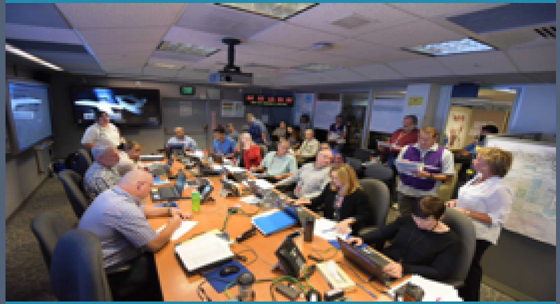
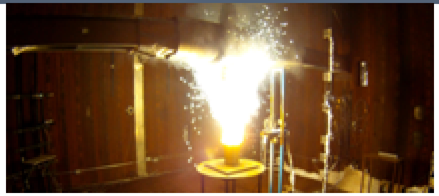
Sandia National Laboratories

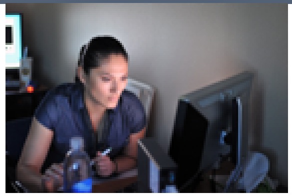
SAND2021-4581C

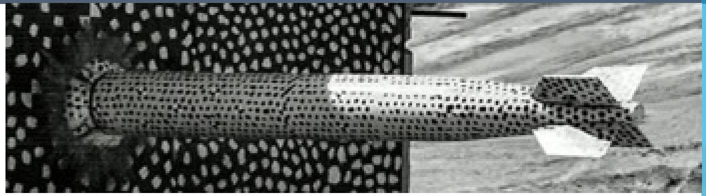
Sandia ATDM: Performance and Portability Accomplishments

April 15, 2021









PRESENTED BY

Bill Rider for the SNL **ATDM Team,** SAND2021-????

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.

SNL ATDM Team Members

Program Management: Jim Stewart, Mike Glass, Bill Rider, Paul Crozier, Richard Kramer, Ken Alvin

EMPIRE PI's: Matt Bettencourt, Eric Cyr
EMPIRE PO: Keith Cartwright
EMPIRE Team:
Kris Beckwith, Adam Blake, Wade Burgess, David Collins, Jason Gates, Daniel Jensen, Paul Lin (now at LBNL), Ed Love, Billy McDoniel, Duncan McGregor, Brandon Medina, Logan Meredith, Sean Miller, Chris Moore, Stan Moore, Edward Philips, Tim Pointon, Troy Powell, Gregg Radtke, Nick Roberds, Sidney Shields, David Sirajuddin, Tom Smith, Scot Swan, Andrew Styer, David Turner

SPARC PI: Micah Howard
SPARC Team:
Anthony Agelastos, Matt Barone, Matt Bopp, Steve Bova, Sam Browne, Chris Bruner, Jeff Fike, Travis Fisher, Mike Hansen, Victor Brunini, Mark Hoemmen (now at Steller Software), Brad Maeng, James Overfelt, Julia Plews, Cory Stack, Ross Wagnild, Jerry Watkins
SPARC V&V Team:
Sarah Kieweg, Brian Carnes, Eric Mussoni, Brian Freno, Greg Weirs, Derek Dinzl, Jaideep Ray, Ryan Keedy, Micah Howard, Ross Wagnild, Justin Winokur

Components Project Managers:
Ron Brightwell, Robert Clay, Rob Hoekstra, Ron Oldfield, Andrew Salinger, Dan Turner, Michael Wolf,
Components Principal Investigators:
Ross Bartlett, Si Hammond, Byron Hanks, Jonathan Hu, Paul Lin (now at LBNL), Ken Moreland, Stephen Olivier, Roger Pawlowski, Eric Phipps, Siva Rajamanickam, Greg Sjaardema, Christian Trott
Components Team:
Seher Acer, Andrew Bradley, Luc Berger-Vergiat, Lewis Cannada, Brian Carnes, Jan Ciesko, Sidafa Conde, Kevin Copps, Vinh Dang, Mohamed Ebeida, H. Carter Edwards (now at Nvidia), Nathan Ellingwood, James Elliott, Noah Evans, Kurt Ferreira, J. Forester, Joe Frye, Chris Glusa, Ryan Grant, G. Hansen, Evan Harvey, Daisy Hollman, Dan Ibanez, Kyungjoo Kim, Kathryn Maupin, Scott Levy, Brian Kelley, Jeff Mauldin, D. Mavriplis (U. Wyoming), Duncan McGregor, Jeff Miles, Sean Miller, Shymali Mukherjee, Curt Ober, Tom Otahal, Steve Owen, Kevin Pedtretti, Mauro Perego, Edward Phillips, David Poliakoff, Nathan Roberts, Roshan Quadros, W. Scott, Chris Siefert, Bill Spotz, Cory Stack, Paul Stallings, Dan Sunderland, Gary Templet, Irina Tezaur, A. Toth, Ray Tuminaro, Craig Ulmer, Lee Ward, Jerry Watkins, Jeremy Wilke, Ichi Yamazaki, Andrew Younge

The Sandia ATDM Strategy was defined at the beginning of 2015, we continue with it today

Vision: In partnership with Sandia's weapon engineers, the ATDM project will help usher in a new era of computational analysis for *weapon life-cycle engineering* by demonstrating *embedded analysis* and *exceptional application performance* on next-generation and exascale high-performance computing systems.

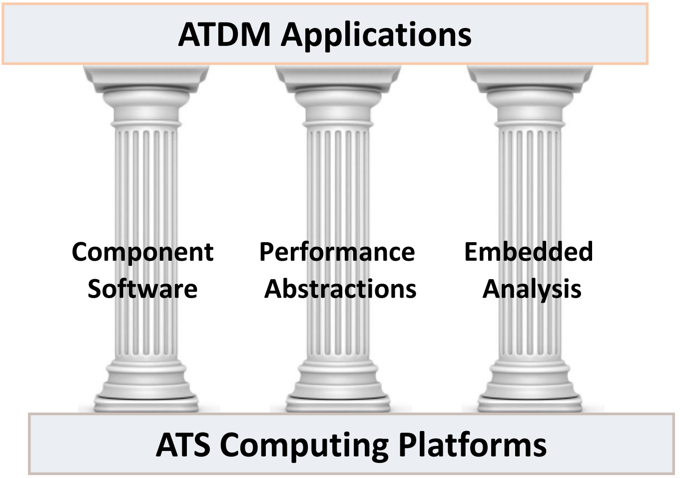
Sandia ATDM Major Technical Themes

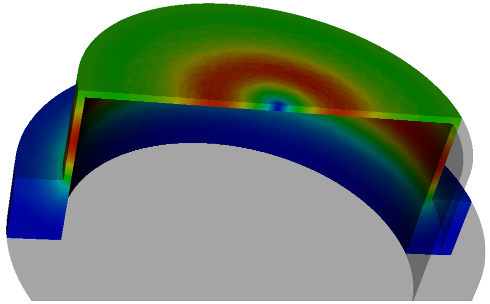
ATDM will provide capabilities utilizing next-generation computing to assess and mitigate
these scenarios:

- Hostile survivability
- Reentry environments

ATDM priorities:

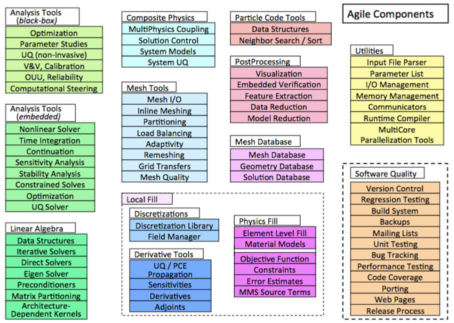
- NW applicability
- Exascale performance
- Flexible software components & abstractions





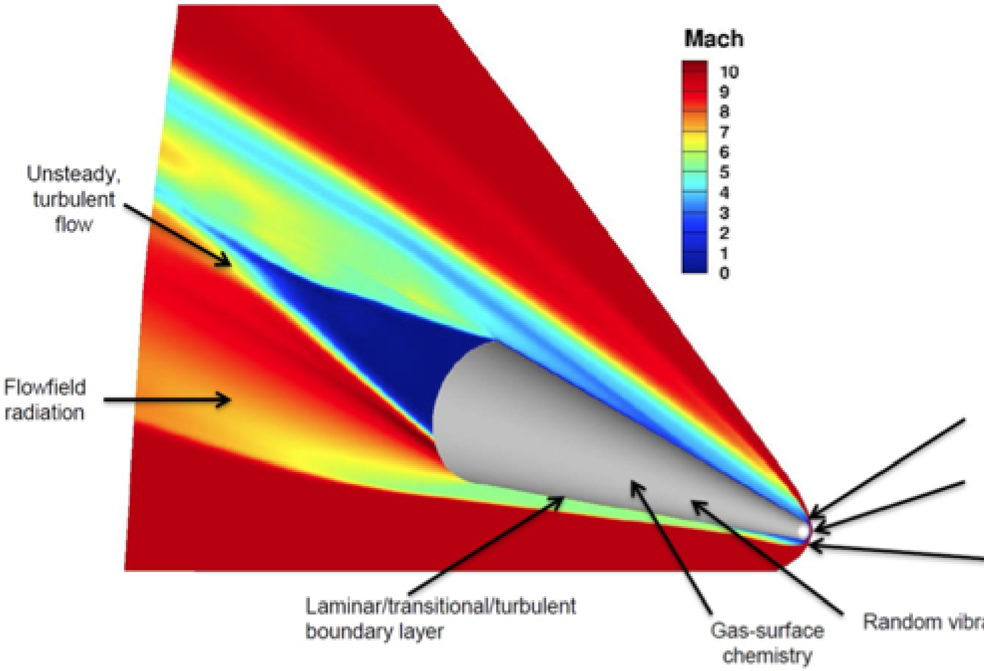
Sierra

Agile Components



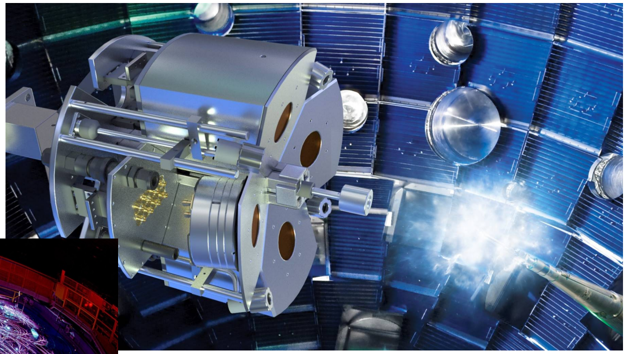
Overarching Goal Credible and Accurate Predictions and Assessments

Re-entry Environments
SPARC

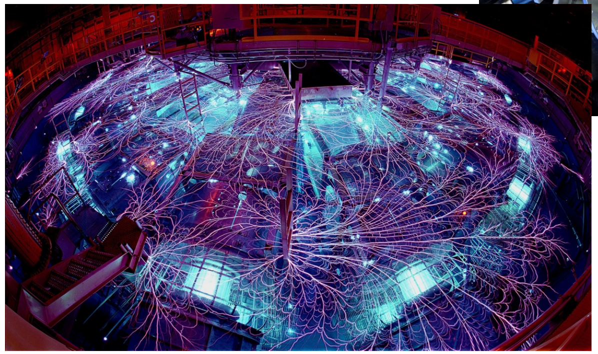


RAMSES
ITS SCEPTRE NuGET EIGER EMPHASIS Xyce Charon
CHEETAH Gemma EMPIRE

Plasma Simulation
EMPIRE



EMPIRE



Gemma
Electromagnetic Radiation (EMR)

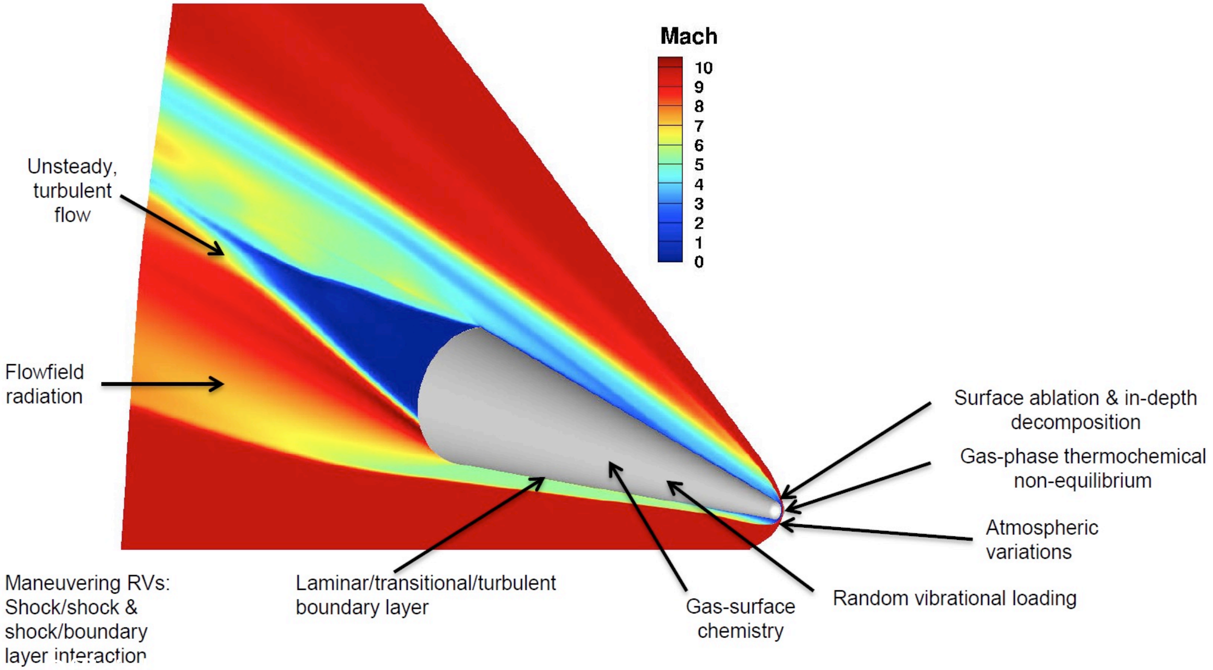
CHEETAH
Radiation Effects

- Productionize ATDM codes (transition to Integrated Codes)
- Develop key combined environments simulation capabilities (new physics + coupling)
- Follow-through on production code preparation for Next-Generation Platforms
- Leverage Strategic Partnership Programs (SPP) where possible

SPARC Basics

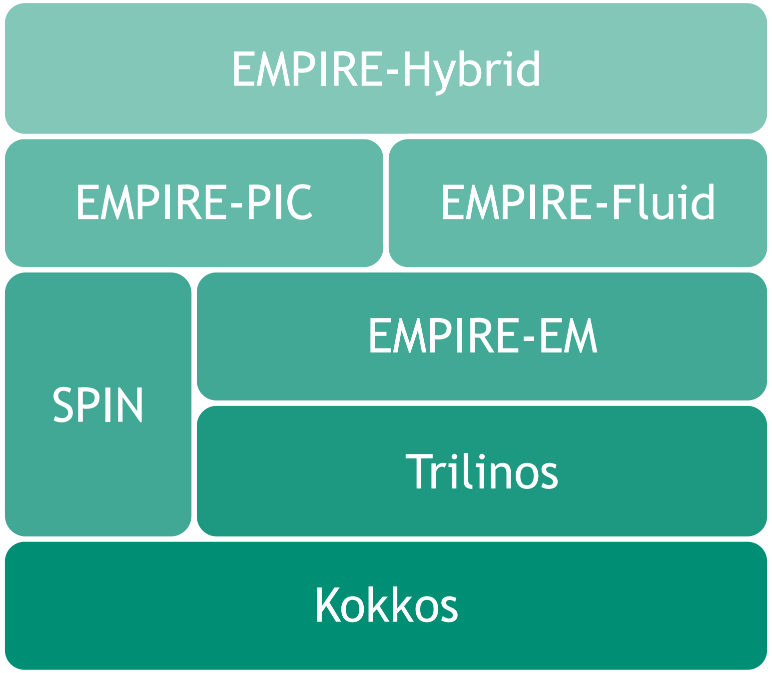
- State-of-the-art reentry simulation on next-gen platforms
  - Continuum compressible CFD (Navier-Stokes), hypersonic gas dynamics
  - Hybrid structured-unstructured finite volume methods. R&D: high order unstructured disc. collocation element methods
  - Perfect and thermo-chemical non-equilibrium gas models
  - RANS and hybrid RANS-LES turbulence models; R&D: Direct Numerical Simulation
- Enabling technologies/components
  - Performance portability through Kokkos
  - Scalable solvers
  - Embedded geometry & meshing
  - Embedded UQ and model calibration

- Credibility
  - Validation against wind tunnel and flight test data
  - Visibility and peer review by external hypersonics community
- Software quality
  - Rigorous regression
  - V&V
  - Performance testing



Advancing Plasma Physics Modeling

- EMPIRE leverages the opportunity from ATDM to advance plasma simulation capability on two fronts:
  - Component-based software design for portability across next-generation hardware architectures
  - New fluid and hybrid kinetic/fluid algorithms for validity and performance across a wider range of plasma density regimes
- EMPIRE is built upon Trilinos components:
  - Panzer: FEM discretization infrastructure
  - Tempus: General time integration package
  - Uses the modern Tpetra-based linear solver stack
  - Kokkos: Portable threading library
- EMPIRE will enable:
  - Higher fidelity modeling of critical plasma applications
  - Towards exascale simulation



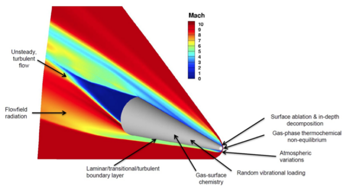
Overview of Productivity, Portability, and Performance

- Developer Productivity
  - … Relative to large, complex scientific software
  - … Relative to advanced parallel programming models for new computer architectures

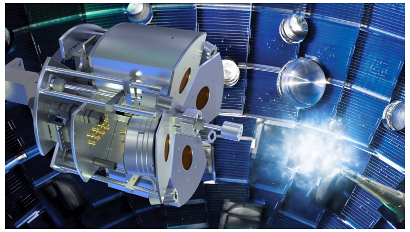
- Portability and Performance
  - … Relative to various computer architectures (CTS-1, Trinity/KNL, and Sierra)

Agile Component Strategy

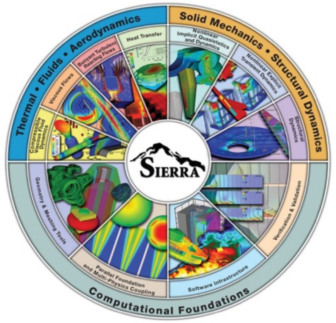
- Sandia has decades of software development experience using component strategy
- Start from current Agile Components (Trilinos)
- Design new components/APIs based on ATDM requirements
- Explore new technologies
- Deep integration of ATDM technologies
- Deep integration of ATDM application and component teams



SPARC



EMPIRE



Engineering Mechanics Code Suite

RAMSES

ITS SCEPTRE NuGET EIGER EMPHASIS Xyce Charon CHEETAH Gemma EMPIRE

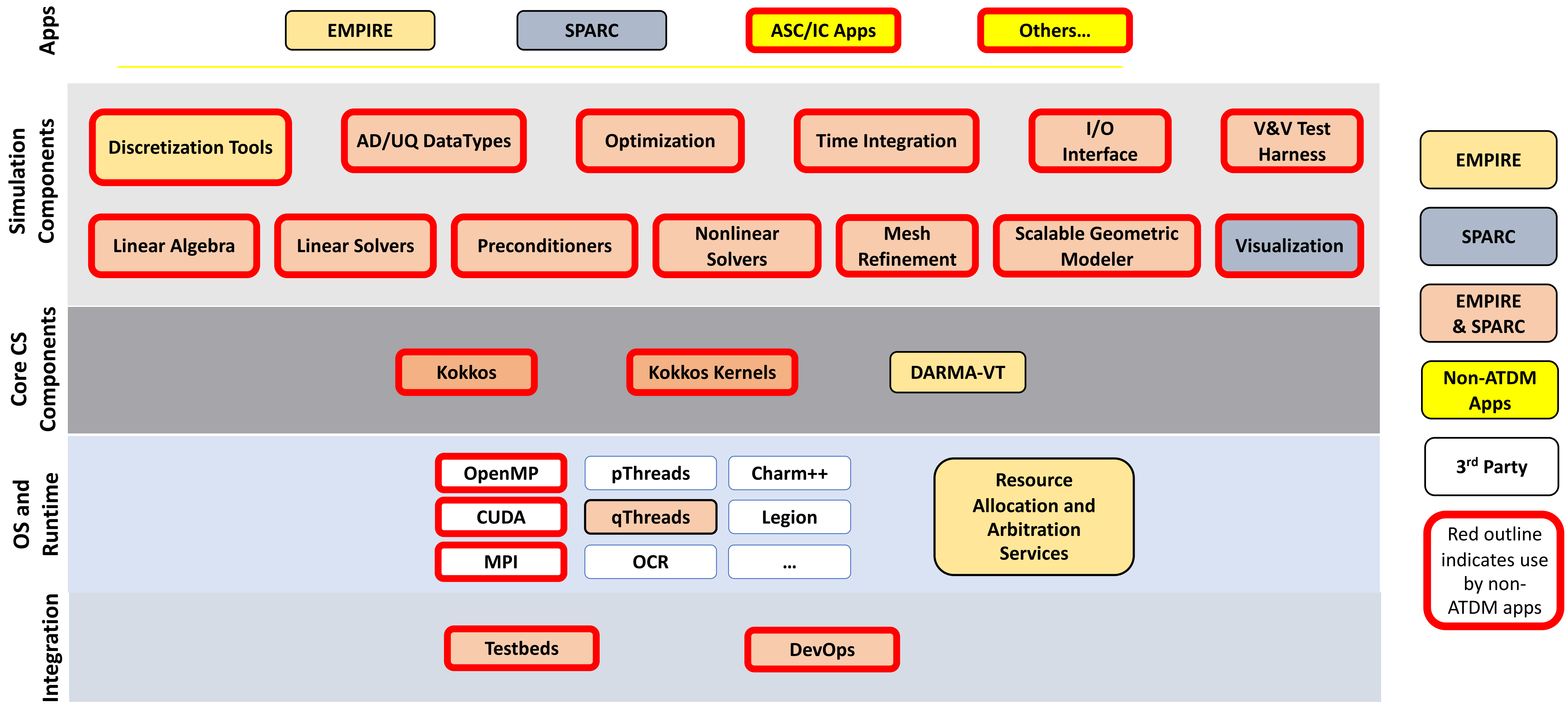
Electromagnetics, Radiation, and Electrical Code Suite

TRILINOS

Agile Components

ATDM applications benefit as well as setting a broad foundation for other applications - *"Write once, use many"*

ATDM Components



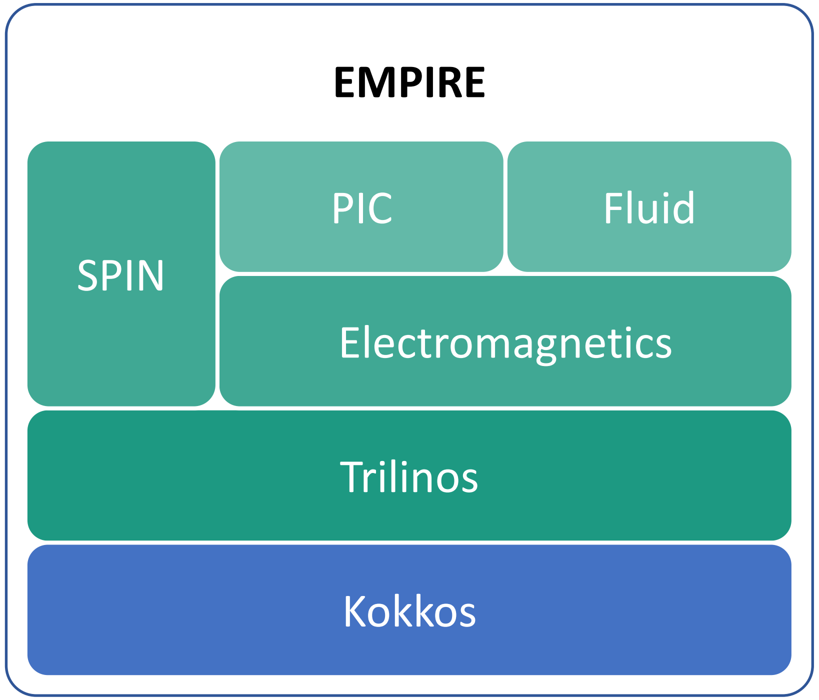
Overview for EMPIRE: Context for the developer challenge

Clean slate approach with designed-in flexibility to meet the ATDM challenge

Mission Impact
Enable new Hybrid plasma simulation capability

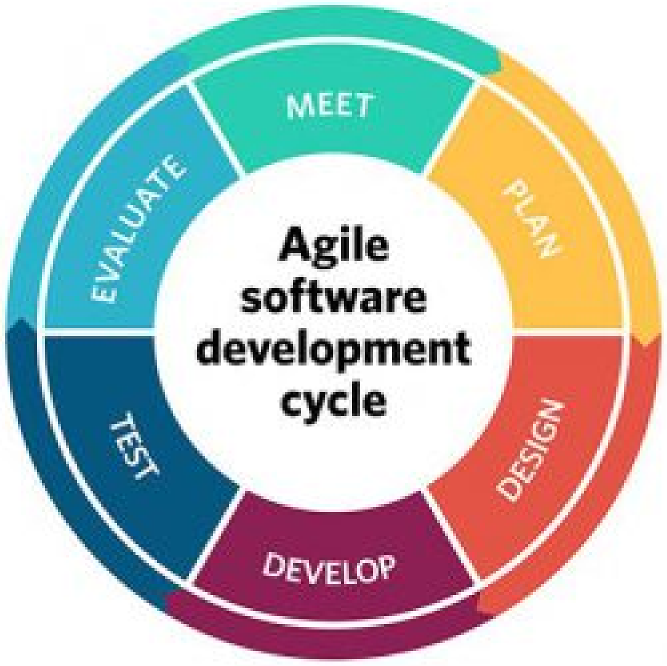
Portability
Enable built-in use of Kokkos for portability

Performance
Enable new algorithms and implementations



Agile Software Development

- Iterative software development
- Engagement with stakeholders
- Regular team meetings
- Team retrospectives
- Team practices & knowledge
- Readable and maintainable code
- Address technical debt



Agile software development and strong team practices support developer productivity.

Performance Portability has an Impact on Developer Productivity

Development Environment

- Long Compile Time
- Poor debugging and profiling tools
- More testing per feature
- Less developer experience

Performance Portability

- Undetermined Loop Order
- Abstract Data Layout
- Language Features Disallowed/Slow
- Poor TPL support

Can feel like someone threw a wrench into the gears!

https://disciplemakingstages.files.wordpress.com/2018/03/shutterstock_96357110.jpg

- Some of these are mitigated by components and the ecosystem
- Some of these are mitigated at the team level.

There is a cost to performance portability but we have strived to minimize it.

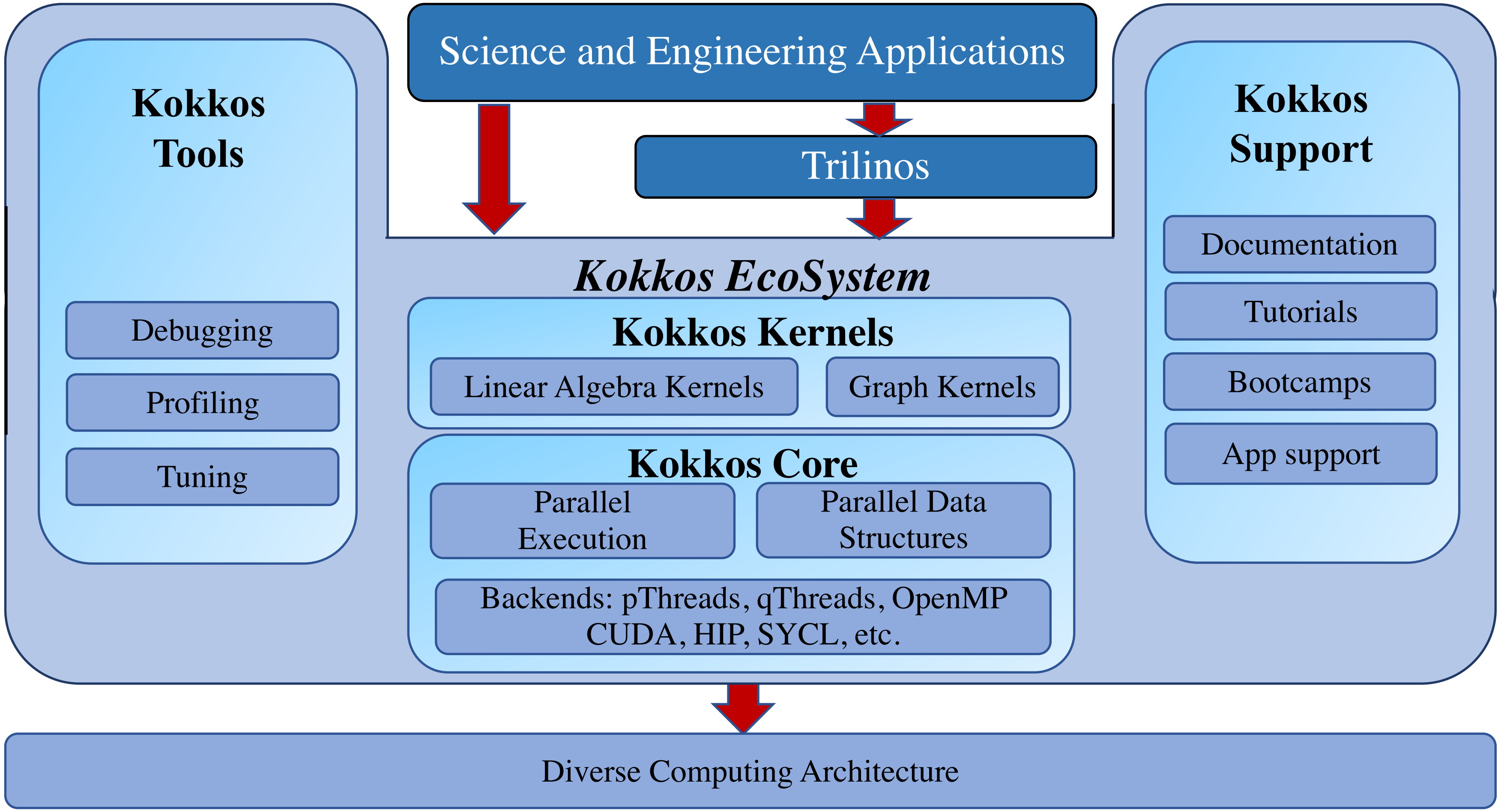
SNL Strategy for New Computer Architectures

- New computer architectures increase the cognitive load on developers
  - Significant increase concurrency
  - New programming models
  - Different memory spaces
  - Hardware differences
- We seek to reduce this load on developers with an abstraction layer – Kokkos
  - Componentized software libraries
    - Deep subject matter expertise consolidated in these components instead of being required on all teams
  - Abstraction layer provides a common programming API – full spectrum of vendor abstraction
    - Hides hardware differences and performance approaches
    - CoDesign interactions
    - Software stack improvements
  - Provides memory management
  - Push programming concepts into the C++ language standard
- Adopted by all ASC codes targeting new HPC architectures, not just ATDM components and applications

Use of the Kokkos abstraction layer significantly reduces the cognitive load on developers for supporting new computer architectures.

The Kokkos EcoSystem

kokkos



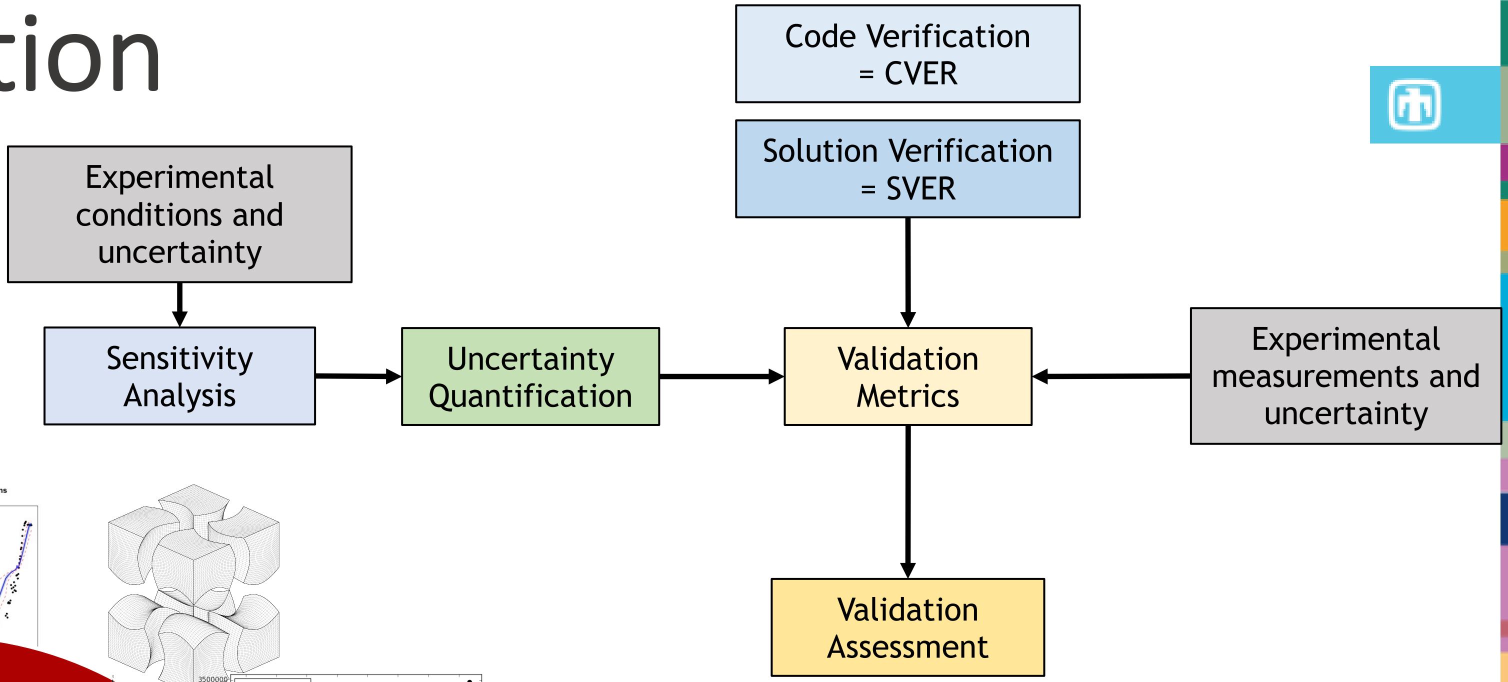
Kokkos Core: parallel patterns and data structures; supports several execution and memory spaces

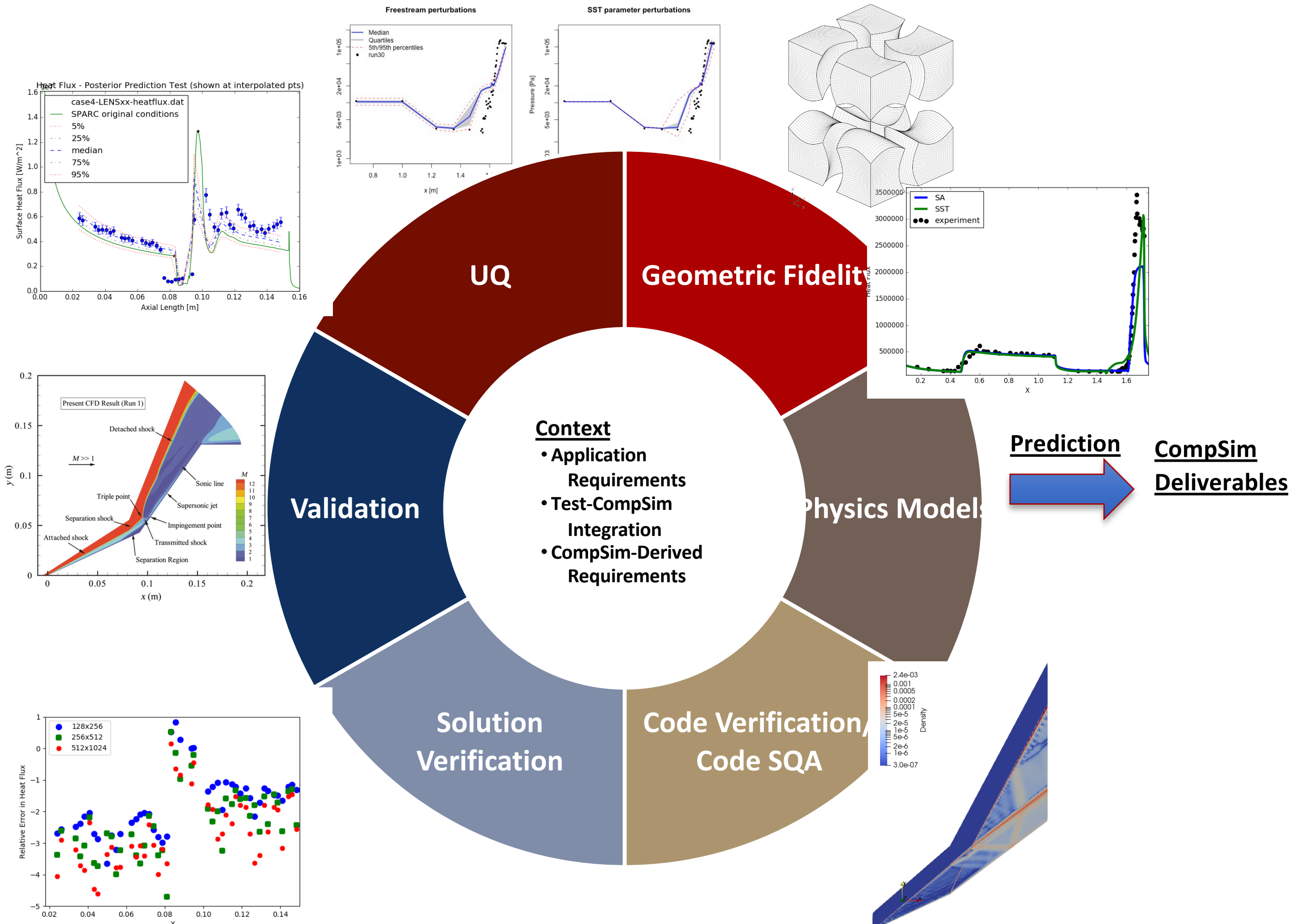
Kokkos Kernels: performance portable BLAS; sparse, dense and graph algorithms

Kokkos Tools: debugging and profiling support

Kokkos enables performance portability and the complexity of supporting numerous architectures that are central to DOE HPC enterprise

Approach: Verification & Validation / UQ Workflow





A Word on Performance and Portability

Relative node performance (**measured against CTS systems**) Courtesy of Si Hammond, SNL

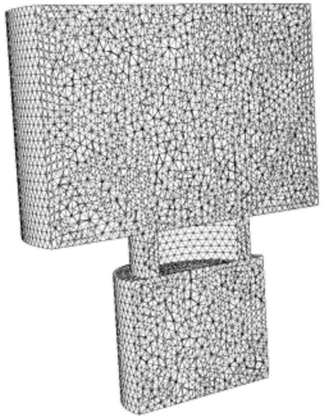
		CTS1	Trinity		Sierra		Astra
		Broadwell	Haswell	KNL	POWER9	V100 GPU	ThunderX2
LINPACK FLOP Rates (per Node)	Perf	1.09 TF/s	~0.86 TF/s	~2.06 TF/s	~1 TF/s	~21.91 TF/s	~0.71 TF/s
	Rel	1.00X	0.79X	1.89X	0.91X	20.01X	0.65X
Memory Bandwidth (STREAM) (per Node)	Perf	~136 GB/s	~120 GB/s	~90 GB/s / ~350 GB/s	~270GB/s	~850 GB/s x 4 = ~3.4 TB/s	~250 GB/s
	Rel	1.00X	0.88X	0.66X	1.99X	25.00X	1.84X
Power (TDP, per Node)	Perf	120W x 2 = 240W	135W x 2 = 270W	~250W	190W x 2 = 380W	~300W x 4 = 1.2kW	~180W x 2 = 360W
	Rel	1.00X	1.13X	1.04X	1.58X	5.00X	1.50X

Defining the performance measurement cases

Defined surrogate problems to measure and track performance

Simple Cavity

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



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

For historical comparison and tracking

Complex Cavity

- Complex geometry.
- Preloaded particles for scaling studies, representative of physical emission drive.
- Run for nominal 100 time-steps for scaling studies.

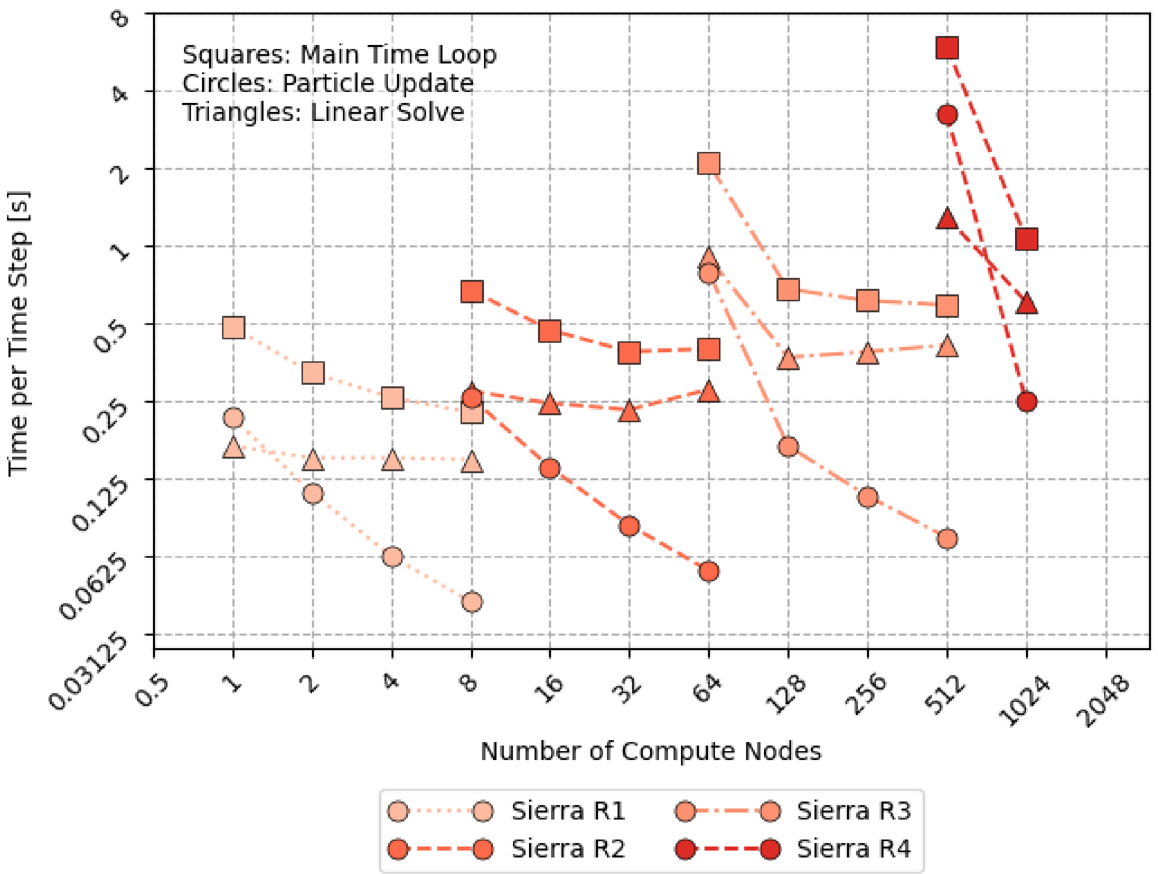
Mesh	Elements	Nodes	Edges	Particles*
R0	3.7M	660k	4.4M	360M
R1	25M	4.4M	30M	2.4B
R2	200M	32M	240M	19B
R3	1.6B	270M	1.9B	160B

*Scaling runs

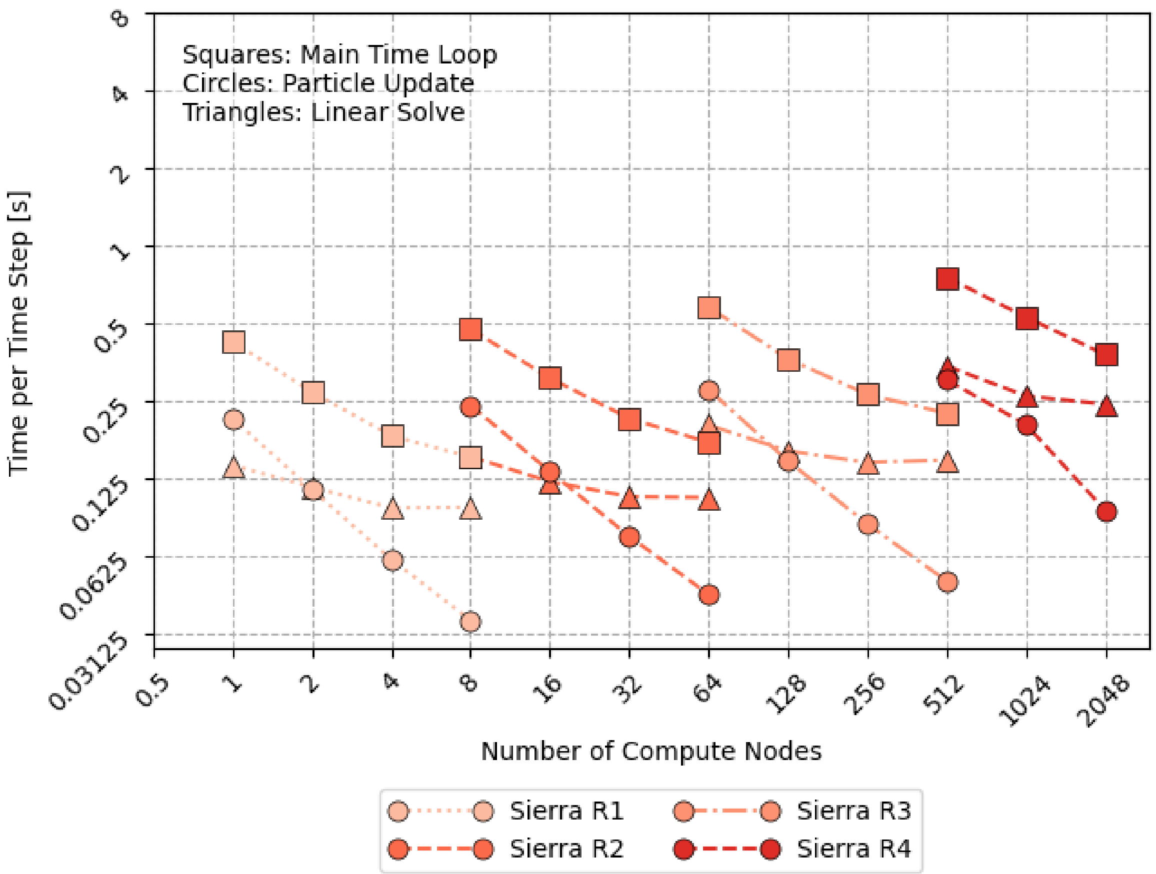
To meet the criteria with a relevant problem

ATS-2 performance improvements for EMPIRE

December 2019 results



August 2020 results

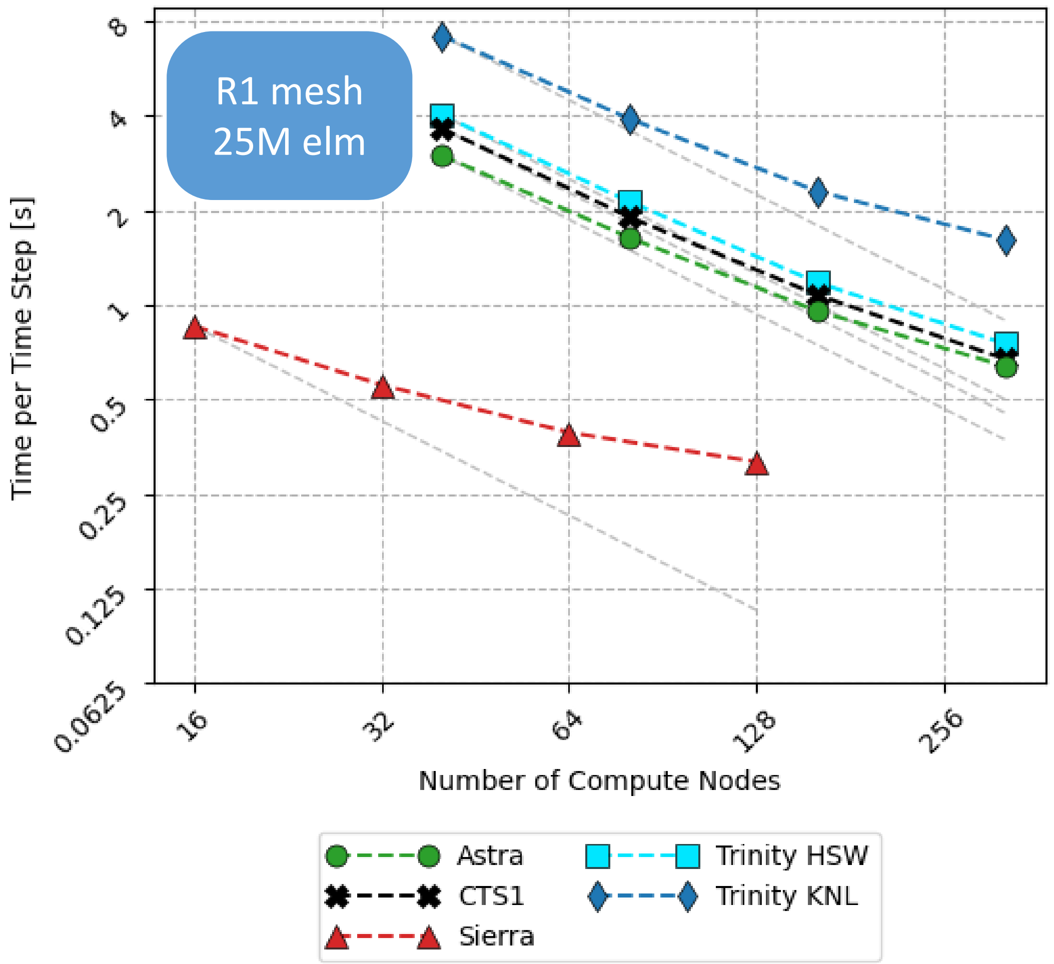


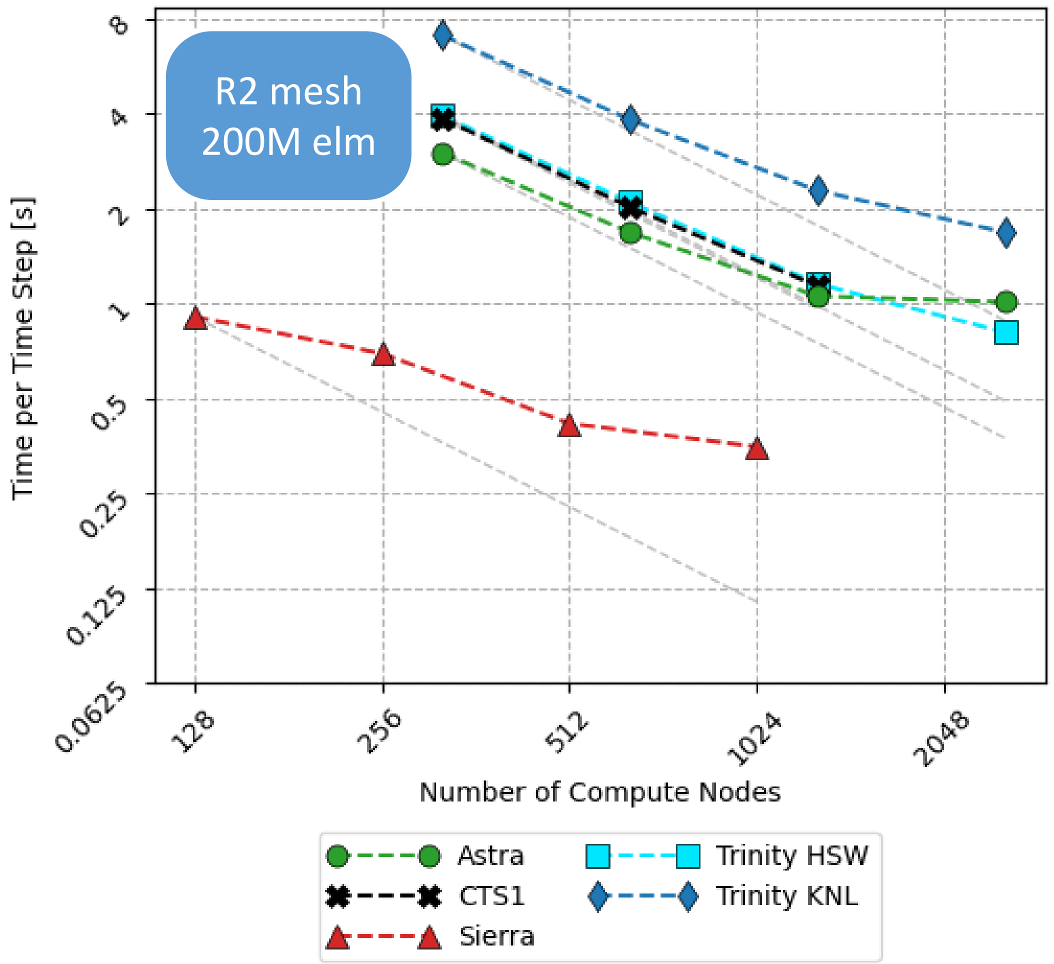
- Linear solver did not weak scale
- Particle update showed strong scaling issues

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

Performance results for the Complex Cavity

Cross-platform strong scaling results

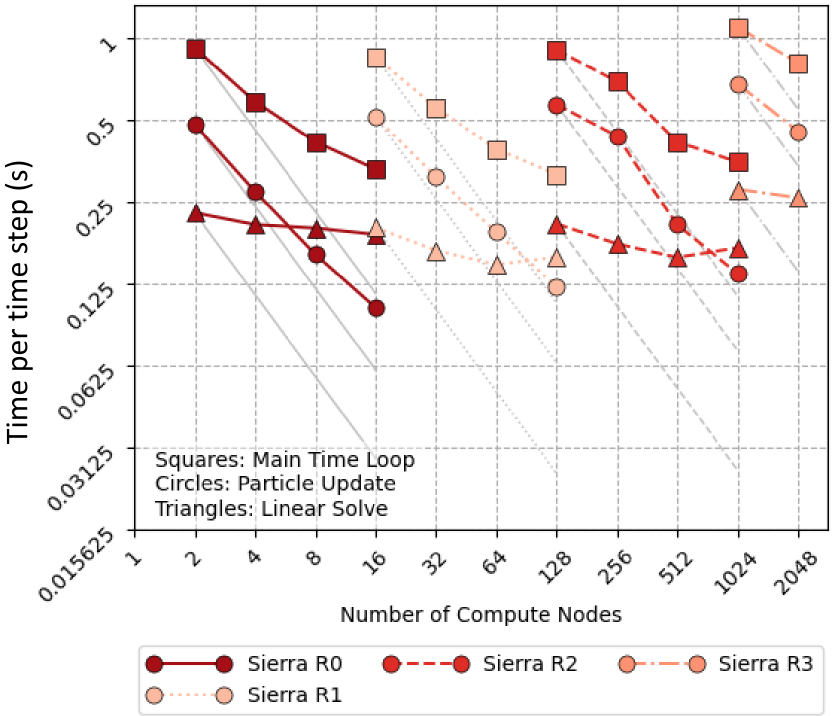




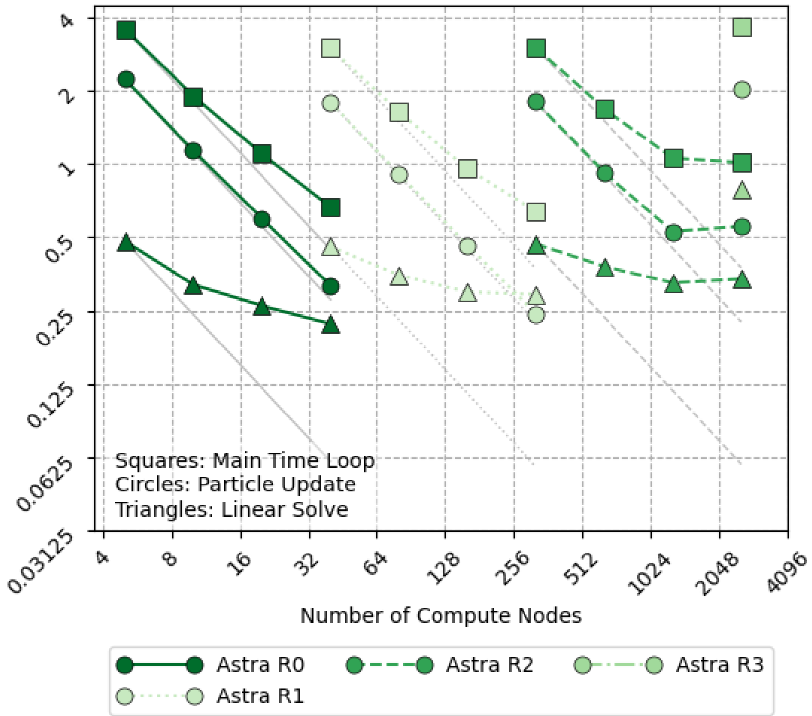
Performance results for the Complex Cavity

Single platform scaling results

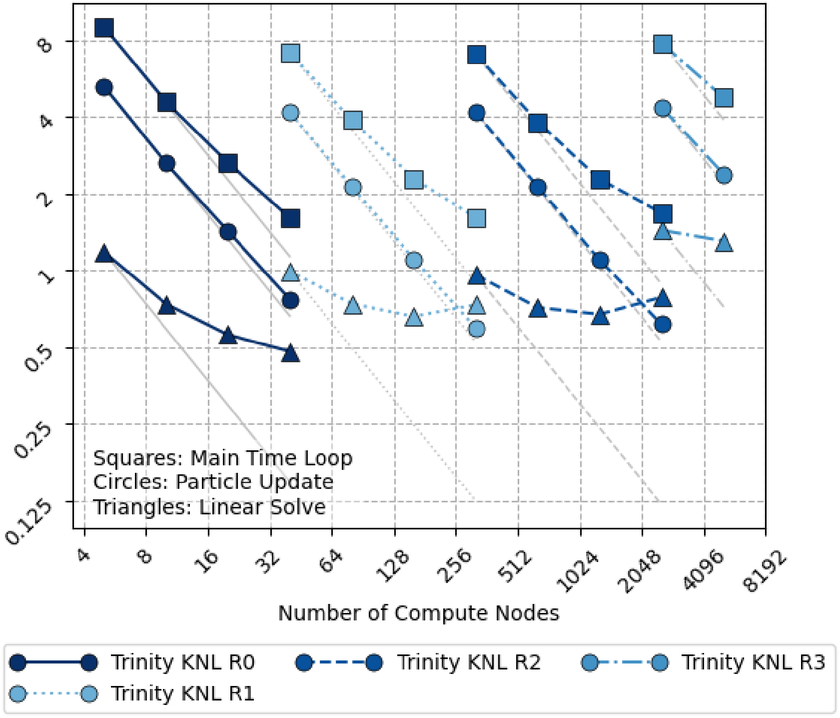
Sierra
4 GPU/socket; 1 MPI/GPU
Up to 2048 nodes (47% of total)



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)

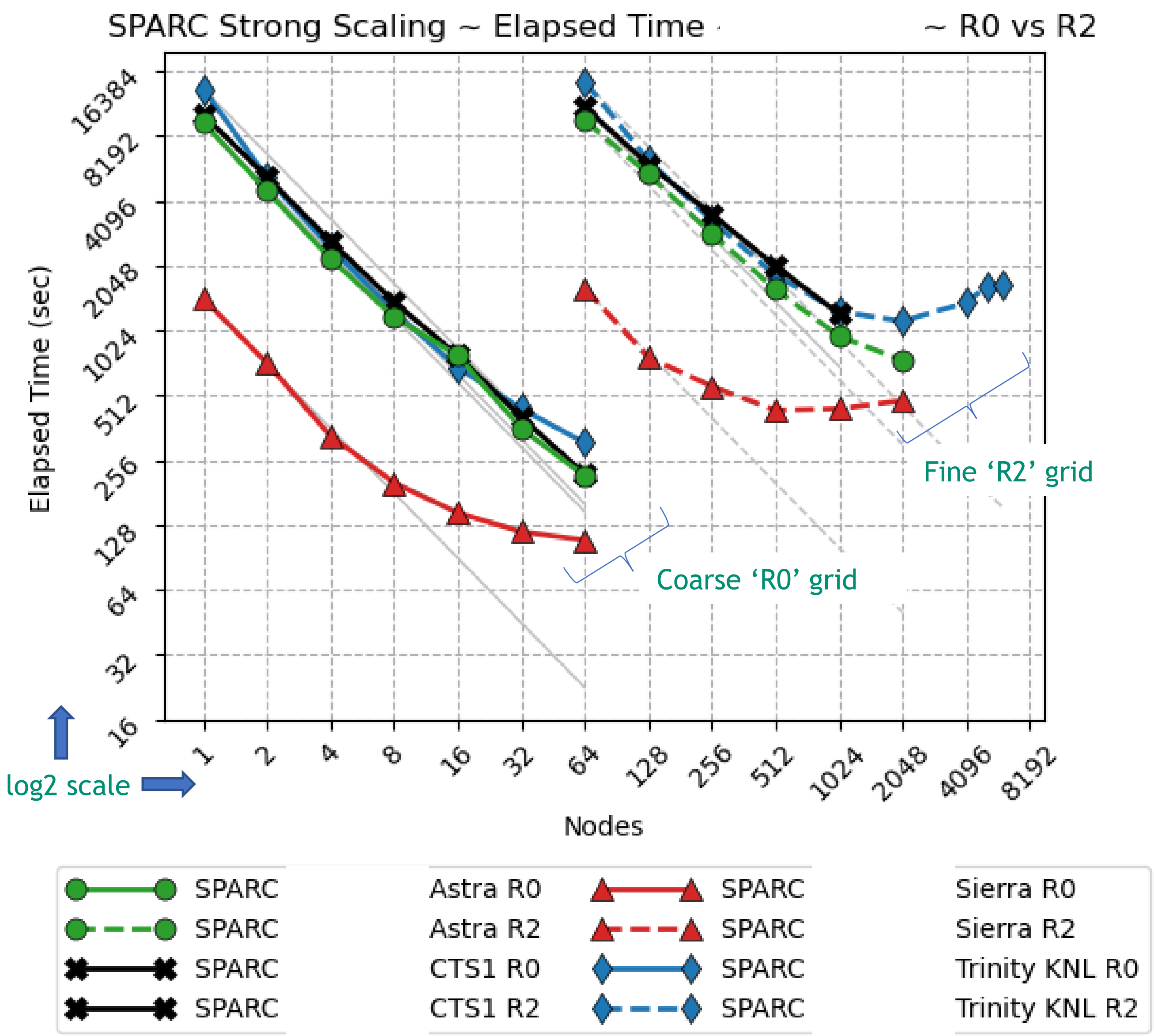


Addresses questions: Performance issues on Trinity/KNL and Kokkos

SPARC Performance Portability Strategy

- Leverage Trilinos for performance-portable linear solvers and I/O facilities
- Implement domain-specific data structures & mesh iteration abstractions on top of Kokkos
  - Can tune implementation for different platforms to maximize performance
    - Atomics vs graph coloring
    - Memory layouts & iteration patterns
    - SIMD types
- All physics code remains platform agnostic
- 96.1% of SPARC code base is platform agnostic

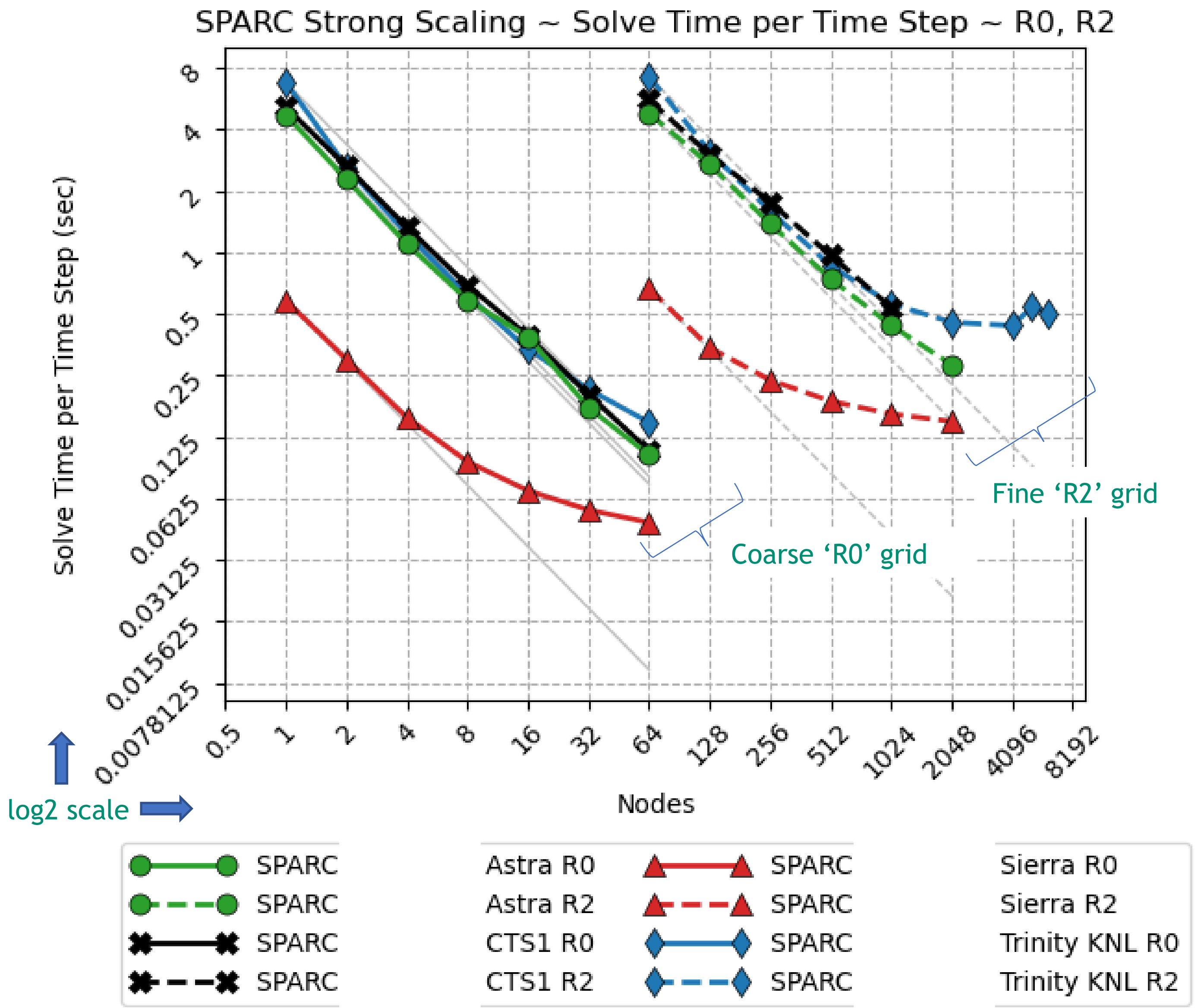
Hypersonics Performance – Strong Scaling, Overall Runtime



CTS1 as baseline:
Trinity: 0.8-1.2x
Astra: 1.2-1.4x
Sierra: 2-8x

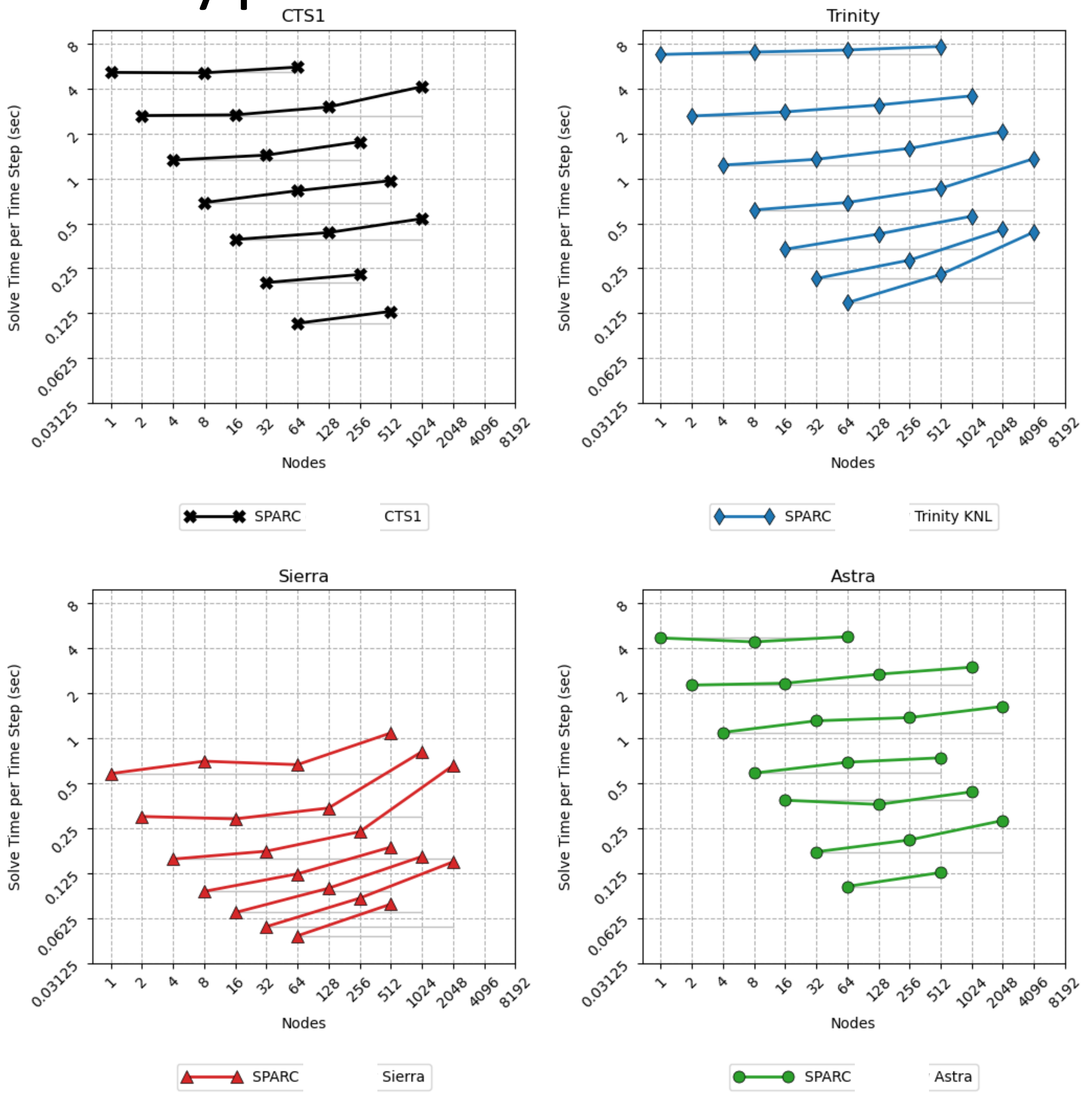
Some increase in elapsed time at largest scales for Trinity and Sierra.

Hypersonics Performance – Strong Scaling, Time per Time Step



Time per time step does not show same increase at large scale on Trinity and Sierra.

Hypersonic Performance – Weak Scaling



Trinity & Astra:

Good weak scaling with sufficient work per node

Sierra:

More work per node required for good weak scaling

Understanding Sierra Strong Scaling

SPARC's block tridiagonal linear solver is the primary limiter of strong scaling.

1. Relative cost of compute kernels is much lower than on other platforms
2. Kernel launch latency sets a high floor on kernel runtime
3. Relatively high cost of MPI getting data to/from GPU
4. Exposing sufficient parallelism to occupy GPU

SPARC Performance Summary

1. Successfully demonstrated SPARC scaling performance at scale on
   - 6144 nodes on Trinity
   - 2048 nodes on Astra
   - 2048 nodes on Sierra
2. SPARC achieves **excellent performance portability** and **speed-ups of up to**
   when compared node-to-node with CTS-1
   - 1.2x on Trinity
   - 1.4x on Astra
   - 8x on Sierra
3. SPARC achieves this with a **code base that is >95% platform agnostic**