CSE383C: Assignment 3

Brady Zhou

September 22, 2016

Exercise 1. Show the modified Gram-Schmidt algorithm is equivalent to the classical algorithm.

Show
1. $w_m = \left[\prod_{j=1}^{i-1} (I - P_j) \right] a_j, \; P_j = q_j^* q_j, \; q_i = \frac{w_m}{\|w_m\|_2}$
2. $w_c = a_j - \sum_{j=1}^{i-1} \lambda_j q_j, \; \lambda_j = q_j^* a_i, \; q_i = \frac{w_c}{\|w_c\|_2}$
yield equivalent $q_i \; \forall 1 \leq i \leq n$.

Proof. 1 and 2 are equivalent.

It is sufficient to show that $w_m = w_c$.

Consider w_c. $(\star)$

$$
\begin{aligned}
w_c &= a_j - \sum_{j=1}^{i-1} \lambda_j q_j, \; \lambda_j = q_j^* a_i \\
&= a_j - \sum_{j=1}^{i-1} (q_j^* a_i) q_j \\
&= a_j - \sum_{j=1}^{i-1} q_j (q_j^* a_i) \text{ Since } q_j^* a_i \text{ is a scalar.} \\
&= a_j - \sum_{j=1}^{i-1} (q_j q_j^*) a_i \text{ Associativity matrix vector multiplication.} \\
&= (I - \sum_{j=1}^{i-1} (q_j q_j^*)) a_i \text{ Distributivity over matrix vector.}
\end{aligned}
$$

Lemma. Show that $\prod_{j=1}^{i-1} (I - q_j q_j^*) = (I - \sum_{j=1}^{i-1} (q_j q_j^*))$.
Proof by induction.

Let $i = 3$;

$$\begin{aligned}
\prod_{j=1}^{2}(I - q_j q_j^*) &= (I - q_1 q_1^*)(I - q_2 q_2^*) \\
&= I^2 - q_2 q_2^* - q_1 q_1^* + (q_1 q_1^*)(q_2 q_2^*) \\
&= I - q_2 q_2^* - q_1 q_1^* + (q_1 q_1^*)(q_2 q_2^*) \\
&= I - q_2 q_2^* - q_1 q_1^* + q_1(q_1^* q_2) q_2^* \text{ Associativity of matrix vector multiplication.} \\
&= I - q_2 q_2^* - q_1 q_1^* + q_1(0) q_2^* \text{ Inner product of orthogonal vectors is 0.} \\
&= I - q_2 q_2^* - q_1 q_1^* \\
&= I - \sum_{j=1}^{2} q_j q_j^*
\end{aligned}$$

Assume for some $i = k$, that $\prod_{j=1}^{k-1}(I - q_j q_j^*) = (I - \sum_{j=1}^{k-1}(q_j q_j^*))$.

Let $i = k + 1$.

$$\begin{aligned}
\prod_{j=1}^{k}(I - q_j q_j^*) &= \Big(\prod_{j=1}^{k-1}(I - q_j q_j^*)\Big)(I - q_k q_k^*) \\
&= \Big(I - \sum_{j=1}^{k-1}(q_j q_j^*)\Big)(I - q_k q_k^*) \text{ By induction hypothesis.} \\
&= I^2 - q_k q_k^* - \sum_{j=1}^{k-1}(q_j q_j^*) + \sum_{j=1}^{k-1}(q_j q_j^*)(q_k q_k^*) \\
&= I - q_k q_k^* - \sum_{j=1}^{k-1}(q_j q_j^*) + \sum_{j=1}^{k-1}(q_j q_j^*)(q_k q_k^*) \\
&= I - q_k q_k^* - \sum_{j=1}^{k-1}(q_j q_j^*) + \sum_{j=1}^{k-1} q_j(q_j^* q_k) q_k^* \text{ Associativity of matrix vector multiplication.} \\
&= I - q_k q_k^* - \sum_{j=1}^{k-1}(q_j q_j^*) + \sum_{j=1}^{k-1} q_j(0) q_k^* \text{ Inner product of orthogonal vectors is 0.} \\
&= I - q_k q_k^* - \sum_{j=1}^{k-1}(q_j q_j^*) \text{ Inner product of orthogonal vectors is 0.} \\
&= I - \sum_{j=1}^{k-1}(q_j q_j^*) - q_k q_k^* \text{ Commutatitivity of matrix vector addition.} \\
&= I - \sum_{j=1}^{k}(q_j q_j^*)
\end{aligned}$$

We see that $\prod_{j=1}^{i-1}(I - q_j q_j^*) = (I - \sum_{j=1}^{i-1}(q_j q_j^*))\ \forall 1 \leq i \leq n$.

Consider w_m. $(\star\star)$

$$\begin{aligned} w_m &= \left[\prod_{j=1}^{i-1}(I - q_j^* q_j)\right]\ a_j \\ &= \left(I - \sum_{j=1}^{i-1}(q_j q_j^*)\right)\ a_j \text{ By Lemma.} \end{aligned}$$

By $(\star)$ and $(\star\star)$, we have shown equality and can see the two algorithms yield the same Q. $\square$

Exercise 2. Implement the classical and modified Gram-Schmidt Orthogonalization.

```
function [Q, R] = gramschmidt(A, flag)
% Decomposes a matrix A into Q, orthogonal and R, upper triangular.
%
% Params:
%   A: m x n matrix, full rank.
%   flag: [0, 1], 0 for classical, 1 for modified.
%
% Returns:
%   Q: m x n matrix, orthogonal, spans A.
%   r: n x n matrix, upper triangular.

[m, n] = size(A);

Q = zeros(m, n);
R = zeros(n, n);

for j = 1:n
    V(:, j) = A(:, j);
    for i = 1:j-1
        if flag == 1
            R(i, j) = Q(:, i)' * A(:, j);
        else
            R(i, j) = Q(:, i)' * V(:, j);
        end
        V(:, j) = V(:, j) - R(i, j) * Q(:, i);
    end
    R(j, j) = norm(V(:, j));
    Q(:, j) = V(:, j) / R(j, j);
end
```

Exercise 3. Suggest tests to judge the numerical accuracy of the algorithms.
The main problem that can arise from this algorithm is rounding errors from the normalization step. To test the accuracy of this algorithm, I checked the 'orthogonality' of the column vectors of Q. I found the maximum inner product value for every pair of column vectors and that was the score. The 'better' the orthogonality, the lower the value.

```
function epsilon = gramschmidt_verification(Q)
% Finds the greatest inner product between the column vectors.
%
% Params:
%   Q: m x n matrix, the Q from a QR decomposition.
%
% Returns:
%   epsilon: a scalar.

[m, n] = size(Q);

epsilon = 0;

for i = 1:n
    for j = i+1: n
        epsilon = max(worst, Q(:, i)' * Q(:, j));
    end
end
```

The results from the tests are as follows.

```
Kappa: 1
Classical:    1.110223e-16
Modified:    1.110223e-16
Matlab:        4.718448e-16

Kappa: 1000
Classical:    1.789457e-12
Modified:    2.171874e-14
Matlab:        4.302114e-16

Kappa: 1000000
Classical:    4.655908e-06
Modified:    9.193776e-12
Matlab:        4.857226e-16

Kappa: 1000000000
Classical:    9.381781e-01
```

```
Modified:    7.949187e-09
Matlab:        4.996004e-16
```

We can see as the condition number increases, the numerical error of the classical algorithm increases. This makes sense, since the condition number tells the limit of error given a small perturbation of the function. The classical algorithm error then compounds on itself, and the higher the condition number the more noticable the numerical accuracy.