HW 11

Ryan Richardson
CS 510

2. $v_1 = u_1$; $q_1 = \left\| \frac{u_1}{\|u_1\|} \right\| = \frac{1}{\|u_1\|} \cdot \|u_1\| = 1$

(a) $v_2 = u_2 - \text{proj}_{q_1} u_2 = u_2 - \frac{u_2 \cdot v_1}{\|v_1 \cdot v_1\|} \cdot v_1$ $\quad q_2 = \frac{u_2}{\|u_2\|}$

$v_3 = u_3 - \text{proj}_{q_2} u_3 = u_3 - \frac{u_3 \cdot v_1}{\|v_1 \cdot v_1\|} v_1 - \frac{u_3 \cdot u_2}{\|v_2 \cdot v_2\|} v_2$ $\quad q_3 = \frac{u_3}{\|u_3\|}$

$\langle q_1, q_2 \rangle = \frac{1}{\|u_2\|} \left(\underbrace{q_1 \cdot u_2}_{\emptyset} - \underbrace{\langle u_2, q_1 \rangle}_{\emptyset} q_1 \right) = 0$ $\quad q_1 = 1$

$\frac{1}{\|u_2\|} \left(u_2 - u_2 \right) = 0$

$\langle q_1, q_3 \rangle = \frac{1}{\|u_3\|} \left(\underbrace{q_1 \cdot u_3}_{\emptyset} - \underbrace{\langle u_3, q_1 \rangle}_{\emptyset} q_1 - \underbrace{\langle u_3, q_2 \rangle}_{\emptyset} q_2 \right) = 0$

$\frac{u_1}{\|u_1\|} \cdot u_3 = 0, \quad \frac{u_1}{\|u_1\|} = 0, \quad \frac{u_2}{\|u_2\|}$

(b) By proper all u's are orthogonal, and we can replace each q with its equivalent $u_i / \|u_i\|$ making each term in the equation $\langle q_i, q_j \rangle = \emptyset$.

3. $A = \begin{bmatrix} 1 & -5 & 4 \\ 1 & -5 & 2 \\ 1 & 7 & 0 \\ 1 & 7 & 6 \end{bmatrix}$ $\quad Q = \begin{bmatrix} .5 & -.5 & .5 \\ .5 & .5 & -.5 \\ .5 & .5 & .5 \\ .5 & -.5 & .5 \end{bmatrix}$ $\quad R = \begin{bmatrix} 2 & 12 & 2 \\ 0 & -2 & -4 \\ 0 & 0 & 6 \end{bmatrix}$

$Q^T Q = I$ [illegible] for Q is [illegible]

[illegible] $QQ^T = I$ [illegible] the size [illegible]

[illegible] square.

4. $A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1.0353 & \\ 0 & 0.7 & \\ -1 & 1.403 & \\ -1 & 0.1 & \\ 0 & -1.1 & \end{bmatrix}$ $v_1 = \begin{bmatrix} 2.7321 \\ 0 \\ 0 \\ -1 \\ -1 \\ 0 \end{bmatrix}$ h_1 [illegible]

$$H_1 = \begin{bmatrix} .5774 & 0 & 0 & -.5774 & -.5774 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ -.5774 & 0 & 0 & .2113 & -.7887 & 0 \\ -.5774 & 0 & 0 & -.7887 & .2113 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_1 \cdot A = \begin{bmatrix} 1.7321 & -.5774 & -.5774 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & .2113 & -.7887 \\ 0 & -.7887 & .2113 \\ 0 & 1 & 1 \end{bmatrix}$$

4. $\alpha = -\text{sign}(a_{22}) \cdot \text{norm}(a_2) = -1 \cdot \sqrt{1^2 + (.7808)^2 + (.2113)^2 + 1^2}$

$\alpha = -1.6133$

$$a_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ .7808 \\ -.2113 \\ -1 \end{bmatrix} \quad e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad v_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ .7808 \\ -.2113 \\ -1 \end{bmatrix} + \begin{bmatrix} 0 \\ -1.6133 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

unit $\{v_2\}$ ⇒

5. $A = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 0 & 3 \\ 2 & 3 & 0 \end{bmatrix}$

Editor - C:\Users\Ryan\Documents\MATLAB\normPower.m

normPower.m

```
function [lam, vec] = normPower(A,err,maxIter)
%  x = normPower(A,err,iter): Normalized Power Iteration of a Matrix to find
%  largest EigenPair
%  input:
%      A = Input matrix
%      err = Maximum error
%      iter = max iterations
% output:
%    lam = largest eigen value
%    vec = corresponding eigen vector

if nargin < 1, error('No matrix supplied'), end
if nargin < 4 || isempty(maxIter),maxIter=50;end
if nargin < 3 || isempty(err), err=.00001;end
[i,k] = size(A);
if i~=k, error('Matrix must be square');end

aLen = length(A);
vec = ones(aLen,1);
lamOld = 0;
currIter = 0;

while(currIter < maxIter)
    currIter = currIter + 1;

    compVec = A*vec;
    lam = max(compVec)/max(vec);
    vec = compVec/norm(compVec);

    currErr = abs((lam-lamOld)/lam)*100;
    lamOld = lam;
    if currErr <= err,break, end
end
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> A = [2,1,-1,3;1,7,0,-1;-1,0,4,-2;3,-1,-2,1]

A =

     2     1    -1     3
     1     7     0    -1
    -1     0     4    -2
     3    -1    -2     1

>> [eigen, vector] = normPower(A)

eigen =

    7.2332


vector =

    0.1185
    0.9862
    0.0322
   -0.1115
```

1.

```
Q =

   -0.5000   -0.5000    0.5000
   -0.5000   -0.5000   -0.5000
    0.5000   -0.5000   -0.5000
    0.5000   -0.5000    0.5000


R =

    2.0000   12.0000    2.0000
         0   -2.0000   -4.0000
         0         0    6.0000

>> Q'*Q

ans =

    1.0000   -0.0000    0.0000
   -0.0000    1.0000         0
    0.0000         0    1.0000
```

3. 6. 1

```
a2 =

         0
    1.0000
         0
    0.7887
   -0.2113
   -1.0000

>> alpha = -sign(a2(2)) * norm(a2)

alpha =

   -1.6330

>> v2 = a2 - alpha*e2

v2 =

         0
    2.6330
         0
    0.7887
   -0.2113
   -1.0000

>> h2 = eye(6)-2*(v2*v2')/(v2'*v2)

h2 =

    1.0000         0         0         0         0         0
         0   -0.6124         0   -0.4830    0.1294    0.6124
         0         0    1.0000         0         0         0
         0   -0.4830         0    0.8553    0.0388    0.1834
         0    0.1294         0    0.0388    0.9896   -0.0491
         0    0.6124         0    0.1834   -0.0491    0.7674

>> h2*h1*A

ans =

   -1.7321    0.5773    0.5773
    0.0000   -1.6330    0.8165
         0         0    1.0000
   -0.0000   -0.0000    0.0333
   -0.0000   -0.0000    0.7232
   -0.0000   -0.0000    0.6899
```

4.

Editor - C:\Users\Ryan\Documents\MATLAB\qrDecomp.m

qrDecomp.m

```
function [v,currErr,currIter] = qrDecomp(A, err, maxIter)
%  v = normPower(A,err,iter): QR iteration to find all eigenvalues of A
%  input:
%      A = Input matrix
%      err = Maximum error
%      iter = max iterations
% output:
%   v = vector of the largest eigenvalues of A

if nargin < 1, error('No matrix supplied'), end
if nargin < 3 || isempty(maxIter),maxIter=100;end
if nargin < 2 || isempty(err), err=.1e-8;end

[i,k] = size(A);

ident = eye(i,k); %create identity matrix same size of A

currIter = 0;

while(currIter < maxIter)
    currIter = currIter + 1;
    [Q,R] = qr(A);
    A = R*Q;
    v=diag(A);
    currErr = max(A(~ident)); %get larger value of A not on main diagonal
    if currErr <= err,break, end
end
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> [eigenValues error iterations] = qrDecomp(A)

eigenValues =

    5.0000
   -3.0000
   -1.0000


error =

   2.7805e-10


iterations =
```

5.