ECP Math Libraries:
Capabilities, Applications Engagement

Sherry Li
Lawrence Berkeley National Laboratory

Lois Curfman McInnes
Argonne National Laboratory

And the ECP Math Libraries Community

ECP Community BOF Days
03/31/2021

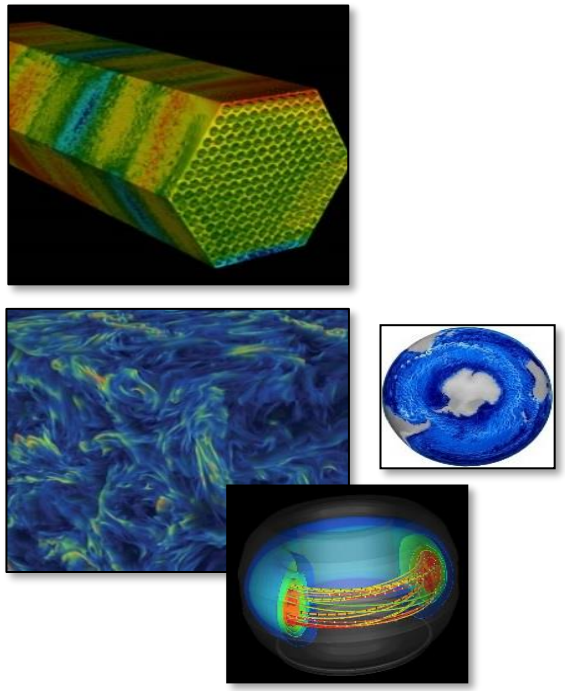
Agenda

- Introduction
- Flash talks of individual math libraries
- Breakout rooms for deep-dive

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

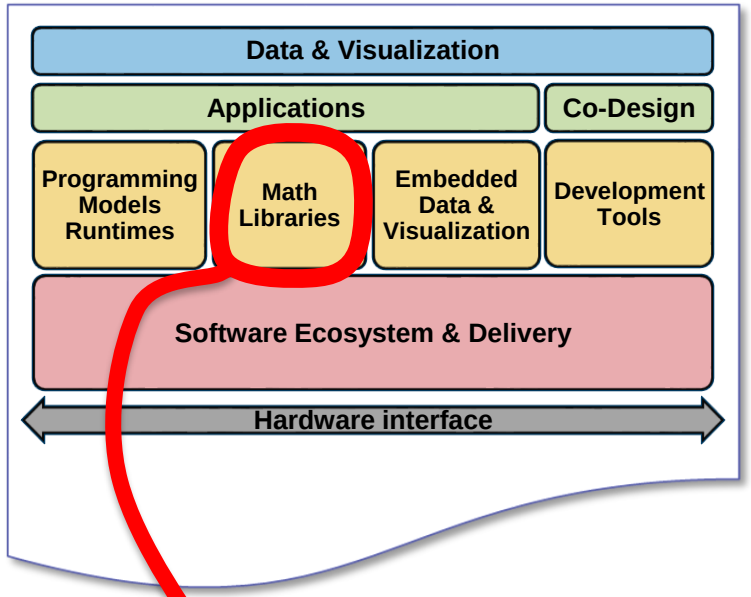
Application Development

Science and mission applications



Software Technology

Scalable software stack



Hardware and Integration

Relationships: facilities with AD/ST, with vendors

Emphasis for this presentation

WBS 2.3.3 Math Libraries: Context for the portfolio

Vision	Provide high-quality, sustainable extreme-scale math libraries that are constantly improved by a robust research and development effort and support exascale needs of the ECP community	
Challenges	Need advances in algorithms and data structures to exploit emerging exascale architectures (high concurrency, limited memory bandwidth, heterogeneity); need new functionality to support predictive simulation and analysis	
Mission	Research, develop, and deliver exascale-ready math libraries to ECP applications	
Objective	Provide scalable, robust, efficient numerical algorithms, encapsulated in libraries that applications can readily use in combination to support next-generation predictive science	
Starting Point	Existing HPC math libraries, used by broad range of ECP applications for the most advanced technologies available in math and computer science R&D	
Portfolio Goals	**Advanced algorithms**	• Advanced, coupled multiphysics and multiscale algorithms (discretizations, preconditioners & Krylov solvers, nonlinear & timestepping solvers, coupling) • Toward predictive simulation & analysis (optimization, sensitivities, UQ, ensembles)
	Performance	• Performance on new node architectures • Extreme strong scalability

	Improving library sustainability & complementarity	• Math library interoperability and complementarity through the xSDK
		  • Improving package usability, quality, sustainability
		  • Community coordination and collaboration while retaining package autonomy

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 …

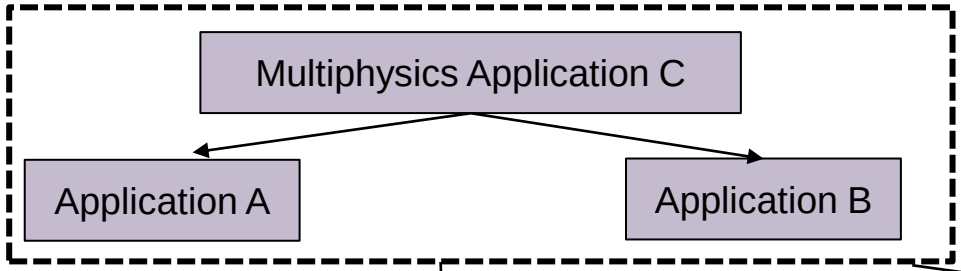
ECP Math Libraries

hypre high performance preconditioners
PETSc
TAO
TRILINOS
STRUMPACK
MFEM
SuperLU
sundials
MAGMA
ARBORX
PLASMA
DTK
ECP SLATE ICL
FFTX
heFFTe
libEnsemble
Ginkgo
TASMANIAN

xSDK Version 0.6.0: November 2020

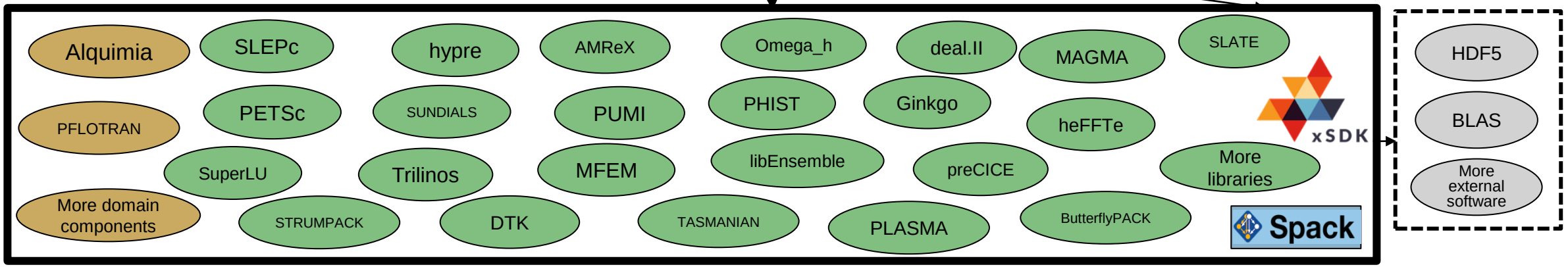
https://xsdk.info

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



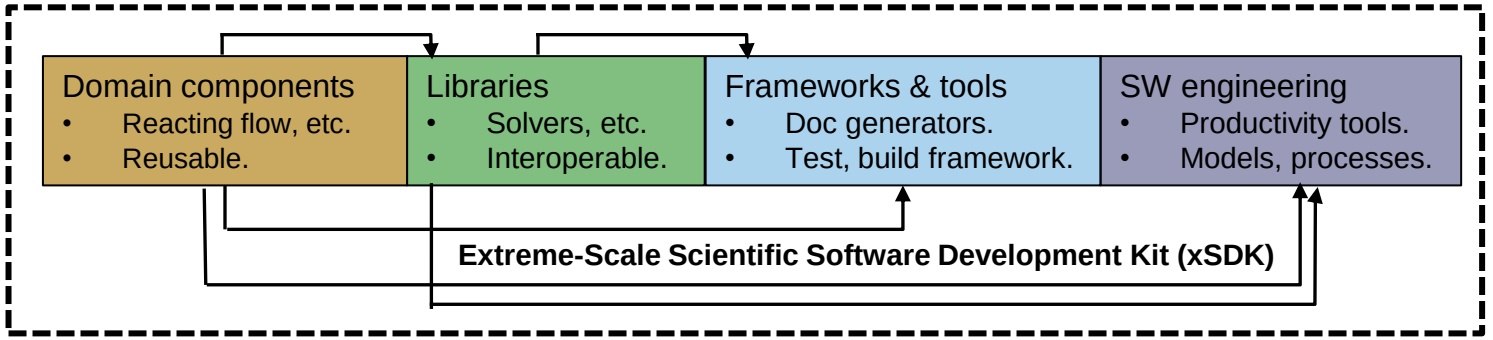
xSDK functionality, Nov 2020

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



November 2020

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

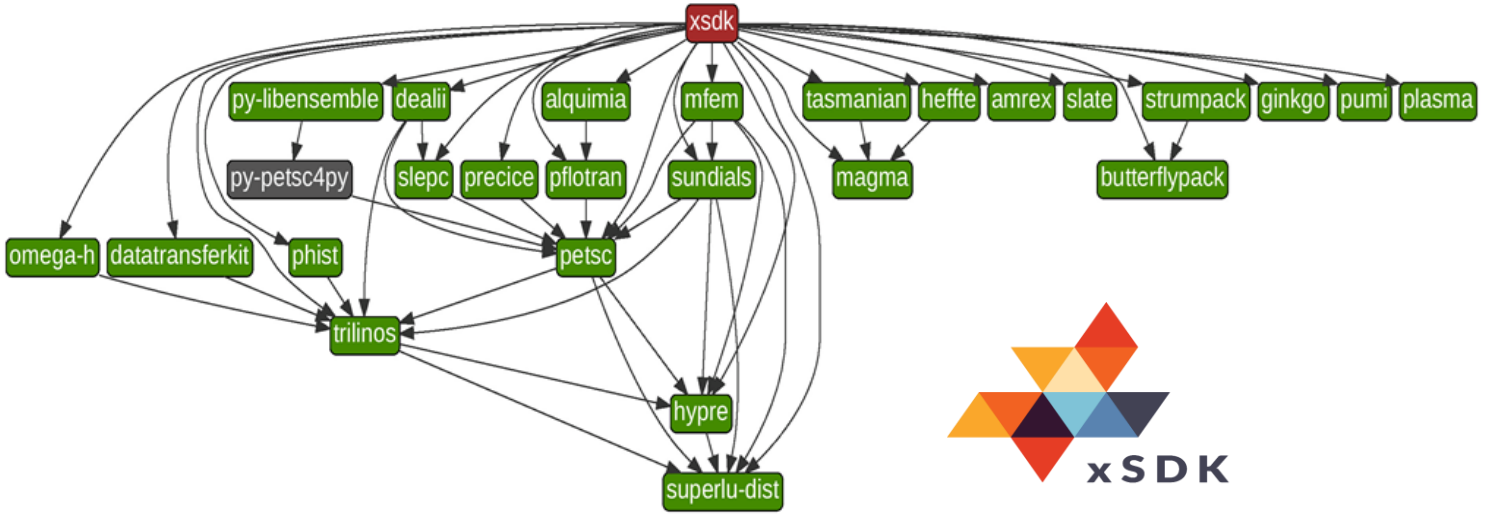


Impact: Improved code quality, usability, access, sustainability

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

Key elements in xSDK

- xSDK Community Policies
https://xsdk.info/policies
- Interoperability
- Spack/Git workflow
  - Installation via spack script
  - CI testing via Gitlab CI infrastructu
  https://gitlab.com/xsdk-project/spack-xsdk
- GPTune autotuner for performance optimization
  https://github.com/gptune/GPTune
- Part of E4S ecosystem: https://e4s.io

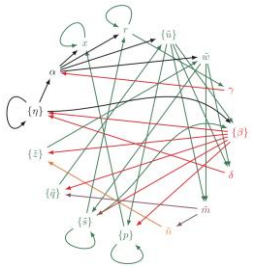


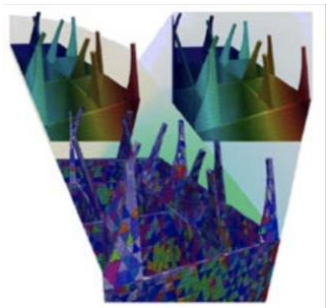
xSDK

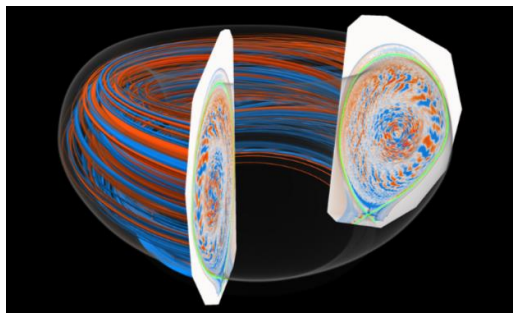
Spack

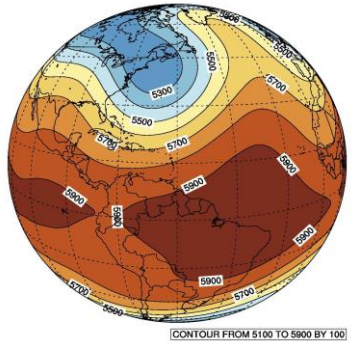
2.3.3 Math Libraries: Projects

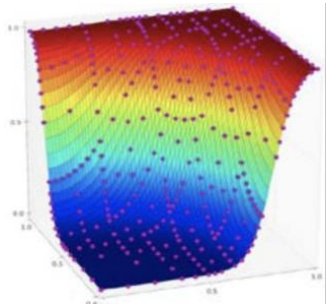
Project Short Name	PI Name, Inst	Short Description/Objective
xSDK	Ulrike Meier Yang, LLNL	xSDK (Extreme-scale Scientific Software Development Kit): community policy-based approach to value-added aggregation of independently developed math libraries (increasing quality, combined usability, interoperability)
PETSc / TAO	Todd Munson, ANL	PETSc (scalable linear & nonlinear solvers, integrators), TAO (numerical optimization), libEnsemble (ensemble management for exascale platforms)
STRUMPACK / SuperLU / FFTX	Xiaoye Li, LBNL	STRUMPACK & SuperLU (scalable sparse direct solvers, preconditioners), FFTX (FFT stack, including symbolic analysis and code generation)
SUNDIALS / hypre	Carol Woodward, LLNL	SUNDIALS (adaptive time integrators, nonlinear solvers), hypre (scalable linear solvers, with emphasis on algebraic multigrid)
CLOVER	Jack Dongarra, UTK	SLATE (exascale-capable dense linear algebra), FFT-ECP (scalable FFTs), Ginkgo (preconditioned iterative solvers), MAGMA-sparse
ALExa / ForTrilinos	John Turner, ORNL	DTK (parallel data transfer between grids, search tree capability), Tasmanian (uncertainty quantification, surrogate modeling), ForTrilinos (automatic generation of Fortran interfaces for Trilinos)
Sake	Siva Rajamanickam, SNL	Trilinos Solvers Stack, KokkosKernels (portable performance kernels for linear algebra and graph algorithms)











ECP Early Access Systems status among the ECP math libraries

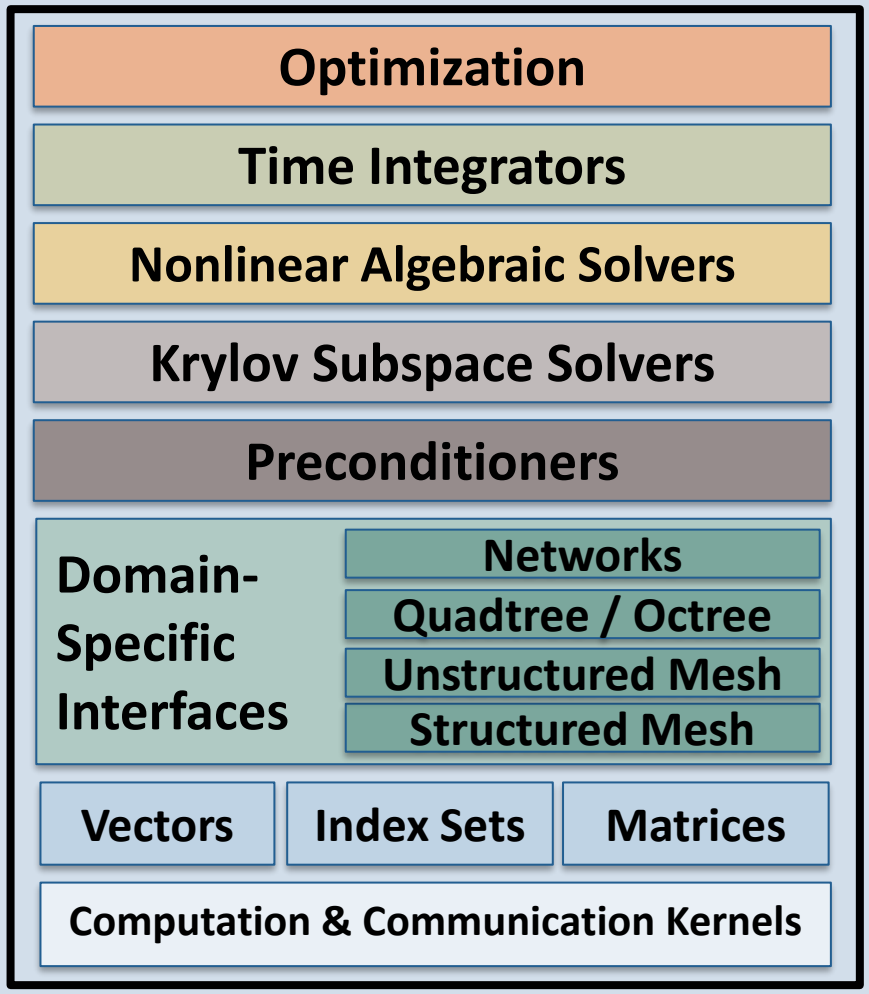
Package	On AMD GPU	On Intel GPU	Installation method	E4S Spack ready for EAS
ArborX	Yes	Yes	Cmake	No
DTK	No	No	Cmake	No
ForTrilinos	No	No	Cmake	No
Ginkgo	yes	yes	Cmake	Tulip HIP; not Intel DPC++
heFFTe	Yes	Yes	Cmake	No
hypre	”Redwood” Nvidia GPU, not yet AMD GPU	No	Autoconf	No
KokkosKernels	Yes	Starting	Cmake	No
libEnsemble	Yes (GPU not applicable)	Yes (GPU not applicable)	Pip, conda, spack	Not applicable
MAGMA	Yes	Yes	Makefiles	No
MFEM	Yes	Yes	Cmake, makefiles	Iris/Yarrow; Tulip in progress
PETSc/TAO	Yes	Yes	Own build system	In progress
PLASMA	Yes	Yes	Cmake	Yes
SLATE	In progress	No	Cmake, makefiles	No
STRUMPACK	yes	No	Cmake	Tulip ROCm
Sundials	Yes	Yes	Cmake	Tulip soon; Yarrow later with SYCL
SuperLU	Yes	No	Cmake	No
SWIG	No	No	Autoconf	No
Tasmanian	Yes	Yes	Cmake	No
Trilinos	Yes (depends on KokkosKernels)	No	Cmake	No

Flash talks of individual packages

PETSc TAO

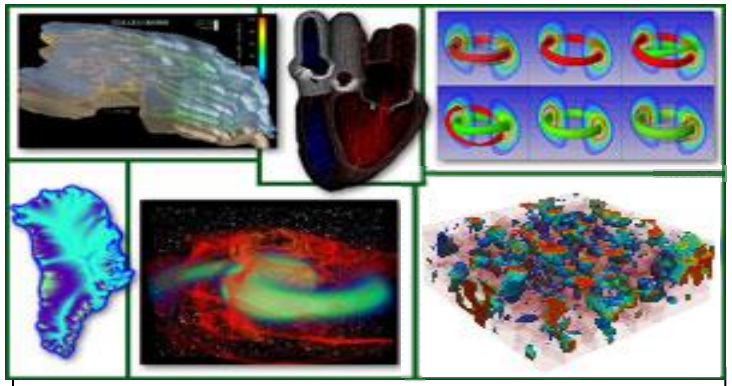
Portable, Extensible Toolkit for Scientific Computation / Toolkit for Advanced Optimization

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



- **Easy customization and composability of solvers at runtime**
  - 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

ECP EXASCALE COMPUTING PROJECT

STRUMPACK

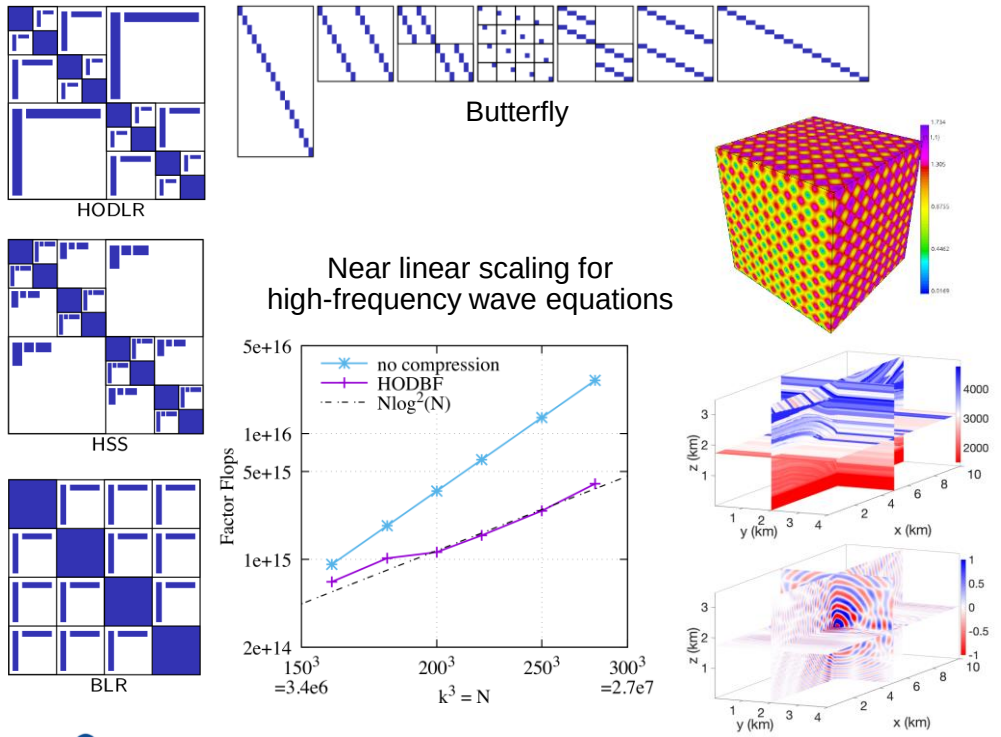
Structured Matrix Package

STRUMPACK

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

BERKELEY LAB

- **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



ECP xSDK FASTMATH

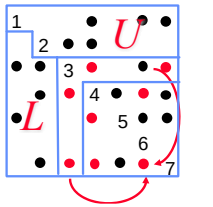
github.com/pghysels/STRUMPACK

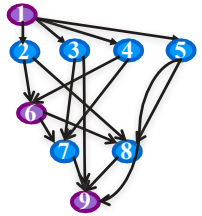
EXASCALE COMPUTING PROJECT

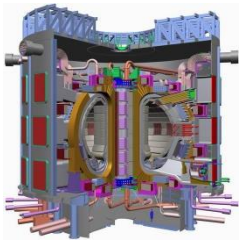
SuperLU

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

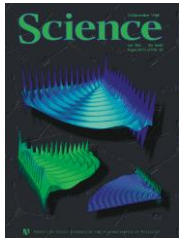
- **Capabilities**
  - Serial (thread-safe), shared-memory (SuperLU_MT, OpenMP or Pthreads), distributed-memory (SuperLU_DIST, hybrid MPI+ OpenM + CUDA/HIP).
    - Written in C, with Fortran interface
  - Sparse LU decomposition (can be nonsymmetric sparsity pattern), triangular solution with multiple 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 (DPC++ planned)
- **Parallel Scalability**
  - Factorization strong scales to 32,000 cores (IPDPS'18, PARCO'19)
  - Triangular solve strong scales to 4000 cores (SIAM CSC'18, SIAM PP'20)
- **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/

EXASCALE COMPUTING PROJECT

FFTX

Next-Generation Fast Fourier Transforms for GPUs. C++ code generation using Spiral analysis tools. Performance portable on CPUs and GPUs. BSD license.

- **Goals**
  - Performance portable, open-source FFT software system for modern heterogeneous architectures (i.e. GPUs) to provide a capability analogous to FFTW.
  - Support applications-specific optimizations corresponding to integrating more of the algorithms into the analysis / code generation process.
- **Approach**
  - Code generation based on Spiral, an analysis and code generation tool chain for discrete Fourier Transforms and tensor algebra algorithms
  - FFTX user API implemented in standard C++.
  - Factored design that allows FFTX / Spiral to be more easily ported across multiple GPU platforms.
- **Capabilities**
  - Complete FFTX C++ API for single-processor / single device. Automated invocation of Spiral to generate code and header file.
  - CPU, cuda code generation. Examples include forward / inverse FFTs (c-to-c and r-to-c); periodic and free-space convolutions.
- **Planned Work**
  - ExaScale platforms: AMD GPU (HIP) (6/2021), Intel GPU (SYCL) (2022).
  - Distributed memory: native FFTX API (7/2021), extensions of single-device API to support other distributed frameworks, e.g. heFFTe (10/2021)
- **Performance**
  - Within 2x of vendor FFTs, with more complete coverage of the algorithm space.
  - Demand-driven performance engineering of specific use cases, higher-level algorithms.

```
const int nx=80;
const int ny=80;
const int nz=80;

box_t<3> domain(point_t<3>({{1,1,1}}), point_t<3>({{nx,ny,nz}}));

array_t<3,std::complex<double>> inputs(domain);
array_t<3,std::complex<double>> outputs(domain);
std::array<array_t<3,std::complex<double>>,1> intermediates {domain};

setInputs(inputs);
setOutputs(outputs);

openScalarDAG();

MDDFT(domain.extents(), 1, intermediates[0], inputs);
IMDDFT(domain.extents(), 1, outputs, intermediates[0]);

closeScalarDAG(intermediates, "fimddft");
```

```
...
if (((threadIdx.x == 1))) {
            for(int i44 = 0; i44 <= 1; i44++) {
                double a300, a301, a302, a303, a304, a305, a306, a307,
                       s111, s112, s113, s114, s115, s116, s117, s118,
                       s119, s120, s121, s122, s123, s124, s125, s126,
                       t333, t334, t335, t336, t337, t338, t339, t340;
                int a298, a299, a308, b62, b63, b64;
                a298 = (2*i44);
                b62 = ((24*threadIdx.y) + a298);
                s111 = T7[((((b62 + 2) + (96*threadIdx.z)) + (3840*blockIdx.x)) + (307200*blockIdx.y)) +
(4915200*blockIdx.z)];
                s112 = T7[((((b62 + 3) + (96*threadIdx.z)) + (3840*blockIdx.x)) + (307200*blockIdx.y)) +
(4915200*blockIdx.z)];
                s113 = T7[((((b62 + 14) + (96*threadIdx.z)) + (3840*blockIdx.x)) + (307200*blockIdx.y)) +
(4915200*blockIdx.z)];
                s114 = T7[((((b62 + 15) + (96*threadIdx.z)) + (3840*blockIdx.x)) + (307200*blockIdx.y)) +
(4915200*blockIdx.z)];
                a299 = (8*i44);
                a300 = D16[a299];
                a301 = D16[(a299 + 1)];
                s115 = ((a300*s111) - (a301*s112));
                s116 = ((a301*s111) + (a300*s112));
                a302 = D16[(a299 + 2)];
                a303 = D16[(a299 + 3)];
                s117 = ((a302*s113) - (a303*s114));
                s118 = ((a303*s113) + (a302*s114));
                t333 = (s115 + s117);
                t334 = (s116 + s118);
                t335 = (s115 - s117);
                t336 = (s116 - s118);
```

Carnegie Mellon University

https://github.com/spiral-software/{spiral-software,fftx

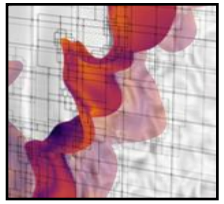
SUNDIALS

Suite of Nonlinear and Differential /Algebraic Equation Solvers

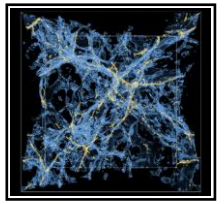
Adaptive time integrators for ODEs and DAEs and efficient nonlinear solvers
Used in a variety of applications. Freely available. Encapsulated solvers & parallelism.

- **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, iterative methods (GMRES, PCG, etc.)
- **Future GPU Features:** Ginkgo linear solver interface, Kokkos vector module
- **Open Source:** Available via LLNL site, GitHub, and Spack; BSD License; Supported by extensive documentation, a user email list, and an active user community

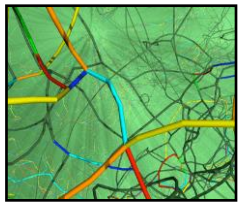
SUNDIALS is used worldwide in applications throughout research and industry



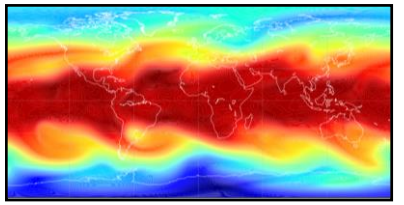
Combustion (Pele)



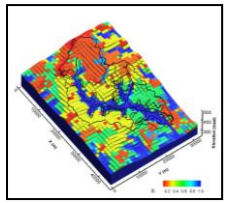
Cosmology (Nyx)



Dislocation dynamics (ParaDiS)



Atmospheric Dynamics (Tempest)



Subsurface flow (ParFlow)

Lawrence Livermore National Laboratory

ECP

FASTMATH

xSDK

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

ECP EXASCALE COMPUTING PROJECT

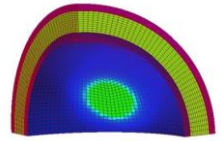
hypre

Lawrence Livermore National Laboratory

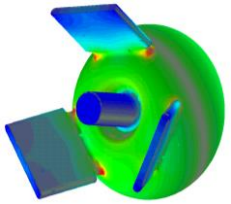
Highly scalable multilevel solvers and preconditioners. Unique user-friendly interfaces. Flexible software design. Used in a variety of applications. Freely available.

- **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, MGR
  - Matrix-free Krylov solvers
- **Exascale early systems GPU-readiness**
  - Available: Nvidia GPU (CUDA)
  - In progress: AMD GPU (HIP), Intel GPU (DPC++ planned)
- **Open-source software**
  - Used worldwide in a vast range of applications
  - Can be used through PETSc and Trilinos
  - Provide CPU and GPU support
  - Available on github: https://www.github.com/hypre-space/hypre

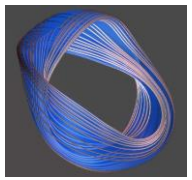
hypre high performance preconditioners



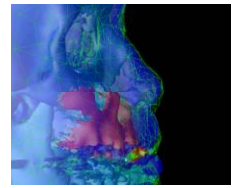
Elasticity / Plasticity



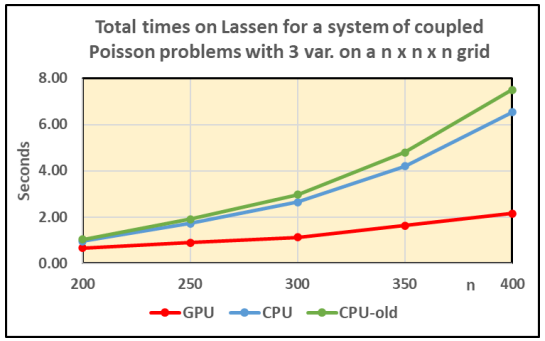
Electro-magnetics



Magneto-hydrodynamics



Facial surgery



xSDK

FASTMATH

ECP

R&D 100

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

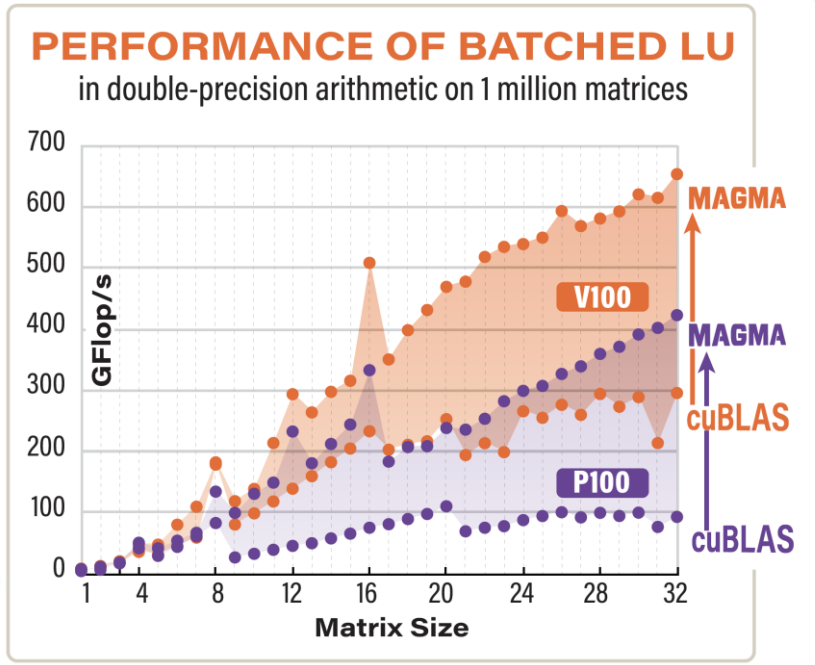
PROJECT

INNOVATIVE COMPUTING LABORATORY

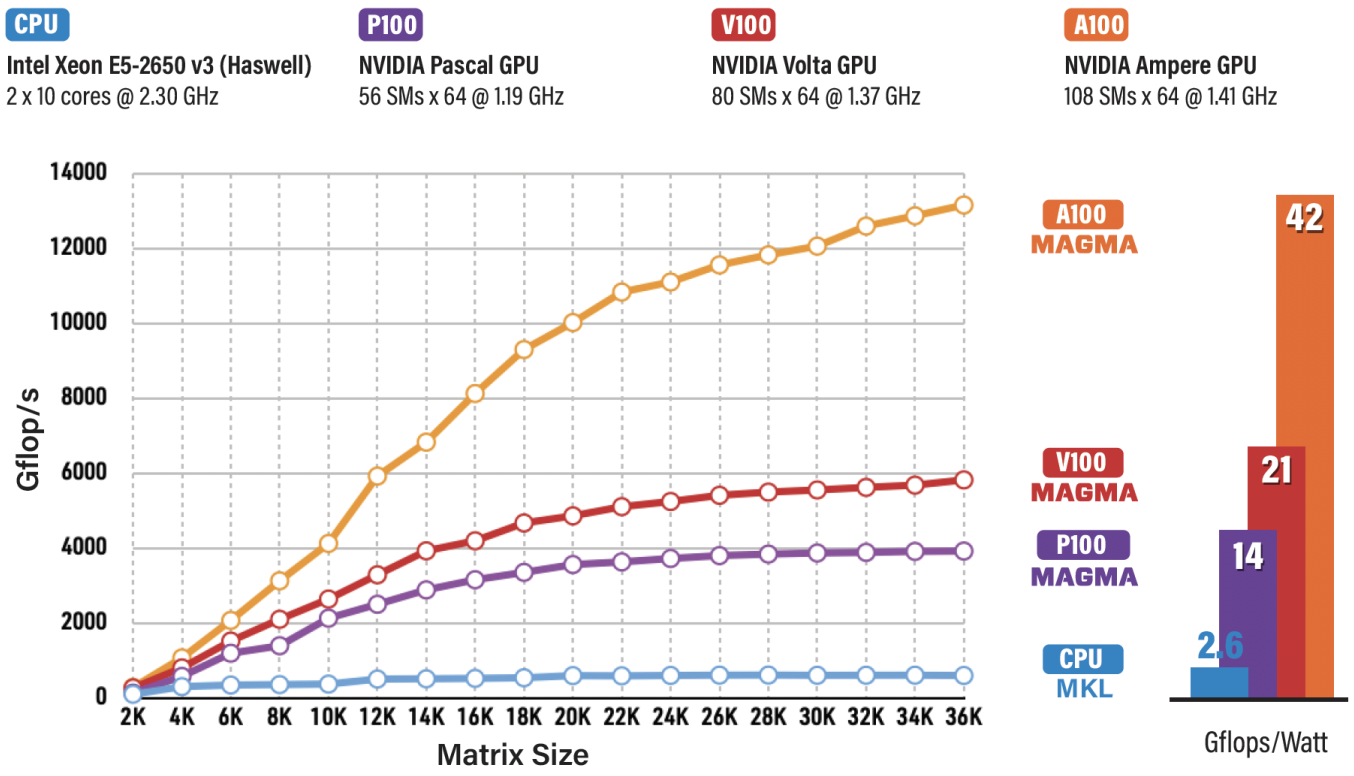
MAGMA

Matrix Algebra on GPU and Multicore

- Shared memory systems
- BLAS/LAPACK on GPUs
- Hybrid CPU-GPU Algorithms
  - Linear system solvers (+ mixed precision)
  - Eigenvalue problem solvers
- Batched LA
  - All BLAS-3 (fixed/variable), LU, QR, Cholesky
- Sparse LA
  - Solvers: BiCG, BiCGSTAB, GMRES
  - Preconditioners: ILU, Jacobi,
  - SPMV, SPMM (CSR, ELL, … etc.)



PERFORMANCE & ENERGY EFFICIENCY



- Support CUDA and ROCM/HIP
- https://icl.utk.edu/magma

IN COLLABORATION WITH SPONSORED BY WITH SUPPORT FROM

Berkeley · University of Colorado Denver · U.S. Department of Defense · U.S. Department of Energy · AMD · intel · Inria · KAUST · ECP · National Science Foundation · MathWorks · NVIDIA

SLATE

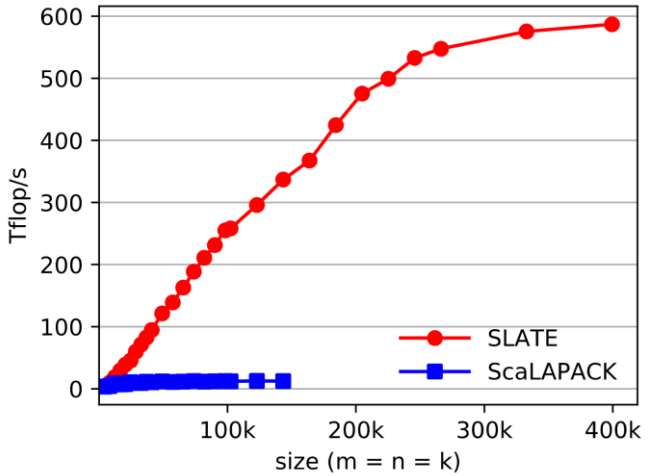
Software for Linear Algebra Targeting Exascale

THE UNIVERSITY OF TENNESSEE KNOXVILLE

Distributed, GPU-accelerated, dense linear algebra library.
Modern replacement for ScaLAPACK. BSD license.

- **Made for distributed HPC with accelerators**
  - BLAS: matrix multiply (C = AB), etc.
  - Linear systems (Ax = b): LU, Cholesky, symmetric indefinite
  - Least squares (Ax ≈ b): QR, LQ
  - Eigenvalue (Ax = λx)
  - SVD (A = UΣVH)
- **GPU-readiness: Uses BLAS++ as abstraction layer**
  - Initial implementation: Nvidia GPUs (cuBLAS)
  - Recent: AMD GPU (hip/rocBLAS).
  - In progress Intel GPUs (OpenMP, oneAPI).
- **Software design**
  - C++ library built on MPI, OpenMP, batch-BLAS, vendor-BLAS.
  - Build: CMake, Makefile, Spack. APIs: C, Fortran, ScaLAPACK.
- **BLAS++ and LAPACK++**
  - C++ wrappers for BLAS and LAPACK routines. Independent projects.

Summit 16 nodes: 672 POWER9 cores+96 NVIDIA V100
CPU + GPU **peak 765 Tflop/s** in double precision



Matrix multiply (dgemm)
47x speedup
77% of peak

https://icl.utk.edu/slate/

EXASCALE COMPUTING PROJECT

PLASMA: Parallel Linear Algebra for Multicore Architectures

Functional scope

- Dense: linear systems, least-squares, EIG/SVD
- Tile matrix layout and tile algorithms
- OpenMP: v4 tasking, v4.5 priorities, v5 offload variants

Compiler framework targets: Clang 11, AOMP 11, XL 16, OneAPI 1, Cray 9, NVHPC

```
double*a_dev=omp_target_alloc(device, a_ld * n);
```

Device-resident pointers for persistent on-device storage

```
int dev = omp_get_default_device(); double *a = mkl_malloc(a_ld * n * 8, 64);
#pragma omp target data map(to:a,b) map(tofrom:c)
{
#pragma omp target variant dispatch use_device_ptr(a,b,c) device(dev) nowait
mkl_dgemm(tA, tB, m, n, k, alpha, a, a_ld, b, b_ld, beta, c, c_ld);
#pragma omp taskwait
}
```

Accessing native libraries for vendor-level on-device performance

Accessing device-specific asynchronous dispatch for low-level runtime integration

```
#pragma omp target data map(a[0:n*n],b[0:n*n]) map(alloc:c[0:n*n])
#pragma omp target data use_device_ptr(a,b,c)
{
  cudaStream_t omp_stream = (cudaStream_t) omp_get_cuda_stream(dev);
  cublasSetStream(handle, stream);
  cublasDgemm(handle, CUBLAS_OP_N, CUBLAS_OP_N, m, n, k, &alpha, a, a_ld, b, b_ld, &beta, c, c_ld);
}
```

Ginkgo

THE UNIVERSITY OF TENNESSEE KNOXVILLE

KIT Karlsruhe Institute of Technology

UJI

GPU-centric high performance sparse linear algebra. Sustainable and extensible C++ ecosystem with full support for AMD, NVIDIA, Intel GPUs.

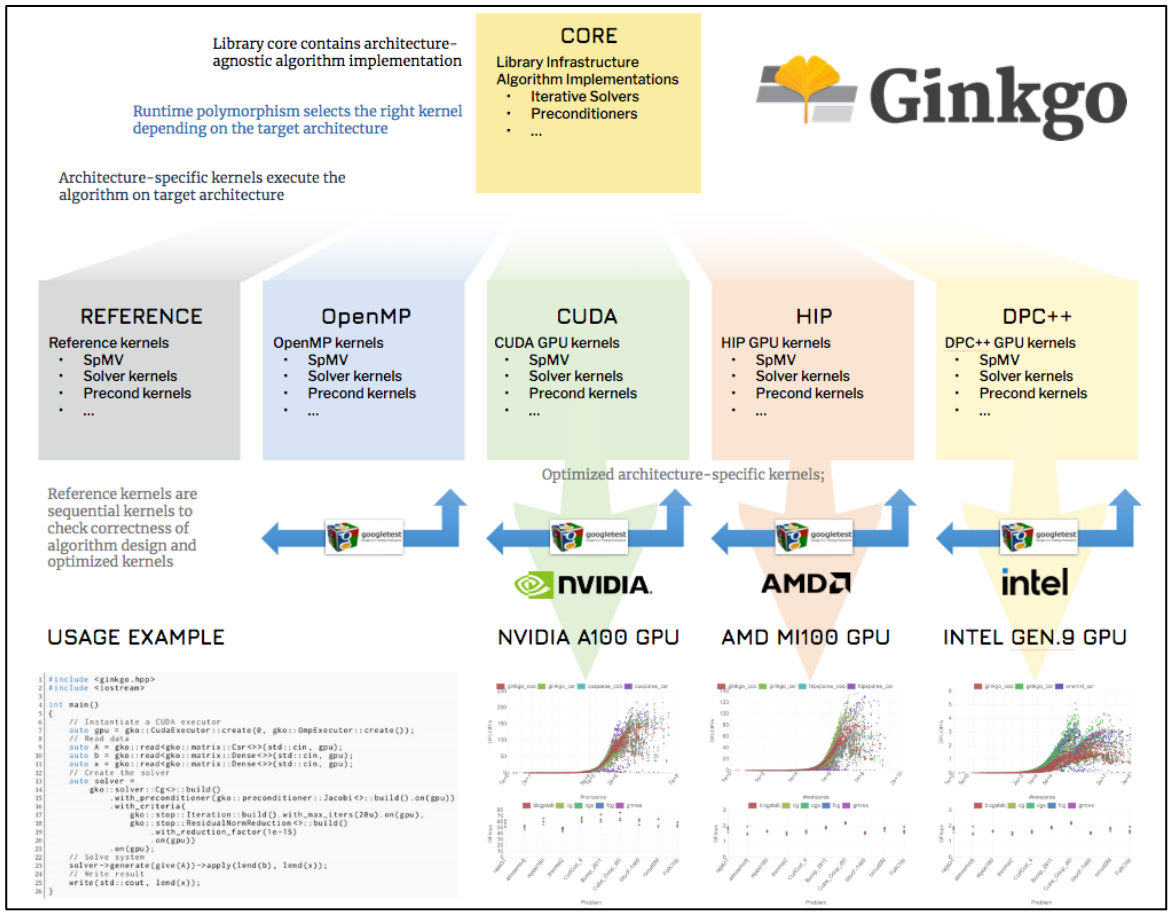
- **High performance sparse linear algebra**
  - Linear solvers: BiCG, BiCGSTAB, CG, CGS, FCG, GMRES, IDR;
  - Advanced preconditioning techniques: ParILU, ParILUT, SAI;
  - Mixed precision algorithms: adaptive precision Jacobi, FSPAI;
  - Decoupling of arithmetic precision and memory precision;
  - *Batched iterative solvers;*
  - Linear algebra building blocks: SpMV, SpGEAM,...;
  - Extensible, sustainable, production-ready;
- **Exascale early systems GPU-readiness**
  - Available: Nvidia GPU (CUDA), AMD GPU (HIP), Intel GPU (DPC++), CPU Multithreading (OpenMP);
  - C++, CMake build;
- **Open source, community-driven**
  - Freely available (BSD License), GitHub, and Spack;
  - Part of the **xSDK and E4S software stack**;
  - Can be used from **deal.II** and **MFEM**;

ECP

E4S

xSDK

deal.II



https://ginkgo-project.github.io/

ECP EXASCALE COMPUTING PROJECT

heFFTe

Highly Efficient FFT for Exascale (heFFTe). Scalable, high-performance multidimensional FFTs; Flexible; User-friendly interfaces (C++/C/Fortran/python); Examples & benchmarks; Testing; Modified BSD license.

Capabilities:

- Multidimensional FFTs
- C2C, R2C, C2R
- Support flexible user data layouts
- Leverage and build on existing **FFT capabilities**

Pre-exascale environment:

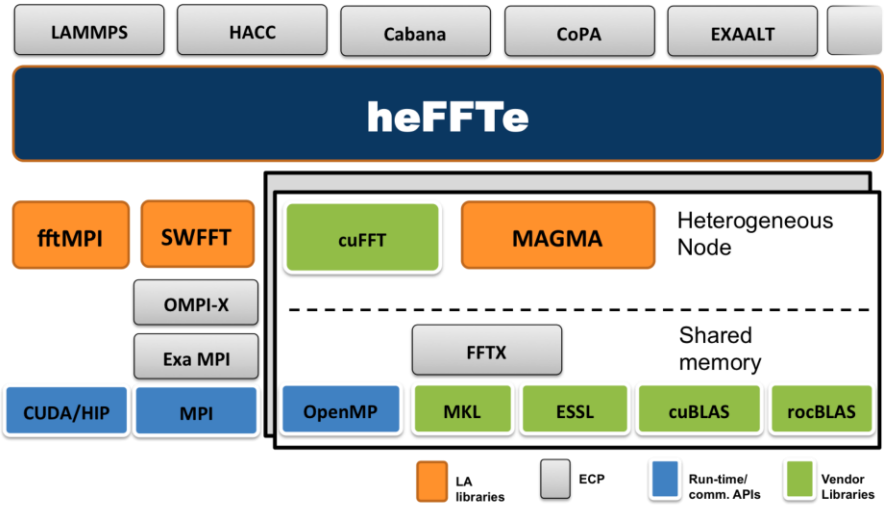
- **Summit @ OLCF (Nvidia GPUs), Poplar (AMD GPUs), and others**
- **In progress: Intel GPU**

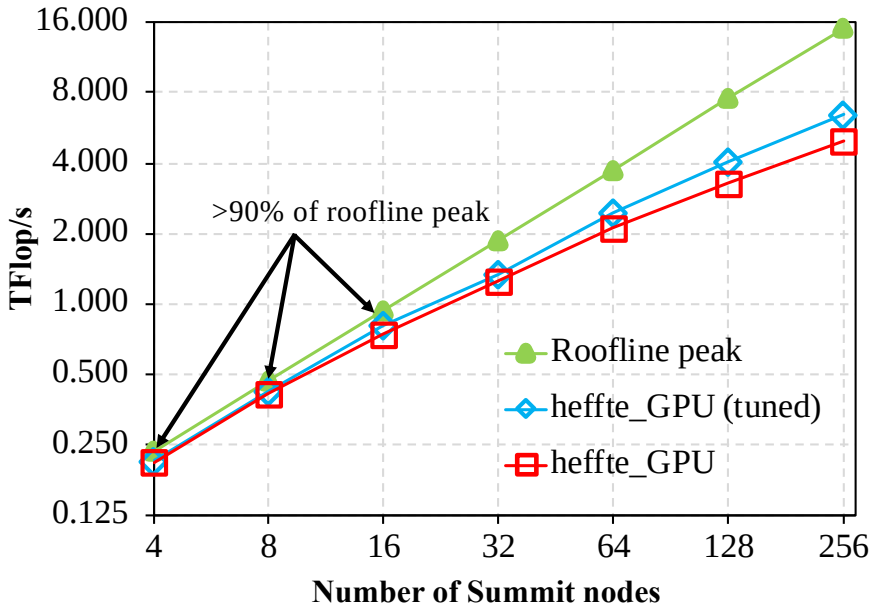
Current status:

- heFFTe 2.0 with support for CPUs, Nvidia GPUs, AMD GPUs
- Very good strong and weak scaling, reaching up to 90% of roofline peak

Open Source Software

- **spack** installation and integration in xSDK
- heFFTe Integration and acceleration of CoPA projects using LAMMPS and HACC
- Homepage: http://icl.utk.edu/fft/
  Repository: https://bitbucket.org/icl/heffte/



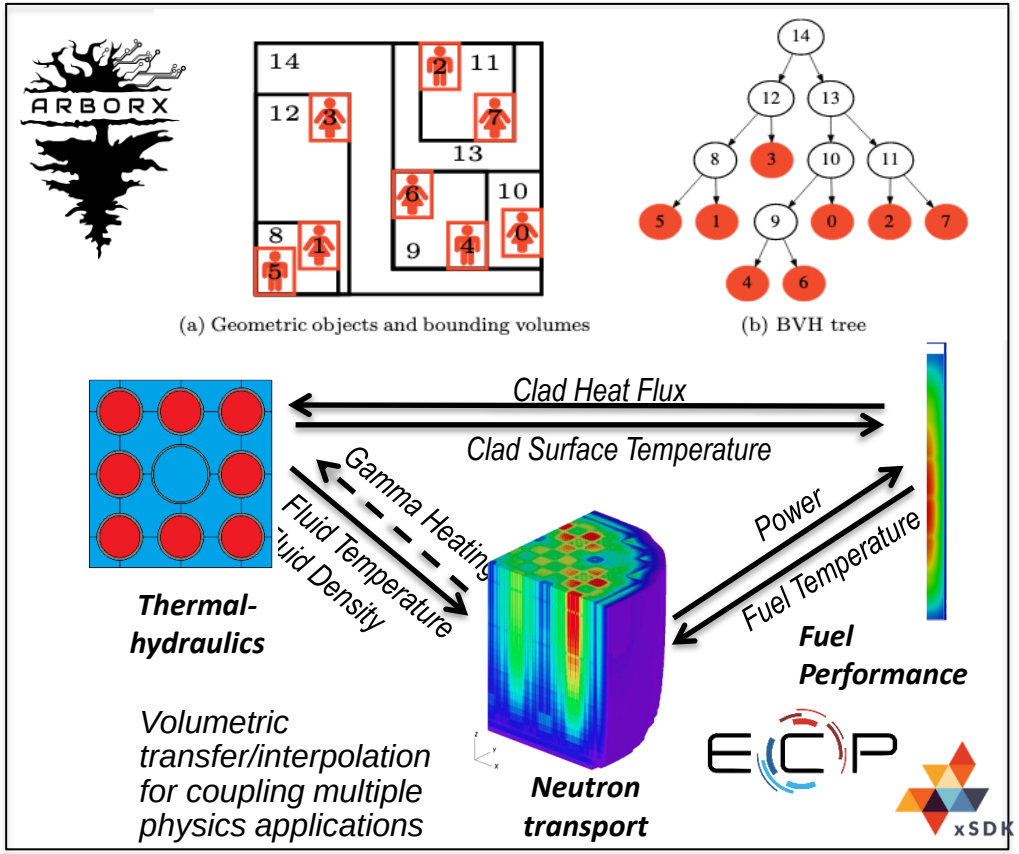


ArborX/ DataTransferKit

OAK RIDGE National Laboratory

Open source libraries for geometric search and parallel solution transfer. Support for grid-based and mesh-free applications.

- **ArborX**
  - Geometric search and clustering algorithms
    - Provides both neighborhood search (rNN) and nearest neighbors (kNN)
    - Provides density-based clustering algorithms (DBSCAN, HDBSCAN)
  - Performance portable
    - Serial performance is comparable to widely used libraries (Boost R-tree, Nanoflann)
    - Supports all DOE leadership class machines
  - Used for Kokkos performance benchmarking
    - The first libraries to support all Kokkos backends (OpenMP, CUDA, HIP, SYCL, OpenMPTarget)
- **DataTransferKit**
  - Efficient and accurate solution transfers between applications of different mesh layouts on parallel accelerated architectures
  - Used for a variety of applications including conjugate heat transfer, fluid structure interaction, computational mechanics, and reactor analysis



https://github.com/arborx/ArborX
https://github.com/ORNL-CEES/DataTransferKit

ECP EXASCALE COMPUTING PROJECT

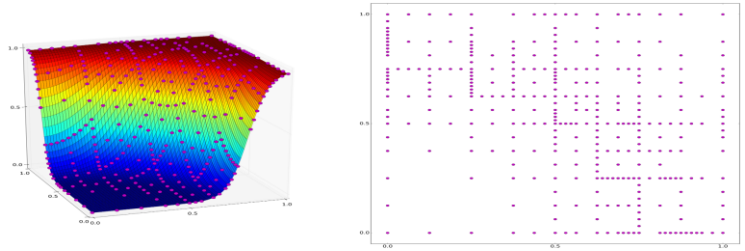
Tasmanian

OAK RIDGE National Laboratory

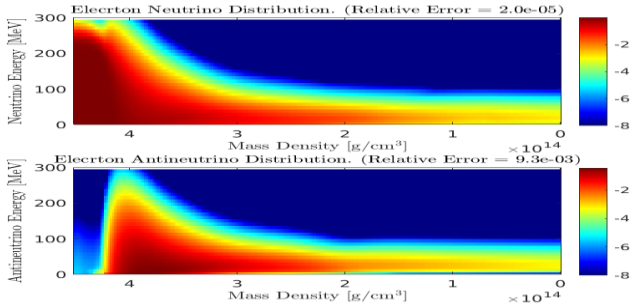
Toolkit for Adaptive Stochastic Modeling and Non-Intrusive ApproximatioN. Open Source Library for Uncertainty Quantification, surrogate modeling, data compression, Bayesian inference, and optimization.

- **Capabilities**
  - Sparse Grid Surrogate modeling using structured and unstructured data
    - Statistical analysis
    - Fast surrogates for multiphysics simulations
  - Hierarchical data representation for data-reduction and data-mining
  - Markov Chain Monte Carlo methods for Bayesian inference
    - Model calibration and validation
    - Sensitivity analysis and multidimensional anaisotropy
- **GPU Accelerated Capabilities**
  - Fast surrogates using Nvidia (CUDA), AMD (HIP), Intel (DPC++)
  - Accelerated linear algebra using UTK MAGMA
  - Parallel surrogate construction using libEnsemble
  - Mixed single-double precision methods for low memory footprint

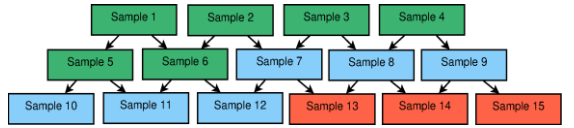
Adaptive Sparse Grid Sampling



Multiphysics simulation of Neutrino Radiation



Dynamic Adaptive Sampling



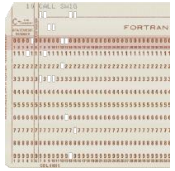
xSDK

ECP

https://github.com/ORNL/TASMANIAN

ECP EXASCALE COMPUTING PROJECT

ForTrilinos



ForTrilinos. Native Fortran interfaces. Extensive examples and testing. BSD license.

- Capabilities
  - **ForTrilinos**: idiomatic Fortran-2003 bindings to Trilinos numerical solvers, linear and nonlinear
  - **SWIG-Fortran**: generate similar bindings for any C++ library/headers
  - **Flibcpp**: Fortran bindings to C++ standard library containers/algorithms/random
- Readiness
  - Thin wrappers require pre-installed Trilinos and Fortran 2003–compatible compiler
  - Supports Trilinos GPU backends, currently with host-only interfaces

TASMANIAN

sundials

DTK

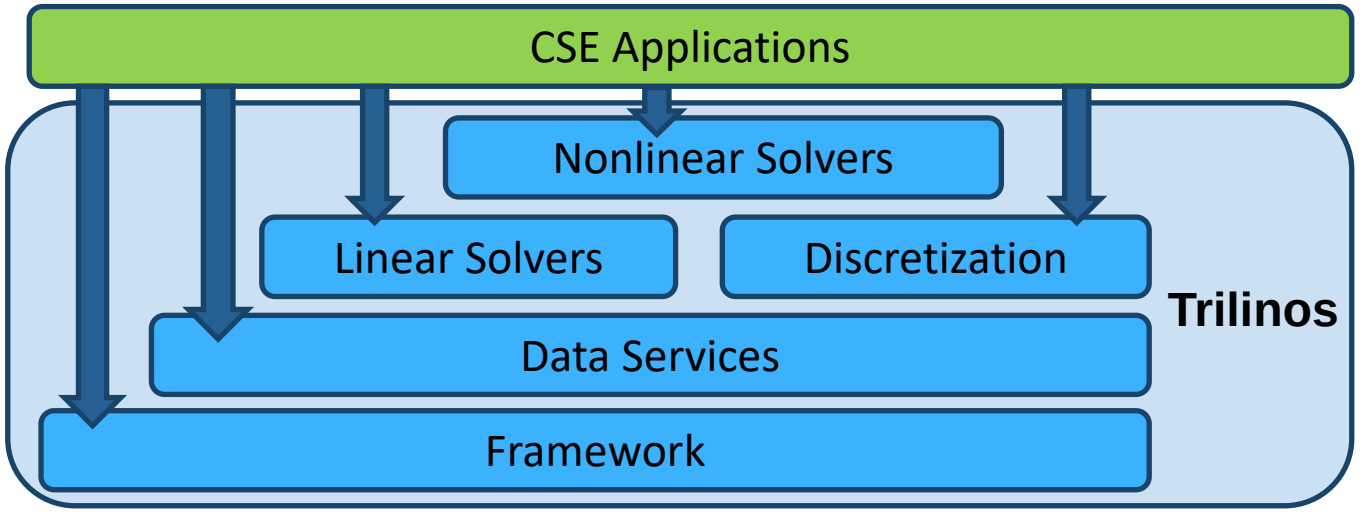
E3SM

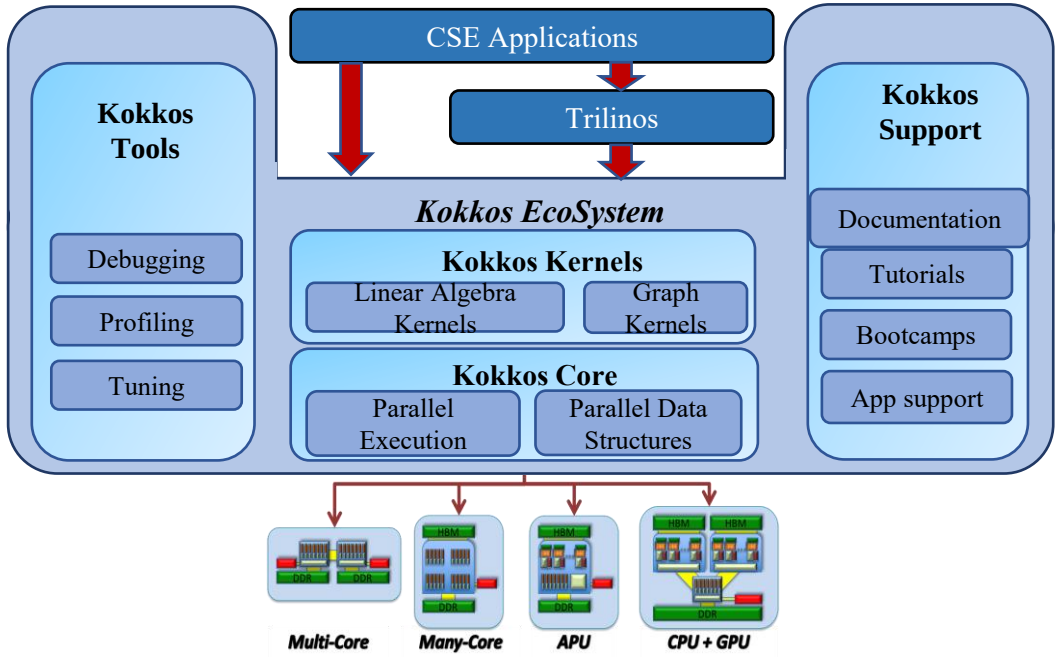
E4S

ECP

https://fortrilinos.readthedocs.io/en/latest/

Trilinos and Kokkos: Open-Source Toolkits of Mathematical Algorithms for HPC





Application Impact

- Solid mechanics, fluid dynamics, electrical circuits, etc.
- SIERRA, Empire, SPARC, Xyce, Drekar, Charon, etc.

Trilinos Software

53 packages in five areas

~100 contributors in total

~50+ active contributors

30-140 commits per week

Kokkos Core: parallel patterns and data structures; supports several execution and memory spaces
Kokkos Kernels: performance portable BLAS; sparse, dense and graph algorithms
Kokkos Tools: debugging and profiling support

Trilinos provides scalable algorithms to CSE applications, enabling high performance on current and next generation HPC platforms including several NVIDIA and AMD GPUs (experimental). Intel GPU support planned.

Kokkos Ecosystem addresses complexity of supporting numerous many/multi-core architectures such as NVIDIA, AMD, and Intel GPUs (planned) that are central to DOE HPC enterprise

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.


Gaps?
Questions?