Overview of Numerical Algorithms and Software for Extreme-Scale Science

Presented to
ATPESC 2018 Participants

Lori Diachin
Deputy Director, Exascale Computing Project

Q Center, St. Charles, IL (USA)
Date 08/06/2018

U.S. DEPARTMENT OF ENERGY | Office of Science

ATPESC Numerical Software Track

ECP EXASCALE COMPUTING PROJECT

Argonne NATIONAL LABORATORY · BERKELEY LAB · Sandia National Laboratories · Rensselaer · SMU

Track 4: Numerical Algorithms and Software for Extreme-Scale Science

MONDAY, August 6, 2018

Time	Title of presentation	Lecturer
9:30 am	**Overview of Numerical Algorithms & Software for Extreme-Scale Science**	Lori Diachin, LLNL
… **with hands-on lessons** throughout the day for various topics (lead: Mark C. Miller, LLNL)		
10:30 am	Break	
11:00 am	**Unstructured Meshing and Discretization**	Tzanio Kolev, LLNL and Mark Shephard, RPI
12:30 pm	Lunch	
1:30 pm	**Time Integration and Nonlinear Solvers**	Barry Smith, ANL
2:15 pm	**Krylov Solvers and Multigrid**	Ulrike Meier Yang, LLNL
3:00 pm	Break	
3:30 pm	**Sparse Direct Solvers**	Sherry Li, LBNL
4:15 pm	**Numerical Optimization**	Todd Munson and Hong Zhang, ANL
5:00 pm	**Putting It All Together: One Perspective**	Cameron Smith and Mark Shephard, RPI
5:30 pm	Dinner + **Panel: Extreme-Scale Algorithms & Software**	Track 4 Team
6:30 pm	**Hands-on Deep Dives ... 1-on-1 Discussions … Prizes!**	Track 4 Team

+ Hands-on lessons

Additional contributors to lectures and hands-on lessons:

Satish Balay (ANL), Alp Denner (ANL), Aaron Fisher (LLNL), Lois Curfman McInnes (ANL)

Additional contributors to gallery of highlights:

Karen Devine (SNL), Mike Heroux (SNL), Piotr Luszczek (U Tennessee), Dan Martin (LBNL), Julianne Müller (LBNL)

See also Track 6:

Software Productivity (Aug 8)

ECP EXASCALE COMPUTING PROJECT

Track 4: Numerical Algorithms and Software: Tutorial Goals

1. Provide a basic understanding of a variety of applied mathematics algorithms for scalable linear, nonlinear, and ODE solvers as well as discretization technologies (e.g., adaptive mesh refinement for structured and unstructured grids)

2. Provide an overview of software tools available to perform these tasks on HPC architectures … including where to go for more info

3. Practice using one or more of these software tools on basic demonstration problems

This presentation gives a high-level introduction to HPC numerical software

- How HPC numerical software addresses challenges in computational science and engineering (CSE)
- Toward extreme-scale scientific software ecosystems
- Using and contributing: Where to go for more info

Why is this important for you?

- Libraries enable users to focus on their primary interests
  - Reuse algorithms and data structures developed by experts
  - Customize and extend to exploit application-specific knowledge
  - Cope with complexity and changes over time
- More efficient, robust, reliable, scalable, sustainable scientific software
- Better science, broader impact of your work

This work is founded on decades of experience and concerted team efforts to improve numerical software

- **FASTMath SciDAC Institute**

fastmath-scidac.llnl.gov

- **IDEAS Scientific Software Productivity**

ideas-productivity.org

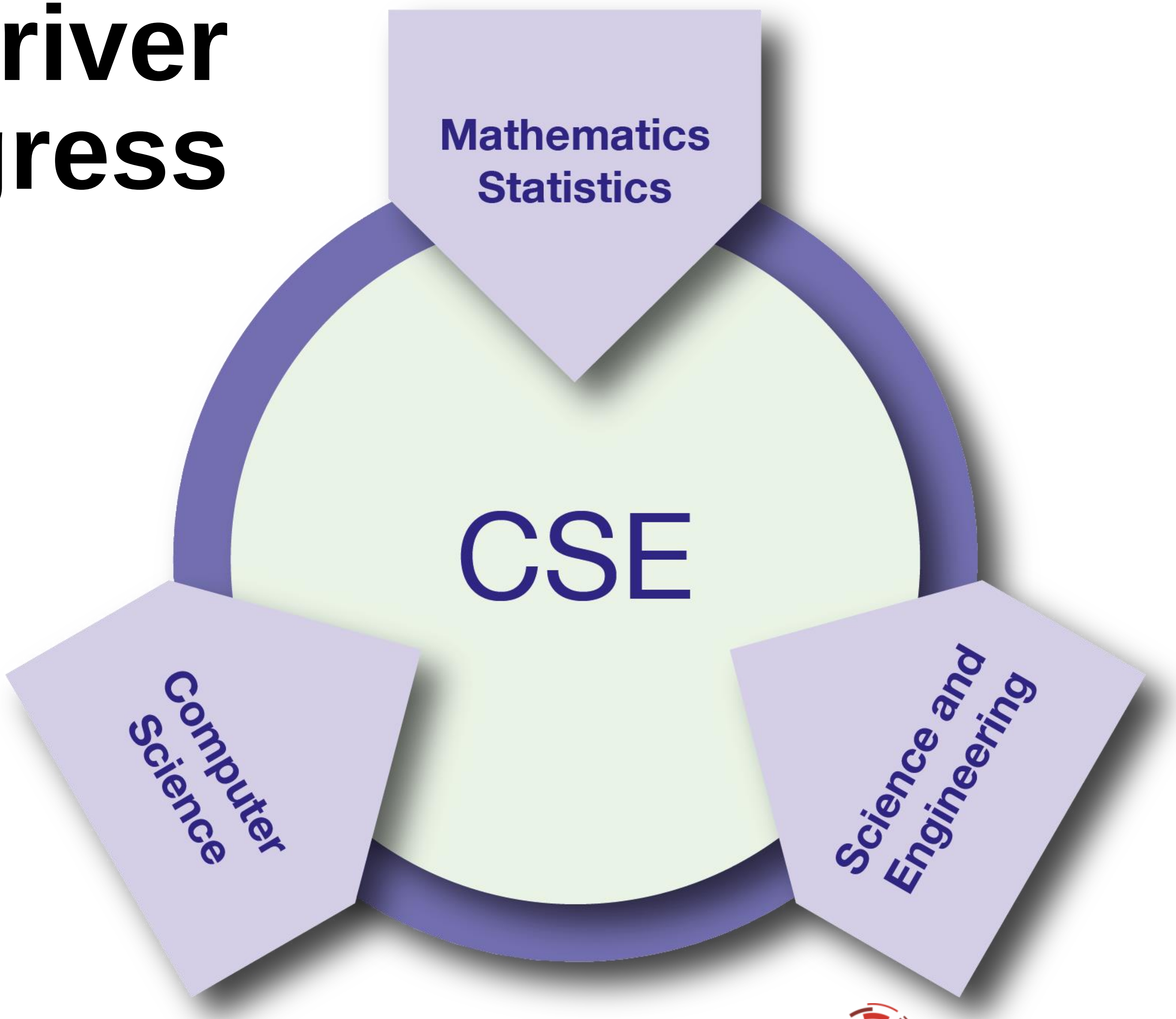
- **Exascale Computing Project**

CSE: Essential driver of scientific progress

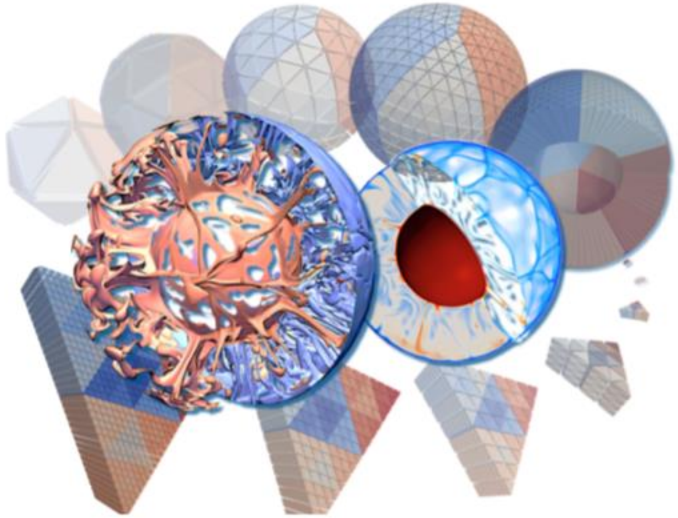
CSE = Computational Science & Engineering

Development and use of computational methods for scientific discovery

- all branches of the sciences
- engineering and technology
- support of decision-making across a spectrum of societally important applications



Rapidly expanding role of CSE: New directions toward predictive science



- Mathematical methods and algorithms
- CSE and HPC: Ubiquitous parallelism
- CSE and the data revolution
- CSE software
- CSE education & workforce development

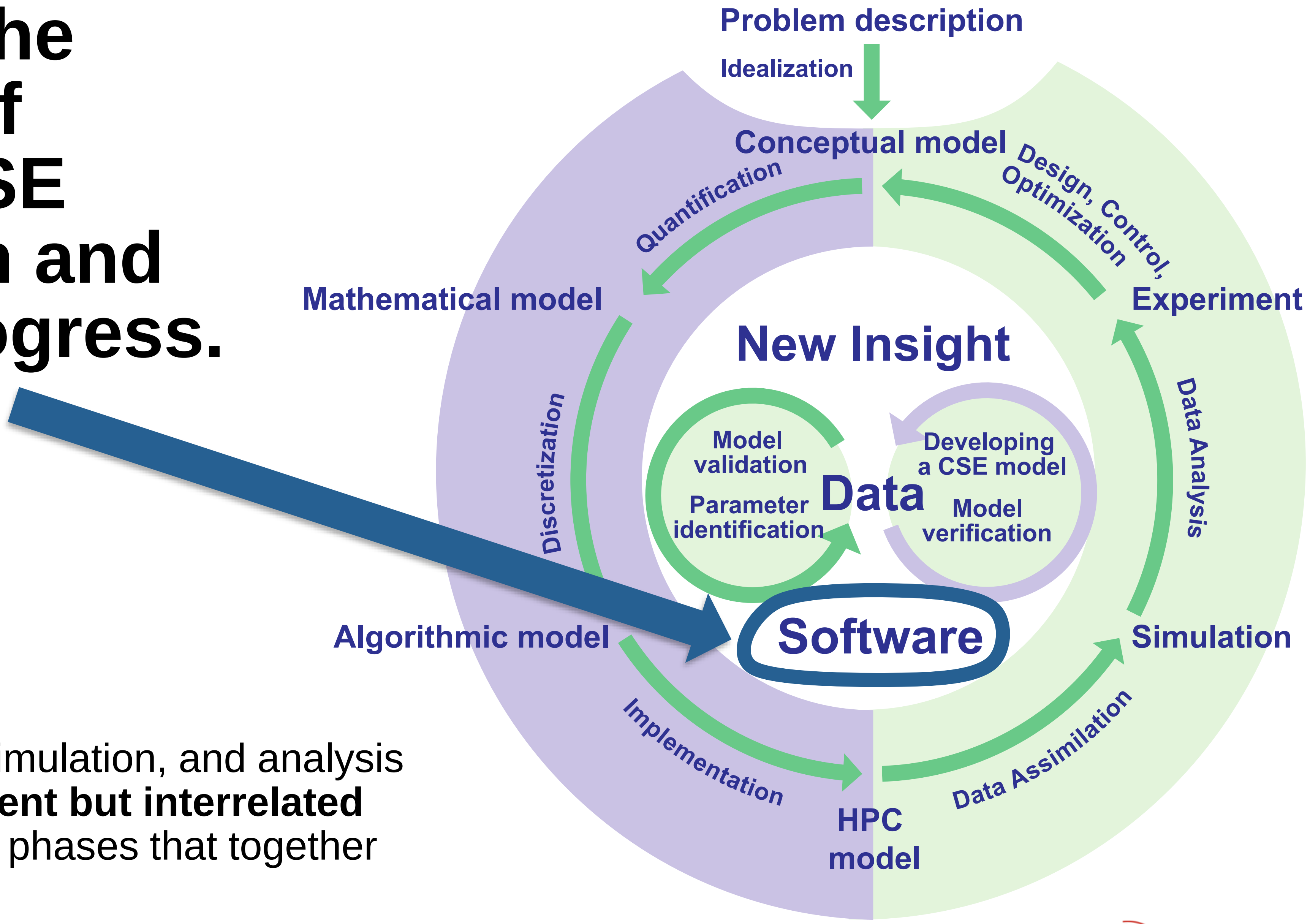
Research and Education in Computational Science & Engineering

U. Rüde, K. Willcox, L.C. McInnes, H. De Sterck, G. Biros, H. Bungartz, J. Corones, E. Cramer, J. Crowley, O. Ghattas, M. Gunzburger, M. Hanke, R. Harrison, M. Heroux, J. Hesthaven, P. Jimack, C. Johnson, K. Jordan, D. Keyes, R. Krause, V. Kumar, S. Mayer, J. Meza, K.M. Mørken, J.T. Oden, L. Petzold, P. Raghavan, S. Shontz, A. Trefethen, P. Turner, V. Voevodin, B. Wohlmuth, C.S. Woodward, to appear in ***SIAM Review****,* Aug 2018

earlier draft*:* Jan 2018 https://arxiv.org/abs/1610.02608

ECP EXASCALE COMPUTING PROJECT

Software is the foundation of sustained CSE collaboration and scientific progress.



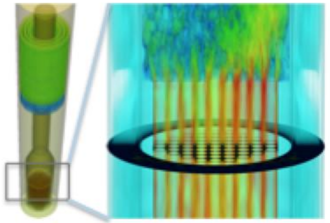
CSE cycle: Modeling, simulation, and analysis

- **Software: independent but interrelated elements** for various phases that together enable CSE

Multiphysics: A primary motivator for exascale

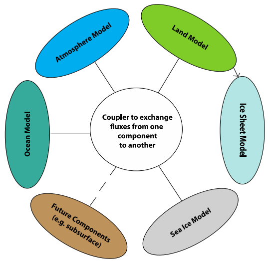
Multiphysics: greater than 1 component governed by its own principle(s) for evolution or equilibrium

– Also: broad class of coarsely partitioned problems possess similarities

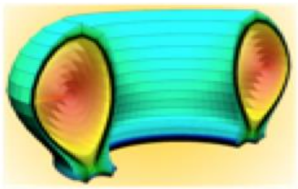


nuclear reactors
A. Siegel, ANL

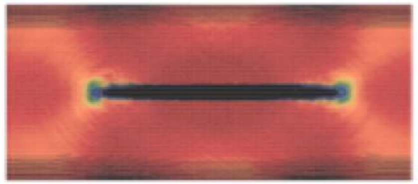
particle accelerators
K. Lee, SLAC



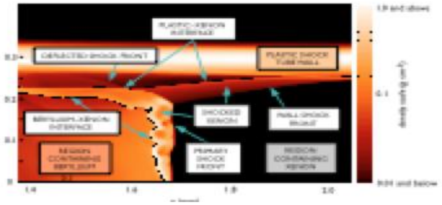
climate
K. Evans, ORNL



fusion
A. Hakim, PPPL



crack propagation
E. Kaxiras, Harvard



radiation hydrodynamics
E. Myra, Univ. of Michigan

IJHPCA, Feb 2013
Vol 27, Issue 1, pp. 4-83

THE INTERNATIONAL JOURNAL of HIGH PERFORMANCE COMPUTING APPLICATIONS

The International Journal of High Performance Computing Applications
27(1) 4–83
© The Author(s) 2012
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/1094342012468181
hpc.sagepub.com
SAGE

Multiphysics simulations: Challenges and opportunities

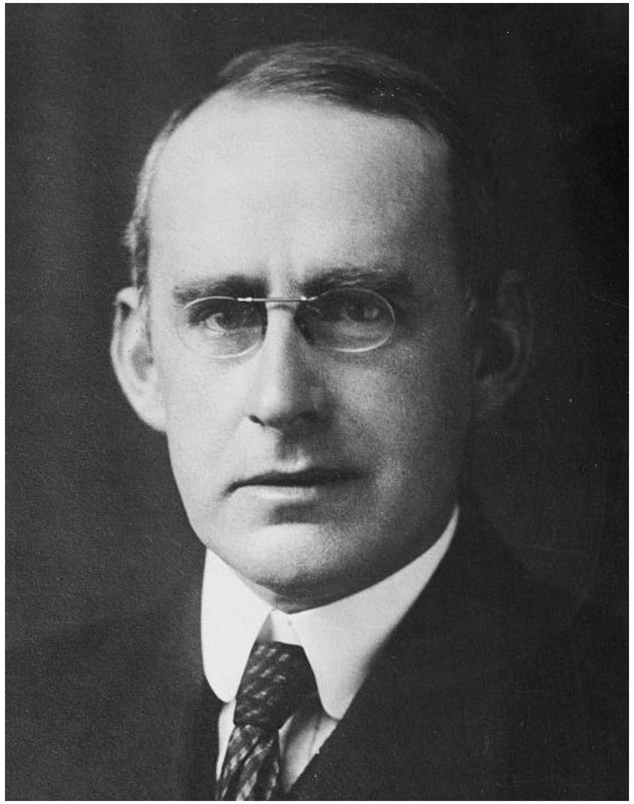
David E Keyes[1,2], Lois C McInnes[3], Carol Woodward[4], William Gropp[5], Eric Myra[6], Michael Pernice[7], John Bell[8], Jed Brown[3], Alain Clo[1], Jeffrey Connors[4], Emil Constantinescu[3], Don Estep[9], Kate Evans[10], Charbel Farhat[11], Ammar Hakim[12], Glenn Hammond[13], Glen Hansen[14], Judith Hill[10], Tobin Isaac[15], Xiangmin Jiao[16], Kirk Jordan[17], Dinesh Kaushik[3], Efthimios Kaxiras[18], Alice Koniges[8], Kihwan Lee[19], Aaron Lott[4], Qiming Lu[20], John Magerlein[17], Reed Maxwell[21], Michael McCourt[22], Miriam Mehl[23], Roger Pawlowski[14], Amanda P Randles[18], Daniel Reynolds[24], Beatrice Rivière[25], Ulrich Rüde[26], Tim Scheibe[13], John Shadid[14], Brendan Sheehan[9], Mark Shephard[27], Andrew Siegel[3], Barry Smith[3], Xianzhu Tang[28], Cian Wilson[2] and Barbara Wohlmuth[23]

doi:10.1177/1094342012468181

EXASCALE COMPUTING PROJECT

Multiphysics challenges … the study of ‘and’

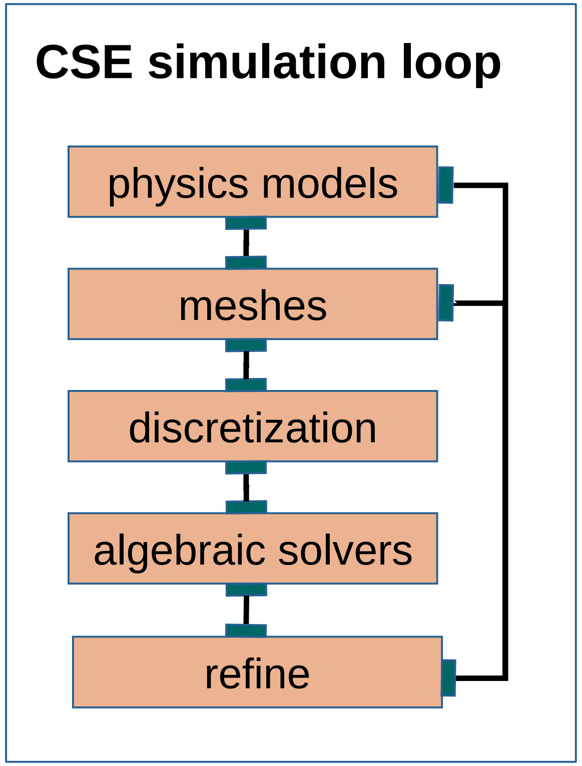
“We often think that when we have completed our study of one we know all about two, because ‘two’ is ‘one and one.’ We forget that we still have to make a study of ‘and.’ ”

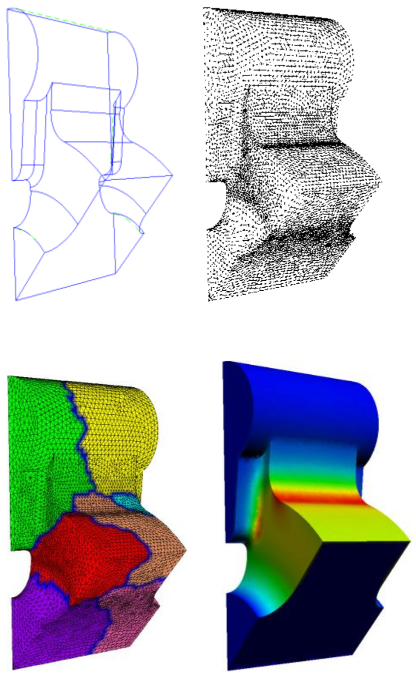


− Sir Arthur Stanley Eddington (1892−1944), British astrophysicist

CSE simulation starts with a forward simulation that capture the physical phenomenon of interest

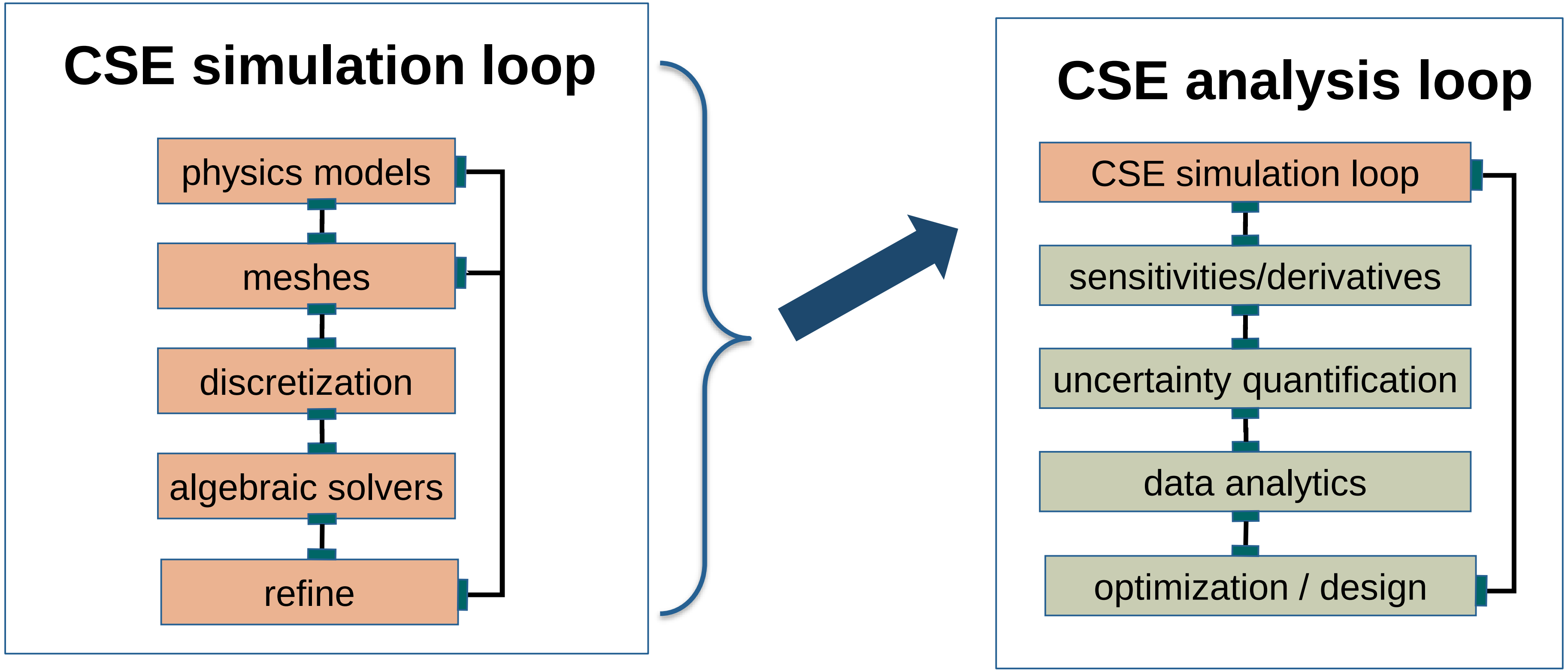
- Develop a mathematical model of the phenomenon of interest
- Approximate the model using a discrete representation
- Solve the discrete representation
- Adapt and refine the mesh or model
- Incorporate different physics, scales





Requires: mesh generation, partitioning, load balancing, high-order discretization, time integration, linear and nonlinear solvers, eigensolvers, mesh refinement, multiscale/multiphysics coupling methods, etc.

CSE analysis builds on the CSE simulation loop … and relies on even more numerical algorithms and software



Beyond interpretive simulations …

working toward predictive science

Requires: adjoints, sensitivities, algorithmic differentiation, sampling, ensemble simulations, uncertainty quantification, data analytics, optimization (derivative free and derivative based), inverse problems, etc.

First consider a very simple example

- 1D rod with one end in a hot water bath, the other in a cold water bath
- Mathematical model

$$\nabla^2 T = 0 \in \Omega$$

$$T(0) = 180^o \quad T(1) = 0^o$$

Hot water bath

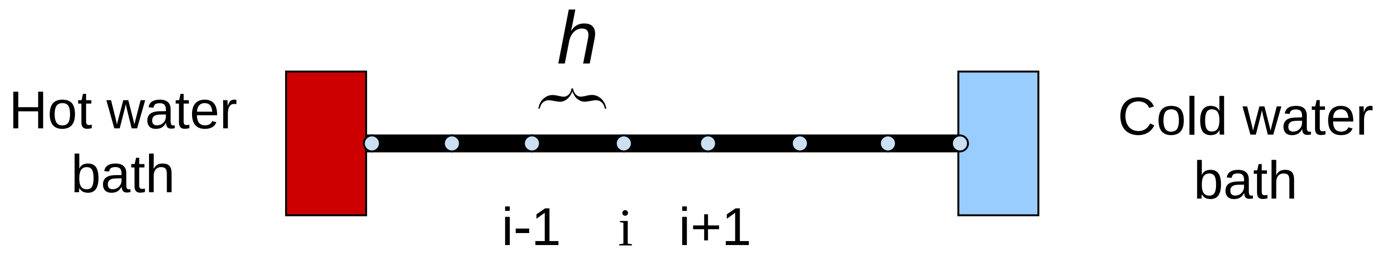
Cold water bath

The first step is to discretize the equations

- Approximate the derivatives in the continuous equations with a discrete representation that is easier to solve
- One approach: Finite differences

$$\nabla^2 T \approx (T_{i+1} - 2T_i + T_{i-1})/h^2 = 0$$

$$T_0 = 180^o \qquad T_n = 0^o$$



Then you can solve for the unknowns T_i

- Set up a matrix of the unknown coefficients
  - include the known boundary conditions
- Solve the linear system for T_i

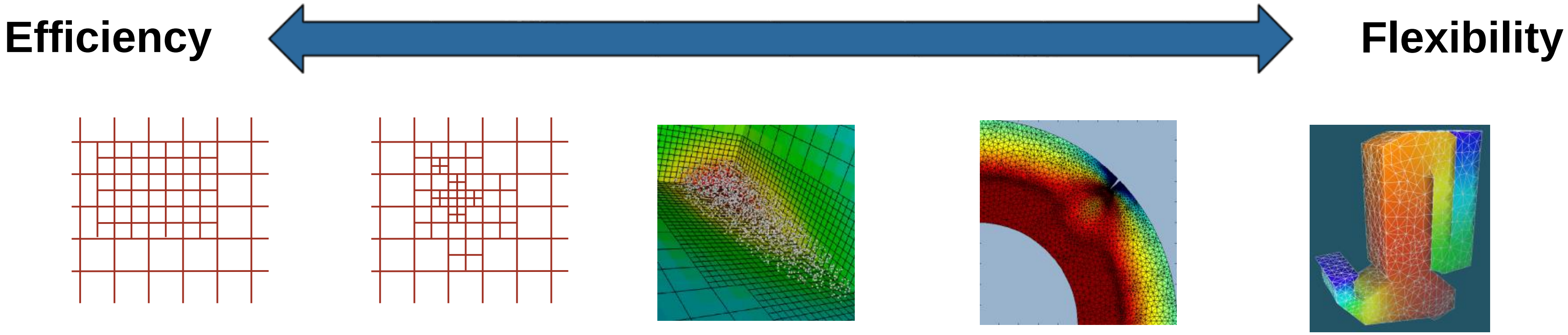
$$\begin{pmatrix} 2 & -1 & 0 & \dots & & 0 \\ -1 & 2 & -1 & 0 & \dots & 0 \\ 0 & -1 & 2 & -1 & 0 \dots & 0 \\ & & \dots & & & \\ 0 & \dots & & 0 & -1 & 2 \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ . \\ T_{n-1} \end{pmatrix} = \begin{pmatrix} 180\, h^2 \\ 0 \\ 0 \\ . \\ 0 \end{pmatrix}$$

- Visualize and analyze the results

EXASCALE COMPUTING PROJECT

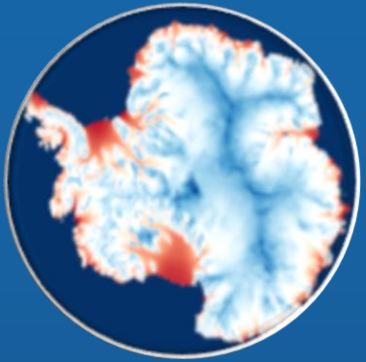
As problems get more complicated, so do the steps in the process

- Different discretization strategies exist for differing needs

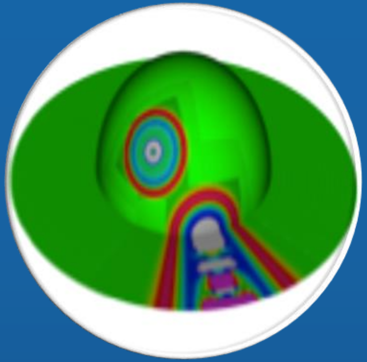


- Most problems are time dependent and nonlinear
  - Need higher algorithmic levels than linear solvers
- Increasingly combining multiple physical processes
  - Interactions require careful handling
- Goal-oriented problem solving requires optimization, uncertainty quantification

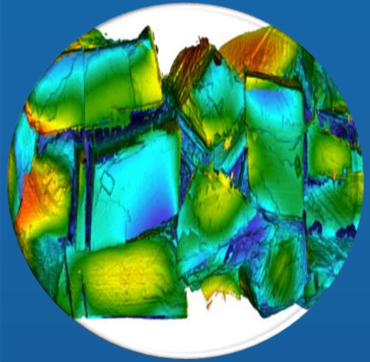
Structured grid efforts focus on high-order, mapped grids, embedded boundaries, AMR, and particles



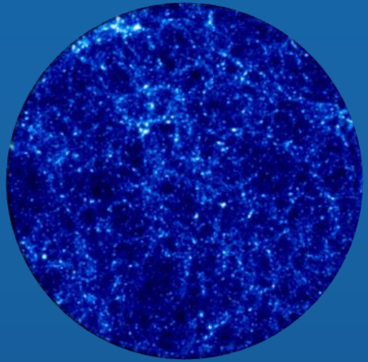
Structured AMR



Mapped-multiblock grids



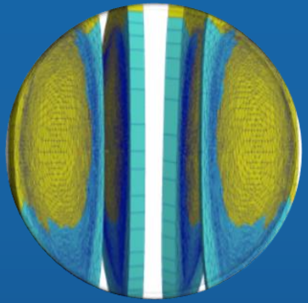
Embedded boundary methods



Particle-based methods

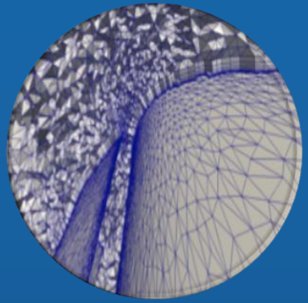
Application to cosmology, astrophysics, accelerator modeling, fusion, climate, subsurface reacting flows, low mach number combustion, etc.

Unstructured grid capabilities focus on adaptivity, high-order, and the tools needed for extreme scaling

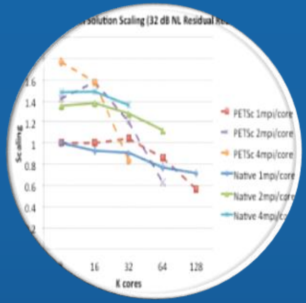


Parallel mesh infrastructures

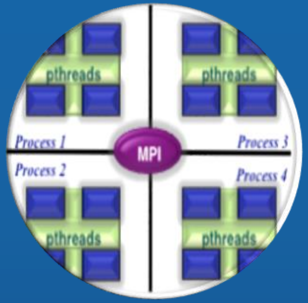
Dynamic load balancing



Mesh adaptation and quality control



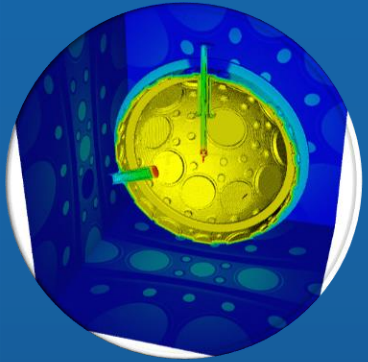
Parallel performance on unstructured meshes



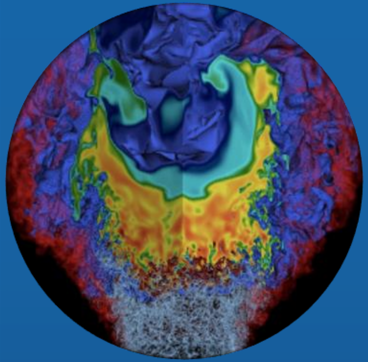
Architecture aware implementations

Application to fusion, climate, accelerator modeling, NNSA applications, nuclear energy, manufacturing processes, etc.

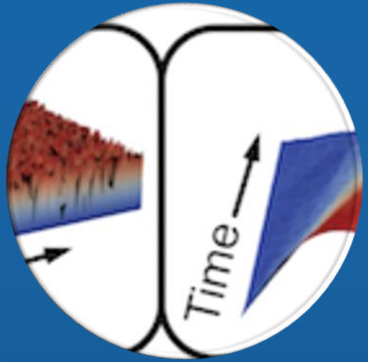
Time discretization methods provide efficient and robust techniques for stiff implicit, explicit and multi-rate systems



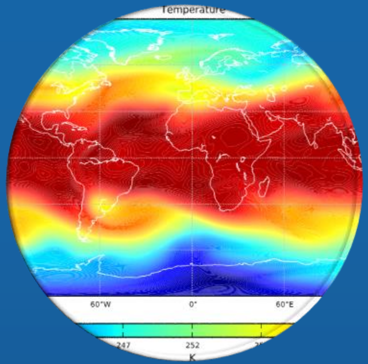
IMEX, Multi-rate methods



Spectral Deferred Correction Techniques



Parallel In Time

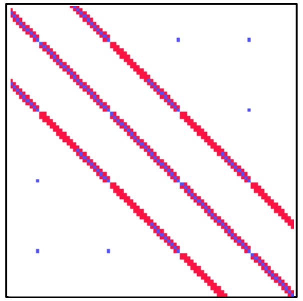


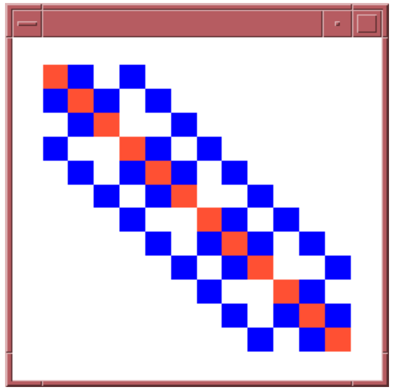
Constrained problems

Application to atmospheric dynamics, ice sheets, combustion, power grid, cosmology, subsurface, etc.

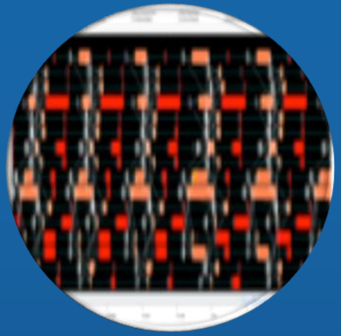
As problems grow in size, so do corresponding discrete systems

- Targeting applications with billions grid points and unknowns
- Most linear systems resulting from these techniques are LARGE and sparse
- Often most expensive solution step
- Solvers:
  - Direct methods (e.g. Gaussian Elimination)
  - Iterative methods (e.g. Krylov Methods)
    - Preconditioning is typically critical
    - Mesh quality affects convergence rate
- Many software tools deliver this functionality as <u>numerical libraries</u>
  - hypre, PETSc, SuperLU, Trilinos, etc.

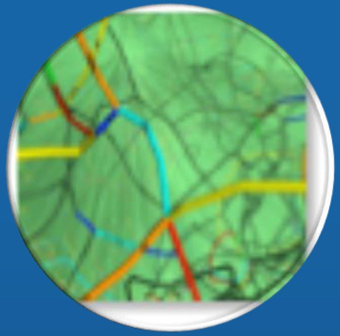




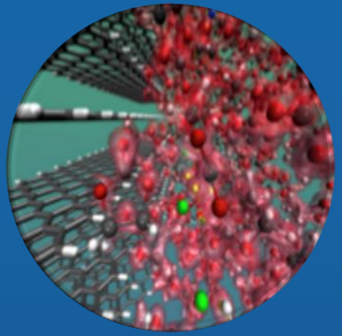
Research on algebraic systems provides key solution technologies to applications



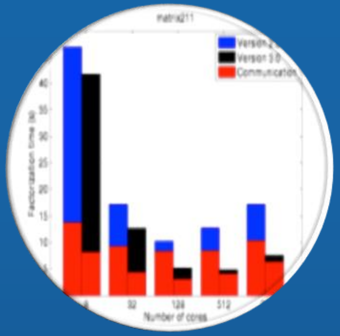
Linear system solution using direct and iterative solvers



Nonlinear system solution using acceleration techniques and globalized Newton methods



Eigensolvers using iterative techniques and optimization



Architecture aware implementations

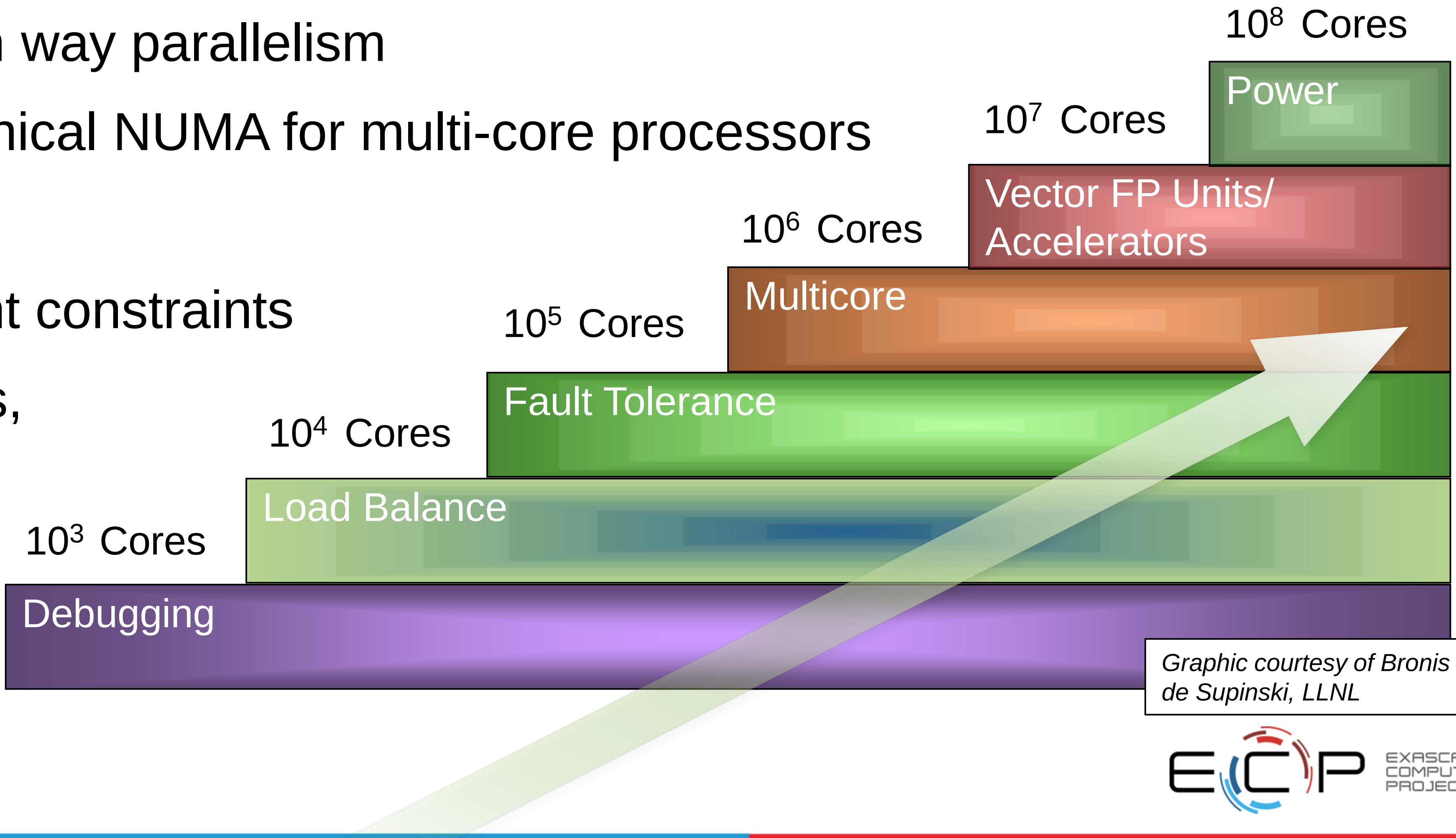
Application to fusion, nuclear structure calculation, quantum chemistry, accelerator modeling, climate, dislocation dynamics etc,

EXASCALE COMPUTING PROJECT

Simulation is significantly complicated by the change in computing architectures

Scientific computing software must address ever increasing challenges:

- Million to billion way parallelism
- Deeply hierarchical NUMA for multi-core processors
- Fault tolerance
- Data movement constraints
- Heterogeneous, accelerated architectures
- Power constraints

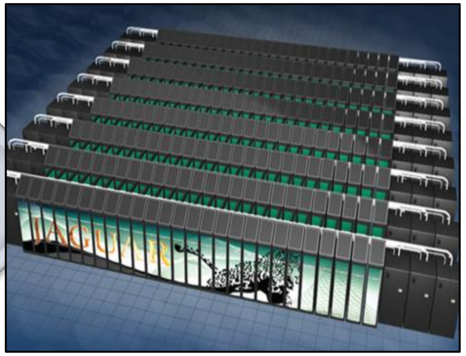


Graphic courtesy of Bronis de Supinski, LLNL

Research to improve performance on HPC platforms focuses on inter- and intra-node issues

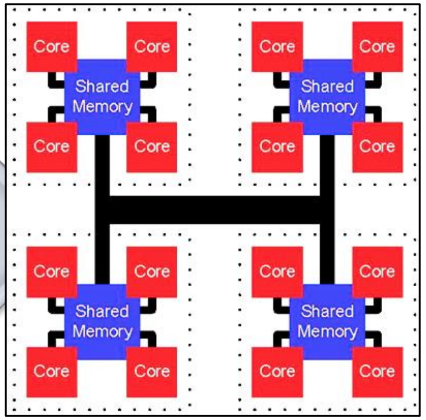
Inter-node: Massive Concurrency

- Reduce communication
- Increase concurrency
- Reduce synchronization
- Address memory footprint
- Enable large communication/computation overlap



Intra-node: Deep NUMA

- MPI + threads for many packages
- Compare task and data parallelism
- Thread communicator to allow passing of thread information among libraries
- Low-level kernels for vector operations that support hybrid programming models



EXASCALE COMPUTING PROJECT

Ongoing research addresses key bottlenecks on modern day computers

Reduce communication

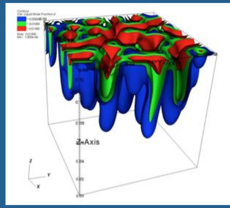
- AMG: develop non-Galerkin approaches, use redundancy or agglomeration on coarse grids, develop additive AMG variants (hypre) (2X improvement)
- Hierarchical partitioning optimizes communication at each level (Zoltan) (27% improvement in matrix-vector multiply)
- Relaxation and bottom solve in AMR multigrid (Chombo) (2.5X improvement in solver, 40% overall)

Increase concurrency

- New spectrum slicing eigensolver in PARPACK (Computes 10s of thousands of eigenvalues in small amounts of time)
- New pole expansion and selected inversion schemes (PEXSI) (now scales to over 100K cores)
- Utilize BG/Q architecture for extreme scaling demonstrations (PHASTA) (3.1M processes on 768K cores unstructured mesh calculation)

Reduce synchronization points

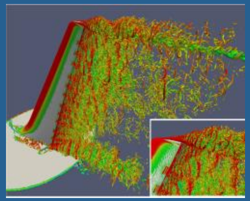
- Implemented pipelined versions of CG and conjugate residual methods; 4X improvement in speed (PETSc) (30% speed up on 32K cores)



Used in PFLOTRAN applications

Address memory footprint issues

- Predictive load balancing schemes for AMR (Zoltan) (Allows AMR runs to complete by maintaining memory footprint)
- Hybrid programming models



Used in PHASTA extreme scale applications

Increase communication and computation overlap

- Improved and stabilized look-ahead algorithms (SuperLU) (3X run time improvement)

Used in Omega3P accelerator simulations

EXASCALE COMPUTING PROJECT

Software libraries facilitate CSE progress

- **Software library:** a high-quality, encapsulated, documented, tested, and multiuse software collection that provides functionality commonly needed by application developers
  - Organized for the purpose of being reused by independent (sub)programs
  - User needs to know only library interface (not internal details) … when and how to use library functionality appropriately

- **Key advantages** of software libraries
  - Contain complexity
  - Leverage library developer expertise
  - Reduce application coding effort
  - Encourage sharing of code, ease distribution of code

- References:
  - https://en.wikipedia.org/wiki/Library_(computing)
  - What are Interoperable Software Libraries? Introducing the xSDK

Broad range of HPC numerical software

Some packages with general-purpose, reusable algorithmic infrastructure in support of high-performance CSE:

- **AMReX** – https://github.com/AMReX-codes/amrex ☆
- **Chombo** - https://commons.lbl.gov/display/chombo ☆
- **Clawpack** - http://www.clawpack.org
- **Deal.II** - https://www.dealii.org
- **FEniCS** - https://fenicsproject.org
- **hypre** - http://www.llnl.gov/CASC/hypre ☆
- **libMesh** - https://libmesh.github.io
- **MAGMA** - http://icl.cs.utk.edu/magma ☆
- **MFEM** - http://mfem.org ☆
- **PETSc/TAO** – http://www.mcs.anl.gov/petsc ☆
- **PUMI** - http://github.com/SCOREC/core ☆
- **SUNDIALS** - http://computation.llnl.gov/casc/sundials ☆
- **SuperLU** - http://crd-legacy.lbl.gov/~xiaoye/SuperLU ☆
- **Trilinos** - https://trilinos.org ☆
- **Uintah** - http://www.uintah.utah.edu
- **waLBerla** - http://www.walberla.net

See info about scope, performance, usage, and design, including:

- tutorials
- demos
- examples
- how to contribute

☆ Discussed today: Gallery of highlights

… and many, many more ... Explore, use, contribute!

ECP applications need sustainable coordination among ECP math libraries

- **ECP application team interviews:**

  Astro, NWChemEx, WDMAPP, ExaFel, GAMESS, ExaSGD, Subsurface, EXAALT, WarpX, ExaAM, MFIX-Exa, ATDM (LANL, LLNL, SNL) apps, CoPA, AMREX, CEED, CODAR

- **Many ECP app teams rely on math libraries, often in combination**

  Need **sustainable coordinated xSDK releases** and **increasing interoperability** among xSDK packages to achieve good performance and easily access advanced algorithms and data structures

hypre high performance preconditioners
PETSc/TAO
SuperLU
TRILINOS
MAGMA
MFEM
sundials
DTK
PLASMA
TASMANIAN
FleCSI
STRUMPACK

Software libraries are not enough

See also Track 6:
Software Productivity (Aug 8)

Apps need to use software packages **in combination**

"The way you get programmer productivity is by eliminating lines of code you have to write."

– Steve Jobs, Apple World Wide Developers Conference, Closing Keynote, 1997

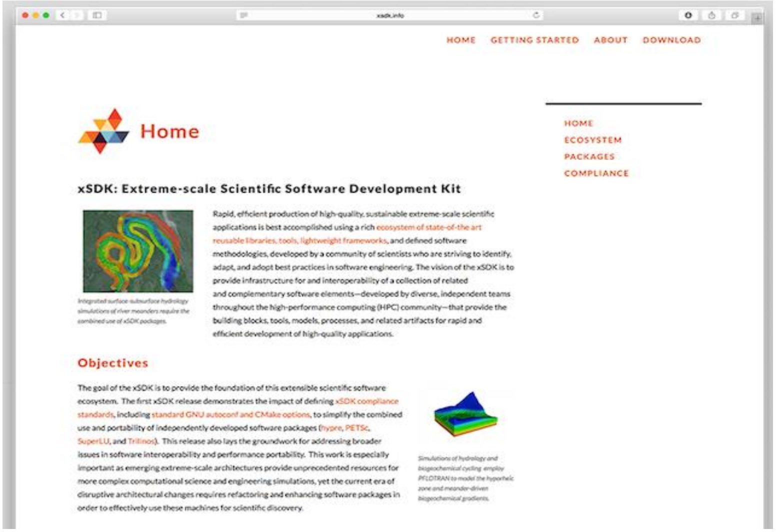
- **Need consistency** of compiler (+version, options), 3rd-party packages, etc.
- **Namespace and version conflicts** make simultaneous build/link of packages difficult
- **Multilayer interoperability** requires careful design and sustainable coordination

Building the foundation of a highly effective extreme-scale scientific software ecosystem

Focus: Increasing the functionality, quality, and interoperability of important scientific libraries, domain components, and development tools

Impact:

- Improved code quality, usability, access, sustainability
- Inform potential users that an xSDK member package can be easily used with other xSDK packages
- Foundation for work on performance portability ,deeper levels of package interoperability

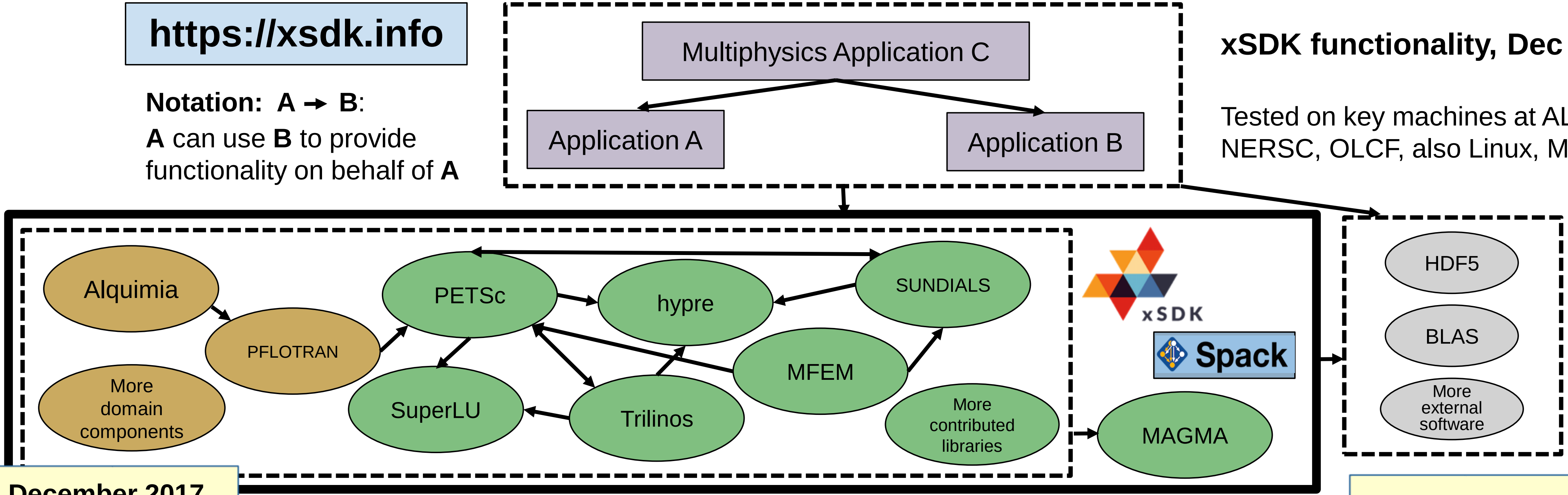


website: xSDK.info

xSDK Version 0.3.0: December 2017

https://xsdk.info

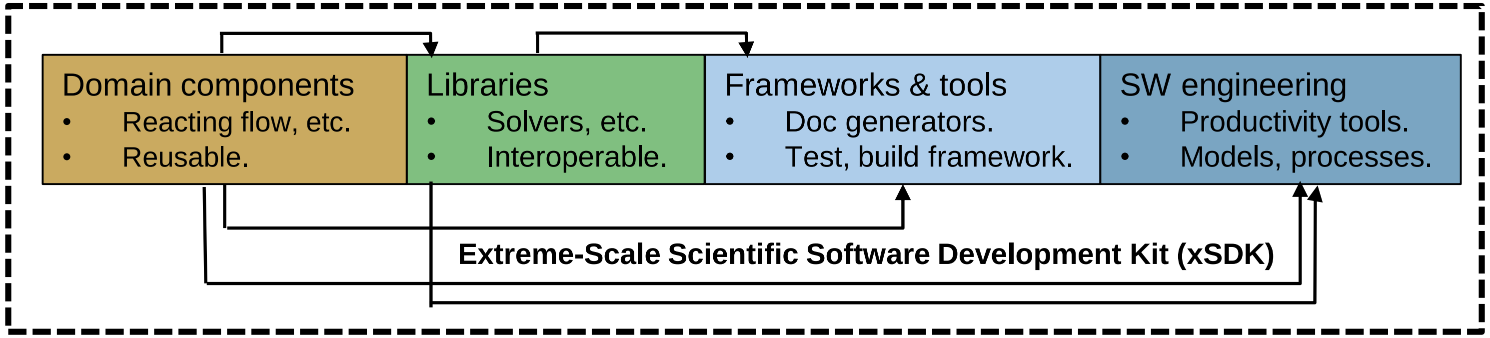
Notation: A → B:
A can use **B** to provide functionality on behalf of **A**



xSDK functionality, Dec 2017

Tested on key machines at ALCF, NERSC, OLCF, also Linux, Mac OS X

December 2017
- 7 math libraries
- 2 domain components
- Spack xSDK installer
- 16 mandatory xSDK community policies



Working toward next xSDK release, fall 2018:

Albany, AMReX, Chombo, CrunchFlow/CrunchTope, deal.II, DTK, Omega_h, PHIST, PLASMA, PUMi, SLEPc, STRUMPACK, TASMANIAN

The xSDK is using Spack to deploy its software

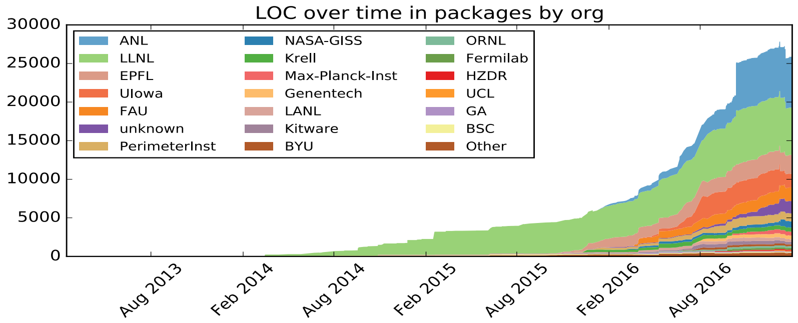
- The xSDK packages depend on a number of open source libraries
- Spack is a package manager for HPC
- Spack allows the xSDK to be deployed with a single command
  - User can optionally choose compilers, MPI implementation, and build options
  - Will soon support combinatorial test dashboards for all xSDK packages

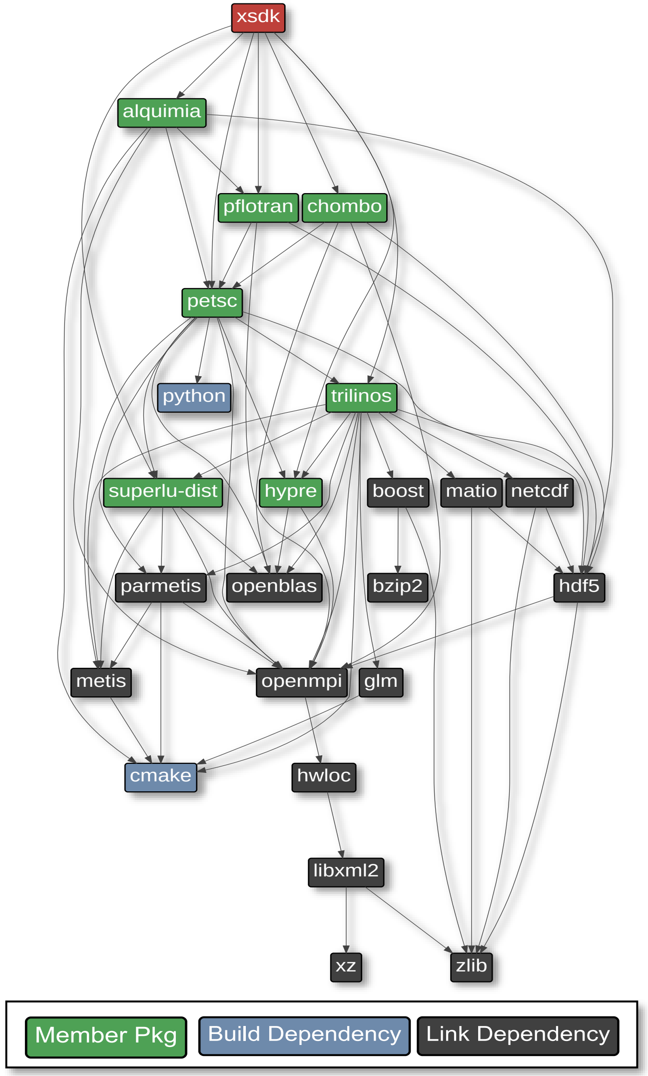
xSDK

Spack

Spack has grown into a thriving open source community

- Over 140 contributors
- Over 40 organizations
- Over 1,400 packages
- Over 75% of package code contributed from outside LLNL





github.com/LLNL/spack

EXASCALE COMPUTING PROJECT

xSDK community policies

xSDK
Version 0.3.0, Nov 2017

<u>xSDK compatible package</u>: Must satisfy mandatory xSDK policies:

M1. Support xSDK community GNU Autoconf or CMake options.
M2. Provide a comprehensive test suite.
M3. Employ user-provided MPI communicator.
M4. Give best effort at portability to key architectures.
M5. Provide a documented, reliable way to contact the development team.
M6. Respect system resources and settings made by other previously called packages.
M7. Come with an open source license.
M8. Provide a runtime API to return the current version number of the software.
M9. Use a limited and well-defined symbol, macro, library, and include file name space.
M10. Provide an accessible repository (not necessarily publicly available).
M11. Have no hardwired print or IO statements.
M12. Allow installing, building, and linking against an outside copy of external software.
M13. Install headers and libraries under <prefix>/include/ and <prefix>/lib/.
M14. Be buildable using 64 bit pointers. 32 bit is optional.
M15. All xSDK compatibility changes should be sustainable.
M16. The package must support production-quality installation compatible with the xSDK install tool and xSDK metapackage.

Also specify **recommended policies**, which currently are encouraged but not required:

R1. Have a public repository.
R2. Possible to run test suite under valgrind in order to test for memory corruption issues.
R3. Adopt and document consistent system for error conditions/exceptions.
R4. Free all system resources it has acquired as soon as they are no longer needed.
R5. Provide a mechanism to export ordered list of library dependencies.

<u>xSDK member package</u>: Must be an xSDK-compatible package, *and* it uses or can be used by another package in the xSDK, and the connecting interface is regularly tested for regressions.

We welcome feedback. What policies make sense for your apps and packages?

https://xsdk.info/policies

xSDK: Primary delivery mechanism for ECP math libraries' continual advancements toward predictive science

As motivated and validated by the needs of ECP applications:

ECP Math libraries

- Performance on new node architectures
- Extreme strong scalability
- Advanced, coupled multiphysics, multiscale
- Optimization, UQ, solvers, discretizations
- Interoperability, complementarity: xSDK

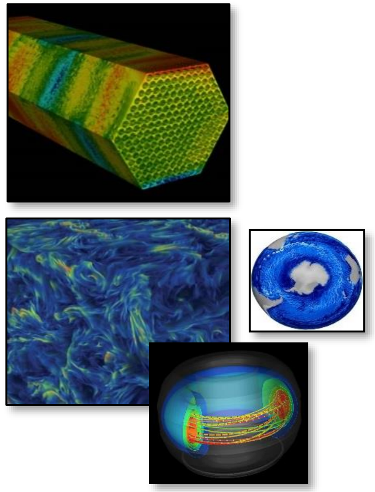
Next-generation algorithms

Advances in data structures for new node architectures

Improving library quality, sustainability, interoperability

Toward predictive scientific simulations

Increasing performance, portability, productivity



Timeline: xSDK release 1 → xSDK release 2 → ..... → xSDK release n

Gallery of highlights

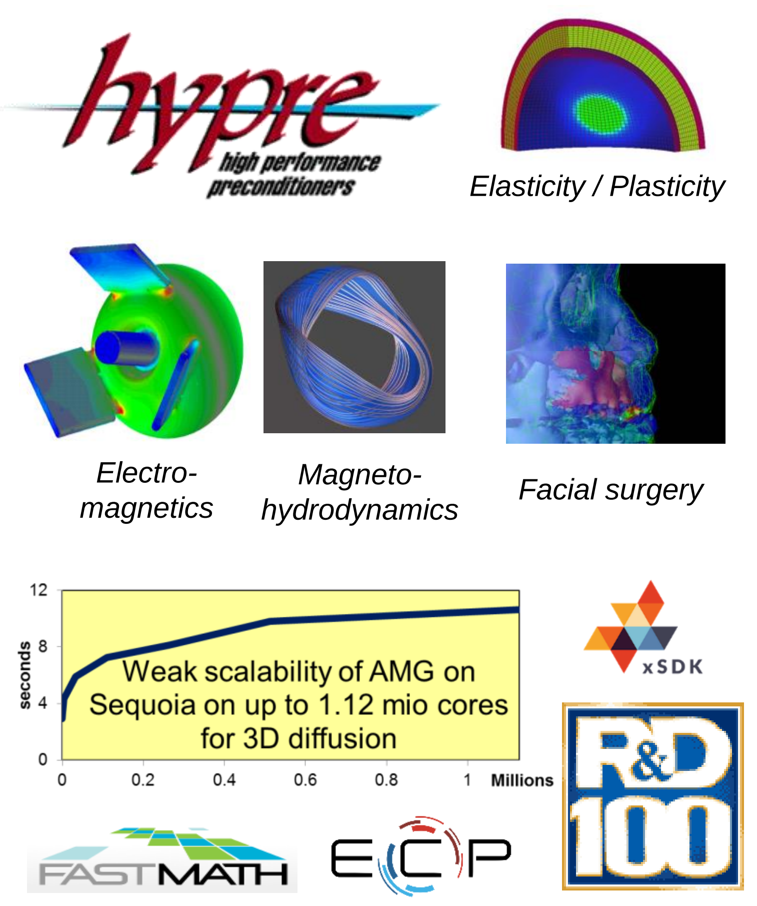
- Overview of HPC numerical software packages
- 1 slide per package, emphasizing key capabilities, highlights, and where to go for more info
  - Listed first
    - Package featured in ATPESC 2018 lectures and hands-on lessons
  - Listed next
    - Additional highlighted packages (<u>not a comprehensive list</u>)

hypre

Lawrence Livermore National Laboratory

Highly scalable multilevel solvers and preconditioners. Unique user-friendly interfaces. Flexible software design. Used in a variety of applications. Freely available.

- **Conceptual interfaces**
  - Structured, semi-structured, finite elements, linear algebraic interfaces
  - Provide natural “views” of the linear system
  - Provide for more efficient (scalable) linear solvers through more effective data storage schemes and more efficient computational kernels
- **Scalable preconditioners and solvers**
  - Structured and unstructured algebraic multigrid solvers
  - Maxwell solvers, H-div solvers
  - Multigrid solvers for nonsymmetric systems: pAIR, MGR
  - Matrix-free Krylov solvers
- **Open source software**
  - Used worldwide in a vast range of applications
  - Can be used through PETSc and Trilinos
  - Available on github: https://www.github.com/LLNL/hypre



http://www.llnl.gov/CASC/hypre

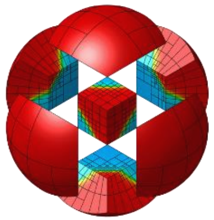
EXASCALE COMPUTING PROJECT

MFEM

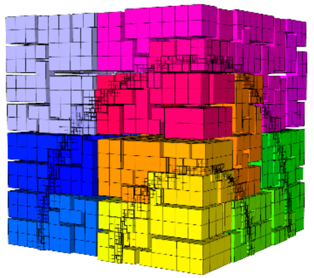
Lawrence Livermore National Laboratory

Free, lightweight, scalable C++ library for finite element methods. Supports arbitrary high order discretizations and meshes for wide variety of applications.

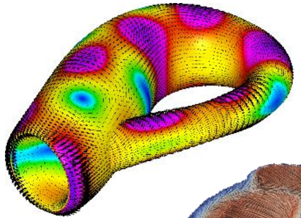
- **Flexible discretizations on unstructured grids**
  - Triangular, quadrilateral, tetrahedral and hexahedral meshes.
  - Local conforming and non-conforming refinement.
  - Bilinear/linear forms for variety of methods: Galerkin, DG, DPG, …
- **High-order and scalable**
  - Arbitrary-order H1, H(curl), H(div)- and L2 elements. Arbitrary order curvilinear meshes.
  - MPI scalable to millions of cores. Enables application development on wide variety of platforms: from laptops to exascale machines.
- **Built-in solvers and visualization**
  - Integrated with: HYPRE, SUNDIALS, PETSc, SUPERLU, …
  - Accurate and flexible visualization with VisIt and GLVis
- **Open source software**
  - LGPL-2.1 with thousands of downloads/year worldwide.
  - Available on GitHub, also via OpenHPC, Spack. Part of ECP's CEED co-design center.



High order curved elements

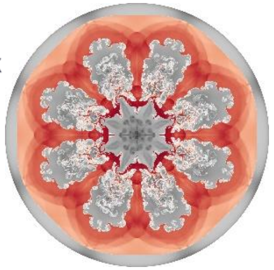


Parallel non-conforming AMR



Surface meshes

Heart modelling



Compressible flow ALE simulations

http://mfem.org

EXASCALE COMPUTING PROJECT

PETSc/TAO

Portable, Extensible Toolkit for Scientific Computation / Toolkit for Advanced Optimization

Scalable algebraic solvers for PDEs. Encapsulate parallelism in high-level objects. Composable, hierarchical, nested, extensible. Active & supported user community. Full API from Fortran, C/C++, Python.

Optimization

Time Integrators

Nonlinear Algebraic Solvers

Krylov Subspace Solvers

Preconditioners

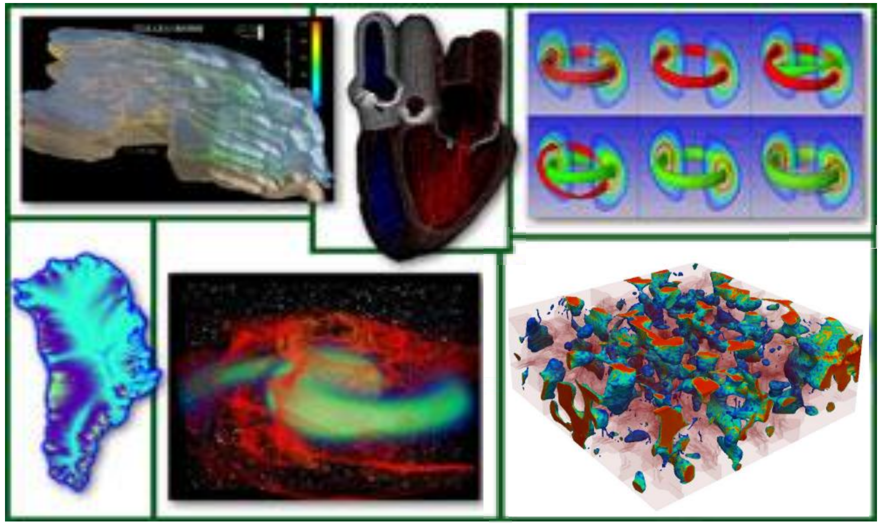
Domain-Specific Interfaces
- **Networks**
- **Quadtree / Octree**
- **Unstructured Mesh**
- **Structured Mesh**

Vectors | **Index Sets** | **Matrices**

Computation & Communication Kernels

- **Easy customization and composability of solvers <u>at runtime</u>**
  - Enables optimality via flexible combinations of physics, algorithmics, architectures
  - Try new algorithms by composing new/existing algorithms (multilevel, domain decomposition, splitting, etc.)
- **Portability & performance**
  - Largest DOE machines, also clusters, laptops
  - Thousands of users worldwide

Argonne NATIONAL LABORATORY



PETSc provides the backbone of diverse scientific applications.

clockwise from upper left: hydrology, cardiology, fusion, multiphase steel, relativistic matter, ice sheet modeling

R&D 100 ECP xSDK FASTMATH

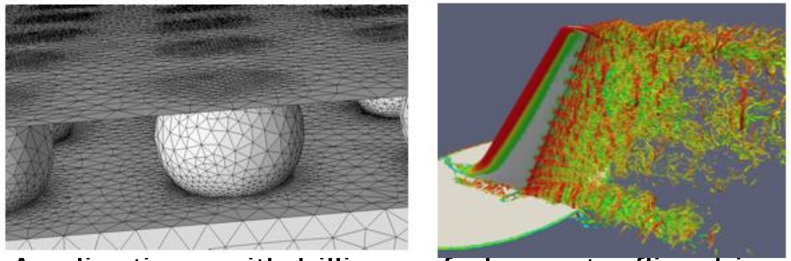
https://www.mcs.anl.gov/petsc

ECP EXASCALE COMPUTING PROJECT

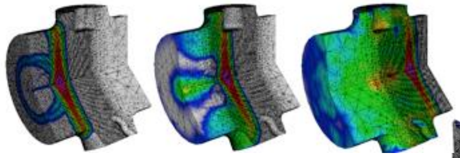
PUMi: Parallel Unstructured Mesh Infrastructure

Parallel management and adaptation of unstructured meshes.
Interoperable components to support the development of unstructured mesh simulation workflows

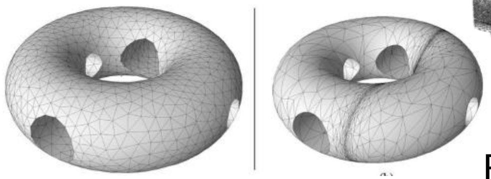
- **Core functionality**
  - Distributed, conformant mesh with entity migration, remote read only copies, fields and their operations
  - Link to the geometry and attributes
  - Mesh adaptation (straight and curved), mesh motion
  - Multi-criteria partition improvement
  - Distributed mesh support for Particle In Cell methods
- **In-memory integrations developed**
  - MFEM: High order FE framework
  - PHASTA: FE for turbulent flows
  - FUN3D: FV CFD
  - Proteus: Multiphase FE
  - ACE3P: High order FE for EM
  - M3D-C1: FE based MHD
  - Nektar++: High order FE for flow
  - Albany/Trilinos: Multi-physics FE
- **Designed for integration into existing codes**
  - Conformant with XSDK
  - Permissive license enables integration with open and closed-source codes



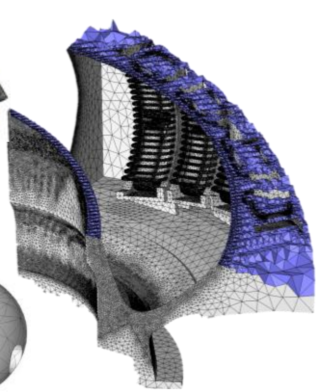
Applications with billions of elements: flip-chip (L), flow control (R)



Mesh adaptation for evolving features



Anisotropic adaptation for curved meshes



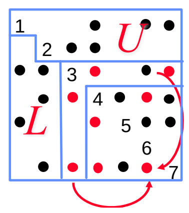
RF antenna and plasma surface in vessel.

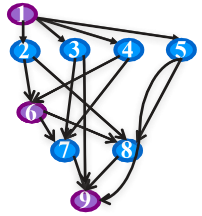
Source Code: github.com/SCOREC/core
Paper: www.scorec.rpi.edu/REPORTS/2014-9.pdf

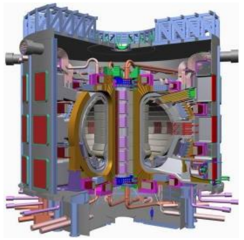
SuperLU

Supernodal Sparse LU Direct Solver. Unique user-friendly interfaces. Flexible software design. Used in a variety of applications. Freely available.

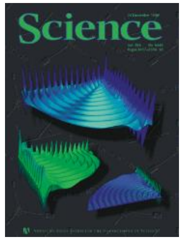
- **Capabilities**
  - Serial (thread-safe), shared-memory (SuperLU_MT, OpenMP or Pthreads), distributed-memory (SuperLU_DIST, hybrid MPI+ OpenM + CUDA).
    - Implemented in C, with Fortran interface
  - Sparse LU decomposition, triangular solution with multiple right-hand sides
  - Incomplete LU (ILU) preconditioner in serial SuperLU
  - Sparsity-preserving ordering:
    - Minimum degree ordering applied to A^TA or A^T+A
    - Nested dissection ordering applied to A^TA or A^T+A [(Par)METIS, (PT)-Scotch]
  - User-controllable pivoting: partial pivoting, threshold pivoting, static pivoting
  - Condition number estimation, iterative refinement.
  - Componentwise error bounds
- **Performance**
  - Factorization strong scales to 24,000 cores (IPDPS'18)
  - Triangular solve strong scales to 4000 cores (CSC'18)
- **Open source software**
  - Used worldwide in a vast range of applications, can be used through PETSc and Trilinos, available on github







ITER tokamak



quantum mechanics

Wiidely adopted in commercial software, including AMD (circuit simulation), Boeing (aircraft design), Chevron, ExxonMobile (geology), Cray's LibSci, FEMLAB, HP's MathLib, IMSL, NAG, SciPy, OptimaNumerics, Walt Disney Animation.

FASTMATH ECP xSDK

http://crd-legacy.lbl.gov/~xiaoye/SuperLU

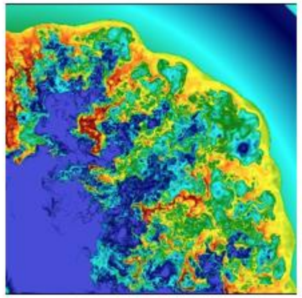
EXASCALE COMPUTING PROJECT

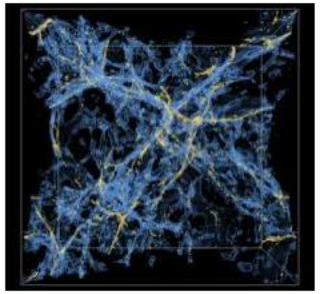
AMReX

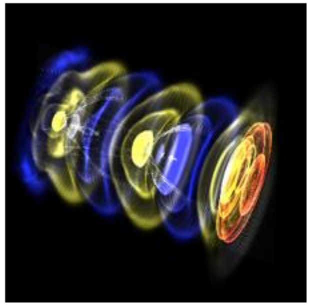
Block-structured adaptive mesh refinement framework. Support for hierarchical mesh and particle data with embedded boundary capability.

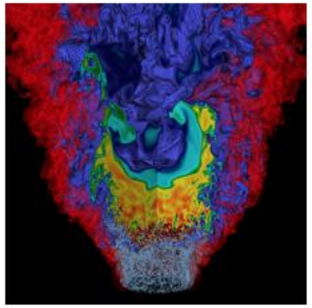
- **Capabilities**
  - Support for solution of PDEs on hierarchical adaptive mesh with particles and embedded boundary representation of complex geometry
    - Core functionality in C++ with frequent use of Fortran90 kernels
  - Support for multiple modes of time integration
  - Support for explicit and implicit single-level and multilevel mesh operations, multilevel synchronization, particle, particle-mesh and particle-particle operations
  - Hierarchical parallelism -- hybrid MPI + OpenMP with logical tiling to work efficiently on new multicore architectures, GPU support in progress
  - Native multilevel geometric multigrid solvers for cell-centered and nodal data
  - Highly efficient parallel I/O for checkpoint/restart and for visualization – native format supported by Visit, Paraview, yt
  - Tutorial examples available in download
- **Open source software**
  - Used for a wide range of applications including accelerator modeling, astrophysics, combustion, cosmology, multiphase flow, phase field modeling…
  - Freely available on github

Examples of AMReX applications









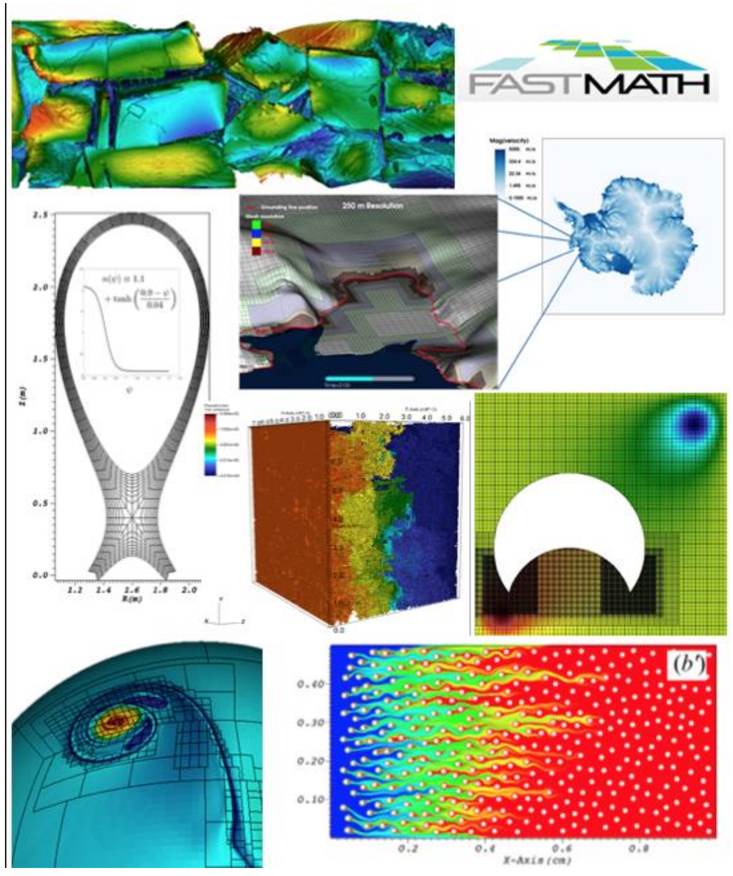
FASTMATH ECP

https://www.github.com/AMReX-Codes/amrex

Chombo

Scalable adaptive mesh refinement framework. Enables implementing scalable AMR applications with support for complex geometries.

- **Adaptive Mesh Refinement (AMR)**
  - Block structured AMR dynamically focuses computational effort where needed to improve solution accuracy
  - Designed as a developers' toolbox for implementing scalable AMR applications
  - Implemented in C++/Fortran
  - Solvers for hyperbolic, parabolic, and elliptic systems of PDEs
- **Complex Geometries**
  - Embedded-boundary (EB) methods use a cut-cell approach to embed complex geometries in a regular Cartesian mesh
  - EB mesh generation is extremely efficient
  - Structured EB meshes make high performance easier to attain
- **Higher-order finite-volume**
  - Higher (4th)-order schemes reduce memory footprint & improve arithmetic intensity
  - Good fit for emerging architectures
  - Both EB and mapped-multiblock approaches to complex geometry



http://Chombo.lbl.gov

MAGMA

THE UNIVERSITY OF TENNESSEE KNOXVILLE

Linear algebra solvers and spectral decompositions for hardware accelerators.
Portable dense direct and sparse iterative solvers for GPUs and coprocessors.

- **Dense Linear Algebra Solvers**
  - Linear systems of equations
  - Linear least squares
  - Singular value decomposition
- **Matrix spectrum methods**
  - Symmetric and non-symmetric eigenvalues
  - Generalized eigenvalue problems
  - Singular Value Decomposition
- **Sparse Solvers & Tensor Computations**

MAGMA SPARSE

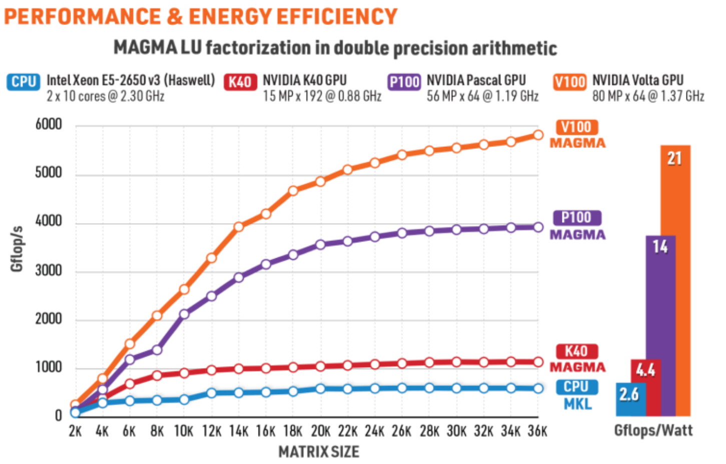
ROUTINES	BiCG, BiCGSTAB, Block-Asynchronous Jacobi, CG, CGS, GMRES, IDR, Iterative refinement, LOBPCG, LSQR, QMR, TFQMR
PRECONDITIONERS	ILU / IC, Jacobi, ParILU, ParILUT, Block Jacobi, ISAI
KERNELS	SpMV, SpMM
DATA FORMATS	CSR, ELL, SELL-P, CSR5, HYB

FEATURES AND SUPPORT

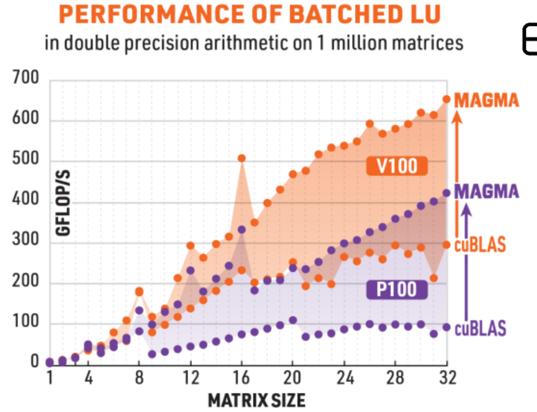
- MAGMA 2.3 FOR CUDA
- clMAGMA 1.4 FOR OpenCL
- MAGMA MIC 1.4 FOR Intel Xeon Phi

CUDA	OpenCL	Intel Xeon Phi	
●	●	●	Linear system solvers
●	●	●	Eigenvalue problem solvers
●	●		Auxiliary BLAS
●			Batched LA
●		●	Sparse LA
●	●	●	CPU/GPU Interface
●	●	●	Multiple precision support
●			Non-GPU-resident factorizations
●	●	●	Multicore and multi-GPU support
● NEW			MAGMA Analytics/DNN
●	●	●	LAPACK testing
●	●	●	Linux
●	●		Windows
●	●		Mac OS

PERFORMANCE & ENERGY EFFICIENCY



PERFORMANCE OF BATCHED LU



xSDK

http://icl.utk.edu/magma/

EXASCALE COMPUTING PROJECT

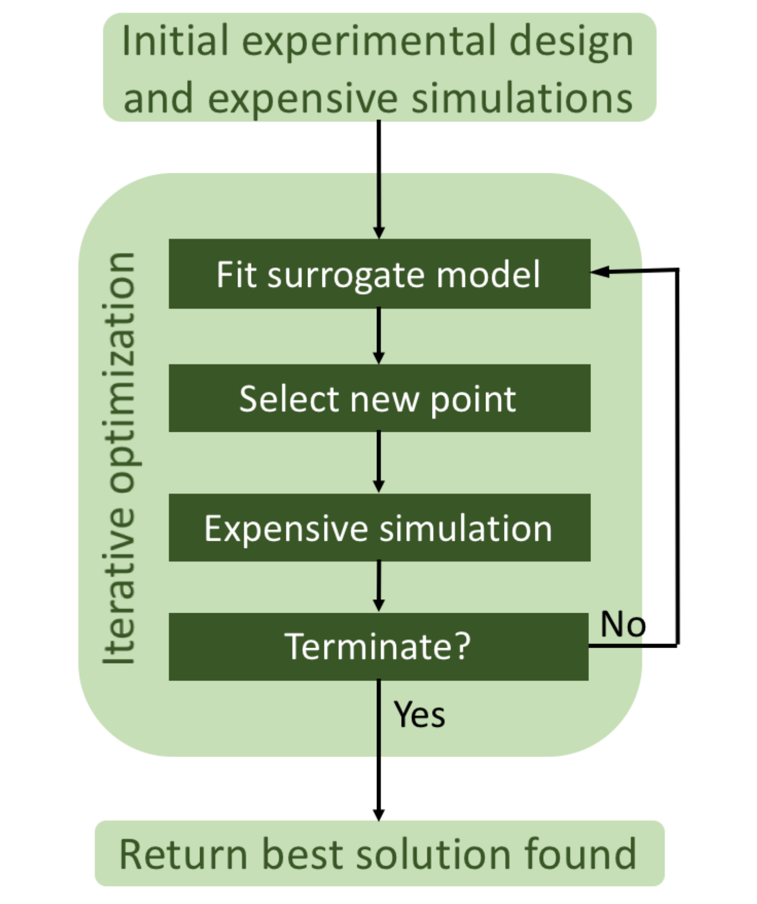
MATSuMoTo

MATLAB Surrogate Model Toolbox

Efficient optimization of computationally-expensive black-box problems. For integer, mixed-integer, and continuous variables. Your choice of surrogate model, sampling method, and initial design strategy. Easy to use. Freely available.

- **Capabilities**
  - Efficient solution of parameter optimization problems that involve time-consuming black-box HPC simulations during the objective function evaluation
  - Surrogate models approximate the expensive function and aid in iterative selection of sample points
  - Adaptive sampling for continuous, integer, and mixed-integer problems *without* relaxation of integer constraints

- **Available User options**
  - **Surrogate model choices:** radial basis functions, polynomial regression, multivariate adaptive regression splines, surrogate model ensembles
  - **Iterative sample point selection:** local perturbations, global candidate points, minimization over the surrogate model
  - **Initial experimental design:** Latin hypercube, symmetric Latin hypercube, design space corners



https://optimization.lbl.gov/downloads

SUNDIALS

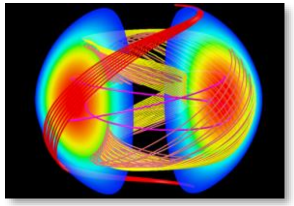
SUite of Nonlinear DIfferential /ALgebraic equation Solvers

Adaptive time integrators for ODEs and DAEs and efficient nonlinear solvers.
Used in a variety of applications. Freely available. Encapsulated parallelism.

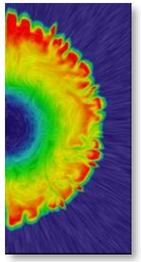
- **ODE integrators:**
  - CVODE(S): variable order and step BDF (stiff) and Adams (non-stiff)
  - ARKode: variable step implicit, explicit, and additive IMEX Runge-Kutta
- **DAE integrators**: IDA(S) - variable order and step BDF integrators
- **Sensitivity Analysis (SA):** CVODES and IDAS provide forward and adjoint SA
- **Nonlinear Solvers:** KINSOL - Newton-Krylov, Picard, and accelerated fixed point
- **Modular Design**
  - Written in C with interfaces to Fortran
  - Users can supply own data structures
  - Optional use structures: serial, MPI, threaded, CUDA, RAJA, hypre, & PETSc
- **Open Source Software**
  - Freely available (BSD License) from LLNL site, github, and Spack
  - CMake-based portable build system
  - Can be used from MFEM and PETSc

Lawrence Livermore National Laboratory

SUNDIALS is used by thousands worldwide in applications from research and industry.



Magnetic reconnection

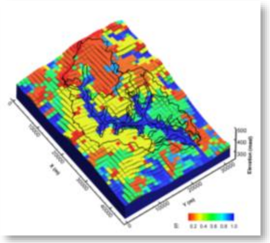


Core collapse super-nova

FASTMATH

ECP

Dislocation dynamics



Subsurface flow

xSDK

SUNDIALS is supported by extensive documentation, a sundials-users email list, and an active user community

http://www.llnl.gov/CASC/sundials

EXASCALE COMPUTING PROJECT

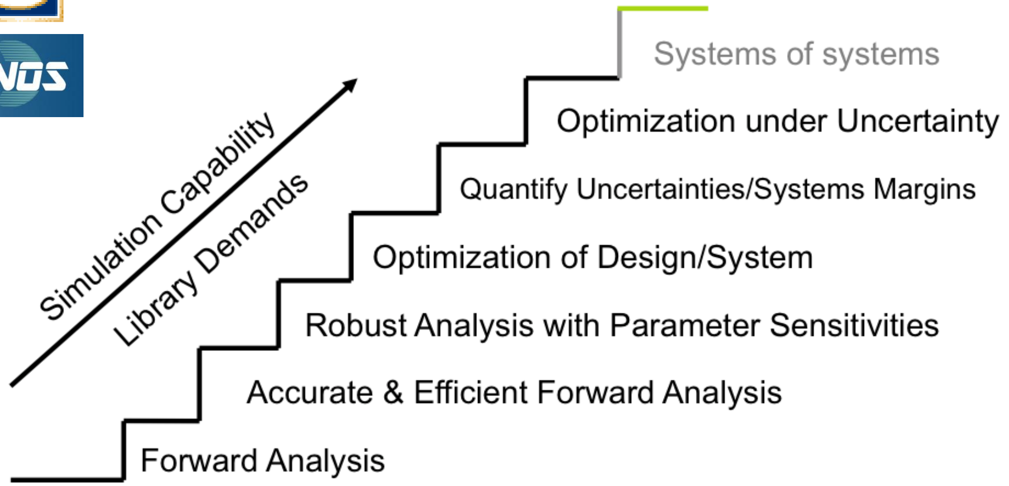
Trilinos

Sandia National Laboratories

Optimal kernels to optimal solutions. Over 60 packages. Laptops to leadership systems. Next-gen systems, multiscale/multiphysics, large-scale graph analysis.

- **Optimal kernels to optimal solutions**
  - Geometry, meshing
  - Discretization, load balancing
  - Scalable linear, nonlinear, eigen, transient, optimization, UQ solvers
  - Scalable I/O GPU, manycore
- **60+ packages**
  - Other distributions: Cray LIBSCI, Github repo
  - Thousands of users, worldwide distribution
  - Laptops to leadership systems

Transforming Computational Analysis To Support High Consequence Decisions



Each stage requires *greater performance* and *error control* of prior stages:
Always will need: more accurate and scalable methods.
more sophisticated tools.

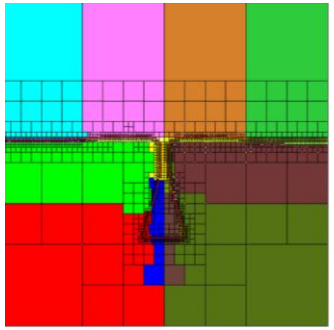
https://trilinos.org

EXASCALE COMPUTING PROJECT

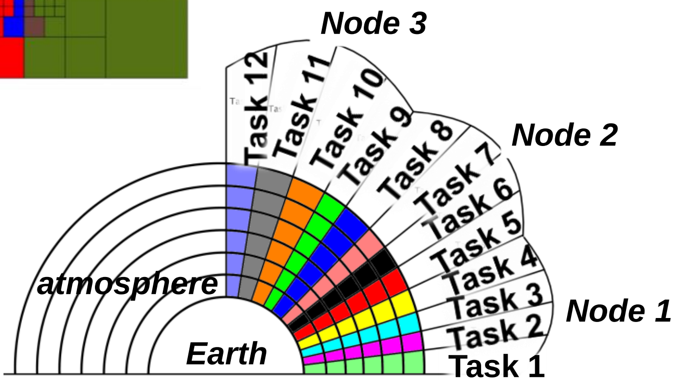
Zoltan/Zoltan2

Parallel partitioning, load balancing, task placement, graph coloring, matrix ordering, unstructured communication utilities, distributed directories

- **Partitioning & load-balancing support many applications**
  - Fast geometric methods maintain spatial locality of data (e.g., for adaptive finite element methods, particle methods, crash/contact simulations)
  - Graph and hypergraph methods explicitly account for communication costs (e.g., for electrical circuits, finite element meshes, social networks)
  - Single interface to popular partitioning TPLs: XtraPuLP (SNL, RPI); ParMA (RPI); PT-Scotch (U Bordeaux); ParMETIS (U Minnesota)
- **Architecture-aware MPI task placement reduces application communication time**
  - Places interdependent MPI tasks on "nearby" nodes in computing architecture
  - Reduces communication time and network congestion
- **Use as a stand-alone library or as a Trilinos component**



Zoltan's fast, geometric partitioner redistributes work to maintain load balance in a surface deposition simulation with adaptive meshing



Zoltan2's task placement reduced communication time in E3SM HOMME (atmospheric modeling) by up to 31% on 16K cores of IBM BG/Q

Sandia National Laboratories · FASTMATH · ASC

https://www.cs.sandia.gov/Zoltan

EXASCALE COMPUTING PROJECT

Sign up for 1-on-1 discussions with numerical software developers

Via Google docs folder: See link in email:

- **Your email address**
- Select 1st, 2nd, and 3rd priorities for short 1-1 meetings with expert developers
- Brief description of interests

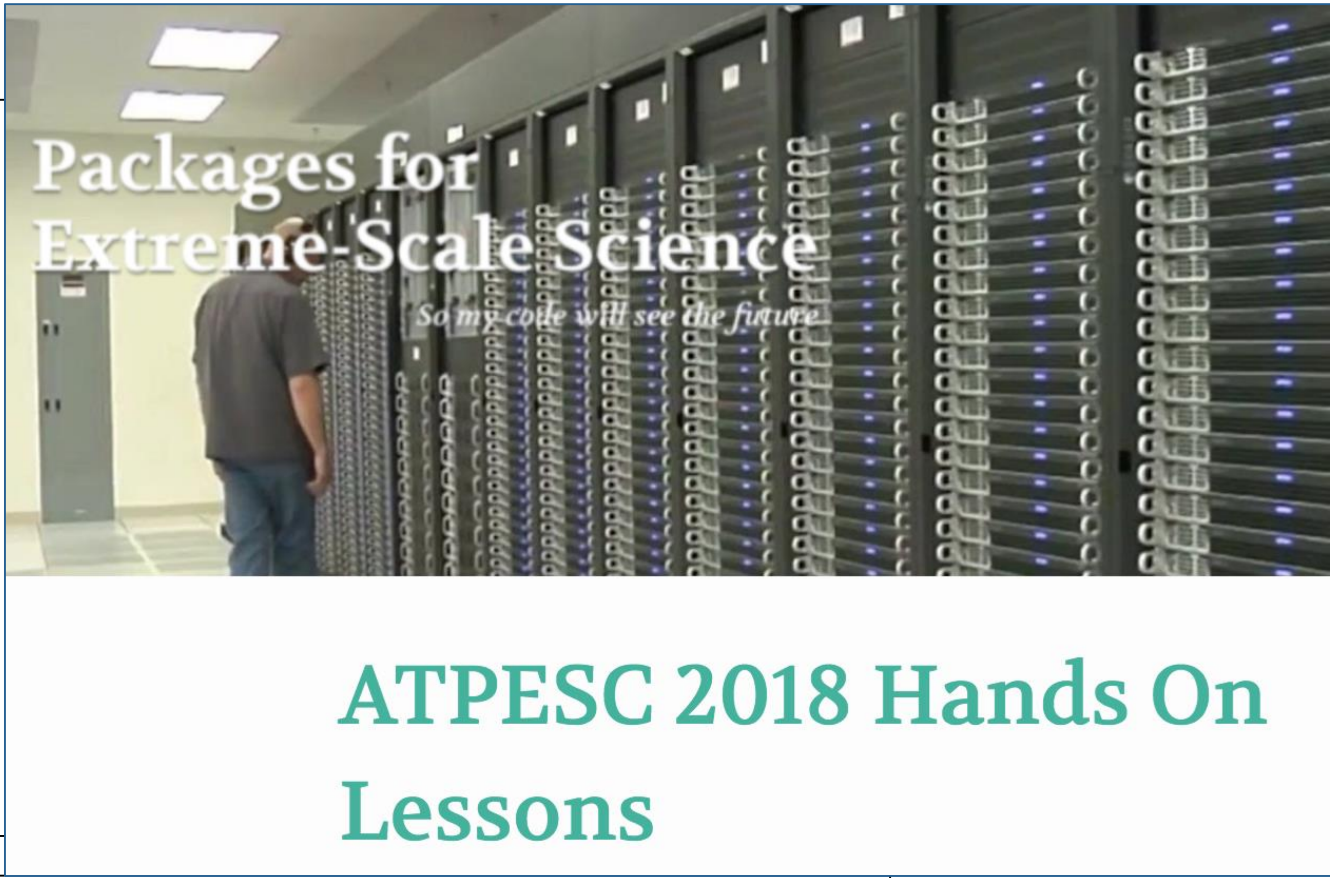
- Complete by 4 pm CDT

Meeting opportunities include:

- Today, 6:30-9:30 pm
- Other days/times, opportunities for communication with developers who are not attending today

HandsOnLessons

- Hand-coded heat equation intro
- Unstructured meshing & finite elements
- Time integration & nonlinear solvers
- Krylov solvers & algebraic multigrid
- Sparse direct solvers
- Numerical optimization and adjoints
- Adaptive workflow



Github pages site:

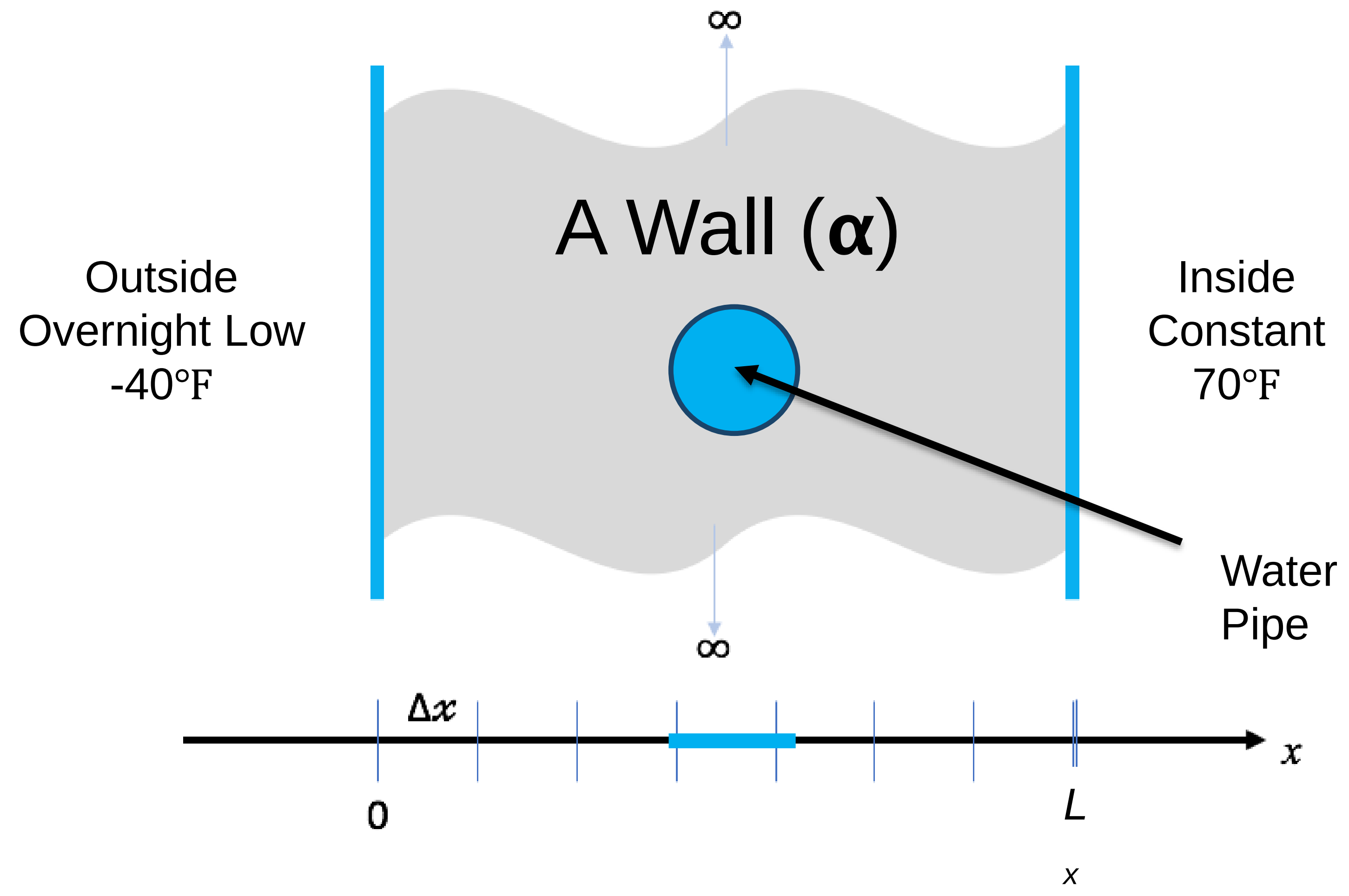
https://xsdk-project.github.io/ATPESC2018HandsOnLessons

Custom-coded heat equation

Why you don’t wanna write custom code
and should instead use numerical packages

Mark C Miller, LLNL

A science problem of interest:
Will my water pipes freeze?



We have a PDE:
The one-dimensional heat equation

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

- $u(x,t)$ is temperature in Kelvin
- x is distance in meters
- t is time in seconds
- **α** is thermal diffusivity of the material (m^2/s)

Given boundary and initial conditions

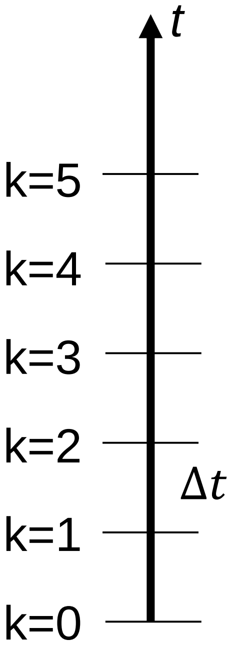
- Left end-point: $u(0,t) = U_0$
- Right end-point: $u(L_x,t)=U_L$
- Initial temperature profile: $u(x,0)=U(x)$

- We seek a numerical software solution for $u(x,t)$

Discretize:
We need a mesh

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

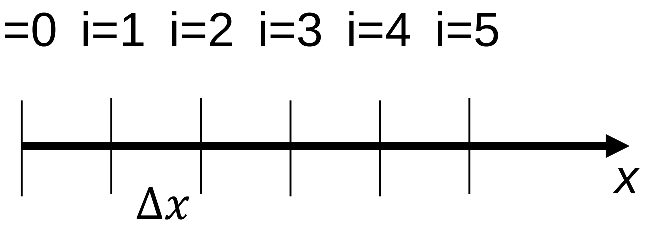
Time



$$\frac{u_i^{k+1} - u_i^k}{\Delta t}$$

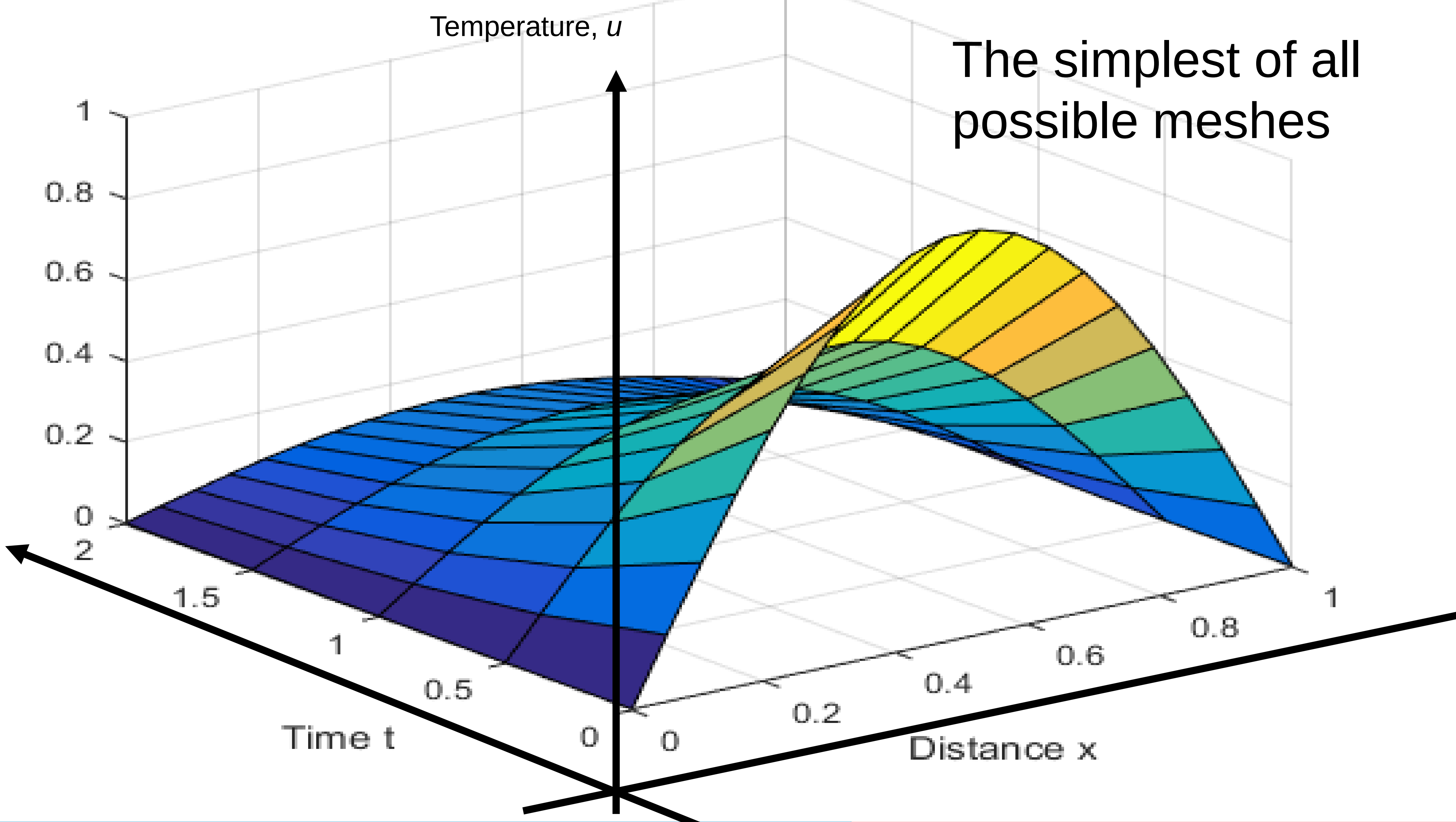
$$\alpha \frac{u_{i-1}^k - 2u_i^k + u_{i+1}^k}{\Delta x}$$

Space



$$\frac{u_i^{k+1} - u_i^k}{\Delta t} = \alpha \frac{u_{i-1}^k - 2u_i^k + u_{i+1}^k}{\Delta x}$$

The simplest of all possible meshes



We need a solver

The *explicit* FTCS algorithm

$$\frac{u_i^{k+1} - u_i^k}{\Delta t} = \alpha \frac{u_{i-1}^k - 2u_i^k + u_{i+1}^k}{\Delta x}$$

$$u_i^{k+1} = r u_{i+1}^k + (1 - 2r) u_i^k + r u_{i-1}^k \qquad r = \alpha \frac{\Delta t}{(\Delta x)^2}$$

- Known to be **unstable** for $r > \frac{1}{2}$
- An explicit solver would involve a matrix, $\overrightarrow{Au}^{k+1} = R\vec{u}^k$

Exercise #1 (3 mins)

Open ftcs.C w/editor and write the body of this function

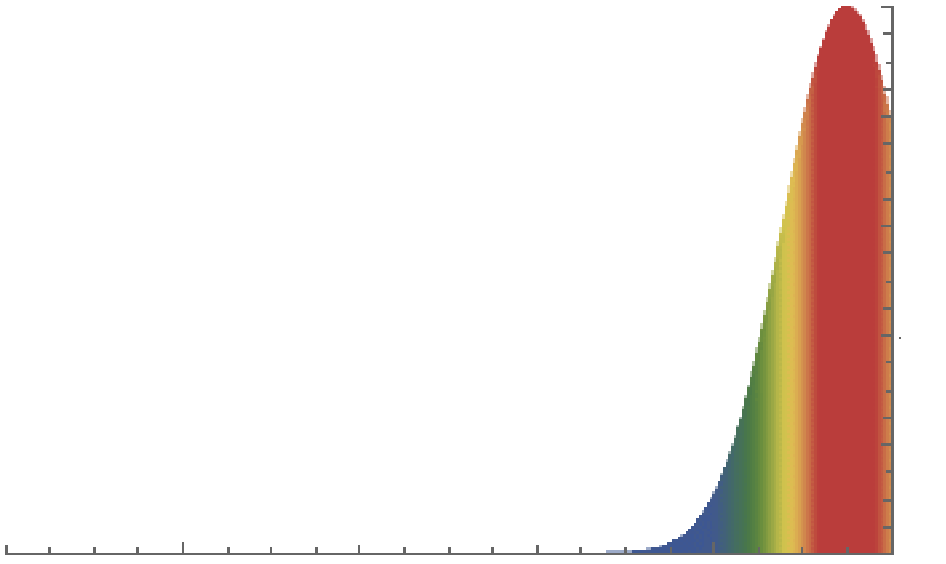
$$u_i^{k+1} = r u_{i+1}^k + (1-2r) u_i^k + r u_{i-1}^k$$

$$r = \alpha \frac{\Delta t}{(\Delta x)^2}$$

```
bool
update_solution_ftcs(
    int n,                    // number of values
    Double *uk1,              // new values: u(i) i=0…n-1 @ t=k+1
    Double const *uk0,        // last values: u(i) i=0…n-1 @ t=k
    Double alpha,             // thermal diffusivity
    Double dx, Double dt,     // spacing in space, x, and time, t.
    Double bc0, Double bc1)   // boundary conditions @ x=0 & x=Lx
{
.
.
.
}
```

Exercise #2 (1 min)
Build and test the application

```
% make
c++ -c   heat.C -o heat.o
c++ -c   utils.C -o utils.o
c++ -c   args.C -o args.o
c++ -c   exact.C -o exact.o
c++ -c   ftcs.C -o ftcs.o
c++ -c   upwind15.C -o upwind15.o
c++ -c   crankn.C -o crankn.o
c++ -o heat heat.o utils.o args.o exact.o ftcs.o upwind15.o crankn.o -lm
```



- How might we test it?
  - We know steady state solution for bc0=A and bc1=B is line from A to B

EXASCALE COMPUTING PROJECT

Exercise #3 (2 mins):
Run application to model problem of interest

- Outside temp has been same as inside temp @ 70 °F for a long time
- Night/storm will last 15.5 hours @ -40 °F
- Walls are 0.25 meters thick wood, pipe is 0.1 meters diameter

Material	Thermal Diffusivity, α, (m^2/s)
Wood	8.2×10^{-8}
Adobe Brick	2.7×10^{-7}
Common (“red”) brick	5.2×10^{-7}

Exercise #4 (1 mins)

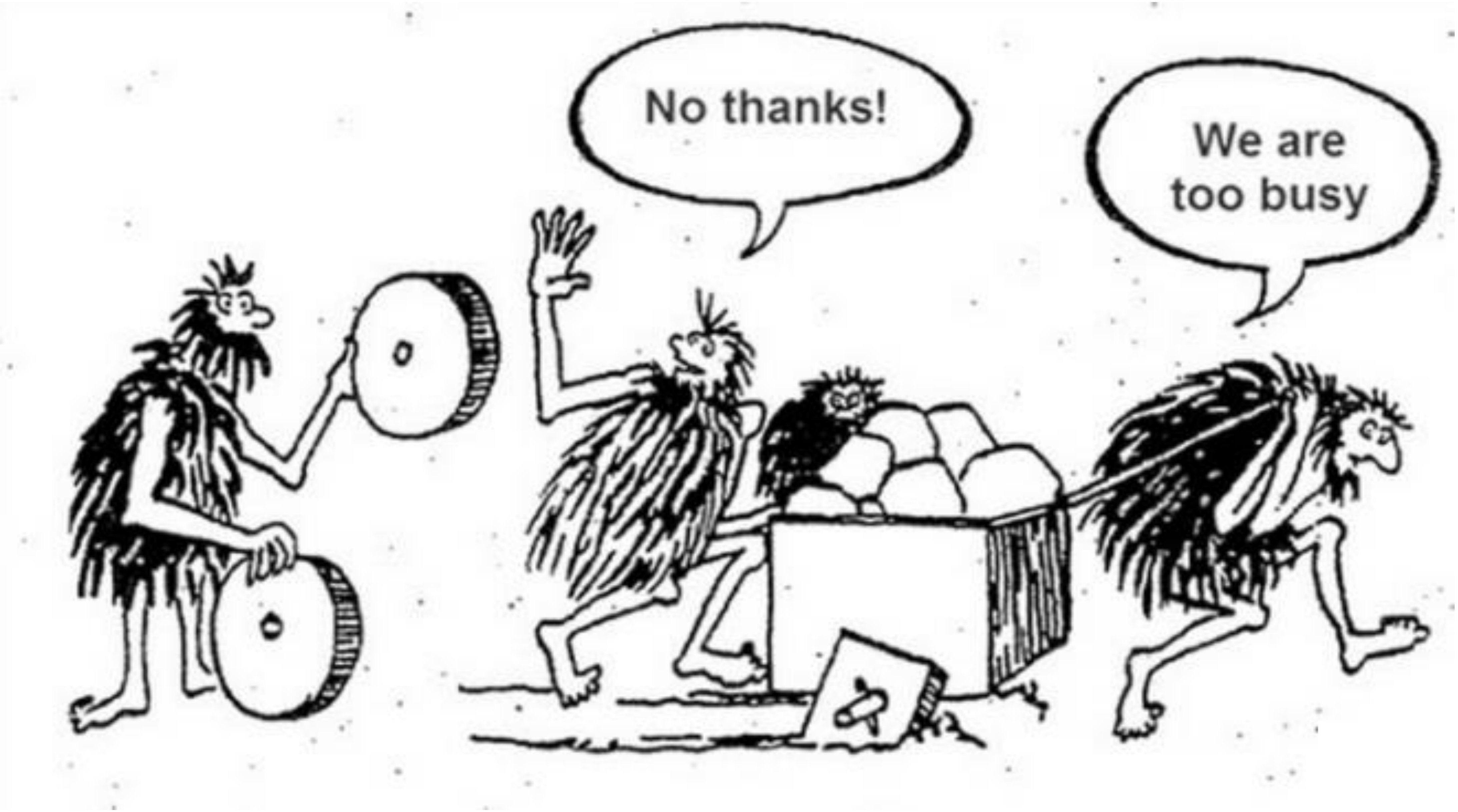
Do some science / analyze the results

Criterion: Will conclude pipe freezes if…
…center point drops below freezing before storm passes

```
make plot PTOOL=[visit|gnuplot|pyplot] RUNAME=<run-name>
```

What if our problem were to find the thinnest wall width?

Custom coded solutions are a slippery slope



HPC software libraries provide powerful algorithms … and enable problem-specific customization

- **Numerical algorithm challenges**
  - Discretizations: Dimensions, orders, geometries, material interfaces, etc…
  - Time integrators: Adaptive, faster convergence, robustness, efficiencies, etc…
  - Solvers: implicit, explicit, iterative, direct, preconditioners, etc..
  - Optimization: Outer loops, nonintrusive, reduced order models, etc…
  - Validation & Verification: Vetted, trusted results, community accepted, etc…
- **Software development challenges**
  - Time and space performance, robustness
  - Scalability, load balance, performance portability
  - Encapsulation, interfaces, interoperability
  - Documentation, ease of installation, ease of use
  - Sustainable open source, supported with regular updates, bug tracking/fixing

Well-designed software libraries enable user-specific customization to exploit problem structure and understanding

Auspices and Disclaimer

Support for this work was provided through Scientific Discovery through Advanced Computing (SciDAC) program and the Exascale Computing Project funded by U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.