High-Order Solvers in MFEM

Will Pazner + the MFEM team

October 20, 2021

pazner1@llnl.gov

Center for Applied Scientific Computing

Lawrence Livermore National Laboratory

CASC
Center for Applied
Scientific Computing

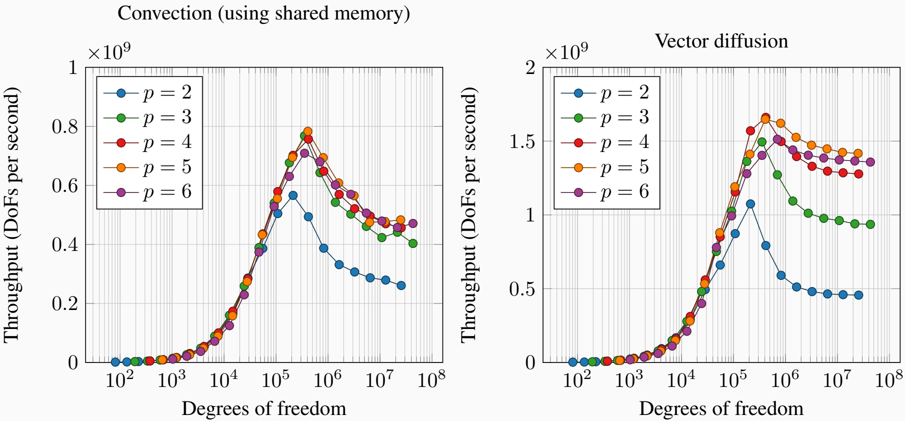
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC. LLNL-PRES-828127.

High-Order Discretizations and Solvers

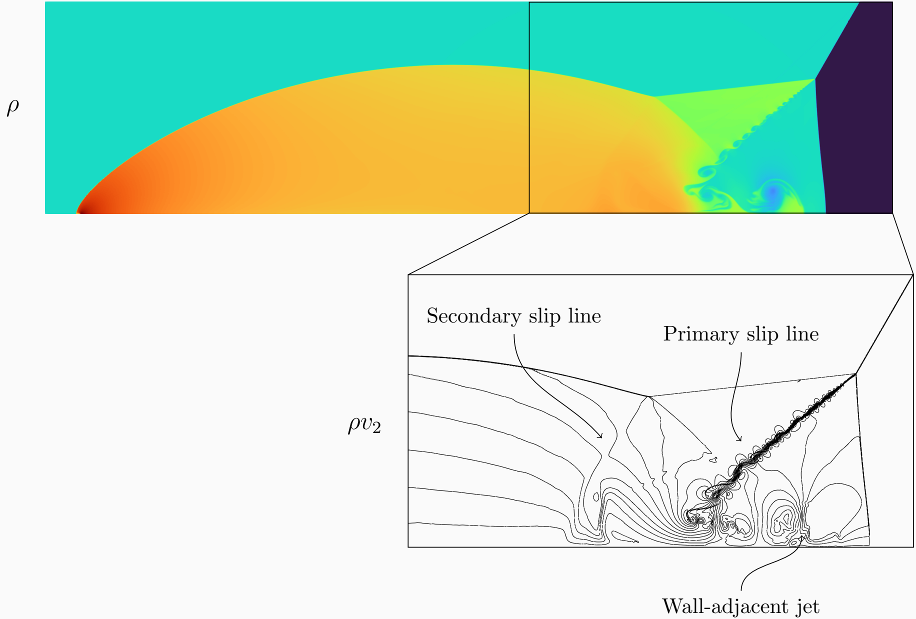
- High-order discretizations present many benefits
  - Very high accuracy on smooth problems
  - With hp-refinement, exponential convergence on problems with singularities
  - Advection-dominated problems: low dispersion, dissipation, better resolution of small-scale features
  - High arithmetic intensity $\implies$ better for GPUs

For example...

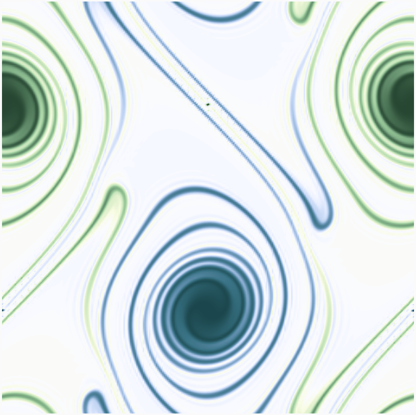
Performant GPU Kernels

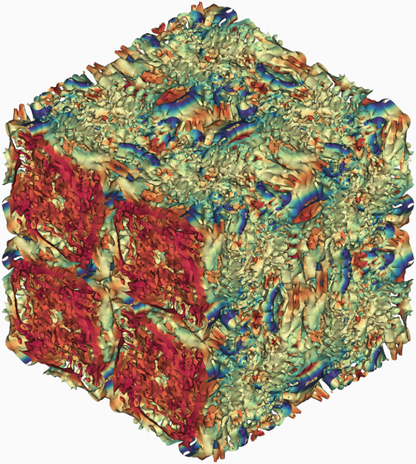


Double Mach Reflection

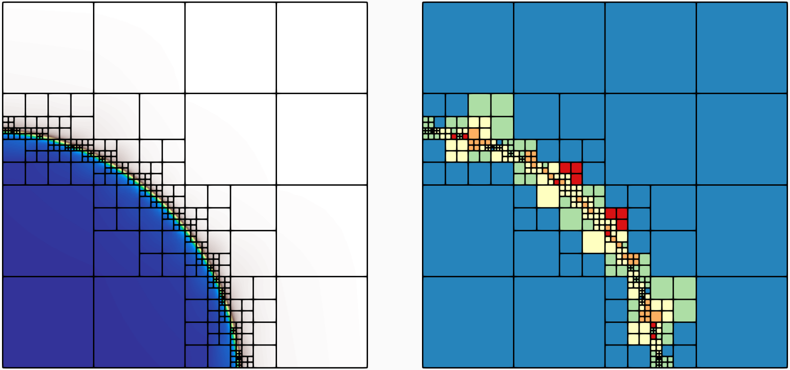


Incompressible Flow





hp-Adaptivity



High-Order Discretizations and Solvers

- High-order discretizations present many benefits
  - Very high accuracy on smooth problems
  - With hp-refinement, exponential convergence on problems with singularities
  - Advection-dominated problems: low dispersion, dissipation, better resolution of small-scale features
  - High arithmetic intensity $\implies$ better for GPUs

But...

High-Order Discretizations and Solvers

- High-order discretizations present many benefits
  - Very high accuracy on smooth problems
  - With hp-refinement, exponential convergence on problems with singularities
  - Advection-dominated problems: low dispersion, dissipation, better resolution of small-scale features
  - High arithmetic intensity $\implies$ better for GPUs

But...

- Difficult to solve the linear systems
  - Poorly conditioned systems ($\kappa \sim p^3/h^2$ or $\kappa \sim p^4/h^2$)
  - Large, dense, element-wise blocks ($\mathcal{O}(p^6)$ nonzeros in 3D)
  - **Expensive** to form/assemble the matrix
  - Classical methods don't always work well

High-Order Solvers in MFEM

MFEM aims to make **efficient solvers** for **high-order** problems **readily available** and **easy to use**

MFEM's Solver Capabilities

Traditional Solver Methods

Iterative methods

- CG, GMRES, FGMRES, MINRES, BiCGSTAB

Sparse direct methods

- UMFPACK, KLU, PARDISO, SuperLU, MUMPS, STRUMPACK

Multigrid and AMG methods

- *hypre*, AMGX, Ginkgo, geometric multigrid

Solvers for High-Order

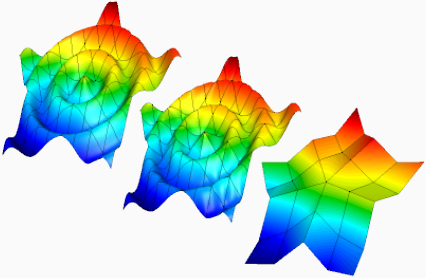
MFEM also provides several specialized high-order solvers

Solvers Designed for High-Order Methods

Solver	Model Problem	Matrix-Free	GPU
hp-multigrid	Diffusion + others...	✓	✓
Low-order refined	Diffusion, curl-curl, grad-div	✓	✓
Matrix-free AMS	curl-curl	✓	✓
Block ILU	Advection-dominated	✗	✗
AIR	Advection-dominated	✗	✗

Geometric multigrid

- Flexible base class `Multigrid`
- BYO:
  - Hierarchy of operators
  - Prolongation operators
  - Smoothers
  - Coarse solvers



- `GeometricMultigrid`
  - Work on a hierarchy of 'FiniteElementSpace' objects
  - h and p refinements supported
- Illustrated in `ex26` and `ex26p`

Matrix-Free Multigrid

- `FiniteElementSpaceHierarchy` (as the name suggests...) manages a hierarchy of finite element spaces
- The user can add h- or p-refined spaces
- Prolongation operators between p-refined spaces are automatically **matrix-free** and **sum-factorized**
- `OperatorJacobiSmoother` and `OperatorChebyshevSmoother` provide **fast**, **matrix-free**, **sum-factorized** access to the diagonal of the high-order operator and associated Chebyshev smoother

Demo

Low-order refined

Main idea

- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh

Low-order refined

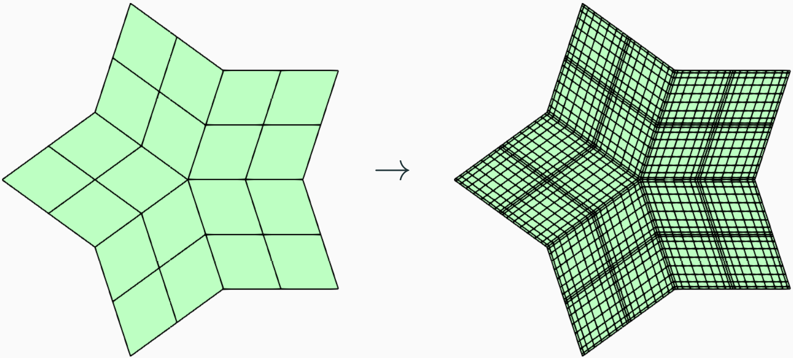
Main idea

- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh
  - Under some assumptions, the two are **spectrally equivalent**
  - Use your favorite classical preconditioner on the LOR system

Low-order refined

Main idea

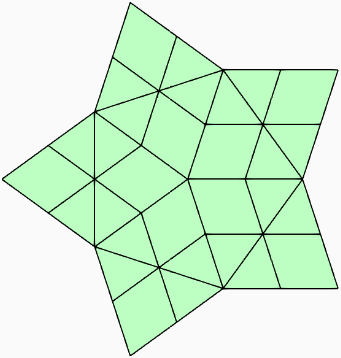
- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh
  - Under some assumptions, the two are **spectrally equivalent**
  - Use your favorite classical preconditioner on the LOR system



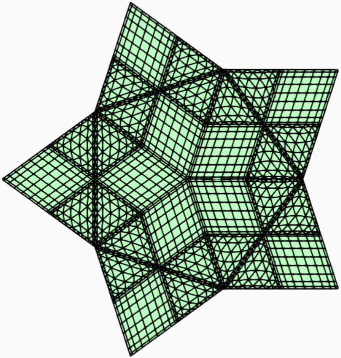
Low-order refined

Main idea

- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh
  - Under some assumptions, the two are **spectrally equivalent**
  - Use your favorite classical preconditioner on the LOR system



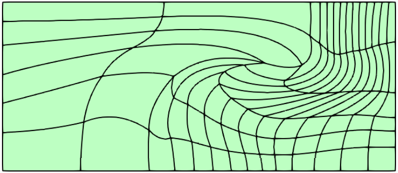
$\rightarrow$



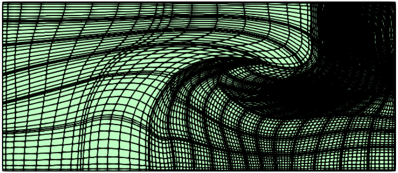
Low-order refined

Main idea

- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh
  - Under some assumptions, the two are **spectrally equivalent**
  - Use your favorite classical preconditioner on the LOR system



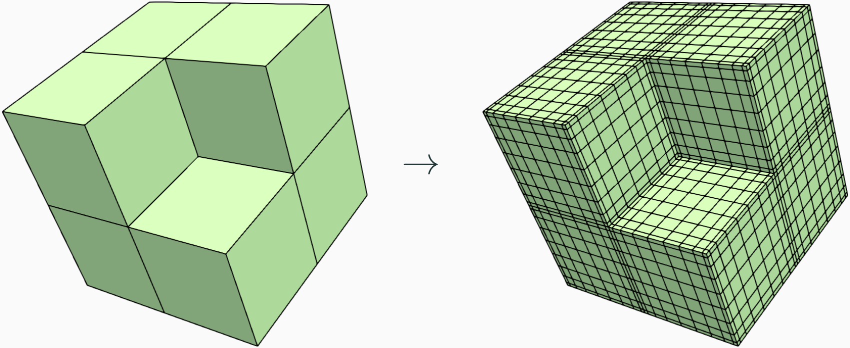
$\rightarrow$



Low-order refined

Main idea

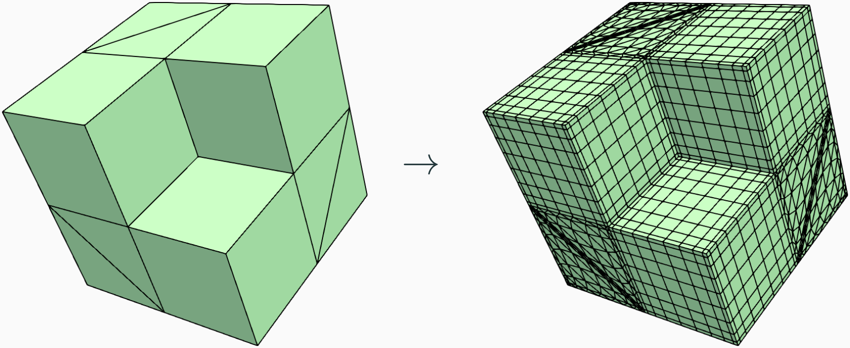
- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh
  - Under some assumptions, the two are **spectrally equivalent**
  - Use your favorite classical preconditioner on the LOR system



Low-order refined

Main idea

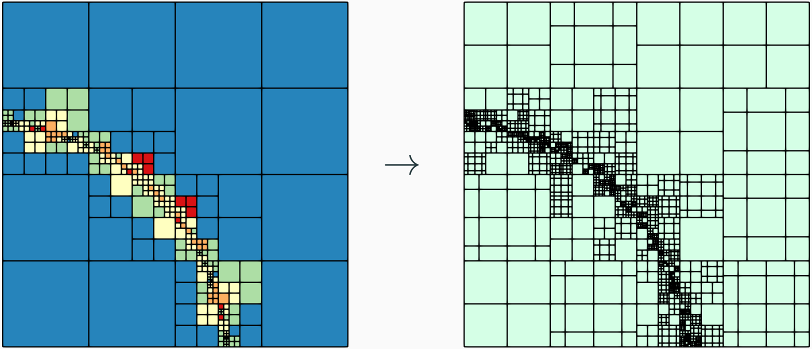
- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh
    - Under some assumptions, the two are **spectrally equivalent**
    - Use your favorite classical preconditioner on the LOR system



Low-order refined

Main idea

- High-order matrix-free operator — **no assembled matrix**!
- Assemble low-order matrix on auxiliary refined mesh
  - Under some assumptions, the two are **spectrally equivalent**
  - Use your favorite classical preconditioner on the LOR system



Low-order refined in MFEM

- MFEM provides the classes `LORDiscretization` and `LORSolver` to help users build these solvers automatically

Low-order refined in MFEM

- MFEM provides the classes `LORDiscretization` and `LORSolver` to help users build these solvers automatically
- `lor_solvers` miniapp demonstrates this capability on:
    - H^1 diffusion
    - Definite Maxwell with $H(\mathrm{curl})$
    - grad-div problem with $H(\mathrm{div})$
    - DG diffusion

Low-order refined in MFEM

- MFEM provides the classes `LORDiscretization` and `LORSolver` to help users build these solvers automatically
- `lor_solvers` miniapp demonstrates this capability on:
  - H^1 diffusion
  - Definite Maxwell with $H(\mathrm{curl})$
  - grad-div problem with $H(\mathrm{div})$
  - DG diffusion
- Parallel example `plor_solvers` provides matrix-free AMG solvers for these problems
  - BoomerAMG, AMS, ADS

Low-order refined in MFEM

- MFEM provides the classes `LORDiscretization` and `LORSolver` to help users build these solvers automatically
- `lor_solvers` miniapp demonstrates this capability on:
  - H^1 diffusion
  - Definite Maxwell with $H(\mathrm{curl})$
  - grad-div problem with $H(\mathrm{div})$
  - DG diffusion
- Parallel example `plor_solvers` provides matrix-free AMG solvers for these problems
  - BoomerAMG, AMS, ADS
- One-liners:
  - `LORSolver<UMFPackSolver> solv_lor(a, ess_dofs);`
  - `LORSolver<HypreBoomerAMG> solv_lor(a, ess_dofs);`

Demo

Advection-dominated problems

- More challenging (from a methods point of view) than elliptic/parabolic problems
- Most robust options are currently **matrix-based**
    - Block ILU(0) with MDF ordering
    - Approximate Ideal Restriction (AIR) AMG
- Both illustrated in `ex9p`
- Both methods can perform “sweeps”, i.e. converge immediately if there is a perfect (triangular) ordering (no cycles)

Demo

- Questions?
- Feedback?
- Suggestions?