Complexity and performance of the Block Low-Rank multifrontal factorization and its variants

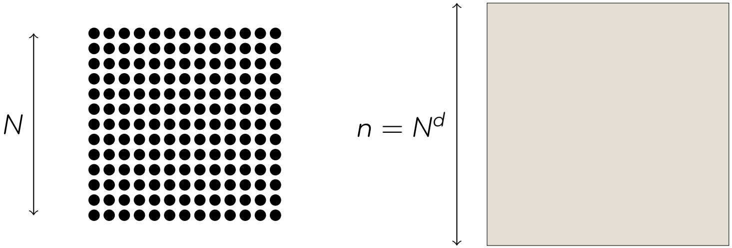
P. Amestoy[*,1] A. Buttari[*,2] J.-Y. L'Excellent[†,3] T. Mary[*,4]

[*]Université de Toulouse [†]ENS Lyon
[1]INPT-IRIT [2]CNRS-IRIT [3]INRIA-LIP [4]UPS-IRIT

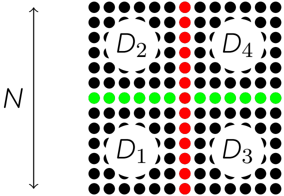
SIAM PP'16, Paris Apr. 12-15

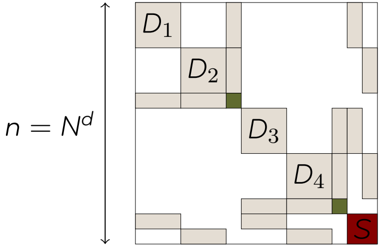
Introduction

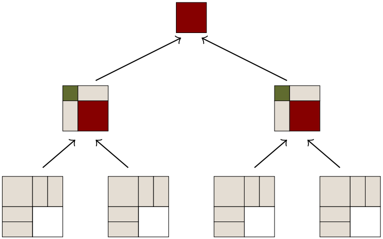
Multifrontal (Duff '83) with Nested Dissection (George '73)



Multifrontal (Duff '83) with Nested Dissection (George '73)

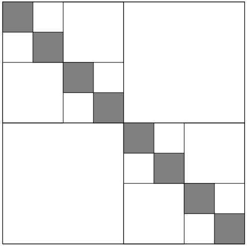




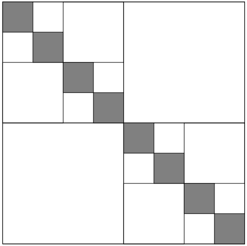


3D problem cost $\propto$

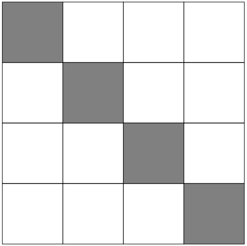
$\rightarrow$ Flops:$O(n^2)$, mem:$O(n^{4/3})$



$\mathcal{H}$-matrix

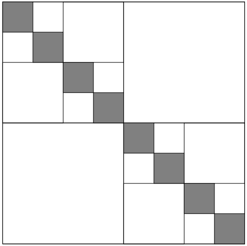


$\mathcal{H}$-matrix

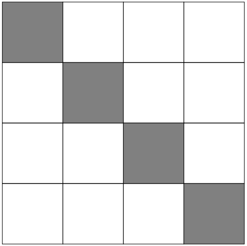


BLR matrix

$\mathcal{H}$ and BLR matrices



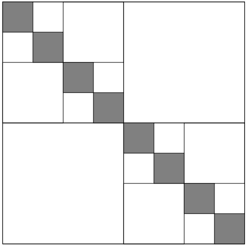
$\mathcal{H}$-matrix



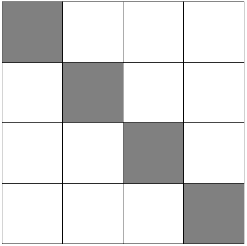
BLR matrix

A block B represents the interaction between two subdomains. If they have a small diameter and are far away their interaction is weak $\Rightarrow$ rank is low.

$\mathcal{H}$ and BLR matrices



$\mathcal{H}$-matrix



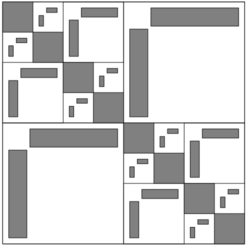
BLR matrix

A block B represents the interaction between two subdomains. If they have a small diameter and are far away their interaction is weak $\Rightarrow$ rank is low.

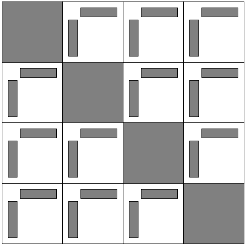
$$\tilde{B} = XY^T \text{ such that } \mathrm{rank}(\tilde{B}) = k_\varepsilon \text{ and } \|B - \tilde{B}\| \leq \varepsilon$$

If $k_\varepsilon \ll \mathrm{size}(B) \Rightarrow$ memory and flops can be reduced with a controlled loss of accuracy ($\leq \varepsilon$)

$\mathcal{H}$ and BLR matrices



$\mathcal{H}$-matrix



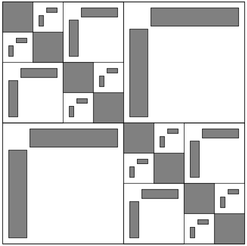
BLR matrix

A block B represents the interaction between two subdomains. If they have a small diameter and are far away their interaction is weak $\Rightarrow$ rank is low.

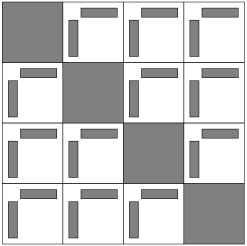
$$\tilde{B} = XY^T \text{ such that } \mathrm{rank}(\tilde{B}) = k_\varepsilon \text{ and } \|B - \tilde{B}\| \leq \varepsilon$$

If $k_\varepsilon \ll \mathrm{size}(B) \Rightarrow$ memory and flops can be reduced with a controlled loss of accuracy ($\leq \varepsilon$)

$\mathcal{H}$ and BLR matrices



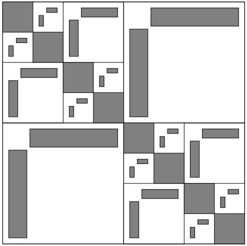
$\mathcal{H}$-matrix



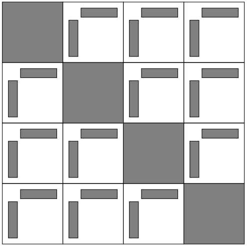
BLR matrix

- Leads to very low theoretical complexity
- Complex, hierarchical structure

- Simple structure
- Theoretical complexity?



$\mathcal{H}$-matrix



BLR matrix

$\Rightarrow$ Our hope is to find a good comprise between theoretical complexity and performance/usability

Questions that will be answered in this talk

- Is the complexity of the BLR factorization asymptotically better than the full-rank one? (i.e., in $O(n^{\alpha})$, with $\alpha < 2$ and where n is the number of unknowns)

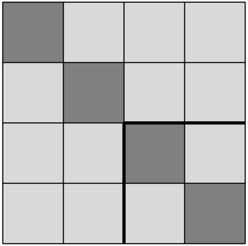
- What are the different variants of the BLR factorization? Do they improve its complexity?

- How well does the complexity improvement translate into a performance gain?

- How parallel is the BLR factorization? What about its variants?

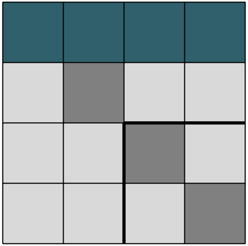
Variants of the BLR factorization

Variants of the BLR LU factorization



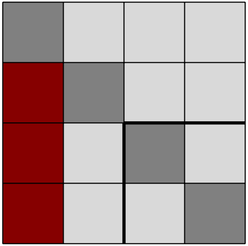
- FSCU

Variants of the BLR LU factorization



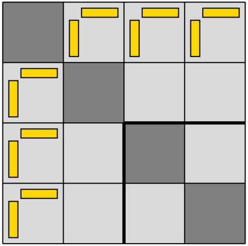
- FSCU (Factor,

Variants of the BLR LU factorization



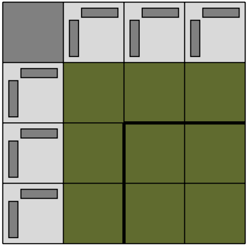
- FSCU (Factor, Solve,

Variants of the BLR LU factorization



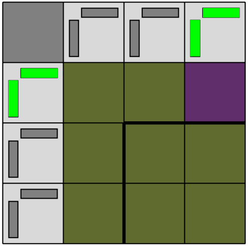
- FSCU (Factor, Solve, Compress,

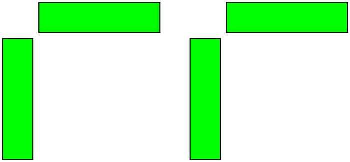
Variants of the BLR LU factorization



- FSCU (Factor, Solve, Compress, Update)

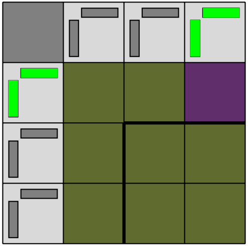
Variants of the BLR LU factorization

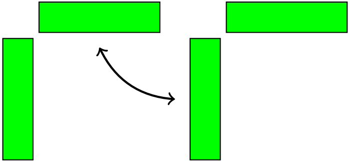




- FSCU (Factor, Solve, Compress, Update)

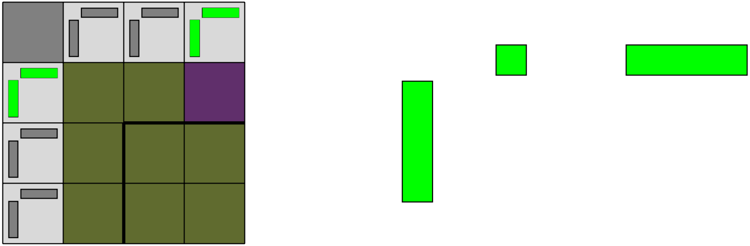
Variants of the BLR LU factorization





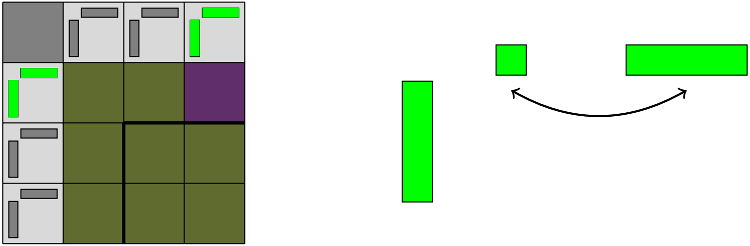
- FSCU (Factor, Solve, Compress, Update)

Variants of the BLR LU factorization



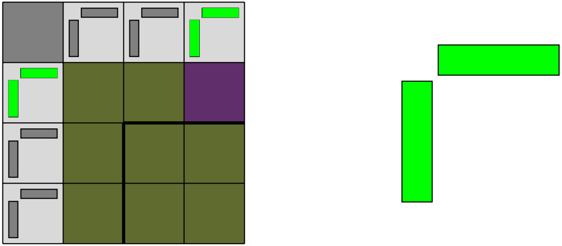
- FSCU (Factor, Solve, Compress, Update)

Variants of the BLR LU factorization



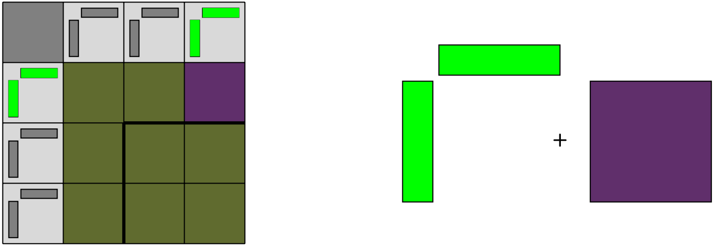
- FSCU (Factor, Solve, Compress, Update)

Variants of the BLR LU factorization



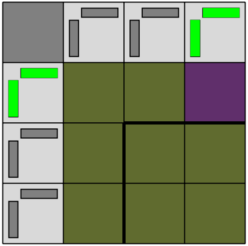
- FSCU (Factor, Solve, Compress, Update)

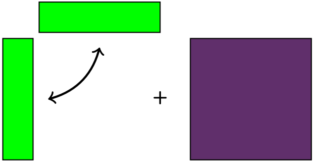
Variants of the BLR LU factorization



- FSCU (Factor, Solve, Compress, Update)

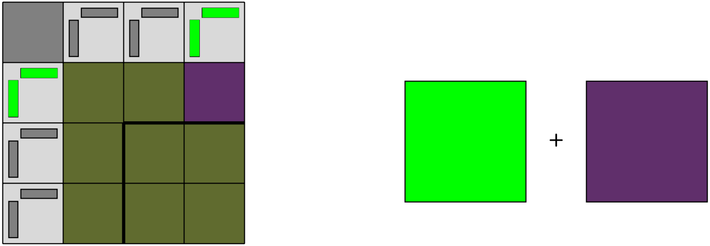
Variants of the BLR LU factorization





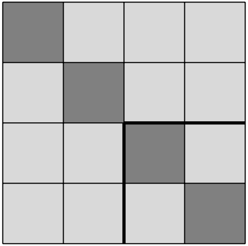
- FSCU (Factor, Solve, Compress, Update)

Variants of the BLR LU factorization



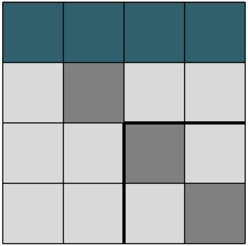
- FSCU (Factor, Solve, Compress, Update)

Variants of the BLR LU factorization



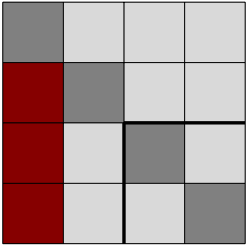
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA

Variants of the BLR LU factorization



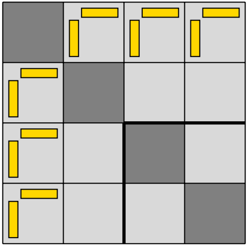
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA

Variants of the BLR LU factorization



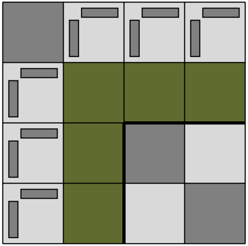
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA

Variants of the BLR LU factorization



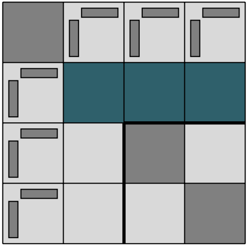
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA

Variants of the BLR LU factorization



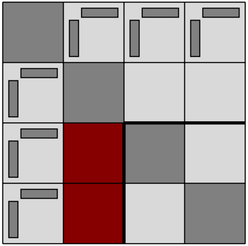
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

Variants of the BLR LU factorization



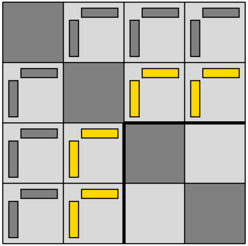
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

Variants of the BLR LU factorization



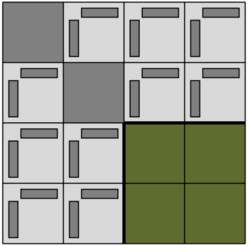
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

Variants of the BLR LU factorization



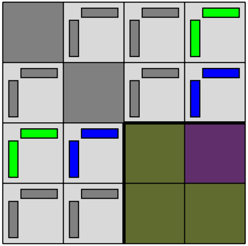
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

Variants of the BLR LU factorization



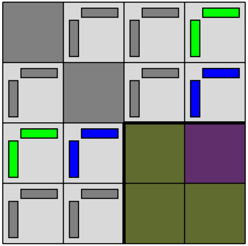
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

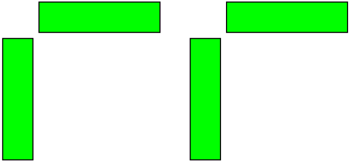
Variants of the BLR LU factorization



- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

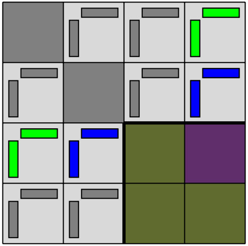
Variants of the BLR LU factorization

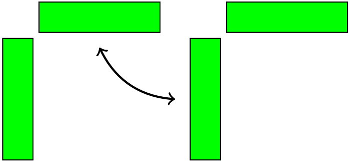




- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

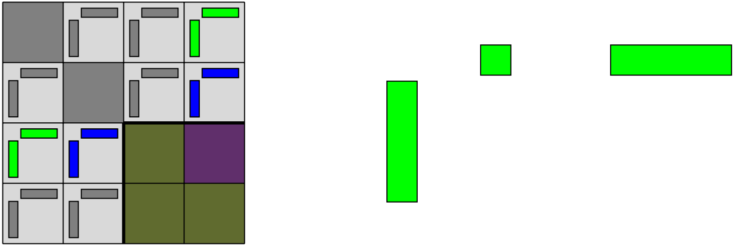
Variants of the BLR LU factorization





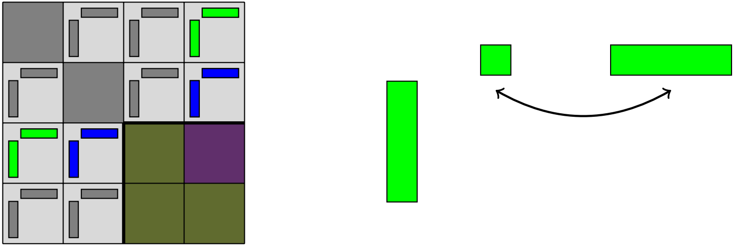
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

Variants of the BLR LU factorization



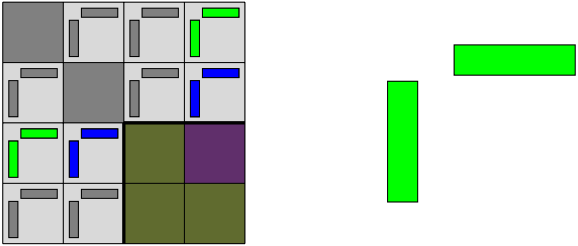
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

Variants of the BLR LU factorization



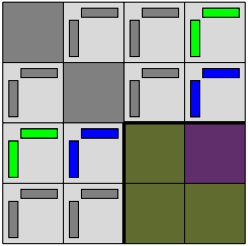
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

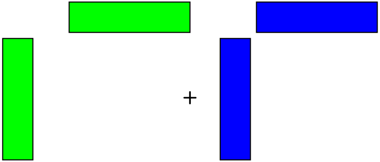
Variants of the BLR LU factorization



- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

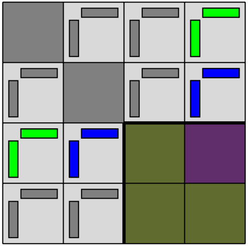
Variants of the BLR LU factorization

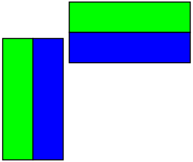




- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking

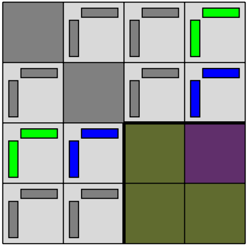
Variants of the BLR LU factorization

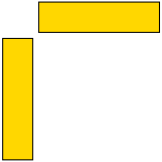




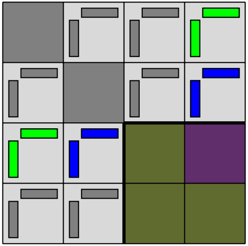
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations

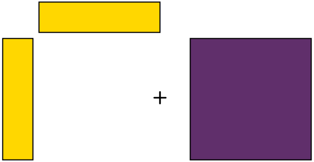
Variants of the BLR LU factorization





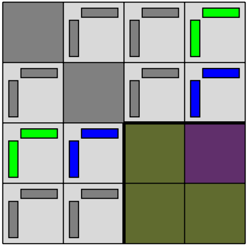
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression

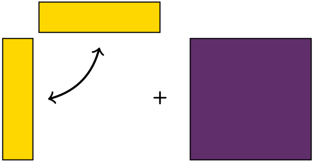




- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression

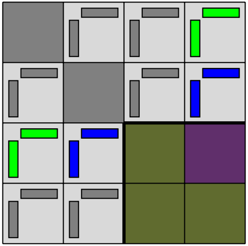
Variants of the BLR LU factorization

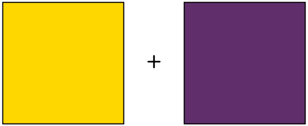




- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression

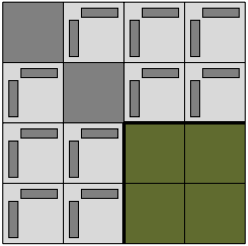
Variants of the BLR LU factorization





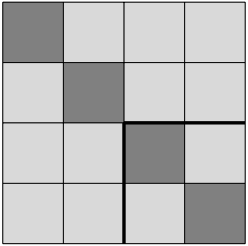
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
    - More natural in Left-looking
    - Better granularity in update operations
    - Potential recompression

Variants of the BLR LU factorization



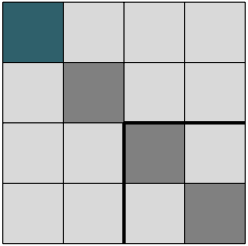
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression

Variants of the BLR LU factorization



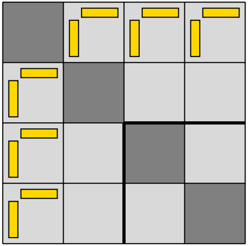
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks

Variants of the BLR LU factorization



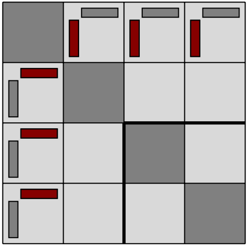
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks

Variants of the BLR LU factorization



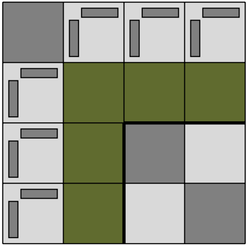
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks

Variants of the BLR LU factorization



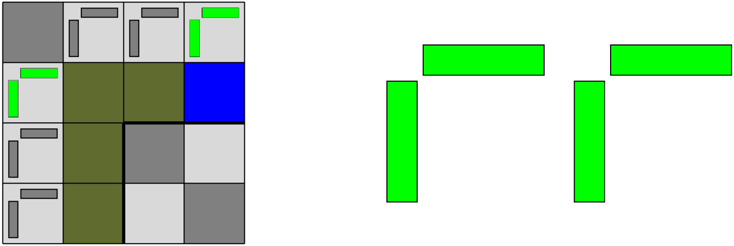
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

Variants of the BLR LU factorization



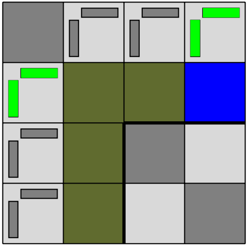
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

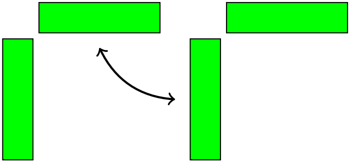
Variants of the BLR LU factorization



- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

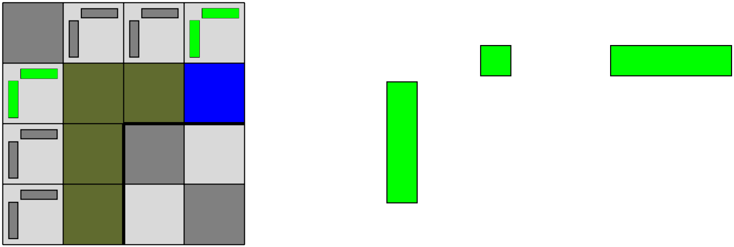
Variants of the BLR LU factorization





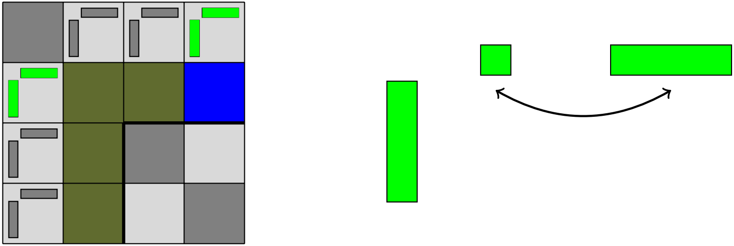
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

Variants of the BLR LU factorization



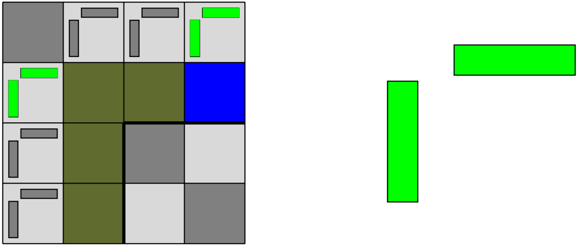
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

Variants of the BLR LU factorization



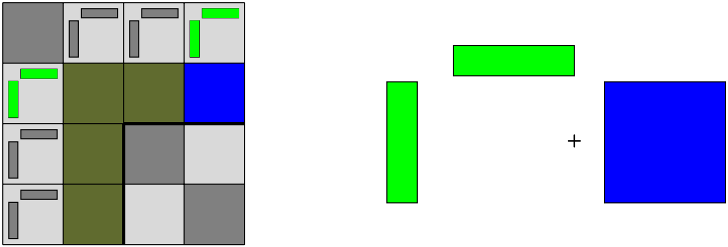
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

Variants of the BLR LU factorization



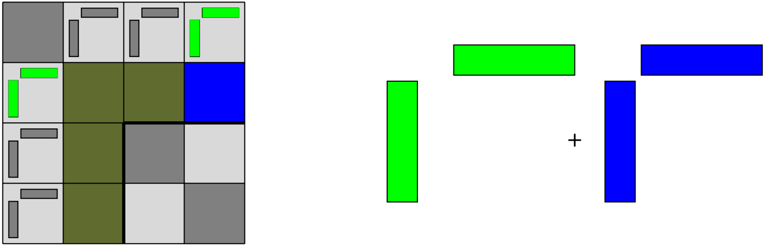
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

Variants of the BLR LU factorization

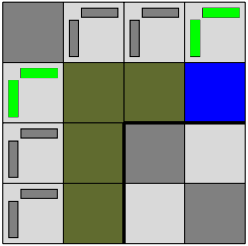


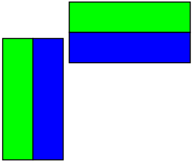
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

Variants of the BLR LU factorization



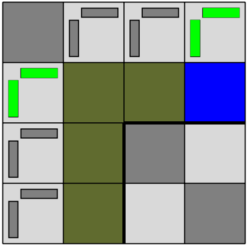
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

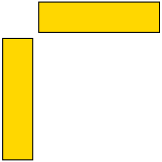




- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

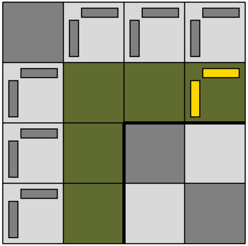
Variants of the BLR LU factorization





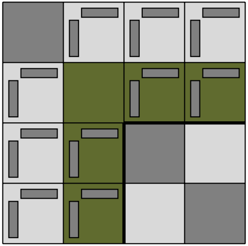
- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve

Variants of the BLR LU factorization



- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve
  - With LUA, no need to decompress accumulators

Variants of the BLR LU factorization



- FSCU (Factor, Solve, Compress, Update)
- FSCU+LUA
  - More natural in Left-looking
  - Better granularity in update operations
  - Potential recompression
- FCSU(+LUA)
  - Restricted pivoting, e.g. to diagonal blocks
  - Low-rank Solve
  - Better ratio BLAS-3/BLAS-2 in Solve
  - With LUA, no need to decompress accumulators

Complexity of the BLR factorization

Complexity of multifrontal BLR factorization

- Extended theoretical work on $\mathcal{H}$-matrices by Hackbush and Bebendorf (2003) and Bebendorf (2005, 2007) to the BLR case. Proof and computation of the theoretical complexity are available in *On the Complexity of the Block Low-Rank Multifrontal Factorization*, P. Amestoy, A. Buttari, J.-Y. L'Excellent and T. Mary (in preparation)

- Today, regarding the complexity, we focus on:
  - Presenting some important properties of the BLR complexity
  - Validating these properties experimentally

Complexity of multifrontal BLR factorization

	operations (OPC)		factor size (NNZ)	
	$r = O(1)$	$r = O(n^{\frac{1}{3}})$	$r = O(1)$	$r = O(n^{\frac{1}{3}})$
FR	$O(n^2)$	$O(n^2)$	$O(n^{\frac{4}{3}})$	$O(n^{\frac{4}{3}})$
BLR FSCU	$O(n^{\frac{5}{3}})$	$O(n^{\frac{11}{6}})$	$O(n \log n)$	$O(n^{\frac{4}{3}})$
BLR FSCU+LUA	$O(n^{\frac{14}{9}})$	$O(n^{\frac{16}{9}})$	$O(n \log n)$	$O(n^{\frac{4}{3}})$
BLR FCSU+LUA	$O(n^{\frac{4}{3}})$	$O(n^{\frac{5}{3}} \log n)$	$O(n \log n)$	$O(n^{\frac{4}{3}})$
$\mathcal{H}$	$O(n^{\frac{4}{3}})$	$O(n^{\frac{5}{3}})$	$O(n)$	$O(n^{\frac{7}{6}})$
$\mathcal{H}$ (fully struct.)	$O(n)$	$O(n^{\frac{4}{3}})$	$O(n)$	$O(n^{\frac{7}{6}})$

in the 3D case (similar analysis possible for 2D)

Important properties:

- The complexity of the standard BLR variant (FSCU) has a lower exponent than the full-rank one
- Each variant further improves the complexity, with the best one (FCSU+LUA) being not so far from the $\mathcal{H}$-case
- These properties hold for different rank bound assumptions, e.g. $r = O(1)$ or $r = O(N) = O(n^{\frac{1}{3}})$

Experimental Setting: Matrices

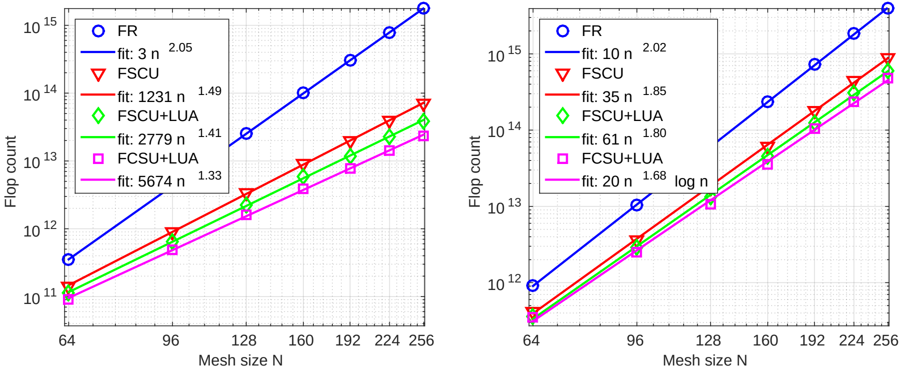
1. **`Poisson`**: N^3 grid with a 7-point stencil with $u = 1$ on the boundary $\partial\Omega$
$$\Delta u = f$$

2. **`Helmholtz`**: N^3 grid with a 27-point stencil, ω is the angular frequency, $v(x)$ is the seismic velocity field, and $u(x,\omega)$ is the time-harmonic wavefield solution to the forcing term $s(x,\omega)$.
$$\left(-\Delta - \frac{\omega^2}{v(x)^2}\right) u(x,\omega) = s(x,\omega)$$

ω is fixed and equal to 4Hz.

Experimental MF complexity: operations

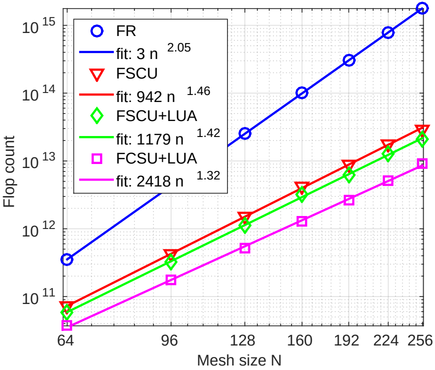
OPC (Poisson, $\varepsilon = 10^{-10}$) OPC (Helmholtz, $\varepsilon = 10^{-5}$)



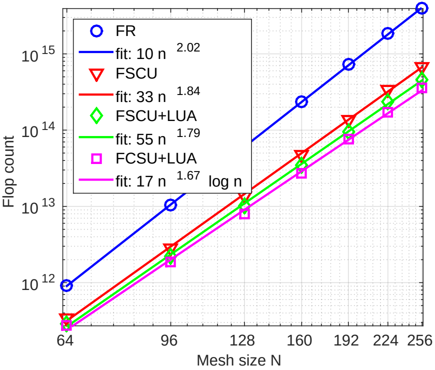
- good agreement with theoretical complexity

Experimental MF complexity: operations

OPC (Poisson, $\varepsilon = 10^{-6}$)



OPC (Helmholtz, $\varepsilon = 10^{-4}$)



- good agreement with theoretical complexity
- ε only plays a role in the constant factor

Performance results

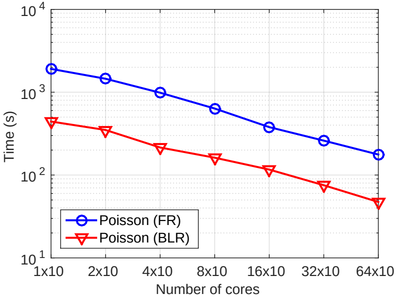
Experimental Setting: Machines

1. Distributed memory experiments are done on the `eos` supercomputer at the CALMIP center of Toulouse (grant 2014-P0989):
   - Two Intel(r) 10-cores Ivy Bridge @ 2,8 GHz
   - Peak per core is 22.4 GF/s
   - 64 GB memory per node
   - Infiniband FDR interconnect

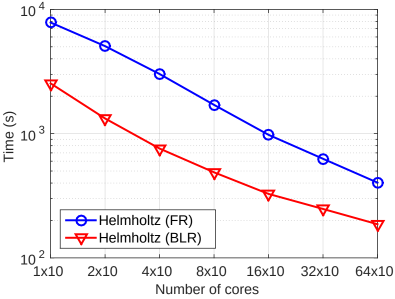
2. Shared memory experiments are done on `grunch` at the LIP laboratory of Lyon:
   - Two Intel(r) 14-cores Haswell @ 2,3 GHz
   - Peak per core is 36.8 GF/s
   - Total memory is 768 GB

Scalability of the BLR factorization (distributed)

Poisson ($\varepsilon = 10^{-6}, N = 192$)



Helmholtz ($\varepsilon = 10^{-4}, N = 192$)

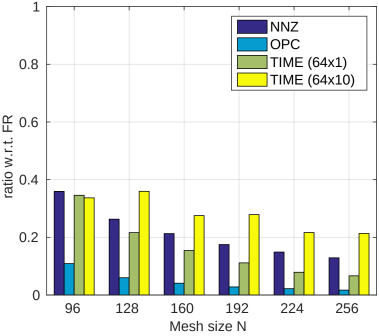


MPI+OpenMP parallelism (10 threads/MPI process, 1 MPI/node)

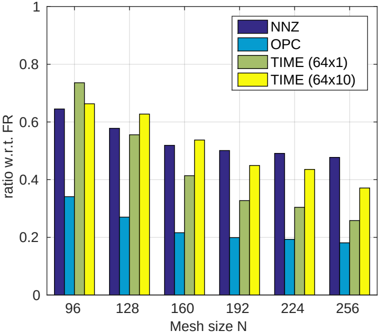
- each time the number of processes doubles, speedup of ~ 1.6
- both FR and BLR scale reasonably well
- gain due to BLR remains constant

Gains due to BLR (distributed, MPI+OpenMP)

Poisson ($\varepsilon = 10^{-6}$)



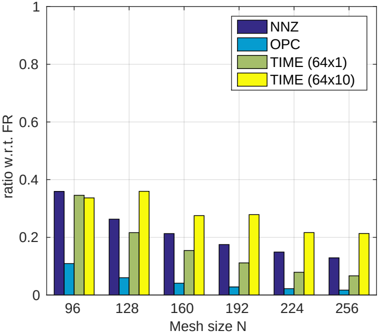
Helmholtz ($\varepsilon = 10^{-4}$)



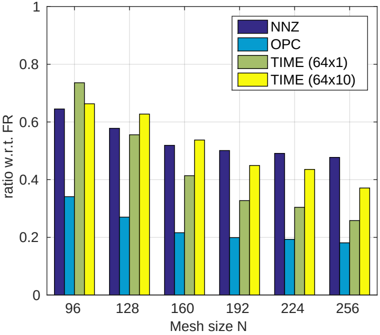
- gains increase with problem size
- gain in flops does not fully translate into gain in time
- multithreaded efficiency lower in LR than in FR
- same remarks apply to Helmoltz, to a lesser extent

Gains due to BLR (distributed, MPI+OpenMP)

Poisson ($\varepsilon = 10^{-6}$)



Helmholtz ($\varepsilon = 10^{-4}$)



- gains increase with problem size
- gain in flops does not fully translate into gain in time
- multithreaded efficiency lower in LR than in FR
- same remarks apply to Helmoltz, to a lesser extent

$\Rightarrow$ *improve multithreading with variants*

Right Looking Vs. Left-Looking (shared)

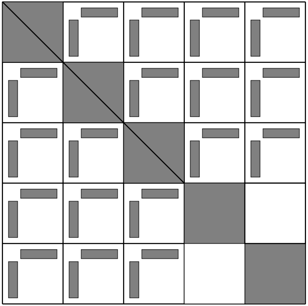
Focus on the Update step (which includes the Decompress)

		1 thread		28 threads	
		RL	LL	RL	LL
Poisson	FR	62294s	65208s	3772s	4092s
($N = 256$)	BLR	2516s	1544s	662s	183s
Helmholtz	FR			9862s	10234s
($N = 256$)	BLR			1694s	1435s

Right Looking Vs. Left-Looking (shared)

Focus on the Update step (which includes the Decompress)

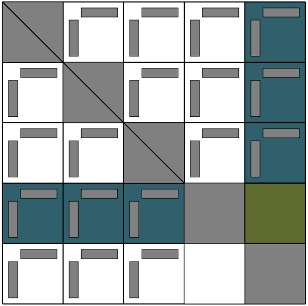
		1 thread		28 threads	
		RL	LL	RL	LL
Poisson	FR	62294s	65208s	3772s	4092s
($N = 256$)	BLR	2516s	1544s	662s	183s
Helmholtz	FR			9862s	10234s
($N = 256$)	BLR			1694s	1435s



Right Looking Vs. Left-Looking (shared)

Focus on the Update step (which includes the Decompress)

		1 thread RL	1 thread LL	28 threads RL	28 threads LL
Poisson	FR	62294s	65208s	3772s	4092s
($N = 256$)	BLR	2516s	1544s	662s	183s
Helmholtz	FR			9862s	10234s
($N = 256$)	BLR			1694s	1435s

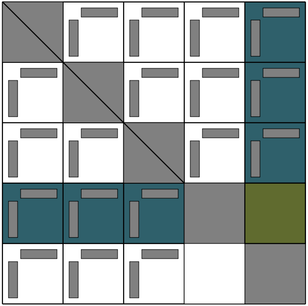


- in RL: FR (green) block is accessed many times; LR (blue) blocks are accessed once
- in LL: FR (green) block is accessed once; LR (blue) blocks are accessed many times

Right Looking Vs. Left-Looking (shared)

Focus on the Update step (which includes the Decompress)

		1 thread		28 threads	
		RL	LL	RL	LL
Poisson ($N = 256$)	FR	62294s	65208s	3772s	4092s
	BLR	2516s	1544s	662s	183s
Helmholtz ($N = 256$)	FR			9862s	10234s
	BLR			1694s	1435s



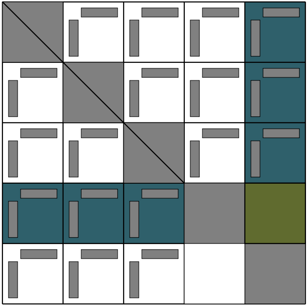
- in RL: FR (green) block is accessed many times; LR (blue) blocks are accessed once
- in LL: FR (green) block is accessed once; LR (blue) blocks are accessed many times

$\Rightarrow$ lower volume of memory transfers (more critical in multithreaded)

Right Looking Vs. Left-Looking (shared)

Focus on the Update step (which includes the Decompress)

		1 thread		28 threads	
		RL	LL	RL	LL
Poisson ($N = 256$)	FR	62294s	65208s	3772s	4092s
	BLR	2516s	1544s	662s	183s
Helmholtz ($N = 256$)	FR			9862s	10234s
	BLR			1694s	1435s

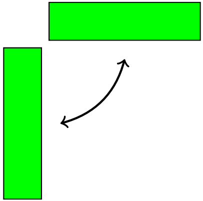


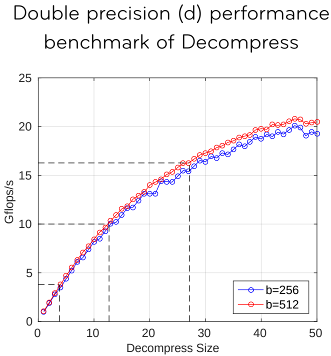
- in RL: FR (green) block is accessed many times; LR (blue) blocks are accessed once
- in LL: FR (green) block is accessed once; LR (blue) blocks are accessed many times

$\Rightarrow$ lower volume of memory transfers (more critical in multithreaded)

$\Rightarrow$ *the Decompress part (135s) remains the bottleneck of the Update (183s)*

Performance of LUA (shared, 28 threads)

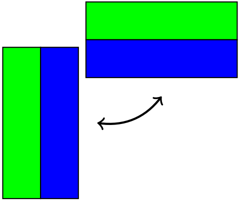




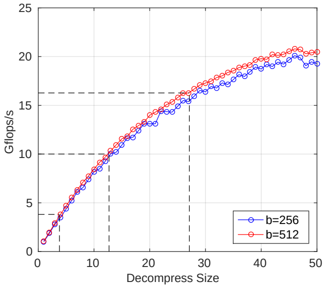
	Poisson ($N = 256$)			Helmholtz ($N = 256$)		
	LL	LUA	LUA +Rec.*	LL	LUA	LUA +Rec.*
Flops in Update ($\times 10^{13}$)	1.0	1.0	0.58	43	43	30
Avg. decompress size	3.8	27.1	12.7	31.3	264.2	136.8
Time in Update	183s	87s	110s	1435s	1304s	1295s
% of peak reached	5%	11%	5%	59%	65%	45%

* All metrics include the Recompression overhead

Performance of LUA (shared, 28 threads)



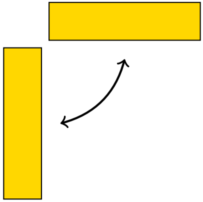
Double precision (d) performance benchmark of Decompress



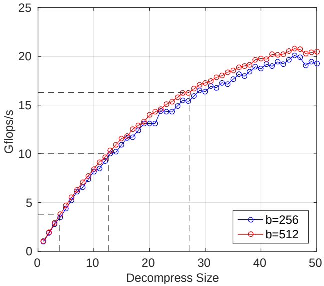
	Poisson ($N = 256$)			Helmholtz ($N = 256$)		
	LL	LUA	LUA +Rec.*	LL	LUA	LUA +Rec.*
Flops in Update ($\times 10^{13}$)	1.0	1.0	0.58	43	43	30
Avg. decompress size	3.8	27.1	12.7	31.3	264.2	136.8
Time in Update	183s	87s	110s	1435s	1304s	1295s
% of peak reached	5%	11%	5%	59%	65%	45%

* All metrics include the Recompression overhead

Performance of LUA (shared, 28 threads)



Double precision (d) performance benchmark of Decompress

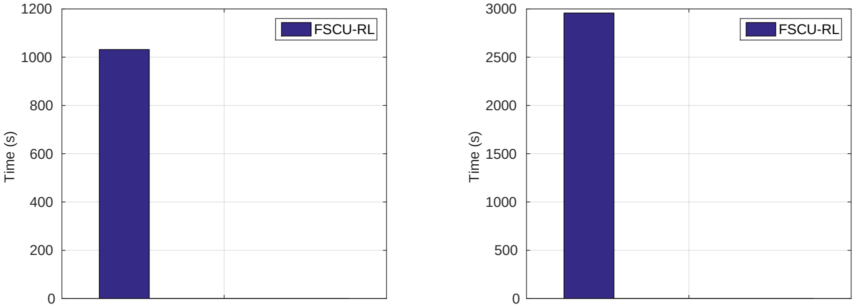


	Poisson ($N = 256$)			Helmholtz ($N = 256$)		
	LL	LUA	LUA +Rec.*	LL	LUA	LUA +Rec.*
Flops in Update ($\times 10^{13}$)	1.0	1.0	0.58	43	43	30
Avg. decompress size	3.8	27.1	12.7	31.3	264.2	136.8
Time in Update	183s	87s	110s	1435s	1304s	1295s
% of peak reached	5%	11%	5%	59%	65%	45%

* All metrics include the Recompression overhead

Performance of BLR variants (shared, 28 threads)

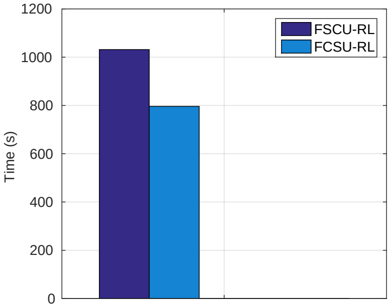
Poisson ($\varepsilon = 10^{-6}, N = 256$) Helmholtz ($\varepsilon = 10^{-3}, N = 256$)



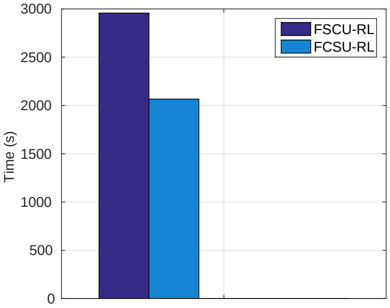
- Non-computational time (~ 300s) is not included $\Rightarrow$ addressed in MPI by tree parallelism and in OpenMP by W. Sid-Lakhdar's PhD thesis work (2014)

Performance of BLR variants (shared, 28 threads)

Poisson ($\varepsilon = 10^{-6}, N = 256$)



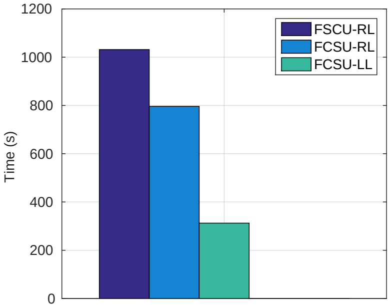
Helmholtz ($\varepsilon = 10^{-3}, N = 256$)



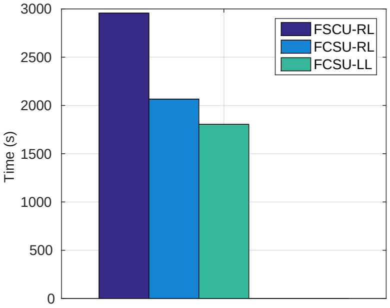
- Non-computational time (~ 300s) is not included $\Rightarrow$ addressed in MPI by tree parallelism and in OpenMP by W. Sid-Lakhdar's PhD thesis work (2014)
- FCSU: Factor+Solve greatly reduced

Performance of BLR variants (shared, 28 threads)

Poisson ($\varepsilon = 10^{-6}, N = 256$)



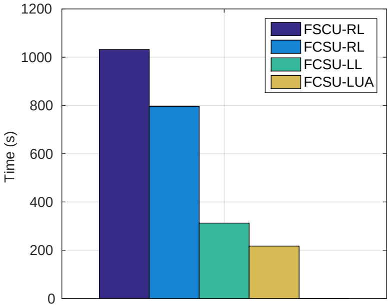
Helmholtz ($\varepsilon = 10^{-3}, N = 256$)



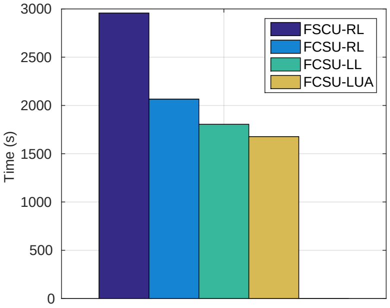
- Non-computational time (~ 300s) is not included $\Rightarrow$ addressed in MPI by tree parallelism and in OpenMP by W. Sid-Lakhdar's PhD thesis work (2014)
- FCSU: Factor+Solve greatly reduced
- LL: Update reduced thanks to lower volume of communications

Performance of BLR variants (shared, 28 threads)

Poisson ($\varepsilon = 10^{-6}, N = 256$)



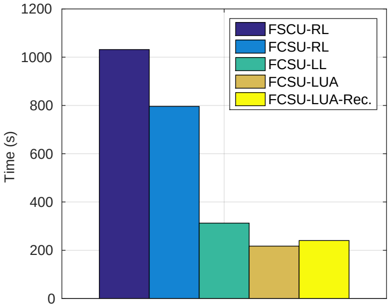
Helmholtz ($\varepsilon = 10^{-3}, N = 256$)



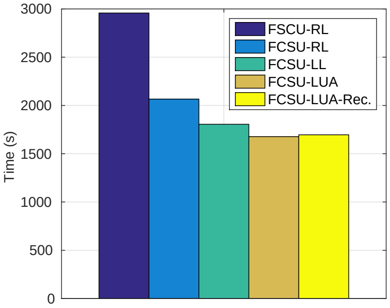
- Non-computational time (~ 300s) is not included $\Rightarrow$ addressed in MPI by tree parallelism and in OpenMP by W. Sid-Lakhdar's PhD thesis work (2014)
- FCSU: Factor+Solve greatly reduced
- LL: Update reduced thanks to lower volume of communications
- LUA: Update (Decompress) reduced thanks to better granularities

Performance of BLR variants (shared, 28 threads)

Poisson ($\varepsilon = 10^{-6}, N = 256$)



Helmholtz ($\varepsilon = 10^{-3}, N = 256$)



- Non-computational time (~ 300s) is not included $\Rightarrow$ addressed in MPI by tree parallelism and in OpenMP by W. Sid-Lakhdar's PhD thesis work (2014)
- FCSU: Factor+Solve greatly reduced
- LL: Update reduced thanks to lower volume of communications
- LUA: Update (Decompress) reduced thanks to better granularities
- Recompression: potential flop reduction not translated into a time gain yet

Conclusion and perspectives

Complexity results

- Theoretical complexity of the BLR (multifrontal) factorization is asymptotically better than FR
- Studied BLR variants to further reduce complexity by achieving higher compression
- Numerical experiments show experimental complexity in agreement with theoretical one

Performance results

- BLR variants possess better properties (efficiency, granularity, volume of communications, number of operations) $\Rightarrow$ leads to a considerable speedup w.r.t. standard BLR variant...
- ...which itself achieves up to 4.7 (Poisson) and 2.7 (Helmholtz) speedup w.r.t. FR

Perspectives

- Implementation and performance analysis of the BLR variants in distributed memory (MPI+OpenMP parallelism)
- Efficient strategies to recompress accumulators (cf. J. Anton's talk)
- Pivoting strategies compatible with the BLR variants
- Influence of the BLR variants on the accuracy of the factorization

Perspectives

- Implementation and performance analysis of the BLR variants in distributed memory (MPI+OpenMP parallelism)
- Efficient strategies to recompress accumulators (cf. J. Anton's talk)
- Pivoting strategies compatible with the BLR variants
- Influence of the BLR variants on the accuracy of the factorization

Acknowledgements

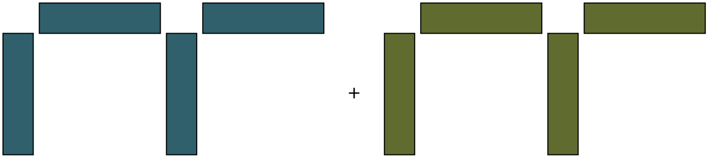
- CALMIP, BULL and LIP for providing access to the machines
- SEISCOPE for providing the Helmholtz Generator
- LSTC members for scientific discussions

Thanks!

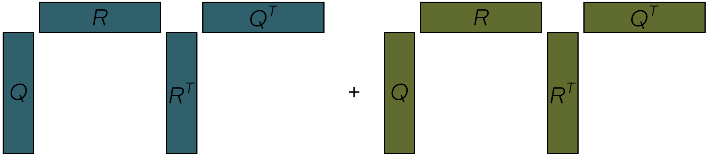
Questions?

Backup Slides

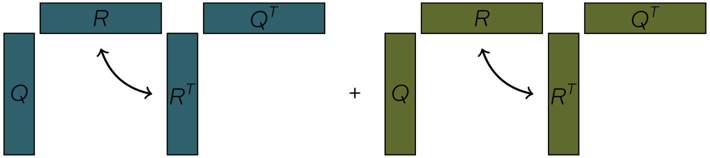
Accumulator recompression



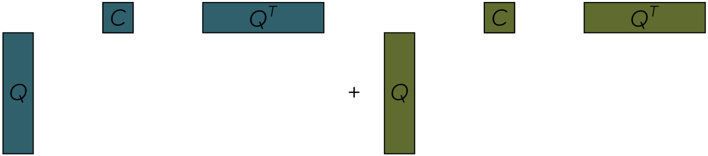
Accumulator recompression



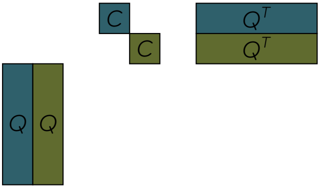
Accumulator recompression



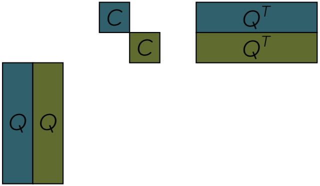
Accumulator recompression



Accumulator recompression



Accumulator recompression



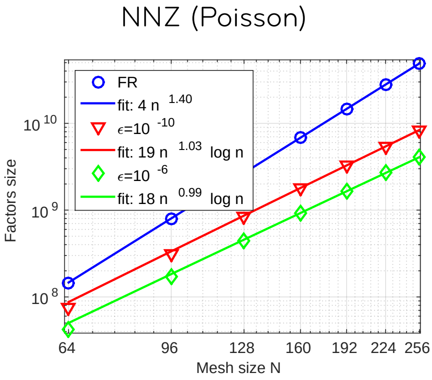
- Weight recompression on $\{C_i\}_i$

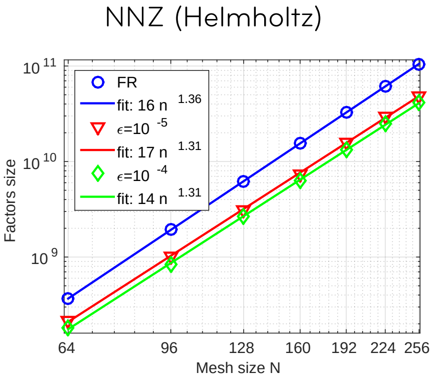
  $\Rightarrow$ With absolute threshold ε, each C_i can be compressed separately

- Redundancy recompression on $\{Q_i\}_i$

  $\Rightarrow$ Bigger recompression overhead, when is it worth it?

Experimental MF complexity: entries in factor





- good agreement with theoretical complexity
- ε only plays a role in the constant factor