Exploiting structure in numerical libraries (PETSc)

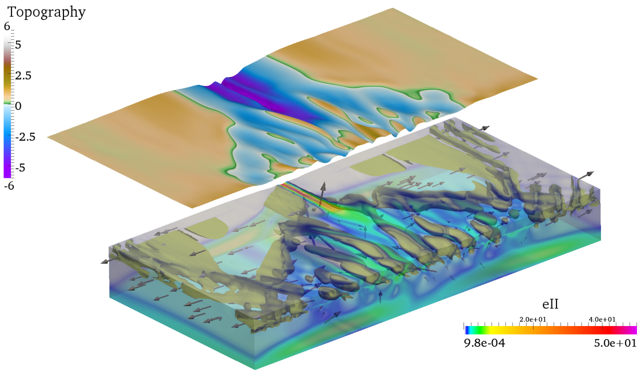
Shrirang Abhyankar, Mark Adams (LBL), Satish Balay, **Jed Brown**, Peter Brune, Emil Constantinescu, Debojyoti Ghosh, Dmitry Karpeev, Matt Knepley (UChicago), Lois Curfman McInnes, Barry Smith, Hong Zhang

Mathematics and Computer Science Division, Argonne National Laboratory

ANL, 2013-06-18

From natural structure to efficient algorithms

- Fluids, structures, plasma, chemistry, mesoscale materials, ...
- Conservation of mass, energy, ...
- Approximately balanced dynamics: weather systems, resonance
- Design accurate numerical methods that preserve compatibility
- Design efficient, scalable algorithms for solving associated algebraic problems



Structure is also needed for analysis of models

- Optimization
  - Smoothness
  - Convexity
  - Computability of gradients
- Data assimilation and experimental design
  - High-dimensional probability distributions
  - Sparsity of observations
- Stability analysis (bifurcations)

PETSc's Goal

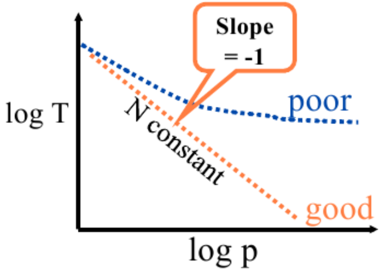
Make the best possible structure-exploiting methods:

- reusable
- easy to use
- extensible
- composable

Scalability definitions

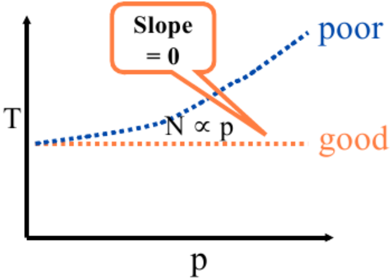
Strong scalability

- Fixed problem size
- execution time T inversely proportional to number of processors p

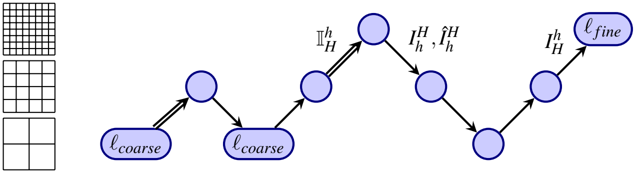


Weak scalability

- Fixed problem size per processor
- execution time constant as problem size increases

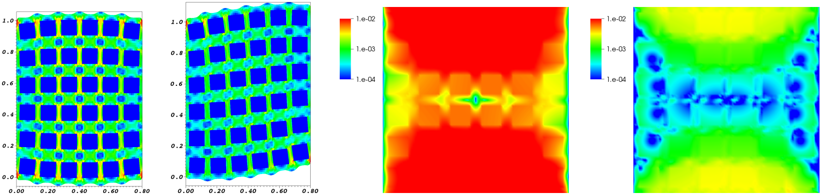


Full Multigrid(FMG)



- start with coarse grid
- x is prolonged using $\mathbb{I}_H^h$ on first visit to each finer level
- truncation error within one cycle
- about five work units for many problems
- highly efficient solution method

τ corrections

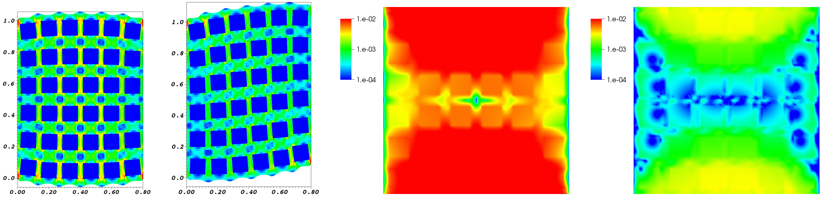


- Plane strain elasticity, $E = 1000, \nu = 0.4$ inclusions in $E = 1, \nu = 0.2$ material, coarsen by 3^2.
- Solve initial problem everywhere and compute $\tau_h^H = A^H \hat{I}_h^H u^h - I_h^H A^h u^h$
- Change boundary conditions and solve FAS coarse problem

$$N^H \acute{u}^H = \underbrace{I_h^H \acute{f}^h}_{\acute{f}^H} + \underbrace{N^H \hat{I}_h^H \tilde{u}^h - I_h^H N^h \tilde{u}^h}_{\tau_h^H}$$

- Prolong, post-smooth, compute error $e^h = \acute{u}^h - (N^h)^{-1} \acute{f}^h$
- Coarse grid *with* τ is nearly $10\times$ better accuracy

τ corrections



- Plane strain elasticity, $E = 1000, \nu = 0.4$ inclusions in $E = 1, \nu = 0.2$ material, coarsen by 3^2.
- Solve initial problem everywhere and compute $\tau_h^H = A^H \hat{I}_h^H u^h - I_h^H A^h u^h$
- Change boundary conditions and solve FAS coarse problem

$$N^H \acute{u}^H = \underbrace{I_h^H \acute{f}^h}_{\acute{f}^H} + \underbrace{N^H \hat{I}_h^H \tilde{u}^h - I_h^H N^h \tilde{u}^h}_{\tau_h^H}$$

- Prolong, post-smooth, compute error $e^h = \acute{u}^h - (N^h)^{-1} \acute{f}^h$
- Coarse grid *with* τ is nearly $10\times$ better accuracy

The Great Solver Schism: Monolithic or Split?

Monolithic

- Direct solvers
- Coupled Schwarz
- Coupled Neumann-Neumann (need unassembled matrices)
- Coupled multigrid
- X Need to understand local spectral and compatibility properties of the coupled system

Split

- Physics-split Schwarz (based on relaxation)
- Physics-split Schur (based on factorization)
  - approximate commutators SIMPLE, PCD, LSC
  - segregated smoothers
  - Augmented Lagrangian
  - "parabolization" for stiff waves
- X Need to understand global coupling strengths

- Preferred data structures depend on which method is used.
- Interplay with geometric multigrid.

Why is exploiting structure hard?

- Black box interfaces tend to exploit only one or two types of structure at a time
- Cutting-edge science often needs to exploit all available structure
- Generic data structures not well matched to evolving hardware
  - More concurrency, less memory per thread
  - Deeper memory hierarchy, heterogeneous execution
- Nonlinearity and coupling with other physical models can change the available structure
- Assumptions break down between model problems and production
- Assumptions fall through the cracks
  - E.g., model nonlinearity breaks assumption of Gaussian probability distribution
  - Switching to more "robust" method makes problem intractable
- Applications seek to satisfy disparate user groups, make assumptions invalid in other contexts