Introduction to Numerical Software

Presented to

ATPESC 2023 Participants

Ulrike Meier Yang

Lawrence Livermore National Laboratory

Date 08/08/2023

U.S. DEPARTMENT OF ENERGY | Office of Science

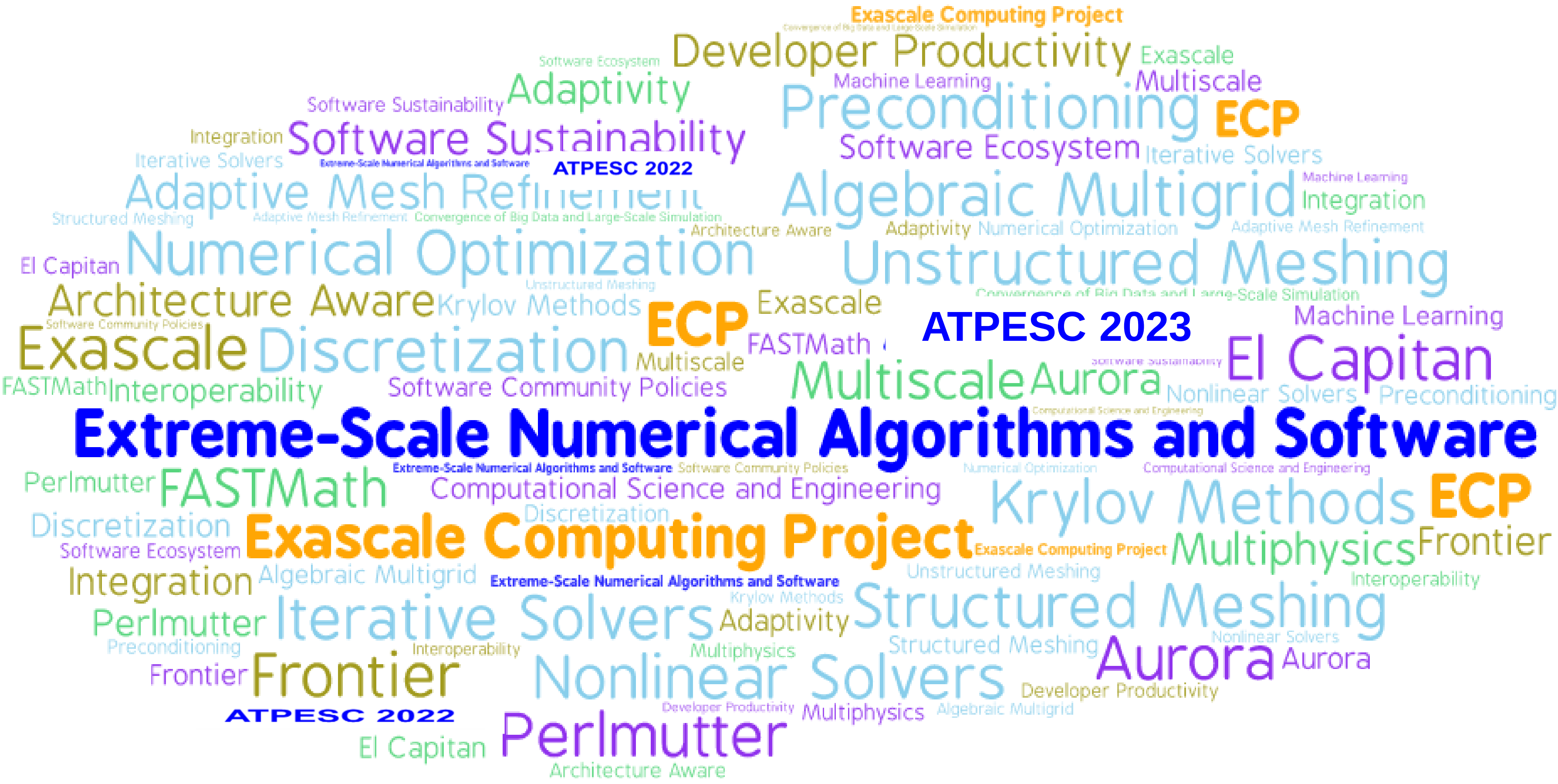
ATPESC Numerical Software Track

EXASCALE COMPUTING PROJECT

Outline

- Logistics for the day

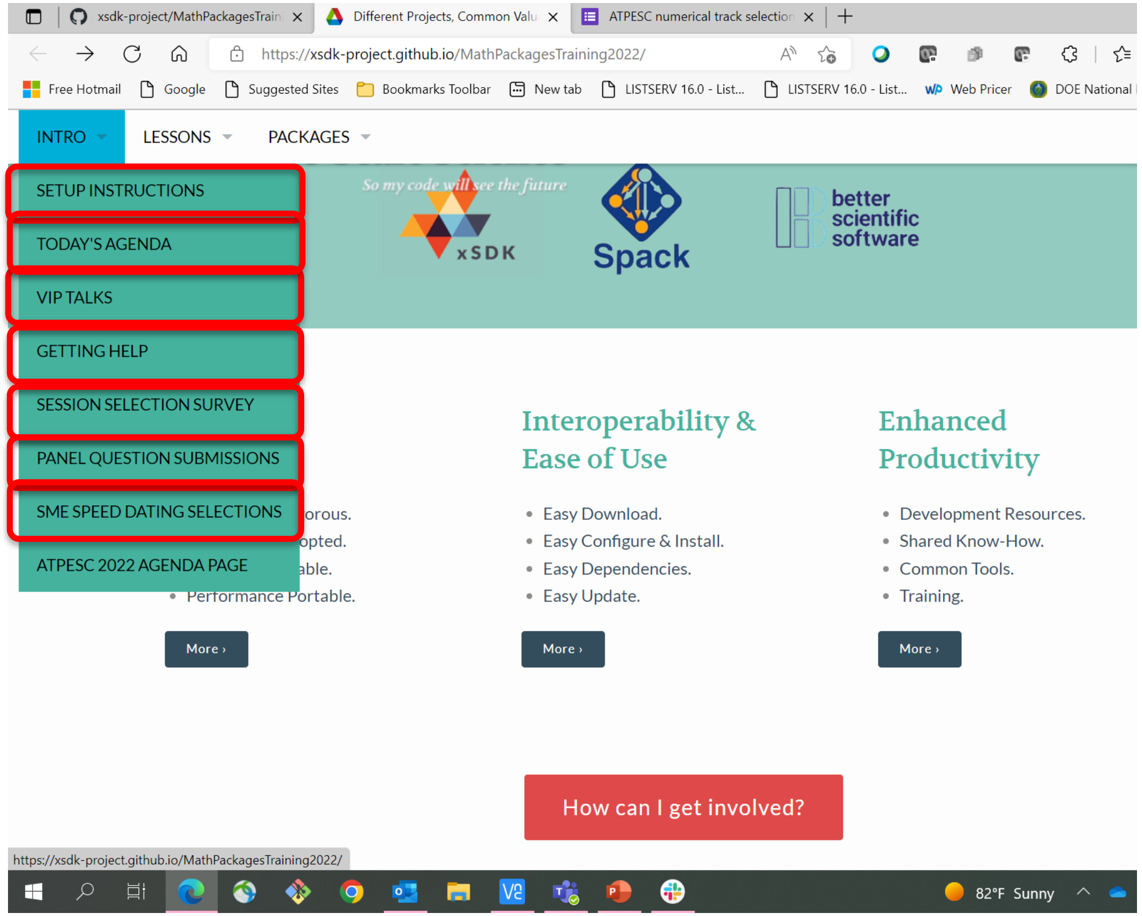
- Intro to numerical algorithms and software for extreme-scale science



Your home bases for the day: ATPESC Track 5
Numerical Algorithms and Software for Extreme-Scale Science

- Main ATPESC Agenda
  - **https://extremecomputingtraining.anl.gov/agenda-2023/#Track-5**
  - slides (pdf) and presenter bios
- Math Packages Training Site
  - session abstracts, links to parallel breakout rooms, hands-on lessons, more
  - **https://xsdk-project.github.io/MathPackagesTraining2023/agenda/**

https://xsdk-project.github.io/MathPackagesTraining2023/



- Setup instructions
- Today’s agenda
- VIP talks
- Getting help
- Session Selection Survey
- Panel question submission
- SME speed dating selections

Agenda

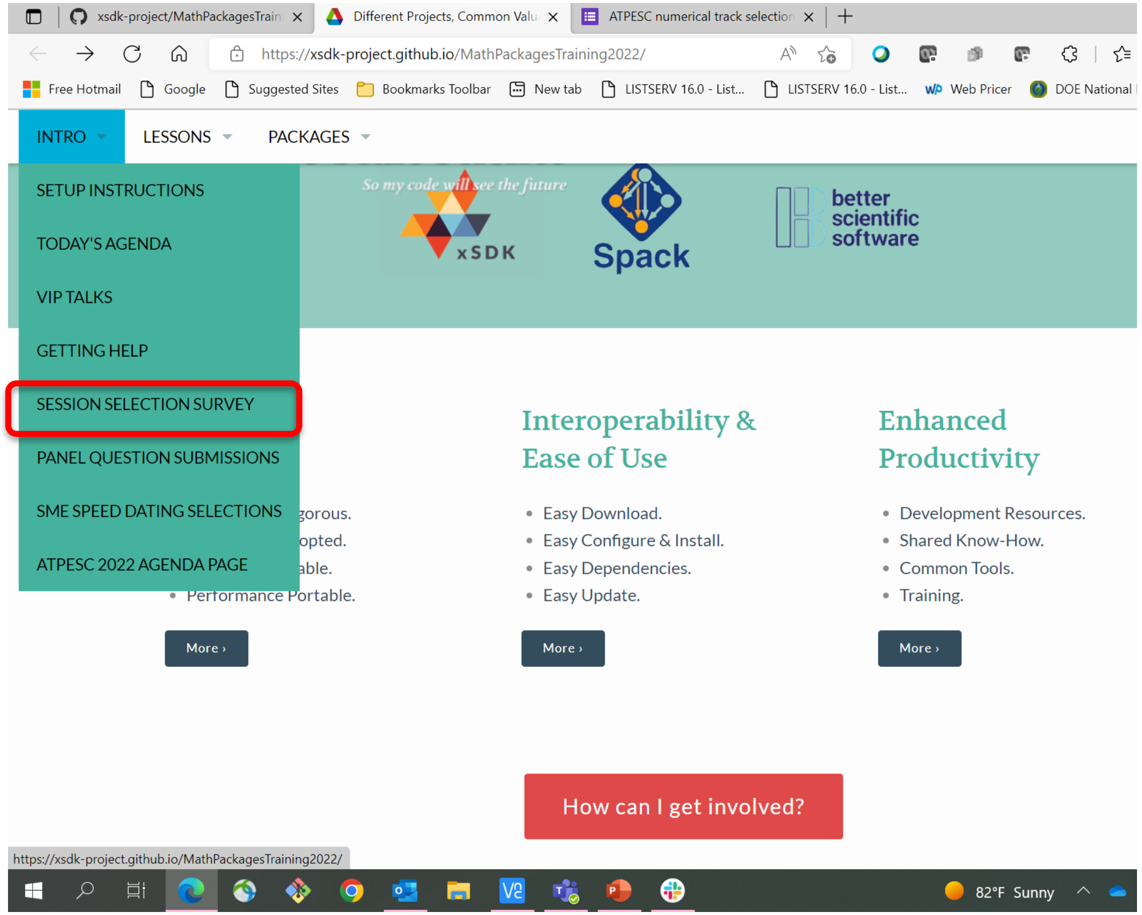
https://extremecomputingtraining.anl.gov/agenda-2023/#Track-5

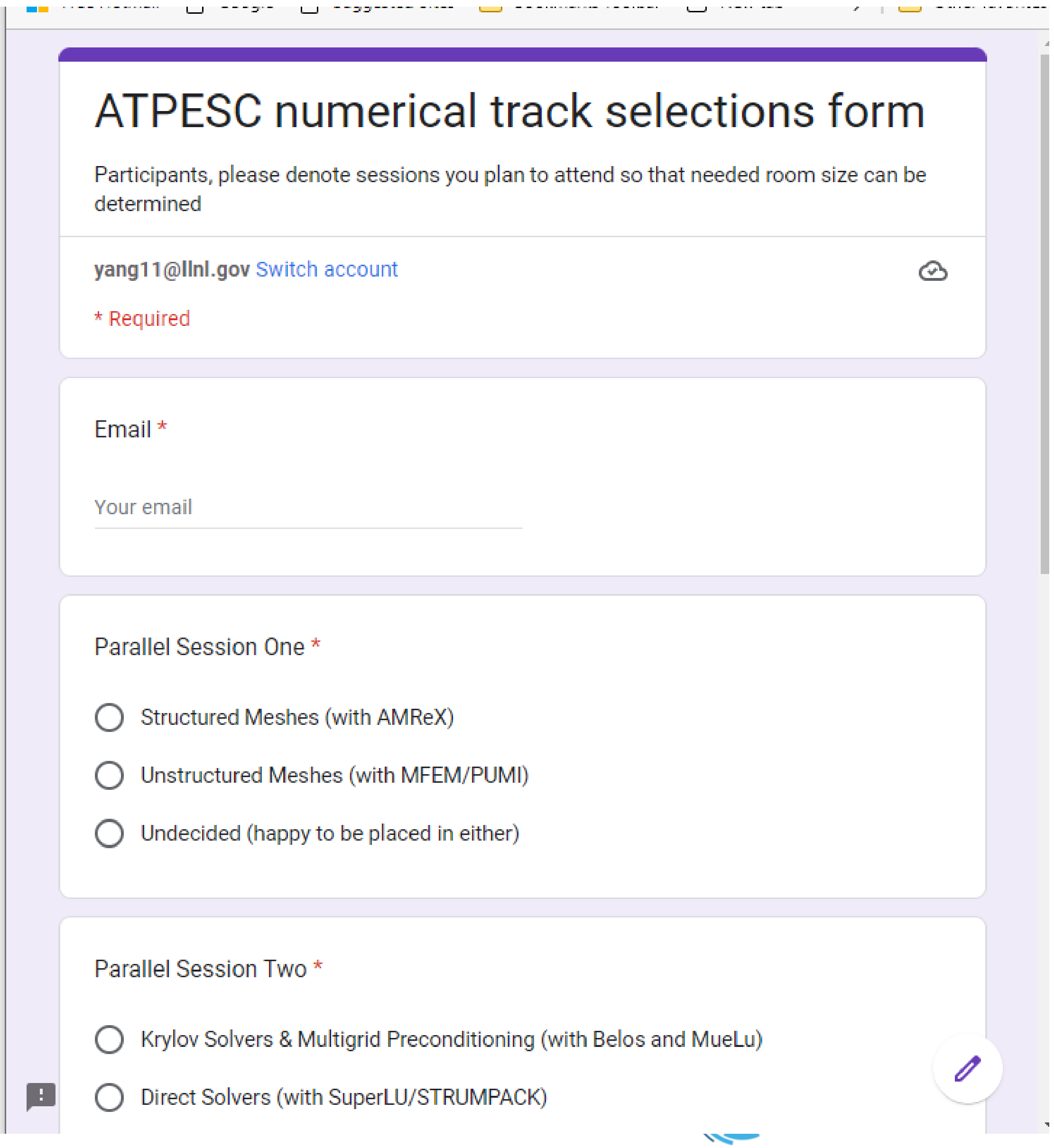
Time	Room?	Room?
8:30 – 9:30	Introduction to Numerical Software – Ulrike Yang	
9:30 – 10:45	Structured Discretization (AMReX) – Ann Almgren, Andrew Myers	Unstructured Discretization (MFEM/PUMI) – Tzanio Kolev, Mark Shephard, Cameron Smith
10:45 – 11:15	Break, Subject Matter Expert (SME) Selections, Panel Questions	
11:15 – 12:30	Iterative Solvers & Algebraic Multigrid (hypre) – Sarah Osborn, Ulrike Yang	Direct Solvers (SuperLU, STRUMPACK) – Sherry Li, Pieter Ghysels
12:30 – 1:30	Lunch, SME Selections, Panel Questions	
1:30 – 2:45	Nonlinear Solvers (PETSc) – Richard Mills	Time Integration (SUNDIALS) – David Gardner
2:45 – 3:15	Break, SME Selections, **Panel Questions Due**	
3:15 – 4:30	Optimization (TAO) – Toby Isaac	Iterative Solvers & Algebraic Multigrid (Trilinos/ Belos/MueLU) – Christian Glusa, Graham Harper
4:30 – 5:30	Wrap-up (Ann Almgren) / Panel: Extreme-Scale Numerical Algorithms and Software	
5:30 – 6:30	Unstructured Time: **SME Selections Due**, Informal Discussion, Continue Hands-on	
6:30 – 7:30	Dinner	
7:30 – 9:30	Optional Activity: SME Speed-dating	

Choose which lecture you want to attend!

- Access: https://forms.gle/axrawtNsTgbjDTJP8

https://xsdk-project.github.io/MathPackagesTraining2022/





Agenda

https://extremecomputingtraining.anl.gov/agenda-2023/#Track-5

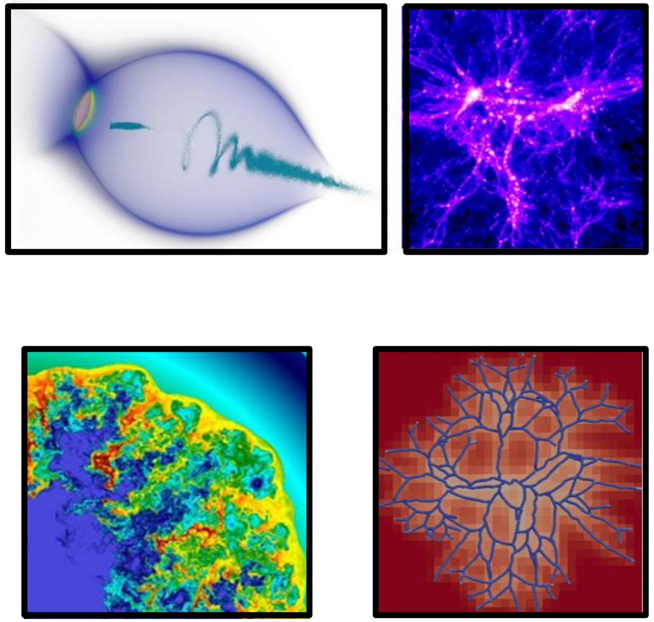
Time	Room?	Room?
8:30 – 9:30	Introduction to Numerical Software – Ulrike Yang	
9:30 – 10:45	**Structured Discretization (AMReX) – Ann Almgren, Andrew Myers**	**Unstructured Discretization (MFEM/PUMI) – Tzanio Kolev, Mark Shephard, Cameron Smith**
10:45 – 11:15	Break	
11:15 – 12:30	Iterative Solvers & Algebraic Multigrid (hypre) – Sarah Osborn, Ulrike Yang	Direct Solvers (SuperLU, STRUMPACK) – Sherry Li, Pieter Ghysels
12:30 – 1:30	Lunch	
1:30 – 2:45	Nonlinear Solvers (PETSc) – Richard Mills	Time Integration (SUNDIALS) – David Gardner
2:45 – 3:15	Break	
3:15 – 4:30	Optimization (TAO) – Toby Isaac	Iterative Solvers & Algebraic Multigrid (Trilinos/ Belos/MueLU) – Christian Glusa, Graham Harper
4:30 – 5:30	Wrap-up (Ann Almgren)/ Panel: Extreme-Scale Numerical Algorithms and Software	
6:30 – 7:30	Dinner	
7:30 – 8:30	SME Speed Dating	

EXASCALE COMPUTING PROJECT

Block-structured adaptive mesh refinement framework. Scalable support for hierarchical mesh and particle data, with embedded boundaries.

- **Capabilities**
  - Support for PDEs on a hierarchical adaptive mesh with particles and embedded boundary representations of complex geometry
  - 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 on multicore architectures
    - hybrid MPI + GPU support for hybrid CPU/GPU systems (NVIDIA CUDA, AMD HIP, Intel SYCL)
  - 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
- **Open source software**
  - Used for diverse apps, including accelerator modeling, astrophysics, combustion, cosmology, multiphase flow, phase field modeling, atmospheric modeling and more
  - Source code and development hosted on github with rigorous testing framework
  - Extensive documentation, examples and tutorials

Examples of AMReX applications



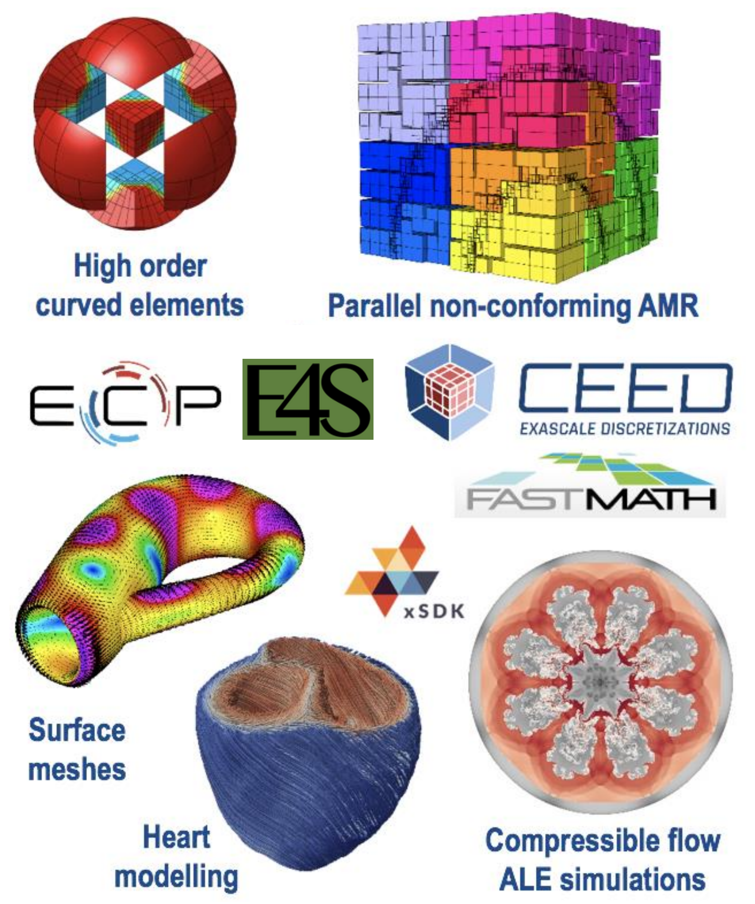
https://www.github.com/AMReX-Codes/amrex

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 and includes initial GPU implementation. 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**
  - BSD with thousands of downloads/year worldwide.
  - Available on GitHub, also via OpenHPC, Spack. Part of ECP's CEED co-design center.



https://mfem.org

ECP EXASCALE COMPUTING PROJECT

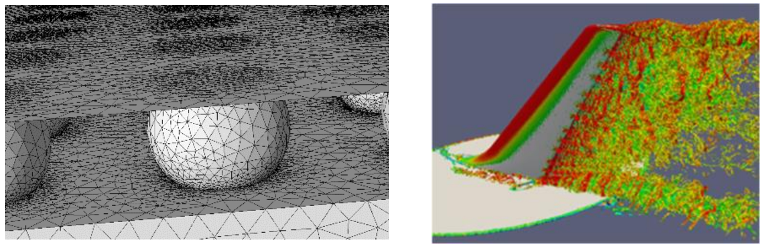
Parallel Unstructured Mesh Infrastructure

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

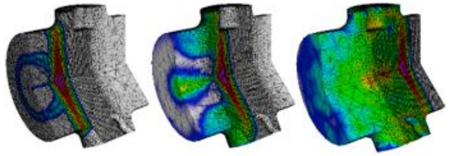
- **Core functionality**
  - Distributed, conformant mesh with entity migration, remote read only copies, fields and their operations
  - Link to the geometry and attributes
  - Mesh adaptation (straight and curved), mesh motion
  - Multi-criteria partition improvement
  - Distributed mesh support for Particle In Cell methods
- **Designed for integration into existing codes**
  - xSDK package; installs with Slack
  - Permissive license enables integration with open and closed-source codes
- **In-memory integrations developed**
  - MFEM: High order FE framework
  - PetraM: Adaptive RF fusion
  - 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

PUMI

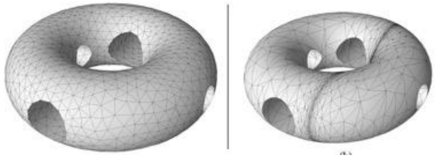
SCOREC Rensselaer



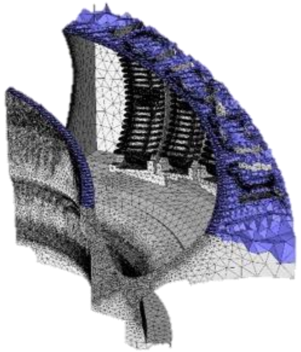
Applications with billions of elements: flip-chip (L), flow control (R)



Mesh adaptation for evolving features



Anisotropic adaptation for curved meshes



RF antenna and plasma surface in vessel.

Source Code: github.com/SCOREC/core
User Guide: scorec.rpi.edu/pumi/PUMI.pdf
Paper: scorec.rpi.edu/REPORTS/2014-9.pdf

Room Choice

- Please raise your hand if you want to attend
  - Structured meshing
  - Unstructured meshing

Agenda

https://extremecomputingtraining.anl.gov/agenda-2023/#Track-5

Time	Room?	Room?
8:30 – 9:30	Introduction to Numerical Software – Ulrike Yang	
9:30 – 10:45	Structured Discretization (AMReX) – Ann Almgren, Andrew Myers	Unstructured Discretization (MFEM/PUMI) – Tzanio Kolev, Mark Shephard, Cameron Smith
10:45 – 11:15	Break	
11:15 – 12:30	**Iterative Solvers & Algebraic Multigrid (hypre) – Sarah Osborn, Ulrike Yang**	**Direct Solvers (SuperLU, STRUMPACK) – Sherry Li, Pieter Ghysels**
12:30 – 1:30	Lunch	
1:30 – 2:45	Nonlinear Solvers (PETSc) – Richard Mills	Time Integration (SUNDIALS) – David Gardner
2:45 – 3:15	Break	
3:15 – 4:30	Optimization (TAO) – Toby Isaac	Iterative Solvers & Algebraic Multigrid (Trilinos/ Belos/MueLU) – Christian Glusa, Graham Harper
4:30 – 5:30	Wrap-up (Ann Almgren) / Panel: Extreme-Scale Numerical Algorithms and Software	
6:30 – 7:30	Dinner	
7:30 – 8:30	SME Speed Dating	

EXASCALE COMPUTING PROJECT

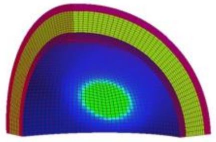
hypre

Lawrence Livermore National Laboratory

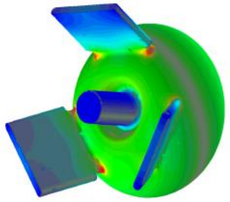
Highly scalable multilevel solvers and preconditioners. Unique user-friendly interfaces. Flexible software design. Used in a variety of applications. Freely available.

- **Conceptual interfaces**
  - Structured, semi-structured, finite elements, linear algebraic interfaces
  - Provide natural “views” of the linear system
  - Provide for efficient (scalable) linear solvers through effective data storage schemes
- **Scalable preconditioners and solvers**
  - Structured and unstructured algebraic multigrid solvers
  - Maxwell solvers, H-div solvers
  - Multigrid solvers for nonsymmetric systems: pAIR
  - Multigrid reduction (MGR) for systems of PDEs
  - Matrix-free Krylov solvers
  - ILU and FSAI preconditioners
- **Exascale early systems GPU-readiness**
  - Nvidia GPU (CUDA), AMD GPU (HIP), Intel GPU (SYCL)
- **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/hypre-space/hypre

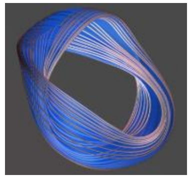
hypre high performance preconditioners



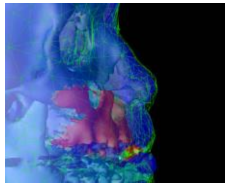
Elasticity / Plasticity



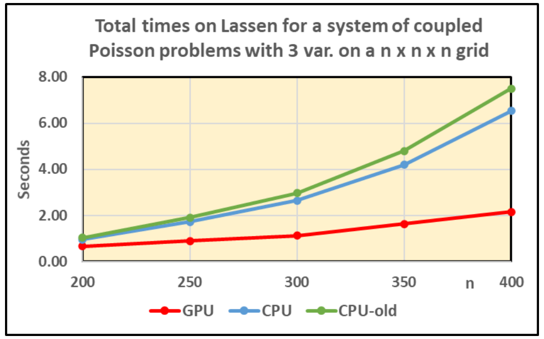
Electro-magnetics



Magneto-hydrodynamics



Facial surgery



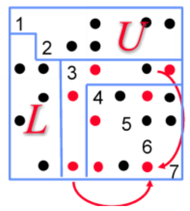
http://www.llnl.gov/CASC/hypre

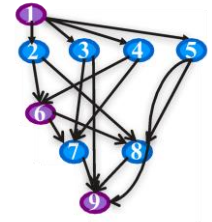
EXASCALE COMPUTING PROJECT

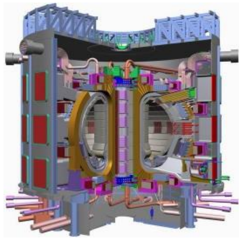
SuperLU

Supernodal Sparse LU Direct Solver. Flexible, user-friendly interfaces. Examples show various use scenarios. Testing code for unit-test. BSD license.

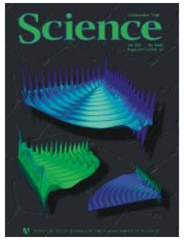
- **Capabilities**
  - Serial (thread-safe), shared-memory (SuperLU_MT, OpenMP or Pthreads), distributed-memory (SuperLU_DIST, hybrid MPI+ OpenM + CUDA/HIP).
    - Written in C, with Fortran interface
  - Sparse LU decomposition (can be nonsymmetric sparsity pattern), triangular solution with multiple right-hand sides
  - Incomplete LU (ILUTP) preconditioner in serial SuperLU
  - Sparsity-preserving ordering: minimum degree or graph partitioning applied to A^TA or A^T+A
  - User-controllable pivoting: partial pivoting, threshold pivoting, static pivoting
  - Condition number estimation, iterative refinement, componentwise error bounds
- **Exascale early systems GPU-readiness**
  - Available: Nvidia GPU (CUDA), AMD GPU (HIP)
  - In progress: Intel GPU (SYCL)
- **Parallel Scalability**
  - Factorization strong scales to 32,000 cores (IPDPS'18, JPDC'19)
  - Triangular solve strong scales to 4000 cores (SIAM CSC'18, SIAM PP'20, SC'23)
- **Open-source software**
  - Used in a vast range of applications, can be used through PETSc and Trilinos, …
  - available on github







ITER tokamak



quantum mechanics

Widely used 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.

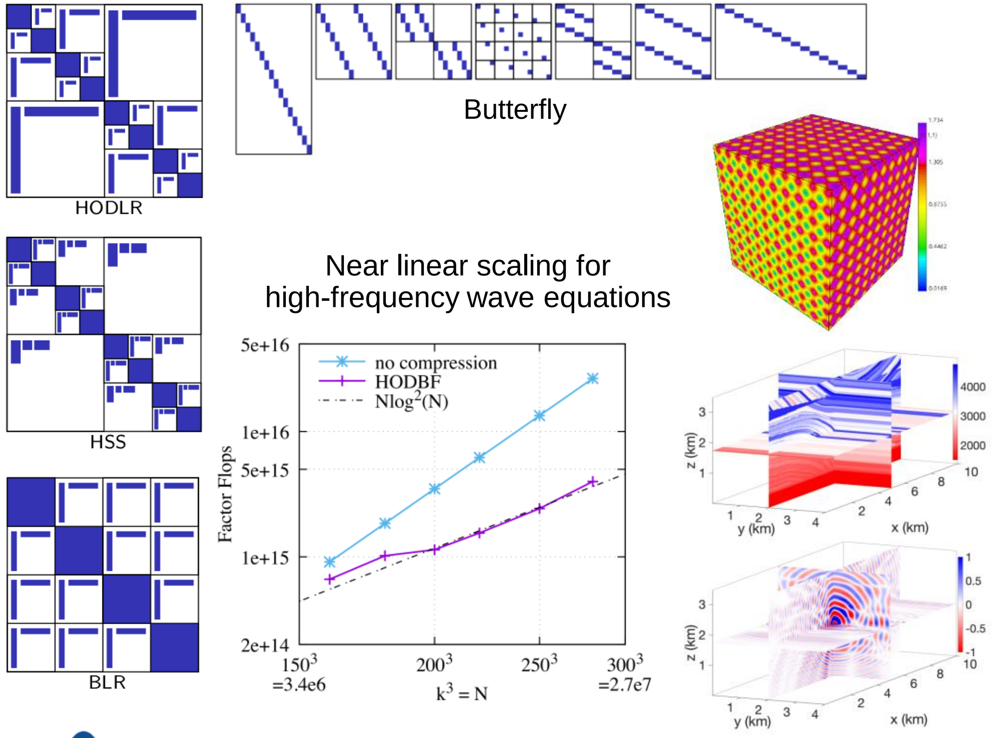
https://portal.nersc.gov/project/sparse/superlu/

STRUMPACK

Structured Matrix Package

Hierarchical solvers for dense rank-structured matrices and fast algebraic sparse solver and robust and scalable preconditioners.

- **Dense Matrix Solvers using Hierarchical Approximations**
  - Hierarchical partitioning, low-rank approximations
  - Hierarchically Semi-Separable (HSS), Hierarchically Off-Diagonal Low-Rank (HODLR), Hierarchically Off-Diagonal Butterfly (HODBF), Block Low-Rank (BLR), Butterfly
  - C++ Interface to ButterflyPACK (Fortran)
  - Applications: BEM, Cauchy, Toeplitz, kernel & covariance matrices, ...
  - Asymptotic complexity much lower than LAPACK/ScaLAPACK routines
- **Sparse Direct Solver**
  - Algebraic sparse direct solver
  - GPU: CUDA, HIP/ROCm, DPC++ (in progress)
  - Orderings: (Par)METIS, (PT)Scotch, RCM
- **Preconditioners**
  - Approximate sparse factorization, using hierarchical matrix approximations
  - Scalable and robust, aimed at PDE discretizations, indefinite systems, …
  - Iterative solvers: GMRES, BiCGStab, iterative refinement
- **Software**
  - BSD license
  - Interfaces from PETSc, MFEM, Trilinos, available in Spack



github.com/pghysels/STRUMPACK

Agenda

https://extremecomputingtraining.anl.gov/agenda-2023/#Track-5

Time	Room?	Room?
8:30 – 9:30	Introduction to Numerical Software – Ulrike Yang	
9:30 – 10:45	Structured Discretization (AMReX) – Ann Almgren, Andrew Myers	Unstructured Discretization (MFEM/PUMI) – Tzanio Kolev, Mark Shephard, Cameron Smith
10:45 – 11:15	Break	
11:15 – 12:30	Iterative Solvers & Algebraic Multigrid (hypre) – Sarah Osborn, Ulrike Yang	Direct Solvers (SuperLU, STRUMPACK) – Sherry Li, Pieter Ghysels
12:30 – 1:30	Lunch	
1:30 – 2:45	**Nonlinear Solvers (PETSc) – Richard Mills**	**Time Integration (SUNDIALS) – David Gardner**
2:45 – 3:15	Break	
3:15 – 4:30	Optimization (TAO) – Toby Isaac	Iterative Solvers & Algebraic Multigrid (Trilinos/ Belos/MueLU) – Christian Glusa, Graham Harper
4:30 – 5:30	Wrap-up (Ann Almgren) / Panel: Extreme-Scale Numerical Algorithms and Software	
6:30 – 7:30	Dinner	
7:30 – 8:30	SME Speed Dating	

EXASCALE COMPUTING PROJECT

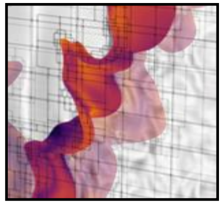
SUNDIALS

Suite of Nonlinear and Differential /Algebraic Equation Solvers

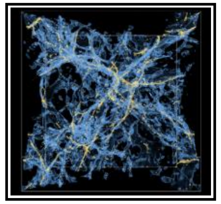
Adaptive time integrators for ODEs and DAEs and efficient nonlinear solvers
Used in a variety of applications. Freely available. Encapsulated solvers & parallelism.

- **ODE and DAE time integrators:**
  — *CVODE:* adaptive order and step BDF (stiff) & Adams (non-stiff) methods for ODEs
  — *ARKODE:* adaptive step implicit, explicit, IMEX, and multirate Runge-Kutta methods for ODEs
  — *IDA*: adaptive order and step BDF methods for DAEs
  — *CVODES* and *IDAS:* provide forward and adjoint sensitivity analysis capabilities
- **Nonlinear Solvers:** *KINSOL* – Newton-Krylov; accelerated Picard and fixed point
- **Modular Design:** Easily incorporated into existing codes; Users can supply their own data structures and solvers or use SUNDIALS provided modules
- **Support on NVIDIA, AMD, and Intel GPUs:**
  — Vectors: CUDA, HIP, OpenMP Offload, RAJA, SYCL (DPC++)
  — Linear solvers: cuSOLVER, MAGMA, matrix-free Krylov methods
- **Open Source:** BSD License; Download from LLNL site, GitHub, or Spack
  — Supported by extensive documentation; user email list with an active community
  — Available through MFEM, AMReX, deal.II, and PETSc

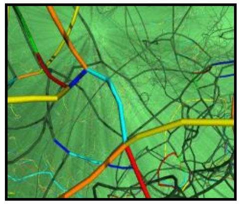
SUNDIALS is used worldwide in applications throughout research and industry



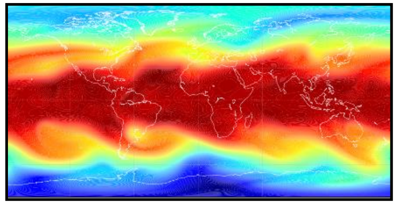
Combustion (Pele)



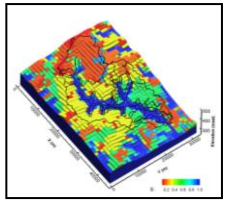
Cosmology (Nyx)



Dislocation dynamics (ParaDiS)



Atmospheric Dynamics (Tempest)



Subsurface flow (ParFlow)

Lawrence Livermore National Laboratory

ECP

FASTMATH

xSDK

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

EXASCALE COMPUTING PROJECT

PETSc TAO

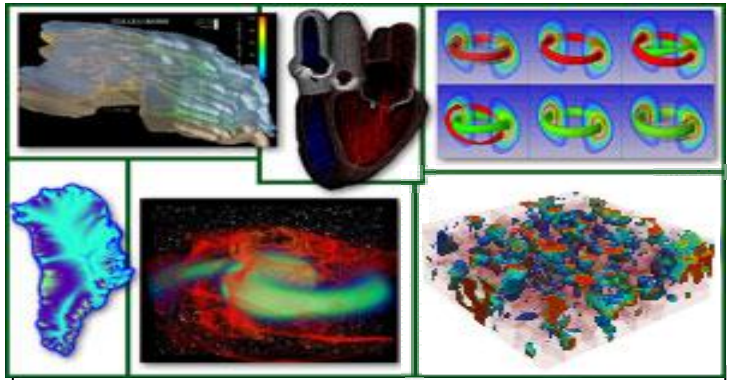
Portable, Extensible Toolkit for Scientific Computation / Toolkit for Advanced

Scalable algebraic solvers for PDEs. Encapsulate parallelism in high-level objects. 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; NVIDIA, AMD, and Intel GPUs
  - 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

ECP xSDK FASTMATH R&D 100

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

Agenda

https://extremecomputingtraining.anl.gov/agenda-2023/#Track-5

Time	Room?	Room?
8:30 – 9:30	Introduction to Numerical Software – Ulrike Yang	
9:30 – 10:45	Structured Discretization (AMReX) – Ann Almgren, Andrew Myers	Unstructured Discretization (MFEM/PUMI) – Tzanio Kolev, Mark Shephard, Cameron Smith
10:45 – 11:15	Break	
11:15 – 12:30	Iterative Solvers & Algebraic Multigrid (hypre) – Sarah Osborn, Ulrike Yang	Direct Solvers (SuperLU, STRUMPACK) – Sherry Li, Pieter Ghysels
12:30 – 1:30	Lunch	
1:30 – 2:45	Nonlinear Solvers (PETSc) – Richard Mills	Time Integration (SUNDIALS) – David Gardner
2:45 – 3:15	Break	
3:15 – 4:30	**Optimization (TAO) – Toby Isaac**	**Iterative Solvers & Algebraic Multigrid (Trilinos/ Belos/MueLU) – Christian Glusa, Graham Harper**
4:30 – 5:30	Wrap-up (Ann Almgren) / Panel: Extreme-Scale Numerical Algorithms and Software	
6:30 – 7:30	Dinner	
7:30 – 8:30	SME Speed Dating	

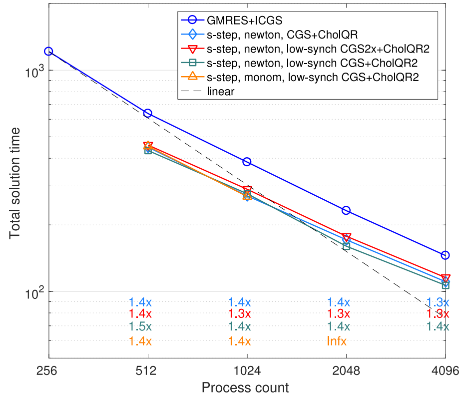
EXASCALE COMPUTING PROJECT

Trilinos/Belos

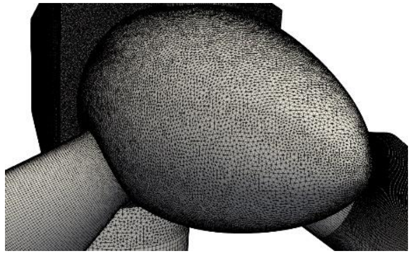
Iterative Krylov-based solvers. Templated C++ allows for generic scalar, ordinal, and compute node types.

- **Ability to solve single or sequence of linear systems**
  - Simultaneously solved systems w/ multiple-RHS: $AX = B$
  - Sequentially solved systems w/ multiple-RHS: $AX_i = B_i,\ i=1,\ldots,t$
  - Sequences of multiple-RHS systems: $A_iX_i = B_i,\ i=1,\ldots,t$
- **Standard methods**
  - Conjugate Gradients (CG), GMRES
  - TFQMR, BiCGStab, MINRES, fixed-point
- **Advanced methods**
  - Block GMRES, block CG/BICG
  - Hybrid GMRES, CGRODR (block recycling GMRES)
  - TSQR (tall skinny QR), LSQR
  - Pipelined and s-step methods
  - Stable polynomial preconditioning
- **Performance portability via Kokkos (CPUs, NVIDIA/Intel/AMD GPUs, Phi)**
- **Ongoing research**
  - Communication avoiding methods

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.



Speed-ups of various s-step Krylov methods within low Mach CFD wind-energy code Nalu-Wind.



Thomas et al., "High-fidelity simulation of wind- turbine incompressible flows", SISC, 2019.

https://trilinos.github.io/belos.html

Trilinos/MueLu

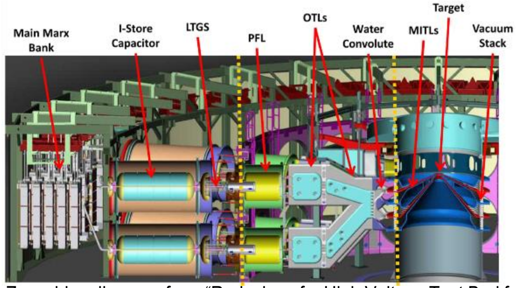
Structured and unstructured aggregation-based algebraic multigrid (AMG) preconditioners

- **Robust, scalable, portable AMG preconditioning critical for many large-scale simulations**
  - Multifluid plasma simulations
  - Shock physics
  - Magneto-hydrodynamics (MHD)
  - Low Mach computational fluid dynamics (CFD)

- **Capabilities**
  - Aggregation-based coarsening
  - Smoothers: Jacobi, GS, *l*1 GS, polynomial, ILU, sparse direct
  - Load-balancing for good parallel performance
  - Structured coarsening, geometric multigrid
  - Setup and solve phases can run on GPUs.
  - Performance portability via Kokkos (CPUs, NVIDIA/Intel/AMD GPUs, Xeon Phi)

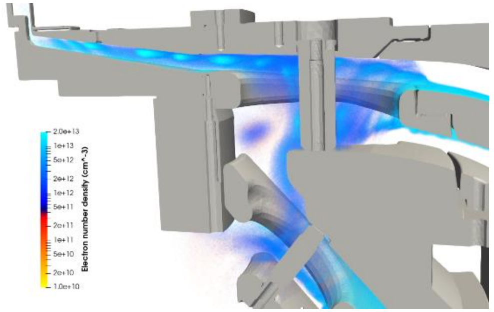
- **Research Areas**
  - AMG for multiphysics
  - Multigrid for coupled structured/unstructured meshes
  - Algorithm selection via machine learning

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.



Z machine diagram, from "Redesign of a High Voltage Test Bed for Marxes on Z", W.M. White et al., 2018.

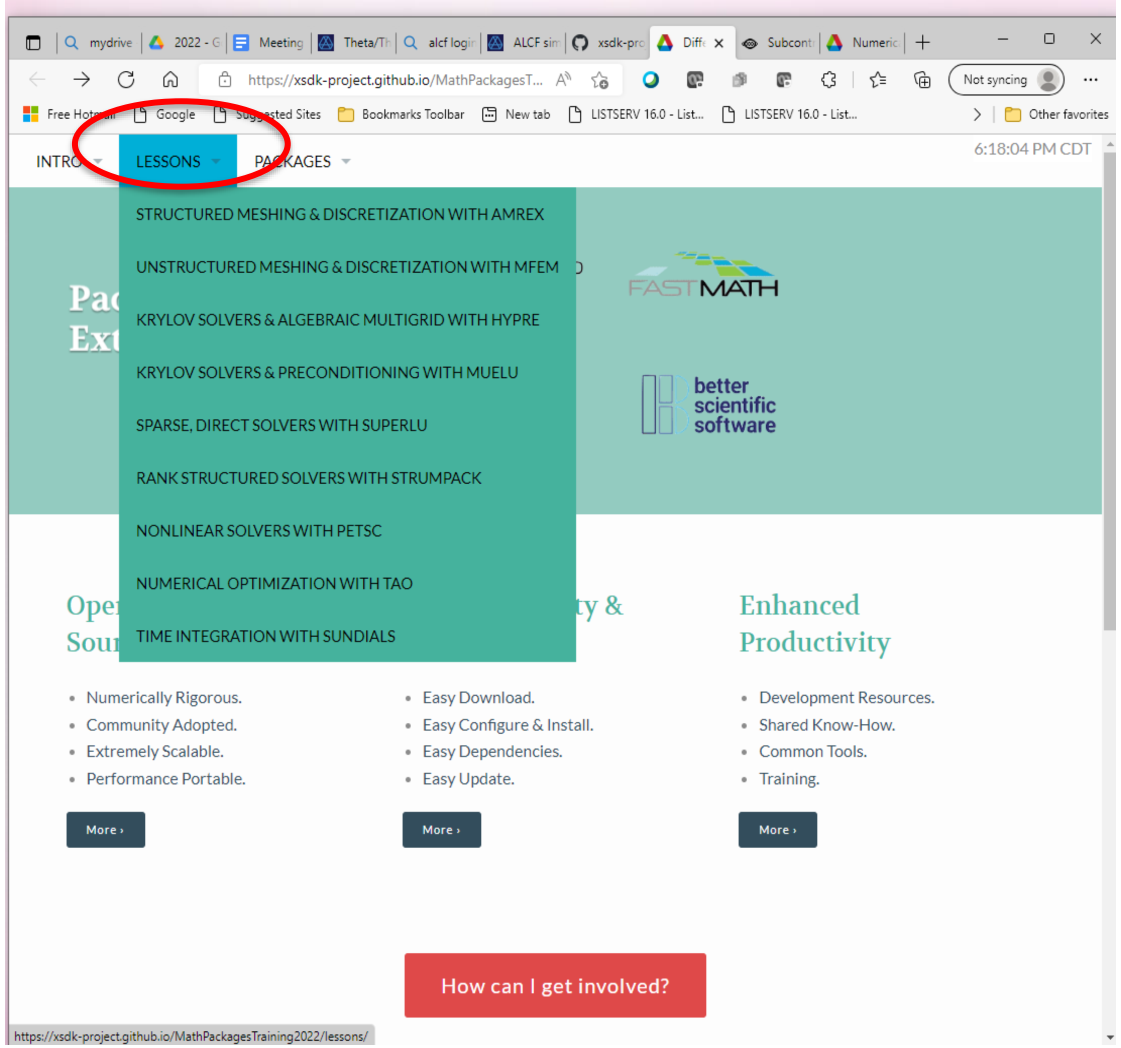
AMG preconditioning for H(curl) systems is key enabling technology in Z machine simulations for determining power from Marx banks to Target.



Plasma density in Z machine Target simulation, courtesy of D. Sirajuddin (SNL).

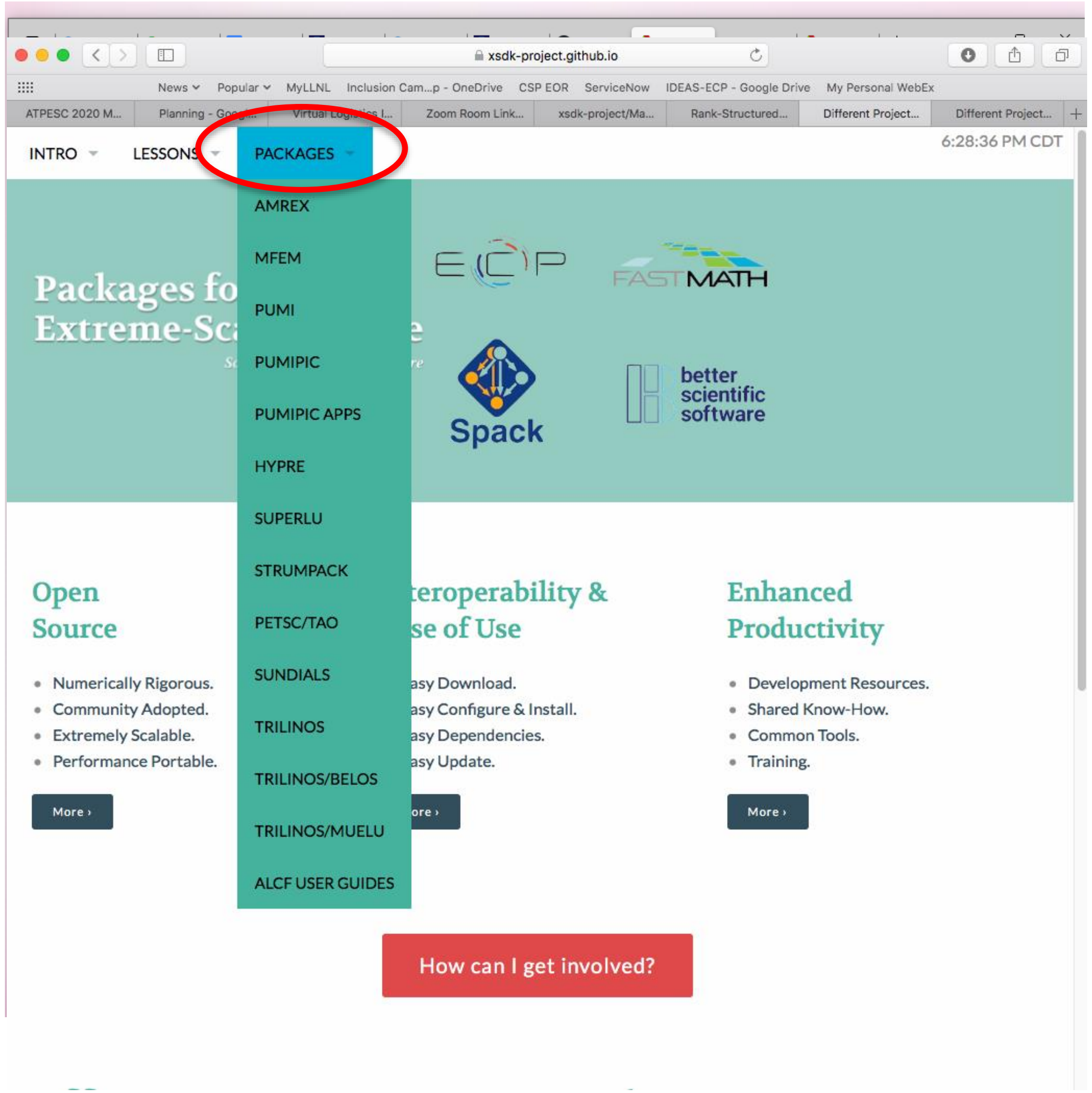
https://trilinos.github.io/muelu.html

https://xsdk-project.github.io/MathPackagesTraining2023/



- Hands-on Lessons

https://xsdk-project.github.io/MathPackagesTraining2023/



- Hands-on Lessons
- Packages

EXASCALE COMPUTING PROJECT

Agenda

https://extremecomputingtraining.anl.gov/agenda-2023/#Track-5

Time	Room?	Room?
8:30 – 9:30	Introduction to Numerical Software – Ulrike Yang	
9:30 – 10:45	Structured Discretization (AMReX) – Ann Almgren, Andrew Myers	Unstructured Discretization (MFEM/PUMI) – Tzanio Kolev, Mark Shephard, Cameron Smith
10:45 – 11:15	Break, Subject Matter Expert (SME) Selections, Panel Questions	
11:15 – 12:30	Iterative Solvers & Algebraic Multigrid (hypre) – Sarah Osborn, Ulrike Yang	Direct Solvers (SuperLU, STRUMPACK) – Sherry Li, Pieter Ghysels
12:30 – 1:30	Lunch, SME Selections, Panel Questions	
1:30 – 2:45	Nonlinear Solvers (PETSc) – Richard Mills	Time Integration (SUNDIALS) – David Gardner
2:45 – 3:15	Break, SME Selections, **Panel Questions Due**	
3:15 – 4:30	Optimization (TAO) – Tobin Isaac	Iterative Solvers & Algebraic Multigrid (Trilinos, Belos & MueLu) – Christian Glusa, Graham Harper
4:30 – 5:30	**Wrap-up (Ann Almgren) / Panel: Extreme-Scale Numerical Algorithms and Software**	
5:30 – 6:30	Unstructured Time: **SME Selections Due**, Informal Discussion, Continue Hands-on	
6:30 – 7:30	Dinner	
7:30 – 9:30	Optional Activity: SME Speed-dating	

Next steps: https://xsdk-project.github.io/MathPackagesTraining2023/agenda

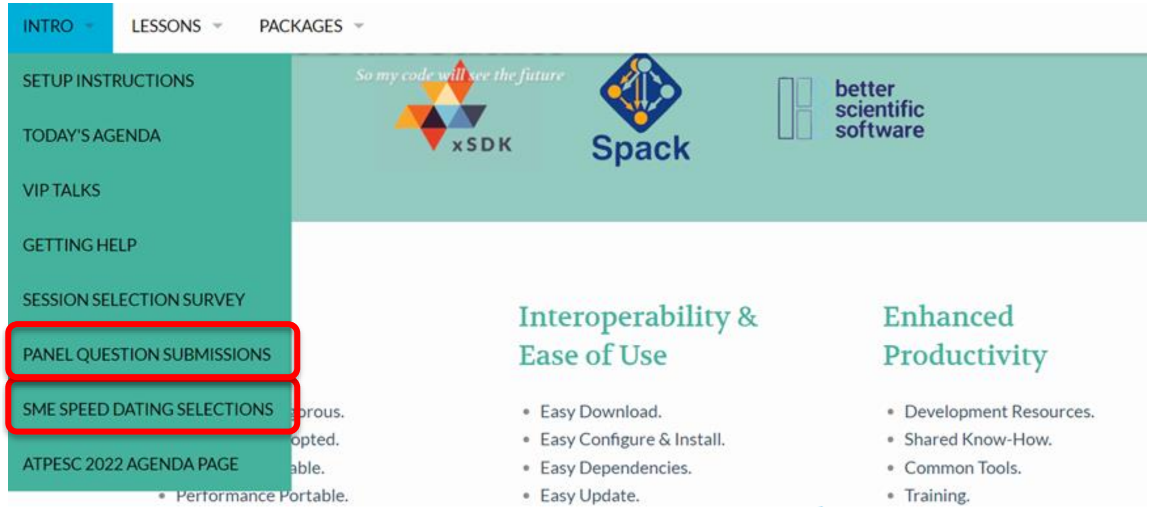
- **WrapUp (Ann Almgren) @ 4:30pm**

  Panel: Main Room @ 4:45 pm

- **SME Speed Dating: @7:30pm**

- **During breaks and lunch**
  - **Provide Panel Questions**
    Due: 3:15 pm
  - **Sign up for discussions with numerical software developers** (optional)
    - Your email address

    Due 6:30 pm



Panel: Extreme-Scale Numerical Algorithms and Software

- **Q&A Session:** ATPESC learners ask questions about working with numerical packages and the community of numerical package developers
  - Questions in **#numerical** slack channel and via Google form
- Panelists


Sherry Li, LBL


Toby Isaac, ANL


Graham Harper, SNL


Andrew Myers, LBL

- Moderator


Ann Almgren, LBL

Panel Question Submission Form

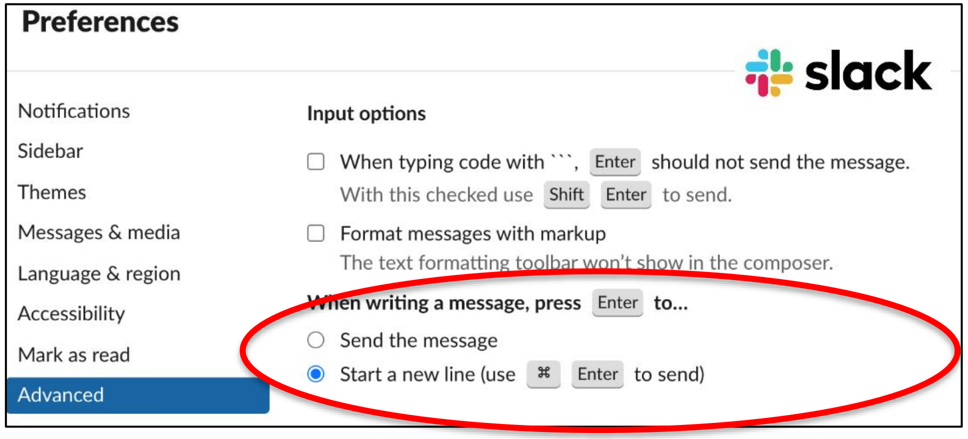
Please enter here a question you would like to ask our panelists during the 45 minute panel session.

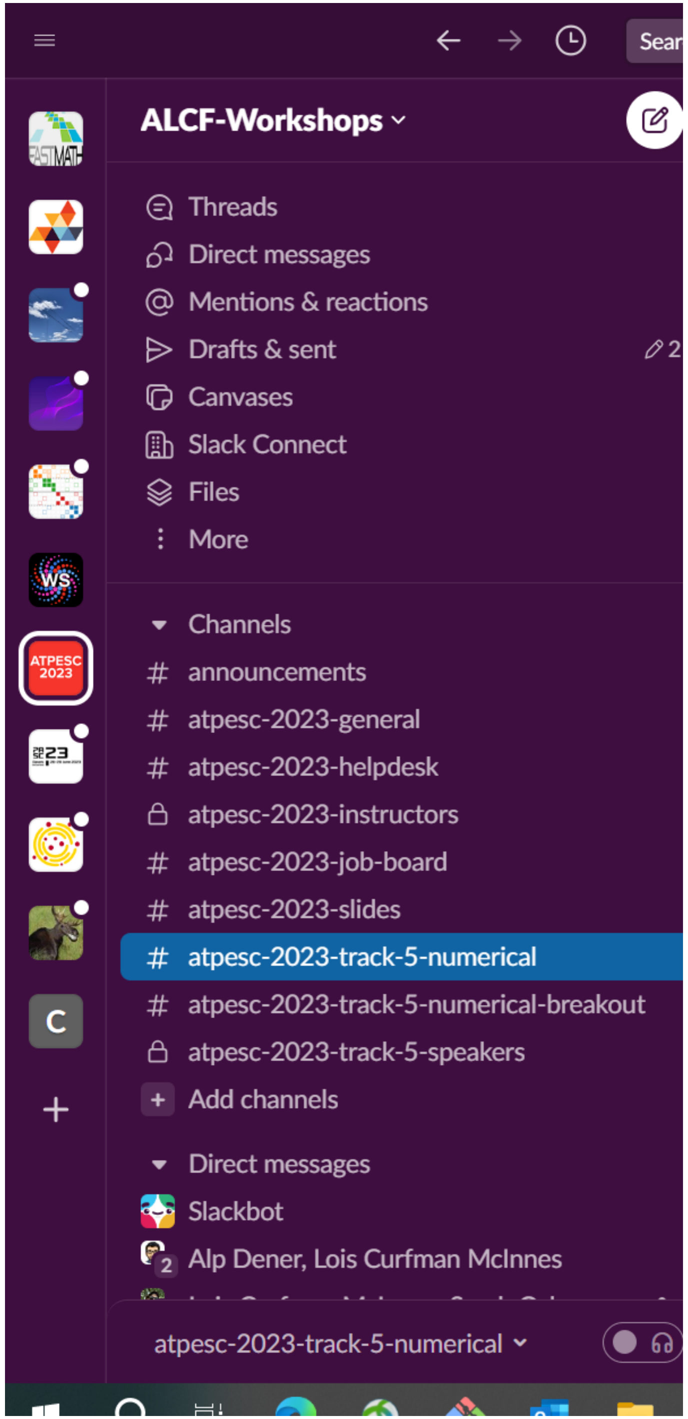
We ask that you please include your name in case we may need to call upon you to clarify your question.

Using Slack

- Recommend using the desktop app, but browser ok too
- **# atpesc-2023-track-5-numerical** channel
- **# atpesc-2023-track-5-numerical-breakout** channel
  - For all chat during presentations
  - For all chat outside any specific parallel session
  - For general help
  - Recommend using the thread option to help keep track of discussions on subtopics

Tip: Consider setting Preferences to customize when to send





Track 5: 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) and numerical optimization

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 provides 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

EXASCALE COMPUTING PROJECT

The ATPESC Team 2023

Extreme-scale numerical algorithms and software
Integrated lectures and hands-on examples, panel session, individual discussions … and more!


Ann Almgren, LBL


Andrew Myers, LBL


Tzanio Kolev, LLNL


Mark Shephard, RPI


Cameron Smith, RPI


Ulrike Yang, LLNL


Christian Glusa, SNL


Graham Harper, SNL


Sherry Li, LBL


Pieter Ghysels, LBL


Satish Balay, ANL


Sarah Osborn, LLNL


Toby Isaac, ANL


David Gardner, LLNL


Richard Mills, ANL

VIPs of ATPESC Extreme-Scale Numerical Software Track

- **Jim Demmel, UC Berkeley** [bio]
  - **Communication-Avoiding Algorithms**
    - ATPESC 2023, Thursday, August 10, 7:30pm
    - ENLA Seminar, June 2020 [video]

- **Jack Dongarra, Univ of Tennessee** [bio]
  - **Growing up at Argonne National Lab**
    - ATPESC 2023, Monday, August 7, 7:30pm
  - Adaptive Linear Solvers and Eigensolvers, ATPESC 2019 [video]

- **David Keyes, KAUST** [bio]
  - **Efficient Computation thorough Tuned Approximation**
    - ATPESC 2023, Tuesday, August 1, 7:30pm
  - Adaptive Nonlinear Preconditioning for PDEs with Error Bounds on Output Functionals, University of Manchester, 2021 [video]

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

EXASCALE COMPUTING PROJECT
https://exascaleproject.org

FASTMATH
https://scidac5-fastmath.lbl.gov

- Exascale Computing Project
- FASTMath SciDAC Institute
- Developers of xSDK packages

Community efforts: Join us!

xSDK
https://xsdk.info

better scientific software
https://bssw.io

E4S
https://e4s.io

… While improving software productivity & sustainability as key aspects of advancing overall scientific productivity

IDEAS productivity
https://ideas-productivity.org

- IDEAS Software Productivity Project
- Better Scientific Software Community

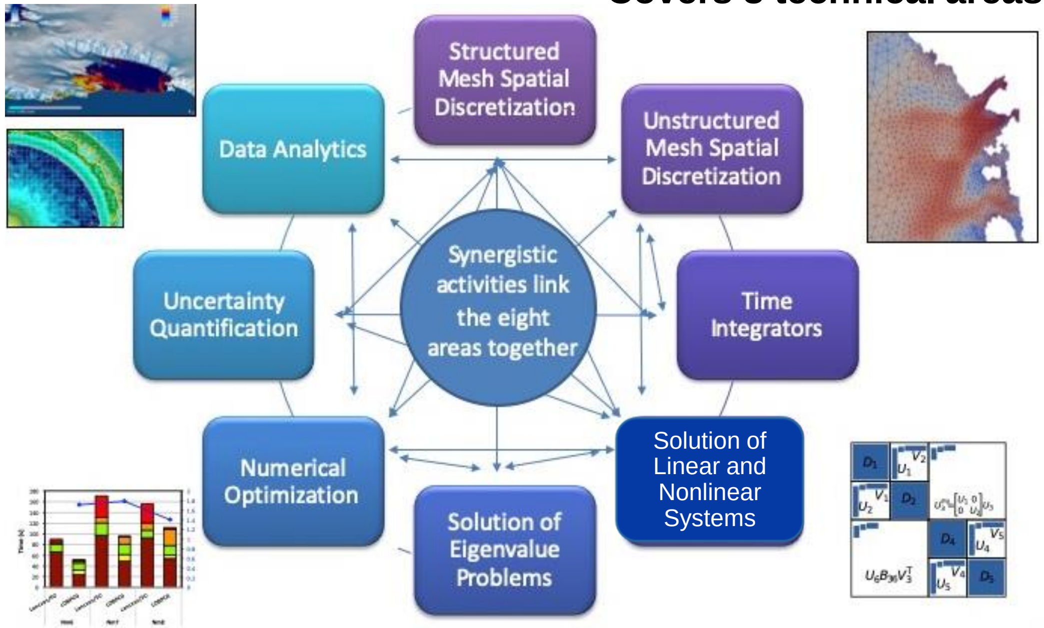
See also Track 3: Software Productivity and Sustainability (Aug 4)

EXASCALE COMPUTING PROJECT

FASTMath: Frameworks, Algorithms & Scalable Technologies for Mathematics

https://scidac5-fastmath.lbl.gov/

Covers 8 technical areas



FASTMath Goals:

- Develop advanced numerical techniques for DOE applications
- Deploy high-performance software on DOE supercomputers
- Demonstrate basic research technologies from applied mathematics
- Engage and support of the computational science community

100's of person years of experience building math software

SuperLU, AMReX, sundials, ZOLTAN, mfem, PETSc, hypre, UQTk, symPACK, PUMI, Trilinos, Albany, DAKOTA

50+ researchers from 5 DOE labs and 5 universities

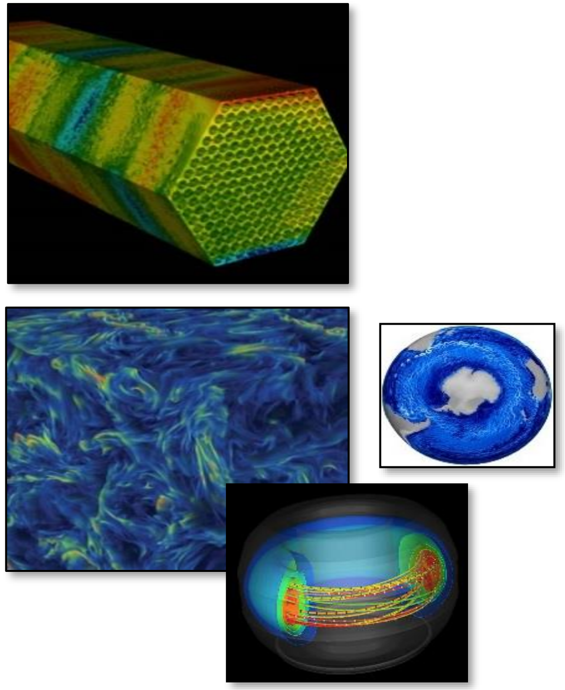
Berkeley Lab, Argonne National Laboratory, LLNL, Oak Ridge National Laboratory, Sandia National Laboratories

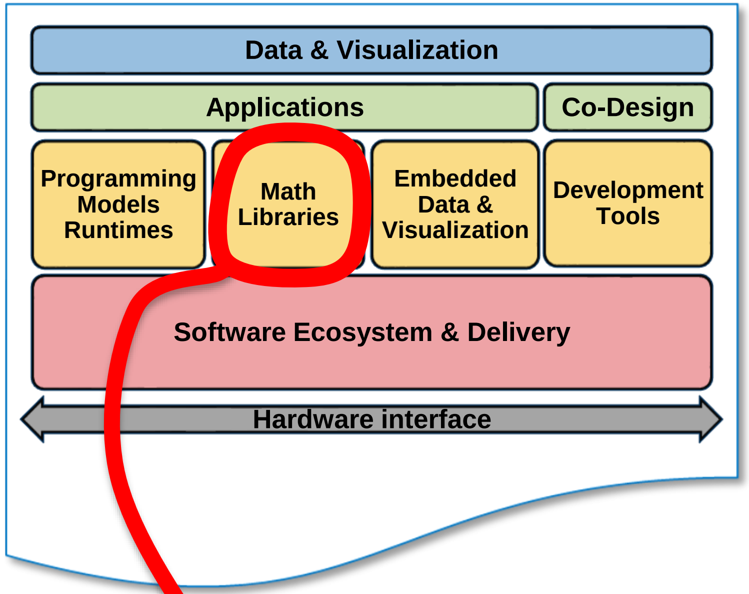
CU Boulder, USC, MIT, Rensselaer, SMU

Esmond G. Ng, FASTMath Institute Director (EGNg@lbl.gov)

ECP's holistic approach uses co-design and integration to achieve exascale computing

Application Development	Software Technology	Hardware and Integration
Science and mission applications	Scalable software stack	Relationships: facilities with AD/ST, with vendors





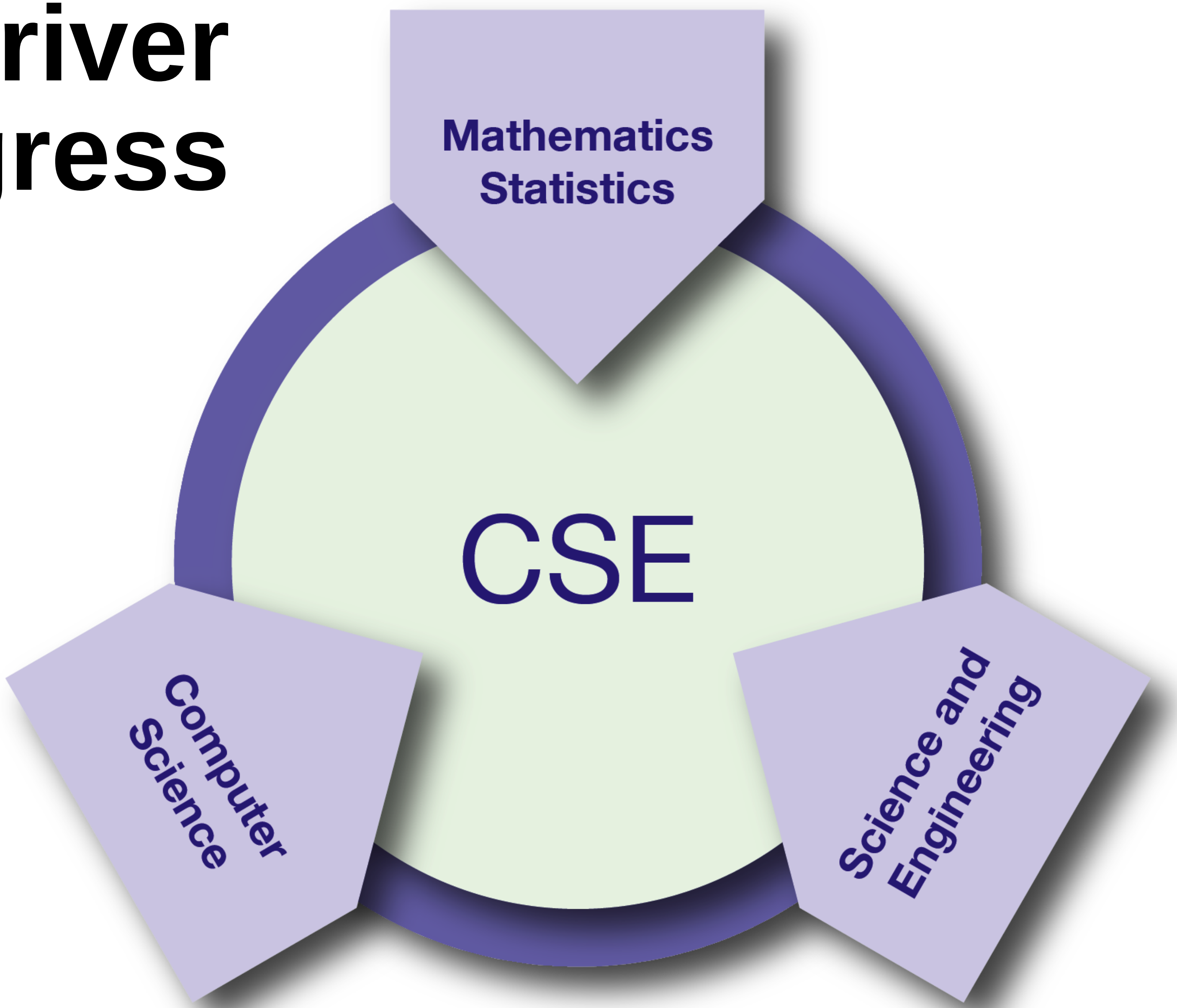
Emphasis for this presentation

CSE: Essential driver of scientific progress

CSE = Computational Science & Engineering

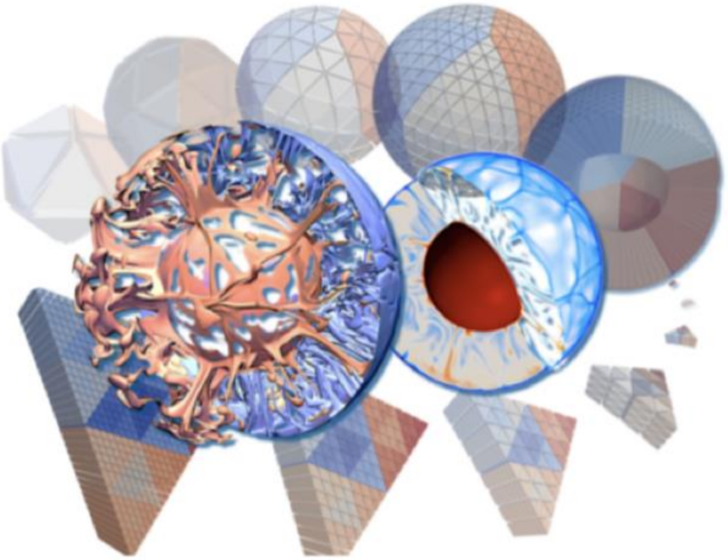
Development and use of computational methods for scientific discovery

- 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

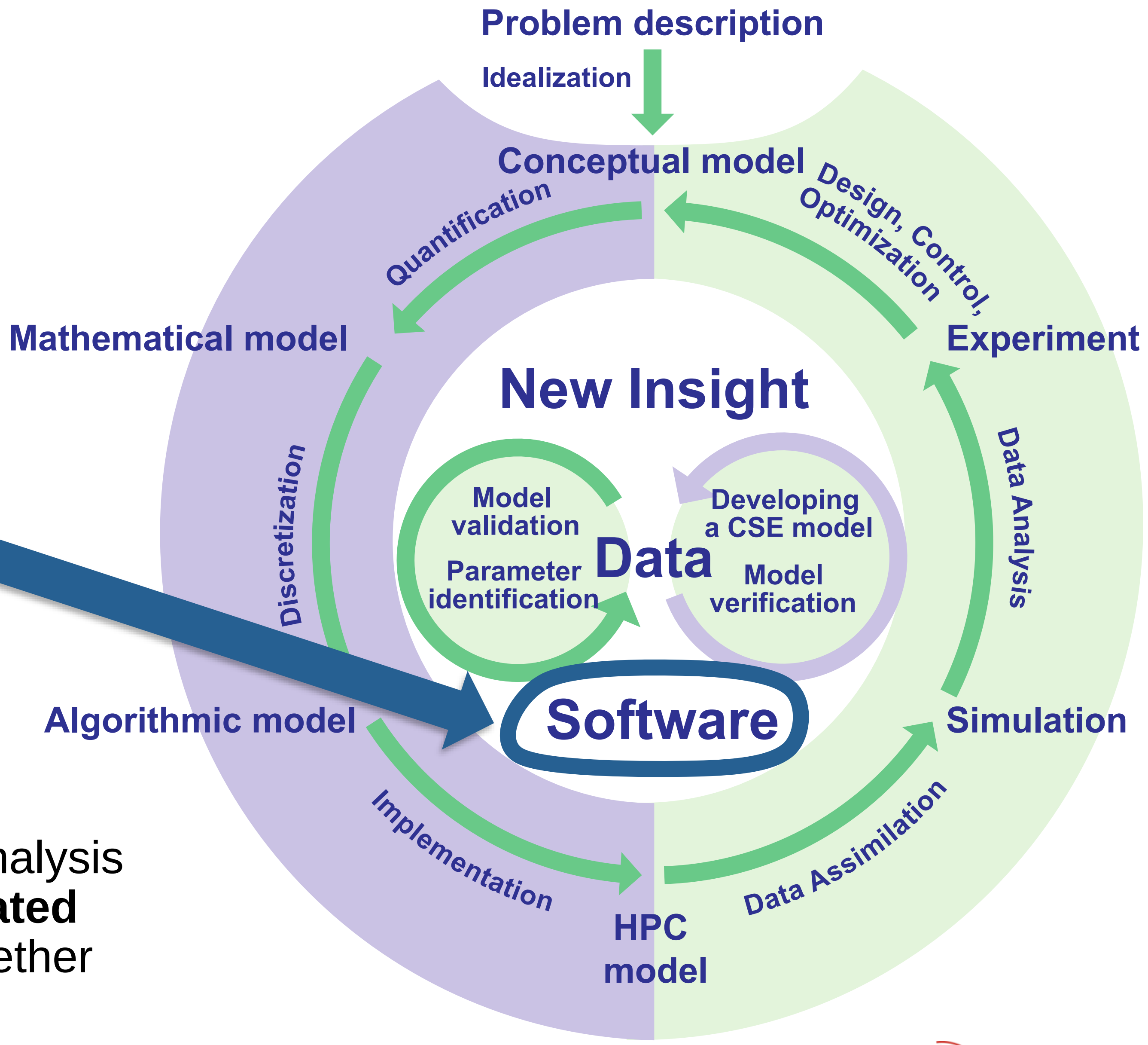


Research and Education in Computational Science & Engineering

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, ***SIAM Review***, *60(3),* Aug 2018, https://doi.org/10.1137/16M1096840.

EXASCALE COMPUTING PROJECT

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



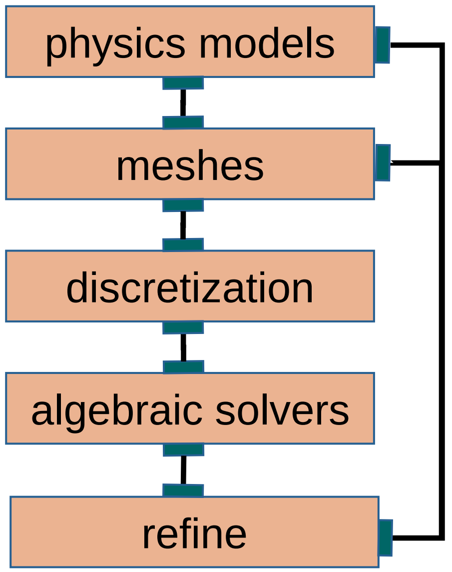
CSE cycle: Modeling, simulation, and analysis

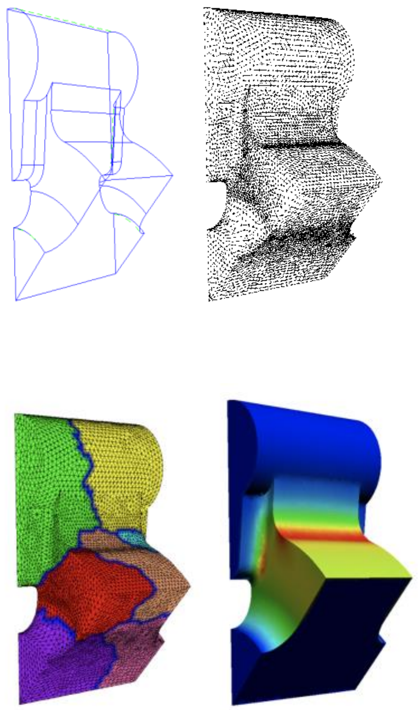
- **Software: independent but interrelated elements** for various phases that together enable CSE

CSE simulation starts with a forward simulation that captures 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

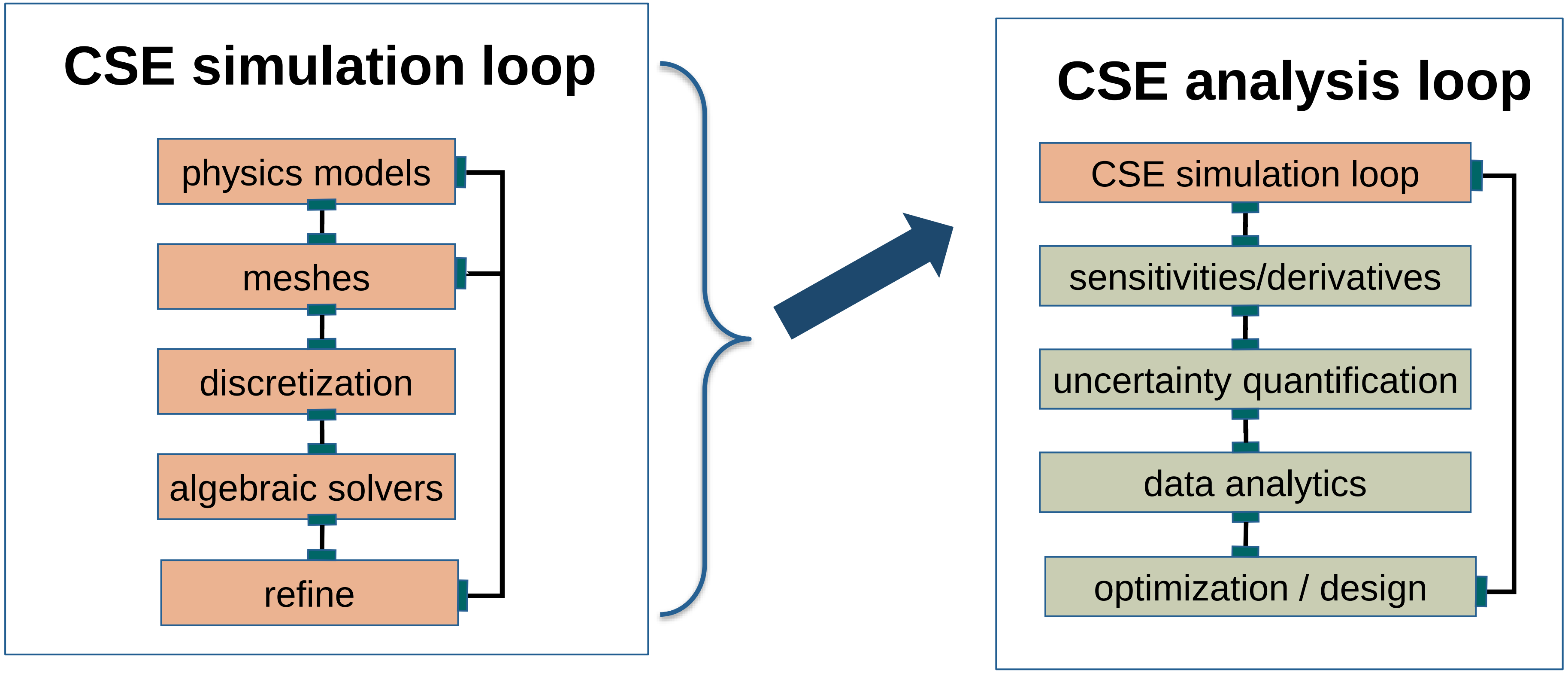
CSE simulation loop





Requires: mesh generation, partitioning, load balancing, high-order discretization, time integration, linear & nonlinear solvers, eigensolvers, mesh refinement, multiscale/multiphysics coupling, 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, ensembles, data analytics, uncertainty quantification, optimization (derivative free & 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

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

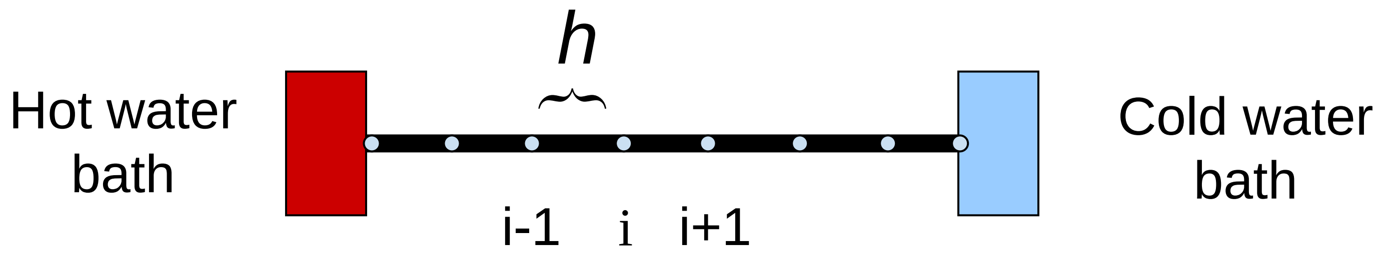
$$T(0) = 180^o \quad T(1) = 0^o$$

The first step is to discretize the equations

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

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

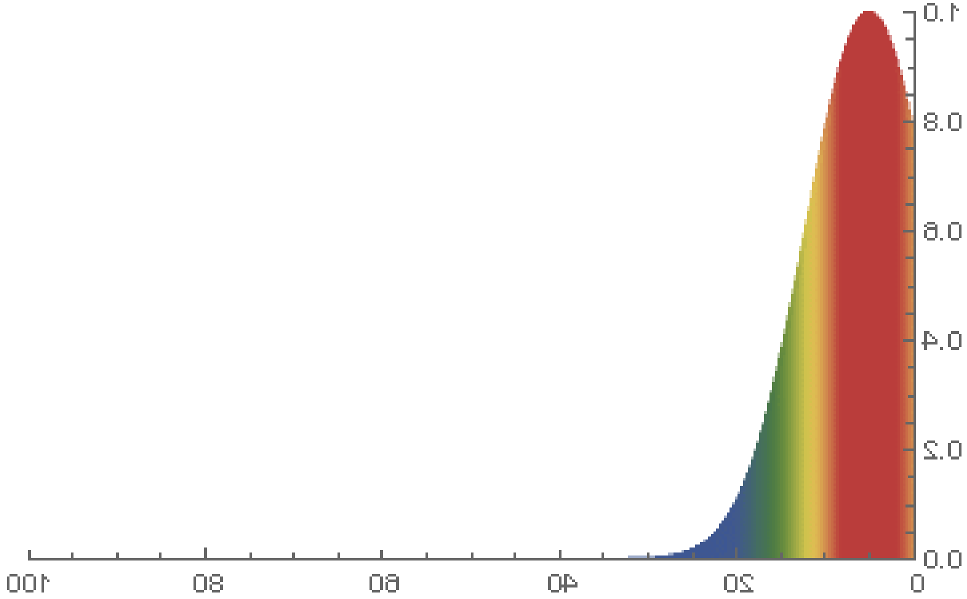
$$T_0 = 180^o \qquad T_n = 0^o$$



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

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

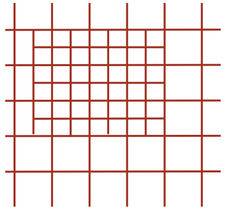
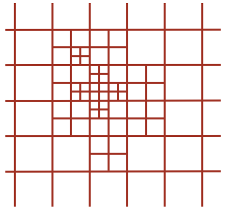
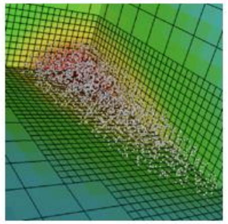
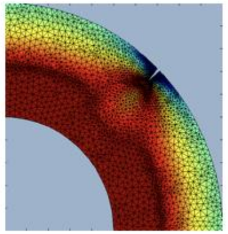
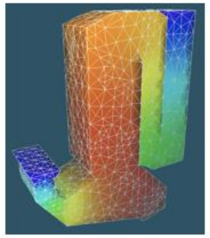


- Visualize and analyze the results

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

- Different discretization strategies exist for differing needs

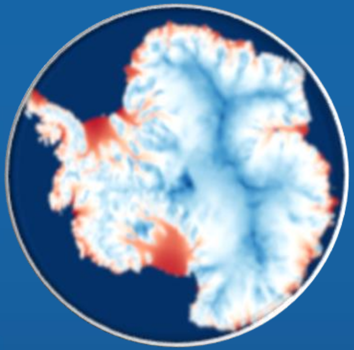
Efficiency ⟷ **Flexibility**

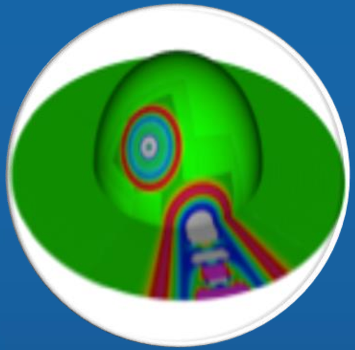
- Most problems are time dependent and nonlinear
  - Need higher algorithmic levels than linear solvers
- Increasingly combining multiple physical processes
  - Interactions require careful handling
- Goal-oriented problem solving requires optimization, uncertainty quantification

EXASCALE COMPUTING PROJECT

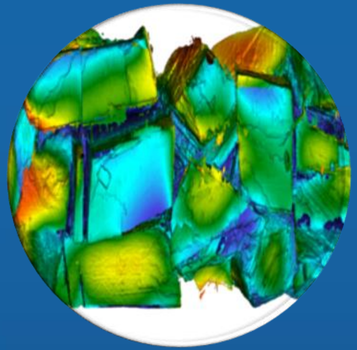
Structured grid efforts focus on high-order, mapped grids, embedded boundaries, AMR, and particles



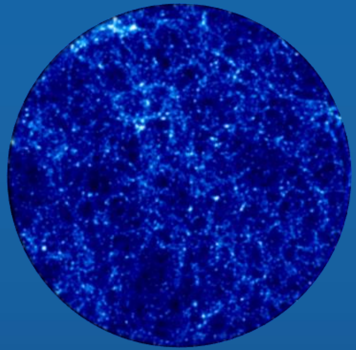
Structured AMR



Mapped-multiblock grids



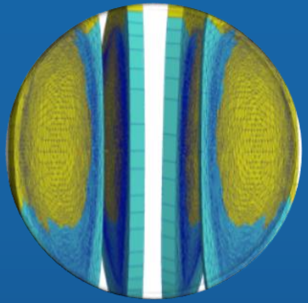
Embedded boundary methods



Particle-based methods

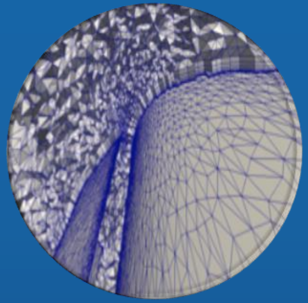
Application to cosmology, astrophysics, accelerator modeling, fusion, climate, subsurface reacting flows, low mach number combustion, etc.

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

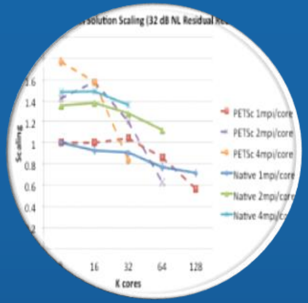


Parallel mesh infrastructures

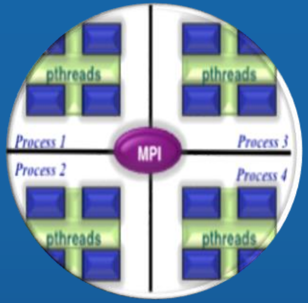
Dynamic load balancing



Mesh adaptation and quality control



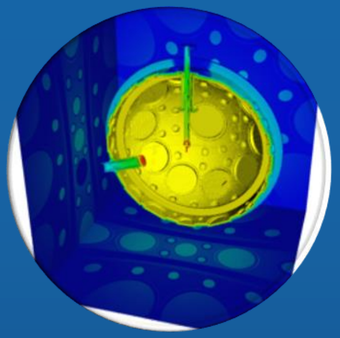
Parallel performance on unstructured meshes



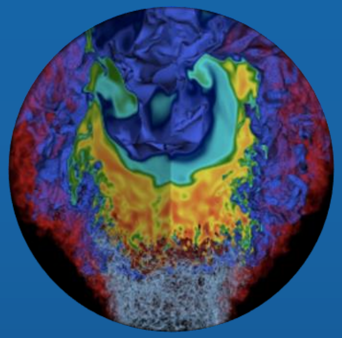
Architecture aware implementations

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

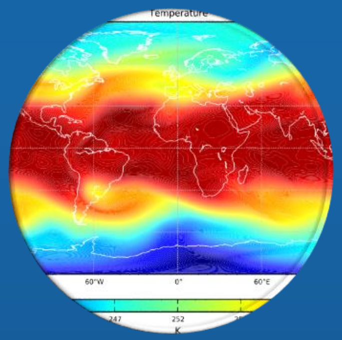


IMEX, multi-rate methods



Spectral deferred correction techniques

Parallel in time

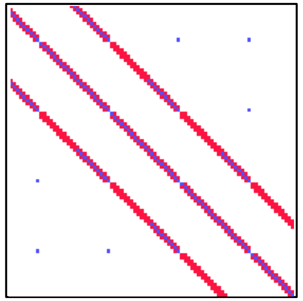


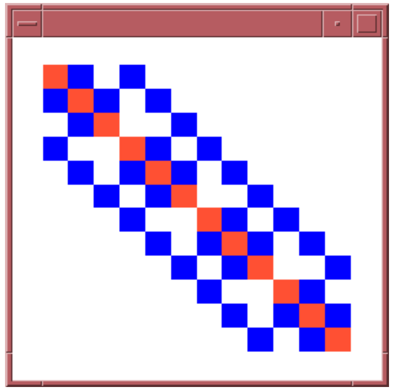
Constrained problems

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

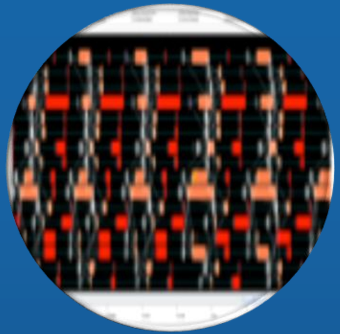
As problems grow in size, so do corresponding discrete systems

- 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.

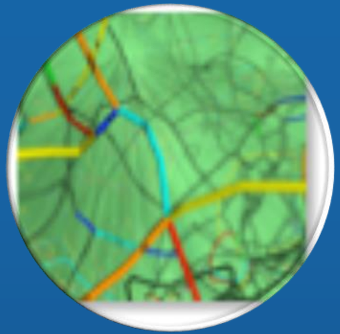




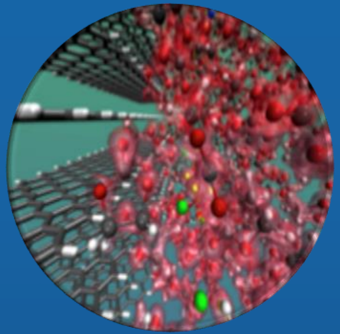
Research on algebraic systems provides key solution technologies to applications



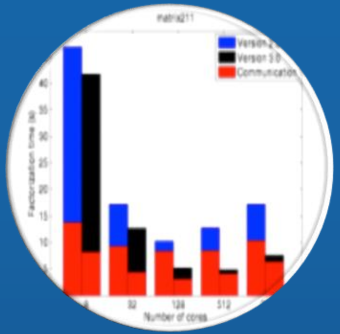
Linear system solution using direct and iterative solvers



Nonlinear system solution using acceleration techniques and globalized Newton methods



Eigensolvers using iterative techniques and optimization



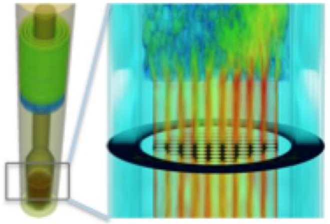
Architecture aware implementations

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

Multiphysics: A primary motivator for exascale

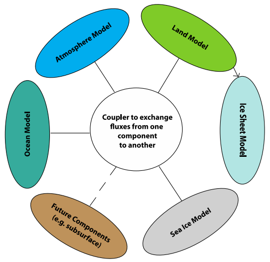
Multiphysics: greater than 1 component governed by its own principle(s) for evolution or equilibrium

– Also: broad class of coarsely partitioned problems possess similarities

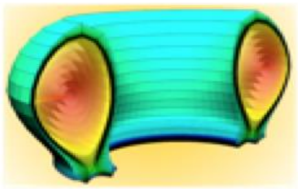


nuclear reactors
A. Siegel, ANL

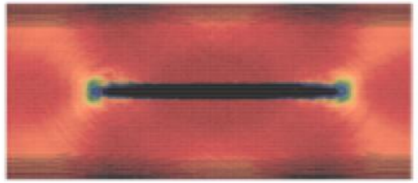
particle accelerators
K. Lee, SLAC



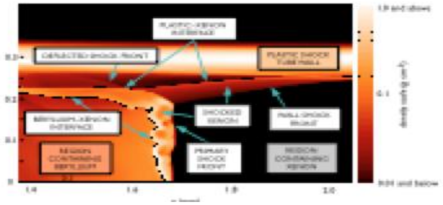
climate
K. Evans, ORNL



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

Multiphysics simulations: Challenges and opportunities

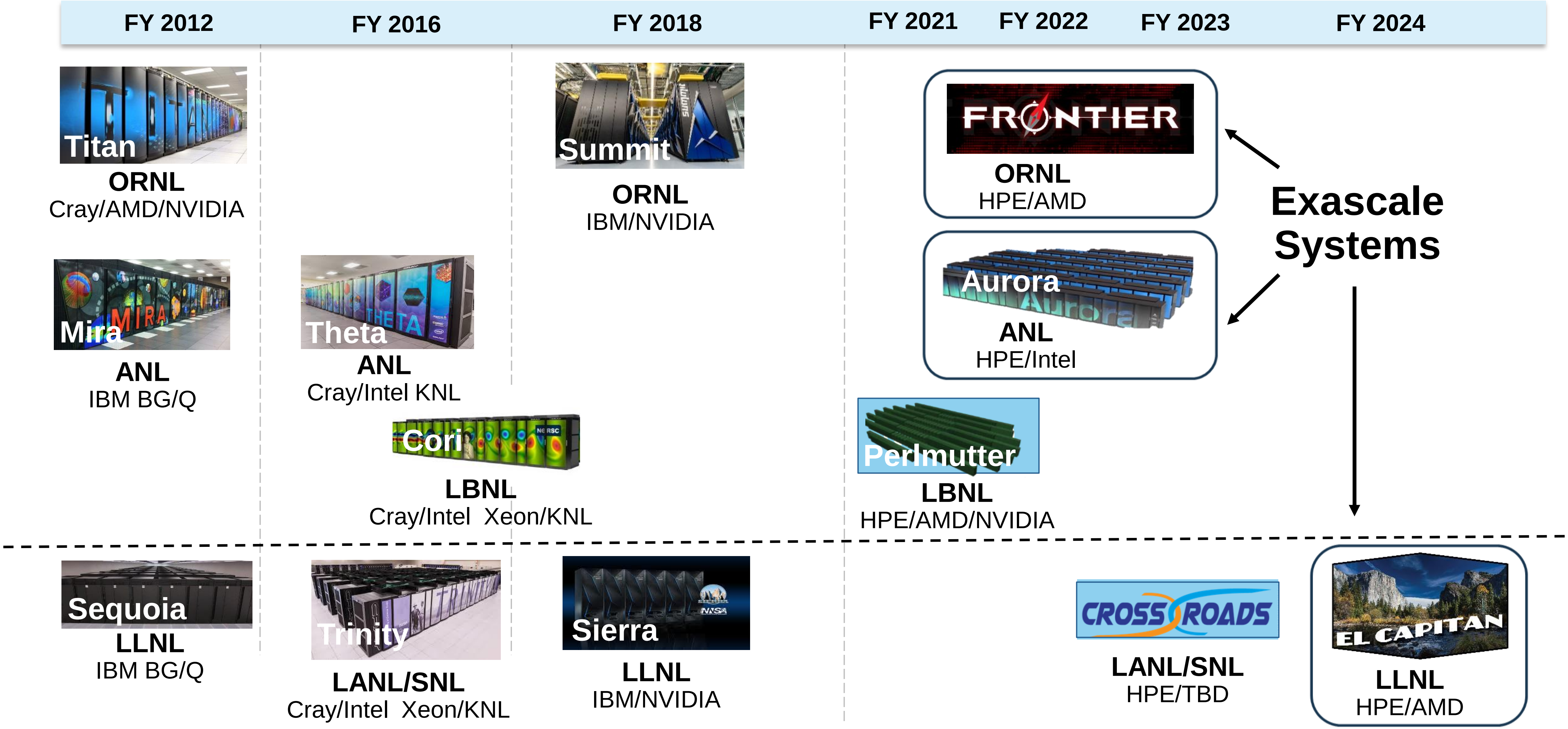
The International Journal of High Performance Computing Applications
27(1) 4–83

DOI: 10.1177/1094342012468181
hpc.sagepub.com
SAGE

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

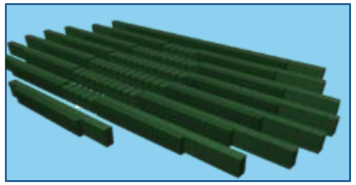
DOE HPC Roadmap to Exascale Systems

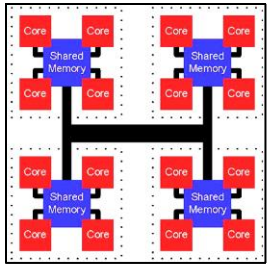


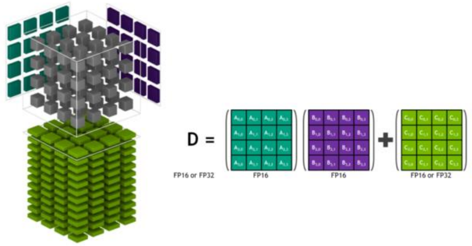
EXASCALE COMPUTING PROJECT

Disruptive changes in HPC architectures

- **Extreme levels of concurrency**
  - Increasingly deep memory hierarchies
  - Very high node and core counts
- **Additional complexities**
  - Hybrid architectures
  - GPUs, multithreading, manycore
  - Relatively poor memory latency and bandwidth
  - Challenges with fault resilience
  - Must conserve power – limit data movement
  - New (not yet stabilized) programming models
  - Etc.







- **Research advances: On-node and inter-node capabilities**
  - Reduce communication and synchronization
  - Increase concurrency
  - Address memory footprint
  - Enable large communication/computation overlap
  - Use GPUs and multithreading
  - Compare task and data parallelism
  - Low-level kernels for vector operations that support hybrid programming models
  - Mixed precision (leverage compute power available in low-precision tensor cores)
  - Etc.

Software libraries facilitate progress in computational science and engineering

- **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.github.io/
- **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

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

EXASCALE COMPUTING PROJECT

ECP applications need sustainable coordination among math libraries

ECP AD Teams

Combustion-Pele, EXAALT, ExaAM, ExaFEL, ExaSGD, ExaSky, ExaStar, ExaWind, GAMESS, MFIX-Exa, NWChemEx, Subsurface, WarpX, WDMApp, WarpX, ExaAM, ATDM (LANL, LLNL, SNL) apps, AMReX, CEED, CODAR, CoPA, ExaLearn

Examples:

- **ExaAM:** DTK, hypre, PETSc, Sundials, Tasmanian, Trilinos, FFT, etc.
- **ExaWind:** hypre, KokkosKernels, SuperLU, Trilinos, FFT, etc.
- **WDMApp:** PETSc, hypre, SuperLU, STRUMPACK, FFT, etc.
- **CEED:** MFEM, MAGMA, hypre, PETSc, SuperLU, Sundials, etc.
- And many more …

ECP Math Libraries

MFEM

DTK

Software libraries are not enough

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

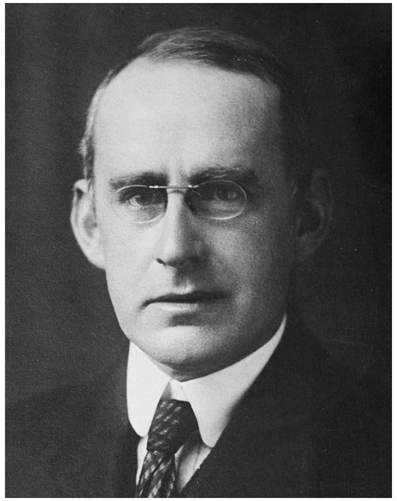
- **Need consistency** of compiler (+version, options), 3rd-party packages, etc.
- **Namespace and version conflicts** make simultaneous build/link of packages difficult
- **Multilayer interoperability** requires careful design and sustainable coordination

Need software <u>ecosystem</u> perspective

Ecosystem: A group of independent but interrelated elements comprising a unified whole

Ecosystems are challenging!

"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.' "



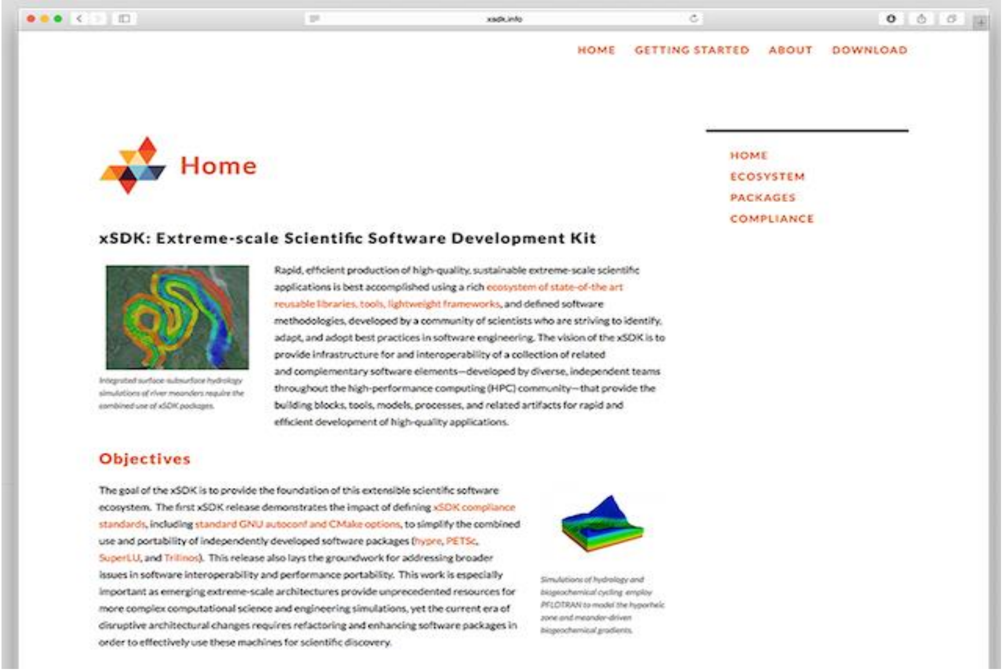
− Sir Arthur Stanley Eddington (1892−1944), British astrophysicist

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



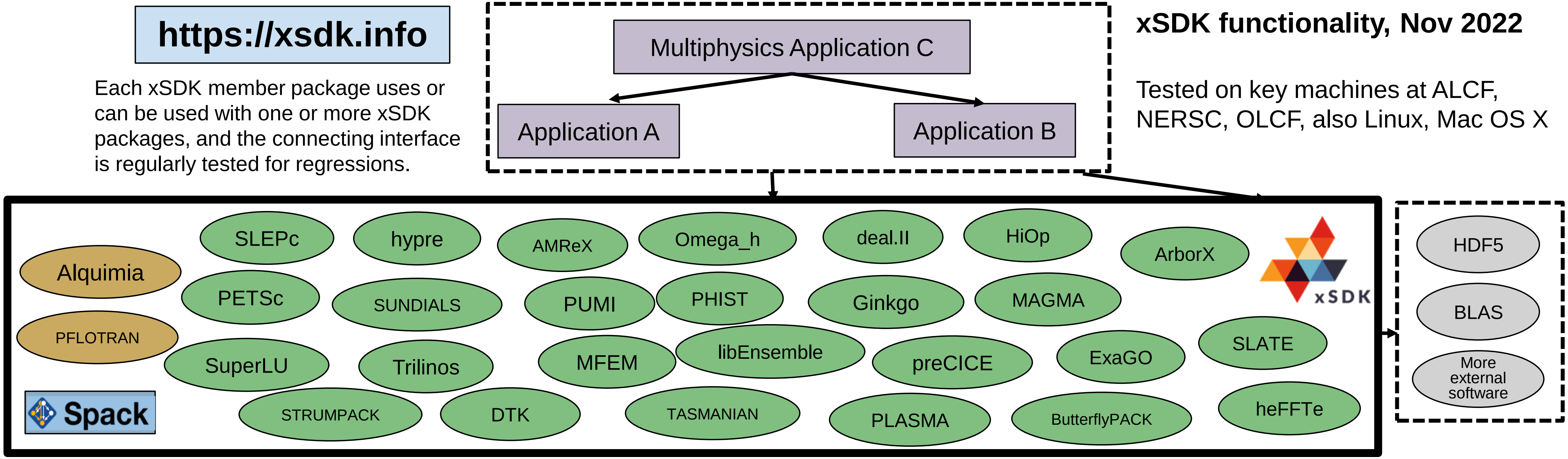
website: xSDK.info

xSDK Version 0.8.0: November 2022

https://xsdk.info

https://xsdk.info

Each xSDK member package uses or can be used with one or more xSDK packages, and the connecting interface is regularly tested for regressions.

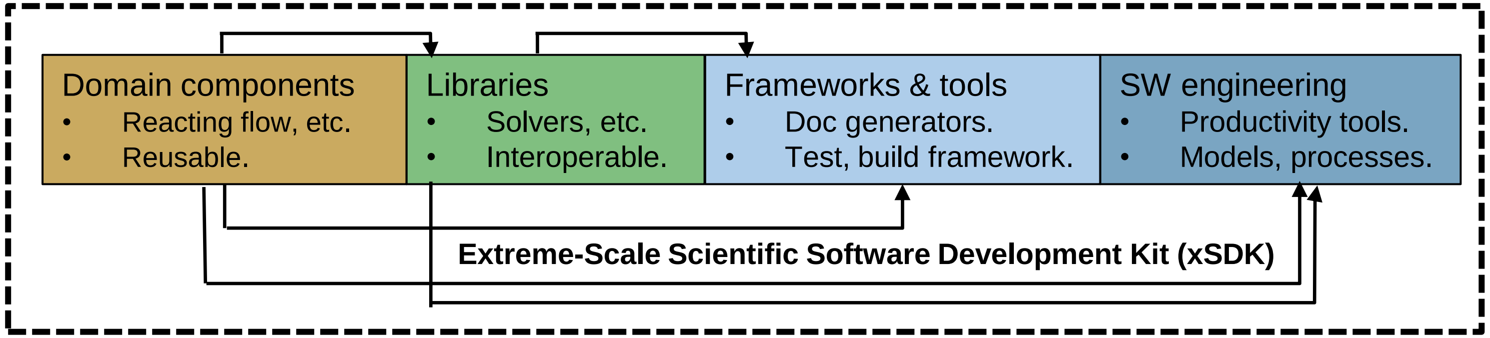


xSDK functionality, Nov 2022

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

November 2022

- 26 math libraries
- 2 domain components
- 16 mandatory xSDK community policies
- Spack xSDK installer

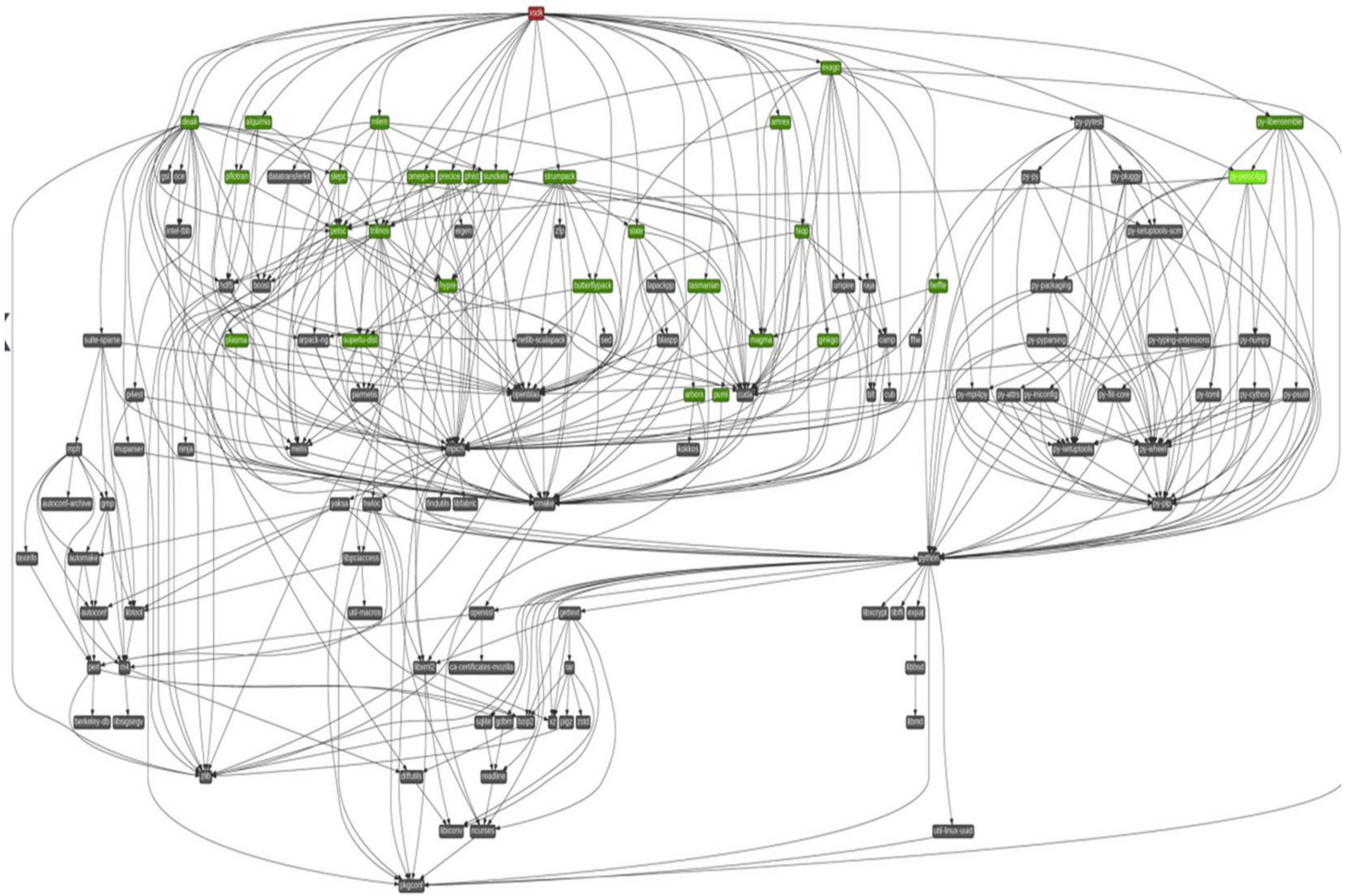


Impact: Improved code quality, usability, access, sustainability

Foundation for work on performance portability, deeper levels of package interoperability

The xSDK is using Spack to deploy its software

- The xSDK packages depend on a number of open-source libraries
- Spack is a flexible package manager for HPC (see Track 3: Software Productivity and Sustainability)
- Spack allows the xSDK to be deployed with a single command
  - User can optionally choose compilers, build options, etc.



Member Pkg | Link Dependency

https://spack.io   github.com/spack

xSDK Libraries

- **AMReX**: Ann Almgren (LBNL)
- **ArborX**: Daniel Arndt (ORNL)
- **DTK**: Bruno Turcksin (ORNL)
- **deal.II**: Wolfgang Bangerth (Colorado State University)
- **ExaGO:** Shrirang Abhyankar (PNNL)
- **Ginkgo**: Hartwig Anzt (Karlsruhe Institute of Technology)
- **heFFTe:** Stan Tomov (UTK)
- **HiOp:** Cosmin Petra (LLNL)
- **hypre**: Rob Falgout, Ulrike Yang (LLNL)
- **libEnsemble**: Steve Hudson (ANL)
- **MAGMA** and **PLASMA**: Piotr Luszczek (UTK)
- **MFEM**: Tzanio Kolev (LLNL)
- **Omega_h, PUMI**: Cameron Smith (RPI)
- **PETSc/TAO**: Satish Balay, Todd Munson (ANL)
- **preCICE:** Frederic Simonis (Technical University Munich)
- **SUNDIALS**: Cody Balos, David Gardner, Carol Woodward (LLNL)
- **SuperLU, STRUMPACK, ButterflyPACK**: Sherry Li, Pieter Ghysels, Yang Liu (LBNL)
- **TASMANIAN**: Miroslav Stoyanov (ORNL)
- **Trilinos**: Jim Willenbring (SNL)
- **PHIST**: Jonas Thies (DLR, German Aerospace Center)
- **SLEPc**: José Roman (Universitat Politècnica de València)

xSDK: https://xsdk.info

Building the foundation of an extreme-scale scientific software ecosystem

xSDK community policies: Help address challenges in interoperability and sustainability of software developed by diverse groups at different institutions

https://github.com/xsdk-project/xsdk-community-policies

<u>xSDK compatible package</u>: must satisfy the mandatory xSDK policies (M1, ..., M17)

Topics include configuring, installing, testing, MPI usage, portability, contact and version information, open-source licensing, namespacing, and repository access

Also specify **recommended policies**, which currently are encouraged but not required (R1, ..., R8)

Topics include public repository access, error handling, freeing system resources, and library dependencies, documentation quality

<u>xSDK member package</u>:

(1) Must be an xSDK-compatible package, *and*

(2) it uses or can be used by another package in the xSDK, and the connecting interface is regularly tested for regressions.

xSDK policies 1.0.0: Feb 2023

- Facilitate combined use of independently developed packages

Impact:

- Improved code quality, usability, access, sustainability
- Foundation for work on deeper levels of interoperability and performance portability

We encourage feedback and contributions!

EXASCALE COMPUTING PROJECT

xSDK community policies

https://github.com/xsdk-project/xsdk-community-policies

https://xsdk.info/policies

xSDK

Version 1.0.0, February 2023

Mandatory xSDK policies: must be satisfied

M1. Support portable installation through Spack (includes xSDK Spack variant guildelines)

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 publicly available repository.

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. Have a debug build option.

M17. Provide sufficient documentation to support use and further development.

Recommended xSDK policies: currently encouraged, but not required

R1. Provide at least one validation test that can be invoked through Spack.

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.

R6. Provide versions of dependencies.

R7. Have README, SUPPORT, LICENSE, and CHANGELOG file in top directory.

R8. Provide version comparison preprocessor macros.

xSDK member package: 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 software?

ECP EXASCALE COMPUTING PROJECT

xSDK community policies

https://github.com/xsdk-project/xsdk-community-policies

https://xsdk.info/policies

Version 1.0.0, February 2023

Mandatory xSDK policies: must be satisfied

M1. Support portable installation through Spack (includes xSDK Spack variant guildelines)

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 publicly available repository.

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. Have a debug build option.

M17. Provide sufficient documentation to support use and further development.

Recommended xSDK policies: currently encouraged, but not required

R1. Provide at least one validation test that can be invoked through Spack.

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.

R6. Provide versions of dependencies.

R7. Have README, SUPPORT, LICENSE, and CHANGELOG file in top directory.

R8. Provide version comparison preprocessor macros.

xSDK member package: 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.

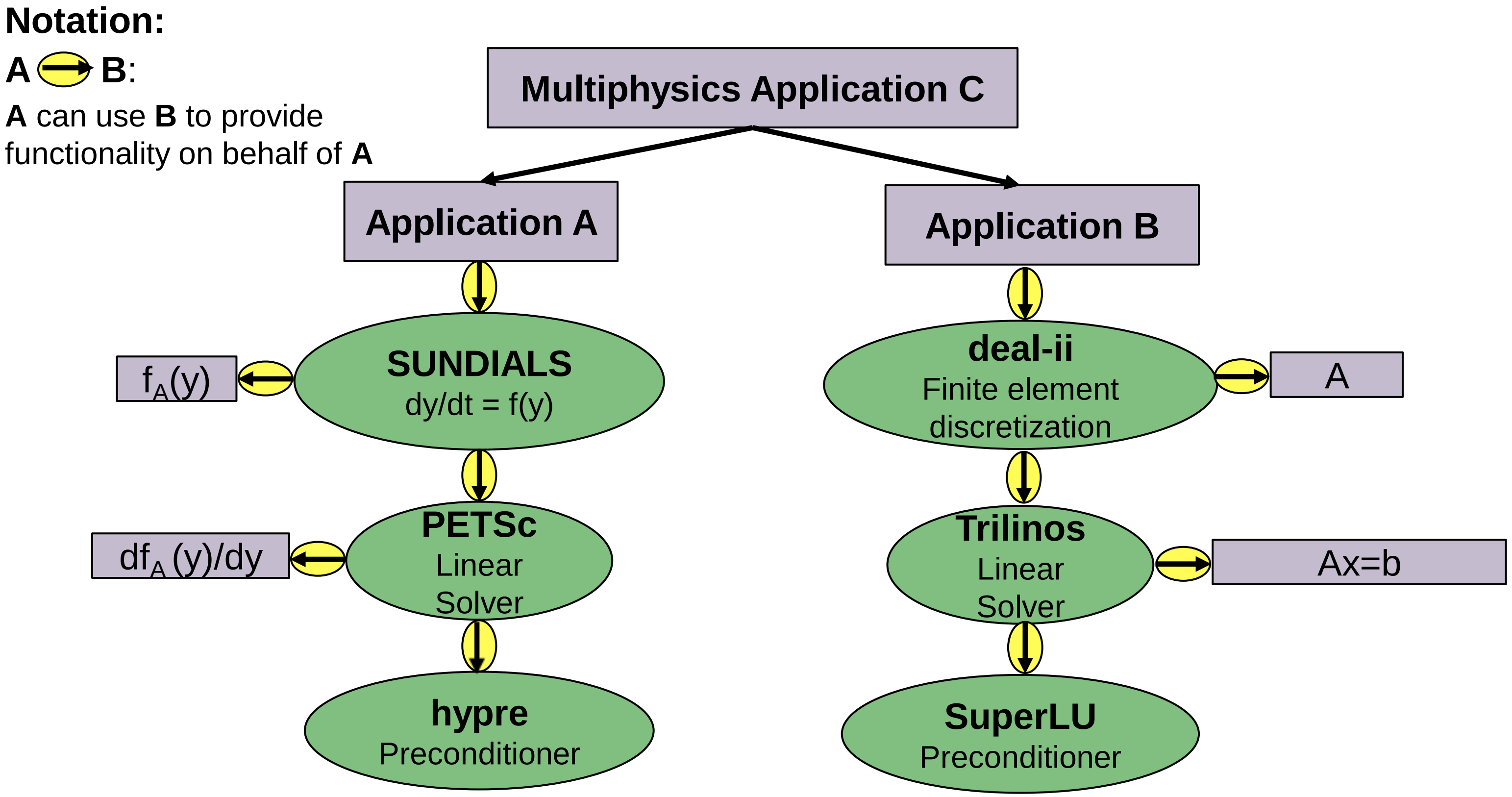
We welcome feedback. What policies make sense for your software?

EXASCALE COMPUTING PROJECT

Interoperability is challenging, particularly for deeper levels!

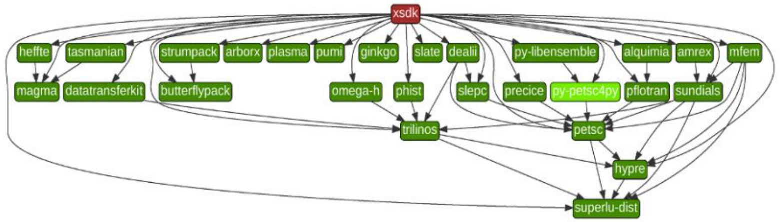
Levels of package interoperability:

- **Interoperability level 1**
  - Both packages can be used (side by side) in an application
- **Interoperability level 2**
  - The libraries can exchange data (or control data) with each other
- **Interoperability level 3**
  - Each library can call the other library to perform unique computations



xSDK4ECP: Focus on inter-package functionality, denoted by
- Coordinating use of on-node resources
- Integrated execution (control inversion, adaptive execution strategies)

Technical Challenges

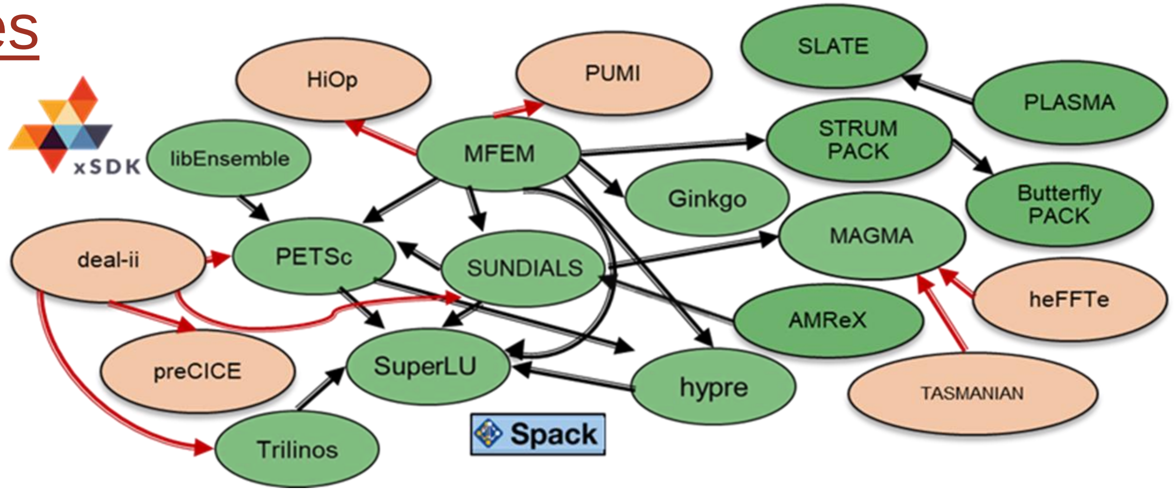


- Staying up to date while facing continual changes
  - xSDK release schedule not aligned with individual xSDK library and Spack release schedule
  - Lower dependencies can cause additional problems
- Testing difficulties
  - CI failures need to be investigated to understand what is broken and who should fix it
  - Often there is more than one package causing the issue, but finding the issues is a sequential process,
    i.e., the first issue needs to be fixed before the next one is discovered
  - The responsible package developers need to be contacted
  - Consistent oversight requires more people to respond to CI failures
- Designed test plan
  - Improve xSDK-examples test suite and integrate it with the xSDK testing process
  - Evaluate and extend current xSDK CI testing through the definition and use of hierarchical test layers, addition of new platforms and increased oversight of test results

EXASCALE COMPUTING PROJECT

Multi-library example codes demonstrating interoperability

- Suite of example codes has been made available in a github repository and included in the xSDK documentation. : https://github.com/xsdk-project/xsdk-examples
- The example codes are a demonstration of interoperability between xSDK libraries and provide training for xSDK library users interested in using these capabilities.



- Difficulty in building via `spack install xsdk-examples', since new interoperabilities generally not enabled in spack and/or xSDK yet. Provide simple build via `cmake'.
- Test suite important piece of xSDK testing strategy plan

xSDK community policies

https://github.com/xsdk-project/xsdk-community-policies

https://xsdk.info/policies

Version 1.0.0, February 2023

Mandatory xSDK policies: must be satisfied

M1. Support portable installation through Spack (includes xSDK Spack variant guildelines)

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 publicly available repository.

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. Have a debug build option.

M17. Provide sufficient documentation to support use and further development.

Recommended xSDK policies: currently encouraged, but not required

R1. Provide at least one validation test that can be invoked through Spack.

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.

R6. Provide versions of dependencies.

R7. Have README, SUPPORT, LICENSE, and CHANGELOG file in top directory.

R8. Provide version comparison preprocessor macros.

xSDK member package: 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 software?

EXASCALE COMPUTING PROJECT

What is performance portability?

- Discussion at 2016 DOE Center of Excellence meeting, Phoenix, AZ, USA (https://performanceportability.org/perfport/definition/)
  - Attendees included scientists, engineers for DOE’s Officie of Science and National Nuclear Security Agency, as well as vendors (Intel, NVIDIA, IBM, etc)
- "For the purposes of this meeting, it is the ability to run an application with acceptable performance across KNL and GPU-based systems with a single version of source code." (Rob Neely)
- "An application is performance portable if it achieves a consistent level of performance (e.g., defined by execution time or other figure of merit (not percentage of peak flops across platforms)) relative to the best-known implementation on each platform." (John Pennycook, Intel)
- "Hard portability = no code changes and no tuning. Software portability = simple code mods with no algorithmic changes. Non-portable = algorithmic changes" (Adrian Pope, Vitali Morozov)
- (Performance portability means) the same source code will run productively on a variety of different architectures" (Larkin)
- "Code is performance portable when the application team says its performance portable!" (Richards)

What is performance portability? (continued)

- Conclusion: There is currently no universally accepted definition of performance portability

An application is performance portable if it achieves a consistent ratio of the actual time to solution to either the best-known or the theoretical best time to solution on each platform with minimal platform specific code required.

BSSw, Anshu Dubey:
An application has portable performance if in addition to running on diverse platforms it exhibits similar accuracy, stability, and reliability across these platforms for a given configuration. Moreover, the time to solution should reflect efficient utilization of available computational resources on each platform.

Portability Strategies of xSDK Libraries

- Use of portable programming models that provide abstractions

  HIP

- Use of abstraction to limit code that interacts with devices
- Use of fast kernel libraries designed for individual architectures

  cuBLAS, cuSPARSE
  rocBLAS, rocSPARSE
  MKL

- Write own CUDA kernels, and use vendor provided tools to port kernels
- Develop new algorithms more suitable for GPUs
  (most challenging, but possibly best results!)

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

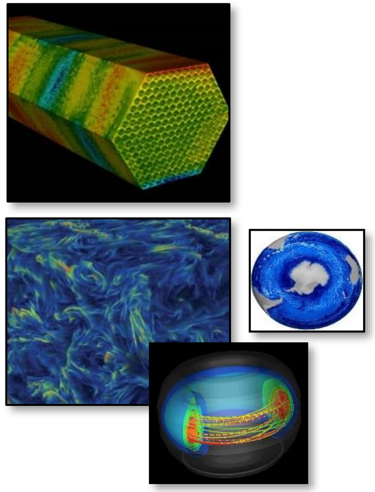
Next-generation algorithms

Advances in data structures for new node architectures

Improving library quality, sustainability, interoperability

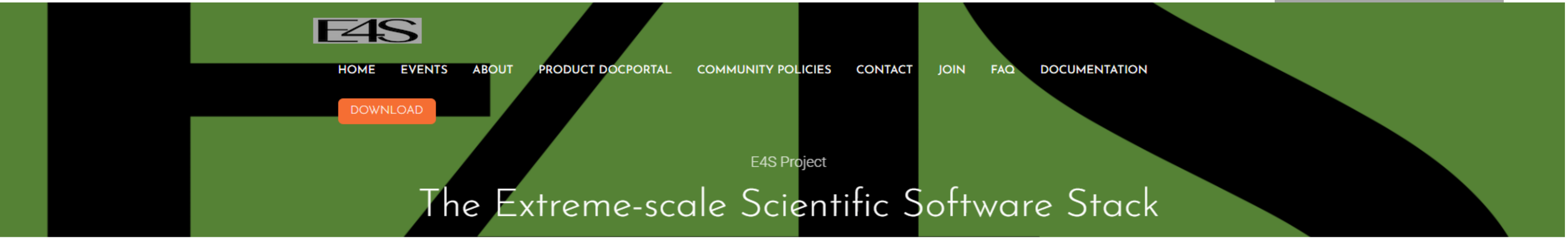
→ **Toward predictive scientific simulations**

→ **Increasing performance, portability, productivity**



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

Extreme-scale Scientific Software Stack (E4S)
https://e4s.io



- E4S is a community effort to provide open-source software packages for developing, deploying, and running scientific applications on HPC platforms.
- E4S provides containers and turn-key, from-source builds of 100+ popular HPC, EDA (e.g., Xyce), and AI/ML packages (e.g., TensorFlow, PyTorch)
- Container images on DockerHub and E4S website of pre-built binaries of ECP ST products
- E4S Spack build cache has over 100,000 binaries.
- Platforms: x86_64, ppc64le, and aarch64. GPUs runtimes: NVIDIA (CUDA), AMD (ROCm), and Intel (OneAPI)
- E4S DocPortal provide a single online location for *accurate* product descriptions for software products.
- E4S helps applications reduce the burden to install dependencies
- Quarterly releases: E4S 23.05 released on May 31, 2023: https://e4s.io/talks/E4S_23.05.pdf

EXASCALE COMPUTING PROJECT

E4S Summary

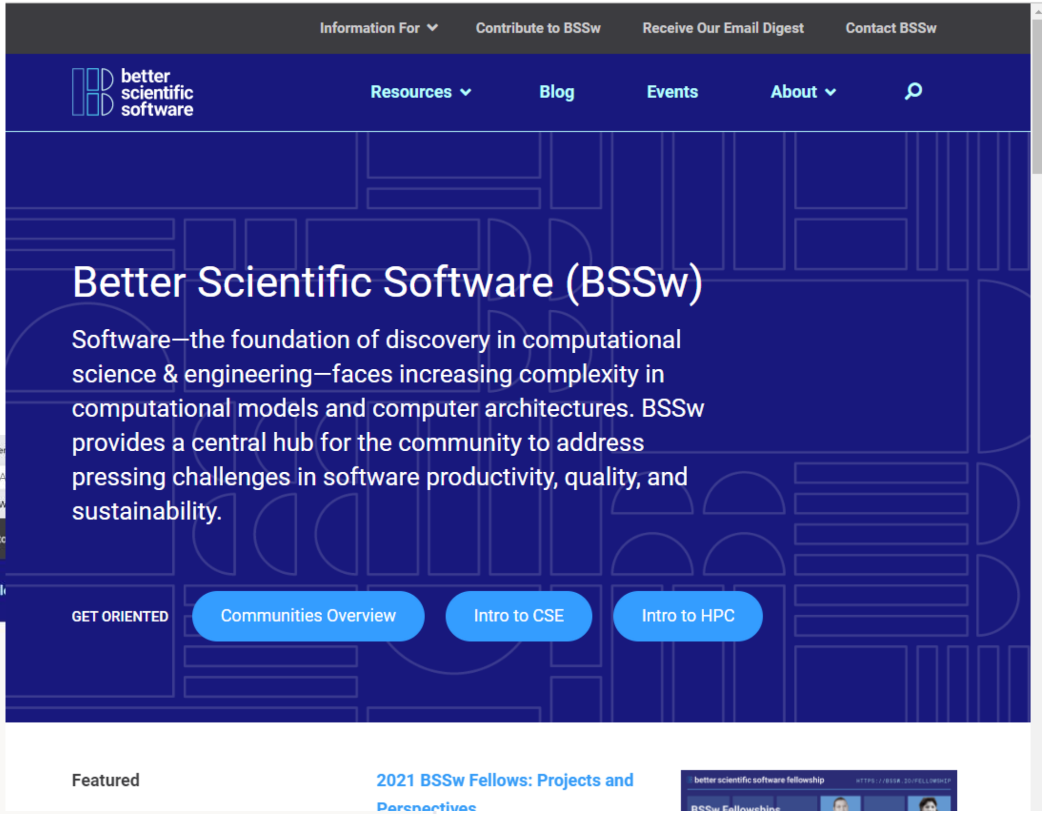
What E4S is not	What E4S is
A closed system taking contributions only from DOE software development teams.	Extensible, open architecture software ecosystem accepting contributions from US and international teams. Framework for collaborative open-source product integration.
A monolithic, take-it-or-leave-it software behemoth.	A full collection if compatible software capabilities **and** A manifest of a la carte selectable software capabilities.
A commercial product.	Vehicle for delivering high-quality reusable software products in collaboration with others.
A simple packaging of existing software.	The conduit for future leading edge HPC software targeting scalable next-generation computing platforms. A hierarchical software framework to enhance (via SDKs) software interoperability and quality expectations.

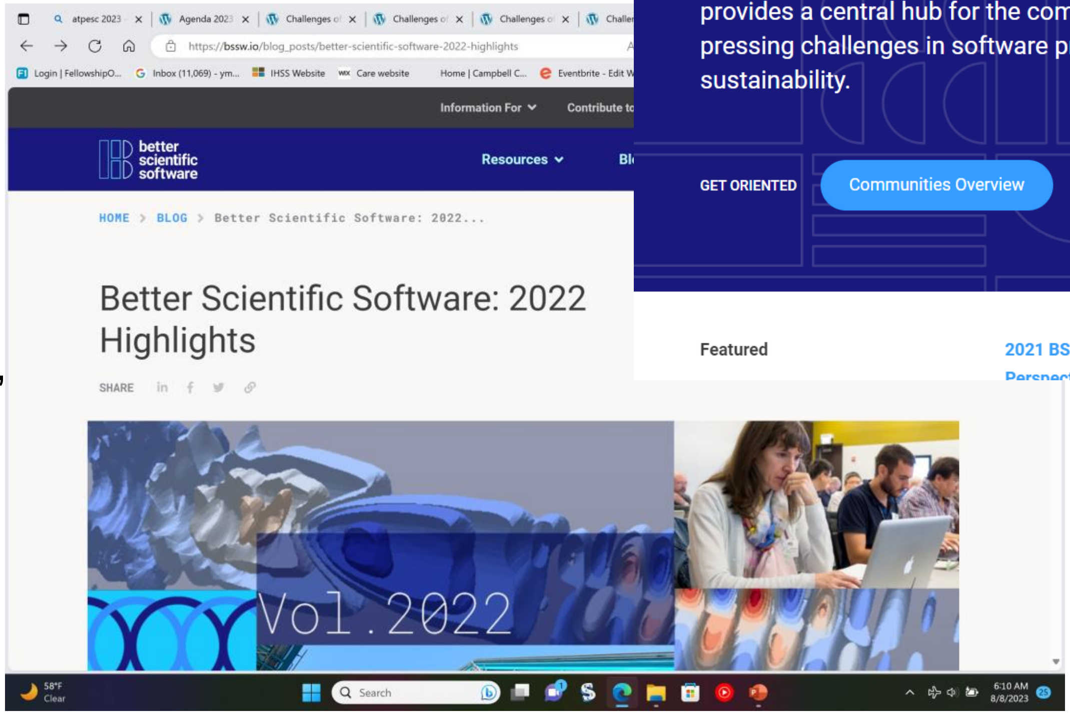
Further reading

better scientific software

https://bssw.io

- Building community through software policies, Piotr Luszczek and Ulrike Yang
- SuperLU: How advances in software practices are increasing sustainability and collaboration, Sherry Li
- Porting the Ginkgo package to AMD's HIP ecosystem, Hartwig Anzt
- Scientific software packages and their communities, Rene Gassmoeller
- Leading a scientific software project: It's all personal, Wolfgang Bangerth
- The art of writing scientific software in an academic environment, Hartwig Anzt
- Working Remotely: The Exascale Computing Project (ECP) panel series, Elaine Raybourn et al.
- Better Scientific Software: 2022 highlights, Rinku Gupta
- And many more …





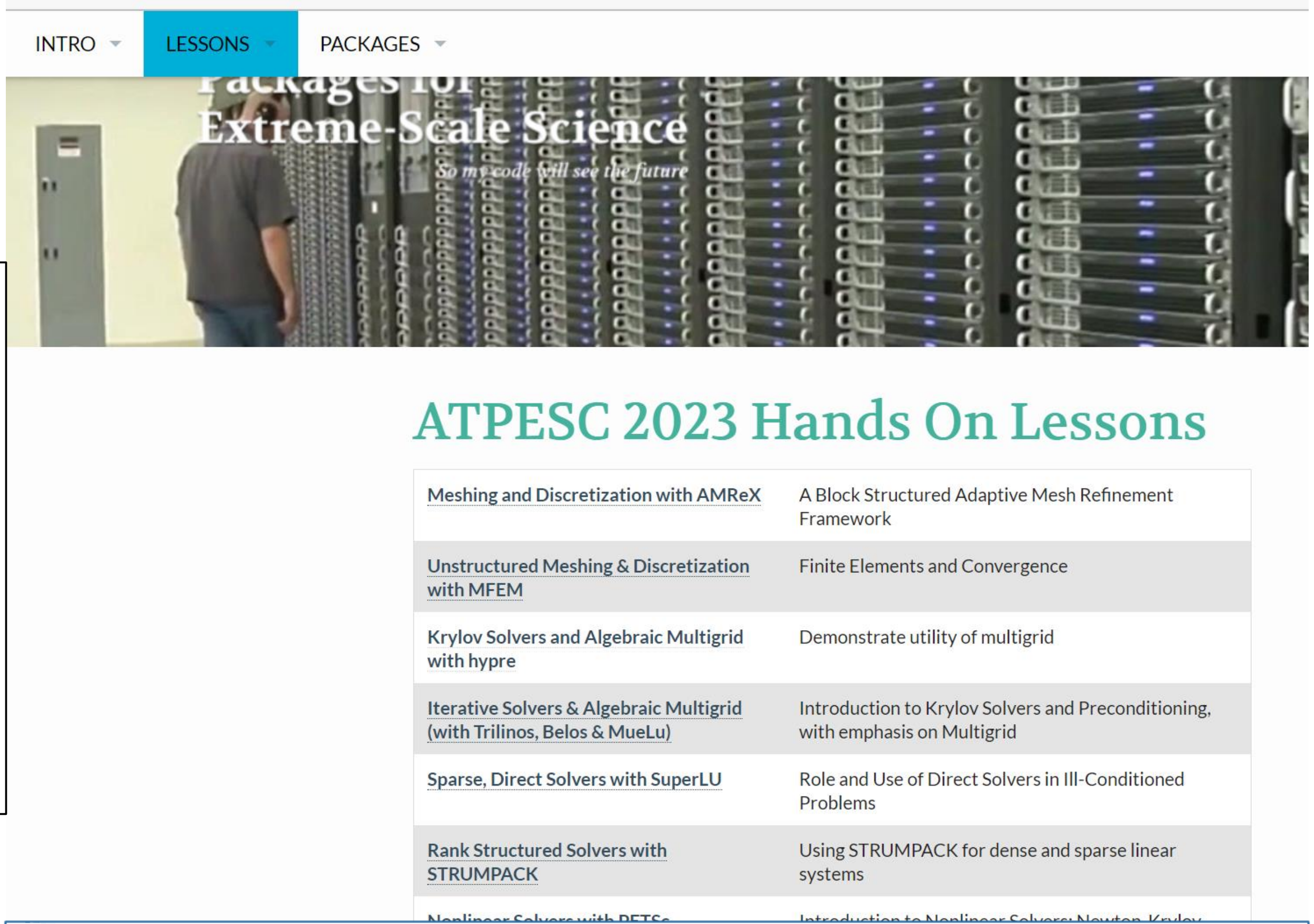
See also Track 3:

Software Productivity & Sustainability (Aug 4)

ECP EXASCALE COMPUTING PROJECT

HandsOn Lessons

- Structured meshing & discretization
- Unstructured meshing & discretization
- Krylov solvers & preconditioners
- Sparse direct solvers
- Nonlinear solvers
- Time integration
- Numerical optimization



And more …

Github pages site:

https://xsdk-project.github.io/MathPackagesTraining2023/lessons/

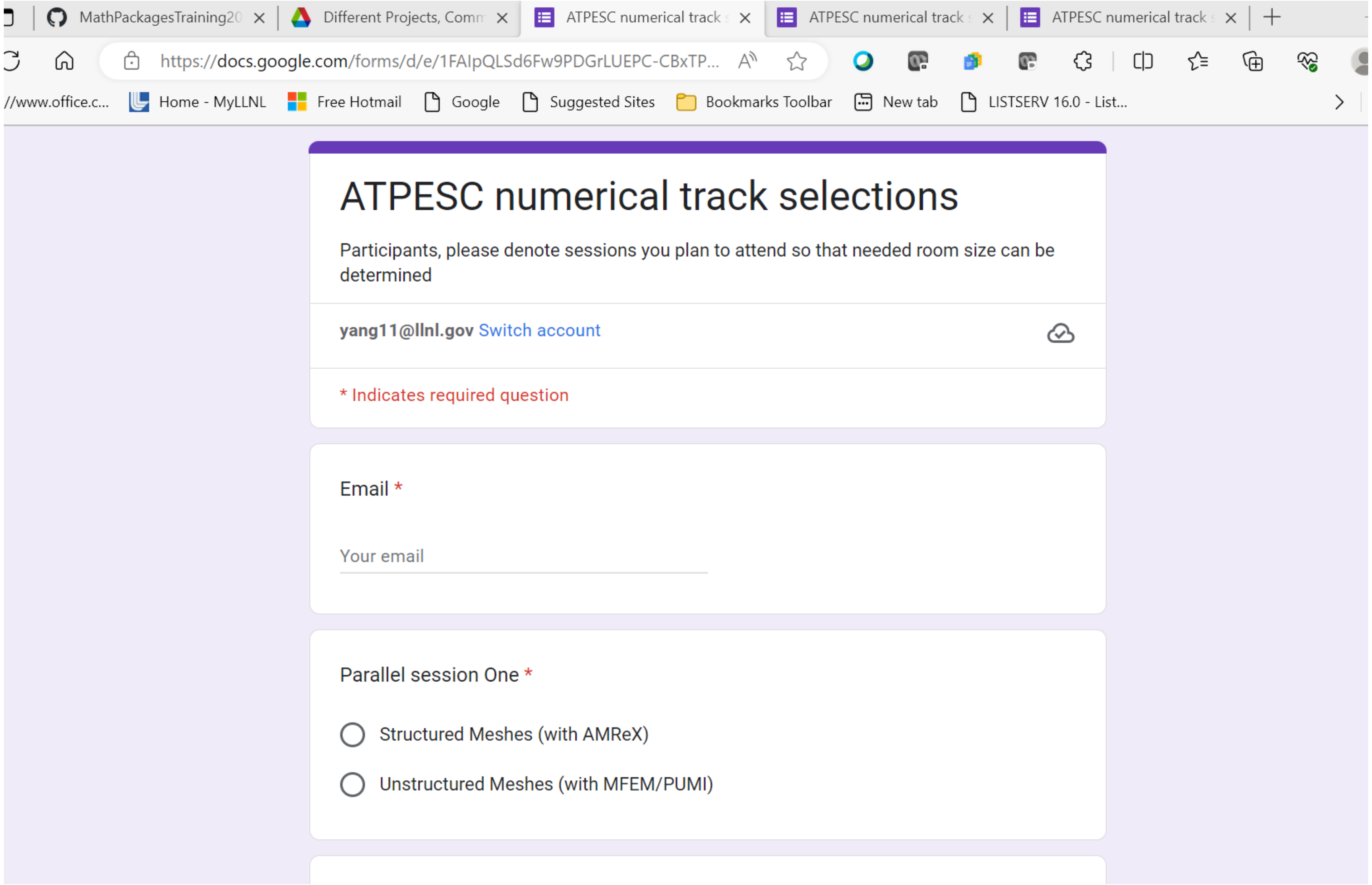
EXASCALE COMPUTING PROJECT

If you haven't yet done so, please choose which session you plan to attend!

Time	Room?	Room?
8:30 – 9:30	Introduction to Numerical Software – Ulrike Yang	
9:30 – 10:45	Structured Discretization (AMReX) – Ann Almgren, Andrew Myers	Unstructured Discretization (MFEM/PUMI) – Tzanio Kolev, Mark Shephard, Cameron Smith
10:45 – 11:15	Break, Subject Matter Expert (SME) Selections, Panel Questions	
11:15 – 12:30	Iterative Solvers & Algebraic Multigrid (hypre) – Sarah Osborn, Ulrike Yang	Direct Solvers (SuperLU, STRUMPACK) – Sherry Li, Pieter Ghysels
12:30 – 1:30	Lunch, SME Selections, Panel Questions	
1:30 – 2:45	Nonlinear Solvers (PETSc) – Richard Mills	Time Integration (SUNDIALS) – David Gardner
2:45 – 3:15	Break, SME Selections, **Panel Questions Due**	
3:15 – 4:30	Optimization (TAO) – Toby Isaac	Iterative Solvers & Algebraic Multigrid (Trilinos/ Belos/MueLU) – Christian Glusa, Graham Harper
4:30 – 5:30	Wrap-up (Ann Almgren) / Panel: Extreme-Scale Numerical Algorithms and Software	
5:30 – 6:30	Unstructured Time: **SME Selections Due**, Informal Discussion, Continue Hands-on	
6:30 – 7:30	Dinner	
7:30 – 9:30	Optional Activity: SME Speed-dating	

Choose which lecture you want to attend!

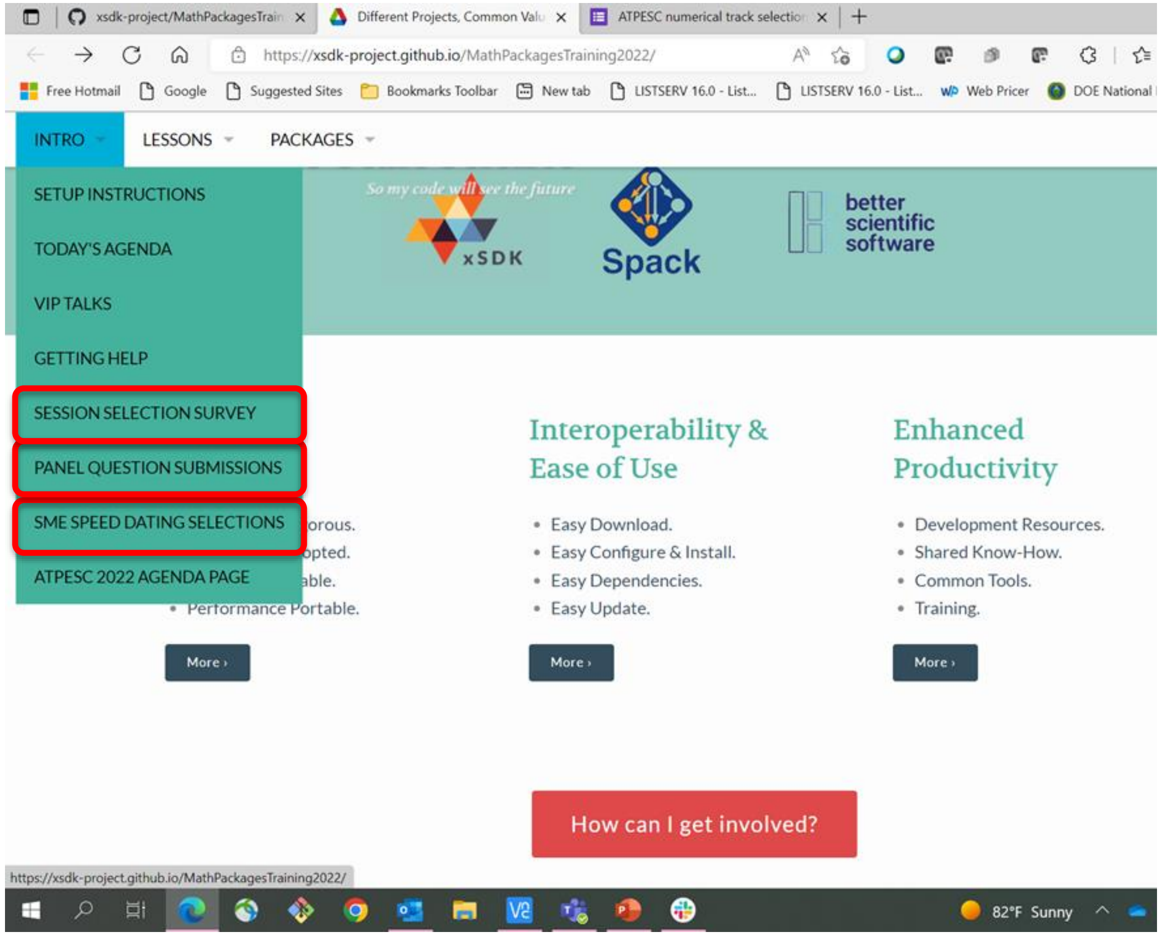
Access form here



Next steps

- **If you haven't done so**
  - **Choose which session you will attend!**

- **During breaks and lunch**
  - **Submit questions for panelists** (optional)
  - **Sign up for discussions with numerical software developers** (optional)
    - **Your email address**
    - Complete by 3:30 pm CDT



EXASCALE COMPUTING PROJECT

Thank you to all ATPESC staff

Special thanks to Ray Loy and Yasaman Ghadar

For their outstanding work in running the 2-week ATPESC program

And thank you to all ATPESC attendees for engaging questions and discussions!

CASC
Center for Applied Scientific Computing

Lawrence Livermore National Laboratory

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC. LLNL-PRES-852746

This work was supported by the U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research (ASCR), Scientific Discovery through Advanced Computing (SciDAC) program, and by the Exascale Computing Project, a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.