11 Dec. 2014, LJLL, Paris

FreeFem++ workshop

A dissection solver with kernel detection for unsymmetric matrices in FreeFem++

Atsushi Suzuki[†]

Atsushi.Suzuki@ann.jussieu.fr

Joint work with

François-Xavier Roux[†,§]

LJLL-UPMC[†], ONERA[§],

Sparse direct solvers

Software	parallel environment	elimination strategy	data management and pivoting	kernel detection
`SuperLU_MT`	shared	super-nodal	dynamic data + pivoting	no
`Pardiso`	shared	super-nodal	dynamic data + $\sqrt{\varepsilon}$-perturbation + pivoting	no
`SuperLU_DIST`	distributed	super-nodal	static data + $\sqrt{\varepsilon}$-perturbation	none
`MUMPS`	distributed	multi-frontal	dynamic data + pivoting	yes

J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, J. W. H. Liu. A supernodal approach to sparse partial pivoting, *SIAM Journal on Matrix Analysis and Applications*, 20 (1999), 720–755.

O. Schenk, K. Gärtner. Solving unsymmetric sparse systems of liner equations with PARDISO, *Future Generation of Computer Systems*, 20 (2004), 475–487.

X. S. Li, J. W. Demmel. SuperLU_DIST : A scalable distributed-memory sparse direct solver for unsymmetric linear systems, *ACM Transactions on Mathematical Software*, 29 (2003), 110–140.

P. R. Amestoy, I. S. Duff, J.-Y. L'Execellent. Mutlifrontal parallel distributed symmetric and unsymmetric solvers, *Computer Methods in Applied Mechanics and Engineering*, 184 (2000) 501-520.

Sparse direct solvers

Software	parallel environment	elimination strategy	data management and pivoting	kernel detection
`SuperLU_MT`	shared	super-nodal	dynamic data + pivoting	no
`Pardiso`	shared	super-nodal	dynamic data + $\sqrt{\varepsilon}$-perturbation + pivoting	no
`SuperLU_DIST`	distributed	super-nodal	static data + $\sqrt{\varepsilon}$-perturbation	none
`MUMPS`	distributed	multi-frontal	dynamic data + pivoting	yes
`Dissection`	shared	multi-frontal	static data + pivoting	yes

J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, J. W. H. Liu. A supernodal approach to sparse partial pivoting, *SIAM Journal on Matrix Analysis and Applications*, 20 (1999), 720–755.

O. Schenk, K. Gärtner. Solving unsymmetric sparse systems of liner equations with PARDISO, *Future Generation of Computer Systems*, 20 (2004), 475–487.

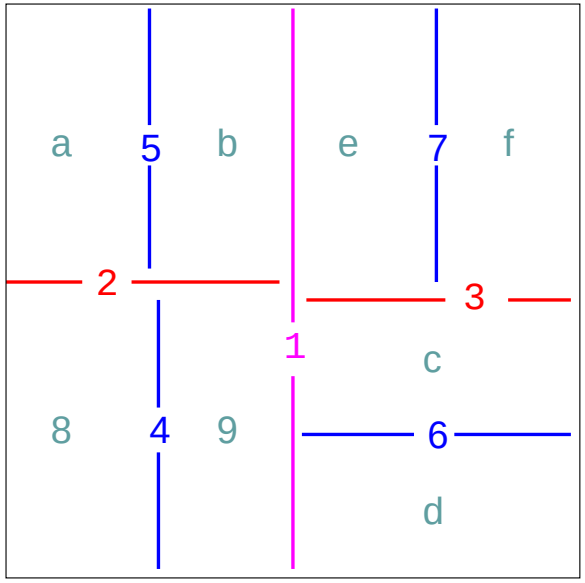
X. S. Li, J. W. Demmel. SuperLU_DIST : A scalable distributed-memory sparse direct solver for unsymmetric linear systems, *ACM Transactions on Mathematical Software*, 29 (2003), 110–140.

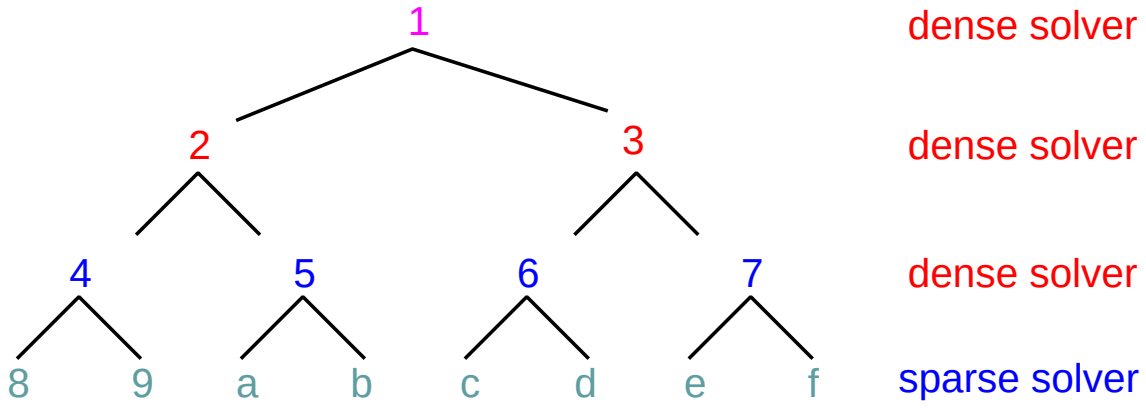
P. R. Amestoy, I. S. Duff, J.-Y. L'Execellent. Mutlifrontal parallel distributed symmetric and unsymmetric solvers, *Computer Methods in Applied Mechanics and Engineering*, 184 (2000) 501-520.

A. Suzuki, F.-X. Roux, A dissection solver with kernel detection for symmetric finite element matrices on shared memory computers, *International Journal for Numerical Methods in Engineering*, 100 (2014) 136-164.

Overview of `Dissection` solver

`Dissection` based on graph decomposition of sparse matrix





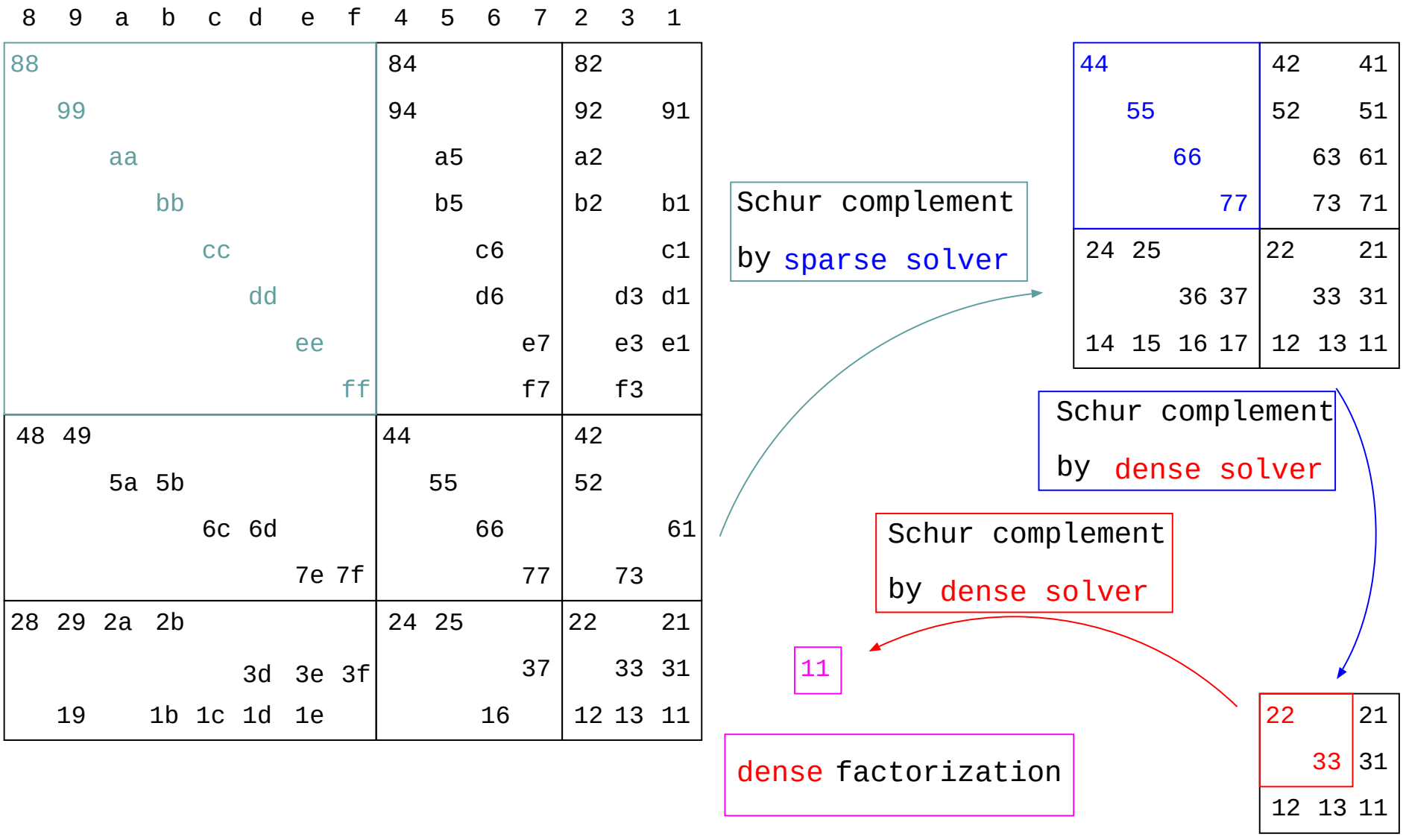
sparse solver (`tridiag`) is written with Fortran90 by F.-X. Roux

$\Rightarrow$ each leaf can be computed in parallel

? load unbalance ? $\Leftarrow$ different sizes of subdomains

? parallel computation of higher levels : smaller subdomains with many cores/processors ?

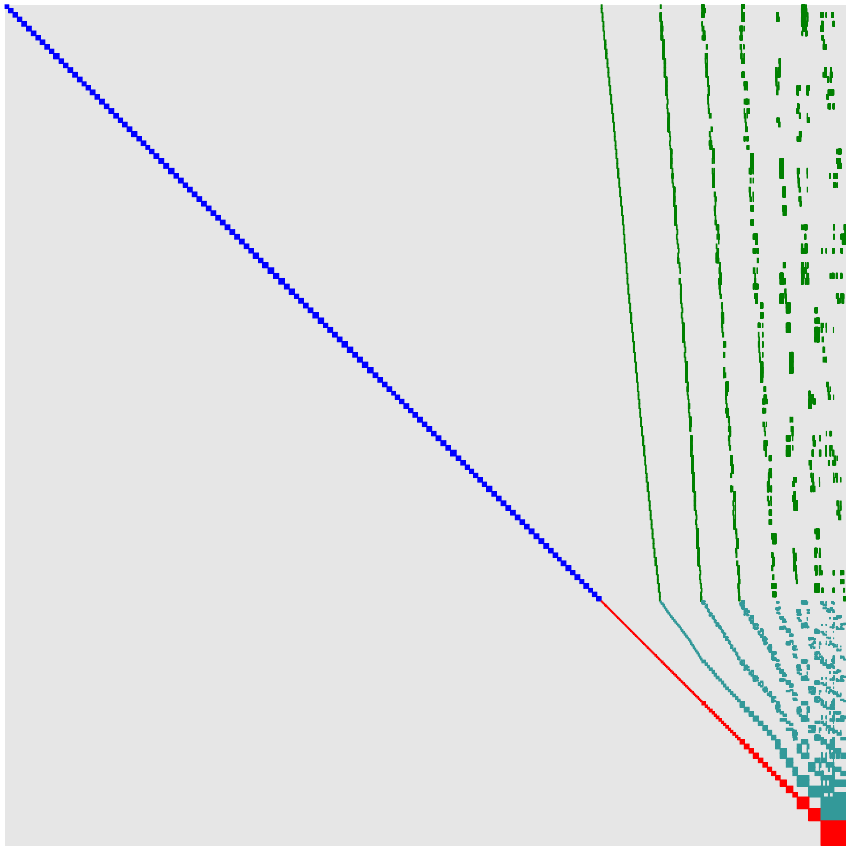
Overview : recursive generation of Schur complement



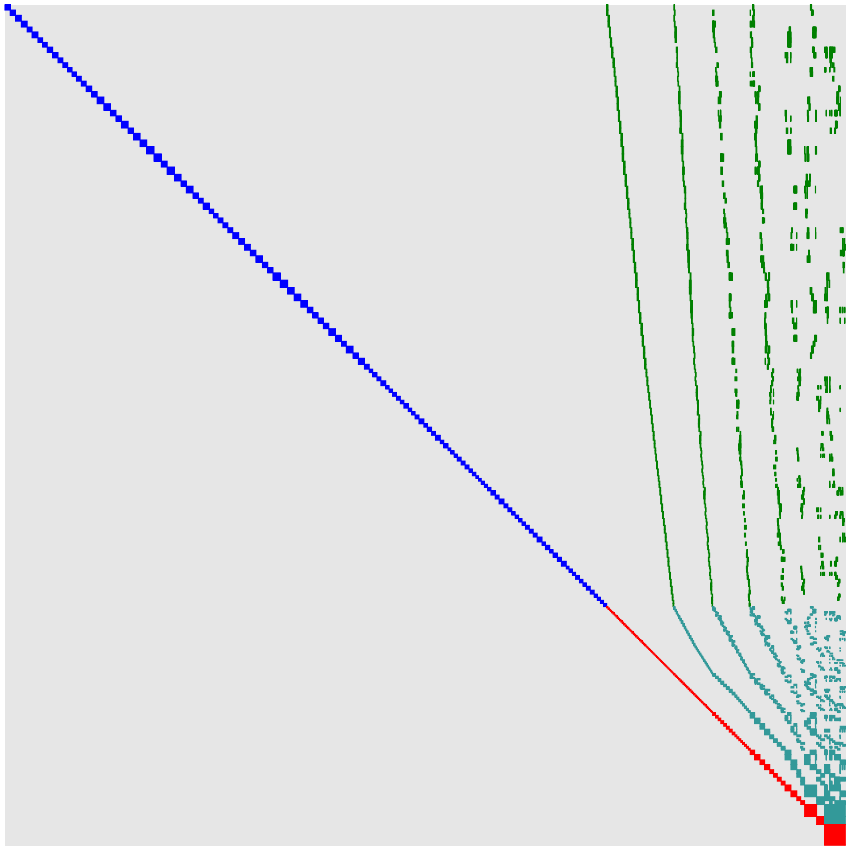
there is no 63 block in sparse matrix. 63 in Schur complement has values as result of fill-in. This is analyzed in “symbolic phase”.

Overview : graph decomposition by METIS/SCOTCH

matrix size = 206,763, bisection level = 8, sub-blocks = 255



METIS



SCOTCH

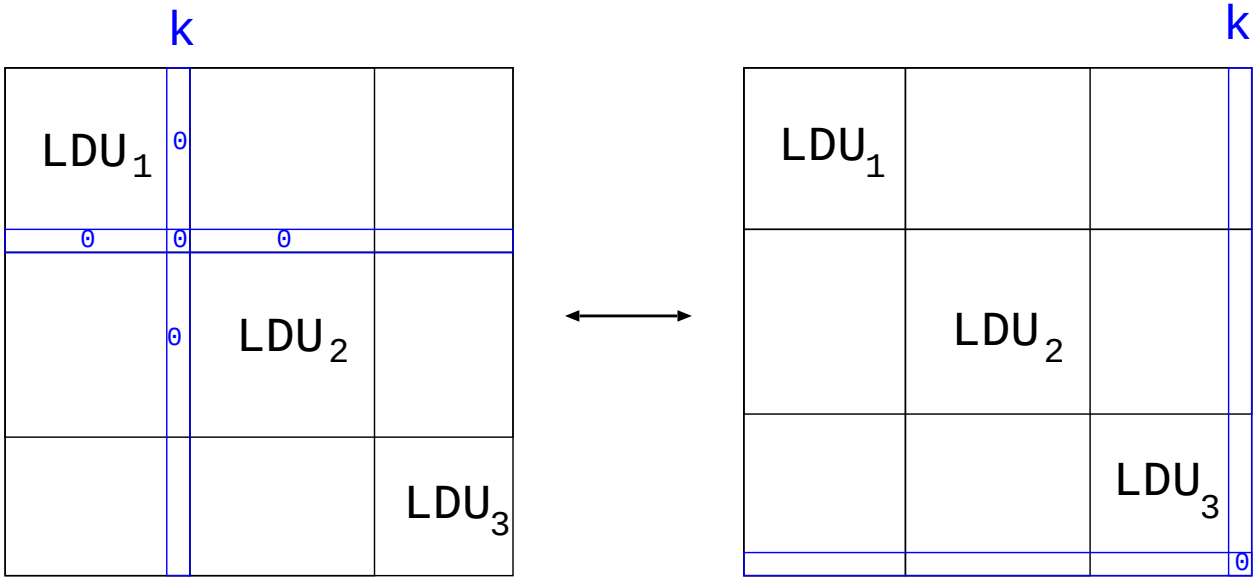
number of nonzeros in dense blocks

	METIS	SCOTCH
diag	82,659,783	66,956,787
off-diag	183,097,704	156,858,381
total	265,739,487	223,815,168

Block factorization and block pivot strategy

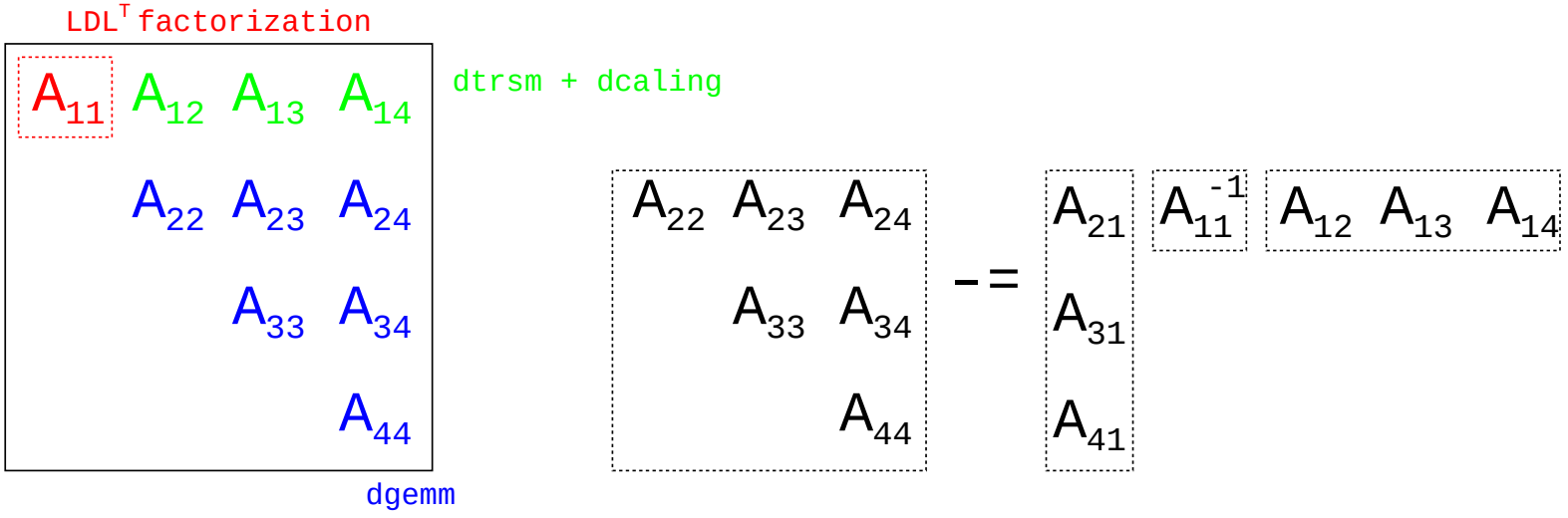
block pivot strategy is employed for parallel efficiency

$$\begin{bmatrix} \Pi_1^T & & \\ & \Pi_2^T & \\ & & \Pi_3^T \end{bmatrix} \begin{bmatrix} L_{1\,1} & & \\ L_{2\,1} & L_{2\,2} & \\ L_{3\,1} & L_{3\,2} & L_{3\,3} \end{bmatrix} \begin{bmatrix} D_1 & & \\ & D_2 & \\ & & D_3 \end{bmatrix} \begin{bmatrix} U_{1\,1} & U_{1\,2} & U_{1\,3} \\ & U_{2\,2} & U_{2\,3} \\ & & U_{3\,3} \end{bmatrix} \begin{bmatrix} \Pi_1 & & \\ & \Pi_2 & \\ & & \Pi_3 \end{bmatrix}$$



When null-pivots appear in a block, they are sent to the last, and corresponding rows and columns are nullified for rank $(n-k)$ update.

Dissection solver : parallelization of dense blocks



$\alpha^{(1)}$: factorization $L_{11} D_{11} L_{11}^T = A_{11}^{(1)}$

$\beta_k^{(1)}$: DTRSM + scaling $D_{11}^{-1}(L_{11}^{-1} A_{1\,k}^{(1)})$

$\gamma_{k,\,l}^{(1)}$: DGEMM $A_{k,\,l}$ -= $(L_{11}^{-1} A_{1\,k})^T D_{11}^{-1}(L_{11}^{-1} A_{1\,l}^{(1)})$

$\alpha^{(2)}$: factorization $L_{22} D_{22} L_{22}^T = A_{22}^{(2)}$

$\alpha^{(1)} \leftarrow \{\beta_2^{(1)}, \beta_3^{(1)}, \beta_4^{(1)}\} \leftarrow \{\gamma_{2,2}^{(1)}, \gamma_{2,3}^{(1)}, \gamma_{3,3}^{(1)}, \cdots, \gamma_{4,4}^{(1)}\} \leftarrow \alpha^{(2)} \leftarrow \{\beta_3^{(2)}, \beta_4^{(2)}\} \leftarrow \{\gamma_{3,3}^{(2)}, \gamma_{3,4}^{(2)}, \gamma_{4,4}^{(2)}\} \leftarrow$

- updating of Schur complement depends on factorization $\Rightarrow$ sequential among levels

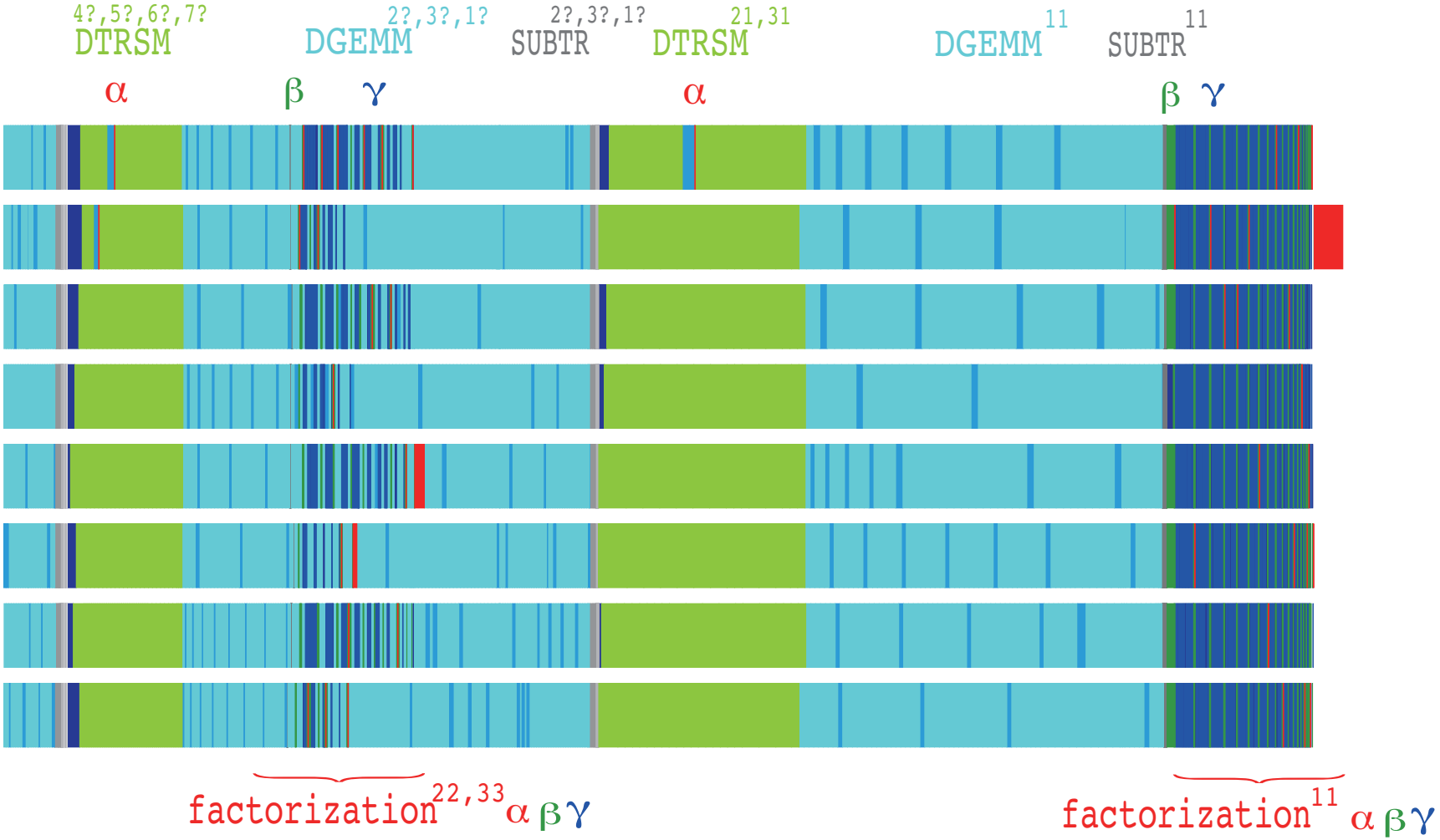
$\alpha^{(1)} \leftarrow \{\beta_2^{(1)}\text{-}\gamma_{2,2}^{(1)}\text{-}\alpha^{(2)}, \beta_3^{(1)}, \beta_4^{(1)}\} \leftarrow \{\gamma_{2,3}^{(1)}, \gamma_{3,3}^{(1)}, \cdots, \gamma_{4,4}^{(1)}\} \leftarrow \{\beta_3^{(2)}\text{-}\gamma_{3,3}^{(2)}\text{-}\alpha^{(3)}, \beta_4^{(2)}\} \leftarrow \{\gamma_{3,4}^{(2)}, \gamma_{4,4}^{(2)}\} \leftarrow$

+ factorization of diagonal block // updating of rest of Schur complement

Task scheduling in `Dissection` solver

$n = 206,763$

Westmere Xeon 5680@3.3GHz, 6core x2, POSIX threads



computation of Schur complement in level 3, factorization in level 2 : scheduled together.

tasks on critical path : scheduled statically

other tasks : executed in greedy way with large group using predicted complexity + fine greedy

Performance comparison

N=1,004,784 symmetric real matrix : Xeon7550 @ 2.0GHz

core	Dissection CPU	Dissection elapsed	/	IntelPardiso CPU	IntelPardiso elapsed	/	MUMPS + paraBLAS CPU	MUMPS + paraBLAS elapsed	/
1	5,607.5	5,607.7	—	5,430.6	5,431.1	—	5,894.4	5,894.9	—
2	5,634.7	2,827.5	1.98	5,676.6	2,838.7	1.92	6,547.5	3,369.3	1.75
4	5,668.4	1,437.9	3.90	6,403.9	1,601.1	3.39	7,457.8	2,003.4	2.94
8	5,784.7	746.7	7.51	6,817.3	852.4	6.37	10,925.2	1,533.5	3.84
12	5,880.5	525.7	10.67	7,049.4	587.9	9.24	14,108.5	1,351.5	4.36
16	5,996.6	406.0	13.81	7,364.2	460.7	11.79	18,388.7	1,375.4	4.28

- automatic parallelization of BLAS by OpenMP is not efficient

+ `Dissection` uses coarse grain parallelization by Pthreads

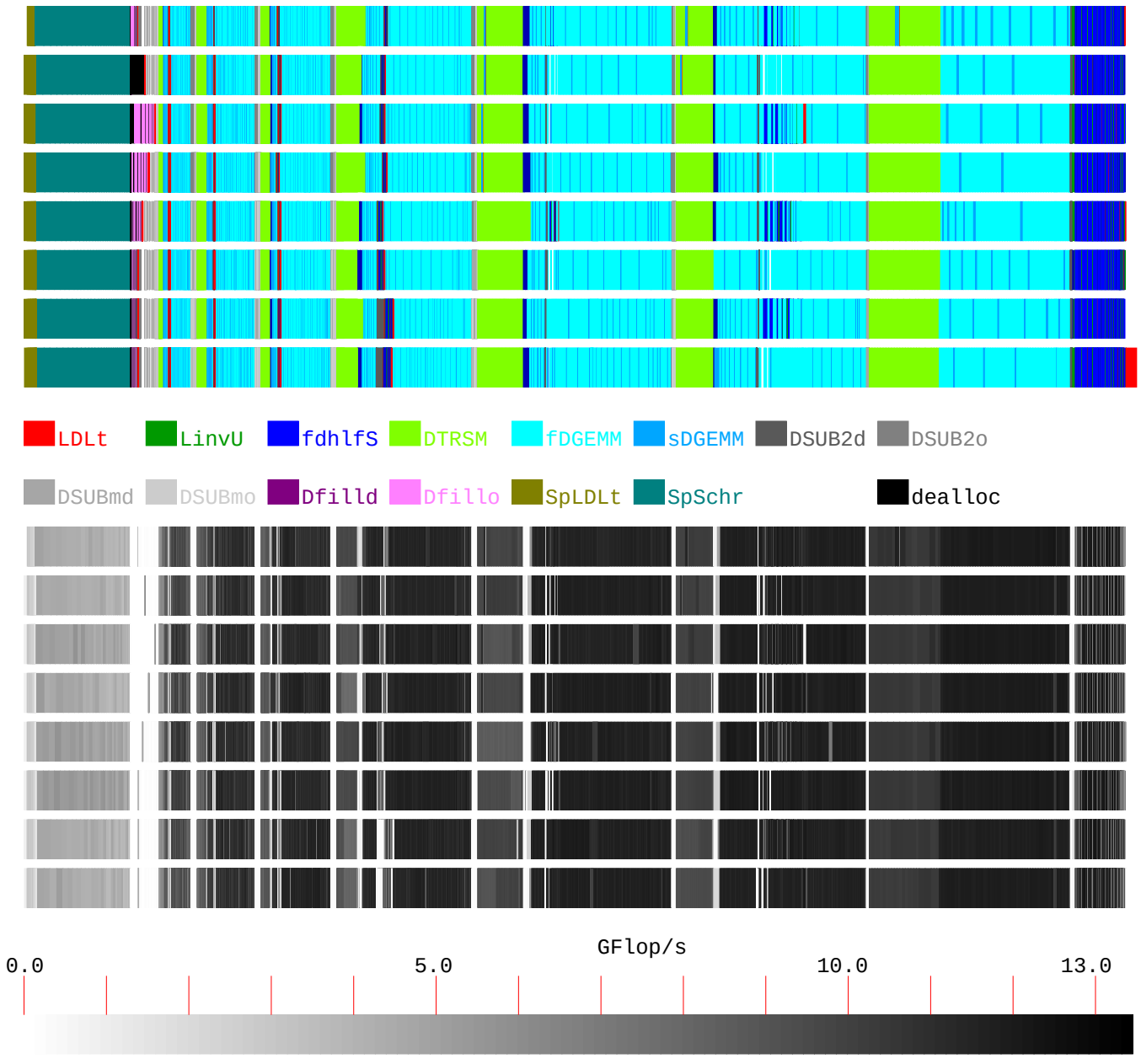
# core	GFlop/s	time for parallel tasks elapsed time	time for parallel tasks idle time of cores	time for the numerical factorization elapsed time	time for the numerical factorization CPU time
1	11.207	76.982	0.000	77.776	77.505
2	22.214	38.840	0.049	39.651	78.133
4	42.473	20.252	1.122	21.089	80.261
6	59.138	14.545	2.368	15.397	85.241
8	75.217	11.414	1.667	12.302	89.434
10	87.651	9.795	2.794	10.681	94.482
12	96.187	8.925	3.622	9.845	102.246

96.187GFlops /159.84GFlops = 60% : running on NUMA is another challenge

Task scheduling and performance of `Dissection` solver

$n = 206,763$

Westmere Xeon 5680@3.3GHz, 6core x2, POSIX threads

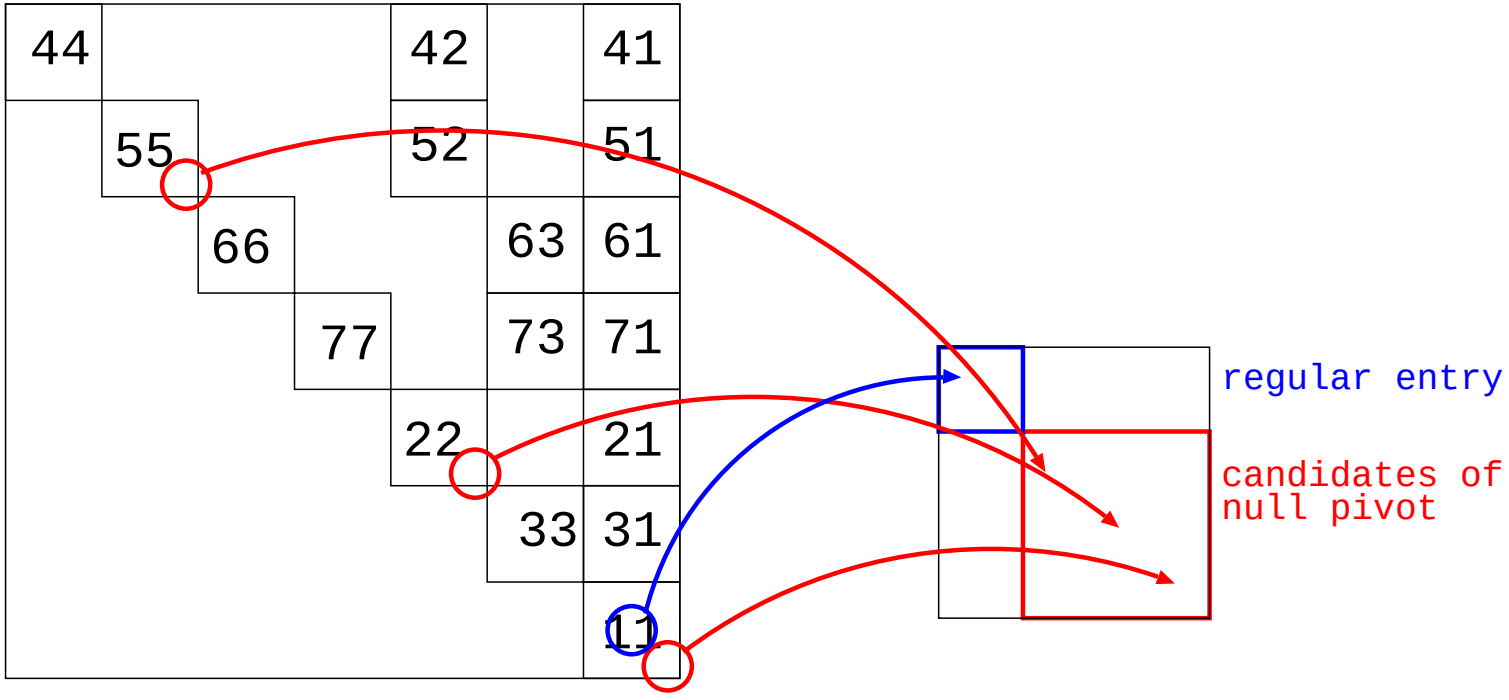


Detection of the kernel of matrix in `Dissection` solver

kernel of stiffness matrix $\longrightarrow$ coarse space for domain decomposition

"theoretically" the last block is singular for semi-positive definite matrices
"numerically" all blocks may be nearly singular

$\tilde{\varepsilon} > 0$: given threshold for null pivot
$|a_{i\,i}|/|a_{i-1\,i-1}| < \tilde{\varepsilon} \;\Rightarrow\; |a_{i\,i}|$ is null pivot.



Schur complement matrix from suspicious null pivots and additional nodes
$\Longrightarrow$ the algorithm to compute dimension of the kernel

How to determine dimension of the kernel?

$$\begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} = \begin{bmatrix} \tilde{A}_{11} & 0 \\ \tilde{A}_{21} & \tilde{S}_{22} \end{bmatrix} \begin{bmatrix} I_1 & \tilde{A}_{11}^{-1}\tilde{A}_{12} \\ 0 & I_2 \end{bmatrix} \qquad \tilde{S}_{22} = 0 \Rightarrow \mathrm{Ker}\tilde{A} = \begin{bmatrix} \tilde{A}_{11}^{-1}\tilde{A}_{12} \\ -I_2 \end{bmatrix}$$

numerical example:

symmetric semi-positive definite, $m + k = 4 + 6 = 10$

by Householder-QR factorization: Matrix Computation; Golub, Van Loan, 1996

4.60e-02	-1.20e-02	2.91e-03	1.16e-02	2.24e-02	-9.33e-05	-3.60e-02	8.22e-03	-7.77e-03	-2.90e-02
0.0	3.84e-02	4.84e-03	-2.21e-02	1.87e-02	1.30e-03	-9.14e-03	-2.74e-02	1.48e-02	-1.91e-02
0.0	0.0	2.96e-02	1.68e-03	-2.55e-02	1.11e-04	1.28e-02	-1.04e-04	-1.12e-03	-1.20e-02
0.0	0.0	0.0	1.28e-02	-1.66e-03	8.48e-04	-4.29e-05	-7.90e-04	-8.56e-03	2.51e-03
0.0	0.0	0.0	0.0	1.23e-11	-5.49e-13	-8.30e-12	1.67e-13	2.10e-14	-7.08e-12
0.0	0.0	0.0	0.0	0.0	6.70e-13	-1.02e-13	-5.33e-13	-1.62e-13	1.18e-13
0.0	0.0	0.0	0.0	0.0	0.0	3.33e-13	-2.48e-14	-6.61e-14	-3.18e-13
0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.22e-13	4.46e-15	-4.34e-14
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.05e-14	8.16e-15
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-1.09e-14

how to set threshold to distinguish non-zero (1.28e-02) and zero (1.23e-11) values ?

$\Longrightarrow$ an algorithm by measuring dimension of a projection onto the image space and detecting existence of inverse of the matrix using higher precision arithmetic

Detection of the kernel of matrix (1/4)

$A \in \mathbb{R}^{N \times N}$, dimKer$A = k \geq 1$, dimIm$A \geq m$.

two parameters: l, n, which define factorization,

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ A_{21} & S_{22} \end{bmatrix} \begin{bmatrix} I_1 & A_{11}^{-1} A_{12} \\ 0 & I_2 \end{bmatrix} \quad A_{11} \in \mathbb{R}^{(N-l)\times(N-l)} \text{ or } \mathbb{R}^{(N-n)\times(N-n)}.$$

projection : $P_n^{\perp} : \mathbb{R}^N \rightarrow \text{span} \begin{bmatrix} \bar{A}_{11}^{-1} A_{12} \\ -I_{22} \end{bmatrix}^{\perp}$, $A_{11} \in \mathbb{R}^{(N-n)\times(N-n)}$

solution in subspace, $\bar{A}_{N-l}^{\dagger} b = \begin{bmatrix} \bar{A}_{11}^{-1} b_1 \\ 0 \end{bmatrix}$, $A_{11} \in \mathbb{R}^{(N-l)\times(N-l)}$, $b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} b_{N-l} \\ b_l \end{bmatrix}$

$\bar{A}_{11}^{-1}$: in quadruple-precision with perturbation to simulate double-precision round-off error.

n : candidate of dimension of the kernel

compute for $l = n-1, n, n+1$

$$\text{err}_l^{(n)} := \max \left\{ \max_{x=[0\ x_l]\neq 0} \frac{||P_n^{\perp}(\bar{A}_{N-l}^{\dagger} A\, x - x)||}{||x||}, \max_{x=[x_{N-l}\ 0]\neq 0} \frac{||\bar{A}_{N-l}^{\dagger} A\, x - x||}{||x||} \right\}$$

$$\begin{array}{rcccccc} n = k+1 & \Leftrightarrow & \text{err}_{n-1} \approx 0 & \wedge & \text{err}_n \approx 0 & \wedge & \text{err}_{n+1} \sim 1 \\ n = k & \Leftrightarrow & \text{err}_{n-1} \gg 0 & \wedge & \text{err}_n \approx 0 & \wedge & \text{err}_{n+1} \sim 1 \\ n = k-1 & \Leftrightarrow & \text{err}_{n-1} \gg 0 & \wedge & \text{err}_n \gg 0 & \wedge & \text{err}_{n+1} \sim 1 \end{array}$$

A_{N-k+1}^{-1} does not exist. $\text{err}_{k-1}^{(k)}$ is computable and is similar order of $||\bar{A}_N^{-1} A_N - I_N||$.

Detection of the kernel of matrix (2/4)

$\bar{A}^{-1}$: in quadruple-precision with perturbation to simulate double-precision round-off error

with decomposition of A into regular part $A_{1\,1} \in \mathbb{R}^{m \times m}$ and others

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \sim \begin{bmatrix} I_{11} & 0 \\ A_{21}\widehat{A_{11}^{-1}} & I_{22} \end{bmatrix} \begin{bmatrix} A_{11} & 0 \\ 0 & S_{22} \end{bmatrix} \begin{bmatrix} I_{11} & \widehat{A_{11}^{-1}}A_{12} \\ 0 & I_{22} \end{bmatrix}$$

$\widehat{A_{11}^{-1}} = U_{11}^{-1} D_{11}^{-1} L_{11}^{-1}$ + artificial perturbation with $\varepsilon_0 \sim 10^{-16}$.

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} \sim \begin{bmatrix} I_{11} & -\widehat{A_{11}^{-1}}A_{12} \\ 0 & I_{22} \end{bmatrix} \begin{bmatrix} \check{A}_{11}^{-1} & 0 \\ 0 & \check{S}_{22}^{-1} \end{bmatrix} \begin{bmatrix} I_{11} & 0 \\ -A_{21}\widehat{A_{11}^{-1}} & I_{22} \end{bmatrix}$$

$\check{A}_{11}^{-1} = U_{11}^{-1} D_{11}^{-1} L_{11}^{-1}$ within quadruple-precision $\sim 10^{-33}$.

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} - \begin{bmatrix} I_1 & 0 \\ 0 & I_2 \end{bmatrix}$$

$$= \begin{bmatrix} (\check{A}_{11}^{-1}A_{11} - I_1) - \widehat{A_{11}^{-1}}A_{12}\check{S}_{22}^{-1}A_{21}(I_1 - \widehat{A_{11}^{-1}}A_{11}) & \check{A}_{11}^{-1}A_{12} - \widehat{A_{11}^{-1}}A_{12}\check{S}_{22}^{-1}\hat{S}_{22} \\ \check{S}_{22}^{-1}A_{21}(I_1 - \widehat{A_{11}^{-1}}A_{11}) & \check{S}_{22}^{-1}\hat{S}_{22} - I_2 \end{bmatrix}$$

$\hat{S}_{22} := A_{22} - A_{21}\widehat{A_{11}^{-1}}A_{12}$

$||\widehat{A_{11}^{-1}}A_{11} - I_1|| \sim 10^{-16}$, $||\check{A}_{11}^{-1}A_{11} - I_1|| \sim 10^{-33}$, $||\check{S}_{22}^{-1}S_{22} - I_2|| \sim 10^{-33}$.

$S_{22} = 0 \Rightarrow \hat{S}_{22} \sim 10^{-16} \Rightarrow ||\bar{A}^{-1}A - I|| \gg 0$

$S_{22} \neq 0 \Rightarrow \quad ||\bar{A}^{-1}A - I|| \sim 10^{-16}$

Detection of the kernel of matrix (3/4)

Stokes equations with full-Neumann bc., $N = 199,808$, $m = 4$, $\tau = 10^{-2}$, $4 + 6 + 1$

eigenvalues by	Householder-QR	$[D]_i$: diagonal entry	$[D]_i^{-1}$
$6.99777789 \cdot 10^{-1}$	$4.98029566 \cdot 10^{-1}$	$3.70161579 \cdot 10^{-1}$	$2.70152295 \cdot 10^{0}$
$6.27846114 \cdot 10^{-1}$	$4.05027660 \cdot 10^{-1}$	$3.06310487 \cdot 10^{-1}$	$3.26466132 \cdot 10^{0}$
$4.80884945 \cdot 10^{-1}$	$3.69900258 \cdot 10^{-1}$	$2.79365437 \cdot 10^{-1}$	$3.57954087 \cdot 10^{0}$
$4.28888921 \cdot 10^{-1}$	$3.57246555 \cdot 10^{-1}$	$2.47548177 \cdot 10^{-1}$	$4.03961772 \cdot 10^{0}$
$-7.02489700 \cdot 10^{-11}$	$6.73940728 \cdot 10^{-11}$	$-6.48523283 \cdot 10^{-11}$	$-1.54196469 \cdot 10^{10}$
$-2.38674355 \cdot 10^{-12}$	$2.05913788 \cdot 10^{-12}$	$-1.84634192 \cdot 10^{-12}$	$-5.41611492 \cdot 10^{11}$
$-1.01390905 \cdot 10^{-12}$	$7.59609792 \cdot 10^{-13}$	$-6.04168305 \cdot 10^{-13}$	$-1.65516792 \cdot 10^{12}$
$-3.51767982 \cdot 10^{-13}$	$3.51718483 \cdot 10^{-13}$	$-4.62451857 \cdot 10^{-13}$	$-2.16238725 \cdot 10^{12}$
$-1.17581650 \cdot 10^{-13}$	$1.46890460 \cdot 10^{-13}$	$-1.31687059 \cdot 10^{-13}$	$-7.59376061 \cdot 10^{12}$
$-2.47928308 \cdot 10^{-14}$	$3.32364425 \cdot 10^{-14}$	$-4.66889871 \cdot 10^{-14}$	$-2.14183271 \cdot 10^{13}$
$-9.43431186 \cdot 10^{-16}$	$-2.92545721 \cdot 10^{-15}$	$-9.02986463 \cdot 10^{-15}$	$-1.10743631 \cdot 10^{14}$

k	$\text{err}_{k-1}^{(k)}$	$\text{err}_{k}^{(k)}$	$\text{err}_{k+1}^{(k)}$
7	$1.61887124 \cdot 10^{-6}$	$2.55270728 \cdot 10^{-16}$	$6.92933699 \cdot 10^{-1}$
6	$6.94434753 \cdot 10^{-5}$	$1.61887124 \cdot 10^{-6}$	$9.62285632 \cdot 10^{-1}$

dim. of kernel = 5	dim. of kernel = 6	dim. of kernel = 7
$8.29092462 \cdot 10^{-13}$	$1.39724349 \cdot 10^{-12}$	$2.68009592 \cdot 10^{-1}$
$2.59219292 \cdot 10^{-12}$	$5.55912542 \cdot 10^{-11}$	$1.20505842 \cdot 10^{-12}$
$8.98148568 \cdot 10^{-13}$	$3.16306840 \cdot 10^{-12}$	$1.44192677 \cdot 10^{-1}$
$7.39122100 \cdot 10^{-13}$	$8.25295635 \cdot 10^{-11}$	$3.61845561 \cdot 10^{-1}$
$2.56624545 \cdot 10^{-12}$	$3.37097407 \cdot 10^{-11}$	$2.01071952 \cdot 10^{-1}$
	$2.58069883 \cdot 10^{-12}$	$6.50183658 \cdot 10^{-2}$
		$1.07433781 \cdot 10^{-1}$

Detection of the kernel of matrix (4/4)

stationary Navier-Stokes equations, $Re = 12,800$, $N = 47,886$, $m = 4$, $\tau = 10^{-2}$, $4 + 1 + 1$

5×5 matrix by Householder QR factorization

1.123921e+0	3.461973e-1	2.459499e-1	3.507836e-1	2.188173e-13	1.809903e-1
	4.178588e-1	-1.530839e-1	-9.949687e-2	-3.980236e-15	-1.652780e-1
		3.408454e-1	-1.846635e-1	-8.625896e-15	-1.561819e-1
			2.801338e-1	-5.087587e-15	-2.801338e-1
				2.588860e-13	-2.587682e-13
					9.055738e-16

k	$\text{err}_{k-1}^{(k)}$	$\text{err}_{k}^{(k)}$	$\text{err}_{k+1}^{(k)}$
2	$3.66949509 \cdot 10^{-4}$	$2.79183050 \cdot 10^{-16}$	$7.22887622 \cdot 10^{-1}$

dim. of kernel = 1	dim. of kernel = 2
$2.36255459 \cdot 10^{-13}$	$3.55789626 \cdot 10^{-1}$
	$2.36642083 \cdot 10^{-13}$

dependency of kernel detection on a parameter in MUMPS

	Threshold for null pivots by CNTL(3)						
	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	automatic
elstct2							1.3095e-14
kernel	6	6	3	3	3	0	0
error	4.3817e-11	←	1.3878e+0	←	←	4.4009e+0	←
residual	7.1626e-14	←	1.0608e-15	←	←	1.2845e-15	←
stokes1							5.0793e-21
kernel	NA$^{(*)}$	6	6	6	5	5	0
error		5.1755e-8	←	←	2.6086e-1	6.9426e-3	2.7784e+1
residual		6.6675e-10	←	←	4.5757e-12	5.6831e-12	6.1865e-12
elstct3							2.8685e-16
kernel	6	6	3	3	1	0	0
error	1.4278e-10	←	2.4022e+0	←	3.1384e+0	3.4278e+0	←
residual	1.8237e-12	←	2.2366e-14	←	1.6868e-15	1.3948e-15	←
Koutsovasilis/F2							5.7237e-14
kernel	6	4	3	3	0	0	0
error	1.8309e-11	7.0498e-2	1.6832e-1	←	1.7918e+0	←	←
residual	1.3557e-13	1.6633e-14	7.4403e-16	←	6.1438e-16	←	←

($*$) stokes1 with CNTL(3)=-10^{-3} exceeds the memory limitation.

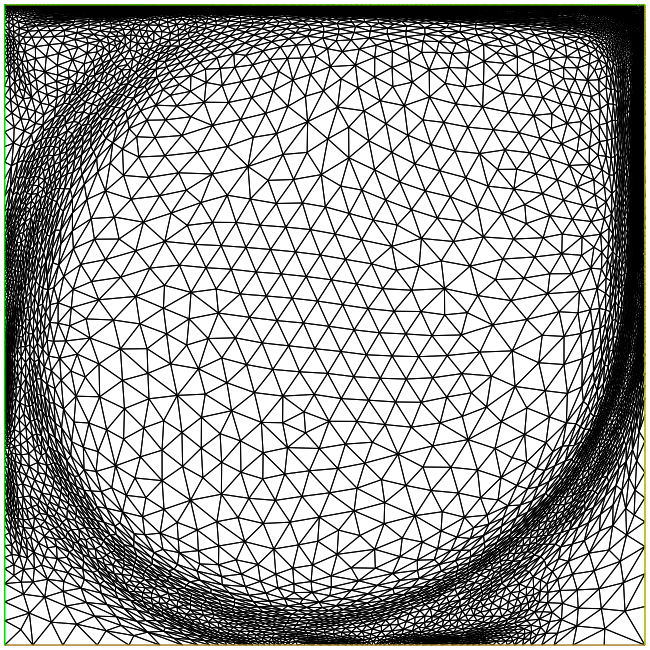
memory consumption: Dissection and IntelPardiso

real 206,763 sym		Dissection			Intel Pardiso		
		min	max	avg	min	max	avg
1 thread	fact	3.105	4.202 (+32%)	3.977 (+25%)	1.234	3.184	3.184
	fw/bw	4.008	4.008	4.007 (+25%)	3.189	3.193	3.193
4 threads	fact	3.315	4.356 (+23%)	4.151 (+17%)	1.551	3.540	3.540
	fw/bw	4.096	4.096	4.096 (+16%)	3.540	3.540	3.540
8 threads	fact	3.315	4.356 (+12%)	4.151 (+7%)	1.849	3.876	3.875
	fw/bw	4.096	4.096	4.096 (+6%)	3.876	3.876	3.876
12 threads	fact	3.816	4.953 (+17%)	4.412 (+5%)	2.140	4.223	4.223
	fw/bw	4.228	4.228	4.228 (+0%)	4.223	4.223	4.223

real 206,763 unsym		Dissection			Intel Pardiso		
		min	max	avg	min	max	avg
1 thread	fact	5.604	6.821 (+17%)	6.426 (+10%)	2.019	5.838	5.838
	fw/bw	6.401	6.401	6.401 (+9%)	5.842	5.847	5.847
4 threads	fact	5.815	6.970 (+13%)	6.657 (+8%)	2.303	6.150	6.150
	fw/bw	6.612	6.612	6.612 (+7%)	6.155	6.160	6.160
8 threads	fact	6.065	7.189 (+11%)	6.751 (+4%)	2.600	6.462	6.462
	fw/bw	6.675	6.675	6.675 (+3%)	6.467	6.472	6.472
12 threads	fact	6.315	7.412 (+9%)	6.881 (+1%)	2.894	6.788	6.787
	fw/bw	6.675	6.675	6.675 (-3%)	6.788	6.793	6.793

FreeFem++ example

stationary Navier-Stokes equations by Newton-Raphson iteration : `cavityNewtow.edp`



find $(u,p) \in V_g \times L^2_0(\Omega)$

$a(u,v) + a_1(u,u,v) + b(v,p) = 0$

$b(u,q) - \varepsilon\,(p,q) = 0$

$\varepsilon = 1.0 \times 10^{-6} \;\Rightarrow 0.$

```
load "Dissection"
defaulttoDissection();
XXMh [up1,up2,pp];
varf vDNS ([u1,u2,p],[v1,v2,q])=
  int2d(Th)(nu*(dx(u1)*dx(v1)+dy(u1)*dy(v1)
           + dx(u2)*dx(v2)+dy(u2)*dy(v2))
 //        + p*q*1.e-6
           + p*dx(v1)+p*dy(v2)
           - dx(u1)*q-dy(u2)*q
           + Ugrad(u1,u2,up1,up2)'*[v1,v2]
           + Ugrad(up1,up2,u1,u2)'*[v1,v2])
    + on(1,2,3,4,u1=0,u2=0);
//...
    up1[]=u1[];
    matrix Ans=vDNS(XXMh,XXMh,tgv=-1);
           set(Ans,solver=sparsesolver,tgv=-1,
               numthreads=4,tolpivot=1.0e-2);
    real[int] b=vNS(0,XXMh,tgv=-1);
    real[int] w=Ans^-1*b;
    u1[]-=w;
```

summary

`diss_int`(pointer dslv, real_or_complex, numthreads)
`diss_s_fact`(pointer dslv, dim, CSR_symboic_data, sym, decomposer, level)
`diss_n_fact`(pointer dslv, matrix_coef, scaling, tolpivot)
`diss_get_kern_dim`(pointer dslv, kern_dim)
`diss_get_kern_vecs`(pointer dslv, vectors)
`diss_fee`(pointer dslv)
written in Dissection.cpp for FreeFem++ load module.

sparse matrix is stored in CSR format with structurally symmetric pattern.
symmetric matrix is stored in either upper or lower.
real symmetric/unsymmetric with kernel detection.
complex general matrix without kernel detection.

symbolic factorization contains scheduling of parallel tasks in addition to ordering by `SCOTCH/METIS` and fill-in analysis.
`SCOTCH/METIS` sometimes does not produce aligned bisection tree.
$\rightarrow$ symbolic factorization phase is slower than one of `UMFPACK`

`double-double` quadruple precision arithmetic library will replace `Fortran90 REAL(16)` usage.