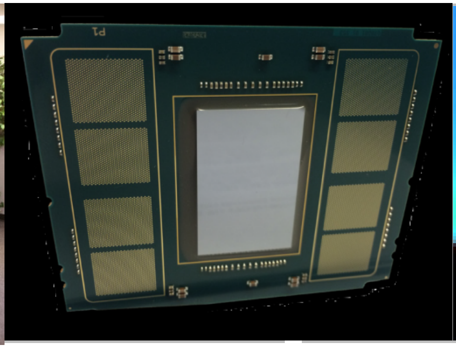
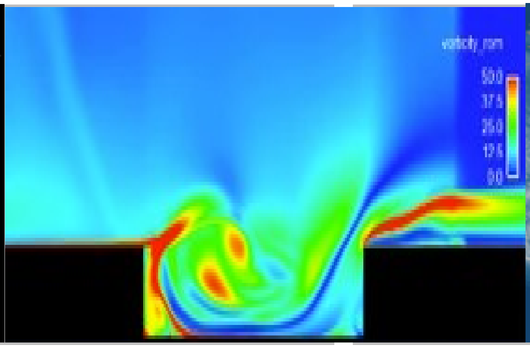
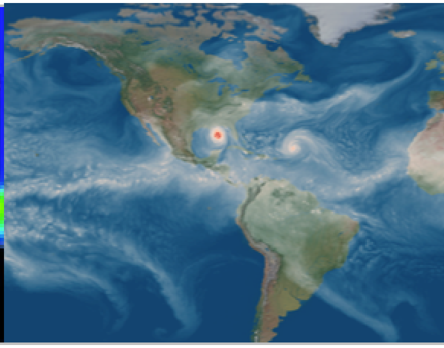
SAND2017-9947C







A Hierarchical Low-Rank Solver for Large, Sparse Linear Systems

Erik Boman, Chao Chen, Eric Darve, Siva Rajamanickam, Ray Tuminaro,

Householder XX, June 2017

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

Hierarchical Low-Rank (HLR) Matrices and Solvers

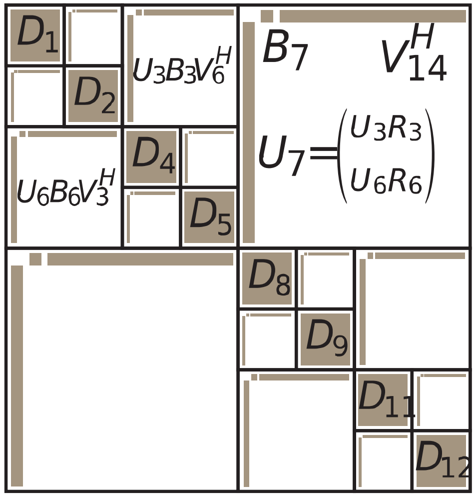
- Low-rank structure is everywhere!
  - Plenary talk by Alex Townsend
- Hierarchical matrices and solvers experiencing a revival
  - Early work by Hackbusch, 1999-2000
- Key insight: Many matrices have useful (rank) structure
  - Blocks far from diagonal can be approximated using low-rank
    - Holds for elliptic PDEs, some other (e.g., advection-diffusion problems)
    - Similar intuition as for Fast Multipole Methods (FMM)
    - May also apply to data science (e.g. covariance matrix)
- Our goals
  - Develop new solver/preconditioner with less memory (and flops)
  - Speed up sparse direct solvers (high accuracy)
  - Use as preconditioner (low accuracy)

Low-Rank Structure

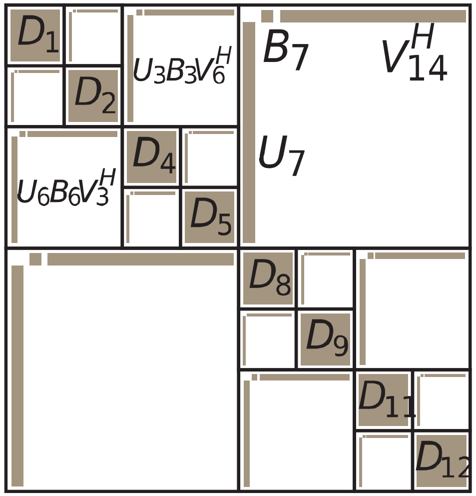
Low-rank structure often occurs in *off-diagonal blocks in*

- The matrix A
- The inverse of A
- The LU factors of A

- Hierarchical low-rank structure: (Figure from Wang et al.)



(a) HSS matrix.

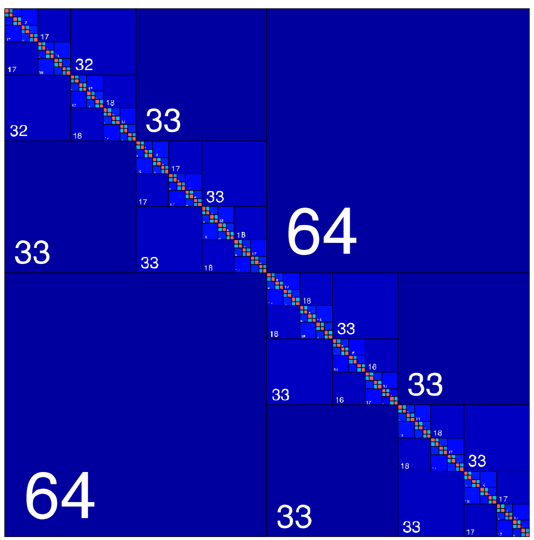


(b) HODLR matrix.

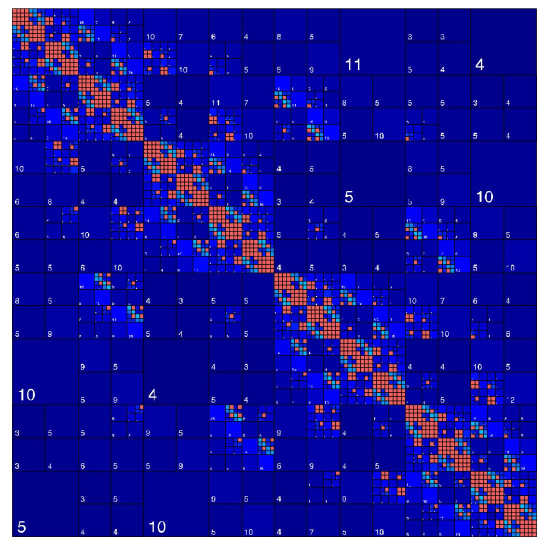
Hierarchical Matrix Formats

	Simple basis	Nested basis
Weak admissibility	HODLR	HSS
Any admissibility	H	**H^2**

- HSS is perhaps most popular but has drawbacks
  - Weak admissibility may require high ranks (esp. 3D problems)
- Example: Inverse of 2D Poisson eqn. *(Courtesy G. Chavez et al.)*



(a) Weak admissibility.



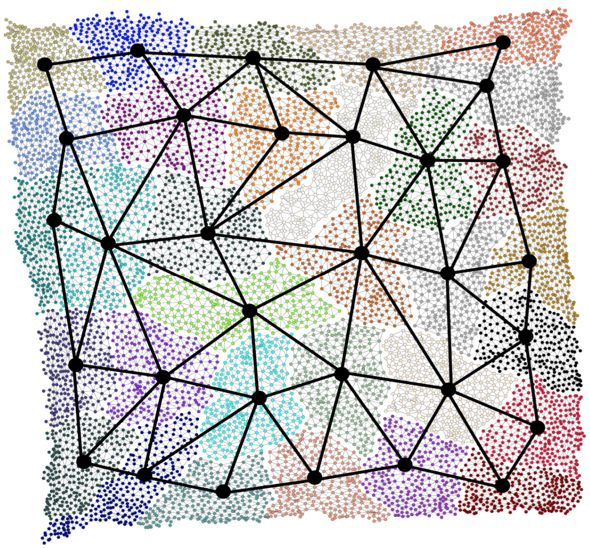
(b) Standard admissibility.

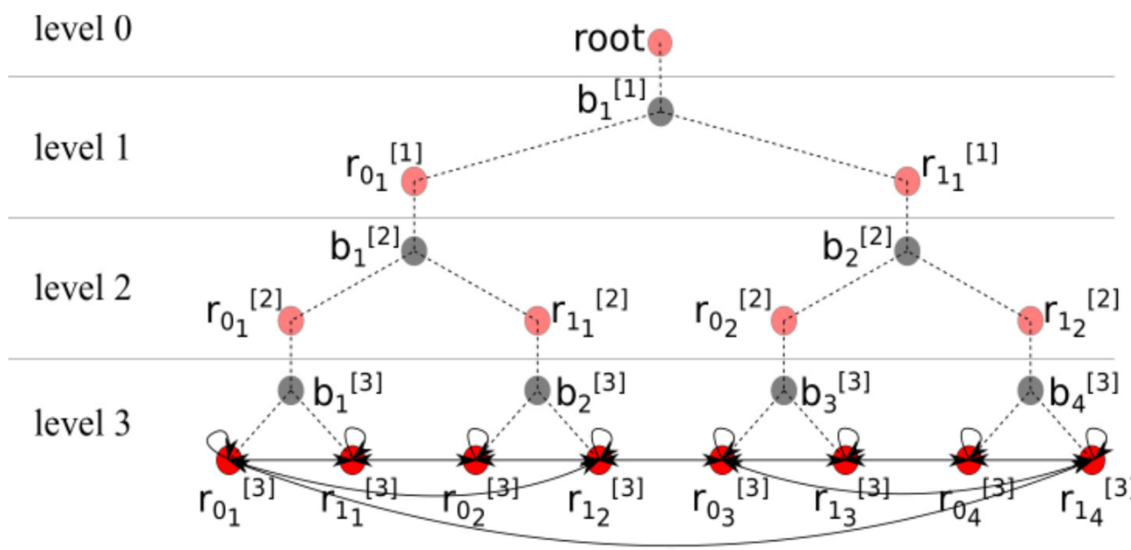
Fast Sparse Solver Approaches

- Early work was on *dense* matrices; we focus on *sparse*
- Approximate A^{-1} directly
  - Use favorite hierarchical format: H, H^2, HSS, HODLR, …
  - Form inverse in $O(k^a * n * \log^b n)$ work, k is the rank, a,b small constants
  - May still be expensive if k is large (aka large "hidden constant")
- Approximate LU factors of A
  - Dense frontal matrices within sparse direct (multifrontal) method
    - Xia, Li, Darve, …
    - Strumpack library (Li, Ghysels, et al.)
  - Work on entire sparse matrix (Sparse triangular factors)
    - H-LU (Hackbusch, Grasedyck, Kriemann, LeBorne, …)
    - *LoRaSp* (Pouransari, Coulier, Darve)

The LoRaSp/H2 Method

- Pouransari, Coulier, Darve, SISC 2017 (in press)
- Partition graph via recursive bisection, construct tree
- Eliminate "clusters" (blocks) starting at leaf level
  - Approximate block LU factorization
  - New "coarse" dof via *extended sparsification*
- Merge neighbors, repeat for each level in the hierarchy (bottom up)





Figures courtesy Chen et al., and Pouransari et al.

6

Multilevel Block Incomplete Factorization

- My Interpretation (algebraic):
  - LoRaSp/H2 solver is really a Block ILU(0) factorization where the "dropped" blocks are approximated using low-rank method
  - A= LU+E, where E has blocks that we approximate by low rank
  - We compensate for the dropped blocks by adding new rows/columns to the matrix (*extended sparsification*)
  - The Schur complement for *coarse* vertices is a smaller matrix we can solve recursively
    - Can eliminate the *black* vertices with little fill
- Low-rank approximation is different from earlier ILU work

7

Extended Sparsification

- For simplicity, assume symmetric A (can be extended)
- Suppose the off-diagonal blocks are (approx) low-rank:

$$\begin{pmatrix} A_1 & UV^T \\ VU^T & A_2 \end{pmatrix}$$

- We solve the equivalent extended system

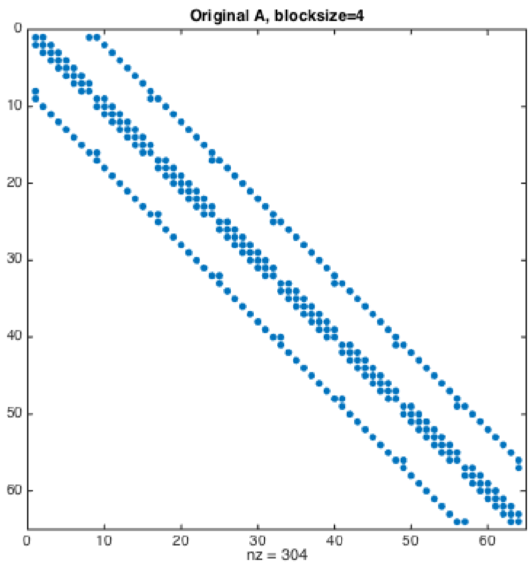
$$\begin{pmatrix} A_1 & 0 & U & 0 \\ 0 & A_2 & 0 & V \\ U^T & 0 & 0 & -I \\ 0 & V^T & -I & 0 \end{pmatrix}$$

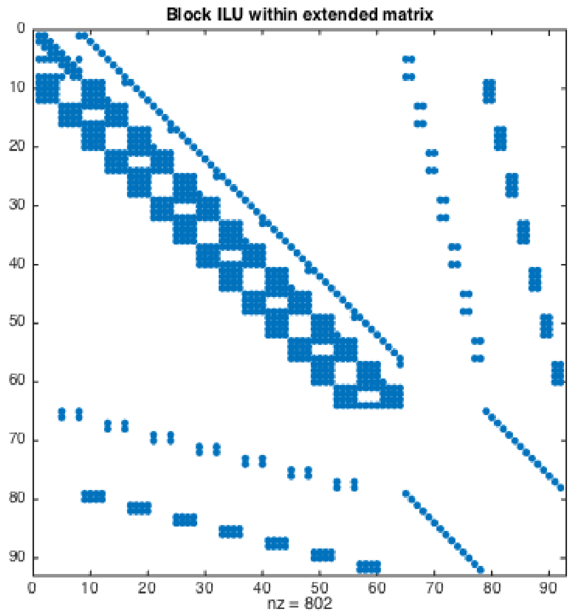
- We sparsify the original matrix, but add extra rows/cols that also need to be factored.
- The lower the ranks of U,V, the smaller extended system

8

ILU and Extended Sparsification

- Note: Fill blocks in the block LU factors often have low rank
  - Schur complements in the Gaussian elimination
- Approach: Compute blocks in ILU(0) exactly, and
  - Approximate blocks in ILU(1) (not in ILU(0)) using low-rank
  - Extend matrix with new rows/cols





9

Related Preconditioners

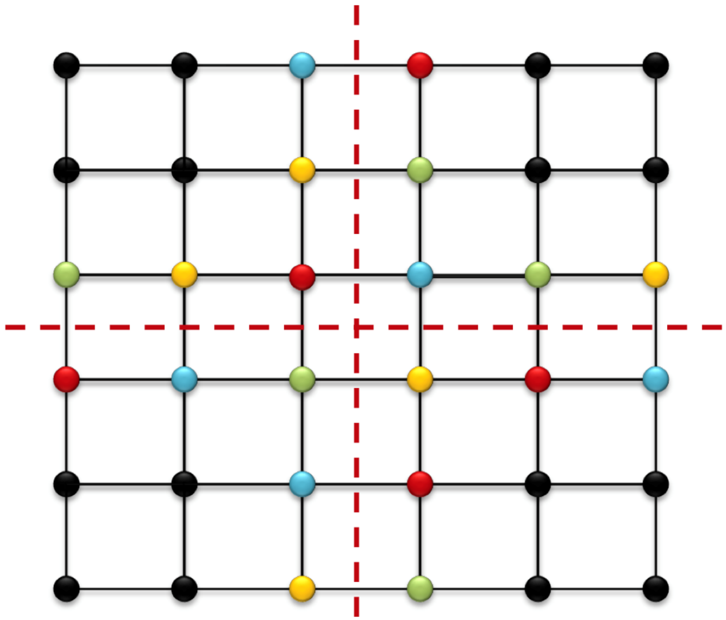
- LoRaSp/H2 method has similarities to several methods we already know:

Method	How LoRaSp/H2 differs
Block ILU	Low-rank compensation for dropped fill
	Extended sparsification
ARMS	Partially factors all blocks, no indep. sets
	Low-rank, extended sparsification
DD	"Coarse grid" represents dropped ILU fill
	Low-rank, extended sparsification
AMG	Coarse grid represents dropped ILU fill
	No smoother

10

Our Parallel Algorithm

- Data parallel: Each processor works on a subgraph (subdomain).
- Consider the cluster graph:
  - Only boundary vertices need communication.
  - Interior vertices can be eliminated independently in parallel.
- Use graph coloring on the boundary to find concurrent work.
- #synchronization points = #colors
- Can overlap communication and computation.
- Repeat for each level.
- Future: Task-based; dynamic sched.



Example: 4 processors. Each vertex (node) corresponds to a cluster of variables.

11

Experiments

- Parallel H2 solver (and results) by Chao Chen
  - Parallel extension of LoRaSp serial code
  - MPI everywhere
  - Eigen library for dense linear algebra (on node)

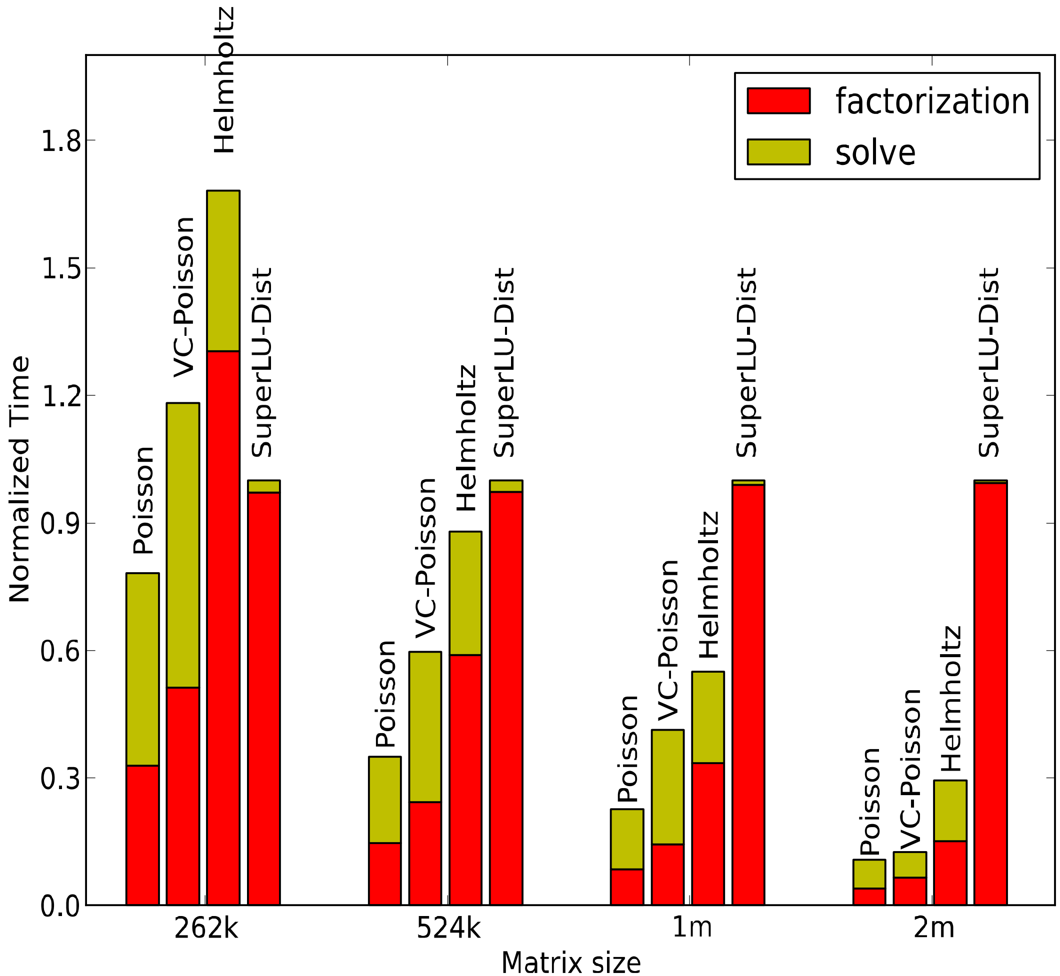
- SVD for low-rank compression
  - a) Fixed eps in matrix approx. (ranks vary)
  - b) Fixed rank in matrix approx. (quality varies)
    - used in most of the experiments

- Platform: Cray XC30 (Edison/NERSC)
  - Used 16 (out of 24) cores per node
  - Used up to 16 nodes (256 cores)

12

Results: Compare SuperLU-Dist

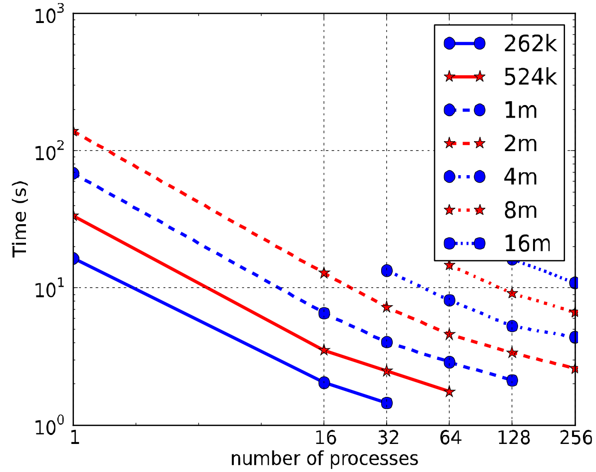
Compare hierarchical solver as preconditioner vs. SuperLU-Dist direct solver on three 3D PDE model problems. 16 processors (MPI ranks).



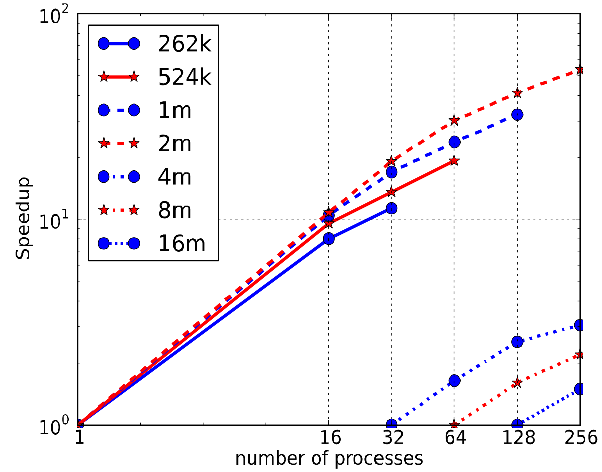
(a) Total time

13

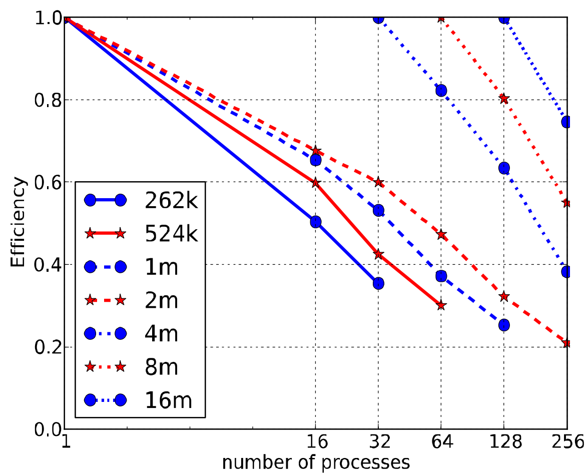
Results: 3D Poisson Eqn.



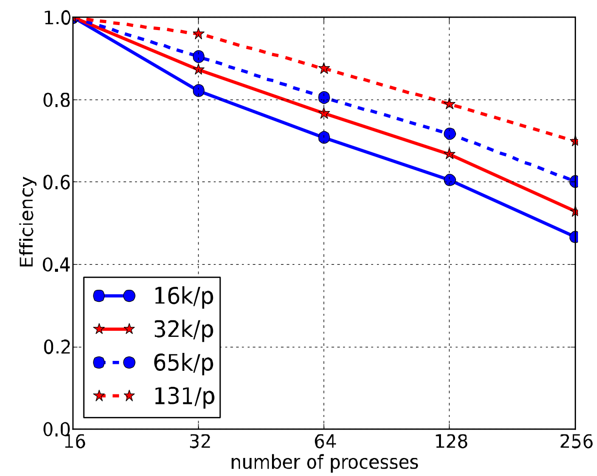
(a) Factorization time



(b) Factorization speedup



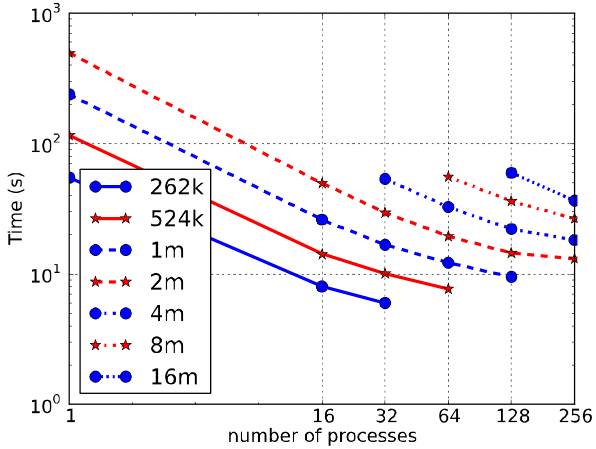
(c) Fixed total problem size



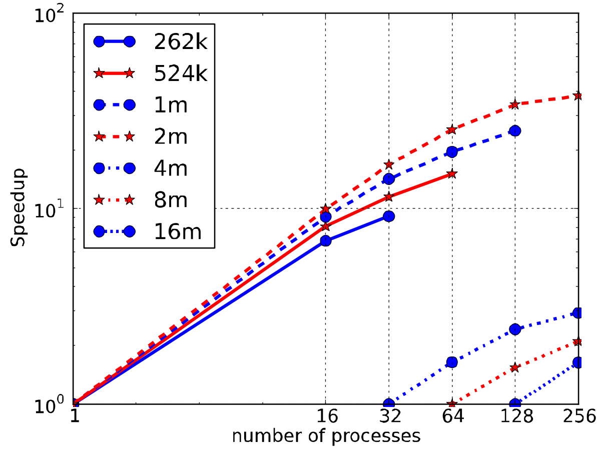
(d) Fixed problem size per processor

14

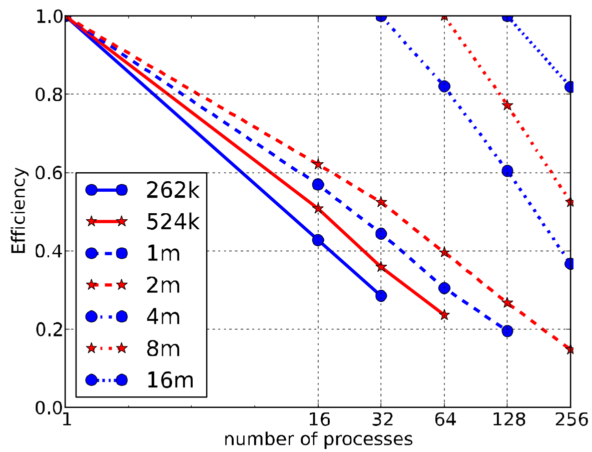
Results: Helmholtz eqn.



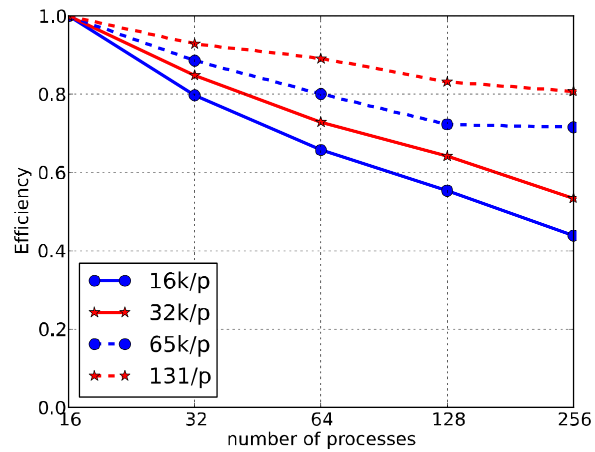
(a) Time



(b) Speedup



(c) Fixed total problem size



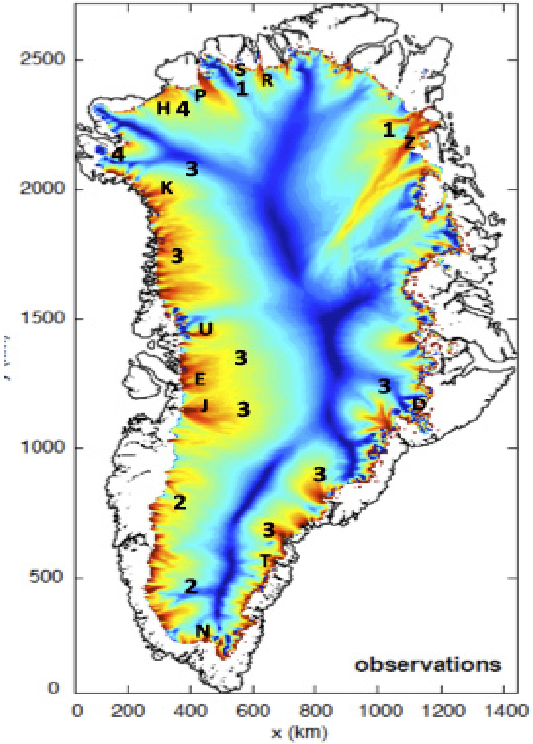
(d) Fixed problem size per processor

15

Ice Sheet Modeling: Greenland

We simulate ice sheet flow using Stokes' eqn. Use Albany/Felix software, Trilinos solvers for linear systems. 2.5D geometry is challenging as the z dimension is very different.

Precond.	8km mesh	4km mesh
ML	-	-
ML/custom	18	17
ILU (custom order)	12	21
H2(1e-1)	141	423
H2(1e-2)	36	153
H2(1e-1)*	19	21
H2(1e-2)*	14	13



* This version uses x-y mesh partitioning and treats "diagonal" grid points as neighbors (not well sep.)

Conclusions (1)

- Hierarchical low-rank methods (HLR) augment the current solver/preconditioner ecosystem.
- Faster than sparse direct
- Most useful as preconditioner
- User must choose accuracy (input arg.)
- Setup phase can be expensive.
  - Can often amortize this cost over multiple rhs
- Theory promises near-linear complexity for many PDE problems
  - Potentially more robust than multigrid/AMG (at a cost)

17

Conclusions (2)

- Well suited for modern architectures
  - Most work is in dense linear algebra (even for sparse problems)
    - High arithmetic intensity
  - Many small dense matrices at same time
    - Could use batched BLAS/LAPACK
- Our hierarchical solver
  - Is motivated by H2 but can also be interpreted as multilevel ILU
  - Many options/variations still to explore
    - Low rank: SVD, RRQR, RRLU, ACA, ID, …
- Early days:
  - Several algorithm options, best choice unclear
  - Codes are still immature but rapidly improving

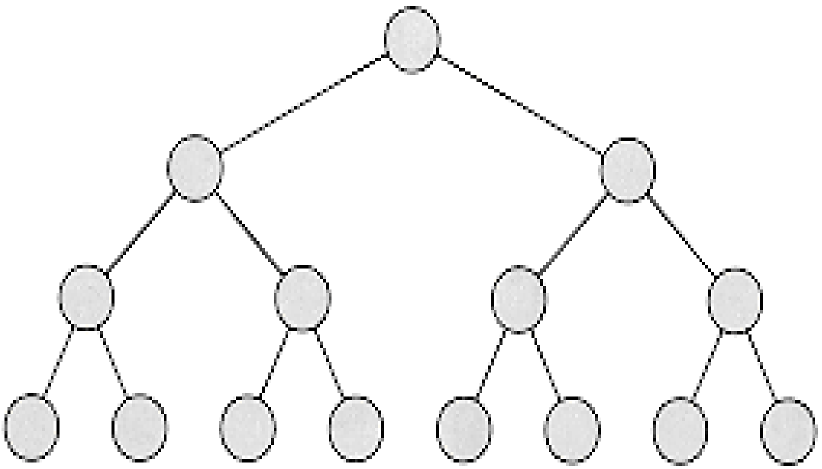
18

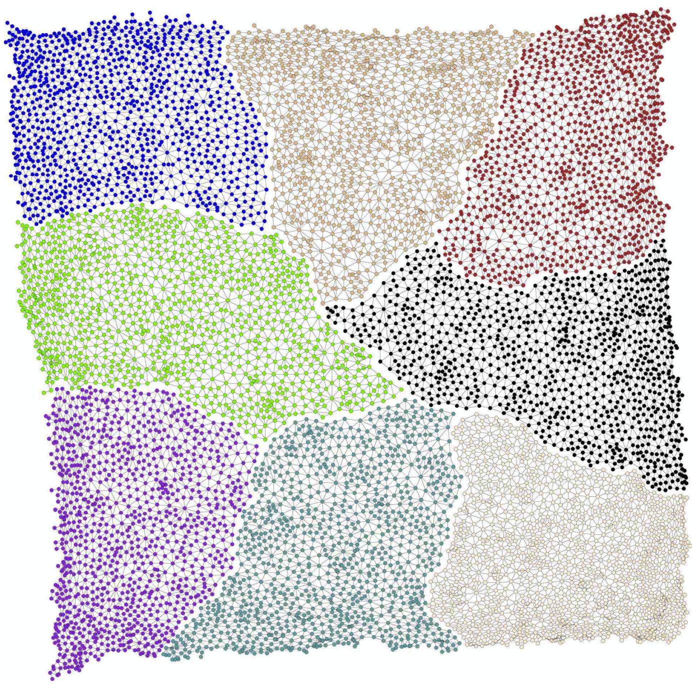
Backup Slides

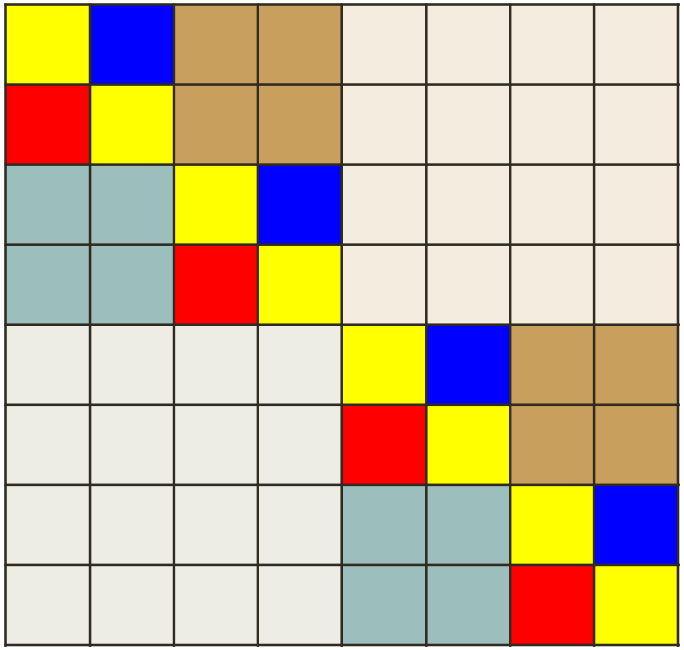
19

HLR: Tree and Matrix

- Recursively bisect the vertices (clustering)
- Corresponds to a binary tree
- Matrix: Low-rank approximation of off-diagonal blocks
  - Only if “well-separated”
  - How to choose ranks?
  - Use SVD, ACA, ULV, or RRQR?





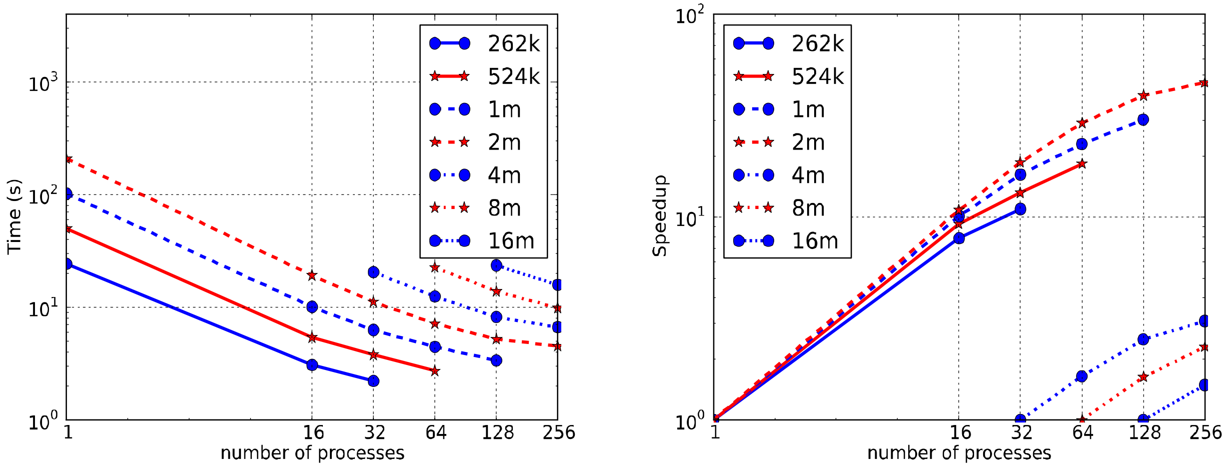


Our Hierarchical Low-Rank Sparse Solver

- Collaboration Darve (Stanford) & Sandia
- Solver based on recent H^2 methods by Darve et al.
  - IFMM (dense), LoRaSp (sparse)
- Uses block approximate LU factorization with low-rank compression for "well separated" interactions
  - Partition matrix, build H-tree, factor approximately
  - Leaves are subdomains, internal vertices correspond to approximate Schur complements (low rank)
  - Tree implicitly gives approximate LU factorization
- New method, differs from multifrontal HSS
  - Simpler, no trees within trees

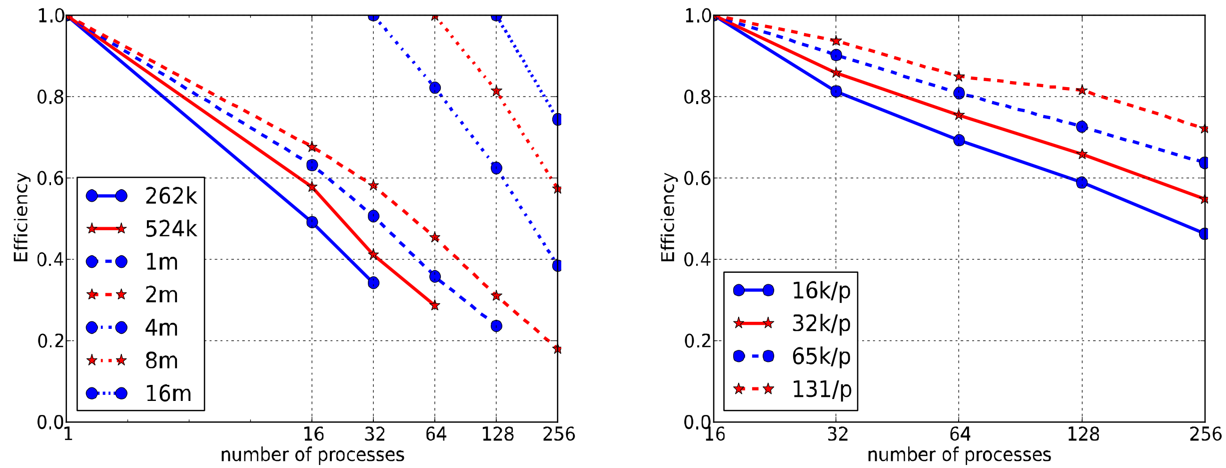
21

Results: Variable coeff. Poisson



(a) Time

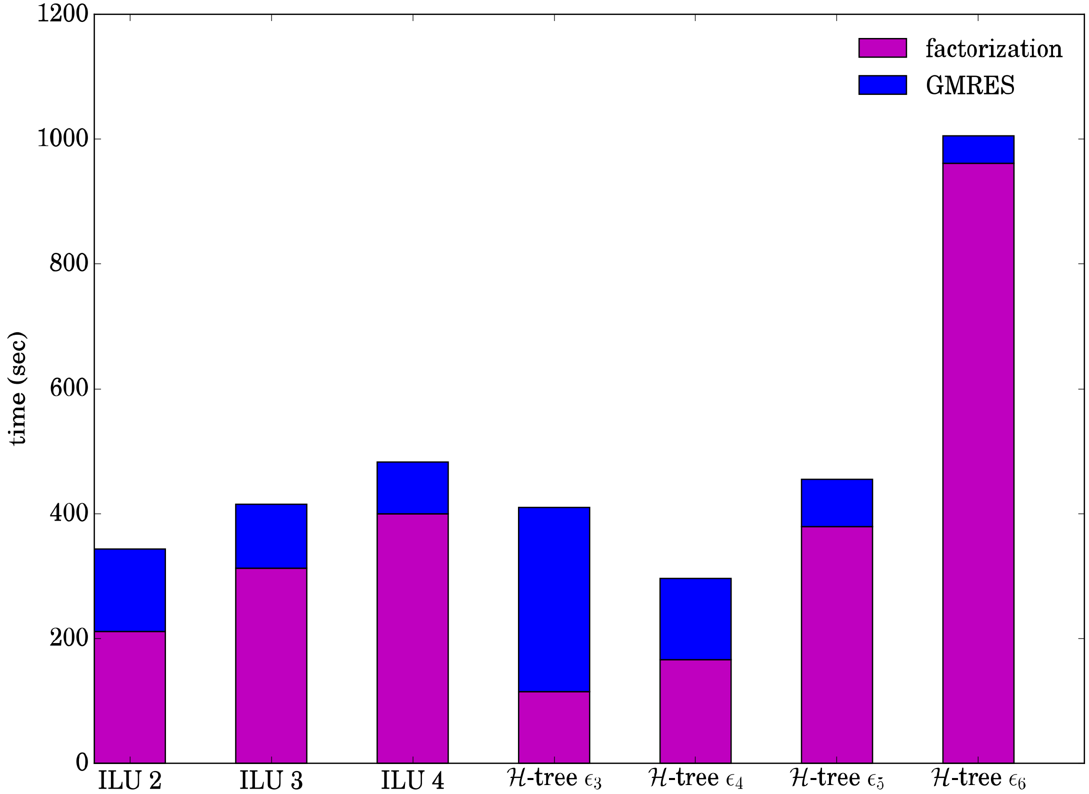
(b) Speedup



(c) Fixed total problem size

(d) Fixed problem size per processor

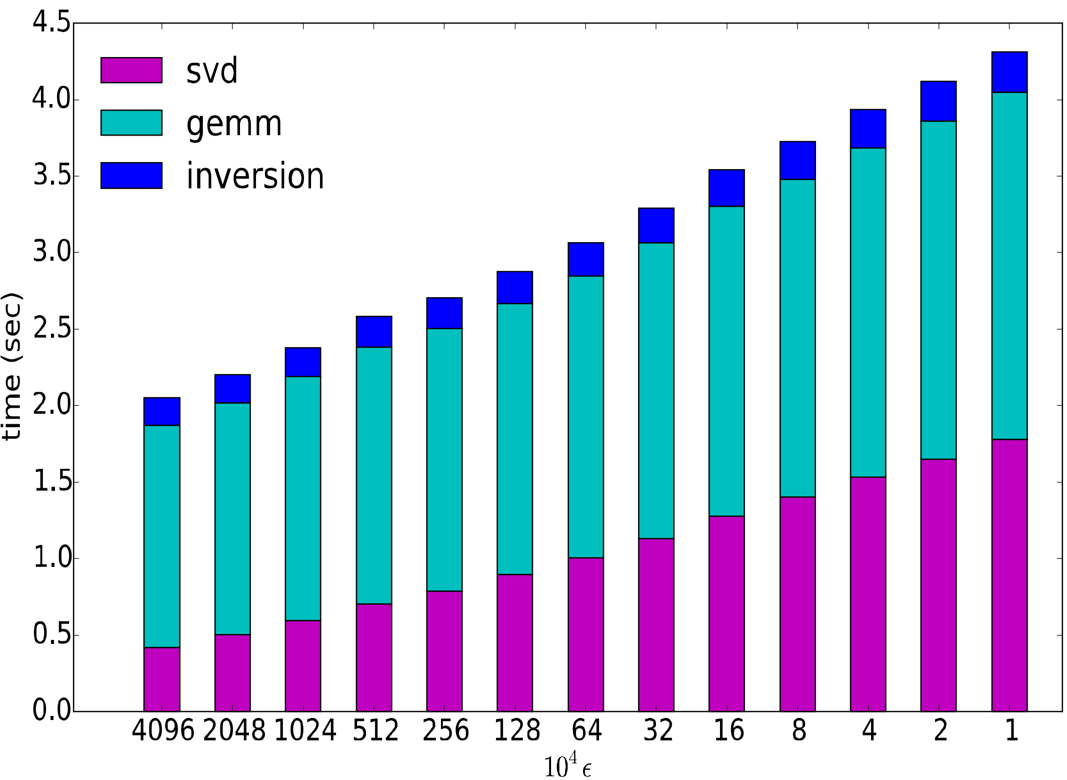
Accuracy Trade-Off



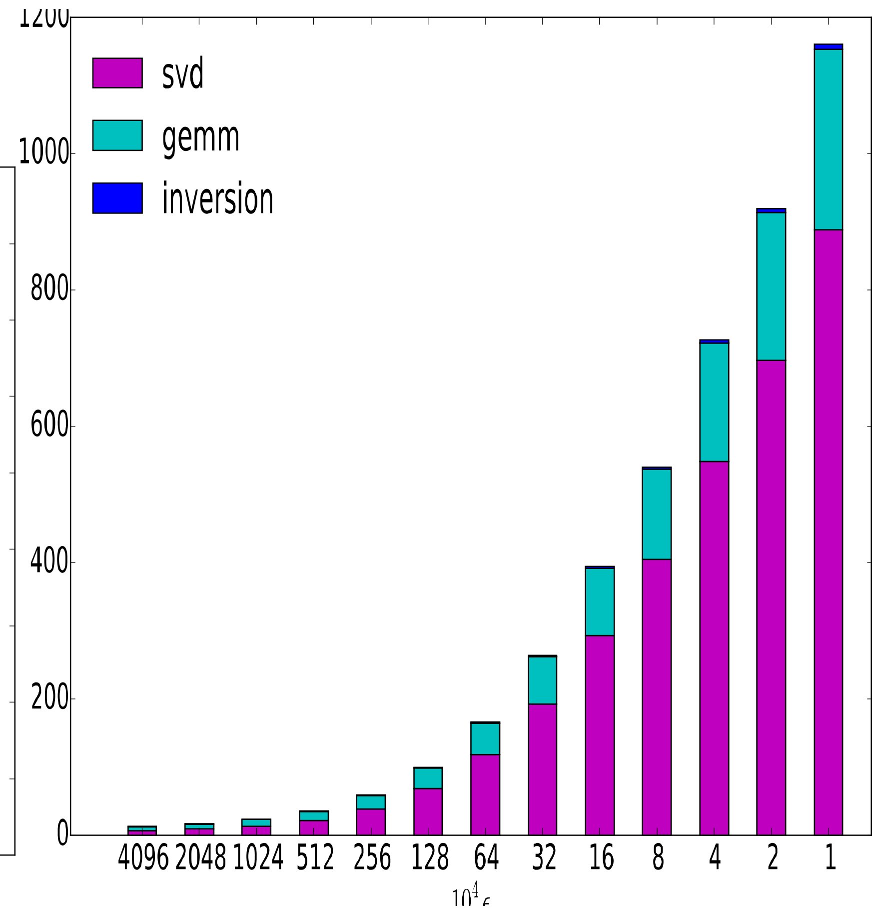
LoRaSp code (Pouransari et al); Poisson 3D

Timing Breakdown

LoRaSp code (Pouransari et al.)



2D PDE



3D PDE

24