W.D. Henshaw

Math 6800: Solutions for Problem Set 4

1. Consider the Householder reflector,

$$F = I - 2uu^*, \quad u^*u = 1.$$

Determine the eigenvalues and eigenvectors, determinant, and singular values of F.

Solution:

Let λ be an eigenvalue of F and $x \neq 0$ the corresponding eigenvector, then since $F^2 = I$,

$$Fx = \lambda x \quad \rightarrow x = Ix = F^2 x = \lambda F x = \lambda^2 x$$

and thus $\lambda^2 = 1$ which implies $\lambda = 1$ or $\lambda = -1$. For $\lambda = -1$,

$$Fx = -x \quad \rightarrow x - 2uu^* x = -x \quad \rightarrow x = (u^*x)u$$

Thus $x = \beta u$ is an eigenvector for $\beta \neq 0$ for $\lambda = -1$. For $\lambda = 1$,

$$Fx = -x \quad \rightarrow x - 2uu^* x = x \quad \rightarrow (u^*x)u = 0$$

which implies $u^*x = 0$. Any vector orthogonal to u will be an eigenvector. We can construct an orthonormal basis q_i, $i = 1, 2, \ldots, m-1$ of vectors orthogonal to u that are eigenvectors for $\lambda = 1$.

The determinant is the product of the eigenvalues and since there is one eigenvalue -1 and $m-1$ eigenvalues of $+1$, then

$$\det(F) = -1(1)^{m-1} = -1.$$

Since $F^*F = I$, the singular values are all $\sigma_i = 1$, $i = 1, 2, \ldots .m$.

2. General Householder reflector. Let $x, y \in \mathbb{C}^m$, with $m > 1$. Show explicitly (using algebra) that if $\|x\|_2 = \|y\|_2$ then there is Householder reflector $F = I - 2uu^*$, $\|u\|_2 = 1$, such that $Fx = \alpha y$ where $\alpha = \pm\text{sign}(y^*x)$. Note: if $z = re^{i\theta} \in \mathbb{C}$, with $r, \theta \in \mathbb{R}$ and $r \geq 0$ then $\text{sign}(z) = e^{i\theta}$, $\text{sign}(0) \equiv 1$). (Hint: if $x \neq \alpha y$ consider $v = x - \alpha y$, $u = v/\|v\|_2$)

Solution:

If $x = y = 0$ then choose any u for F. Otherwise suppose $x = \alpha y$. If $\alpha = -1$ choose $u = x/\|x\|_2$ so that $Fx = -x = \alpha y$. If $\alpha = 1$ then choose u to be a vector orthogonal to x (we can do this if $m > 1$) and then $Fx = x = \alpha y$.

If $x \neq \alpha y$ choose

$$v = x - \alpha y, \qquad u = \frac{x - \alpha y}{\|x - \alpha y\|_2},$$

where $\alpha = \pm\text{sign}(y^*x)$. Note that

$$Fx - \alpha y = (I - 2\frac{vv^*}{v^*v})x - \alpha y = x - \alpha y - 2\frac{vv^*x}{v^*v} = (1 - 2\frac{v^*x}{v^*v})v$$

Thus $Fx = \alpha y$ provided

$$v^*x = \frac{1}{2}(v^*v) \quad \text{(to prove)}.$$

Now

$$v^*v = (x - \alpha y)^*(x - \alpha y) = x^*x - \bar{\alpha}y^*x - \alpha x^*y + y^*y = 2(x^*x - \bar{\alpha}y^*x),$$
$$v^*x = (x - \alpha y)^*x = x^*x - \bar{\alpha}y^*x = \frac{1}{2}v^*v$$

where we have used $\|x\|^2 = x^*x = y^*y = \|y\|^2$, $\bar{\alpha}\alpha = |\alpha|^2 = 1$, and $\bar{\alpha}y^*x = \alpha x^*y = \pm|x^*y|$ since $\alpha = \pm\text{sign}(y^*x)$. Thus $v^*x = \frac{1}{2}(v^*v)$ which proves the result.

3. Write a Matlab function `[W,R] = house(A)` that computes an implicit representation of a full or reduced QR factorization for $A \in \mathbb{C}^{m\times n}$ with $m \geq n$ using Householder reflections. The output variables are a lower triangular matrix $W \in \mathbb{C}^{m\times n}$ whose columns are the Householder vectors v_k, and an upper triangular matrix $R \in \mathbb{C}R^{n\times n}$.

Also write a Matlab function `Q = formQ(W)` that takes the matrix W from `house` and generates the full matrix $Q \in \mathbb{C}^{m\times m}$.

(a) Test your program on the Vandermonde matrix from problem set 3 with $m = 5$. Compare Q and R from the Matlab function `[Q,R]=qr(A)` to the output from `house` and `formQ`. Print Q and R for each case along with $\|A - QR\|_2$, and $\|Q^*Q - I\|_2$ for each factorization.

Solution: Here are the results. The codes are below. Apart from sign's the answers all agree.

```
>> ps4
A =
   1.000000000000000                   0                   0                   0                   0
   1.000000000000000   0.250000000000000   0.062500000000000   0.015625000000000   0.003906250000000
   1.000000000000000   0.500000000000000   0.250000000000000   0.125000000000000   0.062500000000000
   1.000000000000000   0.750000000000000   0.562500000000000   0.421875000000000   0.316406250000000
   1.000000000000000   1.000000000000000   1.000000000000000   1.000000000000000   1.000000000000000

Q =

  -0.447213595499958  -0.632455532033676   0.534522483824849  -0.316227766016837  -0.119522860933439
  -0.447213595499958  -0.316227766016838  -0.267261241912425   0.632455532033676   0.478091443733757
  -0.447213595499958   0.000000000000000  -0.534522483824848  -0.000000000000001  -0.717137165600636
  -0.447213595499958   0.316227766016838  -0.267261241912424  -0.632455532033675   0.478091443733758
  -0.447213595499958   0.632455532033676   0.534522483824849   0.316227766016838  -0.119522860933440


R =

  -2.236067977499790  -1.118033988749895  -0.838525491562421  -0.698771242968684  -0.618412550027285
                   0   0.790569415042095   0.790569415042095   0.760923061978017   0.731276708913938
                   0                   0   0.233853586673371   0.350780380010057   0.415507712035722
                   0                   0                   0   0.059292706128157   0.118585412256314
                   0                   0                   0                   0  -0.011205268212510


W =

   0.850650808352040                   0                   0                   0                   0
   0.262865556059567  -0.748594769695856                   0                   0                   0
   0.262865556059567   0.130537585551298  -0.942100351559142                   0                   0
   0.262865556059567   0.341751835782647  -0.334282967079670  -0.800389015572086                   0
   0.262865556059567   0.552966086013995   0.026491989592190   0.599480961959216  -1.000000000000000
```

```
Rh =

  -2.236067977499789  -1.118033988749895  -0.838525491562421  -0.698771242968684  -0.618412550027286
   0.000000000000000   0.790569415042095   0.790569415042095   0.760923061978016   0.731276708913938
   0.000000000000000   0.000000000000000   0.233853586673371   0.350780380010057   0.415507712035722
   0.000000000000000   0.000000000000000                   0   0.059292706128157   0.118585412256314
   0.000000000000000   0.000000000000000                   0   0.000000000000000   0.011205268212510


Qh =

  -0.447213595499958  -0.632455532033676   0.534522483824848  -0.316227766016838   0.119522860933440
  -0.447213595499958  -0.316227766016838  -0.267261241912424   0.632455532033675  -0.478091443733759
  -0.447213595499958                   0  -0.534522483824849   0.000000000000002   0.717137165600636
  -0.447213595499958   0.316227766016838  -0.267261241912425  -0.632455532033676  -0.478091443733756
  -0.447213595499958   0.632455532033676   0.534522483824849   0.316227766016838   0.119522860933439

Matlab QR: || A - Q*R || = 9.61e-16, ||Q*Q-I|| = 6.26e-16
House QR : || A - Q*R || = 1.68e-15, ||Q*Q-I|| = 9.52e-16
```

(b) (NLA 10.3) Let Z be the matrix

$$Z = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 7 \\ 4 & 2 & 3 \\ 4 & 2 & 2 \end{bmatrix}.$$

Compute three reduced QR factorizations of Z in Matlab: by the Gram-Schmidt routine `mgs`, by the householder routines `house` and `formQ` and by Matlab's builting command `qr`. Compare the three results and comment on the differences.

Solution:

Apart from signs, all methods agree for this problem as well.

```
Z =

     1     2     3
     4     5     6
     7     8     7
     4     2     3
     4     2     2

 --- QR: Modified Gram-Schmidt ---

Qm =

   0.101015254455221   0.316173069524858   0.541996899879458
   0.404061017820884   0.353369901233664   0.516187523694722
   0.707106781186548   0.390566732942470  -0.524790649089634
   0.404061017820884  -0.557952475632103   0.387140642771041
   0.404061017820884  -0.557952475632103  -0.120443755528768
```

```
Rm =

   9.899494936611664   9.495433918790784   9.697464427701226
                   0   3.291919606229404   3.012943368413352
                   0                   0   1.970115715434855

 --- QR: Householder ---

Qh =

  -0.101015254455221  -0.316173069524859   0.541996899879458  -0.684208462935388  -0.357671145388090
  -0.404061017820884  -0.353369901233665   0.516187523694722   0.328008406610176   0.581227435996121
  -0.707106781186548  -0.390566732942472  -0.524790649089634   0.009397216571679  -0.268261242201384
  -0.404061017820884   0.557952475632102   0.387140642771042   0.365597272896889  -0.491817532809417
  -0.404061017820884   0.557952475632102  -0.120443755528768  -0.538998692773657   0.469465057012740


Rh =

  -9.899494936611667  -9.495433918790781  -9.697464427701224
  -0.000000000000001  -3.291919606229405  -3.012943368413354
  -0.000000000000003  -0.000000000000000   1.970115715434856

 --- QR: Matlab ---

Q =

  -0.101015254455221  -0.316173069524858   0.541996899879458
  -0.404061017820884  -0.353369901233665   0.516187523694722
  -0.707106781186548  -0.390566732942472  -0.524790649089634
  -0.404061017820884   0.557952475632102   0.387140642771041
  -0.404061017820884   0.557952475632102  -0.120443755528768


R =

  -9.899494936611665  -9.495433918790779  -9.697464427701222
                   0  -3.291919606229403  -3.012943368413352
                   0                   0   1.970115715434855
```

Listing 1: house.m

```
function [W,R] = house( A )
%
%   Compute an implicit representation of a reduced
%        QR factorization : A = Q R
%
%   A (input) : m x n matrix
%   W (output) : m x n lower triangular matrix with columns the Housholder vectors v_k
%   R (output) : n x n matrix, upper triangular
%

[m,n]=size(A);
R=A;
W=zeros(m,n);

for k=1:n
```

```
  vk = R(k:m,k); % column k of A
  % Householder vector vk = x + sign(x1) ||x|| e_k :

  % Note: sign(0)=0 may fail, A=[0,1;1,0] (2021)
  % vk(1) = vk(1) + sign(vk(1))*norm(vk);
  if( vk(1)>0 ) s = 1; else s=-1; end
  vk(1) = vk(1) + s*norm(vk);

  vk = vk/norm(vk,2);

  R(k:m,k:n) = R(k:m,k:n) - 2*vk*(vk'*R(k:m,k:n));

  W(k:m,k)=vk;  % save vk in lower triangular part of W

end

R = R(1:n,1:n); % remove zero rows at bottom
```

Listing 2: formQ.m

```
function [Q] = formQ( W )
%
%   Compute the unitary matrix Q in the
%        QR factorization : A = Q R
%   given the output W from the function house.
%
%   W (input) : m x n lower triangular matrix with columns the Housholder vectors v_k
%   Q (output) : mxm unitary matrix
%

[m,n]=size(W);

Q=eye(m);

% Q = Q1 * Q2 * ... * Qn
% Qk = I - 2 vk vk^*

if( 1==1 )

  % Version 1: matrix version
  for k=n:-1:1
    Q(k:m,:) = Q(k:m,:) - 2*W(k:m,k)*(W(k:m,k)'*Q(k:m,:)); % this could be optimized
  end

else

  % Version 2: vector version:
  for i=1:m
    % Compute column qi = Q*e_i
    qi=Q(:,i);
    for k=n:-1:1
      qi(k:m) = qi(k:m) - 2*W(k:m,k)*dot(W(k:m,k),qi(k:m)); % this could be optimized
    end
    Q(:,i)=qi;
```

```
  end

end
```

Listing 3: ps4.m

```
% Problem set 4

clear; % clear variables

format long; % show more digts on the output

% --- Construct the Vandermonde matrix:
m=5;
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;

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

if 0==1
  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
end;
A
Q
R

[W,Rh]=house(A)

Qh = formQ(W)

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

% ------- part (b)
Z = [ 1 , 2 , 3;
      4 , 5 , 6;
      7 , 8 , 7;
      4 , 2 , 3;
      4 , 2 , 2];
Z

[Qm,Rm] = mgs(Z); % MGS
```

```
fprintf(' --- QR: Modified Gram-Schmidt ---\n');
Qm
Rm

fprintf(' --- QR: Householder ---\n');
[W,Rh]=house(Z);
Qh = formQ(W);

Qh
Rh

fprintf(' --- QR: Matlab ---\n');
[Q,R] = qr(Z,0); % QR
Q
R
```

4. Take $m = 50$, $n = 12$. Create a vector $t \in \mathbb{R}^m$ of equally spaced points for $t \in [0, 1]$. Create the $m \times n$ Vandermonde matrix using the points t to build the matrix associated with solving a least squares fit of a polynomial of degree $n - 1$ to the function $\cos(4t)$ so that the right-hand-side is $b = \cos(4t)$.

Solve the least squares problem in 7 ways:

(a) by forming and solving the normal equations.

(b) using a QR factorization and Classical Gram-Schmidt, `clgs`.

(c) using a QR factorization and modified Gram-Schmidt, `mgs`.

(d) using a QR factorization and the Householder triangularization function `house` (from ex. 3.).

(e) using a QR factorization and Matlab's `qr`.

(f) using Matlab's backslash function: $x = A\backslash b$.

(g) using the SVD, using Matlab's `svd` function.

For each of the cases, output the solution to 16 digits formatted in 7 columns (4 columns followed by 3 columns) using the following Matlab code:

```
fprintf('       Normal                CLGS                 MGS                    HOUSE\n');
for i=1:n
  fprintf(' %22.15e %22.15e %22.15e %22.15e\n',xa(i),xb(i),xc(i),xd(i))
end;

fprintf('       Matlab QR         Matlab backslash          SVD\n');
for i=1:n
  fprintf(' %22.15e %22.15e %22.15e\n',xe(i),xf(i),xg(i))
end;
```

The results are assumed to be stored in the arrays `xa, xb, xc, xd, xe, xf, xg`. In each case highlight the digits (e.g. underline in red, or with a hilighter marker) that appear to be wrong (due to rounding errors). Comment on the differences you observe. Do the normal equations exhibit numerical instability (large errors due to roundoff) ? You do not have to explain the cause of the differences.

Solution: Here are the results. **Note: your exact answers may differ due to different versions of Matlab or compilers but the general trends should be the same.** Likely incorrect digits are highlighted in red. The code is below. The normal equations are seen to exhibit numerical instability since most of the values are not even accurate to 1 digit. CLGS and MGS are also bad. The other appropaches give reasonably good results, accurate to 6 digits or more in general.

```
>> ls
         Normal                   CLGS                     MGS                     HOUSE
  9.999999923625902e-01  1.000011225050978e+00  9.999999992898986e-01  1.000000000996588e+00
  2.048962895472356e-06 -1.662792514475208e-03  4.573930469634016e-08 -4.227417356325825e-07
 -8.000072992812600e+00 -7.959931253004935e+00 -7.999998299390983e+00 -7.999981235715334e+00
  1.021632584256770e-03 -3.777042686183902e-01 -7.318347646202624e-05 -3.187628950008534e-04
  1.065924069272982e+01  1.245041785531804e+01  1.066758694363621e+01  1.066943079371808e+01
  3.189690756730509e-02 -4.648781695747640e+00 -5.635787158208388e-03 -1.382027931765499e-02
 -5.776045841468008e+00  1.089871050413284e+00 -5.669951359926857e+00 -5.647075649531697e+00
  1.598438340903675e-01 -5.030264007450659e+00 -3.394343256606498e-02 -7.531598710690635e-02
  1.416819166839725e+00  2.632300313280774e+00  1.645260478981480e+00  1.693606922732483e+00
  2.090849375154007e-01  8.395390912699202e-01  4.127191487987137e-02  6.032136822147395e-03
 -4.586683941588203e-01 -7.023456868617821e-01 -3.888095922916242e-01 -3.742417144852864e-01
  1.032344012031382e-01  5.489358237401135e-02  9.064865347155422e-02  8.804057796255686e-02
         Matlab QR             Matlab backslash              SVD
  1.000000000996607e+00  1.000000000996607e+00  1.000000000996607e+00
 -4.227429284014940e-07 -4.227430618397271e-07 -4.227429165929553e-07
 -7.999981235688562e+00 -7.999981235684451e+00 -7.999981235689030e+00
 -3.187631964053238e-04 -3.187632478590228e-04 -3.187631892405303e-04
  1.066943079564523e+01  1.066943079599354e+01  1.066943079558823e+01
 -1.382028686934504e-02 -1.382028829810566e-02 -1.382028660355476e-02
 -5.647075630539033e+00 -5.647075626787011e+00 -5.647075631312937e+00
 -7.531601841262300e-02 -7.531602486139055e-02 -7.531601696351942e-02
  1.693606956409765e+00  1.693606963629702e+00  1.693606954663832e+00
  6.032114028430476e-03  6.032108955685817e-03  6.032115336941257e-03
 -3.742417056679837e-01 -3.742417036369793e-01 -3.742417062231264e-01
  8.804057647508594e-02  8.804057612156030e-02  8.804057657694747e-02
```

Here are results from Maple using 50 digits of precision:

```
x[1]= 1.0000000009966063881 9e+00
x[2]=-4.22743094981515752694e-07
x[3]=-7.99998123568334552546e+00
x[4]=-3.18763262573856304336e-04
x[5]= 1.06694307961016257485e+01
x[6]=-1.38202887804887216493e-02
x[7]=-5.64707562541768382926e+00
x[8]=-7.53160273819227483226e-02
x[9]= 1.69360696662345950911e+00
x[10]= 6.03210674388467005884e-03
x[11]=-3.74241702713363729283e-01
x[12]= 8.80405759551344285895e-02
```

Listing 4: ls.m

```
% Problem set 4

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

% --- Construct the Vandermonde matrix:
m=50;
```

```
n=12;

t=(0:m-1)'/(m-1); % grid for [0,1]
A=t.^0;
for k=1:n-1
  A = [A t.^k]; % Vandermonde matrix
end;

b =cos(4*t);

% Normal equations:
B = A'*A;
ba=A'*b;
xa = B\ba;

% QR and classical GS
[Qc,Rc]=clgs(A); % reduced QR
bc=Qc'*b;
xb = Rc\bc;

% QR and modified GS
[Qm,Rm]=mgs(A);  % reduced QR
bm=Qm'*b;
xc = Rm\bm;

% QR and Householder:
[W,Rh] = house(A); % reduced Rh
Qh = formQ(W);
Qh = Qh(:,1:n); % reduced Qh
bd=Qh'*b;
xd = Rh\bd;

% Matlab QR
[Q,R]=qr(A,0); % reduced QR
be=Q'*b;
xe = R\be;

% Matlab \
xf = A\b;

% Matlab SVD
[U,S,V] = svd(A,0); %reduced SVD
bg = U'*b;
w = S\bg;
xg = V*w;

fprintf('         Normal                 CLGS                  MGS                    HOUSE\n');
for i=1:n
  fprintf(' %22.15e %22.15e %22.15e %22.15e\n',xa(i),xb(i),xc(i),xd(i))
end;

fprintf('         Matlab QR           Matlab backslash             SVD\n');
for i=1:n
```

```
  fprintf('␣%22.15e␣%22.15e␣%22.15e\n',xe(i),xf(i),xg(i))
end;
```