MATLAB PROJECT 4

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 # 4

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

_1._Edwin Salcedo

_2._Anisha Paul

_3._Brandon Tran

_4._John Dinh

_5._Ananya Kakaveti

_6._Yaaseen Mohammad

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

Exercise 1

```
type eigendiag
```

```
function [L,P,D]=eigendiag(A)
format
[~,n]=size(A);
P=[];
D=[];

%Part 1
%output eigenvalues with their multiplicity
L = eig(A);
L = sort(L, 'ascend', 'ComparisonMethod','real');
for i = 2:size(L, 1)
   if isequal(closetozeroroundoff(L(i,1)-L(i-1, 1), 7), 0)
       L(i, :) = L(i-1, :);
   end
end

for i = 1:size(L, 1)
    if closetozeroroundoff(L(i, 1)-real(L(i, 1)), 7)==0
        L(i, 1) = real(L(i, 1));
    end
end

%matrix not invertible
if rank(A)~=size(A, 2)
    closetozeroroundoff(L, 12)
end

disp('all distinct eigenvalues of A')
disp(L)

%Part 2
[m, M] = groupcounts(L);

disp('the distinct eigenvalues of A are')
disp(M)

flag = 0;

for k = 1 : length(M)
    lambda = M(k)
    multiplicity = m(k)
    mod = A - lambda.*eye(size(A));
    mod = rref(mod);
    W = null(mod);
    disp('a basis for the eigenspace for this lambda is')
    disp(W)
    P = [P W];
    dim = size(A, 2) - rank(mod);
    if dim < multiplicity
        disp('dimension of the eigenspace is less than multiplicity of lambda')
        flag = 1;
    end
end

%Part 3
%not diagonalizable
if isequal(flag, 1)
        disp('A is not diagonalizable')
        P = [];
        D = [];
```

```
    return
end


%diagonalizable
D = diag(L);
A*P;
P*D;
if ~any(closetozeroroundoff(A*P - P*D, 7)) & isequal(rank(P), size(P, 2))
    disp('A is diagonalized')
    display(P)
    display(D)
else
    disp('Oops! I got a bug in my code!')
    P = [];
    D = [];
    return
end

%Bonus
symmetric = isequal(transpose(A), A);
if symmetric
    disp('matrix A is symmetric')
else
    return
end
if symmetric & closetozeroroundoff(P'*P-eye(n), 7)==0
    disp('the orthogonal diagonalization is confirmed')
else
    disp('Wow! A symmetric matrix is not orthogonally diagonalizable')
end
```

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

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

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

```
all distinct eigenvalues of A
     3
     3

the distinct eigenvalues of A are
     3
lambda = 3
multiplicity = 2
a basis for the eigenspace for this lambda is
    -1
     0

dimension of the eigenspace is less than multiplicity of lambda
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]=eigendiag(A);
```

```
all distinct eigenvalues of A
   -2.0000
   -2.0000
    1.0000

the distinct eigenvalues of A are
   -2.0000
    1.0000
lambda = -2.0000
multiplicity = 2
a basis for the eigenspace for this lambda is
    0.7071
   -0.7071
         0

dimension of the eigenspace is less than multiplicity of lambda
lambda = 1.0000
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.5774
    0.5774
   -0.5774

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]=eigendiag(A);
```

```
all distinct eigenvalues of A
     3
     3
     5
     5

the distinct eigenvalues of A are
     3
     5
lambda = 3
multiplicity = 2
a basis for the eigenspace for this lambda is
   -0.7071         0
         0   -0.7071
    0.7071         0
         0    0.7071
lambda = 5
multiplicity = 2
a basis for the eigenspace for this lambda is
   -0.7071         0
         0    0.7071
   -0.7071         0
         0    0.7071

A is diagonalized
```

```
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
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

```
% %(d)
% A=jord(4,3)
% [L,P,D]=eigendiag(A);
%(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]=eigendiag(A);
```

```
ans = 4×1
         0
         0
         0
    4.0000
all distinct eigenvalues of A
   -0.0000
   -0.0000
   -0.0000
    4.0000

the distinct eigenvalues of A are
   -0.0000
    4.0000
lambda = -3.5321e-16
multiplicity = 3
a basis for the eigenspace for this lambda is
         0         0    0.8660
   -0.5774   -0.5774   -0.2887
    0.7887   -0.2113   -0.2887
   -0.2113    0.7887   -0.2887
lambda = 4.0000
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.5000
   -0.5000
   -0.5000
   -0.5000

A is diagonalized
P = 4×4
         0         0    0.8660   -0.5000
   -0.5774   -0.5774   -0.2887   -0.5000
    0.7887   -0.2113   -0.2887   -0.5000
   -0.2113    0.7887   -0.2887   -0.5000
```

```
D = 4×4
   -0.0000         0         0         0
         0   -0.0000         0         0
         0         0   -0.0000         0
         0         0         0    4.0000
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

```
%(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]=eigendiag(A);
```

```
all distinct eigenvalues of A
    1.0000
    1.0000
    5.0000
    9.0000

the distinct eigenvalues of A are
    1.0000
    5.0000
    9.0000
lambda = 1
multiplicity = 2
a basis for the eigenspace for this lambda is
   -0.7071         0
         0   -0.7071
    0.7071         0
         0    0.7071
lambda = 5.0000
multiplicity = 1
a basis for the eigenspace for this lambda is
dimension of the eigenspace is less than multiplicity of lambda
lambda = 9
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.5000
   -0.5000
   -0.5000
   -0.5000

A is not diagonalizable
```

```
%(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]=eigendiag(A);
```

```
all distinct eigenvalues of A
    2.0000
```

```
    2.0000
    5.0000

the distinct eigenvalues of A are
    2.0000
    5.0000
lambda = 2.0000
multiplicity = 2
a basis for the eigenspace for this lambda is
         0    0.8165
   -0.7071   -0.4082
    0.7071   -0.4082
lambda = 5.0000
multiplicity = 1
a basis for the eigenspace for this lambda is
dimension of the eigenspace is less than multiplicity of lambda
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]=eigendiag(A);
```

```
ans = 4×1
   -8.9443
         0
    8.9443
   34.0000
all distinct eigenvalues of A
   -8.9443
   -0.0000
    8.9443
   34.0000

the distinct eigenvalues of A are
   -8.9443
   -0.0000
    8.9443
   34.0000
lambda = -8.9443
multiplicity = 1
a basis for the eigenspace for this lambda is
    0.3764
    0.0236
    0.4236
   -0.8236
lambda = -9.6438e-16
multiplicity = 1
a basis for the eigenspace for this lambda is
    0.2236
    0.6708
   -0.6708
   -0.2236
lambda = 8.9443
multiplicity = 1
a basis for the eigenspace for this lambda is
    0.8236
```

```
   -0.4236
   -0.0236
   -0.3764
lambda = 34.0000
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.5000
   -0.5000
   -0.5000
   -0.5000

A is diagonalized
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   -0.0000         0         0
         0         0    8.9443         0
         0         0         0   34.0000
```

```
%(i)
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]=eigendiag(A);
```

```
all distinct eigenvalues of A
  -21.2768
  -13.1263
   13.1263
   21.2768
   65.0000

the distinct eigenvalues of A are
  -21.2768
  -13.1263
   13.1263
   21.2768
   65.0000
lambda = -21.2768
multiplicity = 1
a basis for the eigenspace for this lambda is
    0.0976
    0.3525
    0.5501
   -0.3223
   -0.6780
lambda = -13.1263
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.6330
    0.5895
   -0.3915
```

```
    0.1732
    0.2619
lambda = 13.1263
multiplicity = 1
a basis for the eigenspace for this lambda is
    0.2619
    0.1732
   -0.3915
    0.5895
   -0.6330
lambda = 21.2768
multiplicity = 1
a basis for the eigenspace for this lambda is
dimension of the eigenspace is less than multiplicity of lambda
lambda = 65.0000
multiplicity = 1
a basis for the eigenspace for this lambda is
dimension of the eigenspace is less than multiplicity of lambda
A is not diagonalizable
```

```
%(j)
A=pascal(4)
```

```
A = 4×4
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
```

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

```
all distinct eigenvalues of A
    0.0380
    0.4538
    2.2034
   26.3047

the distinct eigenvalues of A are
    0.0380
    0.4538
    2.2034
   26.3047
lambda = 0.0380
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.3087
    0.7231
   -0.5946
    0.1684
lambda = 0.4538
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.7873
    0.1632
    0.5321
   -0.2654
lambda = 2.2034
multiplicity = 1
a basis for the eigenspace for this lambda is
    0.5304
    0.6403
    0.3918
   -0.3939
```

```
lambda = 26.3047
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.0602
   -0.2012
   -0.4581
   -0.8638

A is diagonalized
P = 4×4
   -0.3087   -0.7873    0.5304   -0.0602
    0.7231    0.1632    0.6403   -0.2012
   -0.5946    0.5321    0.3918   -0.4581
    0.1684   -0.2654   -0.3939   -0.8638
D = 4×4
    0.0380         0         0         0
         0    0.4538         0         0
         0         0    2.2034         0
         0         0         0   26.3047
matrix A is symmetric
the orthogonal diagonalization is confirmed
```

```
%(k)
A=[0 0 .33;.18 0 0;0 .71 .94]
```

```
A = 3×3
         0         0    0.3300
    0.1800         0         0
         0    0.7100    0.9400
```

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

```
all distinct eigenvalues of A
  -0.0218 - 0.2059i
  -0.0218 + 0.2059i
   0.9836 + 0.0000i

the distinct eigenvalues of A are
  -0.0218 - 0.2059i
  -0.0218 + 0.2059i
   0.9836 + 0.0000i
lambda = -0.0218 - 0.2059i
multiplicity = 1
a basis for the eigenspace for this lambda is
   0.6821 + 0.0000i
  -0.0624 + 0.5896i
  -0.0451 - 0.4256i
lambda = -0.0218 + 0.2059i
multiplicity = 1
a basis for the eigenspace for this lambda is
   0.6821 + 0.0000i
  -0.0624 - 0.5896i
  -0.0451 + 0.4256i
lambda = 0.9836
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.3175
   -0.0581
   -0.9465

A is diagonalized
P = 3×3 complex
   0.6821 + 0.0000i   0.6821 + 0.0000i  -0.3175 + 0.0000i
```

```
  -0.0624 + 0.5896i  -0.0624 - 0.5896i  -0.0581 + 0.0000i
  -0.0451 - 0.4256i  -0.0451 + 0.4256i  -0.9465 + 0.0000i
D = 3×3 complex
  -0.0218 - 0.2059i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i  -0.0218 + 0.2059i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9836 + 0.0000i
```

```
%(l)
A=[0 -1;1 0]
```

```
A = 2×2
     0    -1
     1     0
```

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

```
all distinct eigenvalues of A
   0.0000 - 1.0000i
   0.0000 + 1.0000i

the distinct eigenvalues of A are
   0.0000 - 1.0000i
   0.0000 + 1.0000i
lambda = 0.0000 - 1.0000i
multiplicity = 1
a basis for the eigenspace for this lambda is
   0.7071 + 0.0000i
   0.0000 + 0.7071i
lambda = 0.0000 + 1.0000i
multiplicity = 1
a basis for the eigenspace for this lambda is
   0.7071 + 0.0000i
   0.0000 - 0.7071i

A is diagonalized
P = 2×2 complex
   0.7071 + 0.0000i   0.7071 + 0.0000i
   0.0000 + 0.7071i   0.0000 - 0.7071i
D = 2×2 complex
   0.0000 - 1.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 1.0000i
```

```
%(m)
A=[0 0 .33;.3 0 0;0 .71 .94]
```

```
A = 3×3
         0         0    0.3300
    0.3000         0         0
         0    0.7100    0.9400
```

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

```
all distinct eigenvalues of A
  -0.0345 - 0.2617i
  -0.0345 + 0.2617i
   1.0090 + 0.0000i

the distinct eigenvalues of A are
  -0.0345 - 0.2617i
  -0.0345 + 0.2617i
   1.0090 + 0.0000i
lambda = -0.0345 - 0.2617i
```

```
multiplicity = 1
a basis for the eigenspace for this lambda is
   0.5840 + 0.0000i
  -0.0868 + 0.6582i
  -0.0611 - 0.4631i
lambda = -0.0345 + 0.2617i
multiplicity = 1
a basis for the eigenspace for this lambda is
   0.5840 + 0.0000i
  -0.0868 - 0.6582i
  -0.0611 + 0.4631i
lambda = 1.0090
multiplicity = 1
a basis for the eigenspace for this lambda is
   -0.3095
   -0.0920
   -0.9464

A is diagonalized
P = 3×3 complex
   0.5840 + 0.0000i   0.5840 + 0.0000i  -0.3095 + 0.0000i
  -0.0868 + 0.6582i  -0.0868 - 0.6582i  -0.0920 + 0.0000i
  -0.0611 - 0.4631i  -0.0611 + 0.4631i  -0.9464 + 0.0000i
D = 3×3 complex
  -0.0345 - 0.2617i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i  -0.0345 + 0.2617i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0090 + 0.0000i
```

Exercise 2

```
type closetozeroroundoff.m
```

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

```
type qrschmidt.m
```

```
function [U,V,tolerance]=qrschmidt(A)
format
[~,n]=size(A);
U=[];
V=[];
tolerance=[];

if rank(A) ~= size(A)
    disp('columns of A are linear dependent and cannot be orthogonalized')
    return;
else
    U=A;
    V = zeros(n,n);

    for i=1:n
        B=A(:,i);
        for j=1:i-1
            V(j,i)=U(:,j)'*A(:,i);
            B=B-V(j,i)*U(:,j);
        end
        V(i,i)=norm(B);
        U(:,i)=B/V(i,i);
    end

    U(isnan(U)) = 1;
    V(isnan(V)) = 1;

    problem = false;
    if rank(A) ~= rank(U)
        disp('U is not a basis for Col A? Check the code!')
        problem = true;
    end

    if closetozeroroundoff(U'*U-eye(n),0) ~= 0
       disp('no orthonormalization? Check the code!')
       problem = true;
    end

    if problem == true
        U=[];
        return;
    end

    tolerance=norm(U'*U-eye(n),1);
    V=U'*A;

end
```

```
%(a)
I=eye(3);M=magic(3);I(:,3)=M(:,1);
A=I
```

```
A = 3×3
     1     0     8
     0     1     3
     0     0     4
```

```
[U,V,tolerance]=qrschmidt(A)
```

```
U = 3×3
     1     0     0
     0     1     0
     0     0     1
V = 3×3
     1     0     8
     0     1     3
     0     0     4
tolerance = 0
```

```
%(b)
A=[1 2 -1 4 1;0 1 3 -2 2;0 0 0 2 -2]
```

```
A = 3×5
     1     2    -1     4     1
     0     1     3    -2     2
     0     0     0     2    -2
```

```
[U,V,tolerance]=qrschmidt(A)
```

```
U = 3×5
     1     0     1     1     1
     0     1     1     1     1
     0     0     1     1     1
V = 5×5
     1     2    -1     4     1
     0     1     3    -2     2
     1     3     2     4     1
     1     3     2     4     1
     1     3     2     4     1
tolerance = 10
```

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

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

```
[U,V,tolerance]=qrschmidt(A)
```

```
columns of A are linear dependent and cannot be orthogonalized

U =

     []


V =

     []
```

```
tolerance =

     []
```

```
%(d)
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
```

```
[U,V,tolerance]=qrschmidt(A)
```

```
U = 5×5
    0.5234    0.5058   -0.6735    0.1215    0.0441
    0.7081   -0.6966    0.0177   -0.0815    0.0800
    0.1231    0.1367    0.3558    0.6307    0.6646
    0.3079    0.1911    0.4122    0.4247   -0.7200
    0.3387    0.4514    0.4996   -0.6328    0.1774
V = 5×5
   32.4808   26.6311   21.3973   23.7063   25.8615
         0   19.8943   12.3234    1.9439    4.0856
   -0.0000   -0.0000   24.3985   11.6316    3.7415
   -0.0000   -0.0000         0   20.0982    9.9739
   -0.0000   -0.0000   -0.0000   -0.0000   16.0005
tolerance = 6.1756e-16
```

```
%(e)
A=magic(4);A=orth(A)
```

```
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
```

```
[U,V,tolerance]=qrschmidt(A)
```

```
U = 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
V = 3×3
    1.0000   -0.0000   -0.0000
    0.0000    1.0000    0.0000
   -0.0000         0    1.0000
tolerance = 1.1102e-16
```

```
%(f)
A=randi(10,6,4)
```

```
A = 6×4
     5     4     9     3
     7     5     2     5
     1     1    10     5
     9     2     6     8
     6     7     8     9
     9     4    10     2
```

```
[U,V,tolerance]=qrschmidt(A)
```

```
U = 6×4
    0.3026    0.2368    0.3037   -0.3164
    0.4237    0.2147   -0.4951    0.0302
    0.0605    0.0863    0.7896    0.3345
    0.5447   -0.5863   -0.0426    0.5338
    0.3631    0.7126   -0.0297    0.3201
    0.5447   -0.1969    0.1910   -0.6322
V = 4×4
   16.5227    9.1995   15.7964   12.0440
   -0.0000    5.1352    3.6379    3.5445
   -0.0000   -0.0000   11.0561    2.1574
    0.0000    0.0000    0.0000    6.7619
tolerance = 6.3838e-16
```

```
%(g)
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
```

```
[U,V,tolerance]=qrschmidt(A)
```

```
U = 5×5
    0.8266   -0.5334    0.1753   -0.0391    0.0055
    0.4133    0.3741   -0.7173    0.4033   -0.1101
    0.2755    0.4629   -0.0577   -0.6790    0.4954
    0.2066    0.4433    0.3526   -0.2062   -0.7707
    0.1653    0.4059    0.5720    0.5764    0.3853
V = 5×5
    1.2098    0.6888    0.4920    0.3854    0.3178
   -0.0000    0.1301    0.1402    0.1327    0.1223
    0.0000    0.0000    0.0081    0.0126    0.0149
   -0.0000   -0.0000   -0.0000    0.0003    0.0007
   -0.0000   -0.0000    0.0000    0.0000    0.0000
tolerance = 1.0536e-07
```

```
%(i)
A=pascal(5)
```

```
A = 5×5
     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15
     1     4    10    20    35
     1     5    15    35    70
```

```
[U,V,tolerance]=qrschmidt(A)
```

```
U = 5×5
    0.4472   -0.6325    0.5345   -0.3162    0.1195
    0.4472   -0.3162   -0.2673    0.6325   -0.4781
    0.4472         0   -0.5345   -0.0000    0.7171
    0.4472    0.3162   -0.2673   -0.6325   -0.4781
    0.4472    0.6325    0.5345    0.3162    0.1195
V = 5×5
    2.2361    6.7082   15.6525   31.3050   56.3489
```

```
    0.0000    3.1623   11.0680   26.5631   53.1263
         0   -0.0000    1.8708    7.4833   19.2428
    0.0000    0.0000    0.0000    0.6325    2.8460
   -0.0000   -0.0000   -0.0000   -0.0000    0.1195
tolerance = 9.1353e-12
```

Exercise 3

Part 1

```
type qrgivens.m
```

```
function [q,r] = qrgivens(A)
format
[m,n]=size(A);
r=A;
q=eye(m);
k=min(m,n);

for i=1:k       % iterate through columns
    j=m;        % j starts at bottom
    while j > i
        if r(j,i) ~= 0      % checks if entry is NOT 0
            b = r(j,i);     % row entry
            a = r(i,i);     % main diagonal entry
            G = givensrot(m,i,j,a,b);
            q=q*G;
            r = G' * r;
        end
        j = j - 1;          % j goes to row above
    end
end
r=closetozeroroundoff(r,7);
% these tests only go off when code failed
test=0;
if  ~istriu(r)
    disp('r is not upper-triangular?!')
    test=1;
end

if any(closetozeroroundoff(q'*q-eye(m),7),"all")
    disp('q is not orthogonal?!')
    test=1;
end

if any(closetozeroroundoff(A-q*r,7),"all")
    disp('QR factorization is not working?!')
    test=1;
end

if test
    r=[];
    q=[];
end
```

```
type closetozeroroundoff.m
```

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

```
%(a)
A=ones(2)
```

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

```
[q,r] = qrgivens(A)
```

```
q = 2×2
    0.7071   -0.7071
    0.7071    0.7071
r = 2×2
    1.4142    1.4142
         0         0
```

```
%(b)
A=magic(3)
```

```
A = 3×3
     8     1     6
     3     5     7
     4     9     2
```

```
[q,r] = qrgivens(A)
```

```
q = 3×3
    0.8480   -0.5223    0.0901
    0.3180    0.3655   -0.8748
    0.4240    0.7705    0.4760
r = 3×3
    9.4340    6.2540    8.1620
         0    8.2394    0.9655
         0         0   -4.6314
```

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

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

```
[q,r] = qrgivens(A)
```

```
q = 4×4
    0.8230   -0.4186    0.3123    0.2236
    0.2572    0.5155   -0.4671    0.6708
    0.4629    0.1305   -0.5645   -0.6708
    0.2057    0.7363    0.6046   -0.2236
r = 4×4
   19.4422   10.5955   10.9041   18.5164
         0   16.0541   15.7259    0.9848
         0         0    1.9486   -5.8458
         0         0         0         0
```

```
%(d)
A=[magic(3),ones(3,2)]
```

```
A = 3×5
     8     1     6     1     1
     3     5     7     1     1
     4     9     2     1     1
```

```
[q,r] = qrgivens(A)
```

```
q = 3×3
```

```
    0.8480   -0.5223    0.0901
    0.3180    0.3655   -0.8748
    0.4240    0.7705    0.4760
r = 3×5
    9.4340    6.2540    8.1620    1.5900    1.5900
         0    8.2394    0.9655    0.6137    0.6137
         0         0   -4.6314   -0.3088   -0.3088
```

```
%(e)
A=[magic(3);ones(2,3)]
```

```
A = 5×3
     8     1     6
     3     5     7
     4     9     2
     1     1     1
     1     1     1
```

```
[q,r] = qrgivens(A)
```

```
q = 5×5
    0.8386   -0.5286   -0.0928   -0.0614    0.0699
    0.3145    0.3622    0.8725   -0.0614    0.0699
    0.4193    0.7656   -0.4789   -0.0614    0.0699
    0.1048    0.0399    0.0201    0.9918   -0.0575
    0.1048    0.0399    0.0201   -0.0705   -0.9910
r = 5×3
    9.5394    6.3945    8.2815
         0    8.2529    0.9747
         0         0    4.6333
         0         0         0
         0         0         0
```

```
%(f)
A=triu(magic(5))
```

```
A = 5×5
    17    24     1     8    15
     0     5     7    14    16
     0     0    13    20    22
     0     0     0    21     3
     0     0     0     0     9
```

```
[q,r] = qrgivens(A)
```

```
q = 5×5
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1
r = 5×5
    17    24     1     8    15
     0     5     7    14    16
     0     0    13    20    22
     0     0     0    21     3
     0     0     0     0     9
```

```
%(g)
A=[magic(3);hilb(3)]
```

```
A = 6×3
```

```
    8.0000    1.0000    6.0000
    3.0000    5.0000    7.0000
    4.0000    9.0000    2.0000
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
```

```
[q,r] = qrgivens(A)
```

```
q = 6×6
    0.8416   -0.5206   -0.0763   -0.1073   -0.0495    0.0313
    0.3156    0.3660    0.8725    0.0691    0.0185   -0.0056
    0.4208    0.7711   -0.4683   -0.0800   -0.0422    0.0282
    0.1052   -0.0196   -0.1084    0.9882    0.0135   -0.0097
    0.0526    0.0003   -0.0384   -0.0235    0.9976   -0.0067
    0.0351    0.0036   -0.0192   -0.0154   -0.0097   -0.9990
r = 6×3
    9.5058    6.2856    8.1555
         0    8.2410    0.9753
         0         0    4.6637
         0         0         0
         0         0         0
         0         0         0
```

```
%(h)
A=[1 1 2 0;0 0 1 3;0 0 2 4;0 0 3 5]
```

```
A = 4×4
     1     1     2     0
     0     0     1     3
     0     0     2     4
     0     0     3     5
```

```
[q,r] = qrgivens(A)
```

```
q = 4×4
    1.0000         0         0         0
         0    1.0000         0         0
         0         0    0.5547   -0.8321
         0         0    0.8321    0.5547
r = 4×4
    1.0000    1.0000    2.0000         0
         0         0    1.0000    3.0000
         0         0    3.6056    6.3791
         0         0         0   -0.5547
```

```
%(i)
A=pascal(4)
```

```
A = 4×4
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
```

```
[q,r] = qrgivens(A)
```

```
q = 4×4
    0.5000   -0.6708    0.5000   -0.2236
    0.5000   -0.2236   -0.5000    0.6708
    0.5000    0.2236   -0.5000   -0.6708
    0.5000    0.6708    0.5000    0.2236
```

```
r = 4×4
    2.0000    5.0000   10.0000   17.5000
         0    2.2361    6.7082   14.0872
         0         0    1.0000    3.5000
         0         0         0    0.2236
```

Part 2

```
type hreflections.m
```

```
function [q,r] = hreflections(A)
format
[m,n]=size(A);
q=eye(m);
Q=eye(m);
r=A;
k=min(m,n);
for i=1:k
    R=r(i:end,i:end);
    if ~any(R(:,1))
        continue
    else
        if i == 1
            I = eye(m);
        else
            I = eye((m-i)+1);
        end
        colnorm=norm(R(:,1)); % colnorm is like ||x||
        u=R(:,1)-colnorm*I(:,1); % x-||x||e1
        u=closetozeroroundoff(u,7);
        if ~any(u)
            continue
        else
            v = u/norm(u);
            Q = I-2*(v*v');
            %embed Q into mxm indentiy matrix as described in Algorithm
            I=eye(m);
            I(padarray(true(size(Q)),size(I)-size(Q),'pre')) = Q;
            Q = I;
            %Q=closetozeroroundoff(Q,7);
        end
        r=Q*r; % running into problems here
        % 1st column of r has entry |x| on top and everything below is 0
        q=q*Q;
        r=closetozeroroundoff(r,7);
    end
end

test=0;
if ~istriu(r)
    disp('r is not upper-triangular?!')
    test=1;
end

if any(closetozeroroundoff(q'*q-eye(m),7),"all")
    disp('q is not orthogonal?!')
    test=1;
end

if any(closetozeroroundoff(A-q*r,7),"all")
    disp('QR factorization is not working?!')
    test=1;
end
```

```
if test
    q=[];
    r=[];
end
```

```
%(a)
A=zeros(2,4)
```

```
A = 2×4
     0     0     0     0
     0     0     0     0
```

```
[q,r] = hreflections(A)
```

```
q = 2×2
     1     0
     0     1
r = 2×4
     0     0     0     0
     0     0     0     0
```

```
%(b)
A=ones(2)
```

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

```
[q,r] = hreflections(A)
```

```
q = 2×2
    0.7071    0.7071
    0.7071   -0.7071
r = 2×2
    1.4142    1.4142
         0         0
```

```
%(c)
A=zeros(4);A(2,[2 4])=ones(2,1)
```

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

```
[q,r] = hreflections(A)
```

```
q = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
r = 4×4
     0     0     0     0
     0     1     0     1
     0     0     0     0
     0     0     0     0
```

```
%(d)
A=pascal(4)
```

```
A = 4×4
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
```

```
[q,r] = hreflections(A)
```

```
q = 4×4
    0.5000   -0.6708    0.5000   -0.2236
    0.5000   -0.2236   -0.5000    0.6708
    0.5000    0.2236   -0.5000   -0.6708
    0.5000    0.6708    0.5000    0.2236
r = 4×4
    2.0000    5.0000   10.0000   17.5000
         0    2.2361    6.7082   14.0872
         0         0    1.0000    3.5000
         0         0         0    0.2236
```

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

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

```
[q,r] = hreflections(A)
```

```
q = 4×4
    0.8230   -0.4186    0.3123   -0.2236
    0.2572    0.5155   -0.4671   -0.6708
    0.4629    0.1305   -0.5645    0.6708
    0.2057    0.7363    0.6046    0.2236
r = 4×4
   19.4422   10.5955   10.9041   18.5164
         0   16.0541   15.7259    0.9848
         0         0    1.9486   -5.8458
         0         0         0         0
```

```
%(f)
A=[magic(3),ones(3,2)]
```

```
A = 3×5
     8     1     6     1     1
     3     5     7     1     1
     4     9     2     1     1
```

```
[q,r] = hreflections(A)
```

```
q = 3×3
    0.8480   -0.5223   -0.0901
    0.3180    0.3655    0.8748
    0.4240    0.7705   -0.4760
r = 3×5
    9.4340    6.2540    8.1620    1.5900    1.5900
         0    8.2394    0.9655    0.6137    0.6137
         0         0    4.6314    0.3088    0.3088
```

```
%(g)
A=[magic(3);ones(2,3)]
```

```
A = 5×3
     8     1     6
     3     5     7
     4     9     2
     1     1     1
     1     1     1
```

```
[q,r] = hreflections(A)
```

```
q = 5×5
    0.8386   -0.5286   -0.0928   -0.0658   -0.0658
    0.3145    0.3622    0.8725   -0.0658   -0.0658
    0.4193    0.7656   -0.4789   -0.0658   -0.0658
    0.1048    0.0399    0.0201    0.9935   -0.0065
    0.1048    0.0399    0.0201   -0.0065    0.9935
r = 5×3
    9.5394    6.3945    8.2815
         0    8.2529    0.9747
         0         0    4.6333
         0         0         0
         0         0         0
```

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

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

```
[q,r] = hreflections(A)
```

```
q = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
r = 4×4
    16     2     3    13
     0    11    10     8
     0     0     6    12
     0     0     0     1
```

```
%(i)
A=[magic(3);hilb(3)]
```

```
A = 6×3
    8.0000    1.0000    6.0000
    3.0000    5.0000    7.0000
    4.0000    9.0000    2.0000
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
```

```
[q,r] = hreflections(A)
```

```
q = 6×6
    0.8416   -0.5206   -0.0763   -0.1088   -0.0474   -0.0291
    0.3156    0.3660    0.8725    0.0695    0.0170    0.0043
    0.4208    0.7711   -0.4683   -0.0814   -0.0407   -0.0265
    0.1052   -0.0196   -0.1084    0.9883   -0.0081   -0.0062
```

```
    0.0526    0.0003   -0.0384   -0.0017    0.9979   -0.0019
    0.0351    0.0036   -0.0192    0.0005   -0.0007    0.9992
r = 6×3
    9.5058    6.2856    8.1555
         0    8.2410    0.9753
         0         0    4.6637
         0         0         0
         0         0         0
         0         0         0
```

```
%(j)
A=[1 1 2 0;0 0 1 3;0 0 2 4;0 0 3 5]
```

```
A = 4×4
     1     1     2     0
     0     0     1     3
     0     0     2     4
     0     0     3     5
```

```
[q,r] = hreflections(A)
```

```
q = 4×4
    1.0000         0         0         0
         0    1.0000         0         0
         0         0    0.5547    0.8321
         0         0    0.8321   -0.5547
r = 4×4
    1.0000    1.0000    2.0000         0
         0         0    1.0000    3.0000
         0         0    3.6056    6.3791
         0         0         0    0.5547
```

Part 3

```
type qrschmidt.m
```

```
function [U,V,tolerance]=qrschmidt(A)
format
[~,n]=size(A);
U=[];
V=[];
tolerance=[];

if rank(A) ~= size(A)
    disp('columns of A are linear dependent and cannot be orthogonalized')
    return;
else
    U=A;
    V = zeros(n,n);

    for i=1:n
        B=A(:,i);
        for j=1:i-1
            V(j,i)=U(:,j)'*A(:,i);
            B=B-V(j,i)*U(:,j);
        end
        V(i,i)=norm(B);
        U(:,i)=B/V(i,i);
    end

    U(isnan(U)) = 1;
    V(isnan(V)) = 1;
```

```
    problem = false;
    if rank(A) ~= rank(U)
        disp('U is not a basis for Col A? Check the code!')
        problem = true;
    end

    if closetozeroroundoff(U'*U-eye(n),0) ~= 0
        disp('no orthonormalization? Check the code!')
        problem = true;
    end

    if problem == true
        U=[];
        return;
    end

    tolerance=norm(U'*U-eye(n),1);
    V=U'*A;
end
```

```
type hbasis.m
```

```
function [Q,R] = hbasis(A)
[m,n]=size(A);
rankA=rank(A);
[Q,R] = hreflections(A);
if m > n
    tempQ = Q;
    tempR = R;

    Q = tempQ(:,1:n);
    R = tempR(1:n,:);
end

if any(closetozeroroundoff(A-Q*R,7),"all")
    disp('an economy-size factorization is not working?')
    Q=[];
    R=[];
end

if rankA ~= n %if rankA==n continue code
    return
end

[U,V]=qrschmidt(A);
mm = 0;
if closetozeroroundoff(U-Q,7)~= 0
    disp('the basis Q does not match the Gram-Schmidth basis U')
    mm = mm+1;
end
if closetozeroroundoff(V-R,7)~= 0
    disp('R does not match upper-triangular V from the Gram-Schmidt process')
    mm = mm+1;
end
if mm >= 1
    disp('Check the code!')
    Q=[];
    R=[];
end
```

```
type hreflections.m
```

```
function [q,r] = hreflections(A)
format
[m,n]=size(A);
q=eye(m);
Q=eye(m);
r=A;
k=min(m,n);
for i=1:k
    R=r(i:end,i:end);
    if ~any(R(:,1))
        continue
    else
        if i == 1
            I = eye(m);
        else
            I = eye((m-i)+1);
        end
        colnorm=norm(R(:,1)); % colnorm is like ||x||
        u=R(:,1)-colnorm*I(:,1); % x-||x||e1
        u=closetozeroroundoff(u,7);
        if ~any(u)
            continue
        else
            v = u/norm(u);
            Q = I-2*(v*v');
            %embed Q into mxm indentiy matrix as described in Algorithm
            I=eye(m);
            I(padarray(true(size(Q)),size(I)-size(Q),'pre')) = Q;
            Q = I;
            %Q=closetozeroroundoff(Q,7);
        end
        r=Q*r; % running into problems here
        % 1st column of r has entry |x| on top and everything below is 0
        q=q*Q;
        r=closetozeroroundoff(r,7);
    end
end

test=0;
if ~istriu(r)
    disp('r is not upper-triangular?!')
    test=1;
end

if any(closetozeroroundoff(q'*q-eye(m),7),"all")
    disp('q is not orthogonal?!')
    test=1;
end

if any(closetozeroroundoff(A-q*r,7),"all")
    disp('QR factorization is not working?!')
    test=1;
end

if test
    q=[];
    r=[];
end
```

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

A = *4×4*

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

```
[Q,R] = hbasis(A)
```

```
Q = 4×4
    0.8230   -0.4186    0.3123   -0.2236
    0.2572    0.5155   -0.4671   -0.6708
    0.4629    0.1305   -0.5645    0.6708
    0.2057    0.7363    0.6046    0.2236
R = 4×4
   19.4422   10.5955   10.9041   18.5164
         0   16.0541   15.7259    0.9848
         0         0    1.9486   -5.8458
         0         0         0         0
```

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

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

```
[Q,R] = hbasis(A)
```

```
Q = 3×3
    0.3333    0.6667    0.6667
    0.6667    0.3333   -0.6667
    0.6667   -0.6667    0.3333
R = 3×3
    3.0000    6.0000    4.3333
         0         0    1.6667
         0         0    0.6667
```

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

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

```
[Q,R] = hbasis(A)
```

```
Q = 3×3
    0.3333    0.6667    0.6667
    0.6667    0.3333   -0.6667
    0.6667   -0.6667    0.3333
R = 3×3
    3.0000    6.0000    9.0000
         0         0         0
         0         0         0
```

```
%(d)
A=rand(3,5)
```

```
A = 3×5
    0.8147    0.9134    0.2785    0.9649    0.9572
```

```
    0.9058    0.6324    0.5469    0.1576    0.4854
    0.1270    0.0975    0.9575    0.9706    0.8003
```

```
[Q,R] = hbasis(A)
```

```
Q = 3×3
    0.6651    0.7463   -0.0256
    0.7395   -0.6631   -0.1162
    0.1037   -0.0583    0.9929
R = 3×5
    1.2249    1.0853    0.6889    0.8590    1.0785
         0    0.2566   -0.2106    0.5590    0.3458
         0         0    0.8800    0.9207    0.7137
```

```
%(e)
A=randi([-6 6],5,3)
```

```
A = 5×3
    -5     2     3
    -1    -6     3
     5     5    -1
     4     6     2
     6     2    -4
```

```
[Q,R] = hbasis(A)
```

```
Q = 5×3
   -0.4927    0.5562    0.1141
   -0.0985   -0.6355    0.6759
    0.4927    0.2605    0.1312
    0.3941    0.4418    0.6374
    0.5912   -0.1541   -0.3265
R = 3×3
   10.1489    5.6164   -3.8428
         0    8.5707    1.0014
         0         0    4.8198
```

```
%(f)
A=[magic(4),pascal(4)]
```

```
A = 4×8
    16     2     3    13     1     1     1     1
     5    11    10     8     1     2     3     4
     9     7     6    12     1     3     6    10
     4    14    15     1     1     4    10    20
```

```
[Q,R] = hbasis(A)
```

```
Q = 4×4
    0.8230   -0.4186    0.3123   -0.2236
    0.2572    0.5155   -0.4671   -0.6708
    0.4629    0.1305   -0.5645    0.6708
    0.2057    0.7363    0.6046    0.2236
R = 4×8
   19.4422   10.5955   10.9041   18.5164    1.7488    3.5490    6.4293   10.5955
         0   16.0541   15.7259    0.9848    0.9637    3.9489    9.2735   17.6737
         0         0    1.9486   -5.8458   -0.1146    0.1032    1.5703    4.8915
         0         0         0         0         0    1.3416    4.0249    8.2735
```

```
%(g)
A=magic(3)
```

```
A = 3×3
     8     1     6
     3     5     7
     4     9     2
```

```
[Q,R] = hbasis(A)
```

```
Q = 3×3
    0.8480   -0.5223   -0.0901
    0.3180    0.3655    0.8748
    0.4240    0.7705   -0.4760
R = 3×3
    9.4340    6.2540    8.1620
         0    8.2394    0.9655
         0         0    4.6314
```

```
%(h)
A=[magic(3);hilb(3)]
```

```
A = 6×3
    8.0000    1.0000    6.0000
    3.0000    5.0000    7.0000
    4.0000    9.0000    2.0000
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
```

```
[Q,R] = hbasis(A)
```

```
Q = 6×3
    0.8416   -0.5206   -0.0763
    0.3156    0.3660    0.8725
    0.4208    0.7711   -0.4683
    0.1052   -0.0196   -0.1084
    0.0526    0.0003   -0.0384
    0.0351    0.0036   -0.0192
R = 3×3
    9.5058    6.2856    8.1555
         0    8.2410    0.9753
         0         0    4.6637
```

Exercise 4

```
type closetozeroroundoff.m
```

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

```
type shrink.m
```

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

```
type proj.m
```

```
function [p,z]=proj(A,b)
format
[~,A]=shrink(A);
m=size(A,1); % m rows and column vector b
p=[];
z=[];

if m ~= size(b,1)
    disp('No solution: sizes of A and b disagree')
    return % terminates function
end

% Check if b is in Col A
if size(b,1) == rank(A)
    disp('b is in Col A')
    p = b
    z = zeros(m,1)
    return
end

% Check if b is orthogonal to Col A
if closetozeroroundoff(A'*b,7) == 0
    disp('b is orthogonal to Col A')
    p = zeros(m,1)
    z = b
else
    [Q,~] = hbasis(A);
    p = Q*Q'*b;
    z = b - p;
    % verify that p and z were computed correctly
    x = (A'*A)\(A'*b);
    p1 = A*x;

    if closetozeroroundoff(p1-p,7) == 0
        disp('the projection of b onto Col A is')
        p
    else % Something is wrong in code
        disp('Oops! p is not a projection!?')
        p = [];
    end

    if closetozeroroundoff(A'*z,7)== 0
        disp('the component of b orthogonal to Col A is')
        z
    else
```

```
        disp('What?! z is not orthogonal to Col A!')
        z = [];
    end
end

if ~isempty(p) && ~isempty(z)
    d = norm(z);
    fprintf('the distance from b to Col A is %i',d)
end
```

```
type hbasis
```

```
function [Q,R] = hbasis(A)
[m,n]=size(A);
rankA=rank(A);
[Q,R] = hreflections(A);
if m > n
    tempQ = Q;
    tempR = R;

    Q = tempQ(:,1:n);
    R = tempR(1:n,:);
end

if any(closetozeroroundoff(A-Q*R,7),"all")
    disp('an economy-size factorization is not working?')
    Q=[];
    R=[];
end

if rankA ~= n %if rankA==n continue code
    return
end

[U,V]=qrschmidt(A);
mm = 0;
if closetozeroroundoff(U-Q,7)~= 0
    disp('the basis Q does not match the Gram-Schmidth basis U')
    mm = mm+1;
end
if closetozeroroundoff(V-R,7)~= 0
    disp('R does not match upper-triangular V from the Gram-Schmidt process')
    mm = mm+1;
end
if mm >= 1
    disp('Check the code!')
    Q=[];
    R=[];
end
```

```
%(a)
A=[1 2 3;2 4 3;2 4 2], b=ones(3,1)
```

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

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

```
the projection of b onto Col A is
p = 3×1
    0.9310
    1.1379
    0.8966
the component of b orthogonal to Col A is
z = 3×1
    0.0690
   -0.1379
    0.1034
the distance from b to Col A is 1.856953e-01
```

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

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

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

```
the projection of b onto Col A is
p = 3×1
    0.5556
    1.1111
    1.1111
the component of b orthogonal to Col A is
z = 3×1
    0.4444
   -0.1111
   -0.1111
the distance from b to Col A is 4.714045e-01
```

```
%(c)
A=magic(4), b=A(:,4)
```

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

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

```
the projection of b onto Col A is
p = 4×1
   13.0000
    8.0000
   12.0000
    1.0000
the component of b orthogonal to Col A is
z = 4×1
```

```
10^-13 ×
   -0.0533
   -0.0533
    0.1066
    0.0244
the distance from b to Col A is 1.328005e-14
```

```
%(d)
A=magic(5), b=(1:4)'
```

```
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 = 4×1
     1
     2
     3
     4
```

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

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

```
%(e)
A=magic(4), b=randi(10,4,1)
```

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

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

```
the projection of b onto Col A is
p = 4×1
    2.3500
    3.0500
    2.9500
    2.6500
the component of b orthogonal to Col A is
z = 4×1
    0.6500
    1.9500
   -1.9500
   -0.6500
the distance from b to Col A is 2.906888e+00
```

```
%(f)
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
    0.9421
    0.9561
    0.5752
    0.0598
    0.2348
```

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

```
b is in Col A
p = 5×1
    0.9421
    0.9561
    0.5752
    0.0598
    0.2348
z = 5×1
     0
     0
     0
     0
     0
```

```
%(g)
A=rand(4,3), b=ones(4,1)
```

```
A = 4×3
    0.3532    0.1690    0.4509
    0.8212    0.6491    0.5470
    0.0154    0.7317    0.2963
    0.0430    0.6477    0.7447
b = 4×1
     1
     1
     1
     1
```

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

```
the projection of b onto Col A is
p = 4×1
    0.6454
    1.1464
    0.7795
    1.1949
the component of b orthogonal to Col A is
z = 4×1
    0.3546
   -0.1464
    0.2205
   -0.1949
the distance from b to Col A is 4.835224e-01
```

```
%(h)
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 projection of b onto Col A is
p = 4×1
    0.8000
    0.6000
    0.4000
    0.2000
the component of b orthogonal to Col A is
z = 4×1
    0.2000
   -0.6000
    0.6000
   -0.2000
the distance from b to Col A is 8.944272e-01
```

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

```
B = 4×4
      1     5     9    13
      2     6    10    14
      3     7    11    15
      4     8    12    16
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
p = 4×1
      0
      0
      0
      0
z = 4×1
      1
      1
      1
      1
the distance from b to Col A is 2
```

Exercise 5

```
type solveall.m
```

```
function x=solveall(A,b)
format
[m,n]=size(A);
x=zeros(n,1);
consist=0;
uniq=0;
if (m==n & rank(A)==m & rank(A)==n)
    consist=1;
    disp('the matrix is invertible - there is a unique "exact" solution')
    [invA,~] = eluinv(A);
    x = invA*b;
else
    if (rank([A,b]) == rank(A))
        consist=1;
        disp('the system is consistent - look for "exact" solution')
    else
        disp('the system is inconsistent - look for least-squares solution')
    end

    if(rank(A)==n)
        uniq=1;
        disp('the system has a unique solution')
    else
        disp('the system has infinitely many solutions')
    end

    if (uniq == 0)
        [pivot_c,B]=shrink(A);
    else
        B=A;
    end
    [Q,R] = hbasis(B);
    aug=[R Q'*b];
    aug=rref(aug);
    y=aug(:,end);
    if (uniq == 0)
        x(pivot_c,1)=y;
    else
        x=y;
    end
end
if consist == 1 & ~any(closetozeroroundoff(A*x-b,7))
    disp('an "exact" solution of the system is')
    disp(x)
elseif consist == 0 & ~any(closetozeroroundoff((A'*A)*x-A'*b,7))
    disp('a least-squares solution is')
    disp(x)
else
    disp('Check code for inconsistent system')
    x=[];
end
```

```
type shrink.m
```

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

```
type closetozeroroundoff.m
```

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

```
type eluinv.m
```

```
function [invA,detA] = eluinv(A)
[~,n]=size(A);
if rank(A) < n
 fprintf('A is not invertible')
 invA=[];
 detA=0;
 return
else
 %Use L and U outputs from the elu function to compute
 %inverse matrices for each respective variable.
 [L,U,N]=elu(A);
 invL = rref([L eye(n)]);
 invL = invL(1:n,n+1:2*n);
 invU = rref([U eye(n)]);
 invU = invU(1:n,n+1:2*n);

 %Use invL and invU to compute invA
 invA = invU*invL;
 F = inv(A);

 %Check validity of invA computation
 if closetozeroroundoff(invA-F,7) ~= 0
 invA = [];
 end

 %Compute detA using U and N
 detA = (-1)^N*prod(diag(U));
 d = det(A);

 %Check validity of detA computation
 if closetozeroroundoff(detA-d,7) ~= 0
 detA = [];
 end
end
```

```
type hbasis.m
```

```
function [Q,R] = hbasis(A)
[m,n]=size(A);
rankA=rank(A);
[Q,R] = hreflections(A);
if m > n
tempQ = Q;
tempR = R;
Q = tempQ(:,1:n);
R = tempR(1:n,:);
end
if any(closetozeroroundoff(A-Q*R,7),"all")
disp('an economy-size factorization is not working?')
Q=[];
R=[];
end
if rankA ~= n %if rankA==n continue code
return
```

```
end
[U,V]=qrschmidt(A);
mm = 0;
if closetozeroroundoff(U-Q,7) ~= 0
disp('the basis Q does not match the Gram-Schmidth basis U')
mm = mm+1;
end
if closetozeroroundoff(V-R,7) ~= 0
disp('R does not match upper-triangular V from the Gram-Schmidt process')
mm = mm+1;
end
if mm >= 1
disp('Check the code!')
Q=[];
R=[];
end
```

```
%(a)
A=magic(5), H=60*hilb(5); b=H(:,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
    60
    30
    20
    15
    12
```

```
x=solveall(A,b);
```

```
the matrix is invertible - there is a unique "exact" solution
an "exact" solution of the system is
    0.5888
    1.5850
   -1.1785
    0.5081
    0.6042
```

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

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

```
x=solveall(A,b);
```

```
the system is consistent - look for "exact" solution
the system has infinitely many solutions
```

```
an "exact" solution of the system is
    0.0588
    0.1176
   -0.0588
         0
```

```
%(c)
A=magic(4), b=rand(4,1)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
b = 4×1
    0.7441
    0.5000
    0.4799
    0.9047
```

```
x=solveall(A,b);
```

```
the system is inconsistent - look for least-squares solution
the system has infinitely many solutions
a least-squares solution is
    0.0324
   -0.0962
    0.1411
         0
```

```
%(d)
A=[1 2 3;2 4 6;2 4 6], b=ones(3,1)
```

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

```
x=solveall(A,b);
```

```
the system is inconsistent - look for least-squares solution
the system has infinitely many solutions
a least-squares solution is
    0.5556
         0
         0
```

```
%(e)
A=[magic(5);zeros(1,5)], b=rand(6,1)
```

```
A = 6×5
    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
     0     0     0     0     0
b = 6×1
```

```
    0.6099
    0.6177
    0.8594
    0.8055
    0.5767
    0.1829
```

```
x=solveall(A,b);
```

```
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
    0.0011
    0.0104
    0.0096
    0.0217
    0.0106
```

```
%(f)
A=[pascal(5);randi([-5 5],2,5)], b=sum(A,2)
```

```
A = 7×5
     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15
     1     4    10    20    35
     1     5    15    35    70
    -3    -5    -4     2     0
     4     0     5     0    -5
b = 7×1
     5
    15
    35
    70
   126
   -10
     4
```

```
x=solveall(A,b);
```

```
the system is consistent - look for "exact" solution
the system has a unique solution
an "exact" solution of the system is
    1.0000
    1.0000
    1.0000
    1.0000
    1.0000
```

```
%(g)
A=[magic(3),ones(3,2)], b=sum(A,2)
```

```
A = 3×5
     8     1     6     1     1
     3     5     7     1     1
     4     9     2     1     1
b = 3×1
    17
    17
    17
```

```
x=solveall(A,b);
```

```
the system is consistent - look for "exact" solution
the system has infinitely many solutions
an "exact" solution of the system is
    1.1333
    1.1333
    1.1333
         0
         0
```

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

```
A = 3×3
     1     2     3
     2     4     3
     2     5     2
b = 3×1
     1
     1
     1
```

```
x=solveall(A,b);
```

```
the matrix is invertible - there is a unique "exact" solution
an "exact" solution of the system is
   -0.6667
    0.3333
    0.3333
```

Exercise 6

```
type polyplot.m
```

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

```
type solveall
```

```
function x=solveall(A,b)
format
[m,n]=size(A);
x=zeros(n,1);
consist=0;
uniq=0;
if (m==n & rank(A)==m & rank(A)==n)
    consist=1;
    disp('the matrix is invertible - there is a unique "exact" solution')
    [invA,~] = eluinv(A);
    x = invA*b;
else
    if (rank([A,b]) == rank(A))
        consist=1;
        disp('the system is consistent - look for "exact" solution')
    else
        disp('the system is inconsistent - look for least-squares solution')
    end

    if(rank(A)==n)
        uniq=1;
        disp('the system has a unique solution')
    else
        disp('the system has infinitely many solutions')
    end

    if (uniq == 0)
        [pivot_c,B]=shrink(A);
    else
        B=A;
    end
    [Q,R] = hbasis(B);
    aug=[R Q'*b];
    aug=rref(aug);
    y=aug(:,end);
    if (uniq == 0)
        x(pivot_c,1)=y;
    else
        x=y;
    end
end
if consist == 1 && ~any(closetozeroroundoff(A*x-b,7))
    disp('an "exact" solution of the system is')
    disp(x)
elseif consist == 0 && ~any(closetozeroroundoff((A'*A)*x-A'*b,7))
    disp('a least-squares solution is')
    disp(x)
else
    disp('Check code for inconsistent system')
    x=[];
end
```

```
type lstsqpoly.m
```

```
function [c,X,N]=lstsqpoly(x,y,n)
format
m=length(x);
c=[];
N=[];
a=x(1)
b=x(m)

X=zeros(m,n+1);
x
for i = 1:m
    j = n:-1:0;
    %x(i)
    X(i,:) = x(i).^(j);
    %X(i,:)
end

disp('the design matrix is')
X

c=solveall(X,y);

c1=X\y;
if any(closetozeroroundoff(c-c1,7))
    disp('Check the code!')
    c=[];
    return
else
    %disp('code is good')
end

N=norm(y-X*c);
fprintf('the 2-norm of the residual vector e is %i\n',N)
plot(x,y,'*'),hold on
polyplot(a,b,c')
fprintf('the polynomial of degree %i of the best least-squares fit is\n',n)
P=vpa(poly2sym(c),4)

hold off

end
```

```
%(a)
x = [1;2;3;4;5;6;8], y = [1;2;3;5;6;7;4]
```

```
x = 7×1
     1
     2
     3
     4
     5
     6
     8
y = 7×1
     1
     2
     3
     5
     6
     7
```

```
    4
```

```
n=1
```

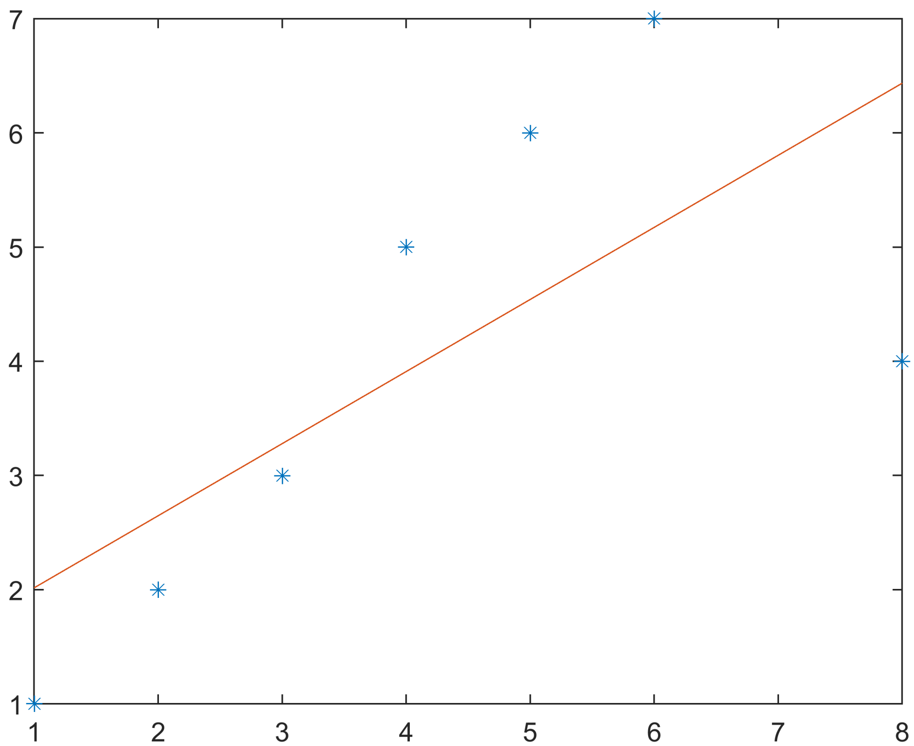
```
n = 1
```

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

```
a = 1
b = 8
x = 7×1
     1
     2
     3
     4
     5
     6
     8
the design matrix is
X = 7×2
     1     1
     2     1
     3     1
     4     1
     5     1
     6     1
     8     1
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
    0.6311
    1.3852

the 2-norm of the residual vector e is 3.756961e+00
the polynomial of degree 1 of the best least-squares fit is
```

P = $0.6311\,x + 1.385$




```
n=2
```

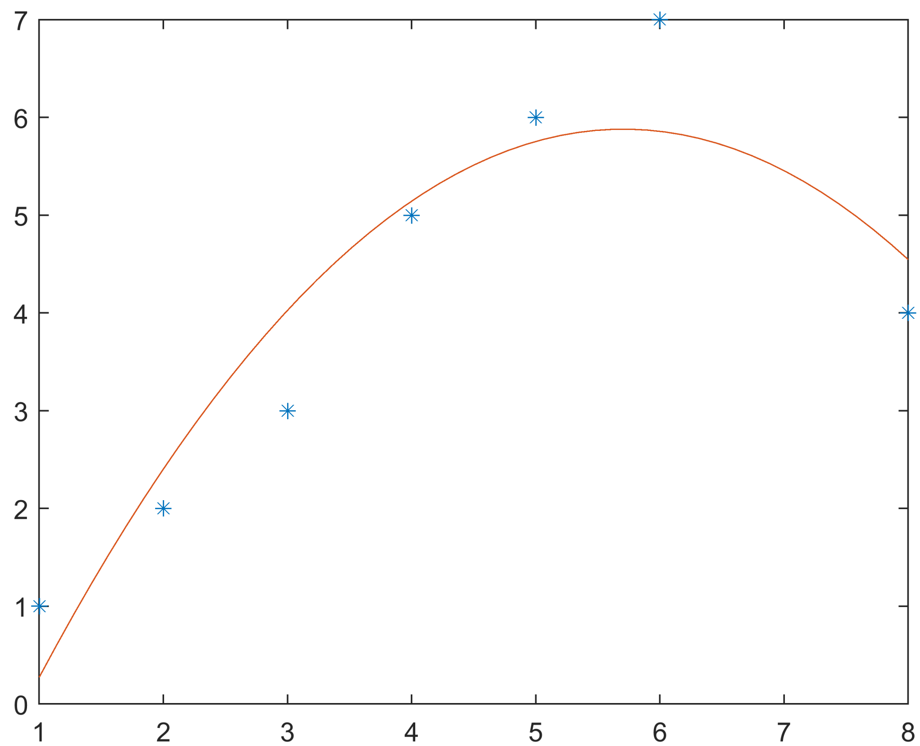
```
n = 2
```

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

```
a = 1
b = 8
x = 7×1
     1
     2
     3
     4
     5
     6
     8
the design matrix is
X = 7×3
     1     1     1
     4     2     1
     9     3     1
    16     4     1
    25     5     1
    36     6     1
    64     8     1
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
   -0.2532
    2.8896
   -2.3636
```

```
the 2-norm of the residual vector e is 1.851640e+00
the polynomial of degree 2 of the best least-squares fit is
```

P = $-0.2532\,x^2 + 2.89\,x - 2.364$




```
n=3
```

```
n = 3
```

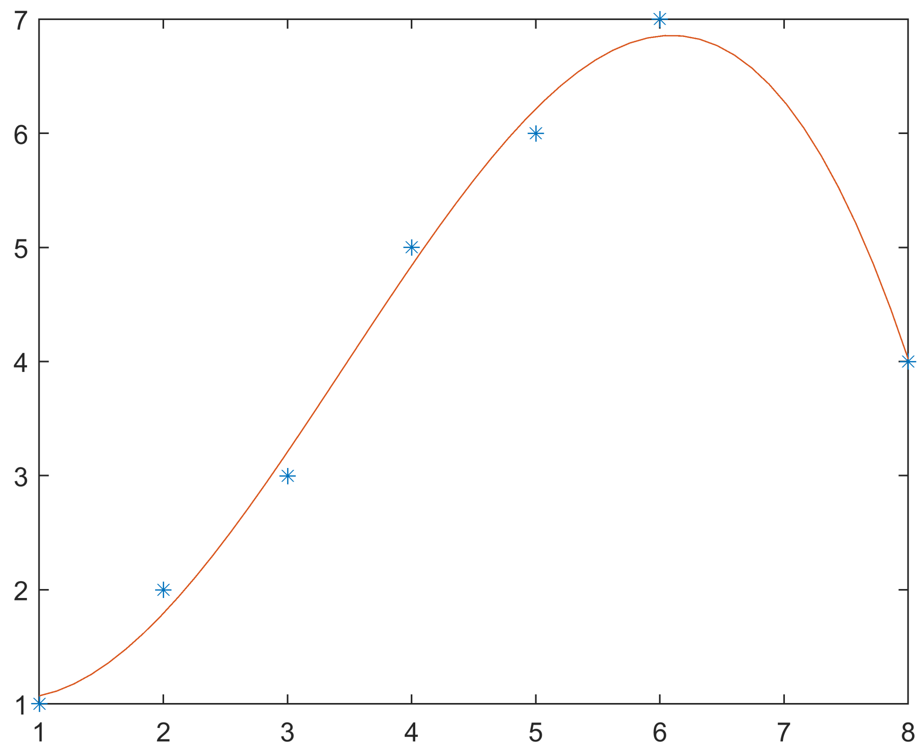
```
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 1
b = 8
x = 7×1
     1
     2
     3
     4
     5
     6
     8
the design matrix is
X = 7×4
     1     1     1     1
     8     4     2     1
    27     9     3     1
    64    16     4     1
   125    25     5     1
   216    36     6     1
   512    64     8     1
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
   -0.0793
```

```
    0.8222
   -1.1892
    1.5173

the 2-norm of the residual vector e is 4.296822e-01
the polynomial of degree 3 of the best least-squares fit is
```

P = $-0.0793\, x^3 + 0.8222\, x^2 - 1.189\, x + 1.517$




```
n=4
```

```
n = 4
```

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

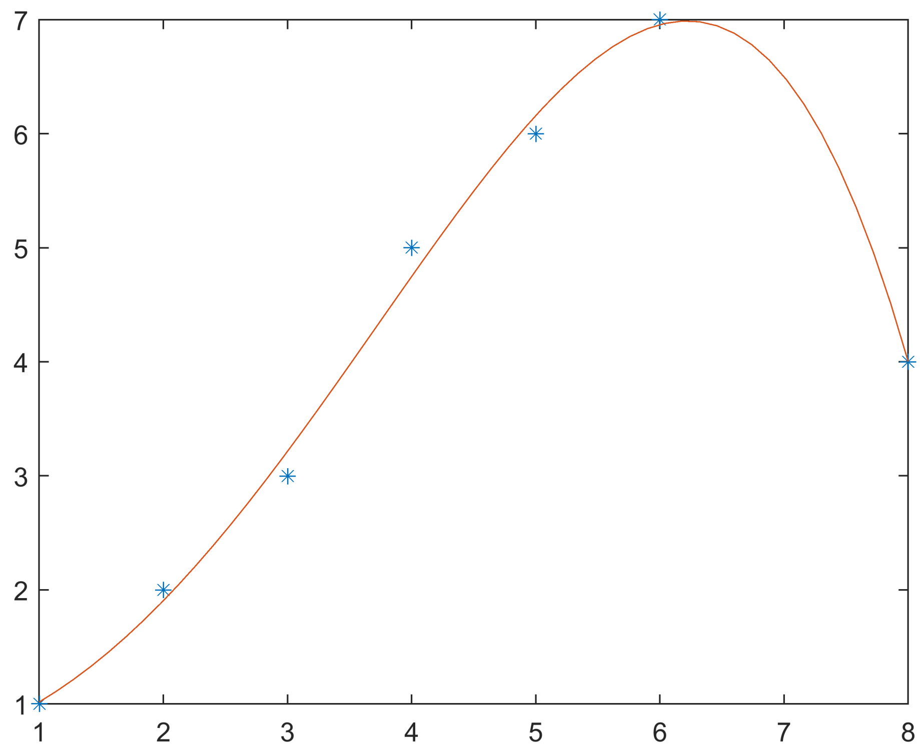
```
a = 1
b = 8
x = 7×1
     1
     2
     3
     4
     5
     6
     8
the design matrix is
X = 7×5
```

1	1	1	1	1
16	8	4	2	1
81	27	9	3	1
256	64	16	4	1
625	125	25	5	1
1296	216	36	6	1
4096	512	64	8	1

```
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
   -0.0061
    0.0286
    0.1927
    0.2026
    0.5994

the 2-norm of the residual vector e is 3.824120e-01
the polynomial of degree 4 of the best least-squares fit is
```

P = $-0.006146\,x^4 + 0.02856\,x^3 + 0.1927\,x^2 + 0.2026\,x + 0.5994$




```
n=5
```

```
n = 5
```

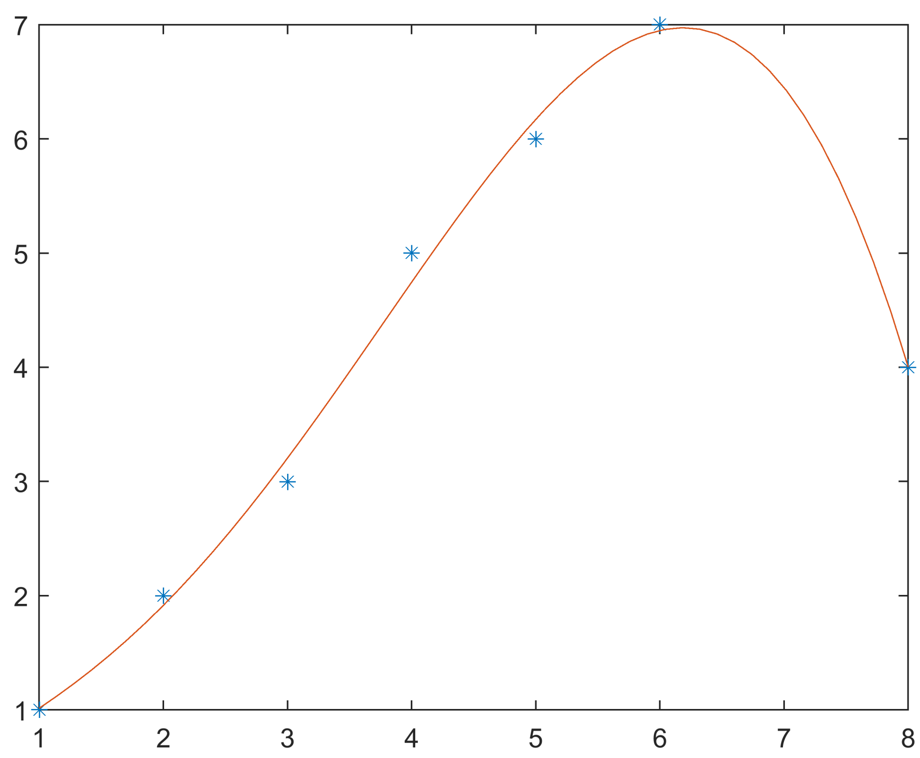
```
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 1
b = 8
x = 7×1
     1
     2
     3
     4
     5
     6
     8
the design matrix is
X = 7×6
           1           1           1           1           1           1
          32          16           8           4           2           1
```

```
         243           81           27            9            3            1
        1024          256           64           16            4            1
        3125          625          125           25            5            1
        7776         1296          216           36            6            1
       32768         4096          512           64            8            1
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
    0.0005
   -0.0176
    0.1179
   -0.1222
    0.6883
    0.3476

the 2-norm of the residual vector e is 3.818946e-01
the polynomial of degree 5 of the best least-squares fit is
```

P = $0.0005348\,x^5 - 0.01756\,x^4 + 0.1179\,x^3 - 0.1222\,x^2 + 0.6883\,x + 0.3476$




```
n=6
```

n = 6

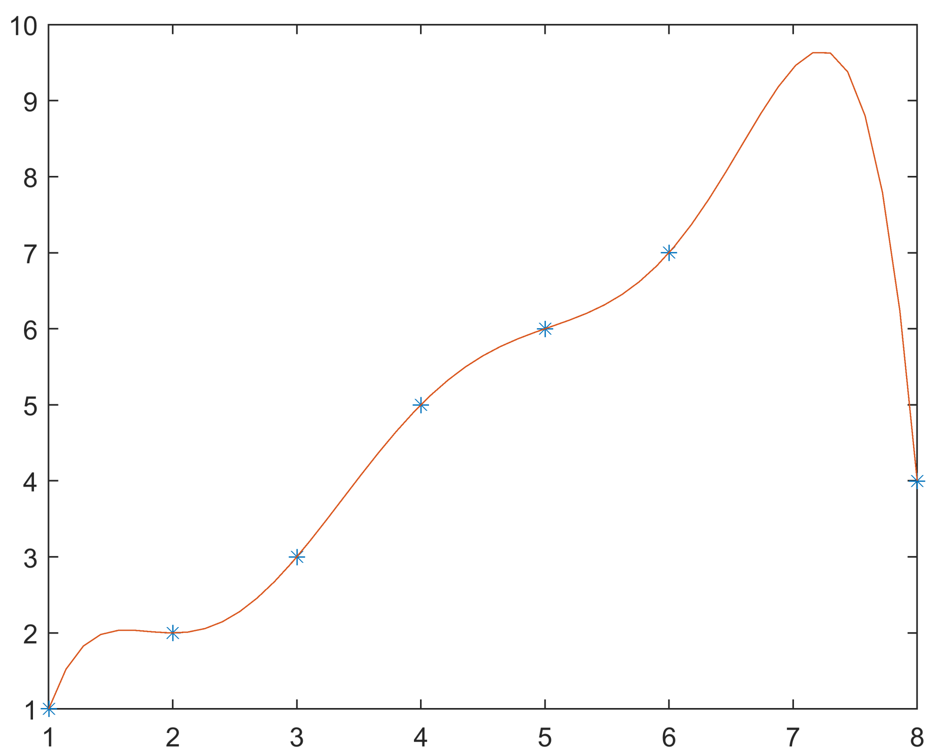
```
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 1
b = 8
x = 7×1
     1
     2
     3
     4
     5
```

```
     6
     8
the design matrix is
X = 7×7
           1           1           1           1           1           1 ···
          64          32          16           8           4           2
         729         243          81          27           9           3
        4096        1024         256          64          16           4
       15625        3125         625         125          25           5
       46656        7776        1296         216          36           6
      262144       32768        4096         512          64           8
the matrix is invertible - there is a unique "exact" solution
an "exact" solution of the system is
   -0.0119
    0.3000
   -2.9583
   14.4167
  -35.9583
   43.7833
  -18.5714

the 2-norm of the residual vector e is 2.568795e-11
the polynomial of degree 6 of the best least-squares fit is
```

P = $-0.0119\,x^6 + 0.3\,x^5 - 2.958\,x^4 + 14.42\,x^3 - 35.96\,x^2 + 43.78\,x - 18.57$




```
%(b)
m=10;
x=(1:m)'/m; y=10*rand(m,1);
n=1
```

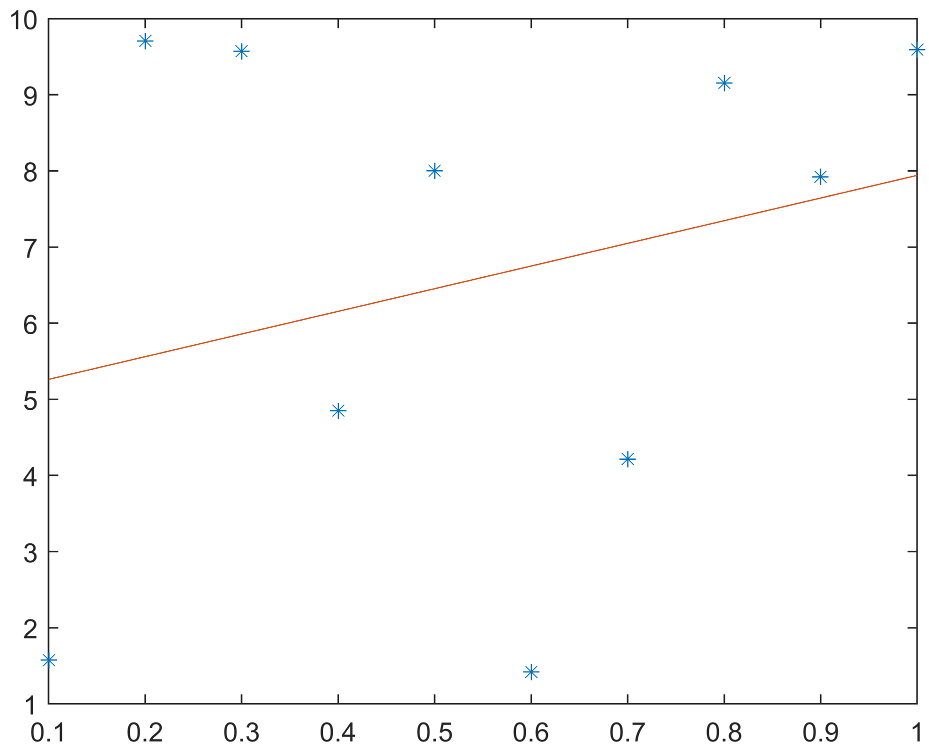
```
n = 1
```

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

```
a = 0.1000
b = 1
x = 10×1
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
the design matrix is
X = 10×2
    0.1000    1.0000
    0.2000    1.0000
    0.3000    1.0000
    0.4000    1.0000
    0.5000    1.0000
    0.6000    1.0000
    0.7000    1.0000
    0.8000    1.0000
    0.9000    1.0000
    1.0000    1.0000
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
    2.9769
    4.9648

the 2-norm of the residual vector e is 9.549901e+00
the polynomial of degree 1 of the best least-squares fit is
```

P = $2.977\,x + 4.965$




```
n=2
```

```
n = 2
```

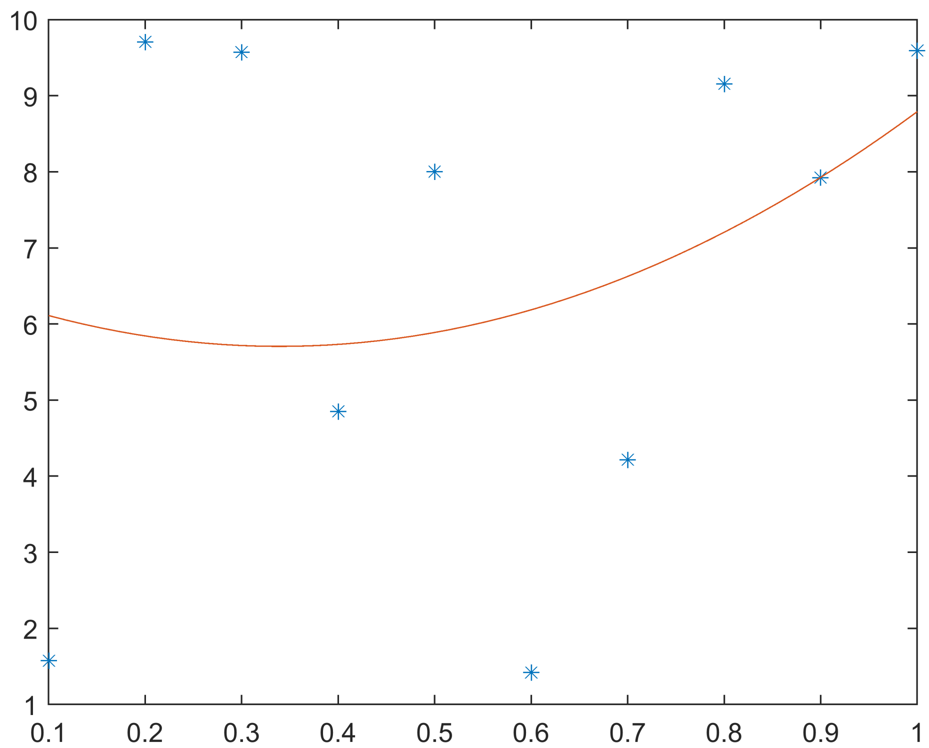
```
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 0.1000
b = 1
x = 10×1
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
the design matrix is
X = 10×3
    0.0100    0.1000    1.0000
    0.0400    0.2000    1.0000
    0.0900    0.3000    1.0000
    0.1600    0.4000    1.0000
    0.2500    0.5000    1.0000
    0.3600    0.6000    1.0000
    0.4900    0.7000    1.0000
    0.6400    0.8000    1.0000
    0.8100    0.9000    1.0000
    1.0000    1.0000    1.0000
the system is inconsistent - look for least-squares solution
the system has a unique solution
```

```
a least-squares solution is
    7.0654
   -4.7950
    6.5192

the 2-norm of the residual vector e is 9.410891e+00
the polynomial of degree 2 of the best least-squares fit is
```

P = $7.065\,x^2 - 4.795\,x + 6.519$




```
n=3
```

```
n = 3
```

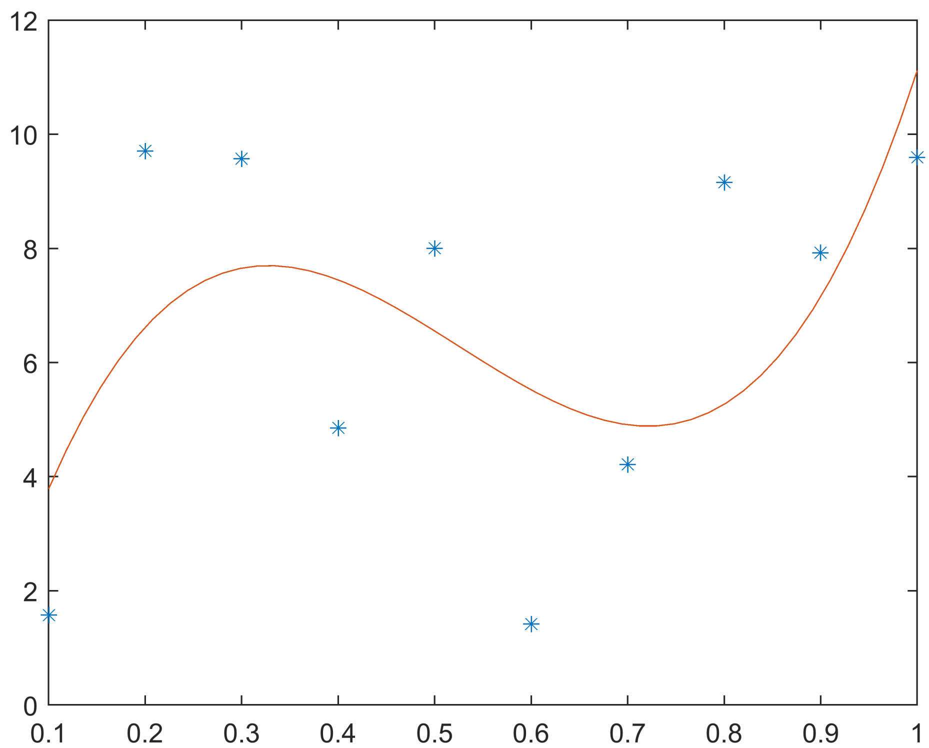
```
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 0.1000
b = 1
x = 10×1
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
the design matrix is
X = 10×4
    0.0010    0.0100    0.1000    1.0000
    0.0080    0.0400    0.2000    1.0000
    0.0270    0.0900    0.3000    1.0000
```

```
    0.0640    0.1600    0.4000    1.0000
    0.1250    0.2500    0.5000    1.0000
    0.2160    0.3600    0.6000    1.0000
    0.3430    0.4900    0.7000    1.0000
    0.5120    0.6400    0.8000    1.0000
    0.7290    0.8100    0.9000    1.0000
    1.0000    1.0000    1.0000    1.0000
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
   92.2673
 -145.1757
   65.4204
   -1.3973

the 2-norm of the residual vector e is 7.891079e+00
the polynomial of degree 3 of the best least-squares fit is
```

P = $92.27\, x^3 - 145.2\, x^2 + 65.42\, x - 1.397$




```
n=4
```

```
n = 4
```

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

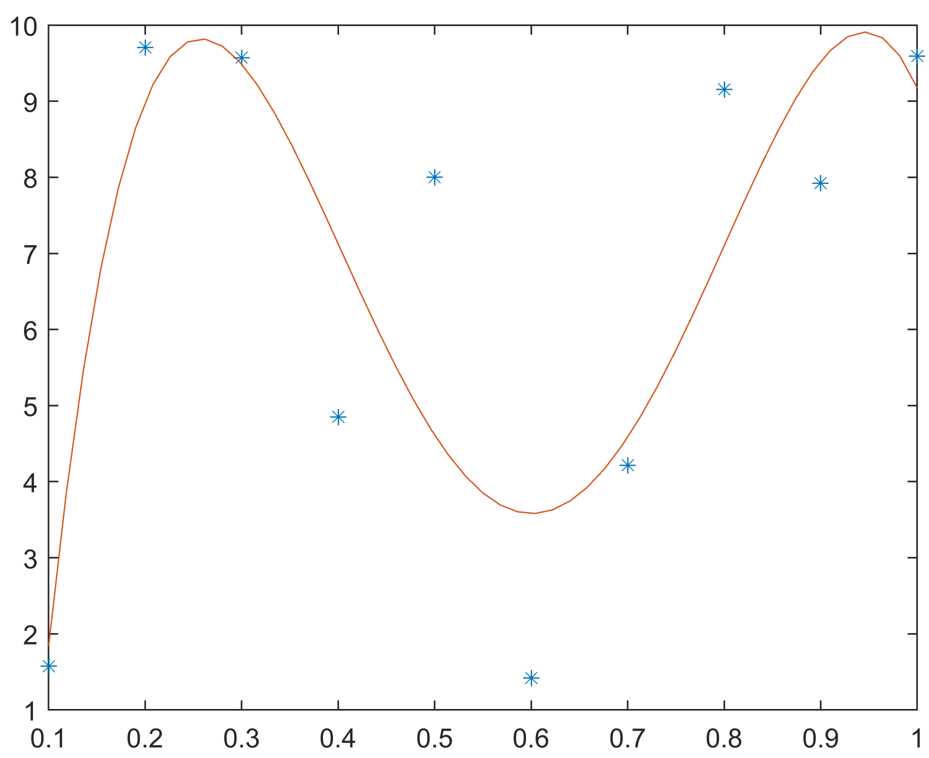
```
a = 0.1000
b = 1
x = 10×1
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
```

```
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
the design matrix is
X = 10×5
    0.0001    0.0010    0.0100    0.1000    1.0000
    0.0016    0.0080    0.0400    0.2000    1.0000
    0.0081    0.0270    0.0900    0.3000    1.0000
    0.0256    0.0640    0.1600    0.4000    1.0000
    0.0625    0.1250    0.2500    0.5000    1.0000
    0.1296    0.2160    0.3600    0.6000    1.0000
    0.2401    0.3430    0.4900    0.7000    1.0000
    0.4096    0.5120    0.6400    0.8000    1.0000
    0.6561    0.7290    0.8100    0.9000    1.0000
    1.0000    1.0000    1.0000    1.0000    1.0000
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
   1.0e+03 *

   -0.4492
    1.0805
   -0.8684
    0.2631
   -0.0168

the 2-norm of the residual vector e is 5.387911e+00
the polynomial of degree 4 of the best least-squares fit is
```

P = $-449.2\,x^4 + 1080.0\,x^3 - 868.4\,x^2 + 263.1\,x - 16.81$




```
n=5
```

```
n = 5
```

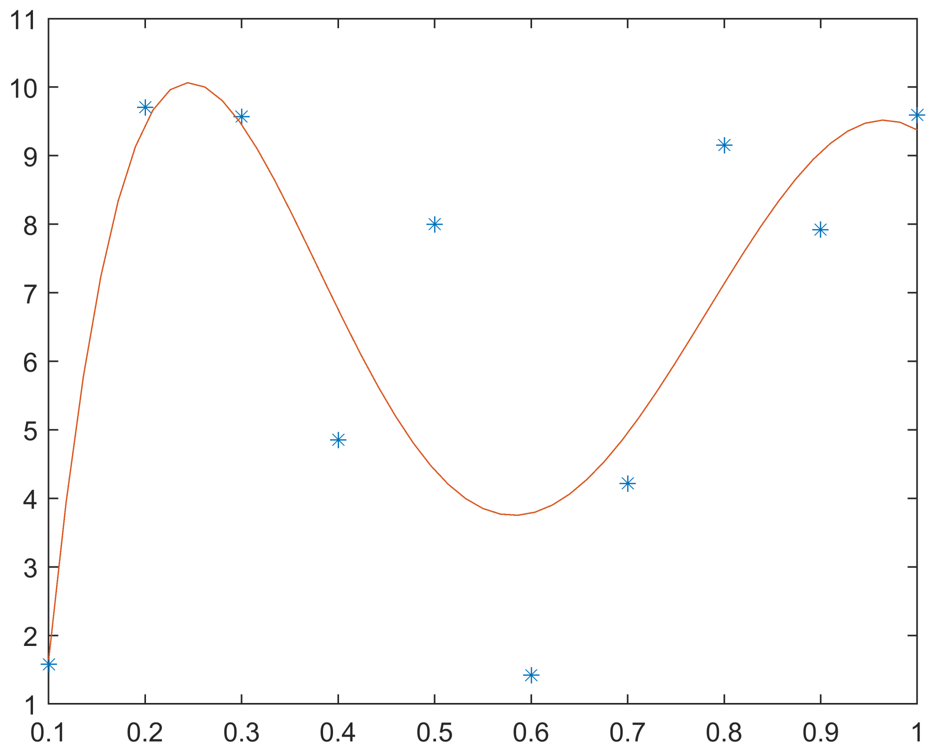
```
[c,X,N]=lstsqpoly(x,y,n);
```

```
a = 0.1000
b = 1
x = 10×1
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
the design matrix is
X = 10×6
    0.0000    0.0001    0.0010    0.0100    0.1000    1.0000
    0.0003    0.0016    0.0080    0.0400    0.2000    1.0000
    0.0024    0.0081    0.0270    0.0900    0.3000    1.0000
    0.0102    0.0256    0.0640    0.1600    0.4000    1.0000
    0.0313    0.0625    0.1250    0.2500    0.5000    1.0000
    0.0778    0.1296    0.2160    0.3600    0.6000    1.0000
    0.1681    0.2401    0.3430    0.4900    0.7000    1.0000
    0.3277    0.4096    0.5120    0.6400    0.8000    1.0000
    0.5905    0.6561    0.7290    0.8100    0.9000    1.0000
    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
the system is inconsistent - look for least-squares solution
the system has a unique solution
a least-squares solution is
   1.0e+03 *

    0.3355
   -1.3719
    2.0088
   -1.2836
    0.3422
   -0.0216

the 2-norm of the residual vector e is 5.305795e+00
the polynomial of degree 5 of the best least-squares fit is
```

P = $335.5\,x^5 - 1372.0\,x^4 + 2009.0\,x^3 - 1284.0\,x^2 + 342.2\,x - 21.61$




```
n=6
```

```
n = 6
```

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

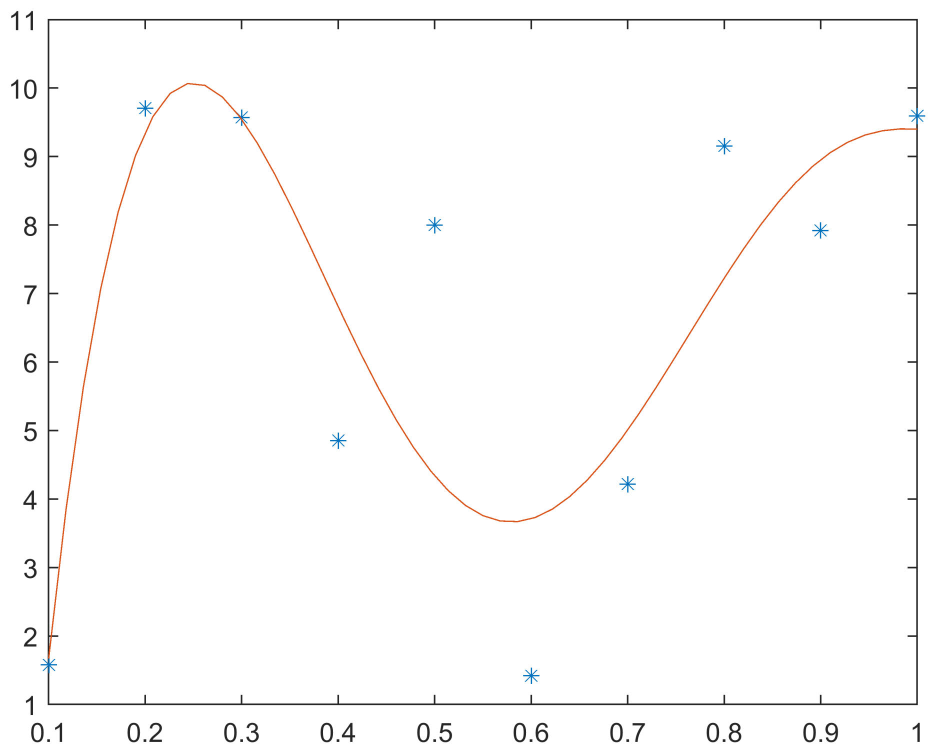
```
a = 0.1000
b = 1
x = 10×1
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
the design matrix is
X = 10×7
    0.0000    0.0000    0.0001    0.0010    0.0100    0.1000    1.0000
    0.0001    0.0003    0.0016    0.0080    0.0400    0.2000    1.0000
    0.0007    0.0024    0.0081    0.0270    0.0900    0.3000    1.0000
    0.0041    0.0102    0.0256    0.0640    0.1600    0.4000    1.0000
    0.0156    0.0313    0.0625    0.1250    0.2500    0.5000    1.0000
    0.0467    0.0778    0.1296    0.2160    0.3600    0.6000    1.0000
    0.1176    0.1681    0.2401    0.3430    0.4900    0.7000    1.0000
    0.2621    0.3277    0.4096    0.5120    0.6400    0.8000    1.0000
    0.5314    0.5905    0.6561    0.7290    0.8100    0.9000    1.0000
    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
the system is inconsistent - look for least-squares solution
the system has a unique solution
```

```
a least-squares solution is
   1.0e+03 *

    0.4132
   -1.0281
    0.3767
    0.9117
   -0.9364
    0.2913
   -0.0190

the 2-norm of the residual vector e is 5.300738e+00
the polynomial of degree 6 of the best least-squares fit is
```

P = $413.2\,x^6 - 1028.0\,x^5 + 376.7\,x^4 + 911.7\,x^3 - 936.4\,x^2 + 291.3\,x - 19.03$



BONUS: When we say polynomial interpolation, we mean we want to create a polynomial that passes through all the given points. Looking at the last graph from part a, each blue point is perfectly passed through by the red polynomial line, so we know this was the right equation.

Exercise 7

```
type invalprob.m
```

```
function []=invalprob(A,X0)
format
[~,n]=size(A);

[L,P,~]=eigenbasis(A);

if isempty(P)
    return
end

fprintf('all eigenvalues of A are\n')
disp(L)
fprintf('an eigenvector basis is\n')
P

syms t

% If all eigenvalues are real
if (isreal(L))
    % Construct E
    Q=exp(t*L);
    for i=1:n
        R(:,i)=P(:,i)*(Q(i));
    end
    E=R;
    fprintf(['an eigenfunction basis for the solution set' ...
        ' is formed the columns of'])
    disp(E)

    S=rref([P X0]);
    S=S(:,3);

    if (n == 2)
        V1=P(:,1);
        V2=P(:,2);
        flag=0;
        if(abs(L(1,1))>0)
            fprintf('the origin is a repeller\nthe line of greatest repulsion goes through the origin and')

            P(:,2)
        else
            if(L(1,1)~=0 && L(1,2)~=0)
                fprintf('the origin is an attractor\nthe line of greatest attraction goes through the origin and')
                flag=1;
                P(:,1)
            else
                fprintf('the origin is a saddle point\nthe line of greatest repulsion goes through the origin and')

                P(:,2)

                fprintf('the line of greatest attraction goes through the origin and')

                P(:,1)
            end
        end

        X0 = [V1,V2,-V1,-V2,X0];
        N=size(X0,2);
        for i=1:N
            C=[V1 V2]\X0(:,i);
```

```
            xt = @(t) C(1)*V1(1)*exp(L(1)*t)+C(2)*V2(1)*exp(L(2)*t);
            yt = @(t) C(1)*V1(2)*exp(L(1)*t)+C(2)*V2(2)*exp(L(2)*t);
            if flag
                fplot(xt,yt,[-1 1])
            else
                fplot(xt,yt,[-.3 .3])
            end
            hold on
        end
        % We are iterating through the matrix and using formula's 4 and 5
        % to plot the parametric functions for the initial vector X0
        hold off
    end

    % Else, eigenvalues are imaginary
else
    V=P(:,2);
    a=real(L(2));
    b=imag(L(2));
    ReV=real(V);
    ImV=imag(V);
    a=closetozeroroundoff(a,7);

    X0 = [ReV,ImV,X0];
    N=length(X0);

    for i=1:N
        C=[ReV ImV]\X0(:,i);
        xt = @(t) (C(1)*(cos(b*t)*ReV(1)-sin(b*t)*ImV(1))+C(2)*(sin(b*t)*ReV(1)+cos(b*t)*ImV(1))).*exp(a*t);
        yt = @(t) (C(1)*(cos(b*t)*ReV(2)-sin(b*t)*ImV(2))+C(2)*(sin(b*t)*ReV(2)+cos(b*t)*ImV(2))).*exp(a*t);
        fplot(xt,yt)
        hold on
    end
    hold off
end
```

```
type eigenbasis.m
```

```
function [L,P,D]=eigenbasis(A)
format
[~,n]=size(A);
P=[];
D=[];
%Part 1
%output eigenvalues with their multiplicity
L = eig(A);
L = sort(L, 'ascend', 'ComparisonMethod','real');
for i = 2:size(L, 1)
 if isequal(closetozeroroundoff(L(i,1)-L(i-1, 1), 7), 0)
 L(i, :) = L(i-1, :);
 end
end
for i = 1:size(L, 1)
 if closetozeroroundoff(L(i, 1)-real(L(i, 1)), 7)==0
 L(i, 1) = real(L(i, 1));
 end
end
%matrix not invertible
if rank(A)~=size(A, 2)
 closetozeroroundoff(L, 12);
end
%Part 2
[m, M] = groupcounts(L);
flag = 0;
```

```
for k = 1 : length(M)
 lambda = M(k);
 multiplicity = m(k);
 mod = A - lambda.*eye(size(A));
 mod = rref(mod);
 W = null(mod, 'r');
 P = [P W];
 dim = size(A, 2) - rank(mod);
 if dim < multiplicity
 flag = 1;
 end
end
%Part 3
%not diagonizable
if isequal(flag, 1)
 disp('A is not diagonizable')
 P = [];
 D = [];
1;
 return
end
%diagonizable
D = diag(L);
A*P;
P*D;
if ~any(closetozeroroundoff(A*P - P*D, 7)) & isequal(rank(P), size(P, 2))
 disp('A is diagonalized')
 return
else
 P = [];
 D = [];
 return
end
```

```
%(a)
A=ones(2)
```

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

```
X0=[[1;-2],[1;2],[-2;5],[0;3],[-2;0]];
invalprob(A,X0)
```

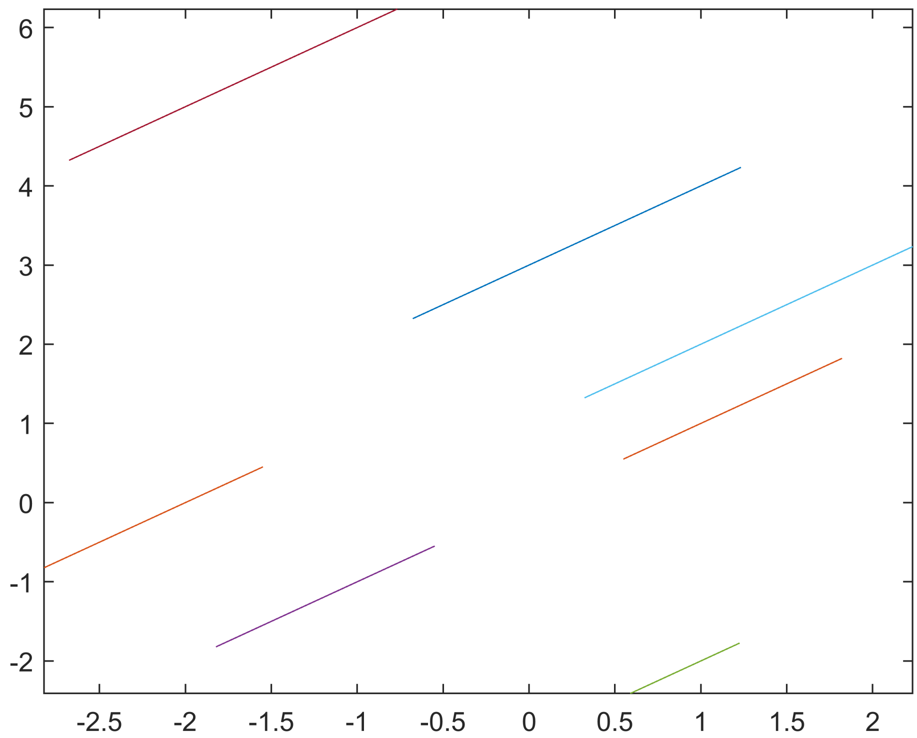
```
A is diagonalized
all eigenvalues of A are
     0
     2

an eigenvector basis is
P = 2×2
    -1     1
     1     1
an eigenfunction basis for the solution set is formed the columns of
```

$$\begin{pmatrix} -1 & e^{2t} \\ 1 & e^{2t} \end{pmatrix}$$

```
the origin is a saddle point
the line of greatest repulsion goes through the origin and
ans = 2×1
     1
```

```
     1
the line of greatest attraction goes through the origin and
ans = 2×1
    -1
     1
```




```
%(b)
A=[-1.5 .5;1 -1]
```

```
A = 2×2
   -1.5000    0.5000
    1.0000   -1.0000
```

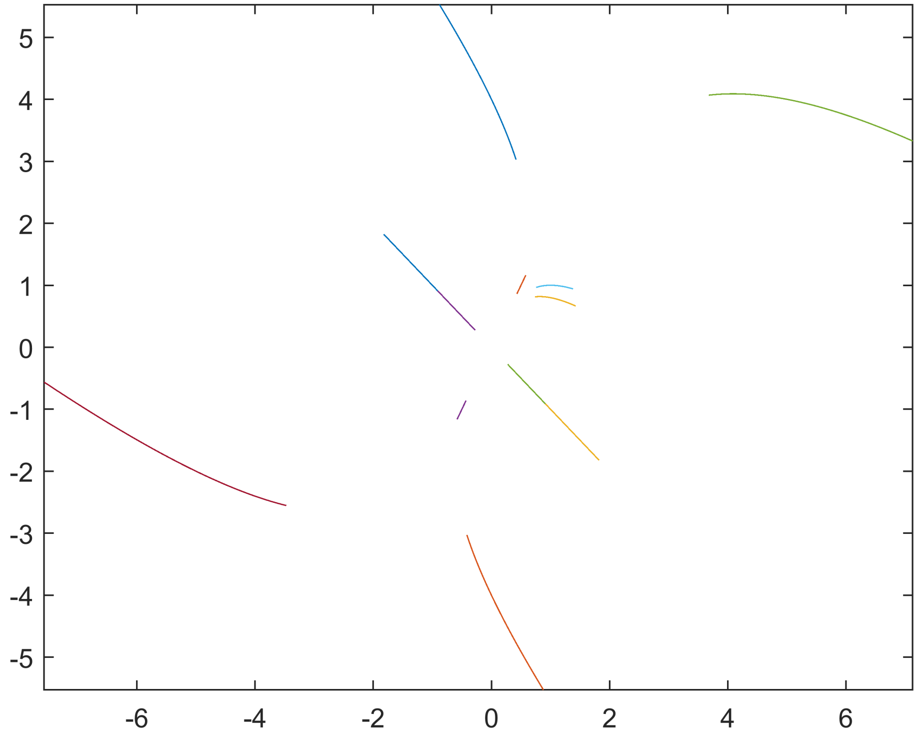
```
X0=[[5;4],[1;1],[-5;-2],[0;4],[0;-4],[1;.8],[-.5;.5],[.5;-.5]];
invalprob(A,X0)
```

```
A is diagonalized
all eigenvalues of A are
   -2.0000
   -0.5000

an eigenvector basis is
P = 2×2
   -1.0000    0.5000
    1.0000    1.0000
an eigenfunction basis for the solution set is formed the columns of
```

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

```
the origin is a repeller
the line of greatest repulsion goes through the origin and
ans = 2×1
    0.5000
    1.0000
```




```
%(c)
A=[4 -5;-2 1]
```

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

```
X0=[[2;1],[0;2],[1;1.2],[-5;0],[-1;-2],[2.9;2.6]];
invalprob(A,X0)
```

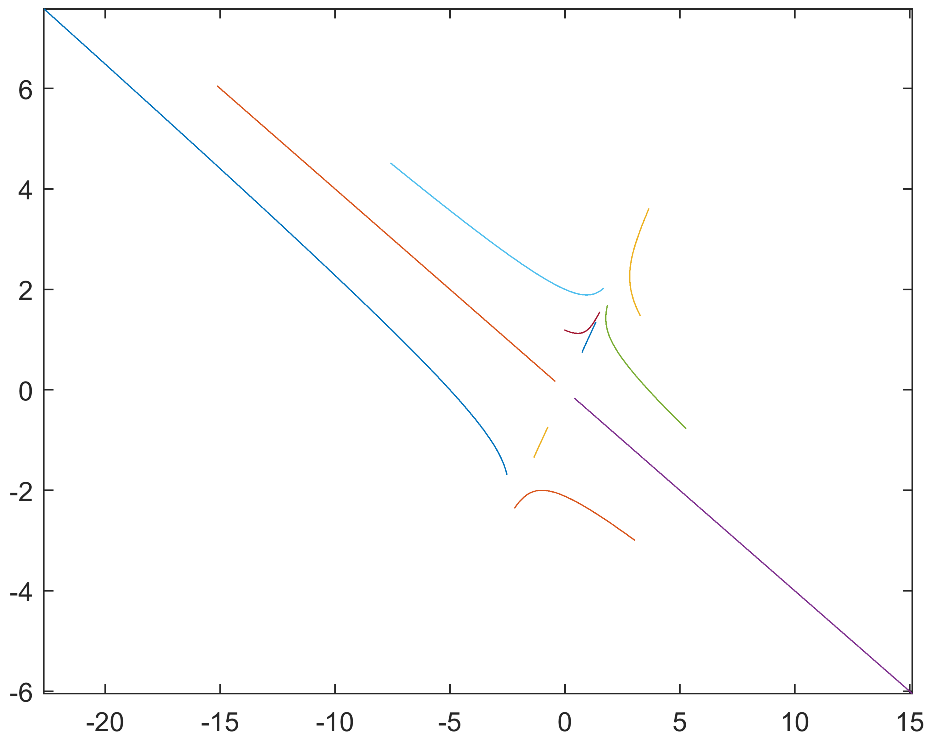
```
A is diagonalized
all eigenvalues of A are
    -1
     6

an eigenvector basis is
P = 2×2
    1.0000   -2.5000
    1.0000    1.0000
an eigenfunction basis for the solution set is formed the columns of
```

$$\begin{pmatrix} e^{-t} & -\frac{5\,e^{6\,t}}{2} \\ e^{-t} & e^{6\,t} \end{pmatrix}$$

```
the origin is a repeller
the line of greatest repulsion goes through the origin and
```

```
ans = 2×1
   -2.5000
    1.0000
```




```
%(d)
A=[2 0; 0 3]
```

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

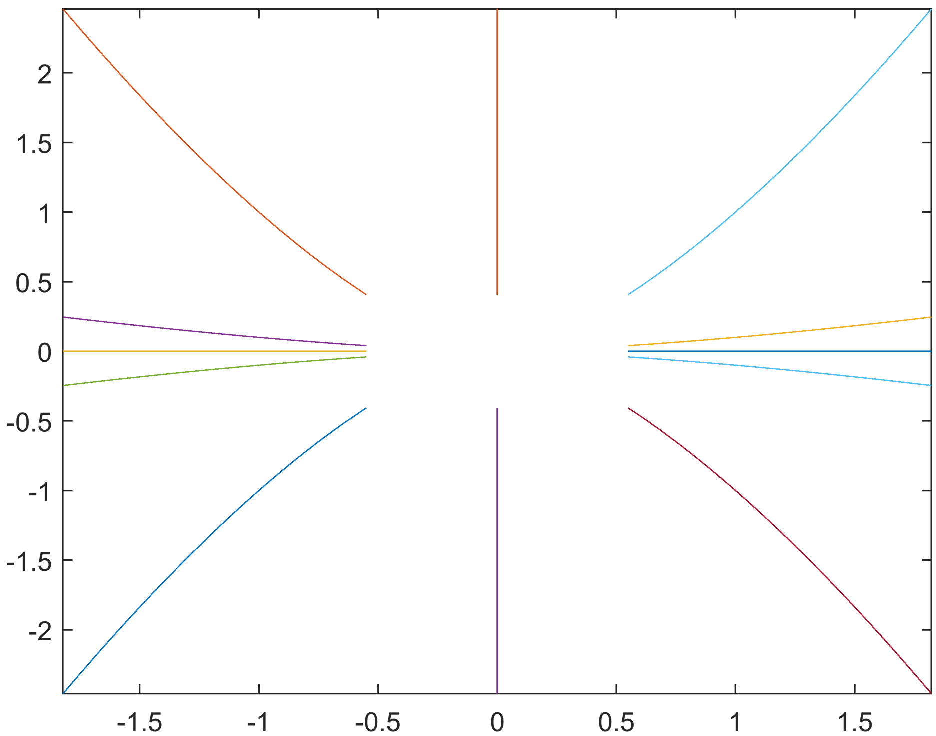
```
X0=[[0;0],[1;1],[1;-1],[-1;-1],[-1;1],[1;.1],[-1;.1],[-1;-.1],[1;-.1]];
invalprob(A,X0)
```

```
A is diagonalized
all eigenvalues of A are
     2
     3

an eigenvector basis is
P = 2×2
     1     0
     0     1
an eigenfunction basis for the solution set is formed the columns of
```

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

```
the origin is a repeller
the line of greatest repulsion goes through the origin and
ans = 2×1
     0
     1
```



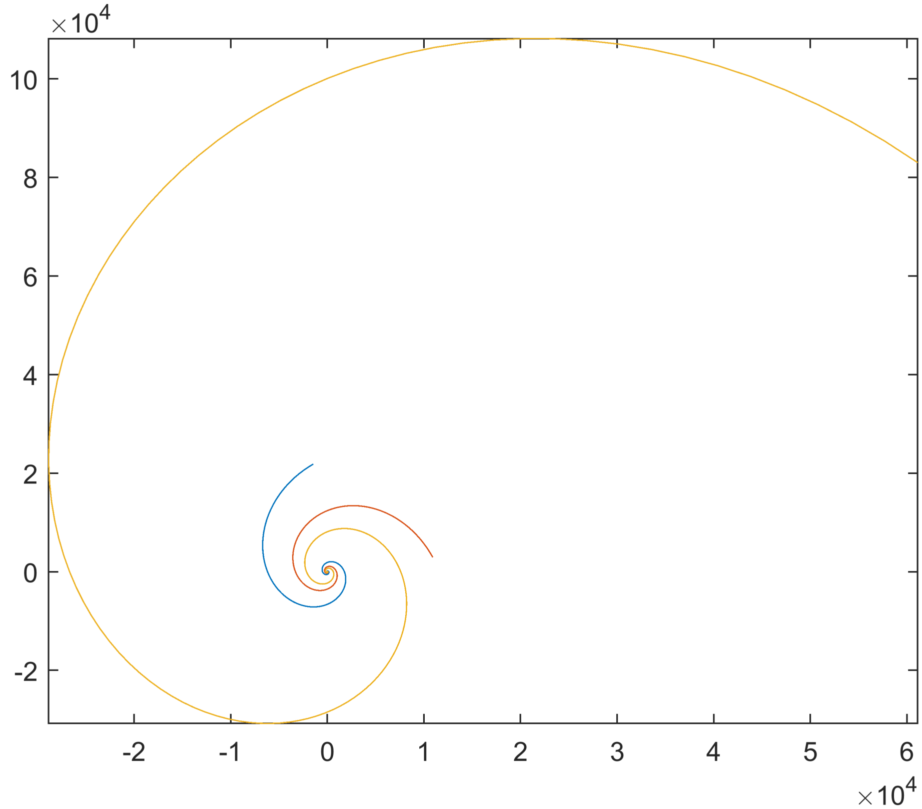

```
%(e)
A=[-2 -2.5;10 -2]
```

```
A = 2×2
   -2.0000   -2.5000
   10.0000   -2.0000
```

```
X0=[3;3];
invalprob(A,X0)
```

```
A is diagonalized
all eigenvalues of A are
  -2.0000 - 5.0000i
  -2.0000 + 5.0000i

an eigenvector basis is
P = 2×2 complex
   0.0000 - 0.5000i   0.0000 + 0.5000i
   1.0000 + 0.0000i   1.0000 + 0.0000i
```



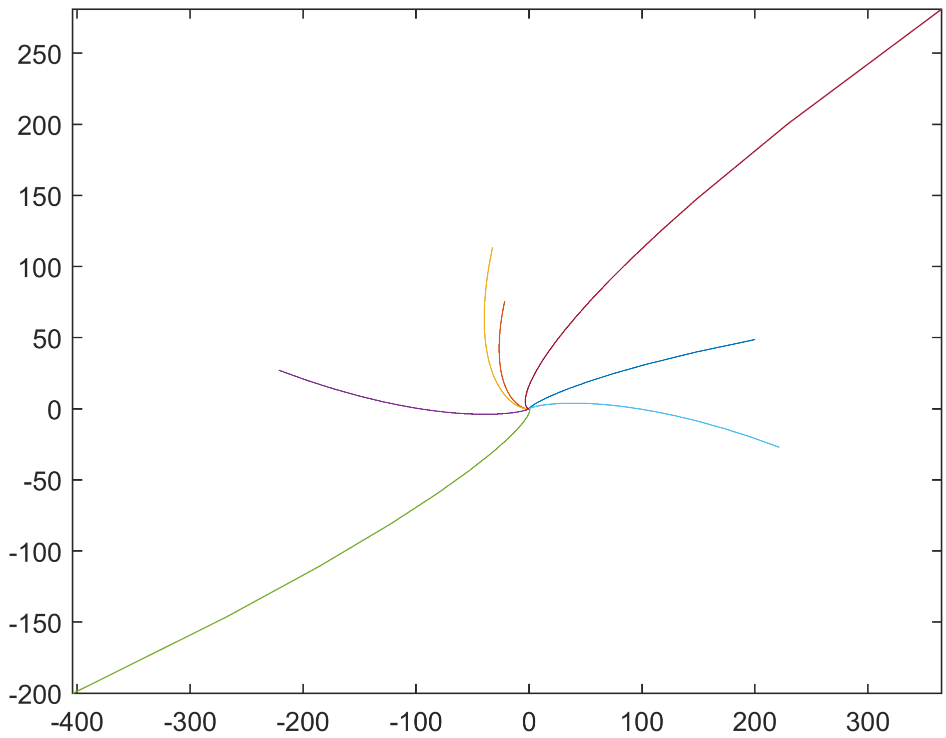

```
%(f)
A=[.8 .5; -.1 1.0]
```

```
A = 2×2
    0.8000    0.5000
   -0.1000    1.0000
```

```
X0=[[-3;0],[-3;-1],[0;-3],[3;1],[-3;3]];
invalprob(A,X0)
```

```
A is diagonalized
all eigenvalues of A are
   0.9000 - 0.2000i
   0.9000 + 0.2000i

an eigenvector basis is
P = 2×2 complex
   1.0000 + 2.0000i   1.0000 - 2.0000i
   1.0000 + 0.0000i   1.0000 + 0.0000i
```




```
%(g)
A=[-2 -5;1 4]
```

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

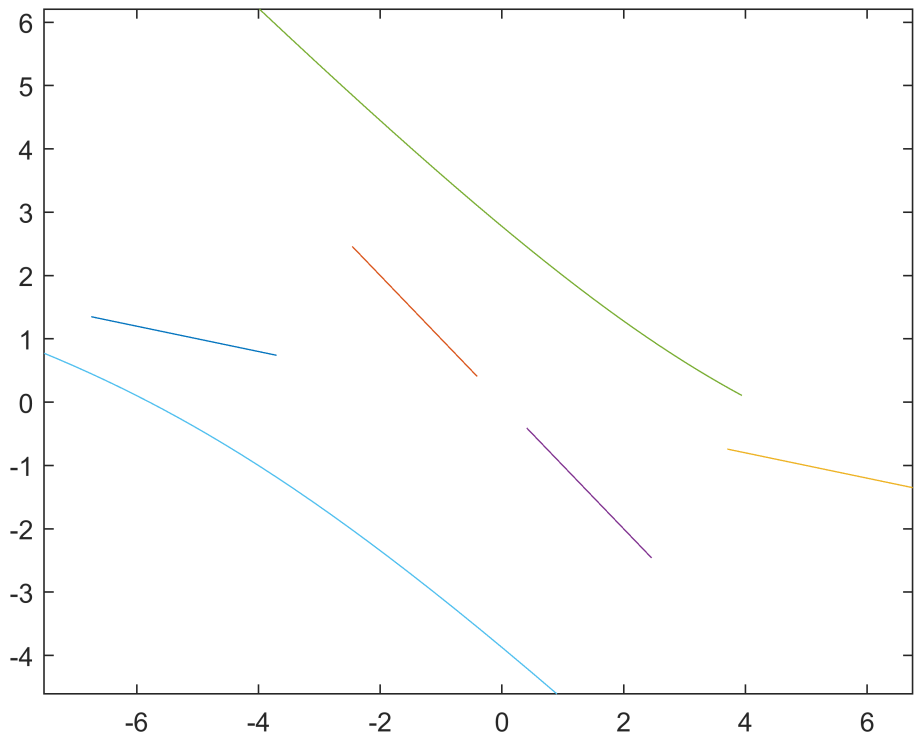
```
x0=[[1;2],[-4;-1]];
invalprob(A,x0)
```

```
A is diagonalized
all eigenvalues of A are
    -1
     3

an eigenvector basis is
P = 2×2
    -5    -1
     1     1
an eigenfunction basis for the solution set is formed the columns of
```

$$\begin{pmatrix} -5\,e^{-t} & -e^{3\,t} \\ e^{-t} & e^{3\,t} \end{pmatrix}$$

```
the origin is a repeller
the line of greatest repulsion goes through the origin and
ans = 2×1
    -1
     1
```




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

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

```
x0=[1;3];
invalprob(A,x0)
```

```
A is not diagonizable
```

```
%(i)
A=diag([1,2,2,3,3,3])
```

```
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=ones(6,1);
invalprob(A,x0)
```

```
A is diagonalized
all eigenvalues of A are
     1
     2
     2
```

```
     3
     3
     3

an eigenvector basis is
P = 6×6
     1     0     0     0     0     0
     0     1     0     0     0     0
     0     0     1     0     0     0
     0     0     0     1     0     0
     0     0     0     0     1     0
     0     0     0     0     0     1
an eigenfunction basis for the solution set is formed the columns of
```

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

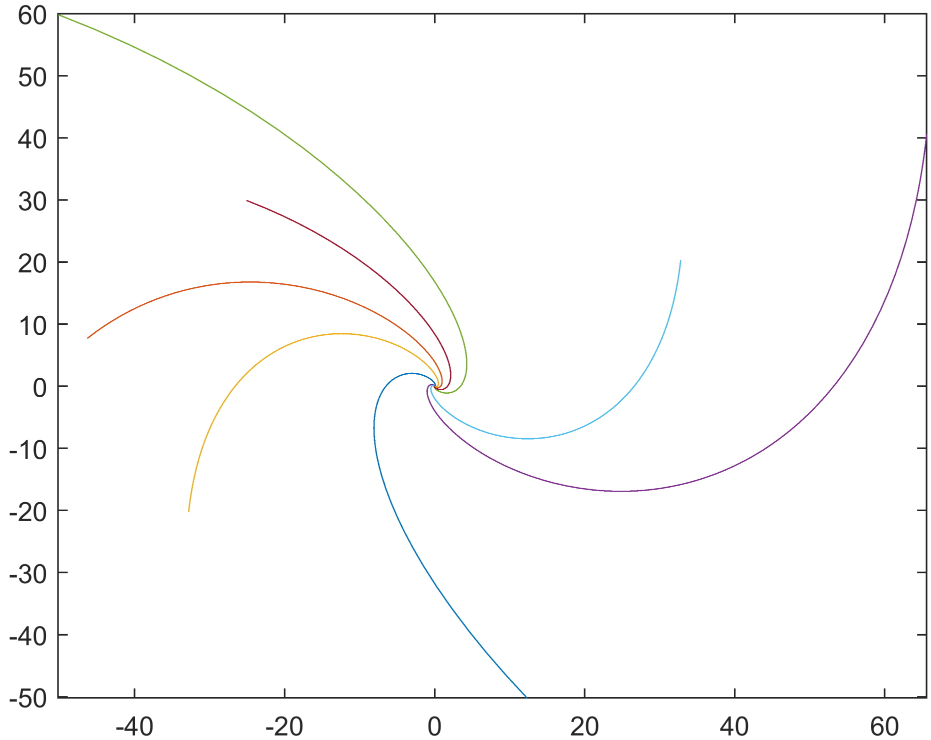
```
%(j)
A=[.5 -.6;.75 1.1]
```

```
A = 2×2
    0.5000   -0.6000
    0.7500    1.1000
```

```
X0=[[.5;.5],[-1;-1],[1;-1],[-.5;-.5],[.5;-.5]];
invalprob(A,X0)
```

```
A is diagonalized
all eigenvalues of A are
   0.8000 - 0.6000i
   0.8000 + 0.6000i

an eigenvector basis is
P = 2×2 complex
  -0.4000 - 0.8000i  -0.4000 + 0.8000i
   1.0000 + 0.0000i   1.0000 + 0.0000i
```



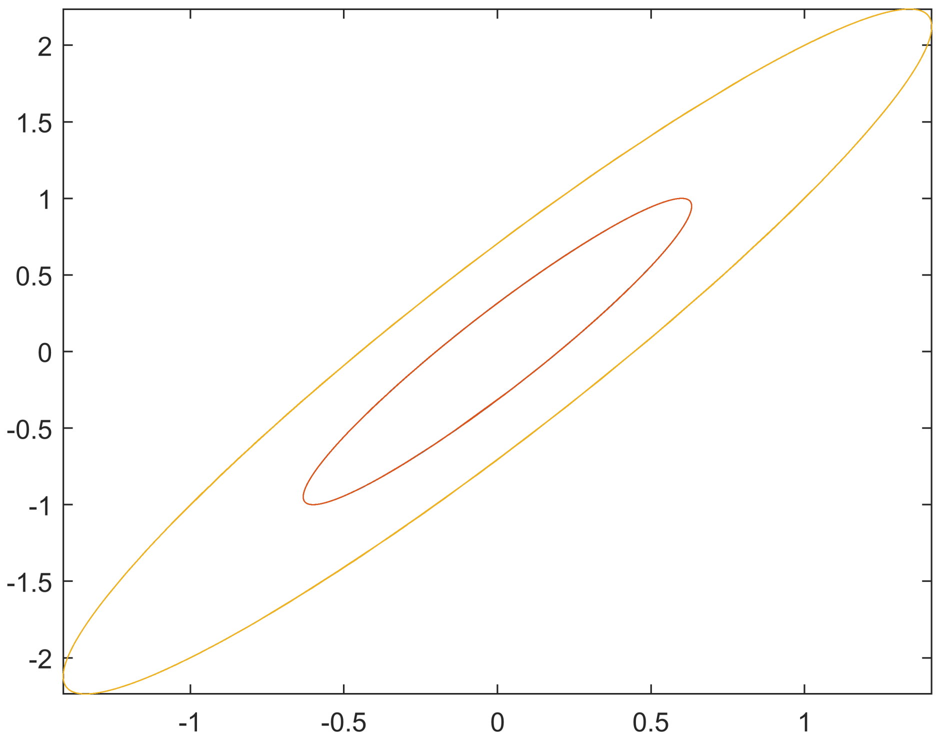

```
%(k)
A=[-6 4;-10 6]
```

```
A = 2×2
    -6     4
   -10     6
```

```
X0=[1;2];
invalprob(A,X0)
```

```
A is diagonalized
all eigenvalues of A are
  -0.0000 - 2.0000i
  -0.0000 + 2.0000i

an eigenvector basis is
P = 2×2 complex
   0.6000 + 0.2000i   0.6000 - 0.2000i
   1.0000 + 0.0000i   1.0000 + 0.0000i
```


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1
-0.5
0
0.5
1