W.D. Henshaw **Math 6800: Solutions for Problem Set 3**

1. NLA exercise 6.5 *Let $P \in \mathbb{C}^{m\times m}$ be a nonzero projector. Show that $\|P\|_2 \ge 1$, with equality if and only if P is an orthogonal projector.*

Solution:
Solution 1:

(1) To show that $\|P\|_2 \ge 1$, use $\|AB\|_2 \le \|A\|_2\|B\|_2$:

$$\|P\|_2 = \|P^2\|_2 \le \|P\|_2^2 \ \rightarrow \|P\|_2 \ge 1 \text{ since } P \ne 0.$$

(2) To show that $P = P^*$ implies $\|P\|_2 = 1$, take $P = P^*$. Then Pv is orthogonal to $(I-P)v$ since $(Pv)^*(I-P)v = v^*(P-P^2)v = 0$ and thus

$$\|v\|_2^2 = \|Pv + (I-P)v\|_2^2 = \|Pv\|_2^2 + \|(I-P)v\|_2^2,$$

and thus $\|Pv\|_2 \le \|v\|_2$ which implies $\|P\|_2 \le 1$. But $\|P\|_2 \ge 1$ and therefore $\|P\|_2 = 1$ if $P = P^*$.

(3) To show that $\|P\|_2 = 1$ implies $P = P^*$, assume $\|P\|_2 = 1$. We know that $P = P^*$ iff $S_1 = \text{range}(P)$ is orthogonal to $S_2 = \text{range}(I-P)$. Thus we will show that S_1 is orthogonal to S_2.

Proof by contradiction. Suppose $\|P\|_2 = 1$ but S_1 is not orthogonal to S_2. Then there exists $v \ne 0$, with $v \perp S_2$ where $v \notin S_2$, $v \notin S_1$, i.e. $Pv \ne v$ (if there is no such v then S_1 is orthogonal to S_2 by dimension arguments). Now $v \perp S_2$ implies v is orthogonal to $(I-P)v \in S_2$ and thus

$$\|Pv\|_2^2 = \|v - (I-P)v\|_2^2 = \|v\|_2^2 + \|(I-P)v\|_2^2 > \|v\|_2^2.$$

But this means $\|P\|_2 > 1$ which is a contradiction. Therefore $\|P\|_2 = 1$ implies S_1 is orthogonal to S_2 which implies $P = P^*$.

2. NLA exercise 7.1: *Consider again the matrices A and B of Exercise 6.4,*

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

(a) Using any method you like, determine (on paper) a reduced QR factorization $A = \hat{Q}\hat{R}$ and a full QR factorization $A = QR$.
(b) Repeat for B.

Solution:

(a) Using Gram-Schmidt, $r_{11} = \|a_1\|_2 = s\sqrt{2}$, and

$$q_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

Then $r_{12} = q_1^* a_2 = 0$ and $r_{22} = 1$ to give the reduced QR factorization

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, = \hat{Q}\hat{R}, \qquad \hat{Q} = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 \\ 0 & 1 \\ \frac{1}{\sqrt{2}} & 0 \end{bmatrix}, \qquad \hat{R} = \begin{bmatrix} \sqrt{2} & 0 \\ 0 & 1 \end{bmatrix}.$$

We can extend to full QR factorization

$$Q = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & 1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \end{bmatrix}, \qquad R = \begin{bmatrix} \sqrt{2} & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},$$

(b) Using Gram-Schmidt, $r_{11} = \sqrt{2}$ *and*

$$q_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

Then $r_{12}q_1 + r_{22}q_2 = b_2$ *implies* $r_{12} = q_1^* b_2 = \sqrt{2}$ *and* $r_{22} = \|b_2 - r_{12}q_1\|_2 = \sqrt{3}$ *to give*

$$B = \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, = \hat{Q}\hat{R}, \qquad \hat{Q} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ 0 & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{3}} \end{bmatrix}, \qquad \hat{R} = \begin{bmatrix} \sqrt{2} & \sqrt{2} \\ 0 & \sqrt{3} \end{bmatrix}.$$

We can extend to full QR factorization

$$Q = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} \end{bmatrix}, \qquad R = \begin{bmatrix} \sqrt{2} & \sqrt{2} \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix}.$$

3. *NLA exercise 7.5* Let A be an $m \times n$ matrix ...
Let $A = \hat{Q}\hat{R}$ *be the reduced QR factorization of* A

(a)

$$A = \left[\begin{array}{c|c|c|c} q_1 & q_2 & \cdots & q_n \end{array}\right] \begin{bmatrix} r_{11} & r_{12} & \cdots & \\ 0 & r_{22} & r_{23} & \cdots \\ & & \vdots & \\ & & & r_{nn} \end{bmatrix},$$

A has full rank iff the columns of A are linearly independent, iff $Ax = 0$ *implies* $x = 0$*. Now* $Ax = 0$ *iff* $\hat{Q}\hat{R}x = 0$ *iff (multiply by* $\hat{Q}^*$*)* $\hat{R}x = 0$*. Thus A has full rank iff* $\hat{R}$ *has full rank iff* $r_{ii} \neq 0$, $i = 1, 2, \ldots, n$ *(*$\hat{R}$ *is nonsingular iff* $det(\hat{R}) = \prod_{i=1}^n r_{ii} \neq 0$*).*

(b) In (a) we showed that the $rank(A) = rank(\hat{R}) = rank(R)$*. The rank of* $\hat{R}$ *is at least equal to the number of non-zero diagonal entries in* $\hat{R}$ *since it is easy see that these columns are linearly independent since if* $r_{ii} > 0$ *then column* $a_i \notin < a_1, a_2, \ldots, a_{i-1} >$*. In addition there could be more linearly independent columns. Consider for example* $\hat{R}$ *with zeros on the diagonal and one's on the first super diagonal in which case the rank is* $n-1$*. Therefore* $rank(A) \geq k$ *and the rank can be greater than* k *in some cases.*

4. *Write matlab functions* `[Qc, Rc]=clgs(A)` *and* `[Qm, Rm]=mgs(A)` *that implement the reduced QR factorization using the classical Gram-Schmidt and modified Gram-Schmidt algorithms, respectively. Test the implementations (in parts (a) and (b) below) by computing the QR factorization* *for the $m \times m$ Vandermonde matrix*

$$A = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{m-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{m-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^{m-1} \end{bmatrix},$$

for points $x_i \equiv (i-1)/(m-1)$, and compare to the results from the built-in Matlab function `[Q,R]=qr(A)`.

WARNING: The matlab `qr` *algorithm may return R with negative diagonal entries, r_{ii}. If $r_{ii} < 0$, you can change the sign of column i of Q and row i of R to obtain a QR factorization with r_{ii} positive.*

(a) For $m = 5$, compute $\|A - QR\|_2$ for each of the three approximations. Also compute the 2-norm differences $\|Qi - Q\|_2$, $\|Ri - R\|_2$, for $i = c$ and $i = m$, and also compute the error $\|Q^*Q - I\|_2$ for each of the three approximations to Q. (Hint: for $m = 5$ the errors should all be small).

(b) Repeat (a) but with $m = 100$.

(a) Here are the results for $m = 5$. Codes are given below.

Listing 1: Results from (a) $m = 5$.

```
>> ps3
clgs :  m=5: ||A-QR||=1.13e-16, ||Qc-Q|| = 1.79e-14, ||Rc-R||=5.11e-16, ||Qc*Qc-I||=1.92e-14
mgs :  m=5: ||A-QR||=1.16e-16, ||Qm-Q|| = 1.26e-14, ||Rm-R||=5.66e-16, ||Qm*Qm-I||=1.27e-14
Matlab QR: || A - Q*R || = 9.61e-16, ||Q*Q-I|| = 6.26e-16
```

(b) Here are the results for $m = 100$. Note that in all cases the product QR is close to A but that the columns of Q from either Gram-Schmidt are far from orthornormal.

Listing 2: Results from (b) $m = 100$.

```
>> ps3
clgs :m=100: ||A-QR||=1.04e-14, ||Qc-Q|| = 9.53, ||Rc-R||=2.36e+01, ||Qc*Qc-I||=8.99e+01
mgs :m=100: ||A-QR||=1.97e-15, ||Qm-Q|| = 1.97, ||Rm-R||=4.55e-03, ||Qm*Qm-I||=1.01e+00
Matlab QR: || A - Q*R || = 9.76e-15, ||Q*Q-I|| = 2.58e-15
```

Listing 3: Classical Gram-Schmidt, clgs.m

```
function [Q,R] = clgs( A )
%
% Classical Gram-Schmidt :
%     Compute the reduced QR factorization : A = Q R
%
%  A (input) : m x n matrix
%  Q (output) : m x n matrix with orthonormal columns
%  R (output) : n x n matrix, upper triangular
%

[m,n]=size(A);
R=zeros(n,n);
Q=zeros(m,n);
I=1:m;        % define an index range 1...m
```

```

for j=1:n
  vj = A(I,j);
  for i=1:j-1
    R(i,j) = dot(Q(I,i),A(I,j));
    % R(i,j) = Q(I,i)'*A(I,j); % same as above
    vj = vj - R(i,j)*Q(I,i);
  end
  R(j,j)= norm(vj,2); % 2-norm
  Q(I,j)=vj/R(j,j);
end
```

Listing 4: Modified Gram-Schmidt, mgs.m

```
function [Q,R] = mgs( A )
%
%  Modified Gram-Schmidt :
%      Compute the reduced QR factorization : A = Q R
%
%   A (input) : m x n matrix
%   Q (output) : m x n matrix with orthonormal columns
%   R (output) : n x n matrix, upper triangular
%

[m,n]=size(A);
R=zeros(n,n);
Q=zeros(m,n);
I=1:m;       % define an index range 1...m

V=A;
for i=1:n
  R(i,i)= norm(V(I,i),2); % 2-norm
  Q(I,i)=V(I,i)/R(i,i);
  for j=i+1:n
    R(i,j) = dot(Q(I,i),V(I,j));
    V(I,j) = V(I,j) - R(i,j)*Q(I,i);
  end
end
```

Listing 5: Problem set 3, Exercise 4.

```
% Problem set 3

clear; clf; % clear variables and figures
set(gca,'FontSize',14);

% --- Construct the Vandermonde matrix:
m=5;
m=100;
x=(0:m-1)'/(m-1); % grid for [0,1]
A=x.^0;
for k=1:m-1
  A = [A x.^k]; % Vandermonde matrix
end;
```

```
% or use A = fliplr(vander(x))

% Matlab qr:
[Q,R] = qr(A); % QR

for k=1:m
  if R(k,k)<0
    % flip the sign of column qk and row k of R
    Q(1:m,k)=-Q(1:m,k);
    R(k,1:m)=-R(k,1:m);
  end
end

% Classical Gram-Schmidt:
[Qc,Rc] = clgs(A);

% Modified Gram-Schmidt:
[Qm,Rm] = mgs(A);

fprintf('clgs:␣m=%d:␣||A-QR||=%8.2e,␣||Qc-Q||␣=␣%8.2e,␣||Rc-R||=%8.2e,␣||Qc*Qc-I||=%8.2e␣\n',m,norm(A-Qc*Rc,2),norm(Qc-Q,2), norm(Rc-R,2),norm(Qc'*Qc-eye(m),2));
fprintf('mgs␣:␣m=%d:␣||A-QR||=%8.2e,␣||Qm-Q||␣=␣%8.2e,␣||Rm-R||=%8.2e,␣||Qm*Qm-I||=%8.2e␣\n',m,norm(A-Qm*Rm,2),norm(Qm-Q,2), norm(Rm-R,2),norm(Qm'*Qm-eye(m),2));

fprintf('Matlab␣QR:␣||␣A␣-␣Q*R␣||␣=␣%8.2e,␣||Q*Q-I||␣=␣%8.2e\n',norm(A-Q*R,2), norm(Q'*Q-eye(m)));
```