

The State of MFEM

MFEM Community Workshop
October 20, 2021

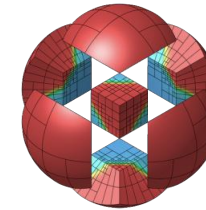
Tzanio Kolev
LLNL



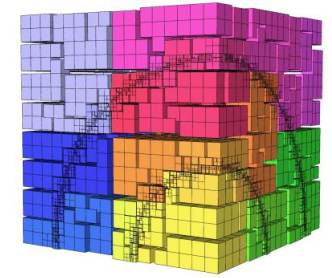
MFEM

Cutting-edge algorithms for powerful applications on HPC architectures

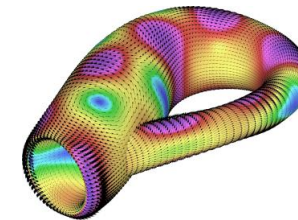
- **Flexible discretizations on unstructured grids**
 - Triangular, quadrilateral, tetrahedral and hexahedral meshes.
 - Local conforming and non-conforming AMR, mesh optimization.
 - Bilinear/linear forms for variety of methods: Galerkin, DG, DPG, ...
- **High-order and scalable**
 - Arbitrary-order H1, H(curl), H(div)- and L2 elements.
 - Arbitrary order curvilinear meshes.
 - MPI scalable to millions of cores and GPU-accelerated.
 - Enables application development from laptops to exascale machines.
- **Built-in solvers and visualization**
 - Integrated with: HYPRE, SUNDIALS, PETSc, SLEPc, SUPERLU, ...
 - AMG preconditioners for full de Rham complex, geometric MG
 - Support for GPU solvers from: HYPRE, PETSc, AmgX
 - Accurate and flexible visualization with VisIt, ParaView and GLVis
- **Open source**
 - Available on GitHub under BSD license. 75+ example codes and miniapps.
 - Part of FASTMath, ECP/CEED, xSDK, OpenHPC, E4S, ...



High-order curved elements



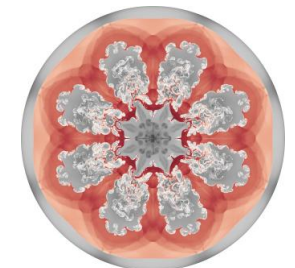
Parallel non-conforming AMR



Surface meshes



Heart modeling



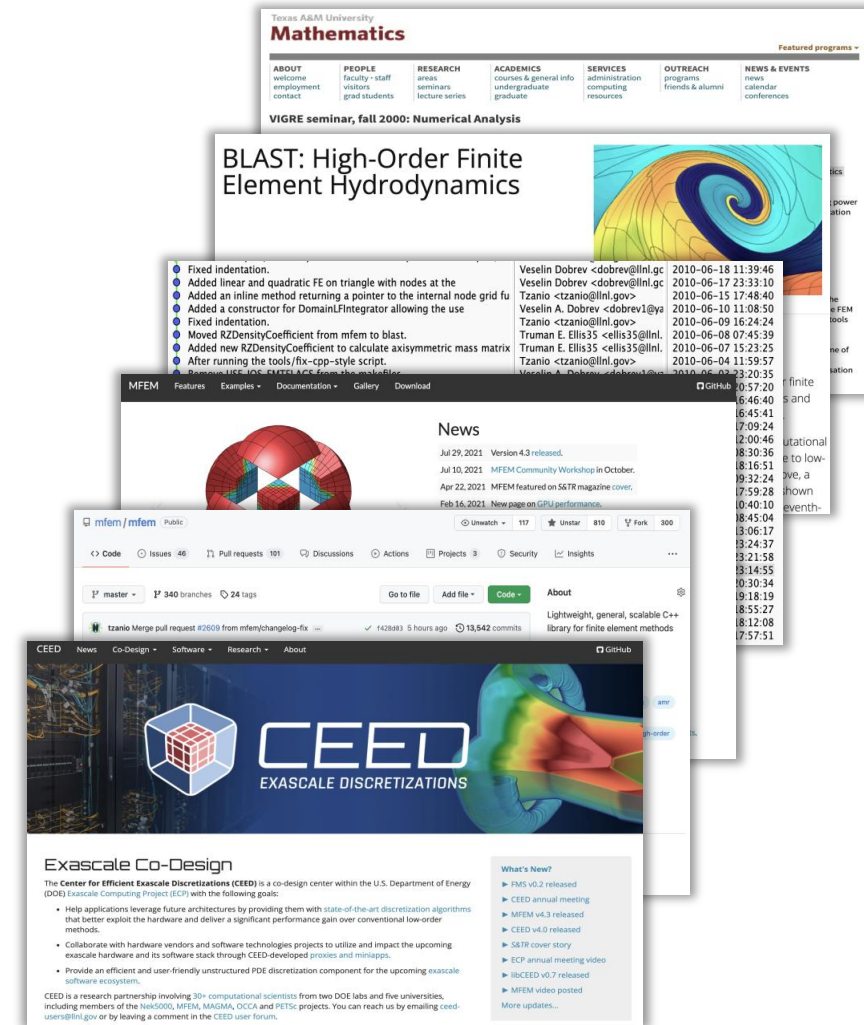
Compressible flow ALE simulations



A Brief History

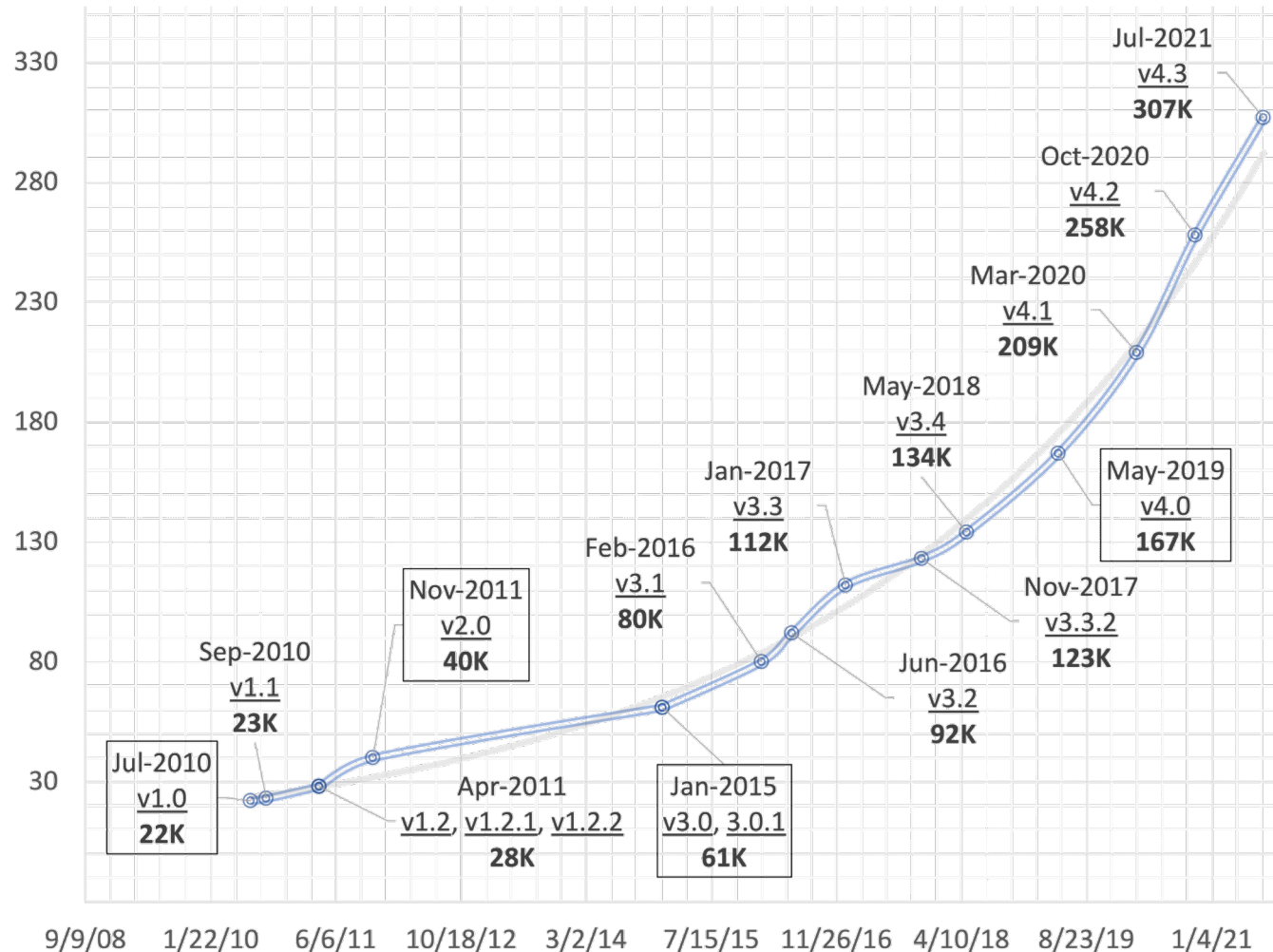
We've been doing this for a long time

- **2000 – “VIGRE seminar: Numerical Analysis,” Texas A&M University**
 - Research code: AggieFEM/aFEM
 - Some of the original contributors: [@v-dobrev](#), [@tzanio](#), [@stomov](#)
 - Used in summer internships at LLNL
- **2010 – BLAST project at LLNL**
 - Motivated high-order, non-conforming AMR and parallel scalability developments
 - MFEM repository starts in May 2010
 - Some of the original contributors: [@v-dobrev](#), [@tzanio](#), [@rieiben1](#), [@trumanellis](#)
 - Project website [mfem.org](#) goes live in August 2015
- **2017 – Development moved to GitHub**
 - First GitHub commits in February 2017
 - Team expands to include many new developers at LLNL and externally
- **2017 – CEED project in the ECP**
 - Motivated partial assembly, GPU, and exascale computing developments



The Source Code Has Grown Significantly

SLOC in MFEM releases over the last 11 years



mfem-4.3.tgz	v4.3	Jul 2021	2.8M	307K	
mfem-4.2.tgz	v4.2	Oct 2020	2.4M	258K	
mfem-4.1.tgz	v4.1	Mar 2020	7.9M	209K	
mfem-4.0.tgz	v4.0	May 2019	5.2M	167K	GPU support
mfem-3.4.tgz	v3.4	May 2018	4.4M	134K	
mfem-3.3.2.tgz	v3.3.2	Nov 2017	4.2M	123K	mesh optimization
mfem-3.3.tgz	v3.3	Jan 2017	4.0M	112K	
mfem-3.2.tgz	v3.2	Jun 2016	3.3M	92K	dynamic AMR, HPC miniapps
mfem-3.1.tgz	v3.1	Feb 2016	2.9M	80K	fem ↔ linear system interface
mfem-3.0.1.tgz	v3.0.1	Jan 2015	1.1M	61K	
mfem-3.0.tgz	v3.0	Jan 2015	1.1M	61K	non-conforming AMR
mfem-2.0.tgz	v2.0	Nov 2011	308K	40K	arbitrary order spaces, NURBS
mfem-v1.2.2.tgz	v1.2.2	Apr 2011	240K	28K	
mfem-v1.2.1.tgz	v1.2.1	Apr 2011	240K	28K	
mfem-v1.2.tgz	v1.2	Apr 2011	240K	28K	MPI parallelism based on hypre
mfem-v1.1.tgz	v1.1	Sep 2010	166K	23K	
mfem-v1.0.tgz	v1.0	Jul 2010	160K	22K	initial release

The Community Has Grown Significantly

GitHub, downloads, and workshop stats

GitHub

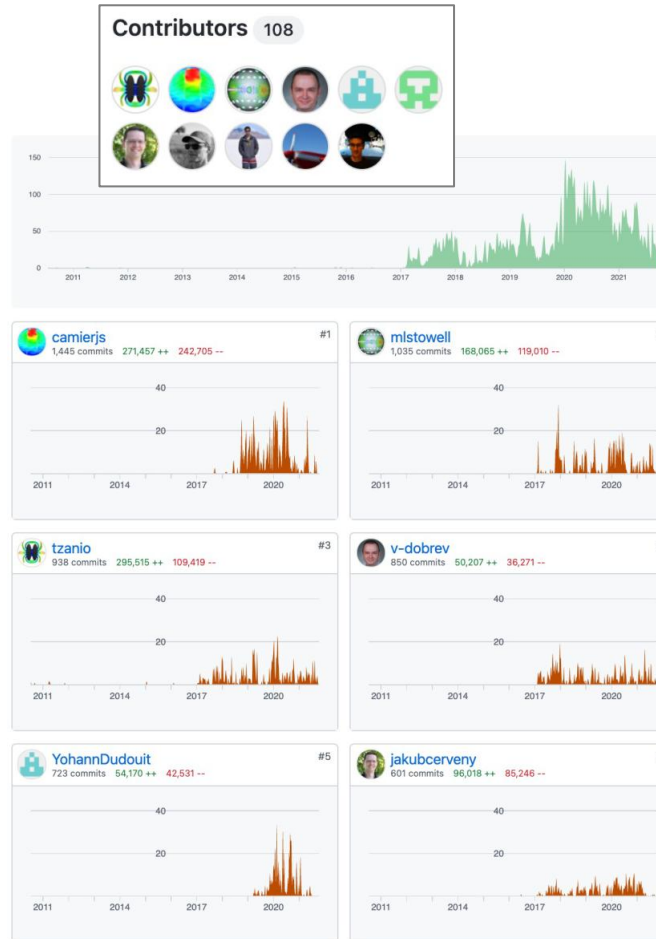
- **108** contributors
- **100** commits / week
- **456** people in the mfem organization – *join to contribute + receive announcements*
- **100** visitors / day
- **810** stars – *thank you!*

Downloads

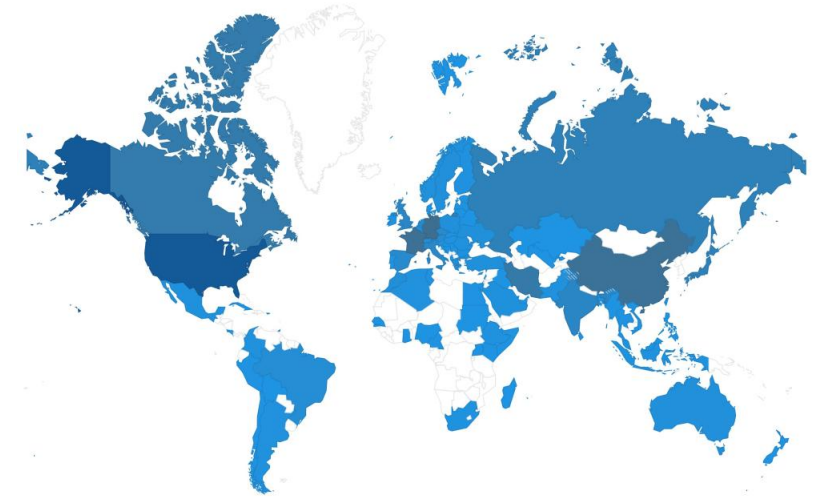
- **35** downloads + clones / day · **12K** / year
- **102** countries total

2021 Community Workshop

- **238** researchers
- **120** organizations
- **28** countries



Top contributors as of Oct 2021



MFEM has been downloaded from 102 countries

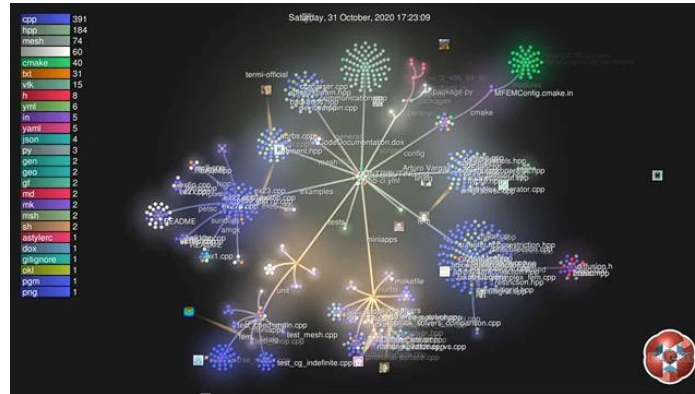
mfem.org	MFEM Community Workshop	October 2021
001 Aaron Fisher	Lawrence Livermore National Laboratory	fisher47@llnl.gov
002 Abdelrahman Elmeligy	North Carolina State University	aelmel1@ncsu.edu
003 Abhilash Reddy Malipeddi	University of Michigan	abhilash@gwu.edu
004 Abhishek Verma	Applied Materials Inc.	AbhishekKumar_Verma@amat.com
005 Aditya Joshi	Rensselaer Polytechnic Institute	joshi14@rpi.edu
006 Adolfo Rodriguez	OpenSim Technology LLC	adolfo@opensim.technology
007 Adriano Cortes	Federal University of Rio de Janeiro	adriano@acad.ufrj.br
008 Adrien Lefieux	Covanos	adrien.lefieux@gmail.com
009 Aidan Hamilton	University of Delaware	aidan@udel.edu
010 Ajay Rawat		ajay.rawat83@gmail.com
011 Alberto Gascón	University of Granada	albergasc@gmail.com
012 Alejandro Campos	Lawrence Livermore National Laboratory	campos33@llnl.gov
013 Alejandro Muñoz Manterola	University of Granada	alevalle71@gmail.com

2021 Community workshop had 238 registrations

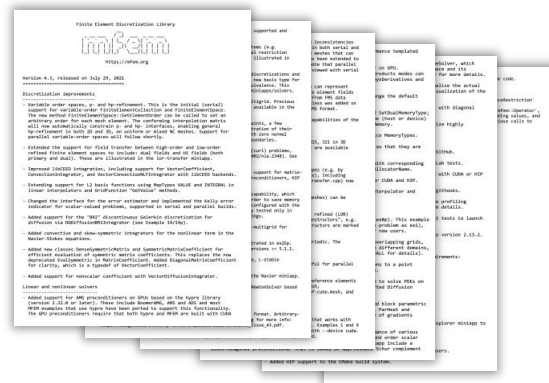
Latest Releases Was a Team Effort

Version 4.3 stats

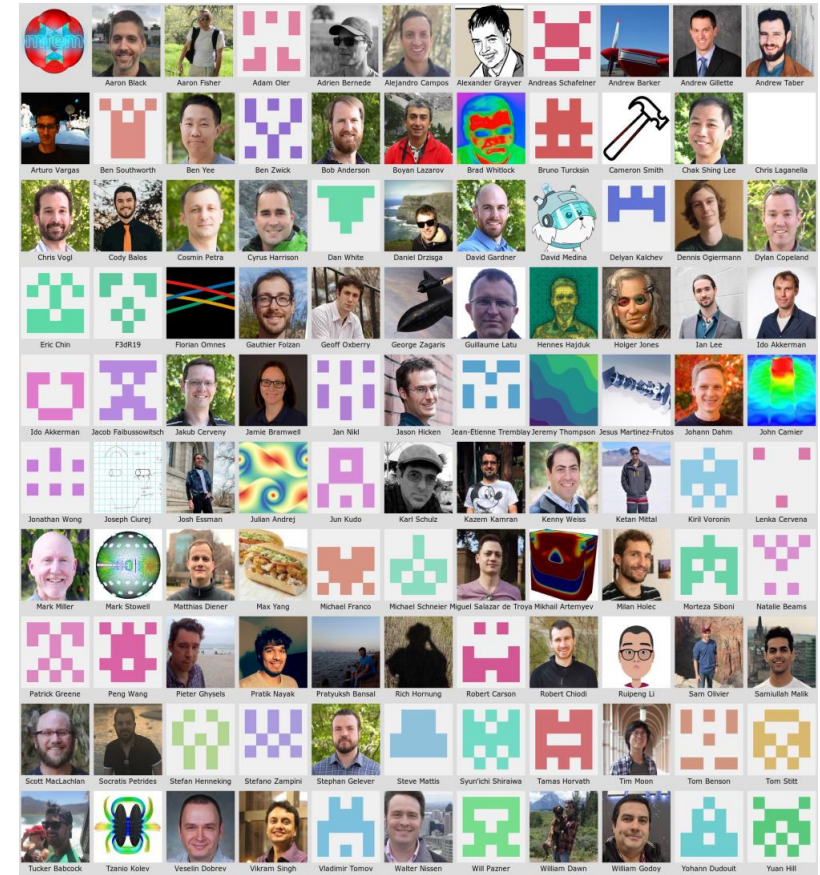
- Released **July 29, 2021**
- 9** months in development
- 81** contributors
- 218** PRs merged
- 265** issues closed
- 48862** new lines of code
- 3806** number of commits
- Many new features:**
 - GPU solvers from hypre + PETSc
 - LOR discretizations, hp-refinement
 - GPU-powered mesh optimization
 - FMS, Caliper, Ginkgo, VTK support
 - 11 new examples + miniapps



The making of mfem-4.3
youtu.be/3Fc1nxQJUVw



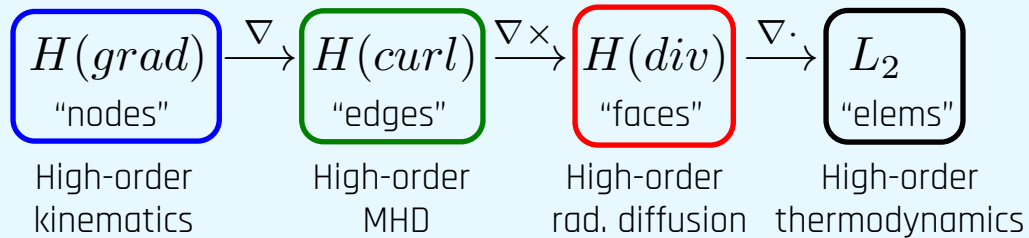
The mfem-4.3 CHANGELOG has 60+ entries



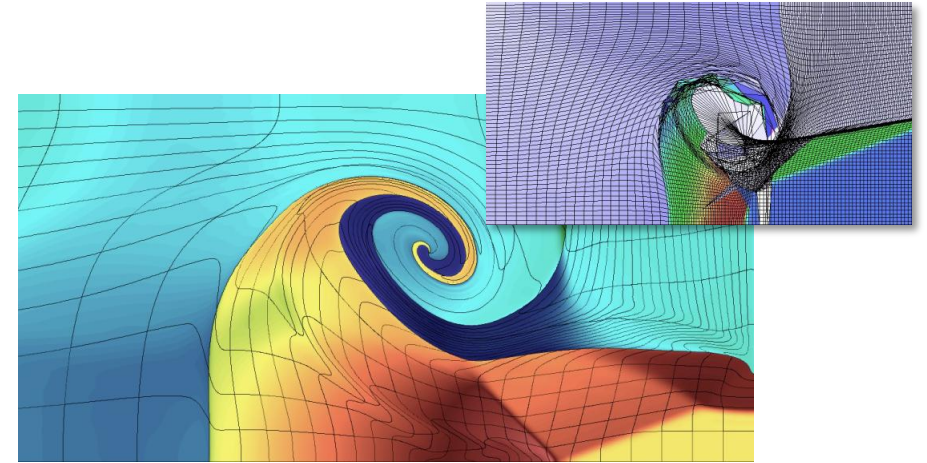
MFEM contributors on GitHub

High-Order Methods for Large-Scale Multi-Physics

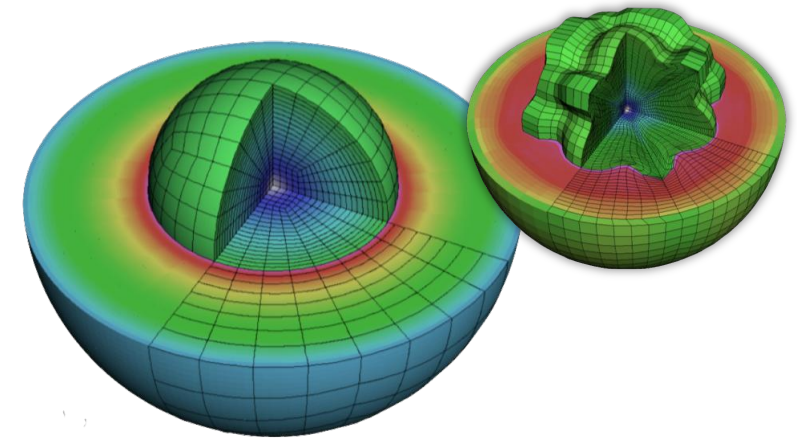
- **Large-scale parallel multi-physics simulations**
 - Radiation diffusion
 - Electromagnetic diffusion
 - Compressible hydrodynamics
- **Finite elements naturally connect different physics**



- **High-order finite elements on high-order meshes**
 - Increased accuracy for smooth problems
 - Sub-element modeling for problems with shocks
 - HPC utilization, FLOPs/bytes increase with the order
- **Need new (interesting!) R&D for full benefits**
 - Meshing, discretizations, solvers, AMR, UQ, visualization, ...



Robustness: 8th order Lagrangian shock triple-point (BLAST) vs. classical low-order method (SGH)

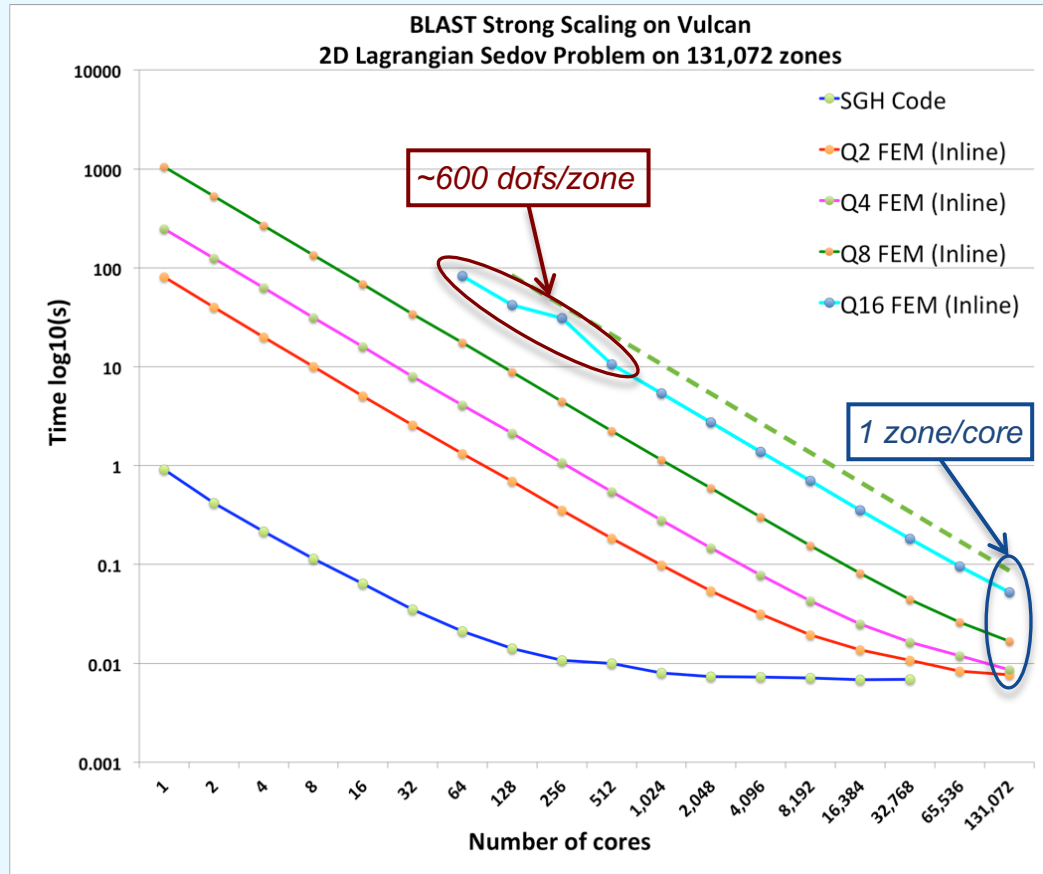


Symmetry: 2nd order Lagrangian ICF-like implosion (BLAST) vs. classical low-order method (SGH)

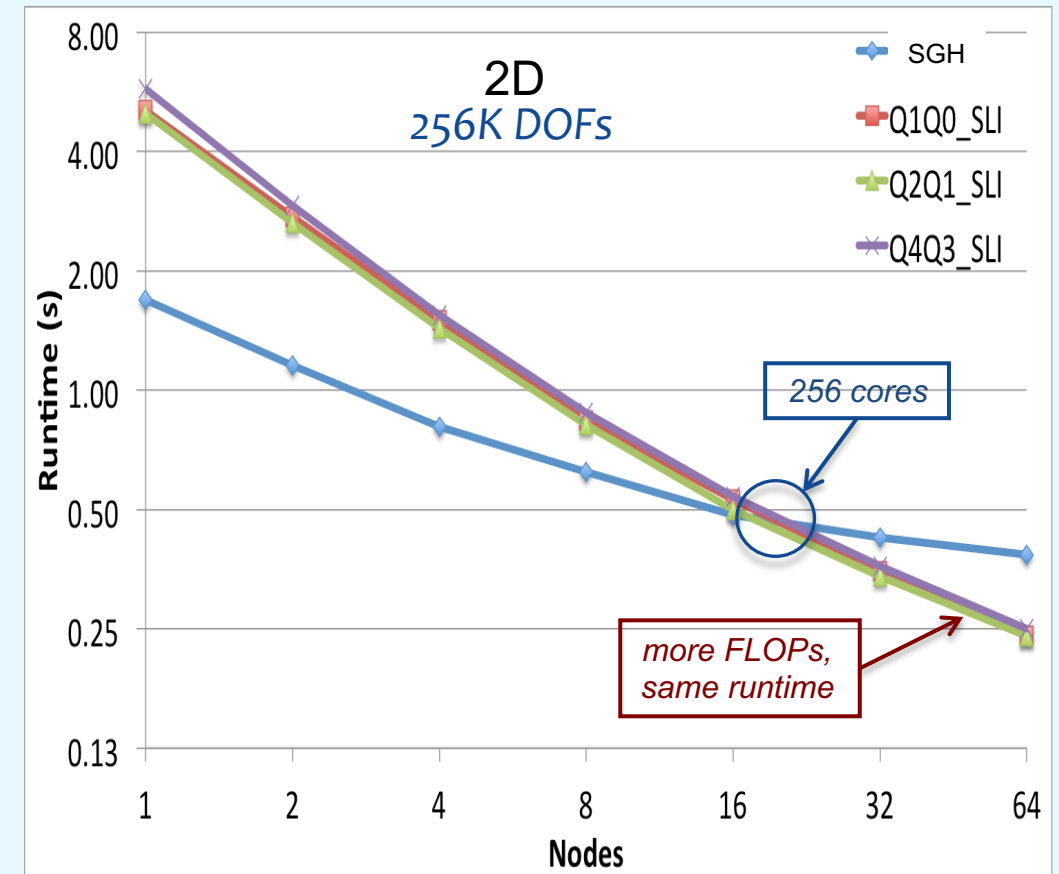
High-Order Methods for Large-Scale Multi-Physics

Parallel scalability

Strong scaling, p-refinement, PA



Strong scaling, fixed #dofs, PA+SLI



Adaptive Mesh Refinement

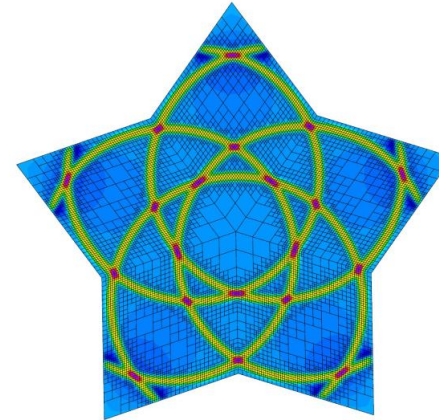
MFEM's unstructured AMR infrastructure

- **AMR on library level**

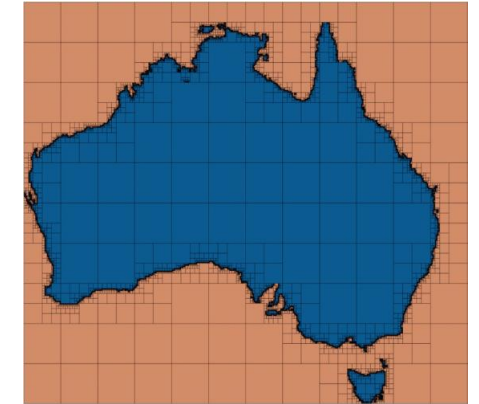
- Conforming local refinement on simplex meshes
- Non-conforming refinement for quad/hex meshes
- Initial hp-refinement

- **General approach**

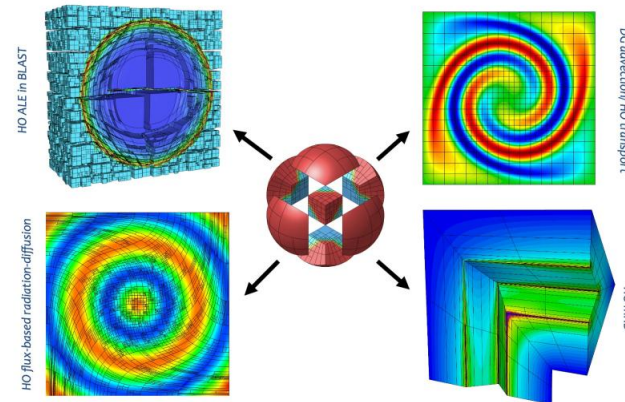
- Any high-order finite element space, H^1 , $H(\text{curl})$, $H(\text{div})$, on any high-order curved mesh
- 2D and 3D · hexes, prisms, tets
- Arbitrary order hanging nodes
- Anisotropic refinement
- Derefinement
- Serial and parallel, including parallel load balancing
- Independent of the physics
- Easy to incorporate in applications



Example 15



Shaper miniapp

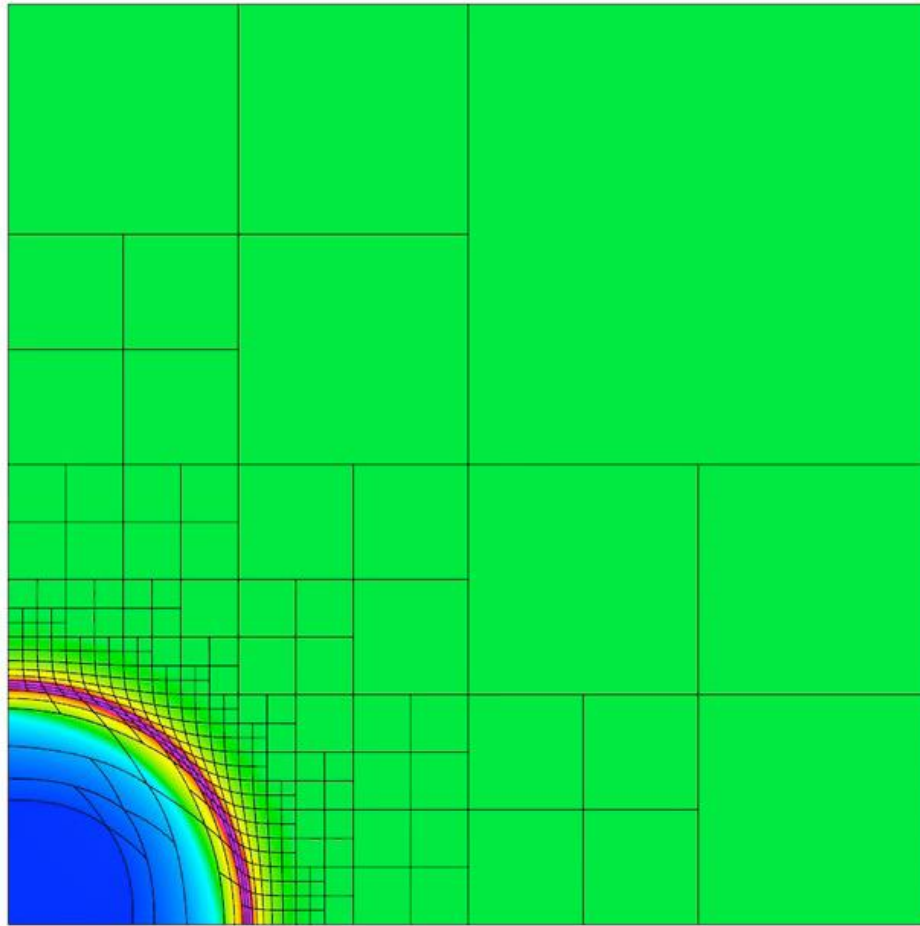


Same AMR algorithms can be applied to a variety of high-order physics

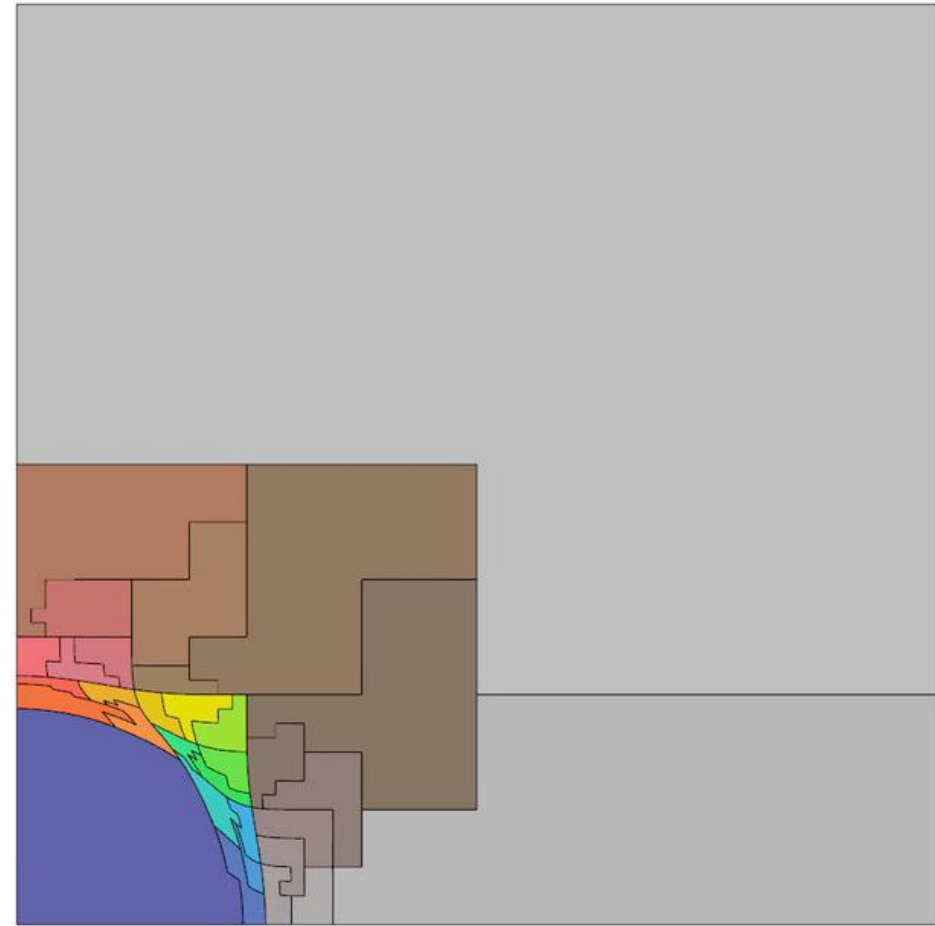


Adaptive Mesh Refinement

Parallel dynamic AMR, 2nd order Lagrangian Sedov problem



Adaptive, viscosity-based refinement and derefinement



Parallel load balancing, 16 cores

Adaptive Mesh Refinement

Parallel AMR scaling to ~400K MPI tasks

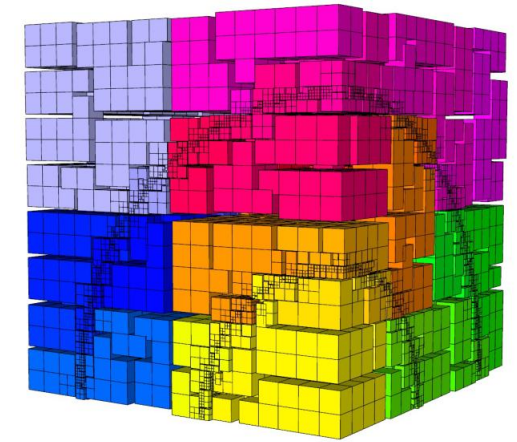
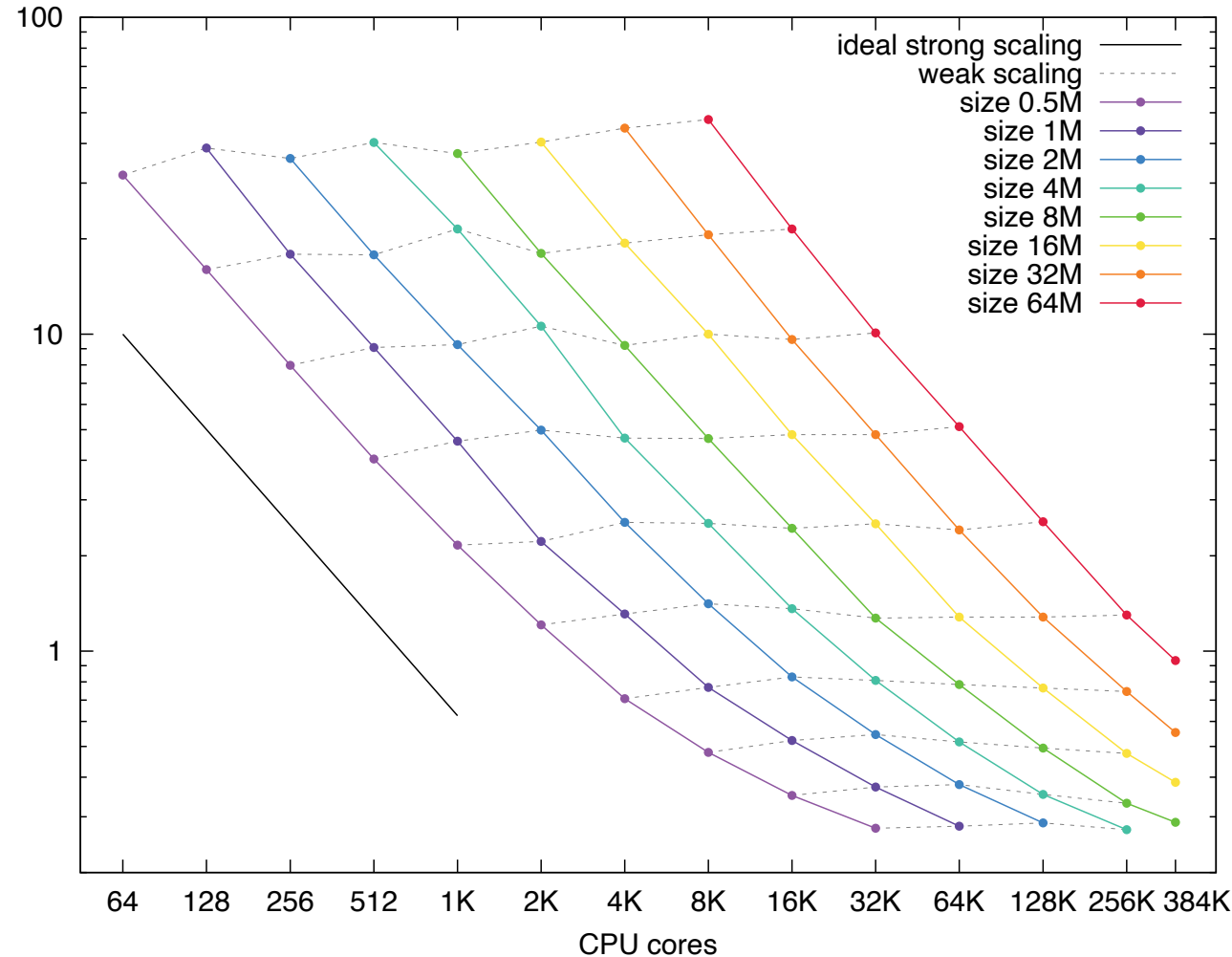
- **Weak + strong scaling**

- ~400K MPI tasks
- BG/Q

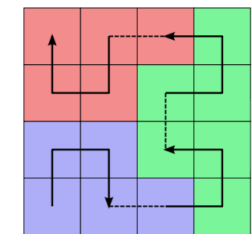
- **Measure only AMR components**

- Interpolation matrix
- Assembly
- Marking
- Refinement
- Rebalancing
- No linear solves
- No “physics”

- **Documented in 2019 SISC paper**



Parallel decomposition (2048 domains shown)

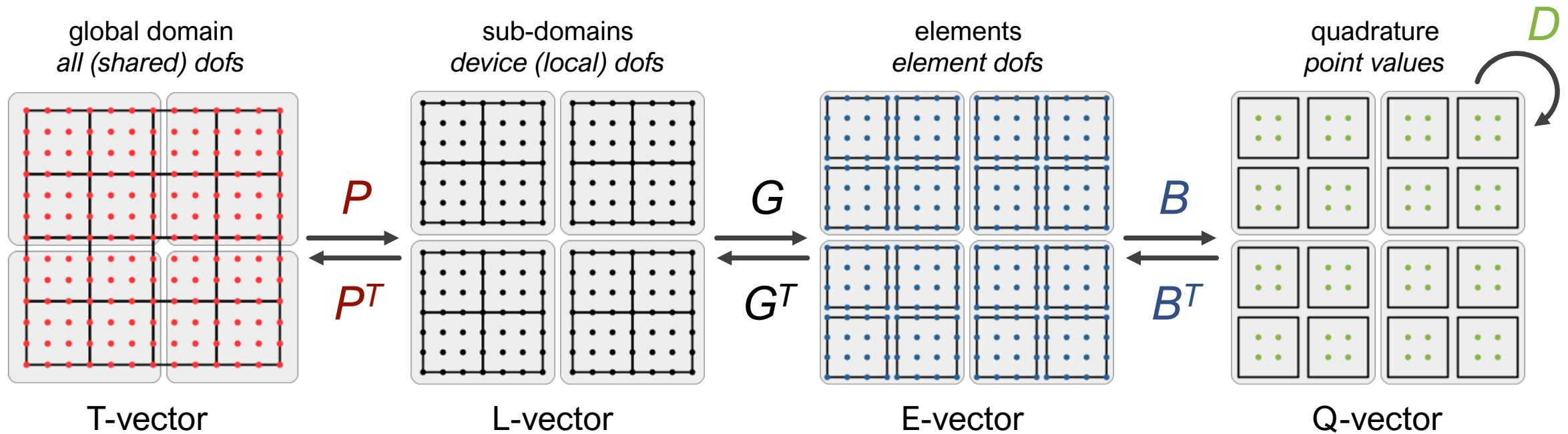


Parallel partitioning via Hilbert curve

FEM Operator Decomposition + Partial Assembly

Decompose A into **parallel**, mesh, **basis**, and **geometry/physics** parts

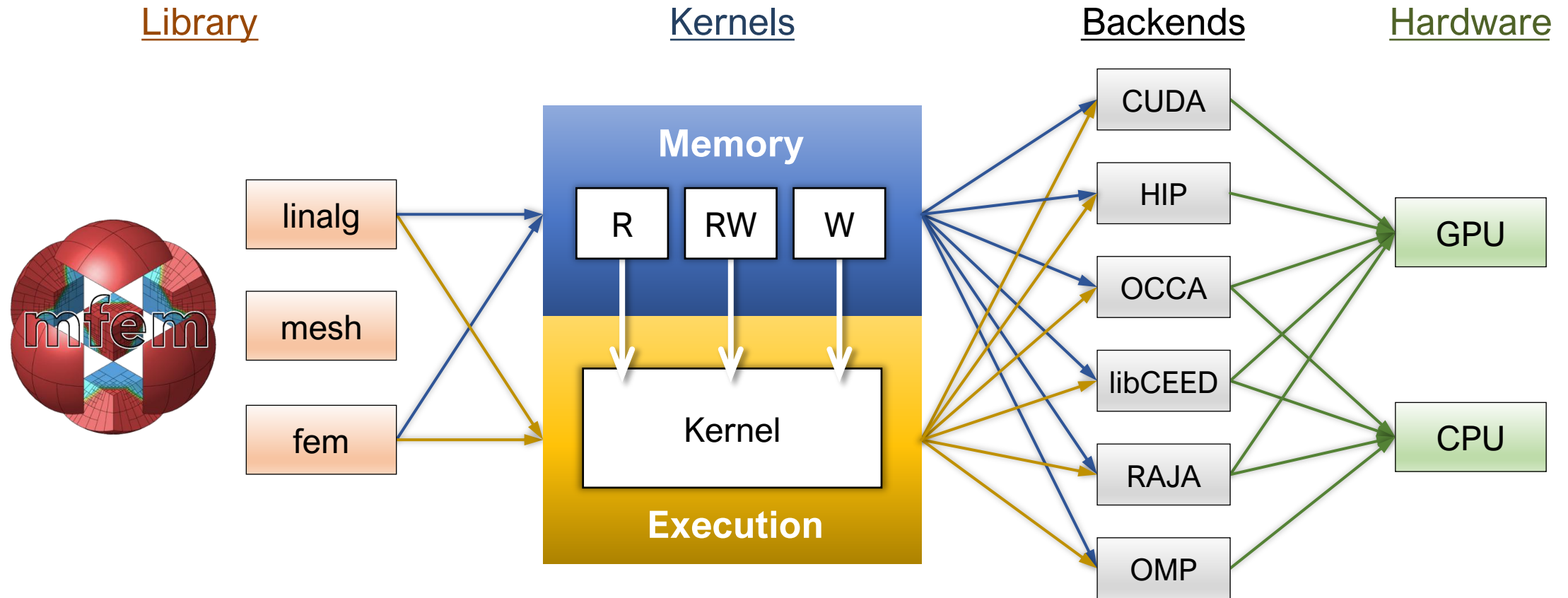
$$A = P^T G^T B^T D B G P$$



- Partial assembly = store only D , evaluate B
- Optimal memory, near-optimal FLOPs compared to A

GPU Support

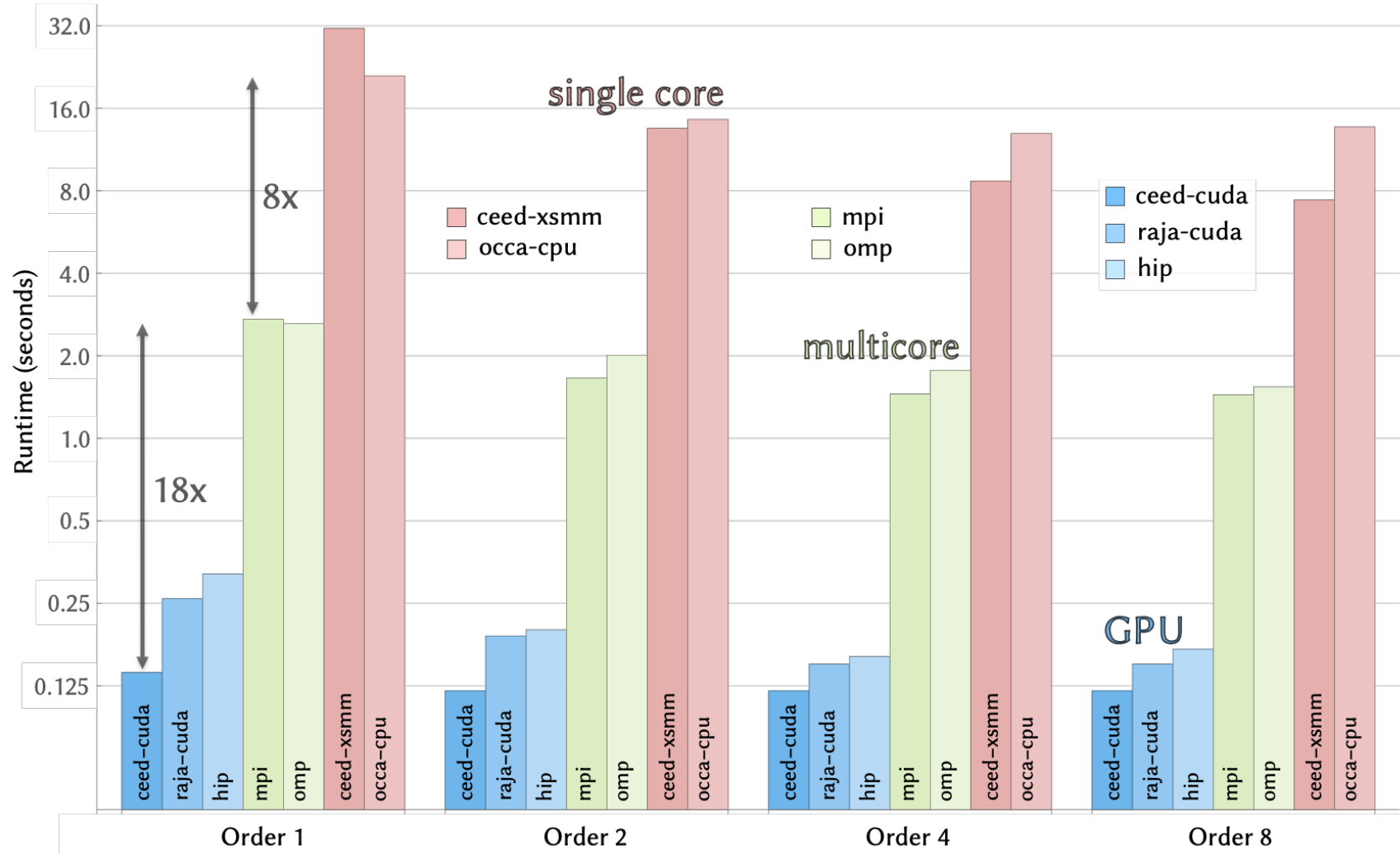
MFEM has provided GPU acceleration for over 2 years (since mfem-4.0)



- Backends are runtime selectable, can be mixed
- Coming soon: support for Intel/SYCL

GPU Support

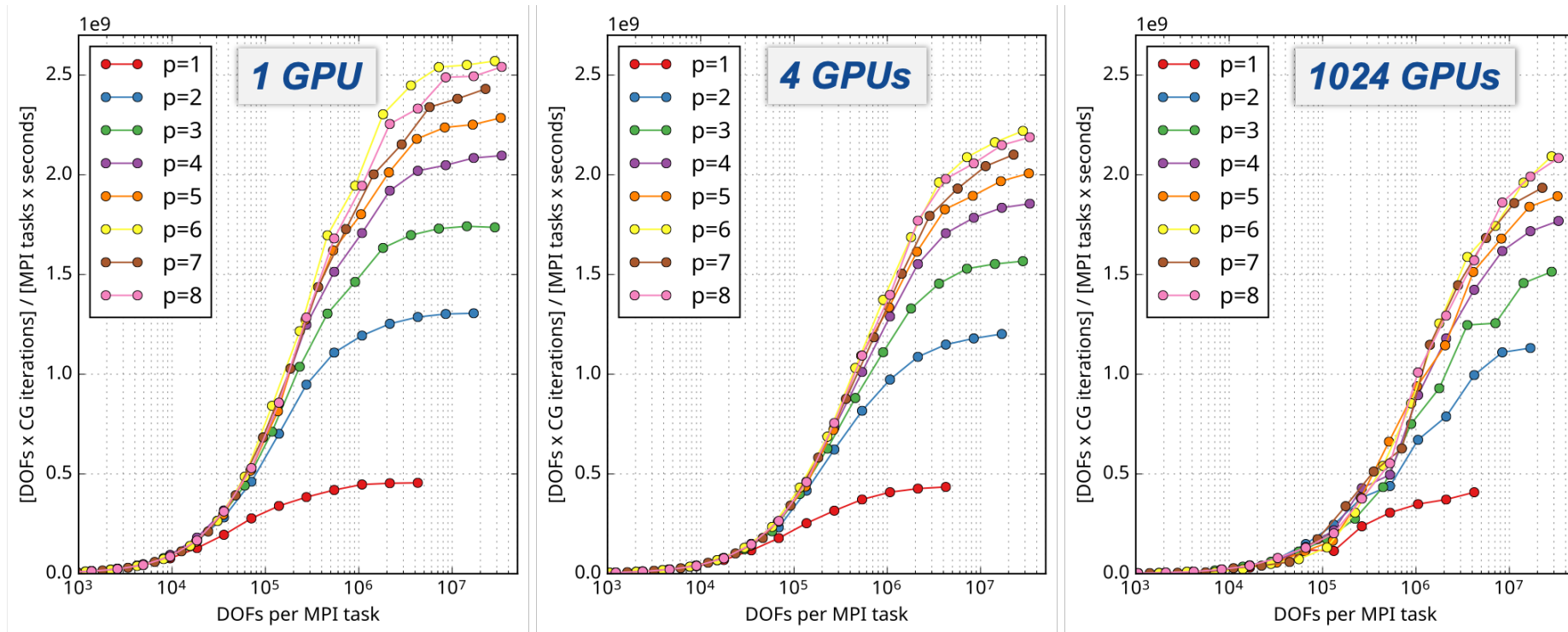
Device backends in MFEM, desktop performance



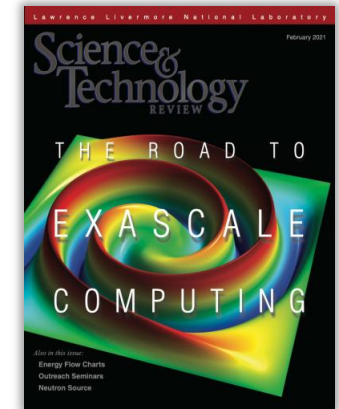
- MFEM-4.2
- October 2020
- Example 1
- 2D, 1.3M dofs
- 200 CG-PA iterations
- Intel Skylake 16 core (Linux)
- AMD MI60 (Corona)
- NVIDIA GV100 (Linux)
- sm_70, CUDA 10.1
- gcc-8.4.0

GPU Support

MFEM performance on multiple GPUs, 3D scalar diffusion, Lassen



- Parallel scaling on Lassen (4 V100 GPUs/node)
- Local performance vs. local problem size
- Best performance: 2.1 TDOF/s
- Largest size: 34 billion dofs



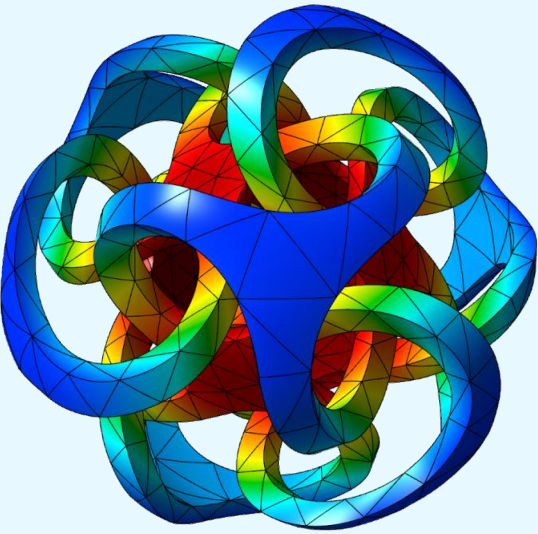
- Benchmarks (BPs)
- Miniapps (Laghos)
- libCEED

ceed.exascaleproject.org

Visualization

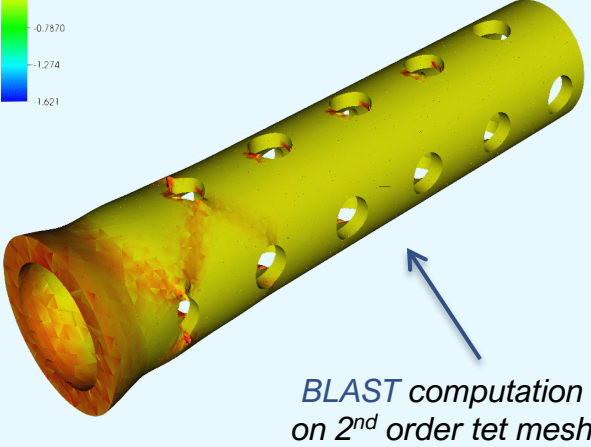
MFEM supports several options for accurate + flexible finite element visualization

GLVis
native lightweight visualization



glvis.org

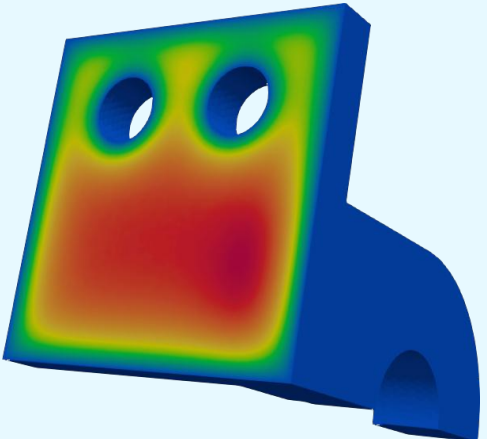
Visit
general data analysis tool



*BLAST computation
on 2nd order tet mesh*

visit.llnl.gov

ParaView
general data analysis tool



paraview.org

Additional I/O support: Conduit, ADIOS, VTK, FMS, ...

Examples

The first stop for new users

Example Codes and Miniapps

This page provides a brief overview of MFEM's example codes and miniapps. For detailed documentation of the MFEM sources, including the examples, see the [online Doxygen documentation](#), or the `doc` directory in the distribution.

The goal of the example codes is to provide a step-by-step introduction to MFEM in simple model settings. The miniapps are more complex, and are intended to be more representative of the advanced usage of the library in physics/application codes. We recommend that new users start with the example codes before moving to the miniapps.

Select from the categories below to display examples and miniapps that contain the respective feature. All examples support (arbitrarily) high-order meshes and finite element spaces. The numerical results from the example codes can be visualized using the GLVis visualization tool (based on MFEM). See the [GLVis website](#) for more details.

Users are encouraged to submit any example codes and miniapps that they have created and would like to share. Contact a member of the MFEM team to report [bugs](#) or post [questions](#) or [comments](#).

Application (PDE) Finite Elements Discretization Solver

Example 1: Laplace Problem

This example code demonstrates the use of MFEM to define a simple isotropic Laplace problem

$$-\Delta u = 1$$

with homogeneous Dirichlet boundary conditions. Specifically, we discretize the domain using a mesh (linear by default, quadratic for quadratic curvilinear mesh, NURBS for NURBS mesh, etc.).

The example highlights the use of mesh refinement, finite element grid functions, as well as linear and bilinear forms corresponding to the left-hand side and right-hand side of the discrete linear system. We also cover the explicit elimination of essential boundary conditions, static condensation, and the optional connection to the GLVis tool for visualization.

The example has a serial (`ex.1.cpp`), a parallel (`ex.1p.cpp`), and HPC versions: [performance/ex.1.cpp](#), [performance/ex.1p.cpp](#). It also has a PETSc modification in [examples/petsc](#), a PUMI modification in [examples/pumi](#) and a Ginkgo modification in [examples/ginkgo](#). Partial assembly and GPU devices are supported.

Example 2: Linear Elasticity

This example code solves a simple linear elasticity problem describing a multi-material cantilever beam. Specifically, we approximate the weak form of

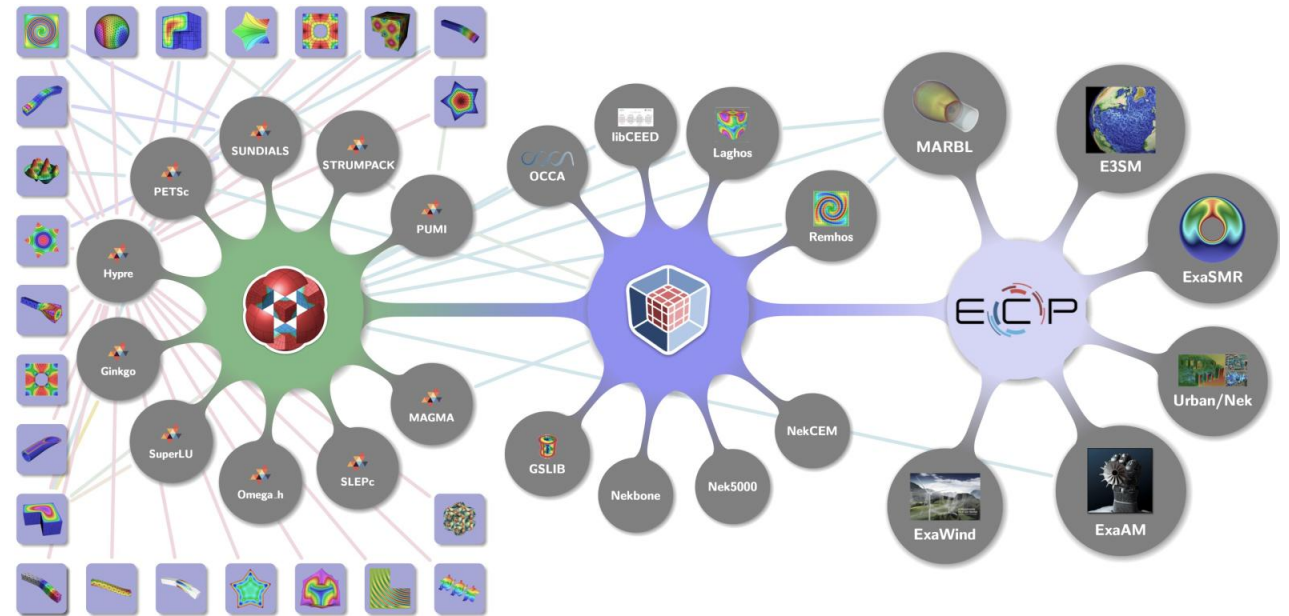
$$-\operatorname{div}(\sigma(\mathbf{u})) = 0$$

where

$$\sigma(\mathbf{u}) = \lambda \operatorname{div}(\mathbf{u}) \mathbf{I} + \mu (\nabla \mathbf{u} + \nabla \mathbf{u}^T)$$

is the stress tensor corresponding to displacement field \mathbf{u} , and λ and μ are the material Lamé constants. The boundary conditions are $\mathbf{u} = 0$ on the fixed part of the boundary with attribute 1, and $\sigma(\mathbf{u}) \cdot \mathbf{n} = \mathbf{f}$ on the remainder with \mathbf{f} being a constant pull down vector on boundary elements with attribute 2, and zero otherwise. The geometry of the domain is assumed to be as follows:

mfem.org/examples



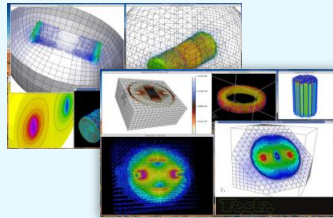
- 30 example codes, most with both serial + parallel versions
- Tutorials to learn MFEM features
- Starting point for new applications
- Show integration with many external packages, miniapps

Miniapps

More advanced, ready-to-use physics solvers

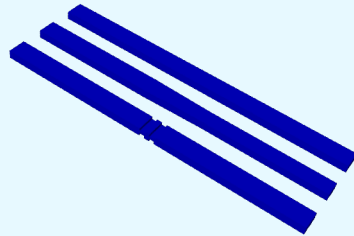
Volta, Tesla, Maxwell and Joule Miniapps *Static and transient electromagnetics*

- **Volta** $-\nabla \cdot \epsilon \nabla \varphi = \rho - \nabla \cdot \vec{P}$
- **Tesla** $\nabla \times \mu^{-1} \nabla \times \vec{A} = \vec{J} + \nabla \times \mu^{-1} \mu_0 \vec{M}$

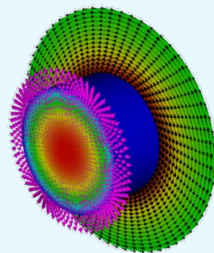


- **Maxwell** · *transient full-wave EM*

$$\frac{\partial(\epsilon \vec{E})}{\partial t} = \nabla \times (\mu^{-1} \vec{B}) - \sigma \vec{E} - \vec{J}$$
$$\frac{\partial \vec{B}}{\partial t} = -\nabla \times \vec{E}$$



- **Joule** · *transient magnetics + Joule heating*
- Arbitrary order elements + meshes
- Adaptive mesh refinement



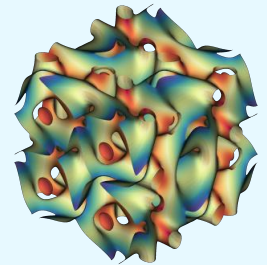
mfem.org/electromagnetics

Navier Miniapp

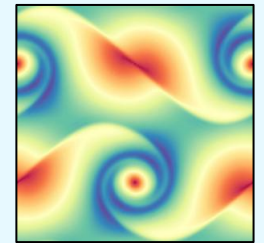
Transient incompressible Navier-Stokes equations

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{f}$$
$$\nabla \cdot \mathbf{u} = 0$$

- Arbitrary order elements
- Arbitrary order curvilinear mesh elements
- Adaptive IMEX (BDF-AB) time-stepping algorithm up to 3rd order
- State-of-the-art HPC performance
- GPU acceleration
- Convenient user interface



3D Taylor-Green vortex, 7th order

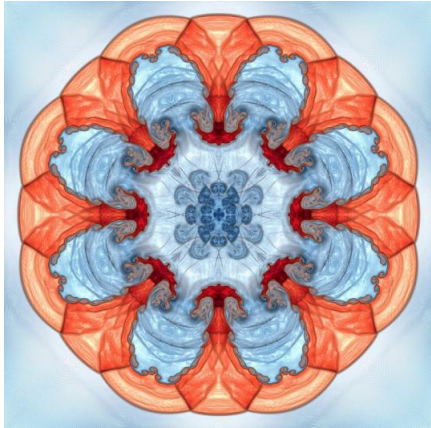


Double shear layer, 5th order, Re = 100000

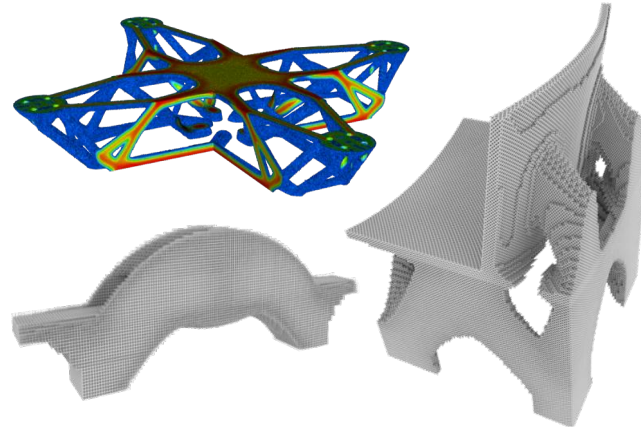
mfem.org/fluids

Applications

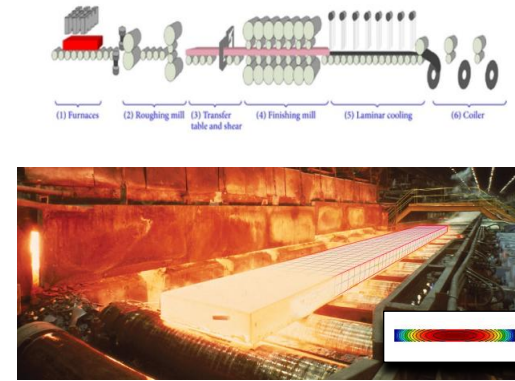
Some of the large-scale simulation codes powered by MFEM



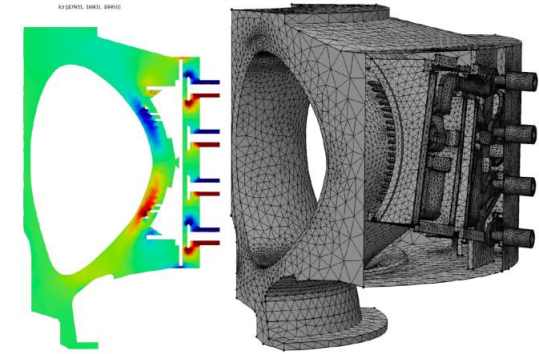
Inertial confinement fusion (BLAST)



Topology optimization for additive manufacturing (LiDO)



Hot strip mill slab modeling (U.S. Steel)



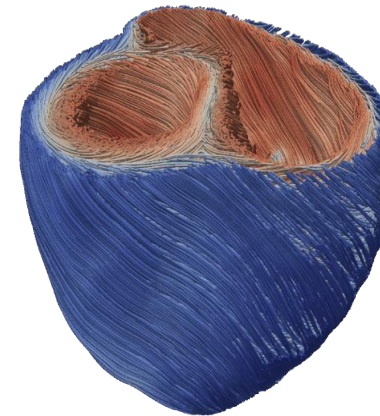
Core-edge tokamak EM wave propagation (SciDAC, RPI)



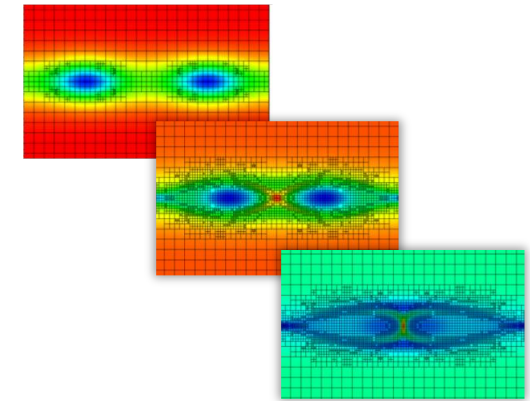
Electric aircraft design (RPI)



MRI modeling (Harvard Medical)



Heart modeling (Cardioid)



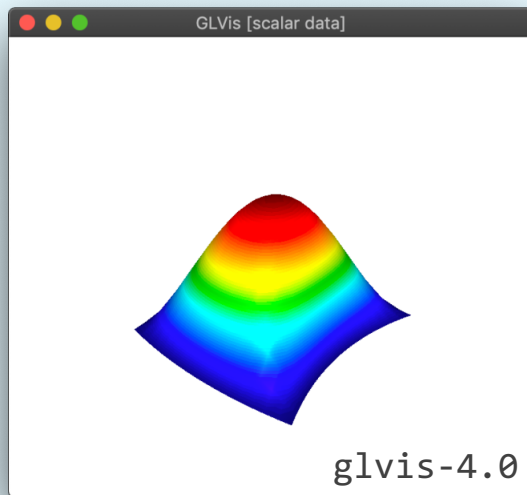
Adaptive MHD island coalescence (SciDAC, LANL)

Visualization

Web + Python support

Modern OpenGL

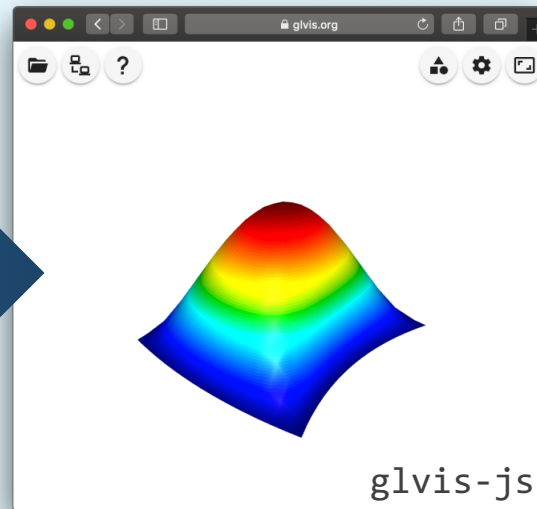
OpenGL3, SDL



glvis.org

Web Visualization

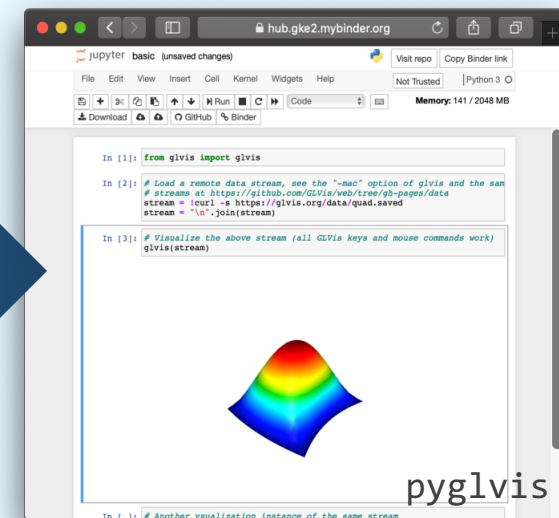
JS/WebAssembly, Emscripten



glvis.org/live

FEM in Python

PyMFEM, Jupyter, Binder



`pip install glvis`

Try glvis-js and pyglvis in your desktop or mobile browser

Roadmap for Next Year

Plans for FY22

▪ GPU support

- Parallel GPU assembly · Scalable solvers for the full HO de Rham complex
- Performance on AMD GPU · Intel GPU support
- Easier to write GPU kernels · Continued performance improvement · Cleaner PA code

▪ Application needs

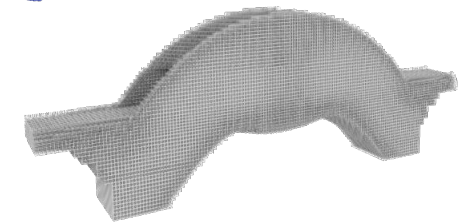
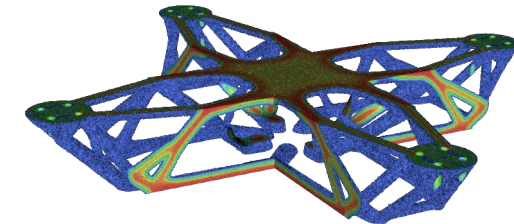
- Subdomain extraction · Different physics in different domains
- User-defined flexible variational forms
- Better handling of nonlinear problems
- Large initial meshes · Mesh partitioning miniapp · Parallel re-partitioning
- Automatic differentiation · Mixed meshes · ML/DRL · Interface sharpening

▪ Code quality

- Improve Doxygen documentation · Additional examples + miniapps
- Unify interfaces · Polish instead of adding new features

▪ New releases

- v4.4 in January · v5.0 coming soon – *expect breaking changes!*



mfem-4.4

📅 Due by January 31, 2022

mfem-5.0

📅 Due by July 31, 2022

Future Roadmap

What should we tackle next?

- **Topics we are considering**

- Design optimization
- Cloud computing
- MFEM in industry
- Connections with ML
- Automated compilation of variational forms · JIT support
- ARM and post-GPU hardware

- **Release schedule**

- Official release every 6 months
- GLVis release 1 month after MFEM release

- **What would you like to see?**

- Geometry pre-processing
- Better error estimators
- Meshes with elements of different dimensions
- Parallel anisotropic AMR
- 4D support

- **Please let us know**

- Slack: [#meet-the-team](#)
- GitHub: github.com/mfem/mfem/issues
- Email: mfem@llnl.gov

Thank you from the MFEM team at LLNL!



**Bob
Anderson**
[@rw-anderson](#)



**Julian
Andrej**
[@jandrej](#)



**Jamie
Bramwell**
[@jamiebramwell](#)



**John
Camier**
[@camierjs](#)



**Jakub
Cerveny**
[@jakubcerveny](#)



**Dylan
Copeland**
[@dylan-copeland](#)



**Veselin
Dobrev**
[@v-dobrev](#)



**Yohann
Dudouit**
[@YohannDudouit](#)



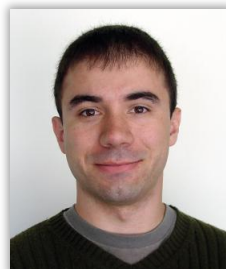
**Aaron
Fisher**
[@acfisher](#)



**Milan
Holec**
[@homijan](#)



**Brendan
Keith**
[@brendankeith](#)



**Tzanio
Kolev**
[@tzanio](#)



**Boyan
Lazarov**
[@bslazarov](#)



**Ketan
Mittal**
[@kmittal2](#)



**Will
Pazner**
[@pazner](#)



**Socratis
Petrides**
[@psocratis](#)



**Mark
Stowell**
[@mlstowell](#)

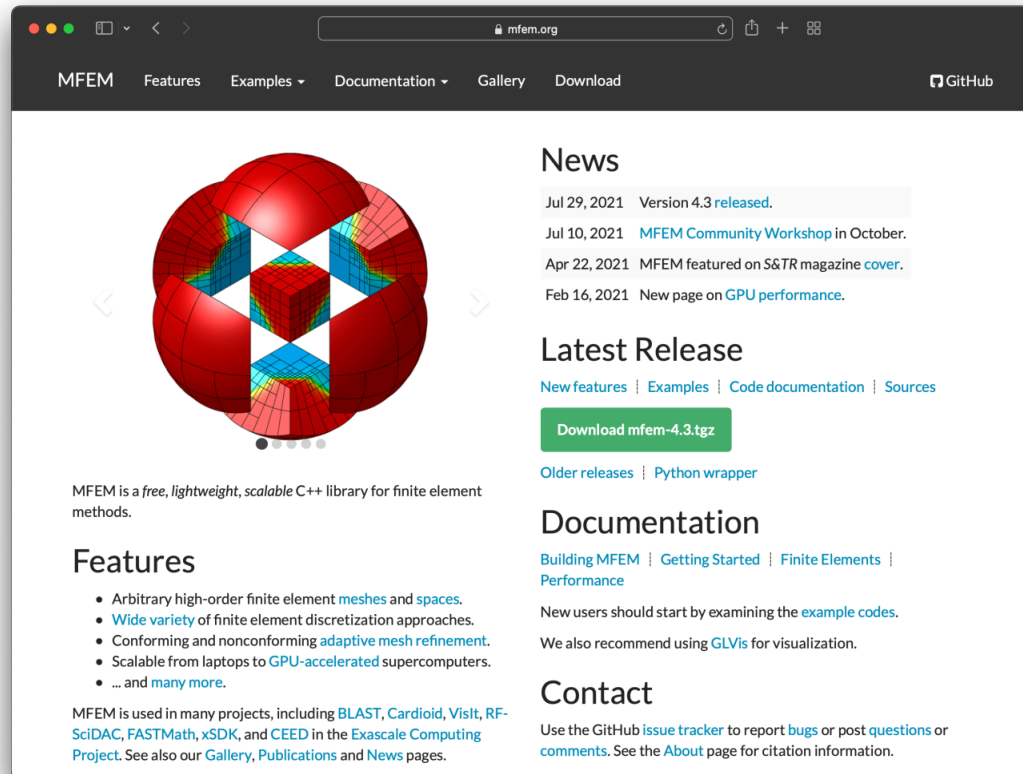


**Vladimir
Tomov**
[@vladotomov](#)



**Chris
Vogl**
[@cjvog1](#)

MFEM Resources



The screenshot shows the MFEM website homepage. The navigation bar includes links for MFEM, Features, Examples, Documentation, Gallery, Download, and GitHub. The main content area features a large 3D visualization of a sphere with a complex internal structure. Below this, there is a 'News' section with four entries dated from February to July 2021. A 'Latest Release' section highlights version 4.3 with a 'Download mfem-4.3.tgz' button. The 'Documentation' section lists links for building MFEM, getting started, and finite elements. A 'Features' section lists several key capabilities like high-order meshes and GPU acceleration. The 'Contact' section provides information on how to report bugs or ask questions.

MFEM is a free, lightweight, scalable C++ library for finite element methods.

Features

- Arbitrary high-order finite element meshes and spaces.
- Wide variety of finite element discretization approaches.
- Conforming and nonconforming adaptive mesh refinement.
- Scalable from laptops to GPU-accelerated supercomputers.
- ...and many more.

MFEM is used in many projects, including BLAST, Cardioid, Visit, RF-SciDAC, FASTMath, xSDK, and CEED in the Exascale Computing Project. See also our Gallery, Publications and News pages.

News

- Jul 29, 2021 Version 4.3 released.
- Jul 10, 2021 MFEM Community Workshop in October.
- Apr 22, 2021 MFEM featured on S&TR magazine cover.
- Feb 16, 2021 New page on GPU performance.

Latest Release

New features | Examples | Code documentation | Sources

Download mfem-4.3.tgz

Older releases | Python wrapper

Documentation

Building MFEM | Getting Started | Finite Elements | Performance

New users should start by examining the example codes.

We also recommend using GLVis for visualization.

Contact

Use the GitHub issue tracker to report bugs or post questions or comments. See the About page for citation information.

Website:
mfem.org

Software:
github.com/mfem

Publications:
mfem.org/publications

Email:
mfem@llnl.gov

- Contact us with questions + feedback
- Contribute to the code
- Explore our publications

mfem.org



Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.