Block Low-Rank (BLR) approximations to improve multifrontal sparse solvers

Joint work with Patrick Amestoy , Cleve Ashcraft, Olivier Boiteau, Alfredo Buttari and Jean-Yves L'Excellent, PhD started on October 1st, 2010 and financed by EDF.

Clément Weisbecker, ENSEEIHT-IRIT, University of Toulouse, France

Sparse Days 2013 — C.E.R.F.A.C.S., Toulouse, France, June 18th 2013

Introduction

Block **Low-Rank approximations** to improve **multifrontal sparse solvers**

Multifrontal solver

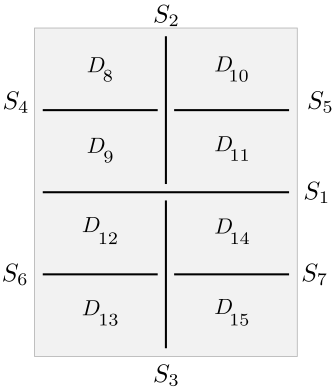
- direct solver for large linear systems
- objective: $A = LU$

Low-rank approximations

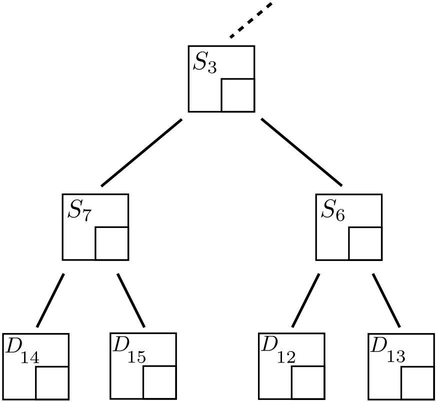
- compression and flop reduction
- accuracy controlled by a numerical parameter

$\Rightarrow$ Combine these two notions to improve multifrontal solvers (in the context of `MUMPS`)

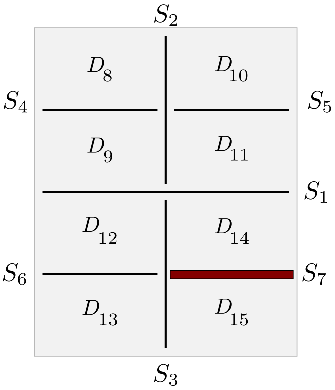
The multifrontal method

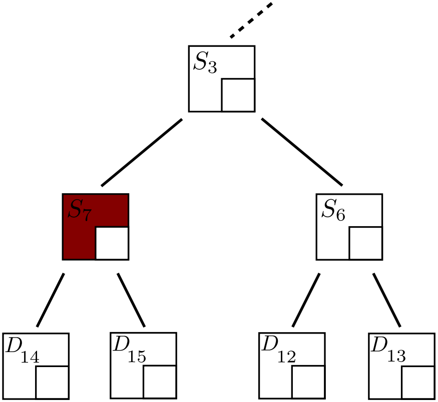


Nested dissection



Elimination tree

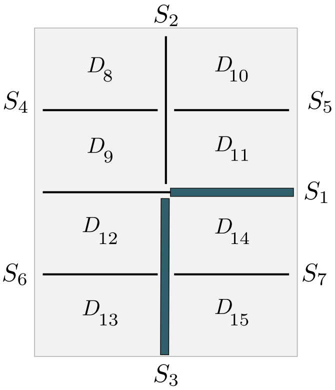


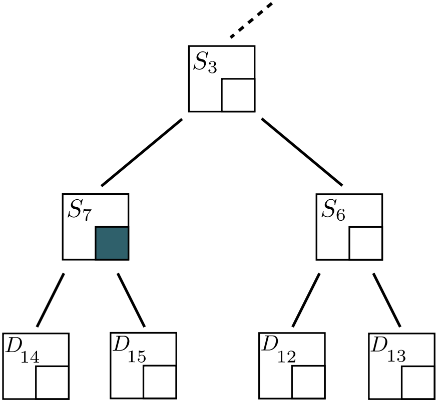


Nested dissection

Elimination tree

Fully-summed variables (FS) = separator



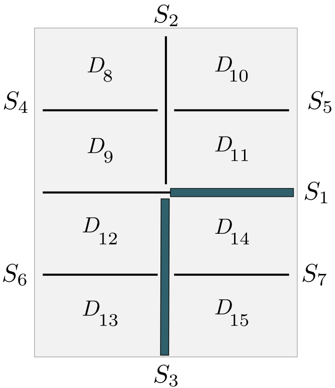


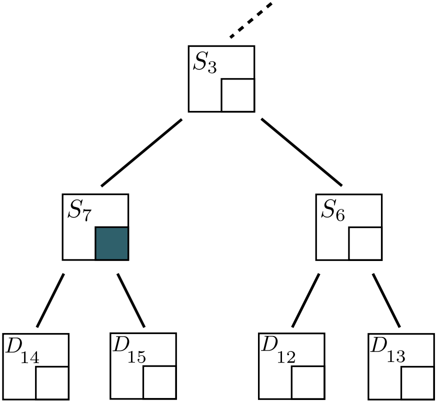
Nested dissection

Elimination tree

Non fully-summed variables (NFS) = border

The multifrontal method [Duff & Reid ’83]





Nested dissection

Elimination tree

Non fully-summed variables (NFS) = border

$\Rightarrow$ stack of CBs

Low-rank approximations

Low-rank matrices

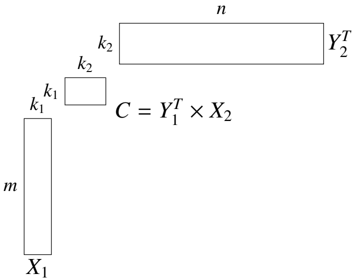
Low-rank block (Bebendorf)

Consider a block B of size $m \times n$ and k_ε its approximated numerical rank at accuracy ε. B is said to be low-rank if it can be written as

$$B = X \cdot Y + E \text{ with } \|E\|_2 \leq \varepsilon \text{ and } k_\varepsilon(m+n) < mn$$

If B is low-rank, storing it as X, Y saves storage and allows faster operations. X, Y can be computed using rank-revealing QR, SVD…

Low-rank product: $X_1(Y_1^T X_2)Y_2^T$

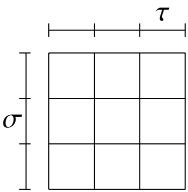


Can we exploit low-rankness in multifrontal methods?

- Fronts are not low-rank but in many applications they exhibit some low-rank blocks.

Idea: find and compress low rank blocks within frontal matrices.
Problem: how to identify low-rank blocks?

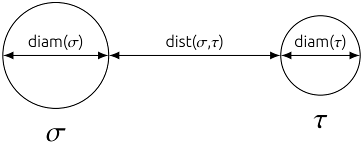
⇒ Define a *clustering* C to obtain low-rank blocks A_b
$(b = \sigma \times \tau \subset I \times I)$.

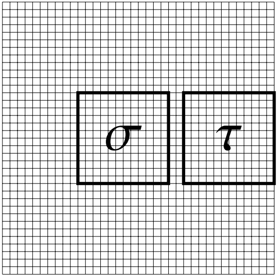


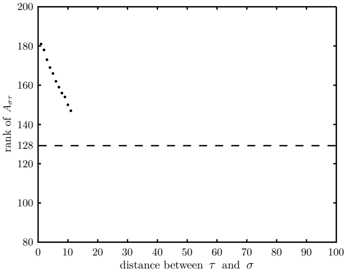
Admissibility condition

How to find a good clustering ?

⇒ Admissibility condition [Börm, Grasedyck, Hackbusch] expresses correlation between distance and rank:

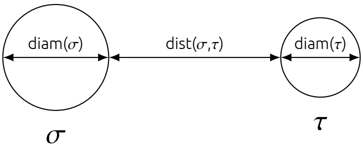


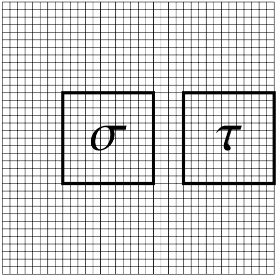


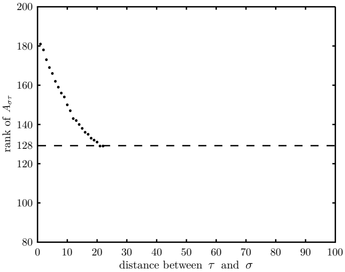


How to find a good clustering ?

⇒ Admissibility condition [Börm, Grasedyck, Hackbusch] expresses correlation between distance and rank:



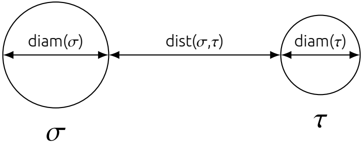


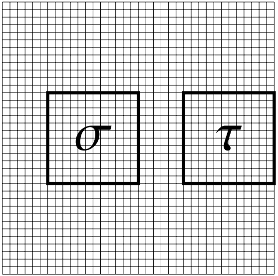


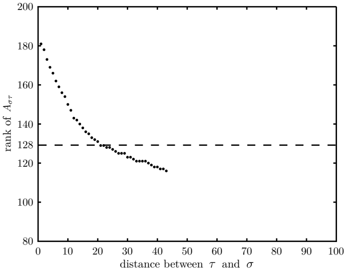
Admissibility condition

How to find a good clustering ?

⇒ Admissibility condition [Börm, Grasedyck, Hackbusch] expresses correlation between distance and rank:



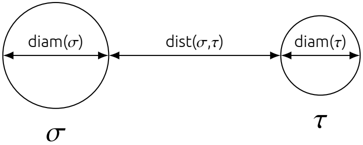


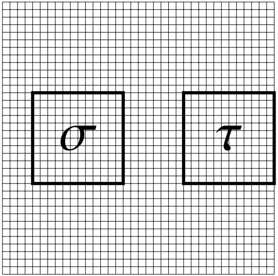


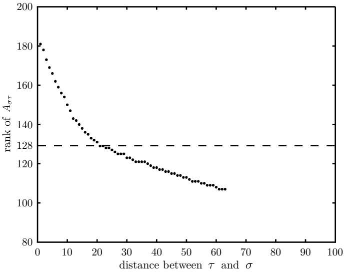
Admissibility condition

How to find a good clustering ?

⇒ Admissibility condition [Börm, Grasedyck, Hackbusch] expresses correlation between distance and rank:

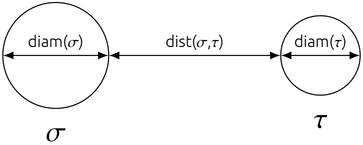


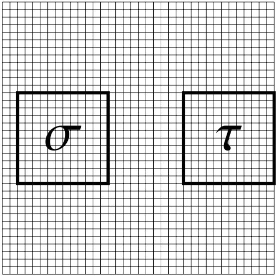


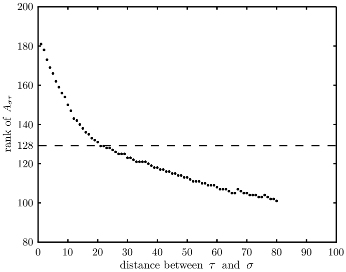


How to find a good clustering ?

⇒ Admissibility condition [Börm, Grasedyck, Hackbusch] expresses correlation between distance and rank:



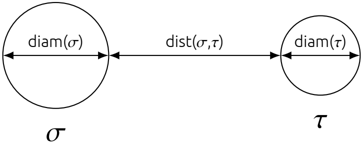


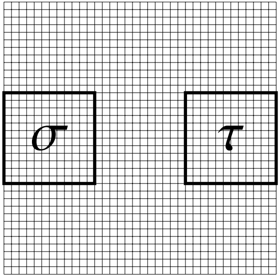


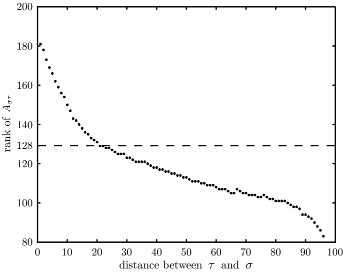
Admissibility condition

How to find a good clustering ?

⇒ Admissibility condition [Börm, Grasedyck, Hackbusch] expresses correlation between distance and rank:







Low-rank representations

- There are different low-rank representations (heuristics to exploit low-rank blocks): $\mathcal{H}, \mathcal{H}^2$ [Bebendof, Börm, Hackbush, Grasedyck,...], Hierarchical/Sequential Semiseparable (HSS/SSS) [Chandrasekaran, Dewilde, Gu, Li, Xia,...], BLR [Amestoy et al.], etc.
- Some representations are simpler and apply to broader classes of problems but provide less gain in memory/operations, while some others are more complex but allow for further gains in complexity.
- We focus on Block Low-Rank (BLR).

HSS vs BLR

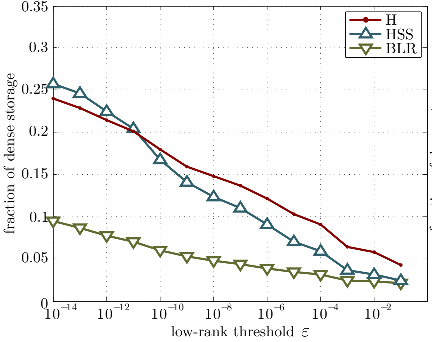
$$HSS = \left[\begin{array}{c|c|cc}
D_1 & X_1 B_1 Y_2^T & X_1 R_1 B_3 W_4^T Y_4^T & X_1 R_1 B_3 W_5^T Y_5^T \\
\cline{1-2}
X_2 B_2 Y_1^T & D_2 & X_2 R_2 B_3 W_4^T Y_4^T & X_2 R_2 B_3 W_5^T Y_5^T \\
\hline
X_4 R_4 B_6 W_1^T Y_1^T & X_4 R_4 B_6 W_2^T Y_2^T & D_4 & \multicolumn{1}{|c}{X_4 B_4 Y_5^T} \\
\cline{3-4}
X_5 R_5 B_6 W_1^T Y_1^T & X_5 R_5 B_6 W_2^T Y_2^T & X_5 B_5 Y_4^T & \multicolumn{1}{|c}{D_5}
\end{array}\right]$$

$$BLR = \left[\begin{array}{c|c|c|c}
D_1 & X_{12} Y_{12}^T & X_{13} Y_{13}^T & X_{14} Y_{14}^T \\
\hline
X_{21} Y_{21}^T & D_2 & X_{23} Y_{23}^T & X_{24} Y_{24}^T \\
\hline
X_{31} Y_{31}^T & X_{32} Y_{32}^T & D_3 & X_{34} Y_{34}^T \\
\hline
X_{41} Y_{41}^T & X_{42} Y_{42}^T & X_{43} Y_{43}^T & D_4
\end{array}\right]$$

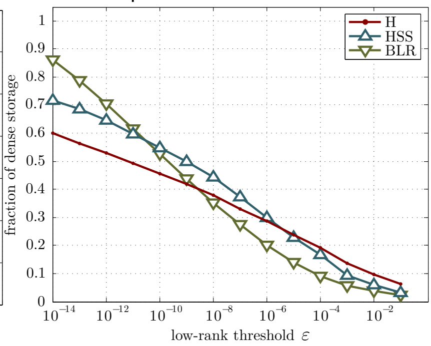
$\Rightarrow$ particular case of $\mathcal{H}$-matrices
$\Rightarrow$ no tree
$\Rightarrow$ natural matrix structure

Compression rates of the frontal matrix at the root of a multifrontal tree, on two 3D stencils (discretization of a 128 x 128 x 128 cube)

Laplacian

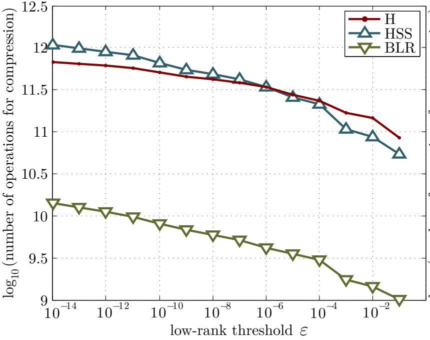


27-pts Geoazur stencil

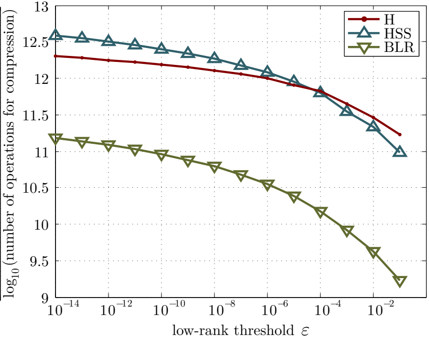


Compression cost of the frontal matrix at the root of a multifrontal tree, on two 3D stencils (discretization of a 128 x 128 x 128 cube)

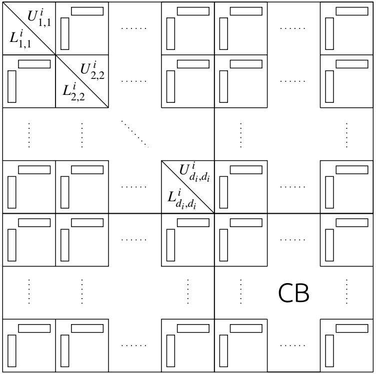
Laplacian



27-pts Geoazur stencil

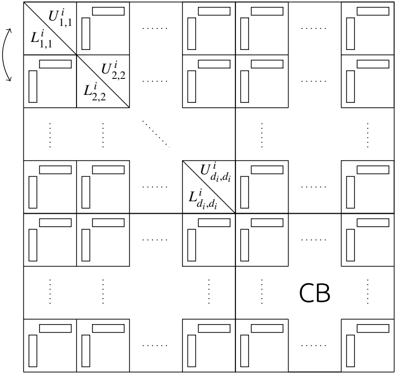


Adaptation to a multifrontal solver



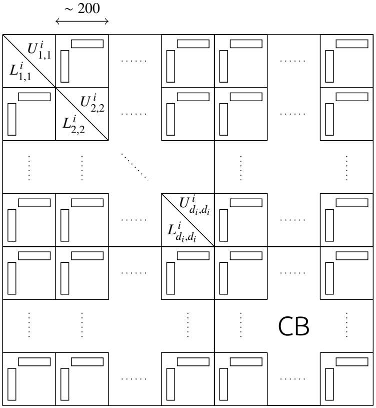
Adaptation to a multifrontal solver

no relative order → efficient clustering

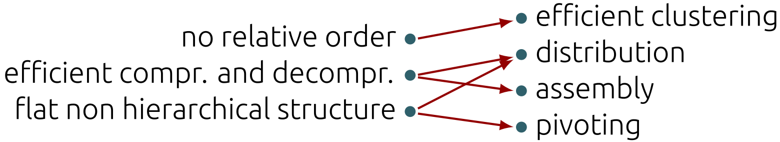


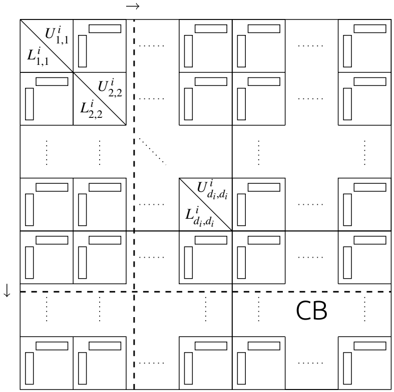
Adaptation to a multifrontal solver

- no relative order → efficient clustering
- efficient compr. and decompr. → distribution, assembly



Adaptation to a multifrontal solver

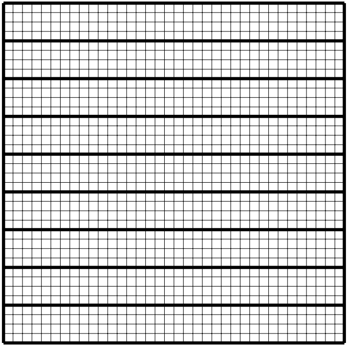




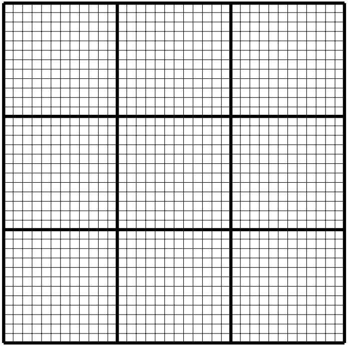
Clustering variables

Admissible clustering

Constraint : the admissibility condition should be satisfied



large diameters
fraction of memory used 83%



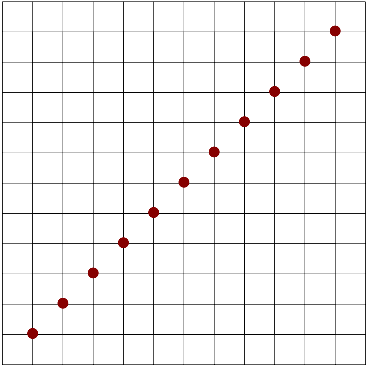
small diameters
fraction of memory used 57%

Halo algorithm for the clustering of a separator

- Designed to catch the geometry of the problem
- Computed with the graph instead of the mesh
- Coupled with a third party partitioning tool

Halo algorithm for the clustering of a separator

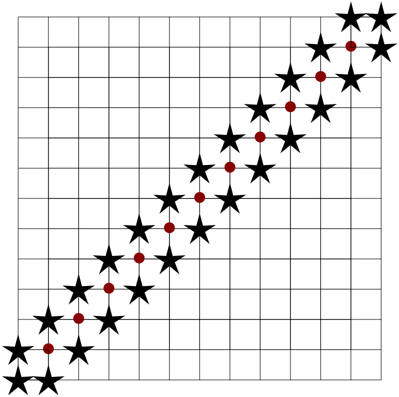
- Designed to catch the geometry of the problem
- Computed with the graph instead of the mesh
- Coupled with a third party partitioning tool



1. The separator

Halo algorithm for the clustering of a separator

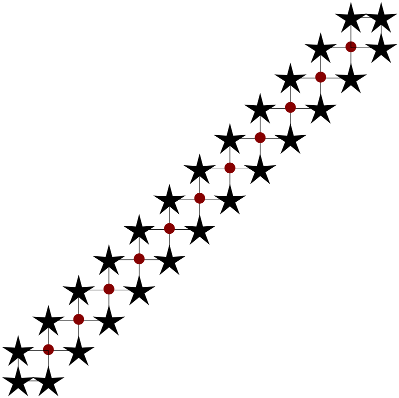
- Designed to catch the geometry of the problem
- Computed with the graph instead of the mesh
- Coupled with a third party partitioning tool



1. The separator
2. The halo

Halo algorithm for the clustering of a separator

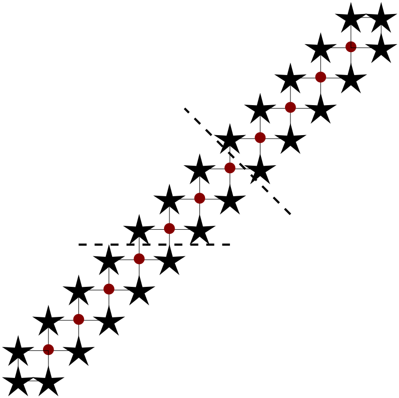
- Designed to catch the geometry of the problem
- Computed with the graph instead of the mesh
- Coupled with a third party partitioning tool



1. The separator
2. The halo
3. Extraction of the halo

Halo algorithm for the clustering of a separator

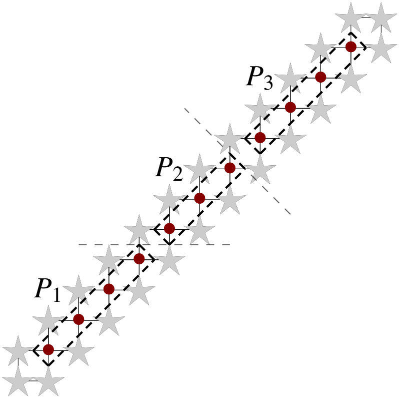
- Designed to catch the geometry of the problem
- Computed with the graph instead of the mesh
- Coupled with a third party partitioning tool



1. The separator
2. The halo
3. Extraction of the halo
4. Partition of the halo

Halo algorithm for the clustering of a separator

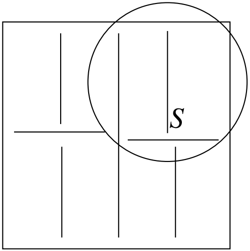
- Designed to catch the geometry of the problem
- Computed with the graph instead of the mesh
- Coupled with a third party partitioning tool



1. The separator
2. The halo
3. Extraction of the halo
4. Partition of the halo
5. Partition of the separator (block size is fixed)

Clustering of the variables of a front

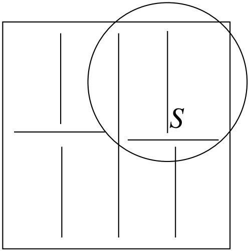
⇒ front = separator + border



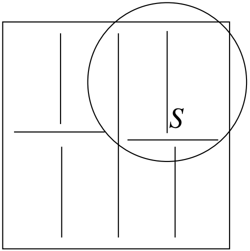
Clustering of the variables of a front

⇒ front = separator + border

1- separator : halo

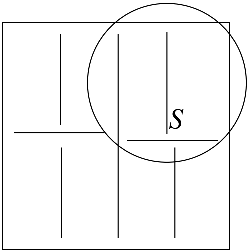


Clustering of the variables of a front



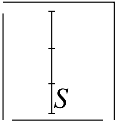
- $\Rightarrow$ front = separator + border
- 1- separator : halo
- 2- border ? 2 choices :

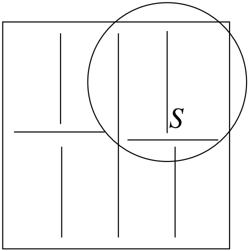
Clustering of the variables of a front



- ⇒ front = separator + border
- 1- separator : halo
- 2- border ? 2 choices :

EXPLICIT



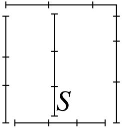


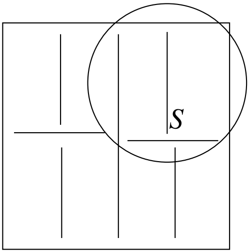
⇒ front = separator + border

1- separator : halo

2- border ? 2 choices :

EXPLICIT



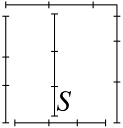


⇒ front = separator + border

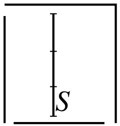
1- separator : halo

2- border ? 2 choices :

EXPLICIT



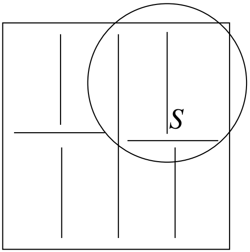
INHERITED (*top down*)



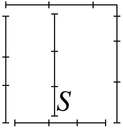
⇒ front = separator + border

1- separator : halo

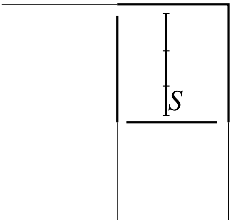
2- border ? 2 choices :



EXPLICIT

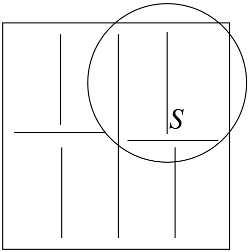


INHERITED (*top down*)

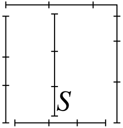


Clustering of the variables of a front

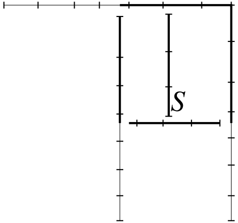
- ⇒ front = separator + border
- 1- separator : halo
- 2- border ? 2 choices :



EXPLICIT



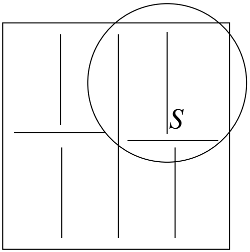
INHERITED (*top down*)



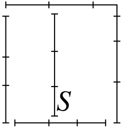
⇒ front = separator + border

1- separator : halo

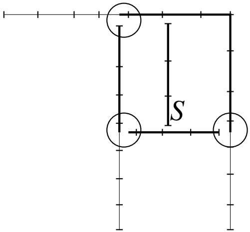
2- border ? 2 choices :



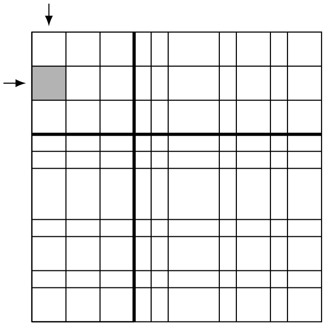
EXPLICIT



INHERITED (*top down*)

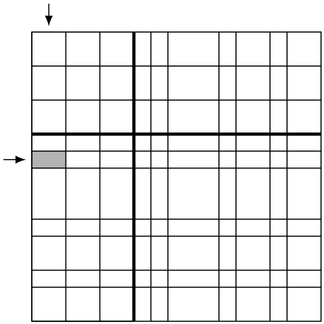


Front structure with inherited clustering



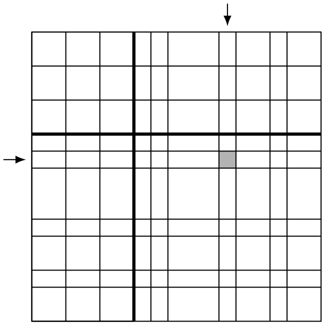
- optimal $\times$ optimal = optimal block

Front structure with inherited clustering



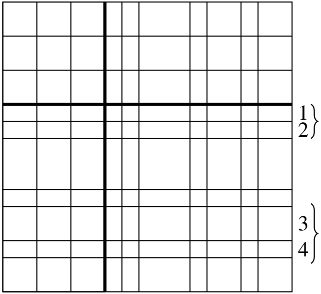
- optimal × optimal = optimal block
- small × optimal = large enough block

Front structure with inherited clustering

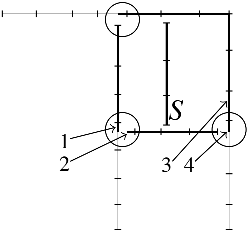


- optimal × optimal = optimal block
- small × optimal = large enough block
- small × small = too small block

Front structure with inherited clustering



- optimal × optimal = optimal block
- small × optimal = large enough block
- small × small = too small block



⇒ reclustering strategies

Block Low-Rank multifrontal method

Factorization algorithms

task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

Factorization algorithms

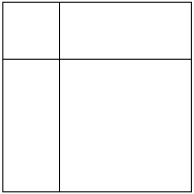
task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

Standard FR algorithm:

F Factor

S Solve

U Update



Factorization algorithms

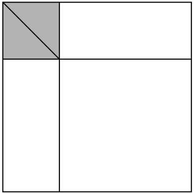
task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

Standard FR algorithm:

F Factor

S Solve

U Update



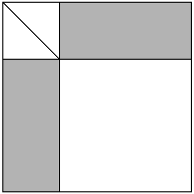
task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

Standard FR algorithm:

F Factor

S Solve

U Update



Factorization algorithms

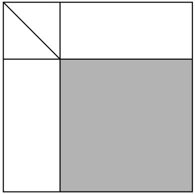
task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

Standard FR algorithm:

F Factor

S Solve

U Update



Factorization algorithms

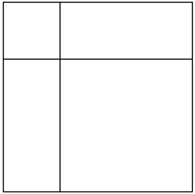
task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

BLR FS**C**U algorithm:

F Factor

S Solve

U Update



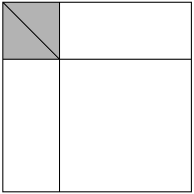
task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

BLR FS**C**U algorithm:

F Factor

S Solve

U Update



Factorization algorithms

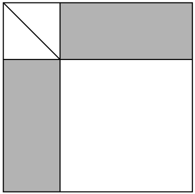
task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

BLR FS**C**U algorithm:

F Factor

S Solve

U Update

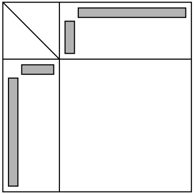


task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

BLR FS**C**U algorithm:

F Factor

S Solve
C Compress
U Update

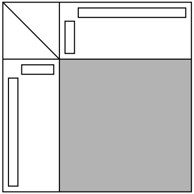


task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

BLR FS**C**U algorithm:

F Factor

S Solve
C Compress
U Update



Factorization algorithms

task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

F Factor

S Solve

C Compress

U Update

FCSU version
more efficient
less stability

Factorization algorithms

task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

F Factor

S Solve

C Compress

U Update

FSUC version
no flop reduction
more stability

Factorization algorithms

task	operation type	dense	low-rank
Factor (F)	$B = LU^T$	$(2/3)n^3$	$(2/3)n^3$
Compress (C)	$C = XY^T$	kn^2	—
Solve (S)	$D = X(Y^T L^{-1})$	n^3	kn^2
CB update (U)	$D = D - X_1(Y_1^T X_2)Y_2^T$	$2n^3$	$2kn^2$

F Factor

S Solve
C Compress
U Update

High algorithmic flexibility

Experiments

Set of problems

Name	Prop.	Arith.	N ($\times 10^6$)	NZ ($\times 10^9$)	mem. LU (GB)	flops LU ($\times 10^{12}$)	CSR	appli.
`Curl` 5000^2	2D/sym.	D	50	0.2	29	5	2E-15	∇
`Geoazur` 128^3	3D/unsym.	Z	2	55	54	60	3E-4	wave prop.
`EDF_A_MECA_R12`	2D/sym.	D	134	1	151	200	4E-15	mechanics
`EDF_D_THER_R7`	3D/sym	D	8	118	229	100	8E-15	thermics

- CSR = Componentwise Scaled Residual
- Code_Aster `tpl101{a,d}` test cases (refined)
- large matrices
- applicative problems

Metrics

memory:

- |L| = fraction of FR factors storage obtained with BLR (%)
- |CB| = fraction of FR maximum size of CB stack obtained with BLR (%)

flops:

- fraction of FR operations needed for the BLR factorization (in percent or absolute data, including the compression cost)

Clustering strategy

	memory				flops		time		
	$\lvert L \rvert$		$\lvert CB \rvert$		facto		clustering		full rank analysis
clustering	inh	exp	inh	exp	inh	exp	inh	exp	
`Curl5000`	63.7%	62.4%	7.0%	5.5%	10.9%	11.1%	13 s	27 s	897 s
`Geoazur128`	79.0%	77.0%	47.0%	45.0%	60.8%	59.1%	5 s	42 s	62 s
`TH_RAFF7`	34.1%	30.7%	17.5%	16.2%	7.2%	6.6%	34 s	206 s	387 s
`ME_RAFF12`	52.9%	51.1%	4.8%	4.1%	6.1%	6.1%	121 s	239 s	1971 s

- “inherited” version is more than 2 times faster
- same results on L_{11}
- comparable results on L_{21}
- a little less good on CBs

$\Rightarrow$ **inherited** clustering used for all the experiments

Results with different orderings on `Geoazur128` problem.

	FR			LR		
ordering	mry	flops	peak	|L|	|CB|	flops
AMD	109GB	$3.9E+14$	92GB	73.5%	40.0%	59.4%
AMF	72GB	$1.8E+14$	45GB	69.9%	53.5%	48.3%
PORD	55GB	$1.0E+14$	28GB	70.4%	49.0%	48.9%
METIS	46GB	$6.2E+13$	20GB	78.7%	46.4%	62.6%
SCOTCH	49GB	$6.6E+13$	21GB	79.4%	48.1%	63.4%

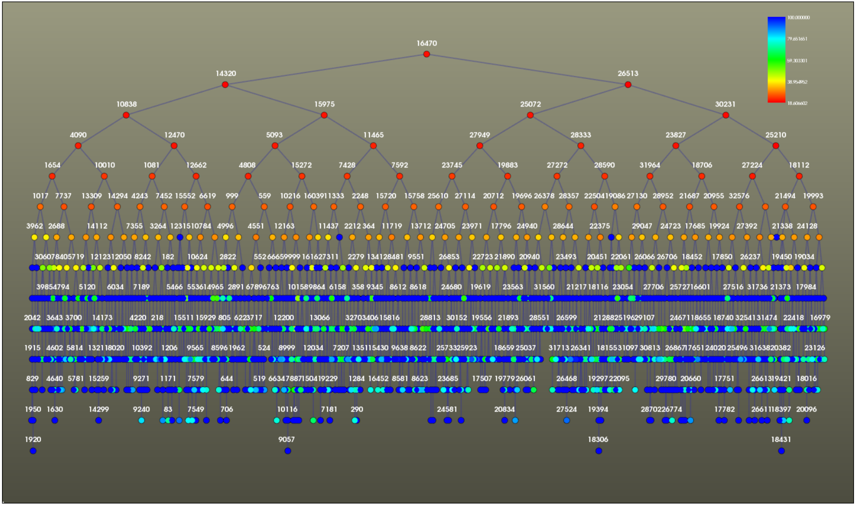
Global ordering of the matrix: general results

Results with different orderings on `Geoazur128` problem.

	FR			LR		
ordering	mry	flops	peak	|L|	|CB|	flops
AMD	109GB	$3.9E+14$	92GB	73.5%	40.0%	59.4%
AMF	72GB	$1.8E+14$	45GB	69.9%	53.5%	48.3%
PORD	55GB	$1.0E+14$	28GB	70.4%	49.0%	48.9%
METIS	46GB	$6.2E+13$	20GB	78.7%	46.4%	62.6%
SCOTCH	49GB	$6.6E+13$	21GB	79.4%	48.1%	63.4%

Algebraic approach

[visualization tool developed by M. Bremond]



Influence of the size of the problem

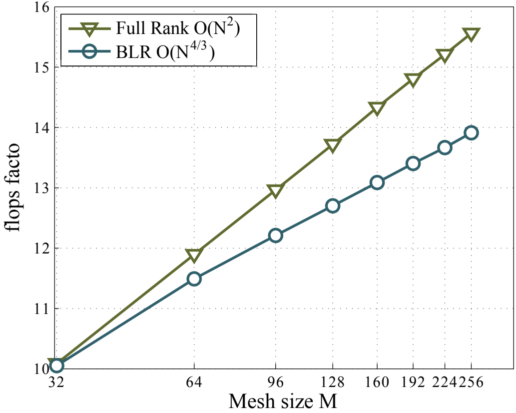
⇒ different refinements of problem `EDF_A_MECA` done with Homard

		memory		flops
Refinement	N	|L|	|CB|	facto
R8	$2,101,258$	71%	43%	37%
R11	$33,570,826$	59%	29%	12%
R12	$134,250,506$	53%	24%	7%

- the larger the problem, the more efficient the method
- target = challenging problems

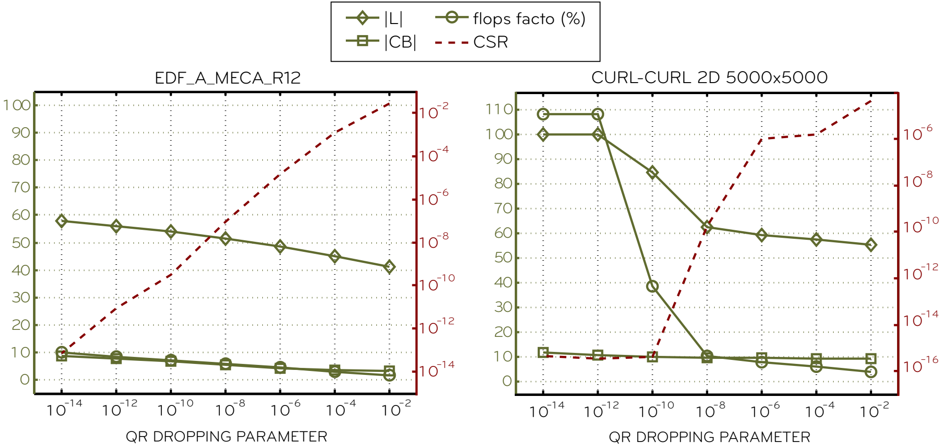
Scalability with respect to the size of the problem

⇒ Laplacian problem, $\varepsilon = 10^{-14}$



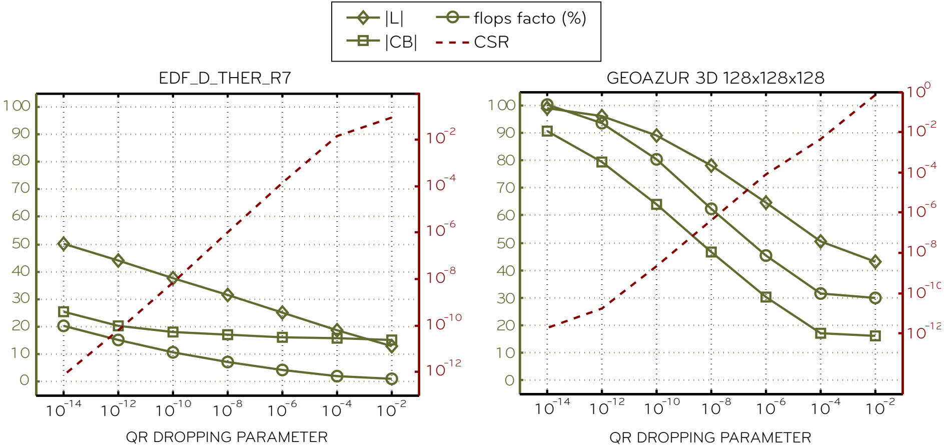
⇒ $\mathcal{O}(N^{4/3})$ complexity (≡ HSS)

Global results: 2D problems



Name	N	mem LU	flops LU	CSR
`EDF_A_MECA_R12`	134E+6	151 GB	2E+14	4E-15
`Curl-Curl` 5000^2	50E+6	29 GB	5E+12	2E-15

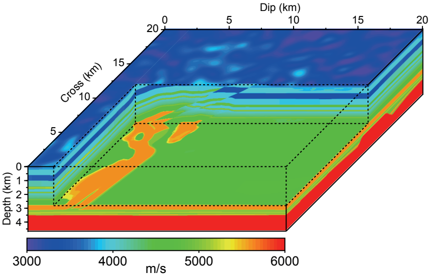
Global results: 3D problems



Name	N	mem LU	flops LU	CSR
EDF_D_THER_R7	8E+6	229 GB	1E+14	8E-15
Geoazur 128^3	2E+6	54 GB	6E+13	3E-4

Application to geophysics (1)

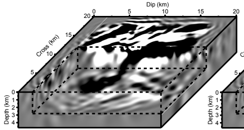
- Helmholtz equation for seismic modeling (SEISCOPE project)
- EAGE overthrust ground model
- single precision computations



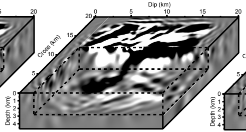
fqcy	Flops LU	Mem LU	Peak memory
2 Hz	8.957E+11	3 GB	4 GB
4 Hz	1.639E+13	22 GB	25 GB
8 Hz	5.769E+14	247 GB	283 GB

Application to geophysics (2)

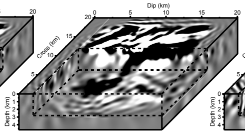
		flops	memory	
ε	fqcy	facto	|L|	|CB|
(10^{-5})	2 Hz	41.8 %	61.8 %	32.3%
	4 Hz	27.4 %	50.0 %	24.4%
	8 Hz	21.8 %	41.6 %	23.9%
(10^{-4})	2 Hz	32.9 %	53.4 %	23.9%
	4 Hz	20.0 %	42.2 %	21.7%
	8 Hz	15.2 %	28.9 %	19.4%
(10^{-3})	2 Hz	24.6 %	44.7 %	16.8%
	4 Hz	13.8 %	34.5 %	19.0%
	8 Hz	9.8 %	21.3 %	15.9%



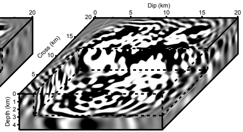
FR



10^{-5}

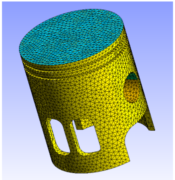
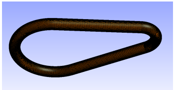
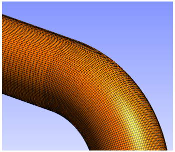


10^{-4}



10^{-3}

Preconditioning with BLR: set of problems

	N	NZ	Cond.	application
`Piston`	1.3E+6	54.7E+6	5.1E+5	external pressure force on the top
`perf001d`	2.0E+6	75.8E+6	1.5E+11	“cavity” hook subjected to internal pressure force (challenging for EDF)

- CG preconditioned with `MUMPS` single precision with BLR
- preliminary study

Preconditioning with BLR: results

`perf001d`, FR SP = 50 iterations

	BLR SP				BLR DP			
ε	#it	|L|	|CB|	flops	#it	|L|	|CB|	flops
10^{-10}	–	–	–	–	2	64.2%	18.3%	31.0%
10^{-9}	–	–	–	–	2	62.2%	16.1%	28.8%
10^{-8}	67	59.4%	18.9%	26.7%	3	58.7%	13.7%	25.5%
10^{-7}	68	56.9%	16.9%	24.3%	4	56.4%	11.4%	23.4%
10^{-6}	66	52.4%	15.2%	20.2%	8	51.9%	9.6 %	19.7%
10^{-5}	67	49.1%	14.1%	17.1%	19	48.6%	8.6 %	17.0%
10^{-4}	81	45.1%	13.5%	14.1%	68	44.4%	8.0 %	14.1%

Preconditioning with BLR: results

`perf001d`, FR SP = 50 iterations

ε	BLR SP #it	BLR SP |L|	BLR SP |CB|	BLR SP flops	BLR DP #it	BLR DP |L|	BLR DP |CB|	BLR DP flops
10^{-10}	–	–	–	–	2	64.2%	18.3%	31.0%
10^{-9}	–	–	–	–	2	62.2%	16.1%	28.8%
10^{-8}	67	59.4%	18.9%	26.7%	3	58.7%	13.7%	25.5%
10^{-7}	68	56.9%	16.9%	24.3%	4	56.4%	11.4%	23.4%
10^{-6}	66	52.4%	15.2%	20.2%	8	51.9%	9.6 %	19.7%
10^{-5}	67	49.1%	14.1%	17.1%	19	48.6%	8.6 %	17.0%
10^{-4}	81	45.1%	13.5%	14.1%	68	44.4%	8.0 %	14.1%

Preconditioning with BLR: results

perf001d, FR SP = 50 iterations

ε	BLR SP				BLR DP			
	#it	|L|	|CB|	flops	#it	|L|	|CB|	flops
10^{-10}	–	–	–	–	2	64.2%	18.3%	31.0%
10^{-9}	–	–	–	–	2	62.2%	16.1%	28.8%
10^{-8}	67	59.4%	18.9%	26.7%	3	58.7%	13.7%	25.5%
10^{-7}	68	56.9%	16.9%	24.3%	4	56.4%	11.4%	23.4%
10^{-6}	66	52.4%	15.2%	20.2%	8	51.9%	9.6 %	19.7%
10^{-5}	67	49.1%	14.1%	17.1%	19	48.6%	8.6 %	17.0%
10^{-4}	81	45.1%	13.5%	14.1%	68	44.4%	8.0 %	14.1%

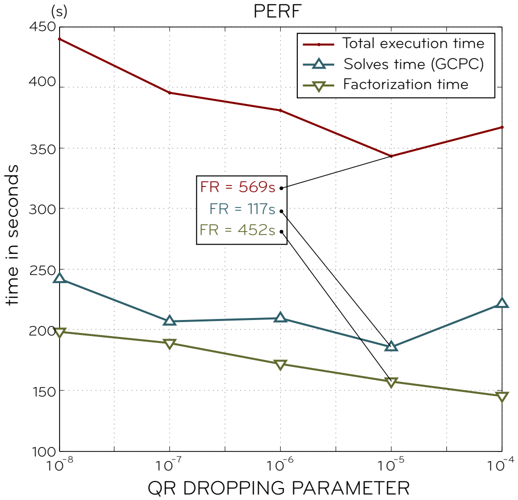
Preconditioning with BLR: results

`perf001d`, FR SP = 50 iterations

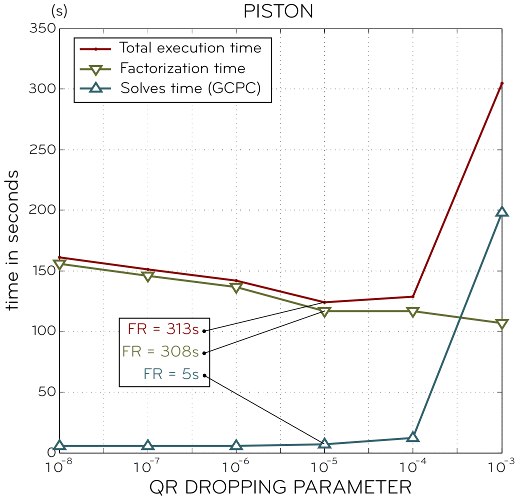
ε	BLR SP #it	BLR SP |L|	BLR SP |CB|	BLR SP flops	BLR DP #it	BLR DP |L|	BLR DP |CB|	BLR DP flops
10^{-10}	–	–	–	–	2	64.2%	18.3%	31.0%
10^{-9}	–	–	–	–	2	62.2%	16.1%	28.8%
10^{-8}	67	59.4%	18.9%	26.7%	3	58.7%	13.7%	25.5%
10^{-7}	68	56.9%	16.9%	24.3%	4	56.4%	11.4%	23.4%
10^{-6}	66	52.4%	15.2%	20.2%	8	51.9%	9.6 %	19.7%
10^{-5}	67	49.1%	14.1%	17.1%	19	48.6%	8.6 %	17.0%
10^{-4}	81	45.1%	13.5%	14.1%	68	44.4%	8.0 %	14.1%

- is it the right definition of *optimality* ?

Preconditioning with BLR: `perf001d` timings



Preconditioning with BLR: `Piston` timings



	FR	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}
# iterations	3	3	3	3	4	7	115	—

Conclusion & perspectives

- efficient method on various applicative problems
- considerable memory reduction & substantial decrease in computations $\Rightarrow \mathcal{O}(N^{4/3})$ complexity on a Laplacian, comparable to HSS
- can be used as a preconditioner or as a direct solver
- good potential for parallelism
- code is stable on tested problems

- MPI
- study on larger and more difficult problems
- error propagation study $\Rightarrow$ absolute or relative dropping parameter ? relative to what ?
  (work with S. Gratton, M. Ngom and D. Titley-Peloquin started)

Aknowledgements

- **O. Boiteau**, **B. Quinnez** and **N. Tardieu** (EDF R&D)
- **S. Operto**, **R. Brossier** and **J. Virieux** (SEISCOPE Project) for their contribution to the geophysics study
- the **Toulouse Computing Center** (CICT) and **N. Renon**
- **S. Li**, **A. Napov** and **F.-H. Rouet** (LBNL Berkeley)

Details on this work can be found 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, IRIT technical report RT/APO/12/6, INRIA technical report INRIA/RR-8199, submitted to SIAM SISC, `http://weisbecker.perso.enseeiht.fr/documents/RT_APO_12_6_BLR.pdf`, 2012.

Thank you !
Any questions?