Lawrence Livermore National Laboratory

Scalable Multigrid Methods

INT Exascale Workshop, Seattle, WA
June 27 - July 1, 2011

Robert D. Falgout

Center for Applied Scientific Computing

This work performed under the auspices of the U.S. Department of Energy by
Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344

LLNL-PRES-459118

Outline

- Motivation / Background
- Basic Multigrid

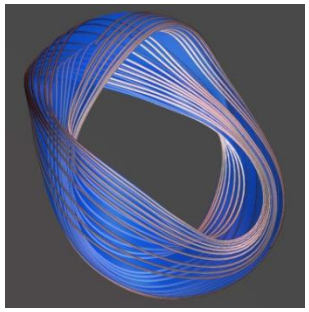
- Parallel Multigrid
- Parallel Algebraic Multigrid
- Multigrid Software Design and Development (*hypre*)

- AMG for Electromagnetic Problems
- Adaptive AMG

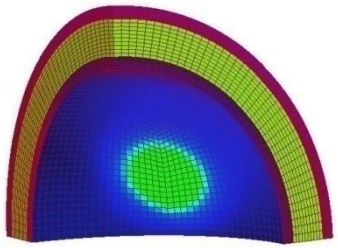
- Summary

The scalable solution of linear systems is crucial in many large-scale simulations

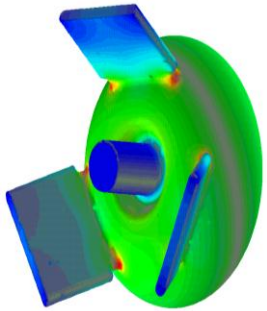
- The solution of linear systems is at the core of many scientific simulation codes



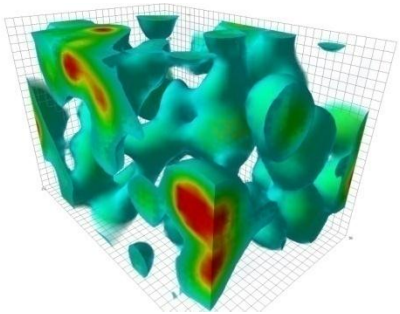
Magnetohydrodynamics



Elasticity / Plasticity



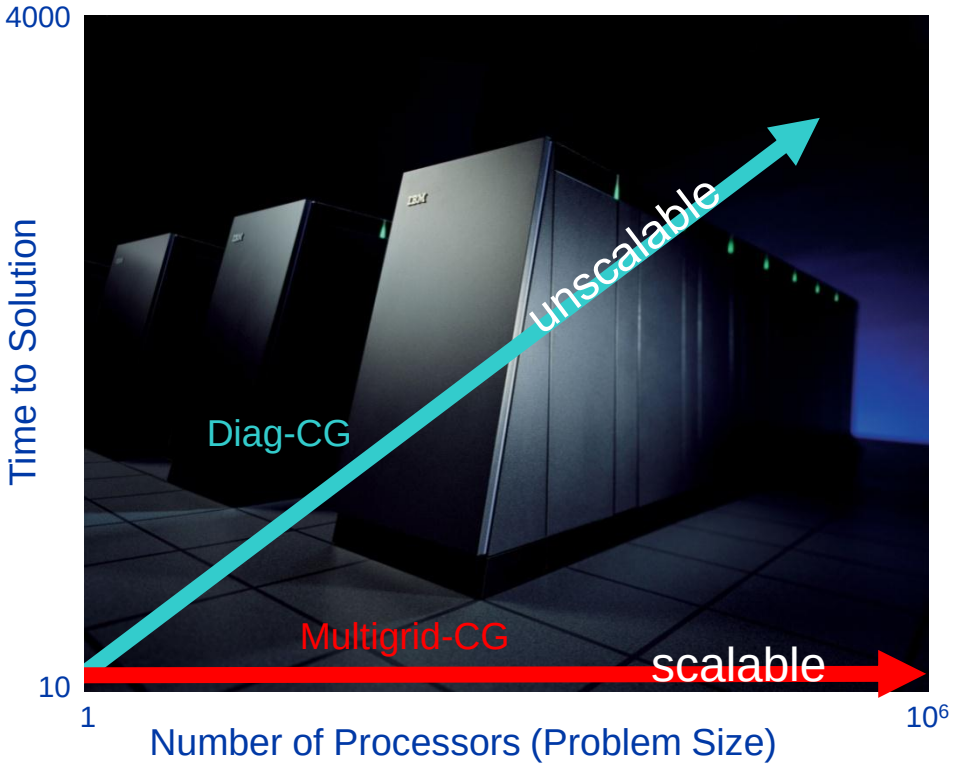
Electromagnetics



Quantum Chromodynamics

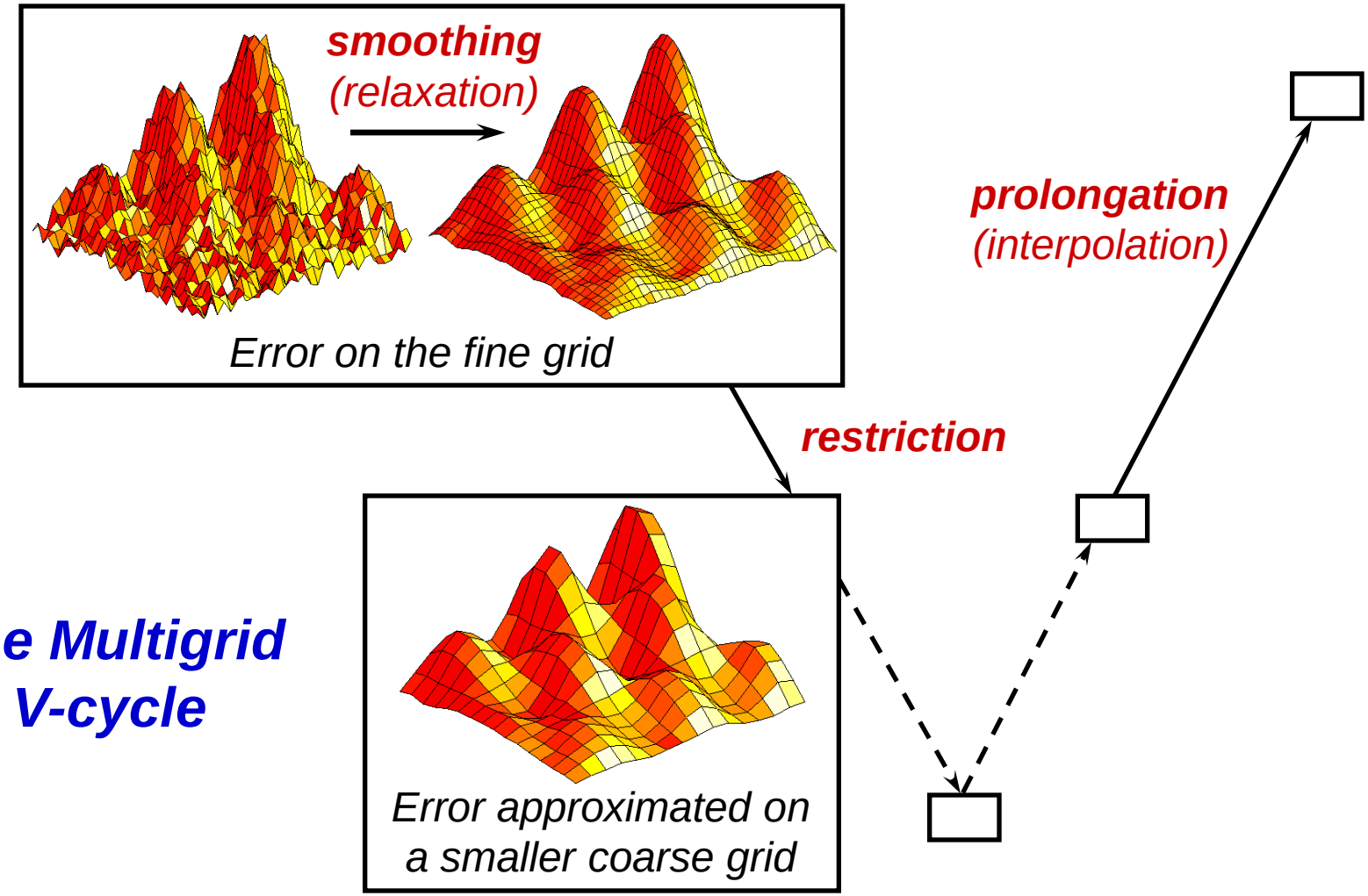
- High fidelity requires huge linear systems and large-scale (e.g., petascale) computing
- We are developing parallel *multigrid* linear solvers and software (*hypre*), driven by applications

Multigrid linear solvers are optimal ($O(N)$ operations), and hence have good scaling potential



- Weak scaling – want constant solution time as problem size grows in proportion to the number of processors

Multigrid uses a sequence of coarse grids to accelerate the fine grid solution



The Multigrid V-cycle

The basic multigrid research challenge

- Optimal O(N) multigrid methods don't exist for some applications, even in serial
- Need to invent methods for these applications

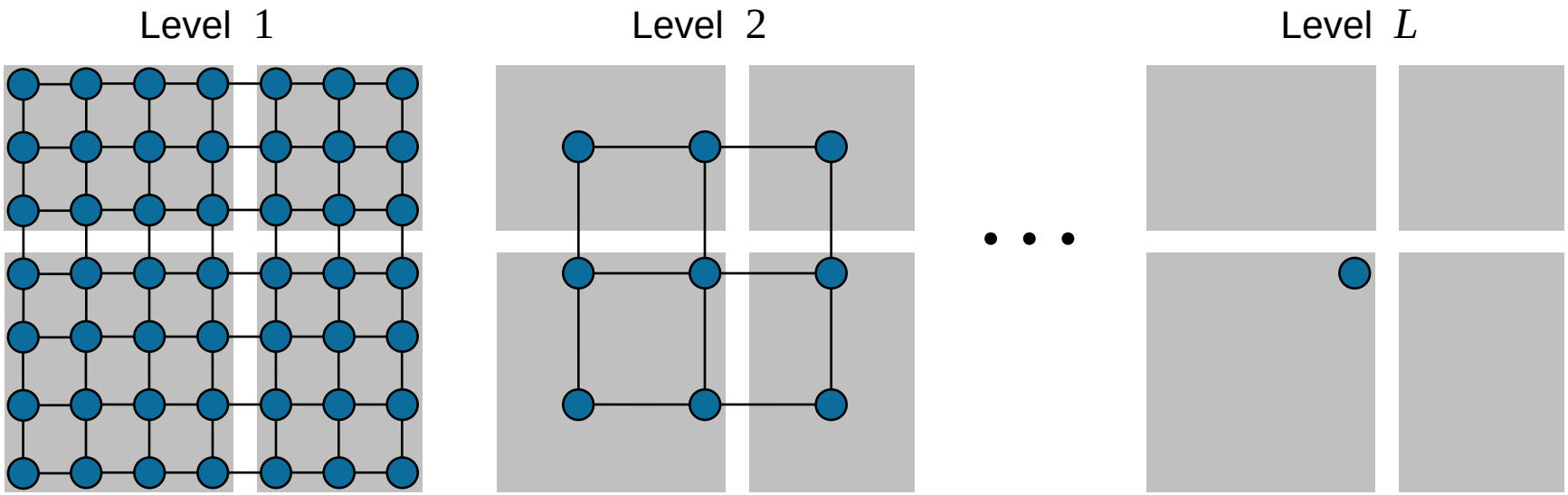
- However …

- Some of the classical and most proven techniques used in multigrid methods don't parallelize
  - Gauss-Seidel smoothers are inherently sequential
  - W-cycles have poor parallel scaling
- Parallel computing imposes additional restrictions on multigrid algorithmic development

- Tomorrow's exascale computers with huge core counts and small memories just magnifies the challenge

Parallel Multigrid

Approach for parallelizing multigrid is straightforward data decomposition



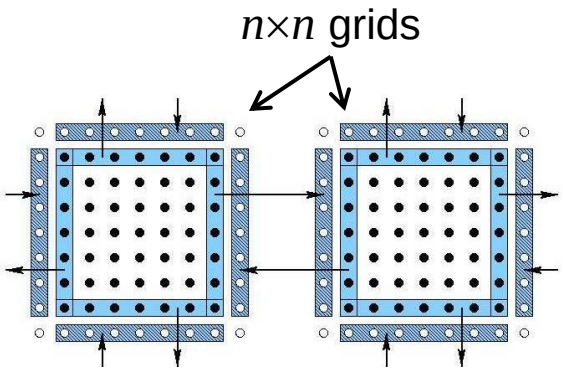
- Basic communication pattern is “nearest neighbor”
  - Relaxation, interpolation, & Galerkin not hard to implement
- Different neighbor processors on coarse grids
- Many idle processors on coarse grids (100K+ on BG/L)
  - Algorithms to take advantage have had limited success

Straightforward parallelization approach is optimal for V-cycles on structured grids (5-pt Laplacian example)

- Standard communication / computation models

$$T_{comm} = \alpha + m\beta \quad \textit{(communicate m doubles)}$$

$$T_{comp} = m\gamma \quad \textit{(compute m flops)}$$



- Time to do relaxation

$$T \approx 4\alpha + 4n\beta + 5n^2\gamma$$

- Time to do relaxation in a V(1,0) multigrid cycle

$$\begin{aligned} T_V &\approx (1+1+\cdots)4\alpha + (1+1/2+\ldots)4n\beta + (1+1/4+\ldots)5n^2\gamma \\ &\approx (\log N)4\alpha + (2)4n\beta + (4/3)5n^2\gamma \end{aligned}$$

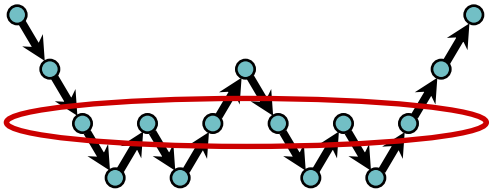
- For achieving optimality in general, the *log* term is unavoidable!

- More precise: $T_{V,better} \approx T_V + (\log P)(4\beta + 5\gamma)$

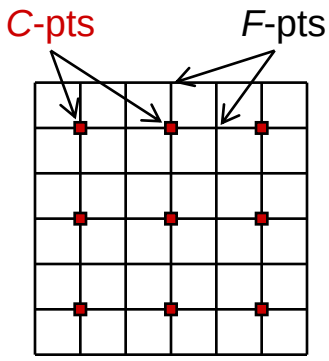
Additional comments on parallel multigrid

- W-cycles scale poorly:

$$T_W \approx (2^{\log N})4\alpha + (\log N)4n\beta + (2)5n^2\gamma$$

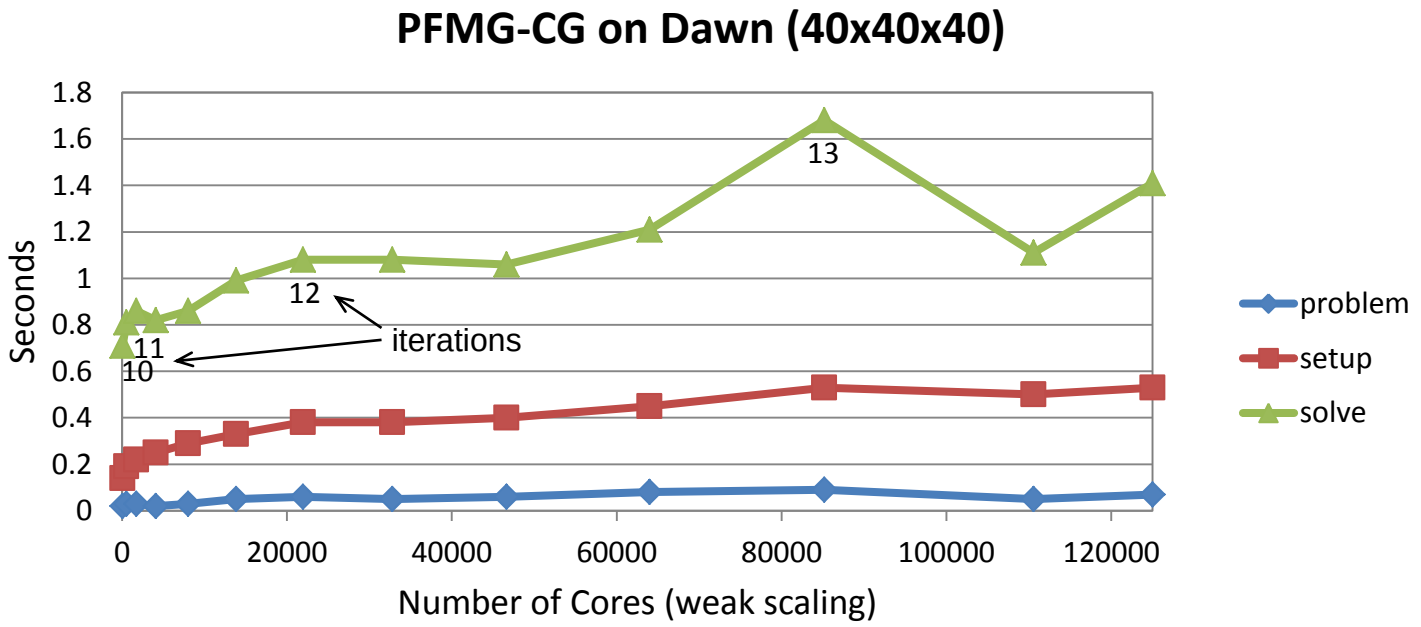


- Lexicographical Gauss-Seidel is too sequential
  - Use red/black or multi-color GS
  - Use weighted Jacobi, hybrid Jacobi/GS, L1
  - Use *C*-*F* relaxation (Jacobi on *C*-pts then *F*-pts)
  - Use Polynomial smoothers
- Parallel smoothers are often less effective



- Recent survey on parallel multigrid:
  - "A Survey of Parallelization Techniques for Multigrid Solvers," Chow, Falgout, Hu, Tuminaro, and Yang, *Parallel Processing For Scientific Computing*, Heroux, Raghavan, and Simon, editors, SIAM, series on Software, Environments, and Tools (2006)
- Recent paper on parallel smoothers:
  - "Multigrid Smoothers for Ultra-Parallel Computing," Baker, Falgout, Kolev, and Yang, *SIAM J. Sci. Comput.*, to appear.

Example weak scaling results on Dawn (an IBM BG/P system at LLNL) in 2011



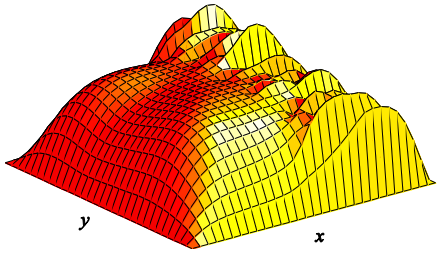
- Laplacian on a cube; 40^3 = 64K grid per processor; largest had 8 billion unknowns
- PFMG is a semicoarsening multigrid solver in *hypre*
- Constant-coefficient version - 1 trillion unknowns on 131K cores in 83 seconds
- Still room to improve setup implementation (these results already employ the assumed partition algorithm described later)

Parallel Algebraic Multigrid (AMG)

Algebraic Multigrid (AMG) is based on MG principles, but uses matrix coefficients

- Many algorithms (AMG alphabet soup)
- Automatically coarsens “grids”

- Error left by pointwise relaxation is called algebraically smooth error
  - Not always geometrically smooth



- Weak approximation property: interpolation must interpolate small eigenmodes well

$$\|E_{TG}\|_A^2 \leq 1 - \frac{1}{K}; \quad K = \sup_e \|A\| \frac{\|(I - P[0\ I])e\|_2}{\|e\|_A^2}$$

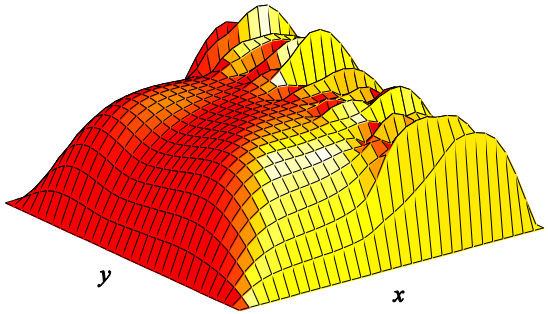
- Near null-space is important!

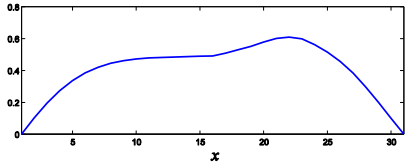
Error left by relaxation can be geometrically oscillatory

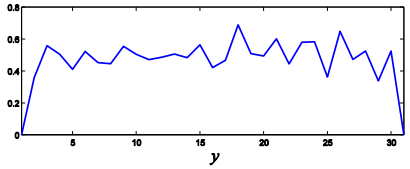
- 7 GS sweeps on

$$-au_{xx} - bu_{yy} = f$$

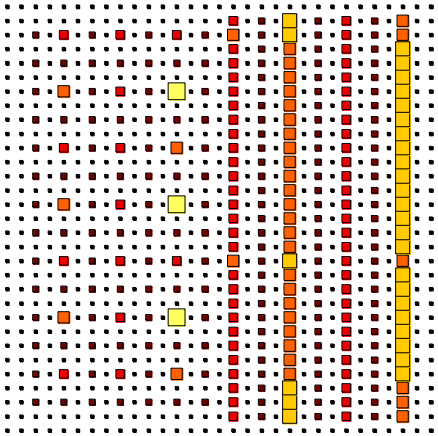
$a = b$	$a \gg b$







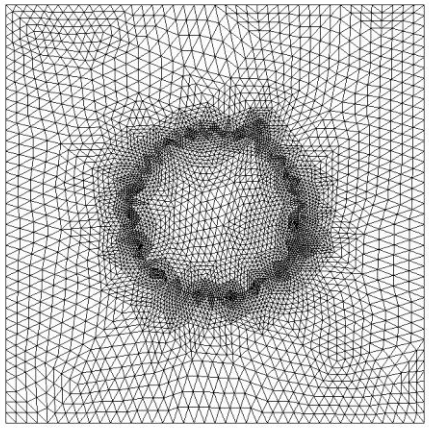
- This example…
  - targets geometric smoothness
  - uses pointwise smoothers
- Not sufficient for some problems!

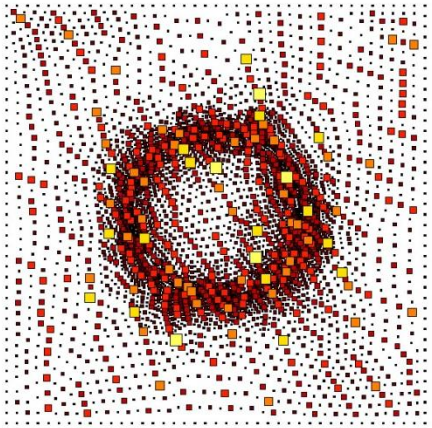


AMG coarsens grids in the direction of geometric smoothness

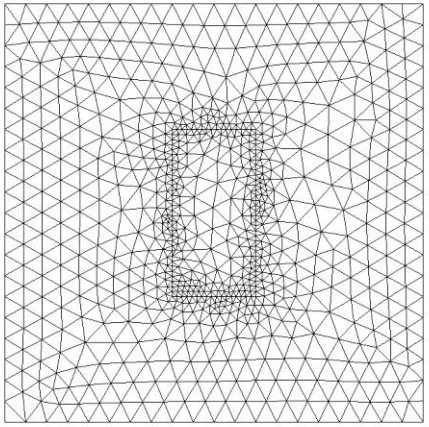
AMG grid hierarchies for several 2D problems

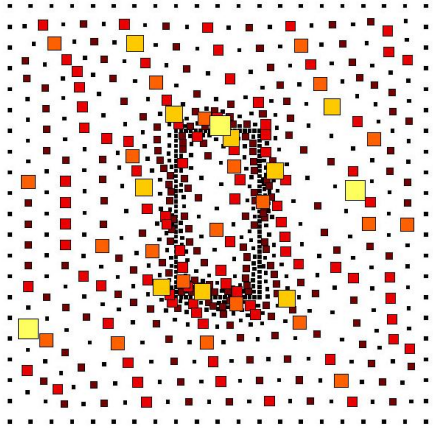
domain1 - 30^o



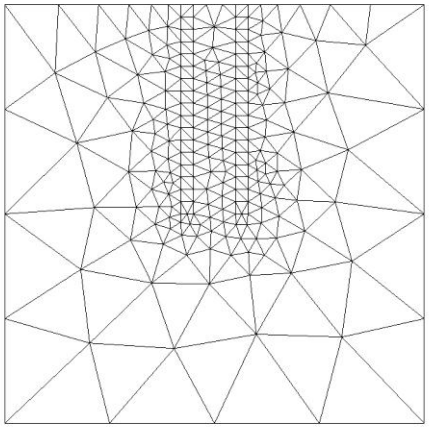


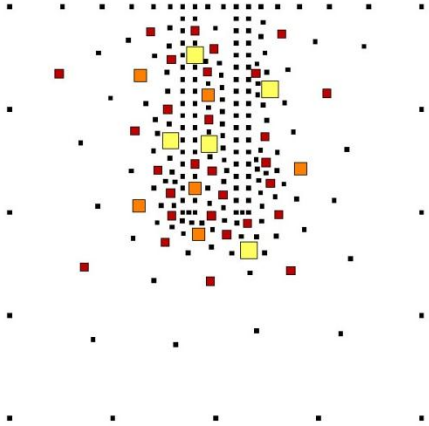
domain2 - 30^o



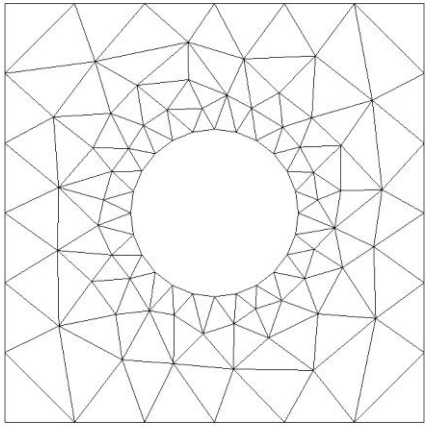


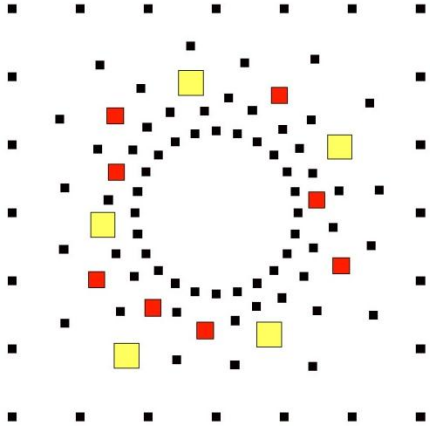
pile





square-hole





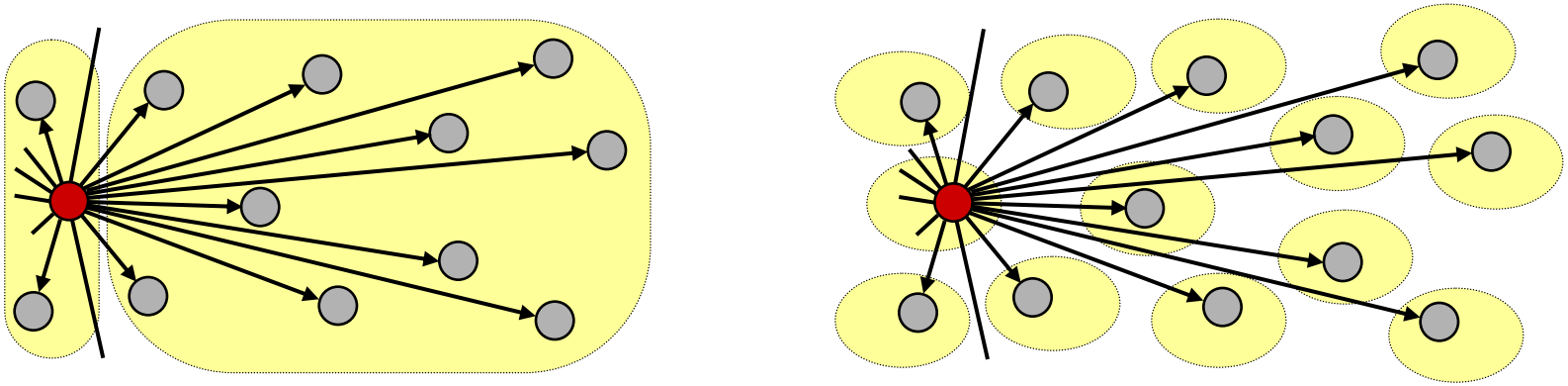
Parallel Coarsening Algorithms

- AMG coarsening algorithm is inherently sequential

- Several parallel algorithms (in *hypre*):
  - CLJP (Cleary-Luby-Jones-Plassmann) – one-pass approach with random numbers to get concurrency
  - Falgout – C-AMG on processor interior, then CLJP to finish
  - PMIS – CLJP without the 'C'; parallel version of C-AMG first pass
  - HMIS – C-AMG on processor interior, then PMIS to finish
  - CGC (Griebel, Metsch, Schweitzer) – compute several coarse grids on each processor, then solve a global graph problem to select the grids with the best "fit"
  - …
- Other parallel AMG codes use similar approaches

Parallel coarse-grid selection in AMG can produce unwanted side effects

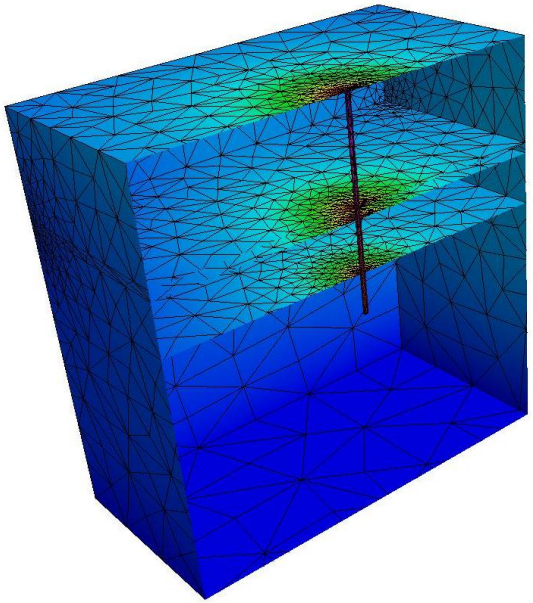
- Non-uniform grids can lead to increased operator complexity and poor convergence
- Operator “stencil growth” reduces parallel efficiency

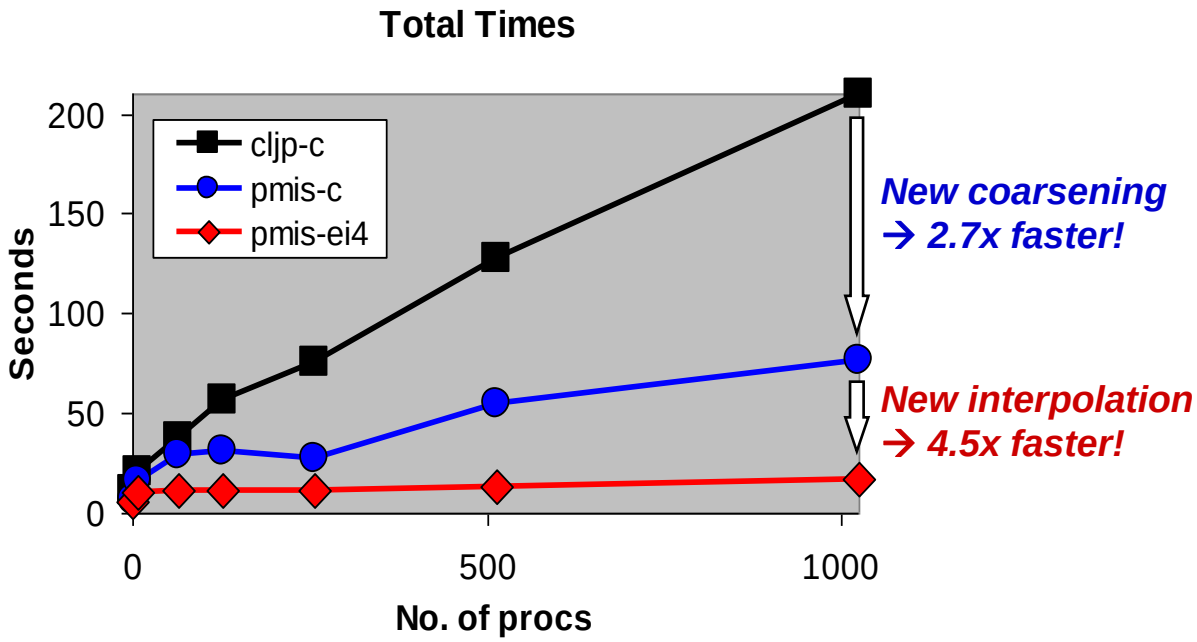


- Currently no guaranteed ways to control complexity
- Can ameliorate with more aggressive coarsening
- Requires long-range interpolation approaches

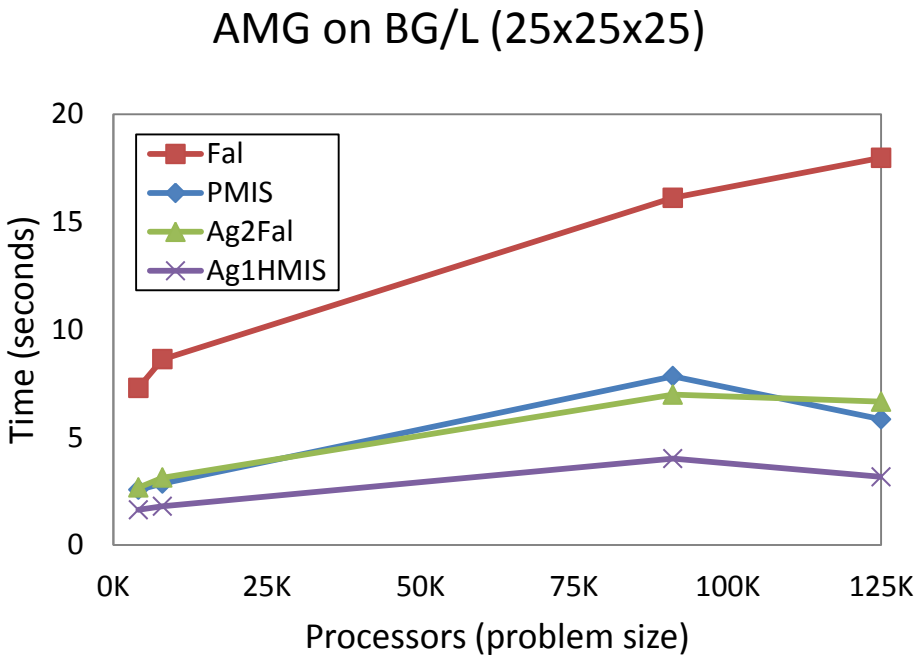
New parallel coarsening and long-range interpolation methods are improving scalability

- Unstructured 3D problem with material discontinuities
- About 90K unknowns per processor on MCR (Linux cluster)
- AMG - GMRES(10)





Parallel AMG in hypre now scales to 130K processors on BG/L … and beyond



- Largest problem above: 2B unknowns
- Largest problem to date: 26B unknowns on 98K processors of BG/L
- Most processors to date: 16B unknowns on 196K cores of Jaguar (Cray XT5 at ORNL)

We analyzed the scalability of several smoothers based on a two-grid multigrid theory

- For a given set of coarse variables, let P be the prolongation that optimizes convergence, then

$$\|E_{TG}\|_A^2 \leq 1 - \frac{1}{K_\star}; \qquad K_\star = \sup_e \frac{\langle S^T \widetilde{M} S e,\ e\rangle}{\langle S^T A S e,\ e\rangle}$$

- In the classical AMG setting, P is "ideal interpolation"

$$R^T = \begin{bmatrix} 0 \\ I_c \end{bmatrix}; \quad S = \begin{bmatrix} I_f \\ 0 \end{bmatrix}; \quad P = \begin{bmatrix} -A_{ff}^{-1} A_{fc} \\ I_c \end{bmatrix}$$

- In the classical setting of smoothing factor analysis, P consists of the smallest eigenvectors of A

$$R^T = [v_1, ..., v_{n_c}]; \quad S = \left[v_{n_c+1}, ..., v_n\right]; \quad P = R^T$$

- We analyzed $K_\star$ for various smoothers

Hybrid Gauss-Seidel smoother is the default smoother in BoomerAMG and scales better than expected

- Block Jacobi

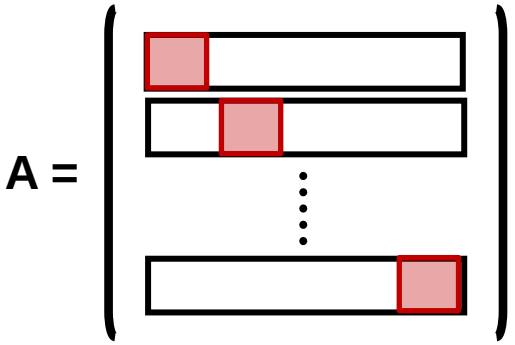
$$I - M_H^{-1} A; \;\; M_H = diag\{A_{kk}\}$$

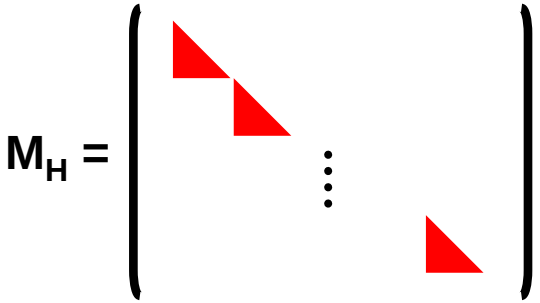
- Hybrid GS – GS on each processor, Jacobi on processor boundaries (inexact block Jacobi)
  - Default smoother used in *hypre*'s BoomerAMG

$$I - M_{HGS}^{-1} A; \;\; M_{HGS} = diag\{D_{kk} + L_{kk}\}$$

- As number of cores increases, block Jacobi convergence approaches that of point Jacobi
- For "large enough" blocks, block Jacobi smoothing does not approach point Jacobi

- Hybrid GS is a better smoother than block Jacobi
- More local work may not be beneficial!



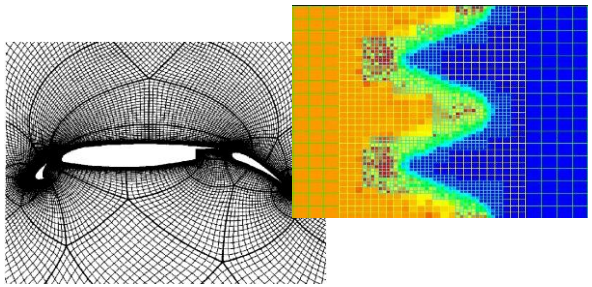


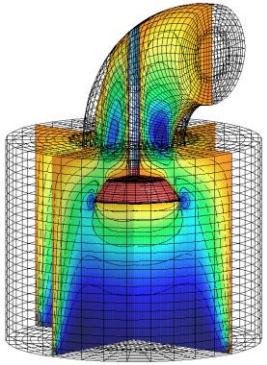
Multigrid Software

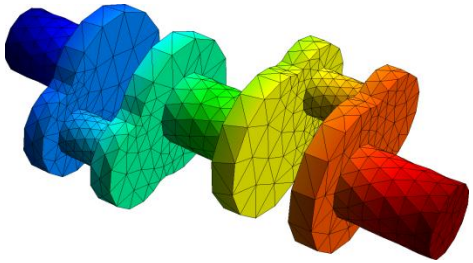
2007 Winner!

Simulation codes present a wide array of challenges for scalable linear solver libraries

- Different applications
  - Diffusion, elasticity, magnetohydrodynamics (MHD)
- Different discretizations and meshes
  - Structured, block-structured, structured AMR, overset, unstructured

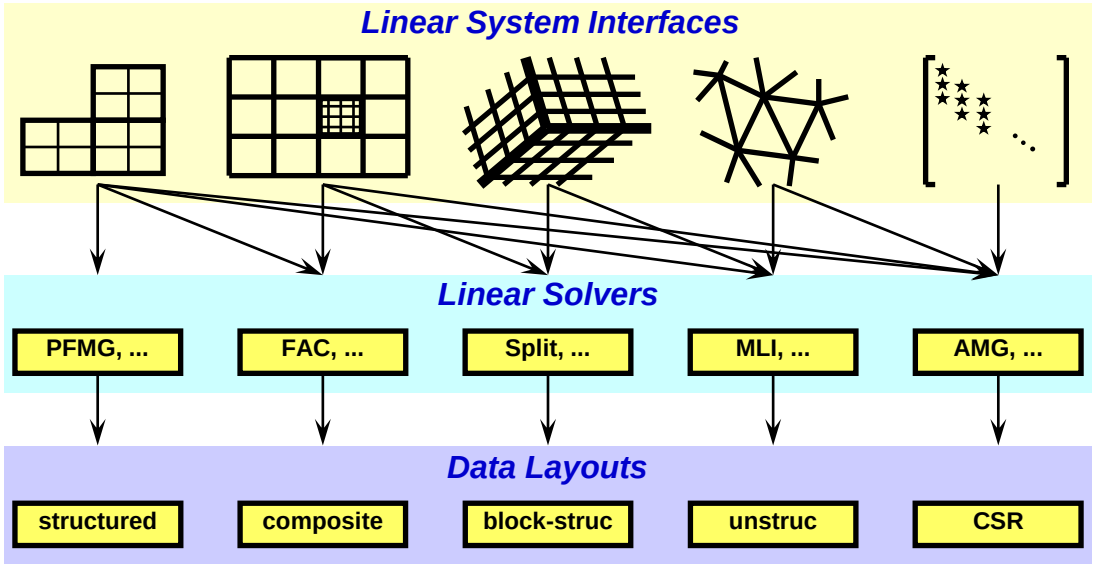


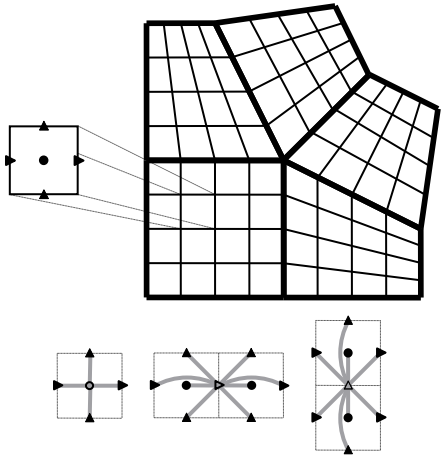




- Different languages – C, C++, Fortran
- Different programming models – MPI, OpenMP
- Scalability beyond 100,000 processors!

Unique software interfaces in *hypre* provide efficient solvers not available elsewhere





Block-structured grid with 3 variable types and 3 discretization stencils

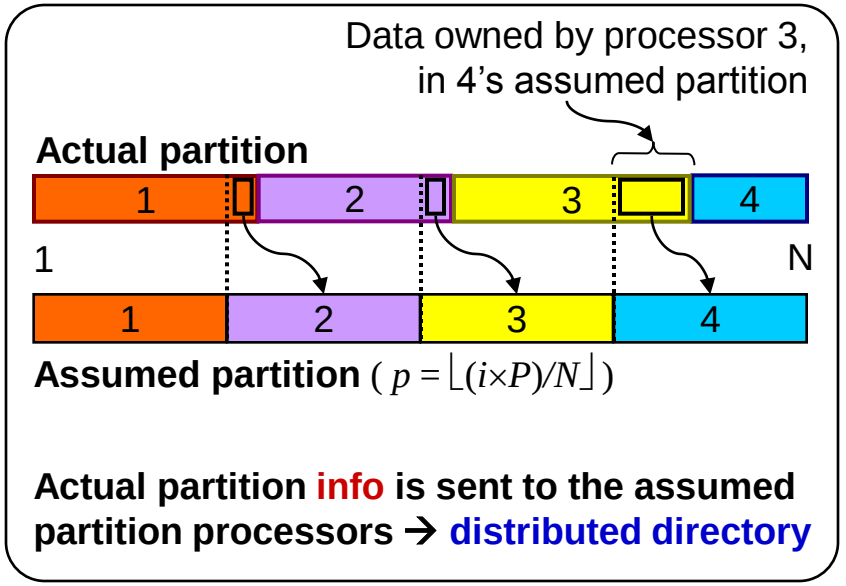
- Example: *hypre*'s interface for semi-structured grids
  - Based on "grids" and either "stencils" or "finite elements" (new)
  - Allows for specialized solvers for structured AMR
  - Also provides for more general solvers like *AMG*

Assumed partition (AP) algorithm enables scaling to 100K+ processors

- Answering global distribution questions previously required $O(P)$ storage & computations
- On BG/L, $O(P)$ storage may not be possible

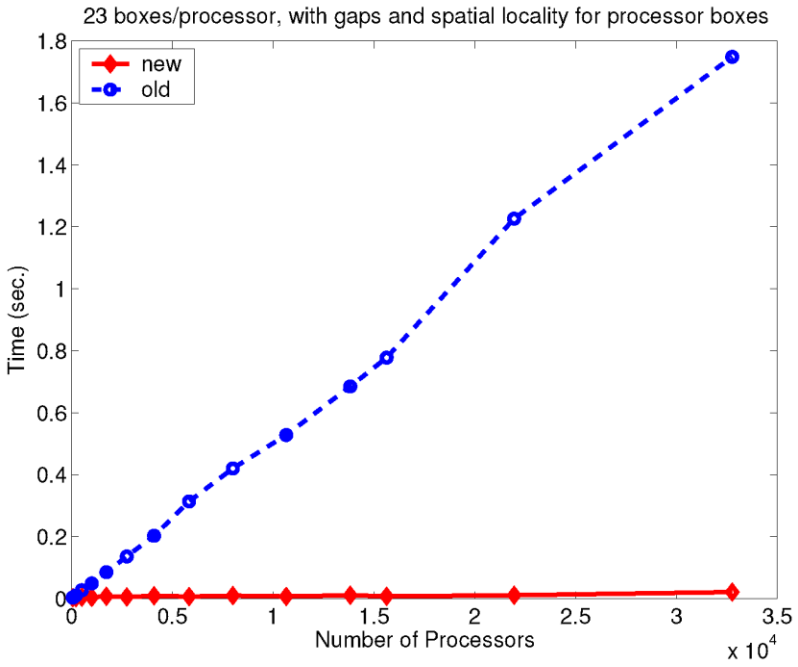
- New algorithm requires
  - $O(1)$ storage
  - $O(\log P)$ computations
- Now available in *hypre*

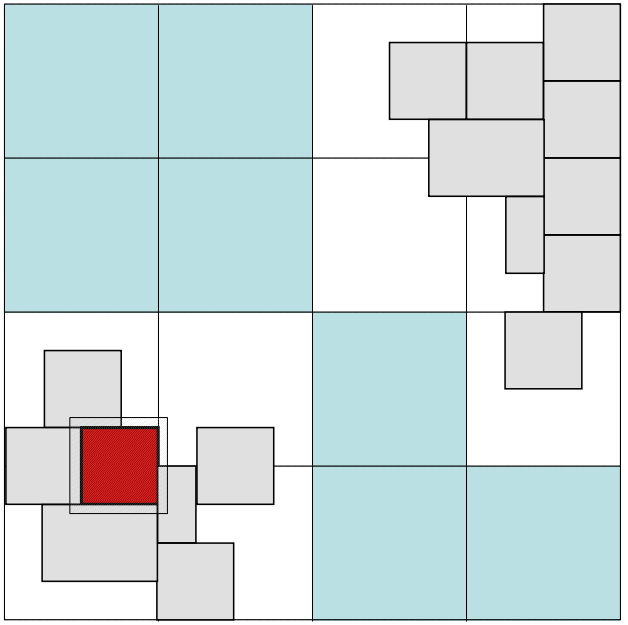
- AP has general applicability beyond *hypre*



Assumed partition (AP) algorithm is more challenging for structured AMR grids

- AMR can produce grids with “gaps”
- Our AP function accounts for these gaps for scalability
- Demonstrated on 32K procs of BG/L

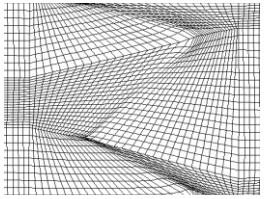




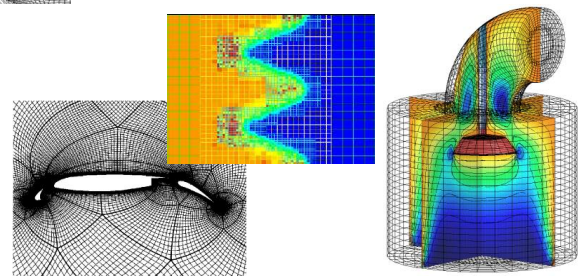
Simple, naïve AP function leaves processors with empty partitions

Currently, *hypre* supports four system interfaces

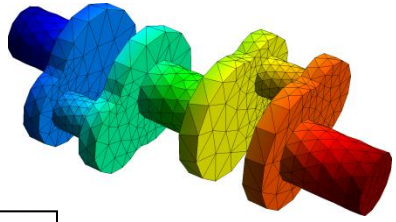
- Structured-Grid (`Struct`)
  - ***logically rectangular grids***



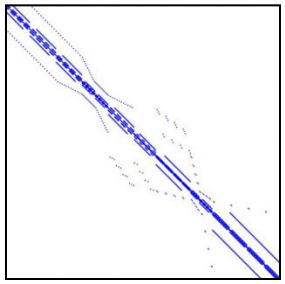
- Semi-Structured-Grid (`SStruct`)
  - ***grids that are mostly structured***



- Finite Element (`FEI`)
  - ***unstructured grids with finite elements***



- Linear-Algebraic (`IJ`)
  - ***general sparse linear systems***



Current solver / preconditioner availability via *hypre*'s system interfaces

Data Layouts	Solvers	System Interfaces			
		Struct	SStruct	FEI	IJ
Structured	Jacobi	✓	✓		
	SMG	✓	✓		
	PFMG	✓	✓		
Semi-structured	Split		✓		
	SysPFMG		✓		
	FAC		✓		
	Maxwell		✓		
Sparse matrix	AMS		✓	✓	✓
	BoomerAMG		✓	✓	✓
	MLI		✓	✓	✓
	ParaSails		✓	✓	✓
	Euclid		✓	✓	✓
	PILUT		✓	✓	✓
Matrix free	PCG	✓	✓	✓	✓
	GMRES	✓	✓	✓	✓
	BiCGSTAB	✓	✓	✓	✓
	Hybrid	✓	✓	✓	✓

Getting the code

- To get the code, go to

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

- User's / Reference Manuals can be downloaded directly
- A short form must be filled out (this is just for our own records)

- To report bugs, request features, or ask general usage questions, send email to

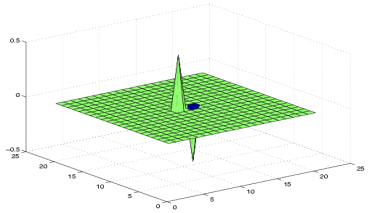
  hypre-support@llnl.gov

- We use a tool called Roundup to automatically tag and track issues

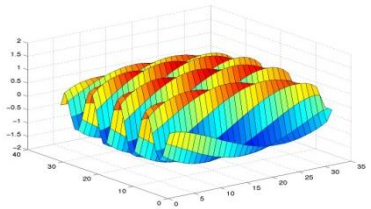
AMG for Electromagnetic Problems

Electromagnetic (EM) problems have huge oscillatory near null spaces

- Definite Maxwell, Indefinite Maxwell, Helmholtz
- Require specialized smoothers and coarse grids



Local: specialized relaxation
(Definite / Indefinite Maxwell)



Global: specialized coarse grids
(Helmholtz, Indefinite Maxwell)

- Definite Maxwell, Nédélec edge FEM discretization

$$\nabla \times \alpha \nabla \times \boldsymbol{E} + \beta \boldsymbol{E} = f \quad \alpha, \beta > 0$$

- Near null-space characterized by gradients

$$\nabla \times (\nabla p_h) = 0$$

Geometric multigrid for definite Maxwell

- Helmholtz decomposition

$$\boldsymbol{E}_h = \boldsymbol{v}_h + \nabla p_h$$

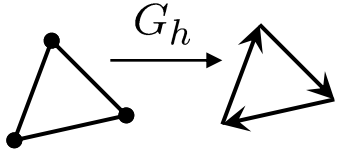
divergence-free curl-free

- Smooth both components (Hiptmair, SINUM 1998)

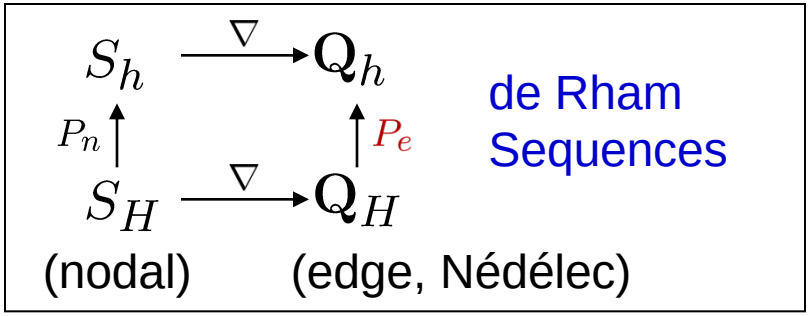
$$R_h = R_{e,h} + G_h R_{v,h} G_h^T$$

Point smoother for A_h Point smoother for $G_h^T A_h G_h$

Discrete Gradient



- Block smoother (Arnold, Falk, Winther, Num. Math. 2000)
- Natural FE interpolation
- Difficulties extending to
  - unstructured meshes
  - variable coefficients



Auxiliary-space Maxwell solver (AMS) utilizes a new decomposition

- Based on Hiptmair, Xu (2006)

$$\boldsymbol{E}_h = \boldsymbol{v}_h + \nabla p_h + \boldsymbol{\Pi}_h \boldsymbol{z}_h$$

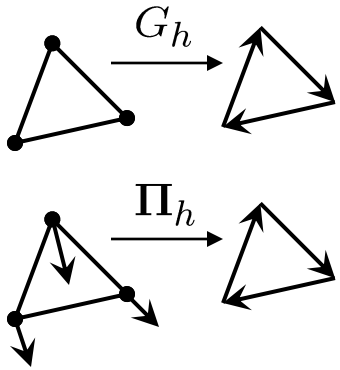
- Define preconditioner based on nodal solvers

$$B_h = R_h + G_h B_{v,h} G_h^T + \boldsymbol{\Pi}_h \boldsymbol{B}_{\boldsymbol{v},h} \boldsymbol{\Pi}_h^T$$

Point smoother for A_h

AMG solver for $G_h^T A_h G_h$

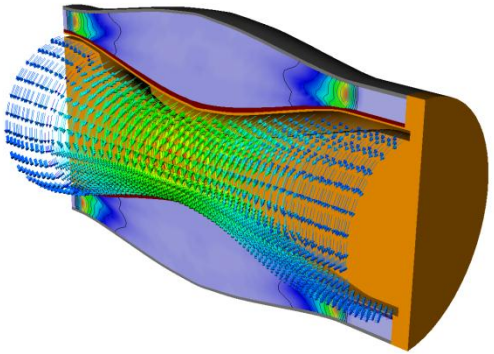
AMG solver for $\boldsymbol{\Pi}_h^T A_h \boldsymbol{\Pi}_h$

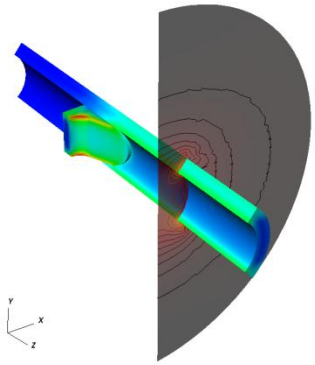


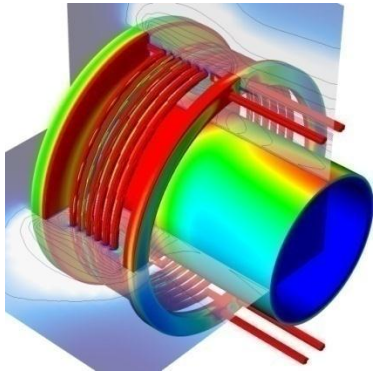
- User provides A, G_h and vertex coordinates
- Fast computation of Π_h (~ 3 mat-vec multiplies)
- AMS is a variational form of Hiptmair-Xu

Auxiliary-space Maxwell Solver (AMS) is improving solve times by up to 25x for some EM problems

- Hiptmair-Xu / AMS are the first provably scalable solvers for EM on unstructured grids
- Employs BoomerAMG
- Highly robust
  - Materials with widely varying electromagnetic properties
  - Unstructured grids
- Example: 1.2B unknowns on 1.9K processors took 355s (23 iterations)



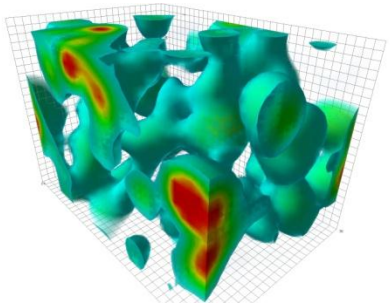




Adaptive AMG

Adaptive *AMG* is well-suited for QCD

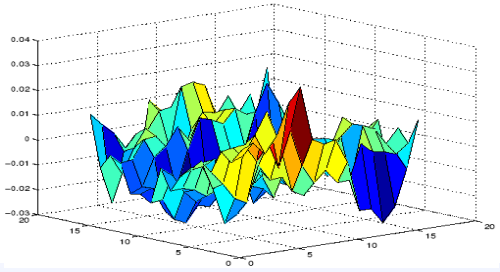
- Quantum Chromodynamics (QCD) is the theory of strong forces in the Standard Model of particle physics



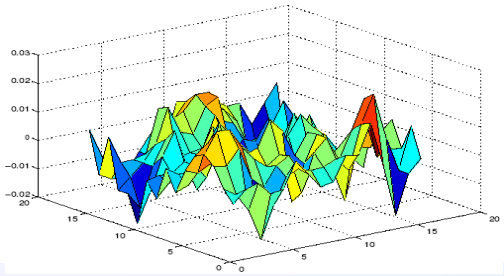
- Scalable solvers for the Dirac equations have been elusive until recently

- Challenges:
  - The system is complex and indefinite
  - The system can be extremely ill-conditioned
  - Near null space is unknown and oscillatory!

Real part



Imaginary part



Adaptive *AMG* idea: use the method to improve the method

- Requires no a-priori knowledge of the near null space
- Idea: uncover *representatives* of slowly-converging error by applying the "current method" to $Ax = 0$, then use these to adapt (improve) the method

- Achi Brandt's *Bootstrap AMG* is an adaptive method
- PCG can be viewed as an adaptive method
  - Not optimal because it uses a global view
  - The key is to view representatives locally

- We developed 2 methods: αAMG and αSA (SISC pubs)

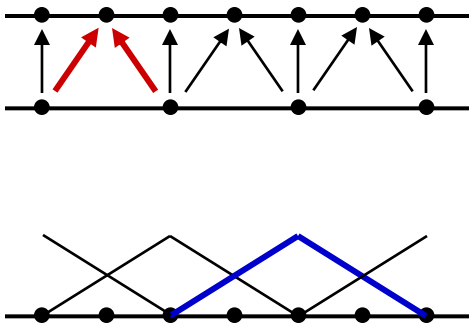
To build effective interpolation, it is important to interpret the near null space in a local way

- (2-level) Coarse-grid correction is a projection

$$(I - P(P^T AP)^{-1} P^T A)e$$

- Better to break up near null space into a local basis

$$P = \frac{1}{2}\begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} \quad \Rightarrow \quad P = \frac{1}{2}\begin{bmatrix} 2 & & & \\ 1 & 1 & & \\ & 2 & & \\ & 1 & 1 & \\ & & 2 & \\ & & 1 & 1 \\ & & & 2 \end{bmatrix}$$



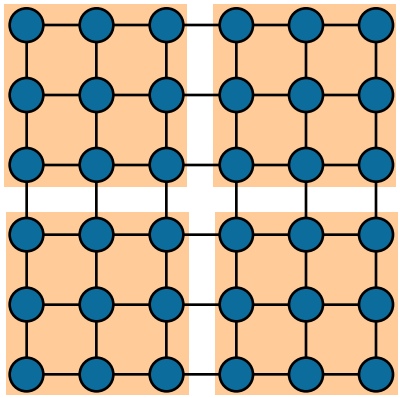
Deflation – not optimal

Multigrid – optimal

- Get full approximation property (low-frequency Fourier modes in this example)

Smoothed Aggregation (SA) builds interpolation by first chopping up a global basis, then smoothing it

- Tentative interpolation is constructed from “aggregates” (local QR factorization is used to orthonormalize)



$$\hat{P} \quad B_c \quad B$$

$$\begin{bmatrix} \\ \\ \\ \\ \\ \\ \end{bmatrix} \begin{bmatrix} \\ \\ \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \\ \\ \\ \end{bmatrix}$$

$$; \quad \hat{P}^T \hat{P} = I$$

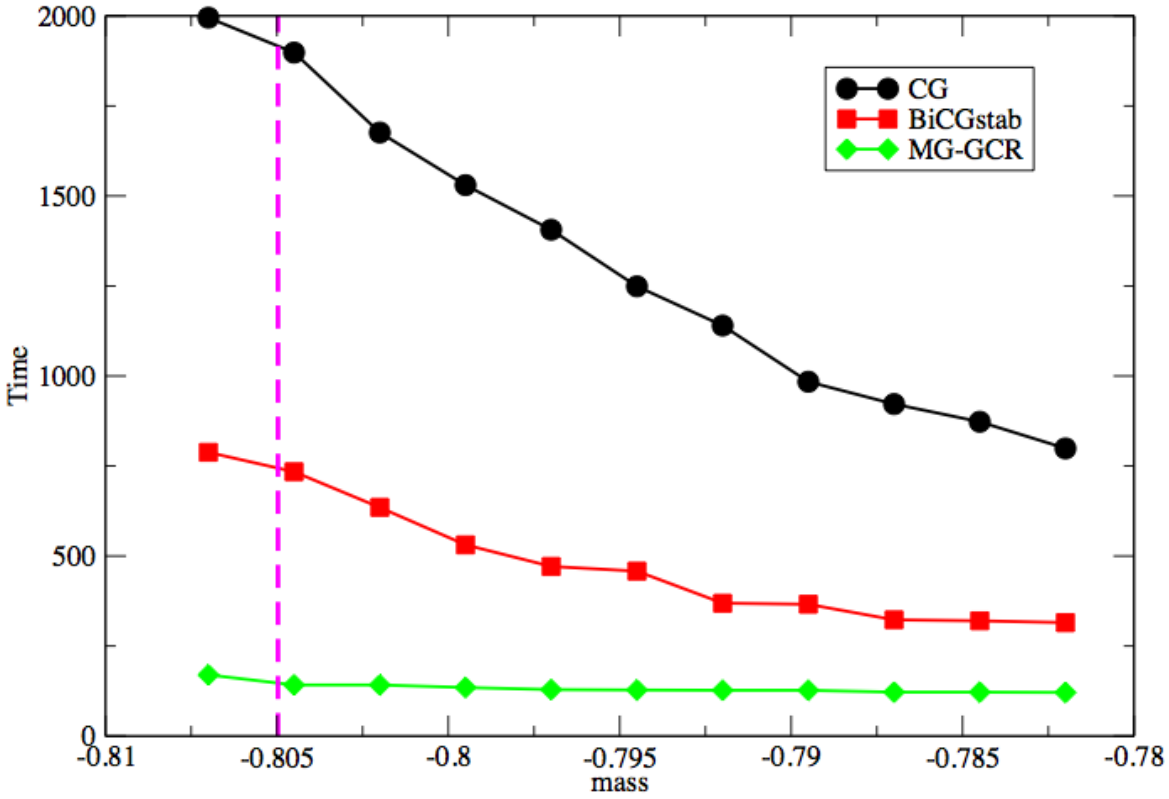
- Smoothing adds basis overlap and improves approximation property

$$P = S\hat{P}$$

Adaptive smoothed aggregation (αSA) automatically builds the global basis for SA

- Generate the basis one vector at a time
  - Start with relaxation on $Au=0$ → u_1 → αSA(u_1)
  - Use αSA(u_1) on $Au=0$ → u_2 → αSA(u_1,u_2)
  - Etc., until we have a good method
- Setup is expensive, but is amortized over many RHS's
- Published in 2004, highlighted in SIAM Review in 2005
  - Brezina, Falgout, MacLachlan, Manteuffel, McCormick, and Ruge, "Adaptive smoothed aggregation (αSA)," *SIAM J. Sci. Comput.* (2004)
- Successfully applied to 2D QED
  - Brannick, Brezina, Keyes, Livne, Livshits, MacLachlan, Manteuffel, McCormick, Ruge, and Zikatanov, "Adaptive smoothed aggregation in lattice QCD," Springer (2006)

4D Wilson-Dirac Results: D-MG shows no critical slowing down (Time)



- Parameters: $N=16^3x32$, $\beta=6.0$, $m_{crit} = -0.8049$
- D-MG Parameters: 4^4x3x2 blocking, 3 levels, W(2,2,4) cycle, $N_v = 20$, setup run at m_{crit}

Summary and Conclusions

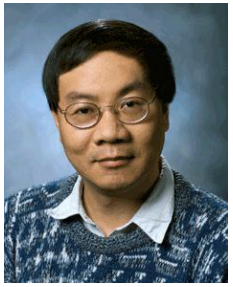
- Multigrid methods are optimal and have good scaling potential
- Many useful tools (GS, W-cycles) cannot be used in parallel
- AMG is based primarily on matrix entries
- In practice, some additional properties of the underlying system are assumed (near null space)
- AMG can solve a large class of problems and can scale to BG/L-class machines
- Parallel computing imposes additional restrictions on MG algorithmic development
- Getting efficient use out of multi-core architectures is challenging!
- Still many outstanding research questions

The Scalable Linear Solvers Team

Allison Baker

Rob Falgout

Tzanio Kolev



Charles Tong

Panayot Vassilevski

Ulrike Yang

See http://www.llnl.gov/casc/linear_solvers for publications, presentations, and software (hypre)

Former

- Chuck Baldwin
- Guillermo Castilla
- Edmond Chow
- Andy Cleary
- Noah Elliott
- Van Henson
- Ellen Hill
- David Hysom
- Jim Jones
- Mike Lambert
- Barry Lee
- Jeff Painter
- Tom Treadway
- Deborah Walker

Thank You!

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.