SAND2021-15038 C
Unclassified Unlimited Release

Sandia National Laboratories

EMPIRE: A Performance Portable Plasma Simulation Code

EMPIRE

Trilinos User Group Meeting, December 1st, 2021

PRESENTED BY

Roger Pawlowski

K. A. Cartwright, D. A. O. McGregor, E. C. Cyr, C. Glusa, J. Hu, S. Miller, E. G. Phillips, E. Love, W. J. McDoniel, P. J. Christenson, R. M. J. Kramer, T. D. Pointon, N. A. Roberds, M. S. Swan, K. S. Bell, T. M. Flanagan, C. H. Moore, T. C. Powell, S. Shields, D. Sirajuddin, J. Elliott, B. Kelley, J. Lifflander, N. Slattengren, P. Miller and M. T. Bettencourt

U.S. DEPARTMENT OF ENERGY NNSA

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Motivation

- Sandia's foundation is science-based engineering, in which fundamental science, computer models, and unique experimental facilities come together so researchers can understand, predict, and verify weapon systems performance.
- EMPIRE (ElectroMagnetic Plasma In Realistic Environments) is a part of Sandia's next-generation plasma modeling and simulation capability.
  - Developed under DOE's ASC/ATDM program starting in 2015
- Goals:
  - Simulate plasmas over a broad density range, with Particle-In-Cell (PIC) dominating at low densities, fluid at high densities, and a hybrid approach in the middle.
  - Performance portability on next-generation architectures
- Code Design:
  - Three distinct physics capabilities that can be run stand-alone or coupled in a hybrid capability: Electromagnetics, PIC, Fluids
  - Built on top of many software components (Trilinos, Kokkos, Darma, …)

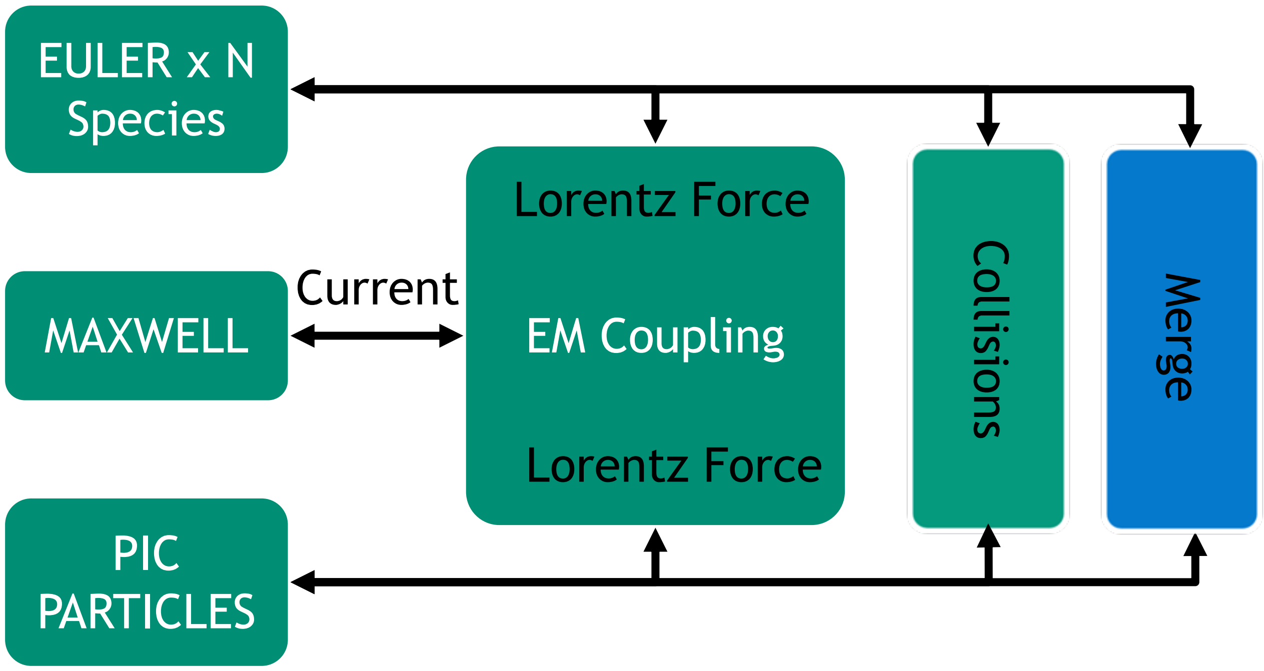
The hybrid kinetic-fluid plasma model

EMPIRE
Hybrid Plasma Modeling

Fluid: for efficiency at high pressures
- Euler's equations evolve species
- DG finite-element discretization

Maxwell:
- Evolves electromagnetic fields
- CG Compatible finite-element discretization

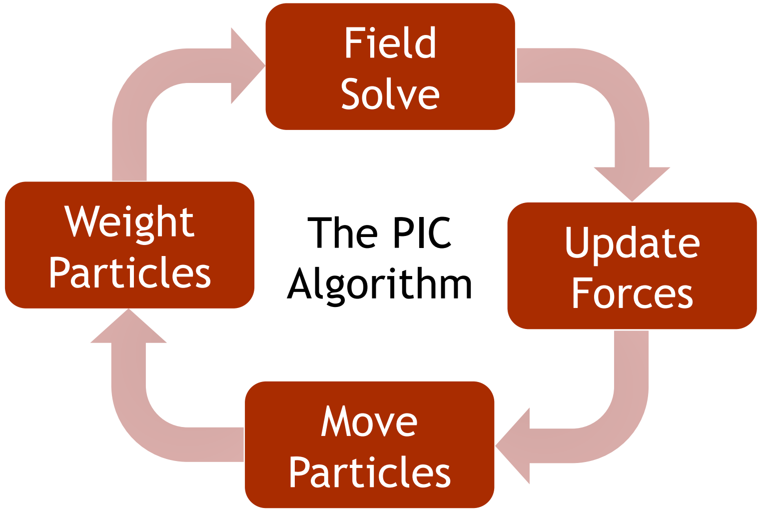
Kinetic: for accuracy at low pressures
- Boltzmann equations evolve species
- PIC: Lagrangian particle discretization



Multiple algorithmic advances have been required for hybrid capability:

- Fluid/Maxwell coupling enforces divergence involution
- Finite-element stabilization methods
- Fluid/particle collisions and merge
- Step over stiff plasma modes using implicit/explicit time integration

The Plasma Model



Relativistic Klimontovich Equation

- Particles can collide: elastic, ionization, excitation etc.

$$\frac{\partial N_s(\boldsymbol{x},\boldsymbol{u},t)}{\partial t} + \boldsymbol{v}\cdot\nabla_x N_s + \frac{q_s}{m_s}\left(\mathbf{E} + \frac{\boldsymbol{v}}{c}\times\mathbf{B}\right)\cdot\nabla_u N_s = \left.\frac{\partial N_s(\boldsymbol{x},\boldsymbol{u},t)}{\partial t}\right|_c$$

$$\rho(\boldsymbol{x},t) = \sum_{species} q_s \int d\boldsymbol{u} N_s(\boldsymbol{x},\boldsymbol{u},t)$$

$$\boldsymbol{J}(\boldsymbol{x},t) = \sum_{species} q_s \int d\boldsymbol{u}\boldsymbol{u} N_s(\boldsymbol{x},\boldsymbol{u},t)$$

Maxwell's Equations

$$\nabla\cdot\boldsymbol{D}(\boldsymbol{x},t) = \frac{\rho(\boldsymbol{x},t)}{\epsilon_0}$$

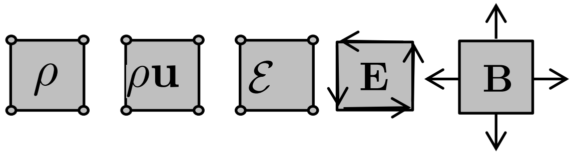
$$\nabla\cdot\mathbf{B}(\boldsymbol{x},t) = 0$$

$$\nabla\times\mathbf{E}(\boldsymbol{x},t) = -\frac{\partial\mathbf{B}(\boldsymbol{x},t)}{\partial t}$$

$$\nabla\times\boldsymbol{H}(\boldsymbol{x},t) = \mu_0\boldsymbol{J}(\boldsymbol{x},t) + \mu_0\epsilon_0\frac{\partial\boldsymbol{D}(\boldsymbol{x},t)}{\partial t}$$

Bettencourt et al, *EMPIRE-PIC: A Performance Portable Unstructured Particle-in-Cell Code*, Communications in Computational Physics, v30, 2021

Fluid Model: 5-Moment Multi-fluid Plasma



Density	$\frac{\partial \rho_a}{\partial t} + \nabla \cdot (\rho_a \mathbf{u}_a) = \sum_{b \neq a} (n_a \rho_b \nu_{ab}^{+} - n_b \rho_a \nu_{ab}^{-})$
Momentum	$\frac{\partial (\rho_a \mathbf{u}_a)}{\partial t} + \nabla \cdot (\rho_a \mathbf{u}_a \otimes \mathbf{u}_a + p_a I + \Pi_a) = q_a n_a (\mathbf{E} + \mathbf{u}_a \times \mathbf{B})$ $- \sum_{b \neq a} [\rho_a (\mathbf{u}_a - \mathbf{u}_b) n_b \nu_{ab}^{M} + \rho_b \mathbf{u}_b n_a \nu_{ab}^{+} - \rho_a \mathbf{u}_a n_b \nu_{ab}^{-}]$
Energy	$\frac{\partial \varepsilon_a}{\partial t} + \nabla \cdot ((\varepsilon_a + p_a) \mathbf{u}_a + \Pi_a \cdot \mathbf{u}_a + \mathbf{h}_a) = q_a n_a \mathbf{u}_a \cdot \mathbf{E} + Q_a^{src}$ $- \sum_{b \neq a} \left[(T_a - T_b) k \nu_{ab}^{E} - \rho_a \mathbf{u}_a \cdot (\mathbf{u}_a - \mathbf{u}_b) n_b \nu_{ab}^{M} - n_a \nu_{ab}^{+} \varepsilon_b + n_b \nu_{ab}^{-} \varepsilon_a \right]$
Charge and Current Density	$q = \sum_k q_k n_k \qquad \mathbf{J} = \sum_k q_k n_k \mathbf{u}_k$
Maxwell's Equations	$\frac{1}{c^2} \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{B} + \mu_0 \mathbf{J} = \mathbf{0} \qquad \nabla \cdot \mathbf{E} = \frac{q}{\epsilon_0}$ $\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = \mathbf{0} \qquad \nabla \cdot \mathbf{B} = 0$

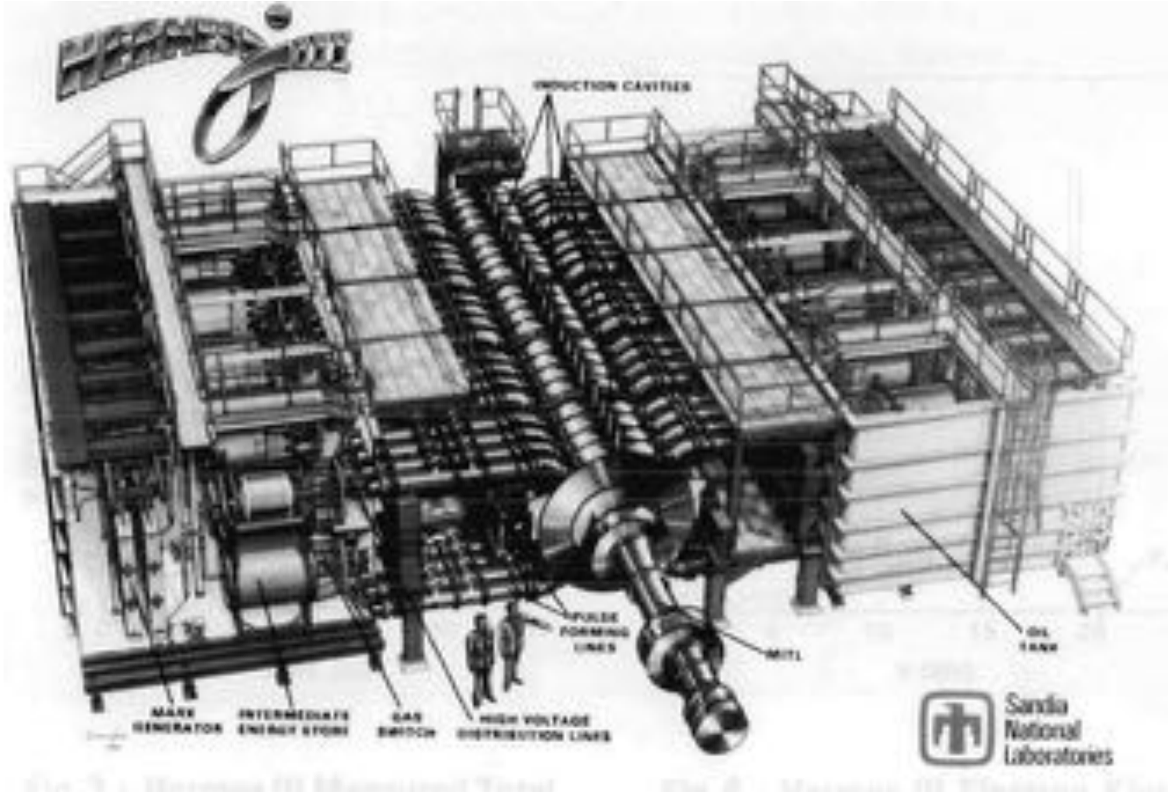
$$\rho_\alpha, \rho \mathbf{u}_\alpha, \varepsilon_\alpha \in H_\nabla(\Omega)$$

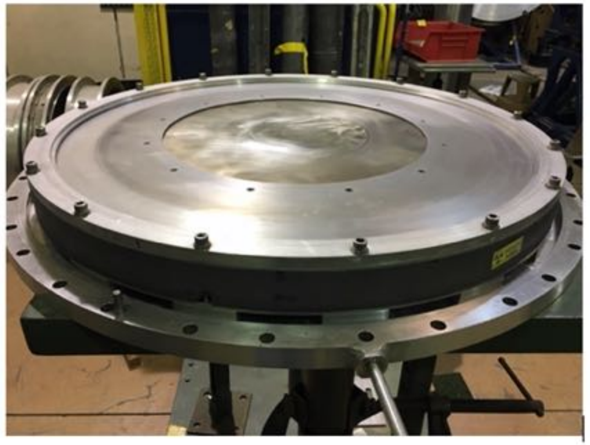
$$\mathbf{E} \in \mathbf{H}_{\nabla\times}(\Omega)$$

$$\mathbf{B} \in \mathbf{H}_{\nabla\cdot}(\Omega)$$

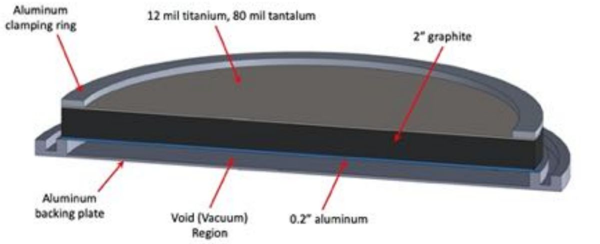
Target Problems

HERMES-III Background

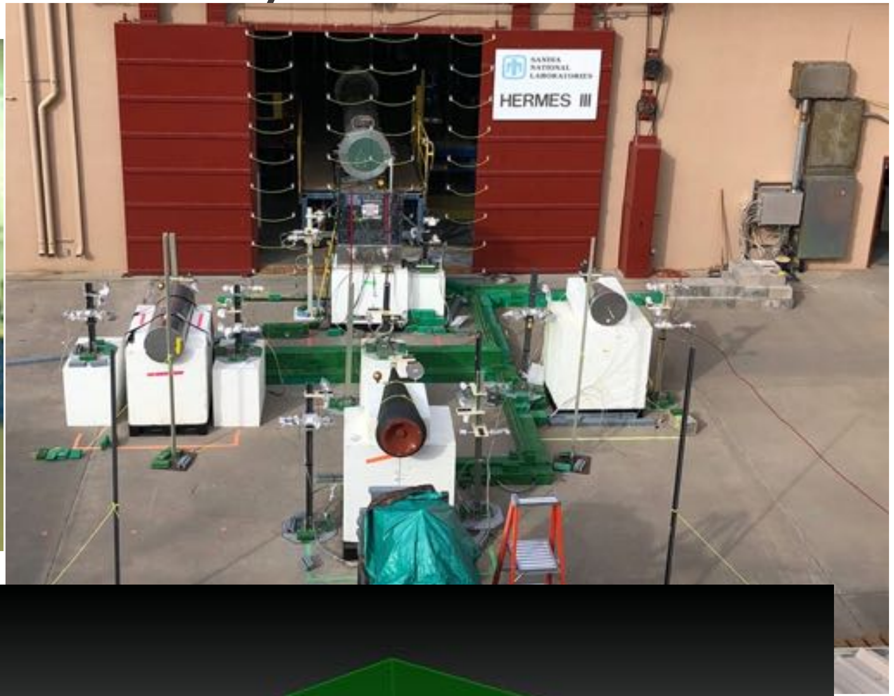


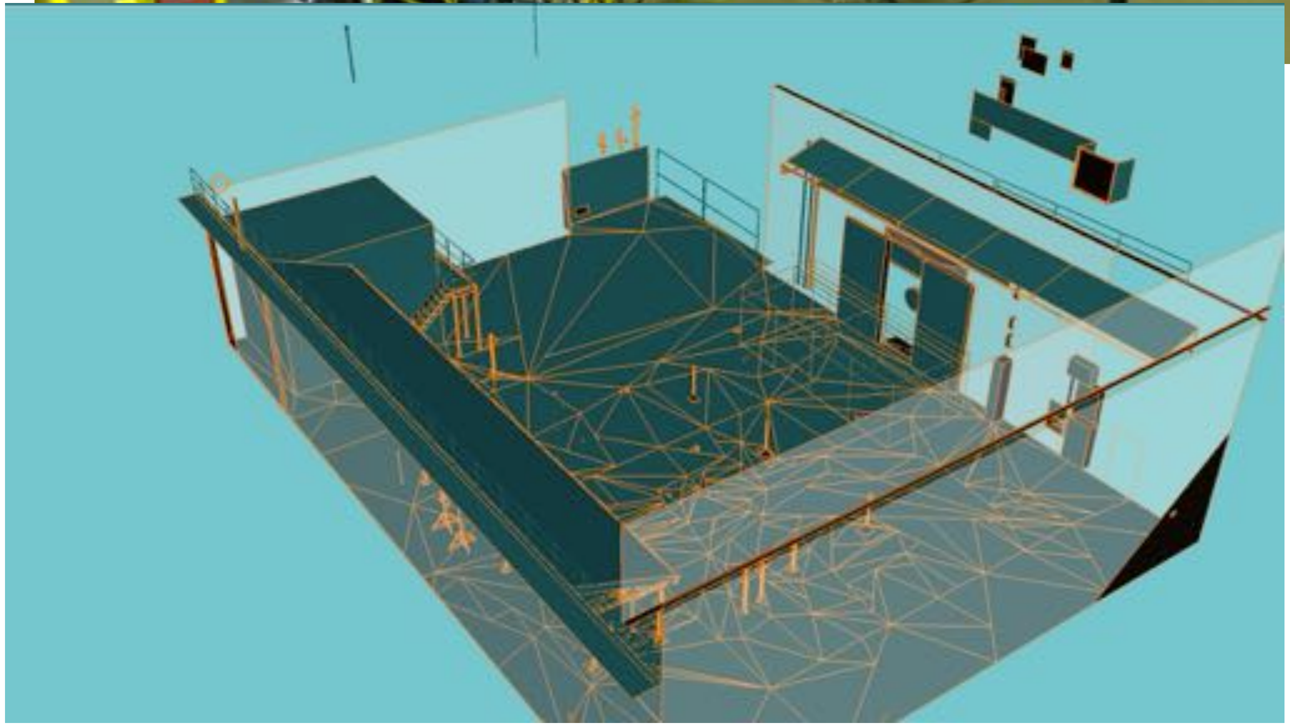


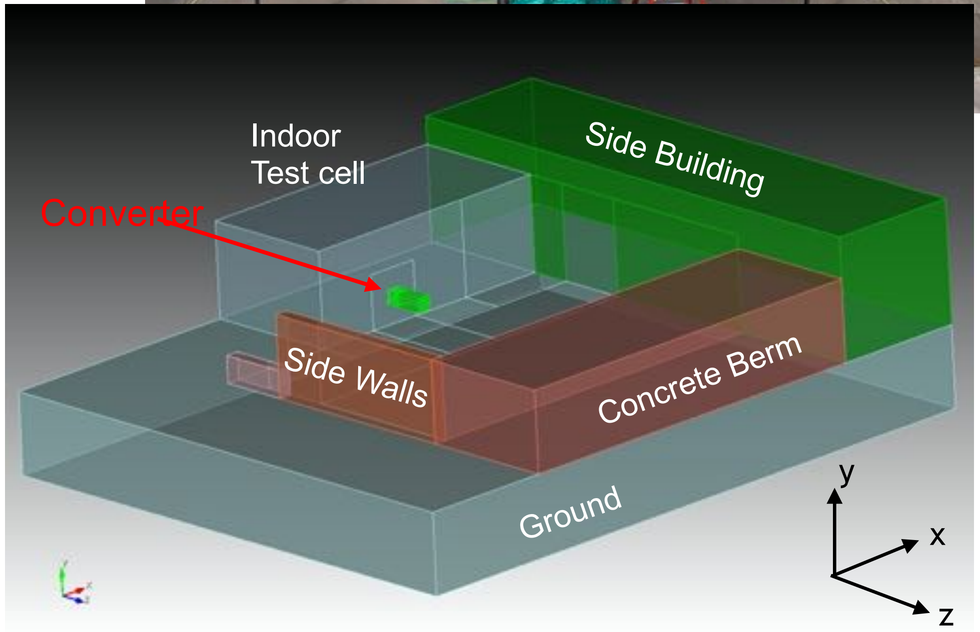
Gamma ray simulator
Nominally an 18MV, 550kA coaxial accelerator
Uses Inductive Voltage Adders (IVAs) to combine Marx pulses
Can operate in bremsstrahlung mode or an ion-diode mode (reverse polarization)



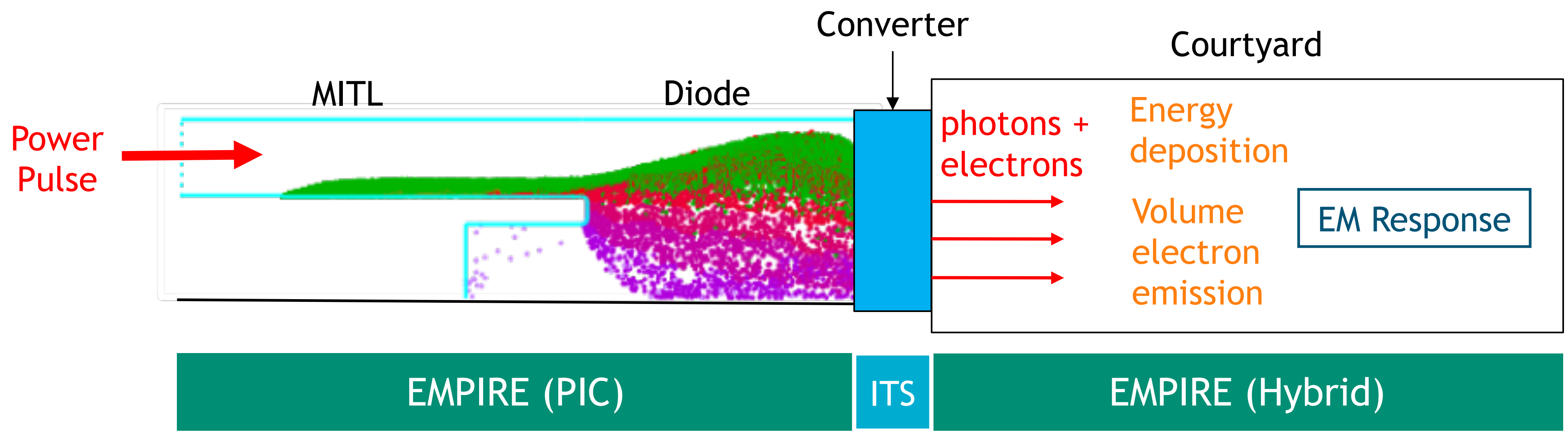
The HERMES III Pulsed Power Accelerator: MITL and courtyard





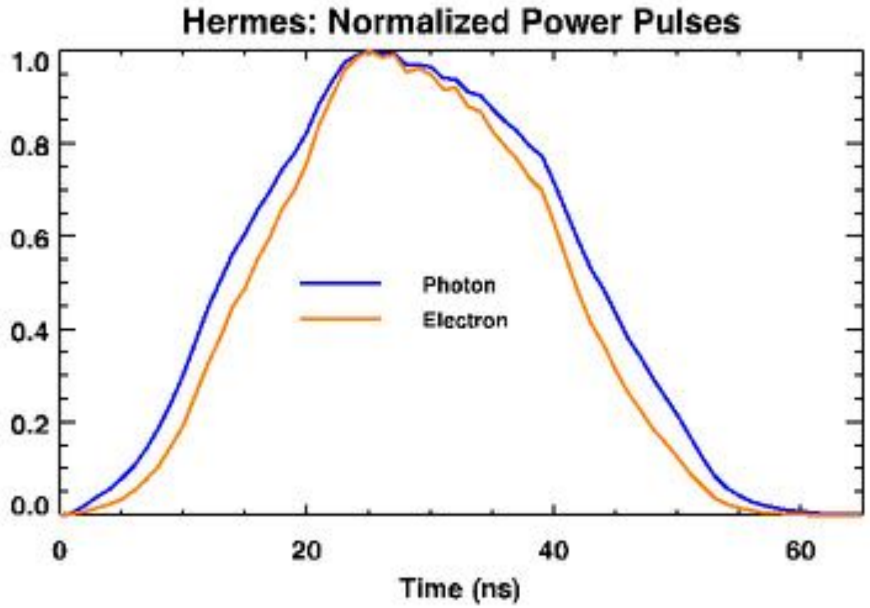


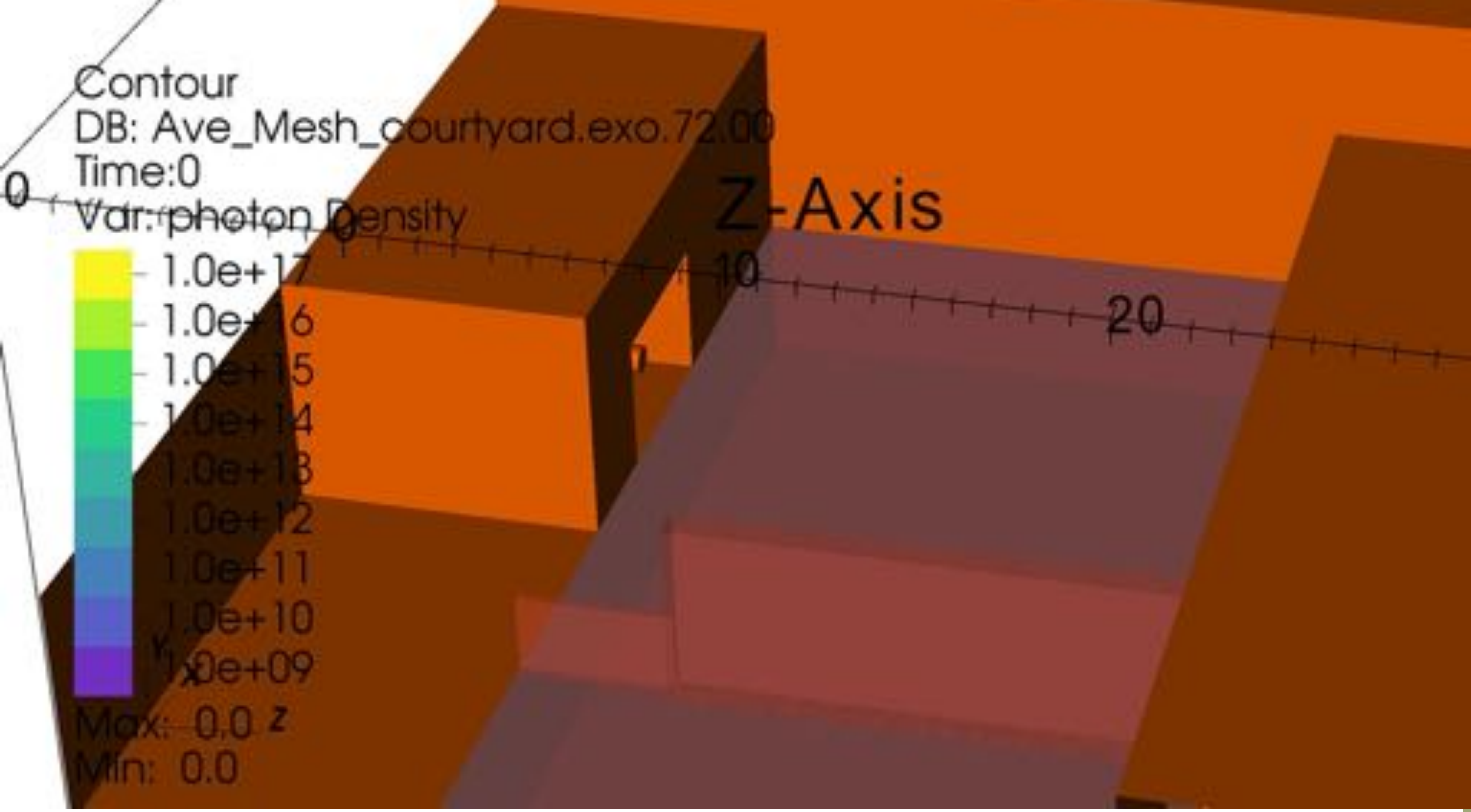
Self-consistent HERMES courtyard simulation



EMPIRE simulates the power flow in the Magnetically Insulated Transmission Line (MITL) and diode to generate the electrons incident on the vacuum side of the converter.

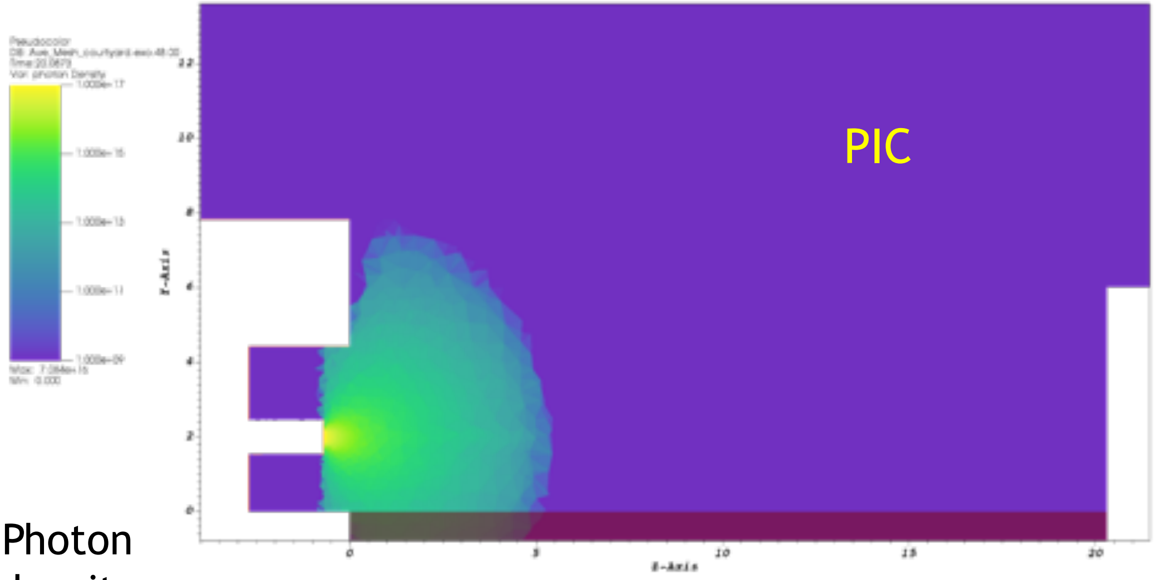
ITS simulates the radiation transport through the converter to generate the volumetric photon and electron plasma source for the courtyard simulation.



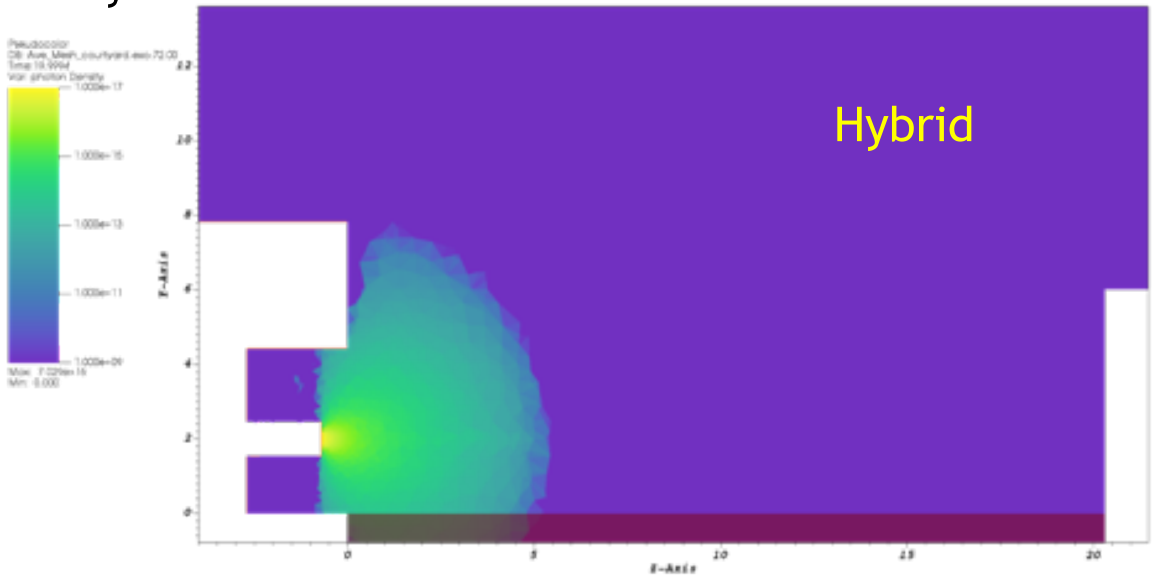


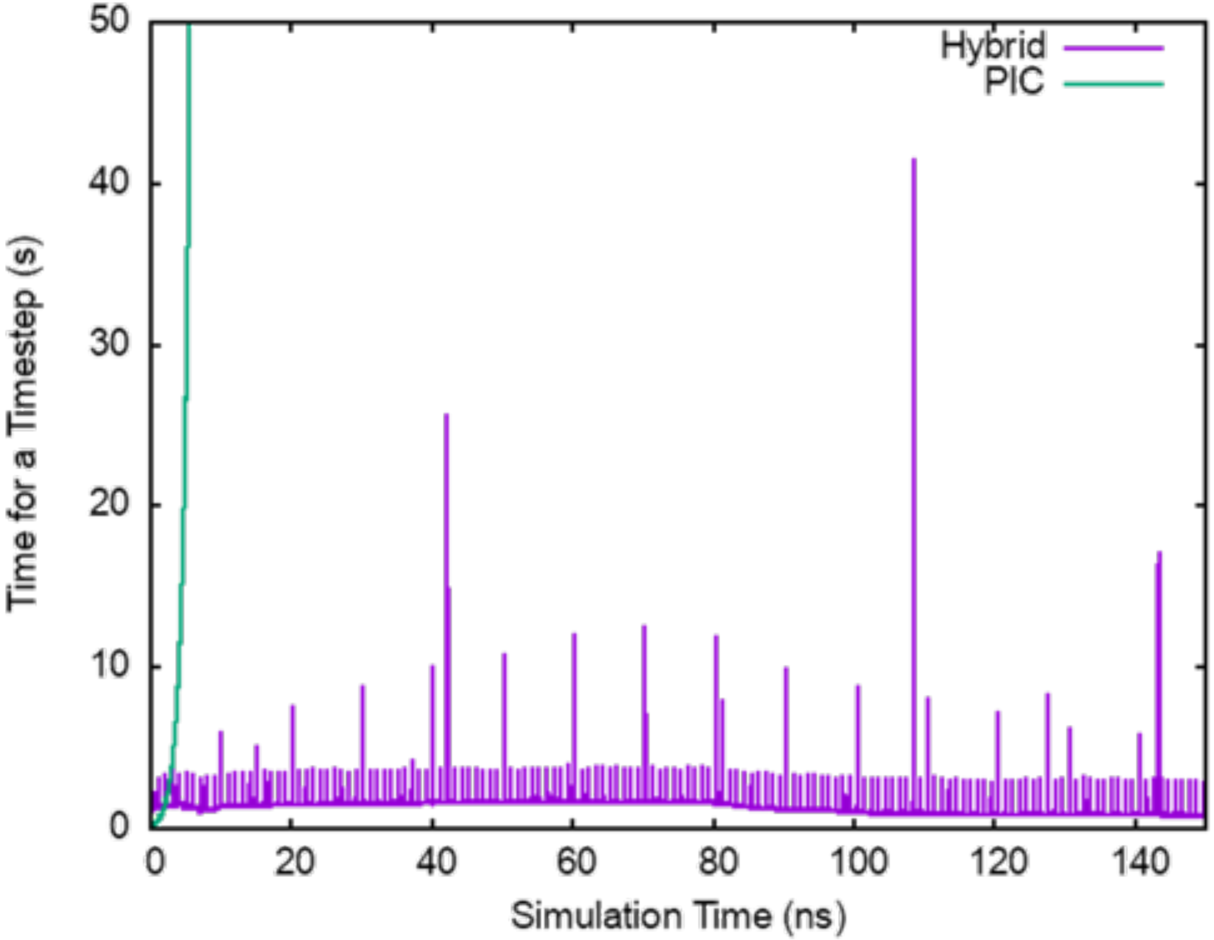
Contour
DB: Ave_Mesh_courtyard.exo.72.00
Time:0
Var: photon Density
1.0e+17
1.0e+16
1.0e+15
1.0e+14
1.0e+13
1.0e+12
1.0e+11
1.0e+10
1.0e+09
Max: 0.0
Min: 0.0
Z-Axis
0
10
20

Solution comparison between PIC and Hybrid



Photon density

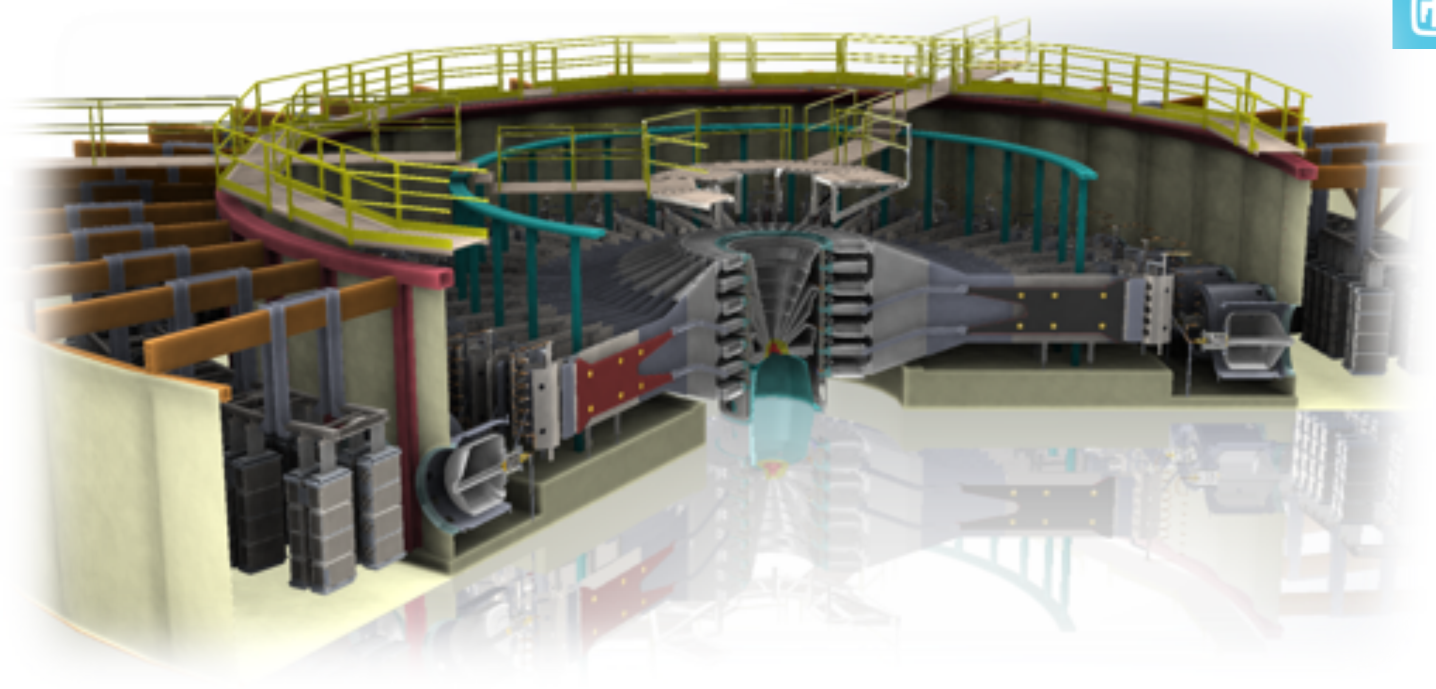


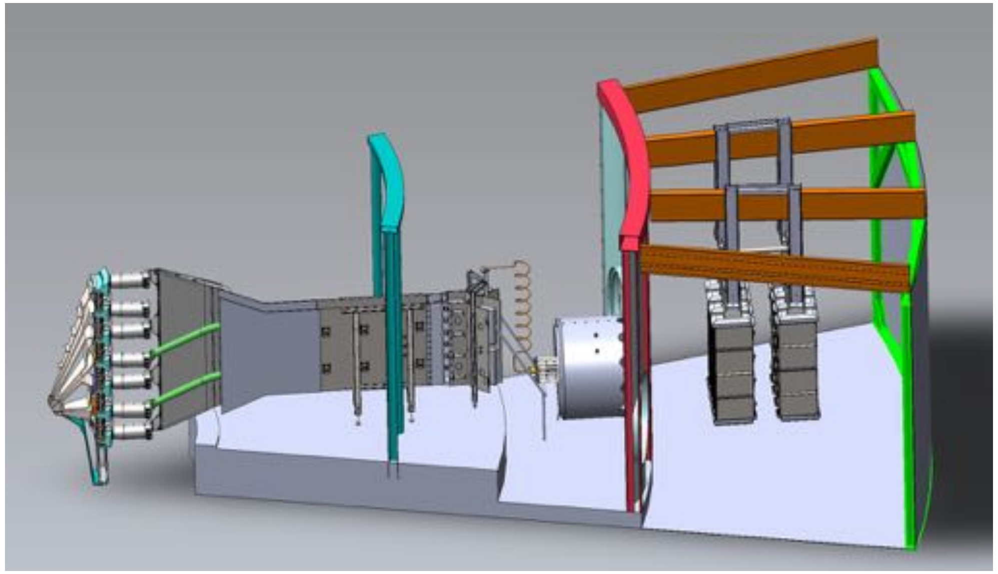


Saturn Accelerator

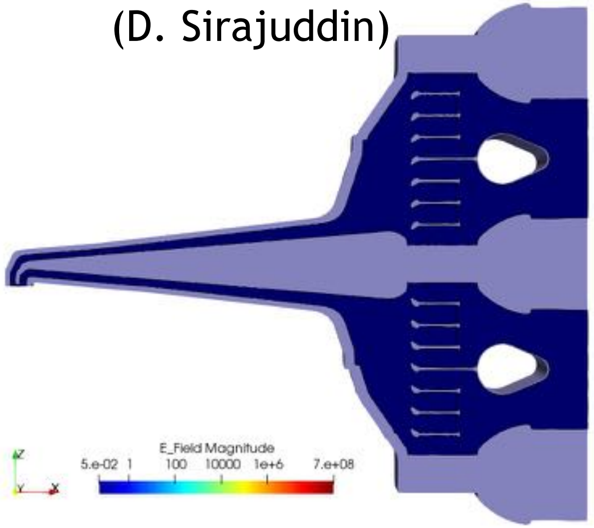
- A modular variable-spectrum X-ray source combining individual pulses from 36 individual modules to generate and convert ion/elec beams to intense X-ray output for component *testing*
  Electron beam: ~1.5 MeV, 8 – 10 MA, ~ 25 ns pulse
  X rays: 100 keV to 1.5 MeV

- SATURN is undergoing a refurbishment effort. EMPIRE simulations will advise new designs and experiments

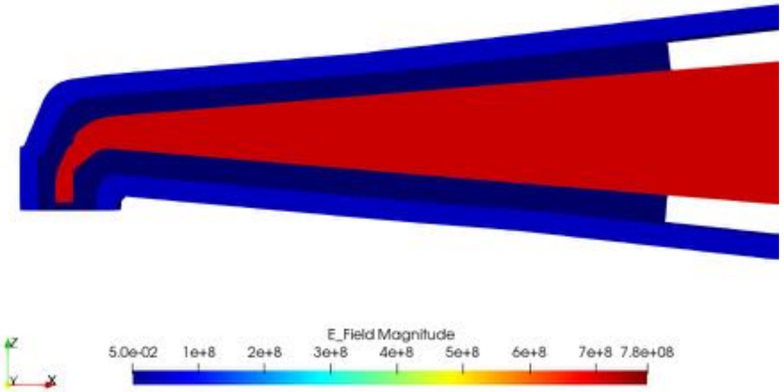




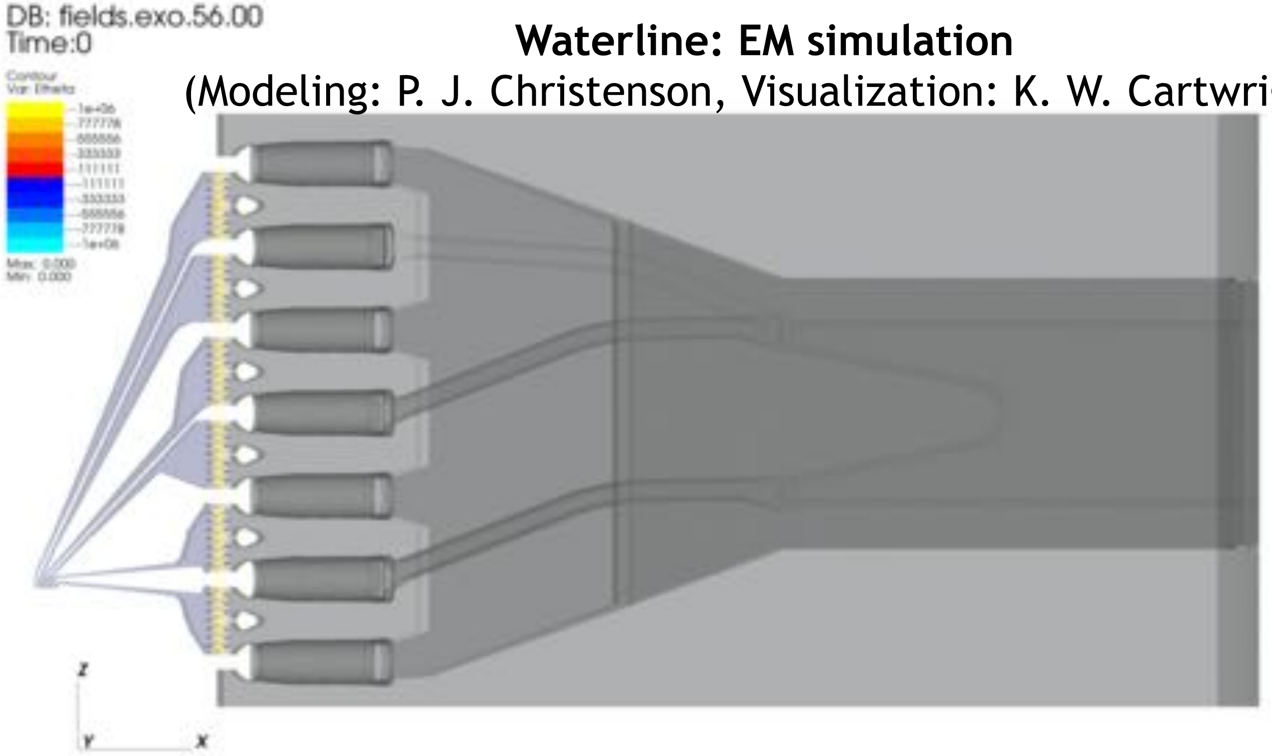
MITLs, stack, water line: EM simulation
(D. Sirajuddin)



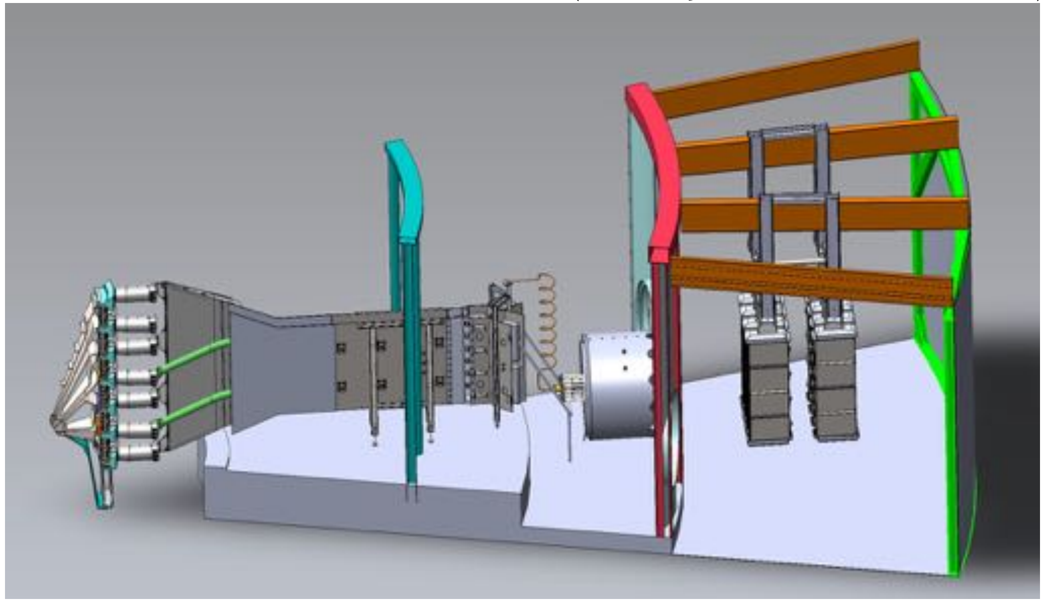
MITLs: EM simulation
(modeling: T. D. Pointon, visualization: D. Sirajuddin)



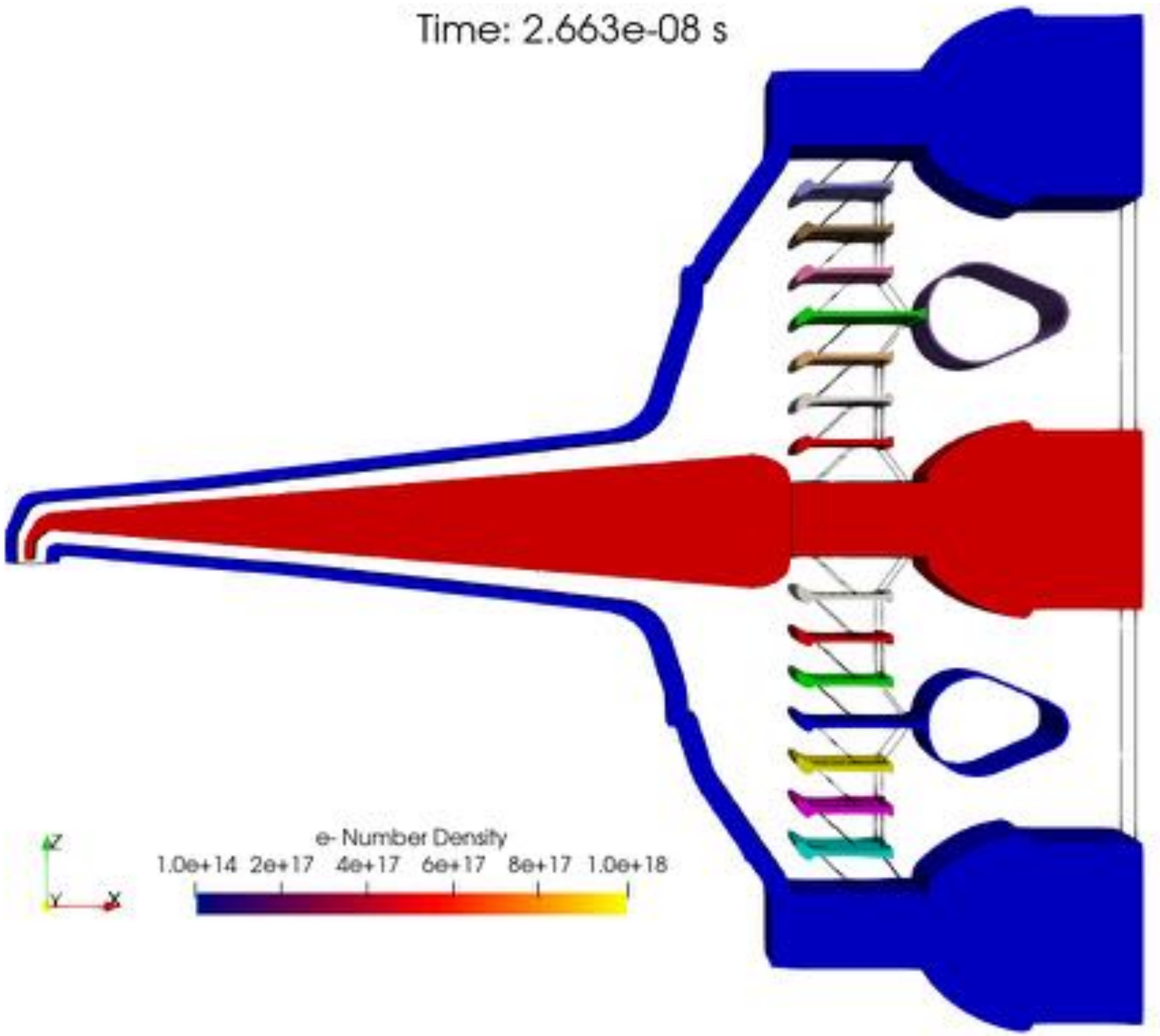
Waterline: EM simulation
(Modeling: P. J. Christenson, Visualization: K. W. Cartwright)



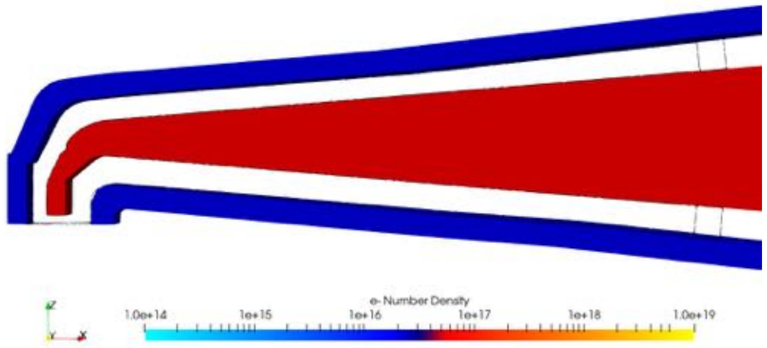
Saturn hardware (one symmetric section)



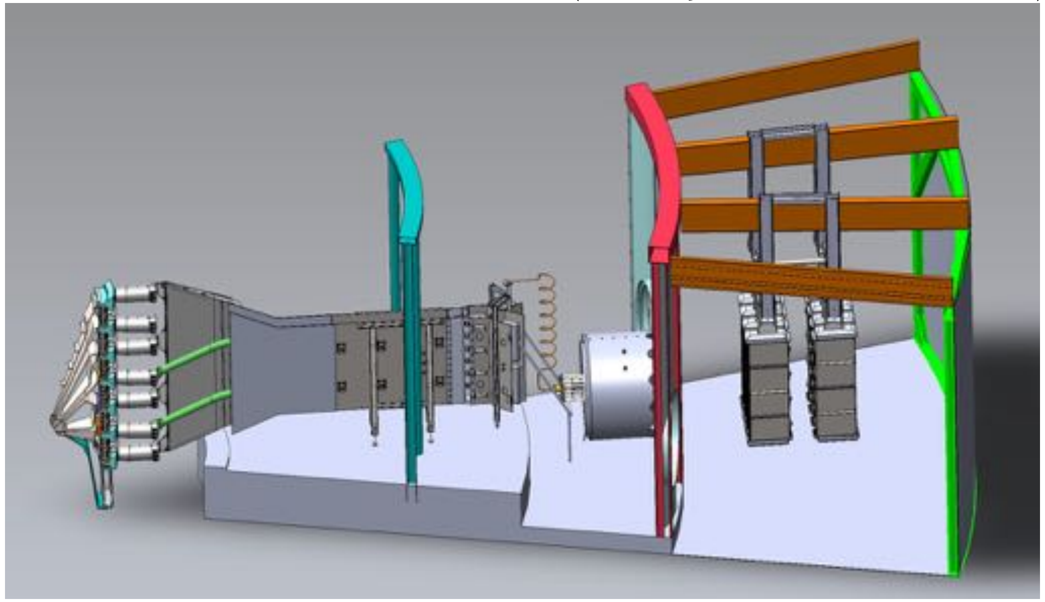
MITLs, stack, water line: e- simulation
(D. Sirajuddin)



MITLs: e- simulation
(modeling: T. D. Pointon,
visualization: D. Sirajuddin)



Saturn hardware (one symmetric section)



Note: waterline geometry not shown (no e- emission)

Trilinos Use

Trilinos Use in EMPIRE

- Directly uses 23 packages (enables 38 due to dependencies)

Data Services	Discretizations	Linear Solvers	Nonlinear
Teuchos	Shards	Belos	Sacado
Kokkos	Intrepid2	Anasazi	Thyra
KokkosKernels	Phalanx	Amesos2	NOX
Tpetra	Panzer	Ifpack2	
Zoltan2	SEACAS	Stratimikos	
Pamgen	STK	Teko	
	Percept	MueLu	

EMPIRE-EM: ElectroMagnetic Solver

$$\begin{pmatrix} \Delta t^{-1}\mathbb{I}_{\mathcal{F}} & \mathbb{K}_h \\ -\mathbb{K}_h^T \mathbb{M}_{\mathcal{F}}(\mu^{-1}) & \Delta t^{-1}\mathbb{M}_{\mathcal{E}}(\epsilon) \end{pmatrix} \begin{pmatrix} \Delta \mathbf{B} \\ \Delta \mathbf{E} \end{pmatrix} = - \begin{pmatrix} \mathbf{r}_{\mathbf{B}} \\ \mathbf{r}_{\mathbf{E}} \end{pmatrix}$$

$$\mathbf{B} \in \mathbf{H}_{\nabla\cdot}(\Omega)$$

$$\mathbf{E} \in \mathbf{H}_{\nabla\times}(\Omega)$$

Block LU Decomposition

$$\begin{pmatrix} \Delta t^{-1}\mathbb{I}_{\mathcal{F}} & \mathbb{K}_h \\ -\mathbb{K}_h^T \mathbb{M}_{\mathcal{F}}(\mu^{-1}) & \Delta t^{-1}\mathbb{M}_{\mathcal{E}}(\epsilon) \end{pmatrix} = \begin{pmatrix} \mathbb{I}_{\mathcal{F}} & 0 \\ -\Delta t \mathbb{K}^T \mathbb{M}(\mu^{-1}) & \mathbb{I}_{\mathcal{E}} \end{pmatrix} \begin{pmatrix} \Delta t^{-1}\mathbb{I}_{\mathcal{E}} & \mathbb{K}_h \\ 0 & \mathbb{S}_{\mathcal{E}} \end{pmatrix}$$

Assemble Schur Compliment as monolithic matrix

$$\mathbb{S}_{\mathcal{E}} = \Delta t^{-1}\mathbb{M}_{\mathcal{E}}(\epsilon) + \Delta t \mathbb{K}_h^T \mathbb{M}(\mu^{-1})\mathbb{K}_h$$

Solve for dE with PCG: $\mathbb{S}_{\mathcal{E}} \Delta \mathbf{E} = -\mathbf{r}_{\mathbf{E}} + \Delta t \mathbb{K}_h^T \mathbb{M}(\mu^{-1})\mathbf{r}_{\mathbf{B}}$

Explicit back solve for dB: $\Delta \mathbf{B} = -\Delta t \mathbb{K}_h \Delta \mathbf{E} - \Delta t \mathbf{r}_{\mathbf{B}}$

Meshing:
STK, Percept, SEACAS, Panzer

Data Structures:
Kokkos, KokkosKernels, Tpetra

Assembly:
Shards, Intrepid2, Panzer, Thyra

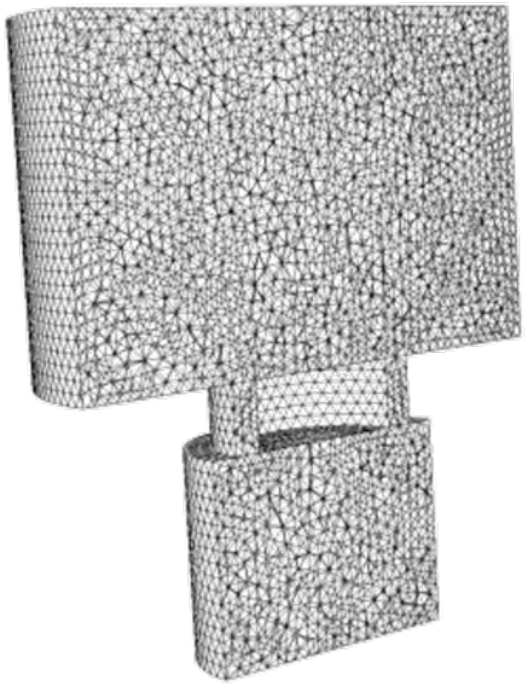
Linear Solve:
- Uses RefMaxwell AMG with Conjugate Gradient
- Chebyshev smoother
- Prec setup done once
- Belos, Teko, MueLu, Ifpack2, Amesos2, KokkosKernels, Zoltan2

Bettencourt, et. al., *EMPIRE-PIC: A Performance Portable Unstructured Particle-in-Cell Code*, 2021
Lourenco Beirao de Veiga, Konstantin Lipnikov, and Marco Manzini, *Mimetic Finite Difference Method for Elliptic Problems*.
Bochev et al., *An algebraic multigrid approach based on a compatible gauge reformulation of Maxwell's equations*, 2008.

Scalability Tests

Simple Cavity

- Simplified physics in similar configuration to B-dot experimental geometry.
- Preloaded particles.
- Run for nominal 100 time-steps to gather metrics.



Generic Cavity

- Complex geometry.
- Preloaded particles for scaling studies.
- Run for nominal 100 time-steps for scaling studies.

Mesh	Elements	Nodes	Edges	Particles
R0	337k	60.4k	406k	16M
R1	2.68M	462k	3.18M	128M
R2	20.7M	3.51M	24.4M	1.0B
R3	166M	27.9M	195M	8.2B
R4	1.33B	223M	1.56B	66B

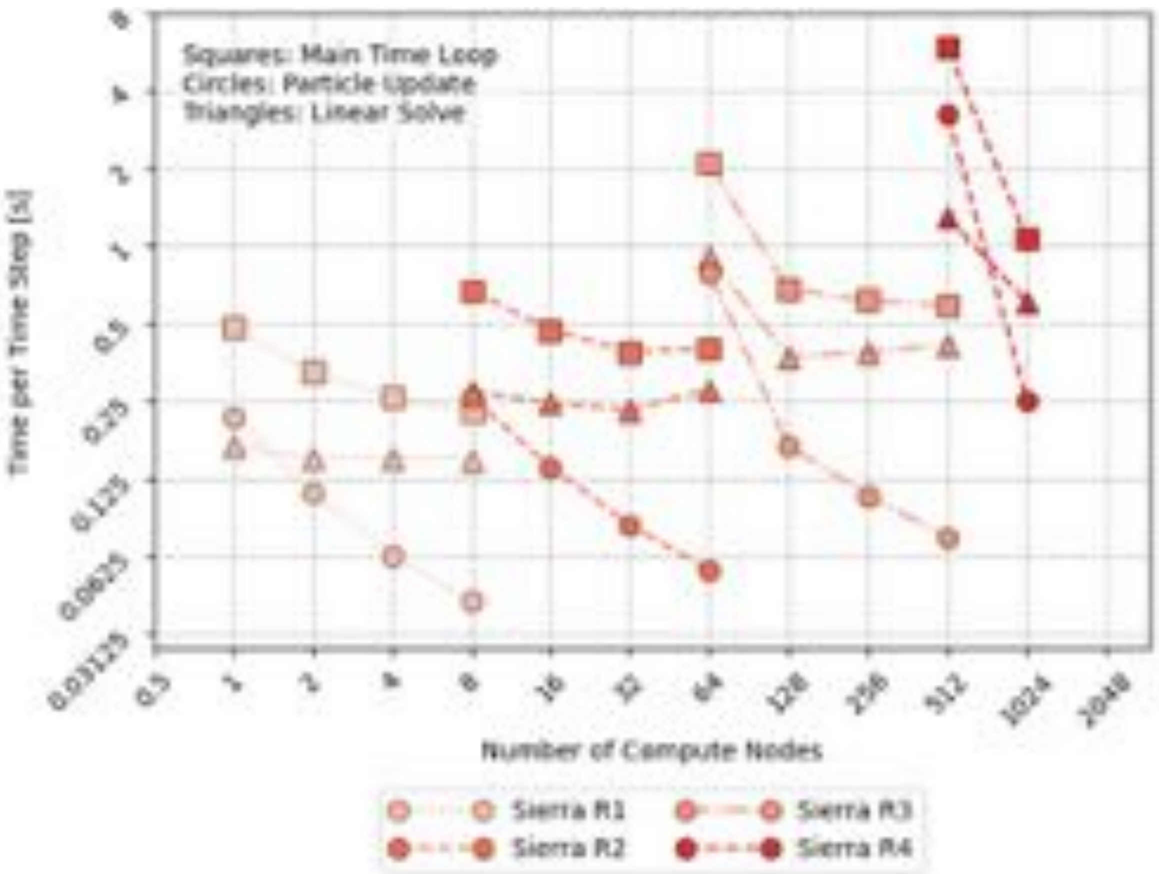
Mesh	Elements	Nodes	Edges	Particles*
R0	3.7M	660k	4.4M	360M
R1	25M	4.4M	30M	2.4B
R2	200M	32M	240M	19B
R3	1.6B	270M	1.9B	160B

Platforms

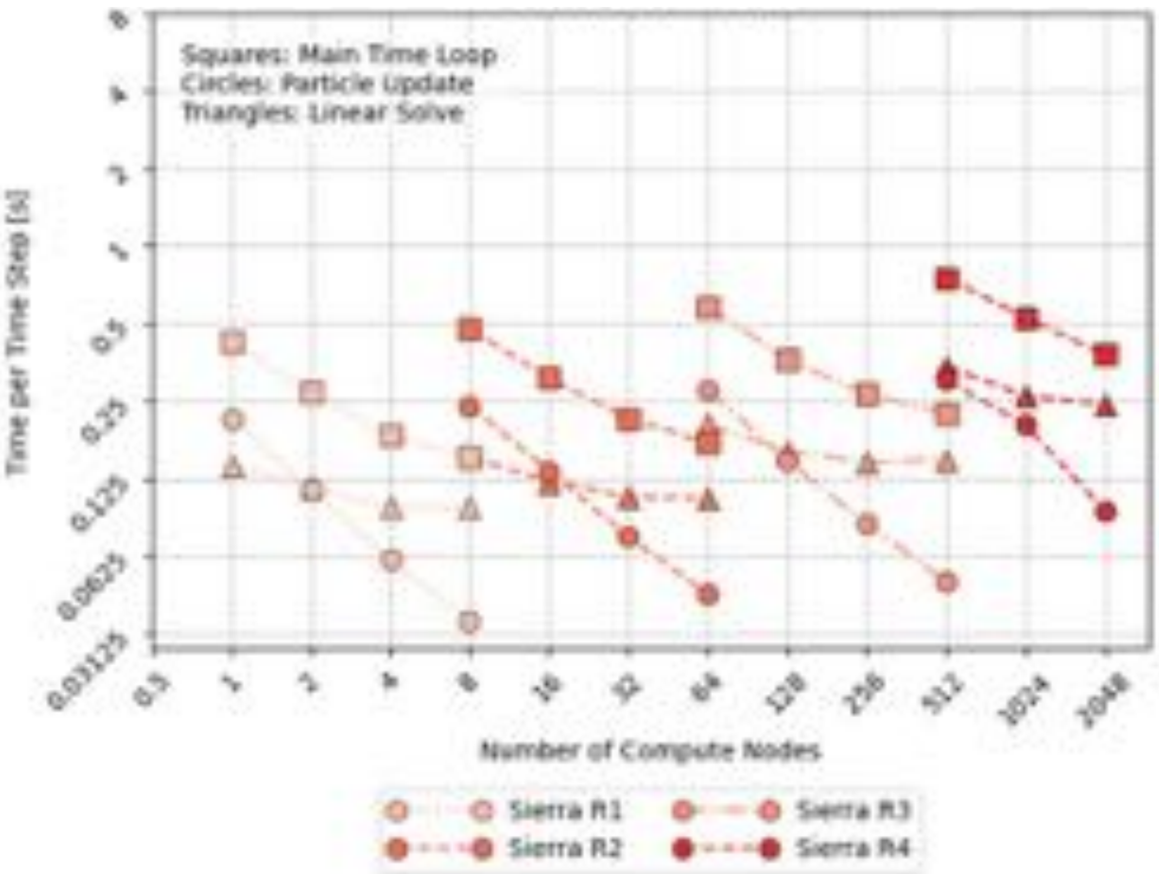
Machine	Nodes	Processor	Accelerator
Trinity (Haswell)	9436	2 × Intel Xeon E5-2698v3	-
Trinity (KNL)	9984	1 × Intel Xeon Phi 7250	-
Astra	2592	2 × Cavium Thunder-X2 CN9975	-
Sierra	4340	2 × IBM POWER9 22C	4 × NVIDIA V100

ATS-2 performance improvements (simple cavity reference problem)

December 2019 results



August 2020 results



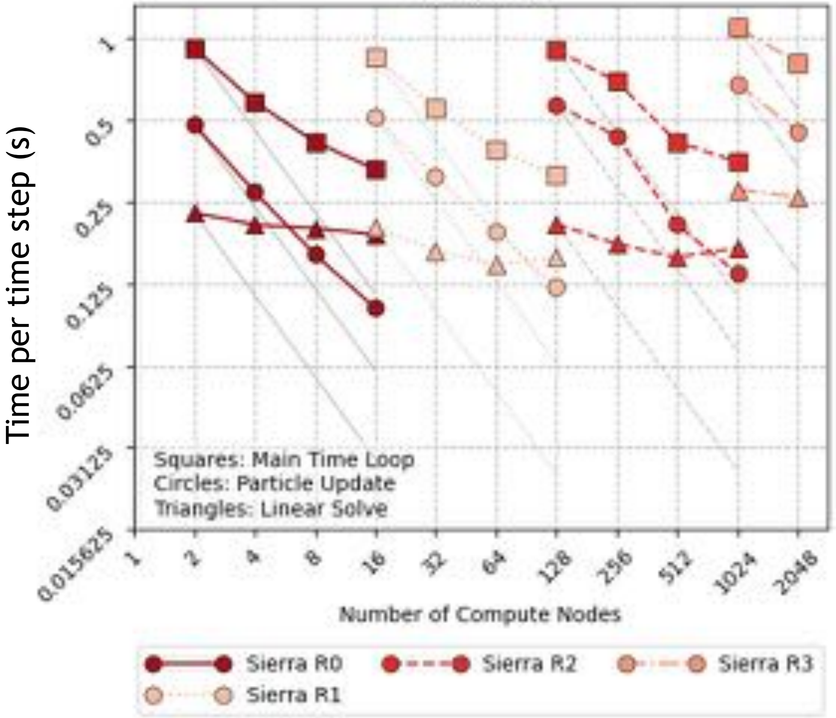
- Linear solver did not weak scale
- Particle update showed strong scaling issues

- Solver performance and scaling improved
- Particle performance and scaling improved

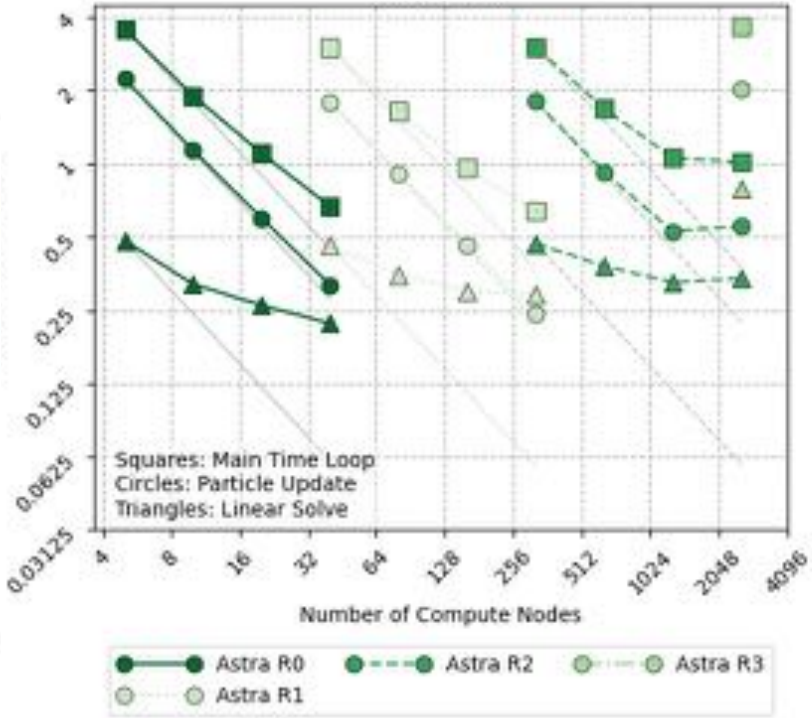
Performance results for the Generic Cavity

Single platform scaling results

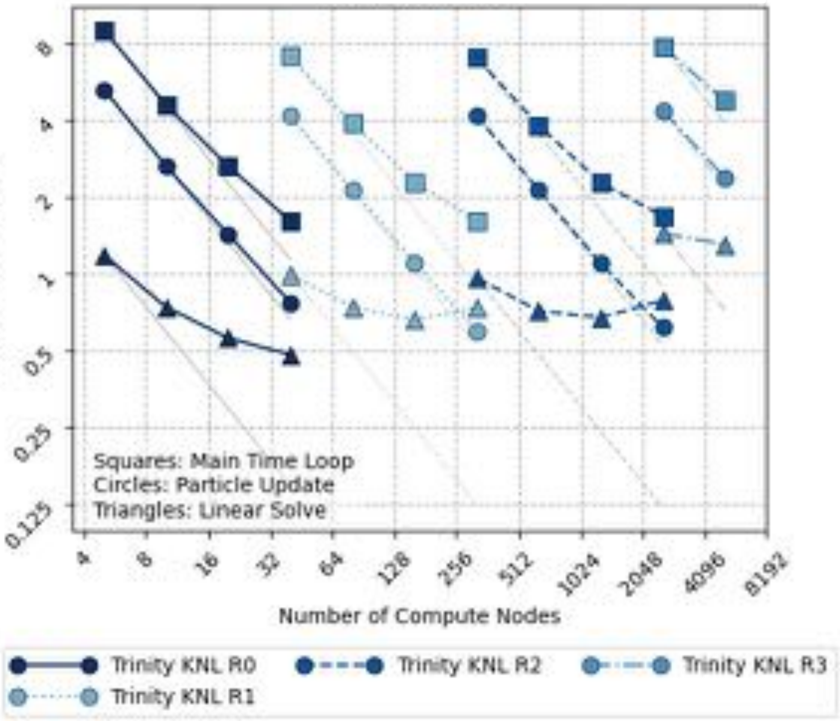
Sierra
4 GPU/socket; 1 MPI/GPU
Up to 2048 nodes (47% of total)



Astra
2 MPI/socket; 9 threads/MPI
Up to 2560 nodes (99% of total)

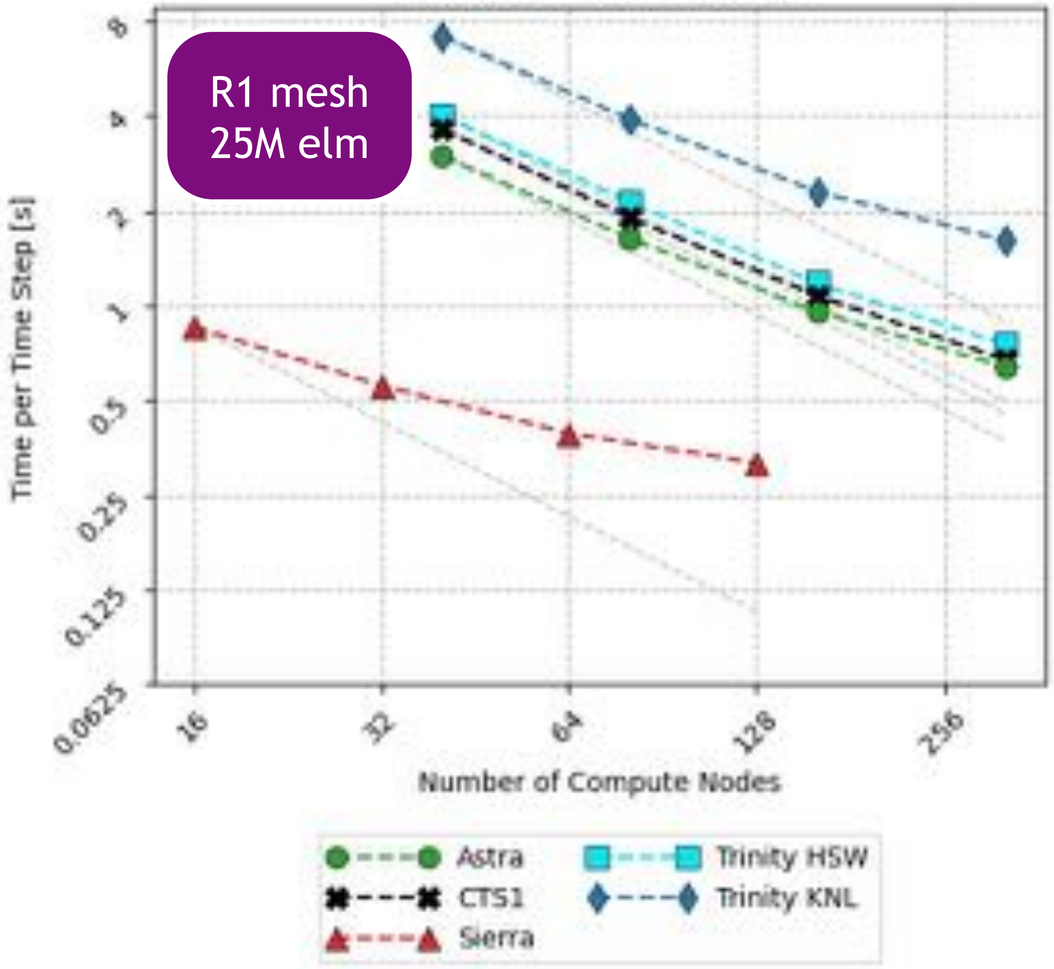


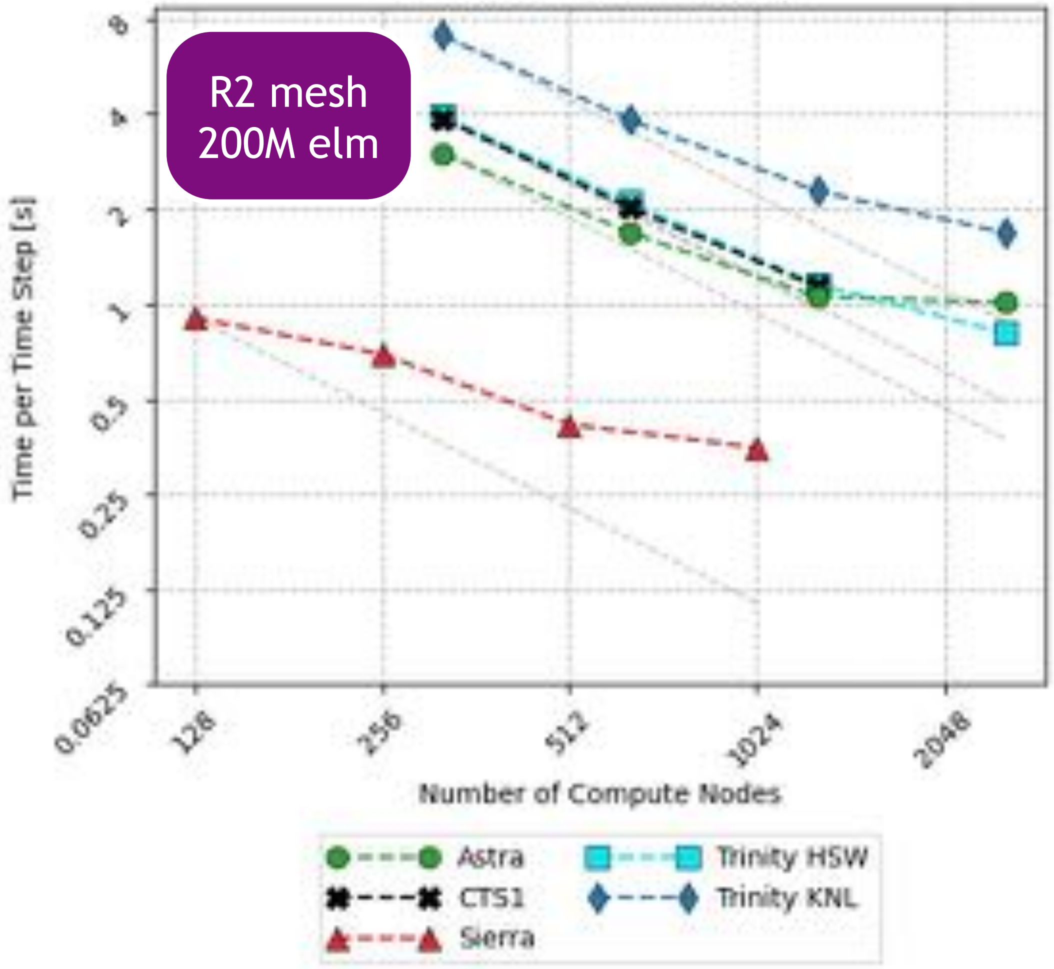
Trinity/KNL
4 MPI/socket; 16 threads/MPI
Up to 5120 nodes (52% of total)



Performance results for the Generic Cavity

Cross-platform strong scaling results



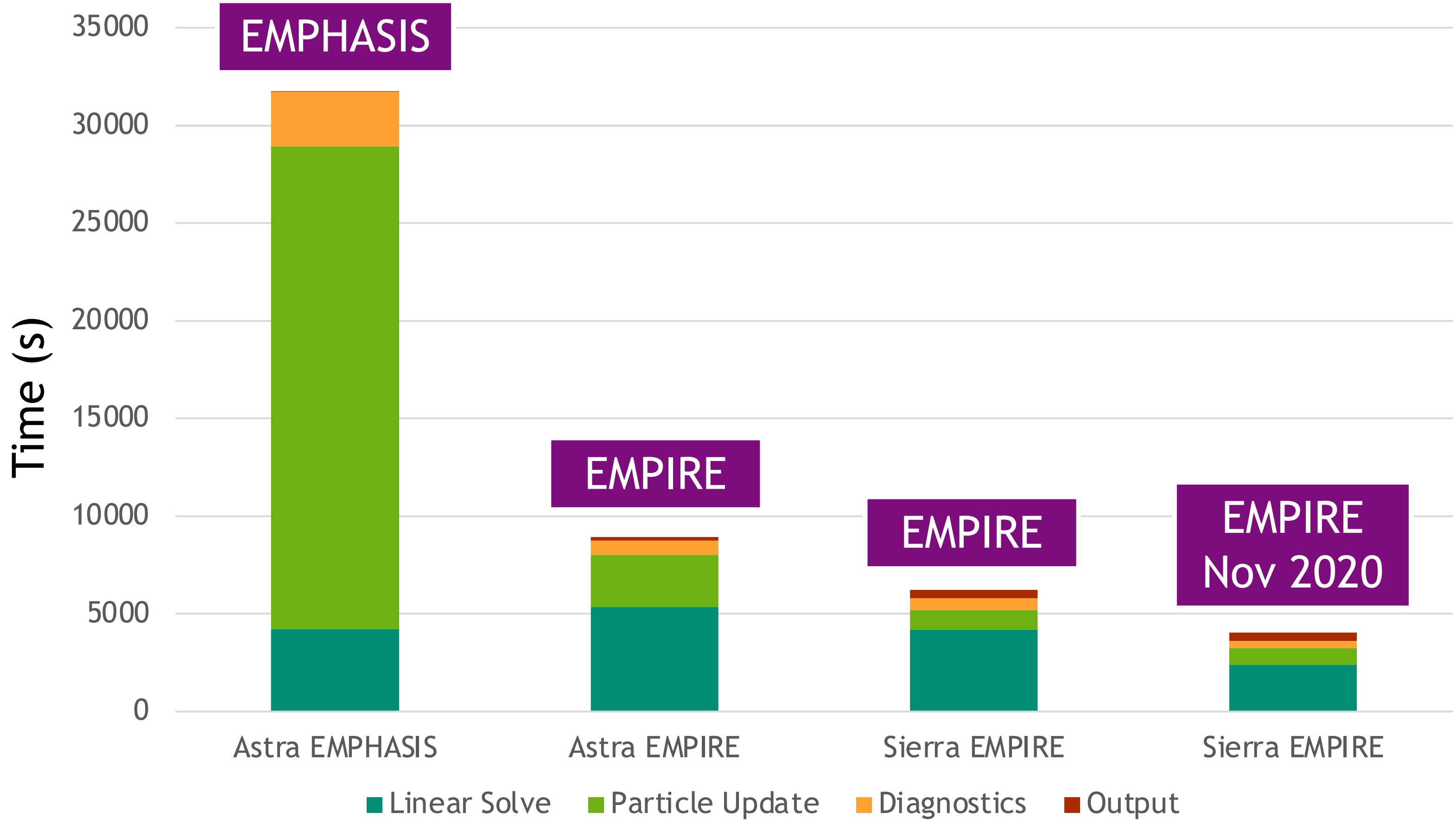


GC performance comparison to the legacy code

Timing results

EMPHASIS uses legacy 32-bit solver stack

EMPIRE uses modern 64-bit solver stack



40 nodes on Astra

16 nodes on Sierra

EMPIRE-PIC Optimized Data Structures

EMPIRE particle update is drastically faster than legacy code despite using the same algorithm

- Optimized data structures improve single node performance

Data models are critical

- AoS and SoA used to be the question people would argue over – EMPIRE uses SoSoAoS

SoSoAoS - ParticleContainer core structure (holds Kokkos::DynamicView of data)

- Enables constant time insertion
- Enables continuous memory access
- Allows code to access just one variable at a time (position, velocity, …)

Particles are marked for deletion and then removed later

- Contiguous memory access for better performance

Atomic operations are available for parallel lock free addition and deletion of particles

Memory pools are used for all temporaries and recycled – allocations are slow

PIC performance improvements in EMPIRE

Linear solvers have been the primary focus until FY20

Reduced atomic load

Using scatter views and parallel scans

Reduced parallel launches

Fused kernels

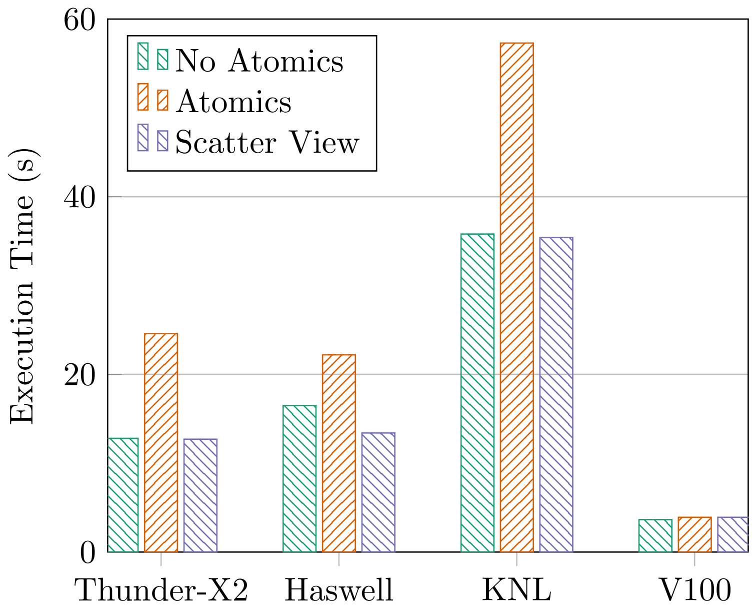
Developed custom particle sort

Replaced Thrust's sort with custom algorithm

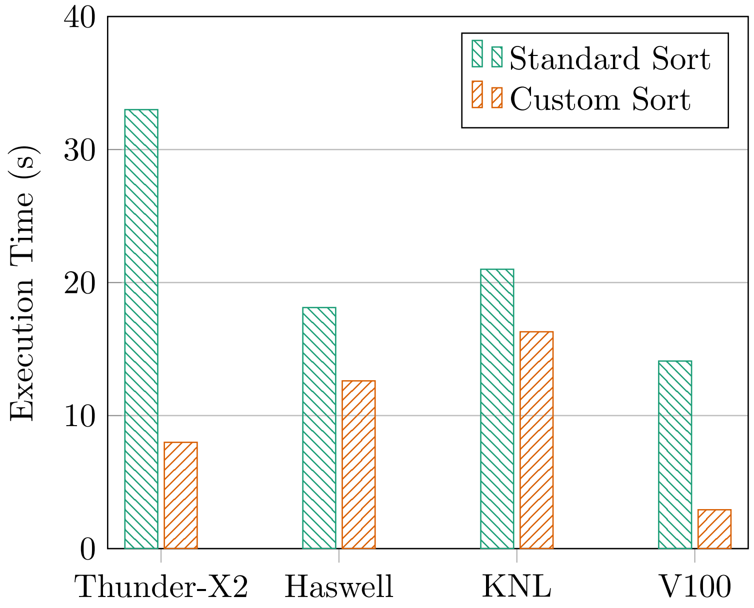
Checkpoint/Restart

Built off VT component for migrating data

Atomics

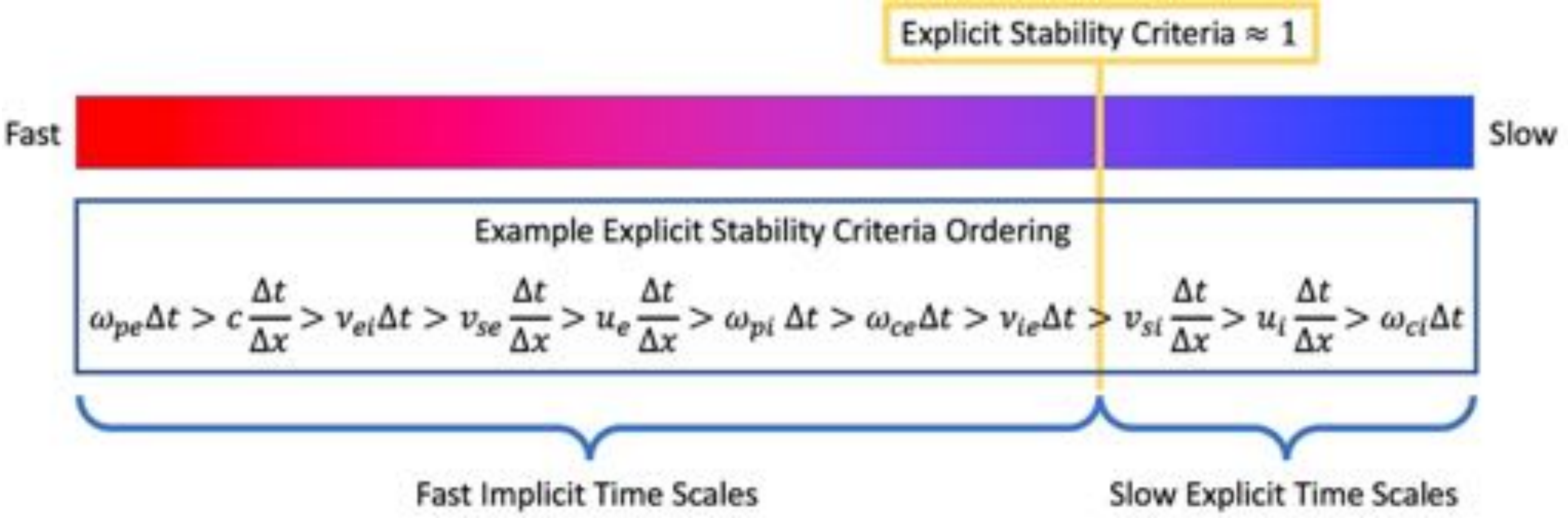


Sort



Fluid/Hybrid Solver

- Discontinuous Galerkin
  - KokkosKernels to invert local mass matrices
- Assembly uses Sacado, Phalanx, Intrepid2, Panzer
  - Automatic Differentiation
- Fluid solver uses an IMEX RK scheme
- Optionally coupled to EMPIRE-EM and EMPIRE-PIC
  - Nonlinear iteration is accelerated using Anderson acceleration (NOX)



Density	$\frac{\partial \rho_a}{\partial t} + \nabla\cdot(\rho_a \mathbf{u}_a) = \sum_{b\neq a}(n_a\rho_b\nu^+_{ab} - n_b\rho_a\nu^-_{ab})$
Momentum	$\frac{\partial(\rho_a\mathbf{u}_a)}{\partial t} + \nabla\cdot(\rho_a\mathbf{u}_a\otimes\mathbf{u}_a + p_a I + \Pi_a) = q_a n_a(\mathbf{E}+\mathbf{u}_a\times\mathbf{B}) - \sum_{b\neq a}[\rho_a(\mathbf{u}_a-\mathbf{u}_b)n_b\nu^M_{ab} + \rho_b\mathbf{u}_b n_a \nu^+_{ab} - \rho_a\mathbf{u}_a n_b\nu^-_{ab}]$
Energy	$\frac{\partial\varepsilon_a}{\partial t} + \nabla\cdot((\varepsilon_a+p_a)\mathbf{u}_a + \Pi_a\cdot\mathbf{u}_a + \mathbf{h}_a) = q_a n_a\mathbf{u}_a\cdot\mathbf{E} + Q^{src}_a - \sum_{b\neq a}\left[(T_a-T_b)k\nu^E_{ab} - \rho_a\mathbf{u}_a\cdot(\mathbf{u}_a-\mathbf{u}_b)n_b\nu^M_{ab} - n_a\nu^+_{ab}\varepsilon_b + n_b\nu^-_{ab}\varepsilon_a\right]$

Explicit Hydrodynamics

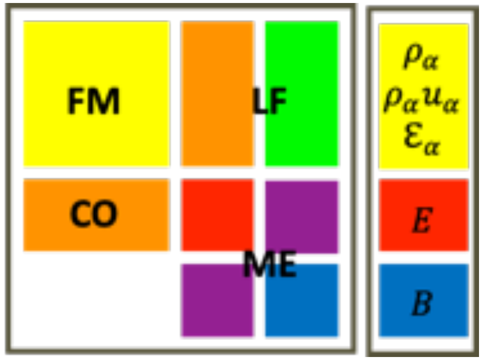
Implicit EM, EM sources, sources for species interactions

Fluid/Hybrid Solver

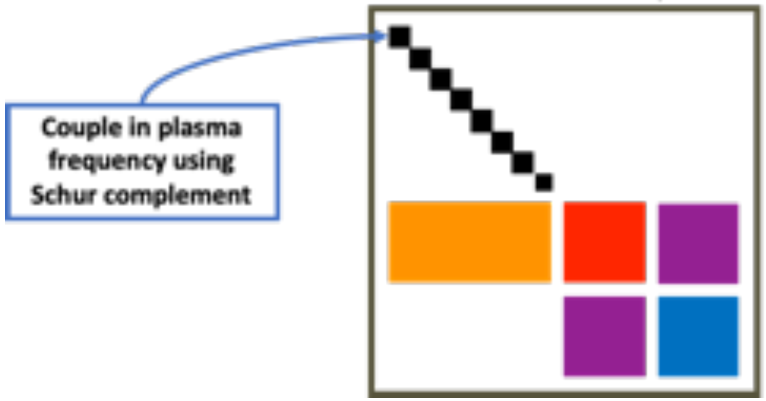
- For an ideal Newton-based solve, we would couple with a Schur complement preconditioner.
- We instead choose the discretization and solution method to avoid a global solve of the fully coupled system
  - Nonlinear solve with Anderson acceleration
  - Discontinuous Galerkin (DG) for the Fluid equations
  - IMEX time integration with **explicit hydrodynamics**
- DG and explicit hydro
  - Decoupled local block diagonal matrices
  - Blocks are solved in parallel using KokkosKernels
  - Leverages fast Maxwell solver described earlier
- Global nonlinear solve uses Anderson acceleration
  - Solve a Quasi-Newton linear system in the Picard mapping: x = g(x)
  - Account for coupling via by adding a local approximation to the Lorentz force operator to the block diagonals.

$$J_k \Delta x_k = -f(x_k)$$
$$x_k = x_k + \Delta x_k$$

$J_k v =$

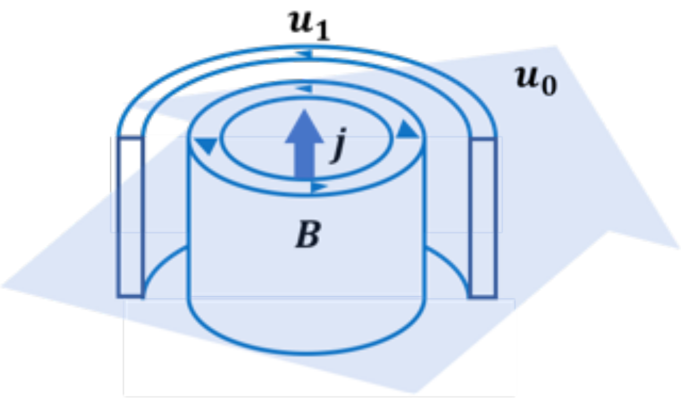


Block GS with Schur Complement

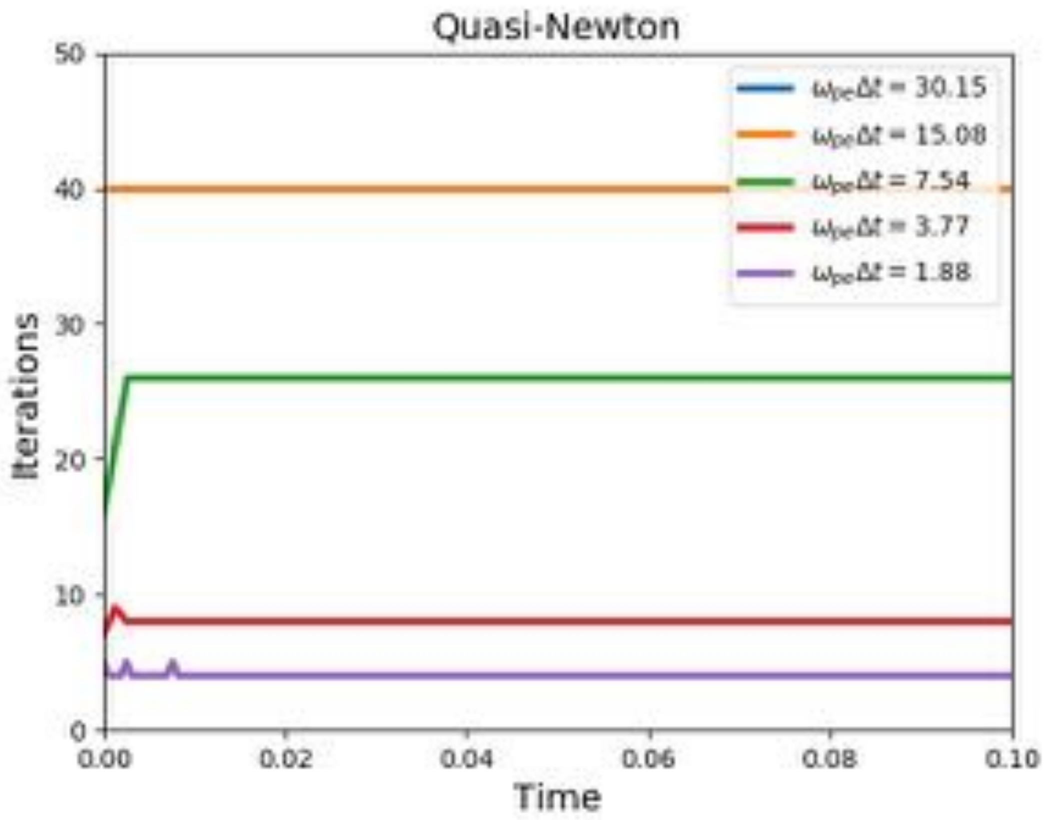


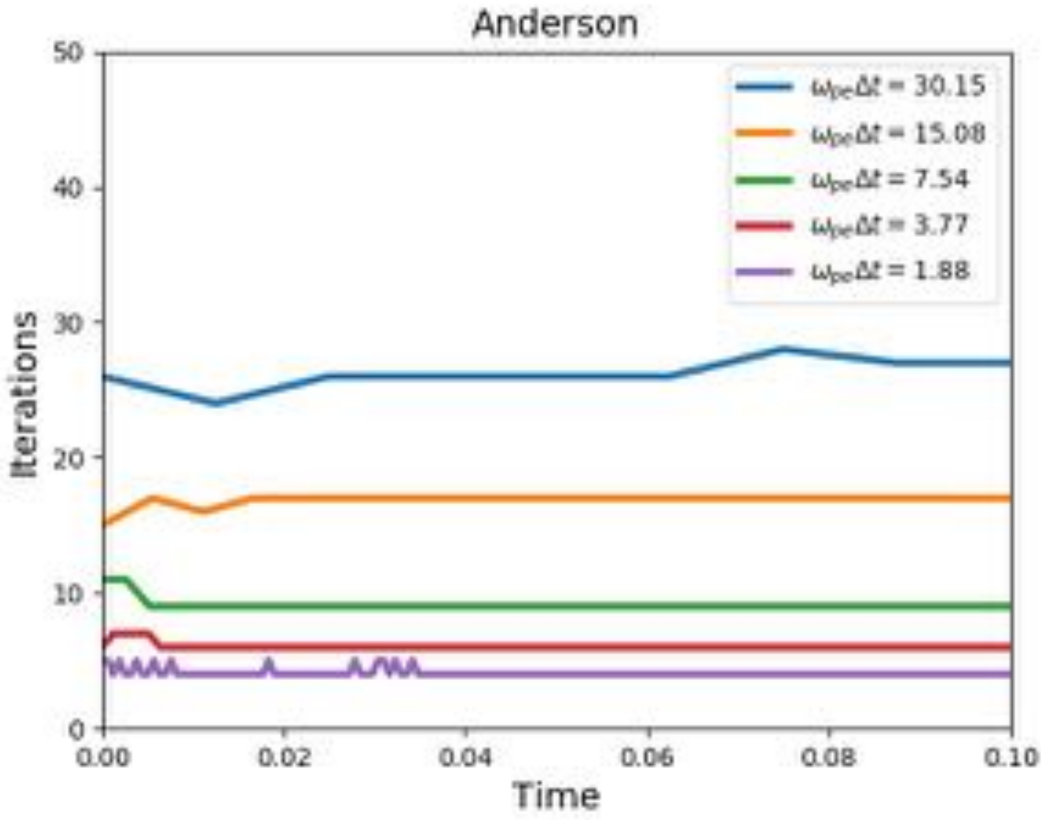
Fluid Linear Solve:
- Block diagonal solver
- KokkosKernels: KokkosBatched_LU and KokkosBatched_Trsm
- NOX

Hybrid Two Fluid Plasma Vortex Verification Problem



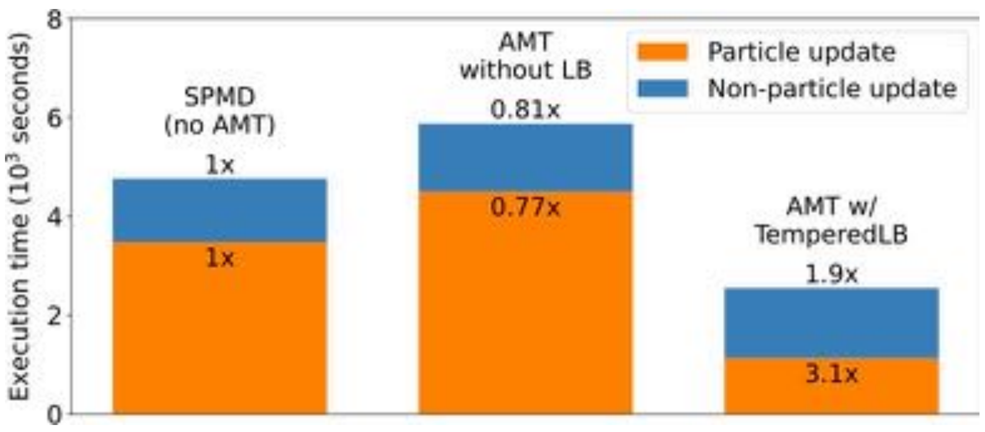
- Two fluid plasma vortex in MHD limit
- IMEX time discretization, DG fluid discretization, CG Maxwell discretization
- Using Schur-Complement in all simulations



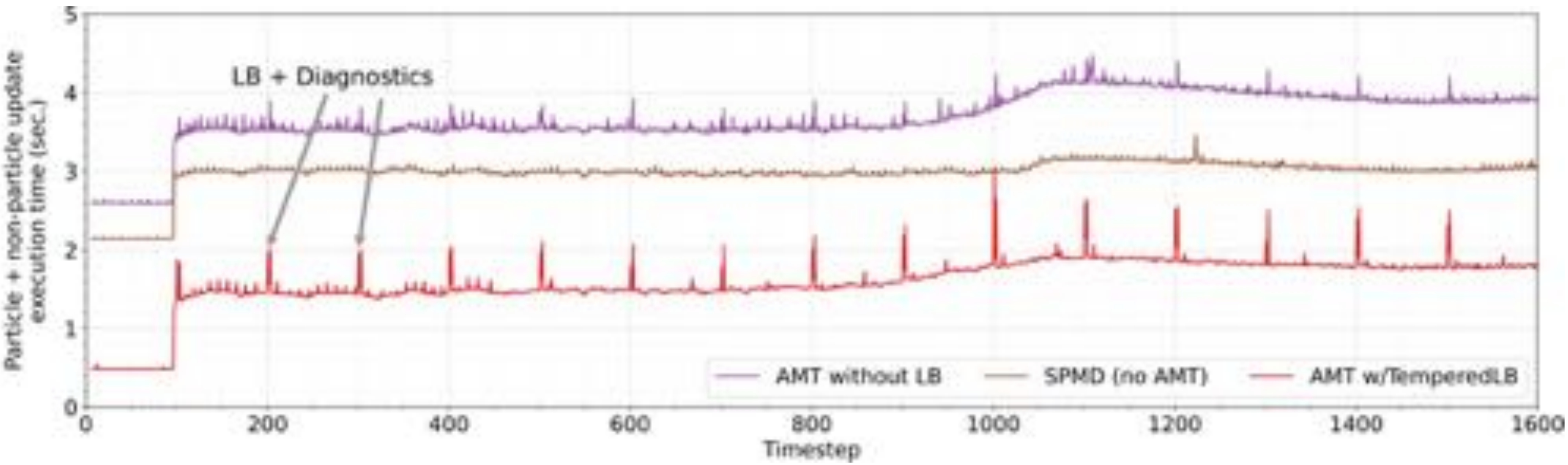


Dynamic task-based load balancing using DARMA

- PIC can spatially concentrate particles, causing load imbalance
- DARMA/vt (virtual transport): C++ asynchronous tasking runtime
- Includes suite of highly scalable, fully distributed load balancers
- Trilinos and Darma coexist, switching between bulk sync and tasking comm layers



(a)



PI: Jonathan Lifflander, Analyst: Brandon Medina

Summary

EMPIRE has demonstrated performance portability

EMPIRE runs on all target platforms	Achieved with **no** platform-dependent code
EMPIRE strong and weak scales	Iterative advances yielded significant improvements
Trilinos is heavily leveraged	Solvers and discretization tools critical to success
Strong partnership with Trilinos	Multiple embedded Trilinos developers on the team

Next steps regarding Trilinos components:

- UVM removal, porting to AMD/HIP
- Memory use improvements
- Algorithmic optimization for more physics