Jurnal Matematika UNAND
Vol. **15** No. **3** Hal. 449 – 459
Edisi Juli 2026
ISSN : 2303–291X
e-ISSN : 2721–9410

https://doi.org/10.25077/jmua.15.3.449-459.2026

SPARSE MATRIX FACTORIZATION USING MULTIFRONTAL QR AND SUPERNODAL CHOLESKY METHODS

MOH HASAN*, CHINTYA MONIKASARI, KUSBUDIONO
Department of Mathematics,
Faculty of Mathematics and Natural Sciences, University of Jember,
Jl. Kalimantan 37, Jember, 68121, Indonesia.
email : hasan.fmipa@unej.ac.id

Received August 21, 2025 Received in revised form July 20, 2026
Accepted July 23, 2026 Available online July 31, 2026

Abstract. This paper discusses about factorization of sparse matrices. A nested dissection method is used to reorder sparse matrices while multifrontal QR and supernodal Cholesky methods are applied to factorize the matrices. Simulations are then carried out to three groups of matrices of the same size, where each group consists of four matrices having various sparsities. The objectives of this study are to investigate the effect of sparsity and the performances of the factorization methods. Results show that the effects of sparsities on the parameters of matrix groups depend on their sparsity slope. It is shown that supernodal Cholesky factorization has better performance than multifrontal QR.

Keywords: Factorization; Ordering; Multifrontal QR; Supernodal Cholesky; sparsity slope

1. Introduction

A sparse matrix is a matrix having most of the elements are zero [1]. Sparse matrices could be found in many areas, such as structural engineering, power network, and numerical computation. Problem of solving a large system of linear equation whose coefficients is sparse matrix becomes a sparse matrix factorization problem. Several methods have been widely applied to factorize sparse matrices including QR and Cholesky methods.

The application of QR factorization in an encrypted domain based on homomorphic encryption and the Gram-Schmidt algorithm obtains low mean square error [2]. QR method with a non-interactive and efficient outsourcing algorithm provide efficiency in reducing computational complexity and time consuming [3]. QR factorization has been developed into multifrontal QR factorization by Duff and Reid

*Corresponding author

[4]. This method was often applied in runtime systems to parallelize the sparse matrix factorization method using COLAMD from the StarPU library which takes less time than the Suite Sparse library [5]. When applied to GPUs, multifrontal QR has a tremendous speed increase to 11 times that of previous methods for large sparse matrices [6]. These advances improve computational efficiency and reduce filling-related problems by maintaining low execution times through minimum degree ordering [7].

Sparse matrix factorization can also be done via the Cholesky method. The Cholesky method with nested dissection ordering can increase processing speed in completing matrices in linear systems [8]. The Cholesky method of the MKL library shows impressive performance, achieving factorization speeds of up to 200 Gflops with the recursive blocking method, which improves computational performance and regularity [9]. Cholesky factorization has an irregular memory access pattern. The Cholesky method has been improved into supernodal Cholesky to overcome this limitation. It consists of merging adjacent matrix columns with similar sparse structures into supernodes to ensure reducing memory access overhead during sparse matrix factorization [10]. In addition, the left-looking algorithm used in the supernodal Cholesky method contributes to reduce memory usage and increase the performance of BLAS (Basic Linear Algebra Subprograms) [11]. Nested dissection ordering can reduce the cost and computational time for calculating sparse matrices [12,13].

This study evaluates the applicability of multifrontal QR and supernodal Cholesky methods to factorize sparse matrices of varying sizes and sparsities. For these factorizations, we propose nested dissection techniques for ordering sparse matrices in order to reducing the occurrence of fill-ins and to clustering nonzero entries as close as to the diagonals. The evaluation is focused on some parameters, namely computational cost, consumption time, speed, and sparsity. The results are expected to provide insight into the usability of multifrontal QR and supernodal Cholesky factorization to solve large system of linear equations.

2. Research Methods

Three sparse matrices available in suite sparse library is used, namely Mesh2e1, 1138_bus, and Dubcova1 [14]. These matrices are all symmetric, but of different sizes, and obtained from problems in structural engineering, power networks, and PDE solvers. For the rest of this paper, these 3 matrices are called sparse matrices **A**, **B**, and **C** respectively. The sparse matrix **A** has 306×306 elements with 2,018 nonzero elements, resulting in a sparsity of 97.84%. The sparse matrix **B** has a size of 1138×1138 having sparsity of 99.69%. Lastly, the sparse matrix **C** has $16,129 \times 16,129$ elements with sparsity of 99.90%. Figure 1 displayed a representation of these sparse matrices. In these pictures, dots represent nonzero valued elements. The more nonzero elements, the higher matrix's density or the less sparsity (sparsity $= 1 -$ density). To perform sparse matrix factorization, Suite Sparse Software could be used [15]. The research consists of three stages, i.e. ordering, factorization, and simulation.

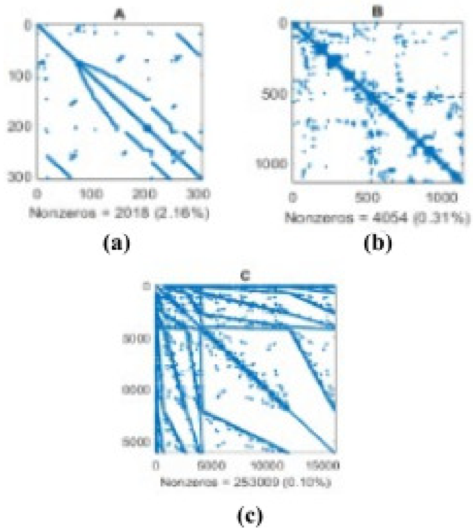

Figure 1. The representation of sparse matrices (a) **A**, (b) **B**, and (c) **C**.

2.1. *Ordering*

The ordering phase is applied to decrease fill-ins (sparse matrix elements transition from zero to nonzero elements) and time spent on factorization. In this research, nested dissection technique is used, since it is more effective for significant problems arising from 2D and 3D spatial creations (finite-element methods) than the minimum degree technique [16]. The nested dissection technique is an algorithm that aims to remove a set of vertices (and the edges incident on those vertices) from a symmetric matrix graph to leave the remaining graph in the disconnected part [17].

2.2. *Factorization*

The two factorization methods used in this study, i.e. multifrontal QR and supernodal Cholesky, explained briefly as follows.

(1) Multifrontal QR is a method that aimed to find QR factorization of sparse matrix through multifrontal. The work of multifrontal QR consists of the computation of a fill-reducing pivotal order and elimination process to determine the size of the factors and frontal matrices [18]. The frontal matrices are a full matrix in which all arithmetic from a front moved along it. The front is a set of active variables ordered systematically from one end of the region to the other, and these variables are involved in 1 or more of the elements sub matrix that was assembled [17]. QR factorization employing Householder's technique [19] and could be written as:

$$K = QR, \tag{2.1}$$

with R is the upper triangular matrix and Q is the orthogonal or the unitary matrix. QR factorization with Householder's method begin by defining P as:

$$P = 1 - 2ww^T, \tag{2.2}$$

with w is a column matrix and P is a symmetric matrix. The next step forms w_1 matrix from the coefficients in the first column of K:

$$w_1^T = \mu_1[(k_{11} - s_1)k_{21}k_{31}k_{n1}],$$

with $\mu_1 = \dfrac{1}{\sqrt{2s_1(s_1 - k_{11})}}$ and $s_1 = \pm(\sum_{j=1}^{n} k_{j1}^2)^{\frac{1}{2}}$.

The value w_1 could be substituted in Equation 2.2 to form an orthogonal matrix $P^{[1]}$ and the matrix $K^{[1]}$ is formed from $P^{[1]}K$. The next step is to generate w_2 from the coefficients in the second column of $K^{[1]}$ until the orthogonal matrix $P^{[2]}$ and then the matrix $K^{[2]}$ could be formed from $P^{[2]}K^{[1]}$. This step is continued $n-1$ times to get an upper triangular matrix R:

$$R = P^{n-1} \cdots P^{[2]}P^{[1]}K \tag{2.3}$$

The matrix $P^{[i]}$ is the orthogonal matrix so $P^{[n-1]} \cdots P^{[2]}P^{[1]}$ is also the orthogonal matrix. Equation 2.1 can be written as $R = Q^{-1}K$ or $R = Q^TK$, and therefore from Equation 2.3, it is obtained $Q^T = P^{[n-1]} \cdots P^{[2]}P^{[1]}$ [20].

(2) Supernodal Cholesky is a method that aimed to transform sparse matrix into a series of dense supernode sub matrices [10]. Cholesky factorization form was:

$$K = LL^T,$$

$$\begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{12} & l_{13} \\ 0 & l_{22} & l_{23} \\ 0 & 0 & l_{33} \end{bmatrix}, \tag{2.4}$$

with L is the lower triangular matrix. Based on Equation 2.4, it is obtained:

$$l_{22} = \sqrt{k_{22} - l_{21}l_{12}}, \tag{2.5}$$

and

$$l_{32} = \frac{k_{32} - l_{31}l_{12}}{l_{22}}. \tag{2.6}$$

The next step is to determine the blocks for columns and rows in sparse matrix. Determination of this block is continued by updating the diagonal blocks with Equation 2.5 according to the size of the sparse matrix and looking for the next L matrix element with Equation 2.6 [16].

2.3. *Simulation*

Simulation is conducted to investigate the performance of multifrontal QR and supernodal Cholesky methods on some parameters, namely computational cost, time consumption, speed, and sparsity. Simulation is done by applying those two factorization methods to 12 different sparse matrices. The 12 matrices are matrices **A**, **B**, and **C** (Figure 1) as well as matrices resulting from powers of 2, 3, and 4 (Figure 2). From Figures 1 and 2, it is seen that the 12 sparse matrices have different densities, meaning they also have different sparsities as shown in Figure 3.

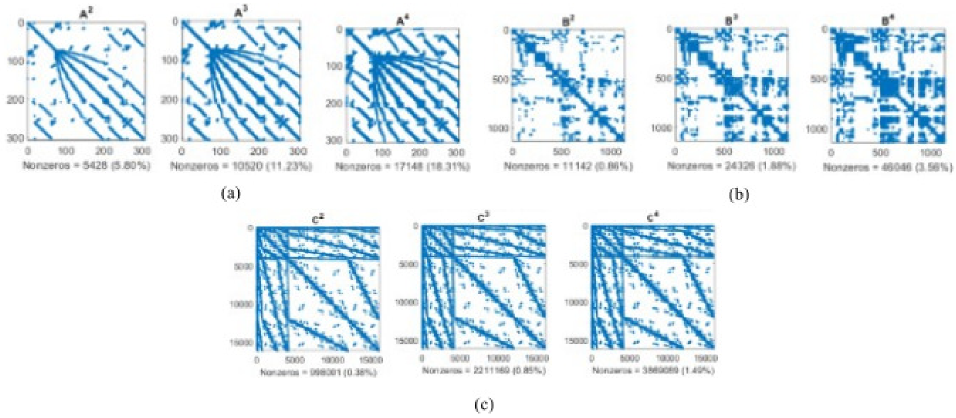

Figure 2. Matrices of power 2, 3, and 4 of **A**, **B**, and **C**: (a) $\mathbf{A}^2$, $\mathbf{A}^3$, and $\mathbf{A}^4$; (b) $\mathbf{B}^2$, $\mathbf{B}^3$, and $\mathbf{B}^4$; (c) $\mathbf{C}^2$, $\mathbf{C}^3$, and $\mathbf{C}^4$.

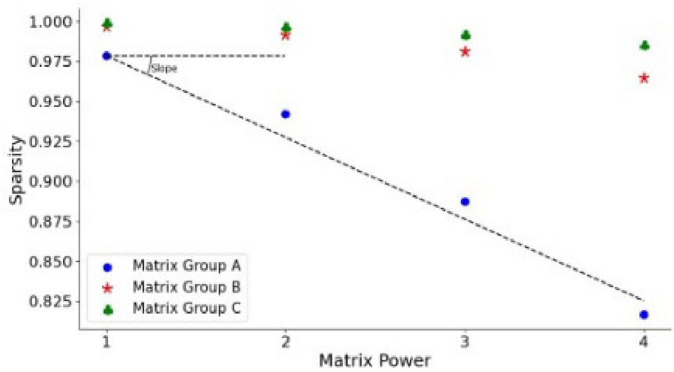

Figure 3. Sparsities of Martices A, B, C, and Their Power Matrices

Figure 3 shows the sparsity of the 12 matrices used for the simulation. The 12 sparse matrices are then clustered into three groups of matrices of the same size. The first group consists of matrix **A** and matrices of power 2, 3, and 4 (**A**, $\mathbf{A}^2$, $\mathbf{A}^3$, and $\mathbf{A}^4$ (matrix group **A**)). The second and third groups include matrices **B** and **C**, and matrices of power 2, 3, and 4 (**B**, $\mathbf{B}^2$, $\mathbf{B}^3$, and $\mathbf{B}^4$ (matrix group **B**), and **C**, $\mathbf{C}^2$, $\mathbf{C}^3$, and $\mathbf{C}^4$ (matrix group **C**)). It can be seen that, for these three groups, the sparsities of the power matrices are smaller than the sparsities of the original matrices (matrices **A**, **B**, and **C**), or we can say the higher the powers, the lower the sparsities. It can be observed that the sparsity slope depends on the sparsity of the original matrix.

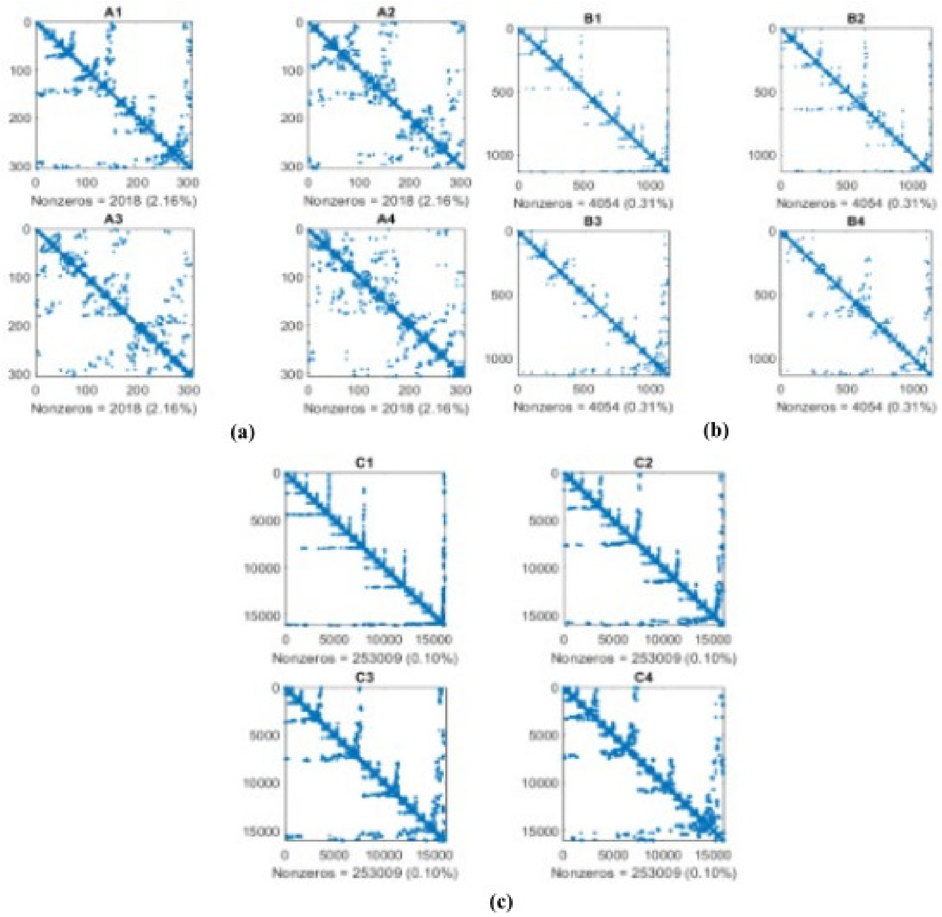

Figure 4. The Results of Ordering Matrix Groups: (a) **A**, (b) **B**, and (c) **C**.

3. Results and Discussion

The results of ordering the three matrix groups consisting of 12 sparse matrices using nested dissection method are shown in Figure 4. It can be seen from Figure 4 that, as expected, reordering yielded symmetric matrices and most of nonzero elements close to the diagonal reducing the bandwidth. It can also be seen that, after reordering, the densities (or sparsities) of matrices within each group are the same, though the positions of nonzero entries of matrices are not precisely the same.

The methods of multifrontal QR and supernodal Cholesky applied to factorize those three groups of sparse matrices. Factoring matrix group **A** produced triangular matrices as shown in Figure 5. In this figure and the following figures, R and L represent the triangular matrices obtained using multifrontal QR and supernodal Cholesky respectively. It can be observed that the sparsities of Rs are different for different original matrices ($\mathbf{A}$, $\mathbf{A}^2$, $\mathbf{A}^3$, and $\mathbf{A}^4$), but they do not a clear pattern. In contrast, the sparsities of Ls vary with the same pattern as sparsities of the matrices before being factorized (see Figure 3). It can be seen from Figure 5 and Table 2 that for the same original matrices, supernodal Cholesky factorization resulted in triangular matrices with higher sparsities compared to the results of multifrontal QR factorization. Analysis of other parameters such as computational cost, time consumption, and speed, shows that supernodal Cholesky is more efficient than multifrontal QR (Table 1). The effect of sparsity on the performance of the two factorization methods is trivial, denser matrices result in bigger parameters quantities.

Table 1. Parameters Values of Matrix Group **A** Factorizations

Sparse Matrix	Parameters	Multifrontal QR	Supernodal Cholesky
$\mathbf{A}$	Computational cost	0.6×10^6	0.306×10^3
	Time (s)	0.41×10^{-2}	0.25×10^{-2}
	Speed (flops)	1.48 M	1.2 K
	Sparsity (%)	90.97	96.58
$\mathbf{A}^2$	Computational cost	0.61×10^6	0.306×10^3
	Time (s)	0.26×10^{-2}	0.07×10^{-2}
	Speed (flops)	2.38 M	4.34 K
	Sparsity (%)	90.96	95.27
$\mathbf{A}^3$	Computational cost	0.61×10^6	0.306×10^3
	Time (s)	0.31×10^{-2}	0.08×10^{-2}
	Speed (flops)	1.95 M	4.06 K
	Sparsity (%)	91	94.35
$\mathbf{A}^4$	Computational cost	0.609×10^6	0.306×10^3
	Time (s)	0.18×10^{-2}	0.04×10^{-2}
	Speed (flops)	3.39 M	6.96 K
	Sparsity (%)	90.9	93.99

Factoring matrix group **B** also produced triangular matrices as shown in Figure 6. The obtained parameters quantities for various sparsities are presented in Table 2. It is seen that, for both factorization methods, the parameters values of 4 matrices in group **B** are only sligthly different. Such results are due to the sparsity slope of matrix group **B** is relatively small compared to sparsity slope of matrix group A (see Figure 3). It can be seen from Figure 6 and Table 2 that supernodal Cholesky factorization of matrix group **B** also resulted in triangular matrices with higher sparsities compared to the results of multifrontal QR. Further observation of the obtained values of computational cost, time consumption, and speed, indicating that supernodal Cholesky has also better performance than the multifrontal QR.

The same as that produced in factoring matrix groups **A** and **B**, factoring matrix group **C** also resulted triangular matrices and parameters quantities as shown in Figure 7 and Table 3 respectively. It is observed from Table 3 that the parameters values as the results of applying multifrontal QR and supernodal Cholesky are not much different. Such results due to the fact that matrix group **C** has smallest sparsity slope. It can be seen from Figure 7 and Table 3 that supernodal Cholesky factorization also resulted triangular matrices with higher sparsities compared to that of multifrontal QR factorization, and hence supernodal Cholesky has better

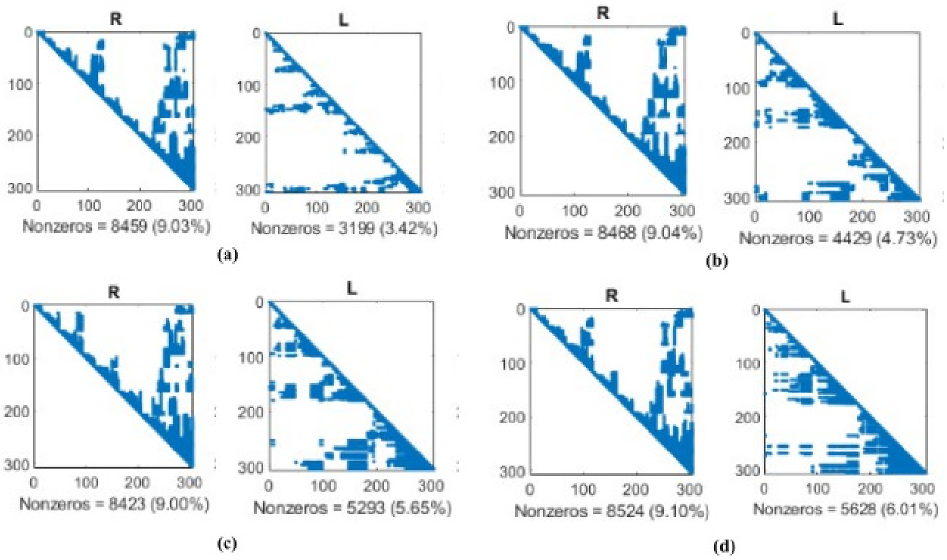

Figure 5. Factorization Results of Matrix Group $\mathbf{A}$: (a) $\mathbf{A}$, (b) $\mathbf{A}^2$, (c) $\mathbf{A}^3$, and (d) $\mathbf{A}^4$.

Table 2. Parameters Values of Matrix Group **B** Factorizations

Sparse Matrix	Parameters	Multifrontal QR	Supernodal Cholesky
$\mathbf{B}$	Computational cost	0.338×10^6	1.138×10^3
	Time (s)	1.63×10^{-2}	0.39×10^{-2}
	Speed (flops)	0.21 M	2.88 K
	Sparsity (%)	99.26	99.74
$\mathbf{B}^2$	Computational cost	0.342×10^6	1.138×10^3
	Time (s)	0.35×10^{-2}	0.08×10^{-2}
	Speed (flops)	0.97 M	14.35 K
	Sparsity (%)	99.26	99.7
$\mathbf{B}^3$	Computational cost	0.338×10^6	1.138×10^3
	Time (s)	0.34×10^{-2}	0.05×10^{-2}
	Speed (flops)	0.99 M	22.22 K
	Sparsity (%)	99.26	99.66
$\mathbf{B}^4$	Computational cost	0.34×10^6	1.138×10^3
	Time (s)	0.37×10^{-2}	0.07×10^{-2}
	Speed (flops)	0.93 M	15.75 K
	Sparsity (%)	99.26	99.59

performance than multifrontal QR.

4. Conclusion

It is clear from the simulations results that the effects of matrix sparsity on the parameters quantities depend on the sparsity slope of the matrix groups, the larger the slope the larger the impact of factorization. In general, supernodal Cholesky

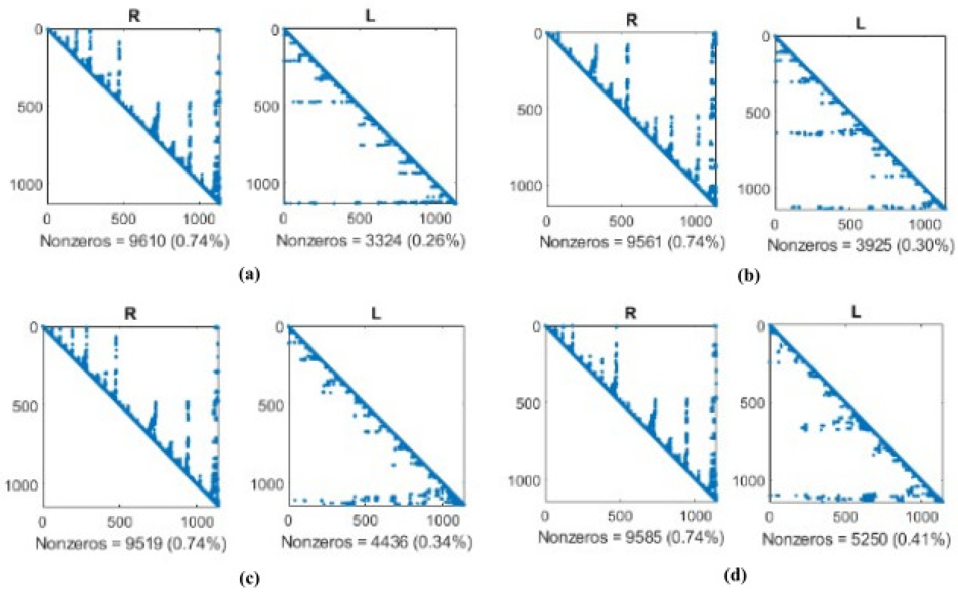

Figure 6. Factorization Results of Matrix Group **B**: (a) **B**, (b) $\mathbf{B}^2$, (c) $\mathbf{B}^3$, and (d) $\mathbf{B}^4$.

Table 3. Parameters Values of Matrix Group **C** Factorizations

Sparse Matrix	Parameters	Multifrontal QR	Supernodal Cholesky
C	Computational cost	1.329×10^9	0.016×10^6
	Time (s)	0.16	0.02
	Speed (flops)	8.49 G	0.74 M
	Sparsity (%)	98.72	99.8
$\mathbf{C}^2$	Computational cost	1.329×10^9	0.016×10^6
	Time (s)	0.16	0.02
	Speed (flops)	8.34 G	0.74 M
	Sparsity (%)	98.67	99.58
$\mathbf{C}^3$	Computational cost	1.351×10^9	0.016×10^6
	Time (s)	0.16	0.04
	Speed (flops)	8.25 G	0.37 M
	Sparsity (%)	98.71	99.56
$\mathbf{C}^4$	Computational cost	1.484×10^9	0.016×10^6
	Time (s)	0.17	0.05
	Speed (flops)	8.92 G	0.3 M
	Sparsity (%)	98.68	99.34

method shows better performances than multifrontal QR. The first method is more efficient in terms computation and is able to produce triangular matrices having higher sparsity, leading to better choice in solving a large system of linear equations.

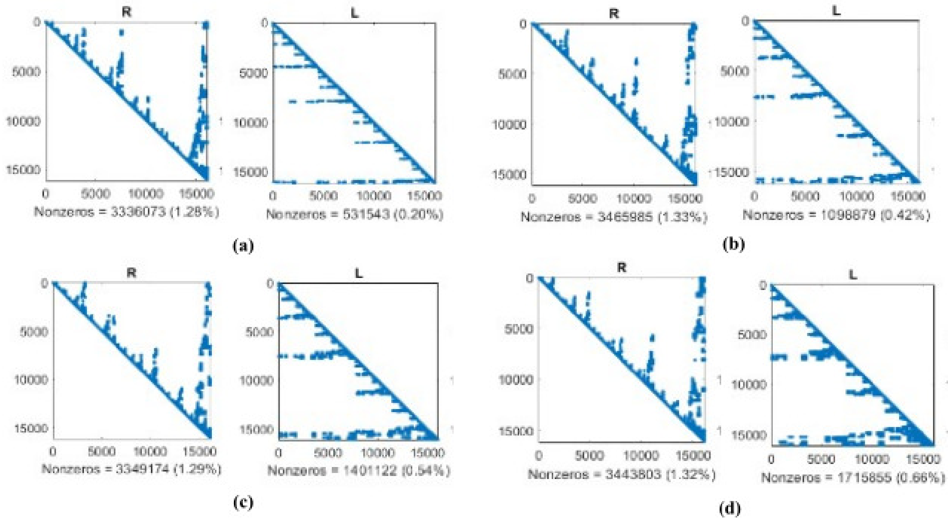

Figure 7. Factorization Results of Matrix Group $\mathbf{C}$: (a) $\mathbf{C}$, (b) $\mathbf{C}^2$, (c) $\mathbf{C}^3$, and (d) $\mathbf{C}^4$.

Bibliography

[1] P. Amestoy, A. Buttari, J.V. L'Excellent, and T. Mary. On the complexity of the block low-rank multifrontal factorization. *SIAM Journal on Scientific Computing*, 39:A1710–A1740, 1 2017.

[2] Y. Zhang, P. Zheng, and W. Luo. Privacy-preserving outsourcing computation of qr decomposition in the encrypted domain. In *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 389–396. IEEE, 8 2019.

[3] L. Yu, Y. Ren, G. Feng, and X. Zhang. Efficient noninteractive outsourcing of large-scale qr and lu factorizations. *Security and Communication Networks*, 2021:1–14, 6 2021.

[4] I.S. Duff and J.K. Reid. The multifrontal solution of indefinite sparse symmetric linear. *ACM Transactions on Mathematical Software*, 9:302–325, 9 1983.

[5] E. Agullo, A. Buttari, A. Guermouche, and F. Lopez. *Multifrontal QR Factorization for Multicore Architectures over Runtime Systems*, pages 521–532. 2013.

[6] S.N. Yeralan, T.A. Davis, W.M. Sid-Lakhdar, and W. Ranka. Algorithm 980. *ACM Transactions on Mathematical Software*, 44:1–29, 6 2018.

[7] G.M. Hebling, J.A.D. Massignan, J.B.A.L. Junior, and M.H.M. Camillo. Sparse and numerically stable implementation of a distribution system state estimation based on multifrontal qr factorization. *Electric Power Systems Research*, 189:106734, 12 2020.

[8] M. Tang, M. Gadou, and S. Ranka. A multithreaded algorithm for sparse cholesky factorization on hybrid multicore architectures. *Procedia Computer Science*, 108:616–625, 2017.

[9] A. Haidar, T. Dong, P. Luszczek, S. Tomov, and J. Dongarra. Retracted: Batched matrix computations on hardware accelerators based on gpus. *The*

International Journal of High Performance Computing Applications, 29:193–208, 5 2015.

[10] D. Zou, Y. Dou, S. Guo, R. Li, and L. Deng. Supernodal sparse cholesky factorization on graphics processing units. *Concurrency and Computation: Practice and Experience*, 26:2713–2726, 11 2014.

[11] S.C. Rennich, D. Stosic, and T.A. Davis. Accelerating sparse cholesky factorization on gpus. *Parallel Computing*, 59:140–150, 11 2016.

[12] Y. Sun, H. Liu, and T Zhou. Sparse cholesky factorization on fpga using parameterized model. *Mathematical Problems in Engineering*, 2017, 1 2017.

[13] E.G. Boman and M.M. Wolf. A nested dissection partitioning method for parallel sparse matrix-vector multiplication. In *2013 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–6. IEEE, 9 2013.

[14] S. Kolodziej, M. Aznaveh, M. Bullock, J. David, T. Davis, M. Henderson, Y. Hu, and R. Sandstrom. The suitesparse matrix collection website interface. *Journal of Open Source Software*, 4:1244, 3 2019.

[15] T.A. Davis. Algorithm 1000. *ACM Transactions on Mathematical Software*, 45:1–25, 12 2019.

[16] T.A. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 1 2006.

[17] I.S. Duff, A.M. Erisman, and J.K. Reid. *Direct Methods for Sparse Matrices*. Oxford University PressOxford, 1 2017.

[18] D. Padua. *Encyclopedia of Parallel Computing*. Springer US, 2011.

[19] A.R. Benson, D.F Gleich, and J. Demmel. Direct qr factorizations for tall-and-skinny matrices in mapreduce architectures. In *2013 IEEE International Conference on Big Data*, pages 264–272. IEEE, 10 2013.

[20] G.R. Lindfield and J.E.T. Penny. *Numerical Methods Using MATLAB*. Elsevier, 4 edition, 2019.