Parallel Linear Solvers for Simulations of Reactor Thermal Hydraulics

Y. Yan[1], S.P. Antal[2], B. Edge[2], D.E. Keyes[1], D. Shaver[2], I.A. Bolotnov[3] and M.Z. Podowski[2]

[1]Columbia University, New York, NY, USA

[2]Rensselaer Polytechnic Institute, Troy, NY, USA

[3]North Carolina State University, Raleigh, NC, USA

Rensselaer

Columbia University

NC STATE UNIVERSITY

Fission and fusion provide carbon-neutral energy

- However, the price we have paid so far for fission is high.
- Part of the high cost is from nuclear accidents.
- Severe nuclear accidents may eventually lead to nuclear core meltdown.
- Understanding the reactor core failure via simulation (and not via reality ☺) is critical to enhance the safety of Generation IV reactors.

NPHASE-CMFD Solver

- Computational Multiphase Fluid Dynamics solver
- Unstructured grids with arbitrary element types
- Capable of modeling an arbitrary number of fields (fluid components and/or phases)
- Has built-in and user defined mechanistic modeling, integrated with numerics
- Characterized by improved robustness and numerical convergence for two-phase flows
- Can be used to model gas/liquid interfaces using level set method
- Uses state-of-the-art multiphase models which have been extensively validated

Where We Need Improved Solver?

- For CFD-based simulations, the linear solver is often the bottleneck for scaling to large number of processors for more detailed resolution.

1. Invert an implicit operator (e.g., ADI)
2. Solve for a Newton update with the Jacobian (delta form)
3. Apply sophisticated physics-based preconditoner, i.e. using a lower order or coarser discretization as the preconditioner
4. Solve the primal and dual system for a control or inverse problem

- Reasons for performance degradation

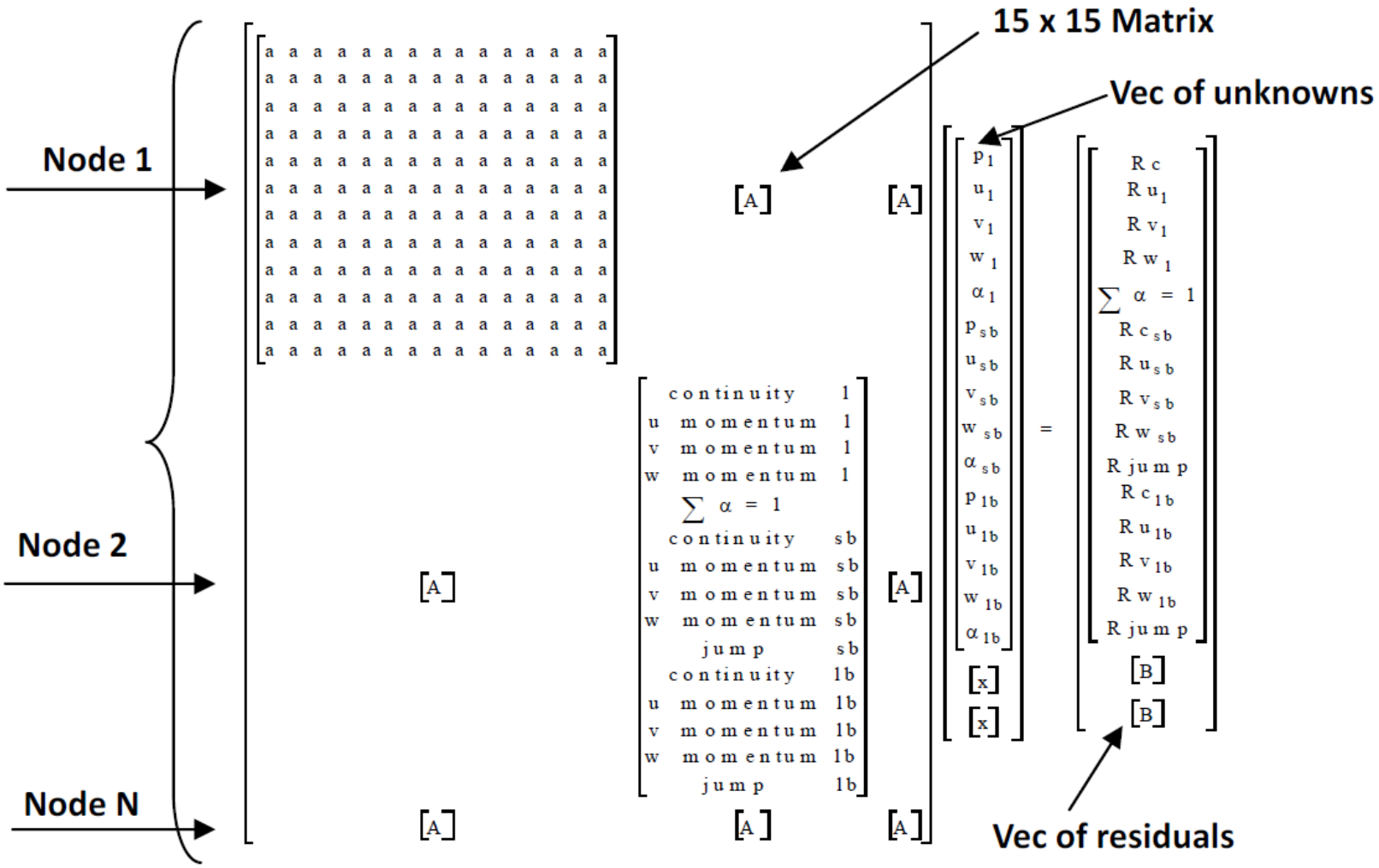
  Linear solver time significantly increases with mesh refinement path as work per iteration grows superlinearly and convergence tends to degrade

- The purpose of the current work is to efficiently solve the Newton system, $A\delta x^n = b^n$, where $\delta x^n = \{u', v', w', p', a'\}$

Current System

- RANS simulations of turbulent flow inside large computational domains at macro-scale.
  - We write the nonlinear steady state conservation law system as $F(x) = 0$
  - After using pseudo-timestepping, we obtain Newton system, $A\delta x^n = b^n$ A is the Jacobian, b is the residual, δx^n is the Newton update.
  - After solving for δx^n, we set $x^{n+1} = x^n - \omega \delta x^n$, where $\delta x^n = \{u', v', w', p', a'\}$

Structure of Newton System



Solution Approach

- Use Preconditioned Krylov iterative method…

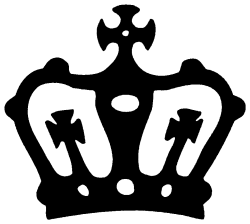
1. Krylov iterative method is robust if workspace can be allowed for the vectors
2. Krylov methods have kernel as a sparse matrix-vector multiply that scales well
3. Krylov iterative method can be combined with various preconditioners from "brute force" algebraic to "physics-based": **ilu, jacobi, sor, domain decomposition, geometric multigrid, algebraic multigrid, physics-based preconditioner, and user designed**
4. For **difficult** problems like multiphase fluid simulation, it is essential to combine Krylov method with powerful preconditioner to accelerate the reduction of low frequency error.
5. An unstructured mesh forces us to use an Algebraic Multigrid Preconditioner
6. Both the parallel implementation of Krylov iterative method and preconditioning method are freely available from **PETSc** and **Hypre**

PETSc and Hypre are part of the U. S. DOE TOPS* project

Mission: enable scientists and engineers to take full advantage of petascale hardware by overcoming the scalability bottlenecks of traditional solvers

Lawrence Livermore National Laboratory

Sandia National Laboratories



Columbia University

University of Colorado

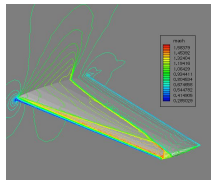
University of Texas

University of California at San Diego

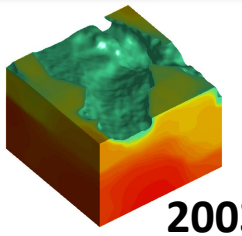
* TOPS (2001-2011) has been succeeded by FASTMath (2011-2016) in SciDAC-3.

TOPS software has taken science applications to the architectural leading edge

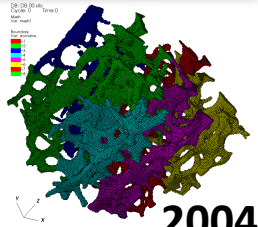
- **TOPS software is at the heart of three Gordon Bell "Special" Prizes**



1999 fluids

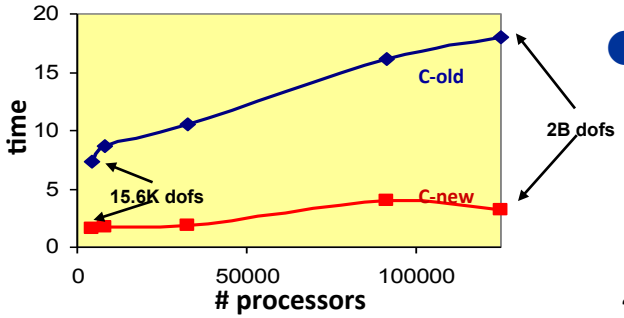


2003 seismic



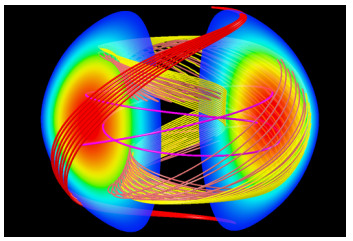
2004 mechanics

After new coarsening algorithm (red), nearly flat scaled speedup for Algebraic Multigrid



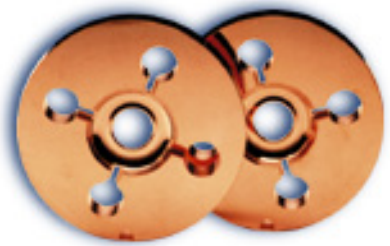
- **Scales to the edge of BlueGene (131,072 processors, 2B unknowns)**

~5X speedup of plasma fusion codes through linear solver replacement – like providing "next generation" computer



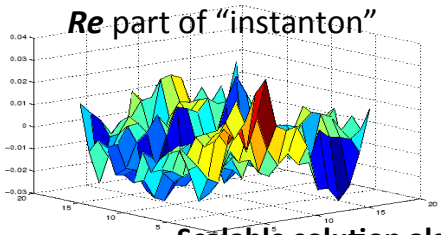
magneto-hydro-dynamics

- **Enabled numerous physics attainments in SciDAC**

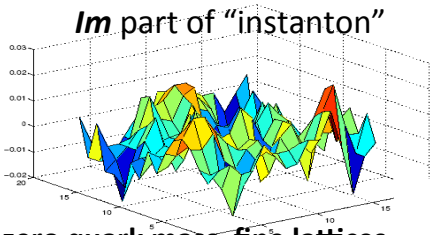


accelerator design

Prototype shape optimization capability



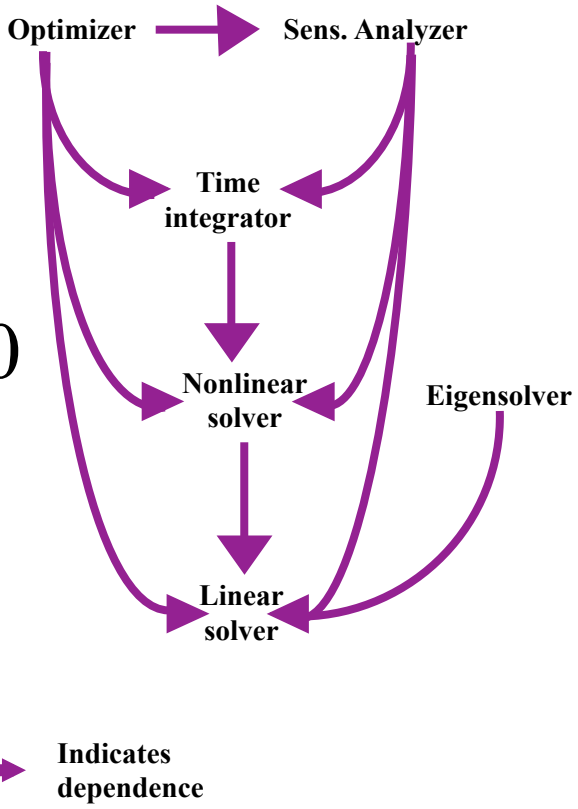
QCD



Scalable solution algorithm for zero quark mass, fine lattices

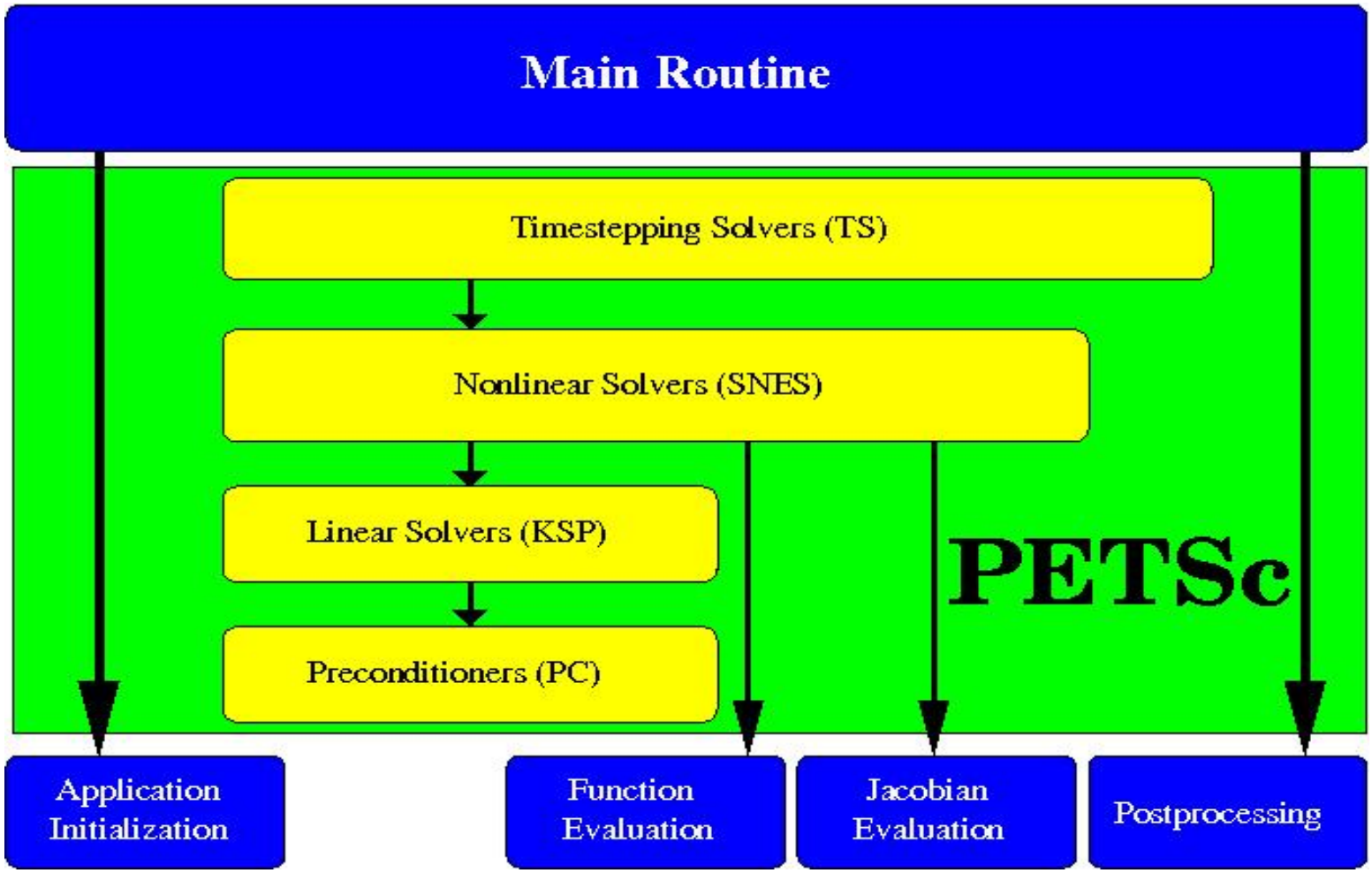
Scope for scalable solver software

- Five types of generalized "solvers"
  - Time integrators (w/ sens. anal.) $f(\dot{x}, x, t, p) = 0$
  - Nonlinear solvers (w/ sens. anal.) $F(x, p) = 0$
  - Constrained optimizers $\min_u \phi(x,u)\ s.t.\ F(x,u) = 0, u \geq 0$
  - Linear solvers $Ax = b$
  - Eigensolvers $Ax = \lambda Bx$
- Software integration
- Performance optimization

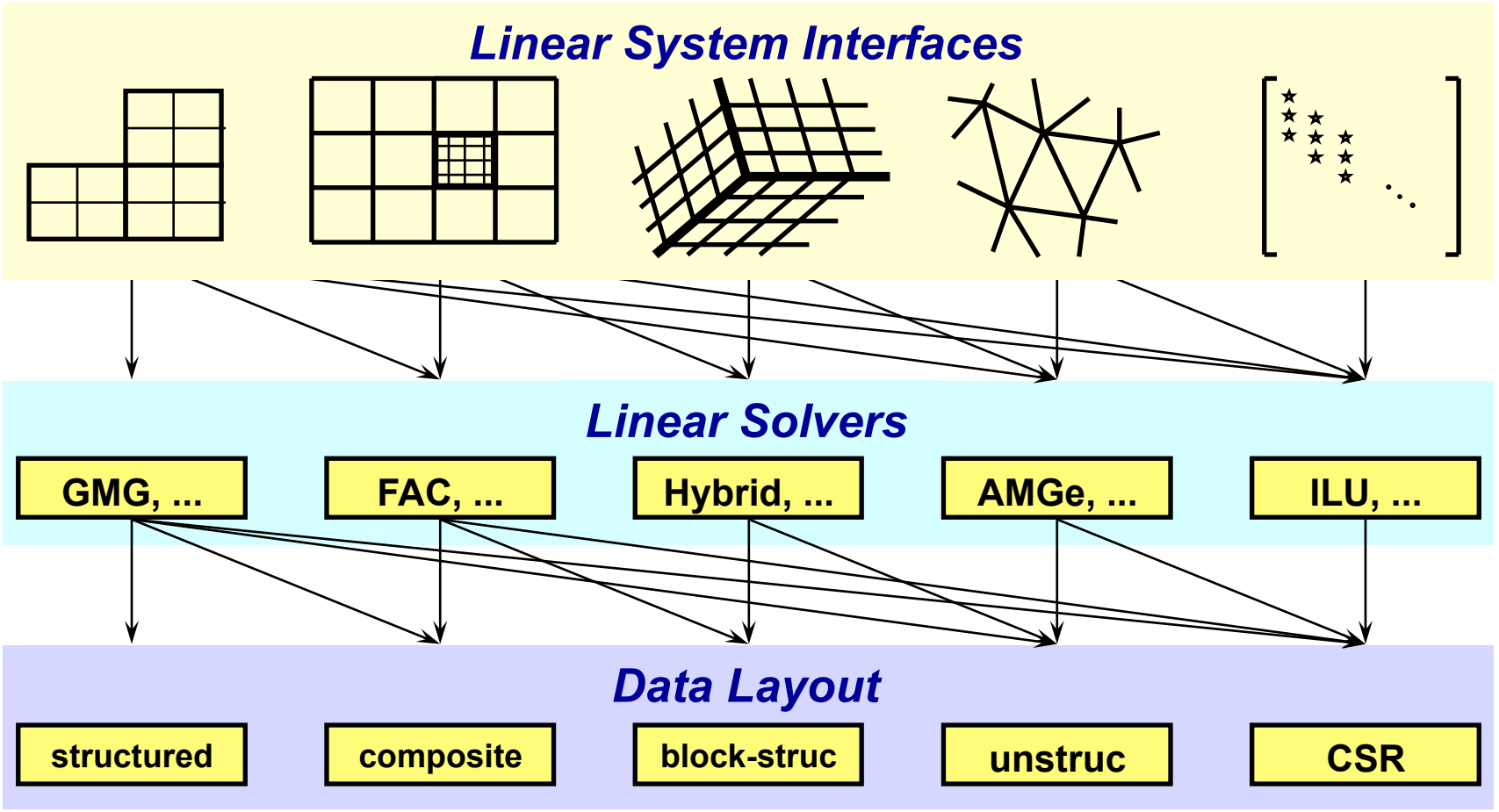


PETSc user code/library interactions

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

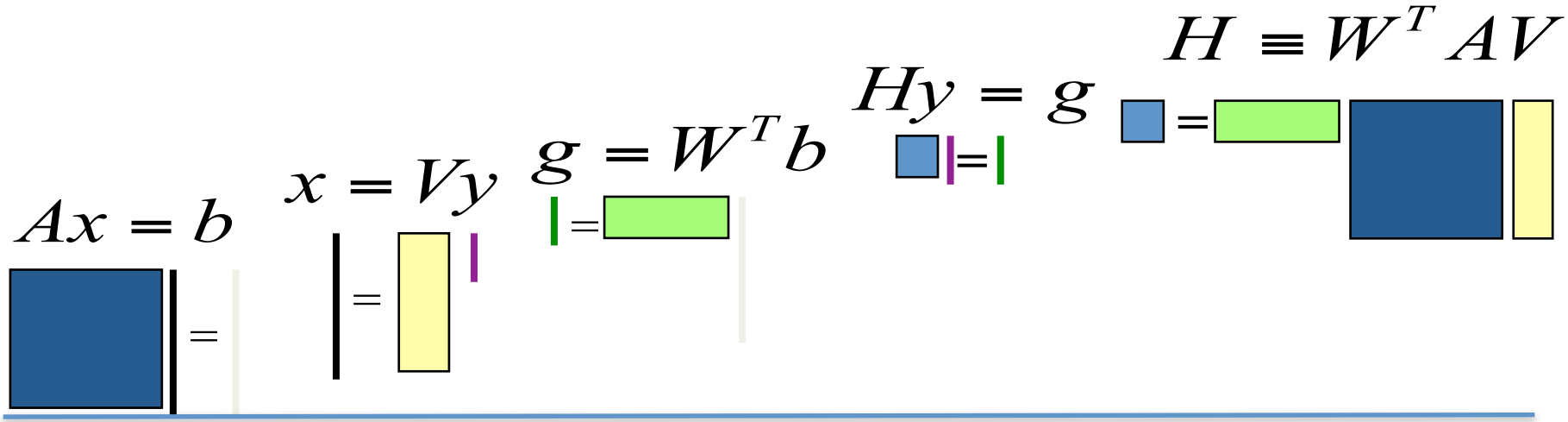


Hypre preconditioner interfaces



Krylov bases for sparse systems

- E.g., conjugate gradients (CG) for symmetric, positive definite systems, and generalized minimal residual (GMRES) for nonsymmetry or indefiniteness
- Krylov iteration is an algebraic projection method for converting a high-dimensional linear system into a lower-dimensional linear system.



Convergence estimates

Symmetric case: $Ax = f$.

Theorem. If A has at most s distinct eigenvalues, CG converges in at most s steps.

Theorem. Let $e_k \equiv x_k - x^*$, where x^* is the exact solution, and let $\kappa \equiv \lambda_{\max}(A)/\lambda_{\min}(A)$. Then

$$||e_k||_A \leq 2\left(\frac{1 - 1/\sqrt{\kappa}}{1 + 1/\sqrt{\kappa}}\right)^k ||e_0||_A \ .$$

Nonsymmetric case: $Ax = f$.

Theorem. If A has at most s distinct eigenvalues, GMRES converges in at most s steps.

Theorem. Let $r_k \equiv f - Ax_k$, $C_p^2 = \lambda_{\max}(A^T A)$, and $c_p = \lambda_{\min}(\frac{1}{2}(A^T + A))$. Then

$$||r_k|| \leq \left(1 - \frac{c_p^2}{C_p^2}\right)^{k/2} ||r_0|| \ .$$

Krylov solver software

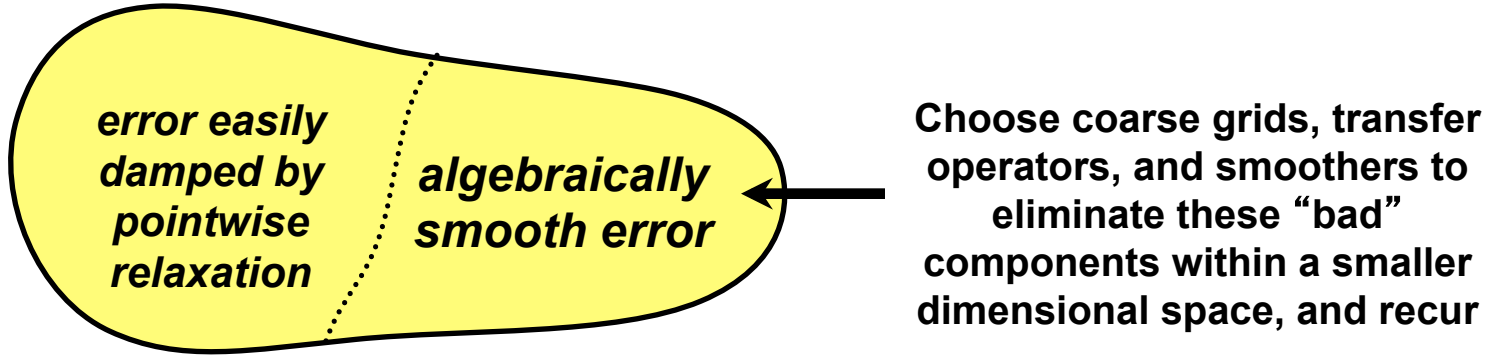
- There is a large variety of Krylov solvers
  - customized to mathematical properties like symmetry or definiteness
  - customized to architectural properties like synchronization cost, memory capacity, communication to computation capabilities
- All Krylov solvers users have found practical are selectable at runtime in libraries like PETSc in TOPS
  - users may register their own Krylov solvers if their favorite variant is missing
- Preconditioning is essential for most applications (see previous slide) to shrink the spectrum into a relatively small number of clumps, each of which requires one Krylov iteration
  - Hypre (called by PETSc) has a number of multilevel preconditioners

Generalization: geometric multigrid

- Geometric multigrid has done well on *structured, homogeneous, Isotropic* problems
- Algebraic multigrid (AMG) has done well with *unstructured*, *inhomogeneous*, and *anisotropic* problems and adaptive AMG (αAMG) has done well with *indefinite* problems
- More work is needed to obtain $O(N)$ in highly *symmetric, multicomponent* problems and for *high-order alternating sign discretizations*

Error modes (think Fourier modes)

AMG Framework

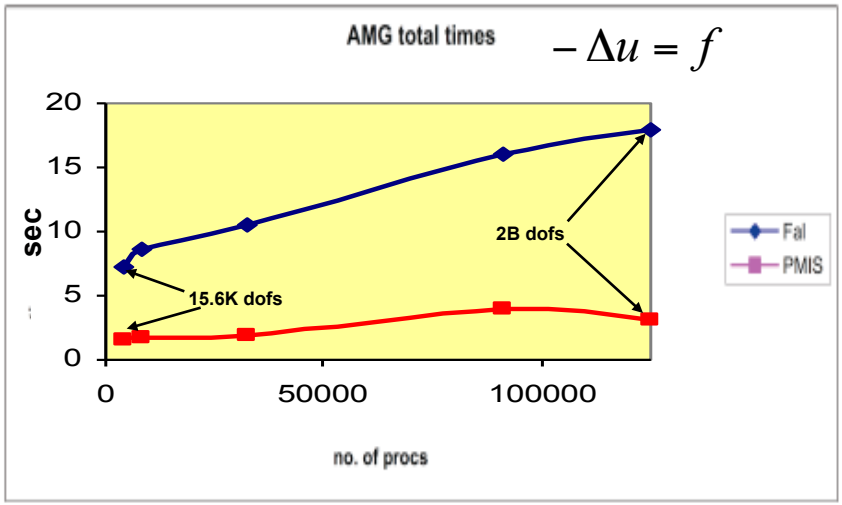


Sparse iterative solvers are scaling: algebraic multigrid (AMG) on BG/L (hypre)

- Algebraic multigrid a key and very general algorithmic technology
  - Discrete operator defined for finest grid by the application, itself, *and* for many recursively derived levels with successively fewer degrees of freedom, for solver purposes
  - Unlike geometric multigrid, AMG not restricted to problems with "natural" coarsenings derived from grid alone
- Optimality (cost per cycle) intimately tied to the ability to coarsen aggressively
- Convergence scalability (number of cycles) and parallel efficiency also sensitive to rate of coarsening

While much research and development remains, multigrid is practical at petascale concurrency (hundreds of thousands of processors)

Figure shows weak scaling result for AMG out to 120K processors, with one 25×25×25 block per processor (up to 1.875B dofs)



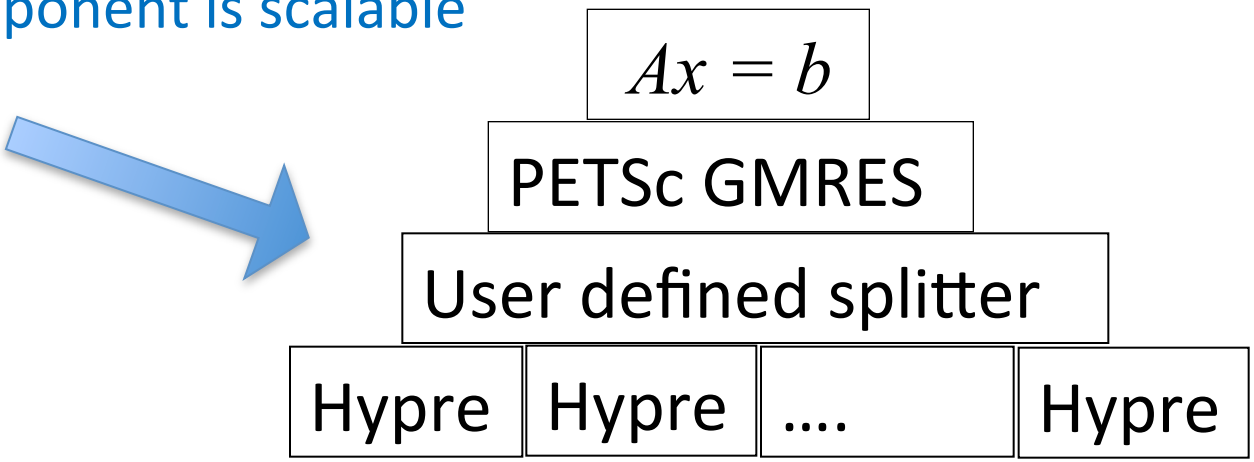
7-pt Laplacian, total execution time, AMG-CG, total problem size ~2 billion

c/o U. M. Yang, LLNL (TOPS)

PETSc and Hypre are extensible toolkits

- Does (GMRES from PETSc)+(AMG from Hypre) solve our problem?
  - *No, because AMG is not yet perfected for multicomponent problems*
- What to do? PETSc allows the user to define split preconditioners
- This does the trick

Each component is scalable



Split preconditioner

- The most basic idea in iterative methods for $Ax = b$ is
$$x \leftarrow x + B^{-1}(b - Ax)$$
- Evaluate residual accurately, but solve approximately, where B^{-1} is an approximate inverse to A
- A sequence of complementary solves can be used, e.g., with first B_1 and then B_2 one has
$$x \leftarrow x + [B_1^{-1} + B_2^{-1} - B_2^{-1} A B_1^{-1}](b - Ax)$$
- B_1 can represent dense diagonal coupling blocks and B_2 sparse scalar solves for each component
- Scale recurrence, e.g., with $B_2^{-1} = R^T (RAR^T)^{-1} R$, where R is a restriction operator taking fine to coarse representations, leads to *multilevel methods*

More details on the splitter

- Consider n-phase fluids, the parallel block matrix will have the following index set for 1 Finite Volume cell:

$$\delta x^n = \{1,2,3,4,5\ldots 5n-4, 5n-3, 5n-2, 5n-1, 5n\}$$

- We define the following splitting strategies:
- Full Split

$$\delta x^n = \{\{1\},\{2\},\{3\},\ldots\{5n-3\},\{5n-2\},\{5n-1\},\{5n\}\}$$

- Partial Split-1

$$\delta x^n = \{\{1,2,3,4\},\{5\}\ldots\{5n-4,5n-3,5n-2,5n-1\},\{5n\}\}$$

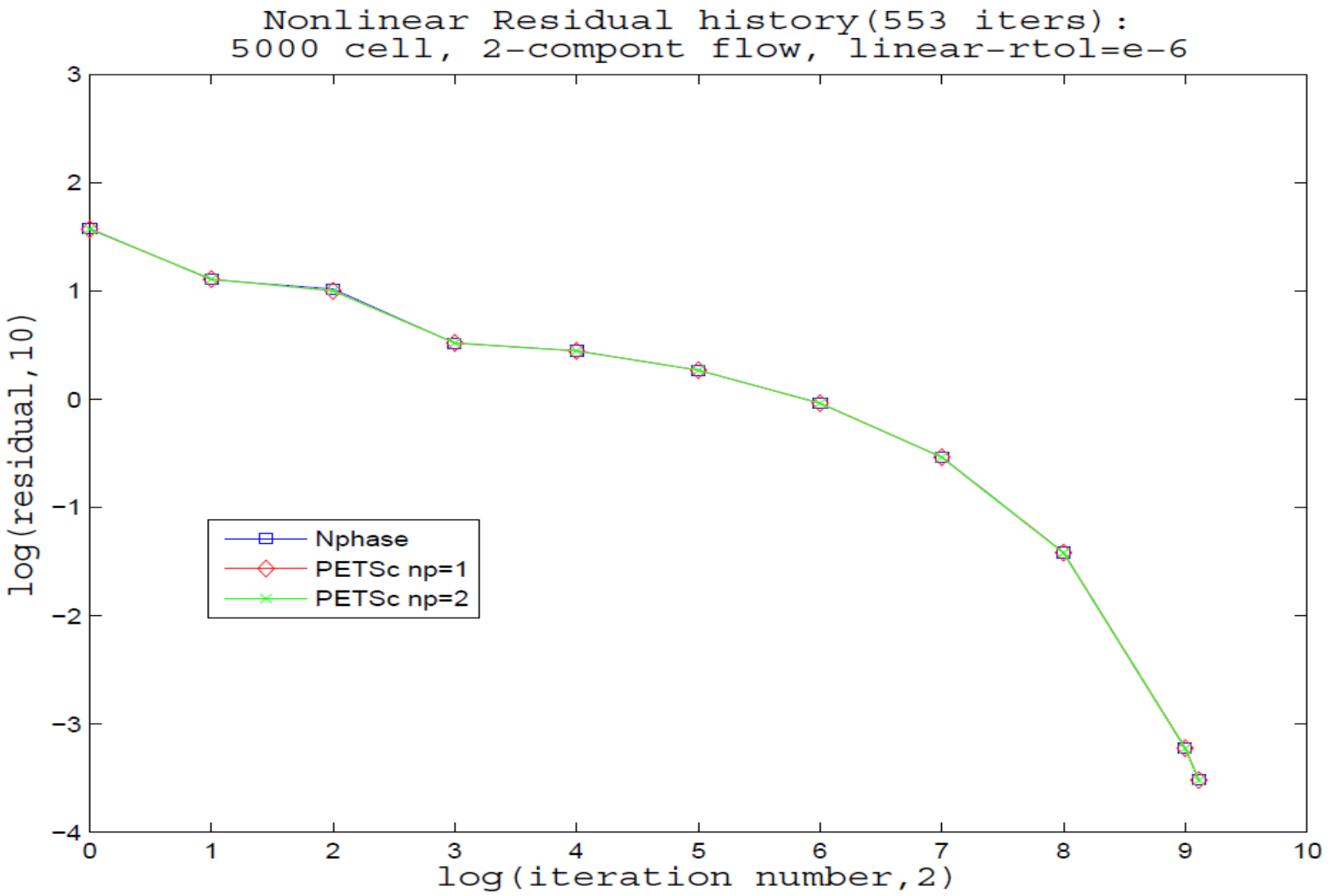
- Partial Split-2

$$\delta x^n = \{\{1\},\{2,3,4\},\{5\}\ldots\{5n-4\},\{5n-3,5n-2,5n-1\},\{5n\}\}$$

- Phase split

$$\delta x^n = \{\{1,2,3,4,5\}\ldots\{5n-4,5n-3,5n-2,5n-1,5n\}\}$$

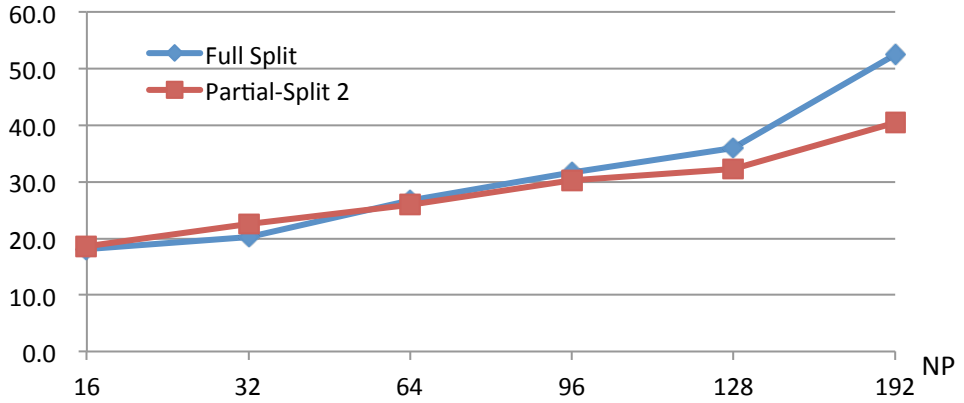
Solver Verification



Solver Scaling

Single-phase flow in the reactor rod geometry with full-splitting and partial-splitting-2 strategies.

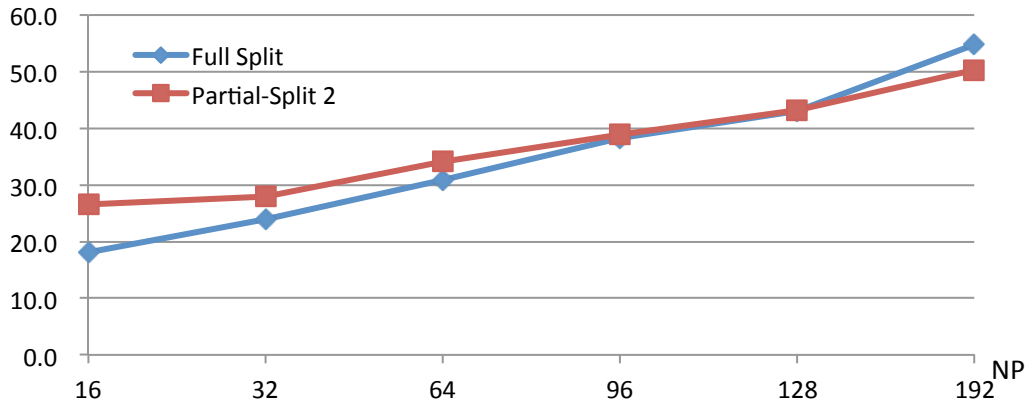
# of Mesh Cells	NP	Full-splitting		Partial-splitting 2	
		Iter. Count	Wall Time	Iter. Count	Wall Time
1.5e6	16	6	1.81e+01	6	1.86e+01
3e6	32	6	2.03e+01	6	2.25e+01
6e6	64	6	2.67e+01	6	2.59e+01
9e6	96	6	3.16e+01	6	3.02e+01
12e6	128	6	3.59e+01	6	3.23e+01
18e6	192	6	5.25e+01	6	4.04e+01



Solver Scaling

Two-phase flow in the reactor rod geometry with full-splitting and partial-splitting-2 strategies.

# of Mesh Cells	NP	Full-splitting		Partial-splitting 2	
		Iter. Count	Wall Time	Iter. Count	Wall Time
3.75e5	16	20	1.81e+01	20	2.65e+01
7.5e5	32	20	2.40e+01	20	2.80e+01
1.5e6	64	20	3.09e+01	20	3.41e+01
2.25e6	96	20	3.82e+01	20	3.88e+01
3e6	128	20	4.30e+01	20	4.32e+01
4.5e6	192	20	5.47e+01	20	5.02e+01



Summary and Future Work

- Solver scalability and robustness is critical for large scale multiphase fluid simulation.

1. To identify a bottleneck, profiling the code is important
2. If it is on linear solver, then a scalable implementation (e.g., Krylov) is important; in this work, we used GMRES
3. Very often, the Krylov solver itself needs a preconditioner to cluster the matrix spectrum for good convergence
4. Choosing a scalable preconditioner is also critical. For more options, see:

5. Applications might need custom preconditioning tuning, too

- Future work

Improving the scalability, and accelerate the nonlinear convergence