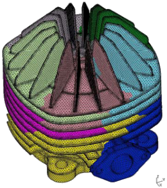
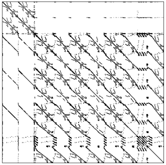
Block Low-Rank Matrices:
Main Results and Recent Advances

Theo Mary
University of Manchester, School of Mathematics

Rutherford Appleton Laboratory, 15 November 2018

M\cr NA
Manchester Numerical Analysis

Context

 $\rightarrow$  $\rightarrow$ 

Linear system $Ax = b$

Often a keystone in scientific computing applications
(discretization of PDEs, step of an optimization method, ...)

Large, sparse matrices

Matrix A is sparse (many zeros) but also large (10^6–10^9 unknowns)

Direct methods

Factorize $A = LU$ and solve $LUx = b$

☺ Numerically reliable ☹ Computational cost

Challenges and opportunities

Asymptotic Complexity

Direct methods require $O(n^2)$ space and $O(n^3)$ work: unfeasible for large n $\Rightarrow$ exploit structural sparsity and data sparsity to achieve $O(n)$ complexity

Performance and Scalability

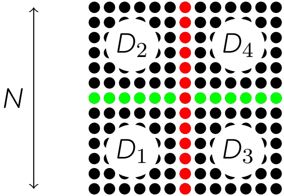
Increasingly faster computers available, need to efficiently make use of them to solve larger and larger problems

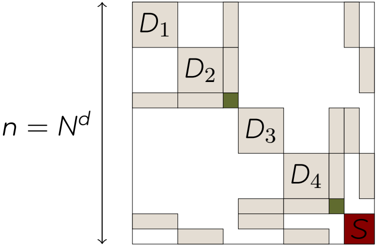
Accuracy and Stability

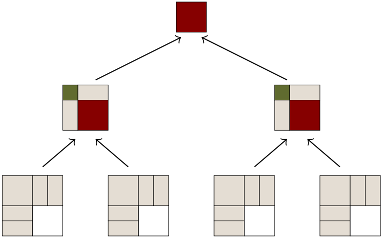
Computations are performed in floating-point arithmetic; increasingly low precisions (e.g. fp16) available

Numerical pivoting needed for stability; important to derive meaningful error bounds for large and/or data sparse problems

Structural sparsity





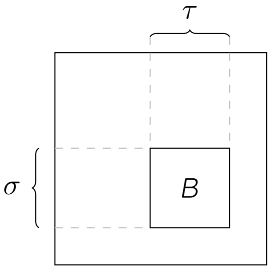


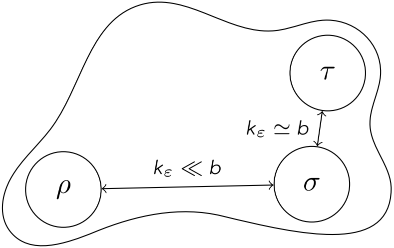
3D problem complexity

- Flops: $O(n^2)$
- Storage: $O(n^{4/3})$

Data sparsity

In many cases of interest the matrix has a block low-rank structure





A block B represents the interaction between two subdomains.
Far away subdomains $\Rightarrow$ block of low numerical rank:

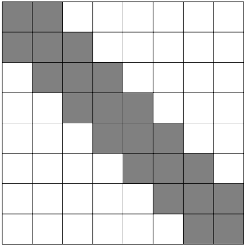
$$\begin{array}{cccc} B & \approx & X & Y^T \\ b \times b & & b \times k_\varepsilon & k_\varepsilon \times b \end{array}$$

with $k_\varepsilon \ll b$ such that $\|B - XY^T\| \leq \varepsilon$

How to choose a good block partitioning of the matrix?

Flat vs hierarchical matrices

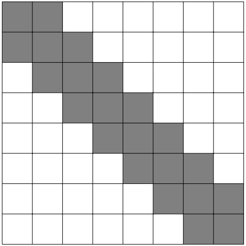
How to choose a good block partitioning of the matrix?



BLR matrix

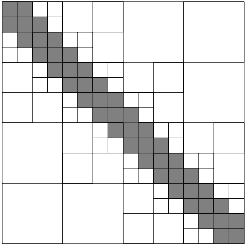
Flat vs hierarchical matrices

How to choose a good block partitioning of the matrix?



BLR matrix

- Superlinear complexity
- Simple, flat structure

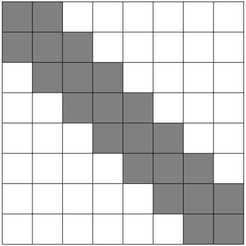


$\mathcal{H}$-matrix

- Nearly linear complexity
- Complex, hierarchical structure

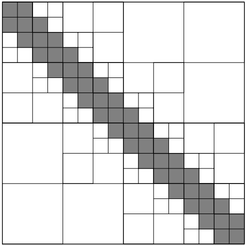
Flat vs hierarchical matrices

How to choose a good block partitioning of the matrix?



BLR matrix

- Superlinear complexity
- Simple, flat structure

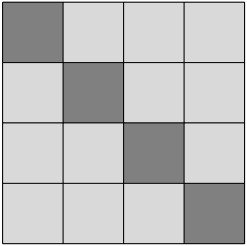


$\mathcal{H}$-matrix

- Nearly linear complexity
- Complex, hierarchical structure

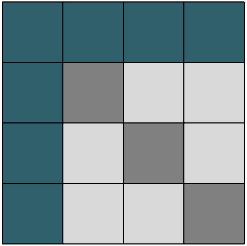
BLR is a comprise between complexity and performance

BLR factorization: standard FCU variant



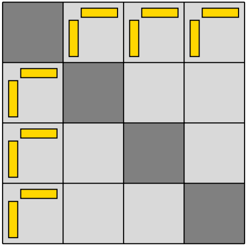
- FCU

BLR factorization: standard FCU variant



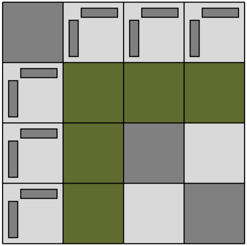
- FCU (Factor,
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



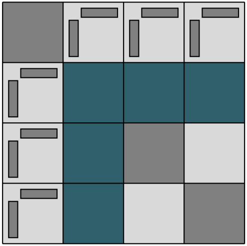
- FCU (Factor, Compress,
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



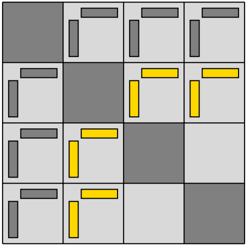
- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



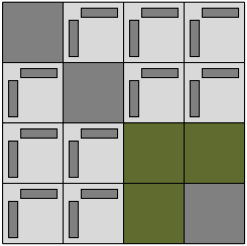
- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



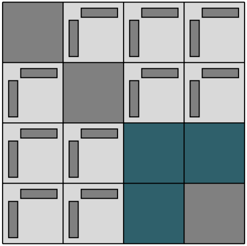
- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



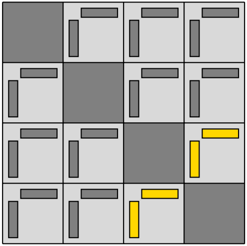
- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



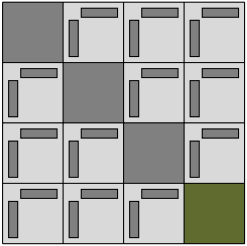
- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



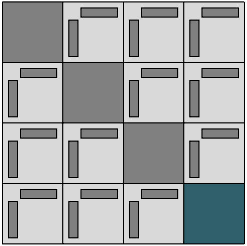
- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



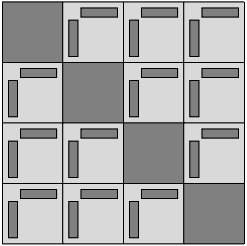
- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant



- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers

BLR factorization: standard FCU variant

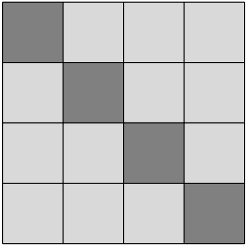


- FCU (Factor, Compress, Update)
- Easy to handle numerical pivoting, a critical feature often lacking in other low-rank solvers
- Potential of this variant was studied in

  P. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L'Excellent, and C. Weisbecker. *Improving Multifrontal Methods by Means of Block Low-Rank Representations*.

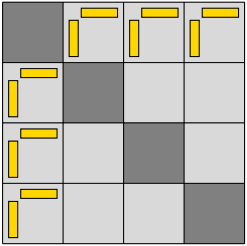
  SIAM J. Sci. Comput. (2015).

BLR factorization: CFU variant



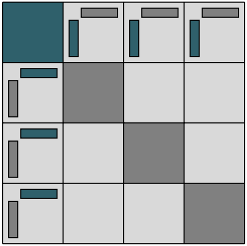
- CFU

BLR factorization: CFU variant



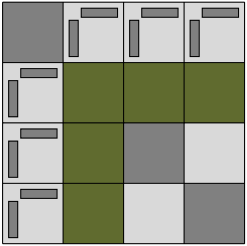
- CFU (Compress,

BLR factorization: CFU variant



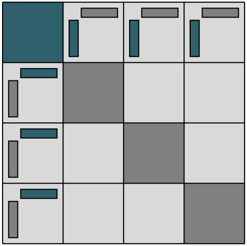
- CFU (Compress, Factor,
- Factor step is performed on compressed blocks ⇒ reduced flops

BLR factorization: CFU variant



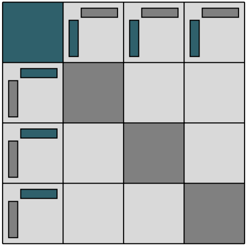
- CFU (Compress, Factor, Update)
- Factor step is performed on compressed blocks ⇒ reduced flops

BLR factorization: CFU variant



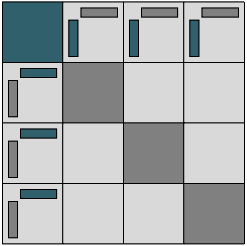
- CFU (Compress, Factor, Update)
- Factor step is performed on compressed blocks ⇒ reduced flops
- How can we handle numerical pivoting?

BLR factorization: CFU variant



- CFU (Compress, Factor, Update)
- Factor step is performed on compressed blocks ⇒ reduced flops
- How can we handle numerical pivoting?
  - Restricting pivot choice to diagonal block is acceptable (in combination with a pivot delaying strategy)

BLR factorization: CFU variant



- CFU (Compress, Factor, Update)
- Factor step is performed on compressed blocks ⇒ reduced flops
- How can we handle numerical pivoting?
  - Restricting pivot choice to diagonal block is acceptable (in combination with a pivot delaying strategy)
  - Must still check entries in off-diagonal blocks: can be estimated from entries in low-rank blocks

Outline

I **Algorithms and Complexity** (joint work with P. Amestoy, A. Buttari, JY. L'Excellent)

- Asymptotic complexity analysis
- Multilevel BLR format

II **Performance and Scalability** (joint work with PA, AB, JYL)

- Multicore performance
- Distributed-memory scalability

III **Accuracy and Stability** (joint work with N. Higham)

- BLR error analysis
- Use as a low-accuracy preconditioner

Algorithms and Complexity

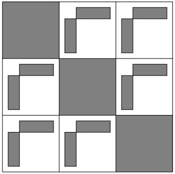
Section 1

Asymptotic complexity analysis

P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *On the Complexity of the Block Low-Rank Multifrontal Factorization*. SIAM J. Sci. Comput. (2017).

Computing the BLR complexity

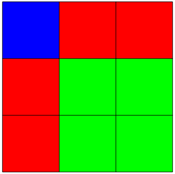
Assume all off-diagonal blocks are low-rank. Then:



$$\begin{aligned} Storage &= cost_{LR} * nb_{LR} + cost_{FR} * nb_{FR} \\ &= O(br) * O((\frac{m}{b})^2) + O(b^2) * O(\frac{m}{b}) \\ &= O(m^2 r/b + mb) \\ &= \mathbf{O(m^{3/2} r^{1/2})} \text{ for } b = (mr)^{1/2} \end{aligned}$$

Computing the BLR complexity

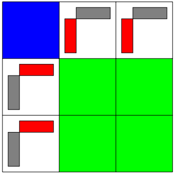
Assume all off-diagonal blocks are low-rank. Then:



$$\begin{aligned} Storage &= cost_{LR} * nb_{LR} + cost_{FR} * nb_{FR} \\ &= O(br) * O((\frac{m}{b})^2) + O(b^2) * O(\frac{m}{b}) \\ &= O(m^2 r/b + mb) \\ &= \mathbf{O(m^{3/2} r^{1/2})} \text{ for } b = (mr)^{1/2} \end{aligned}$$

$$\begin{aligned} FlopLU &= cost_{getrf} * nb_{getrf} + cost_{trsm} * nb_{trsm} + cost_{gemm} * nb_{gemm} \\ &= O(b^3) * O(\frac{m}{b}) + O(b^3) * O((\frac{m}{b})^2) + O(br^2) * O((\frac{m}{b})^3) \\ &= O(mb^2 + m^2 b + m^3 r^2 / b^2) \\ &= \mathbf{O(m^{7/3} r^{2/3})} \text{ for } b = (mr^2)^{1/3} \end{aligned}$$

Assume all off-diagonal blocks are low-rank. Then:



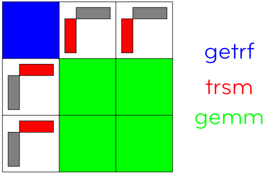
getrf
trsm
gemm

$$\begin{aligned}
Storage &= cost_{LR} * nb_{LR} + cost_{FR} * nb_{FR} \\
&= O(br) * O((\frac{m}{b})^2) + O(b^2) * O(\frac{m}{b}) \\
&= O(m^2 r/b + mb) \\
&= \mathbf{O(m^{3/2} r^{1/2})} \text{ for } b = (mr)^{1/2}
\end{aligned}$$

$$\begin{aligned}
FlopLU &= cost_{getrf} * nb_{getrf} + cost_{trsm} * nb_{trsm} + cost_{gemm} * nb_{gemm} \\
&= O(b^3) * O(\frac{m}{b}) + O(\cancel{b^3} b^2 r) * O((\frac{m}{b})^2) + O(br^2) * O((\frac{m}{b})^3) \\
&= O(mb^2 + m^2 \cancel{b} r + m^3 r^2 / b^2) \\
&= \mathbf{O}(\cancel{m^{7/3} r^{2/3}} \mathbf{m^2 r}) \text{ for } b = \cancel{(mr^2)^{1/3}} (mr)^{1/2}
\end{aligned}$$

CFU variant improves asymptotic complexity!

Assume all off-diagonal blocks are low-rank. Then:



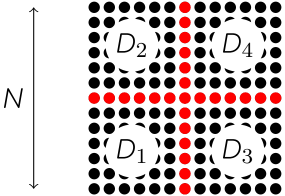
$$\begin{aligned} Storage &= cost_{LR} * nb_{LR} + cost_{FR} * nb_{FR} \\ &= O(br) * O((\frac{m}{b})^2) + O(b^2) * O(\frac{m}{b}) \\ &= O(m^2 r/b + mb) \\ &= \mathbf{O(m^{3/2} r^{1/2})} \text{ for } b = (mr)^{1/2} \end{aligned}$$

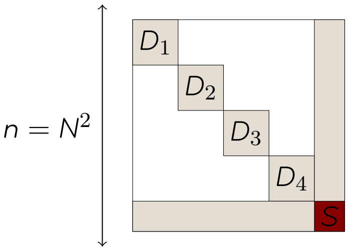
$$\begin{aligned} FlopLU &= cost_{getrf} * nb_{getrf} + cost_{trsm} * nb_{trsm} + cost_{gemm} * nb_{gemm} \\ &= O(b^3) * O(\frac{m}{b}) + O(\cancel{b^3} b^2 r) * O((\frac{m}{b})^2) + O(br^2) * O((\frac{m}{b})^3) \\ &= O(mb^2 + m^2 \cancel{b} r + m^3 r^2 / b^2) \\ &= \mathbf{O}(\cancel{m^{7/3} r^{2/3}} \mathbf{m^2 r}) \text{ for } b = \cancel{(mr^2)^{1/3}} (mr)^{1/2} \end{aligned}$$

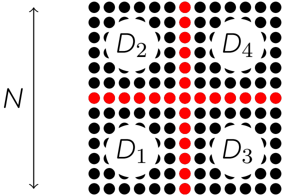
CFU variant improves asymptotic complexity!

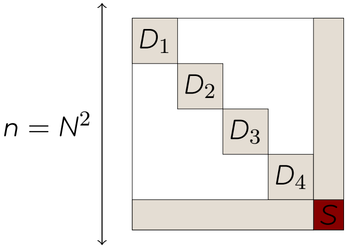
Result holds if a **constant** number of off-diag. blocks is full-rank.

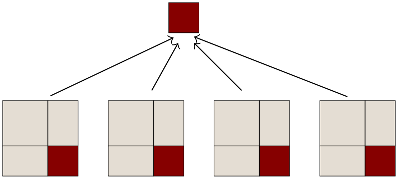
From dense to sparse: nested dissection











Proceed recursively to compute separator tree

Factorizing a sparse matrix amounts to factorizing a sequence of dense matrices

$\Rightarrow$

sparse complexity is directly derived from dense one

Nested dissection complexity formulas

In the **2D** case:

$$\mathcal{C}_{sparse} = \sum_{\ell=0}^{\log N} 4^{\ell} \mathcal{C}_{dense}\left(\frac{N}{2^{\ell}}\right)$$

Nested dissection complexity formulas

In the **2D** case:

$$\mathcal{C}_{sparse} = \sum_{\ell=0}^{\log N} 4^{\ell} \mathcal{C}_{dense}\left(\frac{N}{2^{\ell}}\right) = N^{\alpha} \sum_{\ell=0}^{\log N} 2^{(2-\alpha)\ell}$$

If $\mathcal{C}_{dense} = O(m^{\alpha})$, $\mathcal{C}_{sparse}$ is a geom. series of common ratio $2^{2-\alpha}$:

$$\mathcal{C}_{sparse} = \begin{cases} O(n^{\alpha/2}) & \text{if } \alpha > 2 \\ O(n \log n) & \text{if } \alpha = 2 \\ O(n) & \text{if } \alpha < 2 \end{cases}$$

Nested dissection complexity formulas

In the **2D** case:

$$\mathcal{C}_{sparse} = \sum_{\ell=0}^{\log N} 4^{\ell}\mathcal{C}_{dense}(\frac{N}{2^{\ell}}) = N^{\alpha} \sum_{\ell=0}^{\log N} 2^{(2-\alpha)\ell}$$

If $\mathcal{C}_{dense} = O(m^{\alpha})$, $\mathcal{C}_{sparse}$ is a geom. series of common ratio $2^{2-\alpha}$:

$$\mathcal{C}_{sparse} = \begin{cases} O(n^{\alpha/2}) & \text{if } \alpha > 2 \\ O(n \log n) & \text{if } \alpha = 2 \\ O(n) & \text{if } \alpha < 2 \end{cases}$$

Similar formulas in the **3D** case:

$$\mathcal{C}_{sparse} = \sum_{\ell=0}^{\log N} 8^{\ell}\mathcal{C}_{dense}(\frac{N^2}{4^{\ell}}) = N^{2\alpha} \sum_{\ell=0}^{\log N} 2^{(3-2\alpha)\ell}$$

$$\mathcal{C}_{sparse} = \begin{cases} O(n^{2\alpha/3}) & \text{if } \alpha > 1.5 \\ O(n \log n) & \text{if } \alpha = 1.5 \\ O(n) & \text{if } \alpha < 1.5 \end{cases}$$

Complexity of the BLR factorization

		storage	flops
dense	FR	$O(m^2)$	$O(m^3)$
	BLR	$O(m^{3/2})$	$O(m^2)$
sparse 2D	FR	$O(n \log n)$	$O(n^{3/2})$
	BLR	$O(n)$	$O(n \log n)$

(assuming $r = O(1)$)

- In a **2D** world hierarchical matrices would not be needed

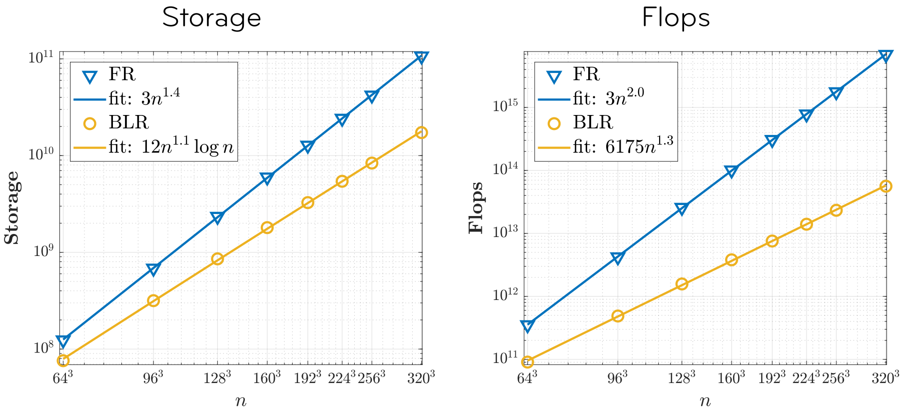
Complexity of the BLR factorization

		storage	flops
dense	FR	$O(m^2)$	$O(m^3)$
	BLR	$O(m^{3/2})$	$O(m^2)$
sparse 2D	FR	$O(n \log n)$	$O(n^{3/2})$
	BLR	$O(n)$	$O(n \log n)$
sparse 3D	FR	$O(n^{4/3})$	$O(n^2)$
	BLR	$O(n \log n)$	$O(n^{4/3})$

(assuming $r = O(1)$)

- In a **2D** world hierarchical matrices would not be needed
- Superlinear complexities in **3D**

Experimental complexity fit: 3D Poisson ($\varepsilon = 10^{-10}$)



- Good agreement with theoretical complexity:
  - Storage: $O(n \log n) \to O(n^{1.1} \log n)$
  - Flops: $O(n^{4/3}) \to O(n^{1.3})$

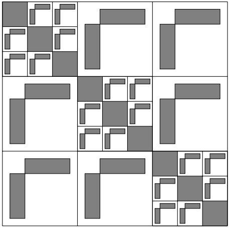
Section 2

The multilevel BLR format

P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *Bridging the gap between flat and hierarchical low-rank matrix formats: the multilevel BLR format*. Submitted (2018).

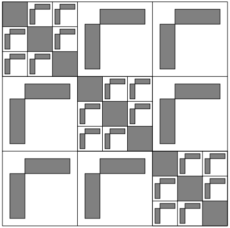
Complexity of the two-level BLR format

Assume all off-diagonal blocks are low-rank. Then:



$$\begin{aligned}
Storage &= cost_{LR} * nb_{LR} + cost_{BLR} * nb_{BLR} \\
&= O(br) * O((\frac{m}{b})^2) + O(b^{3/2}r^{1/2}) * O(\frac{m}{b}) \\
&= O(m^2 r/b + m(br)^{1/2}) \\
&= \mathbf{O(m^{4/3}r^{2/3})} \text{ for } b = (m^2 r)^{1/3}
\end{aligned}$$

Assume all off-diagonal blocks are low-rank. Then:



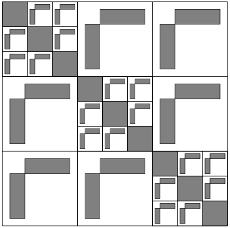
$$\begin{aligned} Storage &= cost_{LR} * nb_{LR} + cost_{BLR} * nb_{BLR} \\ &= O(br) * O((\frac{m}{b})^2) + O(b^{3/2} r^{1/2}) * O(\frac{m}{b}) \\ &= O(m^2 r/b + m(br)^{1/2}) \\ &= \mathbf{O(m^{4/3} r^{2/3})} \text{ for } b = (m^2 r)^{1/3} \end{aligned}$$

Similarly, we can prove:

$$FlopLU = \mathbf{O(m^{5/3} r^{4/3})} \text{ for } b = (m^2 r)^{1/3}$$

Result holds if a **constant** number of off-diag. blocks is BLR.

Assume all off-diagonal blocks are low-rank. Then:



$$Storage = cost_{LR} * nb_{LR} + cost_{BLR} * nb_{BLR}$$
$$= O(br) * O((\frac{m}{b})^2) + O(b^{3/2}r^{1/2}) * O(\frac{m}{b})$$
$$= O(m^2 r/b + m(br)^{1/2})$$
$$= \mathbf{O(m^{4/3}r^{2/3})} \text{ for } b = (m^2 r)^{1/3}$$

Similarly, we can prove:

$$FlopLU = \mathbf{O(m^{5/3}r^{4/3})} \text{ for } b = (m^2 r)^{1/3}$$

Result holds if a **constant** number of off-diag. blocks is BLR.

		FR	BLR	2-BLR	...	$\mathcal{H}$
storage	dense	$O(m^2)$	$O(m^{1.5})$	$O(m^{1.33})$	...	$O(m \log m)$
	sparse	$O(n^{1.33})$	$O(n \log n)$	$O(n)$	...	$O(n)$
flop LU	dense	$O(m^3)$	$O(m^2)$	$O(m^{1.66})$	...	$O(m \log^3 m)$
	sparse	$O(n^2)$	$O(n^{1.33})$	$O(n^{1.11})$	...	$O(n)$

Multilevel BLR complexity

Main result

For $b = m^{\ell/(\ell+1)} r^{1/(\ell+1)}$, the $\ell-$level complexities are:

$$Storage = \mathbf{O}(\mathbf{m}^{(\ell+2)/(\ell+1)} \mathbf{r}^{\ell/(\ell+1)})$$

$$FlopLU = \mathbf{O}(\mathbf{m}^{(\ell+3)/(\ell+1)} \mathbf{r}^{2\ell/(\ell+1)})$$

Multilevel BLR complexity

Main result

For $b = m^{\ell/(\ell+1)} r^{1/(\ell+1)}$, the $\ell-$level complexities are:

$$Storage = \mathbf{O}(\mathbf{m}^{(\ell+2)/(\ell+1)} \mathbf{r}^{\ell/(\ell+1)})$$

$$FlopLU = \mathbf{O}(\mathbf{m}^{(\ell+3)/(\ell+1)} \mathbf{r}^{2\ell/(\ell+1)})$$

	$\ell = 1$	$\ell = 2$
Dense	$O(m^2)$	$O(m^{1.66})$
Sparse 3D	$O(n^{1.33})$	$O(n^{1.11})$

Multilevel BLR complexity

Main result

For $b = m^{\ell/(\ell+1)} r^{1/(\ell+1)}$, the ℓ−level complexities are:

$$Storage = \mathbf{O}(\mathbf{m}^{(\ell+2)/(\ell+1)} \mathbf{r}^{\ell/(\ell+1)})$$

$$FlopLU = \mathbf{O}(\mathbf{m}^{(\ell+3)/(\ell+1)} \mathbf{r}^{2\ell/(\ell+1)})$$

	$\ell = 1$	$\ell = 2$	$\ell = 3$
Dense	$O(m^2)$	$O(m^{1.66})$	$O(m^{1.5})$
Sparse 3D	$O(n^{1.33})$	$O(n^{1.11})$	$O(n \log n)$

Main result

For $b = m^{\ell/(\ell+1)} r^{1/(\ell+1)}$, the ℓ−level complexities are:

$$Storage = \mathbf{O(m^{(\ell+2)/(\ell+1)} r^{\ell/(\ell+1)})}$$

$$FlopLU = \mathbf{O(m^{(\ell+3)/(\ell+1)} r^{2\ell/(\ell+1)})}$$

	$\ell = 1$	$\ell = 2$	$\ell = 3$	$\ell = 4$
Dense	$O(m^2)$	$O(m^{1.66})$	$O(m^{1.5})$	$O(m^{1.4})$
Sparse 3D	$O(n^{1.33})$	$O(n^{1.11})$	$O(n \log n)$	$O(n)$

Multilevel BLR complexity

Main result

For $b = m^{\ell/(\ell+1)} r^{1/(\ell+1)}$, the $\ell-$level complexities are:

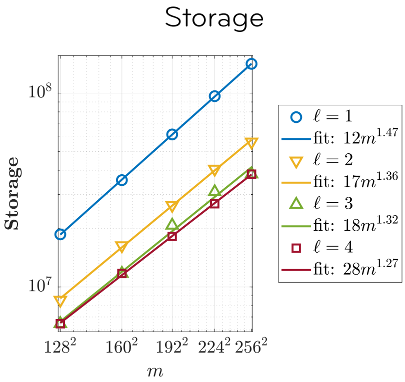
$$Storage = \mathbf{O}(\mathbf{m}^{(\ell+2)/(\ell+1)} \mathbf{r}^{\ell/(\ell+1)})$$

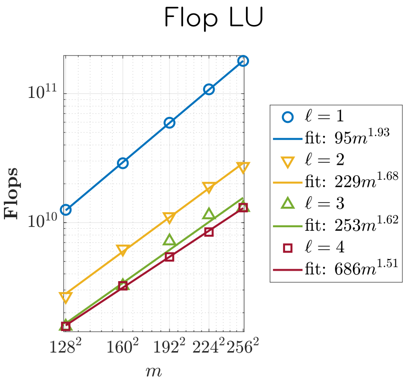
$$FlopLU = \mathbf{O}(\mathbf{m}^{(\ell+3)/(\ell+1)} \mathbf{r}^{2\ell/(\ell+1)})$$

	$\ell = 1$	$\ell = 2$	$\ell = 3$	$\ell = 4$
Dense	$O(m^2)$	$O(m^{1.66})$	$O(m^{1.5})$	$O(m^{1.4})$
Sparse 3D	$O(n^{1.33})$	$O(n^{1.11})$	$O(n \log n)$	$O(n)$

$\Rightarrow$ $\mathcal{H}$ **matrices typically use** $10+$ **levels... but only** 4 **are enough!**

Numerical experiments (3D Poisson)





- Experimental complexity in relatively good agreement with theoretical one
- Asymptotic gain decreases with levels

Performance and Scalability

Section 3

Multicore performance

P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures*. ACM Trans. Math. Soft. (2018).

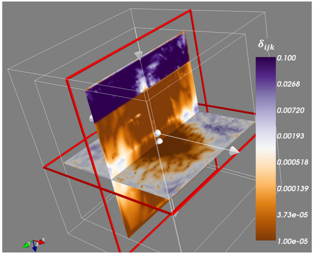
Matrix S3

Double complex (z) symmetric

Electromagnetics application (CSEM)

3.3 millions unknowns

Required accuracy: $\varepsilon = 10^{-7}$



D. Shantsev, P. Jaysaval, S. Kethulle de Ryhove, P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *Large-scale 3D EM modeling with a Block Low-Rank multifrontal direct solver*. Geophys. J. Int (2017).

	flops ($\times 10^{12}$)	time (1 core)	time (24 cores)
FR	78.0	7390	509
BLR	10.2	2242	309
ratio	7.7	3.3	1.7

7.7 gain in flops only translated to a **1.7** gain in time:

Can we do better?

BLR variants

Variant name	$\mathcal{C}$	flops ($\times 10^{12}$)	time
Full-Rank	n^2	78.0	509
FCU	$n^{1.67}r^{0.5}$	10.2	307

BLR variants

Tree parallelism improves performance by reducing the relative cost of the fronts at the bottom of the tree, which achieve poor compression

Variant name	$\mathcal{C}$	flops ($\times 10^{12}$)	time
Full-Rank	n^2	78.0	509
+Tree par.	=	=	418
FCU	$n^{1.67}r^{0.5}$	10.2	307
+Tree par.	=	=	221

BLR variants

Left-looking FCU improves performance thanks to a left-looking approach which reduces memory transfers

Variant name	$\mathcal{C}$	flops ($\times 10^{12}$)	time
Full-Rank	n^2	78.0	509
+Tree par.	=	=	418
FCU	$n^{1.67}r^{0.5}$	10.2	307
+Tree par.	=	=	221
+Left-looking	=	=	175

BLR variants

LUA improves performance because it accumulates multiple low-rank updates and applies them at once increasing the granularity of operations

Variant name	$\mathcal{C}$	flops ($\times 10^{12}$)	time
Full-Rank	n^2	78.0	509
+Tree par.	=	=	418
FCU	$n^{1.67}r^{0.5}$	10.2	307
+Tree par.	=	=	221
+Left-looking	=	=	175
+LUA	=	=	167

BLR variants

LUAR reduces complexity because recompresses accumulated updates on the fly

Variant name	$\mathcal{C}$	flops ($\times 10^{12}$)	time
Full-Rank	n^2	78.0	509
+Tree par.	=	=	418
FCU	$n^{1.67}r^{0.5}$	10.2	307
+Tree par.	=	=	221
+Left-looking	=	=	175
+LUA	=	=	167
+LUAR	$n^{1.55}r^{0.66}$	8.1	160

BLR variants

CFU reduces complexity because solve operations are also done in low-rank

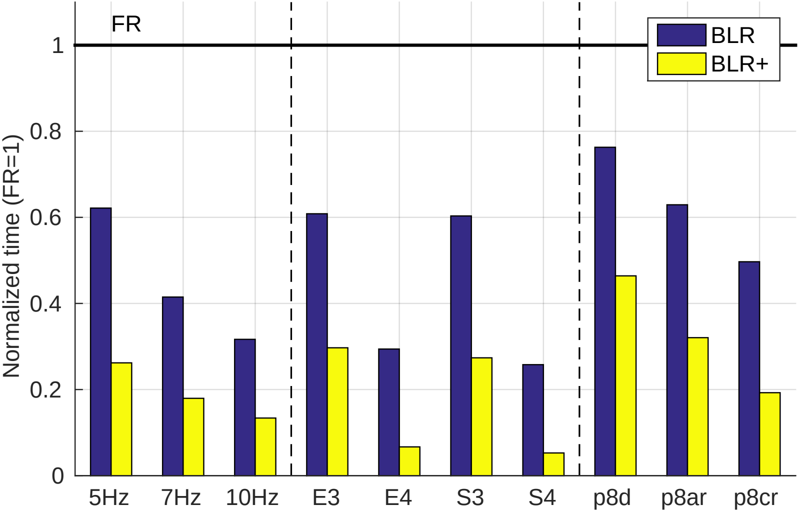
Variant name	$\mathcal{C}$	flops ($\times 10^{12}$)	time
Full-Rank	n^2	78.0	509
+Tree par.	=	=	418
FCU	$n^{1.67} r^{0.5}$	10.2	307
+Tree par.	=	=	221
+Left-looking	=	=	175
+LUA	=	=	167
+LUAR	$n^{1.55} r^{0.66}$	8.1	160
+CFU	$n^{1.33} r$	3.9	111

BLR variants

Variant name	$\mathcal{C}$	flops ($\times 10^{12}$)	time
Full-Rank	n^2	78.0	509
+Tree par.	=	=	418
FCU	$n^{1.67}r^{0.5}$	10.2	307
+Tree par.	=	=	221
+Left-looking	=	=	175
+LUA	=	=	167
+LUAR	$n^{1.55}r^{0.66}$	8.1	160
+CFU	$n^{1.33}r$	3.9	111

$\Rightarrow$**1.7** gain becomes **3.3**

Multicore performance results (24 threads)

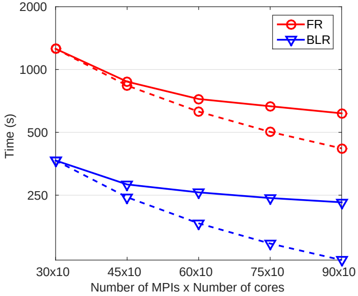


Section 4

Distributed-memory scalability

Distributed-memory scalability analysis

Results on $300 \rightarrow 900$ cores
(eos supercomputer, CALMIP)



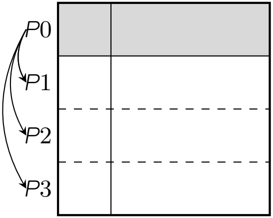
Matrix 10Hz
Single complex (c) unsymmetric
Seismic imaging application (FWI)
17 millions unknowns
Required accuracy: $\varepsilon = 10^{-3}$

P. Amestoy, R. Brossier, A. Buttari, J.-Y. L'Excellent, T. Mary, L. Métivier, A. Miniussi, and S. Operto. *Fast 3D frequency-domain full waveform inversion with a parallel Block Low-Rank multifrontal direct solver: application to OBC data from the North Sea*. Geophysics (2016).

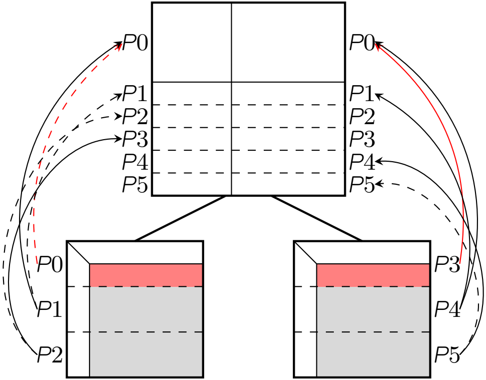
How to improve the scalability of the BLR factorization?
Two main difficulties:

- **Higher weight of communications:** flops reduced by **13** but volume of communications only by **2**
- **Unpredictability of compression:** more difficult to design good mapping and scheduling strategies

Type of messages

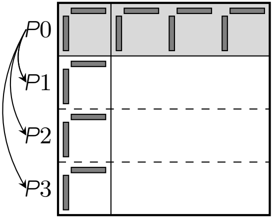


LU messages

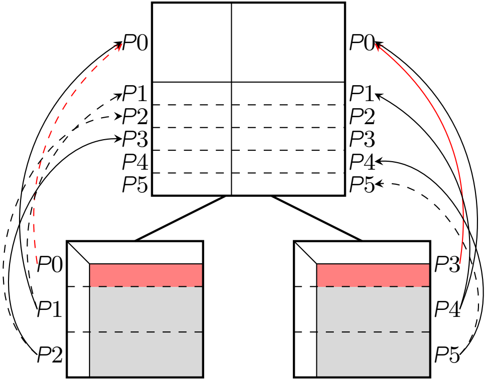


CB messages

Type of messages



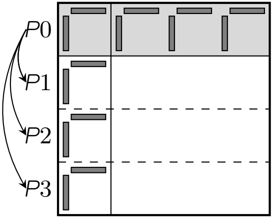
LU messages



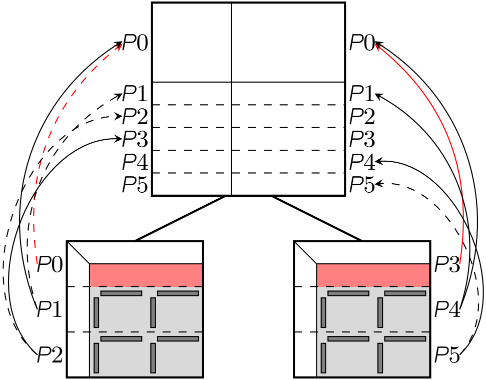
CB messages

- Volume of LU messages is reduced by compressing the factors
  - ☺ Reduces operation count, communications, and memory consumption

Type of messages



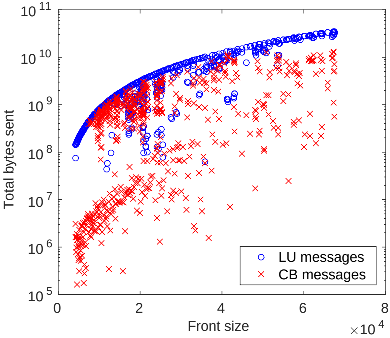
LU messages



CB messages

- Volume of LU messages is reduced by compressing the factors
  - ☺ Reduces operation count, communications, and memory consumption
- Volume of CB messages can be reduced by compressing the CB
  - ☺ Reduces communications and memory consumption
  - ☹ Increases operation count unless assembly is done in LR

Communication analysis

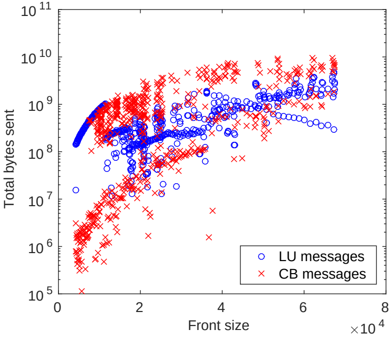


- FR case: *LU* messages dominate

Theoretical communication bounds

	$\mathcal{W}_{LU}$	$\mathcal{W}_{CB}$	$\mathcal{W}_{tot}$
FR	$\mathcal{O}\left(n^{4/3}p\right)$	$\mathcal{O}\left(n^{4/3}\right)$	$\mathcal{O}\left(n^{4/3}p\right)$

Communication analysis

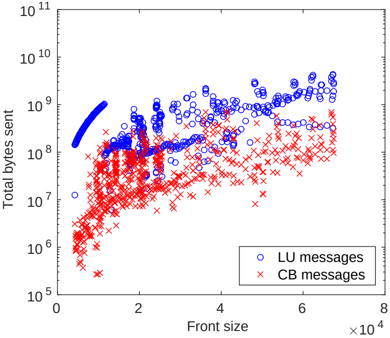


- FR case: *LU* messages dominate
- BLR case: CB messages dominate $\Rightarrow$ underwhelming reduction of communications

Theoretical communication bounds

	$\mathcal{W}_{LU}$	$\mathcal{W}_{CB}$	$\mathcal{W}_{tot}$
FR	$\mathcal{O}\left(n^{4/3}p\right)$	$\mathcal{O}\left(n^{4/3}\right)$	$\mathcal{O}\left(n^{4/3}p\right)$
BLR (CB_{FR})	$\mathcal{O}\left(nr^{1/2}p\right)$	$\mathcal{O}\left(n^{4/3}\right)$	$\mathcal{O}\left(nr^{1/2}p + n^{4/3}\right)$

Communication analysis



- FR case: *LU* messages dominate
- BLR case: CB messages dominate ⇒ underwhelming reduction of communications

⇒ CB compression allows for truly reducing the communications

Theoretical communication bounds

	$\mathcal{W}_{LU}$	$\mathcal{W}_{CB}$	$\mathcal{W}_{tot}$
FR	$\mathcal{O}\left(n^{4/3}p\right)$	$\mathcal{O}\left(n^{4/3}\right)$	$\mathcal{O}\left(n^{4/3}p\right)$
BLR (CB_{FR})	$\mathcal{O}\left(nr^{1/2}p\right)$	$\mathcal{O}\left(n^{4/3}\right)$	$\mathcal{O}\left(nr^{1/2}p + n^{4/3}\right)$
BLR (CB_{LR})	$\mathcal{O}\left(nr^{1/2}p\right)$	$\mathcal{O}\left(nr^{1/2}\right)$	$\mathcal{O}\left(nr^{1/2}p\right)$

Performance impact of the CB compression

matrix	10Hz	15Hz
order	17 M	58 M
factor flops (FR)	2.6 PF	29.6 PF
⇒ BLR (CB_{FR})	0.1 PF (5.3%)	1.0 PF (3.3%)
⇒ BLR (CB_{LR})	0.2 PF (6.1%)	1.1 PF (3.7%)
CB_{LR} flops impact	+15%	+12%
factor time (FR)	601	5,206
⇒ BLR (CB_{FR})	123 (4.9)	838 (6.2)
⇒ BLR (CB_{LR})	213 (2.8)	856 (6.1)
CB_{LR} time impact	+73%	+2%
comm. volume (FR)	5.3 TB	29.6 TB
comm. volume (CB_{FR})	1.7 TB (3.2)	13.3 TB (2.2)
comm. volume (CB_{LR})	0.6 TB (9.1)	1.2 TB (23.2)

⇒ CB compression becomes increasingly critical?

Performance impact of the CB compression

matrix	10Hz	15Hz
order	17 M	58 M
factor flops (FR)	2.6 PF	29.6 PF
⇒ BLR (CB_{FR})	0.1 PF (5.3%)	1.0 PF (3.3%)
⇒ BLR (CB_{LR})	0.2 PF (6.1%)	1.1 PF (3.7%)
CB_{LR} flops impact	+15%	+12%
factor time (FR)	601	5,206
⇒ BLR (CB_{FR})	123 (4.9)	838 (6.2)
⇒ BLR (CB_{LR})	213 (2.8)	856 (6.1)
CB_{LR} time impact	+73%	+2%
comm. volume (FR)	5.3 TB	29.6 TB
comm. volume (CB_{FR})	1.7 TB (3.2)	13.3 TB (2.2)
comm. volume (CB_{LR})	0.6 TB (9.1)	1.2 TB (23.2)

⇒ CB compression becomes increasingly critical?

Performance impact of the CB compression

matrix	10Hz	15Hz
order	17 M	58 M
factor flops (FR)	2.6 PF	29.6 PF
⇒ BLR (CB_{FR})	0.1 PF (5.3%)	1.0 PF (3.3%)
⇒ BLR (CB_{LR})	0.2 PF (6.1%)	1.1 PF (3.7%)
CB_{LR} flops impact	+15%	+12%
factor time (FR)	601	5,206
⇒ BLR (CB_{FR})	123 (4.9)	838 (6.2)
⇒ BLR (CB_{LR})	213 (2.8)	856 (6.1)
CB_{LR} time impact	+73%	+2%
comm. volume (FR)	5.3 TB	29.6 TB
comm. volume (CB_{FR})	1.7 TB (3.2)	13.3 TB (2.2)
comm. volume (CB_{LR})	0.6 TB (9.1)	1.2 TB (23.2)

⇒ CB compression becomes increasingly critical?

Results on very large problems

Results from

T. Mary. *Block Low-Rank multifrontal solvers: complexity, performance, and scalability*. PhD thesis (2017).

(Chapter 9, *Future challenges for large-scale BLR solvers*)

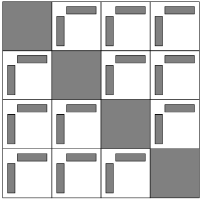
Matrix 20Hz ($N = 130$M, seismic imaging) on 2400 cores:

	flops	factors storage	memory peak	time
FR	150.0 PF	11.0 TB	151.0 GB	OOM
BLR	3.6 PF	1.8 TB	81.0 GB	2641s
ratio	42.0	6.0	1.9	

Accuracy and Stability

Section 5

Error analysis of BLR factorizations



Each off-diagonal block B is approximated by a low-rank matrix $\widetilde{B}$ such that $\|B - \widetilde{B}\| \le \varepsilon$

$\|A - L_\varepsilon U_\varepsilon\| \ne \varepsilon$ because of error propagation
$\Rightarrow$ **What is the overall accuracy $\|A - L_\varepsilon U_\varepsilon\|$?**

- Can we prove that $\|A - L_\varepsilon U_\varepsilon\| = O(\varepsilon)$?
- What is the error growth, i.e., how does the error depend on the matrix size m?
- How do the different variants (FCU, CFU, etc.) compare?
- Should we use an absolute threshold ($\|B - \widetilde{B}\| \le \varepsilon$) or a relative one ($\|B - \widetilde{B}\| \le \varepsilon \|B\|$)?

Some ingredients for the proof

The proof is based on *Stability of Block Algorithms with Fast Level-3 BLAS* (Demmel and Higham, 1992)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

Inductive proof: easy to bound error of computing $S = A_{22} - L_{21}U_{12}$ and error of $S = L_{22}U_{22}$ is obtained by induction

Some ingredients for the proof

The proof is based on *Stability of Block Algorithms with Fast Level-3 BLAS* (Demmel and Higham, 1992)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

Inductive proof: easy to bound error of computing $S = A_{22} - L_{21}U_{12}$ and error of $S = L_{22}U_{22}$ is obtained by induction

For BLR, several specific difficulties arise:

- Need to bound error of low-rank product kernel: $C = \widetilde{A}\widetilde{B} = X_A \left(Y_A^T X_B\right) Y_B^T$
- Choice of norm matters: to obtain best constants possible, we need a consistent, unitarily invariant norm
- Global bound is obtained from blockwise bounds $\Rightarrow$ we work with the Frobenius norm

Main result

Reminder

The full-rank LU factorization of $A \in \mathbb{R}^{n\times n}$ satisfies

$$\|A - LU\| \le nu\|L\|\|U\| + O(u^2)$$

Main result

The FCU BLR factorization of $A \in \mathbb{R}^{n\times n}$ with relative threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \le (nu + \varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

Main result

The FCU BLR factorization of $A \in \mathbb{R}^{n \times n}$ with relative threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \leq (nu + \varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

- $\|L\|\|U\| \leq n^2 \rho_n \|A\|$ where ρ_n is the growth factor
  $\Rightarrow$ with partial pivoting, the BLR factorization is stable!

Main result

The FCU BLR factorization of $A \in \mathbb{R}^{n\times n}$ with relative threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \leq (nu + \varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

- $\|L\|\|U\| \leq n^2 \rho_n \|A\|$ where ρ_n is the growth factor
  $\Rightarrow$ with partial pivoting, the BLR factorization is stable!
- Usually $\varepsilon \gg u$:

$\Rightarrow$ Role of u is limited

$\Rightarrow$ Very slow error growth

$\Rightarrow$ Usage of fast matrix arithmetic may be stable in BLR

Main result

The FCU BLR factorization of $A \in \mathbb{R}^{n \times n}$ with relative threshold ε satisfies

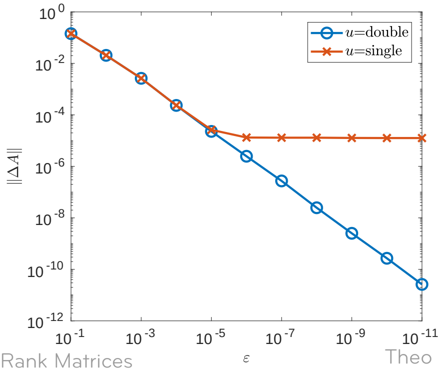
$$\|A - L_\varepsilon U_\varepsilon\| \leq (nu + \varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

- $\|L\|\|U\| \leq n^2 \rho_n \|A\|$ where ρ_n is the growth factor
  $\Rightarrow$ with partial pivoting, the BLR factorization is stable!
- Usually $\varepsilon \gg u$:

$\Rightarrow$ Role of u is limited

$\Rightarrow$ Very slow error growth

$\Rightarrow$ Usage of fast matrix arithmetic may be stable in BLR



Main result

The FCU BLR factorization of $A \in \mathbb{R}^{n \times n}$ with relative threshold ε satisfies

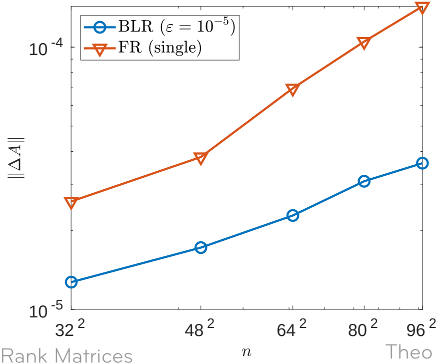
$$\|A - L_\varepsilon U_\varepsilon\| \le (nu + \varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

- $\|L\|\|U\| \le n^2 \rho_n \|A\|$ where ρ_n is the growth factor
  $\Rightarrow$ with partial pivoting, the BLR factorization is stable!
- Usually $\varepsilon \gg u$:

$\Rightarrow$ Role of u is limited

$\Rightarrow$ Very slow error growth

$\Rightarrow$ Usage of fast matrix arithmetic may be stable in BLR



Main result

The FCU BLR factorization of $A \in \mathbb{R}^{n\times n}$ with relative threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \le (nu + \varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

- $\|L\|\|U\| \le n^2 \rho_n \|A\|$ where ρ_n is the growth factor
  $\Rightarrow$ with partial pivoting, the BLR factorization is stable!
- Usually $\varepsilon \gg u$:

$\Rightarrow$ Role of u is limited

$\Rightarrow$ Very slow error growth

$\Rightarrow$ Usage of fast matrix arithmetic may be stable in BLR

For example with Strassen's algorithm, $nu \to n^{\log_2 12} u \approx n^{3.6} u$

Ongoing work with C.-P. Jeannerod, C. Pernet, and D. Roche: *Exploiting fast matrix arithmetic within BLR factorizations*:
$O(n^2)$ complexity $\to O(n^{(\omega+1)/2})$
($\approx O(n^{1.9})$ for Strassen)

Relative vs absolute threshold

Theorem

The FCU BLR factorization of $A \in \mathbb{R}^{n \times n}$ with absolute threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \leq (nu + \theta\varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

where $\theta = \sqrt{n/b - 1} \sum_{i=1}^{n/b} \|L_{ii}\| + \|U_{ii}\|$

The BLR factorization with absolute threshold

- ☹ Has a faster error growth
- ☹ Is scaling-dependent

Relative vs absolute threshold

Theorem

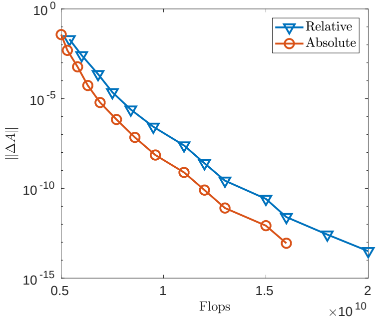
The FCU BLR factorization of $A \in \mathbb{R}^{n\times n}$ with absolute threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \le (nu + \theta\varepsilon)\|L\|\|U\| + O(u\varepsilon) + O(u^2)$$

where $\theta = \sqrt{n/b-1}\sum_{i=1}^{n/b} \|L_{ii}\| + \|U_{ii}\|$

The BLR factorization with absolute threshold

- ☹ Has a faster error growth
- ☹ Is scaling-dependent
- ☺ Is more efficient in practice



Error analysis: CFU variant

Theorem

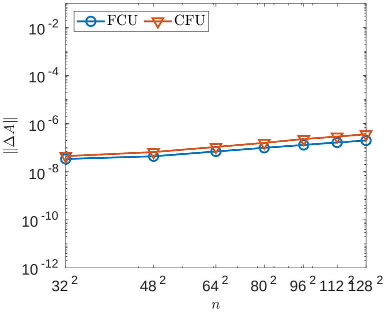
The CFU BLR factorization of $A \in \mathbb{R}^{n\times n}$ with relative threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \le (nu+\varepsilon)\|L\|\|U\| + O(\kappa(A)u\varepsilon) + O(u^2)$$

Theorem

The CFU BLR factorization of $A \in \mathbb{R}^{n\times n}$ with relative threshold ε satisfies

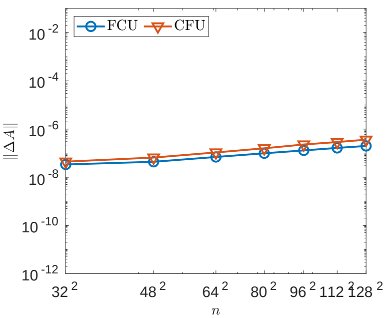
$$\|A - L_\varepsilon U_\varepsilon\| \leq (nu+\varepsilon)\|L\|\|U\| + O(\kappa(A)u\varepsilon) + O(u^2)$$

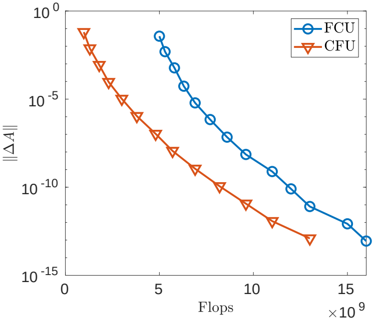


Theorem

The CFU BLR factorization of $A \in \mathbb{R}^{n\times n}$ with relative threshold ε satisfies

$$\|A - L_\varepsilon U_\varepsilon\| \leq (nu+\varepsilon)\|L\|\|U\| + O(\kappa(A)u\varepsilon) + O(u^2)$$





Section 6

Use as a low-accuracy preconditioner

N. Higham and T. Mary. *A New Preconditioner that Exploits Low-Rank Approximations to Factorization Error*. SIAM J. Sci. Comp (2018).

Low-accuracy BLR preconditioners

BLR factorization + GMRES solve with stopping tolerance 10^{-9}

Matrix	n	Time (s)		Storage (GB)	
		$\varepsilon = 10^{-2}$	$\varepsilon = 10^{-8}$	$\varepsilon = 10^{-2}$	$\varepsilon = 10^{-8}$
audikw_1	1.0M	1163	69	5	10
Bump_2911	2.9M	–	282	34	56
Emilia_923	0.9M	304	63	7	12
Fault_639	0.6M	–	45	5	9
Ga41As41H72	0.3M	–	76	12	17
Hook_1498	1.5M	902	75	6	11
Si87H76	0.2M	–	62	10	14

Low-accuracy BLR solvers:

☹ are slower and less robust

☺ but require much less storage

Improved preconditioner: context

Objective

- Compute solution to linear system $Ax = b$
- $A \in \mathbb{R}^{n\times n}$ is ill conditioned

LU-based preconditioner

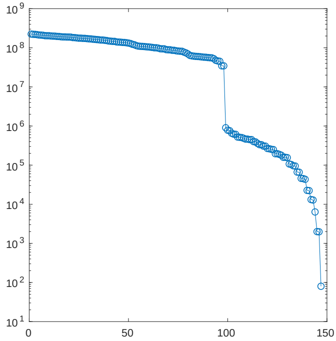
1. Compute approximate factorization $A = \widehat{L}\widehat{U} + \Delta A$
   - Half-precision factorization
   - Incomplete LU factorization
   - Structured matrix factorization: Block Low-Rank, $\mathcal{H}$, HSS,...
2. Solve $\Pi_{LU}Ax = \Pi_{LU}b$ with $\Pi_{LU} = \widehat{U}^{-1}\widehat{L}^{-1}$ via some iterative method

- Convergence to solution may be slow or fail

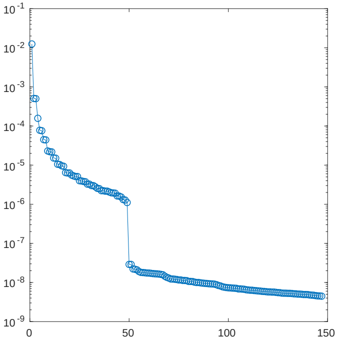
$\Rightarrow$ **Objective: accelerate convergence**

Improved preconditioner: key observation

Matrix lund_a ($n = 147$, $\kappa(A) = 2.8e{+}06$)



SVD of A



SVD of A^{-1}

- Often, A is ill conditioned due to a small number of small singular values
- Then, A^{-1} is numerically low-rank

Improved preconditioner: key idea

Factorization error might be low-rank?

Let the error $E = \widehat{U}^{-1}\widehat{L}^{-1}A - I = \widehat{U}^{-1}\widehat{L}^{-1}(\widehat{L}\widehat{U} + \Delta A) - I$

$$= \widehat{U}^{-1}\widehat{L}^{-1}\Delta A \approx A^{-1}\Delta A$$

Does E retain the low-rank property of A^{-1}?

A novel preconditioner

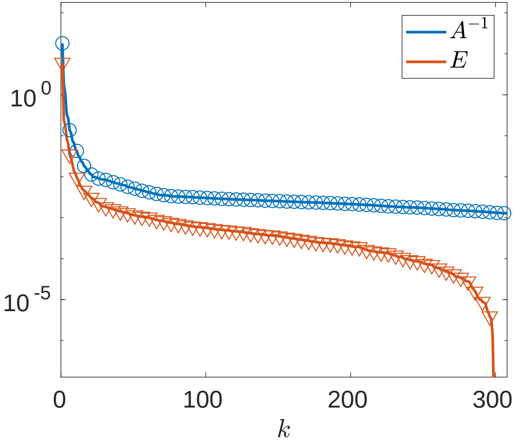
Consider the preconditioner

$$\Pi_{E_k} = (I + E_k)^{-1}\Pi_{LU}$$

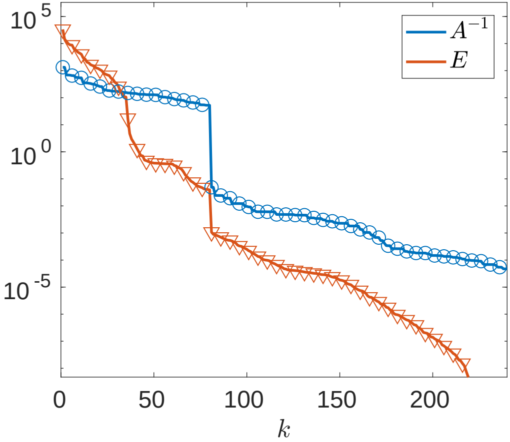
with E_k a rank-k approximation to E.

- If $E = E_k$, $\Pi_{E_k} = A^{-1}$
- If $E \approx E_k$ for some small k, Π_{E_k} can be computed cheaply

Typical SV distributions of A^{-1} and E

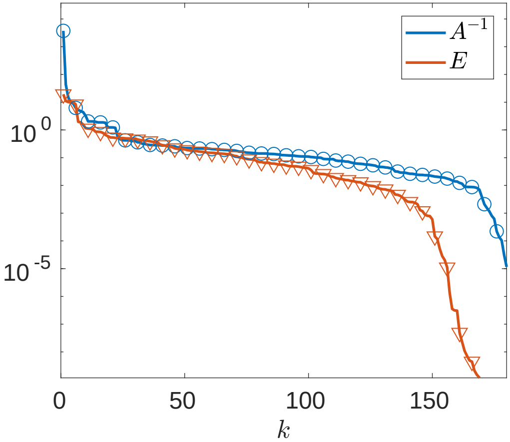


Matrix cz308



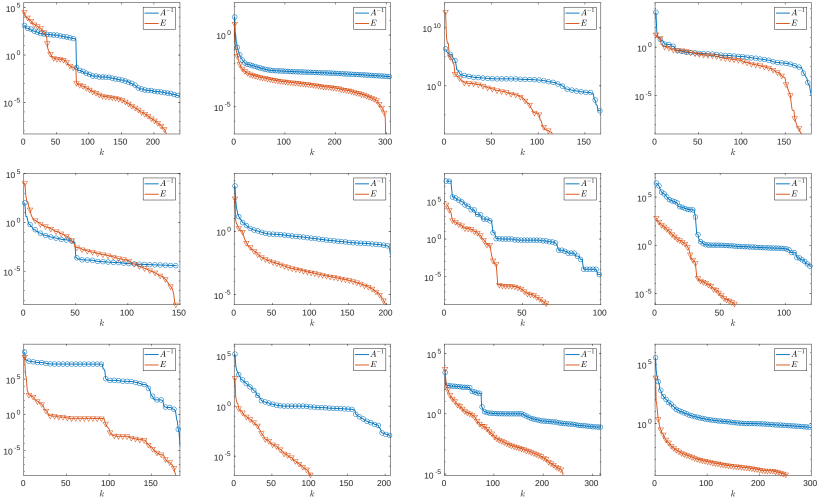
Matrix steam1

Typical SV distributions of A^{-1} and E



Matrix rajat14

Typical SV distributions of A^{-1} and E



We did **not** specifically select matrices for which A^{-1} is low-rank!

Computing E_k

We need to compute a rank-k approximation of

$$E = \widehat{U}^{-1}\widehat{L}^{-1}A - I$$

E cannot be built explicitly! $\Rightarrow$ use **randomized** method

Algorithm 1 Randomized SVD via direct SVD of $V^T E$.

1: Sample E: $S = E\Omega$, with Ω a $n \times (k+p)$ random matrix.
2: Orthonormalize S: $V = \text{qr}(S)$. $\{\Rightarrow E \approx VV^T E.\}$
3: Compute truncated SVD $V^T E \approx X_k \Sigma_k Y_k^T$.
4: $E_k \approx (VX_k)\Sigma_k Y_k^T$.

Results for $\varepsilon = 10^{-2}$:

Matrix	Π_{LU}		Π_{E_k}	
	Iter.	Time	Iter.	Time
audikw_1	691	1163	331	625
Bump_2911	–	–	284	1708
Emilia_923	174	304	136	267
Fault_639	–	–	294	345
Ga41As41H72	–	–	135	143
Hook_1498	417	902	356	808
Si87H76	–	–	131	116

$\Rightarrow$ **performance and robustness improvement with zero storage overhead**

Conclusion

Takeaway messages

- BLR factorization achieves quadratic dense complexity but (quasi-)linear sparse complexity with a small number of levels
- A large fraction of this theoretical reduction is converted into actual time gains, even on large numbers of cores
- It is numerically stable thanks to numerical pivoting and can efficiently exploit low-precision floating-point arithmetic

$\Rightarrow$ **Good compromise between complexity, performance, and accuracy**

Perspectives (my objective for the next $O(10)$ years)

Develop a distributed-memory, high-performance implementation of a multifrontal, multilevel, multiprecision BLR solver

Slides and papers available here

`bit.ly/theomary` (list of references on next slide)

References

P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *On the Complexity of the Block Low-Rank Multifrontal Factorization*. SIAM J. Sci. Comput. (2017).

P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *Bridging the gap between flat and hierarchical low-rank matrix formats: the multilevel BLR format*. Submitted (2018).

P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures*. ACM Trans. Math. Soft. (2018).

C. Gorman, G. Chavez, P .Ghysels, T. Mary, F.-H. Rouet, and X. S. Li. *Matrix-free Construction of HSS Representation Using Adaptive Randomized Sampling*. Submitted (2018).

N. Higham and T. Mary. *A New Preconditioner that Exploits Low-Rank Approximations to Factorization Error*. SIAM J. Sci. Comp (2018).

N. Higham and T. Mary. *A New Approach to Probabilistic Rounding Error Analysis*. Submitted (2018).

P. Amestoy, R. Brossier, A. Buttari, J.-Y. L'Excellent, T. Mary, L. Métivier, A. Miniussi, and S. Operto. *Fast 3D frequency-domain full waveform inversion with a parallel Block Low-Rank multifrontal direct solver: application to OBC data from the North Sea*. Geophysics (2016).

D. Shantsev, P. Jaysaval, S. Kethulle de Ryhove, P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. *Large-scale 3D EM modeling with a Block Low-Rank multifrontal direct solver*. Geophys. J. Int (2017).