Wine or Brandy?
Block low rank vs block separable matrices

Cleve Ashcraft

Livermore Software Technology Corporation

14th Scheduling for Large Scale Systems Workshop
Bordeaux, June 26 – 28, 2019

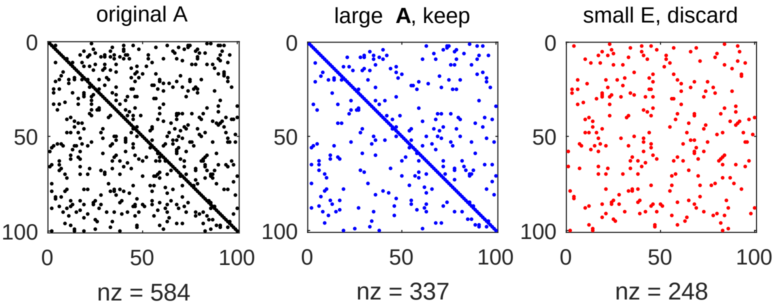
Livermore Software Technology Corporation, est. 1986.

- engineering simulation software
- linear algebra group
  - Cleve Ashcraft
  - Roger Grimes
  - Bob Lucas
  - François-Henry Rouet
  - Eugene Vecharynski
  - Clement Weisbecker
- electromechanics
  - Pierre L'Eplattenier
- transient acoustic fluid-structure interaction
  - Tom Littlewood

Drop tolerance approximation

Pointwise approximation $\quad A = \mathbf{A} + E$

- **disjoint nonzero structure,** $\quad \mathbf{a}_{k,j} \neq 0 \iff e_{k,j} = 0$



Drop tolerance approximation

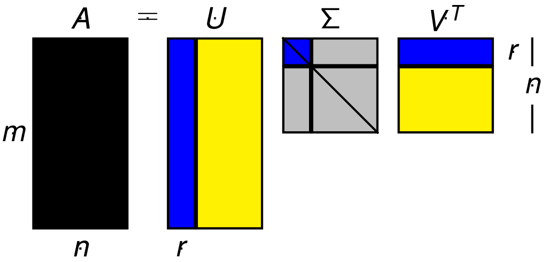
Pointwise approximation $A = \mathbf{A} + E$

- **absolute tolerance**, $$\mathbf{A}_{k,j} \neq 0 \Longrightarrow |a_{k,j}| \geq \epsilon$$
- **relative tolerance w.r.t. global max element** $$\mathbf{A}_{k,j} \neq 0 \Longrightarrow \frac{|a_{k,j}|}{\max_{i,l} |a_{i,l}|} \geq \epsilon$$
- **relative tolerance w.r.t. diagonal elements** $$\mathbf{A}_{k,j} \neq 0 \Longrightarrow \frac{|a_{k,j}|}{\sqrt{|a_{k,k} a_{j,j}|}} \geq \epsilon$$
- LSTC 2004 : did **not** work for BEM matrices, there are not many small entries to drop.
- Sparsification with a $\{0, 1\}$ function **did** not work.

We need a different definition of sparsity.

A low rank approximation from the SVD

From the singular value decomposition (SVD) $A = U \Sigma V^T$



- U and V have orthonormal columns, $U^T U = I$, $V^T V = I$
- Σ diagonal, real and nonnegative
- grey — too small, yellow — ignored, blue — kept

Block Matrices : BLR – Block Low Rank

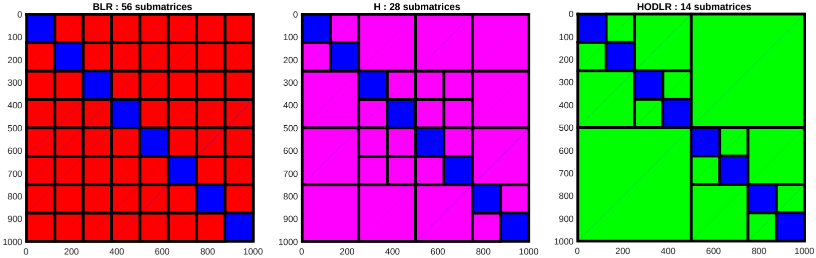
- Our matrix A has a block row and column structure.

$$A = A_{\mathcal{M},\mathcal{M}} = \begin{bmatrix} A_{\mathcal{M}_1,\mathcal{M}_1} & A_{\mathcal{M}_1,\mathcal{M}_2} & \cdots & A_{\mathcal{M}_1,\mathcal{M}_N} \\ A_{\mathcal{M}_2,\mathcal{M}_1} & A_{\mathcal{M}_2,\mathcal{M}_2} & \cdots & A_{\mathcal{M}_2,\mathcal{M}_N} \\ \vdots & \vdots & \ddots & \vdots \\ A_{\mathcal{M}_N,\mathcal{M}_1} & A_{\mathcal{M}_N,\mathcal{M}_2} & \cdots & A_{\mathcal{M}_N,\mathcal{M}_N} \end{bmatrix} \tag{1}$$

- Submatrices are **atomically** distributed across processors
- Submatrices $A_{\mathcal{M}_i,\mathcal{M}_i}$ on the diagonal are square, dense, and well-conditioned.
- Off-diagonal submatrices are **dense**,
  but **most** have **small** numerical rank,
  although a **few** have **large** numerical rank

Blocking Strategies

- Dense matrices from boundary element methods
  — electromagnetics, acoustics, heat transfer

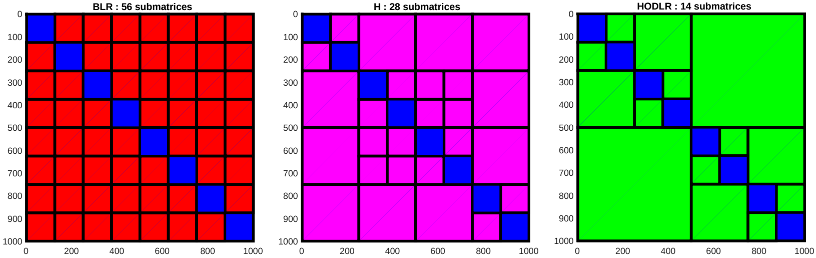


BLR H HODLR

- BLR is a **simplification** of H-matrices
  diagonal blocks are dense,
  off-diagonal blocks are dense, zero, or lowrank, (or sparse)
- 2004 : BLR in LS-DYNA for BEM matrices
- 2012 : BLR in MUMPS for sparse matrices

BLR – Block Low Rank

- Dense matrices from boundary element methods
  — electromagnetics, acoustics, heat transfer



BLR H HODLR

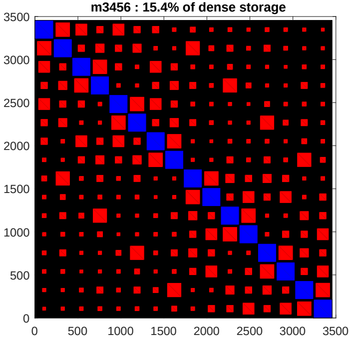
- BLR in LS-DYNA for BEM matrices

2004 : serial *LDU* factor and solve,
2005 : MPI matrix splitting PCG/GMRES, MPI mmm and mvm
2018 : pointwise MPP *LDU* factor and solve

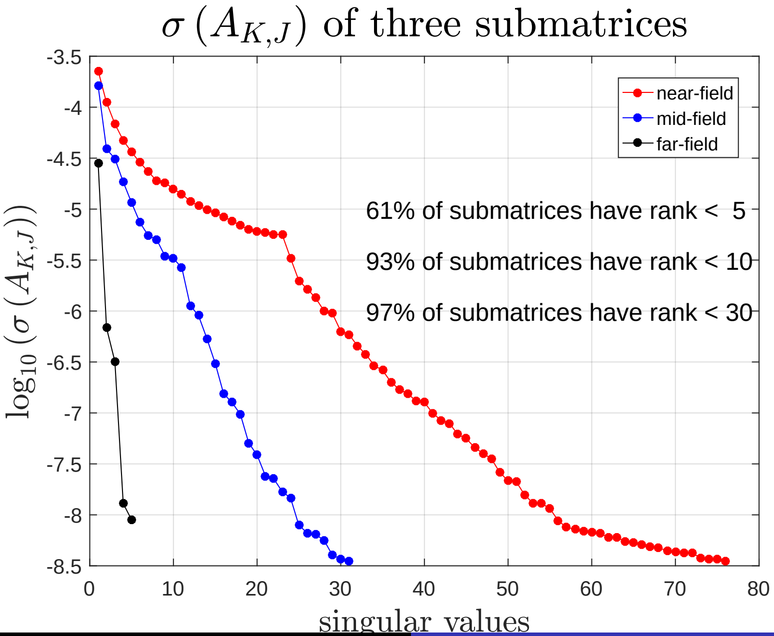
Reduced storage requirements

Matrix from underwater acoustic fluid structure interaction

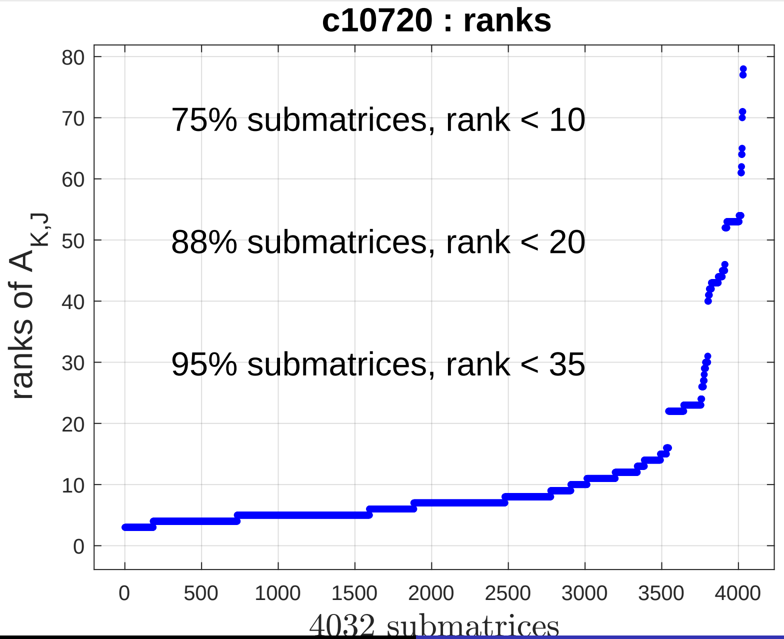
- 3456×3456 dense
- 16 subdomains
- Blue matrices are dense
- Red matrices are low rank
- absolute tolerance = $1e^{-8}$



Reduced storage requirements : M3456



Reduced storage requirements : C10720



BLR functionality

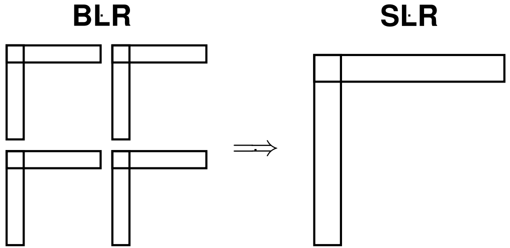
Store and operate with low rank submatrices, A is BLR

- $q = A\,p$ — matrix-vector multiply
- $Q = A\,P$ — matrix-matrix multiply, P and Q are BLR
- factor $A = L\,U$, $L\,L^T$, $L\,D\,U$, $L\,D\,L^T$,
- solve $A\,u = f$ — matrix-vector solve
- solve $A\,U = F$ — matrix-matrix solve, U and F are BLR

We know that BLR is not optimal w.r.t. storage,
since it is a special case of H-matrices.

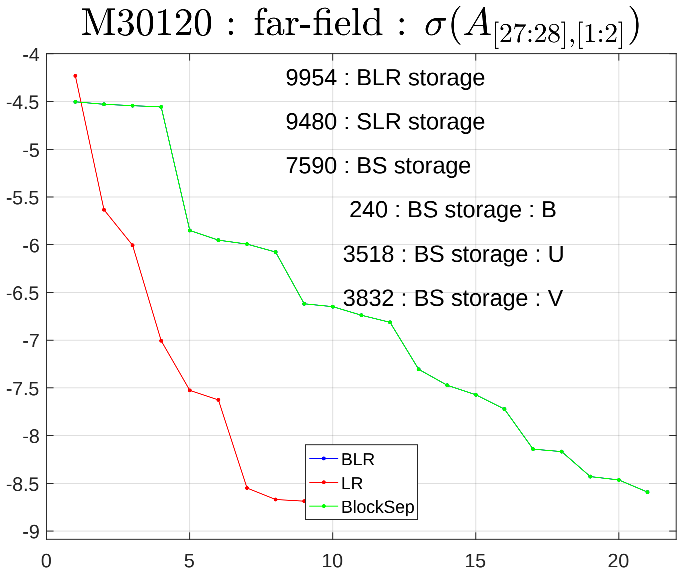
Example : BLR –> SLR (single low rank)

- Combine four submatrices into one submatrix

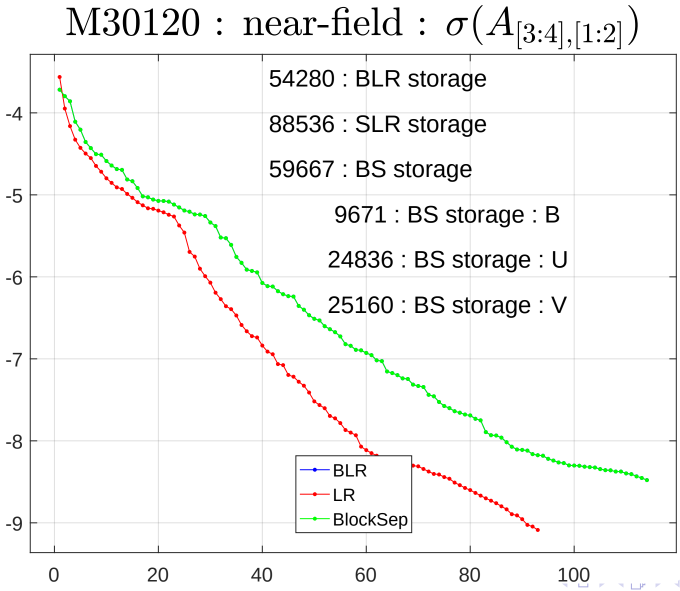


- Which approach uses less storage?
- $8rn$ versus $4sn$
- breakeven point when $s = 2r$

Far field submatrices – better to combine



Near field submatrices – better to leave separate



Building an H-matrix from a BLR matrix

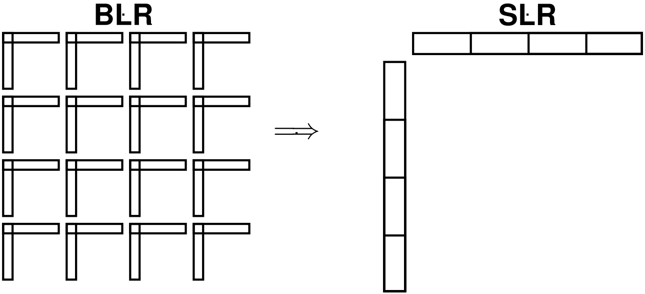
- From the BLR matrix, build a coarse graph
  - vertex weights proportional to domain weight
  - edge weights proportional to submatrix rank
- Build a domain merge tree, (Metis, Scotch, etc), leaves are domains, the root is the entire graph
- Climb the merge tree, combine when appropriate

$$\boxed{\begin{array}{cccc} A_{1;13} & A_{1;14} & A_{1;15} & A_{1;16} \\ A_{2;13} & A_{2;14} & A_{2;15} & A_{2;16} \\ A_{3;13} & A_{3;14} & A_{3;15} & A_{3;16} \\ A_{4;13} & A_{4;14} & A_{4;15} & A_{4;16} \end{array}} \Longrightarrow \boxed{\begin{array}{cc} A_{1:2;13:14} & A_{1:2;15:16} \\ & \\ A_{1:2;13:14} & A_{1:2;15:16} \end{array}} \Longrightarrow A_{1:4;13:16}$$

BLR to SLR and back to BLR

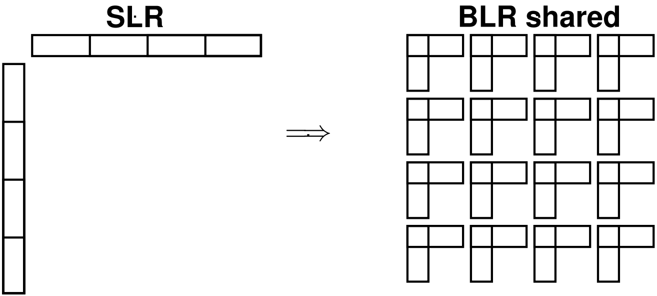
- Each submatrix has a low rank form,
  e.g., $A_{1,13} = X_{1,\alpha}\, Y^T_{13,\alpha}$
- The large submatrix $A_{1:4,13:16}$ has a low rank form
  $A_{1:4,13:16} = X_{1:4,\beta}\, Y^T_{13:16,\beta}$

Sixteen submatrices combine their XY^T data
into a new **single** low rank XY^T matrix.

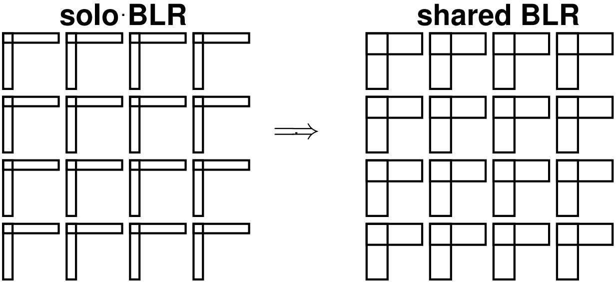


BLR to SLR and back to BLR

- Each submatrix has a **new** low rank form,
  e.g., $A_{1,13} = X_{1,\beta} \, Y_{13,\beta}^T$
  where $X_{1,\beta}$ and $Y_{13,\beta}^T$ come from the H-matrix.
- $X_{1,\beta}$ is larger than $X_{1,\alpha}$,
  but $X_{1,\beta}$ is shared among four submatrices.



Two flavors of BLR



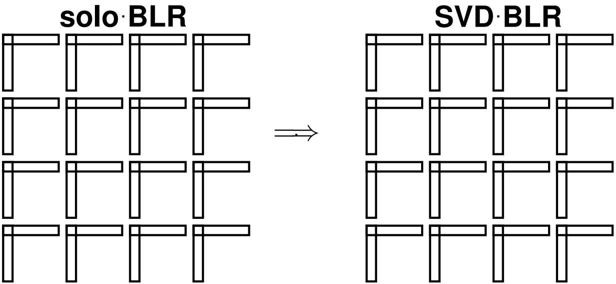
- **Solo** $A_{1,13} = X_{1,\alpha} Y_{13,\alpha}^T$, responsible for $X_{1,\alpha}$ and $Y_{13,\alpha}$.
- **Shared** $A_{1,13} = X_{1,\beta} Y_{13,\beta}^T$,
  responsible for $1/4$ cost of $X_{1,\beta}$ and $1/4$ cost of $Y_{13,\beta}$.

BLR vs H-matrix

- H-matrix always better than BLR. How much better? Enough better to abandon the simpler BLR?
- **Shared** BLR can use the same storage as the best H-matrix ordering. (Minor mods to present code.)
- SLR compression **is** effective in the far field, and there are a lot of far field submatrices.
- SLR compression **is not** effective in the near field, and the near-field is where the entries are concentrated.
- How to distribute in a distributed environment?
- How to deal with load balance?

BLR to Block Separable

First step – compute the SVD of each submatrix.



$$
\begin{aligned}
\text{E.g.,}\quad A_{1,13} &= X_{1,\alpha}\, Y_{13,\alpha}^T + E_{1,13} \qquad (2)\\
&= \left(X_{1,\alpha}\, V_{\alpha,\alpha}\right)\, \Sigma_{\alpha,\alpha}\, U_{13,\alpha}^T + E_{1,13}\\
&= U_{1,\alpha}\, \Sigma_{\alpha,\alpha}\, U_{13,\alpha}^T + E_{1,13}\\
\text{where}\quad \|E_{1,13}\|_2 < \epsilon, &\quad U_{1,\alpha}^T U_{1,\alpha} = I_{\alpha,\alpha} = U_{13,\alpha}^T U_{13,\alpha}
\end{aligned}
$$

BLR to Block Separable

Focus on the first block row with numeric rank s.

$$A_{1,13:16} = \begin{bmatrix} A_{1,13} & A_{1,14} & A_{1,15} & A_{1,16} \end{bmatrix} \tag{3}$$
$$= X_{1,\beta} \begin{bmatrix} Y_{13,\beta}^T & Y_{14,\beta}^T & Y_{15,\beta}^T & Y_{16,\beta}^T \end{bmatrix} \tag{4}$$

This is an dense $n \times 4n$ matrix, $O(n^2 s)$ operations for RRQR. We can get equivalent results with

$$T_{1,13:16} = \begin{bmatrix} U_{1,\alpha}\Sigma_{\alpha,\alpha} & U_{1,\gamma}\Sigma_{\gamma,\gamma} & U_{1,\delta}\Sigma_{\delta,\delta} & U_{1,\epsilon}\Sigma_{\epsilon,\epsilon} \end{bmatrix} \tag{5}$$
$$= X_{1,\beta} \begin{bmatrix} Z_{13,\beta}^T & Z_{14,\beta}^T & Z_{15,\beta}^T & Z_{16,\beta}^T \end{bmatrix} \tag{6}$$

for $O(nrs)$ cost.

$X_{1,\beta^{(1)}}$ **is the shared column space of the first block row.**

$X_{2,\beta^{(2)}}$ **is the shared column space of the second block row.**

... **etc** ...

BLR to Block Separable

Focus on the first block column with numeric rank s.

$$\text{E.g.,}\quad A_{1:4,13} = \begin{bmatrix} A_{1,13} \\ A_{2,13} \\ A_{3,13} \\ A_{4,13} \end{bmatrix} = \begin{bmatrix} Z_{1,\nu} \\ Z_{2,\nu} \\ Z_{3,\nu} \\ Z_{4,\nu} \end{bmatrix} X_{13,\nu}^T \tag{7}$$

This is an dense $4n \times n$ matrix, $O(n^2 s)$ operations for RRLQ.
We can get equivalent results with an RRQR factorization of

$$\begin{aligned} T_{13,:} &= \left[\left(V_{13,\alpha}\Sigma_{\alpha,\alpha}\right) \quad \left(V_{13,\gamma}\Sigma_{\gamma,\gamma}\right) \quad \left(V_{13,\delta}\Sigma_{\delta,\delta}\right) \quad \left(V_{13,\epsilon}\Sigma_{\epsilon,\epsilon}\right) \right] \\ &= X_{13,\nu^{(13)}} \left[Y_{1,\nu^{(13)}}^T \quad Y_{2,\nu^{(13)}}^T \quad Y_{3,\nu^{(13)}}^T \quad Y_{4,\nu^{(13)}}^T \right] \end{aligned} \tag{8}$$

for $O(nrs)$ cost.

$X_{13,\nu^{(13)}}$ **— shared row space of the first block column.**

$X_{14,\nu^{(14)}}$ **— shared row space of the second block column.**

. . . **etc** . . .

BLR to Block Separable

The end result is to factor the 4×4 block matrix into the product of three matrices.

$$A_{1:4,13:16} = X_{1:4,\beta} \, B_{\beta,\nu} \, X_{13:16,\nu}^T \tag{9}$$

The two outer matrices $X_{1:4,\beta}$ and $X_{13:16,\nu}$ are **block diagonal**, and each submatrix has **orthonormal columns**.

$$X_{1:4,\beta} = \begin{bmatrix} X_{1,\beta^{(1)}} & & & \\ & X_{2,\beta^{(2)}} & & \\ & & X_{3,\beta^{(3)}} & \\ & & & X_{4,\beta^{(4)}} \end{bmatrix} \tag{10}$$

$$X_{13:16,\nu} = \begin{bmatrix} X_{13,\nu^{(13)}} & & & \\ & X_{14,\nu^{(14)}} & & \\ & & X_{15,\nu^{(15)}} & \\ & & & X_{16,\nu^{(16)}} \end{bmatrix} \tag{11}$$

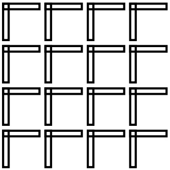
BLR to Block Separable

All "weight" of the matrix concentrated in central matrix $B_{\beta,\nu}$,

$$B_{\beta(1:4),\nu(13:16)} = \begin{bmatrix} B_{\beta(1),\nu(13)} & B_{\beta(1),\nu(14)} & B_{\beta(1),\nu(15)} & B_{\beta(1),\nu(16)} \\ B_{\beta(2),\nu(13)} & B_{\beta(2),\nu(14)} & B_{\beta(2),\nu(15)} & B_{\beta(2),\nu(16)} \\ B_{\beta(3),\nu(13)} & B_{\beta(3),\nu(14)} & B_{\beta(3),\nu(15)} & B_{\beta(3),\nu(16)} \\ B_{\beta(4),\nu(13)} & B_{\beta(4),\nu(14)} & B_{\beta(4),\nu(15)} & B_{\beta(4),\nu(16)} \end{bmatrix} \tag{12}$$

which is also BLR.

$B_{\beta(1:4),\nu(13:16)} =$



How much smaller is $B_{\beta(1:4),\nu(13:16)}$ than the original $A_{1:4,13:16}$?

BLR vs SLR vs Block Separable

Three ways to store 2×2, 4×4 and 8×8 block submatrices.

- **BLR** – block low rank, $X_i Y_i^T$
- **SLR** – single low rank, XY^T
- **BS** – block separable, $X_i B_{i,j} X_j^T$

	storage			
2×2 submatrix	BLR	SLR	BS	field
$[3:4] \times [1:2]$	50112	63072	**45579**	near
$[5:6] \times [1:2]$	49680	62208	**45757**	near
$[9:10] \times [1:2]$	44064	60480	**43344**	near
$[7:8] \times [1:2]$	18144	18144	**14848**	mid
$[11:12] \times [1:2]$	18576	18144	**14353**	mid
$[13:14] \times [1:2]$	19008	19008	**14848**	mid
$[15:16] \times [1:2]$	11664	**6912**	8100	far

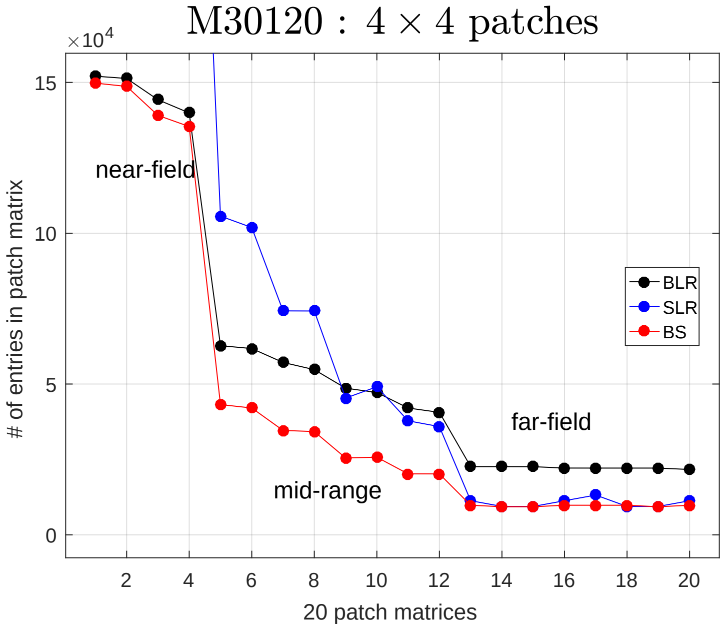
BLR vs SLR vs Block Separable

Three ways to store submatrices.

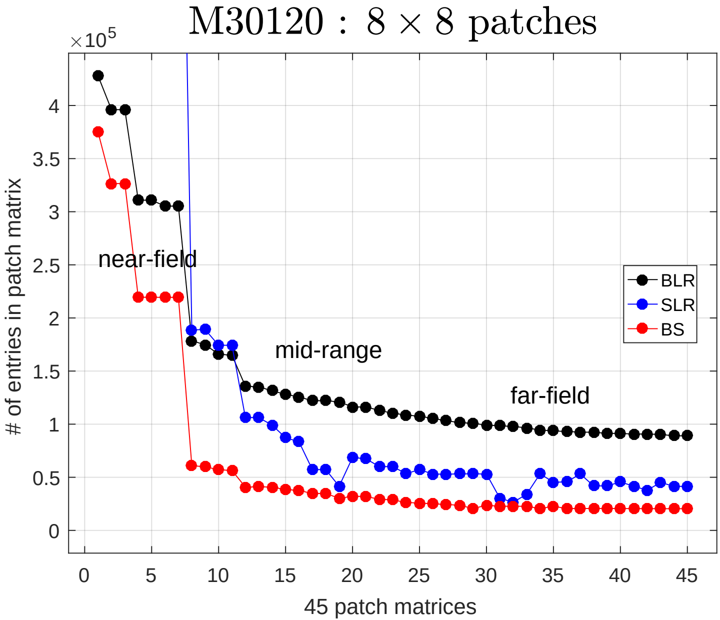
- **BLR** – block low rank, $X_i Y_i^T$
- **SLR** – single low rank, XY^T
- **BS** – block separable, $X_i B_{i,j} X_j^T$

	storage			
4×4 submatrix	BLR	SLR	BS	field
$[5:8] \times [1:4]$	135648	247104	**111288**	near
$[9:12] \times [1:4]$	125712	240192	**107592**	near
$[13:16] \times [1:4]$	61344	69120	**35660**	mid
8×8 submatrix	BLR	SLR	BS	field
$[9:16] \times [1:8]$	374112	974592	**272308**	near

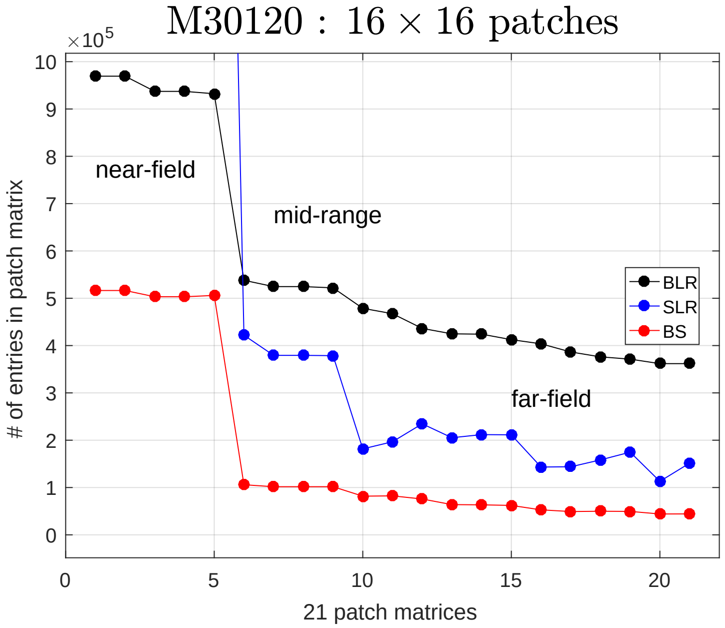
30120 dof and 128 domains



30120 dof and 128 domains



30120 dof and 128 domains



BLR, SLR or BS? or all three?

- BLR — simplicity, good implementations
  - simple factor, solve, multiply
  - matrix-matrix multiply now low rank, not dense
  - simple computations, task DAG easy to construct
  - during factorization, $L_{\mathcal{K},\mathcal{I}}$ and $(D_{\mathcal{I},\mathcal{I}} U_{\mathcal{I},\mathcal{J}})$ are shared among processors
- H-matrices — SLR in each patch, many codes
- H-matrices — BS in each patch, (new, sort of)
- BLR-shared — use the best H-matrix decomposition, (new) point to submatrices, not owned, either SLR or BS

HODLR-SLR or HODLR-BS ? or combination?

- HODLR, largest sized patches, fewest # of patches
- HODLR opposite BLR, other end of spectrum
- use SLR on each patch
  - large scale operations on entire patch
  - $q = Ap = XY^T p$ straightforward in MPP
- use BS on each patch
  - medium scale operations on block row or column
  - $q = Ap = XBY^T p$ less straightforward in MPP, the task DAG changes.
  - BS adds complexity to computation, reduces storage, reduces computations, orthonormal on the outside
- BS on near- and mid-field, SLR on far field ?

Summary

- Discard notion of $\{0, 1\}$ sparsity
  "Concentrate" matrix entries into the singular values, then drop small singular values
- Consider "patches" of low rank submatrices
  - H-SLR, known as H-matrices
  - H-BS, (new, L'Eplattenier's multicenter)
  - Both, cooperation $\Longrightarrow$ reduced storage, operations
- BS on all of $H_{\mathcal{M},\mathcal{M}}$ is **not** very effective
- BS on separate patches of $H_{\mathcal{M},\mathcal{M}}$ **is** very effective
- BS does not gain much from recursion on our patches, two levels is adequate for our present matrix sizes