SAND2018-???? C

SAND2018-9685C

Developing a Hybrid Multi-fluid/PIC Plasma Capability Using Components

Roger Pawlowski*, Edward Phillips**, John Shadid*, Paul Lin*, Eric Cyr*, Sidafa Conde*, Matt Bettencourt**, Sean Miller*, Eric Phipps*, Stephen Bond*, and Christian Trott*

*Center for Computing Research, Sandia National Laboratories

**Radiation and Electrical Sciences Center, Sandia National Laboratories

Smoky Mountains Computational Sciences and Engineering Conference

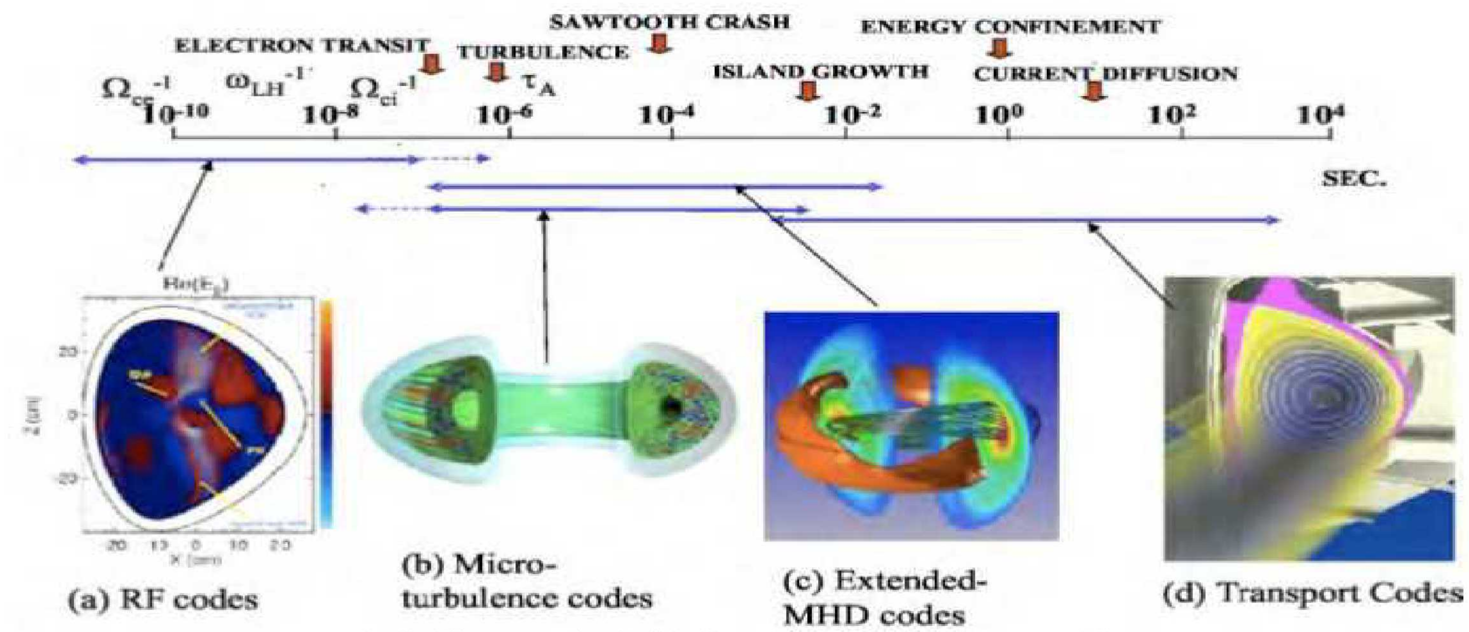
August 28-30, 2018 • Gatlinburg, TN

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and 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.

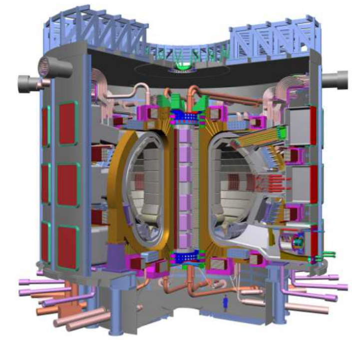
Outline

- Motivation
- Models and Solvers
- Components/Software
- Results
- Perspective on the use of Components
- Summary

ITER: Understanding and controlling instabilities in plasma confinement is critical.



(a) Time scales in fusion plasmas (FSP report)



ITER

Strong external magnetics field used to:

- Confine the hot plasma and keep it from striking wall,
- Attempt is to achieve temperature of about 100M deg K (6x Sun temp.),
- Energy confinement times O(1 – 10) seconds is desired.

Plasma instabilities can cause break of confinement, huge energy loss, and discharge very large electrical currents (~20MA) into structure. ITER can sustain only a limited number of disruptions, O(1 – 5) significant instabilities.

DOE Office of Science ASCR/OFES Reports:
Fusion Simulation Project Workshop Report, 2007, Integrated System Modeling Workshop 2015

Hybrid Code Design

- Goal: Develop a hybrid code with 5-Moment Fluid Model + Electromagnetics coupled to Particle-In-Cell (PIC) Model.

- Requirements
  - **Performance Portable** on emerging HPC architectures: HSW, KNL, CUDA, …
  - **Component-based**, heavily leverage Exascale Computing Project (ECP) software stack: Trilinos, Kokkos, …

- Two Scalable Fluid solvers
  - **Research code:** (Drekar):
    - CGFEM, FTC, HDG
    - Implicit, IMEX, Explicit
    - Turbulent CFD, turbulent MHD, Multispecies Plasma, …
  - **Production code**: DGFEM, Explicit, IMEX

- Scalable EM-PIC solver
  - **Production code:** Implicit and explicit

Trilinos

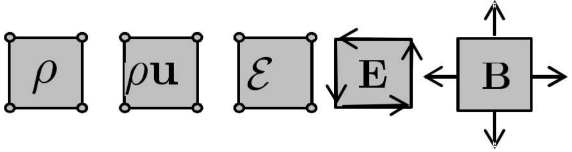
- The Trilinos Project is an effort to develop algorithms and enabling technologies within an object-oriented software framework for the solution of large-scale, complex multi-physics engineering and scientific problems.
- Not monolithic; ~60 separate packages
- Each package
  - Has its own development team & management
  - May or may not depend on other Trilinos packages
  - May even have a different license or release status
    - Most BSD; some LGPL
    - Some not publicly released yet (e.g., "pre-copyright")
  - Benefits from Trilinos' build, test, & release infrastructure (SQA)
- Common build & test framework: TriBITS
  - Lets packages express their dependencies on
    - Other packages
    - Third-party libraries (e.g., HDF5, BLAS, SuperLU, …)

Approach: More like LEGOs than a unified framework!

Trilinos Package Summary

	Objective	Package(s)
Discretizations	Meshing & Discretizations	Intrepid, Pamgen, Sundance, Mesquite, STKMesh, Panzer
	Time Integration	Rythmos
Methods	Automatic Differentiation	Sacado
	Mortar Methods	Moertel
Services	Linear algebra objects	Epetra, Tpetra
	Interfaces	Xpetra, Thyra, Stratimikos, Piro, …
	Load Balancing	Zoltan, Isorropia, Zoltan2
	"Skins"	PyTrilinos, WebTrilinos, ForTrilinos, CTrilinos
	Utilities, I/O, thread API	Teuchos, EpetraExt, Kokkos, Phalanx, Trios, …
Solvers	Iterative linear solvers	AztecOO, Belos, Komplex
	Direct sparse linear solvers	Amesos, Amesos2, ShyLU
	Direct dense linear solvers	Epetra, Teuchos, Pliris
	Iterative eigenvalue solvers	Anasazi
	Incomplete factorizations	AztecOO, Ifpack, Ifpack2
	Multilevel preconditioners	ML, CLAPS, MueLu
	Block preconditioners	Meros, Teko
	Nonlinear solvers	NOX, LOCA
	Optimization	MOOCHO, Aristos, TriKota, GlobiPack, OptiPack, ROL
	Stochastic PDEs	Stokhos

Multi-fluid 5-Moment Plasma System Model



Density	$\frac{\partial \rho_a}{\partial t} + \nabla \cdot (\rho_a \mathbf{u}_a) = \sum_{b \neq a} (n_a \rho_b \bar{\nu}^+_{ab} - n_b \rho_a \bar{\nu}^-_{ab})$
Momentum	$\frac{\partial (\rho_a \mathbf{u}_a)}{\partial t} + \nabla \cdot (\rho_a \mathbf{u}_a \otimes \mathbf{u}_a + p_a I + \Pi_a) = q_a n_a (\mathbf{E} + \mathbf{u}_a \times \mathbf{B})$ $- \sum_{b \neq a} [\rho_a (\mathbf{u}_a - \mathbf{u}_b) n_b \bar{\nu}^M_{ab} + \rho_b \mathbf{u}_b n_a \bar{\nu}^+_{ab} - \rho_a \mathbf{u}_a n_b \bar{\nu}^-_{ab}]$
Energy	$\frac{\partial \varepsilon_a}{\partial t} + \nabla \cdot ((\varepsilon_a + p_a) \mathbf{u}_a + \Pi_a \cdot \mathbf{u}_a + \mathbf{h}_a) = q_a n_a \mathbf{u}_a \cdot \mathbf{E} + Q^{src}_a$ $- \sum_{b \neq a} \left[(T_a - T_b) k \bar{\nu}^E_{ab} - \rho_a \mathbf{u}_a \cdot (\mathbf{u}_a - \mathbf{u}_b) n_b \bar{\nu}^M_{ab} - n_a \bar{\nu}^+_{ab} \varepsilon_b + n_b \bar{\nu}^-_{ab} \varepsilon_a \right]$
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	$\frac{1}{c^2} \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{B} + \mu_0 \mathbf{J} = \mathbf{0}$ $\qquad \nabla \cdot \mathbf{E} = \frac{q}{\epsilon_0}$ $\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = \mathbf{0}$ $\qquad \nabla \cdot \mathbf{B} = 0$

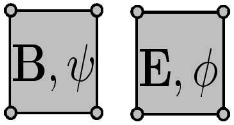
Electromagnetics Summarized Maxwell Formulation

$$\frac{1}{c^2}\frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{B} + \mu_0 \mathbf{J} + \nabla \phi = 0$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} + \nabla \psi = 0$$

$$\frac{1}{c_h}\frac{\partial \phi}{\partial t} + \frac{1}{c_p}\phi + \left[\nabla \cdot \mathbf{E} - \frac{q}{\epsilon_0}\right] = 0$$

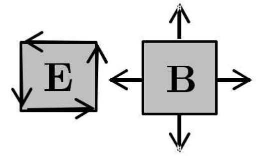
$$\frac{1}{c_h}\frac{\partial \psi}{\partial t} + \frac{1}{c_p}\psi + [\nabla \cdot \mathbf{B}] = 0$$



Or

$$\frac{1}{c^2}\frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{B} + \mu_0 \mathbf{J} = 0 \; ; \quad \nabla \cdot \mathbf{E} = \frac{q}{\epsilon_0}$$

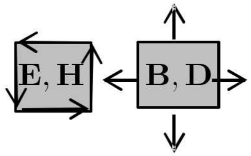
$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = 0 \; ; \quad \nabla \cdot \mathbf{B} = 0$$



Or

$$\frac{\partial \mathbf{D}}{\partial t} - \nabla \times \mathbf{H} + \mathbf{J} = 0 \; ; \quad \nabla \cdot \mathbf{D} = q \; ; \quad \mathbf{E} = \epsilon^{-1}\mathbf{D}$$

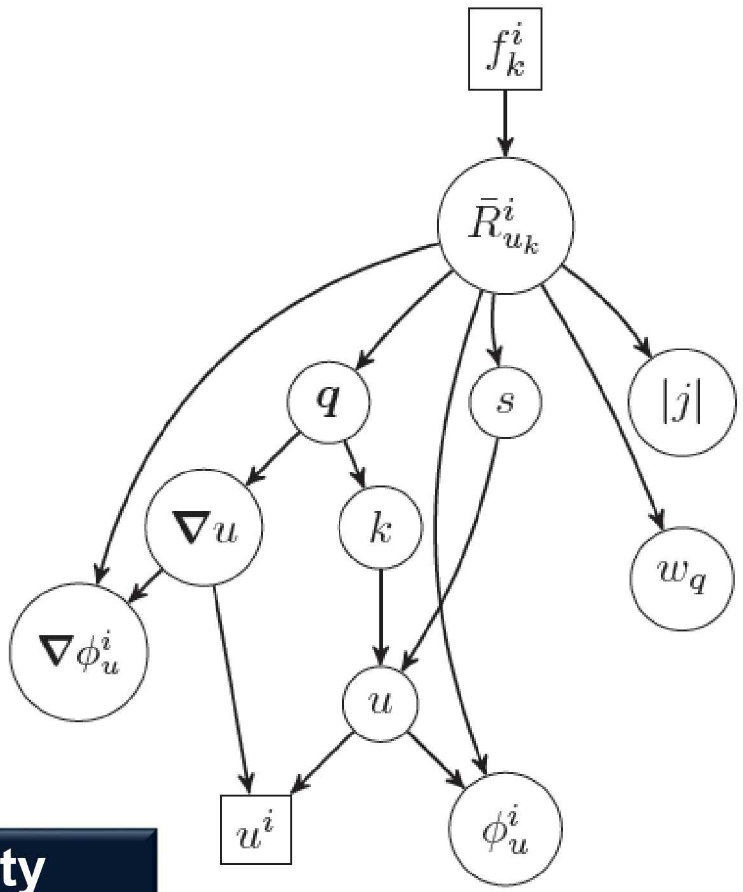
$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = 0 \; ; \quad \nabla \cdot \mathbf{B} = 0 \; ; \quad \mathbf{H} = \mu_0^{-1}\mathbf{B}$$



Phalanx: Lightweight DAG-based Expression Evaluation

$$R_u^i = \int_\Omega \left[\phi_u^i \dot{u} - \nabla \phi_u^i \cdot \boldsymbol{q} + \phi_u^i s\right] \, \mathrm{d}\Omega$$

- Decompose a complex model into a graph of simple kernels (functors)
- A node in the graph evaluates one or more temporary **fields (memory for flexibility)**
- Runtime DAG construction of graph
- Supports rapid development, separation of concerns and extensibility.
- Achieves flexible multiphysics assembly
- Leverages Sacado scalar types for non-invasive Jacobian, Hessian, …
- Combine kernels for performance



DAG-Based Assembly → flexibility

Multi-fluid 5-Moment Plasma System Models

Density	$\frac{\partial \rho_a}{\partial t} + \nabla \cdot (\rho_a \mathbf{u}_a) = \sum_{b \neq a} (n_a \rho_b \bar{\nu}^+_{ab} - n_b \rho_a \bar{\nu}^-_{ab})$
Momentum	$\frac{\partial (\rho_a \mathbf{u}_a)}{\partial t} + \nabla \cdot (\rho_a \mathbf{u}_a \otimes \mathbf{u}_a + p_a I + \Pi_a) = q_a n_a (\mathbf{E} + \mathbf{u}_a \times \mathbf{B})$ $- \sum_{b \neq a} [\rho_a (\mathbf{u}_a - \mathbf{u}_b) n_b \bar{\nu}^M_{ab} + \rho_b \mathbf{u}_b n_a \bar{\nu}^+_{ab} - \rho_a \mathbf{u}_a n_b \bar{\nu}^-_{ab}]$
Energy	$\frac{\partial \varepsilon_a}{\partial t} + \nabla \cdot ((\varepsilon_a + p_a) \mathbf{u}_a + \Pi_a \cdot \mathbf{u}_a + \mathbf{h}_a) = q_a n_a \mathbf{u}_a \cdot \mathbf{E} + Q^{src}$ $- \sum_{b \neq a} \left[(T_a - T_b) k \bar{\nu}^E_{ab} - \rho_a \mathbf{u}_a \cdot (\mathbf{u}_a - \mathbf{u}_b) n_b \bar{\nu}^M_{ab}\right.$
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	$\frac{1}{c^2} \frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{B} + \mu_0 \mathbf{J} = \mathbf{0} \qquad \nabla \cdot \mathbf{E} = \frac{q}{\epsilon_0}$ $\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = \mathbf{0} \qquad \nabla \cdot \mathbf{B} = 0$

Cyclotron Frequency

Strong off diagonal coupling for plasma oscillation

Light wave

IMEX: Time Integration

$$\mathbf{M}\dot{\mathbf{U}} + \mathbf{F} + \mathbf{G} = 0$$

Explicit Hydrodynamics

Implicit EM, EM sources, sources for species $(\rho_a, \rho_a \mathbf{u}_a, \epsilon_a)$ interactions

Tempus Time Integrator: Implicit / Explicit (IMEX) Methods

Governing PDE Semi-discretized in Space (e.g. FV, FD, FE) written as an ODE system

$$\mathbf{u}_t + \underbrace{\mathbf{F}(\mathbf{u})}_{\text{Slow, Explicit}} + \underbrace{\mathbf{G}(\mathbf{u})}_{\text{Fast, Implicit}} = \mathbf{0}$$

IMEX Multi-stage Methods (RK-type) form a consistent set of nonlinear residuals:

$$\mathbf{u}^{(i)} = \mathbf{u}^n + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij} \mathbf{F}(\mathbf{u}^{(j)}) - \Delta t \sum_{j=1}^{i} a_{ij} \mathbf{G}(\mathbf{u}^{(j)}) \qquad \text{for } i = 1 \ldots s,$$

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \sum_{i=1}^{s} \hat{b}_i \mathbf{F}(\mathbf{u}^{(i)}) - \Delta t \sum_{i=1}^{s} b_i \mathbf{G}(\mathbf{u}^{(i)}).$$

$$\begin{array}{c|c} \hat{\mathbf{c}} & \hat{\mathbf{a}} \\ \hline & \hat{\mathbf{b}}^T \end{array} \text{ is explicit, and } \begin{array}{c|c} \mathbf{c} & \mathbf{a} \\ \hline & \mathbf{b}^T \end{array} \text{ is implicit.}$$

High-order accuracy (e.g. 2^{nd} – 5^{th}), with various stability properties have demonstrated:
A-, L-stability, Strong Stability Preserving (SSP), TVB, ….

See for e.g. Ascher, Ruuth and Wetton (1997), Ascher, Ruuth and Spiteri (1997), Carpenter, Kennedy, et. al (2005), Higueras et. al. (2011)

Implicit Sub-problem: NOX, Belos, AztecOO, Ifpack(2), ML, MueLu, Teko

Discrete Nonlinear Sub-problem – Newton's Method

$$\mathcal{F}(\mathbf{u}^{(\mathbf{i})}) = \mathbf{u}^{(i)} - \mathbf{u}^n - \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij} \mathbf{F}(\mathbf{u}^{(j)}) + \Delta t \sum_{j=1}^{i} a_{ij} \mathbf{G}(\mathbf{u}^{(j)}) = 0$$

/* Find u* such that $\mathcal{F}(\mathbf{u}^*) = \mathbf{0}$ */

Until Nonlinear Convergence {

Iteratively solve linear sub-problem (e.g. AMG preconditioned Krylov method)

$$\mathcal{F}'(\mathbf{u_k})\mathbf{s_k} = -\mathcal{F}(\mathbf{u_k}) \quad \text{until} \quad \frac{\|\mathcal{F}'(\mathbf{u_k})\mathbf{s_k} + \mathcal{F}(\mathbf{u_k})\|}{\|\mathcal{F}(\mathbf{u_k})\|} \leq \eta^L$$

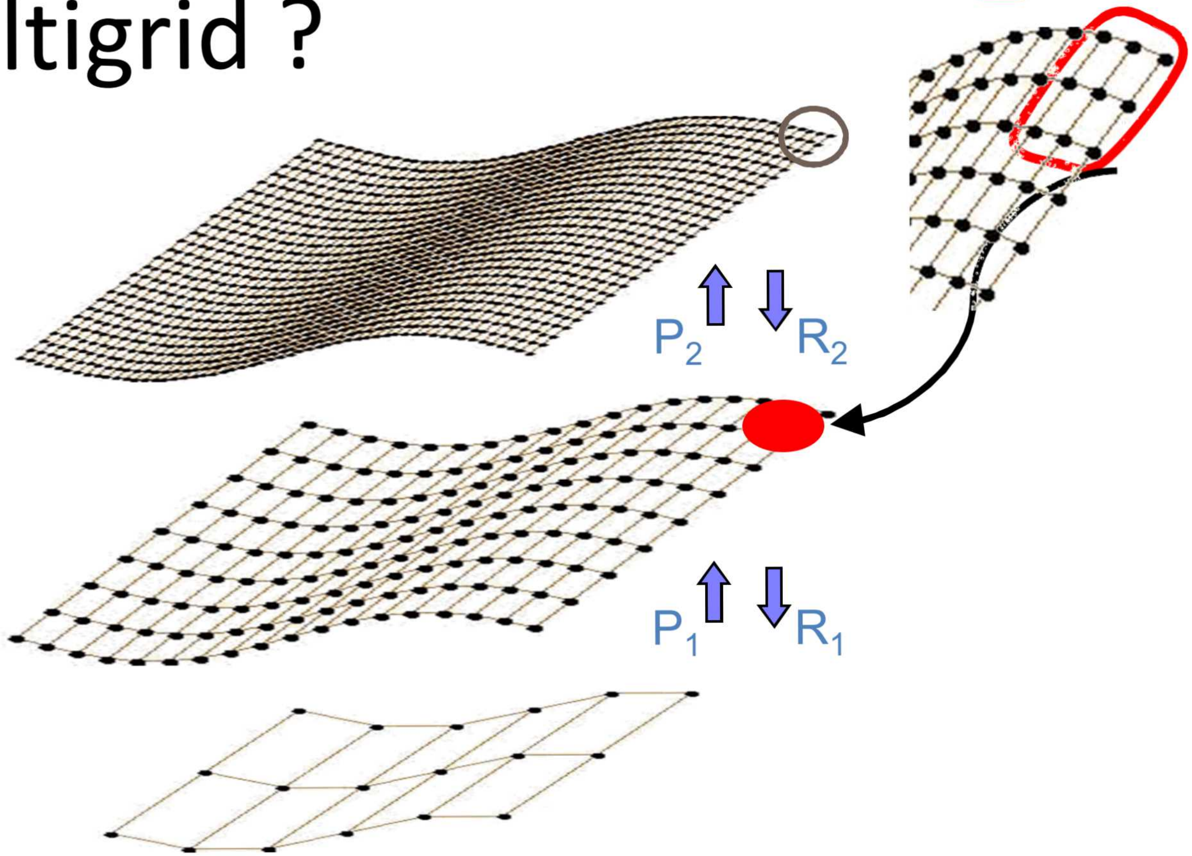
Update Sequence $\mathbf{u_{k+1}} = \mathbf{u_k} + \mathbf{s_k}$

Check nonlinear norms for convergence ($\|\mathcal{F}(\mathbf{u_{k+1}})\|$, $\frac{\|\mathcal{F}(\mathbf{u_{k+1}})\|}{\|\mathcal{F}(\mathbf{u_0})\|}$, $\|\mathbf{s_k}\|_{WRMS}$, nan, max iter);

}

A key technology for implicit/IMEX is AD for Jacobian evaluation!

What is Multigrid ?



Solve $A_3 u_3 = f_3$

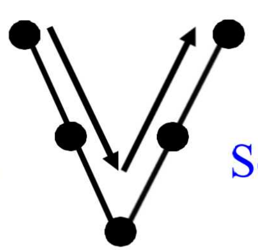
Basic idea:

- Develop coarse approximations on multiple levels (e.g discretize)
- Define prolongation P_i and restriction operator R_i (e.g. for P -FE interpolation)
- Accelerate convergence via coarse iterations to efficiently propagate information across domain

Smooth $A_3 u_3 = f_3$. Set $f_2 = R_2 r_3$.

Smooth $A_2 u_2 = f_2$. Set $f_1 = R_1 r_2$.

Solve $A_1 u_1 = f_1$ directly.



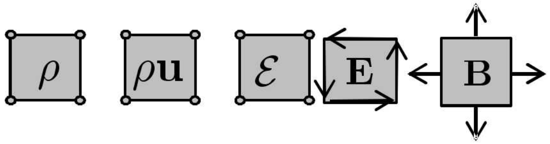
Set $u_3 = u_3 + P_2 u_2$. Smooth $A_3 u_3 = f_3$.

Set $u_2 = u_2 + P_1 u_1$. Smooth $A_2 u_2 = f_2$.

Scalable Physics-based Preconditioners for Physics-compatible Discretizations

$$
\begin{bmatrix}
\mathbf{D}_{\rho_i} & \mathbf{K}^{\rho_i}_{\rho_i u_i} & 0 & \mathbf{Q}^{\rho_i}_{\rho_e} & 0 & 0 & 0 & 0 \\
\mathbf{D}^{\rho_i u_i}_{\rho_i} & \mathbf{D}_{\rho_i u_i} & 0 & \mathbf{Q}^{\rho_i u_i}_{\rho_e} & \mathbf{Q}^{\rho_i u_i}_{\rho_e u_e} & 0 & \mathbf{Q}^{\rho_i u_i}_{E} & \mathbf{Q}^{\rho_i u_i}_{B} \\
\mathbf{D}^{\mathcal{E}_i}_{\rho_i} & \mathbf{D}^{\mathcal{E}_i}_{\rho_i u_i} & \mathbf{D}_{\mathcal{E}_i} & \mathbf{Q}^{\mathcal{E}_i}_{\rho_e} & \mathbf{Q}^{\mathcal{E}_i}_{\rho_e u_e} & \mathbf{Q}^{\mathcal{E}_i}_{\mathcal{E}_e} & \mathbf{Q}^{\mathcal{E}_i}_{E} & 0 \\
\mathbf{Q}^{\rho_e}_{\rho_i} & 0 & 0 & \mathbf{D}_{\rho_e} & \mathbf{K}^{\rho_e}_{\rho_e u_e} & 0 & 0 & 0 \\
\mathbf{Q}^{\rho_e u_e}_{\rho_i} & \mathbf{Q}^{\rho_e u_e}_{\rho_i u_i} & 0 & \mathbf{D}^{\rho_e u_e}_{\rho_e} & \mathbf{D}_{\rho_e u_e} & 0 & \mathbf{Q}^{\rho_e u_e}_{E} & \mathbf{Q}^{\rho_e u_e}_{B} \\
\mathbf{Q}^{\mathcal{E}_e}_{\rho_i} & \mathbf{Q}^{\mathcal{E}_e}_{\rho_i u_i} & \mathbf{Q}^{\mathcal{E}_e}_{\mathcal{E}_i} & \mathbf{D}^{\mathcal{E}_e}_{\rho_e} & \mathbf{D}^{\mathcal{E}_e}_{\rho_e u_e} & \mathbf{D}_{\mathcal{E}_e} & \mathbf{Q}^{\mathcal{E}_e}_{E} & 0 \\
0 & \mathbf{Q}^{E}_{\rho_i u_i} & 0 & 0 & \mathbf{Q}^{E}_{\rho_e u_e} & 0 & \mathbf{Q}_{E} & \mathbf{K}^{E}_{B} \\
0 & 0 & 0 & 0 & 0 & 0 & \mathbf{K}^{B}_{E} & \mathbf{Q}_{B}
\end{bmatrix}
\begin{bmatrix}
\rho_i \\ \rho_i \mathbf{u}_i \\ \mathcal{E}_i \\ \rho_e \\ \rho_e \mathbf{u}_e \\ \mathcal{E}_e \\ \mathbf{E} \\ \mathbf{B}
\end{bmatrix}
$$

16 Coupled Nonlinear PDEs



Group the hydrodynamic variables together (similar discretization)

$$\mathbf{F} = (\rho_i, \rho_i \mathbf{u}_i, \mathcal{E}_i, \rho_e, \rho_e \mathbf{u}_e, \mathcal{E}_e)$$

Resulting 3x3 block system

$$
\begin{bmatrix}
\mathbf{D}_F & \mathbf{Q}^F_E & \mathbf{Q}^F_B \\
\mathbf{Q}^E_F & \mathbf{Q}_E & \mathbf{K}^E_B \\
0 & \mathbf{K}^B_E & \mathbf{Q}_B
\end{bmatrix}
\begin{bmatrix} \mathbf{F} \\ \mathbf{E} \\ \mathbf{B} \end{bmatrix}
$$

Reordered 3x3

$$
\begin{bmatrix}
\mathbf{Q}_B & \mathbf{K}^B_E & 0 \\
\mathbf{K}^E_B & \mathbf{Q}_E & \mathbf{Q}^E_F \\
\mathbf{Q}^F_B & \mathbf{Q}^F_E & \mathbf{D}_F
\end{bmatrix}
\begin{bmatrix} \mathbf{B} \\ \mathbf{E} \\ \mathbf{F} \end{bmatrix}
$$

Physics-based Approach Enables Optimal AMG Sub-block Solvers

$$\begin{bmatrix} \mathbf{Q}_B & \mathbf{K}_E^B & 0 \\ 0 & \hat{\mathcal{D}}_E & \mathbf{Q}_F^E \\ 0 & 0 & \hat{\mathbf{S}}_F \end{bmatrix} \begin{bmatrix} \mathbf{B} \\ \mathbf{E} \\ \mathbf{F} \end{bmatrix}$$

$$\hat{\mathbf{S}}_F = \mathbf{D}_F - \mathcal{K}_E^F \tilde{\tilde{\mathcal{D}}}_E^{-1} Q_F^E$$

CFD type system
node-based coupled
ML: H(grad) AMG
(SIMPLEC: Schur-compl.)

$$\hat{\mathcal{D}}_E = \mathbf{Q}_E - \mathbf{K}_B^E \bar{\mathbf{Q}}_B^{-1} \mathbf{K}_E^B$$

Compare to: $\frac{\partial^2 \mathbf{E}}{\partial t^2} - \frac{1}{\sigma \mu_0} \nabla \times \nabla \times \mathbf{E} = \mathbf{0}$

Electric field system
Edge-based curl-curl type
ML: H(curl) AMG
(lumped mass)

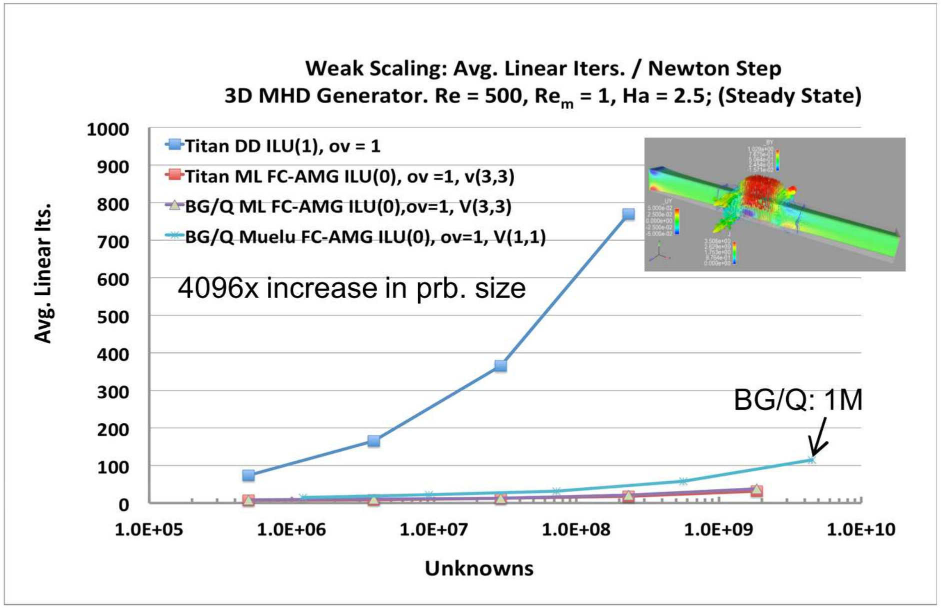
$$\mathbf{B} = -\bar{\mathbf{Q}}_B^{-1} \mathbf{K}_E^B \ \mathbf{E}$$

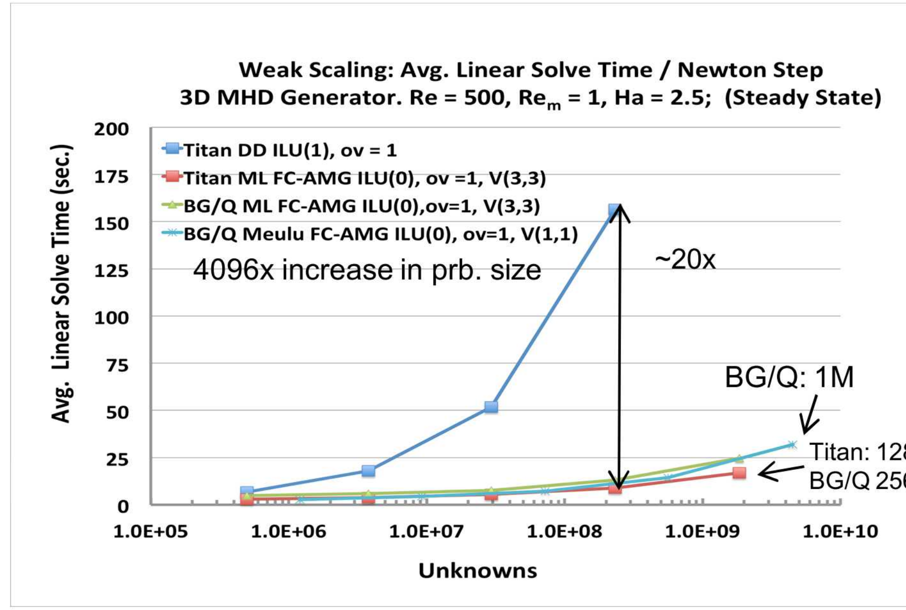
Face-based simple
mass matrix Inversion.

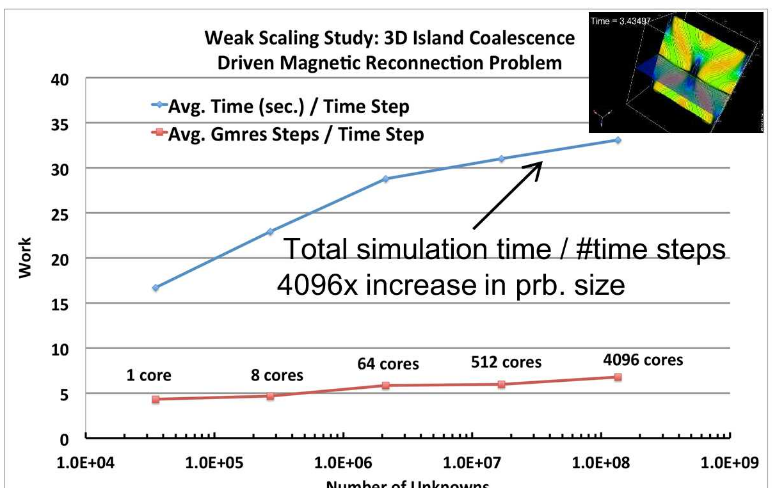
Large-scale Scaling Studies for Cray XK7 AND BG/Q; VMS 3D FE MHD

Sandia National Laboratories

u P B r (similar discretizations for all variables, fully-coupled H(grad) AMG)







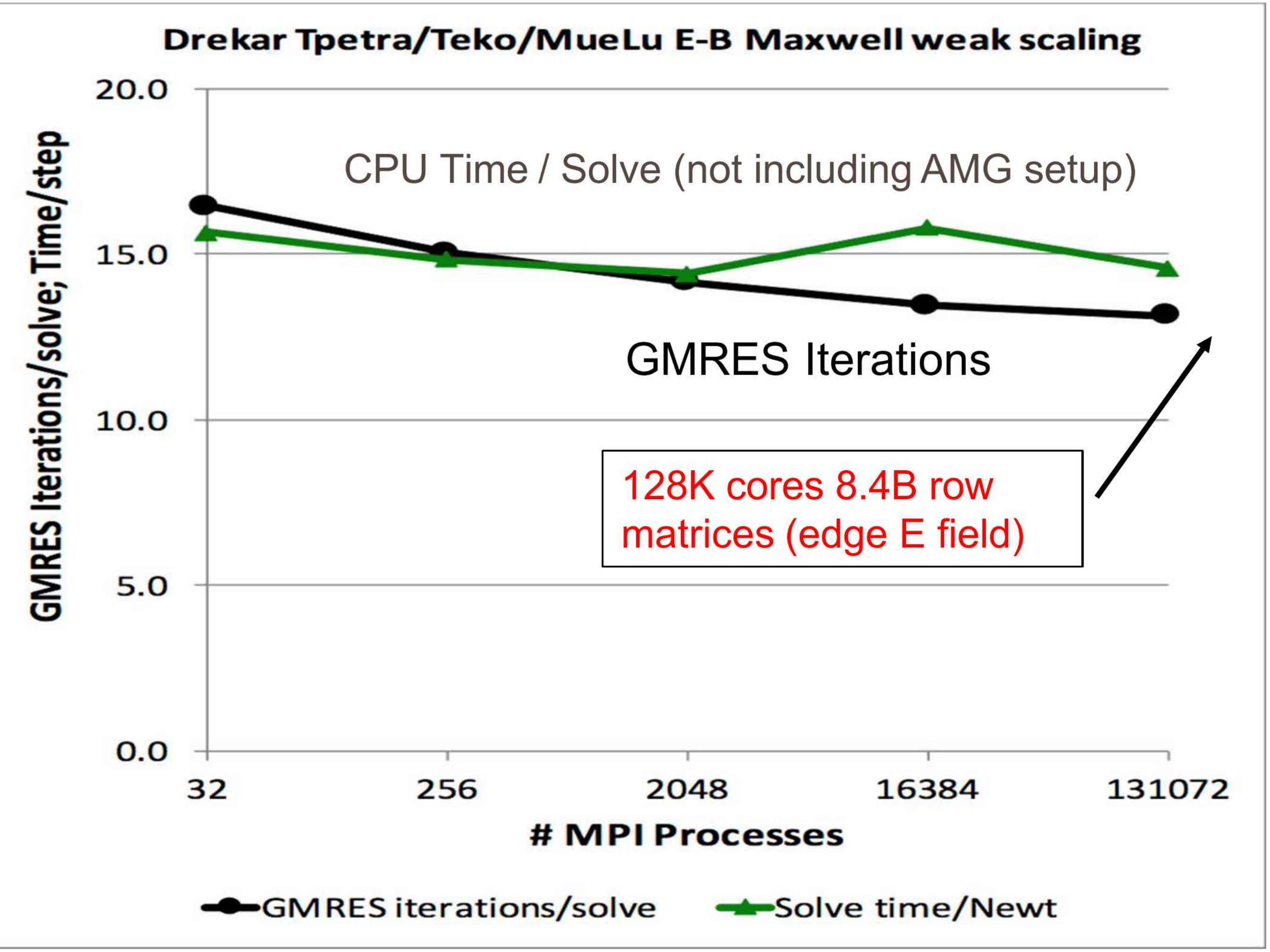
Largest fully-coupled unstructured FE MHD solves demo. to date:

MHD (steady) weak scaling studies to 128K Cray XK7, 1M BG/Q
Large demonstration computations

- MHD (steady): 13B DoF, 1.625B elem, on 128K cores
- CFD (Transient): 40B DoF, 10.0B elem, on 128K cores

Poisson sub-block solvers: 4.1B DoF, 4.1B elem, on 1.6M cores

Weak Scaling for 3D Electro Magnetic Pulse with Block Maxwell Eq. Preconditioners on Trinity



$$\mathcal{D}^E = \mathrm{Q}_E - \mathrm{K}_B^E \bar{\mathrm{Q}}_B^{-1} \mathrm{K}_E^B$$

Maxwell subsystem: electric field Edge-based curl-curl type system.

Good scaling on block solves (at least for solve; setup needs improvement)

Demonstrated to $CFL_c > 10^4$

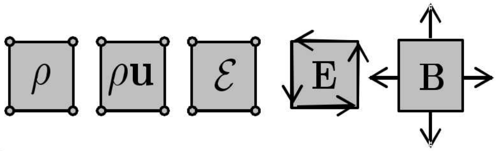
Drekar

GS smoother with H(grad) AMG

Max $CFL_c \sim 200$

Initial Weak Scaling for Longitudinal Electron / Ion Plasma Oscillation and Under-resolved TEM Wave Results (Full Maxwell – two-fluid)

$$\Delta t = 1.1 \times 10^{-11} \approx 0.023\ \tau_{\omega_{pi}} \approx 0.1\ \tau_{\omega_{pe}} \geq 3 \times 10^{2}\ \tau_c$$



Structure-preserving discretization

N	P	Linear its / Newton	Solve time / linear solve	$\frac{\Delta t_{imp}}{\Delta t_{exp}}$
100	1	4.18	0.2	300
200	2	4.21	0.22	600
400	4	4.27	0.23	1.2E+3
800	8	4.4	0.26	2.4E+3
1600	16	4.51	0.35	4.8E+3
3200	32	4.89	0.42	9.6E+3
6400	64	6.21	0.61	1.9e+4

$$\mu = \frac{m_i}{m_e} = 1836.57 \qquad \Delta x \approx 1 \mu m$$

Initial weak scaling of ABF preconditioner

- Domain [0,0.01]x[0,0.0004]x[0,0.0004]; Periodic BCs in all directions
- N elements in x-direction;
- Fixed time step size for SDIRK (2,2): (not resolving TEM wave)

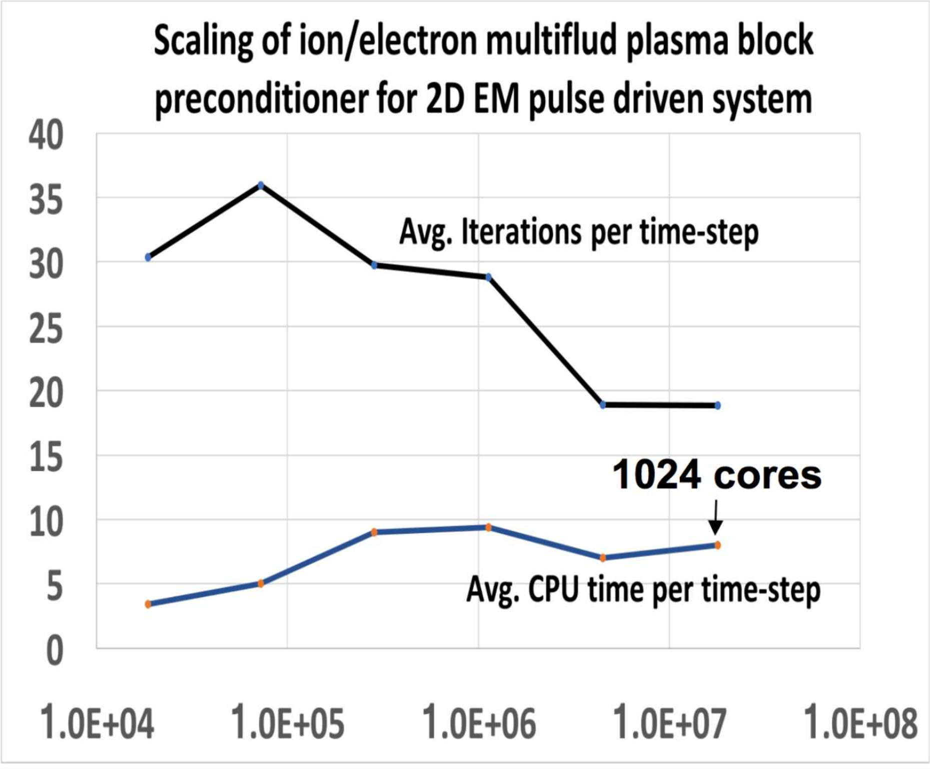
<u>Proof of Principle</u>

SimpleC on fluid Schur-complement
DD-ILU for Euler Eqns.
DD-LU curl-curl

A MORE REALISTIC TEST PROBLEM

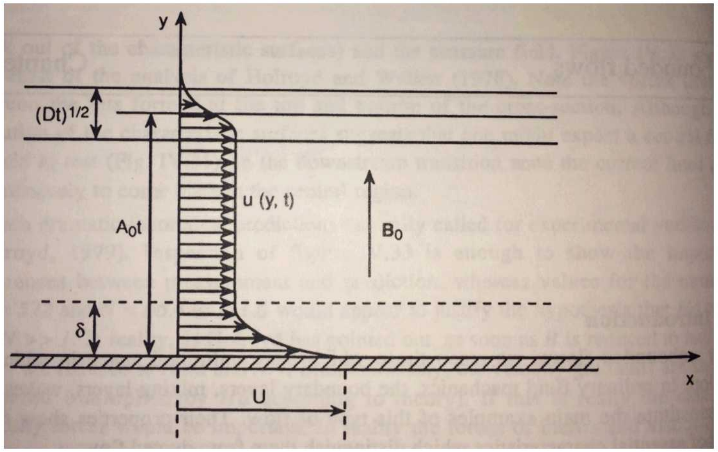
- 2D electron/ion plasma driven by an external current pulse with background magnetic field and density gradient
- Simulation resolves current source

Time-scale	Pulse Problem
CFL_{EM}	$[6.25 \times 10^{-2}, 2.0]$
$CFL_{\mathbf{u}_e}$	$[3.75 \times 10^{-2}, 1.2]$
$CFL_{\mathbf{u}_i}$	$[3.75 \times 10^{-5}, 1.2 \times 10^{-3}]$
CFL_{s_e}	0
CFL_{s_i}	0
$CFL_{\omega_{p,e}}$	1.2×10^2
$CFL_{\omega_{p,i}}$	3.8
$CFL_{\omega_{c,e}}$	2.7
$CFL_{\omega_{c,i}}$	2.7×10^{-3}



Resistive Alfven wave problem

- Solution is derived from resistive/viscous MHD which **ignores Hall effects:**
  - Hall parameter $\mathrm{H} = \frac{\omega_{ce}}{\nu_{ei}} = \frac{\eta B}{n_e e} \ll 1$
  - Reducing Hall effects in magnetized multi-fluid model is tricky - requires large collision frequency
- Problem used for verifying resistive, Lorentz force, and viscous operators:
  - Impulse shear due to a moving wall drives a **Hartmann layer**
  - Hartmann layer shear excites **Alfven wave** traveling along magnetic field
  - Alfven wave front diffuses due to momentum and magnetic diffusivity
  - Profile depends on the effective **Lundquist number** $S = \frac{L v_A}{\lambda}$



R. Moreau, Magnetohydrodynamics, 1990

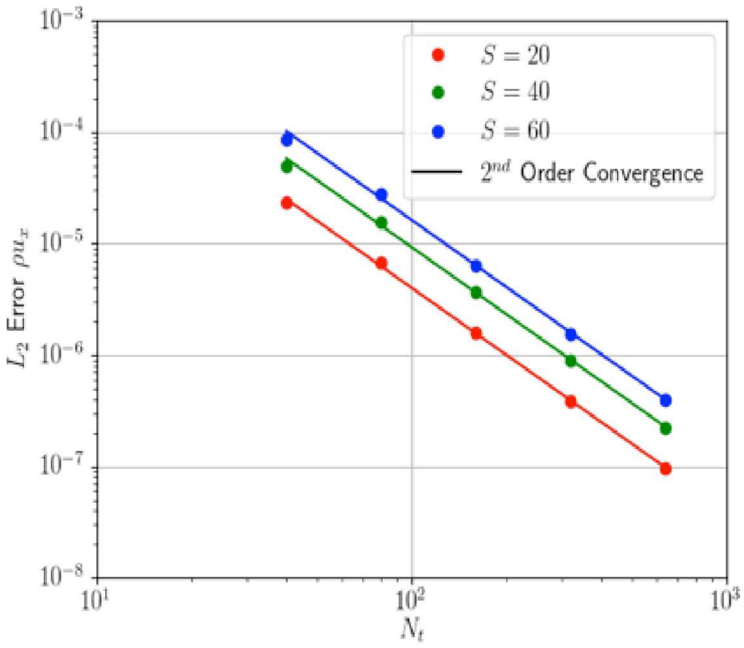
$$u_x = \frac{U}{4}\left(1 + \exp\left(\frac{v_A y}{\lambda}\right)\right)\mathrm{erfc}(\eta_+) + \frac{U}{4}\left(1 + \exp\left(-\frac{v_A y}{\lambda}\right)\right)\mathrm{erfc}(\eta_-)$$

$$B_x = \sqrt{\mu_0 \rho}\frac{U}{4}\left(1 - \exp\left(\frac{v_A y}{\lambda}\right)\right)\mathrm{erfc}(\eta_+) - \sqrt{\mu_0 \rho}\frac{U}{4}\left(1 - \exp\left(-\frac{v_A y}{\lambda}\right)\right)\mathrm{erfc}(\eta_-)$$

$$\eta_\pm = \frac{y \pm v_A t}{2\sqrt{\lambda t}}$$

Asymptotic Solution of multifluid in MHD Limit:

Implicit L-stable and IMEX SSP/L-stable time integration and block preconditioners enable solution of multifluid EM plasma model in the asymptotic resistive MHD limit.
(Simple Visco-resistive Alfven wave)



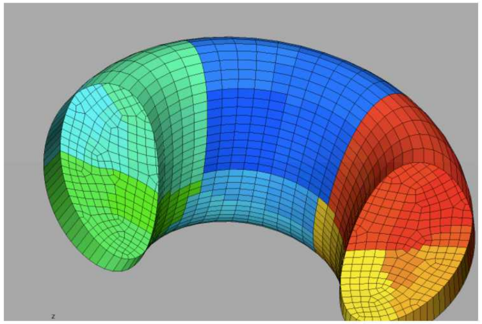
Accuracy in MHD limit (IMEX)

Plasma Scales for S = 60		
	Electrons	Ions
$\omega_p \Delta t$	10^7 - 10^9	10^6 - 10^7
$\omega_c \Delta t$	10^6 - 10^7	10^3 - 10^4
$\nu_{\alpha\beta} \Delta t$	10^{10} - 10^{11}	10^7 - 10^8
$\nu_S \Delta t / \Delta x$	10^{-2}	10^{-4}
$u \Delta t / \Delta x$	10^{-4}	10^{-4}
$\mu \Delta t / \rho \Delta x^2$	10^{-1} - 10^1	10^{-2} - 10^0
$c \Delta t / \Delta x$	10^2	

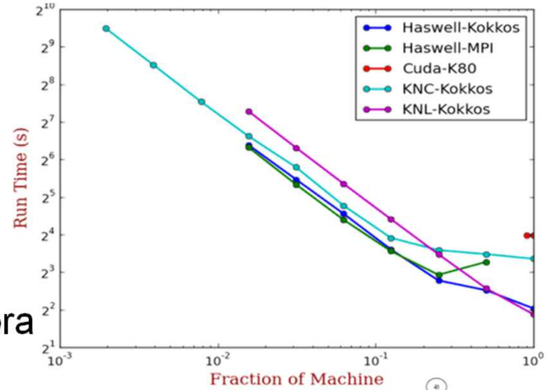
IMEX terms: implicit/explicit

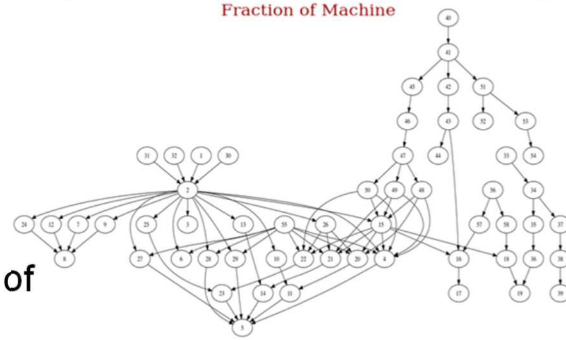
Drekar: Software Infrastructure Pushing Limits of Component Integration

- 1^{st}-5^{th} order fully-implicit and implicit / explicit (IMEX) [Tempus, Rythmos]
- 2D & 3D Unstructured finite elements (FE), HEX and Tet with nodal FE and physics compatible (node, edge, face, …) methods [Drekar, Intrepid2]
- Fully coupled globalized Newton-Krylov (NK) solver
  - Residuals are Programed and Automatic Differentiation (AD generates Jacobian for NK, Sensitivities, Adjoints, etc. [SACADO]
  - GMRES Krylov solvers using compressed sparse row (CSR) [AztecOO, Belos]
  - Scalable Preconditioners: Fully-coupled system AMG, Physics-based block preconditioners with AMG [ML, Muelu, Teko]



- Software architecture:
  - Massively parallel R&D code:
    - MPI version demonstrated weak scaling to 1M cores; sub-block solvers to 1.6M cores
    - MPI+X. Employs Kokkos performance portability abstraction layer interface and Kokkos-kernels utilities. FE assembly and linear algebra gather / scatter to CSR global distributed sparse matrix kernels, demonstrated on a wide variation of advanced node architectures (see figure). Solvers in process.
    - Asynchronous many Tasking (AMT) possible in future with DAG [Phalanx]
  - Solvers/Linear Alg. tools based on Trilinos packages (Aztec/Belos, ML/Muelu, Epetra/TPetra, Teko, Ifpack, Ifpack2, ShyLu, etc.)
  - Template-based generic programming with automatic differentiation [AD] of FE weak forms (Sacado)
  - Core FE assembly capability (Panzer, Intrepid, Kokkos)
  - Asynchronous dependency graphs (DAG) used for multiphysics complexity management and possible AMT capability (Phalanx)





[Drekar: Shadid, Pawlowski, Cyr, Phillips, Lin, Smith, Conde, **Mabuza, Miller]**

EM-PIC Model

Kinetic equation (Klimontovich) for phase space density of each plasma species N_s

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

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

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

- Lagrangian particles updated by F=ma
- Currently solved with explicit time integration
- PIC code contains its own EM field solver

Maxwell's Equations

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

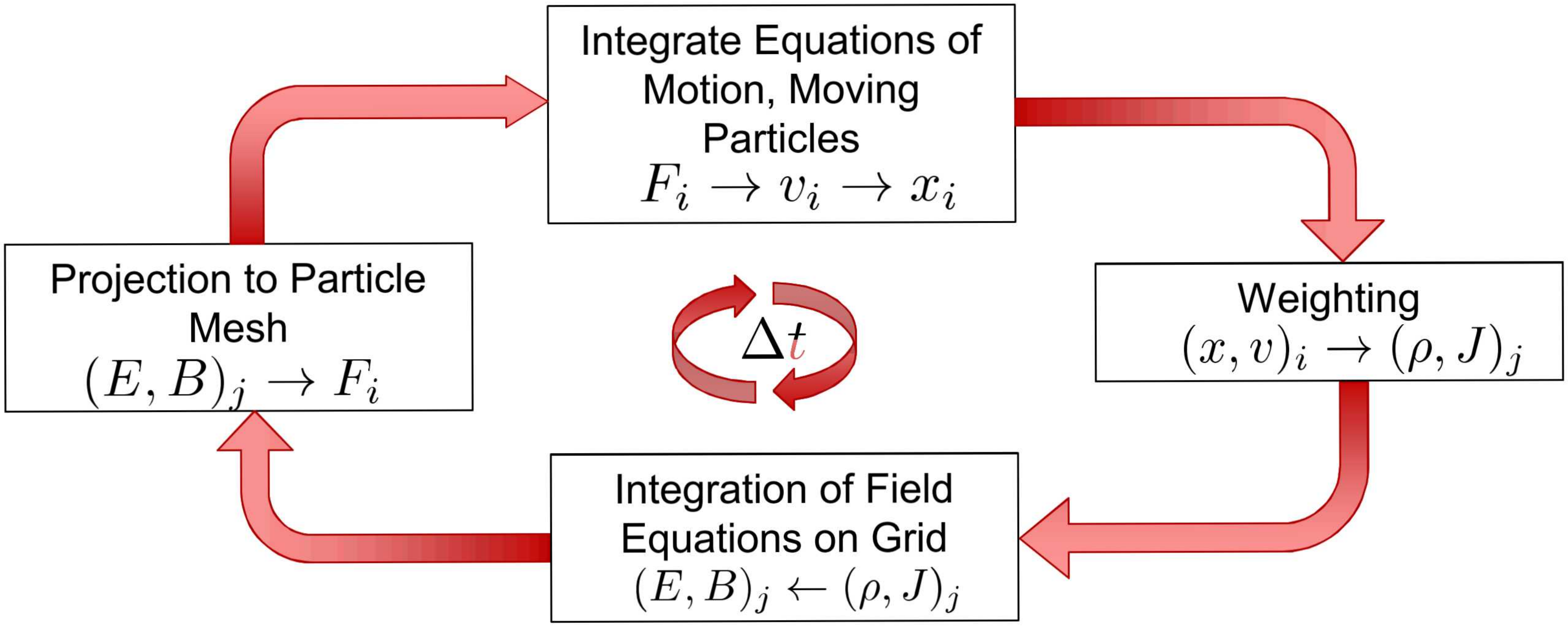
$$\nabla\cdot\mathbf{B}(\boldsymbol{x},t) = 0$$

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

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

Operator Split Coupled Model

$$m_s \frac{d}{dt}\left(\boldsymbol{V}_i(t)\gamma_i(t)\right) = q_s\left(\mathbf{E}(\boldsymbol{X}_i(t),t) + \boldsymbol{V}_i(t) \times \mathbf{B}(\boldsymbol{X}_i(t),t)\right) \qquad \gamma_i(t) = 1/\sqrt{1 - \boldsymbol{V}_i^2(t)/c^2}$$



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

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

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

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

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

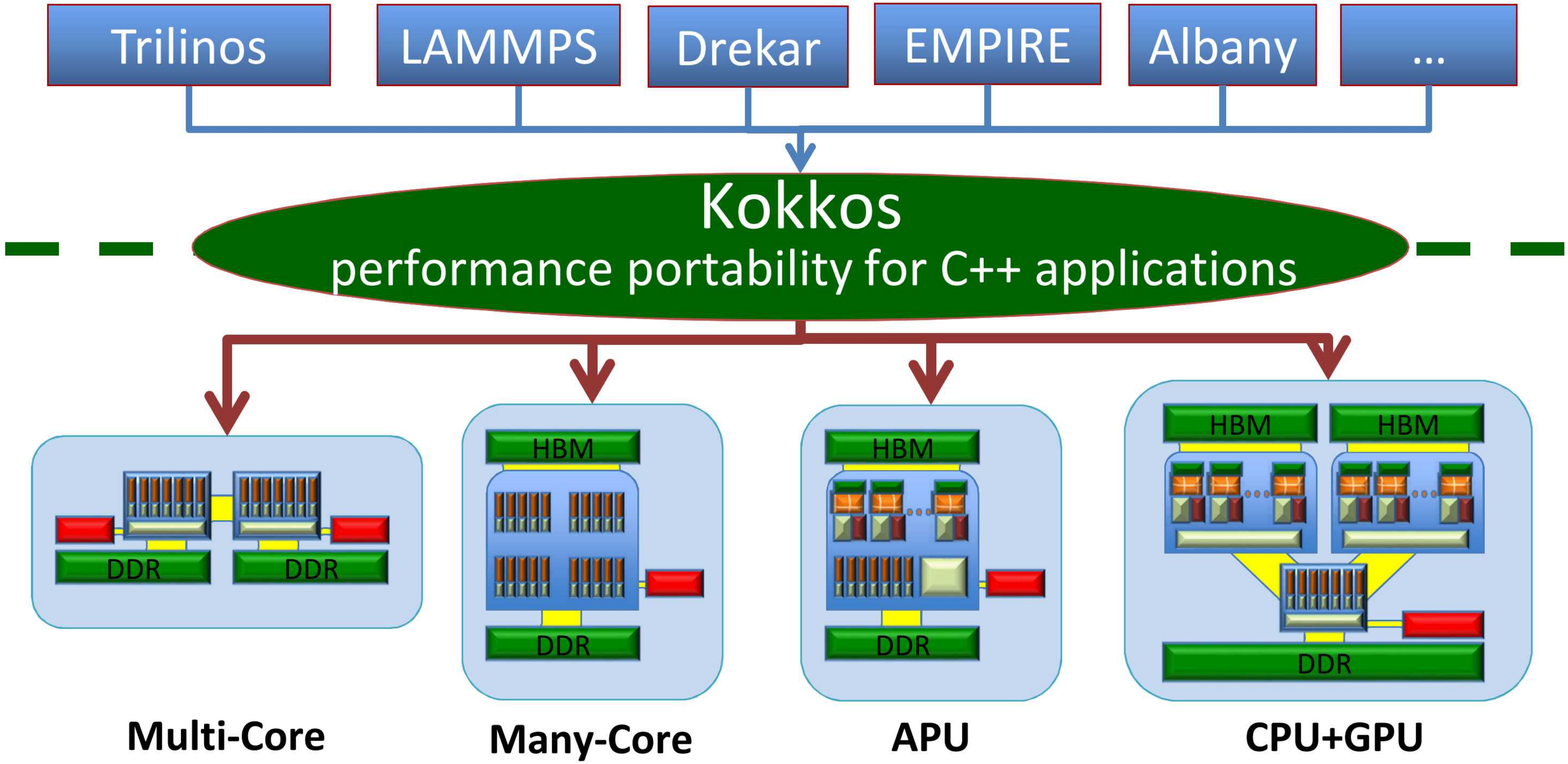
$$\boldsymbol{J}(\boldsymbol{x},t) = \sum_{species} q_s \int d\boldsymbol{u}\boldsymbol{u} N_s(\boldsymbol{x},\boldsymbol{u},t)$$

Software: Critical Technology

- Performance portability on next generation architectures
  - CPU, PHI, GPU, …
  - Easy for application teams to code
- Sensitivities
  - Implicit and IMEX time integrators, parametric sensitivity analysis, optimization, stability, bifurcation analysis
  - Combinatorial explosion of sensitivity requirements → AD is only viable solution!
  - Do not burden analysts/physics experts with analysis algorithm requirements

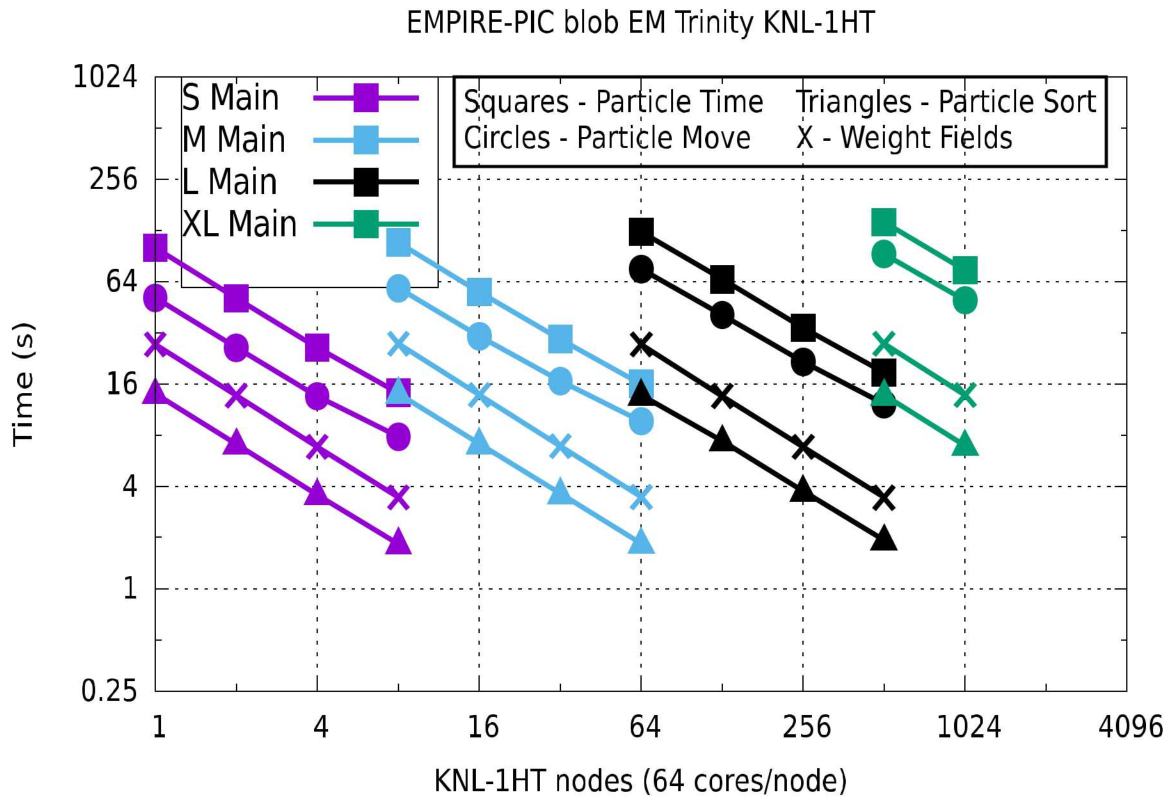
Performance Portability: Kokkos

- Performance Portable Thread-Parallel Programming Model in C++
- Multidimensional Array
- Compiletime polymorphic memory layouts: cached vs coalesced memory
- Asynchronous Many Tasking



https://github.com/kokkos/kokkos

Performance Portability: PIC



Size	# of Elements	# of Nodes	# of Edges	# of Faces	# of Particles
S	337k	60k	406k	683k	16M
M	2.68M	462k	3.18M	5.40M	128M
L	20.7M	3.51M	24.4M	41.6M	1B
XL	166M	27.9M	195M	333M	8.2B
XXL	1.332B	223M	1.56B	2.67B	65.6B

Sacado: Template-based Automatic Differentiation

- Implement equations templated on the scalar type
- Libraries provide new scalar types that ***overload the math operators*** to propagate embedded quantities
  - Expression templates for performance
  - Derivatives: FAD, RAD
  - Hessians
  - Stochastic Galerkin: PCE
  - Multipoint: Ensemble (Stokhos)
- Analytic Values (NO FD involved)!

```
template <typename ScalarT>
void computeF(ScalarT* x, ScalarT* f)
{
  f[0] = 2.0 * x[0] + x[1] * x[1];
  f[1] = x[0] * x[0] * x[0] + sin(x[1]);
}
```

double **Fad<double>**

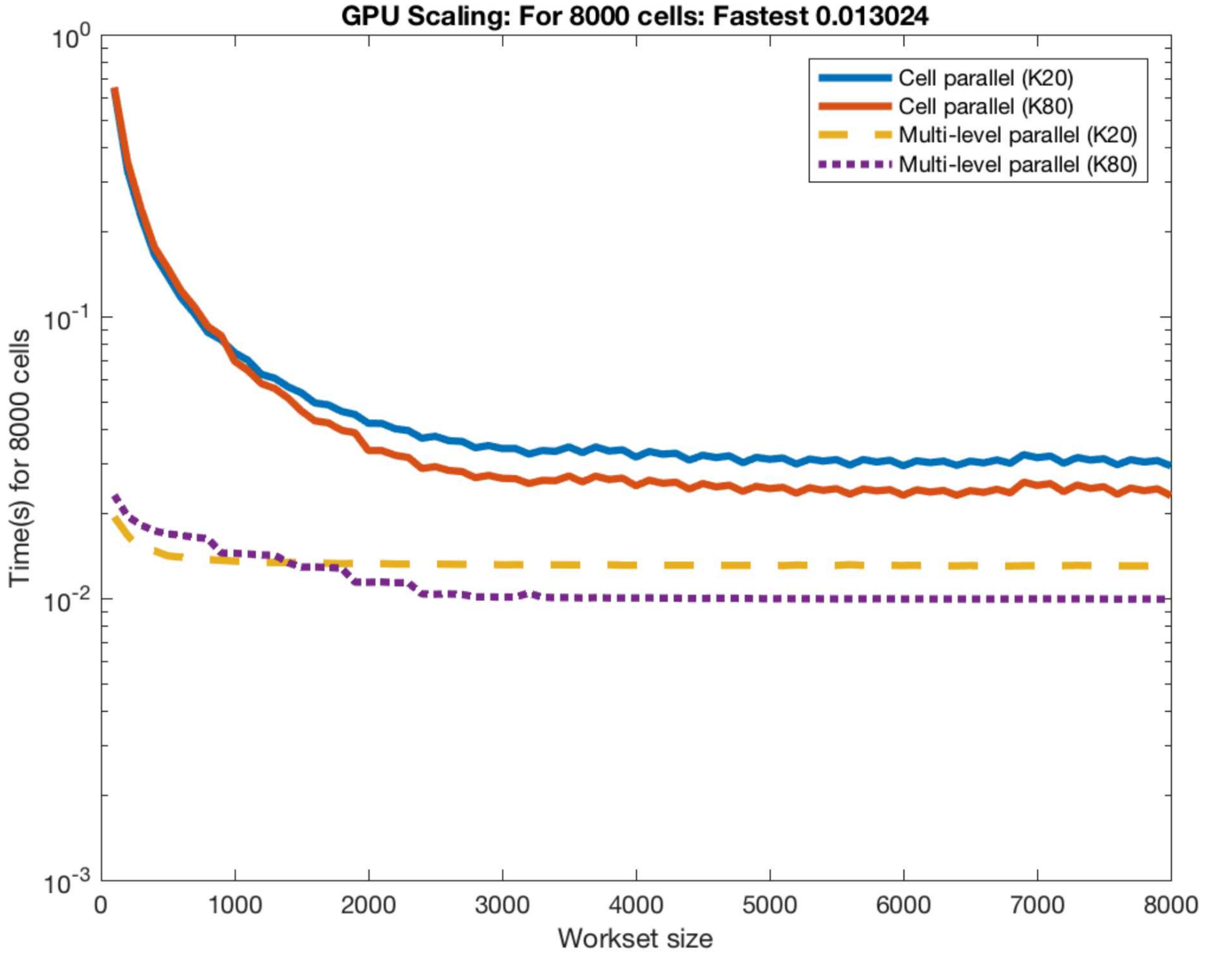
Operation	Forward AD rule
$c = a \pm b$	$\dot{c} = \dot{a} \pm \dot{b}$
$c = ab$	$\dot{c} = a\dot{b} + \dot{a}b$
$c = a/b$	$\dot{c} = (\dot{a} - c\dot{b})/b$
$c = a^r$	$\dot{c} = ra^{r-1}\dot{a}$
$c = \sin(a)$	$\dot{c} = \cos(a)\dot{a}$
$c = \cos(a)$	$\dot{c} = -\sin(a)\dot{a}$
$c = \exp(a)$	$\dot{c} = c\dot{a}$
$c = \log(a)$	$\dot{c} = \dot{a}/a$

```
double* x;
double* f;
…
computeF(x,f);
```

```
DFad<double>* x;
DFad<double>* dfdx;
…
computeF(x,dfdx);
```

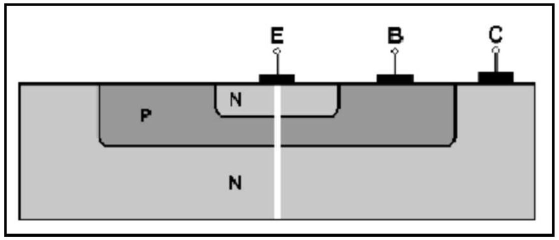
Single CFD Kernel GPU Performance Assessment

- Single level parallelism is insufficient
- Does not expose enough parallelism
- 3-level hierarchical parallelism shows significant improvement
- Hand coded sensitivity array outside libraries
- ***Key is to parallelize (vectorize) over FAD derivative dimension***

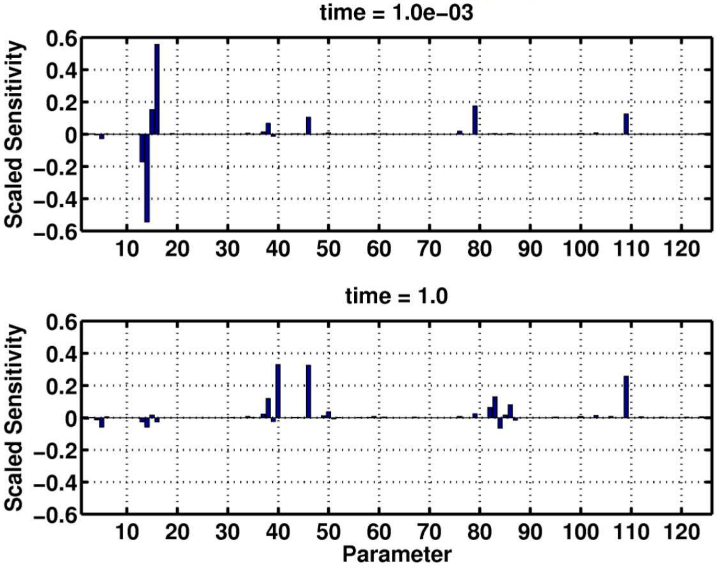


Sensitivity Analysis Capability Demonstrated on the QASPR Simple Prototype

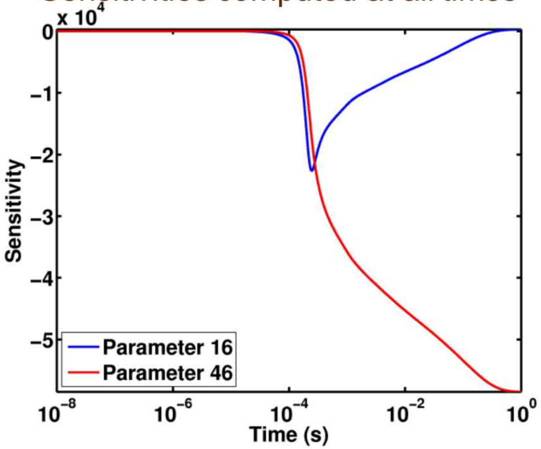
- Bipolar Junction Transistor
- Pseudo 1D strip (9x0.1 micron)
- Full defect physics
- 126 parameters



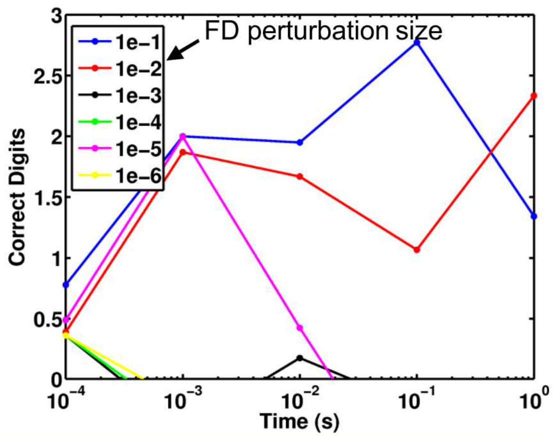
Sensitivities show dominant physics



Sensitivities computed at all times



1st-order Finite Difference Accuracy



Comparison to FD:

- ✓ Sensitivities at all time points
- ✓ More accurate
- ✓ More robust
- ✓ 14x faster!

Perspectives from Application Devs

Sandia National Laboratories

- Every component is a RISK to production software!
  - Applications have 20 to 30+ year lifetimes
- Components really help getting started and really hurt getting finished
  - Tend to be general solutions that improve productivity and not specific solutions which are much faster (scope creep)
  - Suggestion: Expose low level building blocks
- Components can make it difficult for new users to orient themselves in the code
  - Jump between code: orienting in code, new abstractions, new coding styles
  - Simple loops become time consuming code explorations
  - Flip side is components can also hide complexity!
  - Suggestion: allow for simple implementations side-by-side
- Heavy adoption of a component can impact agility
  - Strong interdependencies can discourage change: larger communication/coordination/testing
  - Support for backwards compatibility drives up software development costs
  - Significant resources to integration testing

Summary

- Progress towards a hybrid code
  - Developing new set of codes for Fluid and PIC
  - Verification test suite for individual codes in place
  - Initial coupling of codes performed

- Performance portability and scalability assessment underway
  - PIC shows excellent strong and weak scaling
  - Fluid assembly shows good scaling
  - Preconditioners and solvers are being assessed, low work per core an issue for this application

- Multiphysics heterogeneity
  - AD is essential to prevent combinatorial explosion of code for sensitivities
  - Hybrid parallelism essential for AD performance on next-gen architectures

Extra Slides

Abstract

Building a Scalable Plasma Physics Capability from Components

R. P. Pawlowski, M. Bettencourt, E. C. Cyr, P. T. Lin, C. Ober, E. G. Phillips and J. N. Shadid

Plasma physics systems are often simulated by either a continuum approach (e.g. magnetohydrodynamics or multi-fluid plasma models) in the continuum limit, or as a collection of charged particle (e.g. Boltzmann equation) in the rarified limit. Particle-in-cell (PIC) methods are typically applied to solve the Klimontovich equation that is the fundamental equation for the distribution function describing particle motion in the presence of electromagnetic fields. For some problems, a combined hybrid model that couples the fluid and PIC descriptions may be required for tractability. The resulting systems are characterized by strong nonlinear and nonsymmetric coupling of fluid and electromagnetic phenomena, as well as the significant range of time- and length-scales that the interactions of these physical mechanisms produce. To enable accurate and stable approximation of these systems a range of spatial and temporal discretization methods are commonly employed.

This presentation describes an effort to build a scalable and performant software stack for finite element (FE) multi-species fluids, PIC and coupled fluid-kinetic hybrid capabilities heavily leveraging software components. The software is designed for next-generation architectures using components under development for the ASC and ECP programs. The multi-fluid model consists of continuity, momentum and energy equations for each species coupled to Maxwell's equations for the electromagnetic field. The equations are discretized using the continuous and discontinuous Galerkin FE method with a compatible basis to enforce the electric field (edge basis) and magnetic field (face basis) involutions from Maxwell's equations. The resulting set of fluid equations contains a wide range of multiple time and length-scale physical mechanisms, producing a stiff system. To evolve the fluid models and coupled kinetic-PIC / fluid system for the time scales of interest, we utilize IMplicit-EXplicit (IMEX) Runge-Kutta time integration methods and operator split methods. For robustness, efficiency, and scalability the implicit nonlinear fluid physics are solved using a Newton-Krylov method with a GMRES linear solve and an approximate block factorization preconditioner. Algebraic multigrid is applied within the blocks.

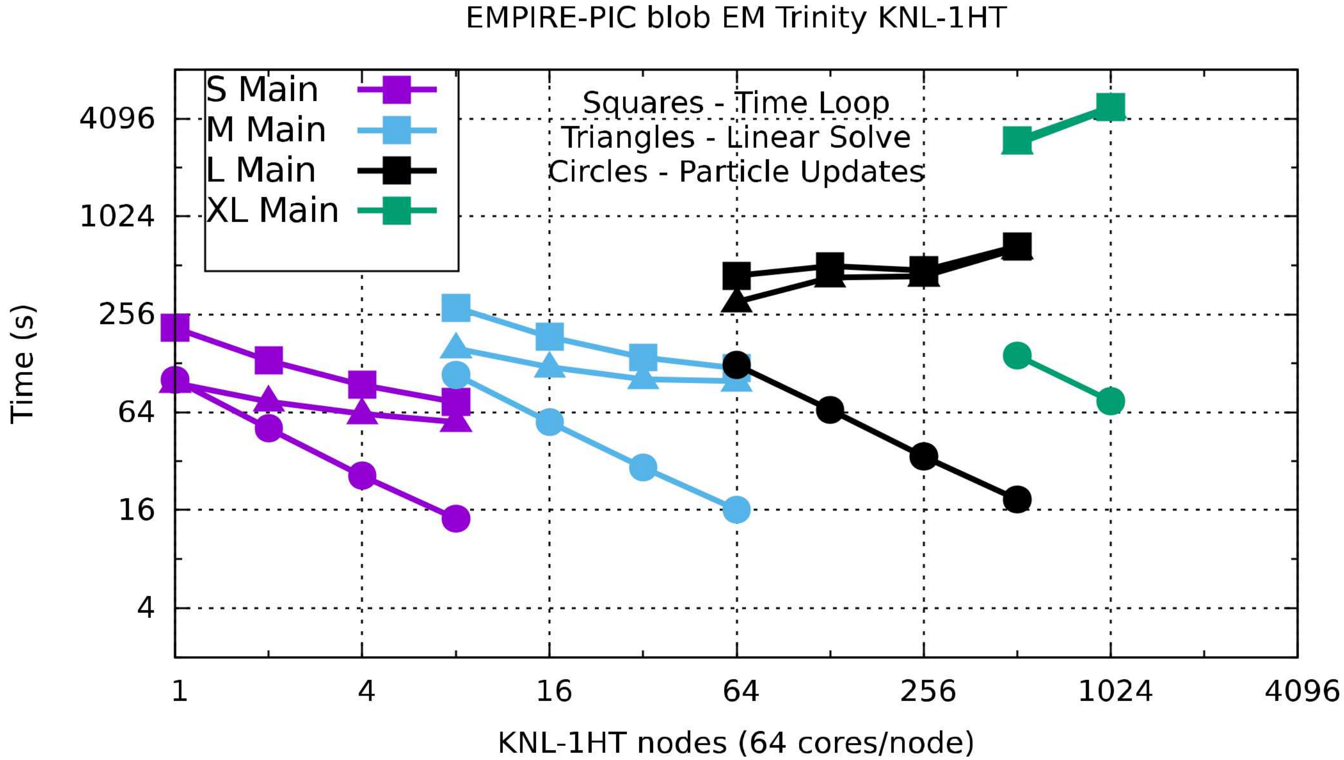
Performance portability is achieved via the Kokkos programming model. The solvers, preconditioners, linear algebra data structures and discretizations leverage 35 packages from the Trilinos Project. This talk will focus on the challenges in developing the components, integrating them with applications and building a long-term sustainable software ecosystem. Results will be shown for Haswell, KNL and CUDA architectures.

Session: Scalable Applications from Scalable Components

This session is intended to discuss the challenges and gaps in algorithms and libraries employed by traditional scalable simulations from both the enabling technologies perspective as well as from an application scientist's view. We would like to invite you to be a speaker in this important session. Participation includes a 30 minute talk on your viewpoint of the role of enabling frameworks in simulation science and any challenges or gaps that must be addressed in the exascale era. In particular, we would like for you to overview your experience in both developing enabling technologies and frameworks for simulation science as well as deploying them in your applications of interest.

Performance Portability (EM-PIC)

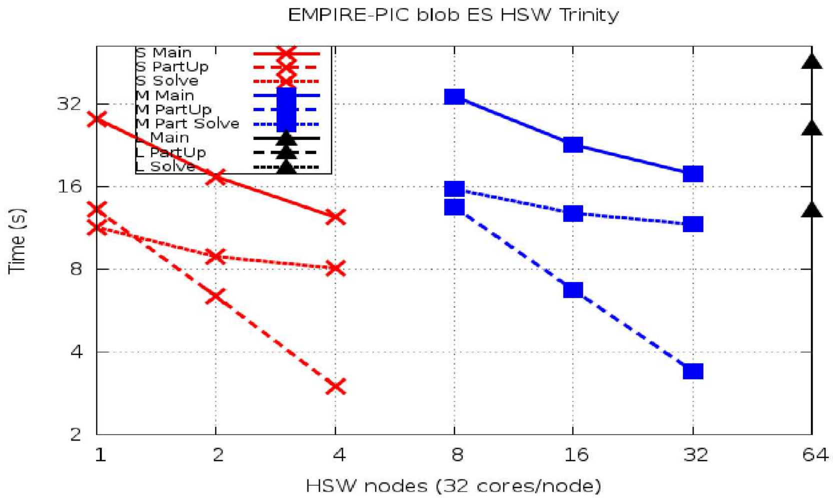
Milestone to assess component status, test new algorithms

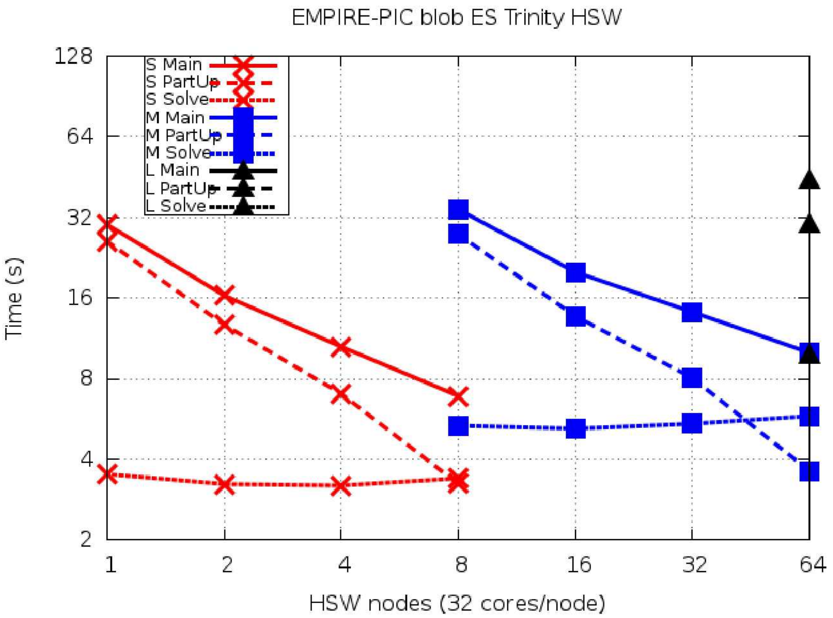


- RefMaxwell preconditioner in Mini-EM showed promise (didn't turn out well)
- ES solve: small work per core - 1800 DOFs/core process in field solve
- EM solve: algorithmic scalability issues with RefMaxwell Preconditioner
- Identified a number of areas for improvement!

Component Improvements

ES Simulation on Trinity: 4 MPI x 16 OMP (1 HT) per KNL node.





May → Aug

- Improved particle sort in application: test problem change - increased Particles ~70%
- New preconditioner: Refmaxwell vs block factorization (app and comp mod)
- Split projections from solve (app mod, comp addition)

Example Scalar Types

(Trilinos Stokhos and Sacado: E. Phipps)

Evaluation Types		Scalar Types
• **Residual**	$F(x,p)$	`double`
• **Jacobian**	$J = \frac{\partial F}{\partial x}$	`DFad<double>`
• **Hessian**	$\frac{\partial^2 F}{\partial x_i \partial x_j}$	`DFad< SFad<double,N> >`
• **Parameter Sensitivities**	$\frac{\partial F}{\partial p}$	`DFad<double>`
• **Jv**	Jv	`DFad<double>`

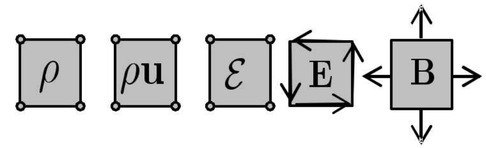
1. All evaluation types are compiled into single library and managed at runtime from a non-template base class via a template manager.
2. Not tied to double (can do arbitrary precision)
3. Can mix multiple scalar types in any evaluation type.
4. Can specialize any node: Write analytic derivatives for performance!

SNL's Mission Requires a Significant Range of Advanced Simulation Capabilities

Sandia National Laboratories

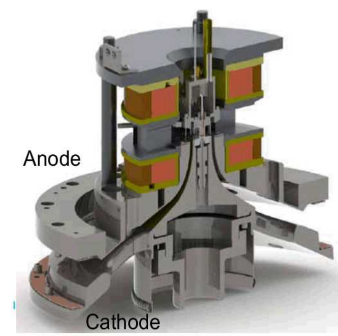
<u>DOE/NNSA and many DOE/SC Mission Drivers are Characterized by:</u>

- Complex strongly coupled physical mechanisms (**multiphysics**)
  - ➢ Strongly coupled nonlinear solvers (**Newton methods**)
  - ➢ Physics-compatible discretizations



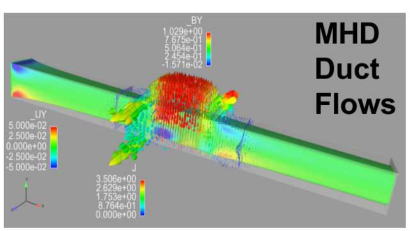
- Large range of interacting time-scales (Multiple-time-scales)
  - ➢ Implicitness (**fully-Implicit** or **implicit/explicit [IMEX]**)

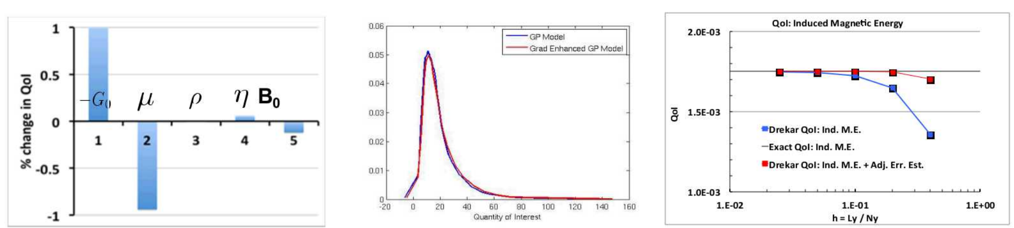
- Complex geometries, multiple length-scales, high-resolution
  - ➢ Unstructured mesh FE (HEX and TET)
  - ➢ Scalable solution (**Krylov methods, physics-based prec., AMG**)



Z Convolute Power-feed

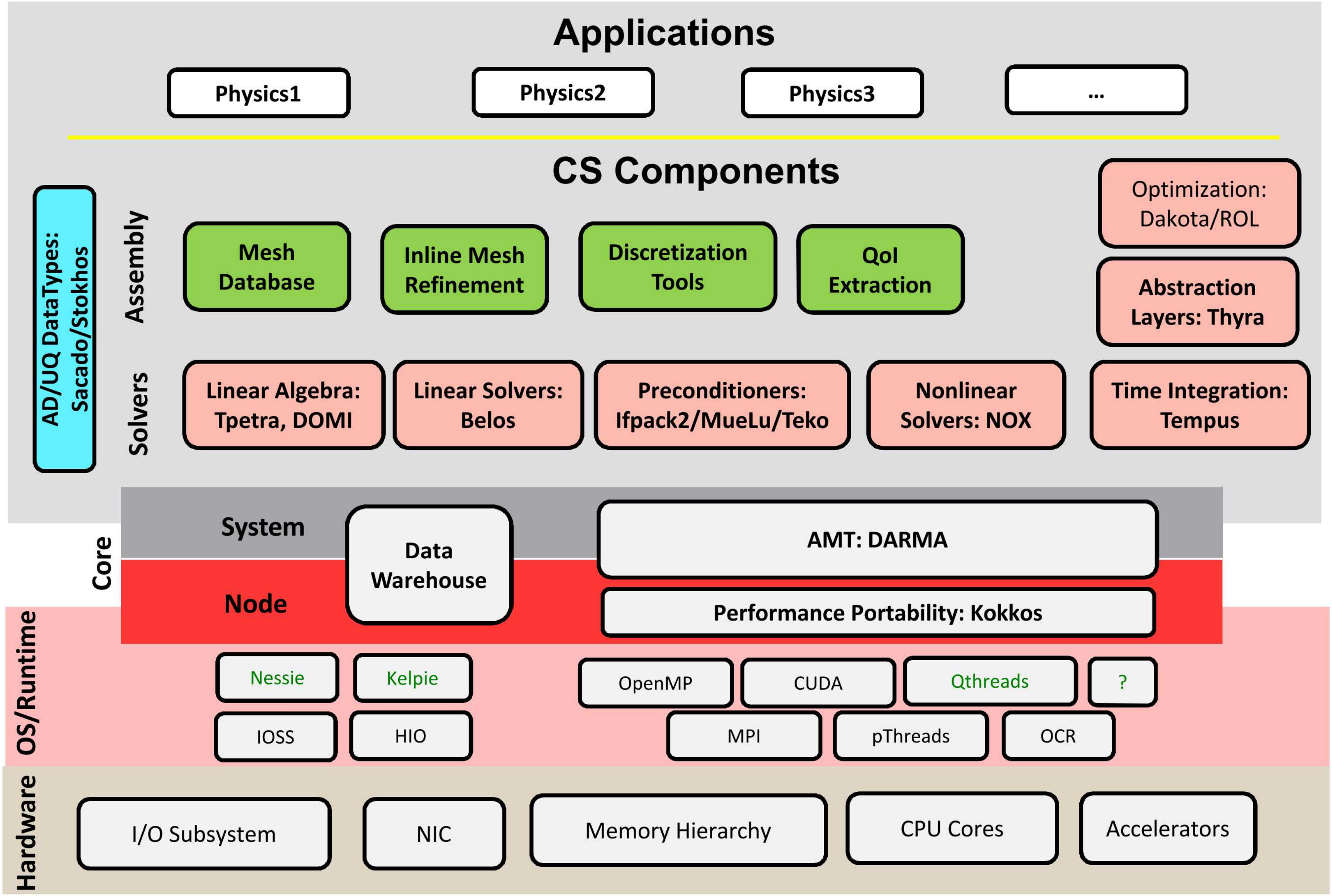
- High consequence decisions informed by modeling / simulation
  - ➢ Beyond forward simulation (**sensitivities, UQ, error est., design opt.**)





Adjoint-enabled Sensitivities, UQ surrogates, Error-estimates

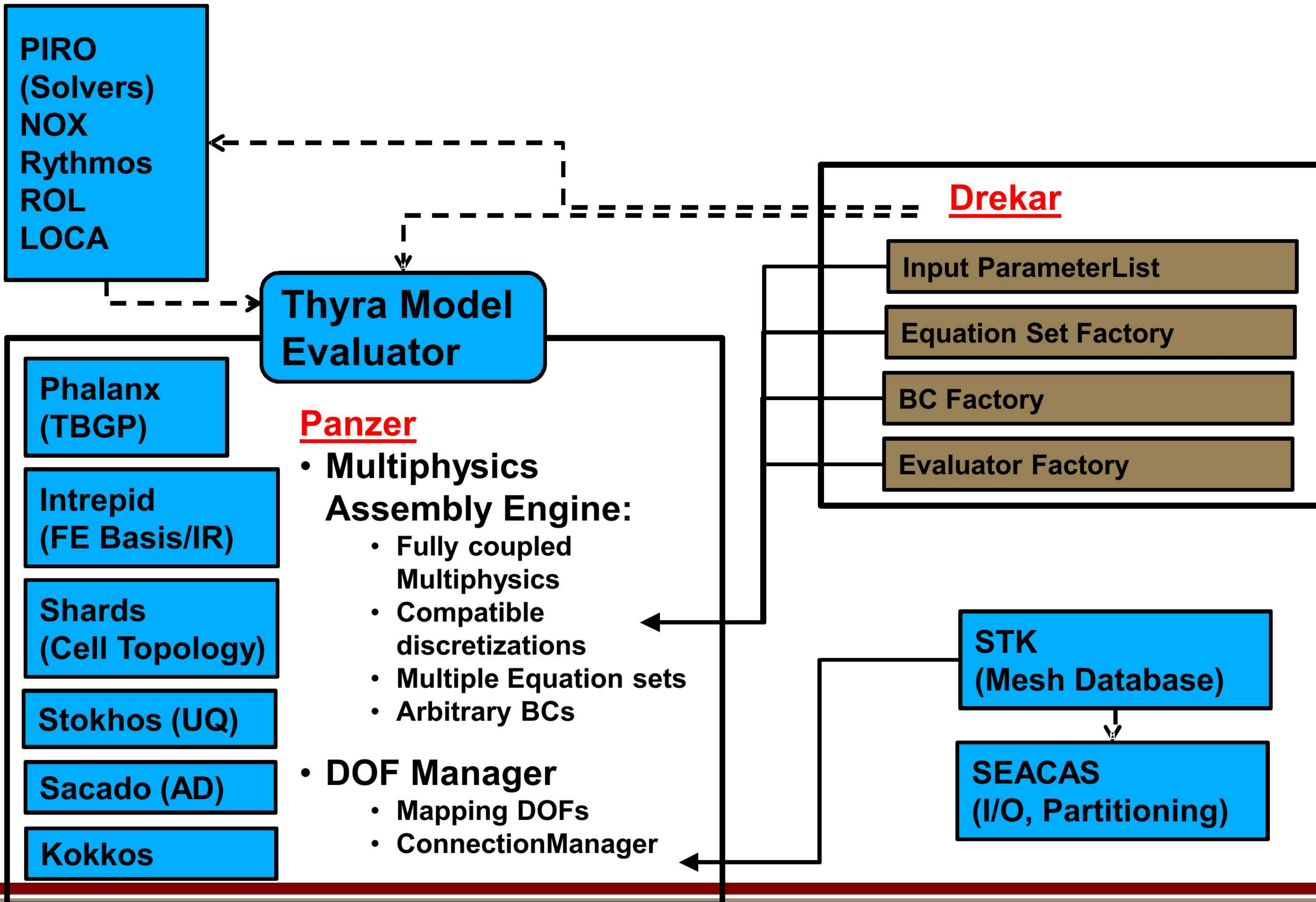
Component Architecture



The Role of Frameworks

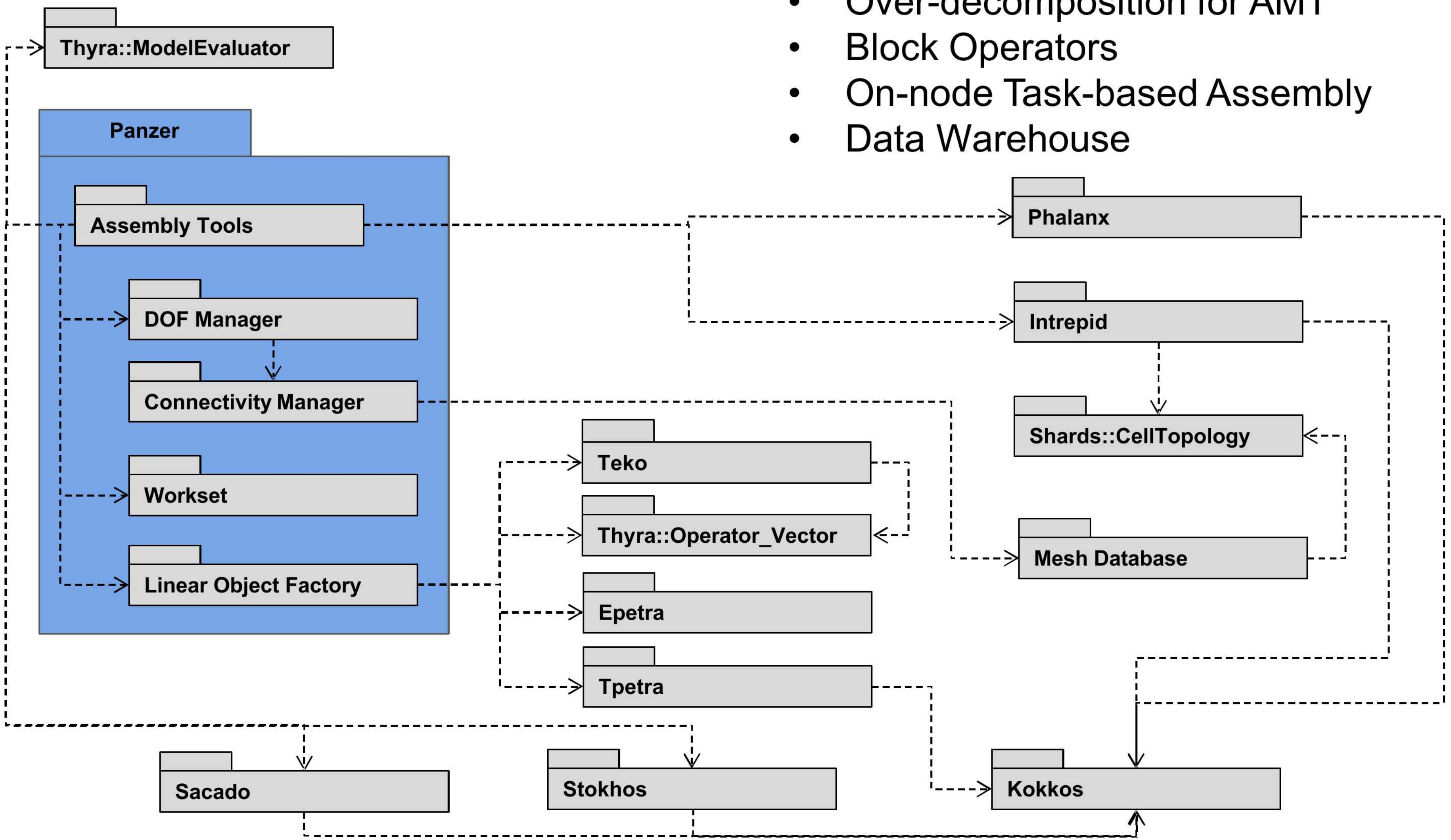
- Pros
  - Fast stand-up of capability
  - Almost a requirement for low budget projects
  - Engagement with community, feedback, improvements, long term sustainability
  - Shared maintenance
  - Focus of expertise
  - Leverage SQA practices
  - …
- Cons
  - Performance can be sacrificed for flexibility
  - General interfaces can cause confusion (supporting every nonlinear analysis type)
    - Explosion of mix-ins (multiple inheritance), all-in-one, …
  - Many changing components can lead to difficulties identifying regressions
  - …

Application Components: Trilinos Discretization Tool Stack



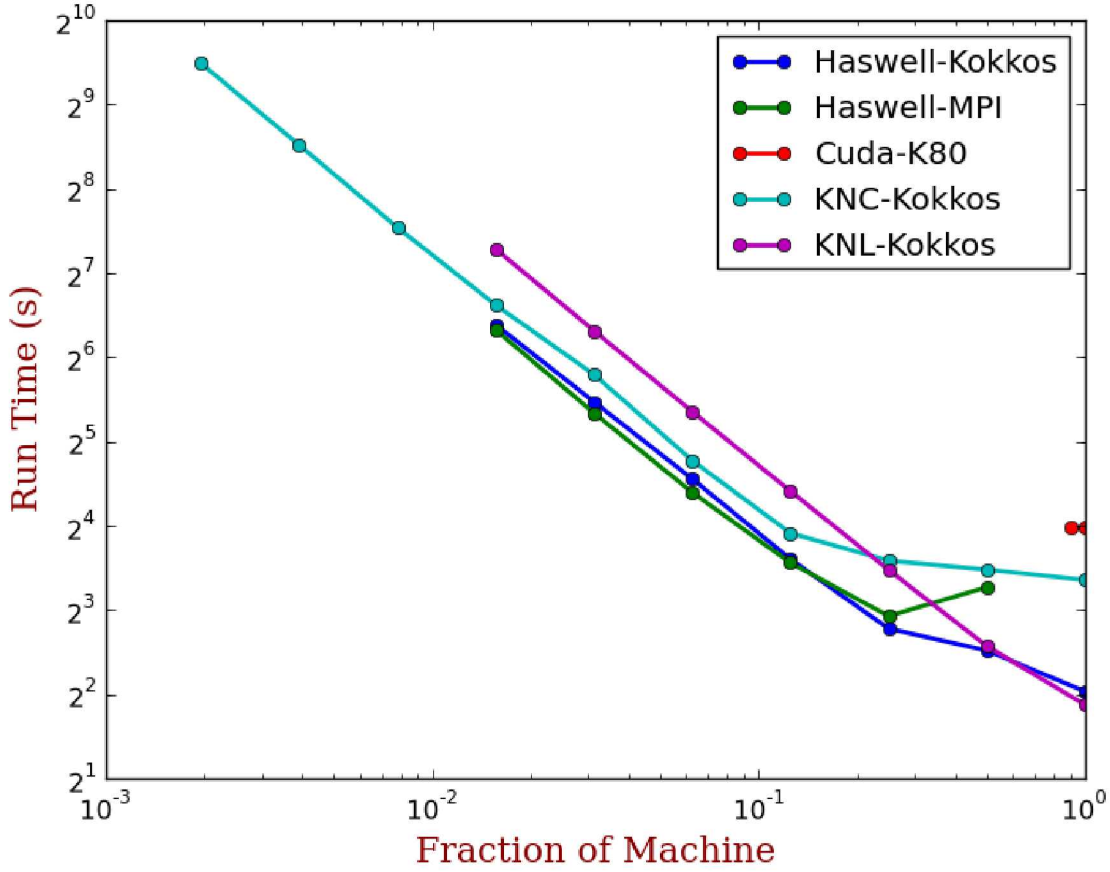
Multiphysics Discretization Tools

- DOF Indexing (Multiphysics)
- Over-decomposition for AMT
- Block Operators
- On-node Task-based Assembly
- Data Warehouse



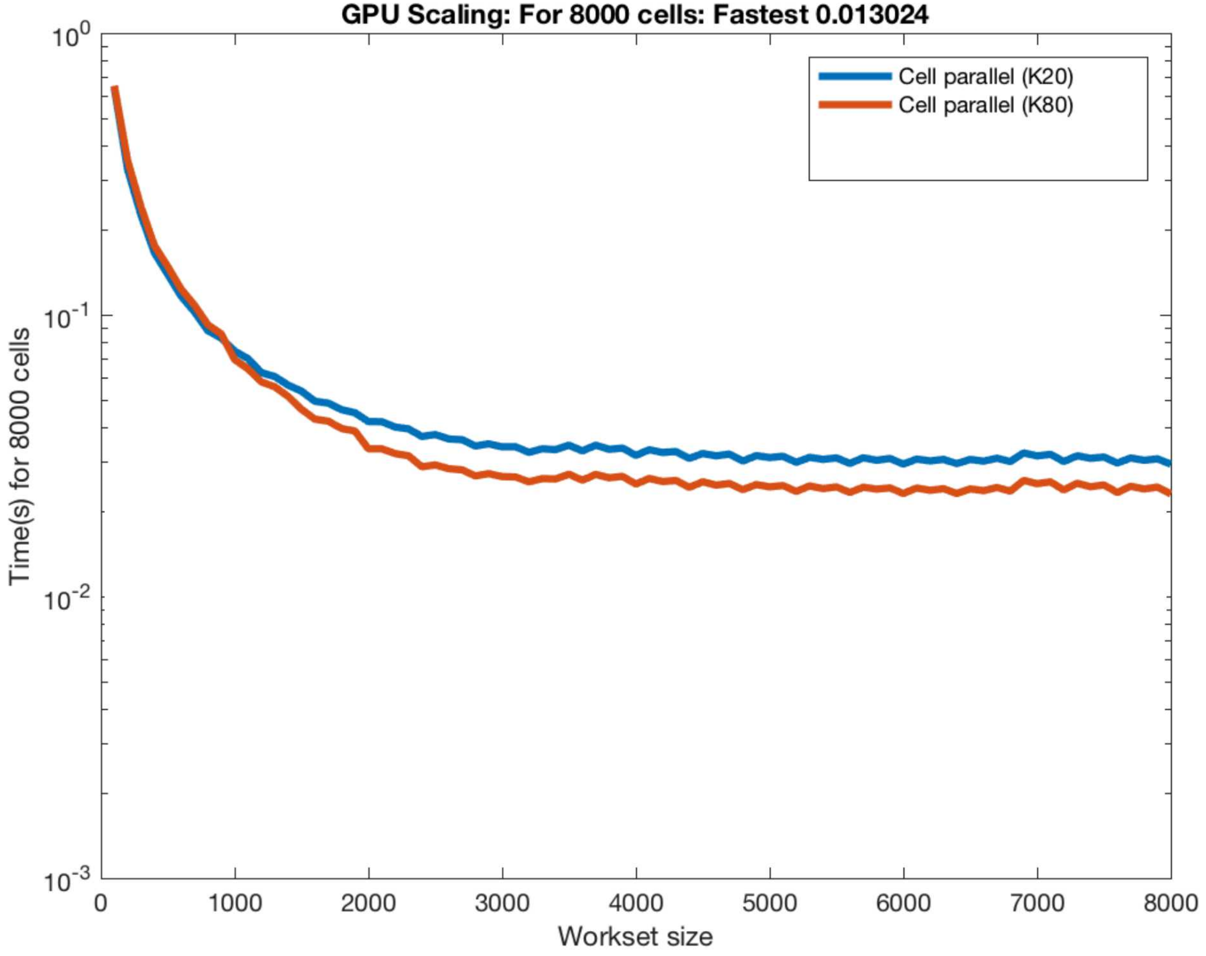
Initial Port for Jacobian Assembly

- 2016 Milestone to demonstrate the “ecosystem”
- 16K elements
- Flat/Single level data parallelism (loop over cells)
- Basic MPI (no thread spec.)



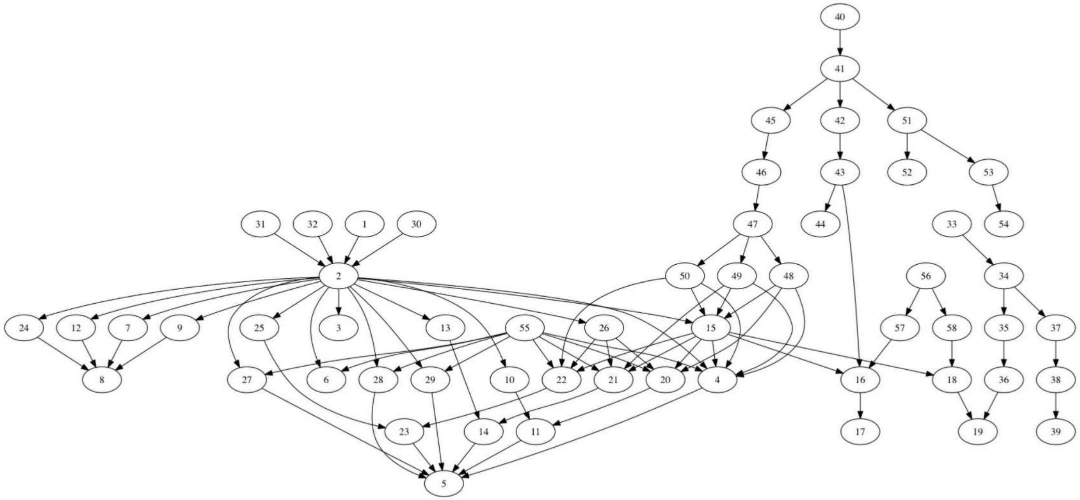
Single CFD Kernel GPU Performance Assessment

- Single level parallelism is insufficient
- Does not expose enough parallelism



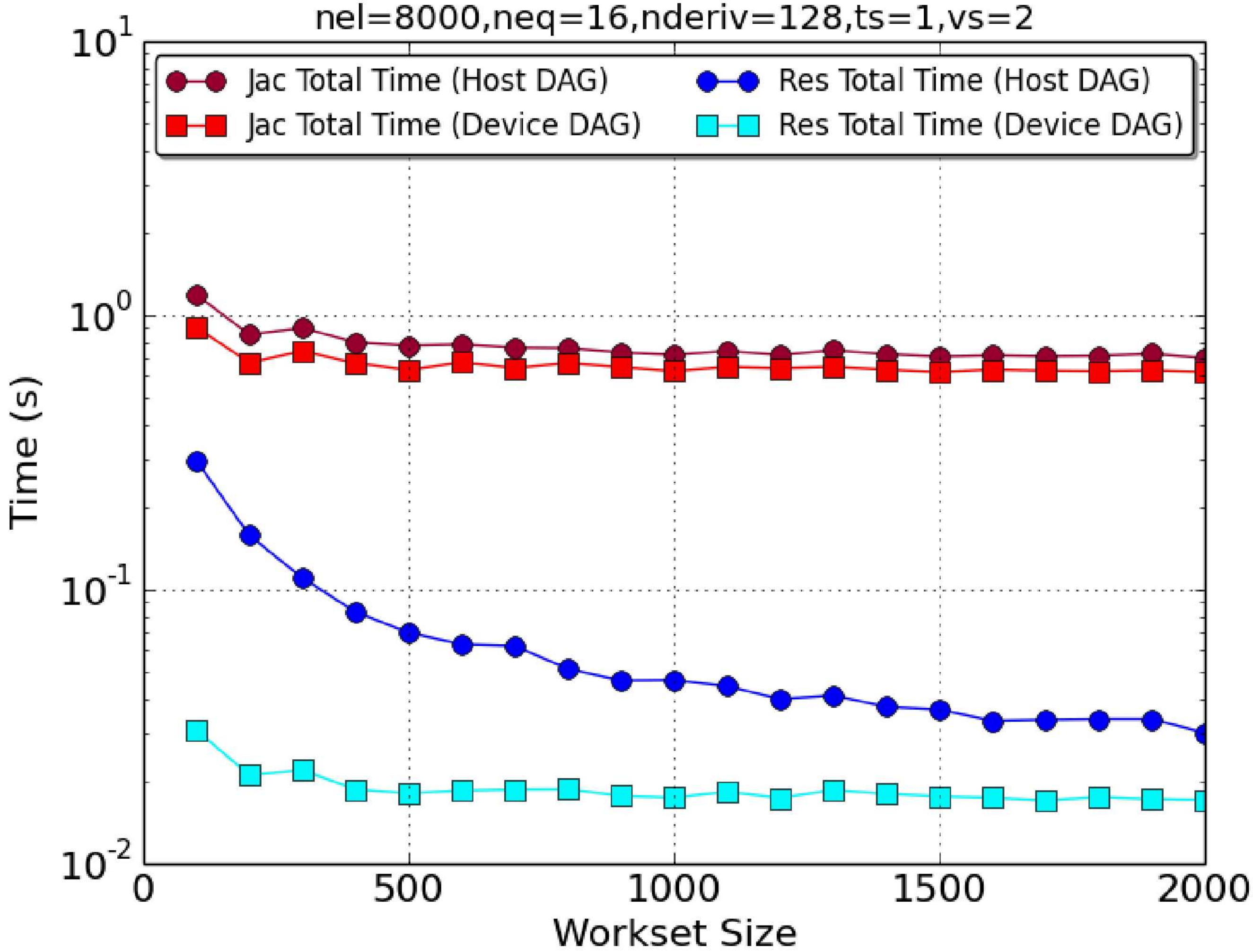
Host vs Device DAG

- **Host DAG:**
  - Each node in DAG launches its own device kernel via parallel_for from host
- **Device DAG:**
  - Single parallel_for for entire DAG
  - Goal: keep values in cache for next functor evaluation
- Device DAG implementation:
  - Need a virtual function call to run through a runtime generated list of functors
  - Copy all functors to device and instantiate
  - Requires relocatable device code (RDC) for CUDA



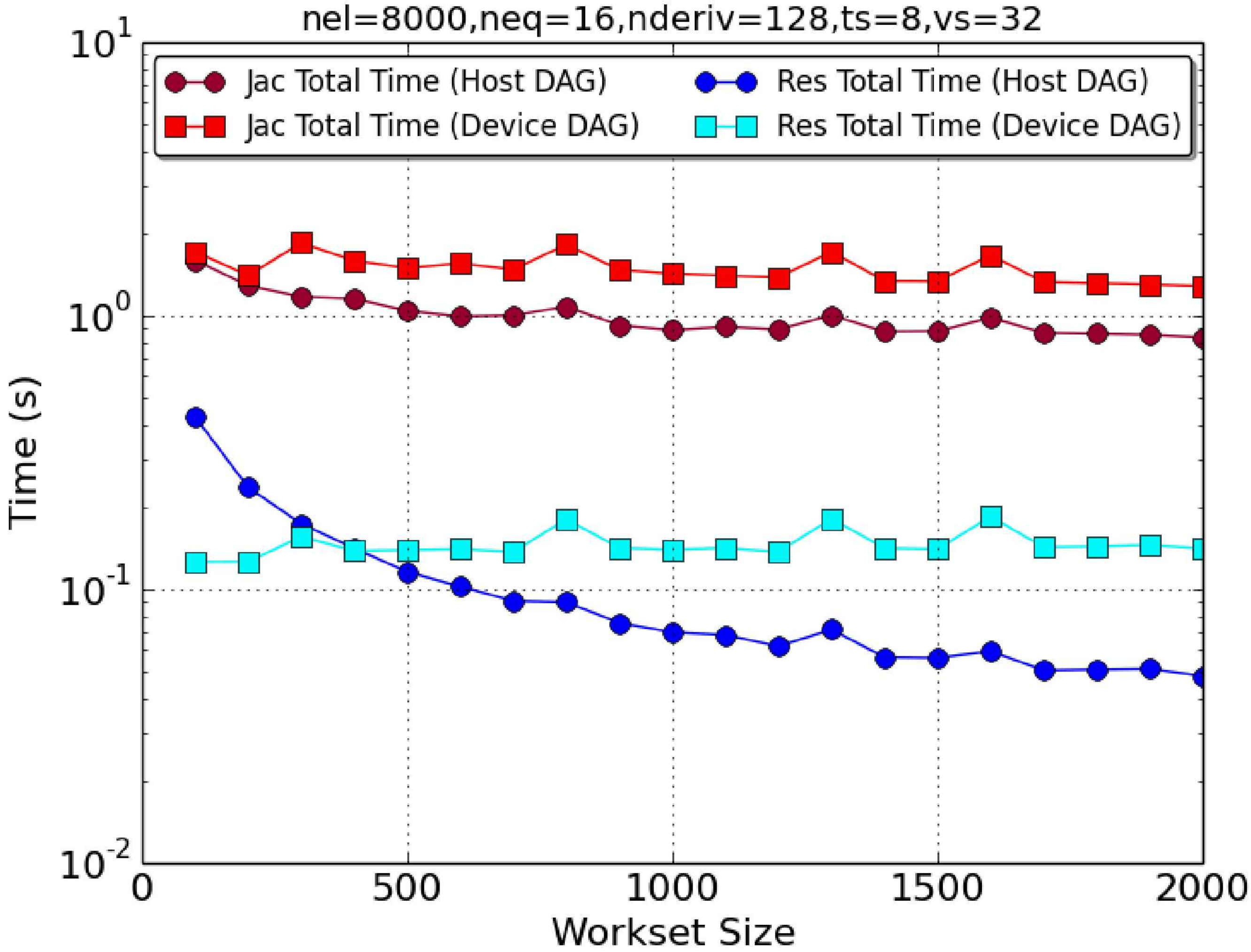
```
template<typename Traits>
struct RunDeviceDag {

  Kokkos::View<PHX::DeviceEvaluatorPtr<Traits>*,PHX::Device> evaluators_;

  ...

  KOKKOS_INLINE_FUNCTION
  void operator()(const TeamPolicy<exec_space>::member_type& team) const
  {
    const int num_evaluators = static_cast<int>(evaluators_.extent(0));
    for (int e=0; e < num_evaluators; ++e) {
      evaluators_(e).ptr->prepareForRecompute(team,data_);
      evaluators_(e).ptr->evaluate(team,data_);

    }
  }

};
```

Host vs Device DAG Performance, 16 Equations



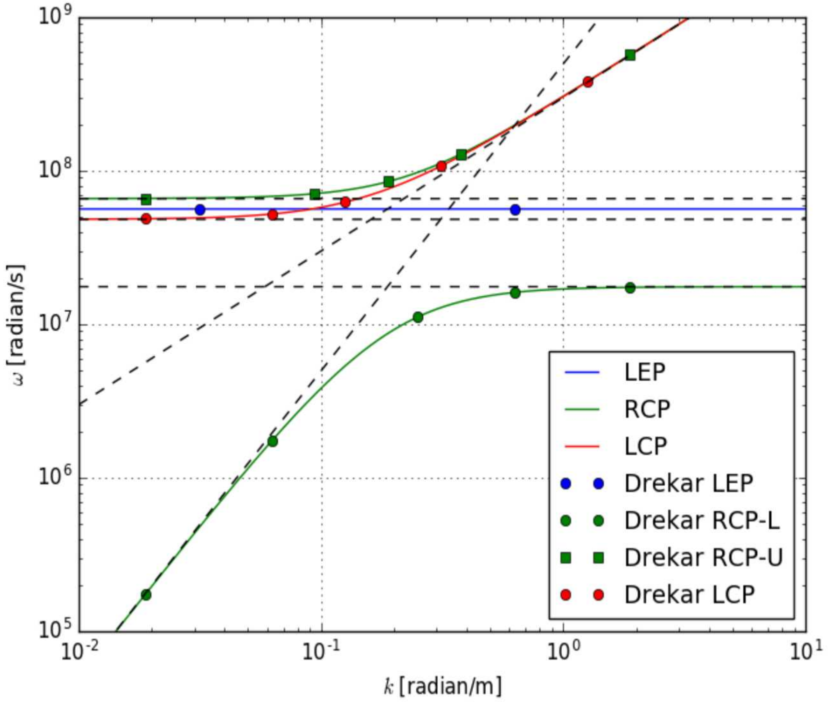
- **OpenMP on Broadwell** node, team size=1, vector size = 2
- Performance gains for residual (significant) and Jacobian (minor)

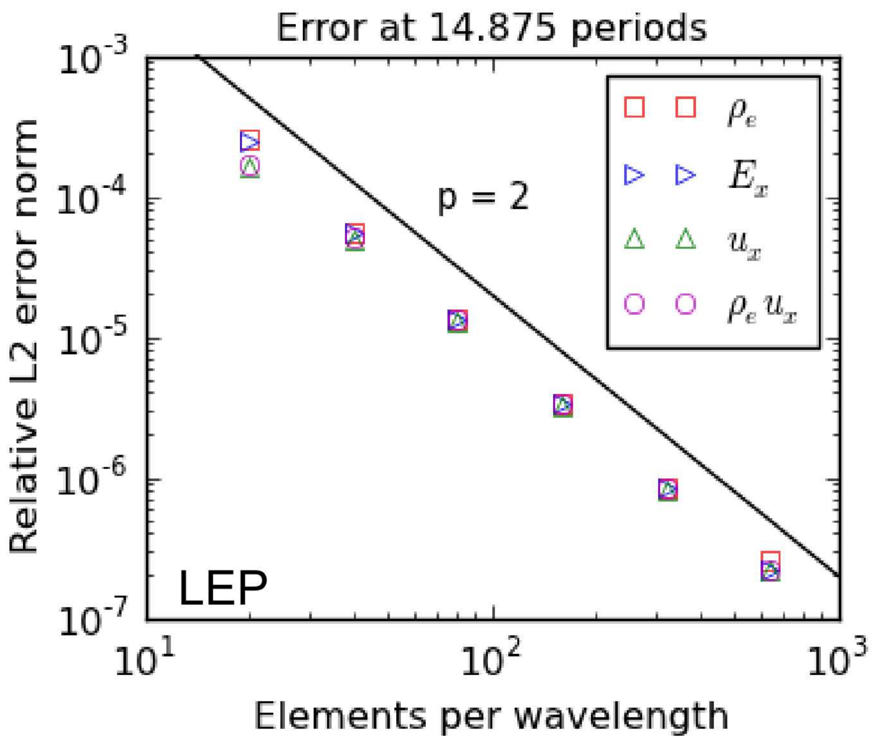
Host vs Device DAG Performance, 16 Equations



- CUDA P100, team size=8, vector size = 32, 128 derivatives
- Significant performance loss

- Demonstration / Verification of Implicit Solution for Longitudinal Electron Plasma (LEP) Oscillation with Under-resolved EM Waves



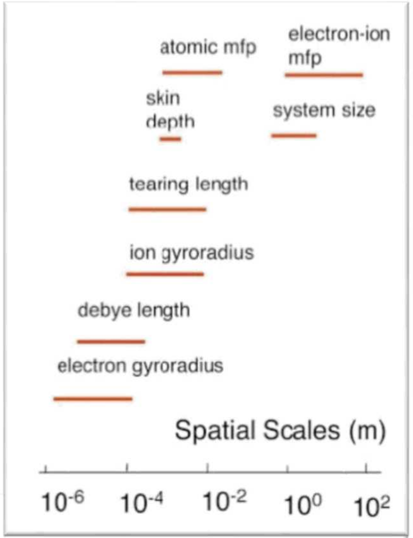


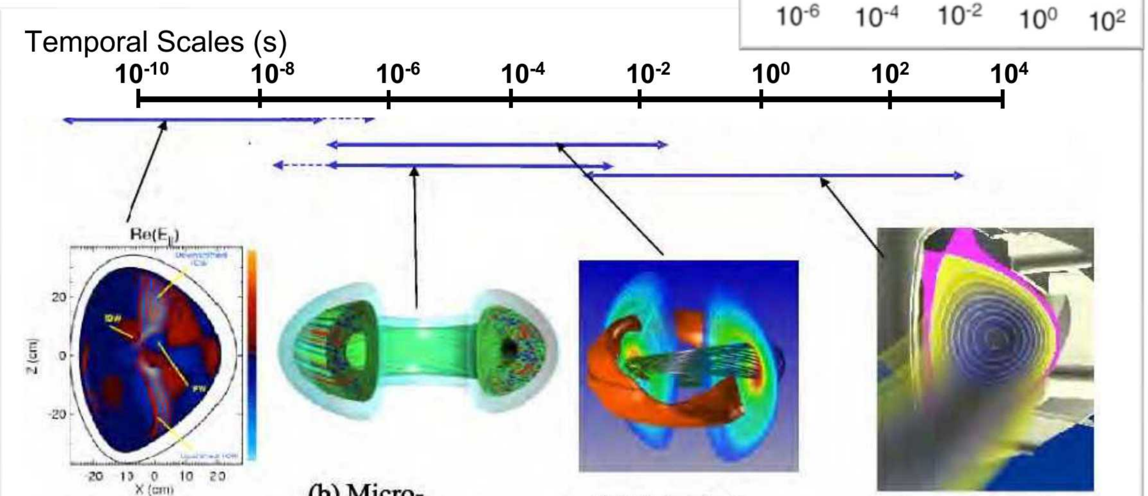
LEP: Longitudinal Electron Plasma Wave
RCP: Right Hand Circularly Polarized Wave
LCP: Left Hand Circularly Polarized Wave
(Cold plasma)

Verification effort with Niederhaus, Radtke, Bettencourt, Cartwright, Kramer, Robinson and ATDM EMPIRE Team

Multiphysics kinetic transport models are particularly ripe for algorithmic development

- Physics models interact in strongly nonlinear way
- Models and/or regions in physical space require kinetic treatment (e.g., Boltzmann): transport in phase space
- Naturally hierarchical formulation: velocity moments contract phase space into macroscopic quantities (e.g., density, momentum, …)
- Longer length-scales / time-scales needed to understand macroscopic instabilities and performance
- Development of moment-based scale-bridging algorithms that embrace heterogeneous architectures is needed





(a) RF codes (b) Micro-turbulence codes (c) Extended-MHD codes (d) Transport Codes

Courtesy: Chacon, Hittinger, Shadid, Wildey (ASCR PI Meeting)

Multi-fluid plasma model

- Continuity equation:

$$\partial_t \rho_\alpha + \nabla \cdot (\rho_\alpha \boldsymbol{u}_\alpha) = S_\alpha$$

Each species α is represented by a separate density ρ, momentum $\rho\boldsymbol{u}$, and isotropic energy ϵ.

- Momentum equation:

$$\partial_t (\rho_\alpha \boldsymbol{u}_\alpha) + \nabla \cdot (\rho_\alpha \boldsymbol{u}_\alpha \otimes \boldsymbol{u}_\alpha + \boldsymbol{P}_\alpha) = \frac{q_\alpha}{m_\alpha} \rho_\alpha (\boldsymbol{E} + \boldsymbol{u}_\alpha \times \boldsymbol{B}) + \boldsymbol{R}_\alpha + \boldsymbol{u}_\alpha S_\alpha$$

- Energy equation:

$$\partial_t \epsilon_\alpha + \nabla \cdot (\boldsymbol{u}_\alpha \cdot (\epsilon_\alpha \boldsymbol{I} + \boldsymbol{P}_\alpha) + \boldsymbol{q}_\alpha) = \frac{q_\alpha}{m_\alpha} \rho_\alpha \boldsymbol{u}_\alpha \cdot \boldsymbol{E} + Q_\alpha + \boldsymbol{u}_\alpha \cdot \boldsymbol{R}_\alpha + \frac{1}{2} \boldsymbol{u}_\alpha^2 S_\alpha$$

- Ampere's Law:

$$\partial_t \boldsymbol{E} - c^2 \nabla \times \boldsymbol{B} = -\frac{1}{\epsilon_0} \sum_\alpha \frac{q_\alpha}{m_\alpha} \rho_\alpha \boldsymbol{u}_\alpha$$

Spatial operators are discretized using a finite element method.

- Faraday's Law:

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

Fluid
Electromagnetic
Inter-fluid

IMEX time integration

- IMEX gives a framework for splitting the model up into implicit and explicit terms:
  - Explicit for slow, non-stiff terms
  - Implicit for fast, stiff terms

Implicit tableau

$$\begin{array}{c|c} c & A \\ \hline & b^t \end{array}$$

Explicit tableau

$$\begin{array}{c|c} \hat{c} & \hat{A} \\ \hline & \hat{b}^t \end{array}$$

$$\partial_t u = f(u,t) + g(u,t)$$

$$u^{(i)} = u^n + \Delta t \sum_{j=0}^{j<i} \hat{A}_{ij} f\left(u^{(j)}, t_n + \hat{c}_j \Delta t\right) + \Delta t \sum_{j=0}^{j\leq i} A_{ij} g\left(u^{(j)}, t_n + c_j \Delta t\right)$$

$$u^{n+1} = u^n + \Delta t \sum_{i=0}^{i<s} \hat{b}_i f\left(u^{(i)}, t_n + \hat{c}_i \Delta t\right) + \Delta t \sum_{i=0}^{i\leq s} b_i g\left(u^{(i)}, t_n + c_i \Delta t\right)$$

- **Objective**: Combine the advantages of implicit and explicit solvers.
  - Take advantage of expensive implicit solver to overstep fast scales, and explicit solver to resolve slow scales.

IMEX splitting for CG

Each operator is associated with one or more plasma scales, which are grouped by color representing their approximate explicit stability limits.

$$\partial_t \rho_\alpha + \boldsymbol{u}_\alpha \cdot \nabla \rho_\alpha = -\rho_\alpha \nabla \cdot \boldsymbol{u}_\alpha$$

$$u_\alpha < \frac{\Delta x}{\Delta t}$$

$$\partial_t \boldsymbol{u}_\alpha + \boldsymbol{u}_\alpha \cdot \nabla \boldsymbol{u}_\alpha = -\boldsymbol{u}_\alpha \nabla \cdot \boldsymbol{u}_\alpha - \frac{1}{\rho_\alpha} \nabla P_\alpha + \frac{1}{\rho_\alpha} \nabla \cdot \left(\mu_\alpha \left(\nabla \boldsymbol{u}_\alpha + \nabla \boldsymbol{u}_\alpha^T - \frac{2}{3} \boldsymbol{I} \nabla \cdot \boldsymbol{u}_\alpha \right) \right)$$

$$u_\alpha < \frac{\Delta x}{\Delta t} \qquad v_{s\alpha} < \frac{\Delta x}{\Delta t} \qquad v_\alpha < \frac{\Delta x^2}{\Delta t}$$

$$+ \frac{q_\alpha}{m_\alpha} \boldsymbol{E} + \frac{q_\alpha}{m_\alpha} \boldsymbol{u}_\alpha \times \boldsymbol{B} - \sum_\beta \nu_{\alpha\beta} (\boldsymbol{u}_\alpha - \boldsymbol{u}_\beta)$$

$$\omega_{p\alpha} \Delta t < 1 \qquad \omega_{c\alpha} \Delta t < 1 \qquad \nu_{\alpha\beta} \Delta t < 1$$

$$\partial_t P_\alpha + \boldsymbol{u}_\alpha \cdot \nabla P_\alpha = -\gamma P_\alpha \nabla \cdot \boldsymbol{u}_\alpha + \nabla \cdot ((\gamma - 1) k_\alpha \nabla T_\alpha) - \sum_\beta \frac{(\gamma - 1)\nu_{\alpha\beta}\rho_\alpha}{m_\alpha + m_\beta} \left(3(T_\alpha - T_\beta) - m_\beta (\boldsymbol{u}_\alpha - \boldsymbol{u}_\beta)^2\right)$$

$$u_\alpha < \frac{\Delta x}{\Delta t} \qquad \kappa_\alpha < \frac{\Delta x^2}{\Delta t} \qquad \nu_{\alpha\beta} \Delta t < 1$$

$$\partial_t \boldsymbol{E} - c^2 \nabla \times \boldsymbol{B} = -\frac{1}{\epsilon_0} \sum_\alpha \frac{q_\alpha}{m_\alpha} \rho_\alpha \boldsymbol{u}_\alpha$$

$$c < \frac{\Delta x}{\Delta t} \qquad \omega_{p\alpha} \Delta t < 1$$

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

For IMEX-CG each operator can be moved between implicit and explicit evaluation depending on the explicit stability limits.

Compatible discretization for EM

- A physics compatible finite element discretization is used to enforce the divergence constraints for the electric and magnetic fields.
- Fluids are represented by an **HGrad** (node) basis $\rho \in V_{\nabla}$.
- The electric field is represented by an **HCurl** (edge) vector basis $\mathbf{E} \in \boldsymbol{V}_{\nabla\times}$.
- The magnetic field is represented by an **HDiv** (face) vector basis $\mathbf{B} \in \boldsymbol{V}_{\nabla\cdot}$.
- Compatibility is defined by the discrete preservation of the **De Rham Complex**:

$$\nabla \phi_{\nabla} \in \boldsymbol{V}_{\nabla\times} \longrightarrow \nabla \times \boldsymbol{\phi}_{\nabla\times} \in \boldsymbol{V}_{\nabla\cdot} \longrightarrow \nabla \cdot \boldsymbol{\phi}_{\nabla\cdot} \in V_{\mathrm{L}_2}$$

- For Faraday's law, we choose a basis for the electric field such that its curl is fully represented by the basis used by the magnetic field.

$$\partial_t \boldsymbol{B} + \nabla \times \boldsymbol{E} = 0 \qquad \nabla \cdot \boldsymbol{B} = 0$$

- Since the curl of the electric field is 'globally continuous' w.r.t. a divergence operator, the divergence of that curl is zero over the domain:

$$\nabla \cdot (\partial_t \boldsymbol{B} + \nabla \times \boldsymbol{E}) = \partial_t(\nabla \cdot \boldsymbol{B}) + \nabla \cdot \nabla \times \boldsymbol{E} = \partial_t(\nabla \cdot \boldsymbol{B}) + \sum_i E_i \underbrace{\nabla \cdot \nabla \times \boldsymbol{\phi}^i_{\nabla\times}}_{0} = \partial_t(\nabla \cdot \boldsymbol{B}) = 0$$

- **Result**: The curl operator does not add divergence errors to the magnetic field

Satisfying Gauss' laws in plasmas

- **Goal**: Solve plasma-coupled Maxwell's equations and satisfy a divergence constraint:

$$\partial_t \boldsymbol{E} - c^2 \nabla \times \boldsymbol{B} = -\frac{1}{\epsilon_0}\boldsymbol{j} \qquad \partial_t \rho_c + \nabla \cdot \boldsymbol{j} = 0$$

$$\nabla \cdot \boldsymbol{E} = \frac{1}{\epsilon_0}\rho_c$$

- In the **strong, non-discretized form**:

$$\nabla \cdot \left(\partial_t \boldsymbol{E} + \frac{1}{\epsilon_0}\boldsymbol{j} - c^2 \nabla \times \boldsymbol{B} \right) = \partial_t \nabla \cdot \boldsymbol{E} + \frac{1}{\epsilon_0} \nabla \cdot \boldsymbol{j} = \partial_t \left(\nabla \cdot \boldsymbol{E} - \frac{1}{\epsilon_0}\rho_c \right) = 0$$

- In the **weak form**: Choose a basis that supports the divergence constraint as HCurl does not support the divergence operation:

$$\int_\Omega \left(\partial_t \boldsymbol{E} - c^2 \nabla \times \boldsymbol{B} + \frac{1}{\epsilon_0}\boldsymbol{j} \right) \cdot \nabla \phi_\nabla \, dV = \int_\Omega \left(\partial_t \boldsymbol{E} \cdot \nabla \phi_\nabla + \frac{1}{\epsilon_0} \nabla \cdot \boldsymbol{j} \, \phi_\nabla \right) dV + c^2 \int_\Omega \underbrace{\boldsymbol{B} \cdot \nabla \times \nabla \phi_\nabla}_{0} dV$$

$$= \int_\Omega \partial_t \left(\boldsymbol{E} \cdot \nabla \phi_\nabla - \frac{1}{\epsilon_0} \rho_c \, \phi_\nabla \right) dV = 0$$

- Assumes that continuity equation is weakly satisfied:

$$\int_\Omega (\partial_t \rho_c - \nabla \cdot \boldsymbol{j}) \phi_\nabla \, dV = \int_\Omega (\partial_t \rho_c \phi_\nabla + \boldsymbol{j} \cdot \nabla \phi_\nabla) dV = 0 \rightarrow \int_\Omega \partial_t \rho_c \phi_\nabla \, dV = -\int_\Omega \boldsymbol{j} \cdot \nabla \phi_\nabla \, dV$$

Discontinuous Galerkin method

- Discontinuous Galerkin FEM does not assume a globally continuous test function:

Weak form

$$\int_\Omega \phi \partial_t u \, dV + \int_\Omega \phi \, \nabla \cdot \boldsymbol{f} \, dV - \int_\Omega \phi s \, dV = 0$$

Break into elements $\mathrm{K} \in \Omega$ with discontinuous element test function ϕ_i^{K}

$$\sum_K \left[\int_{\mathrm{K}} \phi_i^K \partial_t u \, dV + \int_{\mathrm{K}} \phi_i^K \nabla \cdot \boldsymbol{f} dV - \int_{\mathrm{K}} \phi_i^K s \, dV \right] = 0$$

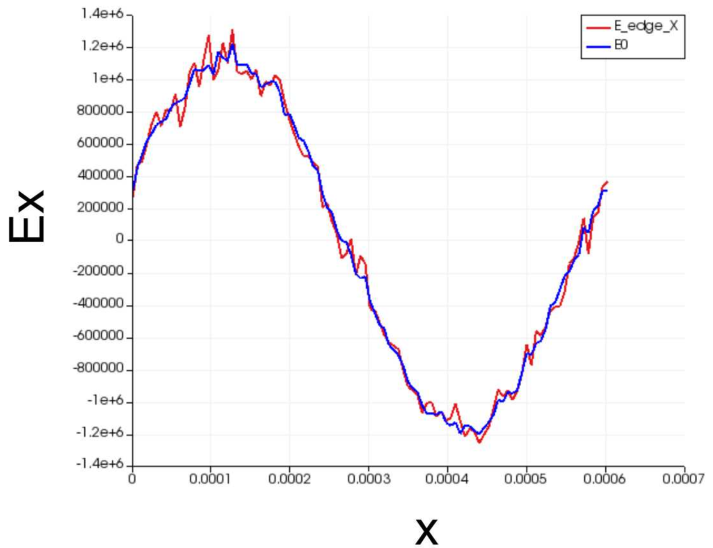
Apply divergence theorem to flux integral

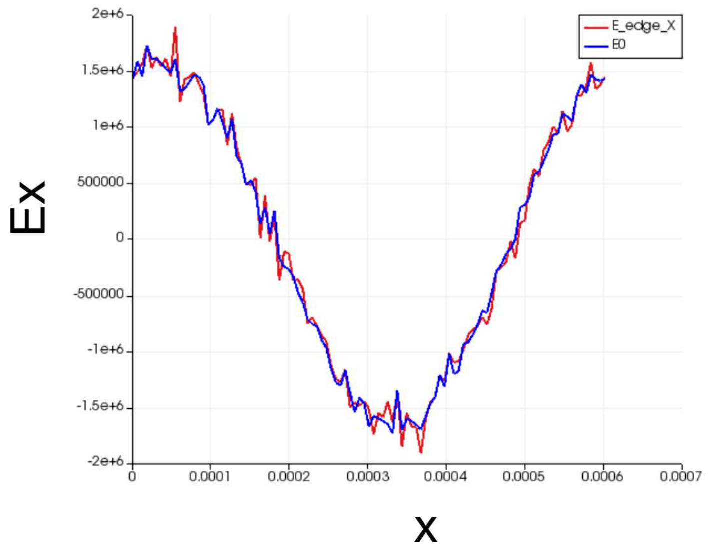
$$\int_{\mathrm{K}} \phi_i^K \partial_t u \, dV + \oint_{\partial \mathrm{K}} \phi_i^K \hat{\boldsymbol{n}} \cdot \boldsymbol{f} \, dS - \int_{\mathrm{K}} \boldsymbol{f} \cdot \nabla \phi_i^K dV - \int_{\mathrm{K}} \phi_i^K s \, dV = 0$$

- **Consistency**: Fluxes must be single valued on interfaces between elements.
  - **Numerical Flux**: Solution to Riemann problem to generate consistent flux on interfaces.

Drekar-PIC Coupling Demonstration

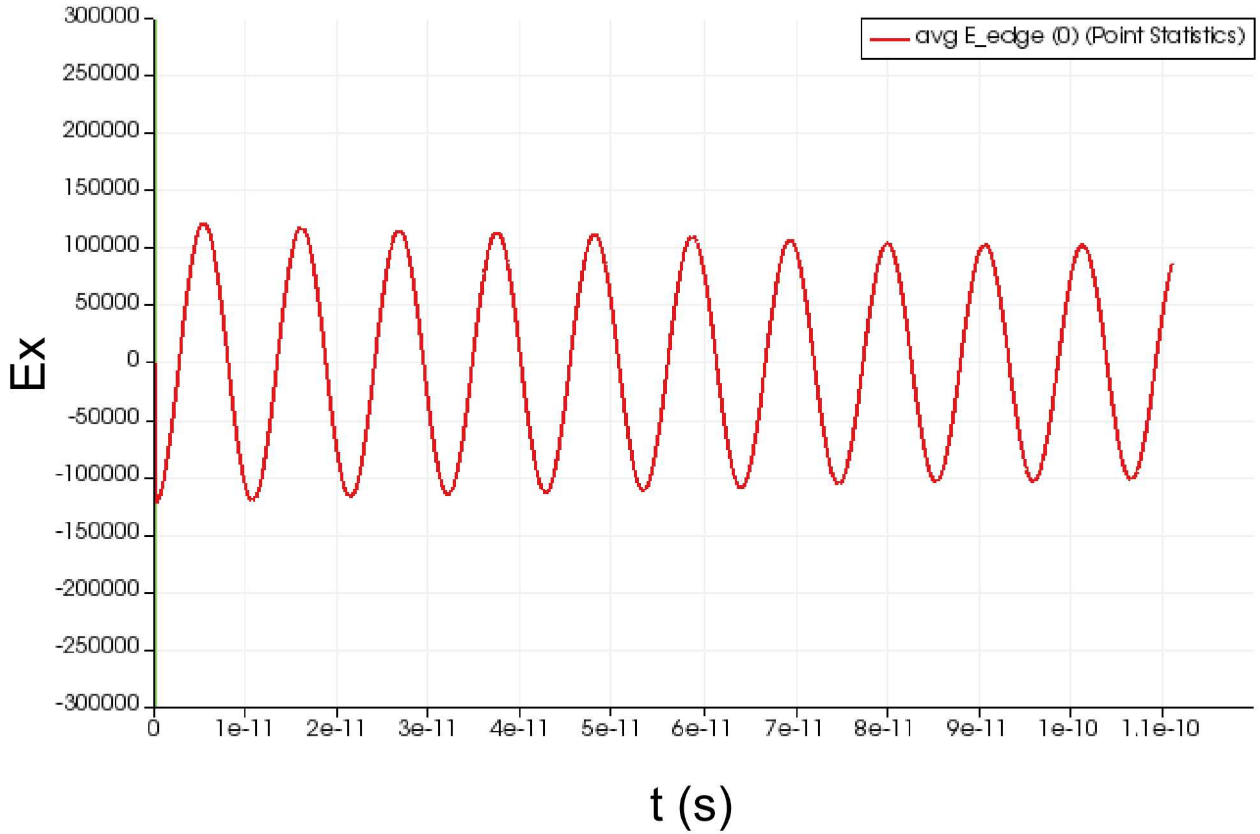
- First coupling of Drekar to EMPIRE-PIC for a Langmuir Wave
- Simple proof-of-principle (Drekar Electrostatic potential, EMPIRE-PIC electrons)
- Replace PIC EM Solver with Drekar EM
- CG fluid, 1024 cells, 9K particles
- Plots show Ex line plot over spatial domain: red is coupled solution (cell avg.), blue is EM-PIC standalone solution at nodes.
- Good agreement





Verification Example: Ion/Electron Plasma Oscillation

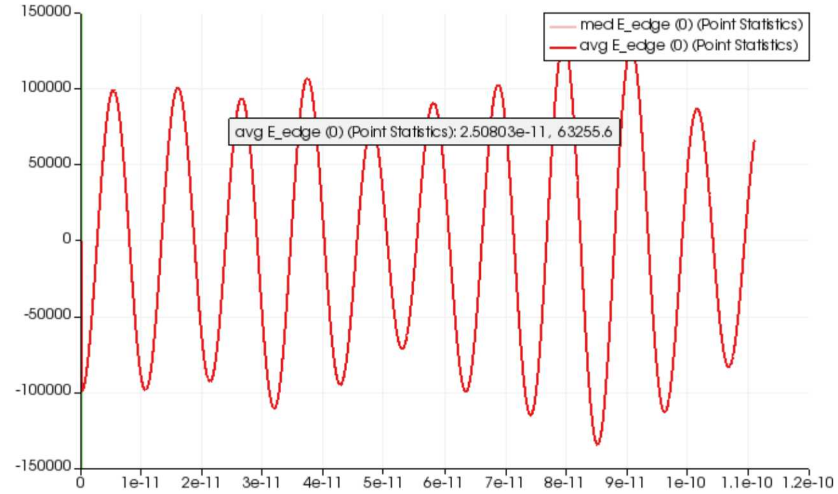
- Coupled fluid (electrons) and PIC (ions)
  - N = 1e+20
  - 16384 particles
  - 32x2 mesh
- Simplified problem for ES formulation
- Theory period: 1.06192e-11
- Simulation period: 1.0593e-11
  - 0.25% error
- New results, just delving into accuracy



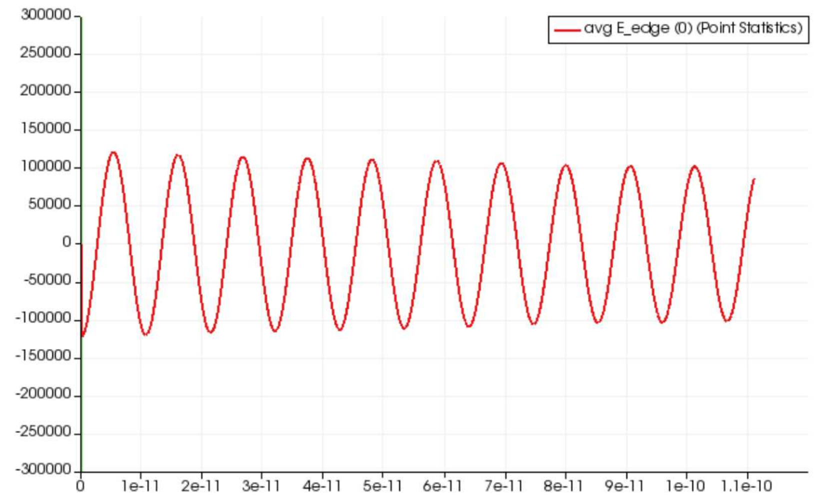
Period Computation

- Frequency/Period values computed using FFT
- Time history data can be sampled from a single point in space or using average value from line across mesh (half plane)
    - 1D problem in a 2D code

Single Point (0.000175, 0.0)

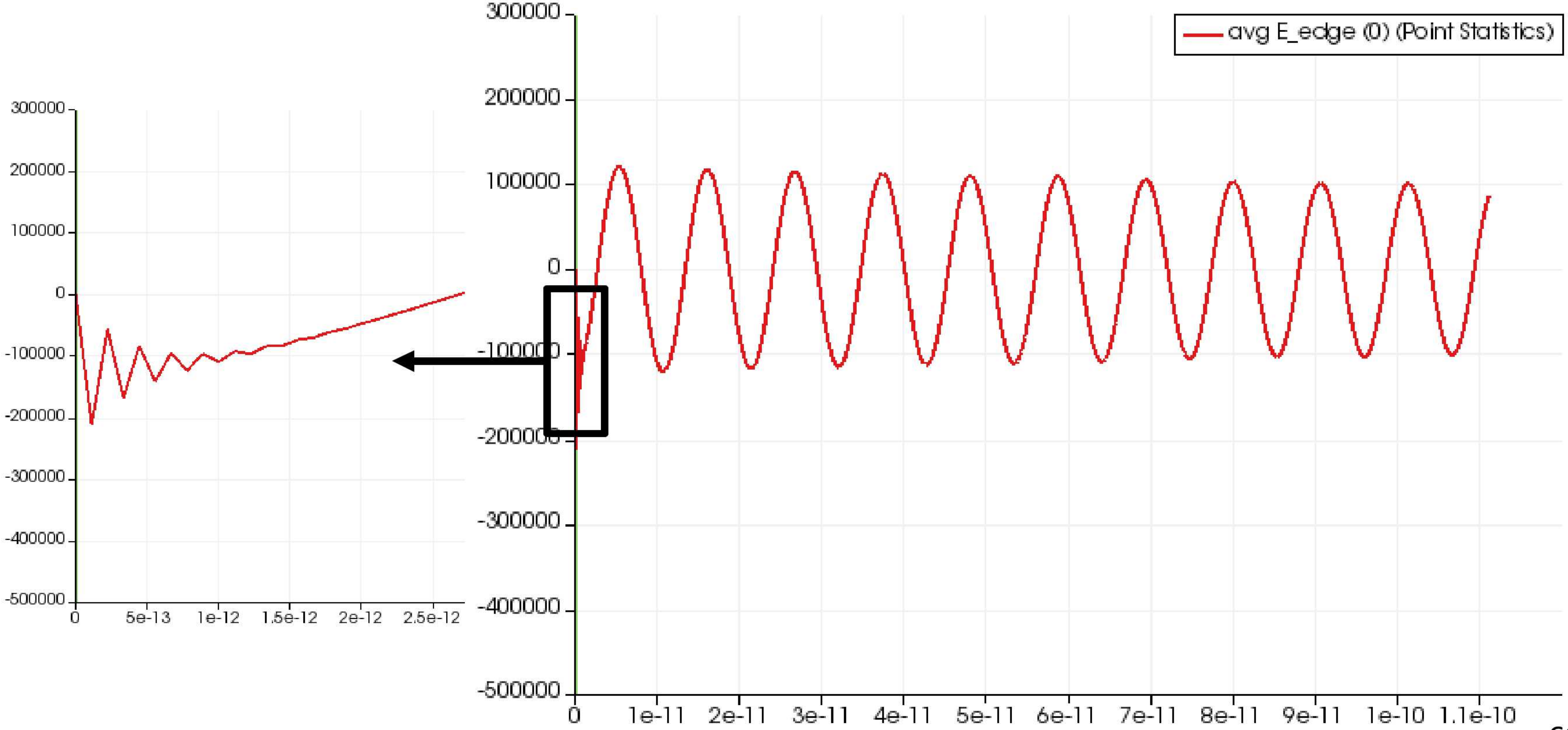


Line Average (center to edge)



Time Integration Effects

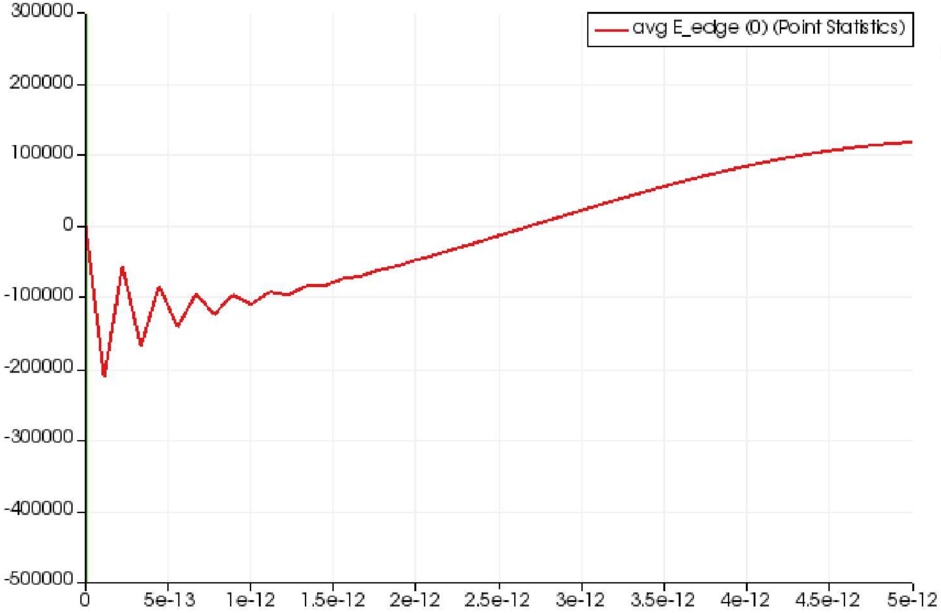
- Can we control stability in fluid solver for coupled system?

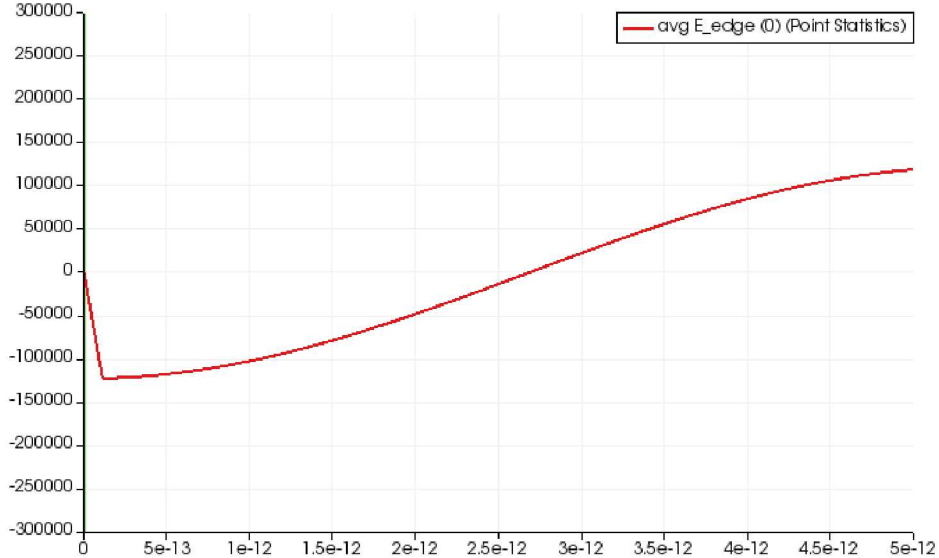


Time Integrator

- Try an L-Stable integrator
  - DIRK 2 stage,2nd Order: L-Stable
  - DIRK 2 stage, 3rd Order: A-Stable
- L-Stability damps out startup oscillations in E-Field
- Simulations are indistinguishable except for short startup.
- Same period values from FFT
- Confidence that we can control part of stability for the fluid and EM

Integrator	Stability	Period	% Error
DIRK 2nd Order, 2 stage	L-Stable	1.1111E-11	4.63E+00
DIRK 3rd Order, 2 stage	A-Stable	1.1111E-11	4.63E+00





Time Step Size

- Operator split solve with implicit electons (fluid) and explicit ions (PIC)
- Stability of ion plasma frequency
- Coupling of Ion Momentum and Energy to Ampere's law
- Failure exhibited by divergence of the Nonlinear solver

Fluid: Implicit Electrons

PIC: Explicit Ions

dt	Result	w_p_e * dt	w_p_i * dt
1.11E-13	Converged	6.26E-02	1.98E-02
2.22E-13	Converged	1.25E-01	3.96E-02
4.44E-13	Converged	2.50E-01	7.92E-02
8.88E-13	Converged	5.01E-01	1.58E-01
2.22E-12	Converged	1.25E+00	3.96E-01
4.44E-12	Converged	2.50E+00	7.92E-01
8.88E-12	Failed	5.01E+00	1.58E+00