Development of a performance portable non-equilibrium plasma fluid solver on adaptive grids

Hari Sitaraman[1], Nicholas Deak[1]

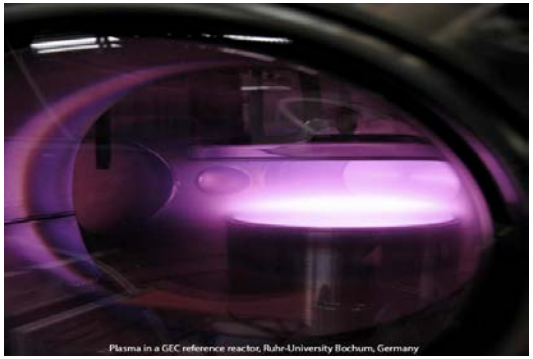
GEC 2024

Date: Oct 1st , 2024

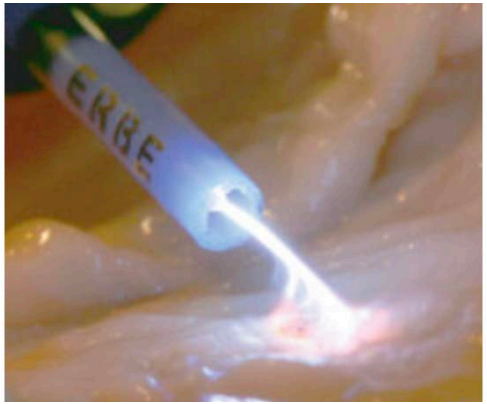
Funded by US, Dept. of Energy, Laboratory directed research and development program

Introduction and motivation

- Non thermal plasmas have numerous applications both at low- and high-pressure regimes
- Fluid models make simulations at higher pressures tractable compared to particle methods
- Plasma fluid models are complex:
  - Stiffness from electron timescales
  - Complex chemistry representation
  - PDE solves and linear systems
- Advent of new compute architectures: need GPU compatible fluid models



GEC reference cell* (~ 0.05-0.5 Torr)



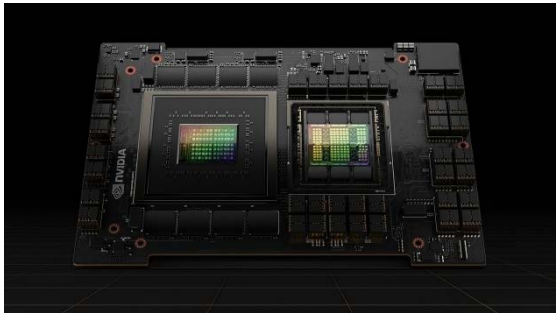
argon plasma coagulator** (~760 Torr)

*https://www.apsgec.org/main/

**Zenker, Matthias. "Argon plasma coagulation." *GMS Krankenhaushygiene interdisziplinar* 3.1 (2008).

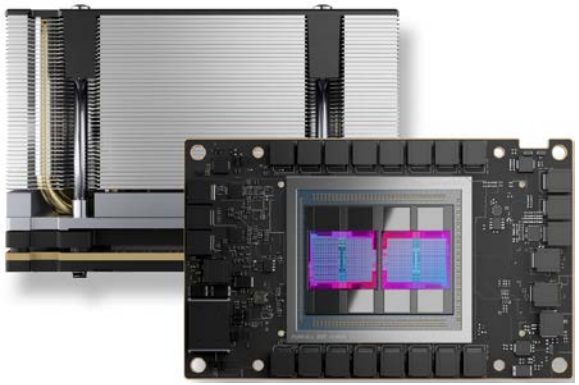
Objective and outline

- Development of a non-equilibrium plasma fluid solver
  - Modern numerical techniques
    - Adaptive meshing
    - Higher order schemes
  - Performance portability
    - Runs on CPUs and GPUs (NVIDIA/AMD/Intel)



NVIDIA H100*

- Outline
  - Mathematical model
  - Numerical methods
  - Programming paradigms
  - Verification tests
  - CPU/GPU performance
  - Example case studies



AMD MI250x**

*https://www.nvidia.com/en-us/data-center/h100/
**https://www.amd.com/en/products/accelerators/instinct/mi200/mi250x.html

Plasma fluid model

Non-equilibrium plasma model – Governing equations

- Two temperature model
  - Non-equilibrium: Electron temperature Te >> Gas temperature Tg
  - Weakly ionized: Electron and ion densities << Neutral gas density

$$\frac{\partial n_k}{\partial t} + \vec{\nabla} \cdot \vec{\Gamma}_k = \dot{G}_k$$

Species Continuity Equation

$$\vec{\Gamma}_k = \mu_k n_k \vec{E} - D_k \vec{\nabla} n_k$$

Drift diffusion approximation

$$\nabla^2 \phi + \sum Z_k n_k = 0$$

$$\vec{E} = -\vec{\nabla}\phi$$

Poisson equation for electrostatic potential

$$\frac{\partial E_e}{\partial t} + \vec{\nabla} \cdot ((E_e + P_e)\vec{u}_e) = \vec{\nabla}.\left(k_e \vec{\nabla} T_e\right) + \dot{S}_e$$

Electron energy equation

$$E_e = \frac{3}{2} n_e k_B T_e \qquad P_e = n_e k_B T_e$$

Plasma model – Governing equations

$$\dot{S}_e = -e\vec{\Gamma}_e \cdot \vec{E} - \frac{3}{2} n_e k_B (T_e - T_g) \frac{2m_e}{m_b} \nu - \sum \Delta E_i r_i$$

Electron energy source term

Electron Joule heating

Electron elastic collisions

Electron inelastic collisions

Boundary conditions at solid surfaces

- **electrons** $\vec{\Gamma}_e.\hat{n} = \frac{1}{4} n_e \left(\frac{8k_B T_e}{\pi m_e} \right)^{1/2}$ — Maxwellian number flux
- **ions** $\vec{\Gamma}_i.\hat{n} = max\left(-\mu_i n_i \vec{\nabla}\phi.\hat{n}, 0 \right)$ — Drift dominated flux
- **neutrals** $\vec{\Gamma}_n.\hat{n} = \frac{1}{4} n_n \left(\frac{8k_B T_g}{\pi m_e} \right)^{1/2}$ — Wall neutral loss
- **electron energy** $\vec{\Gamma}_{EE}.\hat{n} = 2k_B T_e \Gamma_e$ — Maxwellian energy flux

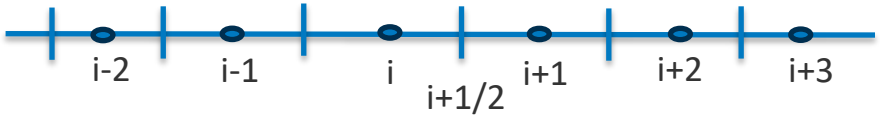
Numerical method

$$\frac{\partial \phi}{\partial t} + \vec{\nabla} \cdot (\vec{V}\phi) = \vec{\nabla} \cdot \left(D\vec{\nabla}\phi \right) + S$$

Unsteady advection-diffusion-reaction equation

- Advection-diffusion-reaction equations are solved using
  - Finite volume method on Cartesian adaptive grids
    - 5th order advection, 2nd order central diffusion
  - Implicit time integration
    - Second order spectral deferred correction (SDC)
  - Can be solved in 2D, 2D-axisymmetric and 3D formulations
    - 1D is 2D with 2-4 cells along transverse axis
  - Standard boundary conditions also included
    - Dirichlet, Homogenous/inhomogenous Neumann, Robin

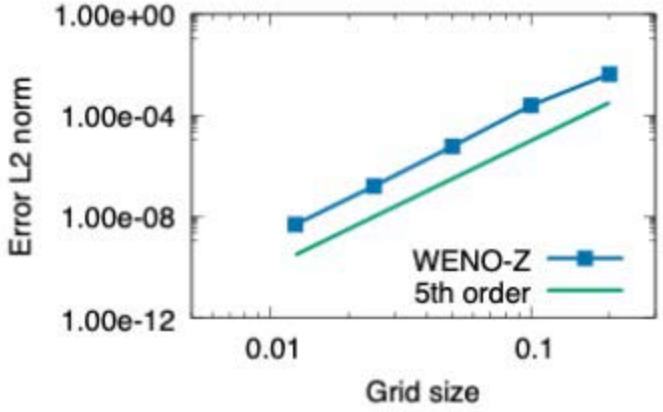
Advection scheme



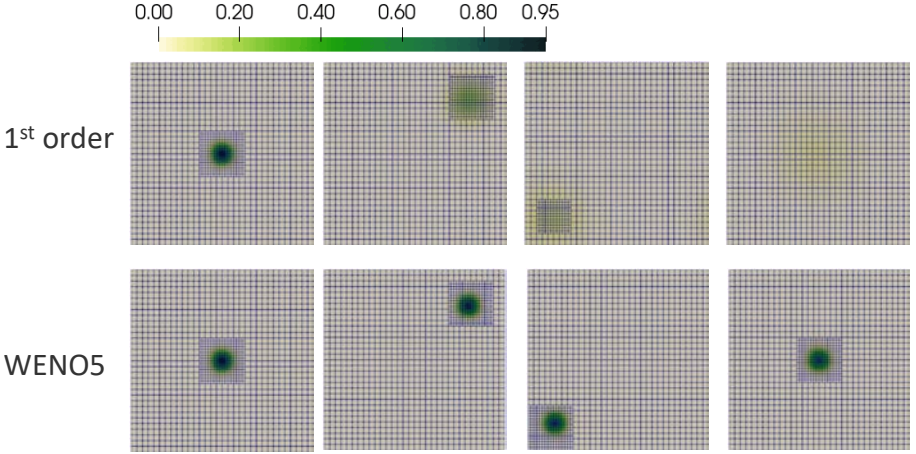
$$f_{i+1/2} = f^+ + f^-$$

$$f^+ = W(f_{i-2}, f_{i-1}, f_i, f_{i+1}, f_{i+2}) + |\lambda| W(\phi_{i-2}, \phi_{i-1}, \phi_i, \phi_{i+1}, \phi_{i+2})$$

$$f^- = W(f_{i-1}, f_i, f_{i+1}, f_{i+2}, f_{i+3}) - |\lambda| W(\phi_{i-1}, \phi_i, \phi_{i+1}, \phi_{i+2}, \phi_{i+3})$$



- 5th order WENO scheme is applied to each PDE
  - Nonlinear dissipation with smoothness indicators from WENO-Z* scheme
- Needs 3 layers of ghost cells as opposed to 1 for 1st order schemes
- Provides greater accuracy with only about 30% increase in cost



Diagonal advection in a periodic domain (32x32) base grid

*Borges et al., JCP, 227.6 (2008): 3191-3211.

Time discretization scheme

$$\frac{du}{dt} = f$$

ODE system

$$u^{n+1} = u^n + \frac{1}{2}\left(f(u^n) + f(u^{n+1})\right)\delta t$$

Second order implicit scheme

$$u^{k+1} = u^n + \frac{1}{2}\left(f(u^n) + f(u^k)\right)\delta t \qquad u^0 = u^n$$

Iterative second order scheme

- Within each k iteration
  - Solve Poisson
  - Solve electron density (Backward Euler)
  - solve electron energy (Backward Euler)
  - Solve ions and neutrals (Backward Euler)
- Just like spectral deferred correction (SDC)* scheme
  - One iteration – 1st order accurate
  - 2 iterations – 2nd order accurate

*Minion, Commun. Math. Sci. 1(3): 471-500 (September 2003)

Chemistry implementation

```yaml
- equation: E + O2 + M => O2- + M  # Reaction 44
  type: three-body
  rate-constant: {A: 3.0e-30, b: 0.0, Ea: 0.0}
  note: Electron attachment to other species
- equation: E + O + M => O- + M  # Reaction 45
  type: three-body
  rate-constant: {A: 1.0e-31, b: 0.0, Ea: 0.0}
- equation: O- + O => O2 + E  # Reaction 46
  rate-constant: {A: 1.5e-10, b: 0.0, Ea: 0.0}
  note: Electron detachment
- equation: O- + H => OH + E  # Reaction 47
  rate-constant: {A: 5.0e-10, b: 0.0, Ea: 0.0}
- equation: O- + H2 => H2O + E  # Reaction 48
  rate-constant: {A: 6.72e-10, b: 0.0, Ea: 0.0}
- equation: O- + C => CO + E  # Reaction 49
  rate-constant: {A: 5.0e-10, b: 0.0, Ea: 0.0}
- equation: O- + CO => CO2 + E  # Reaction 50
  rate-constant: {A: 6.5e-10, b: 0.0, Ea: 0.0}
```

CANTERA yaml format

```
AMREX_GPU_HOST_DEVICE AMREX_FORCE_INLINE void
productionRate(amrex::Real *wdot, const amrex::Real *sc, const amrex::Real T,
               const amrex::Real Te, amrex::Real EN, amrex::Real *enerExch) {

  amrex::Real tc[5] = {log(T), T, T * T, T * T * T,
                       T * T * T * T}; // temperature cache
  const amrex::Real invT = 1.0 / tc[1];
  const amrex::Real logT = log(T/300.0);

  // reference concentration: P_atm / (RT) in inverse mol/m^3
  const amrex::Real refC = 101325 / 8.31446 * invT;
  const amrex::Real refCinv = 1 / refC;

  for (int i = 0; i < 44; ++i) {
    wdot[i] = 0.0;
  }

  {
    // reaction 12:  CO2 + E => CO + O + E
    Janev_sum = 0.0;
    amrex::Real k_f;
    Ffit_coefs = {28.3950215483559, -119.510009432235, 160.437496467960, -74.1425574458809};
    double Ffit_A = 1.50741518933862e-16;
    for(int j = 0; j<4; j++) Janev_sum += Ffit_coefs[j] * invTe_pow[j];
    k_f = Ffit_A * exp(Janev_sum) * 6.02214085774e23;
    const amrex::Real qf = k_f * (sc[E_ID] * sc[CO2_ID]);
    const amrex::Real qr = 0.0;
    const amrex::Real qdot = qf - qr;
    wdot[O_ID] += qdot;
    wdot[CO_ID] += qdot;
    wdot[CO2_ID] -= qdot;
  }
```

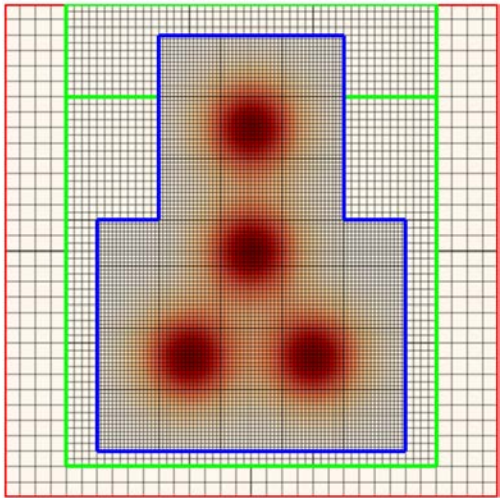
Inline GPU compatible functions

- CANTERA yaml files like combustion chemistry
  - Python parser* converts yaml to C++ functions for production rates
  - Currently hand-written non-Arrhenius rates, transport coefficients
    - **Plasma chemistry is different from combustion!!**

*https://github.com/AMReX-Combustion/PelePhysics

AMReX programming paradigm

- We use performance portable adaptive meshing library, AMReX*
- Block structured adaptive Cartesian grids, hybrid parallelization
- All levels advanced at the same timestep
- Multilevel Multigrid (MLMG) based backward Euler scheme
  - Cell centered implicit diffusion/explicit advection



- For stiff systems, we utilize AMReX's HYPRE** interface for algebraic multigrid
- Performance portability from parallel for lambdas

```
// update residual
amrex::ParallelFor(bx, [=] AMREX_GPU_DEVICE(int i, int j, int k) {
    dsdt_arr(i, j, k) = (flux_arr[0](i, j, k) - flux_arr[0](i + 1, j, k)) / dx[0]
    + rxn_arr(i,j,k,captured_specid);
    dsdt_arr(i,j,k) += (flux_arr[1](i, j, k) - flux_arr[1](i, j + 1, k)) / dx[1];
    dsdt_arr(i,j,k) += (flux_arr[2](i, j, k) - flux_arr[2](i, j, k + 1)) / dx[2];
});
```

*Zhang, Weiqun, et al., *The Journal of Open Source Software* 4.37 (2019): 1370. **Falgout et al., " *Int. Conf. on comp. sci.*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.

Code verification

Method of manufactured solutions (MMS)

Plasma fluid equations

$$\frac{\partial n_e}{\partial t} + \vec{\nabla}.\Gamma_e = S_e$$

$$\Gamma_e = \mu_e n_e \vec{E} - D_e \vec{\nabla} n_e$$

$$\frac{\partial n_i}{\partial t} + \vec{\nabla}.\Gamma_i = S_i$$

$$\Gamma_i = \mu_i n_i \vec{E} - D_i \vec{\nabla} n_i$$

$$\frac{\partial \epsilon_e}{\partial t} + \vec{\nabla}.\Gamma_\epsilon = S_\epsilon$$

$$\epsilon_e = \frac{3}{2} n_e k_B T_e$$

$$\Gamma_\epsilon = \frac{5}{3}\left(\mu_e \epsilon_e \vec{E} - D_e \vec{\nabla}\epsilon_e\right)$$

$$\nabla^2 \phi = \frac{e}{\epsilon_0}\left(n_e - \sum_i Z_i n_i\right)$$

Simplified MMS problem

$$\frac{\partial n}{\partial t} + \frac{\partial \Gamma}{\partial x} = \left(\frac{5}{3}x^4 - \frac{x}{6} - 2\right)$$

$$\Gamma = \mu n E - D\frac{\partial n}{\partial x}$$

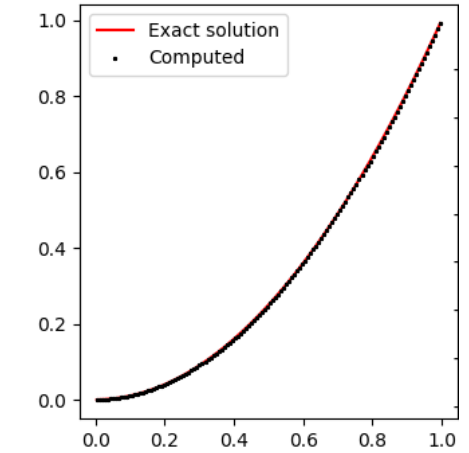
$$E = -\frac{\partial \phi}{\partial x} \quad \mu = -1 \quad D = 1$$

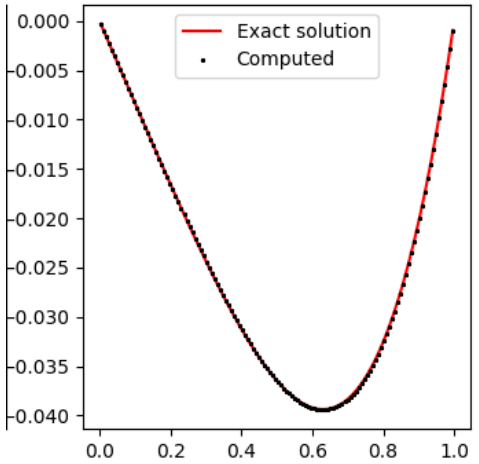
$$\frac{\partial^2 \phi}{\partial x^2} = n$$

$$n(0) = 0 \quad n(1) = 1 \quad \phi(0) = 0 \quad \phi(1) = 0$$

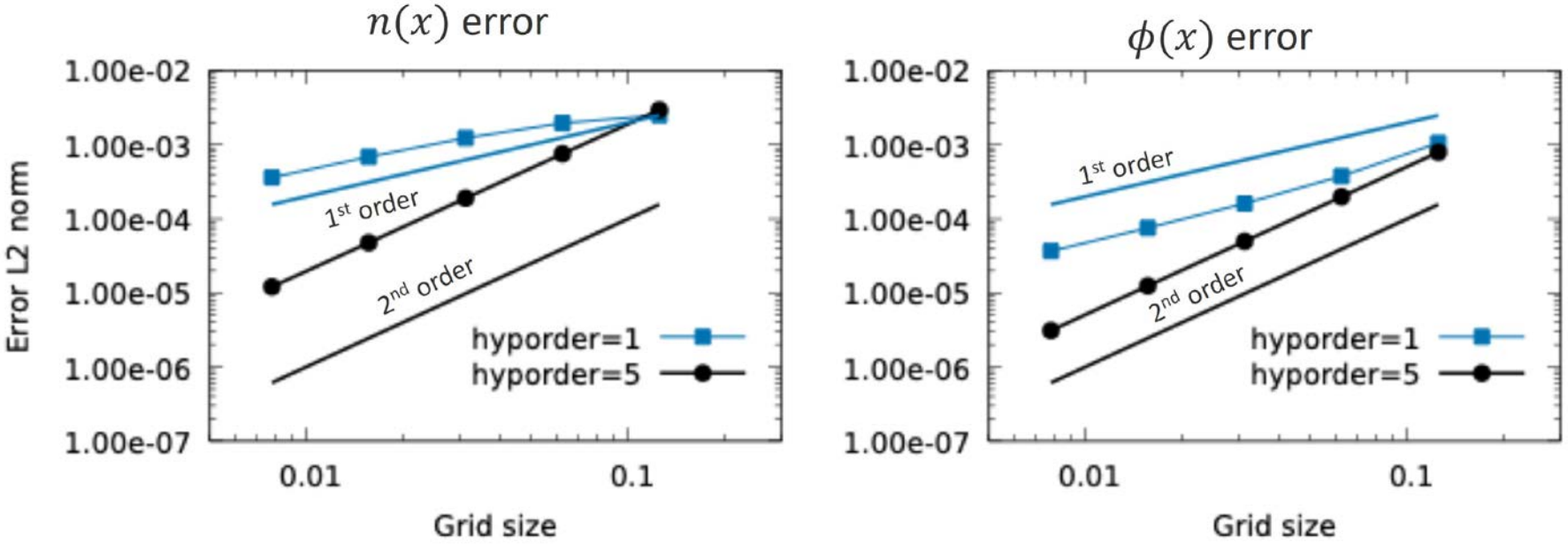
$$\text{solution } n = x^2 \quad \phi = \frac{1}{12}(x^4 - x)$$

MMS provides a way to check the accuracy of our schemes and correctness of our implementation



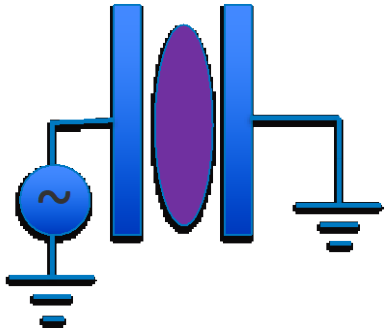


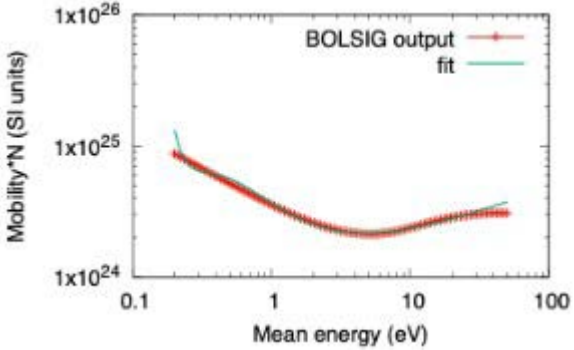
Method of manufactured solutions

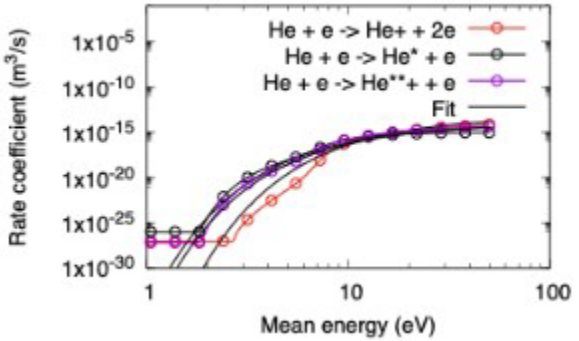


- 5th order WENO advection + 2nd order diffusion follows the leading order 2nd order convergence
- We are getting theoretical convergence rates for our spatial discretization schemes, indicating correctness of our implementation

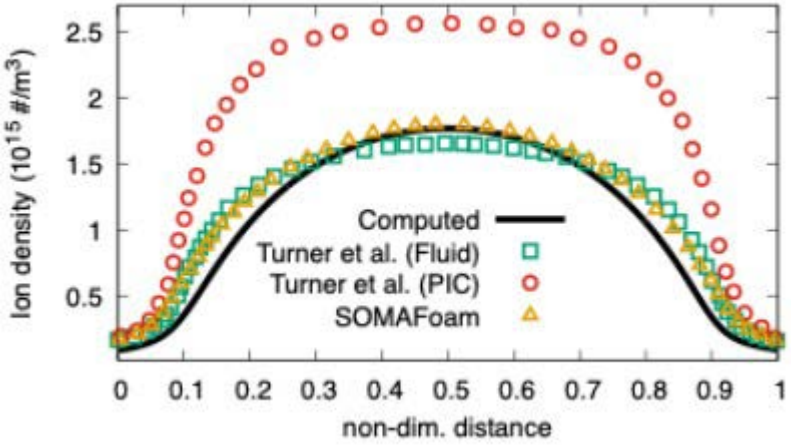
He capacitive discharge verification





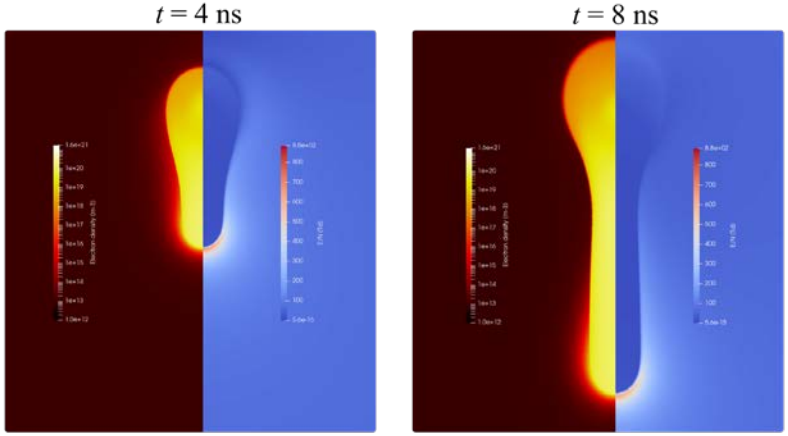


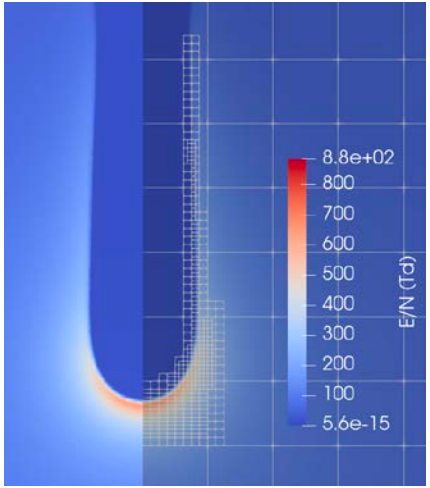
- 1D benchmark case at 1 Torr, 300K from Turner et al. *Physics of Plasmas* 20.1 (2013).
- Helium chemistry with He+,He*,He**
  - Cross sections obtained from Turner paper
  - Offline BOLSIG solve and fitting for rates and electron transport properties
- Compares well with Turner's fluid model and open-source code, SOMAFoam*, after ~ 2000 RF cycles

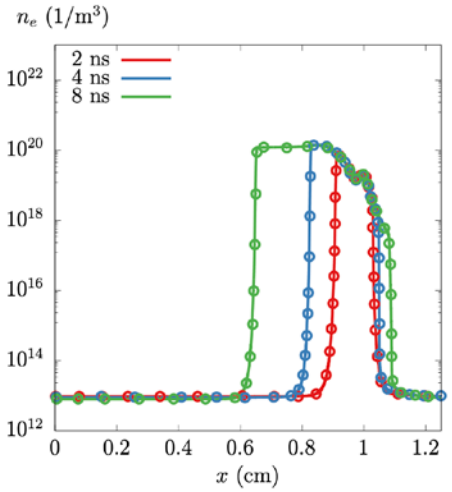


*Verma and Ayyaswamy, *Computer Physics Communications* 263 (2021): 107855.

Streamer verification

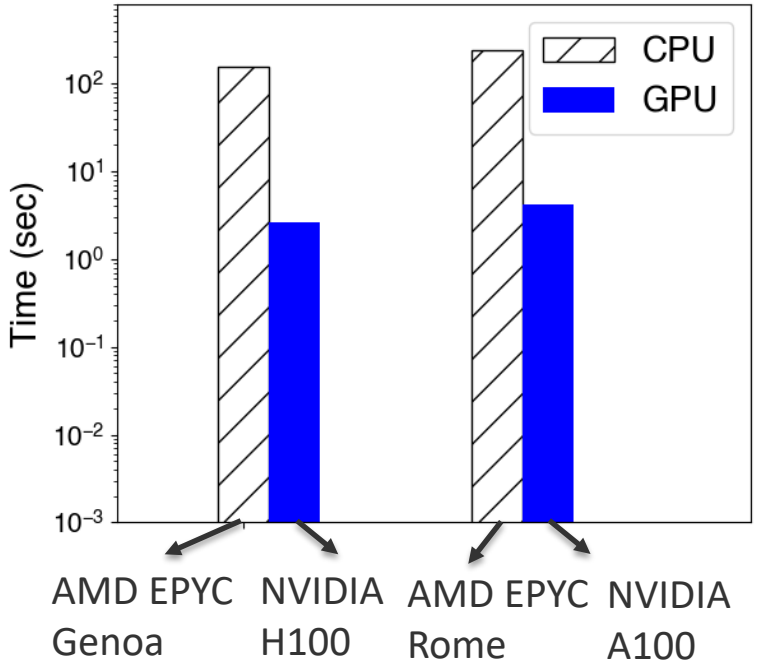


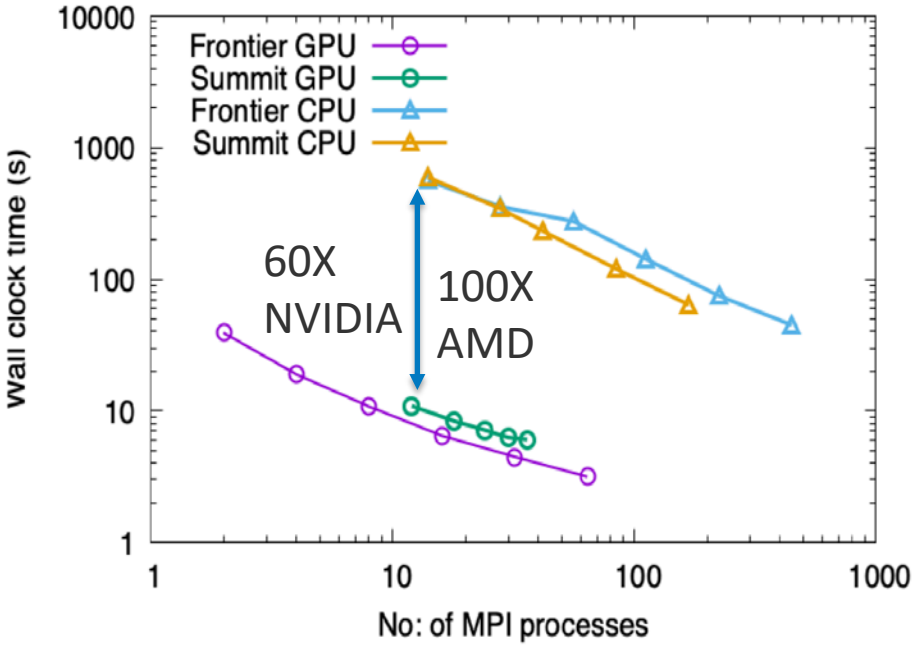




- Axisymmetric streamer test case discussed in Bagheri et al., PSST, 2018
- Initiation with seed charge and applied electric field
- Streamer forms and propagates to the bottom grounded boundary with top-to-bottom applied field.
- Our code agrees well with axial profiles of Electron density and electric field (symbols: literature, line: *Vidyut3d*)
- Simulations conducted using 52 cores for approximately 3 hours to simulate 16 ns

CPU vs GPU performance





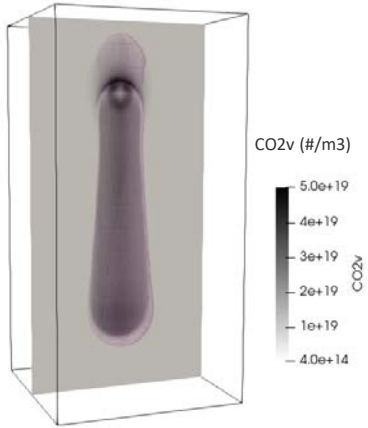
4 million cell 2D streamer case run for 10 steps. 1 GPU ~ 60X faster than 1 CPU

512x1024x512 = 0.25B cell 3D streamer case, ~ 20X speed up at the node level on ORNL Frontier

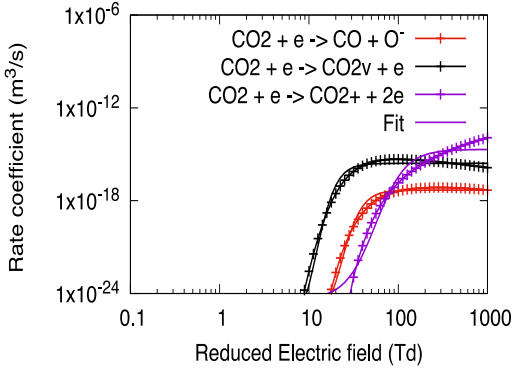
Plasma models for CO_2 conversion

- Atmospheric pressure discharges proceed through a streamer breakdown mechanism
- Local reduced electric fields (E/N) dictate electron impact ionization/excitation/dissociation effects

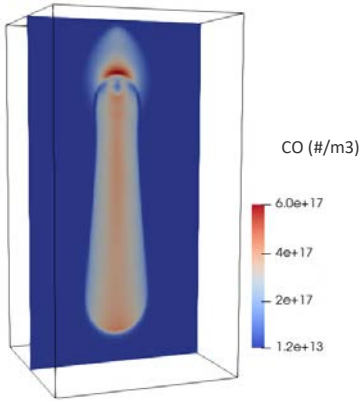
NREL's coaxial DBD



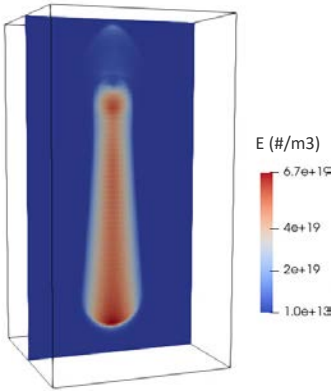
CO2 vibrational states in streamer



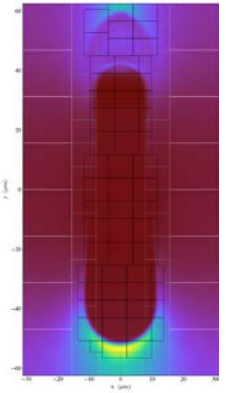
Electron impact rates as a function of E/N



CO formation in the streamer



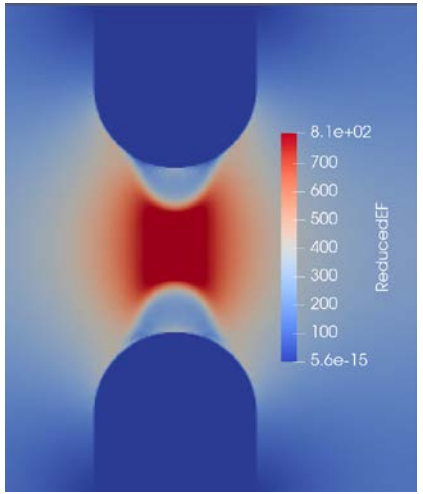
Electron density in streamer



Electric field focusing at streamer head

Conclusions and future work

- Conclusions
  - Developed a non-equilibrium plasma fluid solver
    - Cartesian block structured adaptive meshing
    - 5th order advection, central diffusion
    - AMReX library
  - Verified against benchmark cases
    - Method of manufactured solutions, He Capacitive discharge, Atmospheric streamer
  - Performance
    - 1 NVIDIA GPU ~ 60X faster than 1 CPU
    - On node GPU performance gain ~ 20X on ORNL frontier



- Future work
  - Complex geometry inclusion
  - photoionization
  - New boundary conditions
    - Dielectric charge build up, secondary emission, external circuit

Thank you

www.nrel.gov

NREL/PR-2C00-91564

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. This work was supported by the Laboratory Directed Research and Development (LDRD) Program at NREL. Funding also provided by DOE, Office of Science, Science Foundations for Energy Earthshots (SFEE) program is acknowledged. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.