MATLAB PROJECT 2

Please include this page in your Group file, as a front page. Type in the group number and the names of all members WHO PARTICIPATED in this project.

GROUP # _____26______

FIRST & LAST NAMES (UFID numbers are NOT required):

_1. James Luberisse
_2. Hao Lin

_3. Ryan Houston

_4. _William Liu

_5. Seadn Madsen

_6. Anthony Khmarin

By including your names above, each of you had confirmed that you did the work and agree with the work submitted.

Exercise 1

```
type eigenval.m
```

```
function [q,e,c,S]=eigenval(A)
format
%method 1
[m,n]=size(A);

[Q,R]=qr(A);


if (m == n)
    k = 1;
    P = closetozeroroundoff(A - triu(A), 12);
    none = zeros(m, n);
    while (P ~= none)
        k = k + 1;
        [Q, R] = qr(A);
        A = R * Q;
        P = closetozeroroundoff(A - triu(A), 12);
    end

B = A;

fprintf('the number of iterations is %i\n',k)

q=sort(diag(B));
disp('The eigenvalues of A from QR factorization are:')
q=closetozeroroundoff(q,7)
end

%method 2

e=eig(A);
e=sort(e,'ascend','ComparisonMethod','real');
disp('The eigenvalues of A from a MATLAB function are:')
e=closetozeroroundoff(e,7)

%method 3

S=poly(A);
S=closetozeroroundoff(S,7);

disp('the MATLAB characteristic polynomial of A is:')
Q=poly2sym(S)
c=roots(S);
c=sort(c,'ascend','ComparisonMethod','real');
disp('The eigenvalues from the MATLAB characteristic polynomial are:')
c=closetozeroroundoff(c,7)


%comparisons

eq=norm(e-q)
cq=norm(c-q)
ec=norm(e-c)

disp('the constructed characteristic polynomial of A is:')
R=poly2sym(poly(e))

if(Q == R)
    disp('Yes! Q and R are the same!')
else
```

```
    disp('What?! Q and R do not match?')
end

end
```

```
type jord.m
```

```
function J = jord(n,r)
if mod(n,1)==0 && n>1
    oneVector=r*ones(n,1);
    J=diag(oneVector);
    for i=[1:n-1]
    J(i,i+1)=1;
    end
fprintf('Jordan matrix of the size %i by %i is\n',n,n)
J
else
J=[];
fprintf('n=%i is not valid input and Jordan Block cannot be built\n',n)
end
end
```

```
type quer.m
```

```
function [Q,R] = quer(A)
format
[m,n]=size(A);

[Q,R]=qr(A)
if (closetozeroroundoff(A-Q*R,7)==0)
    disp('the product of Q and R forms a decomposition of A');
else
    disp('no, it cannot be true!');
end

q=0;
r=0;
if(inv(Q) == transpose(Q))
    q=1;
    disp('Q is a unitary matrix');
    %check that Q is unitary
end
if (istriu(R))
    r=1;
    disp('R is an upper-triangular matrix');
    %check R is an upper-triangular matrix
end
if(q==1 && r==1)
    disp('Q*R forms an orthogonal-triangular decomposition of A');
else
    disp('Q*R does NOT forms an orthogonal-triangular decomposition of A. What is wrong?!');
end

if(m == n)
    k=1;
    P=closetozeroroundoff(A-triu(A),7);
while(P~=zeros(m,n))
    A=R*Q;
    [Q,R]=qr(A);
    P=closetozeroroundoff(A-triu(A),7);
    k=k+1;
end
disp('the matrix B');
disp(A);
```

```
disp('the number of iterations:');
disp(k);

disp('the main diagonal of the matrix B');
for i=1:m
    disp(A(i,i));
end

E = eig(A);
disp('the eigenvalues of the input matrix A')
disp(E)
end
end
```

```
type closetozeroroundoff.m
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
%(a)
A=[3 3;0 3]
```

```
A = 2×2
     3     3
     0     3
```

```
[q,e,c,S]=eigenval(A);
```

```
the number of iterations is 1
The eigenvalues of A from QR factorization are:
q = 2×1
     3
     3
The eigenvalues of A from a MATLAB function are:
e = 2×1
     3
     3
the MATLAB characteristic polynomial of A is:
```

Q = $x^2 - 6\,x + 9$

```
The eigenvalues from the MATLAB characteristic polynomial are:
c = 2×1 complex
   3.0000 - 0.0000i
   3.0000 + 0.0000i
eq = 0
cq = 5.2684e-08
ec = 5.2684e-08
the constructed characteristic polynomial of A is:
```

R = $x^2 - 6\,x + 9$

```
Yes! Q and R are the same!
```

```
% QR provides a good approximation of the eigenvalues.

%(b)
A=jord(5,4)
```

```
Jordan matrix of the size 5 by 5 is
J = 5×5
     4     1     0     0     0
```

```
     0     4     1     0     0
     0     0     4     1     0
     0     0     0     4     1
     0     0     0     0     4
A = 5×5
     4     1     0     0     0
     0     4     1     0     0
     0     0     4     1     0
     0     0     0     4     1
     0     0     0     0     4
```

```
[q,e,c,S]=eigenval(A);
```

```
the number of iterations is 1
The eigenvalues of A from QR factorization are:
q = 5×1
     4
     4
     4
     4
     4
The eigenvalues of A from a MATLAB function are:
e = 5×1
     4
     4
     4
     4
     4
the MATLAB characteristic polynomial of A is:
```

Q = $x^5 - 20\,x^4 + 160\,x^3 - 640\,x^2 + 1280\,x - 1024$

```
The eigenvalues from the MATLAB characteristic polynomial are:
c = 5×1 complex
   3.9955 + 0.0000i
   3.9986 - 0.0043i
   3.9986 + 0.0043i
   4.0037 - 0.0027i
   4.0037 + 0.0027i
eq = 0
cq = 0.0101
ec = 0.0101
the constructed characteristic polynomial of A is:
```

R = $x^5 - 20\,x^4 + 160\,x^3 - 640\,x^2 + 1280\,x - 1024$

```
Yes! Q and R are the same!
```

```
% QR provides a good approximation of the eigenvalues.

%(c)
A=ones(5);
[q,e,c,S]=eigenval(A);
```

```
the number of iterations is 1
The eigenvalues of A from QR factorization are:
q = 5×1
     1
     1
     1
     1
     1
The eigenvalues of A from a MATLAB function are:
e = 5×1
     0
     0
```

```
     0
     0
     5
the MATLAB characteristic polynomial of A is:
```

Q = $x^5 - 5\,x^4$

```
The eigenvalues from the MATLAB characteristic polynomial are:
c = 5×1
     0
     0
     0
     0
     5
eq = 4.4721
cq = 4.4721
ec = 0
the constructed characteristic polynomial of A is:
```

R = $x^5 - 5\,x^4$

```
Yes! Q and R are the same!
```

```
% QR  provied a bad approiximaton of the eiegenvalues.

%(d)
A=tril(magic(4))
```

```
A = 4×4
    16     0     0     0
     5    11     0     0
     9     7     6     0
     4    14    15     1
```

```
[q,e,c,S]=eigenval(A);
```

```
the number of iterations is 1
The eigenvalues of A from QR factorization are:
q = 4×1
     1
     6
    11
    16
The eigenvalues of A from a MATLAB function are:
e = 4×1
     1
     6
    11
    16
the MATLAB characteristic polynomial of A is:
```

Q = $x^4 - 34\,x^3 + 371\,x^2 - 1394\,x + 1056$

```
The eigenvalues from the MATLAB characteristic polynomial are:
c = 4×1
    1.0000
    6.0000
   11.0000
   16.0000
eq = 0
cq = 3.5706e-14
ec = 3.5706e-14
the constructed characteristic polynomial of A is:
```

R = $x^4 - 34\,x^3 + 371\,x^2 - 1394\,x + 1056$

```
Yes! Q and R are the same!
```

```
%QR provides a good approximation of the eigenvalues.
```

```
%(e)
A=triu(magic(4),-1)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     0     7     6    12
     0     0    15     1
```

```
[q,e,c,S]=eigenval(A);
```

```
the number of iterations is 1
The eigenvalues of A from QR factorization are:
q = 4×1
     1
     6
    11
    16
The eigenvalues of A from a MATLAB function are:
e = 4×1
  -10.4464
    7.4036
   12.0417
   25.0011
the MATLAB characteristic polynomial of A is:
```

Q = $x^4 - 34\,x^3 + 111\,x^2 + 3781\,x - 23284$

```
The eigenvalues from the MATLAB characteristic polynomial are:
c = 4×1
  -10.4464
    7.4036
   12.0417
   25.0011
eq = 14.6662
cq = 14.6662
ec = 7.2944e-14
the constructed characteristic polynomial of A is:
```

R = $x^4 - 34\,x^3 + 111\,x^2 + 3781\,x - 23284$

```
Yes! Q and R are the same!
```

```
% QR provides a bad approximation of the eigenvalues.

%(f)
A=[4 3 0 0; 7 10 3 0;0 3 1 5;0 0 6 7]
```

```
A = 4×4
     4     3     0     0
     7    10     3     0
     0     3     1     5
     0     0     6     7
```

```
[q,e,c,S]=eigenval(A);
```

```
the number of iterations is 1
The eigenvalues of A from QR factorization are:
q = 4×1
     1
     4
     7
    10
The eigenvalues of A from a MATLAB function are:
```

```
e = 4×1
   -2.9302
    1.8042
    9.7253
   13.4007
the MATLAB characteristic polynomial of A is:
```

Q = $x^4 - 22\,x^3 + 99\,x^2 + 269\,x - 689$

```
The eigenvalues from the MATLAB characteristic polynomial are:
c = 4×1
   -2.9302
    1.8042
    9.7253
   13.4007
eq = 6.2657
cq = 6.2657
ec = 2.0897e-14
the constructed characteristic polynomial of A is:
```

R = $x^4 - 22\,x^3 + 99\,x^2 + 269\,x - 689$

```
Yes! Q and R are the same!
```

```
% QR provides a bad approximation of the eigenvalues.
```

Exercise 2

```
type closetozeroroundoff.m
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
type eigen.m
```

```
 function [L,P,D]=eigen(A)
format


[~,n]=size(A);
P=[];
D=[];


if(rank(A) == n)
    L=eig(A);
    L = transpose(L);
    L=real(L);
    L = sort(L);



    L = closetozeroroundoff(L, 7);



    %(1)
   refNumbers = [];
    prevNum = L(1,1);
    refNumbers = [refNumbers prevNum];
    for i = 1:n
        if(round(L(1,i),7) ~= round(prevNum,7))
               prevNum = L(1,i);
```

```
                refNumbers = [refNumbers prevNum];

        end
    end


    for i = 1:n
      for j = 1:size(refNumbers,2)
          if( closetozeroroundoff(L(1,i) - refNumbers(1,j),7) == 0 )
              L(1,i) = refNumbers(1,j);
          end
      end
    end

    fprintf('all eigenvalues of A are\n')
    display(L)
end

%(2)
if(rank(A) ~= n)
    L = eig(A);
    L = closetozeroroundoff(L, 7);
    L = real(L');
    L = sort(L);



    refNumbers = [];
    prevNum = L(1,1);
    refNumbers = [refNumbers prevNum];
    for i = 1:n
        if(round(L(1,i),7) ~= round(prevNum,7))
                prevNum = L(1,i);
                refNumbers = [refNumbers prevNum];

        end
    end


    for i = 1:n
      for j = 1:size(refNumbers,2)
          if( closetozeroroundoff(L(1,i),7) == closetozeroroundoff(L(1,j),7) )
              L(1,i) = refNumbers(1,j);
          end
      end
    end

    fprintf('all eigenvalues of A are\n')
    display(L)
end




%may need to reformat

M = unique(L);
m = [];




count = 0;
k=1;
for i = 1:size(M, 2)
```

```
    for j = 1:size(L, 2)
        if( round(M(1,i),7) == round(L(1,j),7) )
            count =  count + 1;
        end
    end
        m(1,k) = count;
        k = k + 1;
        count = 0;
end




for i = 1:size(M, 2)


             fprintf('eigenvalue %d has multiplicity %i\n',M(i),m(i))


  end

  d = [];
  k = 1;
  for i = 1:size(M,2)
            tempA = A;

      for j = 1:size(A,2)

      tempA(j,j) = tempA(j,j) - M(1,i);


      end
%       z = zeros(size(A,2),1)

      fprintf('a basis for the eigenspace for lambda=%d is:\n',M(i));
      W=null(tempA)
      P = [P W];
      d(1,k) = size(W, 2);
      k = k + 1;

  end
for i = 1:size(M,2)
  fprintf('dimension of eigenspace for lambda = %d is %i\n',M(i),d(i))
end




diagonlizable = true;
for i = 1:size(M, 2)
 if (~( m(i) == d(i)))
         diagonlizable = false;
 end
end



if(diagonlizable == true)
    fprintf("A is diagonalizable")
    %construction P
    P
    %construction D

    D = eye(size(L, 2));
    for i = 1: size(D, 2)
```

```
        D(i,i) = L(1,i);
    end

    D

    if(closetozeroroundoff(A*P - P*D, 7) == 0 & rank(P) == n)
         fprintf('Great! I got a diagonalization!')
    else
        fprintf('Oops! I got a bug in my code!')
    end

    [U, V] = eig(A)

    pSum = 0;
    uSum = 0;
    matches = 0;
    for i = 1:n
        pSum = pSum + abs(P(:,i));
        uSum = uSum + abs(U(:,i));
    end

    uSum = sort(uSum);
    pSum = sort(pSum);


    if(closetozeroroundoff(pSum - uSum,7) == 0)
        fprintf('Columns of P and U are the same or match up to scalar (-1).')
    else
        fprintf('There is no specific match among the columns of P and U' )
    end

    sumD = 0;
    sumV = 0;

    for i = 1:n
        sumD = sumD + abs(D(i,i));
        sumV = sumV + abs(V(i,i));
    end

    if(closetozeroroundoff(sumD - sumV,7) == 0)
        fprintf('The diagonal elements of D and V match')
    else
        fprintf('That cannot be true!')
    end


    %BONUS
    disp(' Bonus*')
    %symmetric check
    if(size(P,1) == size(P,2))
        disp('matrix A is symmetric')
           if(closetozeroroundoff((P*D*inv(P)) - A,7) == 0)
               disp('the orthogonal diagonalization is confirmed')


       else
           disp('Wow! A symmetric matrix is not orthogonally diagonalizable?!')
       end

    else
         disp('diagonalizable matrix A is not orthogonally diagonalizable')
    end

else
```

```
    fprintf("A is not diagonalizable")
end

end
```

```
type jord.m
```

```
function J = jord(n,r)
if mod(n,1)==0 && n>1
    oneVector=r*ones(n,1);
    J=diag(oneVector);
    for i=[1:n-1]
    J(i,i+1)=1;
    end
fprintf('Jordan matrix of the size %i by %i is\n',n,n)
J
else
J=[];
fprintf('n=%i is not valid input and Jordan Block cannot be built\n',n)
end
end
```

```
%(a)
A=[3 3; 0 3]
```

```
A = 2×2
     3     3
     0     3
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×2
     3     3
eigenvalue 3 has multiplicity 2
a basis for the eigenspace for lambda=3 is:
W = 2×1
    -1
     0
dimension of eigenspace for lambda = 3 is 1
A is not diagonalizable
```

```
%(b)
A=[2 4 3;-4 -6 -3;3 3 1]
```

```
A = 3×3
     2     4     3
    -4    -6    -3
     3     3     1
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×3
   -2.0000   -2.0000    1.0000
eigenvalue -2.000000e+00 has multiplicity 2
eigenvalue 1.000000e+00 has multiplicity 1
a basis for the eigenspace for lambda=-2.000000e+00 is:
W = 3×1
    0.7071
   -0.7071
   -0.0000
```

```
a basis for the eigenspace for lambda=1.000000e+00 is:
W = 3×1
    0.5774
   -0.5774
    0.5774
dimension of eigenspace for lambda = -2.000000e+00 is 1
dimension of eigenspace for lambda = 1.000000e+00 is 1
A is not diagonalizable
```

```
%(c)
A=[4 0 1 0; 0 4 0 1; 1 0 4 0; 0 1 0 4]
```

```
A = 4×4
     4     0     1     0
     0     4     0     1
     1     0     4     0
     0     1     0     4
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×4
     3     3     5     5
eigenvalue 3 has multiplicity 2
eigenvalue 5 has multiplicity 2
a basis for the eigenspace for lambda=3 is:
W = 4×2
    0.7071         0
         0   -0.7071
   -0.7071         0
         0    0.7071
a basis for the eigenspace for lambda=5 is:
W = 4×2
   -0.7071         0
         0    0.7071
   -0.7071         0
         0    0.7071
dimension of eigenspace for lambda = 3 is 2
dimension of eigenspace for lambda = 5 is 2
A is diagonalizable
P = 4×4
    0.7071         0   -0.7071         0
         0   -0.7071         0    0.7071
   -0.7071         0   -0.7071         0
         0    0.7071         0    0.7071
D = 4×4
     3     0     0     0
     0     3     0     0
     0     0     5     0
     0     0     0     5
Great! I got a diagonalization!
U = 4×4
   -0.7071         0         0   -0.7071
         0    0.7071    0.7071         0
    0.7071         0         0   -0.7071
         0   -0.7071    0.7071         0
V = 4×4
     3     0     0     0
     0     3     0     0
     0     0     5     0
     0     0     0     5
Columns of P and U are the same or match up to scalar (-1).The diagonal elements of D and V match Bonus*
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

```
%(d)
A=jord(5,4);
```

```
Jordan matrix of the size 5 by 5 is
J = 5×5
     4     1     0     0     0
     0     4     1     0     0
     0     0     4     1     0
     0     0     0     4     1
     0     0     0     0     4
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×5
     4     4     4     4     4
eigenvalue 4 has multiplicity 5
a basis for the eigenspace for lambda=4 is:
W = 5×1
     1
     0
     0
     0
     0
dimension of eigenspace for lambda = 4 is 1
A is not diagonalizable
```

```
%(e)
A=ones(4)
```

```
A = 4×4
     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1     1
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×4
    4.0000    4.0000         0         0
eigenvalue 0 has multiplicity 2
eigenvalue 4.000000e+00 has multiplicity 2
a basis for the eigenspace for lambda=0 is:
W = 4×3
         0         0    0.8660
   -0.5774   -0.5774   -0.2887
    0.7887   -0.2113   -0.2887
   -0.2113    0.7887   -0.2887
a basis for the eigenspace for lambda=4.000000e+00 is:
W = 4×1
   -0.5000
   -0.5000
   -0.5000
   -0.5000
dimension of eigenspace for lambda = 0 is 3
dimension of eigenspace for lambda = 4.000000e+00 is 1
A is not diagonalizable
```

```
%(f)
A=[4 1 3 1;1 4 1 3;3 1 4 1;1 3 1 4]
```

```
A = 4×4
     4     1     3     1
     1     4     1     3
     3     1     4     1
     1     3     1     4
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×4
    1.0000    1.0000    5.0000    9.0000
eigenvalue 1.000000e+00 has multiplicity 2
eigenvalue 5.000000e+00 has multiplicity 1
eigenvalue 9.000000e+00 has multiplicity 1
a basis for the eigenspace for lambda=1.000000e+00 is:
W = 4×2
    0.5215    0.4775
    0.4775   -0.5215
   -0.5215   -0.4775
   -0.4775    0.5215
a basis for the eigenspace for lambda=5.000000e+00 is:
W = 4×1
    0.5000
   -0.5000
    0.5000
   -0.5000
a basis for the eigenspace for lambda=9.000000e+00 is:
W = 4×1
    0.5000
    0.5000
    0.5000
    0.5000
dimension of eigenspace for lambda = 1.000000e+00 is 2
dimension of eigenspace for lambda = 5.000000e+00 is 1
dimension of eigenspace for lambda = 9.000000e+00 is 1
A is diagonalizable
P = 4×4
    0.5215    0.4775    0.5000    0.5000
    0.4775   -0.5215   -0.5000    0.5000
   -0.5215   -0.4775    0.5000    0.5000
   -0.4775    0.5215   -0.5000    0.5000
D = 4×4
    1.0000         0         0         0
         0    1.0000         0         0
         0         0    5.0000         0
         0         0         0    9.0000
Great! I got a diagonalization!
U = 4×4
    0.0000    0.7071    0.5000   -0.5000
    0.7071   -0.0000   -0.5000   -0.5000
    0.0000   -0.7071    0.5000   -0.5000
   -0.7071         0   -0.5000   -0.5000
V = 4×4
    1.0000         0         0         0
         0    1.0000         0         0
         0         0    5.0000         0
         0         0         0    9.0000
There is no specific match among the columns of P and UThe diagonal elements of D and V match Bonus*
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

```
%(g)
A=[3 1 1;1 3 1;1 1 3]
```

```
A = 3×3
     3     1     1
     1     3     1
     1     1     3
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×3
    2.0000    2.0000    5.0000
eigenvalue 2.000000e+00 has multiplicity 2
eigenvalue 5.000000e+00 has multiplicity 1
a basis for the eigenspace for lambda=2.000000e+00 is:
W = 3×2
    0.8165         0
   -0.4082   -0.7071
   -0.4082    0.7071
a basis for the eigenspace for lambda=5.000000e+00 is:

W =

  3×0 empty double matrix

dimension of eigenspace for lambda = 2.000000e+00 is 2
dimension of eigenspace for lambda = 5.000000e+00 is 0
A is not diagonalizable
```

```
%(h)
A=magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×4
   -8.9443         0    8.9443   34.0000
eigenvalue -8.944272e+00 has multiplicity 1
eigenvalue 0 has multiplicity 1
eigenvalue 8.944272e+00 has multiplicity 1
eigenvalue 3.400000e+01 has multiplicity 1
a basis for the eigenspace for lambda=-8.944272e+00 is:
W = 4×1
   -0.3764
   -0.0236
   -0.4236
    0.8236
a basis for the eigenspace for lambda=0 is:
W = 4×1
    0.2236
    0.6708
   -0.6708
   -0.2236
a basis for the eigenspace for lambda=8.944272e+00 is:
W = 4×1
    0.8236
   -0.4236
   -0.0236
   -0.3764
```

```
a basis for the eigenspace for lambda=3.400000e+01 is:
W = 4×1
    0.5000
    0.5000
    0.5000
    0.5000
dimension of eigenspace for lambda = -8.944272e+00 is 1
dimension of eigenspace for lambda = 0 is 1
dimension of eigenspace for lambda = 8.944272e+00 is 1
dimension of eigenspace for lambda = 3.400000e+01 is 1
A is diagonalizable
P = 4×4
   -0.3764    0.2236    0.8236    0.5000
   -0.0236    0.6708   -0.4236    0.5000
   -0.4236   -0.6708   -0.0236    0.5000
    0.8236   -0.2236   -0.3764    0.5000
D = 4×4
   -8.9443         0         0         0
         0         0         0         0
         0         0    8.9443         0
         0         0         0   34.0000
Great! I got a diagonalization!
U = 4×4
   -0.5000   -0.8236    0.3764   -0.2236
   -0.5000    0.4236    0.0236   -0.6708
   -0.5000    0.0236    0.4236    0.6708
   -0.5000    0.3764   -0.8236    0.2236
V = 4×4
   34.0000         0         0         0
         0    8.9443         0         0
         0         0   -8.9443         0
         0         0         0    0.0000
Columns of P and U are the same or match up to scalar (-1).The diagonal elements of D and V match Bonus*
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

```
%(k)
 A=magic(5)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

```
 [L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×5
  -21.2768  -13.1263   13.1263   21.2768   65.0000
eigenvalue -2.127677e+01 has multiplicity 1
eigenvalue -1.312628e+01 has multiplicity 1
eigenvalue 1.312628e+01 has multiplicity 1
eigenvalue 2.127677e+01 has multiplicity 1
eigenvalue 6.500000e+01 has multiplicity 1
a basis for the eigenspace for lambda=-2.127677e+01 is:
W = 5×1
   -0.0976
   -0.3525
   -0.5501
    0.3223
    0.6780
a basis for the eigenspace for lambda=-1.312628e+01 is:
```

```
W = 5×1
   -0.6330
    0.5895
   -0.3915
    0.1732
    0.2619
a basis for the eigenspace for lambda=1.312628e+01 is:
W = 5×1
    0.2619
    0.1732
   -0.3915
    0.5895
   -0.6330
a basis for the eigenspace for lambda=2.127677e+01 is:
W = 5×1
    0.6780
    0.3223
   -0.5501
   -0.3525
   -0.0976
a basis for the eigenspace for lambda=6.500000e+01 is:
W = 5×1
   -0.4472
   -0.4472
   -0.4472
   -0.4472
   -0.4472
dimension of eigenspace for lambda = -2.127677e+01 is 1
dimension of eigenspace for lambda = -1.312628e+01 is 1
dimension of eigenspace for lambda = 1.312628e+01 is 1
dimension of eigenspace for lambda = 2.127677e+01 is 1
dimension of eigenspace for lambda = 6.500000e+01 is 1
A is diagonalizable
P = 5×5
   -0.0976   -0.6330    0.2619    0.6780   -0.4472
   -0.3525    0.5895    0.1732    0.3223   -0.4472
   -0.5501   -0.3915   -0.3915   -0.5501   -0.4472
    0.3223    0.1732    0.5895   -0.3525   -0.4472
    0.6780    0.2619   -0.6330   -0.0976   -0.4472
D = 5×5
  -21.2768         0         0         0         0
         0  -13.1263         0         0         0
         0         0   13.1263         0         0
         0         0         0   21.2768         0
         0         0         0         0   65.0000
Great! I got a diagonalization!
U = 5×5
   -0.4472    0.0976   -0.6330    0.6780   -0.2619
   -0.4472    0.3525    0.5895    0.3223   -0.1732
   -0.4472    0.5501   -0.3915   -0.5501    0.3915
   -0.4472   -0.3223    0.1732   -0.3525   -0.5895
   -0.4472   -0.6780    0.2619   -0.0976    0.6330
V = 5×5
   65.0000         0         0         0         0
         0  -21.2768         0         0         0
         0         0  -13.1263         0         0
         0         0         0   21.2768         0
         0         0         0         0   13.1263
Columns of P and U are the same or match up to scalar (-1).The diagonal elements of D and V match Bonus*
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

```
%(l)
A=hilb(5)
```

```
A = 5×5
    1.0000    0.5000    0.3333    0.2500    0.2000
    0.5000    0.3333    0.2500    0.2000    0.1667
    0.3333    0.2500    0.2000    0.1667    0.1429
    0.2500    0.2000    0.1667    0.1429    0.1250
    0.2000    0.1667    0.1429    0.1250    0.1111
```

```
[L,P,D]=eigen(A);
```

```
all eigenvalues of A are
L = 1×5
    0.0000    0.0003    0.0114    0.2085    1.5671
eigenvalue 3.287929e-06 has multiplicity 1
eigenvalue 3.058980e-04 has multiplicity 1
eigenvalue 1.140749e-02 has multiplicity 1
eigenvalue 2.085342e-01 has multiplicity 1
eigenvalue 1.567051e+00 has multiplicity 1
a basis for the eigenspace for lambda=3.287929e-06 is:
W = 5×1
   -0.0062
    0.1167
   -0.5062
    0.7672
   -0.3762
a basis for the eigenspace for lambda=3.058980e-04 is:
W = 5×1
    0.0472
   -0.4327
    0.6674
    0.2330
   -0.5576
a basis for the eigenspace for lambda=1.140749e-02 is:
W = 5×1
   -0.2142
    0.7241
    0.1205
   -0.3096
   -0.5652
a basis for the eigenspace for lambda=2.085342e-01 is:
W = 5×1
   -0.6019
    0.2759
    0.4249
    0.4439
    0.4290
a basis for the eigenspace for lambda=1.567051e+00 is:
W = 5×1
   -0.7679
   -0.4458
   -0.3216
   -0.2534
   -0.2098
dimension of eigenspace for lambda = 3.287929e-06 is 1
dimension of eigenspace for lambda = 3.058980e-04 is 1
dimension of eigenspace for lambda = 1.140749e-02 is 1
dimension of eigenspace for lambda = 2.085342e-01 is 1
dimension of eigenspace for lambda = 1.567051e+00 is 1
A is diagonalizable
P = 5×5
   -0.0062    0.0472   -0.2142   -0.6019   -0.7679
    0.1167   -0.4327    0.7241    0.2759   -0.4458
   -0.5062    0.6674    0.1205    0.4249   -0.3216
    0.7672    0.2330   -0.3096    0.4439   -0.2534
   -0.3762   -0.5576   -0.5652    0.4290   -0.2098
```

```
D = 5×5
    0.0000         0         0         0         0
         0    0.0003         0         0         0
         0         0    0.0114         0         0
         0         0         0    0.2085         0
         0         0         0         0    1.5671
Great! I got a diagonalization!
U = 5×5
   -0.0062    0.0472    0.2142   -0.6019    0.7679
    0.1167   -0.4327   -0.7241    0.2759    0.4458
   -0.5062    0.6674   -0.1205    0.4249    0.3216
    0.7672    0.2330    0.3096    0.4439    0.2534
   -0.3762   -0.5576    0.5652    0.4290    0.2098
V = 5×5
    0.0000         0         0         0         0
         0    0.0003         0         0         0
         0         0    0.0114         0         0
         0         0         0    0.2085         0
         0         0         0         0    1.5671
Columns of P and U are the same or match up to scalar (-1).The diagonal elements of D and V match Bonus*
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

Exercise 3

```
type closetozeroroundoff
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
type shrink
```

```
function B=shrink(A)
[~,pivot]=rref(A);
B=A(:,pivot);
end
```

```
type proj
```

```
function [p,z]=proj(A,b)
format
A=shrink(A);
m=size(A,1);
p=[];
z=[];

if m == size(transpose(b), 2)
    if rank(A) == rank([A b])
        disp('b is in the Col A')
        p = b;
        disp(p)
        z = b - p;
        disp(z)
        return
    else
        R = colspace(sym(A));
        T = 0;
        for i = 1:size(R,2)
```

```
            T = T + dot(R(:,i),b);
        end
        T = closetozeroroundoff(T,7);
        if T == 0
            disp('b is orthogonal to Col A')
            z = b;
            p = z - b;
            disp(p)
            disp(z)
        return
        else
            x = A \ b;
            disp('the least squares solution of inconsistent system Ax=b is')
            disp(x)
            x1 = A \ b;
            c1 = closetozeroroundoff(x,12);
            c2 = closetozeroroundoff(x1,12);
            if c1 == c2
                disp('A\b returns the least-squares solution of inconsistent system Ax=b')
            end
            p = A*x;
            disp('the projection of b onto Col A is')
            disp(p)
            z = b - p;
            if closetozeroroundoff(dot(p,z),7) == 0
                disp('the component of b orthogonal to Col A is')
                disp(z)
            else
                disp('Oops! Is there a bug in my code?')
            end
            d = norm(z);
            fprintf('the distance from b to Col A is %i',d)
        end
    end
else
    disp('No solution: sizes of A and b disagree')
    p=[];
    z=[];
end
end
```

```
%(a)
A=magic(4), b=(1:4)'
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
b = 4×1
     1
     2
     3
     4
```

```
[p,z]=proj(A,b);
```

```
the least squares solution of inconsistent system Ax=b is
    0.0471
    0.1941
```

```
    0.0529

A\b returns the least-squares solution of inconsistent system Ax=b
the projection of b onto Col A is
    1.3000
    2.9000
    2.1000
    3.7000

the component of b orthogonal to Col A is
   -0.3000
   -0.9000
    0.9000
    0.3000

the distance from b to Col A is 1.341641e+00
```

```
%(b)
A=magic(6), b=A(:,6)
```

```
A = 6×6
    35     1     6    26    19    24
     3    32     7    21    23    25
    31     9     2    22    27    20
     8    28    33    17    10    15
    30     5    34    12    14    16
     4    36    29    13    18    11
b = 6×1
    24
    25
    20
    15
    16
    11
```

```
[p,z]=proj(A,b);
```

```
b is in the Col A
    24
    25
    20
    15
    16
    11

     0
     0
     0
     0
     0
     0
```

```
%(c)
A=magic(6), b=(1:5)'
```

```
A = 6×6
    35     1     6    26    19    24
     3    32     7    21    23    25
    31     9     2    22    27    20
     8    28    33    17    10    15
    30     5    34    12    14    16
     4    36    29    13    18    11
b = 5×1
     1
```

```
     2
     3
     4
     5
```

```
[p,z]=proj(A,b);
```

```
No solution: sizes of A and b disagree
```

```
%(d)
A=magic(5), b = rand(5,1)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
b = 5×1
    0.1493
    0.2575
    0.8407
    0.2543
    0.8143
```

```
[p,z]=proj(A,b);
```

```
b is in the Col A
    0.1493
    0.2575
    0.8407
    0.2543
    0.8143

     0
     0
     0
     0
     0
```

```
%(e)
A=ones(4); A(:)=1:16, b=[1;0;1;0]
```

```
A = 4×4
     1     5     9    13
     2     6    10    14
     3     7    11    15
     4     8    12    16
b = 4×1
     1
     0
     1
     0
```

```
[p,z]=proj(A,b);
```

```
the least squares solution of inconsistent system Ax=b is
   -0.4500
    0.2500

A\b returns the least-squares solution of inconsistent system Ax=b
the projection of b onto Col A is
    0.8000
```

```
    0.6000
    0.4000
    0.2000

the component of b orthogonal to Col A is
    0.2000
   -0.6000
    0.6000
   -0.2000

the distance from b to Col A is 8.944272e-01
```

```
%(f)
B=ones(4); B(:)=1:16; A=null(B,'r'), b=ones(4,1)
```

```
A = 4×2
     1     2
    -2    -3
     1     0
     0     1
b = 4×1
     1
     1
     1
     1
```

```
[p,z]=proj(A,b);
```

```
b is orthogonal to Col A
     0
     0
     0
     0

     1
     1
     1
     1
```

Exercise 4

```
type closetozeroroundoff
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
type shrink
```

```
function B=shrink(A)
[~,pivot]=rref(A);
B=A(:,pivot);
end
```

```
type solveall
```

```
function [x1,x2] = solveall(A,b)
```

```
format compact
format long
[m,n] = size(A);
if rank(A) < rank([A b])
    disp('the system is inconsistent - find the least-squares solution')
else
    disp('the system is consistent - find the "exact" solution')
end

x1=A\b;
disp('the solution calculated by the backslash operator is')
display(x1)

if closetozeroroundoff(transpose(A)*A-eye(n),7) == 0
    disp('A has orthonormal columns')
    x2 = inv(eye(n))*transpose(A)*b;
    disp('solution calculated by the Orthogonal Decomposition Theorem is')
    display(x2)
else
    disp('A does not have orthonormal columns')
    disp('an orthonormal basis for Col A is')
    U=orth(A)
    b1=U*transpose(U)*b;
    disp('the projection of b onto Col A is')
    display(b1)
    x2 = A\b1;
    disp('the solution calculated using the projection b1 is')
    display(x2)
end
if closetozeroroundoff(x1-x2,12) == 0
    disp('solutions x1 and x2 are sufficiently close to each other')
else
    disp('Check the code!')
    return
end
n1=norm(b-A*x1);
disp('least-squares error of approximation of b by elements of Col A is')
display(n1)
x = rand(n,1);
n2=norm(b-A*x);
disp('an error of approximation of b by A*x of Col A for a random x is')
display(n2)
```

```
%(a)
A=magic(4); b=A(:,4), A=orth(A)
```

```
b = 4×1
    13
     8
    12
     1
A = 4×3
   -0.5000    0.6708    0.5000
   -0.5000   -0.2236   -0.5000
   -0.5000    0.2236   -0.5000
   -0.5000   -0.6708    0.5000
```

```
[x1,x2]=solveall(A,b);
```

```
the system is consistent - find the "exact" solution
the solution calculated by the backslash operator is
x1 = 3×1
 -16.999999999999996
   8.944271909999152
```

```
  -3.000000000000001
A has orthonormal columns
solution calculated by the Orthogonal Decomposition Theorem is
x2 = 3×1
 -17.000000000000000
   8.944271909999159
  -3.000000000000003
solutions x1 and x2 are sufficiently close to each other
least-squares error of approximation of b by elements of Col A is
n1 =
     6.646518689688359e-15
an error of approximation of b by A*x of Col A for a random x is
n2 =
  19.308078682693104
```

```
%(b)
A=magic(5), b=rand(5,1)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
b = 5×1
   0.196595250431208
   0.251083857976031
   0.616044676146639
   0.473288848902729
   0.351659507062997
```

```
[x1,x2]=solveall(A,b);
```

```
the system is consistent - find the "exact" solution
the solution calculated by the backslash operator is
x1 = 5×1
  -0.008184129724147
   0.002796278350050
   0.010980107448299
   0.013475558291106
   0.009988680104223
A does not have orthonormal columns
an orthonormal basis for Col A is
U = 5×5
  -0.447213595499958  -0.545634873129948   0.511667273601714   0.195439507584854 ···
  -0.447213595499958  -0.449758363151205  -0.195439507584838  -0.511667273601691
  -0.447213595499958  -0.000000000000024  -0.632455532033676   0.632455532033676
  -0.447213595499958   0.449758363151189  -0.195439507584872  -0.511667273601694
  -0.447213595499958   0.545634873129987   0.511667273601672   0.195439507584856
the projection of b onto Col A is
b1 = 5×1
   0.196595250431209
   0.251083857976031
   0.616044676146639
   0.473288848902729
   0.351659507062997
the solution calculated using the projection b1 is
x2 = 5×1
  -0.008184129724147
   0.002796278350050
   0.010980107448299
   0.013475558291106
   0.009988680104223
solutions x1 and x2 are sufficiently close to each other
```

```
least-squares error of approximation of b by elements of Col A is
n1 =
     1.922962686383564e-16
an error of approximation of b by A*x of Col A for a random x is
n2 =
  91.520577981895116
```

```
%(c)
A=magic(4); A=shrink(A), b=ones(4,1)
```

```
A = 4×3
    16     2     3
     5    11    10
     9     7     6
     4    14    15
b = 4×1
     1
     1
     1
     1
```

```
[x1,x2]=solveall(A,b);
```

```
the system is consistent - find the "exact" solution
the solution calculated by the backslash operator is
x1 = 3×1
   0.058823529411765
   0.117647058823529
  -0.058823529411765
A does not have orthonormal columns
an orthonormal basis for Col A is
U = 4×3
  -0.363225569906992  -0.839773278980323   0.335928601456289
  -0.511952614823082   0.201051551215513  -0.497476425501395
  -0.413098103234259  -0.228706948321817  -0.571876812690966
  -0.659789104673461   0.449502219631667   0.559129763025005
the projection of b onto Col A is
b1 = 4×1
   1.000000000000000
   0.999999999999999
   0.999999999999999
   0.999999999999999
the solution calculated using the projection b1 is
x2 = 3×1
   0.058823529411765
   0.117647058823529
  -0.058823529411764
solutions x1 and x2 are sufficiently close to each other
least-squares error of approximation of b by elements of Col A is
n1 =
     4.577566798522237e-16
an error of approximation of b by A*x of Col A for a random x is
n2 =
  30.400634900305239
```

```
%(d)
A=magic(4); A=shrink(A), b=rand(4,1)
```

```
A = 4×3
    16     2     3
     5    11    10
     9     7     6
     4    14    15
b = 4×1
```

```
   0.567821640725221
   0.075854289563064
   0.053950118666607
   0.530797553008973
```

```
[x1,x2]=solveall(A,b);
```

```
the system is inconsistent - find the least-squares solution
the solution calculated by the backslash operator is
x1 = 3×1
   0.019179654123210
  -0.204958989282863
   0.221909441099766
A does not have orthonormal columns
an orthonormal basis for Col A is
U = 4×3
  -0.363225569906992  -0.839773278980323   0.335928601456289
  -0.511952614823082   0.201051551215513  -0.497476425501395
  -0.413098103234259  -0.228706948321817  -0.571876812690966
  -0.659789104673461   0.449502219631667   0.559129763025005
the projection of b onto Col A is
b1 = 4×1
   0.562684810704940
   0.060443799502221
   0.069360608727450
   0.535934383029253
the solution calculated using the projection b1 is
x2 = 3×1
   0.019179654123210
  -0.204958989282863
   0.221909441099766
solutions x1 and x2 are sufficiently close to each other
least-squares error of approximation of b by elements of Col A is
n1 =
   0.022972602228420
an error of approximation of b by A*x of Col A for a random x is
n2 =
  30.843125516205774
```

```
%(e)
A=magic(6); A=orth(A), b=rand(6,1)
```

```
A = 6×5
  -0.408248290463863   0.557411249186207   0.045556029986867  -0.418231543932495 ···
  -0.408248290463863  -0.231229647863760   0.630136447921892  -0.257134627176704
  -0.408248290463863   0.436200183527296   0.269634326252200   0.539061664088837
  -0.408248290463863  -0.395374353495895  -0.242232078683333  -0.458962384952503
  -0.408248290463863   0.149577987800958  -0.684940943059626   0.096936068904036
  -0.408248290463863  -0.516585419154806  -0.018153782418000   0.498330823068829
b = 6×1
   0.568823660872193
   0.469390641058206
   0.011902069501241
   0.337122644398882
   0.162182308193243
   0.794284540683907
```

```
[x1,x2]=solveall(A,b);
```

```
the system is inconsistent - find the least-squares solution
the solution calculated by the backslash operator is
x1 = 5×1
  -0.956813912617036
  -0.305623197007150
```

```
   0.117736214712119
  -0.095369464592118
   0.161013718125105
A has orthonormal columns
solution calculated by the Orthogonal Decomposition Theorem is
x2 = 5×1
  -0.956813912617037
  -0.305623197007150
   0.117736214712119
  -0.095369464592118
   0.161013718125105
solutions x1 and x2 are sufficiently close to each other
least-squares error of approximation of b by elements of Col A is
n1 =
   0.507041743827988
an error of approximation of b by A*x of Col A for a random x is
n2 =
   1.749227110692347
```

It seems that the error is larger for consistent systems, but the error of approximation is many magnitudes larger for inconsistent ones when using a random x. The least squares solution definitely minimizes the distance, as it has a much lower error than the random x's approximation of B

Exercise 5

```
type closetozeroroundoff
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
type shrink_1
```

```
function B=shrink_1(A)
[~,pivot]=rref(A);
disp(pivot)
B=A(:,pivot);
end
```

```
type orthonorm
```

```
function U=orthonorm(A)
format compact
format long
if(rank(A) == size(A,2))
    disp('a basis X for Col A is formed by all columns of A')
    X=A
else
    disp('a basis X for Col A is formed by the columns of A with indexes')
    X=shrink_1(A)
end
U = X;
n = size(U,2);
if(closetozeroroundoff(U'*U-eye(n),15) == zeros(n))
    disp('a required orthonormal basis U for Col A is X')
    display(U)
    return
end
count = 1;
```

```
while(true)
temp = (U(:,1))/norm(U(:,1));
for i = 2:n
    tSum = zeros(size(U,1),1);
    curr = U(:,i);
    for j = temp
        tSum = tSum + (dot(curr,j)/(norm(j))^2)*j;
    end
    %tSum
    %temp
    temp(:,i) = (curr - tSum)/norm((curr - tSum));
end
U = temp;
if(count == 1)
    if(rank(U) ~= rank(X))
        disp('U is not a basis for Col A? Check the code!')
    end
    if(closetozeroroundoff(U'*U-eye(n),0) ~= zeros(n))
        disp('No orthonormalization? Check the code!')
    end
end
p = 1;
while(closetozeroroundoff(U'*U-eye(n),p) == zeros(n))
    if(p >= 15)
        fprintf('accuracy after %i orthonormalisation(s) is at least %d\n',count,p)
        break
    end
    p = p + 1;
end
if(p < 15)
    fprintf('accuracy after %i orthonormalization(s) is %d\n',count,p)
elseif(p >= 15)
    break;
end
count = count + 1;
end
disp('orthonormal basis for Col A that satisfies the required accuracy is')
display(U)
fprintf('U has been generated after %i orthonormalization(s)\n',count)
```

```
%(a)
A=[1 2 -1 4 1;0 1 3 -2 2;0 0 0 2 -2];
U=orthonorm(A);
```

```
a basis X for Col A is formed by the columns of A with indexes
     1     2     4
X = 3×3
     1     2     4
     0     1    -2
     0     0     2
accuracy after 1 orthonormalisation(s) is at least 15
orthonormal basis for Col A that satisfies the required accuracy is
U = 3×3
     1     0     0
     0     1     0
     0     0     1
U has been generated after 1 orthonormalization(s)
```

```
%(b)
A=magic(5);
U=orthonorm(A);
```

```
a basis X for Col A is formed by all columns of A
X = 5×5
```

```
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
accuracy after 1 orthonormalisation(s) is at least 15
orthonormal basis for Col A that satisfies the required accuracy is
U = 5×5
   0.523386711036017   0.505754231034130  -0.673469759051680   0.121541529711433 ···
   0.708111432578141  -0.696573420416297   0.017727408849190  -0.081541631655458
   0.123149814361416   0.136741841080354   0.355750617506511   0.630744102816935
   0.307874535903540   0.191057415586139   0.412230966630674   0.424671602073494
   0.338661989493894   0.451438656528345   0.499631214910400  -0.632767355365590
U has been generated after 1 orthonormalization(s)
```

```
%(c)
A=magic(4);
U=orthonorm(A);
```

```
a basis X for Col A is formed by the columns of A with indexes
     1     2     3
X = 4×3
    16     2     3
     5    11    10
     9     7     6
     4    14    15
accuracy after 1 orthonormalisation(s) is at least 15
orthonormal basis for Col A that satisfies the required accuracy is
U = 4×3
   0.822951199797823  -0.418557220533073   0.312347844383391
   0.257172249936820   0.515451569223407  -0.467088978298099
   0.462910049886276   0.130510755378817  -0.564518581133284
   0.205737799949456   0.736265221000698   0.604636652888949
U has been generated after 1 orthonormalization(s)
```

```
%(d)
A=randi(10,4,6);
U=orthonorm(A);
```

```
a basis X for Col A is formed by the columns of A with indexes
     1     2     3     4
X = 4×4
     7     1     9     5
     7     3     6     2
     8    10    10    10
     5     2     1     1
accuracy after 1 orthonormalisation(s) is at least 15
orthonormal basis for Col A that satisfies the required accuracy is
U = 4×4
   0.511890696888991  -0.543425898078570   0.559495518713382   0.360029126982312
   0.511890696888991  -0.225364417826011  -0.164394420753576  -0.812498164946569
   0.585017939301705   0.787499814742967   0.152936884975707   0.119198832581982
   0.365636212063565  -0.183693261322334  -0.797840553104859   0.442738521018790
U has been generated after 1 orthonormalization(s)
```

```
%(e)
A=hilb(7);
U=orthonorm(A);
```

```
a basis X for Col A is formed by all columns of A
X = 7×7
   1.000000000000000   0.500000000000000   0.333333333333333   0.250000000000000 ···
   0.500000000000000   0.333333333333333   0.250000000000000   0.200000000000000
```

```
   0.333333333333333   0.250000000000000   0.200000000000000   0.166666666666667
   0.250000000000000   0.200000000000000   0.166666666666667   0.142857142857143
   0.200000000000000   0.166666666666667   0.142857142857143   0.125000000000000
   0.166666666666667   0.142857142857143   0.125000000000000   0.111111111111111
   0.142857142857143   0.125000000000000   0.111111111111111   0.100000000000000
accuracy after 1 orthonormalization(s) is 1
accuracy after 2 orthonormalisation(s) is at least 15
orthonormal basis for Col A that satisfies the required accuracy is
U = 7×7
   0.813304645434915  -0.543794290342009   0.199095307552452  -0.055113252545769 ⋯
   0.406652322717457   0.303317262369621  -0.688560547825031   0.475965264949984
   0.271101548478305   0.393949644093289  -0.207134625521039  -0.490111167000235
   0.203326161358729   0.381744394201061   0.112389155791078  -0.439598472616485
   0.162660929086983   0.351412667964099   0.291518454908453  -0.112276608353538
   0.135550774239152   0.320235052233921   0.389191824334151   0.230886766394466
   0.116186377919274   0.292095791941544   0.440744902840134   0.520623748142910
U has been generated after 2 orthonormalization(s)
```

```
%(f)
A=hilb(9);
U=orthonorm(A);
```

```
a basis X for Col A is formed by all columns of A
X = 9×9
   1.000000000000000   0.500000000000000   0.333333333333333   0.250000000000000 ⋯
   0.500000000000000   0.333333333333333   0.250000000000000   0.200000000000000
   0.333333333333333   0.250000000000000   0.200000000000000   0.166666666666667
   0.250000000000000   0.200000000000000   0.166666666666667   0.142857142857143
   0.200000000000000   0.166666666666667   0.142857142857143   0.125000000000000
   0.166666666666667   0.142857142857143   0.125000000000000   0.111111111111111
   0.142857142857143   0.125000000000000   0.111111111111111   0.100000000000000
   0.125000000000000   0.111111111111111   0.100000000000000   0.090909090909091
   0.111111111111111   0.100000000000000   0.090909090909091   0.083333333333333
accuracy after 1 orthonormalization(s) is 1
accuracy after 2 orthonormalization(s) is 9
accuracy after 3 orthonormalisation(s) is at least 15
orthonormal basis for Col A that satisfies the required accuracy is
U = 9×9
   0.805883739588529  -0.548744524629204   0.212016522663850  -0.064769402599622 ⋯
   0.402941869794265   0.266771946706805  -0.666365298228793   0.505198450145897
   0.268627913196176   0.358229367478340  -0.264391854571240  -0.378623524673989
   0.201470934897132   0.349843656961966   0.013832223213607  -0.449277132479077
   0.161176747917706   0.323166462291285   0.174177938201102  -0.266484898870000
   0.134313956598088   0.295074157100900   0.264550563796192  -0.043737917413956
   0.115126248512647   0.269486345138161   0.314672091320436   0.155525245103848
   0.100735467448566   0.247074389683837   0.341242501100238   0.317624240622229
   0.089542637732059   0.227638630963728   0.353725848217708   0.444267113807640
U has been generated after 3 orthonormalization(s)
```

```
%(g)
X=orth(hilb(5));
U=orthonorm(X);
```

```
a basis X for Col A is formed by all columns of A
X = 5×5
  -0.767854735065807   0.601871478353973  -0.214213624140621   0.047161806831170 ⋯
  -0.445791060462709  -0.275913417432099   0.724102131480104  -0.432667334694428
  -0.321578294480220  -0.424876622351522   0.120453278536018   0.667350443883441
  -0.253438943245175  -0.443903038699774  -0.309573969853397   0.233024515280419
  -0.209822636563631  -0.429013353681693  -0.565193410522136  -0.557599947804642
a required orthonormal basis U for Col A is X
U = 5×5
  -0.767854735065807   0.601871478353973  -0.214213624140621   0.047161806831170 ⋯
  -0.445791060462709  -0.275913417432099   0.724102131480104  -0.432667334694428
```

```
  -0.321578294480220  -0.424876622351522   0.120453278536018   0.667350443883441
  -0.253438943245175  -0.443903038699774  -0.309573969853397   0.233024515280419
  -0.209822636563631  -0.429013353681693  -0.565193410522136  -0.557599947804642
```

```
%(h)
A=[orth(hilb(5)), ones(5)];
U=orthonorm(A);
```

```
a basis X for Col A is formed by the columns of A with indexes
     1     2     3     4     5
X = 5×5
  -0.767854735065807   0.601871478353973  -0.214213624140621   0.047161806831170 ···
  -0.445791060462709  -0.275913417432099   0.724102131480104  -0.432667334694428
  -0.321578294480220  -0.424876622351522   0.120453278536018   0.667350443883441
  -0.253438943245175  -0.443903038699774  -0.309573969853397   0.233024515280419
  -0.209822636563631  -0.429013353681693  -0.565193410522136  -0.557599947804642
a required orthonormal basis U for Col A is X
U = 5×5
  -0.767854735065807   0.601871478353973  -0.214213624140621   0.047161806831170 ···
  -0.445791060462709  -0.275913417432099   0.724102131480104  -0.432667334694428
  -0.321578294480220  -0.424876622351522   0.120453278536018   0.667350443883441
  -0.253438943245175  -0.443903038699774  -0.309573969853397   0.233024515280419
  -0.209822636563631  -0.429013353681693  -0.565193410522136  -0.557599947804642
```

```
%(k)
A=orth(hilb(7));
U=orthonorm(A);
```

```
a basis X for Col A is formed by all columns of A
X = 7×7
  -0.733225603080614   0.623238511591099   0.260842643619283  -0.075187279139429 ···
  -0.436359150069654  -0.163071523456998  -0.670558489510334   0.526777549010783
  -0.319779114044051  -0.321514146331300  -0.295329507071720  -0.425656183169809
  -0.254885556321454  -0.357367406020183   0.023046632443994  -0.461676123346221
  -0.212844074668574  -0.357068305453527   0.233687618042103  -0.171195538795242
  -0.183143115876330  -0.344569978962172   0.367875851069302   0.182664476543393
  -0.160939670445336  -0.328134700342623   0.452348557834480   0.509754876662158
accuracy after 1 orthonormalisation(s) is at least 15
orthonormal basis for Col A that satisfies the required accuracy is
U = 7×7
  -0.733225603080613   0.623238511591098   0.260842643619283  -0.075187279139429 ···
  -0.436359150069654  -0.163071523456998  -0.670558489510334   0.526777549010783
  -0.319779114044051  -0.321514146331300  -0.295329507071720  -0.425656183169809
  -0.254885556321454  -0.357367406020183   0.023046632443994  -0.461676123346220
  -0.212844074668574  -0.357068305453527   0.233687618042103  -0.171195538795242
  -0.183143115876329  -0.344569978962171   0.367875851069302   0.182664476543393
  -0.160939670445336  -0.328134700342622   0.452348557834480   0.509754876662158
U has been generated after 1 orthonormalization(s)
```

Exercise 6

```
type lstsqpoly.m
```

```
function [c,X,N]=lstsqpoly(x,y,n)
format
m=length(x);
%1
a=x(1)
b=x(end)
X=ones(m,n+1);
for i=1:m
    for j=0:n
```

```
    X(i,j+1)= power(x(i,1),n-j);
    end
end
%2
disp('the design matrix is')
disp(X)
%3
c=X\y
disp('the parameter vector is')
display(c)
%4
e=y-X*c;
disp('the norm of the residual vector is')
N=norm(e);
disp(N)
%5
plot(x,y,'*'),hold on
polyplot(a,b,c')
%6
fprintf('the polynomial of degree %i of the best least-squares fit is\n',n)
P=poly2sym(c);
%7
hold off

end
```

```
type polyplot.m
```

```
function [] = polyplot(a,b,p)
x = (a:b-a)/50:b;
y= polyval(p,x);
plot(x,y);
end
```

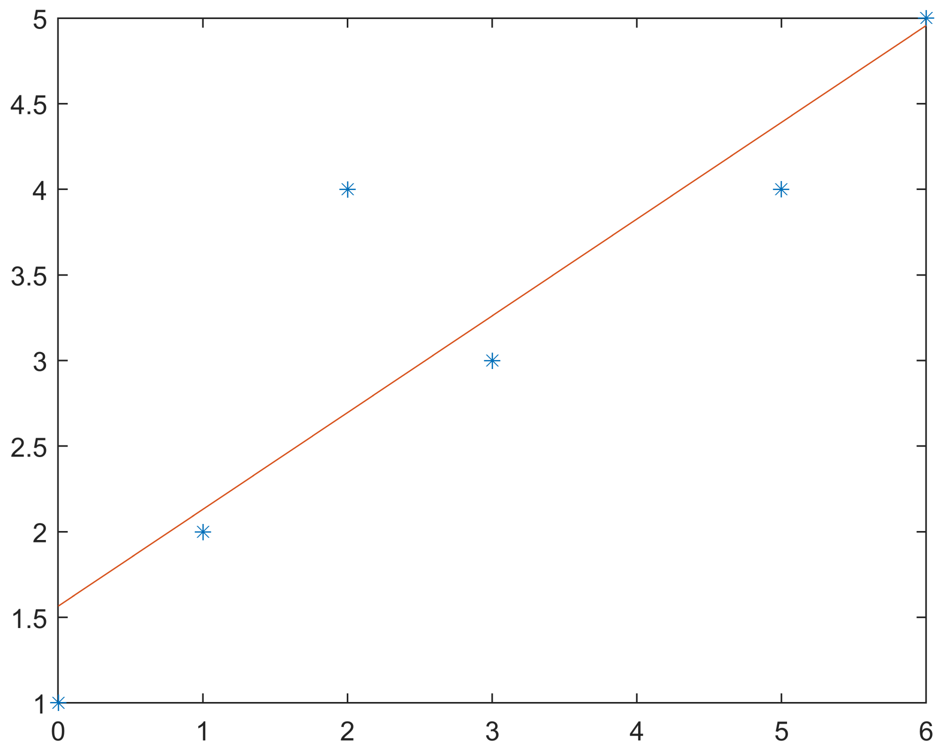
```
x = [0;1;2;3;5;6], y = [1;2;4;3;4;5]
```

```
x = 6×1
     0
     1
     2
     3
     5
     6
y = 6×1
     1
     2
     4
     3
     4
     5
```

```
n=1;
[c,X,N]=lstsqpoly(x,y,n);
```

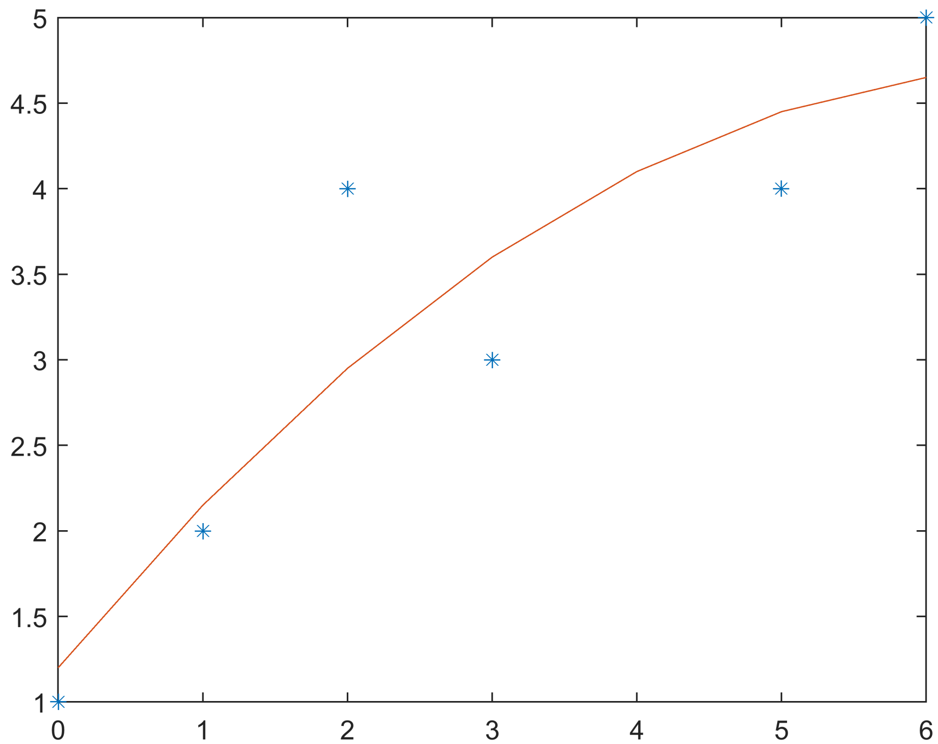
```
a = 0
b = 6
the design matrix is
     0     1
     1     1
     2     1
     3     1
     5     1
     6     1
```

```
c = 2×1
    0.5652
    1.5652
the parameter vector is
c = 2×1
    0.5652
    1.5652
the norm of the residual vector is
    1.5036
the polynomial of degree 1 of the best least-squares fit is
```



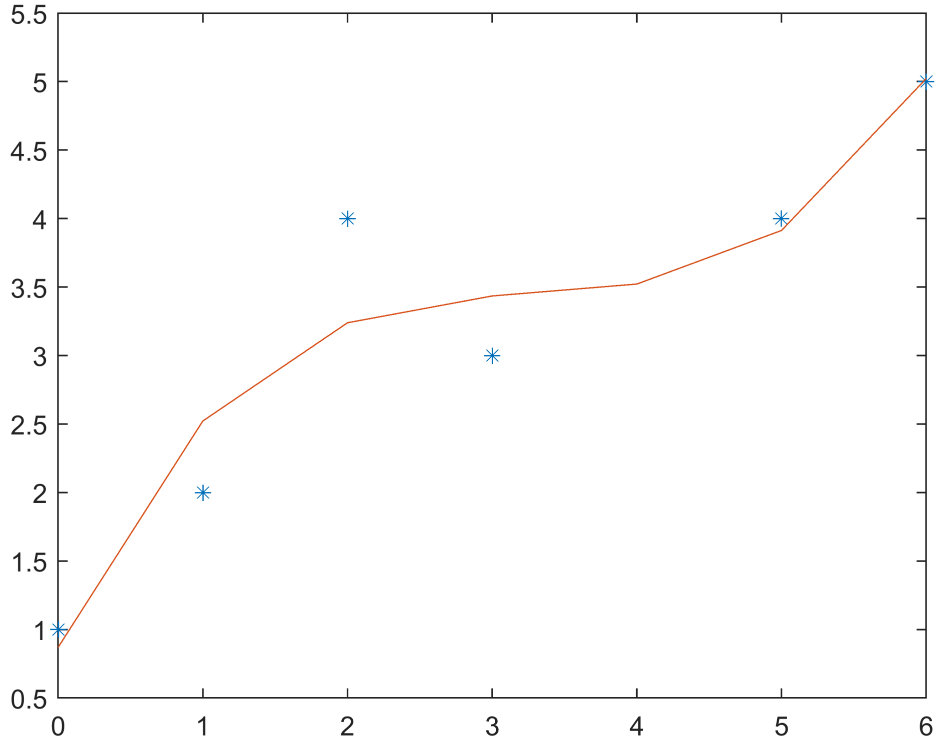

```
n=2;
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 0
b = 6
the design matrix is
     0     0     1
     1     1     1
     4     2     1
     9     3     1
    25     5     1
    36     6     1
c = 3×1
   -0.0750
    1.0250
    1.2000
the parameter vector is
c = 3×1
   -0.0750
    1.0250
    1.2000
the norm of the residual vector is
    1.3601
the polynomial of degree 2 of the best least-squares fit is
```



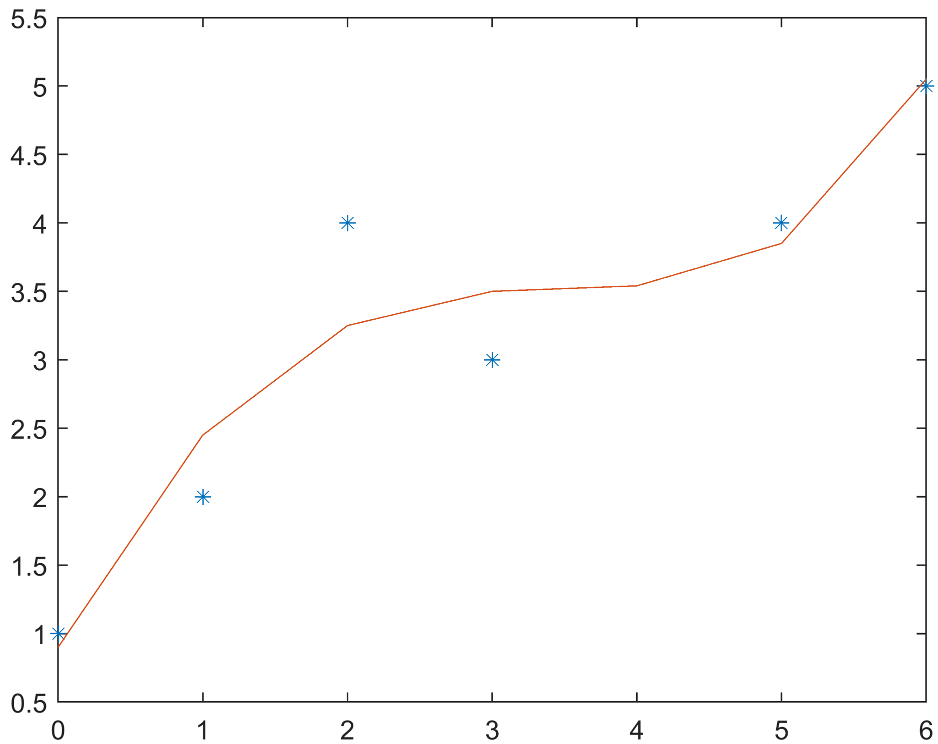

```
n=3;
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 0
b = 6
the design matrix is
     0     0     0     1
     1     1     1     1
     8     4     2     1
    27     9     3     1
   125    25     5     1
   216    36     6     1
c = 4×1
    0.0688
   -0.6739
    2.2572
    0.8696
the parameter vector is
c = 4×1
    0.0688
   -0.6739
    2.2572
    0.8696
the norm of the residual vector is
    1.0321
the polynomial of degree 3 of the best least-squares fit is
```



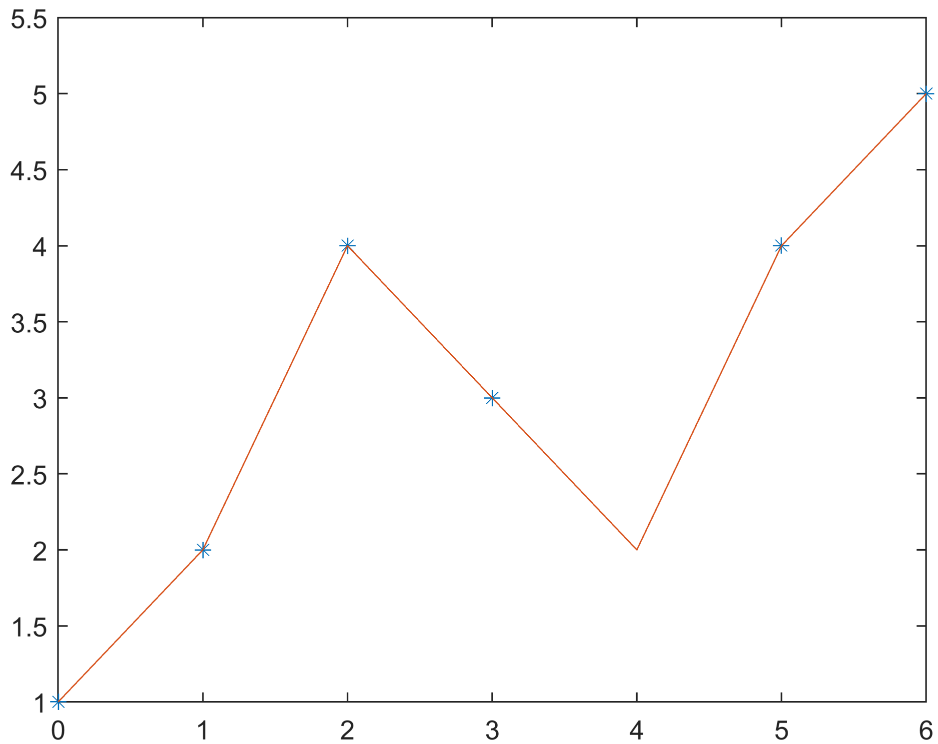

```
n=4;
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 0
b = 6
the design matrix is
           0           0           0           0           1
           1           1           1           1           1
          16           8           4           2           1
          81          27           9           3           1
         625         125          25           5           1
        1296         216          36           6           1
c = 5×1
    0.0058
   -0.0017
   -0.4108
    1.9567
    0.9000
the parameter vector is
c = 5×1
    0.0058
   -0.0017
   -0.4108
    1.9567
    0.9000
the norm of the residual vector is
    1.0247
the polynomial of degree 4 of the best least-squares fit is
```




```
n=5;
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 0
b = 6
the design matrix is
           0           0           0           0           0           1
           1           1           1           1           1           1
          32          16           8           4           2           1
         243          81          27           9           3           1
        3125         625         125          25           5           1
        7776        1296         216          36           6           1
c = 6×1
   -0.0583
    0.8750
   -4.4583
    8.6250
   -3.9833
    1.0000
the parameter vector is
c = 6×1
   -0.0583
    0.8750
   -4.4583
    8.6250
   -3.9833
    1.0000
the norm of the residual vector is
   1.6758e-14
the polynomial of degree 5 of the best least-squares fit is
```



BONUS:

For n=5, the best least-squares fit polynomial of degree 5 interpolates

the 6 data points because the polynomial gets close enough to the data

points. Each time n is incremented the design matrices become more similar

to the points. For data with x points, you need a x-l polynomial.With the 6 data points we have, a 5th degree polynomial is suficient to pass

through every point.

Exercise 7

```
type invalprob.m
```

```
function []=invalprob(A,x0)
format
[~,n]=size(A);
L = eigenForExercise7(A);
L = transpose(L);
L = real(L);
L = sort(L);
P = [];
counter = numel(L);
%Calculating L
for i=1:(size(L,2)-1)
    for j=2:size(L,2)
        if(closetozeroroundoff(L(i)-L(j),7)==0)
```

```
            L(i) = L(j);
        end
    end
end

%Continued Calculation of L
if (rank(A) ~= n)
    for i=1:size(L,2)
        if(closetozeroroundoff(L(i),7) == 0)
            L(i) = 0;
        end
    end
end           %L finished calculating



UniL = unique(L);
zeroUniL = zeros(numel(UniL));
for i = 1:numel(UniL)
    for j = 1: counter
        if UniL(i)== L(j)
            zeroUniL(i) = 1 + zeroUniL(i);
        end
    end

end


zeroUniLTwo = zeros(numel(UniL));
for i = 1:numel(UniL)
    temp = UniL(i) * eye(size(A));
    temp = A - temp;
    temp = null(temp);
    zeroUniLTwo(i) = size(temp,2);
    P = [P temp];
end

%Reset rount to zero
count = 0;
[~,k] = size(zeroUniLTwo);
for i=1:k
    if zeroUniLTwo(i) == zeroUniL(i)
        count = 1 + count;
    end
end
if count == k
    disp('A is diagonlizable')
else
    disp('A is not diagonlizable')
    return;
end
fprintf('all eigenvalues of A are\n')
display(L)
fprintf('an eigenvector basis for R^%i\n',n)
display(P)

syms t
Eye = eye(n);
Eye = sym(Eye);
Eyes = Eye;
for i=1:n
    Eyes(i,i)=exp(L(i)*t);
end
Eye=Eyes;
Eye = P*Eye;
```

```
display(Eye)
C = inv(P)*x0;
display(C)
disp('Entries are the weights in the representation of the solution x through the eigenfunction basis E')
if n == 2
    if rank(A) == 2
        if (L(2) > 0) && (L(1) > 0)
            disp('the origin is a repellor')
            if L(1) > L(2)
                F = P(:,1);
                disp('the direcrtion of greatest repulsion is')
                display(F)
            else
                F = P(:,2);
                disp('the direction of greatest repulsion is')
                display(F)
            end
        elseif(L(1) < 0) && (L(2) < 0)
            disp('the origin is an attractor')
            if L(1) < L(2)
                F = P(:,1);
                disp('the direcrtion of greatest attraction is')
                display(F)
            else
                F = P(:,2);
                disp('the direction of greatest attraction is')
                display(F)
            end
        else
            disp('the origin is a saddle point')
            if L(1) > L(2)
                F = P(:,1);
                %display(F)
                disp('the direcrtion of greatest repulsion is')
                display(F)
                G = P(:,2);
                disp('the direction of greatest attraction is')
                display(G)
            else
                F = P(:,2);
               % display(F)
                disp('the direcrtion of greatest repulsion is')
                display(F)
                G = P(:,1);
                disp('the direction of greatest attraction is')
                display(G)
            end
        end
    end
else
    disp('A has a 0 eigenvalue')
end
```

```
% (a)
A = ones(2), x0=[1;2]
```

```
A = 2×2
     1     1
     1     1
x0 = 2×1
     1
     2
```

```
invalprob(A,x0)
```

```
A is diagonlizable
all eigenvalues of A are
L = 2×1
     0
     2
an eigenvector basis for R^2
P = 2×2
   -0.7071    0.7071
    0.7071    0.7071
Eye =
```

$$\begin{pmatrix} -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}\,e^{2t}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}\,e^{2t}}{2} \end{pmatrix}$$

```
C = 2×1
    0.7071
    2.1213
Entries are the weights in the representation of the solution x through the eigenfunction basis E
```

```
%(b)
A=[-2 -5;1 4], x0=[2;3]
```

```
A = 2×2
    -2    -5
     1     4
x0 = 2×1
     2
     3
```

```
invalprob(A,x0)
```

```
A is diagonlizable
all eigenvalues of A are
L = 2×1
    -1
     3
an eigenvector basis for R^2
P = 2×2
   -0.9806   -0.7071
    0.1961    0.7071
Eye =
```

$$\begin{pmatrix} -\frac{5\,\sqrt{26}\,e^{-t}}{26} & -\frac{\sqrt{2}\,e^{3t}}{2} \\ \frac{\sqrt{26}\,e^{-t}}{26} & \frac{\sqrt{2}\,e^{3t}}{2} \end{pmatrix}$$

```
C = 2×1
   -6.3738
    6.0104
Entries are the weights in the representation of the solution x through the eigenfunction basis E
the origin is a saddle point
the direcrtion of greatest repulsion is
F = 2×1
   -0.7071
    0.7071
the direction of greatest attraction is
G = 2×1
   -0.9806
```

```
    0.1961
```

```
%(c)
A=[7 -1;3 3], x0=[-2;1]
```

```
A = 2×2
     7    -1
     3     3
x0 = 2×1
    -2
     1
```

```
invalprob(A,x0)
```

```
A is diagonlizable
all eigenvalues of A are
L = 2×1
     4
     6
an eigenvector basis for R^2
P = 2×2
   -0.3162   -0.7071
   -0.9487   -0.7071
Eye =
```

$$\begin{pmatrix} -\frac{\sqrt{10}\,e^{4t}}{10} & -\frac{\sqrt{2}\,e^{6t}}{2} \\ -\frac{3\,\sqrt{10}\,e^{4t}}{10} & -\frac{\sqrt{2}\,e^{6t}}{2} \end{pmatrix}$$

```
C = 2×1
   -4.7434
    4.9497
Entries are the weights in the representation of the solution x through the eigenfunction basis E
the origin is a repellor
the direction of greatest repulsion is
F = 2×1
   -0.7071
   -0.7071
```

```
%(d)
A=[-1.5 .5; 1 -1], x0=[5;4]
```

```
A = 2×2
   -1.5000    0.5000
    1.0000   -1.0000
x0 = 2×1
     5
     4
```

```
invalprob(A,x0)
```

```
A is diagonlizable
all eigenvalues of A are
L = 2×1
   -2.0000
   -0.5000
an eigenvector basis for R^2
P = 2×2
   -0.7071    0.4472
    0.7071    0.8944
Eye =
```

$$\begin{pmatrix} -\frac{\sqrt{2}\,e^{-2t}}{2} & \frac{\sqrt{5}\,e^{-\frac{t}{2}}}{5} \\ \frac{\sqrt{2}\,e^{-2t}}{2} & \frac{2\,\sqrt{5}\,e^{-\frac{t}{2}}}{5} \end{pmatrix}$$

```
C = 2×1
   -2.8284
    6.7082
Entries are the weights in the representation of the solution x through the eigenfunction basis E
the origin is an attractor
the direcrtion of greatest attraction is
F = 2×1
   -0.7071
    0.7071
```

```
%(e)
A=[3 3;0 3], x0=[1;3]
```

```
A = 2×2
     3     3
     0     3
x0 = 2×1
     1
     3
```

```
invalprob(A,x0)
```

```
A is not diagonlizable
```

```
%(f)
A=diag([1,2,2,3,3,3]), x0=ones(6,1)
```

```
A = 6×6
     1     0     0     0     0     0
     0     2     0     0     0     0
     0     0     2     0     0     0
     0     0     0     3     0     0
     0     0     0     0     3     0
     0     0     0     0     0     3
x0 = 6×1
     1
     1
     1
     1
     1
     1
```

```
invalprob(A,x0)
```

```
A is diagonlizable
all eigenvalues of A are
L = 6×1
     1
     2
     2
     3
     3
     3
an eigenvector basis for R^6
P = 6×6
```

```
     1     0     0     0     0     0
     0     1     0     0     0     0
     0     0     1     0     0     0
     0     0     0     0     0     1
     0     0     0     1     0     0
     0     0     0     0     1     0
Eye =
```

$$\begin{pmatrix} e^{t} & 0 & 0 & 0 & 0 & 0 \\ 0 & e^{2t} & 0 & 0 & 0 & 0 \\ 0 & 0 & e^{2t} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & e^{3t} \\ 0 & 0 & 0 & e^{3t} & 0 & 0 \\ 0 & 0 & 0 & 0 & e^{3t} & 0 \end{pmatrix}$$

```
C = 6×1
     1
     1
     1
     1
     1
     1
Entries are the weights in the representation of the solution x through the eigenfunction basis E
A has a 0 eigenvalue
```