Fall 2016 CSE/CS 383 HW#3 Po-Cheng Pan (UT EID: pp22828)

1. Let $A \in \mathbb{R}^{m\times n}$ be full rank and let $\{q_j\}_{j=1}^{n}$ be the orthonormal vectors from classical Gram-Schmidt orthogonalization of the columns of A

Let $P_j = q_j q_j^*$

$$W = \left[\prod_{j=1}^{i-1}(I-P_j)\right] A(:,i), \quad q_i = \frac{W}{\|W\|_2}, \quad i=1,\cdots,m$$

From the textbook, we know the classical Gram-Schmidt as below:

$$q_1 = \frac{A(:,1)}{\|A(:,1)\|_2} \quad q_2 = \frac{A(:,2) - q_1^* \cdot A(:,1)\, q_1}{\|A(:,2) - q_1^* A(:,1) \cdot q_1\|_2} \quad \cdots \quad q_n = \frac{A(:,n) - \sum_{i=1}^{n-1} q_i^* A(:,i) q_i}{\|A(:,n) - \sum_{i=1}^{n-1} q_i^* A(:,i) q_i\|_2}$$

& From (8.1) & (8.3)

q_i can be also expressed as $q_1 = \frac{A(:,1)}{\|A(:,1)\|_2}$ $\quad q_2 = \frac{P_2' A(:,2)}{\|P_2' A(:,2)\|_2} \quad \cdots \quad q_n = \frac{P_n' A(:,n)}{\|P_n' A(:,n)\|_2}$ where $P_j' = I - Q_{j-1} Q_{j-1}^*$

$$Q_{j-1} = [q_1 | q_2 | \cdots | q_{j-1}]$$

Then consider modified Gram-Schmidt algorithm, $P_j = q_j q_j^*$

$i=1$: $\prod_{j=1}^{0}(I-P_j) = I \Rightarrow \frac{W_1}{\|W_1\|_2} = \frac{A(:,1)}{\|A(:,1)\|_2}$

$i=2$: $\prod_{j=1}^{1}(I-P_j) = I - P_1 = I - q_1 q_1^* = I - Q_1 Q_1^* = P_2' \Rightarrow \frac{W_2}{\|W_2\|_2} = \frac{P_2' A(:,2)}{\|P_2' A(:,2)\|_2} = q_2$

We suppose when $i=k$, $\prod_{j=1}^{k-1}(I-P_j) = P_k' \Rightarrow \frac{W_k}{\|W_k\|_2} = q_k$

Consider $i=k+1$

$$\begin{aligned}
\prod_{j=1}^{k}(I-P_j) &= \prod_{j=1}^{k-1}(I-P_j) \cdot (I-P_k) \\
&= P_k' \cdot (I - q_k q_k^*) \\
&= (I - Q_{k-1} Q_{k-1}^*)(I - q_k q_k^*) \\
&= I - Q_{k-1}Q_{k-1}^* - q_k q_k^* + Q_{k-1}Q_{k-1}^* q_k q_k^*
\end{aligned}$$

($\because$ $q_1 \cdot q_2 \cdots q_k$ all are orthonormal $\therefore Q_{k-1}Q_{k-1}^* q_k q_k^* = 0$)

$$= I - Q_{k-1}Q_{k-1}^* - q_k q_k^*$$

$$= I - (Q_{k-1}Q_{k-1}^* + q_k q_k^*)$$

($\because$ $Q_{k-1}Q_{k-1}^*$ & $q_k q_k^*$ both are $\mathbb{R}^{n\times n}$ $\therefore$ We can combine q_k into Q_{k-1} as $Q_k = [q_1 | q_2 | \cdots | q_k]$ s.t. $Q_{k-1}Q_{k-1}^* + q_k q_k^* = Q_k Q_k^*$)

$$= I - Q_k Q_k^*$$

$$= P_{k+1}' \Rightarrow \frac{W_{k+1}}{\|W_{k+1}\|_2} = q_{k+1}$$

By induction, we prove $q_i = \frac{W}{\|W\|_2}$, $i=1,\cdots,m$. Hence, both classical and modified algorithm are mathematically equivalent. //

Po-Cheng Pan
UT EID: pp22828

Homework 3

Fall 2016, CSE/CS 383, Linear Algebra

Coding Parts:

2. **Write a single Matlab function [Q,R] = gramschmidt(A,flag) that computes a reduced QR fac- torization A = QR of an m × n full rank matrix A with m ≥ n using the classical Gram Schmidt orthogonalization when flag == true and the modified Gram Schmidt orthogonalization when flag == false or flag is left unspecified. Q should be an m × n matrix and R an n × n matrix.**

Ans:

Code:

```
%%gramschmidt
function[Q, R] = gramschmidt(A, flag)
if ~exist('flag', 'var')
    flag = false;
end
[m, n] = size(A);
Q = zeros(m, n);
R = zeros(n, n);
% Classical Gram-Schmidt
if (flag == true)
    for j = 1:n
        v = A(:, j);
        for i = 1:(j-1)
            R(i,j) = Q(:,i)'*A(:,j);
            v = v - R(i, j)'*Q(:,i);
        end
        R(j,j) = norm(v, 2);
        Q(:, j) = v/R(j,j);
    end
% Modified Gram-Schmidt
else
    for i = 1:n
        R(i, i) = norm(A(:,i),2);
        Q(:, i) = A(:,i)/R(i,i);
        for j = i+1:n
            R(i,j) = Q(:,i)'*A(:,j);
            A(:, j) = A(:, j) - R(i, j)'*Q(:,i);
        end
    end
end

```

3. **Suggest a test (or tests) to (quantitatively) check the accuracy of your QR factorization. Apply the test(s) to QR factorizations obtained by the two Gram Schmidt variants and Matlab's [Q,R]=qr(A) to the following matrices: A = gallery('randsvd',100,kappa); for kappa=1,1E3,1E6, 1E9. Discuss your results.**

Ans:

I consider three different directions to check the accuracy of my QR factorization : Accuracy, Orthogonality and whether R is upper triangular or not.

For these three directions, I have following definitions to evaluate them.

Accuracy : error = $\|QR-A\|_2 / \|A\|_2$

Orthogonality: $\|Q^*Q - I\|_2 / \|I\|_2$

R is upper triangular or not : Using the Matlab function : istriu

In these tests, test matrices are A = gallery('randsvd',100,kappa); for kappa=1,1E3,1E6, 1E9. (kappa is the value of the condition number.) Besides, to be general, I ran each test for 100 times and then calculated their averages. The test results are as following three tables.

Accuracy:

	Matlab	Classical	Modified
1	1.8851E-15	5.9620E-16	6.1756E-16
1E+03	8.0518E-16	2.2419E-16	2.2264E-16
1E+06	7.1646E-16	1.7984E-16	1.7920E-16
1E+09	7.0958E-16	1.6370E-16	1.6333E-16

Orthogonality:

	Matlab	Classical	Modified
1	2.4466E-15	1.0949E-15	1.0256E-15
1E+03	2.4262E-15	7.8979E-12	6.2346E-14
1E+06	2.3803E-15	6.2346E-05	3.4670E-11
1E+09	2.4021E-15	1.1393E+01	2.6848E-08

Upper Triangular:

	Matlab	Classical	Modified
1	The R matrices of all tests are still upper triangular matrices		
1E+03			
1E+06			
1E+09			

Discussion:

From the test results, we can find that

1. As the condition number increases, the accuracy of classical and modified QR method is better than the Matlab intrinsic QR function. And the errors of modified QR method are a little lesser than the errors of simplified QR method when the condition number is large
2. However, as the condition number increases, the orthogonality of Matlab intrinsic function is much better than other two methods. In addition, the orthogonality of modified method is better than the classical method.
3. The R matrices created by all methods all satisfy the conditions of upper triangular matrices.

Test Code: (P.S. All my codes are also uploaded to Canvas)

```
%% Main for Question 2
clc; clear; close all;

repeattime = 100;
kappa = [1,1e3, 1e6, 1e9];
error = zeros(3, size(kappa,2));
orthogonality = zeros(3, size(kappa,2));
isUpperTriangular = zeros(3, size(kappa,2));

for j = 1 : repeattime
    for i = 1:size(kappa,2)

    A = gallery('randsvd', 100, kappa(i));
    [Q_matlab, R_matlab] = qr(A);

    error(1, i) =  error(1, i)+ norm((A-Q_matlab*R_matlab))/norm(A);
    orthogonality(1, i) = orthogonality(1, i)+ norm(Q_matlab*Q_matlab'-eye(size(Q_matlab)))/norm(eye(size(Q_matlab)));
    isUpperTriangular(1, i) = isUpperTriangular(1, i) + istriu(R_matlab);

    [Q_classical, R_classical] = gramschmidt(A, true);

    error(2, i) = error(2, i)+ norm((A-Q_classical*R_classical))/norm(A);
    orthogonality(2, i) =orthogonality(2, i)+  norm(Q_classical*Q_classical'-eye(size(Q_classical)))/norm(eye(size(Q_classical)));
    isUpperTriangular(2, i) = isUpperTriangular(2, i) + istriu(R_classical);

    [Q_modified, R_modified] = gramschmidt(A);

    error(3, i) = error(3, i) + norm((A-Q_modified*R_modified))/norm(A);
    orthogonality(3, i) = orthogonality(3, i) +  norm(Q_modified*Q_modified'-eye(size(Q_modified)))/norm(eye(size(Q_modified)));
    isUpperTriangular(3, i) = isUpperTriangular(3, i)+ istriu(R_modified);

    end

end

error = error./repeattime;
orthogonality = orthogonality./repeattime;
isUpperTriangular = isUpperTriangular./repeattime;
```