芝浦工業大学

博士学位論文

Studies on Accurate Numerical Computations of Thin QR Decomposition and Verified Numerical Computations for Matrix Equations

Thin QR 分解に対する高精度数値計算法と行列方程式に対する精度保証付き数値計算の研究

令和 2年 3月

Takeshi Terao

寺尾 剛史

Abstract

Numerical computations are widely used in scientific computing and can be performed quickly on modern computers. With the rapid development of computer architecture, the number of cores has increased to achieve high performance in terms of speed. Consequently, parallel computing has been the subject of much research for high-performance computing. However, there are problems involving rounding errors due to finite precision arithmetic. If a problem is ill-conditioned or large-scale, the computed results may be inaccurate due to the accumulation of rounding errors. Therefore, in this thesis we focus on the computational performance of numerical algorithms in terms of speed and accuracy. There are verified numerical computations that produce an approximate solution of a problem and its error bound. In this thesis, we provide the following:

- accurate numerical computations of QR decomposition and their rounding error analysis,
- fast methods proving the nonsingularity of real matrices, and
- verified numerical computations for eigenvalue problems.

QR decomposition of a matrix A is a decomposition of the matrix into a product $A = QR$ of an orthogonal matrix Q and an upper triangular matrix R. QR decomposition is applied to the linear least squares problem and eigenvalue algorithms. In this thesis, we focus on thin QR decomposition (also called economy size QR decomposition or reduced QR decomposition). CholeskyQR is a fast algorithm employed for thin QR decomposition. CholeskyQR2 aims to improve the orthogonality of the Q-factor computed by CholeskyQR. Although Cholesky QR algorithms can be effectively implemented in high-performance computing environments, they are unlike the Householder QR and Gram–Schmidt algorithms, not suitable for ill-conditioned matrices. To address this problem, we apply the concept of LU decomposition to the Cholesky QR algorithms; that is, the principle is to use the LU-factors of a given matrix as preconditioning before applying Cholesky decomposition. We call this method LU-Cholesky QR. We also perform rounding error analysis of the proposed algorithms on the orthogonality and residual of computed the QR-factors. The numerical examples provided in this thesis illustrate the efficiency of the proposed algorithms in parallel computing on both shared and distributed memory computers. In addition, the preconditioning method can be extended to thin QR decomposition in an oblique inner product.

Next, we provide a computer-assisted proof of the nonsingularity of a real and dense matrix, which is an important problem in verified numerical computations, in particular in a system of linear equations. Several verification methods have been proposed using factors of LU decomposition and their approximate inverses. We propose fast and efficient methods using the LU factors and their inverse matrices, and the proposed methods can be extended to the verification of the nonsingularity of interval matrices.

Finally, we focus on verification methods for eigenvalues for large-scale and real symmetric matrices. Solving standard and generalized symmetric eigenproblems is essential for many applications. For large-scale problems, numerical results may be inaccurate; thus, we propose an efficient verification method that provides quantitative error bounds of computed eigenvalues. Because the main cost of the proposed method is devoted to matrix multiplication, the method is expected to have high scalability on large-scale parallel systems. We present numerical results demonstrating the performance of the proposed method in terms of speed and accuracy on the RIKEN K computer and FUJITSU Supercomputer PRIMEHPC FX100. In addition, we provide quantitative error bounds of the computed eigenvalues of problems arising from the physics of a material, in particular, electronic state calculations. We succeed in obtaining verified eigenvalues of large-scale problems up to 10^6 dimensions with a reasonable computational cost.

Contents

I Accurate Numerical Computation of Thin QR Decomposition **5**

1 Introduction **6**
1.1 Introduction . . . 6
1.2 Background . . . 7
1.2.1 Notation . . . 7
1.2.2 Cholesky QR algorithms . . . 7

2 LU-Cholesky QR algorithms for thin QR decomposition **9**
2.1 Proposed algorithms . . . 9
2.2 Rounding error analysis of the proposed algorithms . . . 12
2.2.1 Preliminaries . . . 12
2.2.2 Proof of Theorem I.1 . . . 18
2.2.3 Proof of Corollary I.2 . . . 20
2.2.4 Proof of Theorem I.3 . . . 20
2.2.5 Proof of Theorem I.4 . . . 21
2.3 Numerical results . . . 22
2.3.1 Shared memory computer environments . . . 22
2.3.2 Distributed memory computer environments . . . 25

3 Preconditioned Cholesky QR algorithms in an oblique inner product **30**
3.1 Introduction . . . 30
3.2 Preliminaries . . . 30
3.2.1 Cholesky QR algorithm . . . 30
3.2.2 Refinement of a Q-factor . . . 31
3.2.3 Shifted Cholesky QR algorithm . . . 31
3.3 LU-Cholesky QR algorithms in an oblique inner product . . . 32
3.4 Numerical results . . . 33
3.5 Conclusion for Part I . . . 35

II Fast verification methods of nonsingularity for matrices **36**

4 Introduction **37**
4.1 Introduction . . . 37
4.2 Previous studies . . . 38
4.2.1 Notation . . . 38

4.2.2 A priori error analysis . . . 38
4.2.3 Verification methods . . . 39

5 Proposed verification method using LU-factors and their inverse matrices 44
5.1 Proposed methods . . . 44
5.1.1 Setting of v_L and v_U . . . 47
5.1.2 Algorithm flow . . . 48
5.2 Numerical results . . . 48
5.3 Conclusion of Part Ⅱ . . . 51

III Validated numerical computations of all eigenvalues for large-scale matrices 54

6 Introduction 55
6.1 Preliminaries . . . 55
6.2 Previous studies . . . 56

7 Proposed method 57
7.1 Rounding error analysis . . . 57
7.2 Numerical results . . . 58
7.3 Conclusion of Part Ⅲ . . . 60

Part I

Accurate Numerical Computation of Thin QR Decomposition

Chapter 1

Introduction

1.1 Introduction

In this paper, we propose the LU-Cholesky QR algorithms for thin QR decomposition. Suppose $A \in \mathbb{R}^{m\times n}, m \geq n$ has full column rank. The thin QR decomposition of A such that

$$A = QR, \quad Q \in \mathbb{R}^{m\times n}, \quad R \in \mathbb{R}^{n\times n}$$

is unique where Q has orthogonal columns satisfying $Q^TQ = I$ with I being the identity matrix, and R is an upper triangular matrix with positive diagonal entries. Algorithms employed for thin QR decomposition are proposed, for example Householder QR (cf. e.g. [1, p. 248]), CGS (Classical Gram-Schmidt) [2], MGS (Modified Gram-Schmidt) [2], SVQR (Singular Value QR) [3], CAQR (Communication-Avoiding QR) [4], Cholesky QR [3], and so forth.

Let u denote the unit round-off of floating-point numbers in working precision, for example, $u = 2^{-53}$ for IEEE standard 754 [5] binary64 (so-called "double precision"). Let $\kappa_2(A)$ be the generalized condition number (cf. e.g. [1, p. 284], [6]) of A such that $\kappa_2(A) := \sigma_{\max}(A)/\sigma_{\min}(A)$ if A has full rank, where $\sigma_{\max}(A)$ and $\sigma_{\min}(A)$ are the maximum and the minimum singular values of A, respectively.

The Cholesky QR algorithms, such as CholeskyQR (cf. e.g., [1, Theorem 5.2.3]) and CholeskyQR2 [7], are ideally employed for thin QR decomposition due to their communication avoidance for tall-skinny matrices. CholeskyQR2 first applies CholeskyQR to A and then applies it again to the computed Q-factor to refine the orthogonality. A rounding error analysis of CholeskyQR2 is presented in [8]. Computational kernels of Householder QR, CGS, MGS, and CAQR algorithms are basic linear algebra subprograms (BLAS) -level 1 and -level 2 routines. However, Cholesky QR algorithms can be implemented using primarily BLAS-level 3 and linear algebra package (LAPACK) routines, which reflects their high computational performance and parallelization efficiency. A major drawback of Cholesky QR algorithms involves the squaring of the generalized condition number of a given problem, i.e., $\kappa_2(A^TA) = \kappa_2(A)^2$, since they directly compute a Cholesky decomposition of the matrix A^TA. Let $\hat{B}$ be a computed result of the matrix multiplication A^TA in floating-point arithmetic. If $\kappa_2(A) > \sqrt{u^{-1}}$, then $\kappa_2(\hat{B}) > u^{-1}$ is expected. In this case, the Cholesky decomposition of $\hat{B}$ tends to fail; thus, the Cholesky QR algorithms are not applicable.

To solve the problem, we propose two algorithms for thin QR decomposition using LU decomposition, hereinafter referred to LU-CholeskyQR and LU-CholeskyQR2. Our focus is on Doolittle's LU decomposition of a matrix A such that $PA = LU$, where $L \in \mathbb{R}^{m\times n}$ is a unit lower triangular

matrix, $U \in \mathbb{R}^{n\times n}$ is an upper triangular matrix, and $P \in \mathbb{R}^{m\times m}$ is a permutation matrix. After the LU decomposition of A, Cholesky decomposition is used for the matrix L^TL. Then, it is likely that it runs to completion, since L tends to be fairly well-conditioned even if A is ill-conditioned (cf. e.g. [1, p. 142], [9, p. 297]). This suggests that we can apply CholeskyQR to L even if $\kappa_2(A) > \sqrt{u^{-1}}$ is satisfied. We call this algorithm by LU-CholeskyQR. We also develop an algorithm called LU-CholeskyQR2, which is a refinement of LU-CholeskyQR in terms of the orthogonality of a Q-factor computed by LU-CholeskyQR. This study is related to [A1, B2, C1–C5, C7, C9, D1, D2] in the list of publications.

1.2 Background

1.2.1 Notation

Here, we introduce the notation used in Part I. Let $\mathbb{F}$ be a set of floating-point numbers as defined by IEEE Std. 754 [5]. The notation $fl(\cdot)$ indicates that all operations inside parentheses are evaluated using floating-point arithmetic in round-to-nearest (ties to even) mode. The number of floating-point operations is counted in flops[1]. For simplicity, an expression with only the maximum degree of a polynomial is used for flops. For example, the cost of matrix multiplication for n-by-n matrices is simply represented as $2n^3$ flops instead of $2n^3 - n^2$ flops.

1.2.2 Cholesky QR algorithms

CholeskyQR is a fast algorithm for the thin QR decomposition of full column rank matrices. For a full column rank matrix $A \in \mathbb{R}^{m\times n}$, $m \geq n$, if $B = A^TA = R^TR$, where $R \in \mathbb{R}^{n\times n}$ is the Cholesky-factor of A^TA, then $Q = AR^{-1} \in \mathbb{R}^{m\times n}$ defines the thin QR decomposition of A such that $A = QR$ [1, Theorem 5.2.3]. The thin QR decomposition of A in floating-point arithmetic aims to compute QR-factors such as $A \approx \hat{Q}\hat{R}$, where $\hat{Q} \in \mathbb{F}^{m\times n}$ has approximately orthogonal columns and $\hat{R} \in \mathbb{F}^{n\times n}$ is an upper triangular matrix.

Here, CholeskyQR is introduced in MATLAB-like notations.

Algorithm I.1. *CholeskyQR (cf. e.g. [1, Theorem 5.2.3])*
For a full column rank matrix $A \in \mathbb{F}^{m\times n}$, the following algorithm produces computed thin QR-factors such that $A \approx \hat{Q}_1\hat{R}_1$.

function $[\hat{Q}_1, \hat{R}_1] = \mathrm{CholQR}(A)$
  $\hat{B} = A' * A$; % $\hat{B} = fl(A^TA)$
  $\hat{R}_1 = \mathrm{chol}(\hat{B})$; % *Cholesky decomposition* $\hat{B} \approx \hat{R}_1^T\hat{R}_1$
  $\hat{Q}_1 = A/\hat{R}_1$; % *Solve* $Q_1\hat{R}_1 = A$ *for* Q_1
end

Here, $\mathrm{chol}(\hat{B})$ produces an upper triangular matrix as a computed Cholesky-factor of $\hat{B}$. The computational cost of $\hat{B} = A' * A$ in Algorithm I.1 computed by `dsyrk` in BLAS is mn^2 flops. In addition, the cost of $\mathrm{chol}(\hat{B})$ and $A/\hat{R}_1$ is $n^3/3$ and mn^2 flops, respectively. Therefore, the total cost of Algorithm I.1 is $2mn^2 + n^3/3$ flops. Matrix $\hat{B}$ becomes ill-conditioned if $\kappa_2(A) > \sqrt{u^{-1}}$,

[1]It is not FLOPS, Floating-point Operation Per Second.

and it is likely that the Cholesky decomposition of $\hat{B}$ fails because it produces a square root of a negative number.

The CholeskyQR2 presented next refines the orthogonality of a computed Q-factor $\hat{Q}_1$ obtained by CholeskyQR.

Algorithm I.2. *CholeskyQR2 [7]*
For a full column rank matrix $A \in \mathbb{F}^{m\times n}$, the following algorithm produces computed thin QR-factors such that $A \approx \hat{Q}_2\hat{R}_2$.

function $[\hat{Q}_2, \hat{R}_2] = \mathrm{CholQR2}(A)$
  $[\hat{Q}_1, \hat{R}_1] = \mathrm{CholQR}(A)$; % $A \approx \hat{Q}_1\hat{R}_1$
  $[\hat{Q}_2, \tilde{R}] = \mathrm{CholQR}(\hat{Q}_1)$; % $\hat{Q}_1 \approx \hat{Q}_2\tilde{R}$
  $\hat{R}_2 = \tilde{R} * \hat{R}_1$;
end

The computational cost of Algorithm I.2 is $4mn^2 + n^3$ flops. This algorithm provides computed QR-factors $\hat{Q}_2, \hat{R}_2$ such that $A \approx \hat{Q}_1\hat{R}_1 \approx \hat{Q}_2\tilde{R}\hat{R}_1 \approx \hat{Q}_2\hat{R}_2$. Rounding error analysis of Algorithms I.1 and I.2 is provided in [8]. Here, assuming that the condition

$$\delta := 8\kappa_2(A)\sqrt{(mn + n(n+1))u} \leq 1, \tag{1.1}$$

is satisfied, it holds that

$$\|\hat{Q}_1^T\hat{Q}_1 - I\|_2 \leq \frac{5}{64}\delta^2, \tag{1.2}$$

$$\|\hat{Q}_2^T\hat{Q}_2 - I\|_2 \leq 6(mn + n(n+1))u, \tag{1.3}$$

provided that neither underflow nor overflow occurs in floating-point computations. The orthogonality of $\hat{Q}_2$ is thus more refined than that of $\hat{Q}_1$. As can be seen, the orthogonality of $\hat{Q}_1$ in (1.2) depends on $\kappa_2(A)$, while that of $\hat{Q}_2$ does not; that is, $\|\hat{Q}_2^T\hat{Q}_2 - I\|_2$ is bounded by $\mathcal{O}(u)$ regardless of $\kappa_2(A)$ under condition (1.1). However, as can also be seen from (1.1), the above Cholesky QR algorithms cannot be applied if $\kappa_2(A) > \sqrt{u^{-1}}$.

Chapter 2

LU-Cholesky QR algorithms for thin QR decomposition

2.1 Proposed algorithms

Suppose that $A \in \mathbb{F}^{m\times n}$ has full column rank. Our approach involves avoiding the Cholesky decomposition of A^TA. This is possible using Doolittle's LU decomposition of A such that $PA = LU$, where L is a unit lower triangular matrix, U is an upper triangular matrix, and P is a permutation matrix. It is heuristically expected that L is fairly well-conditioned, even if A is ill-conditioned (cf. e.g. [1, p. 142], [9, p. 297]). Then, if the QR-factors of P^TL are obtained such that $P^TL = QS$ where Q is orthogonal and S is upper triangular, then $A = P^TLU = QSU =: QR$. Here, the QR-factors of P^TL can be efficiently obtained by CholeskyQR through the Cholesky decomposition of $L^TPP^TL = L^TL$. Doolittle's LU decomposition function can work as preconditioning for the Cholesky QR algorithms.

Let $\hat{L}$, $\hat{U}$, and P be the computed LU-factors of A such that $PA \approx \hat{L}\hat{U}$. We then perform the Cholesky decomposition of $fl(\hat{L}^T\hat{L})$ in floating-point arithmetic, which is expected to run to completion. Our algorithms can thus produce computed QR-factors of A even if $\kappa_2(A) > \sqrt{u^{-1}}$, which signifies that our algorithms are applicable to a wider class of problems than the original Cholesky QR algorithms.

The following algorithm, called LU-CholeskyQR, generates computed QR-factors utilizing LU-factors.

Algorithm I.3. *LU-CholeskyQR*
For a full column rank matrix $A \in \mathbb{F}^{m\times n}$, the following algorithm produces computed thin QR-factors

such that $A \approx \hat{Q}_1 \hat{R}_1$.

function $[\hat{Q}_1, \hat{R}_1] = \mathrm{LU\_CholQR}(A)$
  $[\hat{L}, \hat{U}, P] = \mathrm{lu}(A)$; % $PA \approx \hat{L}\hat{U}$ *(P is not used hereafter.)*
  $\hat{B} = L' * L$; % $\hat{B} = fl(\hat{L}^T \hat{L})$
  $\hat{S} = \mathrm{chol}(\hat{B})$; % *Cholesky decomposition* $\hat{B} \approx \hat{S}^T \hat{S}$
  $\hat{R}_1 = \hat{S} * \hat{U}$;
  $\hat{Q}_1 = A/\hat{R}_1$; % *Solve* $\hat{Q}_1 \hat{R}_1 = A$ *for* $\hat{Q}_1$
end

Because the computational cost of the LU decomposition for A is $2mn^2 - n^3/3$ flops, the total cost of Algorithm I.3 is $4mn^2 - n^3/3$ flops.

Next, we propose a variant of the LU-CholeskyQR algorithm.

Algorithm I.4. *A variant of LU-CholeskyQR*
For a full column rank matrix $A \in \mathbb{F}^{m \times n}$*, the following algorithm produces computed thin* QR*-factors such that* $A \approx \hat{Q}_1 \hat{R}_1$.

function $[\hat{Q}_1, \hat{R}_1] = \mathrm{variant\_LU\_CholQR}(A)$
  $[\hat{L}, \hat{U}, P] = \mathrm{lu}(A)$; % $PA \approx \hat{L}\hat{U}$
  $\hat{B} = L' * L$; % $\hat{B} = fl(\hat{L}^T \hat{L})$
  $\hat{S} = \mathrm{chol}(\hat{B})$; % *Cholesky decomposition* $\hat{B} \approx \hat{S}^T \hat{S}$
  $\hat{Q}_1 = (P' * \hat{L})/\hat{S}$; % P *is the permutation matrix*
  $\hat{R}_1 = \hat{S} * \hat{U}$;
end

It should be noted that Algorithm I.1 is applicable for $\hat{L}$ such that

$$A \approx P^T \hat{L}\hat{U} \approx P^T \hat{Q}\hat{R}\hat{U},$$

where $P^T \hat{Q} =: \tilde{Q}_1$ has approximately orthogonal columns, $\hat{R}\hat{U} =: \tilde{R}_1$ is an upper triangular matrix, and $\hat{Q}, \hat{R}$ are QR-factors computed by Algorithm I.1. This achieves numerical stability, even if A is ill-conditioned. In addition, an upper bound of the orthogonality of $\hat{Q}_1$ is given as

$$\|\hat{Q}_1^T \hat{Q}_1 - I\|_2 \leq \kappa_2(\hat{L})^2 (mn + n(n+1))u, \tag{2.1}$$

from (1.2), if $8\kappa_2(\hat{L})\sqrt{(mn + n(n+1))u} \leq 1$. However, the numerical stability of Algorithm I.4 is identical to that of Algorithm I.3, because it depends on the applicability of Cholesky decomposition for $\hat{L}^T \hat{L}$. The residual of the QR-factors computed by Algorithm I.4 is poorer than that of Algorithm I.3, which is stated in Theorem I.4. The upper bounds of the residuals of the QR-factors computed by Algorithms I.3 and I.4 depend on $\|A\|_2$ and $\|L\|_2\|U\|_2$, respectively. Therefore, the QR-factors computed by Algorithm I.3 have a better upper bound of their residuals than that computed by Algorithm I.4. Therefore, Algorithm I.3 is suitable for preconditioning in many cases.

The following algorithm called LU-CholeskyQR2 refines the orthogonality computed by $\hat{Q}_1$ obtained by Algorithm I.3.

Algorithm I.5. *LU-CholeskyQR2*
For a full column rank matrix $A \in \mathbb{F}^{m\times n}$, the following algorithm produces computed thin QR-factors such that $A \approx \hat{Q}_2\hat{R}_2$ using Algorithm I.3.

function $[\hat{Q}_2, \hat{R}_2] = \mathrm{LU\_CholQR2}(A)$
  $[\hat{Q}_1, \hat{R}_1] = \mathrm{LU\_CholQR}(A)$; % $A \approx \hat{Q}_1\hat{R}_1$ *(Algorithm I.3)*
  $[\hat{Q}_2, \tilde{R}] = \mathrm{CholQR}(\hat{Q}_1)$; % $\hat{Q}_1 \approx \hat{Q}_2\tilde{R}$
  $\hat{R}_2 = \tilde{R} * \hat{R}_1$;
end

Thus, the orthogonality of $\hat{Q}_1$ is refined by CholeskyQR as $\hat{Q}_1 \approx \hat{Q}_2\tilde{R}$, and the computational cost of Algorithm I.5 is $6mn^2 + n^3/3$ flops.

In a similar way to the Cholesky QR algorithms, our LU-Cholesky QR algorithms (Algorithms I.3 and I.5) can be implemented with standard numerical linear algebra libraries, such as BLAS and LAPACK on shared memory computers and PBLAS and ScaLAPACK on distributed memory computers. Therefore, our proposed algorithms effectively benefit from highly optimized routines in these libraries, in particular, in parallel computing. However, while Cholesky QR has high grain parallelism, LU-Cholesky QR does not, as it uses LU decomposition with pivoting. Accordingly, in a parallel environment with large communication latency, LU-Cholesky QR can be significantly slower than Cholesky QR.

As a result of rounding error analysis of Algorithms I.3 and I.5 on the orthogonality and residual of the computed QR-factors, we present the following theorems and corollary.

Theorem I.1. *Let $\hat{Q}_1$ be obtained by Algorithm I.3, where neither overflow nor underflow occurs in floating-point operations. Under the assumptions*

$$\delta_L := 8\kappa_2(\hat{L})\sqrt{(mn + n(n+1))u} \le 1, \tag{2.2}$$

$$\delta_{LU} := 64\kappa_2(\hat{L})\kappa_2(\hat{U})n^2u \le 1, \tag{2.3}$$

it holds that

$$\|\hat{Q}_1^T\hat{Q}_1 - I\|_2 \le \frac{1}{8}\max(\delta_{LU}, \delta_L^2). \tag{2.4}$$

Corollary I.2. *Let $\hat{Q}_2$ be obtained by Algorithm I.5, where neither overflow nor underflow occurs in floating-point operations. Under the assumptions* (2.2), (2.3), *and*

$$8\kappa_2(\hat{Q}_1)\sqrt{(mn + n(n+1))u} \le 1,$$

it hold that

$$\|\hat{Q}_2^T\hat{Q}_2 - I\|_2 \le 6.5(mn + n(n+1))u.$$

Theorem I.3. *Let $\hat{Q}_2, \hat{R}_2$ be obtained by Algorithm I.5, where neither overflow nor underflow occurs in floating-point operations. Under the assumptions* (2.2) *and* (2.3),

$$\|\hat{Q}_2\hat{R}_2 - A\|_2 \le 4.09n^2u\|A\|_2. \tag{2.5}$$

Theorem I.4. *Let $\hat{Q}_1, \hat{R}_1$ be obtained by Algorithm I.4, where neither overflow nor underflow occurs in floating-point operations. Under the assumptions* (2.2) *and* (2.3)*,*

$$\|\hat{Q}_1\hat{R}_1 - A\|_2 \le 3.15n^2u\|\hat{L}\|_2\|\hat{U}\|_2. \tag{2.6}$$

Proofs of Theorems I.1, I.3, and I.4 and Corollary I.2 are provided in Section 2.2. From Theorem I.4, the residual norm of the QR-factors of Algorithm I.4 worse, which depends on $\|\hat{L}\|_2\|\hat{U}\|_2 (\gtrsim \|A\|_2)$. Therefore, Algorithm I.3 is superior to Algorithm I.4 for the residual norm, and orthogonality can be refined as in Algorithm I.5. Next, we modify Algorithm I.3.

2.2 Rounding error analysis of the proposed algorithms

We present the rounding error analysis of Algorithms I.3 and I.5 on the orthogonality of the computed Q-factors by providing the proofs of Theorem I.1 and Corollary I.2.

For $X = (x_{ij}), Y = (y_{ij}) \in \mathbb{R}^{m\times n}$, the notation $|X|$ signifies $|X| = (|x_{ij}|) \in \mathbb{R}^{m\times n}$. The inequality $X \le Y$ signifies that $x_{ij} \le y_{ij}$ for all (i, j). We assume that all floating-point operations are performed with unit roundoff u, that neither overflow nor underflow occurs in the floating-point operations, and that no divide-and-conquer methods are used for matrix multiplication.

2.2.1 Preliminaries

We begin with standard rounding error analysis for matrix multiplication, LU decomposition, and triangular systems. This applies to blocking strategies; however, it does not apply to more sophisticated methods, such as those proposed by Strassen [10], Coppersmith and Winograd [11], or Williams [12]. For example, given $A \in \mathbb{F}^{m\times n}$ and $B \in \mathbb{F}^{n\times k}$, the product AB is computed by using mk inner products in dimension n in any order of evaluation.

Lemma I.5 (Jeannerod–Rump [13])**.** *For matrices $A \in \mathbb{F}^{m\times n}$ and $B \in \mathbb{F}^{n\times k}$, a computed result of matrix multiplication $C := fl(AB) \in \mathbb{F}^{m\times k}$ satisfies*

$$|AB - C| \le nu|A||B|.$$

Lemma I.6 (Rump–Jeannerod [14])**.** *Suppose that $\hat{L} \in \mathbb{F}^{m\times n}$ and $\hat{U} \in \mathbb{F}^{n\times n}$ are computed LU-factors of $A \in \mathbb{F}^{m\times n}$. Then,*

$$\hat{L}\hat{U} - A = E_5, \quad |E_5| \le nu|\hat{L}||\hat{U}|.$$

Lemma I.7 (Rump–Jeannerod [14])**.** *Suppose that $\hat{R} \in \mathbb{F}^{n\times n}$ is a computed Cholesky-factor of $A \in \mathbb{F}^{n\times n}$. Then,*

$$\hat{R}^T\hat{R} - A = E_5, \quad |E_5| \le (n+1)u|\hat{R}^T||\hat{R}|.$$

Lemma I.8 (Rump–Jeannerod [14])**.** *For a nonsingular triangular matrix $T \in \mathbb{F}^{n\times n}$ and $B \in \mathbb{F}^{n\times k}$, suppose triangular systems $TX = B$ are solved by forward or backward substitution. Then, a computed solution $\hat{X} \in \mathbb{F}^{n\times k}$ satisfies*

$$T\hat{X} - B = \Delta, \quad |\Delta| \le nu|T||\hat{X}|.$$

For $A \in \mathbb{F}^{m\times n}$, $m \geq n$, suppose that matrices $\hat{L} \in \mathbb{F}^{m\times n}$, $\hat{U} \in \mathbb{F}^{n\times n}$, and $P \in \mathbb{F}^{m\times m}$ are computed by Doolittle's LU decomposition with partial pivoting such that $\hat{L}\hat{U} \approx PA$. From Lemmas II.1, II.2, I.7, and II.3,

$$PA - \hat{L}\hat{U} = E_1, \quad |E_1| \leq nu|\hat{L}||\hat{U}|, \tag{2.7}$$
$$\hat{B} - \hat{L}^T\hat{L} = E_2, \quad |E_2| \leq mu|\hat{L}^T||\hat{L}|, \tag{2.8}$$
$$\hat{S}^T\hat{S} - \hat{B} = E_3, \quad |E_3| \leq (n+1)u|\hat{S}^T||\hat{S}|, \tag{2.9}$$
$$\hat{R}_1 - \hat{S}\hat{U} = E_4, \quad |E_4| \leq nu|\hat{S}||\hat{U}|, \tag{2.10}$$
$$\hat{Q}_1\hat{R}_1 - A = E_5, \quad |E_5| \leq nu|\hat{Q}_1||\hat{R}_1|, \tag{2.11}$$

are satisfied for the matrices in Algorithm I.3. It is known [9, p. 111] that, for $M, N \in \mathbb{R}^{m\times n}$ with $|M| \leq N$,

$$\|M\|_2 \leq \| \, |M| \, \|_2 \leq \sqrt{\operatorname{rank}(M)}\|M\|_2, \tag{2.12}$$
$$\|M\|_2 \leq \|N\|_2. \tag{2.13}$$

Therefore, $E_1, \ldots, E_5$ satisfy the following inequalities.

$$\|E_1\|_2 \leq n^2u\|\hat{L}\|_2\|\hat{U}\|_2, \tag{2.14}$$
$$\|E_2\|_2 \leq mnu\|\hat{L}\|_2^2, \tag{2.15}$$
$$\|E_3\|_2 \leq n(n+1)u\|\hat{S}\|_2^2, \tag{2.16}$$
$$\|E_4\|_2 \leq n^2u\|\hat{S}\|_2\|\hat{U}\|_2, \tag{2.17}$$
$$\|E_5\|_2 \leq n^2u\|\hat{Q}_1\|_2\|\hat{R}_1\|_2. \tag{2.18}$$

Assume that

$$\delta_L := 8\kappa_2(\hat{L})\sqrt{(mn + n(n+1))u} \leq 1, \tag{2.19}$$
$$\delta_{LU} := 64\kappa_2(\hat{L})\kappa_2(\hat{U})n^2u \leq 1 \tag{2.20}$$

are satisfied. From assumptions (2.19) and (2.20), $\hat{L}$ and $\hat{U}$ are nonsingular, and

$$mnu \leq \frac{1}{64}, \quad n(n+1)u \leq \frac{1}{64}.$$

From (2.11), $\hat{Q}_1 = (A + E_5)\hat{R}_1^{-1}$, and

$$\begin{aligned}\hat{Q}_1^T\hat{Q}_1 &= \hat{R}^{-T}(A+E_5)^T(A+E_5)\hat{R}^{-1} \\ &= \hat{R}_1^{-T}A^TA\hat{R}_1^{-1} + \hat{R}_1^{-T}(E_5^TA + A^TE_5 + E_5^TE_5)\hat{R}_1^{-1}.\end{aligned} \tag{2.21}$$

Substituting (2.7) into (2.21) yields

$$\begin{aligned}&\hat{R}_1^{-T}A^TA\hat{R}_1^{-1} \\ &= \hat{R}_1^{-T}(\hat{L}\hat{U} + E_1)^T(\hat{L}\hat{U} + E_1)\hat{R}_1^{-1} \\ &= \hat{R}_1^{-T}\hat{U}^T\hat{L}^T\hat{L}\hat{U}\hat{R}_1^{-1} + \hat{R}_1^{-T}(E_1^T\hat{L}\hat{U} + \hat{U}^T\hat{L}^TE_1 + E_1^TE_1)\hat{R}_1^{-1}.\end{aligned} \tag{2.22}$$

In addition, substituting (2.8) and (2.9) into (2.22), we have

$$\hat{R}_1^{-T}\hat{U}^T\hat{L}^T\hat{L}\hat{U}\hat{R}_1^{-1} = \hat{R}_1^{-T}\hat{U}^T(\hat{S}^T\hat{S} - E_2 - E_3)\hat{U}\hat{R}_1^{-1}. \tag{2.23}$$

From (2.10), we obtain

$$\begin{aligned}\hat{R}_1^{-T}\hat{U}^T\hat{S}^T\hat{S}\hat{U}\hat{R}_1^{-1} &= \hat{R}_1^{-T}(\hat{R}_1 - E_4)^T(\hat{R}_1 - E_4)\hat{R}_1^{-1} \\ &= I - \hat{R}_1^{-T}E_4^T - E_4\hat{R}_1^{-1} + \hat{R}_1^{-T}E_4^TE_4\hat{R}_1^{-1}. \end{aligned}\tag{2.24}$$

Next, we introduce several lemmas.

Lemma I.9. *The matrix $\hat{S}$ in Algorithm I.3 satisfies*

$$\|\hat{S}^{-1}\|_2^2 \le 1.02\sigma_{\min}(\hat{L})^{-2}. \tag{2.25}$$

Proof. From (2.8), (2.9), (2.15), and (2.16),

$$\hat{S}^T\hat{S} = \hat{B} + E_3 = \hat{L}^T\hat{L} + E_2 + E_3 \tag{2.26}$$

and

$$\begin{aligned}\sigma_{\min}(\hat{S})^2 = \sigma_{\min}(\hat{L}^T\hat{L} + E_2 + E_3) &\ge \sigma_{\min}(\hat{L})^2 - \|E_2\|_2 - \|E_3\|_2 \\ &\ge \sigma_{\min}(\hat{L})^2 - mnu\|\hat{L}\|_2^2 - n(n+1)u\|\hat{S}\|_2^2. \end{aligned}\tag{2.27}$$

From (2.15), (2.16), and (2.26),

$$\|\hat{S}\|_2^2 - n(n+1)u\|\hat{S}\|_2^2 \le \|\hat{L}\|_2^2 + mnu\|\hat{L}\|_2^2,$$

and

$$\|\hat{S}\|_2^2 \le \frac{1+mnu}{1-n(n+1)u}\|\hat{L}\|_2^2. \tag{2.28}$$

Hence, from (2.27) and (2.28),

$$\begin{aligned}\sigma_{\min}(\hat{S})^2 &\ge \sigma_{\min}(\hat{L})^2 - mnu\|\hat{L}\|_2^2 - n(n+1)u\frac{1+mnu}{1-n(n+1)u}\|\hat{L}\|_2^2 \\ &\ge \sigma_{\min}(\hat{L})^2 - (mn+n(n+1))u\frac{1+mnu}{1-n(n+1)u}\|\hat{L}\|_2^2.\end{aligned}$$

From the assumption (2.19),

$$64(mn+n(n+1))u\|\hat{L}\|_2^2 \le \sigma_{\min}(\hat{L})^2.$$

Then,

$$\sigma_{\min}(\hat{S})^2 \ge \sigma_{\min}(\hat{L})^2 - \frac{\sigma_{\min}(\hat{L})^2}{64}\cdot\frac{1+1/64}{1-1/64} \ge 0.983\sigma_{\min}(\hat{L})^2. \tag{2.29}$$

□

Lemma I.10. *The matrix $\hat{R}_1$ in Algorithm I.3 satisfies*

$$\|\hat{R}_1^{-1}\|_2 \leq 1.03\sigma_{\min}(\hat{L})^{-1}\sigma_{\min}(\hat{U})^{-1}. \tag{2.30}$$

Proof. From (2.28),

$$\|\hat{S}\|_2^2 \leq \frac{1+mnu}{1-n(n+1)u}\|\hat{L}\|_2^2 \leq \frac{65}{63}\|\hat{L}\|_2^2$$

and from (2.10) and (2.17),

$$\begin{aligned}\sigma_{\min}(\hat{R}_1) &= \sigma_{\min}(\hat{S}\hat{U}+E_4) \geq \sigma_{\min}(\hat{S}\hat{U}) - \|E_4\|_2 \\ &\geq \sigma_{\min}(\hat{S})\sigma_{\min}(\hat{U}) - n^2u\|\hat{S}\|_2\|\hat{U}\|_2.\end{aligned}$$

From these, (2.20), and (2.29),

$$\begin{aligned}\sigma_{\min}(\hat{R}_1) &\geq \sigma_{\min}(\hat{S})\sigma_{\min}(\hat{U}) - n^2u\|\hat{S}\|_2\|\hat{U}\|_2 \\ &\geq \sqrt{0.983}\sigma_{\min}(\hat{L})\sigma_{\min}(\hat{U}) - \sqrt{\frac{65}{63}}n^2u\|\hat{L}\|_2\|\hat{U}\|_2 \\ &\geq 0.99\sigma_{\min}(\hat{L})\sigma_{\min}(\hat{U}) - \sqrt{\frac{65}{63}}\frac{\sigma_{\min}(\hat{L})\sigma_{\min}(\hat{U})}{64} \\ &\geq 0.974\sigma_{\min}(\hat{L})\sigma_{\min}(\hat{U}).\end{aligned}$$

□

Lemma I.11. *The matrix $\hat{R}_1$ in Algorithm I.3 satisfies*

$$\kappa_2(\hat{R}_1) \leq 1.03(1+n^2u)\sqrt{\frac{1+mnu}{1-n(n+1)u}}\kappa_2(\hat{L})\kappa_2(\hat{U}). \tag{2.31}$$

Proof. From (2.10) and (2.17), and (2.28),

$$\|\hat{R}_1\|_2 \leq (1+n^2u)\|\hat{S}\|_2\|\hat{U}\|_2$$

and

$$\|\hat{S}\|_2^2 \leq \frac{1+mnu}{1-n(n+1)u}\|\hat{L}\|_2^2. \tag{2.32}$$

Therefore,

$$\|\hat{R}_1\|_2 \leq (1+n^2u)\sqrt{\frac{1+mnu}{1-n(n+1)u}}\|\hat{L}\|_2\|\hat{U}\|_2.$$

Combining this and Lemma I.10 proves the lemma. □

Lemma I.12. *The matrices $\hat{L}, \hat{U}, \hat{R}_1$ in Algorithm I.3 satisfy*

$$\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2^2 \leq 1.06.$$

Proof. From (2.28) and Lemma I.10,

$$\begin{aligned}\|\hat{S}\|_2\|\hat{U}\|_2\|\hat{R}_1^{-1}\|_2 &\le \sqrt{\frac{1+mnu}{1-n(n+1)u}}\|\hat{L}\|_2\|\hat{U}\|_2\|\hat{R}_1^{-1}\|_2 \\ &\le 1.03\sqrt{\frac{65}{63}}\kappa_2(\hat{L})\kappa_2(\hat{U}).\end{aligned}$$

From this, (2.10), (2.17), and Lemma I.9,

$$\begin{aligned}\|\hat{U}\hat{R}_1^{-1}\|_2 &\le \|\hat{S}^{-1}\|_2(1+\|E_4\|_2\|\hat{R}_1^{-1}\|_2) \\ &\le \|\hat{S}^{-1}\|_2(1+n^2u\|\hat{S}\|_2\|\hat{U}\|_2\|\hat{R}_1^{-1}\|_2) \\ &\le 1.01\sigma_{\min}(\hat{L})^{-1}\left(1+1.03\sqrt{\frac{65}{63}}n^2u\kappa_2(\hat{L})\kappa_2(\hat{U})\right).\end{aligned}$$

From the assumption (2.20),

$$\begin{aligned}\|\hat{U}\hat{R}_1^{-1}\|_2 &\le 1.01\sigma_{\min}(\hat{L})^{-1}\left(1+\frac{1.03}{64}\sqrt{\frac{65}{63}}\right) \\ &\le 1.03\sigma_{\min}(\hat{L})^{-1}.\end{aligned} \tag{2.33}$$

Moreover, from (2.10), (2.23), and (2.24),

$$\begin{aligned}\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2^2 &= \|\hat{R}_1^{-T}\hat{U}^T\hat{L}^T\hat{L}\hat{U}\hat{R}_1^{-1}\|_2 \\ &\le \|\hat{R}_1^{-T}\hat{U}^T\hat{S}^T\hat{S}\hat{U}\hat{R}_1^{-1}\|_2 + \|\hat{R}_1^{-T}\hat{U}^T(E_2+E_3)\hat{U}\hat{R}_1^{-1}\|_2 \\ &\le \|(\hat{R}_1-E_4)\hat{R}_1^{-1}\|_2^2 + \|\hat{U}\hat{R}_1^{-1}\|_2^2(\|E_2\|_2+\|E_3\|_2) \\ &\le (1+\|E_4\|_2\|\hat{R}_1^{-1}\|_2)^2 + \|\hat{U}\hat{R}_1^{-1}\|_2^2(\|E_2\|_2+\|E_3\|_2).\end{aligned}$$

From (2.17), (2.28), and Lemma I.10,

$$\begin{aligned}\|E_4\|_2\|\hat{R}_1^{-1}\|_2 &\le 1.03n^2u\|\hat{S}\|_2\|\hat{U}\|_2\sigma_{\min}(\hat{L})^{-1}\sigma_{\min}(\hat{U})^{-1} \\ &\le 1.03n^2u\frac{1+mnu}{1-n(n+1)u}\kappa_2(\hat{L})\kappa_2(\hat{U}),\end{aligned}$$

and, form (2.15), (2.16), and (2.28),

$$\begin{aligned}\|E_2\|_2+\|E_3\|_2 &\le mnu\|\hat{L}\|_2^2 + n(n+1)u\|\hat{S}\|_2^2 \\ &\le \frac{1+mnu}{1-n(n+1)u}(mn+n(n+1))u\|\hat{L}\|_2^2\end{aligned}$$

Therefore, from these and (2.33),

$$\begin{aligned}\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2^2 \le &\left(1+\frac{1.03\cdot 65}{63}n^2u\kappa_2(\hat{L})\kappa_2(\hat{U})\right)^2 \\ &+\frac{1.07\cdot 65}{63}(mn+n(n+1))u\kappa_2(\hat{L})^2.\end{aligned}$$

Then, from (2.19) and (2.20),

$$\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2^2 \le \left(1 + \frac{1.03 \cdot 65}{64 \cdot 63}\right)^2 + \frac{1.07 \cdot 65}{63 \cdot 64} \le 1.06.$$

□

Lemma I.13. *The matrix $\hat{Q}_1$ computed by Algorithm I.3 satisfies*

$$\|\hat{Q}_1\|_2 \le 1.112 =: \beta. \tag{2.34}$$

Proof. From (2.11) and (2.18),

$$\begin{aligned}\|\hat{Q}_1\|_2 &\le \|A\hat{R}_1^{-1}\|_2 + \|E_5\hat{R}_1^{-1}\|_2 \\ &\le \|A\hat{R}_1^{-1}\|_2 + n^2u\|\hat{Q}_1\|_2\kappa_2(\hat{R}_1),\end{aligned}$$

then

$$(1 - n^2u\kappa_2(\hat{R}_1))\|\hat{Q}_1\|_2 \le \|A\hat{R}_1^{-1}\|_2. \tag{2.35}$$

From (2.14) and (2.22),

$$\begin{aligned}\|A\hat{R}_1^{-1}\|_2^2 &= \|\hat{R}_1^{-T}A^TA\hat{R}_1^{-1}\|_2 \\ &\le \|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2^2 + 2\|E_1\|_2\|R_1^{-1}\|_2\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2 + \|E_1\|_2^2\|\hat{R}_1^{-1}\|_2^2 \\ &\le \|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2^2 + 2n^2u\|\hat{L}\|_2\|\hat{U}\|_2\|\hat{R}_1^{-1}\|_2\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2 \\ &\quad + n^4u^2\|\hat{L}\|_2^2\|\hat{U}\|_2^2\|\hat{R}_1^{-1}\|_2^2.\end{aligned}$$

Here, from Lemmas I.10 and I.12,

$$\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2^2 \le 1.06, \quad \|\hat{R}_1^{-1}\|_2 \le 1.03\sigma_{\min}(\hat{L})^{-1}\sigma_{\min}(\hat{U})^{-1}. \tag{2.36}$$

Hence,

$$\begin{aligned}\|A\hat{R}_1^{-1}\|_2^2 \le 1.06 &+ 2 \cdot 1.03\sqrt{1.06}n^2u\kappa_2(\hat{L})\kappa_2(\hat{U}) \\ &+ (1.03 \cdot n^2u\kappa_2(\hat{L})\kappa_2(\hat{U}))^2.\end{aligned}$$

From the assumption (2.20),

$$\|A\hat{R}_1^{-1}\|_2^2 \le 1.06 + \frac{2 \cdot 1.03\sqrt{1.06}}{64} + \left(\frac{1.03}{64}\right)^2 \le 1.094. \tag{2.37}$$

Also, from (2.20) and (2.31),

$$\begin{aligned}n^2u\kappa_2(\hat{R}_1) &\le 1.03(1 + n^2u)\sqrt{\frac{1 + mnu}{1 - n(n+1)u}}n^2u\kappa_2(\hat{L})\kappa_2(\hat{U}) \\ &\le 1.03 \cdot \frac{65}{64} \cdot \sqrt{\frac{65}{63}} \cdot \frac{1}{64}.\end{aligned}$$

From this, (2.35), and (2.37), we obtain

$$\|\hat{Q}_1\|_2 \le \frac{\|A\hat{R}_1^{-1}\|_2}{1 - n^2u\kappa_2(\hat{R}_1)} \le 1.112. \tag{2.38}$$

□

2.2.2 Proof of Theorem I.1

We estimate $\|\hat{Q}_1^T\hat{Q}_1 - I\|_2$, where $\hat{Q}_1$ is computed by Algorithm I.3. From (2.21), (2.22), and (2.23), let

$$\begin{aligned}
\delta_1 &:= \|\hat{R}_1^{-T}(E_5^T A + A^T E_5 + E_5^T E_5)\hat{R}_1^{-1}\|_2, \\
\delta_2 &:= \|\hat{R}_1^{-T}(-\hat{U}^T(E_2 + E_3)\hat{U} + \hat{U}^T\hat{L}^T E_1 + E_1^T\hat{L}\hat{U} + E_1^T E_1)\hat{R}_1^{-1}\|_2, \\
\delta_3 &:= \|\hat{R}_1^{-T}(-E_4^T\hat{R}_1 - \hat{R}_1^T E_4 + E_4^T E_4)\hat{R}_1^{-1}\|_2.
\end{aligned}$$

Then,

$$\|\hat{Q}_1^T\hat{Q}_1 - I\| \le \delta_1 + \delta_2 + \delta_3.$$

We first estimate δ_1. From (2.18) and (2.34),

$$\|E_5\|_2 \le n^2 u\|\hat{Q}_1\|_2\|\hat{R}_1\|_2 \le \beta n^2 u\|\hat{R}_1\|_2.$$

From this and (2.37),

$$\begin{aligned}
\delta_1 &\le 2\|\hat{R}_1^{-1}\|_2\|E_5\|_2\|A\hat{R}_1^{-1}\|_2 + \|\hat{R}_1^{-1}\|_2^2\|E_5\|_2^2 \\
&\le 2\sqrt{1.094}\beta n^2 u\kappa_2(\hat{R}_1) + \beta^2 n^4 u^2\kappa_2(\hat{R}_1)^2. \qquad (2.39)
\end{aligned}$$

Substituting (2.20) and (2.31) into (2.39),

$$\begin{aligned}
\delta_1 &\le 2.06\sqrt{1.094}\beta n^2 u(1+n^2u)\sqrt{\frac{1+mnu}{1-n(n+1)u}}\kappa_2(\hat{L})\kappa_2(\hat{U}) \\
&\quad + \beta^2 n^4 u^2\left(1.03(1+n^2u)\sqrt{\frac{1+mnu}{1-n(n+1)u}}\kappa_2(\hat{L})\kappa_2(\hat{U})\right)^2 \\
&= \frac{2.06}{64}\sqrt{1.094}\beta(1+n^2u)\sqrt{\frac{1+mnu}{1-n(n+1)u}}\delta_{LU} \\
&\quad + \frac{\beta^2}{64^2}1.03^2(1+n^2u)^2\frac{1+mnu}{1-n(n+1)u}\delta_{LU}^2 \\
&\le 0.0393\delta_{LU} + 0.00035\delta_{LU}^2 \le 0.0397\delta_{LU}. \qquad (2.40)
\end{aligned}$$

We next estimate δ_2. From (2.14), (2.15), (2.16), (2.28), (2.33), and Lemmas I.10 and I.12,

$$\begin{aligned}
\delta_2 &\le \|\hat{U}\hat{R}_1^{-1}\|_2^2(\|E_2\|_2 + \|E_3\|_2) \\
&\quad + 2\|E_1\|_2\|\hat{L}\hat{U}\hat{R}_1^{-1}\|_2\|\hat{R}_1^{-1}\|_2 + \|\hat{R}_1^{-1}\|_2^2\|E_1\|_2^2 \\
&\le 1.03^2\sigma_{\min}(\hat{L})^{-2}(mn\|\hat{L}\|_2^2 + n(n+1)\|\hat{S}\|_2^2)u \\
&\quad + 2.06n^2u\|\hat{L}\|_2\|\hat{U}\|_2 \cdot 1.03\sigma_{\min}(\hat{L})^{-1}\sigma_{\min}(\hat{U})^{-1} \\
&\quad + 1.03^2\sigma_{\min}(\hat{L})^{-2}\sigma_{\min}(\hat{U})^{-2}n^4u^2\|\hat{L}\|_2^2\|\hat{U}\|_2^2 \\
&\le 1.03^2\sigma_{\min}(\hat{L})^{-2}(mn + \frac{(1+mnu)n(n+1)}{1-n(n+1)u})u\|\hat{L}\|_2^2 \\
&\quad + 2.06n^2u\|\hat{L}\|_2\|\hat{U}\|_2 \cdot 1.03\sigma_{\min}(\hat{L})^{-1}\sigma_{\min}(\hat{U})^{-1} \\
&\quad + 1.03^2\sigma_{\min}(\hat{L})^{-2}\sigma_{\min}(\hat{U})^{-2}n^4u^2\|\hat{L}\|_2^2\|\hat{U}\|_2^2 \\
&= 1.03^2(mn + \frac{(1+mnu)n(n+1)}{1-n(n+1)u})u\kappa_2(\hat{L})^2 \\
&\quad + 2.06n^2u \cdot 1.03\kappa_2(\hat{L})\kappa_2(\hat{U}) + 1.03^2n^4u^2\kappa_2(\hat{L})^2\kappa_2(\hat{U})^2 \\
&\le \frac{1.03^2}{64}\frac{1+mnu}{1-n(n+1)u}\delta_L^2 + \frac{2.06 \cdot 1.03}{64}\delta_{LU} + \frac{1.03^2}{64^2}\delta_{LU}^2 \\
&\le 0.0172\delta_L^2 + 0.0333\delta_{LU} + 0.00026\delta_{LU}^2 \\
&\le 0.0172\delta_L^2 + 0.034\delta_{LU}.
\end{aligned} \tag{2.41}$$

We finally estimate δ_3. From (2.17), (2.28), and Lemma I.10,

$$\begin{aligned}
\delta_3 &\le 2\|\hat{R}_1^{-1}\|_2\|E_4\|_2 + \|\hat{R}_1^{-1}\|_2^2\|E_4\|_2^2 \\
&\le 2.06\sigma_{\min}(\hat{L})^{-1}\sigma_{\min}(\hat{U})^{-1}n^2u\|\hat{S}\|_2\|\hat{U}\|_2 \\
&\quad + 1.03^2\sigma_{\min}(\hat{L})^{-2}\sigma_{\min}(\hat{U})^{-2}n^4u^2\|\hat{S}\|_2^2\|\hat{U}\|_2^2 \\
&\le 2.06\sigma_{\min}(\hat{L})^{-1}\sigma_{\min}(\hat{U})^{-1}n^2u\sqrt{\frac{1+mnu}{1-n(n+1)u}}\|\hat{L}\|_2\|\hat{U}\|_2 \\
&\quad + 1.03^2\sigma_{\min}(\hat{L})^{-2}\sigma_{\min}(\hat{U})^{-2}n^4u^2\frac{1+mnu}{1-n(n+1)u}\|\hat{L}\|_2^2\|\hat{U}\|_2^2 \\
&= \frac{2.06}{64}\sqrt{\frac{1+mnu}{1-n(n+1)u}}\delta_{LU} + \frac{1.03^2}{64^2}\frac{1+mnu}{1-n(n+1)u}\delta_{LU}^2 \\
&\le 0.0328\delta_{LU} + 0.000268\delta_{LU}^2 \le 0.034\delta_{LU}.
\end{aligned} \tag{2.42}$$

Thus, combining (2.40), (2.41), and (2.42),

$$\begin{aligned}
\|\hat{Q}_1^T\hat{Q}_1 - I\|_2 &\le \delta_1 + \delta_2 + \delta_3 \\
&\le 0.0397\delta_{LU} + 0.0172\delta_L^2 + 0.034\delta_{LU} + 0.034\delta_{LU} \\
&\le 0.1077\delta_{LU} + 0.0172\delta_L^2 \le 0.1249\max(\delta_{LU}, \delta_L^2) \\
&\le \frac{1}{8}\max(\delta_{LU}, \delta_L^2),
\end{aligned}$$

which proves Theorem I.1.

2.2.3 Proof of Corollary I.2

We estimate $\|\hat{Q}_2^T\hat{Q}_2 - I\|_2$, where $\hat{Q}_2$ is computed by Algorithm I.5. In a similar way to rounding error analysis of CholeskyQR2 [8], an upper bound of $\|\hat{Q}_2^T\hat{Q}_2 - I\|_2$ can be obtained.

From (2.19), (2.20), and (2.4),

$$\|\hat{Q}_1^T\hat{Q}_1 - I\|_2 \leq \frac{1}{8}.$$

Then,

$$\sqrt{1-\frac{1}{8}} \leq \sigma_{\min}(\hat{Q}_1), \quad \sigma_{\max}(\hat{Q}_1) \leq \sqrt{1+\frac{1}{8}}$$

and

$$\kappa_2(\hat{Q}_1) = \frac{\sigma_{\max}(\hat{Q}_1)}{\sigma_{\min}(\hat{Q}_1)} \leq 1.134. \tag{2.43}$$

From the assumption

$$\alpha := 8\kappa_2(\hat{Q}_1)\sqrt{mnu + n(n+1)u} \leq 1,$$

it holds from (1.2) that

$$\|\hat{Q}_2^T\hat{Q}_2 - I\|_2 \leq \frac{5}{64}\alpha^2. \tag{2.44}$$

From this and (2.43),

$$\begin{aligned}\|\hat{Q}_2^T\hat{Q}_2 - I\|_2 &\leq \frac{5}{64}\left(8 \cdot 1.134\sqrt{(mn + n(n+1))u}\right)^2 \\ &\leq 6.5(mn + n(n+1))u,\end{aligned}$$

which proves Corollary I.2.

2.2.4 Proof of Theorem I.3

From Lemmas II.1 and II.3, $\hat{Q}_1$, $\hat{R}_1$, $\hat{Q}_2$, $\hat{R}_2$, and $\tilde{R}$ in Algorithm I.5 (through Algorithms I.1 and I.3) satisfy

$$\hat{Q}_2\tilde{R} - \hat{Q}_1 = E_6, \quad |E_6| \leq nu|\hat{Q}_2||\tilde{R}|, \tag{2.45}$$

$$\tilde{R}\hat{R}_1 - \hat{R}_2 = E_7, \quad |E_7| \leq nu|\tilde{R}||\hat{R}_1|, \tag{2.46}$$

$$\hat{Q}_1\hat{R}_1 - A = E_8, \quad |E_8| \leq nu|\hat{Q}_1||\hat{R}_1|. \tag{2.47}$$

Suppose that $\hat{R}_1$ is non-singular, then $\tilde{R}$ and $\hat{Q}_1$ satisfy

$$\tilde{R} = (\hat{R}_2 + E_7)\hat{R}_1^{-1}, \quad \hat{Q}_1 = (A + E_8)\hat{R}_1^{-1} \tag{2.48}$$

from (2.46) and (2.47). To substitute (2.48) into (2.45), the residual can be estimated as follows.

$$\begin{aligned}\hat{Q}_2(\hat{R}_2 + E_7)\hat{R}_1^{-1} - (A + E_8)\hat{R}_1^{-1} &= E_6 (= \hat{Q}_2\tilde{R} - \hat{Q}_1), \\ \hat{Q}_2(\hat{R}_2 + E_7) - A - E_8 &= E_6\hat{R}_1, \\ \hat{Q}_2\hat{R}_2 - A &= E_6\hat{R}_1 - \hat{Q}_2E_7 + E_8.\end{aligned}$$

With this, the norm of residual is bounded by

$$\begin{aligned}\|\hat{Q}_2\hat{R}_2 - A\|_2 &\le \|E_6\|_2\|\hat{R}_1\|_2 + \|\hat{Q}_2\|_2\|E_7\| + \|E_8\|_2 \\ &\le n^2u(2\|\hat{Q}_2\|_2\|\tilde{R}\|_2 + \|\hat{Q}_1\|_2)\|\hat{R}_1\|_2 \end{aligned} \tag{2.49}$$

from (2.12). Also, $\|\tilde{R}\|_2^2$ satisfies

$$\|\tilde{R}\|_2^2 \le \frac{1+mnu}{1-n(n+1)u}\|\hat{Q}_1\|_2 \le \frac{65}{63}\|\hat{Q}_1\|_2 \tag{2.50}$$

as the same of proof of (2.32), since $\hat{S}$ and $\tilde{R}$ are Cholesky factors of L^TL and $\hat{Q}_1^T\hat{Q}_1$ respectively. From (2.34) and (2.44),

$$\|\hat{Q}_1\|_2 \le 1.112, \quad \|\hat{Q}_2\|_2 \le \sqrt{1+\frac{5}{64}} \le 1.0384.$$

Therefore, from (2.49) and (2.50),

$$\|\hat{Q}_2\hat{R}_2 - A\|_2 \le 3.495n^2u\|\hat{R}_1\|_2. \tag{2.51}$$

Next, we analyze upper bound of $\|\hat{R}_1\|_2$. From Theorem I.1,

$$\sigma_{\min}(\hat{Q}_1) \ge \sqrt{1-\frac{1}{8}} \ge 0.875, \sigma_{\max}(\hat{Q}_1) \le \sqrt{1+\frac{1}{8}} \le 1.125.$$

From this and (2.47), we obtain

$$\begin{aligned}\|\hat{Q}_1\hat{R}_1 - E_8\|_2 &\ge \|\hat{Q}_1\hat{R}_1\|_2 - n^2u\|\hat{Q}_1\|_2\|\hat{R}_1\|_2 \\ &\ge \sigma_{\min}(\hat{Q}_1)\|\hat{R}_1\|_2 - n^2u\|\hat{Q}_1\|_2\|\hat{R}_1\|_2\end{aligned}$$

and

$$\begin{aligned}\|\hat{R}_1\|_2 &\le \frac{\|A\|_2}{\sigma_{\min}(\hat{Q}_1) - n^2u\|\hat{Q}_1\|_2} \\ &\le \frac{\|A\|_2}{0.875 - 1.125n^2u} \le 1.17\|A\|_2.\end{aligned} \tag{2.52}$$

To substitute (2.52) into (2.51),

$$\begin{aligned}\|\hat{Q}_2\hat{R}_2 - A\|_2 &\le 3.495 \cdot 1.17n^2u\|A\|_2 \\ &\le 4.09n^2u\|A\|_2.\end{aligned}$$

2.2.5 Proof of Theorem I.4

From (2.7), (2.10), and (2.12),

$$\hat{L}\hat{U} - PA = E_1, \quad \|E_1\| \le n^2u\|\hat{L}\|_2\|\hat{U}\|_2, \tag{2.53}$$

$$\hat{R}_1 - \hat{S}\hat{U} = E_4, \quad \|E_4\| \le n^2u\|\hat{S}\|_2\|\hat{U}\|_2, \tag{2.54}$$

and from Lemma II.3,

$$\hat{Q}_1\hat{S} - P^T\hat{L} = E_9, \quad \|E_9\|_2 \le n^2 u\|\hat{Q}_1\|_2\|\hat{S}\|_2. \tag{2.55}$$

From (2.28), (2.54), and (2.55), we obtain

$$\|E_4\| \le n^2\frac{1+mnu}{1-n(n+1)u}u\|\hat{L}\|_2\|\hat{U}\|_2 \le \frac{65}{63}n^2u\|\hat{L}\|_2\|\hat{U}\|_2, \tag{2.56}$$

$$\|E_9\| \le \frac{65}{63}n^2u\|\hat{Q}_1\|_2\|\hat{L}\|_2. \tag{2.57}$$

From (2.55),

$$\hat{Q}_1\hat{S}\hat{U} - P^T\hat{L}\hat{U} = E_9\hat{U},$$

and from (2.53) and (2.54),

$$\begin{aligned}\hat{Q}_1(\hat{R}_1 - E_4) - A - P^TE_1 &= E_9\hat{U},\\ \hat{Q}_1\hat{R}_1 - A &= \hat{Q}_1E_4 + P^TE_1 + E_9\hat{U}.\end{aligned}$$

From this, (2.53), (2.56), and (2.57),

$$\begin{aligned}\|\hat{Q}_1\hat{R}_1 - A\|_2 &\le \|\hat{Q}_1\|_2\|E_4\|_2 + \|E_1\|_2 + \|E_9\|_2\|\hat{U}\|_2\\ &\le n^2u\left(2\frac{65}{63}\|\hat{Q}_1\|_2\|\hat{L}\|_2\|\hat{U}\|_2 + \|\hat{L}\|_2\|\hat{U}\|_2\right),\end{aligned}$$

and, from (1.2),

$$\|\hat{Q}_1\hat{R}_1 - A\|_2 \le \left(1 + 2\frac{65}{63}\sqrt{1+\frac{5}{64}}\right)n^2u\|\hat{L}\|_2\|\hat{U}\|_2 \tag{2.58}$$

$$\le 3.15n^2u\|\hat{L}\|_2\|\hat{U}\|_2. \tag{2.59}$$

2.3 Numerical results

Here, we present the numerical results of our proposed algorithms in both shared and distributed memory computing environments.

2.3.1 Shared memory computer environments

We compared the orthogonality of the computed Q-factors, residual norms of the computed QR-factors, and computation times for the above algorithms (Algorithms I.1, I.2, I.3, and I.5) and a standard Householder QR algorithm through numerical examples in the following two shared memory computer environments:

Env. 1 CPU: Intel(R) Core(TM) i7-8550U, 4 cores, Memory: 16 GB, OS: Windows 10, Software: MATLAB R2018a

Env. 2 CPU: Intel(R) Core(TM) i9-7900X, 10 cores, Memory: 128 GB, OS: Windows 10, Software: MATLAB R2018a

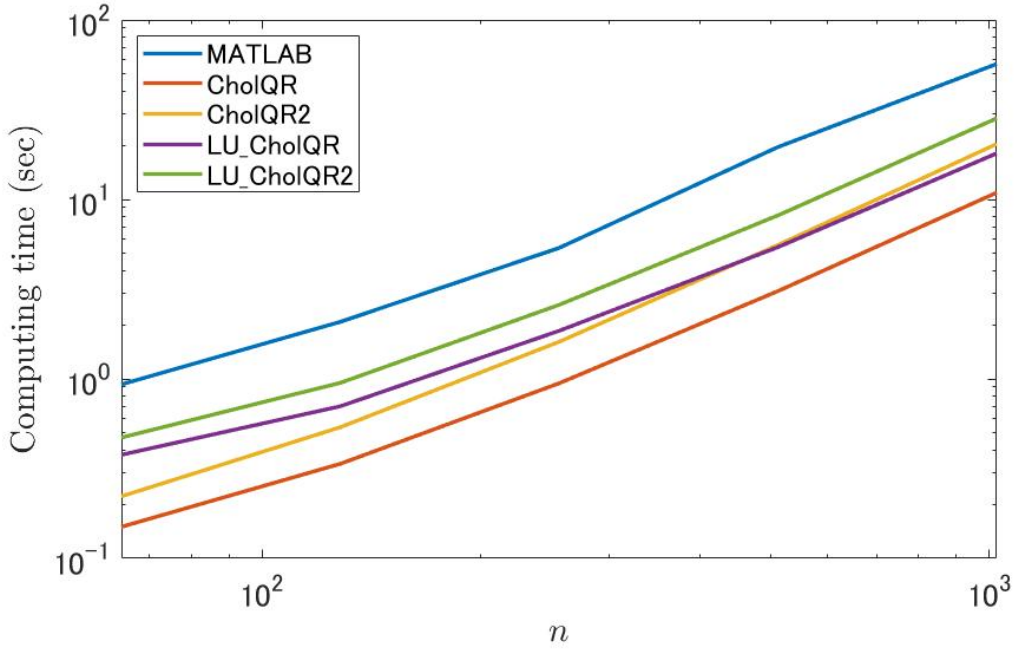

Figure 2.1: Comparison of the computation times for various n with $m = 500{,}000$ in Env. 1

We first compare the computation time, and a test matrix $A \in \mathbb{F}^{m\times n}$ is generated using MATLAB's function as `A = rand(m,n)`. With respect to the computation times, Figs. 2.1 and 2.2 indicate that the proposed algorithms were faster than MATLAB's `qr` function. Moreover, when n reached a certain size, the cost of preconditioning by LU decomposition was relatively less expensive, because the computational performance of LU decomposition was improved for larger n, especially in Env. 2.

Next, we compared the strong-scalability of MATLAB's `qr` function, CholeskyQR2, and LU-CholeskyQR2 algorithms in Env. 2. From Fig. 2.3, strong-scalability of LU-CholeskyQR2 is superior to that of MATLAB's QR, and comparable to that of CholeskyQR2.

Next we compare orthogonality and residual on Env. 1, and generate test matrices using Higham's `randsvd` function [9] as

```
A = gallery('randsvd',[m,n],cnd,mode,m,n,1),
```

where `cnd` is a specified generalized condition number $\kappa_2(A)$, and `mode` can be selected as follows:

1. One large singular value.
2. One small singular value.
3. Geometrically distributed singular values.
4. Arithmetically distributed singular values.
5. Random singular values with uniformly distributed logarithm.

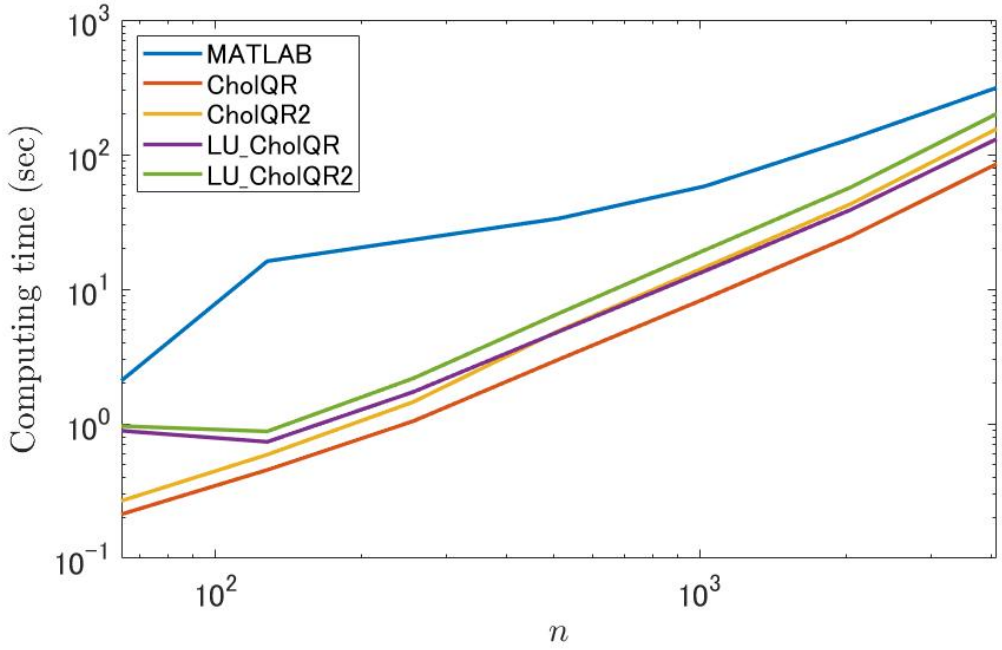

Figure 2.2: Comparison of the computation times for various n with $m = 1{,}000{,}000$ in Env. 2

In numerical examples, we choose `mode` $= 3$ unless otherwise specified. As a standard Householder QR algorithm, we use MATLAB's `qr` function for thin QR decomposition as

```
[Q,R] = qr(A,0).
```

For comparison on the orthogonality $\|\hat{Q}^T\hat{Q} - I\|_2$ and the residual norms $\|\hat{Q}\hat{R} - A\|_2$ for computed QR-factors, we use the Advanpix Multiprecision Computing Toolbox [15] for calculating them precisely.

Figure 2.4 compares the orthogonality as $\|\hat{Q}^T\hat{Q} - I\|_2$, and Fig. 2.5 compares the residuals as $\|\hat{Q}\hat{R} - A\|_2$. As can be seen, the proposed algorithms (LU-CholeskyQR and LU-CholeskyQR2) run to completion even if $\kappa_2(A) > \sqrt{u^{-1}}$. Moreover, the Q-factors computed by LU-CholeskyQR are successfully refined by LU-CholeskyQR2 in terms of orthogonality. It should be noted that neither CholeskyQR nor CholeskyQR2 cannot produce QR-factors if $\kappa_2(A) > \sqrt{u^{-1}}$, as Cholesky decomposition of $fl(A^TA)$ breaks down. Both the orthogonality and the residual norms of the computed QR-factors obtained by LU-CholeskyQR2 are comparable to the results produced by MATLAB's `qr` function.

Moreover, Figs. 2.6 and 2.7 compare the orthogonality and the residual norms for $m = 1024$, $32 \le n \le 1024$, and $\kappa_2(A) \approx 10^7$. Similar to the previous results, both the orthogonality and the residual norms of the computed QR-factors obtained by LU-CholeskyQR2 are comparable to the results produced by MATLAB's `qr` function.

Furthermore, Figs. 2.8 and 2.9 display the orthogonality and the residual norms of the QR-factors computed by LU-CholeskyQR2, respectively, for all `mode` $\in \{1, 2, 3, 4, 5\}$. When $\kappa_2(A) \approx u^{-1}$, the orthogonality becomes slightly worse in the cases of mode $\in \{1, 2\}$. However, the residual norms are still small for all the modes.

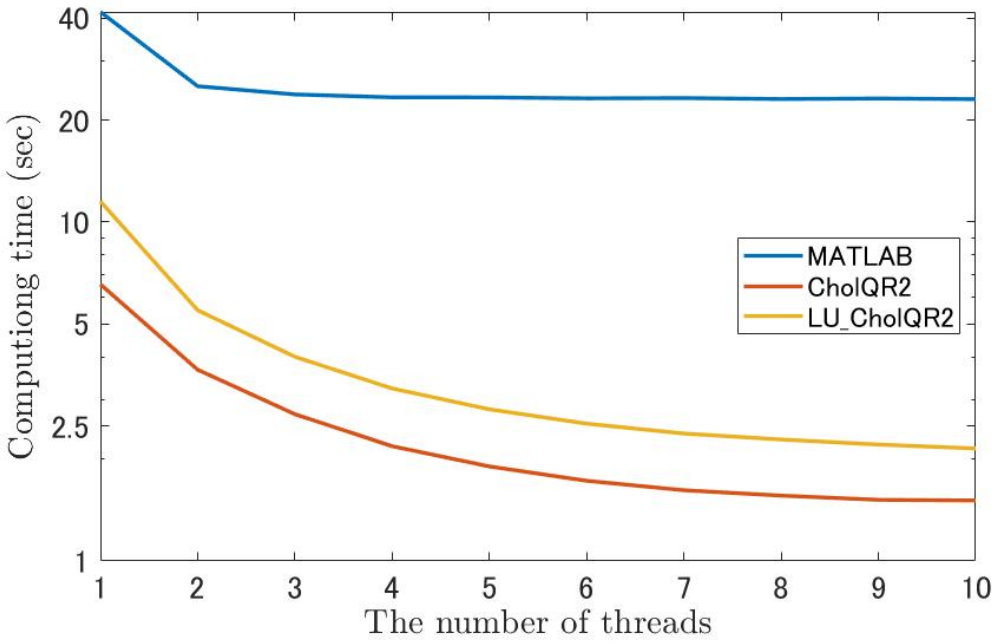

Figure 2.3: Comparison of the computation times for various threads with $m = 1{,}000{,}000, n = 256$ in Env. 2

2.3.2 Distributed memory computer environments

Finally, we show some numerical results on parallel distributed memory computers using RIKEN's K computer and FUJITSU Supercomputer PRIMEHPC FX100. Computing times of the above algorithms are compared in the following two computational environments:

K computer CPU: SPARC6$^{\text{TM}}$ VIIIfx (8 cores), Memory: 16 GB / node

FX100 CPU: SPARC6$^{\text{TM}}$ XIfx (32 cores), Memory: 32 GB (HMC) / node

Here, we use `mpifccpx` as the Fujitsu C compiler command with MPI with the options

```
-Kfast,parallel,openmp -SCALAPACK -SSL2BLAMP
```

on K computer and FX100 in common.

We generate $m \times n$ matrices whose elements are pseudo-random numbers uniformly distributed in the interval $(0, 1)$ with $m = 1{,}048{,}576$, $n = 256$. Note that the generated test matrices are not ill-conditioned so that the Cholesky QR algorithms are applicable.

Tables 2.1 and 2.2 compare the computation times for the Householder QR algorithm (labeled 'HouseholderQR'), CholeskyQR2, and LU-CholeskyQR2. We use the ScaLAPACK routines listed below:

Algorithm	Major ScaLAPACK routines
HouseholderQR	`pdgeqrf`, `pdorgqr`
CholeskyQR2	`dsyrk`, `dpotrf`, `dtrsv`, `dtrmm` $(*)$
LU-CholeskyQR2	`pdgetrf` and all routines in $(*)$

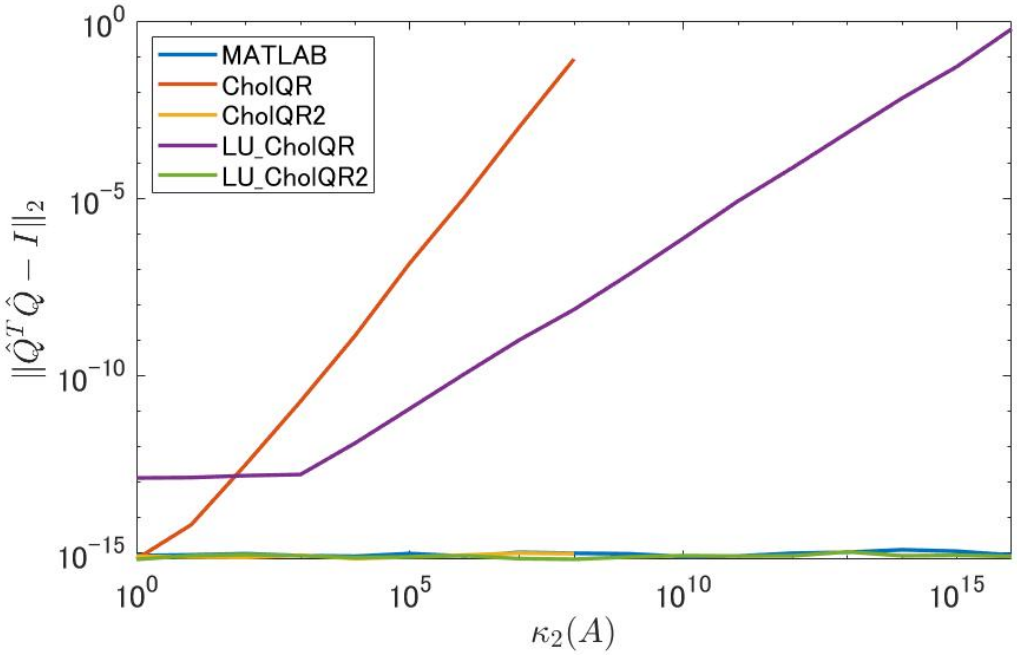

Figure 2.4: Comparison of the orthogonality $\|\hat{Q}^T\hat{Q} - I\|_2$ for various $\kappa_2(A)$ with $m = 1024$, $n = 128$ in Env. 1

As can be seen, CholeskyQR2 is the fastest among them in all cases, as expected. Although the computational performance of Doolittle's LU decomposition is not very high due to partial pivoting, the proposed algorithm (LU-CholeskyQR2) is faster than the Householder QR algorithm (HouseholderQR) in almost all the cases.

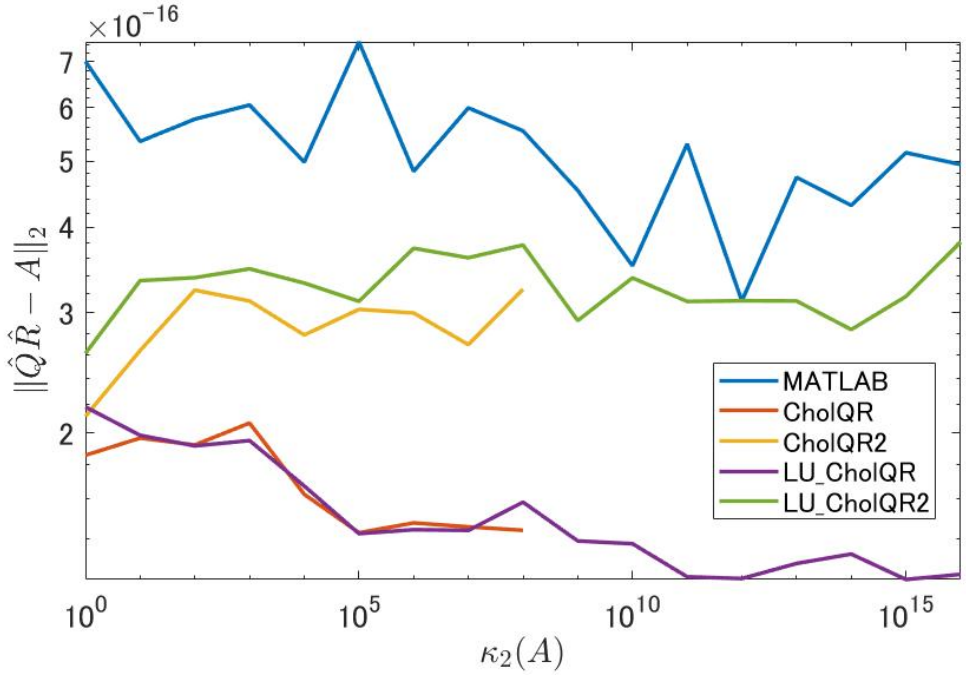

Figure 2.5: Comparison of the residual norms $\|\hat{Q}\hat{R}-A\|_2$ for various $\kappa_2(A)$ with $m = 1,024$, $n = 128$ in Env. 1

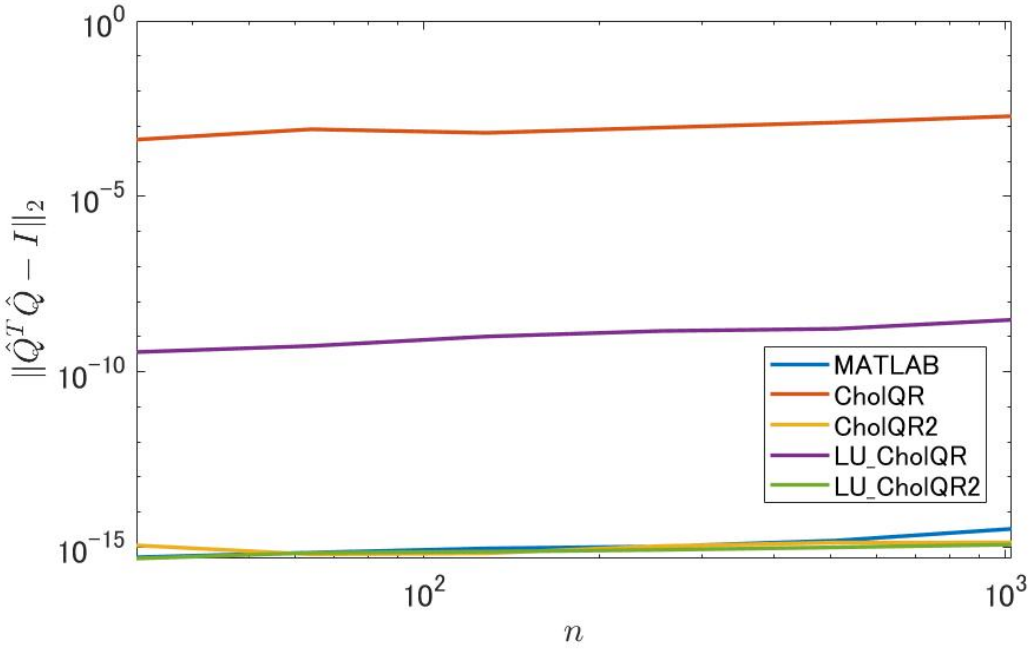

Figure 2.6: Comparison of the orthogonality $\|\hat{Q}^T\hat{Q} - I\|_2$ for various n with $m = 1024$, cnd $= 10^7$ in Env. 1

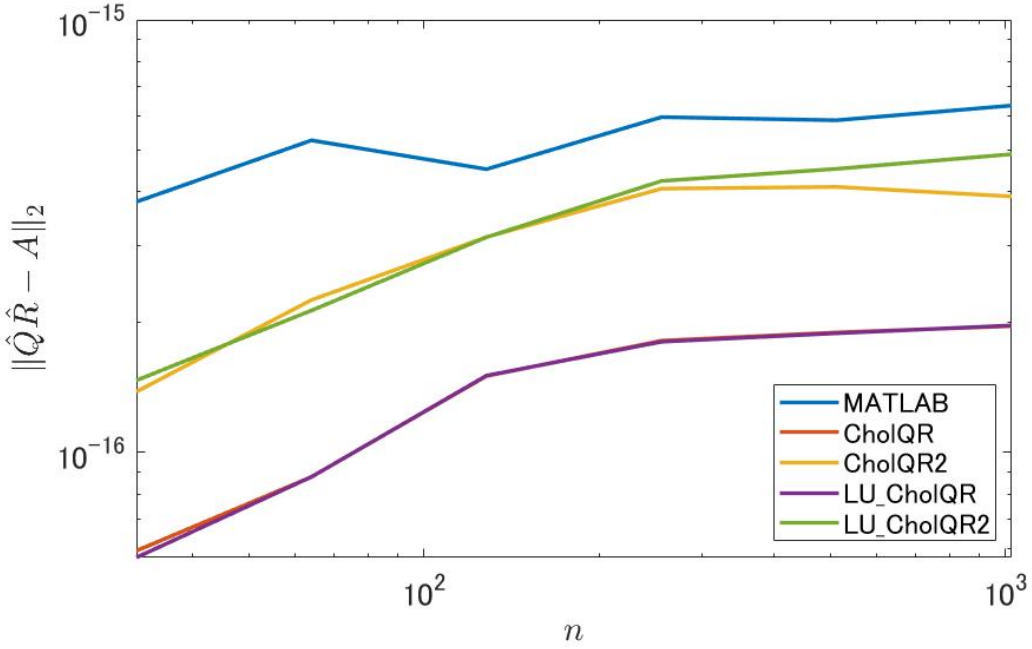

Figure 2.7: Comparison of the residual norms $\|\hat{Q}\hat{R} - A\|_2$ for various n with $m = 1024$, cnd $= 10^7$ in Env. 1

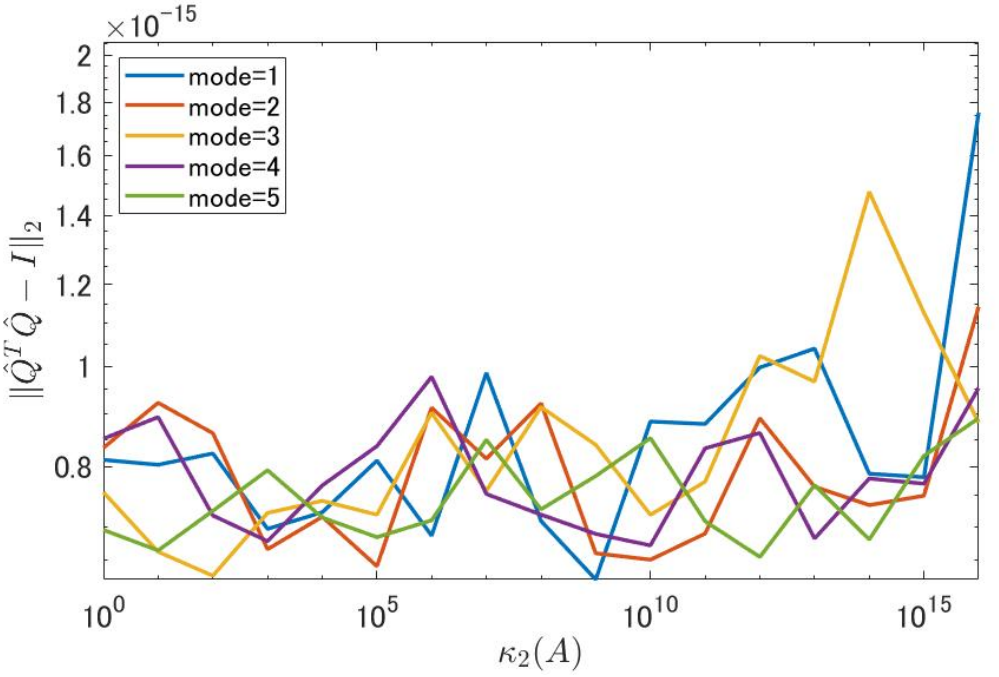

Figure 2.8: The orthogonality $\|\hat{Q}^T\hat{Q} - I\|_2$ for Algorithm I.5 (LU-CholeskyQR2) for various $\kappa_2(A)$ with several singular value distributions with $m = 1024$, $n = 128$

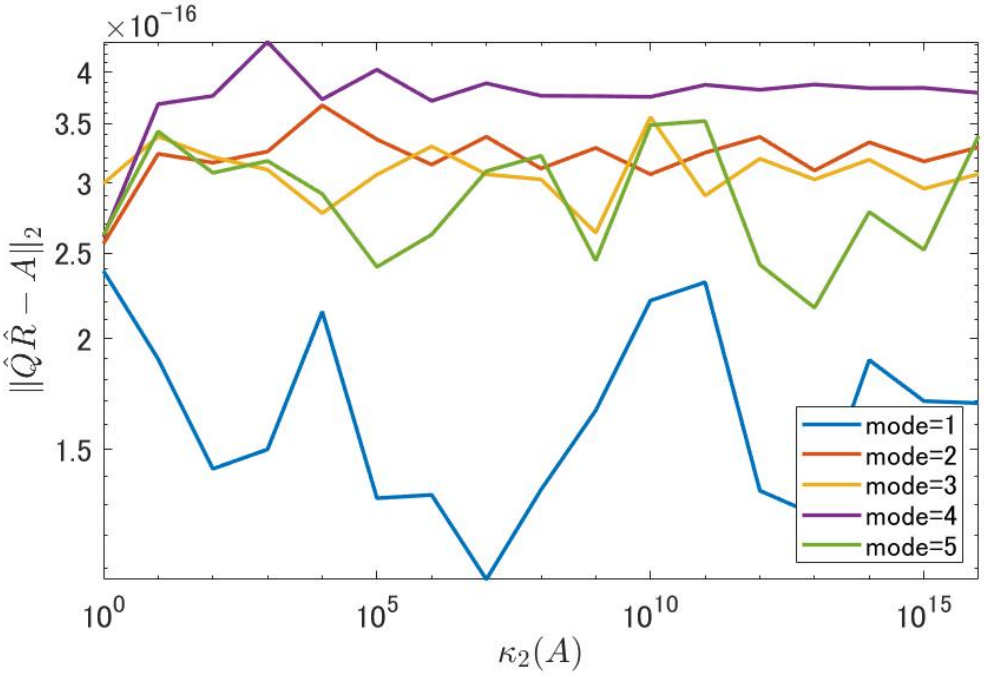

Figure 2.9: The residual norms $\|\hat{Q}\hat{R} - A\|_2$ for Algorithm I.5 (LU-CholeskyQR2) for various $\kappa_2(A)$ with several singular value distributions with $m = 1024$, $n = 128$

Table 2.1: Comparison of computation times (sec) on RIKEN's K computer (m = 1,048,576, $n = 256$)

Algorithm \ # nodes	1	2	4	8	16	32	64	128
HouseholderQR	33.89	27.01	12.51	6.10	3.09	1.64	0.89	0.86
CholeskyQR2	7.78	2.58	1.29	0.67	0.36	0.20	0.14	0.12
LU-CholeskyQR2	23.93	12.39	6.24	2.79	1.48	0.82	0.50	0.43

Table 2.2: Comparison of computation times (sec) on FUJITSU FX100 (m = 1,048,576, $n = 256$)

Algorithm \ # nodes	1	2	4	8	16	32	64
HouseholderQR	15.75	8.12	3.97	2.03	1.09	0.62	0.35
CholeskyQR2	1.79	0.94	0.50	0.27	0.16	0.11	0.13
LU-CholeskyQR2	7.97	3.36	1.74	0.68	0.57	0.42	0.38

Chapter 3

Preconditioned Cholesky QR algorithms in an oblique inner product

3.1 Introduction

In this chapter, we consider thin QR decomposition in an oblique inner product for full rank matrices $A \in \mathbb{R}^{m\times n}, m \geq n$ and $B \in \mathbb{R}^{m\times m}$ with B being positive definite. This decomposition produces B-orthogonal columns $Q \in \mathbb{R}^{m\times n}$ and an upper triangular matrix $R \in \mathbb{R}^{n\times n}$ such that

$$A = QR, \quad Q^T BQ = I, \tag{3.1}$$

where I is the identity matrix. For the QR-factors computed by numerical computations, B-orthogonality as $\|Q^T BQ - I\|$ and residual as $\|QR - A\|$ are significant. Although CholeskyQR has weak numerical stability, Cholesky QR is a fast algorithm employed for thin QR decomposition [16]. In addition, when CholeskyQR runs to completion, we can refine B-orthogonality using CholeskyQR2 [16]. In Part I, we propose the fast and accurate numerical algorithms for this QR decomposition in an oblique inner product using Doolittle's LU decomposition. There are advantages in terms of B-orthogonality, residual, and computation times, that is shown in numerical examples.

3.2 Preliminaries

We first define the notation in Part I. Let $\mathbb{F}$ be a set of binary floating-point numbers, and let u be the unit roundoff (binary64: $u = 2^{-53}$). The 2-norm of vector $x = (x_i) \in \mathbb{R}^n$ and matrix $A = (a_{ij}) \in \mathbb{R}^{m\times n}$ indicates that

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}, \quad \|A\|_2 = \max_{\|x\|_2=1} \|Ax\|_2.$$

Matrix A^+ denotes the Moore-Penrose pseudoinverse matrix of A; that is, $A^+ = (A^T A)^{-1} A^T$. $\kappa_2(A)$ is the condition number such that $\kappa_2(A) = \|A\|_2 \|A^+\|_2$.

3.2.1 Cholesky QR algorithm

In this subsection, we first introduce Cholesky QR algorithms in an oblique inner product for $A \in \mathbb{F}^{m\times n}, B \in \mathbb{F}^{m\times m}$ using MATLAB-like notation.

Algorithm I.6. *CholeskyQR algorithm in an oblique inner product*

function $[Q_1, R_1] = \mathrm{CholQR}(A, B)$
  $C = A' * B * A$; % $C \approx A^T B A$
  $R_1 = \mathrm{chol}(C)$; % $C \approx R_1^T R_1$
  $Q_1 = A/R_1$; % $Q_1 \approx A R_1^{-1}$
end

Since this algorithm is implementable using Level-3 routines in basic linear algebra subprograms (BLAS) and linear algebra package (LAPACK), CholeskyQR achieves high performance on speed. However, the paper [16] reports numerical instability of CholeskyQR. For a matrix C as $\kappa_2(C) \gtrsim u^{-1}$, Cholesky decomposition for C breaks down in many cases. Here, we have

$$\kappa_2(A^T B A) \leq \kappa_2(A)^2 \kappa_2(B). \tag{3.2}$$

Therefore, even if matrices A and B are well-conditioned, there is possibility that C is ill-conditioned. This indicates that CholeskyQR has weak numerical stability.

3.2.2 Refinement of a Q-factor

Next, we consider the refinement after applying CholeskyQR for A and B. The following algorithm named CholeskyQR2 [16] refines B-orthogonality $\|Q^T B Q - I\|_2$.

Algorithm I.7. *CholeskyQR2 algorithm in an oblique inner product*

function $[Q_2, R_2] = \mathrm{CholQR2}(A, B)$
  $[Q_1, R_1] = \mathrm{CholQR}(A, B)$;
  $[Q_2, R] = \mathrm{CholQR}(Q_1, B)$;
  $R_2 = R * R_1$;
end

For $m \gg n$, the cost of CholeskyQR2 is almost twice as much as that of CholeskyQR.

3.2.3 Shifted Cholesky QR algorithm

We introduce the shifted Cholesky QR algorithms [17] whose numerical stability is stronger than that of the standard Cholesky QR algorithms.

Algorithm I.8. *Shifted CholeskyQR algorithm in an oblique inner product*

function $[Q_1, R_1] = \mathrm{sCholQR}(A, B, s)$
  $C = A' * B * A$;
  $R_1 = \mathrm{chol}(C + s * I)$; % *$s$ is a positive constant*
  $Q_1 = A/R_1$;
end

Even if $\kappa_2(C) > u^{-1}$, $\kappa_2(C + sI) \leq u^{-1}$ is satisfied by the diagonal shift. In [17], the amount of shift is calculated as

$$s \approx 11(2m\sqrt{mn} + n(n+1))u\|A\|_2^2\|B\|_2. \tag{3.3}$$

Similarly, the shifted CholeskyQR2 and shifted CholeskyQR3 are introduced as follows [17]:

Algorithm I.9. *Shifted CholeskyQR2 algorithm in an oblique inner product*

function $[Q_2, R_2] = \text{sCholQR2}(A, B, s)$
  $[Q_1, R_1] = \text{sCholQR}(A, B, s)$;
  $[Q_2, R] = \text{CholQR}(Q_1, B)$;
  $R_2 = R * R_1$;
end

Algorithm I.10. *Shifted CholeskyQR3 algorithm in an oblique inner product*

function $[Q_3, R_3] = \text{sCholQR3}(A, B)$
  $[Q_1, R_1] = \text{sCholQR}(A, B, s)$;
  $[Q_3, R] = \text{CholQR2}(Q_1, B)$;
  $R_3 = R * R_1$;
end

3.3 LU-Cholesky QR algorithms in an oblique inner product

In this section, we propose the LU-Cholesky QR algorithm employed for thin QR decomposition in an oblique inner product. We focus on the preconditioning using numerical computations of Doolittle's LU decomposition of a given matrix A such that

$$PA \approx \hat{L}\hat{U},$$

where $\hat{L}$ is a unit lower triangular matrix, $\hat{U}$ is an upper triangular matrix, and P is a permutation matrix. It is known that $\hat{L}$ tends to be fairly well-conditioned even if A is ill-conditioned.

We apply Doolittle's LU decomposition to preconditioning of CholeskyQR.

Algorithm I.11. *LU-CholeskyQR algorithm in an oblique inner product*

function $[Q_1, R_1] = \text{LU\_CholQR}(A, B)$
  $[\hat{L}, \hat{U}, p] = \text{lu}(A)$; % $PA \approx \hat{L}\hat{U}$
  $C = \hat{L}' * B(p, p) * \hat{L}$; % $B(p, p) = PBP^T$
  $R = \text{chol}(C)$;
  $R_1 = R * U$;
  $Q_1 = A/R_1$;
end

If a given matrix A is ill-conditioned, $\kappa_2(A) \geq \kappa_2(L)$, so that the point of this algorithm is that $\kappa_2(\hat{L}^T P B P^T \hat{L}) \lesssim \kappa_2(A^T B A)$ is expected. Hence, even if a matrix A is ill-conditioned, the proposed algorithm for A and B being $\kappa_2(B) < u^{-1}$ can run to completion.

Next, LU-CholesktQR2 algorithm is explained.

Algorithm I.12. *LU-CholeskyQR2 algorithm in an oblique inner product*

function $[Q_2, R_2] = \text{LU\_CholQR2}(A, B)$
  $[Q_1, R_1] = \text{LU\_CholQR}(A, B)$;
  $[Q_2, R] = \text{CholQR}(Q_1, B)$;
  $R_2 = R * R_1$;
end

LU-CholekyQR2 aims to refine B-orthogonality such as the original CholeskyQR2 algorithm introduced in Section 3.2.2.

3.4 Numerical results

Here, we provide the numerical results. Matrices A and B are generated by MATLAB as follows:

$$A = \texttt{gallery}('randsvd', [m, n], \texttt{cndA}, 3, m, n, 1),$$
$$B = \texttt{gallery}('randsvd', m, \texttt{cndB}, 3, m, m, 1).$$

These matrices A and B satisfy $\kappa_2(A) \approx \texttt{cndA}, \kappa_2(B) \approx \texttt{cndB}$ and $\|A\|_2, \|B\|_2 \approx 1$. Therefore, for simplicity, we obtain the shift amount s in (3.3) as $s \approx 11(2m\sqrt{mn} + n(n+1))u$ for sCholQR, sCholQR2, and sCholQR3. Figure 7.2 compares B-orthogonality of the shifted Cholesky QR and LU-Cholesky QR algorithms for various $\kappa_2(B)$ for $\texttt{cndA} = 10^9$ and $\texttt{cndA} = 10^{14}$. The figure indicates that the B-orthogonality of the Q-factor computed by the proposed algorithms is comparable to that computed by the shifted Cholesky QR. From right side in Fig. 7.2, although the standard Cholesky QR algorithms break down when $\kappa_2(B) \gtrsim 10^{10}$ and $\texttt{cndA} = \texttt{10}^{\texttt{14}}$, LU-Choleksy QR algorithms can be applied to ill-conditioned matrices.

Figure 3.2 compares residual of shifted Cholesky QR and LU-Cholesky QR algorithms for various $\kappa_2(B)$ for $\texttt{cndA} = 10^9$ and $\texttt{cndA} = 10^{14}$. The residual of the QR-factors computed by the proposed algorithms is comparable to that computed by the shifted Cholesky QR.

Finally, we compare the computation times for the following random matrices generated by the MATLAB function; $A = \texttt{randn}(m, n)$ and $B = \texttt{randn}(m)$. The computation environment of the computer and MATLAB are as follows:

CPU: Intel Core i7-8550U, Memory: 16 GB, MATLAB R2019a

Figure 3.3 reveals that the computation times of sCholQR, sCholQR2, and sCholQR3 are 1, 2, and 3 times that of the standard CholeskyQR algorithm, respectively. However, the cost of LU decomposition is much lower than that of CholeskyQR algorithm. Hence, computation times of CholeskyQR and LU-CholeskyQR algorithms are comparable.

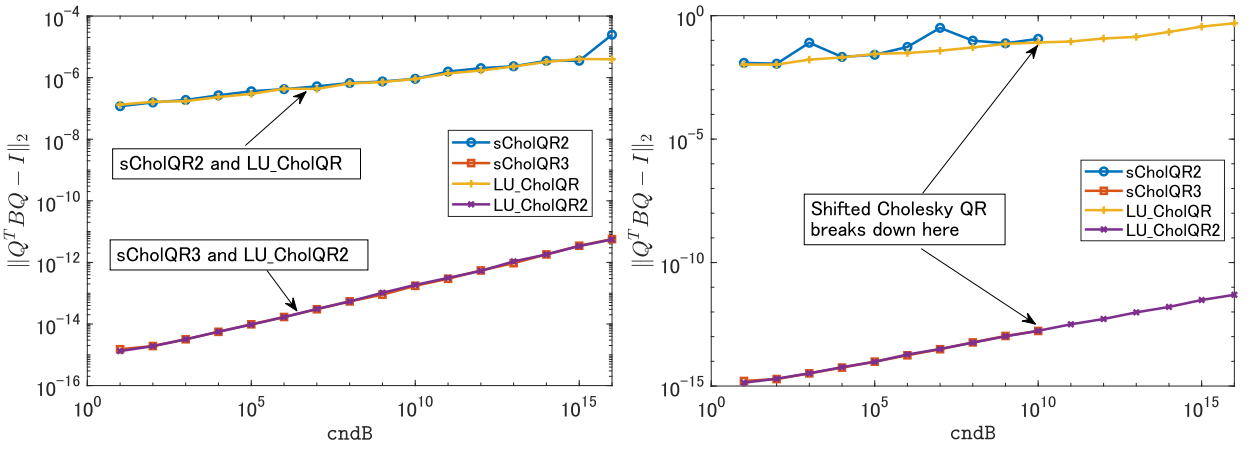

Figure 3.1: Comparison of B-orthogonality ($m = 1024,\ n = 256$, $\texttt{cndA} = 10^9$ (left) and $\texttt{cndA} = 10^{14}$ (right)).

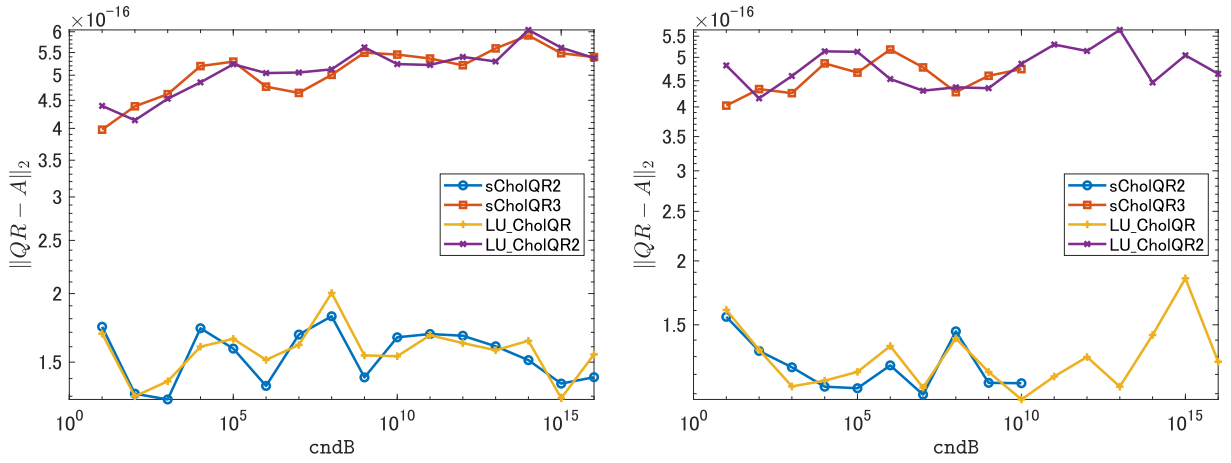

Figure 3.2: Comparison of residual. $m = 1024,\ n = 256$ ($m = 1024,\ n = 256$, $\texttt{cndA} = 10^9$ (left) and $\texttt{cndA} = 10^{14}$ (right)).

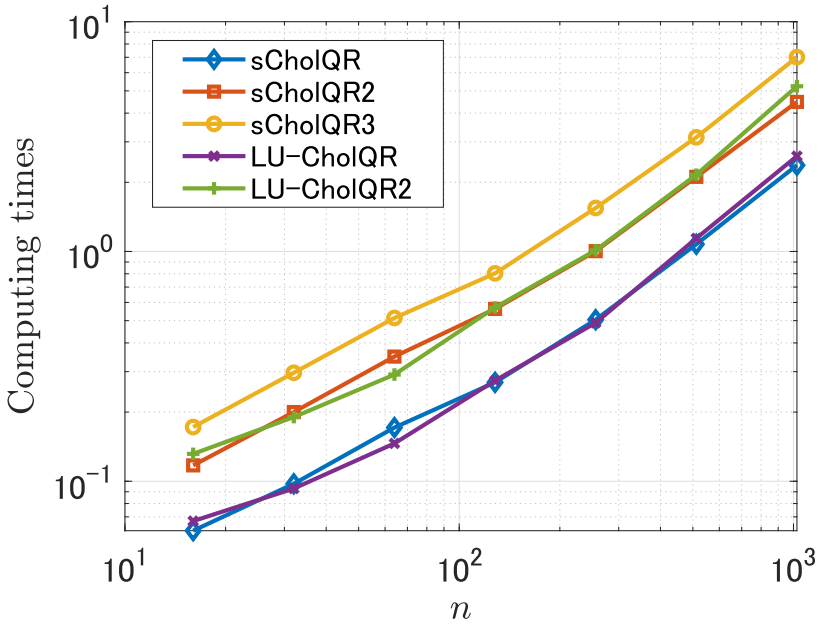

Figure 3.3: Comparison of computation times [sec] for various n. ($m = 10,000$)

3.5 Conclusion for Part I

In Part I, we propose LU-Cholesky QR algorithms for thin QR decomposition to improve the robustness of existing Cholesky QR algorithms. Our investigations in Part I indicate that the proposed algorithms would effectively work for ill-conditioned matrices while Cholesky QR algorithms, such as CholeskyQR and CholeskyQR2 would not be applicable.

With respect to computation time, the comparisons in our numerical examples reveal that

- LU-CholeskyQR2 is faster than the Householder QR algorithm on both the shared memory computers and distributed memory computers used in Part I,
- the computation time for LU-CholeskyQR2 is approximately 1.5 times greater than that of CholeskyQR2 on our shared memory computers, and
- the computation time for LU-CholeskyQR2 is between 3 and 5 times greater than that of CholeskyQR2 on the distributed memory computers used in Part I.

With respect to the orthogonality and norms of residuals, LU-CholeskyQR2 is comparable to CholeskyQR2 and the Householder QR algorithm.

We also presented the results of rounding error analysis of the proposed algorithms in a similar way to the Cholesky QR algorithms.

Moreover, We proposed the preconditioned Cholesky QR algorithms for thin QR decomposition in an oblique inner product. The cost of preconditioning is significantly smaller than the cost of the standard CholeskyQR algorithm. In addition, the numerical stability of the proposed algorithms is superior to that of the shifted Cholesky QR algorithms. Thus, the proposed algorithms are practical due to their high computational performance in speed, accuracy, and stability.

Part II

Fast verification methods of nonsingularity for matrices

Chapter 4

Introduction

4.1 Introduction

The goal of this chapter is to propose fast methods for proving the nonsingularity of a matrix $A \in \mathbb{R}^{n\times n}$ using only numerical computations based on IEEE 754 [5]. To prove the nonsingularity of a given matrix, one of the following may be demonstrated:

- that the determinant is not zero,
- that the matrix inverse exists,
- that there are no zero eigenvalues.

If numerical computations are used for these approaches, rounding error problems arise. For example, we cannot obtain the exact determinant, inverse matrix, and eigenvalues due to the accumulation of rounding errors. We aim to prove the nonsingularity of a given matrix using only floating-point arithmetic.

It is known that the matrix A is nonsingular if there exists a matrix $R \in \mathbb{R}^{n\times n}$ such that

$$\|RA - I\| < 1, \tag{4.1}$$

where I is the identity matrix. This theory is often used for computer-assisted proofs of nonsingularity of matrices. It is difficult to rigorously compute $\|RA - I\|$ using floating-point arithmetic due to rounding error problem. Therefore, an upper bound of $\|RA - I\|$ can be computed. Proving $\|RA - I\| < 1$ is also essential for verifying numerical computations for linear systems. Let a linear system be $Ax = b,\ x, b \in \mathbb{R}^n$ and an approximate solution of $Ax = b$ be $\hat{x}$. If $\|RA - I\| < 1$, then the upper bound of the error is bounded by

$$\|\hat{x} - x\| \leq \frac{\|R(b - A\hat{x})\|}{1 - \|RA - I\|}.$$

In Part II, we use the maximum norm and focus on how to obtain upper the bound of $\|RA - I\|_\infty$ using only floating-point arithmetic.

There are two strategies for setting R. One is using LU factors and their approximate inverses [18, 19, 20, 21]. Let $\hat{L}$ and $\hat{U}$ be the computed LU factors of PA, where P is a permutation matrix, that is, $PA \approx \hat{L}\hat{U}$. Matrices X_L and X_U are the approximate inverse matrices of $\hat{L}$ and

$\hat{U}$, respectively. Then, R is set by $R := X_U X_L P$. The second strategy is to compute R as the approximate inverse of a matrix A directly [18, 19].

In this chapter, we propose four new methods for obtaining the upper bound of $\|RA - I\|_\infty$ by setting $R := (\hat{L}\hat{U})^{-1}P$. These methods have advantages for proof of nonsingularity of interval matrices. Numerical results illustrate their efficiency. In addition, the proposed methods often result in superior upper bounds of $\|RA - I\|_\infty$ although their computational cost is lower than that of previous studies.

The remainder of this chapter is organized as follows. We introduce the target problem and present an overview of previous studies in Section 4.2.1. In Section 4.2.2, we summarize notations and lemmas for rounding error analysis. In addition, we describe previous studies on obtaining the upper bound of $\|RA - I\|_\infty$. In Section 4.2.3, we introduce new methods and their extension to interval matrices with numerical examples. This study is related to [B1, C10–C22] in the list of publications.

4.2 Previous studies

4.2.1 Notation

Here, we introduce the notation used in Part II. Let $\mathbb{F}$ be a set of binary floating-point numbers as defined by IEEE Std. 754 [5]. Notations $fl(\cdot), fl_\bigtriangledown(\cdot)$, and $fl_\bigtriangleup(\cdot)$ indicate that all operations inside parentheses are evaluated using floating-point arithmetic with the following rounding modes: rounding to the nearest (roundTiesToEven), rounding downward (roundTowardNegative) and upward (roundTowardPositive), respectively. Let u be the unit roundoff and u_s be the minimum positive value in floating point numbers, for example, $u = 2^{-53}, u_s = 2^{-1074}$ for binary64 in IEEE Std. 754. For $x, y \in \mathbb{R}^n$, the notation $|x|$ is defined as $|x| = (|x_1|, |x_2|, \ldots, |x_n|)^T$ and $x < y$ signifies $x_i < y_i$ for all i. This notation can easily be extended to matrices. $E \in \mathbb{F}^{n\times n}$ and $e \in \mathbb{F}^n$ are an n-by-n matrix and n-vector of ones, respectively, while matrices E_L and E_U are lower and upper triangular matrices whose all elements are 1, respectively. As in Section 1.2.1, we also count the number of floating-point operations in flops[1]. For simplicity, we only use the maximum degree of a polynomial for flops. For example, the number of floating-point operations of a product of $A, B \in \mathbb{F}^{n\times n}$ is considered by $2n^3$ flops excluding the $\mathcal{O}(n^2)$ terms. For a matrix $A \in \mathbb{R}^{n\times n}$, $\mathrm{diag}(A)$ represents the vector $(a_{11}, \ldots, a_{nn})^T \in \mathbb{R}^n$. For $A, B \in \mathbb{F}^{n\times n}$, a function $\max(A, B)$ returns a matrix with the largest elements taken from A or B.

4.2.2 A priori error analysis

Here, we perform standard rounding error analysis for matrix multiplication, LU decomposition, and triangular systems. For $A \in \mathbb{F}^{n\times n}$, suppose that matrices $\hat{L}$ and $\hat{U}$ are computed by several variants of Gaussian elimination with partial pivoting, where pivoting information is stored in permutation matrix P such that $\hat{L}\hat{U} \approx PA$.

Lemma II.1 ([13]). *For $A, B \in \mathbb{F}^{n\times n}$, a computed result of matrix multiplication $C := \mathtt{fl}(AB)$ satisfies*

$$|AB - C| \le nu|A||B| + \frac{nu_s}{2}E.$$

[1]It is not FLOPS, Floating-point Operation Per Second.

Lemma II.1 applies to blocking strategies, however, it does not apply to more sophisticated methods, such as those proposed by Strassen [10], Coppersmith and Winograd [11], or Williams [12]. For example, given $A \in \mathbb{F}^{m\times n}$ and $B \in \mathbb{F}^{n\times k}$, let C be the approximation of their product AB returned by using mk inner products in dimension n in any order of evaluation.

The following lemmas are related to the residual for LU decomposition and triangular systems. We introduce results from [14] with an underflow term from [18].

Lemma II.2 ([14] and [18]). *For $A \in \mathbb{F}^{n\times n}$, suppose that matrices $\hat{L}$ and $\hat{U}$ are the computed LU factor. If $nu < 1$, then, also in the presence of underflow,*

$$\hat{L}\hat{U} - PA = \Delta_A, \quad |\Delta_A| \le nu|\hat{L}||\hat{U}| + \frac{u_s}{1-nu}(ne + \mathrm{diag}(|U|))e^T.$$

Lemma II.3 ([14] and [18]). *Let matrix equation $TX = B$, where $T \in \mathbb{F}^{n\times n}$ is a nonsingular triangular matrix, be solved by forward / backward substitution. If $nu < 1$, then the computed solution $\hat{X}$ satisfies*

$$T\hat{X} = B + \Delta, \quad |\Delta| \le nu|T||\hat{X}| + \frac{u_s}{1-nu}(nI + D)E_T,$$

where D is the diagonal matrix of T. If T is a lower triangular matrix, then $E_T := E_L$. Otherwise, $E_T := E_U$. If matrix T is a unit triangular matrix, then

$$|\Delta| \le nu|T||\hat{X}| + \frac{nu_s}{1-nu}E_T$$

is satisfied.

Lemma II.4 ([22, p. 145]). *If $T = \{t_{ij}\}$ is a nonsingular triangular matrix, then*

$$|T^{-1}| \le M(T)^{-1},$$

where the triangular matrix $M(T) = \{m_{ij}\}$ is the comparison matrix of T:

$$m_{ij} = \begin{cases} |t_{ii}|, & i = j \\ -|t_{ij}|, & i \ne j \end{cases}.$$

4.2.3 Verification methods

In this subsection, we introduce previous studies on computing the upper bound of $\|RA-I\|_\infty$. First, we introduce verification methods for computing an approximate inverse matrix of A. Assuming that R is a computed inverse matrix, the computational cost of obtaining R is $2n^3$ flops. Then, an upper bound of $|RA - I|$ can be obtained by

$$|RA - I| \le \max(\mathtt{fl}_\bigtriangledown(|RA - I|), \mathtt{fl}_\bigtriangleup(|RA - I|)). \tag{4.2}$$

Next, we introduce an algorithm based on (4.2). All algorithms in this chapter are written in MATLAB-like style. It should be noted that we use the absolute value $|\cdot|$ for matrices instead of abs$(\cdot)$ and omit operation "$*$" for simplicity.

Algorithm II.1 (Oishi-Rump [18]). *For $A \in \mathbb{F}^{n\times n}$, the following algorithm computes an upper bound of $\|RA - I\|_\infty$.*

function $res = \mathrm{Method1}(A)$
  $R = \mathrm{inv}(A)$; % *R is an approximate inverse of A*
  $\mathrm{feature}('setround', -\mathrm{inf})$; % *Change the rounding mode to rounding downward*
  $S_1 = |RA - I|$;
  $\mathrm{feature}('setround', \mathrm{inf})$; % *Change the rounding mode to rounding upward*
  $S_2 = |RA - I|$;
  $S = \max(S_1, S_2)$;
  $res = \mathrm{norm}(S, \mathrm{inf})$; % *$res \geq \|S\|_\infty$*
end

Algorithm II.1 computes the approximate inverse performs matrix and performs two matrix multiplications. Therefore, the computational cost of Algorithm II.1 is $6n^3$ flops.

Here, we introduce a faster method. From Lemma II.1, an upper bound of $|RA - I|$ can be obtained by

$$|RA - I|e \leq \mathtt{fl}(|RA - I|)e + (n+1)u(|R|(|A|e) + e) + \frac{nu_s}{2}Ee.$$

Thus, we have an upper bound of $\|RA - I\|_\infty$ as

$$\begin{aligned} \|RA - I\|_\infty &= \||RA - I|e\|_\infty \leq \|\mathtt{fl}(|RA - I|)e + (n+1)u(|R|(|A|e) + e)\|_\infty + n^2u_s/2 \\ &\leq \|\mathtt{fl}_\triangle(\mathtt{fl}(|RA - I|)e + (n+1)u(|R|(|A|e) + e)\|_\infty + n^2u_s/2). \end{aligned} \tag{4.3}$$

We introduce an algorithm based on (4.3) using directed rounding[2].

Algorithm II.2. *Let $A, R \in \mathbb{F}^{n\times n}$. This algorithm computes an upper bound of $\|RA - I\|_\infty$*

function $res = \mathrm{Method2}(A)$
  $n = \mathrm{size}(A, 1)$;
  $e = \mathrm{ones}(n, 1)$;
  $R = \mathrm{inv}(A)$; % *R is an approximate inverse of A*
  $S = |RA - I|$;
  $\mathrm{feature}('setround', \mathrm{inf})$; % *Change the rounding mode to rounding upward*
  $T = Se + (n+1)u(|R|(|A|e) + e) + n^2u_s/2$;
  $res = \mathrm{norm}(T, \mathrm{inf})$;
end

Algorithm II.2 computes the approximate inverse matrix and performs matrix multiplication. The computational cost of Algorithm II.2 is $4n^3$ flops, which is smaller than that of Algorithm II.1. However, res obtained by Algorithm II.1 is often significantly smaller than that obtained by Algorithm II.2.

[2]In the original paper [19], the upper bound of $\|RA - I\|_\infty$ is obtained using only rounding to the nearest mode. In this paper, we use direct rounding for simplicity.

Next, we introduce verification methods using LU factors as $PA \approx \hat{L}\hat{U}$ and their inverse matrices $X_L \approx L^{-1}$ and $X_U \approx U^{-1}$. Suppose that $\hat{L}$ and $\hat{U}$ are computed LU factors that satisfy Lemma II.2. X_L and X_U are computed inverse matrices of $\hat{L}$ and $\hat{U}$, respectively by a successive solution of $\hat{L}^T x = e_i$, $\hat{U}^T x = e_i$ in any order of evaluation and satisfy Lemma II.3. Here, we define a function computing X_L and X_U as follows.

Algorithm II.3. *The following function returns the LU factors and their approximate inverse matrices.*

function $[\hat{L}, \hat{U}, p, X_L, X_U] = \mathrm{invlu}(A)$
  $I = \mathrm{eye}(\mathrm{size}(A))$; *% I is the identity matrix*
  $[\hat{L}, \hat{U}, p] = \mathrm{lu}(A, \ 'vector')$; *% LU decomposition* $A(p,:) \approx \hat{L}\hat{U}$
  $X_L = I/\hat{L}$; *% Solve* $X_L\hat{L} = I$ *for* X_L
  $X_U = I/\hat{U}$; *% Solve* $X_U\hat{U} = I$ *for* X_U
end

It should be noted that if we use $X_L = I/\hat{L}$ and $X_U = I/\hat{U}$, then the computational cost is n^3 flops for both. Thus, we implement original codes for $I/\hat{L}$ and $X_U = I/\hat{U}$ in the numerical examples. The cost of Algorithm II.3 is $4/3\ n^3$ flops because LU decomposition involves $2/3\ n^3$ flops and solving a triangular system requires $1/3\ n^3$ flops.

Next, we introduce several lemmas pertaining to the upper bounds of $|RA - I|$ with $R := X_U X_L P$.

Lemma II.5 (Oishi-Rump [18])**.** *Let $\hat{L}, \hat{U}$ be the computed LU factors of $A \in \mathbb{F}^{n\times n}$, P be the permutation matrix, and X_L, X_U be approximate inverse matrices of $\hat{L}, \hat{U}$ using Algorithm II.3. Then, including possible underflow, the bounds for $\|X_U X_L PA - I\|_\infty$ can be obtained by*

$$\|X_U X_L PA - I\|_\infty \le nu\|2|X_U||X_L||\hat{L}||\hat{U}| + |X_U||\hat{U}|\ \|_\infty + \epsilon u_s$$

where

$$\epsilon = \frac{nu}{1-nu}((\|\ |X_U||X_L|\ \|_\infty + 1)(n + \max(\mathrm{diag}(|U|))) + n\|X_U\|_\infty\|U\|_\infty).$$

Using Lemma II.5 and the switching of rounding modes, we can obtain the upper bound of $\|X_U X_L PA - I\|_\infty$ using only floating-point arithmetic.

Algorithm II.4 (Oishi-Rump [18])**.** *This function returns an upper bound of $\|X_U X_L PA - I\|_\infty$.*

function $res = \mathrm{Method3}(A)$
  $n = \mathrm{size}(A, 1)$;
  $e = \mathrm{ones}(n, 1)$;
  $[\hat{L}, \hat{U}, p, X_L, X_U] = \mathrm{invlu}(A)$; *% Algorithm II.3*
  $\mathrm{feature}('setround', \mathrm{inf})$; *% Change the rounding mode to rounding to upward*
  $s_1 = 2nu(|X_U|(|X_L|(|\hat{L}|(|\hat{U}|e))))$;
  $s_2 = nu(|X_U|(|U|e))$;
  $\epsilon = nu/(1-nu)((s_2+1)(n + \max(\mathrm{diag}(|U|))) + n\|\ |X_U|e\|_\infty\|\ |U|e\|_\infty)$
  $res = \mathrm{norm}(s_1 + s_2 + \epsilon u_s, \mathrm{inf})$;
end

It should be noted that the cost after invlu(A) is $\mathcal{O}(n^2)$ flops; thus, Algorithm II.4 requires $4/3n^3$ flops. Next, we introduce two methods proposed in [20] and [21]. In the original papers, there is no treatment of underflow; however, we introduce these methods in the presence of underflow.

Lemma II.6 (Ogita-Oishi [20]). *Let $\hat{L}, \hat{U}$ be the computed LU factors of A, P be permutation matrix ($PA \approx \hat{L}\hat{U}$), and X_L, X_U be the approximate inverse matrices of $\hat{L}, \hat{U}$ using Algorithm II.3. Then, including possible underflow, the bounds for $\|X_U X_L PA - I\|_\infty$ can be obtained by*

$$\begin{aligned} \|X_U X_L PA - I\|_\infty &\leq \| \, |X_U|(|X_L PA - \hat{U}| + nu|U| + \epsilon E_U)\|_\infty, \\ \epsilon E_U &= \frac{nu_s(n + \max(\mathrm{diag}(\hat{U})))}{1 - nu} \end{aligned} \tag{4.4}$$

where the upper bound of $|X_L PA - U|$ is computed as

$$|X_L PA - \hat{U}| \leq \max(\mathtt{fl}_\triangledown(|X_L(PA) - \hat{U}|), \mathtt{fl}_\triangle(|X_L(PA) - \hat{U}|)).$$

We introduce an algorithm obtaining the upper bound of $\|X_U X_L PA - I\|_\infty$ on the basis of Lemma II.6.

Algorithm II.5 (Ogita-Oishi [20]). *This function returns upper bounds of $\|RA - I\|_\infty = \|X_U X_L PA - I\|_\infty$.*

```
function res = Method4(A)
  n = size(A, 1);
  e = ones(n, 1);
  [L̂, Û, p, X_L, X_U] = invlu(A);    % Algorithm II.3
  feature('setround', -inf);    % Change the rounding mode to rounding to downward
  S_1 = X_L A(p, :) − Û;
  feature('setround', inf);    % Change the rounding mode to rounding to upward
  S_2 = X_L A(p, :) − Û;
  S = max(|S_1|, |S_2|);
  s = |X_U|(Se + nu|Û|e + nu_s(n + max(diag(Û)))/(1 − nu));
  res = norm(s, inf);
end
```

Algorithm II.5 involves Algorithm II.3 ($4/3n^3$ flops) and two triangular-dense matrix multiplications (n^3 flops for a multiplication). The cost of Algorithm II.5 is $\frac{10}{3}n^3$ flops.

Lemma II.7 (Ozaki-Ogita-Oishi [21]). *Let $\hat{L}, \hat{U}$ be computed LU factors of A, P be permutation matrix, and X_L, X_U be approximate inverse matrices of $\hat{L}, \hat{U}$ using Algorithm II.3. Then, including possible underflow, $\|X_U X_L PA - I\|_\infty$ is bounded by*

$$\begin{aligned} & \|X_U X_L PA - I\|_\infty \\ \leq \; & \||X_U|(|\mathtt{fl}(X_L(PA) - \hat{U})| + (n+1)u(|X_L||PA| + |\hat{U}|) + nu|U| + \epsilon_1 E_U + \epsilon_2 E)\|_\infty, \\ \epsilon_1 E_U = \; & \frac{nu_s(n + \max(\mathrm{diag}(\hat{U})))}{1 - nu}, \quad \epsilon_2 E = \frac{n^2 u_s}{2}. \end{aligned}$$

We introduce an algorithm based on Lemma II.7 using direct rounding.

Algorithm II.6 (Ozaki-Ogita-Oishi [21])**.** *This function returns an upper bound of* $\|RA - I\|_\infty = \|X_U X_L PA - I\|_\infty$.

function $res = \text{Method5}(A)$
  $n = \text{size}(A, 1)$;
  $e = \text{ones}(n, 1)$;
  $[\hat{L}, \hat{U}, p, X_L, X_U] = \text{invlu}(A)$; % *Algorithm II.3*
  $S = X_L A(p, :) - \hat{U}$;
  feature($'setround'$, inf); % *Change the rounding mode to rounding to upward*
  $t = nu_s(n + \max(\text{diag}(\hat{U})))/(1 - nu) + n^2 u_s/2$;
  $s = |X_U|(|S|e + (n + 1)u(|X_L|(|A(p, :)|e) + |\hat{U}|e) + nu|\hat{U}|e + te)$;
  $res = \text{norm}(s, \text{inf})$;
end

This algorithm involves $\frac{7}{3}n^3$ flops, since it involves Algorithm II.3 ($4/3n^3$ flops) and a triangular-dense matrix multiplication (n^3 flops).

Chapter 5

Proposed verification method using LU-factors and their inverse matrices

5.1 Proposed methods

We set $R := (\hat{L}\hat{U})^{-1}P$ and aim to obtain the upper bound of $\|RA - I\|_\infty$. Note that the rigorous $(\hat{L}\hat{U})^{-1}$ is not required for the proposed method. We first define a function computing the LU factors and their inverse matrices.

Algorithm II.7. *This function returns LU factors and its approximate inverse matrices.*

function $[\hat{L}, \hat{U}, p, X_L, X_U] = \text{invlu2}(A)$
  $I = \text{eye}(\text{size}(A))$; % *I is the identity matrix*
  $[\hat{L}, \hat{U}, p] = \text{lu}(A)$; % *LU decomposition* $A(p,:) \approx \hat{L}\hat{U}$
  $X_L = \hat{L}\backslash I$; % *Solve* $\hat{L}X_L = I$ *for* X_L
  $X_U = I/\hat{U}$; % *Solve* $X_U\hat{U} = I$ *for* X_U
end

The difference in Algorithm II.3 and Algorithm II.7 is only the computation of X_L. For computed results of Algorithm II.7, we define matrices Δ_L, Δ_U and Δ_A as follows:

$$\Delta_L := I - \hat{L}X_L, \quad \Delta_U := I - X_U\hat{U}, \quad \Delta_A = \hat{L}\hat{U} - PA. \tag{5.1}$$

Here, $\hat{L}, X_L$ and Δ_L are lower triangular matrices, and $\hat{U}, X_U$ and Δ_U are upper triangular matrices. Assume that matrices X_L and X_U are computed by backward substitution for linear systems $\hat{L}X = I$ and $X\hat{U} = I$.

We first introduce the variant of Lemma II.4.

Lemma II.8. *If all diagonal elements of* $I - |T|$ *are positive, then*

$$|(I - T)^{-1}| \le (I - |T|)^{-1},$$

where T *is a triangular matrix and* I *is the identity matrix.*

Proof 1. *From Lemma II.4, $I-T$ satisfies $|(I-T)^{-1}| \le M(I-T)^{-1} =: S$. Assume that T is a lower triangular matrix, $S = \{s_{ij}\}$ satisfies*

$$s_{ij} = \begin{cases} 1/(1-t_{ii}), & i = j, \\ \displaystyle\sum_{k=j}^{i-1} |t_{ik}||s_{ki}|/(1-t_{ii}), & i > j. \end{cases}$$

Here,

$$\{(I-|T|)^{-1}\}_{ij} = \begin{cases} 1/(1-|t_{ii}|), & i = j, \\ \displaystyle\sum_{k=j}^{i-1} |t_{ik}||s_{ki}|/(1-|t_{ii}|), & i > j, \end{cases}$$

then,

$$|(I-T)^{-1}| \le M(I-T)^{-1} \le (I-|T|)^{-1}$$

is satisfied. The case of an upper triangular matrix can similarly be proved. □

The following theorem provides a sufficient condition for nonsingularity of matrices.

Theorem II.9. *For $A \in \mathbb{F}^{n\times n}$, assume that LU decomposition successfully runs to completion. Matrices $\hat{L}, \hat{U}$, and P are computed LU factors such as $\hat{L}\hat{U} \approx PA$ and $\hat{L}$ and $\hat{U}$ are non-singular. Matrices X_L and X_U are approximate solutions of $\hat{L}X = I$ and $X\hat{U} = I$ by backward substitution. For Δ_L and Δ_U in (5.1), assume that there exist $v_L > 0$ and $v_U > 0$ such that*

$$(I-|\Delta_L|)v_L > 0, \quad (I-|\Delta_U|)v_U > 0, \tag{5.2}$$

Then, matrix A is non-singular if

$$\|(I-|\Delta_U|)^{-1}|X_U X_L|(I-|\Delta_L|)^{-1}|\Delta_A|\ \| < 1. \tag{5.3}$$

Proof 2. *Note that off-diagonal elements of triangular matrices $I-|\Delta_L|$ and $I-|\Delta_U|$ are not positive. From assumption (5.2), all diagonal elements of $I-|\Delta_L|$ and $I-|\Delta_U|$ are positive. Since $I-|\Delta_L| \le I-\Delta_L$ and $I-|\Delta_U| \le I-\Delta_U$, all diagonal elements of $I-\Delta_L$ and $I-\Delta_U$ are also positive. Therefore triangular matrices $I-\Delta_L$ and $I-\Delta_U$ are non-singular, and from (5.1), we have*

$$\hat{L}^{-1} = X_L(I-\Delta_L)^{-1}, \quad \hat{U}^{-1} = (I-\Delta_U)^{-1}X_U. \tag{5.4}$$

Next, we derive an upper bound of $|(\hat{L}\hat{U})^{-1}PA - I|$. Using (5.1), (5.4) and Lemma II.8 in turn,

$$\begin{aligned} |RA-I| = |(\hat{L}\hat{U})^{-1}PA - I| &= |(\hat{L}\hat{U})^{-1}(\hat{L}\hat{U} - \Delta_A) - I| = |(\hat{L}\hat{U})^{-1}\Delta_A| \\ &\le |\hat{U}^{-1}\hat{L}^{-1}||\Delta_A| \\ &\le |(I-\Delta_U)^{-1}| \cdot |X_U X_L| \cdot |(I-\Delta_L)^{-1}| \cdot |\Delta_A| \\ &\le (I-|\Delta_U|)^{-1}|X_U X_L|(I-|\Delta_L|)^{-1}|\Delta_A|. \end{aligned}$$

Therefore, if $\|(I-|\Delta_U|)^{-1}|X_U X_L|(I-|\Delta_L|)^{-1}|\Delta_A|\ \| < 1$, then A is non-singular. □

From Theorem II.9, we derive an upper bound $|RA - I|$, where $R = (\hat{L}\hat{U})^{-1}P$. Next, we introduce a theorem concerning with an upper bound of $\|RA - I\|_\infty$. The critical point of the following theorem is to obtain an upper bound without computing $(I - |\Delta_L|)^{-1}$ and $(I - |\Delta_U|)^{-1}$.

Theorem II.10. *Assume that (5.2) is satisfied for $\exists v_L, v_U > 0$, then*

$$\|(I - |\Delta_U|)^{-1}|X_U X_L|(I - |\Delta_L|)^{-1}|\Delta_A| \ \|_\infty \leq \max_i \frac{(|X_U X_L|v_L)_i}{(w_U)_i} \max_i \frac{(|\Delta_A|e)_i}{(w_L)_i} \|v_U\|_\infty,$$

where $w_L = (I - |\Delta_L|)v_L > 0$ and $w_U = (I - |\Delta_U|)v_U > 0$.

Proof 3. *We obtain*

$$|\Delta_A|e = \left(\frac{(|\Delta_A|e)_1}{(w_L)_1}(w_L)_1, \ldots, \frac{(|\Delta_A|e)_n}{(w_L)_n}(w_L)_n \right)^T \leq \max_i \frac{(|\Delta_A|e)_i}{(w_L)_i} w_L. \tag{5.5}$$

From the definition of w_L, we have $(I - |\Delta_L|)^{-1}w_L = v_L$. This and (5.5) derives

$$(I - |\Delta_L|)^{-1}|\Delta_A|e \leq \max_i \frac{(|\Delta_A|e)_i}{(w_L)_i} v_L. \tag{5.6}$$

Similarly, we obtain

$$(I - |\Delta_U|)^{-1}|X_U X_L|v_L \leq \max_i \frac{(|X_U X_L|v_L)_i}{(w_U)_i} v_U.$$

Then,

$$\begin{aligned} \| \ |(\hat{L}\hat{U})^{-1}PA - I|e\|_\infty &\leq \|(I - |\Delta_U|)^{-1}|X_U X_L|(I - |\Delta_L|)^{-1}|\Delta_A|e\|_\infty \\ &\leq \max_i \frac{(|X_U X_L|v_L)_i}{(w_U)_i} \max_i \frac{(|\Delta_A|e)_i}{(w_L)_i} \|v_U\|_\infty \end{aligned}$$

is satisfied. □

The manner of setting v_L and v_U is important. This discussion is provided in Section 3.1. From Theorem II.10, if we obtain upper bounds of $|X_U X_L|v_L$ and $|\Delta_A|e$, and lower bounds of w_L and w_U, then we can obtain the upper bound of $\|(\hat{L}\hat{U})^{-1}PA - I\|_\infty$. We first explain how to compute the upper bound of $|X_U X_L|v_L$ and $|\Delta_A|e$.

Method A. $|X_U X_L|v_L \leq |X_U|(|X_L|v_L)$

Method B. $|X_U X_L|v_L \leq \mathtt{fl}(|X_U X_L|)v_L + nu|X_U|(|X_L|v_L) + \dfrac{nu_s}{2}v_L$ from Lemma II.1

Method C. $|\Delta_A|e \leq nu|\hat{L}|(|\hat{U}|e) + \dfrac{nu_s}{1 - nu}(ne + \mathrm{diag}(|U|))$ from Lemma II.2

Method D. $|\Delta_A|e \leq \max(\mathtt{fl}_\triangledown(|\hat{L}\hat{U} - PA|)), \mathtt{fl}_\triangle(|\hat{L}\hat{U} - PA|)e$

Table 5.1: Comparison of computational cost of proposed methods

Name	Method		Cost
	$\lvert X_U X_L\rvert v$	$\lvert\Delta_A\rvert e$	
T(A,C)	A	C	$\frac{4}{3}n^3$
T(B,C)	B	C	$2n^3$
T(A,D)	A	D	$\frac{8}{3}n^3$
T(B,D)	B	D	$\frac{10}{3}n^3$

Methods A, B, and methods C, D produce upper bounds of $|X_U X_L|v_L$ and $|\Delta_A|e$, respectively. The computational cost of combinations of methods A, B and methods C, D are presented in Table 5.1. For example, T(A, C) signifies that method A is used for the upper bound for $|X_U X_L|v_L$ and method C is used for the upper bound for $|\Delta_A|e$. We note that the cost of $\texttt{fl}(X_U X_L)$ and $\texttt{fl}(LU)$ is $2n^3/3$ flops.

Next, we describe how to compute the lower bounds of $w_L := (I - |\Delta_L|)v_L$ and $w_U := (I - |\Delta_U|)v_U$. We can compute the lower bounds from Lemma II.3 using direct rounding as follows:

$$\begin{aligned} w_L = (I - |\Delta_L|)v_L &\geq -(nu|\hat{L}||X_L|v_L + \frac{nu_s}{1-nu}ee^T v_L - v_L) \\ &\geq -\texttt{fl}_\triangle(nu|\hat{L}|(|X_L|v_L) + \frac{nu_s}{1-nu}e(e^T v_L) - v_L) =: w'_L, \qquad (5.7) \\ w_U = (I - |\Delta_U|)v_U &\geq -(nu|X_U||\hat{U}|v_U + \frac{u_s}{1-nu}(ne + \mathrm{diag}(|\hat{U}|))e^T v_U - v_U) \\ &\geq -\texttt{fl}_\triangle(nu|X_U|(|\hat{U}|v_U) + \frac{u_s e^T v_U}{1-nu}(ne + \mathrm{diag}(|\hat{U}|)) - v_U) \\ &=: w'_U. \qquad (5.8) \end{aligned}$$

5.1.1 Setting of v_L and v_U

Here, we introduce how to obtain v_L and v_U defined in (5.2). Let $s \in \mathbb{F}^n$ and $r \in \mathbb{F}^n$ be the upper bounds of $|X_U X_L|v_L$ and $|\Delta_A|e$, respectively. These vectors can be obtained using a combination either of method A or B and either method C or D. In addition, from (5.7) and (5.8), w'_L and w'_U are computable lower bounds of w_L and w_U. Thus, from Theorem II.10,

$$\begin{aligned} \|(\hat{L}\hat{U})^{-1}PA - I\|_\infty &\leq \max_i \frac{(|X_U X_L|v_L)_i}{(w_U)_i} \max_i \frac{(|\Delta_A|e)_i}{(w_L)_i} \|v_U\|_\infty \\ &\leq \max_i \frac{s_i}{(w'_U)_i} \max_i \frac{r_i}{(w'_L)_i} \|v_U\|_\infty. \qquad (5.9) \end{aligned}$$

To avoid the overestimation of (5.9), we aim to have

$$\max_i \frac{s_i}{(w'_U)_i} \approx 1, \quad \max_i \frac{r_i}{(w'_L)_i} \approx 1, \quad \text{i.e.,} \quad w'_U \approx s, \quad w'_L \approx r.$$

Therefore, we set the linear systems as follows:

$$w'_U \approx (I - nu|X_U||\hat{U}|)v_U^* = s, \quad w'_L \approx (I - nu|\hat{L}||X_L|)v_L^* = r. \qquad (5.10)$$

Let the approximation of v_U^* and v_L^* be v_U and v_L, respectively. The linear systems are solved by an iterative method without matrix multiplications $|\hat{L}||X_L|$ and $|X_U||\hat{U}|$ from initial vectors s and r. It should be noted that the coefficient matrices in (5.10) are often strongly diagonally dominant so that we can obtain an accurate approximate solution of the linear systems with several iterations. Because $s, r > 0$, we can expect that $w'_L, w'_U > 0$. In addition,

$$v_L \approx (I - nu|\hat{L}||X_L|)^{-1}r, \quad v_U \approx (I - nu|X_U||\hat{U}|)^{-1}s,$$

and all diagonal elements in $(I - nu|\hat{L}||X_L|)$ and $(I - nu|X_U||\hat{U}|)$ are expected to be positive. In this case, from Lemma II.8, all elements in $(I - nu|\hat{L}||X_L|)^{-1}$ and $(I - nu|X_U||\hat{U}|)^{-1}$ are positive, and $v_L, v_U > 0$.

5.1.2 Algorithm flow

Here, we present the flow of the proposed method.

Step 1 Compute the LU decomposition for a given matrix A and its inverse matrices by Algorithm II.7.

Step 2 Solve the linear system $(I - nu|\hat{L}||X_U|)v_L^* = r$ and obtain approximation v_L by an iterative method, where r is the upper bound of $|\Delta_A|e$.

Step 3 Solve the linear system $(I - nu|X_U||\hat{U}|)v_U^* = s$ and obtain approximation v_U by an iterative method, where s is the upper bound of $|X_U X_L|v_L$.

Step 4 Compute w'_L and w'_U based on (5.7) and (5.8), respectively.

Step 5 If v_L, w'_L, v_U or $w'_U \leq 0$ then verification fails and this algorithm completes.

Step 6 Compute the upper bound of $\|(\hat{L}\hat{U})^{-1}PA - I\|_\infty$ based on Theorem II.10. If the bound is strictly less than 1, then the given matrix is nonsingular.

The details of this algorithm are provided in Appendix A.

5.2 Numerical results

In this section, we present numerical results to compare previous studies with our proposed methods. Test matrices $A \in \mathbb{F}^{n\times n}$ were generated using MATLAB by

$$A = \texttt{gallery}(\text{`}\texttt{randsvd}', n, \text{cond}, \text{mode}, n, n, 1), \tag{5.11}$$

where cond is the expected condition number of A, and mode is one of the following values: (1) one large singular value, (2) one small singular value, (3) geometrically distributed singular values. We used the Jacobi method for linear systems in **STEP 2** and **STEP 3** and obtained an accurate approximation of the solution of linear systems in several iterations.

There is a routine trmm that performs triangular-dense matrix multiplication in BLAS; however, a routine computing triangular-triangular matrix multiplication is not supported. We implemented it using block matrix multiplication using dgemm in BLAS.

Table 5.2: Comparison of computational cost of verification methods.

Method	Cost	Ratio
Method3	$\frac{4}{3}n^3$	2
T(A, C)	$\frac{4}{3}n^3$	2
T(B, C)	$2n^3$	3
Method5	$2n^3$	3.5
T(A, D)	$\frac{8}{3}n^3$	4
Method4	$\frac{10}{3}n^3$	5
T(B, D)	$\frac{10}{3}n^3$	5
Method2	$4n^3$	6
Method1	$6n^3$	9

Table 5.2 presents the computational cost of the verification methods. The ratio is the cost of each method divided by the cost of LU decomposition. We compared the upper bounds of $\|RA-I\|_\infty$ for various condition numbers. Figures 5.1-5.6 present the upper bounds of $\|RA-I\|_\infty$, where the mode of the gallery function is 1, 2, and 3, respectively. Figures 5.1 and 5.2 present the computed results when mode = 1. In this case, T(B, D) can verify the nonsingularity of ill-conditioned matrices more effectively than Method2. The cost of these methods is $\frac{10}{3}n^3$ and $4n^3$ flops. This result indicates that the proposed method is fast and robust when mode = 1. Figures 5.3 and 5.4 present the computed results when mode = 2. In this case, the proposed methods do not display superior performance to that of previous methods. The reason is as follows: For Method5, we have

$$|RA - I| \approx (n+1)u|X_U||X_L||PA|. \tag{5.12}$$

However,

$$|RA - I| \approx |X_U X_L||\Delta_A| \approx |X_U||X_L||\Delta_A|. \tag{5.13}$$

tends to be satisfied for mode=2. Because $|X_U||X_L|$ is common for (5.12) and (5.13), we compare $(n+1)u|PA|$ and $|\Delta_A|$ as

$$Y_1 := (n+1)u|PA|, \quad Y_2 := \max(|\mathtt{fl}_\bigtriangledown(PA - \hat{L}\hat{U})|, |\mathtt{fl}_\bigtriangleup(PA - \hat{L}\hat{U})|), \quad Y_3 := nu|\hat{L}||\hat{U}|,$$

where Y_2 and Y_3 are the upper bounds of $|\Delta_A|$ using method D and Method C, respectively. Table 5.3 presents the maximum norm of Y_1, Y_2, and Y_3. The table indicates that $Y_1 \lesssim Y_2 \lesssim Y_3$ tends to be satisfied; therefore, Method5 produces better upper bounds of $\|RA-I\|_\infty$ than those produced by the proposed methods. This implies that Method4 is also superior to the proposed methods.

Figures 5.5 and 5.6 present the upper bound of $\|RA-I\|_\infty$ when mode = 3. In this case, the computed results of Methods2, 4, and T(B, D) are almost identical.

Table 7.1 compares the computation times of the verification methods. Here, we define two notations, T_* and κ_*. T_* is the computation time and κ_* is a condition number while the upper bound of $\|RA-I\|_\infty \approx 1$. For example, T_{M1} indicates the computation time of Method1, and the nonsingularity of a given matrix can be verified up to κ_{M1} by Method1. Therefore, small T_* and large κ_* indicate a more effective method. We summarize the numerical results as follows.

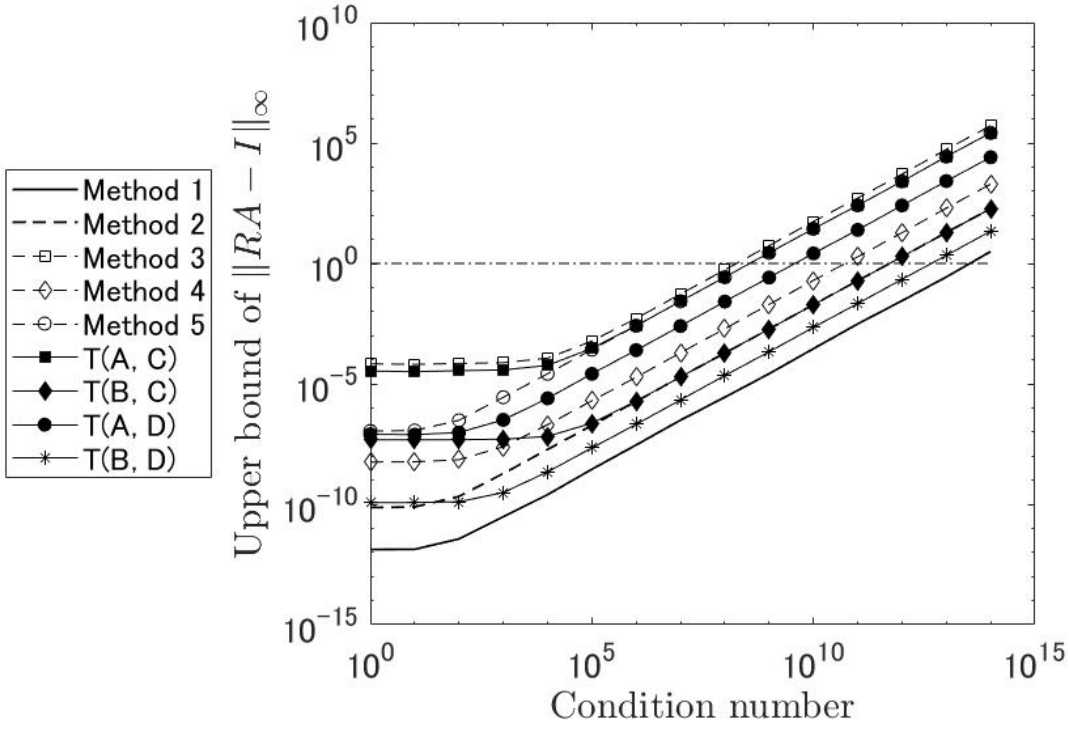

Figure 5.1: $n = 1,000, \text{mode} = 1$

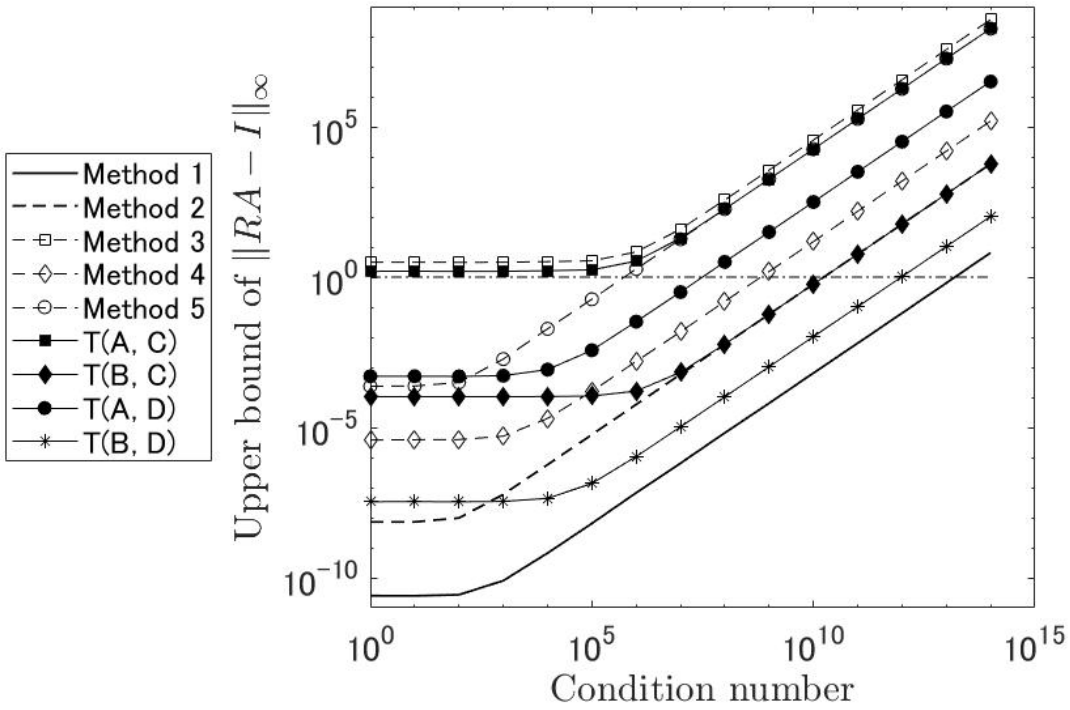

Figure 5.2: $n = 10,000, \text{mode} = 1$

Table 5.3: Comparison of the maximum norm of Y_1, Y_2, and Y_3 when $n = 1000$

method \ cnd	10^6	10^8	10^{10}	10^{12}
Y_1	2.87e-12	2.87e-12	2.87e-12	2.86e-12
Y_2	1.85e-11	1.77e-11	1.96e-11	1.97e-11
Y_3	4.23e-09	4.02e-09	4.04e-09	4.46e-09

mode = 1. $T_{T(B,C)} < T_{M4}, \quad \kappa_{M4} < \kappa_{T(B,C)}$. T(B,C) is superior to Method4.

mode = 2. $T_{M5} < T_{T(B,D)} \approx T_{M4}, \quad \kappa_{T(A,D)} < \kappa_{M5} < \kappa_{M4}$.
T(B,D) is inferior to Methods4 and Method5.

mode = 3. $T_{T(B,D)} \approx T_{M4}, \quad \kappa_{M4} < \kappa_{T(B,D)}$. T(B,D) is superior to Method4.
$T_{T(B,C)} < T_{M5}, \quad \kappa_{M5} < \kappa_{T(B,C)}$. T(B,C) is superior to Method5.

Table 5.4: Comparison of computation times [s] and their ratio (verification/LU decomposition).

Method	Cost	Ratio	10,000	20,000	30,000
Method3	$\frac{4}{3}n^3$	2	6.87 (2.63)	46.3 (2.64)	143 (2.59)
T(A, C)	$\frac{4}{3}n^3$	2	7.76 (2.98)	47.6 (2.70	145 (2.63)
T(B, C)	$2n^3$	3	10.7 (4.07)	65.9 (3.79)	204 (3.66)
Method5	$2n^3$	3.5	11.2 (4.32)	73.0 (4.18)	227 (4.11)
T(A, D)	$\frac{8}{3}n^3$	4	12.7 (4.96)	83.0 (4.76)	256 (4.62)
Method4	$\frac{10}{3}n^3$	5	14.2 (5.47)	96.7 (5.53)	304 (5.49)
T(B, D)	$\frac{10}{3}n^3$	5	15.3 (5.87)	99.4 (5.74)	311 (5.88)
Method2	$4n^3$	6	14.9 (5.55)	101 (5.79)	327 (5.91)
Method1	$6n^3$	9	23.1 (7.63)	143 (8.19)	469 (8.44)

5.3 Conclusion of Part Ⅱ

In Part Ⅱ, we propose fast verification methods for proving the nonsingularity of real matrices. For several test matrices, the proposed methods are faster and more robust than existing methods. In addition, the proposed methods can be easily extended to proving the nonsingularity of interval matrices.

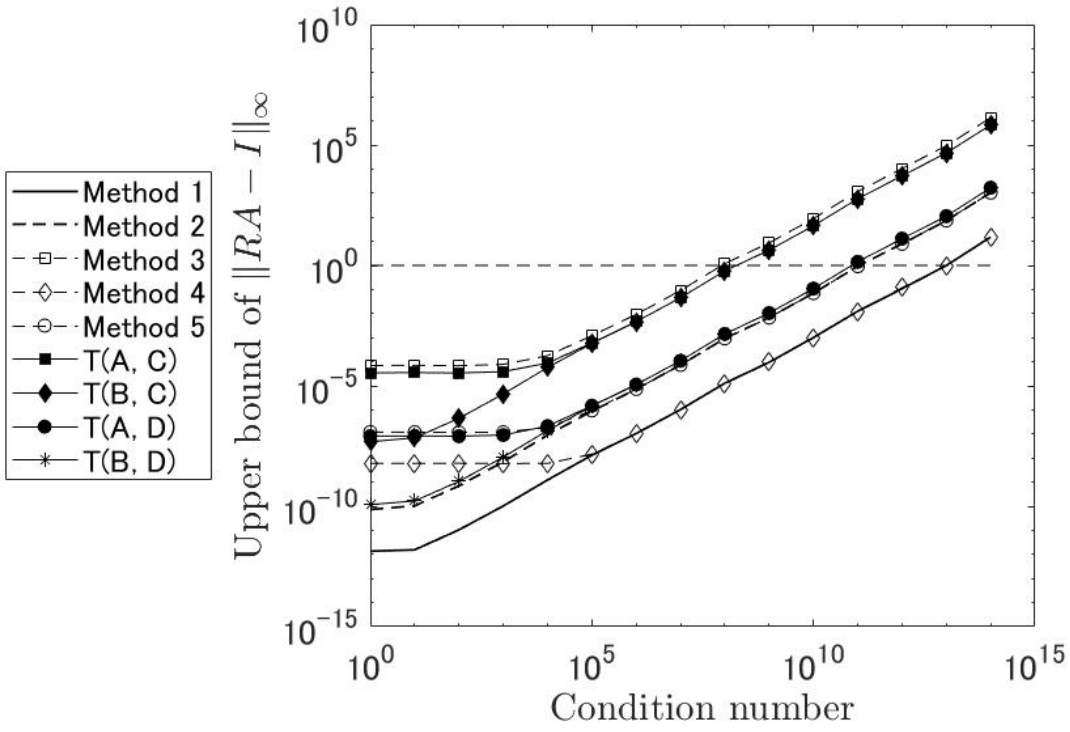

Figure 5.3: $n = 1,000, \text{mode} = 2$

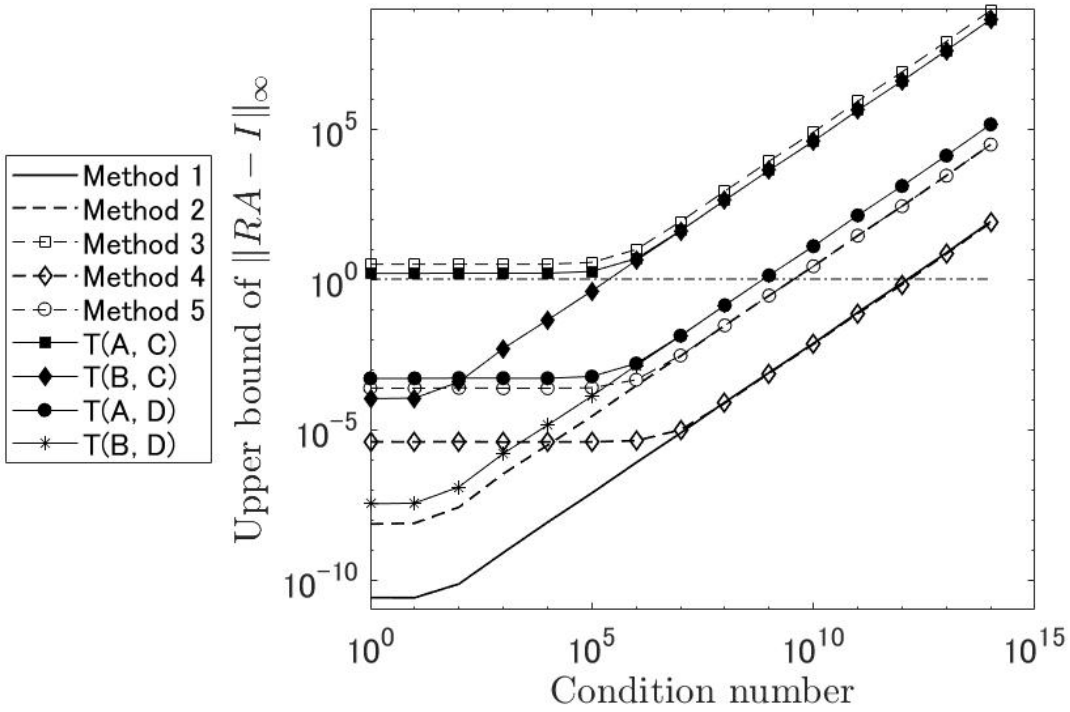

Figure 5.4: $n = 10,000, \text{mode} = 2$

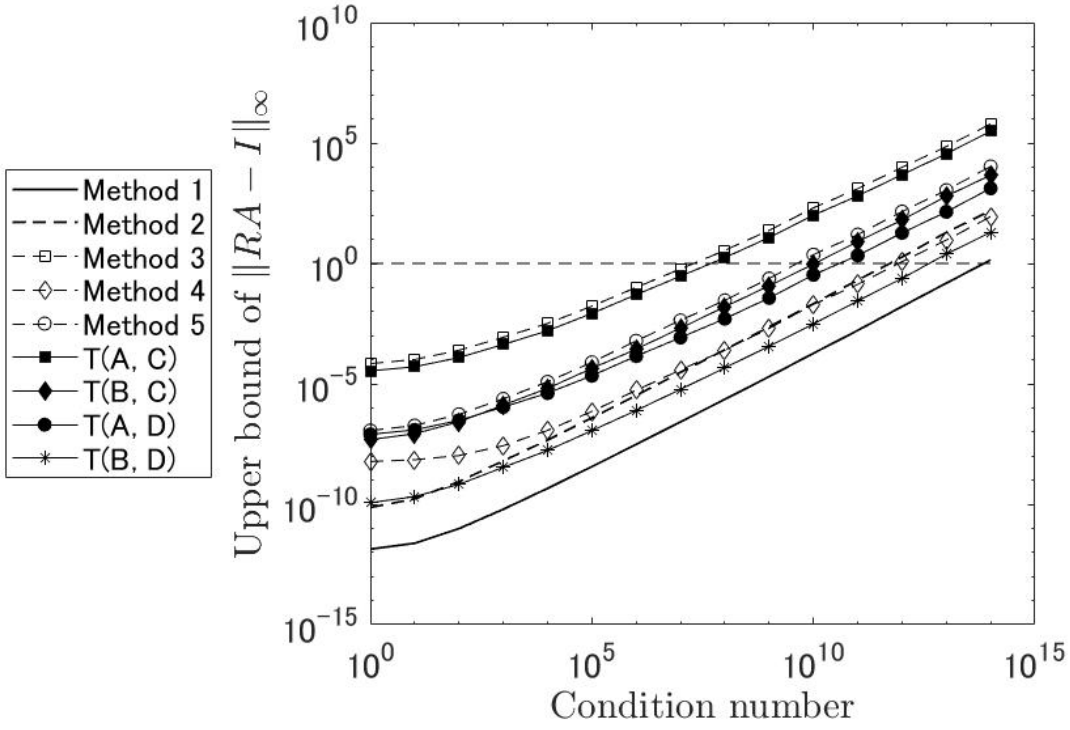

Figure 5.5: $n = 1,000, \text{mode} = 3$

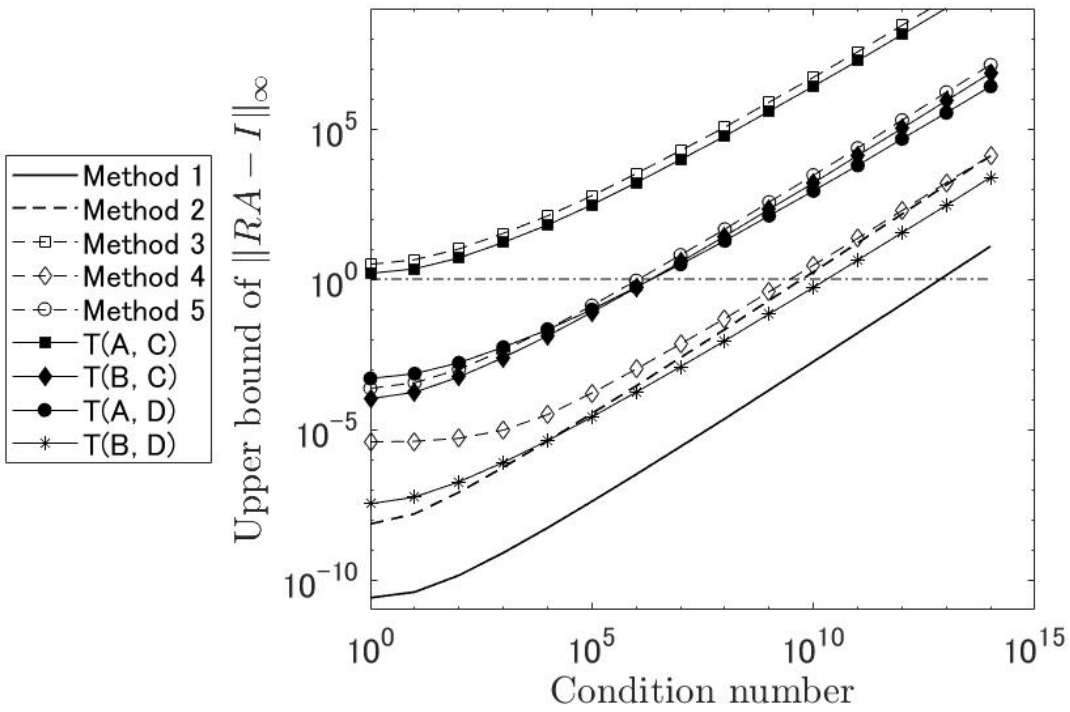

Figure 5.6: $n = 10,000, \text{mode} = 3$

Part III

Validated numerical computations of all eigenvalues for large-scale matrices

Chapter 6

Introduction

Let $A = A^T, B = B^T \in \mathbb{R}^{n\times n}$ with B being positive definite. We consider a generalized eigenproblem

$$Ax^{(i)} = \lambda_i Bx^{(i)},$$

where $\lambda_i \in \mathbb{R}$ and $x^{(i)} \in \mathbb{R}^n$, $i = 1, \ldots, n$, are eigenvalues and eigenvectors, respectively. Assume that $\lambda_i \leq \lambda_{i+1}$ for $1 \leq i \leq n-1$, and let $\hat{\lambda}_i$ and $\hat{x}^{(i)}$, $i = 1, \ldots, n$, be the computed eigenvalues and eigenvectors, respectively. The proposed verification method produces upper bounds of $|\hat{\lambda}_i - \lambda_i|$ for all i on the basis of the Gershgorin circle theorem (cf. e.g. [1, p.357]) using $\hat{\lambda}_i$ and $\hat{x}^{(i)}$ for all i. The proposed method also utilizes verified solutions of linear systems proposed in [23]. We also improve error analysis for approximate solutions of linear systems to obtain better error bounds of computed eigenvalues.

To demonstrate the effectiveness of the proposed method, we present numerical results on the scalability of the proposed method on FUJITSU PRIMEHPC FX100. In addition, as a practical application, we provide quantitative error bounds of the computed eigenvalues for both standard and generalized eigenproblems arising from quantum materials physics, where the results are obtained on the RIKEN K computer. This study is related to [B3] in the list of publications.

6.1 Preliminaries

First, we introduce the notation used in Part III. Let $P = (p_{ij}), Q = (q_{ij}) \in \mathbb{R}^{n\times n}$. Inequalities for matrices are understood componentwise, for example, $P > Q$ signifies $p_{ij} > q_{ij}$ for all (i, j). The absolute value notation $|P|$ signifies $|P| = (|p_{ij}|) \in \mathbb{R}^{n\times n}$, a nonnegative matrix consisting of componentwise absolute values of P. Similar notation is applied to real vectors. Let $\|P\|_\infty$ denote the maximum norm of P such that $\|P\|_\infty = \max_{1\leq i\leq n} \sum_{j=1}^n |p_{ij}|$.

Next, we introduce the Gershgorin circle theorem: $\mathcal{G}_i := \{\mu : |\mu - p_{ii}| \leq \sum_{i\neq j} |p_{ij}|\}$, then $\lambda(P) \subseteq \bigcup_{i=1}^n \mathcal{G}_i$. If the union of k sets $\mathcal{G}_i$, $i_1, \ldots, i_k$, are disjoint from the others, then that union contains exactly k eigenvalues of P.

Here, we review previous studies pertaining to verification methods for all eigenvalues [24, 25, 26]. Define D and X such that $D \in \mathbb{R}^{n\times n}$ is a diagonal matrix with $d_{ii} = \lambda_i$ for $i = 1, \ldots, n$ and $X = [x^{(1)}, \ldots, x^{(n)}] \in \mathbb{R}^{n\times n}$ with $X^T BX = I$, where I is the identity matrix. Then, the matrix form $AX = BXD$ is obtained, and $B^{-1}AX = XD$ is satisfied. For any nonsingular $\hat{X} \in \mathbb{R}^{n\times n}$, we have $\lambda_k(B^{-1}A) = \lambda_k(\hat{X}^{-1}B^{-1}A\hat{X}) = d_{kk}$ for all k. Assuming that $\hat{X} \approx X$ and $\hat{D} = \mathrm{diag}(\hat{\lambda}) \approx D$,

it is expected that $\hat{X}^T \approx (B\hat{X})^{-1}$ and $A\hat{X} \approx B\hat{X}\hat{D}$. Then, the problem of enclosing all eigenvalues is reduced to the verified solutions of linear systems such that $(B\hat{X})Y = A\hat{X}$ for Y, where $\hat{D}$ is expected to be a good approximation of Y. If we obtain an upper bound of $|\hat{D} - Y|$, then all eigenvalues can be enclosed by the Gershgorin circle theorem.

6.2 Previous studies

We first describe rounding error analysis for all eigenvalues using a scaler. For a matrix $A + \Delta$, all eigenvalues are enclosed by

$$\lambda_i(A) - \|\Delta\|_2 \leq \lambda_i(A + \Delta) \leq \lambda_i(A) + \|\Delta\|_2$$

Here, we have $\lambda_i(\hat{X}^{-1}A\hat{X}) = \lambda_i(A)$. In addition,

$$|\hat{D} - \hat{X}^{-1}A\hat{X}| \leq \hat{X}^{-1}|\hat{X}\hat{D} - A\hat{X}|$$

and, if $\|I - \hat{X}^T\hat{X}\|_2 < 1$

$$\|\hat{X}^{-1}\|_2 \leq \frac{1}{\sqrt{1 - \|I - \hat{X}^T\hat{X}\|_2}}.$$

Therefore,

$$\|\hat{D} - \hat{X}^{-1}A\hat{X}\|_2 \leq \frac{\|\hat{X}\hat{D} - A\hat{X}\|_2}{\sqrt{1 - \|I - \hat{X}^T\hat{X}\|_2}} =: \alpha$$

is satisfied. Then,

$$\lambda_i(A) - \alpha \leq \lambda_i(A) = \lambda_i(\hat{X}^{-1}A\hat{X}) \leq \lambda_i(A) + \alpha$$

is obtained.

Next, we describe element-wise error analysis for all eigenvalues, which was proposed by Miyajima [25]. This paper provides the upper bound of $|\hat{D} - \hat{X}^{-1}B^{-1}A\hat{X}|$ such that

$$|\hat{D} - \hat{X}^{-1}B^{-1}A\hat{X}| \leq |R| + \max_i \frac{(|R|e)_i}{(e - |G|e)_i}|G|, \tag{6.1}$$

where $e = (1, \ldots, 1)^T$.

Chapter 7

Proposed method

7.1 Rounding error analysis

We define

$$R = \hat{X}^T(B\hat{X}\hat{D} - A\hat{X}), \quad G = I - \hat{X}^T B\hat{X}, \tag{7.1}$$

and assume that there exists an n-vector $v > 0$ satisfying $w := (I - |G|)v > 0$. This means that $(I - |G|)$ and $\hat{X}$ are nonsingular. Then,

$$\hat{D} - \hat{X}^{-1}B^{-1}A\hat{X} = (\hat{X}^T B\hat{X})^{-1}\hat{X}^T(B\hat{X}\hat{D} - \hat{A}\hat{X}) = (I - G)^{-1}R.$$

From the assumption for v,

$$|(I - G)^{-1}| \leq (I - |G|)^{-1} = I + (I - |G|)^{-1}|G|$$

is satisfied. Therefore, we obtain

$$|\hat{D} - \hat{X}^{-1}B^{-1}A\hat{X}| \leq |R| + (I - |G|)^{-1}|G||R|.$$

Suppose an n-vector $v > 0$ satisfies $w := (I - |G|)v > 0$. Using [27, p. 134],

$$|\hat{D} - \hat{X}^{-1}B^{-1}A\hat{X}|e \leq |R|e + \max_i \frac{(|G||R|e)_i}{w_i}v =: r, \tag{7.2}$$

where $e = (1, \ldots, 1) \in \mathbb{R}^n$. From this and the Gershgorin circle theorem, all eigenvalues are enclosed by

$$\lambda(B^{-1}A) \subseteq \bigcup_{i=1}^{n}[\hat{\lambda}_i - r_i, \hat{\lambda}_i + r_i].$$

If $\hat{\lambda}_i + r_i \leq \hat{\lambda}_{i+1} - r_{i+1}$ for $1 \leq i \leq n-1$, no multiple eigenvalue exists. Here, it is better that v satisfies $(I - |G|)v \approx |G||R|e$. In addition, if $\|G\|_\infty \ll 1$, $(I - |G|)^{-1} \approx (I + |G|)$ is satisfied [1]. Thus, we set $v = (I + |G|)|G||R|e$. Then,

$$w = (I - |G|)v = (I - |G|)(I + |G|)|G||R| = (I - |G|^2)|G||R|$$

and

$$\max_i \frac{(|G||R|e)_i}{w_i} = (1 + \mathcal{O}(u^2)) \approx 1$$

are obtained. Here, we compare error analysis of the proposed method and Miyajima's method [25]. From (6.1),

$$\max_i \frac{(|R|e)_i}{(e - |G|e)_i}|G|e \geq \|R\|_\infty |G|e$$

However, from (7.2)

$$\max_i \frac{(|G||R|e)_i}{w_i} v \leq (1 + \mathcal{O}(u^2))(I + |G|)|G||R|e \approx |G||R|e$$

In addition, we have $|G||R|e \leq \|R\|_\infty |G|e$. Therefore,

$$\max_i \frac{(|G||R|e)_i}{w_i} v \lesssim \max_i \frac{(|R|e)_i}{(e - |G|e)_i}|G|e$$

is expected. This signifies that the proposed upper bound is superior to that of the previous study.

7.2 Numerical results

Here we discuss the performance of the proposed verification method. We used the ScaLAPACK routines PDSYEVD and PDSYGVX as eigensolvers for standard and generalized eigenproblems, respectively.

First, we present the performance in terms of computational speed on FUJITSU FX100 for standard eigenproblems using pseudo-random matrices with various n. The specification of FX100 is as follows: CPU: SPARC64 XIfx with 32 cores, RAM: 32 GB, Total number of nodes: 2,880. Figure 7.1 displays the ratio of computation times of the proposed verification method T_{veri} and the eigensolver T_{eig} as T_{veri}/T_{eig}. When the number of nodes increases, the ratio decreases, which signifies that the proposed method has higher strong-scalability than the ScaLAPACK routine PDSYEVD. The reason for this is that the proposed method is based primarily on matrix multiplication.

Next, as a practical application, we consider a quantum materials simulation that aims to understand electronic structures in material physics. To correctly understand the properties of materials, it is crucial to determine the order of eigenvalues [28]. Our verification method is useful for this purpose. The used matrix data were stored in the ELSES matrix library [29, 30] and were generated by ELSES [31], a quantum mechanical nanomaterial simulator. The number in a problem name indicates the dimension of the problem; for example, the matrix size of VCNT400000std is 400,000. In addition, std in VCNT400000std indicates the standard eigenvalue problem. To address larger problems, we performed numerical experiments on the RIKEN K computer for both standard and generalized eigenproblems. The specification of the K computer is as follows: CPU: SPARC64 VIIIfx with 8 cores, RAM: 8 GB, Total number of nodes: 82,944.

From Tab. 7.1, which presents the computation times for obtaining verified solutions and the ratio as T_{veri}/T_{eig}, the proposed verification method is faster than the eigensolver on K computer. Define the difference $\delta_k := \hat{\lambda}_{k+1} - \hat{\lambda}_k$ and the radius sum $\rho_k := r_k + r_{k+1}$ for $1 \leq k \leq n-1$. If

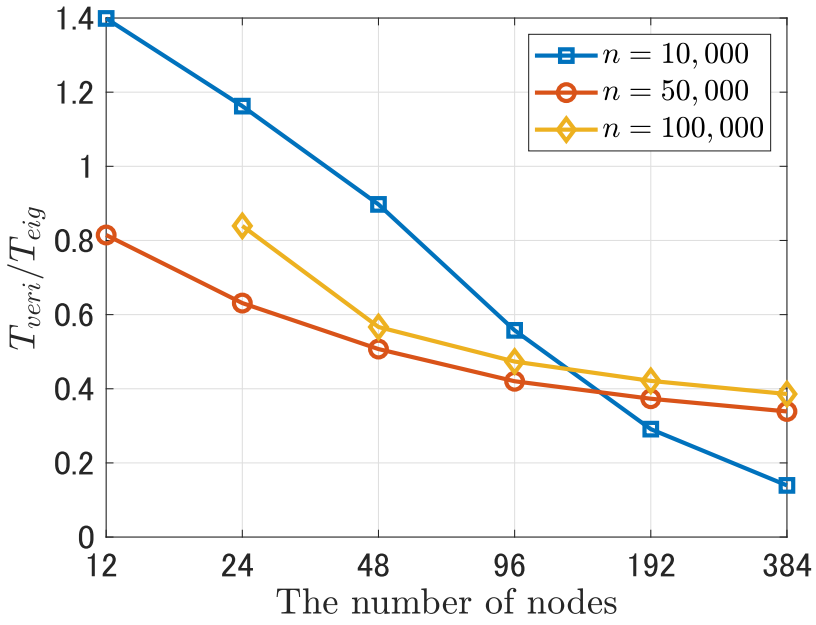

Figure 7.1: Ratio of computation times on FX100

Table 7.1: Computation times [s] obtaining verified solutions on K computer

Problem name	No. of nodes	T_{eig}	T_{veri}	Ratio
VCNT225000	2,025	2.62e+03	1.77e+03	0.67
VCNT400000std	1,600	9.01e+03	7.14e+03	0.79
VCNT1000000std	10,000	2.67e+04	2.08e+04	0.77

$\rho_k/\delta_k > 1$ for all k, then all eigenvalues can be separated. We present the results in Fig. 7.2, where the left figure shows the results of generalized eigenproblems and the right figure shows the results of standard eigenproblems. As can be seen from the left figure, we succeed in confirming that there are no multiple eigenvalues for generalized eigenproblems with $n \leq 225,000$. In addition, as seen from the right figure, although separation of all eigenvalues fails for standard eigenproblems with $n \geq 4 \times 10^5$, enclosure of all eigenvalues is still possible in the case of $n = 10^6$. For the problem with $n = 10^6$, only 82 Gershgorin circles have intersections. If the eigenpairs corresponding to such circles have high accuracy, then it may be possible to guarantee that all eigenvalues are separated. Therefore, in future work, we will develop a method for the refinement of eigenpairs.

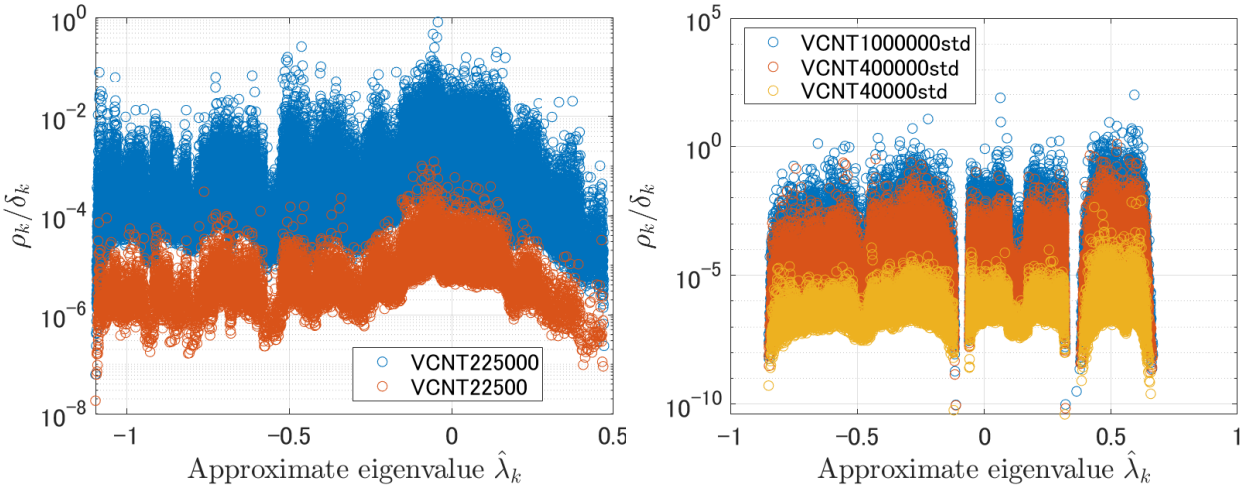

Figure 7.2: Ratio of difference δ_k and radius sum ρ_k on K computer

7.3 Conclusion of Part Ⅲ

In Part Ⅲ, we derived the new upper bounds for all eigenvalues. In addition, we implemented the verification method of all eigenvalues using the proposed error analysis for supercomputers. The advantages of the proposed method are as follows:

- The proposed method can be applied for any eigensolvers that can produce all eigenpairs.
- On the large-scale parallel systems, the proposed verification method has high strong scalability due to strong dependency on matrix multiplication.

We success to generate the intervals that enclose all eigenvalues of the matrix whose dimension is 10^6.

List of publications

Journal article

A1 T. Terao, K. Ozaki, T. Ogita, “LU-Cholesky QR algorithms for thin QR decomposition”, Accepted for Parallel Computing.

International conference proceedings

B1 T. Terao, K. Ozaki, “Verification of positive definiteness using approximate inverse matrix of computed Cholesky factor”, Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science.

B2 T. Terao, K. Ozaki, T. Ogita, “LU-Cholesky QR algorithms for thin QR decomposition in an oblique inner product”, Submitted for publication of proceedings of International Conference on Mathematics: Pure, Applied and Computation, 2019.

B3 T. Terao, K. Ozaki, T. Ogita, “Verified numerical computations for standard eigenvalue problems on supercomputer”, Accepted for publication of proceedings of the 38th JSST Annual International Conference on Simulation Technology, 2019.

conferences

C1 T. Terao, K. Ozaki, T. Ogita, “Preconditioned Cholesky QR Algorithms for Ill-conditioned Matrices”, Workshop on Large-scale Parallel Numerical Computing Technology (Kobe), 2019/6/7.

C2 T. Terao, K. Ozaki, T. Ogita, “Robust and efficient Cholesky QR algorithms for thin QR decomposition”, The 3rd UOG-SIT Workshop in Pure/Applied Mathematics and Computer Science (Guam), 2019/3/22.

C3 T. Terao, K. Ozaki, T. Ogita, “Robust Preconditioned Cholesky QR algorithms for ill-conditioned matrices on large-scale parallel systems”, 2019 Conference on Advanced Topics and Auto Tuning in High-Performance Scientific Computing (Taiwan), 2019/3/15.

C4 寺尾 剛史，尾崎 克久, 荻田 武史,「悪条件行列に対する Cholesky QR アルゴリズムとその比較」，2018 年度応用数理学会研究部会連合発表会 (大阪大学)，2019/3/5.

C5 T. Terao, K. Ozaki, Takeshi Ogita, “Thin QR Decomposition using LU Factors and its Refinement”, SIAM Conference on Computational Science and Engineering (Spokane), 2019/2/25.

C6 T. Terao, K. Ozaki, ”Generation of Test Matrices with Specified Eigenvalues on Parallel Distributed Computers”, The 37th JSST Annual International Conference on Simulation Technology (Muroran), 2018/9/18.

C7 寺尾 剛史，尾崎 克久, 荻田 武史,「LU 分解を用いた CholeskyQR アルゴリズムの丸め誤差解析」，2018 年度日本応用数理学会年会 (名古屋大学)，2018/9/3.

C8 T. Terao, K. Ozaki,“Generation of large scale matrices for numerical examples”, 10th International Workshop on Parallel Matrix Algorithms and Applications (Switzerland), 2018/6/27.

C9 T. Terao, K. Ozaki, T. Ogita,“ Rounding Error Analysis of QR Decomposition using LU Factors Based on CholeskyQR Algorithm”, IX Pan-American Workshop Applied Mathematics & Computational Science (Cuba), 2018/6/14.

C10 T. Terao, K. Ozaki,“Verification of Positive Definiteness of Symmetric Sparse Matrices”, 2018 Conference on Advanced Topics and Auto Tuning in High-Performance Scientific Computing (Taiwan), 2018/6/14.

C11 寺尾 剛史，尾崎 克久,「区間行列に対する正則性の高速な保証法」，2017 年度日本応用数理学会研究部会連合発表会 (電気通信大学)，2018/3/15.

C12 T. Terao, K. Ozaki,“ Validated Solution of Linear Systems for Real Symmetric and Positive Definite Matrices”, SIAM Conference on Parallel Processing for Science Computing (Waseda University), 2018/3/9.

C13 寺尾 剛史，尾崎 克久,「実対称正定値行列を係数行列とする連立一次方程式の数値解の高速精度保証法」，精度保証付き数値計算の実問題への応用研究集会発表会 (北九州)，2017/12/9.

C14 寺尾 剛史，尾崎 克久,「超大規模な線形数値計算に対する精度保証付き数値計算法の開発と実装」,「若手・女性利用者推薦」成果報告会 (東京大学)，2017/12/6.

C15 寺尾 剛史，尾崎 克久，南畑 淳史「連立一次方程式の数値解に対する高速精度保証法」，RIMS 共同研究（公開型） 数値解析学最前線 ー理論・方法・応用ー (京都大学)，2017/11/10.

C16 寺尾 剛史，尾崎 克久,「大規模疎行列を係数行列に持つ連立 1 次方程式の数値解に対する精度保証付き数値計算」，第 17 回 AT 研究会オープンアカデミックセッション (山梨大学)，2017/10/7.

C17 寺尾 剛史，尾崎 克久,「行列の正則性を高速に保証するための理論と実装法」，2017 年度日本応用数理学会年会 (武蔵野大学)，2017/9/7.

C18 寺尾 剛史，尾崎 克久,「実対称行列を係数行列とする連立 1 次方程式の数値解に対する精度保証付き数値計算」，Summer United Workshops on Parallel, Distributed and Cooperative Processing (秋田)，2017/7/27.

C19 寺尾 剛史，尾崎 克久,「行列の正則性を保証する高速な手法について」，第 26 回環瀬戸内ワークショップ (愛媛大学)，2017/7/22.

C20 寺尾 剛史，尾崎 克久,「超大規模な線形計算に対する精度保証付数値計算法の開発と評価」，学際大規模情報基盤共同利用・共同研究拠点，2017/7/13.

C21 T. Terao, K. Ozaki,“Verification of Positive Definiteness using Approximate Inverse Matrices of Computed Cholesky Factors”, International Conference on Computational and Mathematical Methods in Science and Engineering (Spain), 2017/7/4.

C22 T. Terao, K. Ozaki,“Fast verification methods for proving non-singularity of matrices”, 10th Summer Workshop on Interval Methods, and 3rd International Symposium on Set Membership - Applications, Reliability and Theory (England), 2017/6/14.

C23 寺尾 剛史，尾崎 克久,「ブロックコレスキー分解を用いた正定値性の保証法」，精度保証付き数値計算と高性能計算に関するワークショップ (東京女子大学)，2017/4/10.

Poster

D1 T. Terao, K. Ozaki, T. Ogita,“High-Performance Computing of Thin QR Decomposition on Parallel Systems”, International conference on high performance computing in Asia-Pacific Region (Germany), 2019/6/19.

D2 T. Terao, K. Ozaki, T. Ogita,“High-Performance Computing of Thin QR Decomposition on Parallel Systems”, International Supercomputing Conference (Germany), 2019/6/18.

Acknowledgment

I would like to express my appreciation to my thesis advisor Professor Katsuhisa Ozaki. I give special gratitude to Prof. Takeshi Ogita and Dr. Atsushi Minamihata for their thoughtful guidance.

These works used super high-performance computing environments for extreme research using computational resources of the K computer and other computers of the HPCI system provided by RIKEN R-CCS and Nagoya University through the HPCI System.

Bibliography

[1] G. H. Golub, C. F. Van Loan, Matrix Computations, 4th edition, Johns Hopkins University Press, 2013.

[2] Å. Björck, Solving linear least squares problems by gram-schmidt orthogonalization, BIT Numerical Mathematics 7 (1) (1967) 1–21.

[3] A. Stathopoulos, K. Wu, A block orthogonalization procedure with constant synchronization requirements, SIAM Journal on Scientific Computing 23 (6) (2002) 2165–2182.

[4] J. Demmel, L. Grigori, M. Hoemmen, J. Langou, Communication-optimal parallel and sequential qr and lu factorizations, SIAM Journal on Scientific Computing 34 (1) (2012) A206–A239.

[5] ANSI/IEEE, IEEE Standard for Floating-Point Arithmetic, New York (2008).

[6] P. S. Stanimirović, Generalizations of the condition number, Mathematica Balkanica 15 (2001) 35–48.

[7] T. Fukaya, Y. Nakatsukasa, Y. Yanagisawa, Y. Yamamoto, CholeskyQR2: a simple and communication-avoiding algorithm for computing a tall-skinny QR factorization on a large-scale parallel system, in: Proceedings of the 5th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems, IEEE Press, 2014, pp. 31–38.

[8] Y. Yamamoto, Y. Nakatsukasa, Y. Yanagisawa, T. Fukaya, Roundoff error analysis of the CholeskyQR2 algorithm, Electronic Transactions on Numerical Analysis 44 (2015) 306–326.

[9] N. J. Higham, Accuracy and Stability of Numerical Algorithms, 2nd ed., SIAM, 2002.

[10] V. Strassen, Gaussian elimination is not optimal, Numerische mathematik 13 (4) (1969) 354–356.

[11] D. Coppersmith, S. Winograd, Matrix multiplication via arithmetic progressions, Journal of symbolic computation 9 (3) (1990) 251–280.

[12] V. V. Williams, Multiplying matrices faster than coppersmith-winograd., in: STOC, Vol. 12, Citeseer, 2012, pp. 887–898.

[13] C.-P. Jeannerod, S. M. Rump, Improved error bounds for inner products in floating-point arithmetic, SIAM J. Matrix Anal. Appl. 34 (2013) 338–344.

[14] S. M. Rump, C.-P. Jeannerod, Improved backward error bounds for LU and Cholesky factorization, SIAM J. Matrix Anal. Appl. 35 (2014) 684–698.

[15] Advanpix, Multiprecision computing toolbox for MATLAB, ver. 3.8.3.8882 (2015).

[16] Y. Yamamoto, Y. Nakatsukasa, Y. Yanagisawa, T. Fukaya, Roundoff error analysis of the choleskyqr2 algorithm in an oblique inner product, JSIAM Letters 8 (2016) 5–8.

[17] T. Fukaya, R. Kannan, Y. Nakatsukasa, Y. Yamamoto, Y. Yanagisawa, Shifted choleskyqr for computing the qr factorization of ill-conditioned matrices, arXiv preprint arXiv:1809.11085.

[18] S. Oishi, S. M. Rump, Fast verification method of solution of matrix equations, Numer. Math. 90 (2002) 755–773.

[19] T. Ogita, S. M. Rump, S. Oishi, Verified solution of linear systems without directed rounding, Advance Research Institute for Science and Engineering 2005-04.

[20] T. Ogita, S. Oishi, Fast verified solutions of linear systems, IPSJ Trans. 46 (2005) 10–18, (in Japanese).

[21] K. Ozaki, T. Ogita, S. Oishi, An algorithm for automatically selecting a suitable verification method for linear systems, Numerical Algorithms 56 (2011) 363–382.

[22] N. J. Higham, Accuracy and Stability of Numerical Algorithms, 2nd edition, SIAM, 2002.

[23] T. Yamamoto, Error bounds for approximate solutions of systems of equations, Japan Journal of Applied Mathematics 1 (1) (1984) 157–171.

[24] S. Miyajima, Numerical enclosure for each eigenvalue in generalized eigenvalue problem, Journal of Computational and Applied Mathematics 236 (9) (2012) 2545–2552.

[25] S. Miyajima, Fast enclosure for all eigenvalues and invariant subspaces in generalized eigenvalue problems, SIAM Journal on Matrix Analysis and Applications 35 (3) (2014) 1205–1225.

[26] T. Hoshi, T. Ogita, K. Ozaki, T. Terao, An a posteriori verification method for generalized hermitian eigenvalue problems in large-scale electronic state calculations, submitted for publication (2019).

[27] A. Neumaier, A simple derivation of the hansen-bliek-rohn-ning-kearfott enclosure for linear interval equations, Reliable Computing 5 (2) (1999) 131–136.

[28] D. Lee, T. Miyata, T. Sogabe, T. Hoshi, S.-L. Zhang, An interior eigenvalue problem from electronic structure calculations, Japan Journal of Industrial and Applied Mathematics 30 (3) (2013) 625–633.

[29] T. Hoshi, H. Imachi, A. Kuwata, K. Kakuda, T. Fujita, H. Matsui, Numerical aspect of large-scale electronic state calculation for flexible device material, Japan Journal of Industrial and Applied Mathematics 36 (2) (2019) 685–698.

[30] T. Hoshi, Elses matrix library, `http://www.elses.jp/matrix/` (2019).

[31] T. Hoshi, S. Yamamoto, T. Fujiwara, T. Sogabe, S.-L. Zhang, An order-n electronic structure theory with generalized eigenvalue equations and its application to a ten-million-atom system, Journal of Physics: Condensed Matter 24 (16) (2012) 165502.