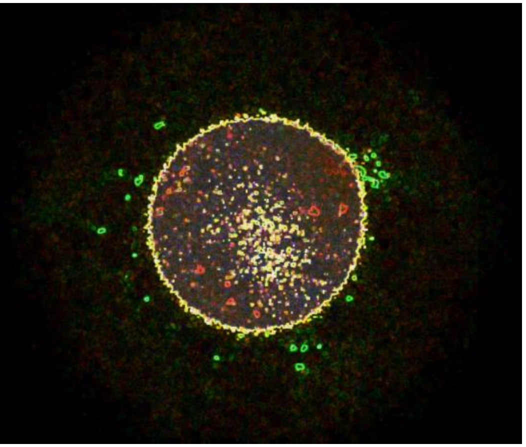
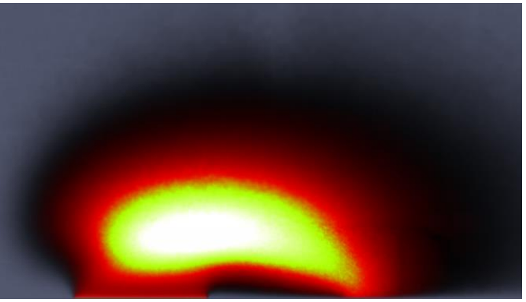




Sandia National Laboratories

Exceptional service in the national interest

Computational Efficiency for Kinetic Simulation of Vacuum Arcs

Matthew M. Hopkins, Jeremiah J. Boerner, Christopher H. Moore, Paul S. Crozier, Stan G. Moore, Matthew T. Bettencourt, and Russell Hooper

Sandia National Labs

4th International Workshop on Mechanisms of Vacuum Arcs

November 4-7, 2013, Chamonix, France

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2013-9311C

Introduction

Vacuum arc discharge is a dominant failure mechanism in many vacuum electronic devices. The same basic failure mechanism is also described as high voltage breakdown (HVB), or electrostatic discharge (ESD). There are also numerous devices that operate based on intended discharge of an arc, e.g., plasma switches, spark plugs, and ion sources. In an effort to better understand the initiation process and post-breakdown evolution to a steady arc, we have developed a 3D massively parallel electrostatic low temperature plasma simulation tool, *Aleph*. *Aleph* includes a number of algorithm and model advances to understand the mechanisms and key phases of vacuum arc discharge. Our long-term goal is to provide predictive capability for breakdown in complex 3D vacuum devices in a production environment.

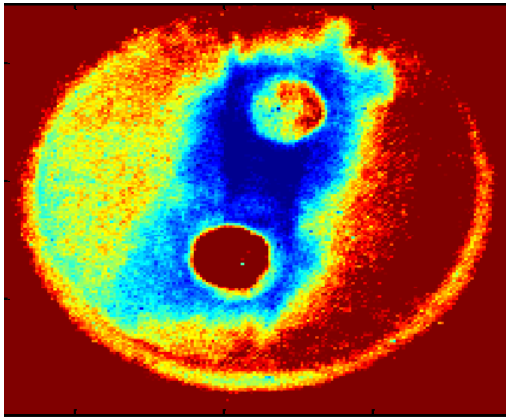
The spatial, temporal, and model capability demands for simulating vacuum arc discharges are enormous. The simulation must evolve from an initial collisionless vacuum (or near vacuum) state through a sputtering phase with surface interaction and low collisionality and ionization, into a growing quasi-neutral plasma with increasing collisionality and ionization, to an explosive growth electron avalanche process, and finally to a steady current-carrying arc plasma. The modeling demands change drastically as each of these phases is encountered. We describe a number of model advances to address these challenges.

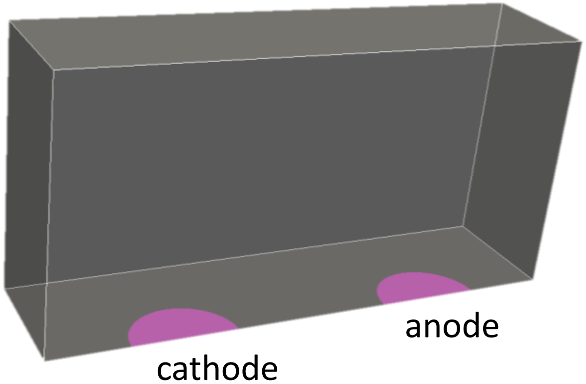
Outline

- Typical application
- Description of PIC-DSMC code, Aleph
- Simulation requirements & cost
- Successive refinement in Δx and Δt
- Particle merging
- Explicit adaptive particle move
- Dynamic sizing of DSMC cells
- Quasi-static acceleration

Typical Application

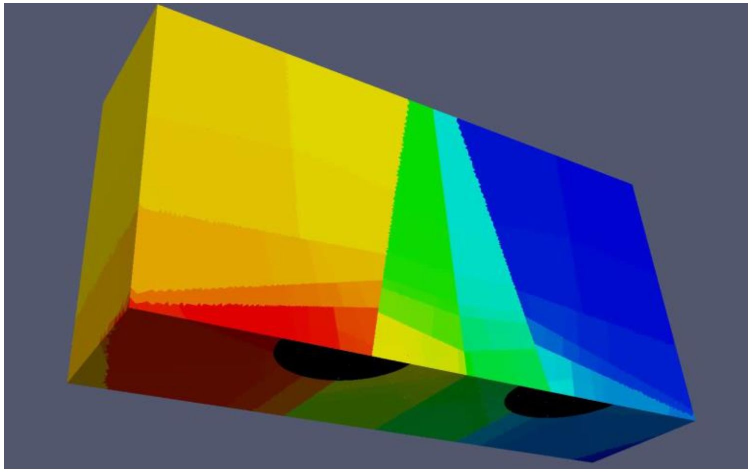
- In vacuum or 4 Torr Ar background
- 1.5 mm inner-to-inner distance
- 0.75 mm diameter electrodes
- Copper electrodes (this picture is Cu-Ti)
- 2 kV drop across electrodes
- 20Ω resistor in series
- Steady conditions around 50V, 100A
- Breakdown time << 100ns
- To meet an ionization mean free path of 1.5 mm at maximum σ, $n_i \sim 10^{16} - 10^{17}$ #/cm^3





3D computational domain

Description of *Aleph*



- 1, 2, or 3D Cartesian
- Unstructured FEM (compatible with CAD)
- Massively parallel
- Hybrid PIC + DSMC (PIC-MCC)
- Electrostatics
- Fixed B field
- Solid conduction
- e- approximations (quasi-neutral ambipolar, Boltzmann)
- Dual mesh (Particle and Electrostatics/Output)
- Advanced surface (electrode) physics models
- Collisions, charge exchange, chemistry, excited states, ionization
- Advanced particle weighting methods
- Dynamic load balancing (tricky)
- Restart (with all particles)
- Agile software infrastructure for extending BCs, post-processed quantities, etc.
- Currently utilizing up to 64K processors (>1B elements, >1B particles)

Description of *Aleph*

Basic algorithm for one time step of length Δt:

1. Given known electrostatic field $\mathbf{E}^n$, move each particle for $\frac{\Delta t}{2}$ via:

$$v_i^{n+1/2} = v_i^n + \frac{\Delta t}{2}\left(\frac{q_i}{m_i}\mathbf{E}^n\right)$$

$$x_i^{n+1} = x_i^n + \Delta t v_i^{n+1/2}$$

2. Compute intersections (non-trivial in parallel).
3. Transfer charges from particle mesh to static mesh.
4. Solve for $\mathbf{E}^{n+1}$.

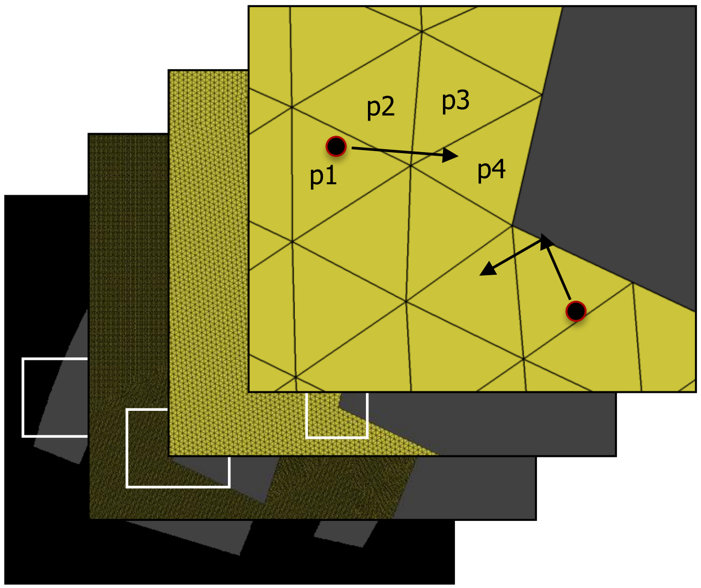
$$\nabla \cdot \left(\epsilon \nabla V^{n+1}\right) = -\rho(\mathbf{x}^{n+1})$$

$$\mathbf{E}^{n+1} = -\nabla V^{n+1}$$

5. Transfer fields from static mesh to dynamic mesh.
6. Update each particle for another $\frac{\Delta t}{2}$ via:

$$v_i^{n+1} = v_i^{n+1/2} + \frac{\Delta t}{2}\left(\frac{q_i}{m_i}\mathbf{E}^{n+1}\right)$$

7. Perform DSMC collisions: sample pairs in element, determine cross section and probability of collision. Roll a digital die, and if they collide, re-distribute energy.
8. Perform chemistry: for each reaction, determine expected number of reactions. Sample particles of those types, perform reaction (particle creation/deletion).
9. Reweight particles.
10. Compute post-processing and other quantities and write output.
11. Rebalance particle mesh if appropriate (variety of determination methods).



Simulation Requirements

Temporal scales dominated by plasma electron frequency ω_p, CFL, and collision frequency ν_c at different phases of breakdown:

$$\Delta t < \min\left(\frac{2}{\omega_p}, \frac{\Delta x}{\sqrt{\frac{m_e \Delta V}{2 q_e}}}, \frac{1}{n_n \sigma \bar{v}}\right)$$

Spatial scales dominated by Debye length λ_D and collision mean free path λ_{mfp} at different phases of breakdown:

$$\Delta x < \min\left(\lambda_D, \frac{1}{n_n \sigma}\right)$$

Number densities increase from "0" to 10^{17} #/cm^3. Using same fixed particle weight p_{weight} isn't an option.

Typical Vacuum Arc Progression

A: Initial injection of e- (no plasma yet)

B: Growth of cathode plasma

C: Breakdown

D: Relax to steady operation (ΔV drops to ~50V)

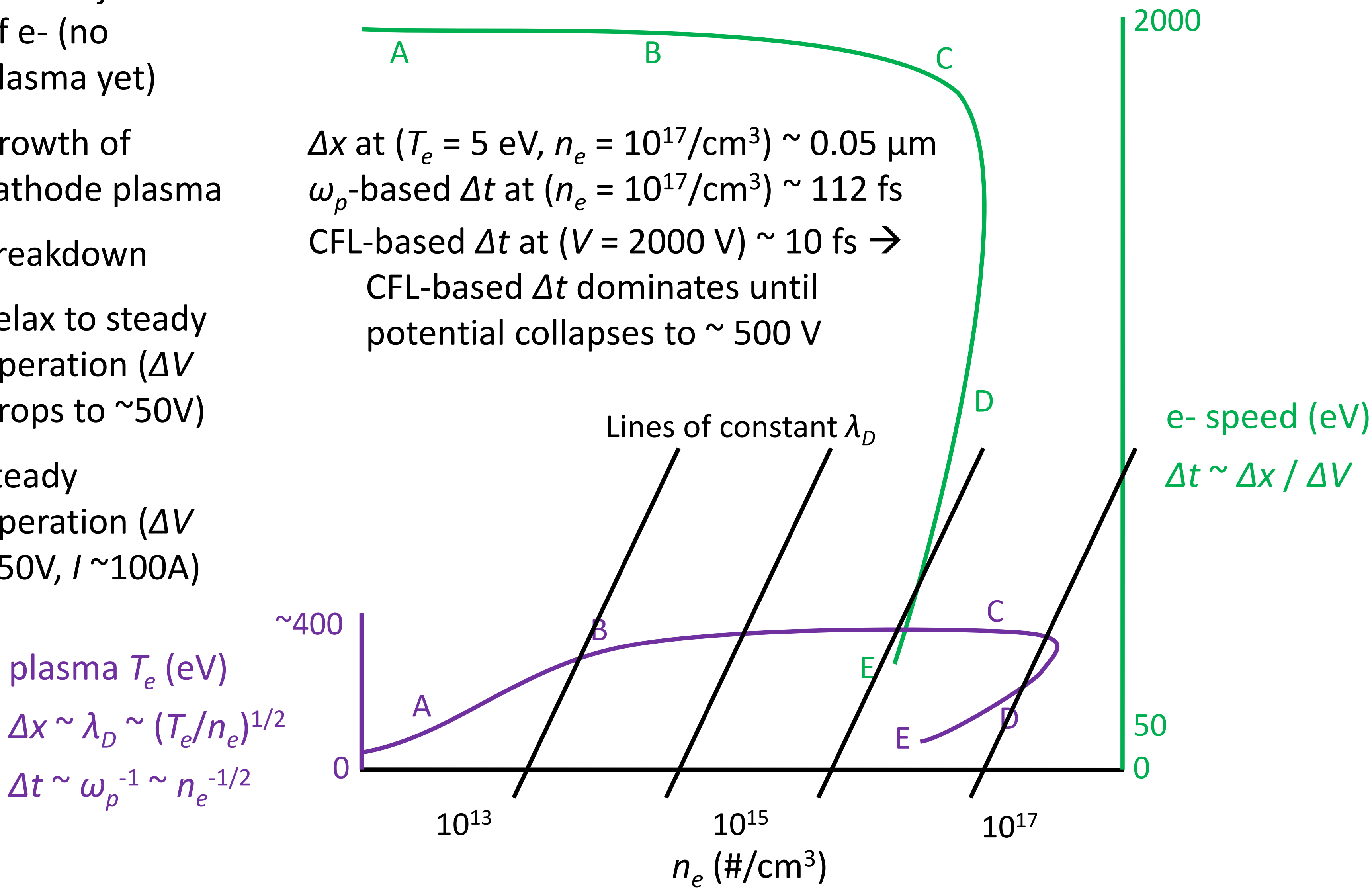
E: Steady operation (ΔV ~50V, I ~100A)

Δx at (T_e = 5 eV, n_e = 10^{17}/cm^3) ~ 0.05 µm

ω_p-based Δt at (n_e = 10^{17}/cm^3) ~ 112 fs

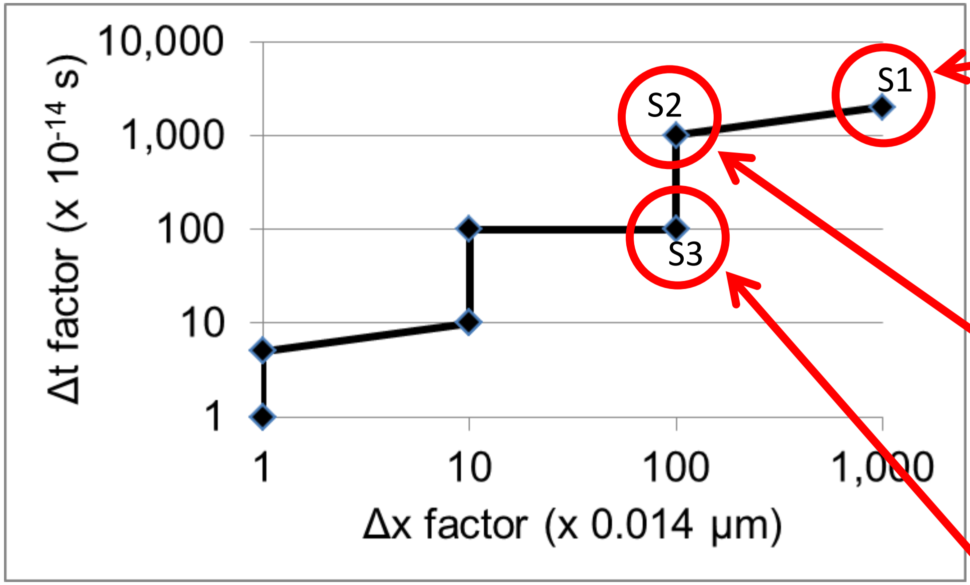
CFL-based Δt at (V = 2000 V) ~ 10 fs →

CFL-based Δt dominates until potential collapses to ~ 500 V



Managing Δx, Δt: Successive Refinement

Discretely refine in (Δx, Δt) by stopping simulation near stability/fidelity limits and perform full particle restart on Δx- and/or Δt-refined simulation. A typical progression to (Δx, Δt) = (0.014 µm, 10 fs) looks like:



S1: (Δx, Δt) = (0.014 mm, 20 ps), or 2,000,000 x less work than final solution steps.
... after 160 ns, both λ_D and ω_p are being challenged, so move to ...

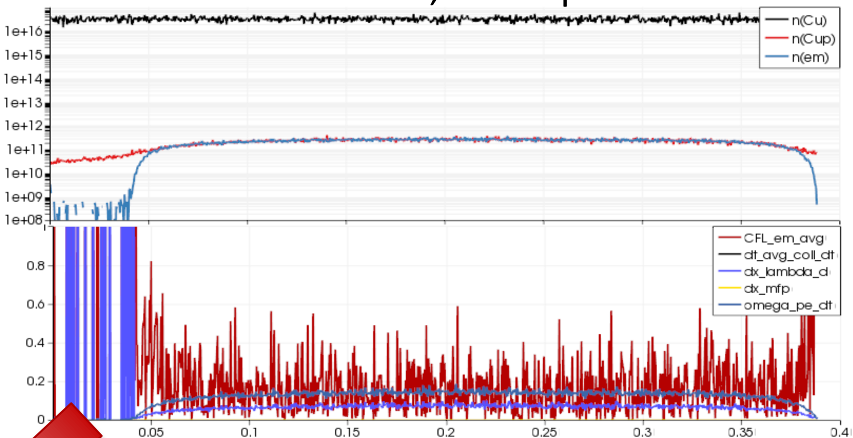
S2: (Δx, Δt) = (0.0014 mm, 10 ps).
... after another 190 ns, only ω_p is being challenged, so move to ...

S3: (Δx, Δt) = (0.0014 mm, 1 ps).

... and continue ... (just started dynamic global Δt selection, want to do something about Δx, too ...). Total savings to 1.35 µs (this case) is tremendous, but still need many small steps on small mesh at end...

Managing Δx, Δt: Successive Refinement

t = 166 ns, Δt = 5 ps

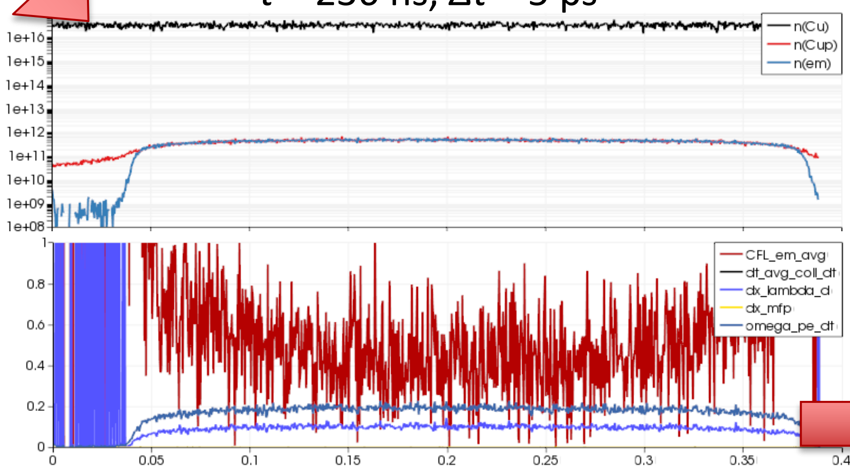


Simulation diagnostics: average e- CFL, $\Delta t \cdot \nu_C$, $\Delta t \cdot \omega_p$, $\Delta x/\lambda_D$, $\Delta x/\lambda_{mfp}$

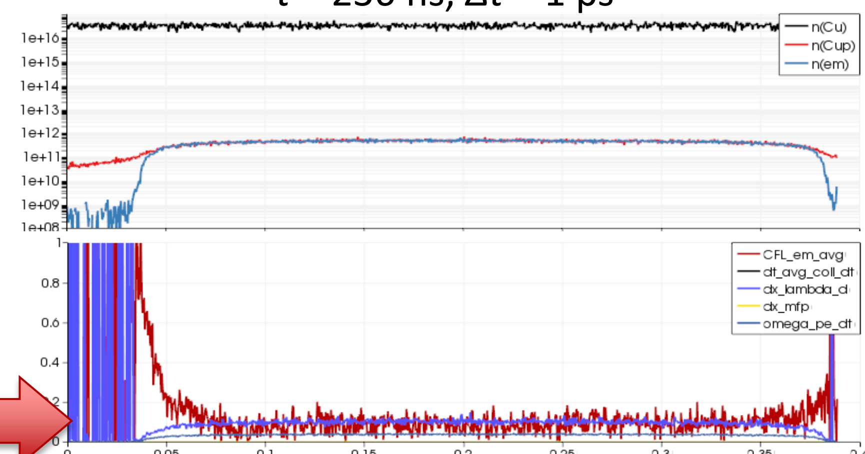
Cathode on left, anode on right,
120 V drop across 3.88 mm,
1 Torr background Cu,
Trickle influx of cold e- (10^{10} #/cm^2/µs),
300 K Cu "sputters" at:
- 1% vs. e-,
- 100% vs. Cu and Cu+,

1 eV SEE from Cu+ impact,
Δx = 1.38 µm, 2812 cells.

t = 236 ns, Δt = 5 ps

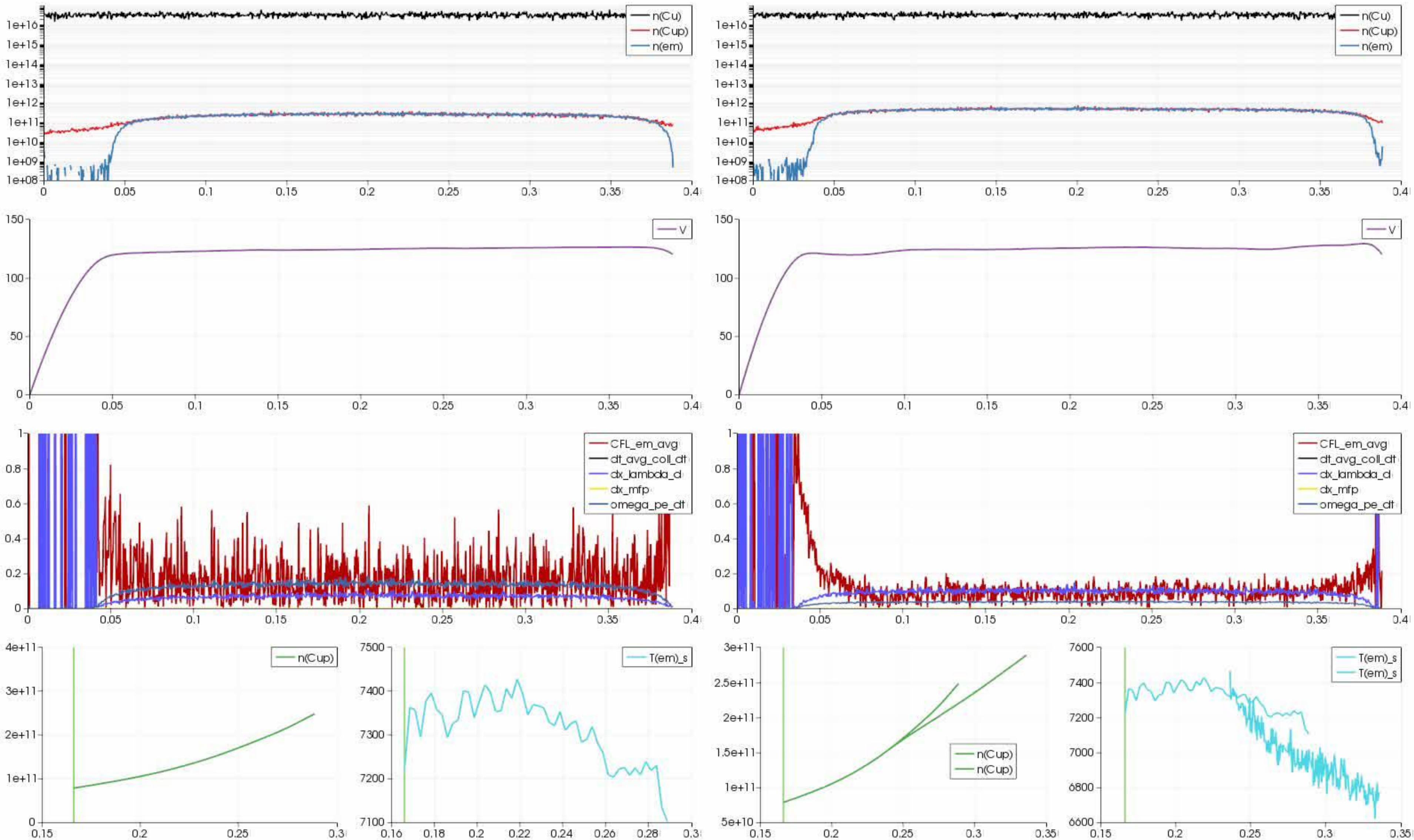


t = 236 ns, Δt = 1 ps



Growing average e- CFL prompts restarting with smaller Δt.

Managing *Δx*, *Δt*: Successive Refinement

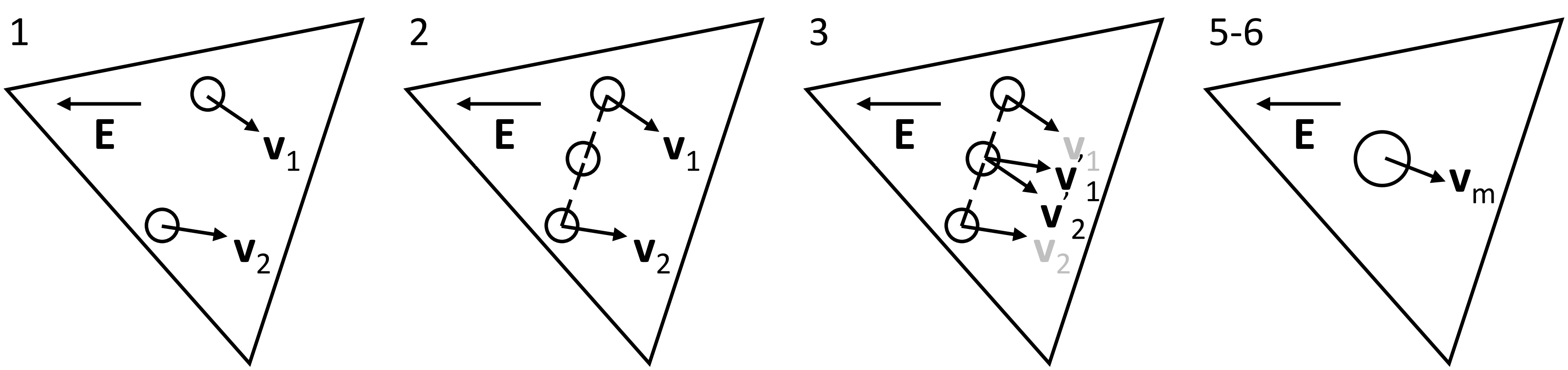


Managing p_{weight}: Particle Merging

We assume the discrete particle sample is the best representation of the "true" particle distribution. This drives us to use particle-only merge methods.

1. Choose a random pair of species S particles in the cell.
2. Compute center of mass position.
3. Compute modified velocities at the center of mass by accounting for displacement in the potential field.
4. If velocities are "too different," reject pair and repeat 1-3.
5. Calculate average velocity, conserving momentum.
6. Adjust (to target) weight and record difference in kinetic energy.

Repeat 1-6 until target number or limiter is met.



Managing p_{weight}: Particle Merging

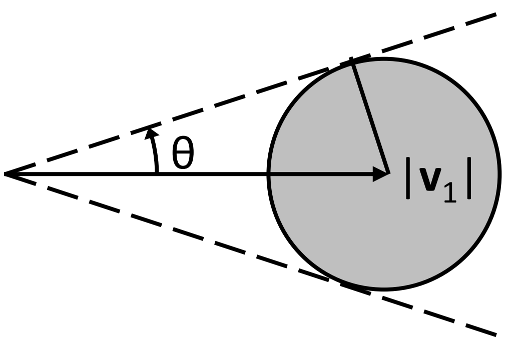
Only approve merge pairs that are close in both position and velocity.

- The spatial bin is the element, approves any pair.
- The velocity bin has many options. We use velocity interval, since it is easy to compute and adjusts based on local temperature.

Much faster to sort particles in element by speed, then choose one at random and check neighbors for valid merge partner.

Velocity Sphere

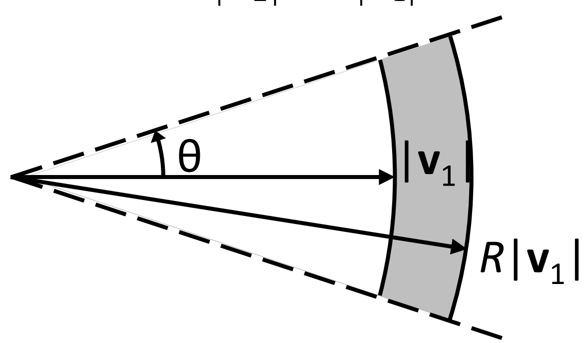
$$\left|\mathbf{v}_2 - \mathbf{v}_1\right| < \left|\mathbf{v}_1\right| \sin(\theta)$$



Velocity Proportion

$$\mathbf{v}_1 \cdot \mathbf{v}_2 > \left|\mathbf{v}_1\right|\left|\mathbf{v}_2\right| \cos(\theta)$$

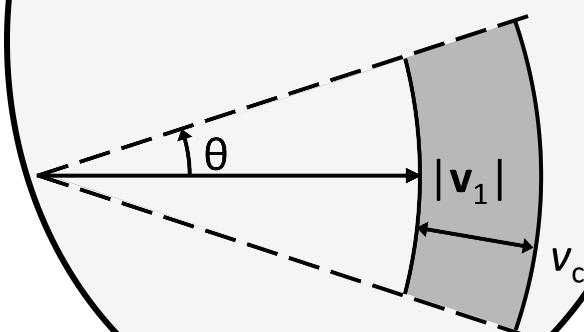
$$\left|\mathbf{v}_2\right| < R\left|\mathbf{v}_1\right|$$



Velocity Interval

$$\mathbf{v}_1 \cdot \mathbf{v}_2 > \left|\mathbf{v}_1\right|\left|\mathbf{v}_2\right| \cos(\theta)$$

$$\left|\mathbf{v}_2\right| - \left|\mathbf{v}_1\right| < v_c = \alpha\sqrt{k_B T / m}$$



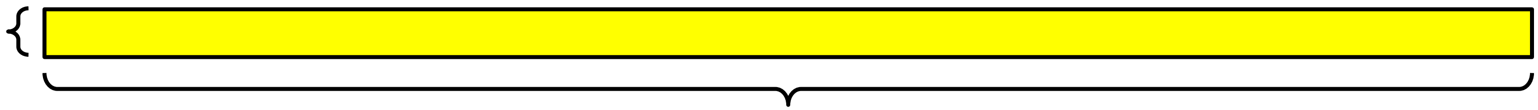
Managing p_{weight}: Particle Merging

Example of using dynamic particle weighting is a growing Xenon sheath.

<u>Injection</u>
V = 5 V
$n_{Xe+} = n_e$ = 10^{10} #/cm^3 to 10^{12} #/cm^3 over 20 ion transit times
v_D = 3 cm/μs
T_e = 1 eV
T_{Xe+} = 300 K

<u>Side walls</u>
$dV/dn = 0$
specular

<u>Wall</u>
V = 0 V



Δx

$(10 \text{ to } 100)\lambda_D = 300\Delta x$

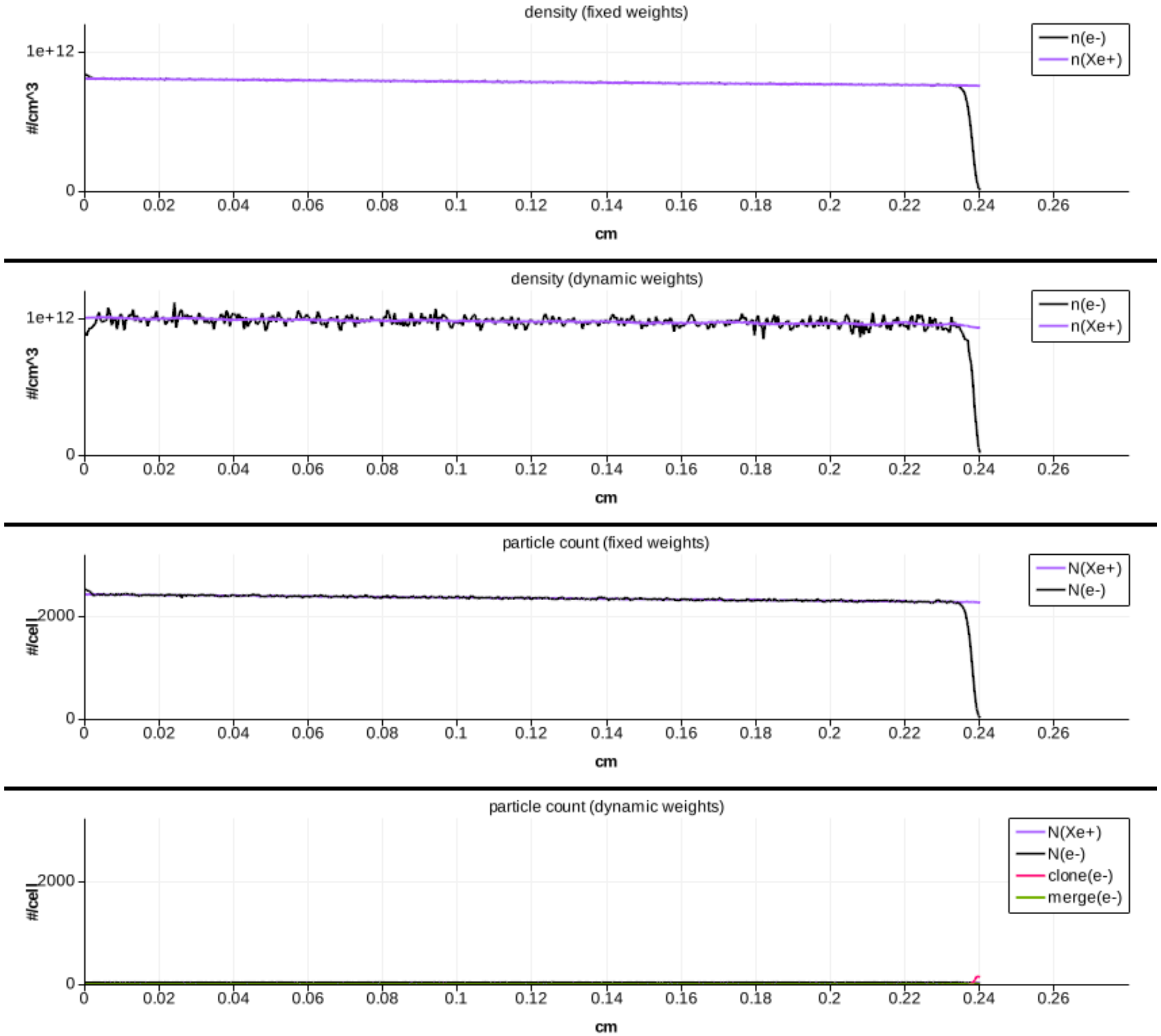
<u>Bulk plasma parameters</u>
v_{Bohm} = 0.086 cm/μs
λ_D = 7.4 x 10^{-3} cm to 7.4 x 10^{-4} cm
Δx = 2.5 x 10^{-4} cm
Δt = 20 ps
$\lambda_D / \Delta x$ = 30 to 3
$\omega_p \cdot \Delta t$ = 0.11 to 1.1

Two solutions:
- Fixed particle weight
- Dynamic particle weight (merging)

Small weight vs. large weight vs. requirements...

Managing p_{weight}: Particle Merging

Solution at high end, $n_{Xe+} = 10^{12}$ #/cm^3.



Runtime 147371 secs

particles = 11M

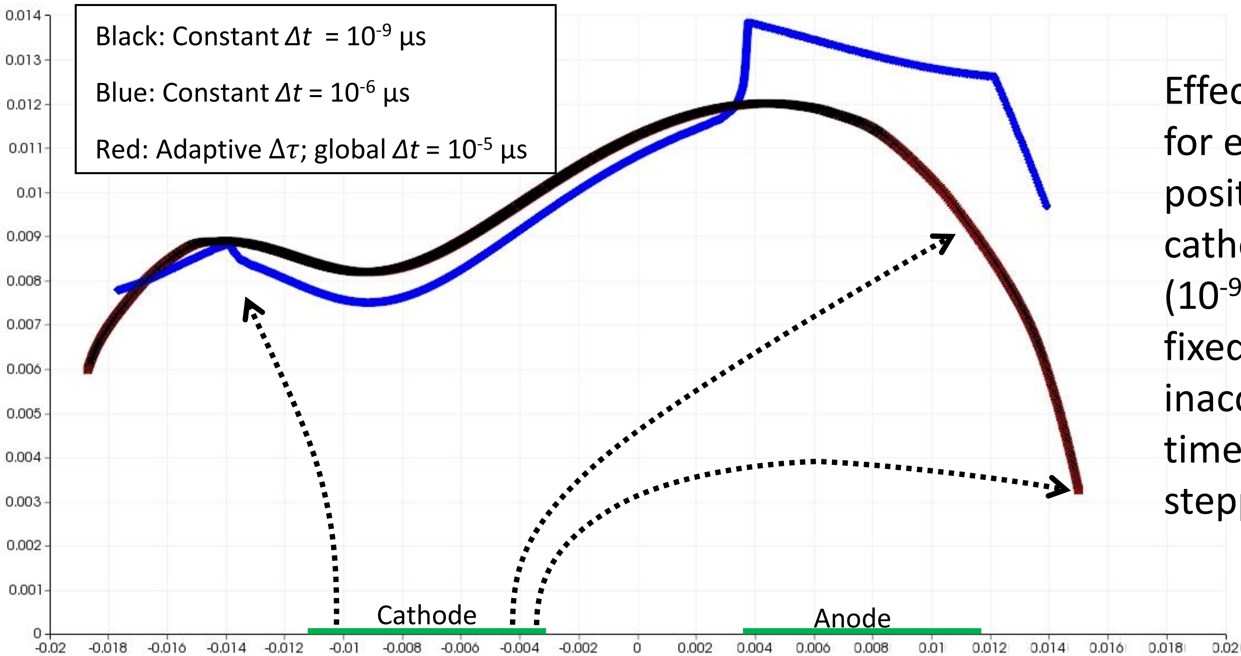
Runtime 7435 secs

particles = 150K

20x speed up!

Managing Δt: Explicit Adaptive Time-Stepping

In many initiation processes there is no significant space charge – only the initial applied field is relevant. In these cases we only need to accurately integrate particle trajectories. To mitigate the cost of using the most restrictive CFL-based Δt, we use a large "global" timestep Δt and force individual particles to use smaller adaptive timesteps $\{\Delta\tau_{i,j}\}$ within the global step ($\sum_j \Delta\tau_{i,j} = \Delta t$). $\Delta\tau_{i,j}$ is a function of particle velocity v_i, and the field **E** and field gradient ∇**E** along the particle trajectory.

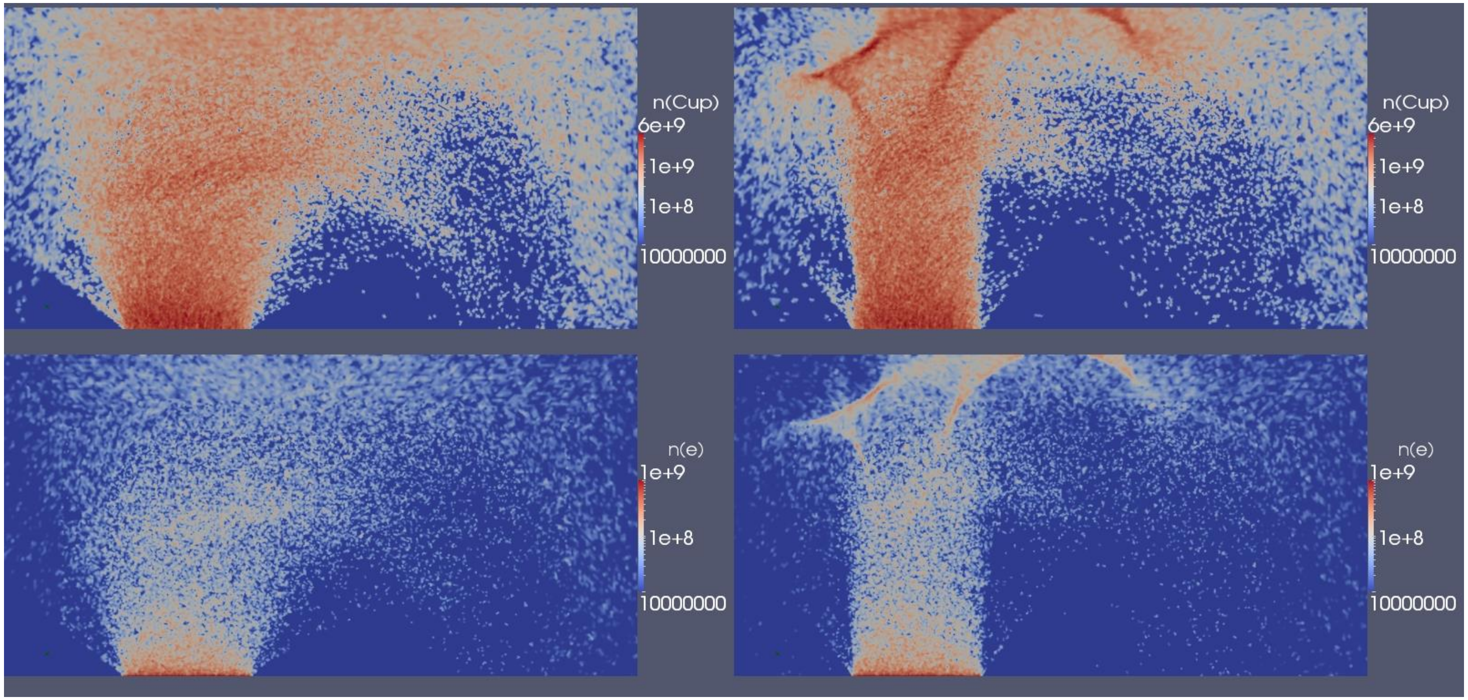


Effect of using adaptive time-stepping for e- trajectories. Thick lines are final positions of e- injected along the cathode after 10^{-5} µs. Small fixed time (10^{-9} µs) gives correct answer. Larger fixed time (10^{-6} µs) is significantly inaccurate. Using an even larger global timestep (10^{-5} µs) but adaptive time-stepping again gives correct answer.

Managing Δt: Explicit Adaptive Time-Stepping

Sandia National Laboratories

2D domain with ~3 Torr background neutral gas – consistent with experiments. Small flux of e- from cathode, should ionize background gas. Ions can generate electrons at cathode. Run 3 cases out to 1.5 x 10^{-3} μs. Constant $\Delta t = 10^{-8}$ μs and adaptive $\Delta t = 10^{-5}$ μs results overlap.



n_{Cu+}

n_{e-}

Constant $\Delta t = 10^{-8}$ μs runtime 24.6 hours
Adaptive $\Delta t = 10^{-5}$ μs runtime 1.6 hours
(solutions essentially identical)

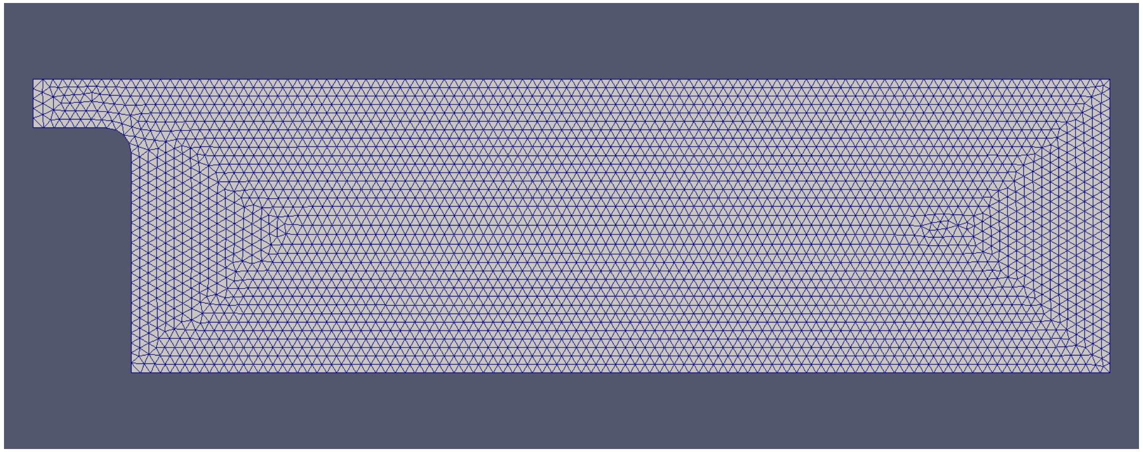
Constant $\Delta t = 10^{-5}$ μs -- 0.024 hours

15x speed up!

Managing Δx: Dynamic Sizing of DSMC Cells

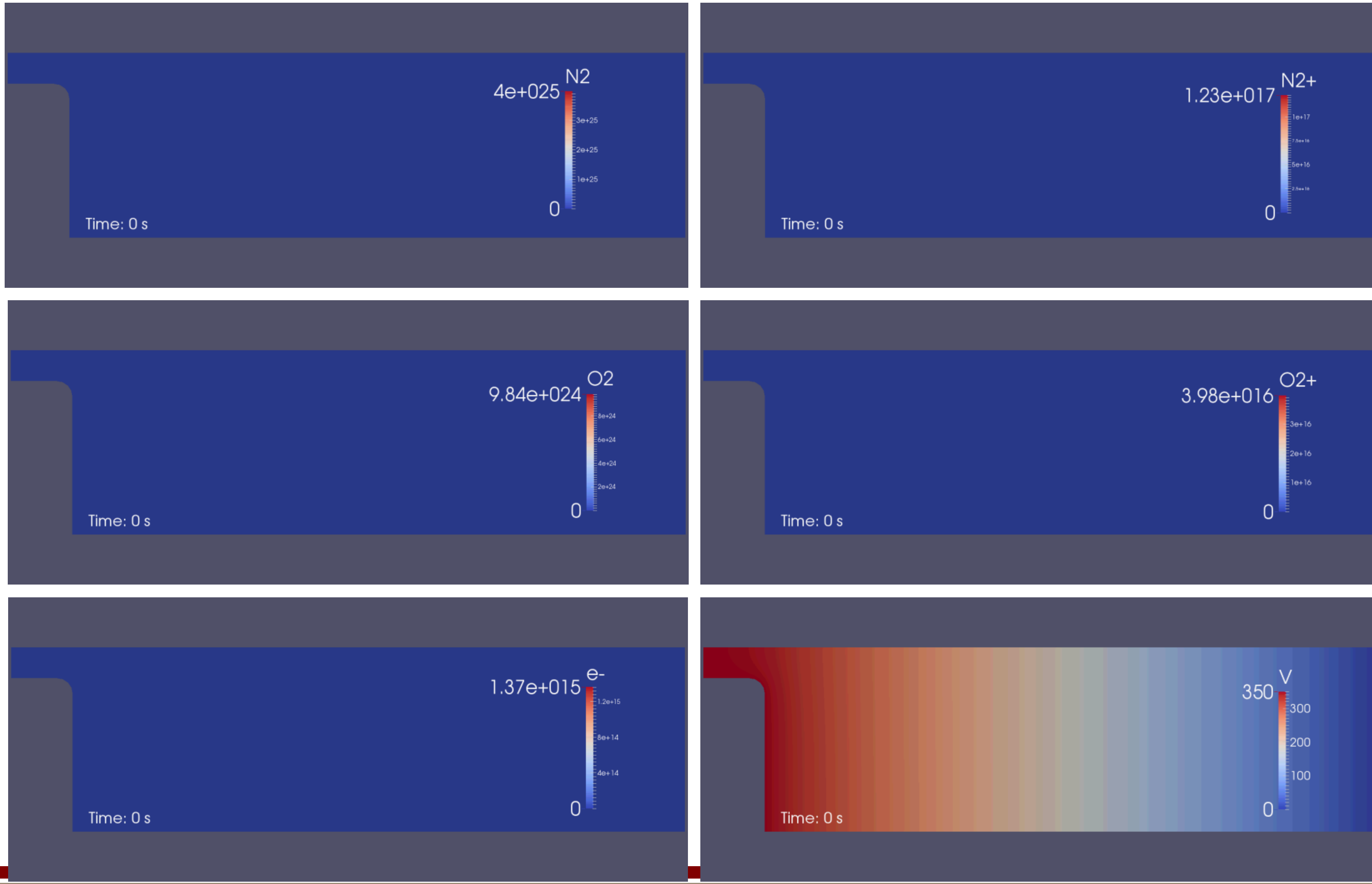
DSMC patch size is dynamically adjusted based on the local mean free path λ_{mfp}:

1. Compute λ_{mfp} for each interaction on an elemental basis (using all species)
2. For each interaction, average λ_{mfp} over elements in the oct-tree cell
3. Take the minimum of all the average λ_{mfp} and divide by 2, use this to size patches using the oct-tree algorithm



- Air injected at high velocity and high temperature from the anode
- Low density electrons injected from the cathode
- Air ionizes and eventually will form plasma and break the gap

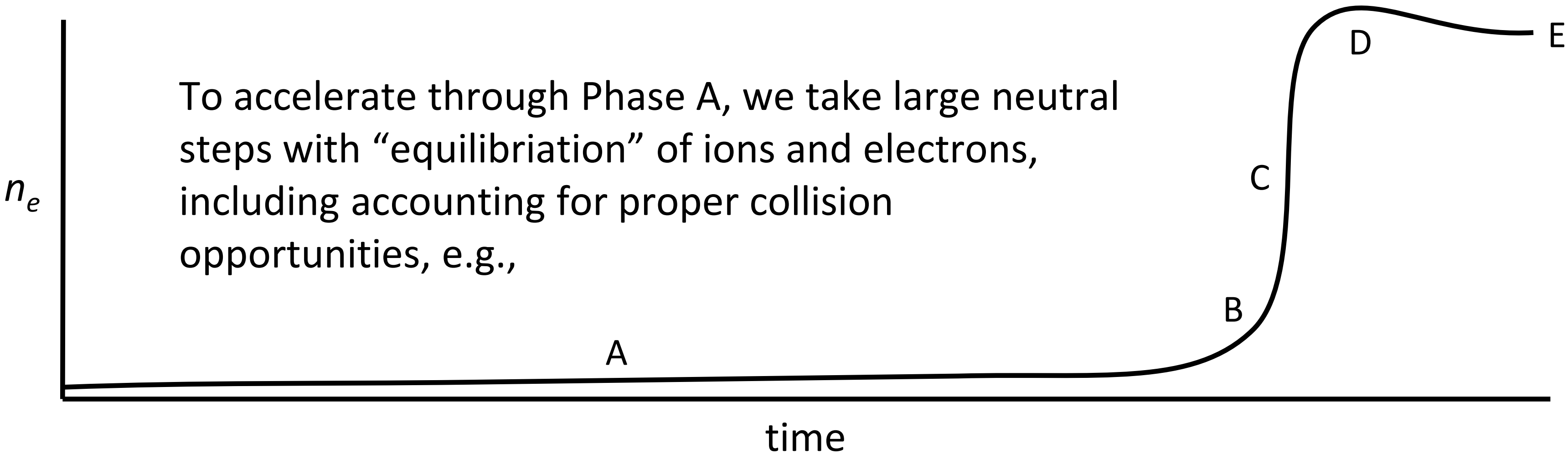
Managing *Δx*: Dynamic Sizing of DSMC Cells



Managing *Δx*: Dynamic Sizing of DSMC Cells

20x speed up on "box" problem

Managing Δt: Quasi-Static Acceleration



To accelerate through Phase A, we take large neutral steps with “equilibriation” of ions and electrons, including accounting for proper collision opportunities, e.g.,

- For each of 400 $\Delta t_{neutral}$ steps,
  - move neutrals
  - neutral-neutral interactions
  - for each of 10 Δt_{ion} steps,
    - move ions
    - ion-neutral interactions
    - ion-ion interactions
    - for each of 10 $\Delta t_{electron}$ steps,
      - move electrons
      - enhanced electron-* interactions

and

- For each of 40 10 x $\Delta t_{neutral}$ steps,
  - move neutrals
  - neutral-neutral interactions
  - for each of 100 Δt_{ion} steps,
    - move ions
    - ion-neutral interactions
    - ion-ion interactions
    - for each of 10 $\Delta t_{electron}$ steps,
      - move electrons
      - enhanced electron-* interactions

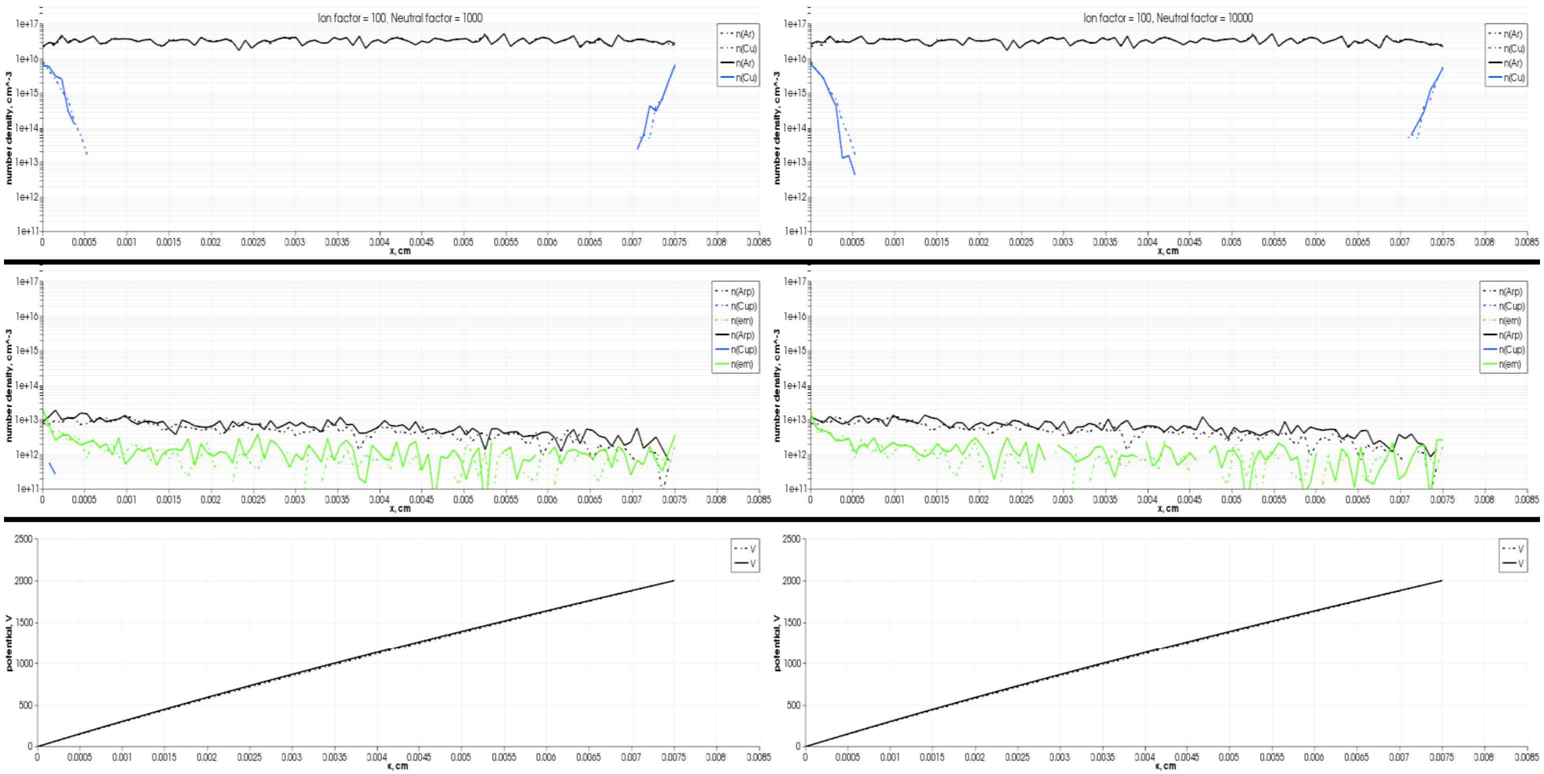
Managing Δt: Quasi-Static Acceleration

- Dashed lines are no acceleration.
- Neutral sputtering BC's.
- Cathode on left, anode on right.
- Influx of e- from cathode.

n_n
Ar background
Cu from surface

n_i for Ar+, Cu+
n_e

V

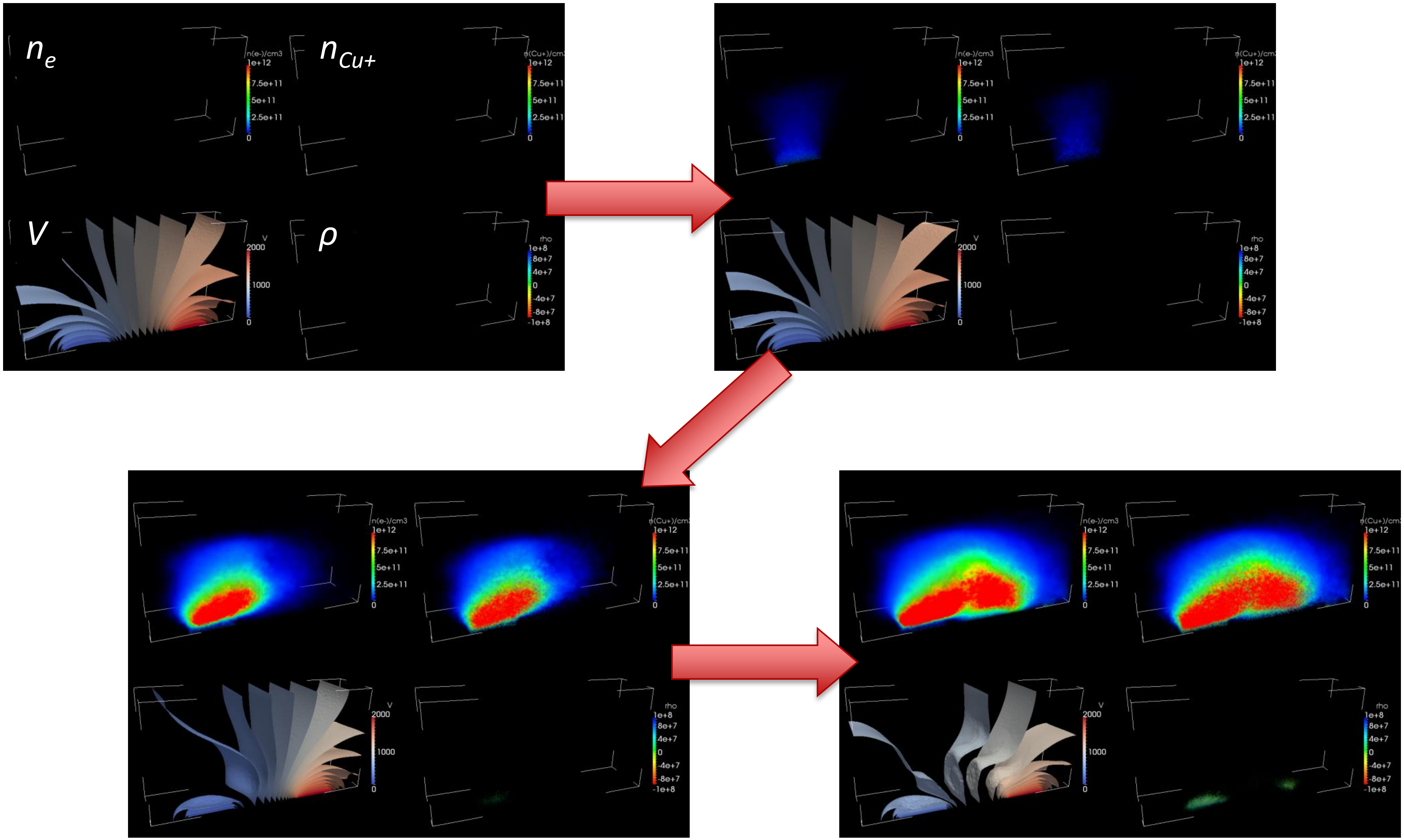


400 neutral steps

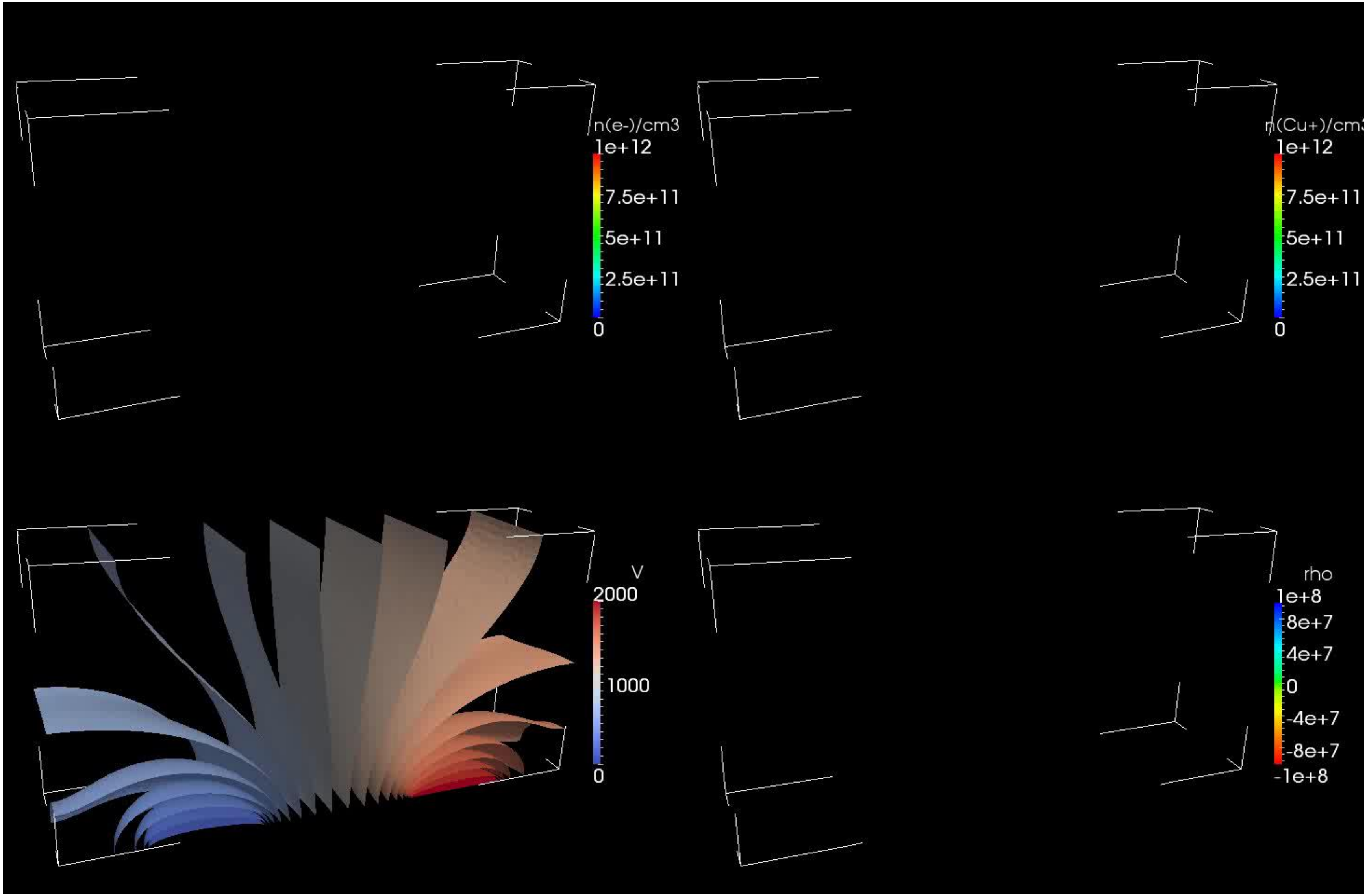
10x speed up!

40 (larger) neutral steps

3D Simulation (Not Vacuum)



3D Simulation (Not Vacuum)



Conclusions & Other Pursuits

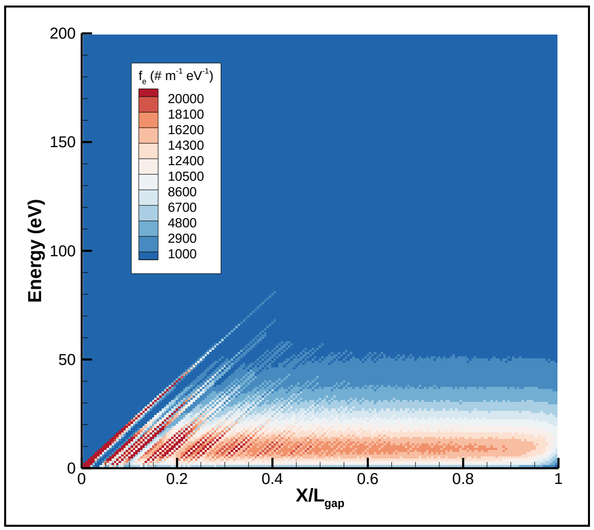
Simulating vacuum arcs is *extremely* expensive with vanilla PIC-DSMC methods. We are concurrently pursuing better physics models (not presented here) and more efficient algorithms with acceptable approximation errors to address these extreme simulation challenges.

Other areas we are pursing / have pursued include:

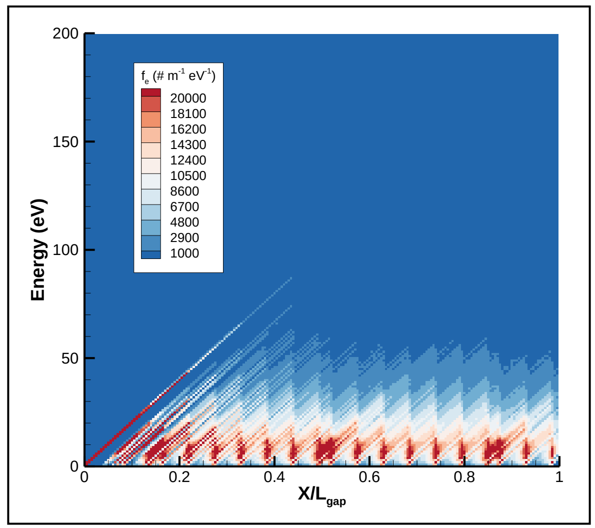
- Implicit kinetic methods
- Oct-tree DSMC collision mesh separate from PIC mesh
- Particle-Particle Particle-Mesh (P^3M) methods
- Dynamic load balancing and other scaling improvements
- Stochastic cathode hot spot models
- Photoionization, photoemission

Electron Energy Distribution Function (EEDF)

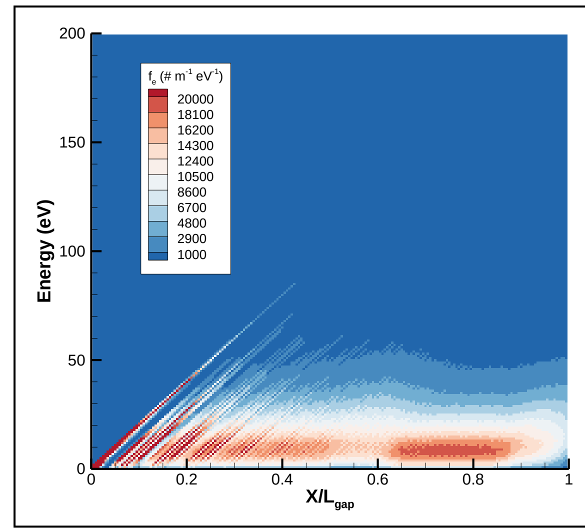
10 N_2 particles per cell



0.1 N_2 particles per cell



0.1 N_2 particles per cell with patching



- Using patching gives a more realistic EEDF (crucial for simulating accurate breakdown voltages)
- Patching allows one to use fewer N_2 particles