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 # 5

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

1. Corey Wolfe
2. Thomas Pena Reina
3. Manuel Vera Miranda
4. Elton Li
5. Ekaterina Krysova
6. Dany Rashwan

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

Part I. Elementary Row Operations

Exercise 1

```
type ele1
```

```
function E1=ele1(n,r,i,j)

E1=eye(n);

E1(j,:)=E1(j,:)+r*E1(i,:);

end
```

```
type ele2
```

```
function E2=ele2(n,i,j)

E2=eye(n);

temp=E2(j,:);

E2(j,:)=E2(i,:);

E2(i,:)=temp;

end
```

```
type ele3
```

```
function E3=ele3(n,j,k)

E3=eye(n);

E3(j,:)=k*E3(j,:);

end
```

```
format
format compact

A=[0 1 3 1; 2 4 6 -2; 3 1 4 2]
```

```
A = 3×4
     0     1     3     1
     2     4     6    -2
     3     1     4     2
```

```
E2=ele2(3,1,2);
%switches first and second rows
```

```
A1=E2*A
```

```
A1 = 3×4
     2     4     6    -2
     0     1     3     1
     3     1     4     2
```

```
E3=ele3(3,1,1/2);
%multiplies row 1 by 1/2
A2=E3*A1
```

```
A2 = 3×4
     1     2     3    -1
     0     1     3     1
     3     1     4     2
```

```
E1=ele1(3,-3,1,3);
%multiplies row 1 by -3 and adds it to the 3rd row
A3=E1*A2
```

```
A3 = 3×4
     1     2     3    -1
     0     1     3     1
     0    -5    -5     5
```

```
E1=ele1(3,5,2,3);
%multiplies row 2 by 5 and adds it to the 3rd row
A4=E1*A3
```

```
A4 = 3×4
     1     2     3    -1
     0     1     3     1
     0     0    10    10
```

```
E3=ele3(3,3,1/10);
%multplies the 3rd row by 1/10
A5=E3*A4
```

```
A5 = 3×4
     1     2     3    -1
     0     1     3     1
     0     0     1     1
```

```
E1=ele1(3,-3,3,2);
%multiplies the 3rd row by -3 and adds it to the 2nd row
A6=E1*A5
```

```
A6 = 3×4
     1     2     3    -1
     0     1     0    -2
     0     0     1     1
```

```
E1=ele1(3,-3,3,1);
```

```
%multiplies the 3rd row by -3 and adds it to the 1st row
A7=E1*A6
```

```
A7 = 3×4
     1     2     0    -4
     0     1     0    -2
     0     0     1     1
```

```
E1=ele1(3,-2,2,1);
%multiplies the 2nd row by -2 and adds it to the 1st row.
A8=E1*A7
```

```
A8 = 3×4
     1     0     0     0
     0     1     0    -2
     0     0     1     1
```

```
%A8 is AN, and is in final reduced echelon form.
Ared=rref(A);
%Ared is matlabs reduced form matrix.
if (Ared == A8)
    fprintf 'Your reduced form matches MATLABs reduced form!'
end
```

```
Your reduced form matches MATLABs reduced form!
```

Part II. Basic Operations

Exercise 2

```
type inverses
```

```
% function used to find the inverse of a matrix
function F=inverses(A)
% matrix A is (n x n), for it to be inveretible
% A needs to have the n pivot points

rankA = rank(A);
n = size(A,1);

% determines if A is invertible
if (rankA ~= n)%returns empty matrix F and message
    F = []
    fprintf("matrix A is not invertible");
    return
else
    F = rref([A eye(n)]);
    %eliminates the first block of the matrix, results in inverse
    F = (F(:,(n+1 :(2 * n))))

    fprintf("P is the inverse of A using MATLAB function");
```

```
    P = inv(A) %MATLAB build in inverse

    % function closetozeroroundoff used with p=5
    DiffMat = closetozeroroundoff(abs(F-P),5);

    % compares matrices F and P
    if isequal(DiffMat,zeros(n))
        fprintf("\nMatrices F and P match");
    else
        fprintf("check code");
    end
end
```

```
type closetozeroroundoff
```

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

```
%(a)
A=[4 0 -7 -7;-6 1 11 9;7 -5 10 19;-1 2 3 -1]
```

```
A = 4×4
     4     0    -7    -7
    -6     1    11     9
     7    -5    10    19
    -1     2     3    -1
```

```
F= inverses(A);
```

```
F = 4×4
   -19   -14     0     7
  -549  -401    -2   196
   267   195     1   -95
  -278  -203    -1    99
P is the inverse of A using MATLAB function
P = 4×4
  -19.0000  -14.0000   -0.0000    7.0000
 -549.0000 -401.0000   -2.0000  196.0000
  267.0000  195.0000    1.0000  -95.0000
 -278.0000 -203.0000   -1.0000   99.0000
Matrices F and P match
```

```
%(b)
A=[1 -3 2 -4;-3 9 -1 5;2 -6 4 -3;-4 12 2 7]
```

```
A = 4×4
     1    -3     2    -4
    -3     9    -1     5
     2    -6     4    -3
    -4    12     2     7
```

```
F = inverses(A);
```

```
F =
     []
matrix A is not invertible
```

```
%(c)
A=magic(3);
F = inverses(A);
```

```
F = 3×3
     0.1472   -0.1444    0.0639
    -0.0611    0.0222    0.1056
    -0.0194    0.1889   -0.1028
P is the inverse of A using MATLAB function
P = 3×3
     0.1472   -0.1444    0.0639
    -0.0611    0.0222    0.1056
    -0.0194    0.1889   -0.1028
Matrices F and P match
```

```
%(d)
A=hilb(5);
F = inverses(A);
```

```
F = 5×5
          25        -300        1050       -1400         630
        -300        4800      -18900       26880      -12600
        1050      -18900       79380     -117600       56700
       -1400       26880     -117600      179200      -88200
         630      -12600       56700      -88200       44100
P is the inverse of A using MATLAB function
P = 5×5
10^5 ×
     0.0002   -0.0030    0.0105   -0.0140    0.0063
    -0.0030    0.0480   -0.1890    0.2688   -0.1260
     0.0105   -0.1890    0.7938   -1.1760    0.5670
    -0.0140    0.2688   -1.1760    1.7920   -0.8820
     0.0063   -0.1260    0.5670   -0.8820    0.4410
Matrices F and P match
```

```
%(e)
A=magic(6);
F = inverses(A);
```

```
F =
     []
matrix A is not invertible
```

Part III. Solving Equations

Exercise 3

Part 1. Solving a system Ax = b

```
type solvesys.m
```

```
function [C,N]=solvesys(A,b)
    % This function takes as input an n*n matrix A and an n*1 vector b
    %   If matrix A is invertible, the outputs are matrix C, whose
    % columns x1, x2, x3 are the three solutions to Ax=b obtained
    % using the three methods \, inv, and rref; and N is the vector whose
    % 3 entries are the 2-norms of the vectors of the differences
    % between each two distinct solutions.
    [~,n]=size(A);
    format long

    if rank(A) < n %checks if the matrix is not invertible
        disp('A is not invertible'); %returns the empty matrices C and N
        C = [];
        N = [];

        if rank([A b]) ~= rank(A) %checks if the system is inconsistent
            disp('The system is inconsistent');
        else
            disp('The system is consistent, but the solution is not unique');
        end
        return %ends the function if A is not invertible
    end

    x1 = A\b; %first method for solution
    x2 = inv(A)*b; %second method for solution
    refMat = rref([A b]); %matrix obtained from reducing the extended matrix
    x3 = refMat(:, end); %third method; last column of reduced matrix

    C = [x1 x2 x3]; %solution matrix

    disp('Solutions obtained by different methods are the columns of ');
    disp(C);

    n1=norm(x1-x2);
    n2=norm(x2-x3);
    n3=norm(x3-x1);

    disp('Norms of differences between solutions are the entries of');
    N=[n1;n2;n3] %matrix of differences between solutions
end
```

```
%(a)
A=magic(4); I=eye(size(A,1)); b=I(:,end)
```

```
b = 4×1
     0
     0
     0
     1
```

```
solvesys(A, b);
```

```
A is not invertible
The system is inconsistent
```

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

```
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
```

```
solvesys(A,b);
```

```
A is not invertible
The system is consistent, but the solution is not unique
```

```
%(c)
A=magic(5); b=fix(10*rand(size(A,1),1))
```

```
b = 5×1
     0
     2
     5
     9
     9
```

```
solvesys(A, b);
```

```
Solutions obtained by different methods are the columns of
  Column 1
  -0.033653846153846
   0.029807692307692
   0.376923076923077
   0.099038461538461
  -0.087500000000000
  Column 2
  -0.033653846153846
   0.029807692307692
   0.376923076923077
   0.099038461538461
  -0.087500000000000
  Column 3
  -0.033653846153846
   0.029810298102981
   0.376923076923077
   0.099041533546326
  -0.087500000000000
```

```
Norms of differences between solutions are the entries of
N = 3×1
10^-5 ×
   0.000000000005732
   0.402832488834191
   0.402832488834704
```

```
%(d)
A=eye(6); b=fix(10*rand(size(A,1),1))
```

```
b = 6×1
     1
     9
     9
     4
     8
     1
```

```
solvesys(A, b);
```

```
Solutions obtained by different methods are the columns of
    1     1     1
    9     9     9
    9     9     9
    4     4     4
    8     8     8
    1     1     1
Norms of differences between solutions are the entries of
N = 3×1
     0
     0
     0
```

In part (d), since matrix A was the 6x6 identity matrix, the solution to the system is vector b.

```
%(e)
A=magic(7), b=fix(10*rand(size(A,1),1))
```

```
A = 7×7
    30    39    48     1    10    19    28
    38    47     7     9    18    27    29
    46     6     8    17    26    35    37
     5    14    16    25    34    36    45
    13    15    24    33    42    44     4
    21    23    32    41    43     3    12
    22    31    40    49     2    11    20
b = 7×1
     4
     9
     7
     9
     6
     0
     8
```

```
solvesys(A, b);
```

```
Solutions obtained by different methods are the columns of
  Column 1
  -0.026359753203607
   0.091580446131941
  -0.074532510678690
   0.106530612244898
  -0.119383009017560
   0.179620313241576
   0.088258186995729
  Column 2
  -0.026359753203607
   0.091580446131941
  -0.074532510678690
   0.106530612244898
  -0.119383009017560
   0.179620313241576
   0.088258186995729
  Column 3
  -0.026359143327842
   0.091575091575092
  -0.074534161490683
   0.106529209621993
  -0.119383825417202
   0.179620034542314
   0.088258471237195
Norms of differences between solutions are the entries of
N = 3×1
10^-5 ×
    0.000000000010292
    0.587883777771398
    0.587883777778084
```

```
%(f)
A=hilb(7), b
```

```
A = 7×7
    1.000000000000000   0.500000000000000   0.333333333333333 ⋯
    0.500000000000000   0.333333333333333   0.250000000000000
    0.333333333333333   0.250000000000000   0.200000000000000
    0.250000000000000   0.200000000000000   0.166666666666667
    0.200000000000000   0.166666666666667   0.142857142857143
    0.166666666666667   0.142857142857143   0.125000000000000
    0.142857142857143   0.125000000000000   0.111111111111111
b = 7×1
     4
     9
     7
     9
     6
     0
     8
```

```
solvesys(A, b);
```

```
Solutions obtained by different methods are the columns of
   1.0e+08 *
  Column 1
   0.001739080006816
  -0.074064480272672
   0.745516802628954
  -2.991962410221816
   5.619398418736615
  -4.949073376185741
   1.650208565312696
  Column 2
   0.001739080008332
  -0.074064480336399
   0.745516803265350
  -2.991962412762030
   5.619398423488756
  -4.949073380358478
   1.650208566700776
  Column 3
   0.001739080000000
  -0.074064480000000
   0.745516800000000
  -2.991962410000000
   5.619398420000000
  -4.949073370000000
   1.650208560000000
Norms of differences between solutions are the entries of
N = 3×1
    0.698448925446246
    1.351941001755965
    0.866713758011968
```

Of the above exercises, it can be seen that the difference between solutions is small enough in most cases except in part (f), where it is greater than 2 between x1 and x2. The only one where it is truly 0 is in part (d), because of what was stated above.

Part 2. Condition numbers

```
c1 = cond(magic(7))
```

```
c1 =    7.111323446624705
```

```
c2 = cond(hilb(7))
```

```
c2 =      4.753673563768867e+08
```

c1 is significantly smaller than c2. This indicates that having a large condition number increases the error when solving a system of equations, and is the reason why the difference in the norms in part (f) is so much greater than in part (e).

```
A = hilb(7)
```

```
A = 7×7
    1.000000000000000   0.500000000000000   0.333333333333333 ⋯
    0.500000000000000   0.333333333333333   0.250000000000000
    0.333333333333333   0.250000000000000   0.200000000000000
    0.250000000000000   0.200000000000000   0.166666666666667
    0.200000000000000   0.166666666666667   0.142857142857143
    0.166666666666667   0.142857142857143   0.125000000000000
    0.142857142857143   0.125000000000000   0.111111111111111
```

```
b=ones(7,1)
```

```
b = 7×1
     1
     1
     1
     1
     1
     1
     1
```

```
x=A\b
```

```
x = 7×1
10^4 ×
    0.000700000004222
   -0.033600000168537
    0.378000001622415
   -1.680000006300972
    3.465000011539436
   -3.326400009961335
    1.201200003267738
```

```
b1=b+0.01
```

```
b1 = 7×1
    1.010000000000000
    1.010000000000000
    1.010000000000000
    1.010000000000000
    1.010000000000000
    1.010000000000000
    1.010000000000000
```

```
y=A\b1
```

```
y = 7×1
10^4 ×
    0.000707000004265
   -0.033936000170225
    0.381780001638653
   -1.696800006364036
    3.499650011654935
   -3.359664010061042
    1.213212003300447
```

```
norm(x-y)
```

```
ans =      5.242180560287886e+02
```

```
c3=rcond(A)
```

```
c3 =      1.015027595488996e-09
```

```
A = magic(7)
```

```
A = 7×7
    30    39    48     1    10    19    28
    38    47     7     9    18    27    29
    46     6     8    17    26    35    37
     5    14    16    25    34    36    45
    13    15    24    33    42    44     4
    21    23    32    41    43     3    12
    22    31    40    49     2    11    20
```

```
b=ones(7,1)
```

```
b = 7×1
     1
     1
     1
     1
     1
     1
     1
```

```
x=A\b
```

```
x = 7×1
   0.005714285714286
   0.005714285714286
   0.005714285714286
   0.005714285714286
   0.005714285714286
   0.005714285714286
   0.005714285714286
```

```
b1=b+0.01
```

```
b1 = 7×1
   1.010000000000000
   1.010000000000000
   1.010000000000000
   1.010000000000000
   1.010000000000000
   1.010000000000000
   1.010000000000000
```

```
y=A\b1
```

```
y = 7×1
    0.005771428571429
    0.005771428571429
    0.005771428571429
    0.005771428571429
    0.005771428571429
    0.005771428571429
    0.005771428571429
```

```
norm(x-y)
```

```
ans =       1.511857892036916e-04
```

```
c3=rcond(A)
```

```
c3 =    0.116711903838697
```

Comparing the sensibility perturbations, we can see that the difference of the norms of x and y in hilb(7) is much greater (>10^6) than in magic(7). And comparing the reciprocal condition numbers c3, we can see that it was significantly smaller in hilb(7) than in magic(7).

In other words, when a slight change was made in b, the solution with matrix hilb(7) was much more sensitive to this change than the matrix magic(7) because the condition number of hilb(7) is much greater than in magic(7).

Generally, we can see that a matrix with a large condition number is more sensitive to perturbations than a matrix with a smaller condition number.

Part IV. Area, Graphics, and Volume in Matlab

Exercise 4

```
format
type areavol
```

```
%Creates the function 'areavol'
function D = areavol(A)

% Variable to determine if parallelogram or parallelipiped
isParallelogram = 0;

% Gets the number of columns to detect different vectors
% If it's 2 then it will be parallelogram, otherwise it's a parallelipiped
if isequal(size(A,2),2)
    isParallelogram = 1;
end

% Checks if linearly dependent
r = rank(A); % gets the rank
[rows, ~] = size(A); % gets the number of rows

% rows > rank, so these vectors are not independent.
if rows > r
    if isequal(isParallelogram, 1) % Cannot be built - parallelogram
```

```
        disp('Parallelogram cannot be built.');
    else    % Cannot be built - parallelpiped
        disp('Parallelipiped cannot be built.');
    end
        D = 0;    %Assigns empty output to D and terminates program
        return;
end

D = abs(det(A)); % gets the determinant

if isequal(isParallelogram, 1) % if parallelogram, outputs area
    disp('The area of the parallelogram is');
else % if parallelpiped, outputs volume
    disp('The volume of the parallelpiped is');
end

end
```

```
%(a)
A=randi(10,2)
```

```
A = 2×2
    10     8
     7     8
```

```
D=areavol(A)
```

```
The area of the parallelogram is
D = 24.0000
```

```
%(b)
A=fix(10*rand(3))
```

```
A = 3×3
     3     7     0
     6     0     0
     1     2     8
```

```
D=areavol(A)
```

```
The volume of the parallelpiped is
D = 336
```

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

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

```
D=areavol(A)
```

```
The volume of the parallelpiped is
D = 360
```

```
%(d)
B=randi([-10,10],2,1); A = [B,3*B]
```

```
A = 2×2
     4    12
    -4   -12
```

```
D=areavol(A)
```

```
Parallelogram cannot be built.
D = 0
```

```
%(e)
X=randi([-10,10],3,1);Y=randi([-10,10],3,1);A=[X,Y,X-Y]
```

```
A = 3×3
     9    -2    11
   -10     6   -16
    -1     6    -7
```

```
D=areavol(A)
```

```
Parallelipiped cannot be built.
D = 0
```

Exercise 5

```
R1 = [1, 0; 0, -1]
```

```
R1 = 2×2
     1     0
     0    -1
```

```
R2 = [-1, 0; 0, 1]
```

```
R2 = 2×2
    -1     0
     0     1
```

```
VS = [1, 0; 2, 1]
```

```
VS = 2×2
     1     0
     2     1
```

```
type transf
```

```
function C=transf(A,E)
% Create a function which transforms an image and plots it
C=A*E;
```

```
% Multiply matrices A and E. Here A is the standard matrix of a
% linear transformation;Call the obtained matrix C,C represents an image
% represented by E under the transformation whose standard matrix is A
x=C(1,:);y=C(2,:);
% Call the first row of C x and call the second row y
plot(x,y)
% Plot x against y
v=[-5 5 -5 5];
% Create vector v which will be holding the boundaries of the plot
axis(v)
% Add axes and grid to the plot
end
```

```
E=[0 1 1 0 0;0 0 1 1 0];
A=eye(2);
hold on
```

```
Warning: MATLAB has disabled some advanced graphics rendering features by switching to
software OpenGL. For more information, click here.
```

```
grid on
E=transf(A,E)
```

```
E = 2×5
     0     1     1     0     0
     0     0     1     1     0
```

```
A=VS;
E=transf(A,E)
```

```
E = 2×5
     0     1     1     0     0
     0     2     3     1     0
```

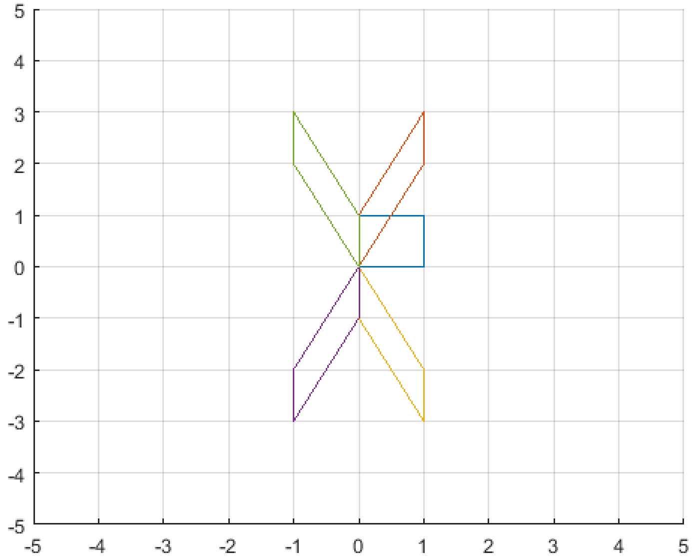
```
A=R1;
E=transf(A,E)
```

```
E = 2×5
     0     1     1     0     0
     0    -2    -3    -1     0
```

```
A=R2;
E=transf(A,E)
```

```
E = 2×5
     0    -1    -1     0     0
     0    -2    -3    -1     0
```

```
A=R1;
E=transf(A,E)
```




```
E = 2×5
     0    -1    -1     0     0
     0     2     3     1     0
```

Part V. Cofactors, Determinants, and Inverse Matrices

Exercise 6

```
type closetozeroroundoff.m
```

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

```
type cofactor.m
```

```
function C=cofactor(a)
format compact
[m,n] = size(a);
C = zeros(m,n);
for i = 1:m
    for j = 1:n
        a_i_j = zeros(m-1, n-1);
```

```
        a_i_j(1:(i-1), 1:(j-1)) = a(1:(i-1), 1:(j-1));
        a_i_j(i:(m-1), 1:(j-1)) = a((i+1):m, 1:(j-1));
        a_i_j(1:(i-1), j:(n-1)) = a(1:(i-1), (j+1):n);
        a_i_j(i:(m-1), j:(n-1)) = a((i+1):m, (j+1):n);
        C(i,j) = (-1)^(i+j) * det(a_i_j);
    end
end
disp( 'the matrix of cofactors is')
disp(C)
end
```

```
type determine.m
```

```
function D = determine(a,C)
D = [];
n = size(a,1);

if(rank(a) < n)
    fprintf('the determinant of the matrix is');
    D = 0
    return
else
   E = zeros(n,2);
   for i = 1:n
        E(i,1) = C(i,:) * transpose(a(i,:));
        E(i,2) = transpose(C(:,i)) * a(:,i);
    end

    for i=1:n
        for j=1:2
            if closetozeroroundoff((E(1,1)-E(i,j)), 7) ~= 0
                disp('Something went wrong!');
                return
            end
        end
    end
    d=det(a);

    if closetozeroroundoff((d-E(1,1)),7)  == 0
        disp('The determinant is ')
        D=E(1,1)
    else
        disp('Check the code!')
    end
end
end
```

```
type inverse.m
```

```
function B = inverse(a,C,D)
B = [];
if(rank(a) < size(a, 1))
    fprintf('a is not invertible');
    return
else
    B = (1/D) * transpose(C);
```

```
end

F = inv(a);

if (closetozeroroundoff(abs(B-F),7) == zeros(size(B)))
    fprintf('the inverse is calculated correctly and it is the matrix');
else
    B
    fprintf('What is wrong?!')
end
end
```

```
%(a)
a=diag([1,2,3,4])
```

```
a = 4×4
     1     0     0     0
     0     2     0     0
     0     0     3     0
     0     0     0     4
```

```
C=cofactor(a);
```

```
the matrix of cofactors is
    24     0     0     0
     0    12     0     0
     0     0     8     0
     0     0     0     6
```

```
D=determine(a,C);
```

```
The determinant is
D = 24
```

```
B=inverse(a,C,D)
```

```
the inverse is calculated correctly and it is the matrix
B = 4×4
    1.0000         0         0         0
         0    0.5000         0         0
         0         0    0.3333         0
         0         0         0    0.2500
```

```
%(b)
a=ones(4)
```

```
a = 4×4
     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1     1
```

```
C=cofactor(a);
```

```
the matrix of cofactors is
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

```
D=determine(a,C);
```

```
the determinant of the matrix is
D = 0
```

```
B=inverse(a,C,D)
```

```
a is not invertible
B =
     []
```

```


%(c)
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
```

```
C=cofactor(a);
```

```
the matrix of cofactors is
   1.0e+05 *
   -0.2503    2.1873   -1.5340    0.2373    0.1397
    2.5935   -1.8915    0.1560   -0.3315    0.2535
   -1.7940   -0.2340    0.1560    0.5460    2.1060
    0.0585    0.6435    0.1560    2.2035   -2.2815
    0.1723    0.0747    1.8460   -1.8753    0.5622
```

```
D=determine(a,C);
```

```
The determinant is
D = 5.0700e+06
```

```
B=inverse(a,C,D)
```

```
the inverse is calculated correctly and it is the matrix
B = 5×5
   -0.0049    0.0512   -0.0354    0.0012    0.0034
    0.0431   -0.0373   -0.0046    0.0127    0.0015
   -0.0303    0.0031    0.0031    0.0031    0.0364
    0.0047   -0.0065    0.0108    0.0435   -0.0370
    0.0028    0.0050    0.0415   -0.0450    0.0111
```

```
%(d)
a=magic(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
```

```
C=cofactor(a);
```

```
the matrix of cofactors is
   1.0e+06 *
    2.5894    2.5894   -1.2947   -2.5894   -2.5894    1.2947
   -0.0000    0.0000   -0.0000         0   -0.0000   -0.0000
   -2.5894   -2.5894    1.2947    2.5894    2.5894   -1.2947
   -2.5894   -2.5894    1.2947    2.5894    2.5894   -1.2947
    0.0000    0.0000   -0.0000    0.0000   -0.0000   -0.0000
    2.5894    2.5894   -1.2947   -2.5894   -2.5894    1.2947
```

```
D=determine(a,C);
```

```
the determinant of the matrix is
D = 0
```

```
B=inverse(a,C,D)
```

```
a is not invertibleB =
     []
```

```
%(e)
a=hilb(4)
```

```
a = 4×4
    1.0000    0.5000    0.3333    0.2500
    0.5000    0.3333    0.2500    0.2000
    0.3333    0.2500    0.2000    0.1667
    0.2500    0.2000    0.1667    0.1429
```

```
C=cofactor(a);
```

```
the matrix of cofactors is
    0.0000   -0.0000    0.0000   -0.0000
   -0.0000    0.0002   -0.0004    0.0003
    0.0000   -0.0004    0.0011   -0.0007
   -0.0000    0.0003   -0.0007    0.0005
```

```
D=determine(a,C);
```

```
The determinant is
```

```
D = 1.6534e-07
```

```
B=inverse(a,C,D)
```

```
the inverse is calculated correctly and it is the matrix
B = 4×4
10³ ×
    0.0160   -0.1200    0.2400   -0.1400
   -0.1200    1.2000   -2.7000    1.6800
    0.2400   -2.7000    6.4800   -4.2000
   -0.1400    1.6800   -4.2000    2.8000
```