Matrix Computation and Application
− MATLAB Project 1 −

20190262 Jeongwoo Park

Contents

Factorization is one of the most useful technique in mathematics. There are several kind of factorization of matrices, for example, diagonalization, singular value decomposition, and QR decomposition. In this article, I'll introduce about QR decompositions and algorithms for to do this.

Terminologies

$\mathbb{C}^{m \times n}$ The matrix algebra consist of all m by n complex matrices.

$U(m)$ The matrix group consist of all m by m unitary matrices.

1 QR decomposition Reduced QR decomposition

Let's begin with the definition of QR decomposition.

Definition 1.1 (QR decomposition)**.** Let $A \in \mathbb{C}^{m \times n}$ be a matrix. If a unitary matrix $Q \in U(m)$ and a upper triangular matrix $R \in \mathbb{C}^{m \times n}$ satisfies the equality $A = QR$, then this form of equation is called a QR decomposition of a matrix A.

Sometimes, it is also convenient to consider a variation of the QR decomposition, which is called reduced QR decomposition.

Definition 1.2 (reduced QR decomposition). Let $A \in \mathbb{>} \times \mathbb{K}$ be a matrix. If a matrix $Q \in \mathbb{C}^{m \times n}$ consist of orthonormal columns and a upper triangular matrix $R \in \mathbb{C}^{n \times n}$ satisfies the equality $A = QR$, then this form of equation if called a <u>reduced QR decomposition</u>.

To distinguish two kind of QR decompositions, sometimes we use call <u>full QR decomposition</u> to denote the original QR decomposition. Then, how can we find a (reduced) QR decomposition?

Before to do that, notice that full QR decomposition and reduced QR decomposition are the same thing for square matrices. Also, if we can calculate QR decomposition for all square matrices, then we can find a QR decomposition of any matrix. For example, if $A \in \mathbb{C}^{m \times n}$ is a matrix and $m \geq n$, then we can consider a square matrix

$$\begin{pmatrix} A & 0 \end{pmatrix} \in \mathbb{C}^{m \times m}$$

by adding zero columns. Assume that $Q'R'$ be a QR decomposition of that matrix, then the equality

$$\begin{pmatrix} A & 0 \end{pmatrix} = Q \begin{pmatrix} R & R'' \end{pmatrix}$$

holds where $R \in \mathbb{C}^{m \times n}$ and $R'' \in \mathbb{C}^{m \times (m-n)}$. In this situation, the equality $A = QR$ gives a QR decomposition. For the case $m \leq n$, let's write

$$A = \begin{pmatrix} A' & A'' \end{pmatrix}$$

where $A' \in \mathbb{C}^{m \times m}$ and $A'' \in \mathbb{C}^{m \times (n-m)}$. If $A' = Q'R'$ is a QR decomposition, then we can obtain a QR decomposition

$$A = Q' \cdot \begin{pmatrix} R' & Q'^* \cdot R'' \end{pmatrix}$$

Similarly, we can obtain a reduced QR decomposition for any matrix. In conclusion, it is enough to consider the method to find a reduced QR decomposition, because a full QR decomposition can be immediately calculated from the reduced QR decomposition of a suitable matrix. In the next section, I'll introduce some method to calculate a reduced QR decomposition.

2 Algorithms for reduced QR decomposition

There are several algorithms to obtain a reduced QR decomposition, like <u>classical Gram–Schmidt process</u>, <u>modified Gram–Schmidt process</u>, and <u>Householder triangularization</u>. The built-in QR decomposition function of MATLAB is based on the Householder triangularization. Example MATLAB codes for classical and modified Gram–Schmidt processes are given in the next subsections.

2.1 Classical Gram–Schmidt process

```
1  function [Q,R] = cgs(A) % Classical Gram—Schmidt process for columns of a matrix
       A
2
3  [m,n] = size(A);
4  Q = zeros(m,n);
5  R = zeros(n,n);
6  N = 0;
7
8  for j = 1 : n
```

```matlab
     v = A(:,j);

     for i = 1 : j—1
         R(i,j) = dot(Q(:,i), A(:,j));
         v = v — R(i,j) * Q(:,i);
     end

     l = norm(v);
     disp(l)

     if l > 10^—18 & N < m % This is due to the error arose by the numerical
         approximation.
         R(j,j) = l;
         Q(:,j) = v/R(j,j);
         N = N + 1;
     end
 end

 end
```

2.2 Modified Gram–Schmidt process

```matlab
function [Q,R] = classical(A) % Classical Gram—Schmidt process for columns of a
    matrix A

[m,n] = size(A); % size of the given matrix A
Q = zeros(m,n);
R = zeros(n,n);
N = 0;

for j = 1 : n
    v = A(:,j);

    for i = 1 : j—1
        R(i,j) = dot(Q(:,i), v);
        v = v — R(i,j) * Q(:,i);
    end

    l = norm(v);
    disp(l)

    if l > 10^—18 & N < m % This is due to the error arose by the numerical
        approximation.
        R(j,j) = l;
        Q(:,j) = v/R(j,j);
        N = N + 1;
    end
```
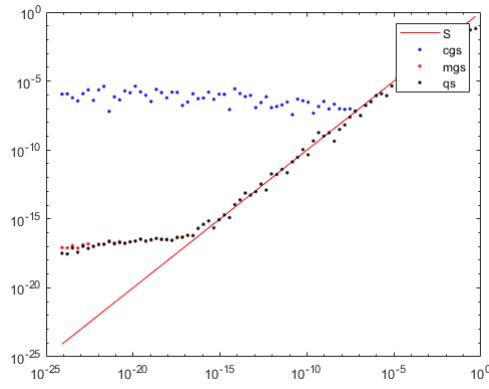
```
24   end
25
26   end
```
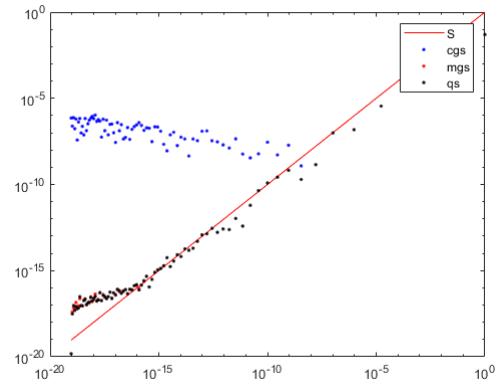
3 Numerical stability of the algorithms
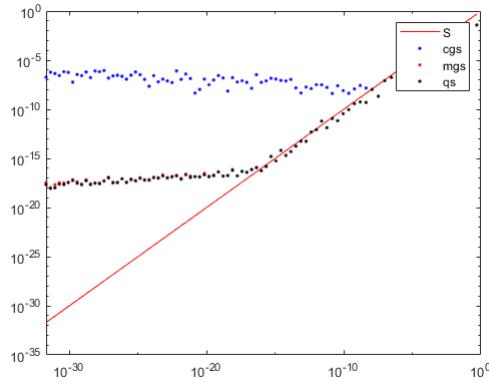
Theoretically, the three algorithms for QR decomposition don't have any problem. However, if we do the calculation including numerical errors, then we have to consider the numerical stability of each algorithm. It is well-known that the modified Gram–Schmidt process and Householder triangularization are numerically stable, but the classical Gram–Schmidt process does not. Here are some tables showing the errors of each algorithm.
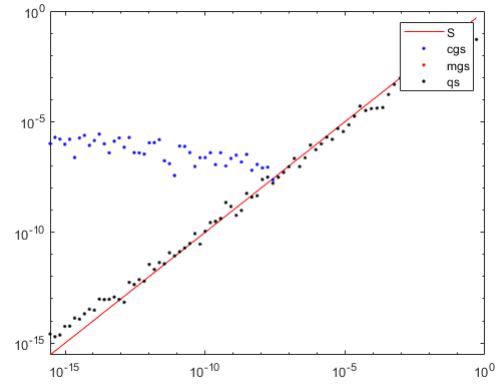

(a) $S = \text{diag} \left(2^{-i}\right)_{1 \leq i \leq 80}$

(b) $S = \text{diag} \left(i^{-10}\right)_{1 \leq i \leq 80}$

(c) $S = \text{diag} \left(i^{-4} 2^{-i}\right)_{1 \leq i \leq 80}$

(d) $S = \text{diag} \left(2^{-i^{0.9}}\right)_{1 \leq i \leq 80}$

Figure 1: Error plots of the classical Gram–Schmidt process (cgs), modified Gram–Schmidt process (mgs), and Householder triangularization (qs)

3.1 Analysis of the result

Why the modified Gram–Schmidt process gives better result? Let's estimate the error with acceptable assumptions. We can assume that the numerical error $\varepsilon_{i,j} \doteq \delta_{i,j} - \langle q_i | q_j \rangle$, where the sign $\doteq$ means the numerical approximation of the value. Simple calculation shows that

$$\langle q_{j'} | v_j \rangle \doteq \langle q_{j'} | u_j \rangle - \sum_{j=1}^{j-1} \langle q_i | u_j \rangle \cdot \langle q_{j'} | q_i \rangle \doteq \sum_{i=1}^{j-1} \langle q_i | u_j \rangle \cdot \varepsilon_{i,j'}$$

where the vector v_j is the j-th stage of the classical Gram–Schmidt process, i.e., $v_j = u_j - \sum_{i=1}^{j-1} \langle q_i | u_j \rangle \cdot q_i$. If we assume that $\{\varepsilon_{i,j}, q_i\}_{i,j}$ are independent uniform random variables such that the image of $\varepsilon_{i,j}$ is the interval $[-\varepsilon, \varepsilon]$ and the image of q_i is the sphere S^{n-1}, then the expectation of the value $\langle q_{j'} | v_j \rangle^2$ can be calculated as

$$\mathbb{E}\left(\left(\sum_{i=1}^{j-1} \langle q_i | u_j \rangle \cdot \varepsilon_{i,j'} \right)^2 \right) = \mathbb{E}\left(\sum_{i=1}^{j-1} \langle q_i | u_j \rangle^2 \varepsilon_{i,j'}^2 \right)$$

$$= \mathbb{E}\left(\sum_{i=1}^{j-1} \langle q_i | \hat{u}_j \rangle^2 \right) \cdot \|u_j\|^2 \cdot \frac{\varepsilon^2}{3}$$

$$= \frac{j-1}{3n} \cdot \varepsilon^2 \cdot \|u_j\|^2$$

where the hat denotes the normalization of a vector. For the modified Gram–Schmidt, it is more complicated, but we can show that the expectation is approximately

$$\frac{j-j'}{3n}\left(1 - \frac{1}{m} \right)^{j'-1} \cdot \varepsilon^2 \cdot \|u_j\|^2$$

when we treat the value ε^2 as a zero.[1] Hence, the error of the modified Gram–Schmidt process is much smaller that the error of the classical Gram–Schmidt process.

---

[1] This is reasonable because number terms including ε is not too large, but ε is very small.