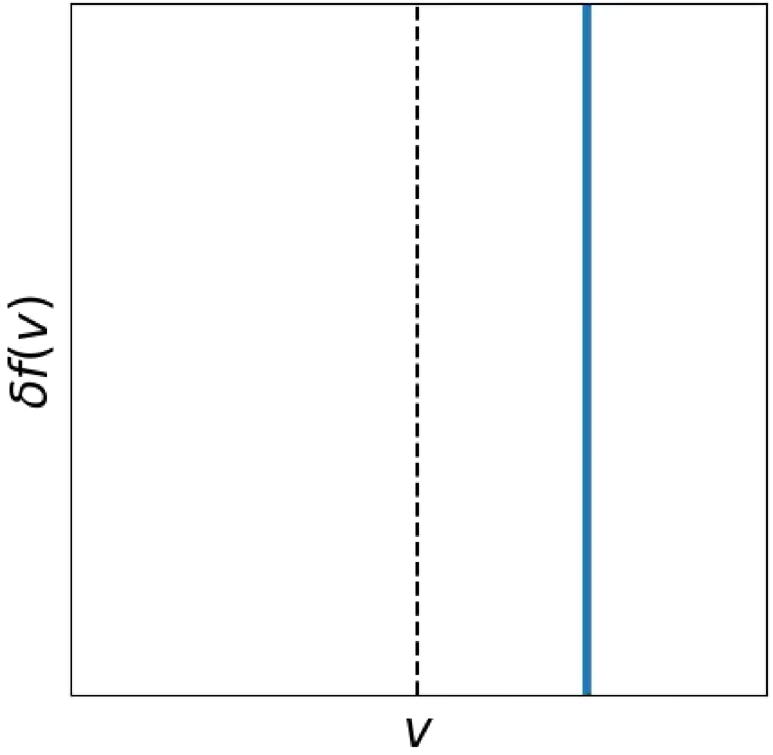
1D: $N_D = n_{mac}\lambda_D$

2D: $N_D = n_{mac}\lambda_D^2$

3D: $N_D = n_{mac}\lambda_D^3$

Thermalization timescale can be obtained from velocity drag and diffusion

Example initial test particle $\delta(\mathrm{v})$ EVDF evolving in a background Maxwellian



Fokker-Planck form of numerical collision operator [1] yields

- Drag coefficients $(\mathrm{A} = \Delta\mathrm{v}/\Delta t)$
- Diffusion coefficients $(\mathrm{D}_{\parallel} = \Delta\mathrm{v}_{\parallel}^2/\Delta t$ and $\mathrm{D}_{\perp} = \Delta\mathrm{v}_{\perp}^2/\Delta t\ )$

From A, $\mathrm{D}_{\parallel}$, $\mathrm{D}_{\perp}$ can directly and extract timescales. [2]

$$\left(\frac{\partial f}{\partial t}\right)_c = -\frac{\partial}{\partial \boldsymbol{v}} \cdot [A(\mathbf{v}) f(\mathbf{v})] + \frac{1}{2}\frac{\partial}{\partial \boldsymbol{v}}\frac{\partial}{\partial \boldsymbol{v}} : [\boldsymbol{D}(\mathbf{v}) f(\mathbf{v})]$$

$$\boldsymbol{\tau_S = \frac{\mathbf{v}}{-A(\mathbf{v})}}, \qquad \boldsymbol{\tau_R \sim \frac{\mathbf{v}_{\mathbf{Te}}}{-\langle A \rangle}}$$

[1] Jubin et al. *Phys. Plasmas* **31**, 023902 (2024). doi: 10.1063/5.0180421

[2] Hockney. *J. Comput. Phys.* **8**, (19-44) (1971); doi: 10.1016/0021-9991(71)90032-5.

The numerical collision operator

The numerical collision operator for electrostatic PIC is an analogue of the Balescu-Lenard collision operator. It is described in textbooks [1] and was recently carefully derived for a multiple-species plasma [2]. However, **it is rarely considered in practice!**

Neglecting aliasing effects due to the timestep:

$$\left(\frac{\partial f_\alpha}{\partial t}\right)_c = \sum_\beta \frac{n_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}}\cdot\int d\boldsymbol{v}'\,\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')\cdot\left(\frac{\delta N_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}}-\frac{\delta N_\alpha}{m_\beta}\frac{\partial}{\partial \boldsymbol{v}'}\right)f_\alpha(\boldsymbol{v})f_\beta(\boldsymbol{v}')$$

with collision tensor $\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')$ given below:

$$\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \equiv \int \frac{d\boldsymbol{k}}{(2\pi)^{\mathrm{d}}}\boldsymbol{K}\otimes\boldsymbol{K}\sum_{\boldsymbol{p}}\left|\frac{S(\boldsymbol{k})S(\boldsymbol{k_p})q_\alpha q_\beta}{\varepsilon_0\epsilon(\boldsymbol{k}\cdot\boldsymbol{v},\boldsymbol{k})K^2}\right|^2 \pi\delta(\boldsymbol{k}\cdot\boldsymbol{v}-\boldsymbol{k_p}\cdot\boldsymbol{v}')$$

[1] Birdsall and Langdon. *Plasma Physics via Computer Simulation*. (2004)
[2] M. Touati et al. *Plasma Phys. Control. Fusion* **64**, 115014 (2022); doi: 10.1088/1361-6587/ac9016

Key differences from real Coulomb collisions

Neglecting aliasing effects due to the timestep:

$$\left(\frac{\partial f_\alpha}{\partial t}\right)_c = \sum_\beta \frac{n_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} \cdot \int d\boldsymbol{v}'\, \boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \cdot \left(\frac{\delta N_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} - \frac{\delta N_\alpha}{m_\beta}\frac{\partial}{\partial \boldsymbol{v}'}\right) f_\alpha(\boldsymbol{v}) f_\beta(\boldsymbol{v}')$$

with collision tensor $\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')$ given below:

$$\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \equiv \int \frac{d\boldsymbol{k}}{(2\pi)^{\mathrm{d}}} \boldsymbol{K} \otimes \boldsymbol{K} \sum_{\boldsymbol{p}} \left|\frac{S(\boldsymbol{k})S(\boldsymbol{k_p})q_\alpha q_\beta}{\varepsilon_0 \epsilon(\boldsymbol{k}\cdot\boldsymbol{v},\boldsymbol{k})K^2}\right|^2 \pi\delta(\boldsymbol{k}\cdot\boldsymbol{v} - \boldsymbol{k_p}\cdot\boldsymbol{v}')$$

Differences from standard Balescu-Lenard:

Key differences from real Coulomb collisions

Neglecting aliasing effects due to the timestep:

$$\left(\frac{\partial f_\alpha}{\partial t}\right)_c = \sum_\beta \frac{n_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} \cdot \int d\boldsymbol{v}' \, \boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \cdot \left(\frac{\delta N_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} - \frac{\delta N_\alpha}{m_\beta}\frac{\partial}{\partial \boldsymbol{v}'}\right) f_\alpha(\boldsymbol{v}) f_\beta(\boldsymbol{v}')$$

with collision tensor $\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')$ given below:

$$\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \equiv \int \frac{d\boldsymbol{k}}{(2\pi)^{\mathrm{d}}} \boldsymbol{K} \otimes \boldsymbol{K} \sum_{\boldsymbol{p}} \left| \frac{S(\boldsymbol{k}) S(\boldsymbol{k_p}) q_\alpha q_\beta}{\varepsilon_0 \epsilon(\boldsymbol{k}\cdot\boldsymbol{v}, \boldsymbol{k}) K^2} \right|^2 \pi \delta(\boldsymbol{k}\cdot\boldsymbol{v} - \boldsymbol{k_p}\cdot\boldsymbol{v}')$$

$S(\boldsymbol{k})$, the Fourier transform of the shape function used in particle / field interpolation and additional filtering

Differences from standard Balescu-Lenard:

- Shape functions / filtering

Key differences from real Coulomb collisions

Neglecting aliasing effects due to the timestep:

$$\left(\frac{\partial f_\alpha}{\partial t}\right)_c = \sum_\beta \frac{n_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}}\cdot\int d\boldsymbol{v}'\,\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')\cdot\left(\frac{\delta N_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} - \frac{\delta N_\alpha}{m_\beta}\frac{\partial}{\partial \boldsymbol{v}'}\right) f_\alpha(\boldsymbol{v}) f_\beta(\boldsymbol{v}')$$

with collision tensor $\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')$ given below:

$$\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \equiv \int \frac{d\boldsymbol{k}}{(2\pi)^{\mathrm{d}}}\boldsymbol{K}\otimes\boldsymbol{K}\sum_{\boldsymbol{p}}\left|\frac{S(\boldsymbol{k})S(\boldsymbol{k_p})q_\alpha q_\beta}{\varepsilon_0\epsilon(\boldsymbol{k}\cdot\boldsymbol{v},\boldsymbol{k})K^2}\right|^2 \pi\delta(\boldsymbol{k}\cdot\boldsymbol{v} - \boldsymbol{k_p}\cdot\boldsymbol{v}')$$

Differences from standard Balescu-Lenard:

- Shape functions / filtering
- Aliasing from finite spatial grid

$$\boldsymbol{k_p} = \boldsymbol{k} - \boldsymbol{p}\odot\boldsymbol{k_g} \qquad \boldsymbol{p}\in\mathbb{Z}, \qquad \boldsymbol{k_g} = 2\pi\boldsymbol{\Delta}_{\boldsymbol{x}}^{-1}$$

Wavenumbers separated by an integer number of Nyquist wavenumbers $\boldsymbol{k_g}$ are coupled.

Key differences from real Coulomb collisions

Neglecting aliasing effects due to the timestep:

$$\left(\frac{\partial f_\alpha}{\partial t}\right)_c = \sum_\beta \frac{n_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}}\cdot\int d\boldsymbol{v}'\,\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')\cdot\left(\frac{\delta N_\beta}{m_\alpha}\frac{\partial}{\partial\boldsymbol{v}} - \frac{\delta N_\alpha}{m_\beta}\frac{\partial}{\partial \boldsymbol{v}'}\right) f_\alpha(\boldsymbol{v}) f_\beta(\boldsymbol{v}')$$

with collision tensor $\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')$ given below:

$$\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \equiv \int \frac{d\boldsymbol{k}}{(2\pi)^{\mathrm{d}}}\, \boldsymbol{K}\otimes\boldsymbol{K}\sum_{\boldsymbol{p}}\left|\frac{S(\boldsymbol{k})S(\boldsymbol{k_p})q_\alpha q_\beta}{\varepsilon_0\epsilon(\boldsymbol{k}\cdot\boldsymbol{v},\boldsymbol{k})K^2}\right|^2 \pi\delta(\boldsymbol{k}\cdot\boldsymbol{v}-\boldsymbol{k_p}\cdot\boldsymbol{v}')$$

Differences from standard Balescu-Lenard:

- Shape functions / filtering
- Aliasing from finite spatial grid
- Dependence on discretization method

$\boldsymbol{K}$, a wavenumber-like quantity dependent on the method by which Maxwell's equations / the Poisson equation are solved on the grid.

Key differences from real Coulomb collisions

Neglecting aliasing effects due to the timestep:

$$\left(\frac{\partial f_\alpha}{\partial t}\right)_c = \sum_\beta \frac{n_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} \cdot \int d\boldsymbol{v}' \, \boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \cdot \left(\frac{\delta N_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} - \frac{\delta N_\alpha}{m_\beta}\frac{\partial}{\partial \boldsymbol{v}'}\right) f_\alpha(\boldsymbol{v}) f_\beta(\boldsymbol{v}')$$

with collision tensor $\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')$ given below:

$$\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \equiv \int \frac{d\boldsymbol{k}}{(2\pi)^{\mathrm{d}}} \boldsymbol{K} \otimes \boldsymbol{K} \sum_{\boldsymbol{p}} \left| \frac{S(\boldsymbol{k}) S(\boldsymbol{k_p}) q_\alpha q_\beta}{\varepsilon_0 \epsilon(\boldsymbol{k}\cdot\boldsymbol{v},\boldsymbol{k}) K^2} \right|^2 \pi\delta(\boldsymbol{k}\cdot\boldsymbol{v} - \boldsymbol{k_p}\cdot\boldsymbol{v}')$$

Differences from standard Balescu-Lenard:

- Shape functions / filtering
- Aliasing from finite spatial grid
- Dependence on discretization method
- The number of spatial dimensions

The number of dimensions, d, only affects the number of factors of $\frac{1}{2\pi}$ in the inverse Fourier transform*

*There are complications in 1D

Key differences from real Coulomb collisions

Neglecting aliasing effects due to the timestep:

$$\left(\frac{\partial f_\alpha}{\partial t}\right)_c = \sum_\beta \frac{n_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} \cdot \int d\boldsymbol{v}'\, \boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \cdot \left(\frac{\delta N_\beta}{m_\alpha}\frac{\partial}{\partial \boldsymbol{v}} - \frac{\delta N_\alpha}{m_\beta}\frac{\partial}{\partial \boldsymbol{v}'}\right) f_\alpha(\boldsymbol{v}) f_\beta(\boldsymbol{v}')$$

with collision tensor $\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}')$ given below:

$$\boldsymbol{Q}_{\alpha\beta}(\boldsymbol{v},\boldsymbol{v}') \equiv \int \frac{d\boldsymbol{k}}{(2\pi)^{\mathrm{d}}} \boldsymbol{K}\otimes\boldsymbol{K} \sum_{\boldsymbol{p}} \left|\frac{S(\boldsymbol{k})S(\boldsymbol{k_p})q_\alpha q_\beta}{\varepsilon_0 \epsilon(\boldsymbol{k}\cdot\boldsymbol{v},\boldsymbol{k})K^2}\right|^2 \pi\delta(\boldsymbol{k}\cdot\boldsymbol{v} - \boldsymbol{k_p}\cdot\boldsymbol{v}')$$

Differences from standard Balescu-Lenard:

- Shape functions / filtering
- Aliasing from finite spatial grid
- Dependence on discretization method
- The number of spatial dimensions
- The macroparticle weighting!

Macroparticle weight $\delta N_\alpha = \frac{\mathrm{Q}_\alpha}{\mathrm{q}_\alpha} = \frac{\mathrm{M}_\alpha}{\mathrm{m}_\alpha}$ for species α

The whole collision operator is scaled by the number of real particles represented by a macroparticle!

Accurate kinetic behavior requires sufficiently small numerical thermalization/relaxation time τ_R

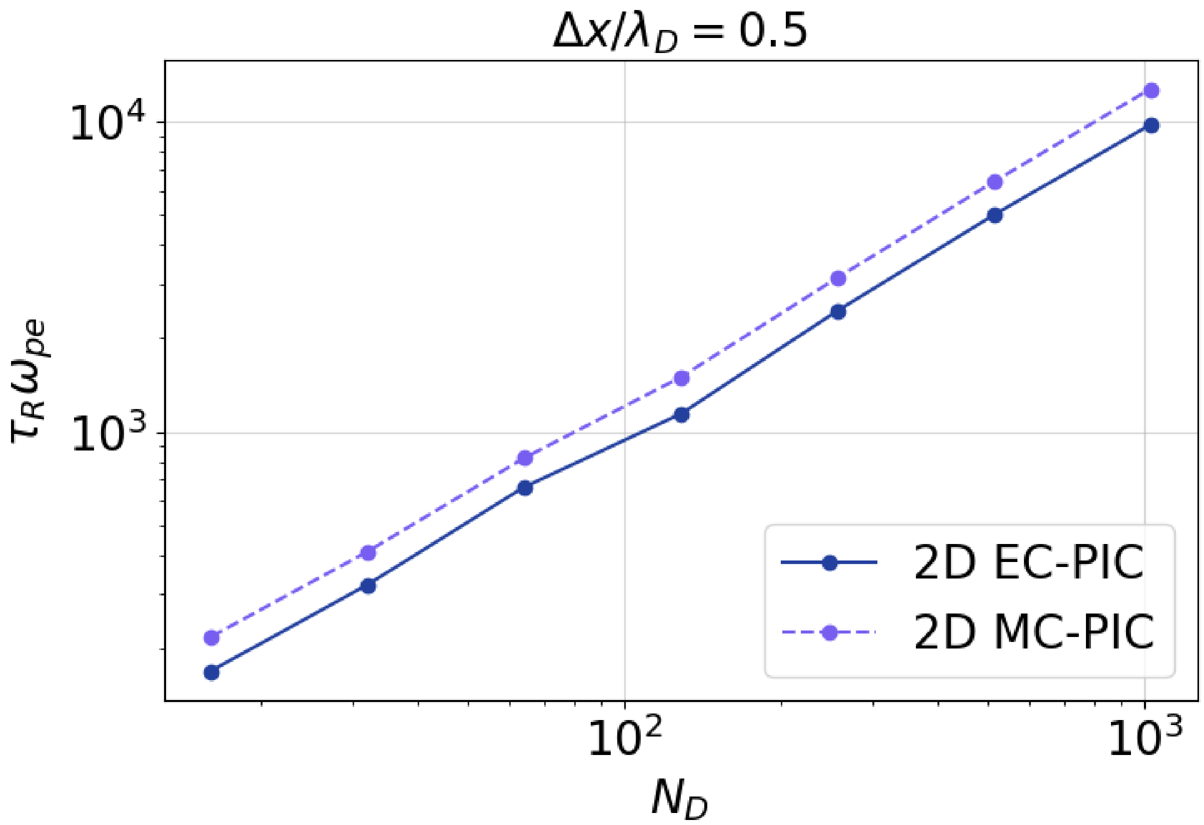
- Long PIC simulations do not exactly act as Vlasov solvers
- Convergence tests (increasing ppc until results no longer change) are useful
- **Thermalization timescale gives a quantitative answer to the question of: "How many particles per cell do we need?"**
- We require numerical thermalization time $>$ real timescale of EVDF evolution or particle residence time in simulation

Thermalization timescales of PIC plasmas

Thermalization in energy conserving (EC) PIC vs momentum conserving (MC) PIC

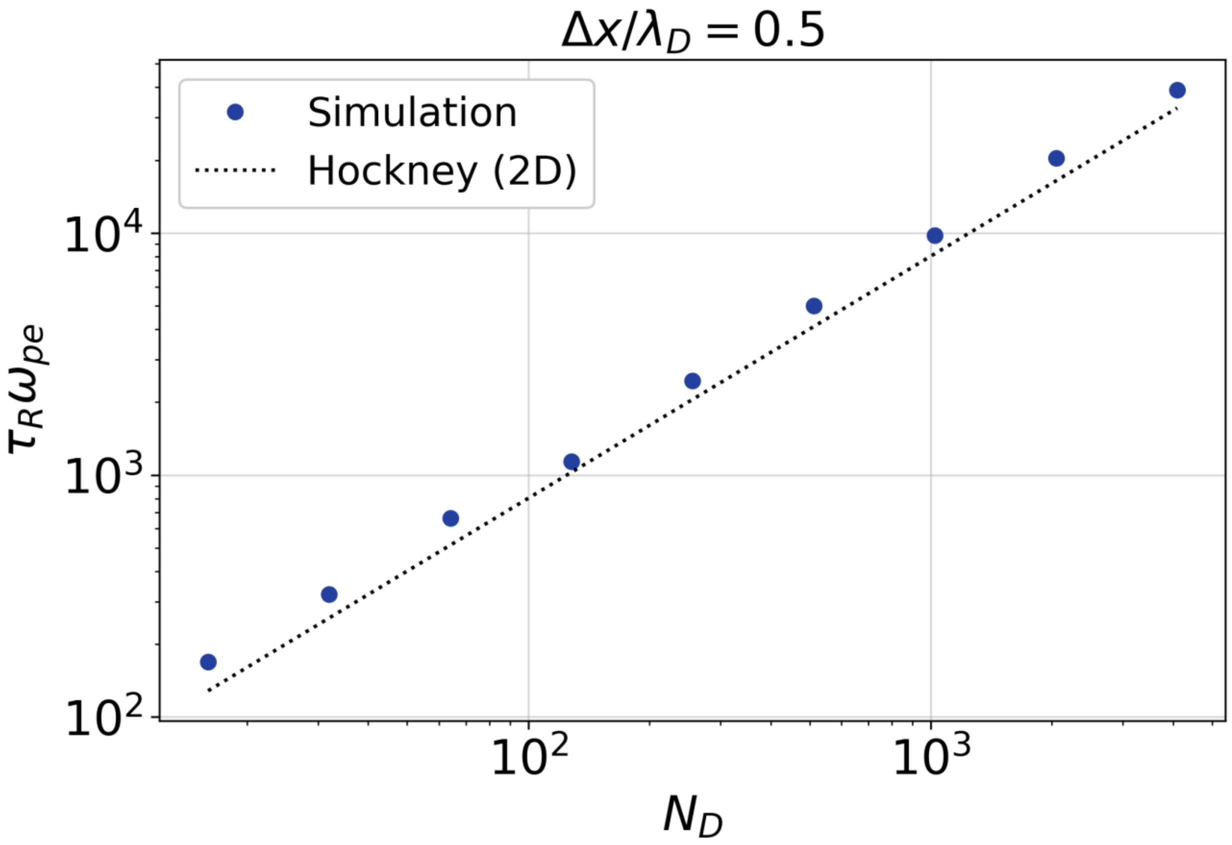
Fig: thermalization time in 2D EC-PIC and MC-PIC as a function of N_D.

- Scaling in EC-PIC and MC-PIC is comparable
- EC-PIC was used for the following work to allow broader parameter space, namely: $\frac{\Delta x}{\lambda_D} > 1$



Thermalization in 2D PIC – N_D scaling

Fig: Thermalization time as a function of N_D in 2D EC-PIC using CIC scheme



Linear in number of macroparticles per Debye volume, $N_D = \mathrm{n}\lambda_D^2$

$\tau_R \propto N_D$ in 2D PIC is directly obtainable from the numerical collision operator.

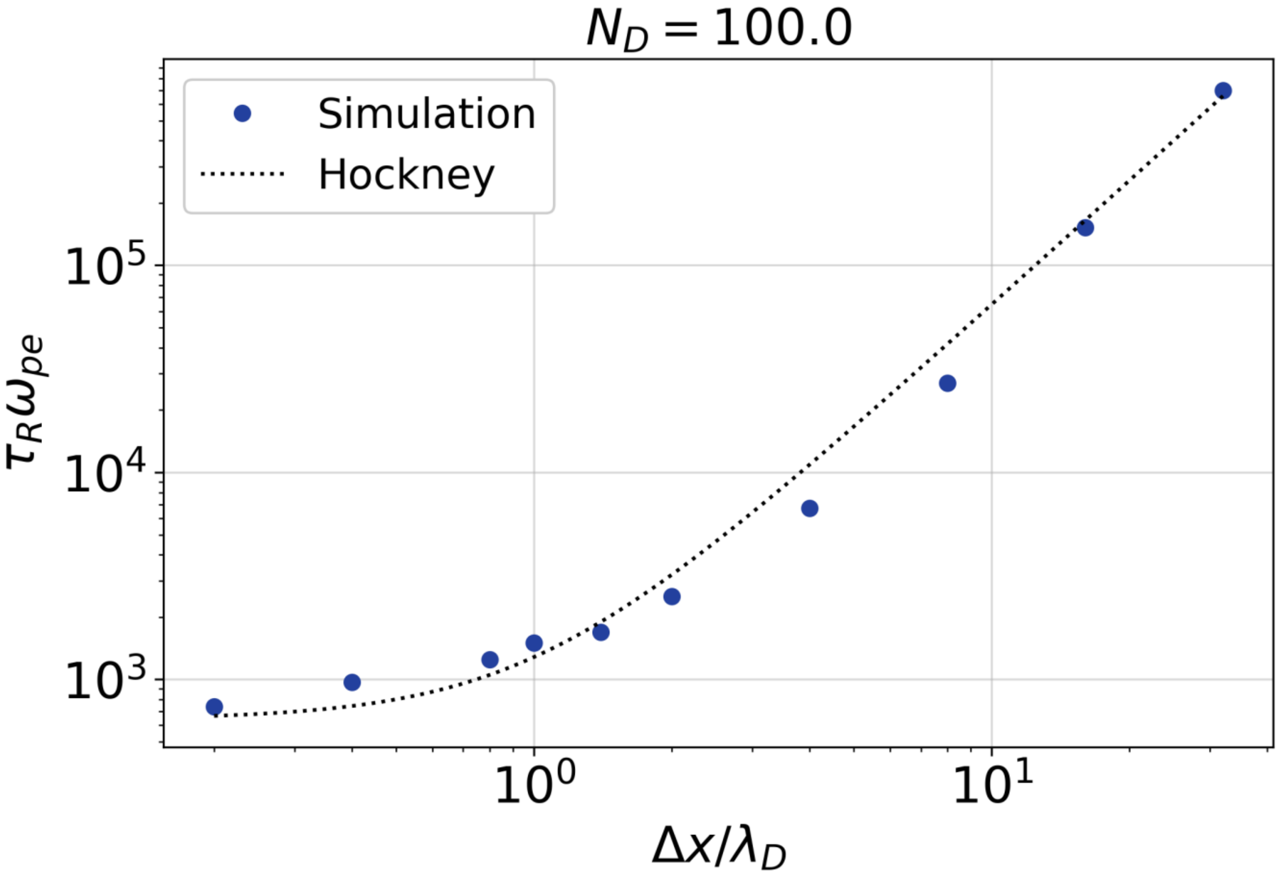
Hockney's empirical formula [1]:

$$\tau_R \omega_{pe} = \frac{2\pi N_D}{0.98}\left(1 + \left(\frac{\Delta \mathrm{x}}{\lambda_D}\right)^2\right)$$

[1] Hockney. *J. Comput. Phys.* **8**, (19-44) (1971); doi: 10.1016/0021-9991(71)90032-5.

Thermalization in 2D PIC - $\Delta x/\lambda_D$ scaling

Fig: Thermalization time as a function of grid spacing in 2D EC-PIC using CIC scheme



Increasing $\frac{\Delta x}{\lambda_D}$ broadens effective particle width and reduces rate of thermalization.

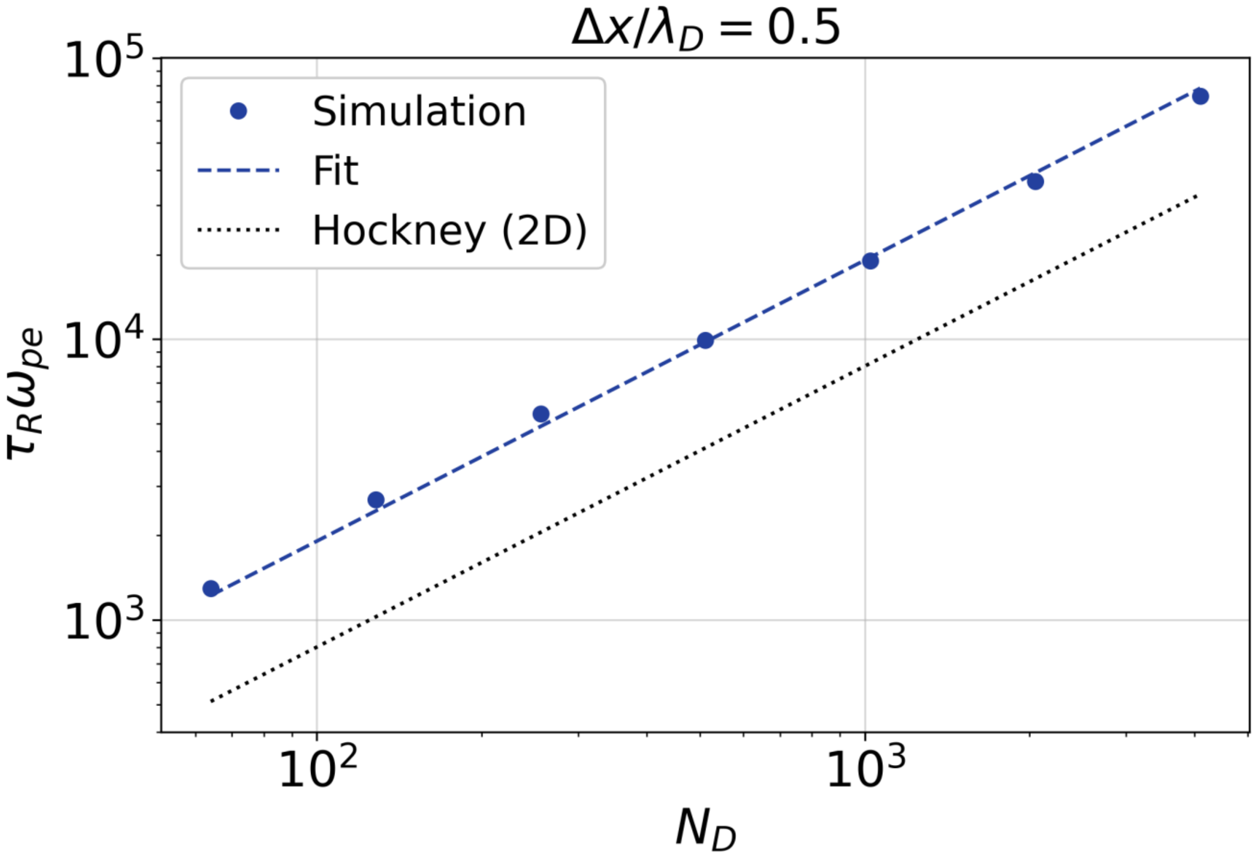
Hockney's empirical formula [1]:

$$\tau_R\omega_{pe} = \frac{2\pi N_D}{0.98}\left(1 + \left(\frac{\Delta \mathrm{x}}{\lambda_D}\right)^2\right)$$

[1] Hockney. *J. Comput. Phys.* **8**, (19-44) (1971); doi: 10.1016/0021-9991(71)90032-5.

Thermalization in 3D PIC – N_D scaling

Fig: Thermalization time as a function of N_D in 3D EC-PIC using CIC scheme



Real Coulomb collisions in 3D space:

$$\tau_R\omega_{pe} \propto \frac{N_D}{\ln(N_D)}$$

But this is incorrect for 3D PIC! Instead:

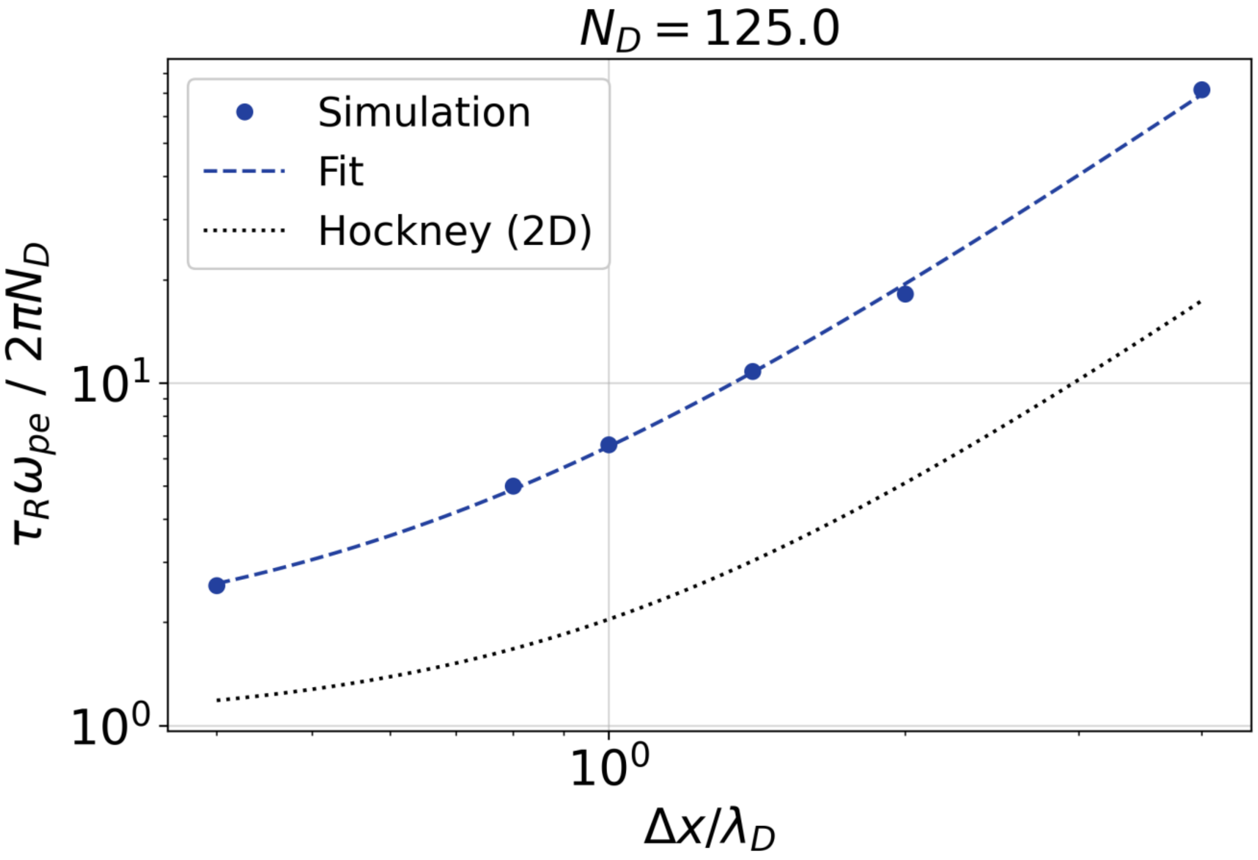
$$\tau_R\omega_{pe} \propto N_D$$

Our empirical fit to measured τ_R:

$$\tau_R\omega_{pe} = 2\pi N_D\left(1.58 + 0.92\left(\frac{\Delta x}{\lambda_D}\right) + 4.0\left(\frac{\Delta x}{\lambda_D}\right)^2\right)$$

Thermalization in 3D PIC – $\Delta x/\lambda_D$ scaling

Fig: Thermalization time as a function of grid spacing in 3D EC-PIC using CIC scheme



Same story - increasing $\Delta x/\lambda_D$:

- Broadens effective particle width
- Reduces rate of thermalization
- Increases τ_R

Our empirical fit to measured 3D PIC τ_R:

$$\tau_R \omega_{pe} = 2\pi N_D \left(1.58 + 0.92 \left(\frac{\Delta x}{\lambda_D} \right) + 4.0 \left(\frac{\Delta x}{\lambda_D} \right)^2 \right)$$

Slow thermalization in 1D PIC – N_D^2 scaling

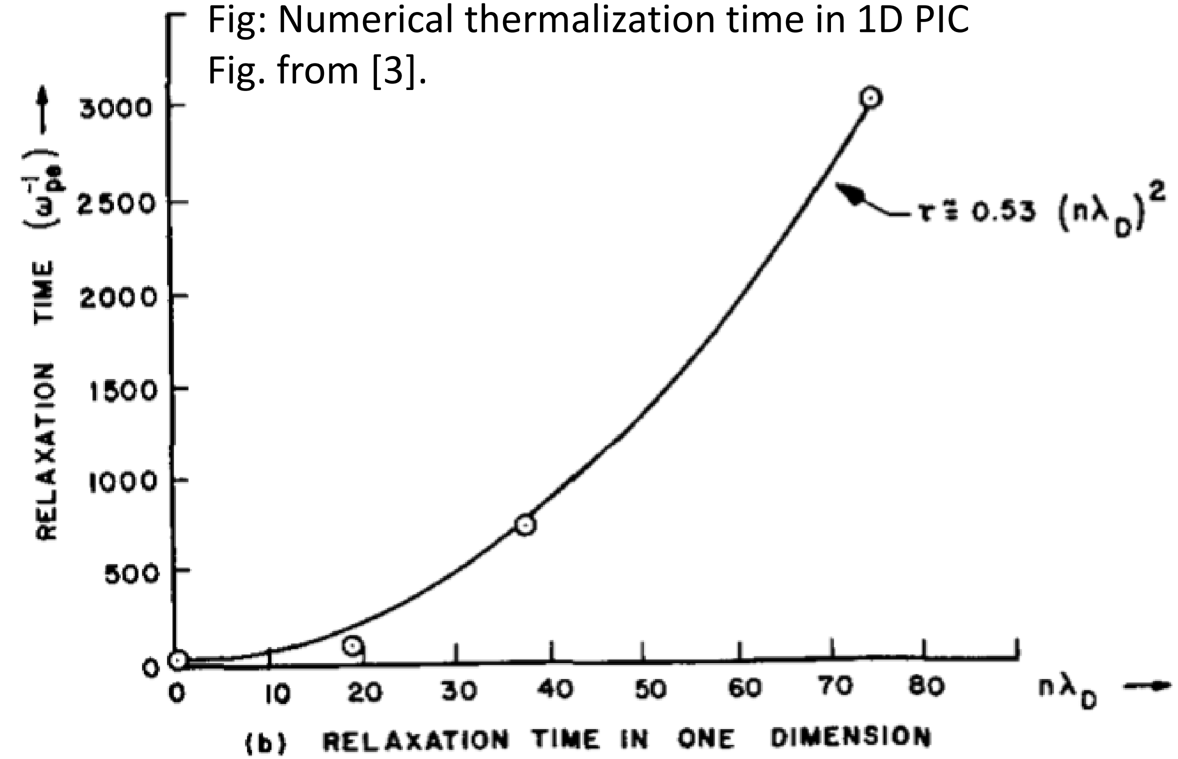
To first order in $1/N_D$ the collision operator cancels *any* stable distribution in a single-species plasma, not just a Maxwellian [1,2].*

Velocities can be exchanged, but not changed.

Relaxation rates depend on next order. [2,3,4]

$$\tau_R \omega_{pe} \propto N_D^2$$

3-particle correlations required for relaxation! [2]



(b) RELAXATION TIME IN ONE DIMENSION

Fig: Numerical thermalization time in 1D PIC
Fig. from [3].

*Exceptions occur when MCC added! [5]

[1] Eldridge and Feix. *Phys. Fluids* **6**, 398 (1963); doi: 10.1063/1.1706746.
[2] Dawson. *Phys. Fluids* **7**, 419 (1964); doi: 10.1063/1.1711214.
[3] Montgomery and Nielson. *Phys. Fluids* **13**, 1405 (1970); doi: 10.1063/1.1693081.
[4] Virtamo and Tuomisto. *Phys. Fluids* **22**, 172 (1979); doi: 10.1063/1.862453.
[5] Turner. *Phys. Plasmas* **13**, 033506 (2006). doi: 10.1063/1.2169752

Exception: 1D PIC/MCC

1D PIC numerical thermalization can speed up significantly when using a Monte Carlo collision algorithm. [1]

$$\tau_R \omega_{pe} \propto N_D^2 \quad \rightarrow \quad \tau_R \omega_{pe} \propto N_D$$

$$\tau_R \omega_{pe} = \frac{34.4}{N_D^{-2} + 28 N_D^{-1} (\nu / \omega_{pe})} \qquad [1]$$

Return of the first order $1/N_D$ numerical collision operator term and breaking the kinetic block: see [2,3]

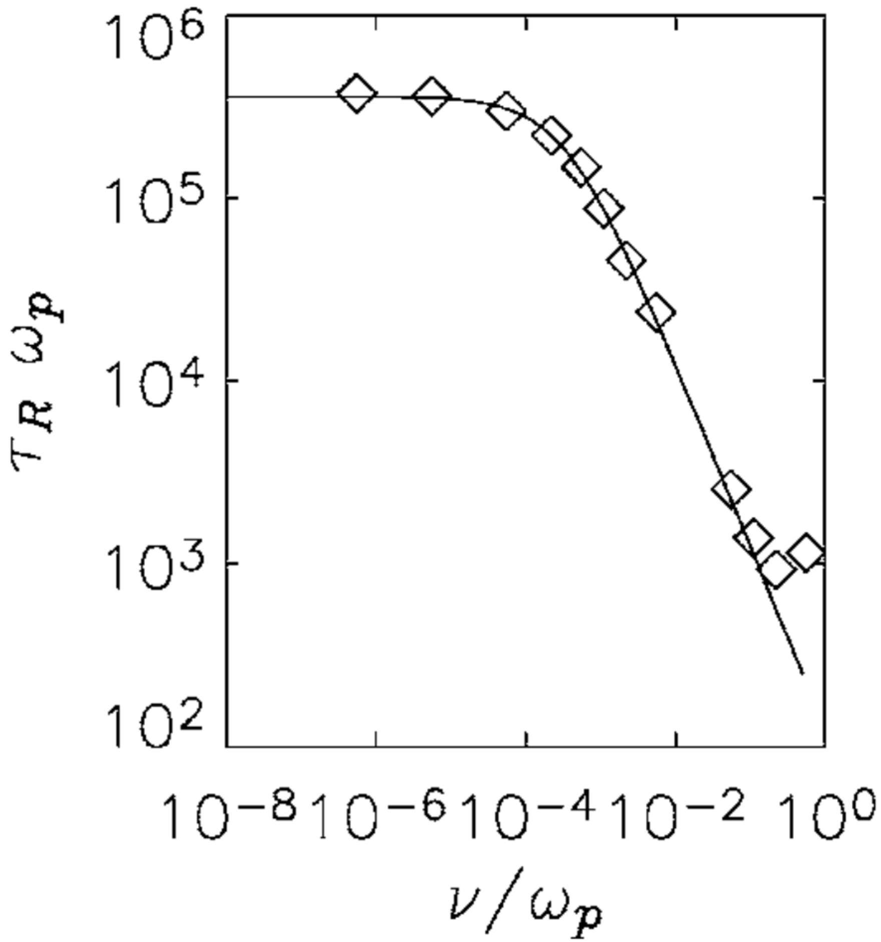


Fig: Dependence of thermalization time on the ratio of the collision frequency to the plasma frequency. Fig. from [1].

[1] Turner. *Phys. Plasmas* **13**, 033506 (2006). doi: 10.1063/1.2169752
[2] Lai et al. *Phys. Plasmas* **21**, 122111 (2014); doi: 10.1063/1.4904307
[3] Lai et al. *Phys. Plasmas* **22**, 092127 (2015); doi: 10.1063/1.4931741

Mitigation strategies

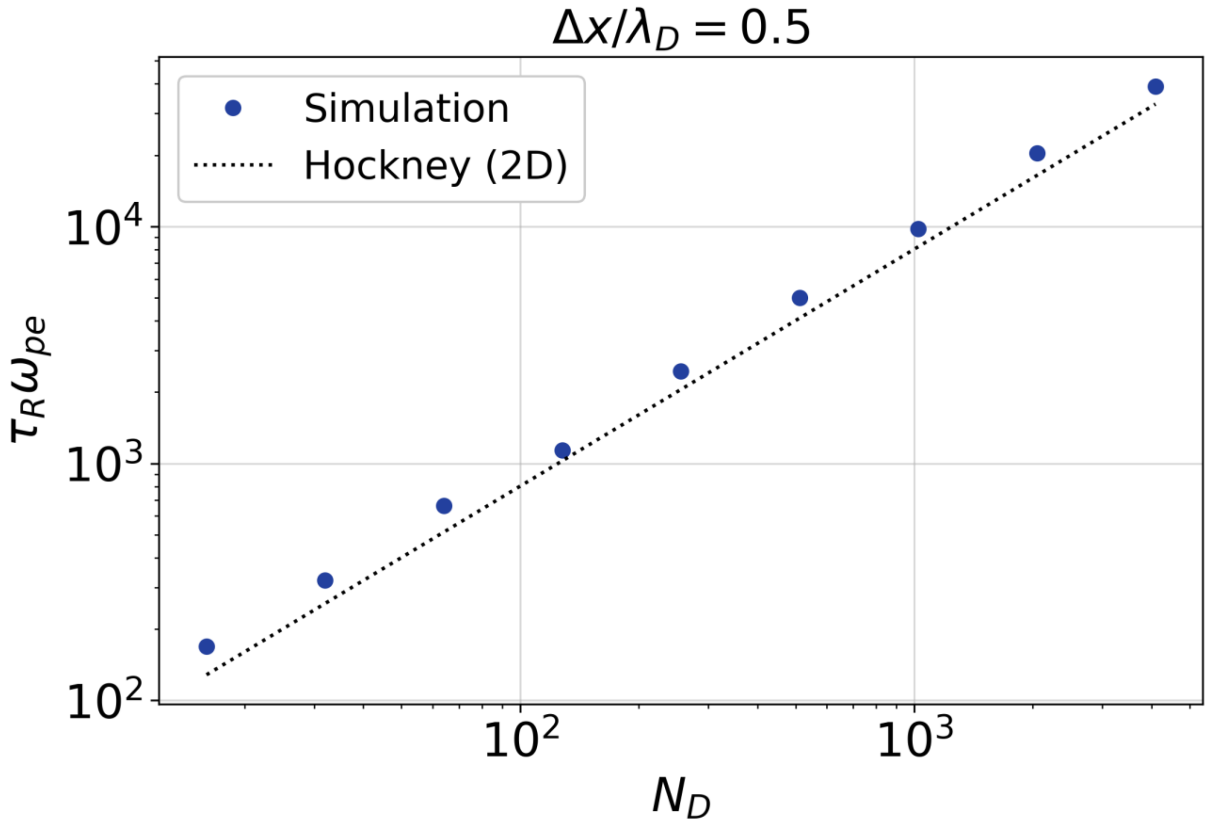
Mitigation strategy: Increase ppc

In any dimension, $N_D \propto ppc$

More markers in phase space →PIC behaves more like a fluid in phase space

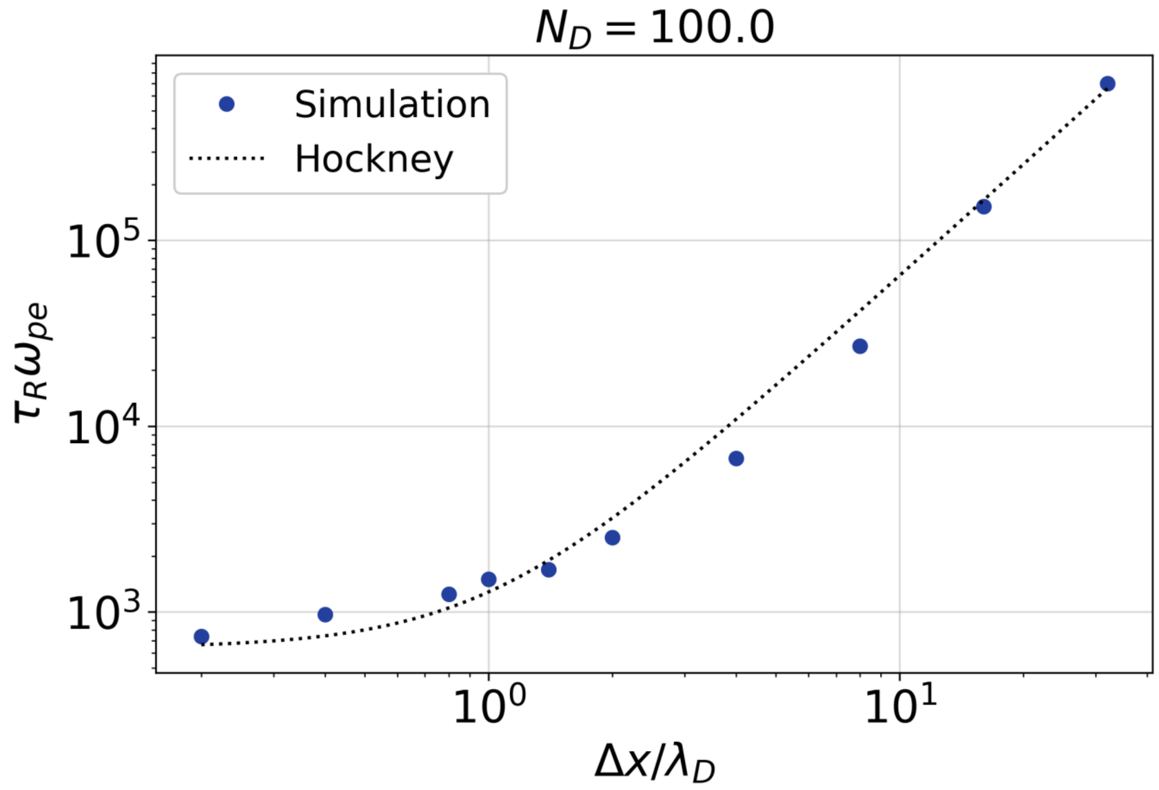
More macroparticles → fewer real particles represented by each marker

Fig: Thermalization time as a function of N_D in 2D EC-PIC using CIC scheme



Mitigation strategy: Increase Δx

Fig: Thermalization time as a function of grid spacing in 2D EC-PIC using CIC scheme



Best impact at $\frac{\Delta x}{\lambda_D} \gg 1$

Use energy conserving codes which allow under-resolution of λ_D.

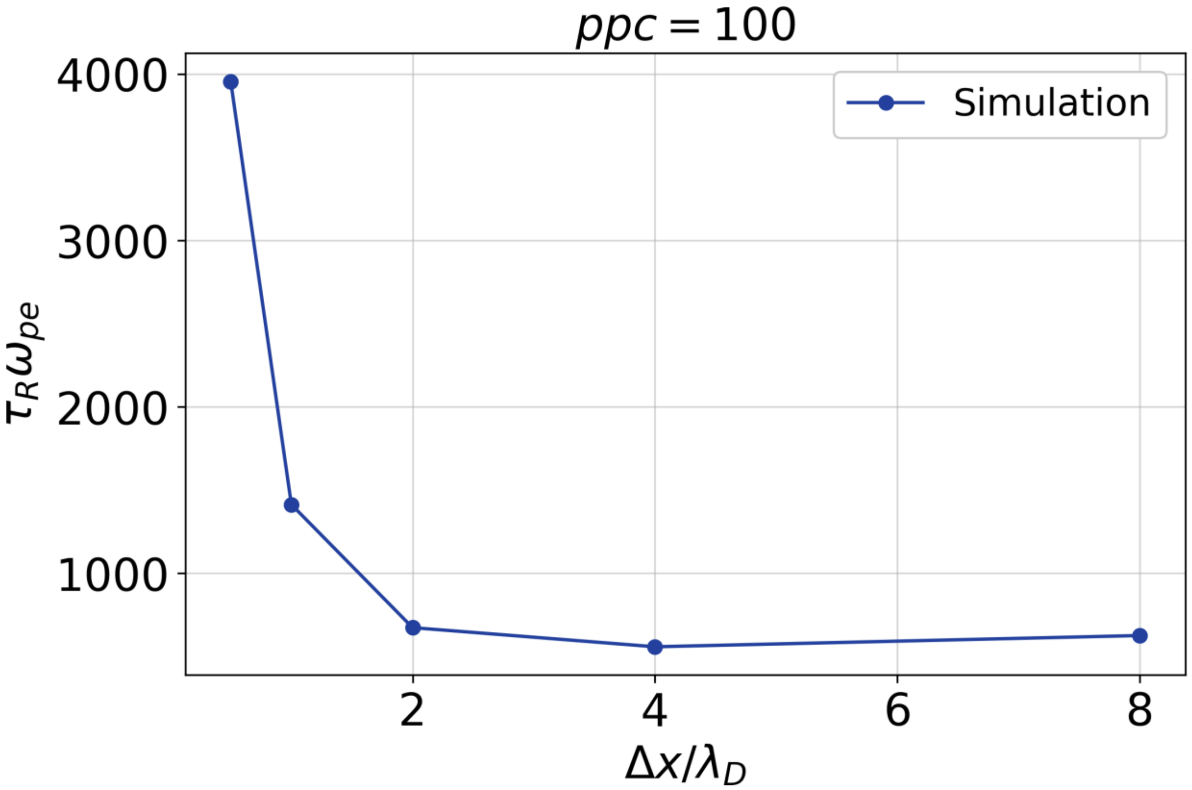
Consider also:

- Using higher order shape functions (not just CIC)
- Filtering

Don't forget to increase ppc when increasing Δx!

Remember that increasing $\varDelta x$ without increasing ppc reduces total macroparticles!

Fig: 2D electrostatic PIC thermalization time with fixed ppc, increasing grid spacing



Increasing Δx without increasing ppc:

- Lowers actual macroparticle density
- Reduces $N_D = ppc\left(\frac{\lambda_D}{\Delta x}\right)^{dim}$
- Ultimately enhances thermalization

Maintaining timescale ordering

If we can order τ_R^{num} with other timescales (collision times, transport times, etc) in the same manner as a real Coulomb collision time τ_R^{real} this might be sufficient.

For this simulation:

$$\tau_{Lx} > \tau_R^{real} > \tau_R^{num} > \ \tau_{e-Ar}$$

Thermalization will happen faster but it is nonetheless accurate.

Fig: 2D PIC simulation of an electron-beam generated plasma in a magnetic field. A) Potential colormap, B) 1D EVDF cross sections from rectangle in Fig A.

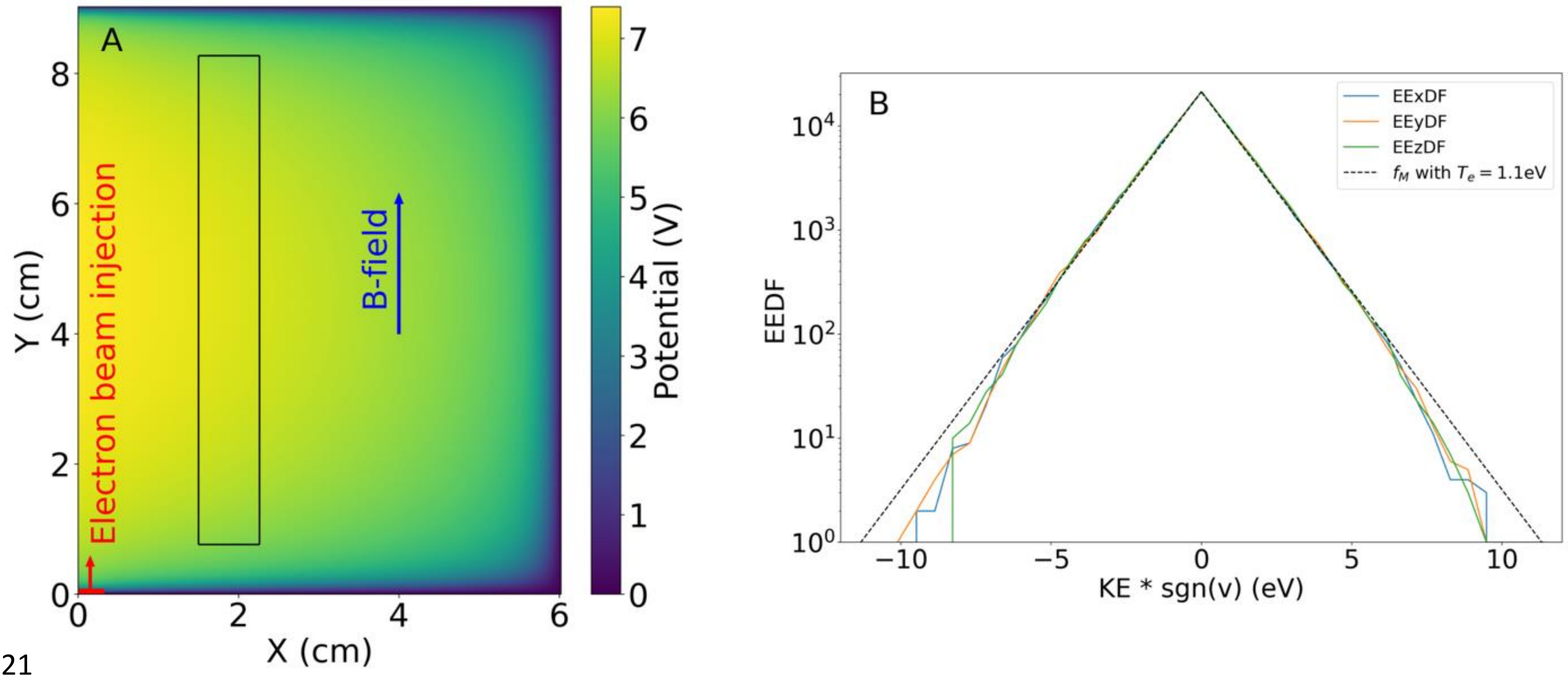


Fig. modified from [1].
Thanks to Shahid Rauf for the simulation data.

[1] Jubin et al. *Phys. Plasmas* **31**, 023902 (2024). doi: 10.1063/5.0180421

Mitigation strategy: Artificially scale λ_D, ω_{pe}

Artificially increase permittivity of free space ε_0, thereby reducing effective ω_{pe} and increasing λ_D. This will delay thermalization: $\tau_R \propto n\lambda_D^{dim}/\omega_{pe}$.

For this simulation:

$$\tau_{Ly} > \tau_R^{num} > \tau_{e-Ar} \sim \tau_R^{real}$$

By scaling $\boldsymbol{\varepsilon_0}$ → $\boldsymbol{1059\varepsilon_0}$ we made $\boldsymbol{\tau_R^{num} > \tau_R^{real}}$

2D PIC simulation of a hollow cathode plasma. A) Potential colormap, B) 1D EVDF cross sections from cathode region in Fig A.

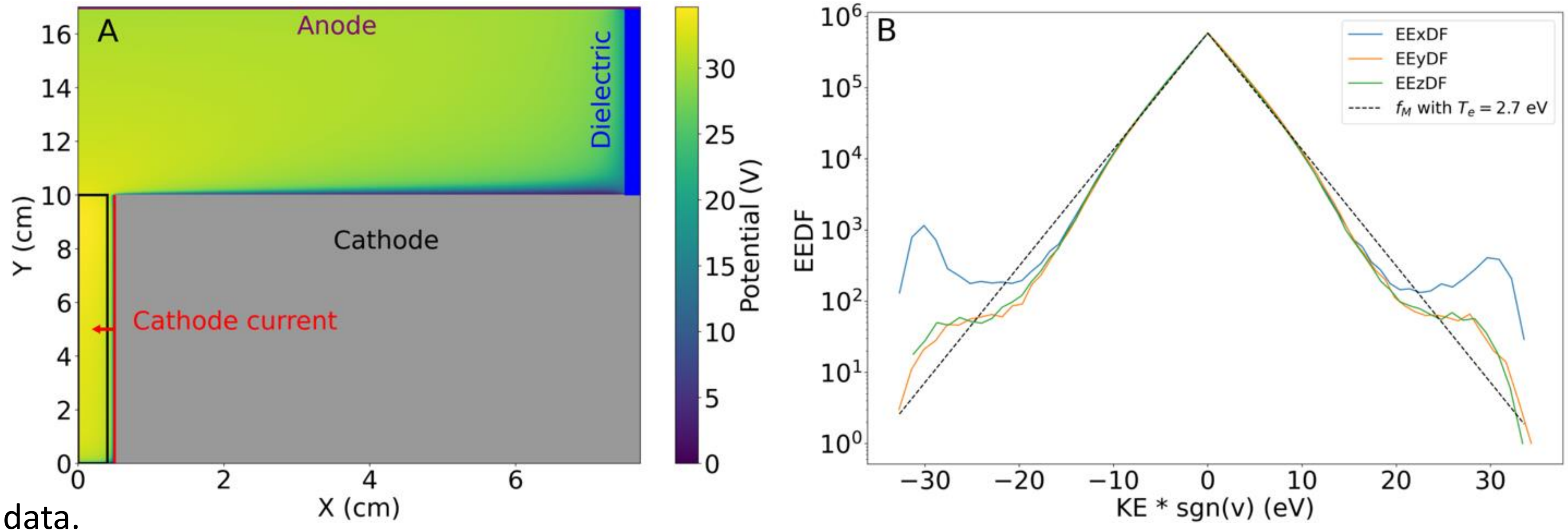


Fig. modified from [1].
Thanks to Willca Villafana for the simulation data.

[1] Jubin et al. *Phys. Plasmas* **31**, 023902 (2024). doi: 10.1063/5.0180421

Summary

- Numerical thermalization is faster in multidimensional PIC than 1D PIC
- It is faster than real thermalization due to real Coulomb collisions when CIC schemes are used and Debye length is well-resolved.
- **Accurate kinetic behavior requires:**

  Numerical thermalization timescales $>$ real timescale of EVDF evolution
- Increasing $\Delta x/\lambda_D$ while maintaining N_D reduces the numerical thermalization rate
- More information is needed regarding impact of irregular grid spacing, higher order shape functions.

Acknowledgements

Thank you to all the members of my research group and our collaborators!
Igor Kaganovich, Andrew Tasman Powis, Alexander Khrabrov, Willca Villafana, Mikhail Mokrov, Salman Sarwar, Yuri Barsukov, Stéphane Ethier, Dmytro Sydorenko, and Shahid Rauf

Special thanks to Andrew Tasman Powis for writing the PIC code optimized for use in thermalization measurement that was used to produce the results shown here.

This research was funded by the Department of Energy's Laboratory Directed Research and Development (LDRD) program at Princeton Plasma Physics Laboratory (PPPL) and the PPPL CRADA agreement with Applied Materials entitled "Two-Dimensional Modeling of Plasma Processing Reactors."

Questions?