Hierarchical Algorithms for Computational Linear Algebra

Yuval Harness - INRIA Team HiePACS

INVENTORS FOR THE DIGITAL WORLD

Joint work of the FastLA Associate Team members

The 6th annual Inria@SiliconValley workshop with California partners:
Paris June 8-10, 2016

The FastLA Associate Team

Fast and Scalable Hierarchical Algorithms for Computational Linear Algebra

Collaboration

- INRIA project-team HiePacs.
- Scientific Computing Group, LBNL.
- Mechanics and Computation Group, Stanford.

Theme

- Study & design hierarchical parallel & scalable numerical techniques.
- Applications: N-body interaction calculations and the solution of large sparse linear systems.
- Implementation: heterogeneous manycore platforms by using task based runtime systems.

Outline

Hierarchical Numerical Techniques

The Basics

Introduction

Motivation

- Problem: solution/factorization of extremeley large dense linear systems:

$$Ax = b$$

- Consider a matrix of dimesnion $n \times n$:

$$\begin{array}{lcl} \text{Sparse} & \Rightarrow & \mathcal{O}(n) \text{ storage units} \\ \text{Dense} & \Rightarrow & \mathcal{O}(n^2) \text{ storage units} \end{array}$$

- Memory consumption in the dense case is a major bottleneck in extending our capability to handle larger and more challenging linear systems.

Introduction

Motivation

- Problem: solution/factorization of extremeley large dense linear systems:

$$Ax = b$$

- Consider a matrix of dimesnion $n \times n$:

$$\begin{array}{rcl} \text{Sparse} & \Rightarrow & \mathcal{O}(n) \text{ storage units} \\ \text{Dense} & \Rightarrow & \mathcal{O}(n^2) \text{ storage units} \end{array}$$

- Memory consumption in the dense case is a major bottleneck in extending our capability to handle larger and more challenging linear systems.

Remarks

- In typical applications, forming A explicitly is prohibitive.
- Such matrices also emerege while solving large sparse systems.

Hierarchical Matrices

Hierarchical Matrix

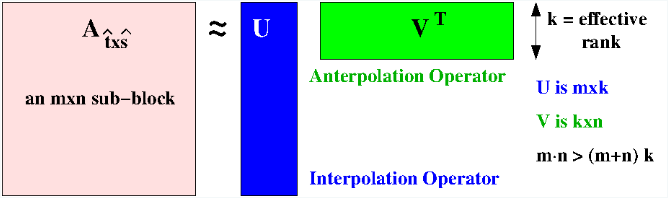
- **Hierarchical matrix (H-matrix)** is a data sparse approximation of a non-sparse matrix.

Hierarchical Matrices

Hierarchical Matrix

- **Hierarchical matrix (H-matrix)** is a data sparse approximation of a non-sparse matrix.

- **Basic principles**
  1. perform rows and columns permutations
  2. replace sub-blocks by low-rank factorizations.

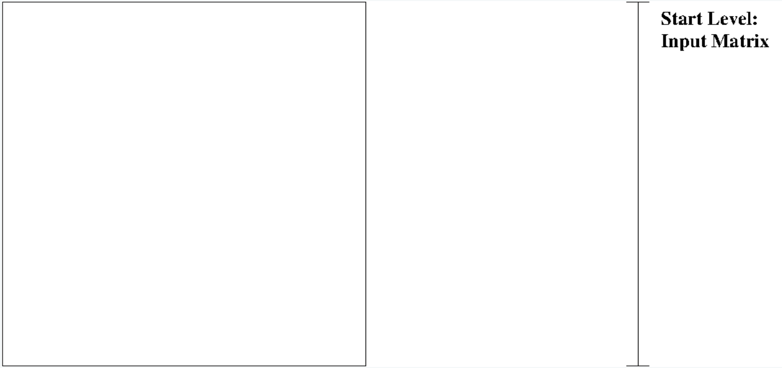


  $A_{\hat{t}\times\hat{s}}$ is a sub-block of $A \in \mathbb{F}^{N\times N}$, $\hat{s}, \hat{t} \subset \mathcal{I} = \{1, 2, ..., N\}$.

  3. **Hierarchical partitioning** $\Rightarrow$ almost linear complexity.

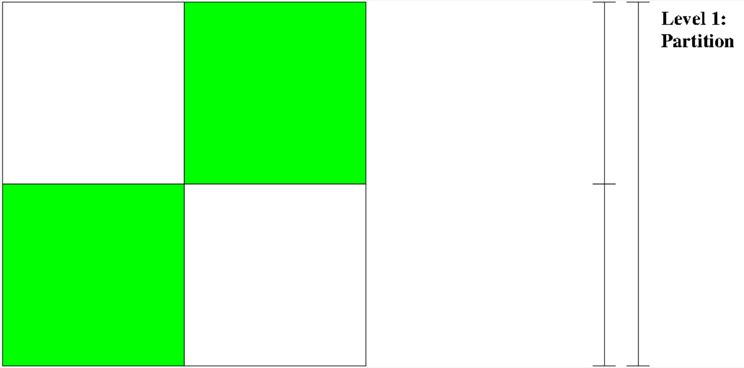
Hierarchical Partitioning

Strong Hierarchical Partitioning



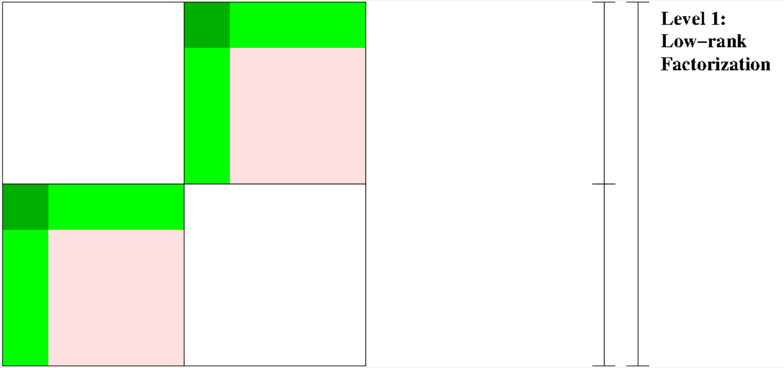
Hierarchical Partitioning

Strong Hierarchical Partitioning



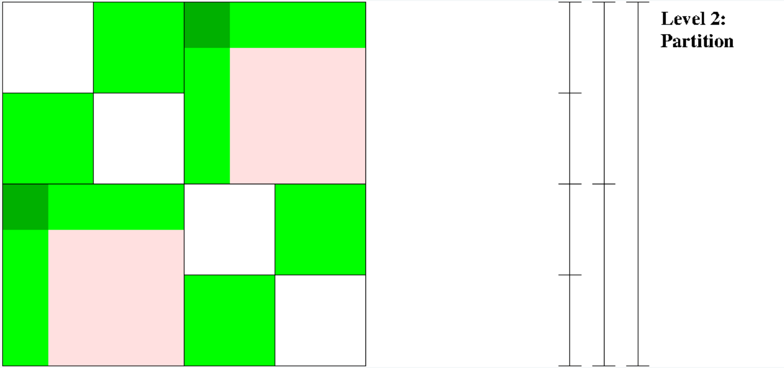
Hierarchical Partitioning

Strong Hierarchical Partitioning



Hierarchical Partitioning

Strong Hierarchical Partitioning

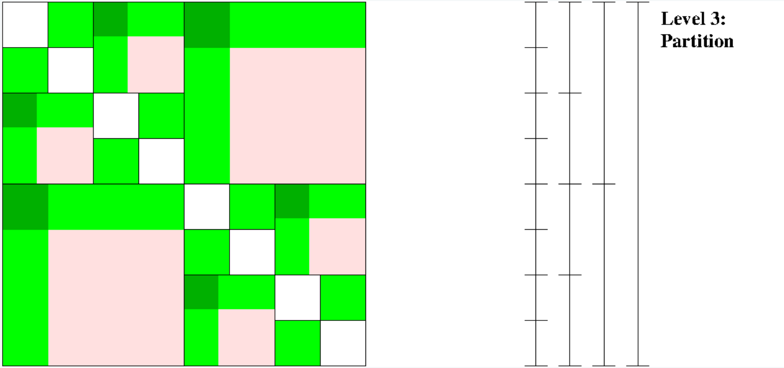


Hierarchical Partitioning

Strong Hierarchical Partitioning

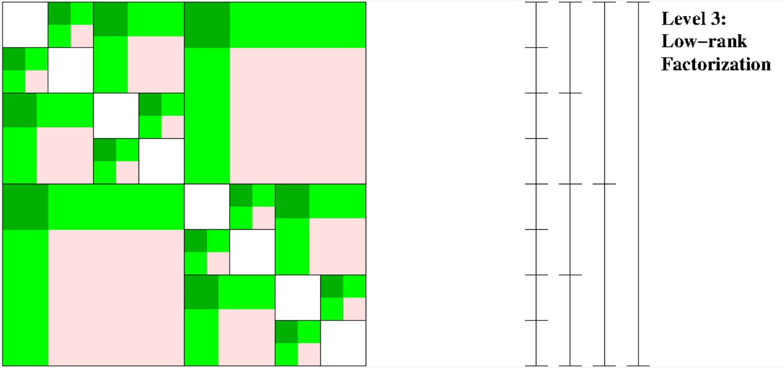
Hierarchical Partitioning

Strong Hierarchical Partitioning



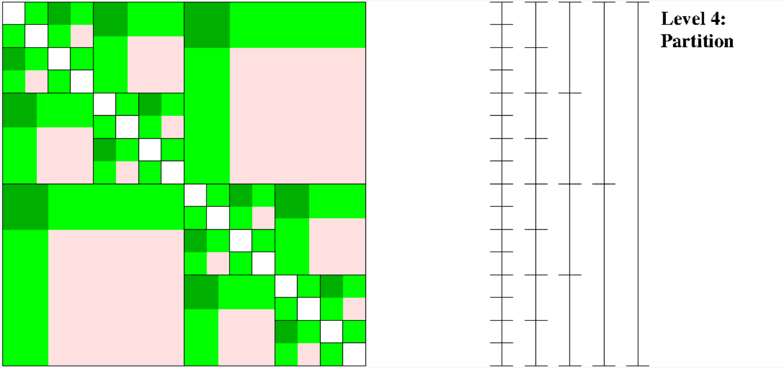
Hierarchical Partitioning

Strong Hierarchical Partitioning



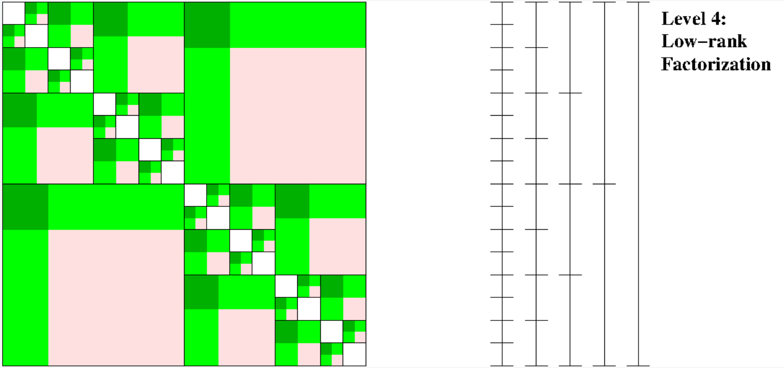
Hierarchical Partitioning

Strong Hierarchical Partitioning



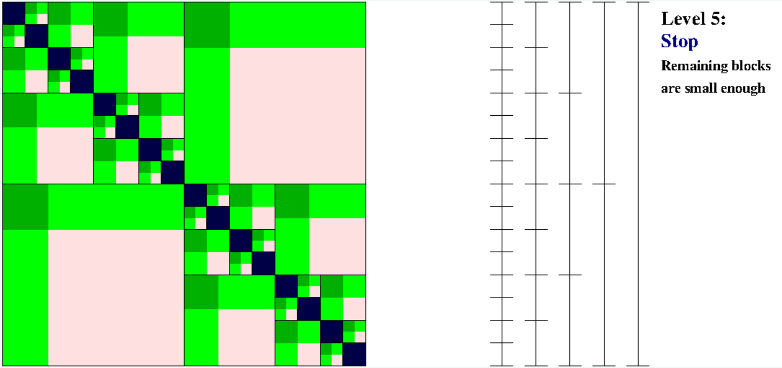
Hierarchical Partitioning

Strong Hierarchical Partitioning



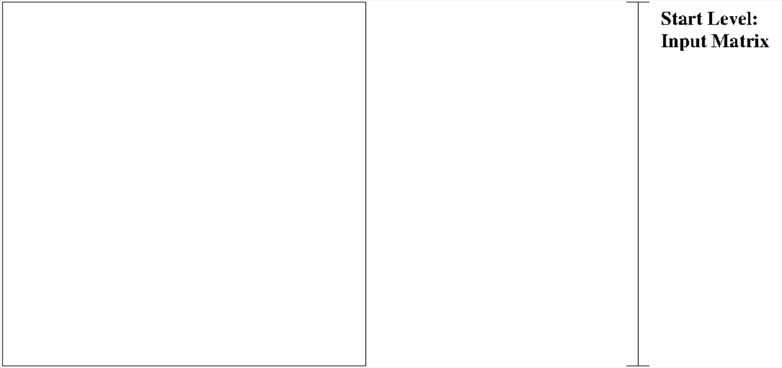
Hierarchical Partitioning

Strong Hierarchical Partitioning



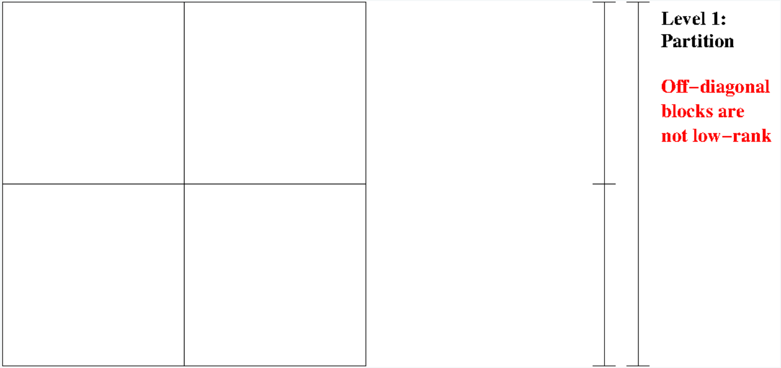
Hierarchical Partitioning

Weak Hierarchical Partitioning



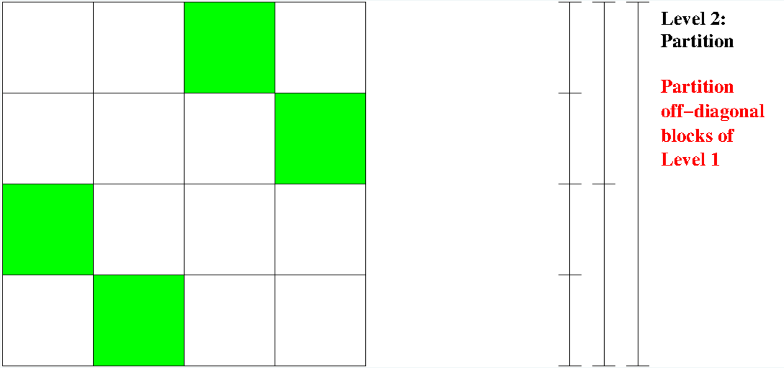
Hierarchical Partitioning

Weak Hierarchical Partitioning



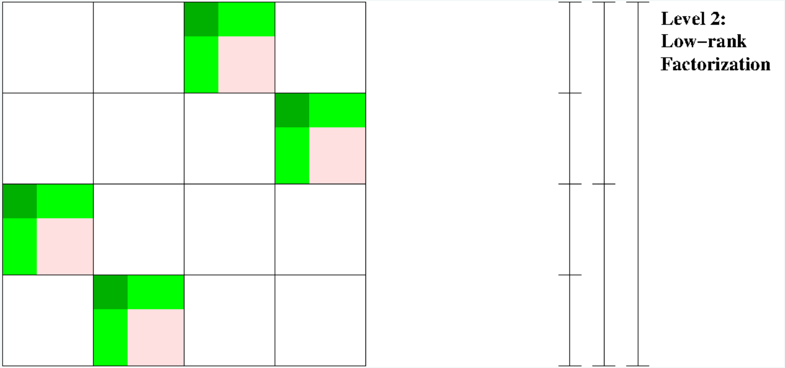
Hierarchical Partitioning

Weak Hierarchical Partitioning



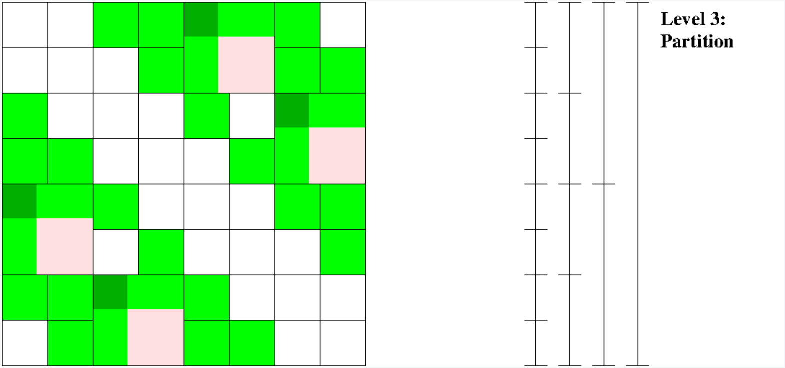
Hierarchical Partitioning

Weak Hierarchical Partitioning



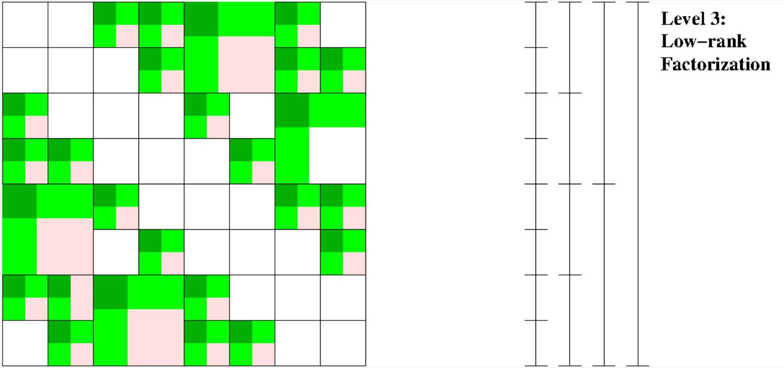
Hierarchical Partitioning

Weak Hierarchical Partitioning



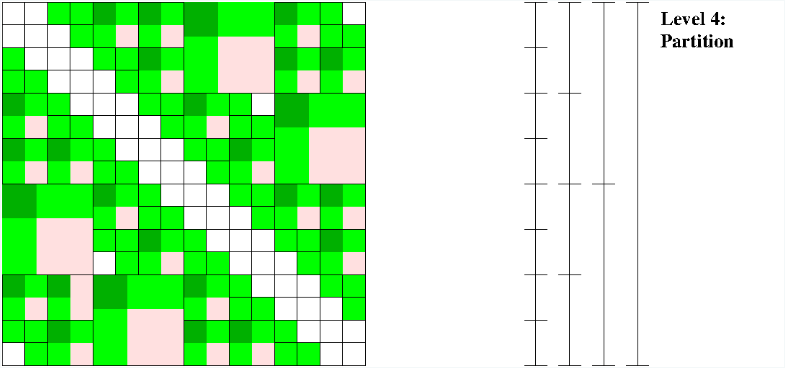
Hierarchical Partitioning

Weak Hierarchical Partitioning



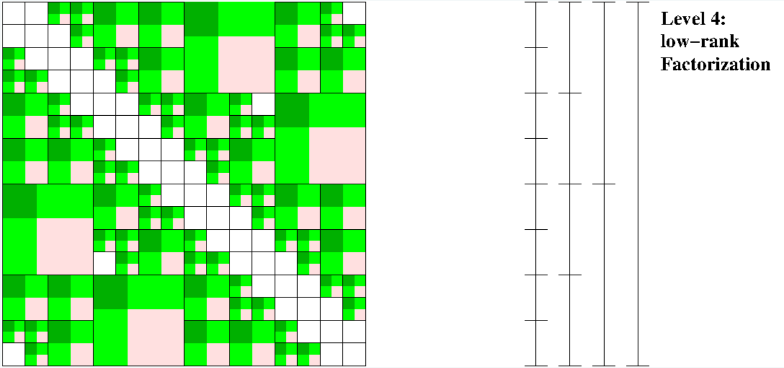
Hierarchical Partitioning

Weak Hierarchical Partitioning



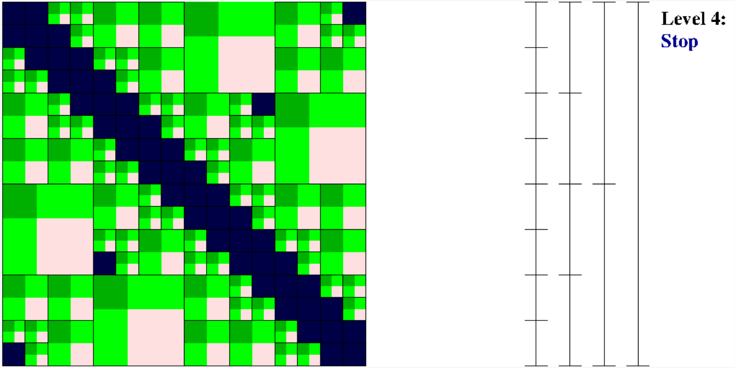
Hierarchical Partitioning

Weak Hierarchical Partitioning



Hierarchical Partitioning

Weak Hierarchical Partitioning



Compression, Complexity and Challenges

Matrix Compression

- Essentially H-matrix approximation is a matrix compression method.
- Works well for discretizations of Integral equations and elliptic PDEs.

Compression, Complexity and Challenges

Matrix Compression

- Essentially H-matrix approximation is a matrix compression method.
- Works well for discretizations of Integral equations and elliptic PDEs.

Complexity of operations

- Complexity of obtaining the hierarchical matrix should be almost linear:

$$\mathcal{O}\left(N \log^{\alpha} N\right), \quad \alpha \text{ is 'small'}$$

- Arithmetics $(+, -, \cdot, \mathrm{inv})$ should be possible in almost linear complexity.
- Hierarchical techniques a.k.a. Fast hierarchical methods.

Compression, Complexity and Challenges

Matrix Compression

- Essentially H-matrix approximation is a matrix compression method.
- Works well for discretizations of Integral equations and elliptic PDEs.

Complexity of operations

- Complexity of obtaining the hierarchical matrix should be almost linear:

$$\mathcal{O}\left(N \log^{\alpha} N\right), \quad \alpha \text{ is 'small'}$$

- Arithmetics $(+, -, \cdot, \mathrm{inv})$ should be possible in almost linear complexity.
- Hierarchical techniques a.k.a. Fast hierarchical methods.

Challenges

- Fast identification & factorization of low-rank structures.
- Prohibitively expensive to form the large dense blocks.

Fast Methods for Geostatistics

H-matrix accelerated Randomized SVD

Introduction

Collaborators

Pierre Blanchard, Olivier Coulaud (INRIA) & Eric Darve (Stanford)

Problem: Generation of Gaussian Random Fields

- $\mathbf{Y} \sim \mu(\mathbf{0}, \mathbf{C})$ is a multivariate Gaussian random field (GRF).
- The covariance $\mathbf{C} \in \mathbb{R}^{N \times N}$ can be prescribed as a kernel matrix

$$\mathbf{C} = \{k(\|\mathbf{x}_i - \mathbf{x}_j\|_2)\}_{i,j=1\dots N}$$

  - $\mathbf{x}$: large and highly heterogenous 3D grid
  - k: correlation kernel such as

$$k_{1/2}(r) = e^{-r/\ell} \quad \text{or} \quad k_{\infty}(r) = e^{-r^2/(2\ell^2)}$$

- Generating $\mathbf{Y}$ requires computing a square root $\mathbf{A}$

$$\mathbf{C} = \mathbf{A}\mathbf{A}^{\mathsf{T}} \quad \rightarrow \quad \mathbf{Y} = \mathbf{A} \cdot \mathbf{X} \quad : \quad \mathbf{X} \sim \mu(\mathbf{0}, \mathbf{I}_N)$$

H-matrix accelerated Randomized SVD

Standard Methods: (Often become computationally prohibitive for large N)

- Cholesky ($\mathcal{O}(N^3)$).
- circulant embedding ($\mathcal{O}(N \log N)$ for equispaced grids)
- turning bands method (approximate).

H-matrix accelerated Randomized SVD

Standard Methods: (Often become computationally prohibitive for large N)

- Cholesky ($\mathcal{O}(N^3)$).
- circulant embedding ($\mathcal{O}(N \log N)$ for equispaced grids)
- turning bands method (approximate).

Solution: H-matrix accelerated Randomized SVD

- Randomized range evaluation:

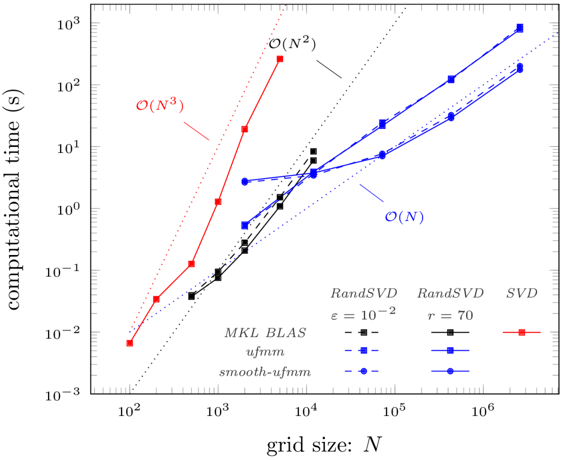
$$\mathbf{Z} = \mathbf{C} \cdot \mathbf{\Omega} \quad : \quad \mathbf{\Omega} \in \mathbb{R}^{N \times r} \text{ is a random Gaussian matrix}.$$

- Approximate Square root:

$$\mathbf{Z} = \mathbf{QR} \quad \rightarrow \quad \mathbf{A} = \mathbf{QU\Sigma}^{1/2} \, : \, \mathbf{U\Sigma U}^T = \mathbf{Q}^T\mathbf{CQ} \in \mathbb{R}^{r \times r}.$$

- H-matrix matrix product acceleration: $\mathbf{C} \rightarrow \mathbf{C}^{\mathcal{H}}$
  - Approximating A in $\mathcal{O}(r^2 \times N)$ operations.
  - Matrix-free method with $\mathcal{O}(r \times N)$ memory footprint.
  - Handles highly heterogeneous grids more efficiently than standard methods.

H-matrix accelerated Randomized SVD: time=f(n)



H-Matrices in Sparse Direct Solvers

Low-rank Operations in a Supernodal Solver

Introduction

Collaborators

Gregoire Pichon, Mathieu Faverge, Pierre Ramet, Jean Roman (INRIA) & Eric Darve (Stanford)

Problem: Solve $Ax = b$ where $A = A^T$ is large and sparse

- Cholesky: factorize $A = LL^T$ (symmetric pattern for LU)
- Solve $Ly = b$
- Solve $L^T x = y$

Introduction

Collaborators

Gregoire Pichon, Mathieu Faverge, Pierre Ramet, Jean Roman (INRIA) & Eric Darve (Stanford)

Problem: Solve $Ax = b$ where $A = A^T$ is large and sparse

- Cholesky: factorize $A = LL^T$ (symmetric pattern for LU)
- Solve $Ly = b$
- Solve $L^T x = y$

Solution: Direct Solver

- Expensive with respect to iterative solvers.
- More robust, and allow to tackle hard problems.
- "Fill-ins" $\Rightarrow$ dense blocks $\Rightarrow$ high memory consumption.

Reducing Fill-ins with Nested Dissection

Objective

- Reorder A to reduce Fill-ins.

Nested Dissection

- Associate A as a graph: $G = (V, E, \sigma_p)$

  V: vertices, E: edges, σ_p: unknowns permutation
- The Algorithm: (computing σ_p)
  1. Partition $V = A \cup B \cup C$
  2. Order C with larger numbers: $V_A < V_C$ and $V_B < V_C$
  3. Apply the process recursively on A and B

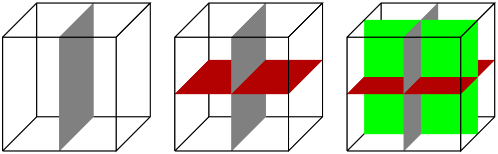
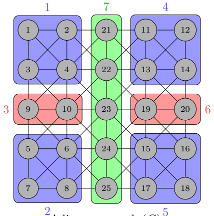


Figure : Three-levels of nested dissection on a regular cube

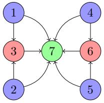
Reducing Fill-ins with Nested Dissection

Symbolic Factorization

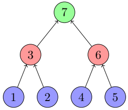
1. Build a partition with the nested dissection process.
2. Compute block elimination tree thanks to the block quotient graph.



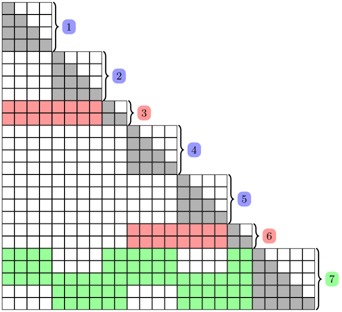
Adjacency graph (G).



Quotient graph $(G^*/P) = (G/P)^*$



Elimination tree (T).



Factorized matrix (L).

Block-Low-Rank Compression

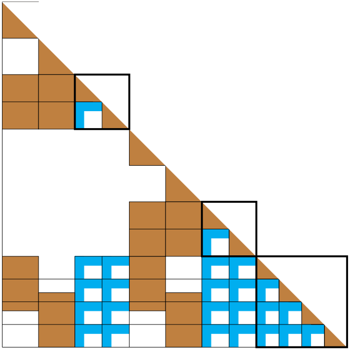
Definition

Block-Low-Rank (BLR) compression of a dense block:

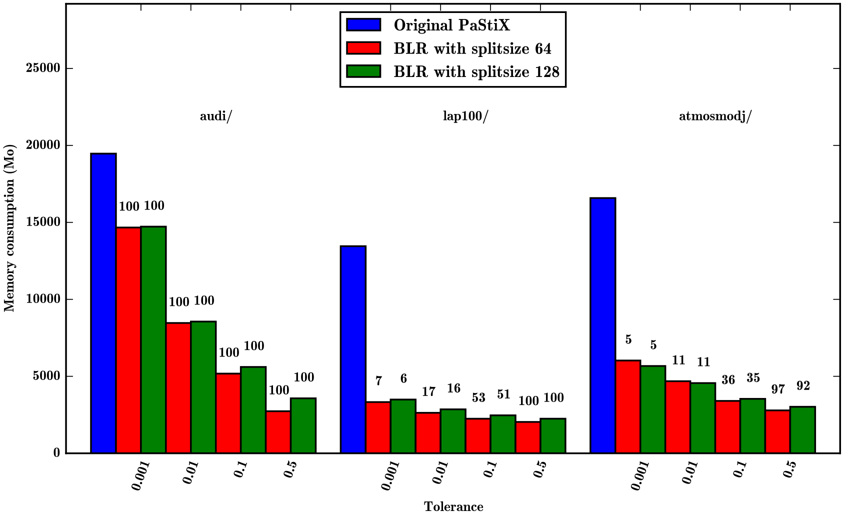
- dividing the block into equally sized sub-blocks.
- replacing each sub-block by a low-rank factorization.

Current Implementation

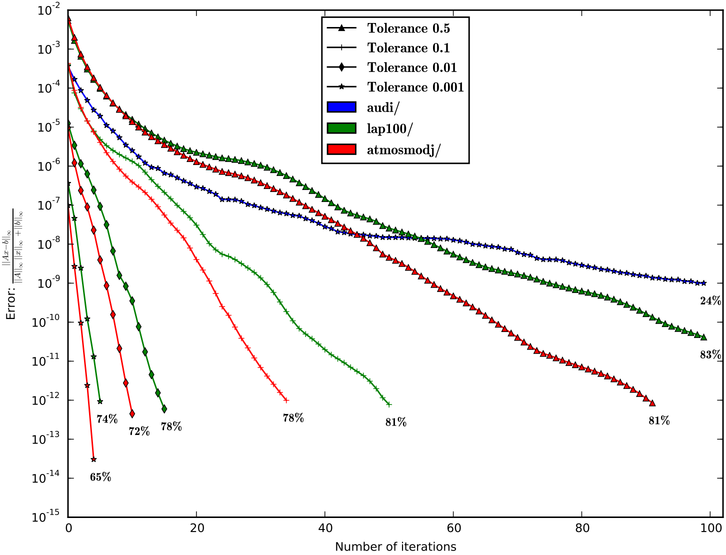
- Use BLR representation for large off-diagonal blocks
- Ordering strategy and kernels will form the foundation for future extensions.



Memory Consumption depending on Tolerance



Accuracy depending on Tolerance, Blocksize=128



Hierarchical Multilevel Preconditioning

Spectral Analysis

Introduction

Collaborators

Yuval Harness, Emanuel Agullo, Luc Giraud (INRIA) &
Eric Darve (Stanford)

Problem: Solve $Ax = b$ where $A = A^T > 0$ is **extremely** large and sparse

- The problem is too big for a direct solver.

Introduction

Collaborators

Yuval Harness, Emanuel Agullo, Luc Giraud (INRIA) &
Eric Darve (Stanford)

Problem: Solve $Ax = b$ where $A = A^T > 0$ is **extremely** large and sparse

- The problem is too big for a direct solver.

Solution: Algebraic Domain Decomposition

$$A = \left(\begin{array}{cccc|c} A_{I_1 I_1} & & & & A_{I_1 \Gamma} \\ & A_{I_2 I_2} & & & A_{I_2 \Gamma} \\ & & \ddots & & \vdots \\ & & & A_{I_p I_p} & A_{I_p \Gamma} \\ \hline A_{\Gamma I_1} & A_{\Gamma I_2} & \cdots & A_{\Gamma I_p} & A_{\Gamma \Gamma} \end{array}\right)$$

- Each $A_{I_j I_j}$ can be inverted in parallel by a direct solver.

The Schur System

The (Global) Schur System

$$Ax = \begin{pmatrix} A_{II} & A_{I\Gamma} \\ A_{\Gamma I} & A_{\Gamma\Gamma} \end{pmatrix} \begin{pmatrix} x_I \\ x_\Gamma \end{pmatrix} = \begin{pmatrix} b_I \\ b_\Gamma \end{pmatrix}$$

- $A_{II} \leftrightarrow$ interior subdomains, $A_{\Gamma\Gamma} \leftrightarrow$ separators.
- If x_Γ is known $\Rightarrow x_I = A_{II}^{-1}(b_I - A_{I\Gamma}x_\Gamma)$.
- The dense Schur system: $Sx_\Gamma = b_\Gamma$, $S = A_{\Gamma\Gamma} - A_{\Gamma I}A_{II}^{-1}A_{I\Gamma}$.

The Schur System

The (Global) Schur System

$$Ax = \begin{pmatrix} A_{II} & A_{I\Gamma} \\ A_{\Gamma I} & A_{\Gamma\Gamma} \end{pmatrix} \begin{pmatrix} x_I \\ x_\Gamma \end{pmatrix} = \begin{pmatrix} b_I \\ b_\Gamma \end{pmatrix}$$

- $A_{II} \leftrightarrow$ interior subdomains, $A_{\Gamma\Gamma} \leftrightarrow$ separators.
- If x_Γ is known $\Rightarrow x_I = A_{II}^{-1}(b_I - A_{I\Gamma}x_\Gamma)$.
- The dense Schur system: $Sx_\Gamma = b_\Gamma$, $S = A_{\Gamma\Gamma} - A_{\Gamma I}A_{II}^{-1}A_{I\Gamma}$.

Iterative Solution

- $Sx_\Gamma = b_\Gamma$ is solved iteratively.
- $\kappa(S) < \kappa(A)$.
- S is never formed, but assembled at each iteration:

$$Sx_\Gamma = \sum_{i=1}^{N} R_i^T S_i R_i x_\Gamma \quad : \quad R_i : \mathbb{R}^N \to \mathbb{R}^{n_i}$$

- All the local components, $\{S_i\}$, are computed in parallel.

Hierarchical Matrix Preconditioning

Motivation

- Schur system $\sim$ preconditioning.
- Further preconditioning for Krylov iterations.

Hierarchical Matrix Preconditioning

Motivation

- Schur system $\sim$ preconditioning.
- Further preconditioning for Krylov iterations.

Hierarchical Matrix Preconditioning

- Let $\widehat{S}$ be an H-matrix approximation of $S = S^T > 0$.
- The preconditioned system: $\widehat{S}^{-1/2} S \widehat{S}^{-1/2} y = \widehat{S}^{-1/2} b$.
- How do we guarantee $\widehat{S}$ is SPD as well?
- Can we estimate/control the condition number?

Hierarchical Matrix Preconditioning

Motivation

- Schur system $\sim$ preconditioning.
- Further preconditioning for Krylov iterations.

Hierarchical Matrix Preconditioning

- Let $\widehat{S}$ be an H-matrix approximation of $S = S^T > 0$.
- The preconditioned system: $\widehat{S}^{-1/2} S \widehat{S}^{-1/2} y = \widehat{S}^{-1/2} b$.
- How do we guarantee $\widehat{S}$ is SPD as well?
- Can we estimate/control the condition number?

Notes

- Consider S be close to singularity: $\|S\| < \epsilon$.
- If $\|S - \widehat{S}\| \geq \epsilon \Rightarrow \widehat{S}$ can be arbitrarily close to singularity.
- We want $\widehat{S}$ to be inaccurate as possible.

Two-Level Analysis

Objective: estimation of spectral bounds

$$\alpha := \inf_{x \neq 0} \frac{x^T \widehat{S} x}{x^T S x}, \quad \beta := \sup_{x \neq 0} \frac{x^T \widehat{S} x}{x^T S x}.$$

- $\alpha > 0 \Leftrightarrow \widehat{S}$ is SPD.
- β / α is the spectral condition number, $\kappa(\widehat{S}^{-1/2} S \widehat{S}^{-1/2})$.

Two-Level Analysis

Objective: estimation of spectral bounds

$$\alpha := \inf_{x\neq 0} \frac{x^T\widehat{S}x}{x^TSx}, \quad \beta := \sup_{x\neq 0} \frac{x^T\widehat{S}x}{x^TSx}.$$

- $\alpha > 0 \Leftrightarrow \widehat{S}$ is SPD.
- β/α is the spectral condition number, $\kappa(\widehat{S}^{-1/2}S\widehat{S}^{-1/2})$.

The Two-Level Problem

$$S = \left(\begin{array}{c|c} S_1 & M \\ \hline M^T & S_2 \end{array}\right), \quad \widehat{A} = \left(\begin{array}{c|c} \widehat{S}_1 & \widehat{M} \\ \hline \widehat{M}^T & \widehat{S}_2 \end{array}\right),$$

- The matrices S, S_1 and S_2 are SPD.
- The matrices $\widehat{S}_1$ and $\widehat{S}_2$ are symmetric as well, and satisfy:

$$\forall x_i \;\; 0 < \alpha_i \leq \frac{x_i^T\widehat{S}_i x_i}{x_i^T S_i x_i} \leq \beta_i \quad \Leftrightarrow \quad \alpha_i S_i \leq \widehat{S}_i \leq \beta_i S_i .$$

Two-Level Analysis

Assumptions

Let $\widehat{M}$ be a GSVD truncation of M,

$$\widehat{M} = \widehat{S}_1^{1/2}\widehat{\mathcal{M}}\widehat{S}_2^{1/2} \quad : \quad \widehat{\mathcal{M}} = \mathcal{U}_p \Sigma_p \mathcal{V}_p^T \approx \mathcal{M} = \widehat{S}_1^{-1/2} M \widehat{S}_2^{-1/2},$$

and assume that $\underline{S} = \left(\begin{array}{c|c} \frac{1}{\beta_1}\widehat{S}_1 & M \\ \hline M^T & \frac{1}{\beta_2}\widehat{S}_2 \end{array}\right) > 0.$

Two-Level Analysis

Assumptions

Let $\widehat{M}$ be a GSVD truncation of M,

$$\widehat{M} = \widehat{S}_1^{1/2} \widehat{\mathcal{M}} \widehat{S}_2^{1/2} \quad : \quad \widehat{\mathcal{M}} = \mathcal{U}_p \Sigma_p \mathcal{V}_p^T \approx \mathcal{M} = \widehat{S}_1^{-1/2} M \widehat{S}_2^{-1/2},$$

and assume that $\underline{S} = \left(\begin{array}{c|c} \frac{1}{\beta_1}\widehat{S}_1 & M \\ \hline M^T & \frac{1}{\beta_2}\widehat{S}_2 \end{array} \right) > 0.$

Main Result

$$\frac{x^T \widehat{S} x}{x^T S x} \leq \max\left\{ \frac{\beta_{\max} - \sqrt{\beta_1\beta_2}\sigma_1}{1 - \sqrt{\beta_1\beta_2}\sigma_1}, \frac{\beta_{\max}}{1 - \sqrt{\beta_1\beta_2}\sigma_{p+1}} \right\} \geq \beta_{\max},$$

$$\frac{x^T \widehat{S} x}{x^T S x} \geq \min\left\{ \frac{\alpha_{\min} - \sqrt{\alpha_1\alpha_2}\sigma_1}{1 - \sqrt{\alpha_1\alpha_2}\sigma_1}, \frac{\alpha_{\min}}{1 + \sqrt{\alpha_1\alpha_2}\sigma_{p+1}} \right\} \leq \alpha_{\min}.$$

The singular value of $\mathcal{M}$: $\sigma_1 \geq \sigma_2 \geq \ldots$.

Multi-Level Implementation

The Multi-Level case

$$\widehat{S} = \widehat{S}_1^{(0)}, \quad \widehat{S}_k^{(\ell)} = \left(\begin{array}{c|c} \widehat{S}_{2k-1}^{(\ell+1)} & \widehat{M}_k^{(\ell)} \\ \hline \widehat{M}_k^{(\ell)^T} & \widehat{S}_{2k}^{(\ell+1)} \end{array} \right) \in \mathbb{R}^{n_k^{(\ell)} \times n_k^{(\ell)}},$$

$$S = S_1^{(0)}, \quad S_k^{(\ell)} = \left(\begin{array}{c|c} S_{2k-1}^{(\ell+1)} & M_k^{(\ell)} \\ \hline M_k^{(\ell)^T} & S_{2k}^{(\ell+1)} \end{array} \right) \in \mathbb{R}^{n_k^{(\ell)} \times n_k^{(\ell)}},$$

Recursive Estimation

$$\alpha_k^{(\ell)} = \frac{1 - \sqrt{\frac{\alpha_{k,\max}^{(\ell)}}{\alpha_{k,\min}^{(\ell)}}} \sigma_{k,1}^{(\ell)}}{\frac{1}{\alpha_{k,\min}^{(\ell)}} - \sqrt{\frac{\alpha_{k,\max}^{(\ell)}}{\alpha_{k,\min}^{(\ell)}}} \sigma_{k,1}^{(\ell)}} \leq \frac{1 - \sqrt{\frac{\beta_{k,\min}^{(\ell)}}{\beta_{k,\max}^{(\ell)}}} \sigma_{k,1}^{(\ell)}}{\frac{1}{\beta_{k,\max}^{(\ell)}} - \sqrt{\frac{\beta_{k,\min}^{(\ell)}}{\beta_{k,\max}^{(\ell)}}} \sigma_{k,1}^{(\ell)}} = \beta_k^{(\ell)},$$

where $\alpha_{k,\min}^{(\ell)} = \min\left\{\alpha_{2k-1}^{(\ell+1)}, \alpha_{2k}^{(\ell+1)}\right\}$, $\beta_{k,\max}^{(\ell)} = \max\left\{\beta_{2k-1}^{(\ell+1)}, \beta_{2k}^{(\ell+1)}\right\}$.

Concluding Remarks

Summary, Conclusions and Future Study

Concluding Remarks

Summary

- Hierarchical Matrices.
- Applications & Challenges in Computational Linear Algebra.

Concluding Remarks

Summary

- Hierarchical Matrices.
- Applications & Challenges in Computational Linear Algebra.

Current Challenges

- Direct Solvers: move from BLR to better compression schemes.
- Preconditioning: Optimal (adaptive) H-matrix preconditioner.
- Main difficulty is to do it in a 'reasonable' complexity.

Concluding Remarks

Summary

- Hierarchical Matrices.
- Applications & Challenges in Computational Linear Algebra.

Current Challenges

- Direct Solvers: move from BLR to better compression schemes.
- Preconditioning: Optimal (adaptive) H-matrix preconditioner.
- Main difficulty is to do it in a 'reasonable' complexity.

Future Plans

- Exascale simulations.
- More realistic/industrial problems.

Concluding Remarks

Summary

- Hierarchical Matrices.
- Applications & Challenges in Computational Linear Algebra.

Current Challenges

- Direct Solvers: move from BLR to better compression schemes.
- Preconditioning: Optimal (adaptive) H-matrix preconditioner.
- Main difficulty is to do it in a 'reasonable' complexity.

Future Plans

- Exascale simulations.
- More realistic/industrial problems.

Thank You