A hybrid factorization solver with mixed precision arithmetic for sparse matrices

Atsushi Suzuki,
RIKEN Center for Computational Science (R-CCS), Japan

Overview

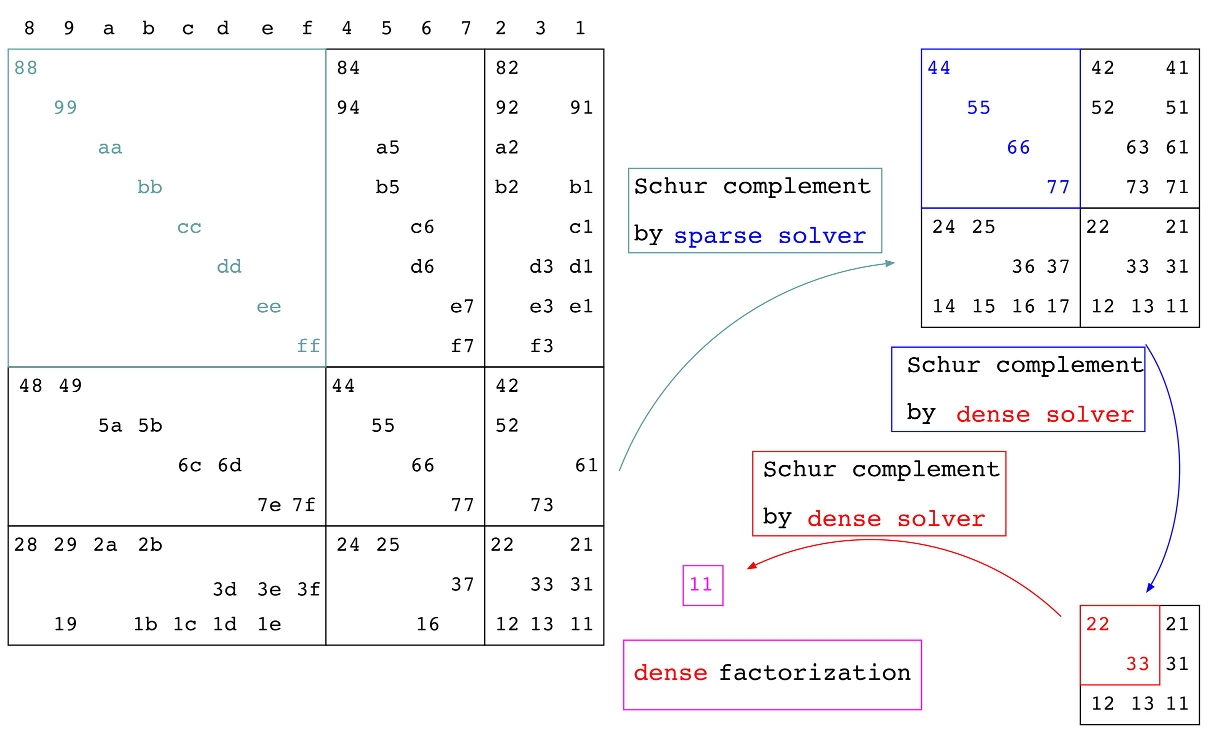
- LDU-factorization with pivoting strategy provides robust solver for sparse matrices with large condition number. Even when a matrix is singular, the kernel of the matrix could be detected numerically by using higher precision arithmetic.
- Computational complexity of the factorization solver is high about $O(N^{2.5})$ for sparse matrix for finite element analysis with lower order elements such as P1 or P2. This complexity in order of N cannot be reduced, but by using lower precision arithmetic, computational cost and memory usage could be reduced.
- Nested-dissection ordering provides multi-frontal factorization and threshold pivoting for each submatrix can extract a set of submatrices with moderate condition number.
- Generation of the Schur complement matrix against the moderate part needs to be performed in higher precision to keep the accuracy of the whole factorization algorithm.
- A hybrid factorization algorithm consists of "factorization with threshold pivoting in lower precision", "generation of the Schur complement in higher precision by iterative solver using factorized matrix in lower precision as a preconditioner", and "factorization with full pivoting and kernel detection in higher precision".
- True mixed precision arithmetic is used in forward/backward substitution procedure with factorized matrix in lower precision and RHS vectors in higher precision for the preconditioner whose input and output are higher precision, where thanks to no truncation of floating-point data during substitution, the preconditioner is kept as a linear operator.

Condition number $\kappa(A)$ of stiffness matrix A in finite element analysis

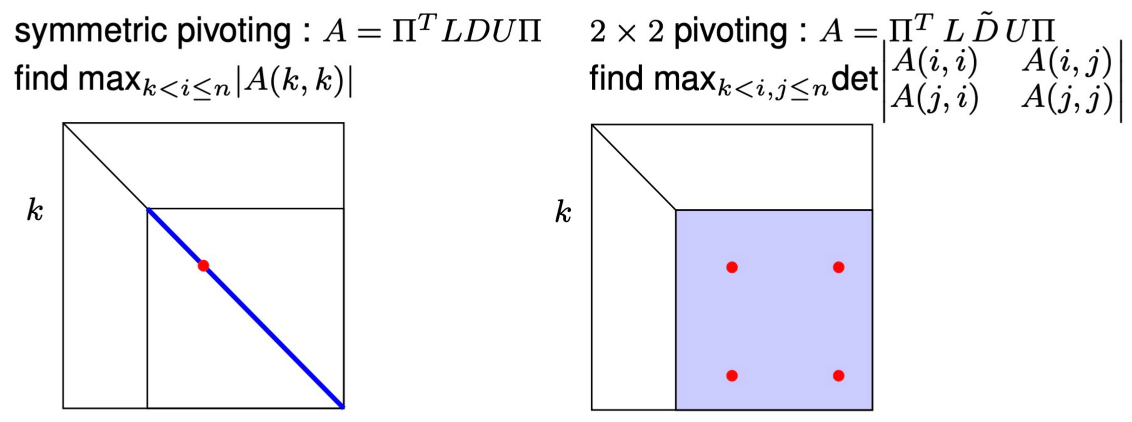
- elasticity problem with composite material (different material property) 10^9
- incompressible Naiver-Stokes flow (divergence freeness) 10^6
- semi-conductor problem (exponential weight in drift term) 10^{14}
- level set method for free-boundary problem (adaptive mesh refinement) 10^3

Accuracy of linear solver needs to cover $(1/h)^p \times \kappa(A)$ during nonlinear solution process with p-th order finite element approximation with mesh size h.

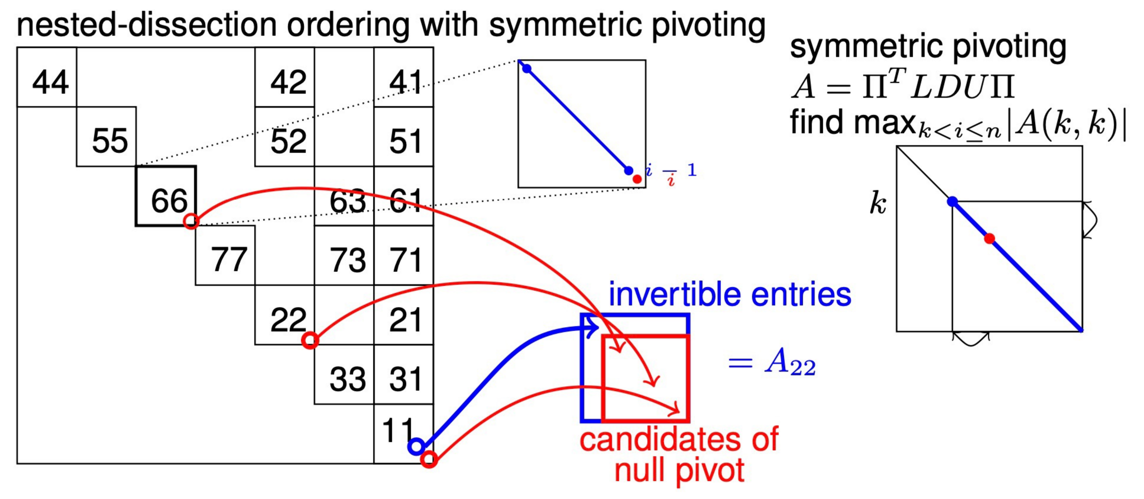
Factorization procedure with threshold pivoting



- Nested-dissection ordering is obtained by METIS or SCOTCH graph partitioner.
- The lowest level consists of sparse sub-matrices, and dense ones in above levels, where BLAS level 3 operations are dominant.



- Sparse matrix from FEM or FVM usually consists of symmetric nonzero pattern, "structurally symmetric" and symmetric pivoting is more efficient than partial pivoting because it keeps structural symmetry during factorization.
- However, symmetric pivoting may meet break down for indefinite matrix.
- 2x2 pivoting allows stable factorization and will be applied to postponed pivots by a strategy with threshold.
- No perturbation is added to diagonal entries, whereas PARDISO [2] and SuperLU_DIST[1] add $\sqrt{\varepsilon}$ -perturbation for stabilization of the factorization procedure.



- Threshold pivoting uses user defined parameter $\tau(\simeq 10^{-2})$.
- When the jump of diagonal entries becomes bigger as $|A(k,k)/A(k+1,k+1)| > 1/\tau$ then entries with index larger than k are postponed to the last Schur complement.
- The last Schur complement is generated from all postponed entries and whose singularity is analyzed by kernel detection algorithm[4] .
- Overall strategy of factorization is similar to MUMPS [3] except for the place where entries are postponed.
- This procedure brings us automatic decomposition of the matrix into union of moderate and hard parts.

Hybrid factorization algorithm

Decomposition into moderate part A_{11} and hard part A_{22} by threshold pivoting with symmetric permutation

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ A_{21} & S_{22} \end{bmatrix} \begin{bmatrix} I_{11} & A_{11}^{-1}A_{12} \\ 0 & I_{22} \end{bmatrix} \qquad \begin{matrix} \kappa(A_{11}) \simeq 10^6 \\ \kappa(A_{22}) \geqq 10^{12} \end{matrix}$$

The Schur complement $S_{22} = A_{22} - A_{21}A_{11}^{-1}A_{12}$ is generated via solving a linear system with multiple RHS $A_{11}X_{12} = A_{12}$ by block-GCR solver with preconditioner[6].

- ✓ Solution of $Q\,X_0 = B$ in lower precision is obtained by forward/backward substitutions using LDU-factorization of A_{11} in lower precision, which is used as a preconditioner.
- ✓ Solution of $A_{11}X_{12} = A_{12}$ is computed in higher precision.
- ✓ The Schur complement is factorized in higher precision with kernel detection to verify singularity and overall accuracy of the factorization is kept in higher precision.

linear system with multiple RHS $B = [b_1, \ldots, b_m]$
solution in higher precision : $A\,[x_1, \ldots, x_m] = B$
solution in lower precision : $Q\,X_0 = B$
residual at the initial stage : $QP_0 = R_0 = B - AX_0$
loop j = 0, ...

$$\mathcal{M}_j = (AQ^{-1}P_j)^T(AQ^{-1}P_j)$$
$$\mathcal{A}_j = R_j^T(AQ^{-1}P_j)$$
$$X_{j+1} = X_j + Q^{-1}P_j\mathcal{M}_j^{-1}\mathcal{A}_j$$
$$R_{j+1} = R_j - AQ^{-1}P_j\mathcal{M}_j^{-1}\mathcal{A}_j$$
$$\mathcal{B}_{i,j} = -(AQ^{-1}R_{j+1})^T(AQ^{-1}P_i)$$
$$P_{j+1} = R_{j+1} + \sum_{0\leq i\leq j} P_i\,\mathcal{M}_i^{-1}\mathcal{B}_{i,j}$$

block-GCR uses SpMM with sym. permutation.

Mixed precision arithmetic for substitution solver

Forward/backward substitutions performed for multiple RHS in higher precision, with factorized matrix in lower precision

A dense sub-matrix in the nested dissection layer is factorized in a block way for parallel task execution by a DAG tree.

$$A_{11} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} = \begin{bmatrix} L_1 & & \\ \widetilde{K}_{21} & L_2 & \\ \widetilde{K}_{31} & \widetilde{K}_{32} & L_3 \end{bmatrix} \begin{bmatrix} D_1 & & \\ & D_2 & \\ & & D_3 \end{bmatrix} \begin{bmatrix} U_1 & \widetilde{K}_{12} & \widetilde{K}_{13} \\ & U_2 & \widetilde{K}_{23} \\ & & U_3 \end{bmatrix}$$

- forward substitution : $L_1X_1 = Y_1$ by TRSM with α=1
- updating contribution from off-diagonal : $Y_2' = Y_2 - \widetilde{K}_{21}X_1$ by GEMM with α=−1, β=1
- diagonal scaling : $X_3' = D_3^{-1}X_3$

double double double → double quadruple quadruple

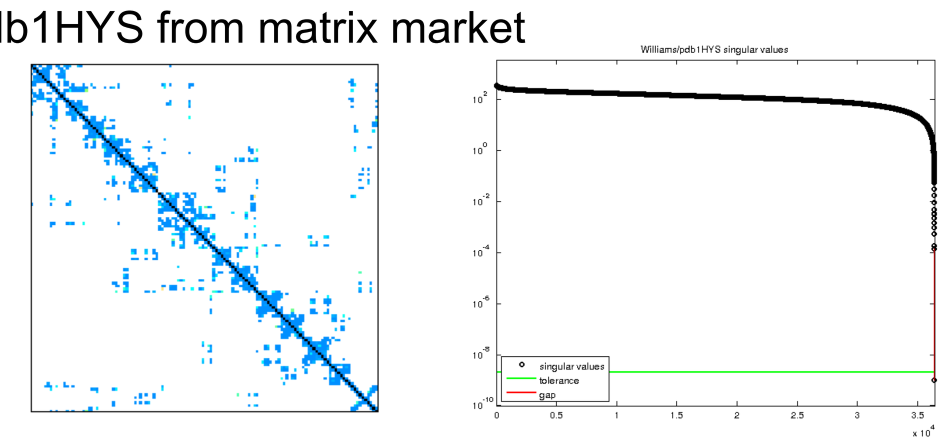
$L_1X_1 = Y_1$ → $L_1\widehat{X_1} = \widehat{Y_1}$

avoiding downward casting + increasing accuracy + linearity

- TRSM and GEMM in tmBLAS [5] for mixed precision data and operator are used.
- Other mathematical operations also follow mixed precision arithmetic defined by C++ templated functions in tmBLAS.

Numerical examples

pdb1HYS from matrix market



figures taken from https://sparse.tamu.edu/Williams/pdb1HYS

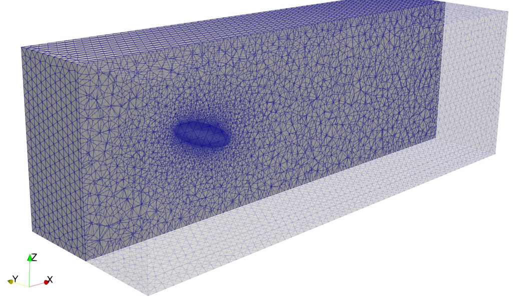
symmetric matrix $\mathcal{A}$: $n = 36,414, nnz = 4,344,765$
$\tau = 0.05$, # postponed entries = 17, S_{22} : 17×17 matrix, invertible, dim Ker $S_{22} = 0$

$\lambda_{\max}(\mathcal{A}) = 34.854$	$\lambda_{\min}(\mathcal{A}) = 9.9876 \times 10^{-10}$	$\kappa(\mathcal{A}) = 3.4962 \times 10^{11}$
$\lambda_{\max}(\mathcal{A}_{11}) = 1.0002$	$\lambda_{\min}(\mathcal{A}_{11}) = 3.8109 \times 10^{-4}$	$\kappa(\mathcal{A}_{11}) = 2.6448 \times 10^{3}$

	double	mixed(double+single)	single
error	1.2650×10^{-6}	1.6647×10^{-6}	3.1178×10^{-2}
residual	7.6201×10^{-16}	9.6442×10^{-16}	4.4906×10^{-7}
time in second	0.5328	0.4604	0.4053
memory (MB)	410	286	240

Apple M1 Max @ 3.23GHz

incompressible flow equations with full-Neumann B.C.



$\mathcal{A} = \begin{bmatrix} A & B^T \\ B & -\delta D \end{bmatrix}$, indefinite, singular

$[A]_{i\,j} = \int_\Omega D(\varphi_j) : D(\varphi_i)$,
$[B]_{i\,j} = -\int_\Omega \nabla\cdot\varphi_j\psi_i$,
$[D]_{i\,j} = \sum_K \int_K \nabla\psi_j \cdot \nabla\psi_i$

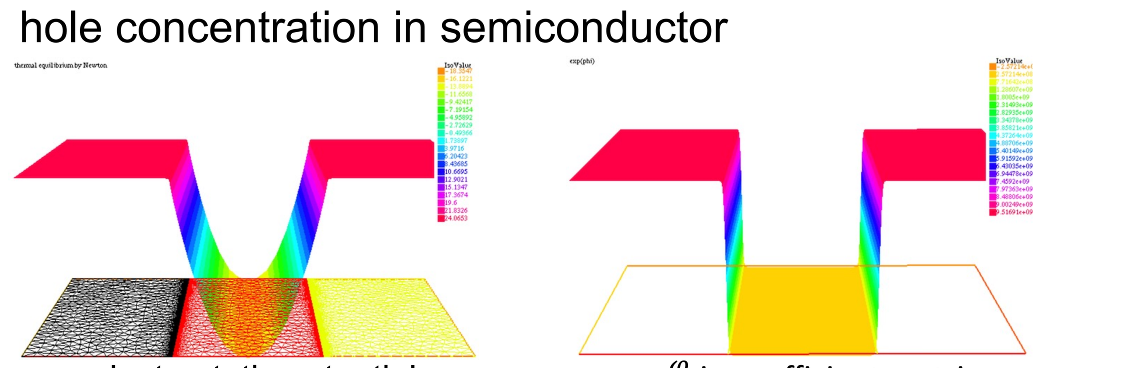
$n = 374,136\ nnz = 27,146,848$,
$\tau = 0.75$, # postponed entries = 16, S_{22} : 16×16 matrix, dim Ker$S_{22} = 6$.

$\lambda_{\max}(\mathcal{A}) = 2.5003$	
$\lambda_{\min}(\mathcal{A}) = 1.1437 \times 10^{-19}$	$\kappa(\mathcal{A}) = 2.1863 \times 10^{19}$
$\lambda_{\min}(\mathcal{A}\vert_{\mathrm{Im}\mathcal{A}}) = 8.0038 \times 10^{-8}$	$\kappa(\mathcal{A}\vert_{\mathrm{Im}\mathcal{A}}) = 3.1240 \times 10^{7}$
$\lambda_{\min}(\mathcal{A}_{11}) = 1.6677 \times 10^{-8}$	$\kappa(\mathcal{A}_{11}) = 5.9962 \times 10^{7}$

	double	mixed(double+single)	single
error	4.3646×10^{-13}	2.0301×10^{-12}	1.7046×10^{-2}
residual	1.2730×10^{-15}	5.1881×10^{-15}	7.0494×10^{-7}
dim. of kernel	6	6	0
time in second	33.390	22.983	15.228
memory (MB)	17,055	10,116	9,423

Apple M1 Max @ 3.23GHz

hole concentration in semiconductor



electrostatic potential φ $\qquad$ e^{φ} in coefficient matrix

Slotboom variable ξ and gradient fields J_p: $\mathrm{div}J_p = 0$, $e^{\varphi}J_p = -\nabla\xi$.

mixed finite element method with Raviart-Thomas, $n = 40,323$, $nnz = 401,243$
$\tau = 0.01$, # postponed entries = 86, S_{22} : 86×86 matrix, dim Ker$S_{22} = 1$.

$\lambda_{\max}(\mathcal{A}) = 6.1349 \times 10^{10}$	$\lambda_{\min}(\mathcal{A}\vert_{\mathrm{Im}\mathcal{A}}) = 3.5704 \times 10^{-12}$	$\kappa(\mathcal{A}\vert_{\mathrm{Im}\mathcal{A}}) = 1.7183 \times 10^{22}$
$\lambda_{\max}(\mathcal{A}_{11}) = 2.0080 \times 10^{2}$	$\lambda_{\min}(\mathcal{A}_{11}) = 3.6520 \times 10^{-12}$	$\kappa(\mathcal{A}_{11}) = 5.5472 \times 10^{13}$

	quadruple	mixed(quadruple+double)	double
error	5.3652×10^{-20}	1.3061×10^{-21}	8.9382×10^{-6}
residual	5.3850×10^{-32}	1.5514×10^{-32}	4.7860×10^{-16}
dim. of kernel	1	1	1
time in second	16.599	2.6459	0.4064
memory (MB)	387	255	195

Apple M1 Max @ 3.23GHz

References

1) X.S, Li, J.W. Demmel. SuperLU_DIST : a scalable distributed-memory sparse direct solver for unsymmetric linear systems. ACM Transactions on Mathematical Software 2003, 29: 110-140, DOI: 10.1145/779359.779361
2) O. Schenk K. Gärtner. Solving unsymmetric sparse systems of liner equations with PARDISO. Future Generation Computer Systems 2004, 20: 475-497, DOI: 10.1016/j.future.2003.07.011
3) P.R. Amestoy, I.S. Duff, J.-Y. L'Excellent. Multifrontal parallel distributed symmetric and unsymmetric solvers. CMAME 2000, 184: 501-520, DOI: 10.1016/S0045-7825(99)00242-X
4) A. Suzuki, F.-X. Roux. A dissection solver with kernel detection for symmetric finite element matrices on shared memory computers, IJNME 2014, 100: 136-164,DOI:10.1002/nme.4729
5) A. Suzuki, D. Mukunoki T. Imamura, tmBLAS : a mixed precision BLAS by C++ tempalte, ISC2023, May 2023, https://www.r-ccs.riken.jp/labs/lpnctrl/projects/tmblas
6) A. Suzuki. A factorization algorithm for sparse matrix with mixed precision arithmetic, ECCOMAS 2022, DOI:10.23967/eccomas.2022.006