2020-2021 BAHAR DÖNEMİ

YMH214
SAYISAL ANALIZ
LAB. DERSİ

6.DERS

Arş. Gör. Alev KAYA

09.04.2021
SAAT:09:00-10:00

Lineer Denklem Sistemlerinin Çözümü

- Doğrudan Yöntemler
- A-Cramer Yöntemi
- B-Gauss Jordan Yöntemi
- LAB: Cramer yöntemi Matlab örnek programı

Lineer Denklem Sistemi

Belirli sayıda bilinmeyen ve belli sayıda denklemden oluşan bir denklem sistemi lineer terimlerden oluşuyorsa bu sistem lineer denklem sistemi olarak adlandırılır.

$$\begin{aligned} 2x - 3y &= 5 \\ -2x + y &= -1 \end{aligned}$$

denklem sistemi iki bilinmeyen içeren lineer bir denklem sistemidir. Genel olarak *n* adet bilinmeyen (x_1, x_2, x_3,….., x_n) içeren lineer bir denklem sistemi aşağıda gösterildiği gibi çık halde veya daha basit olarak matris formunda yazılabilir.

$$\left.\begin{array}{cccccc} a_{11}x_1 + & a_{12}x_2 + & a_{13}x_3 + & \ldots\ldots\ldots & a_{1n}x_n = & b_1 \\ a_{21}x_1 + & a_{22}x_2 + & a_{23}x_3 + & \ldots\ldots\ldots & a_{2n}x_n = & b_2 \\ a_{31}x_1 + & a_{32}x_2 + & a_{33}x_3 + & \ldots\ldots\ldots & a_{3n}x_n = & b_3 \\ \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots \\ a_{n1}x_1 + & a_{n2}x_2 + & a_{n3}x_3 + & \ldots\ldots\ldots & a_{nn}x_n = & b_n \end{array}\right\} \quad A.\vec{x} = \vec{b}$$

Lineer Denklem Sistemi

$$\begin{array}{cccccc}
a_{11}x_1 + & a_{12}x_2 + & a_{13}x_3 + & \ldots\ldots\ldots & a_{1n}x_n = & b_1 \\
a_{21}x_1 + & a_{22}x_2 + & a_{23}x_3 + & \ldots\ldots\ldots & a_{2n}x_n = & b_2 \\
a_{31}x_1 + & a_{32}x_2 + & a_{33}x_3 + & \ldots\ldots\ldots & a_{3n}x_n = & b_3 \\
\ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots \\
a_{m1}x_1 + & a_{m2}x_2 + & a_{m3}x_3 + & \ldots\ldots\ldots & a_{mn}x_n = & b_m
\end{array}$$

$$\overset{A}{\begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix}}_{m\times n} \times \overset{\vec{x}}{\begin{bmatrix} x_1 \\ x_2 \\ \ldots \\ x_n \end{bmatrix}}_{n\times 1} = \overset{\vec{b}}{\begin{bmatrix} b_1 \\ b_2 \\ \ldots \\ b_m \end{bmatrix}}_{m\times 1}$$

A: katsayılar matrisi

x: bilinmeyen vektör

b: sağ taraf vektörü

Örnek

$$\begin{bmatrix} 3 & 2 & 4 \\ 1 & -2 & 0 \\ -1 & 3 & 2 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 14 \\ 7 \\ 2 \end{bmatrix}$$

$$\begin{array}{rcrcrl} 3x_1 & + & 2x_2 & + & 4x_3 & = 14 \\ x_1 & - & 2x_2 & + & 0x_3 & = 7 \\ -x_1 & + & 3x_2 & + & 4x_3 & = 2 \end{array}$$

Lineer Denklem Sistemlerinin Çözümü

Lineer bir denklem sisteminin çözülerek bilinmeyen x_i değerlerinin bulunmasında değişik yöntemler kullanılır. Bu yöntemler 2 grup halinde ayrılabilir.

1-) Analitik (Direkt) Yöntemler: Denklem sisteminin çözümünü matematik anlamda tam olarak veren yöntemlerdir. Bu yöntemler sayesinde doğrudan aranan çözüm elde edilir.

- Matris Tersi Yöntemi
- Cramer Yöntemi
- Eliminasyon Yöntemi
- Gauss Eliminasyon Yöntemi
- Gauss-Jordan Yöntemi
- LU Ayırma Yöntemi

Lineer Denklem Sistemlerinin Çözümü

2-) İteratif (Dolaylı) Yöntemler: Çözümü bulmak için öncelikle tahmini değerlerden başlanır ve adım adım ardışık hesaplamalarla belirli tolerans sınırları içinde aranan çözüme ulaşılır.

- Basit İterasyon (Jacobi) Yöntemi
- Gauss-Seidel Yöntemi
- Rölaksasyon (SOR) Yöntemi

Matrisin Tersi ile Çözümleme

Verilen $A.x=b$ denklem sisteminde, katsayılar matrisinin tersi (A^{-1}) hesaplandığında çözüm vektörü iki matrisin çarpımından elde edilebilir.

$$A.\vec{x} = \vec{b} \qquad \Rightarrow \qquad \vec{x} = A^{-1}.\vec{b}$$

Matris tersinin hesabı için farklı yöntemler vardır. Fakat bu yöntemlerde, eleman sayısı ne kadar fazla ise matris tersini bulmak için daha fazla bilgisayar hafızasına ve hesaplama zamanına ihtiyaç duyulur.

Matrisin Tersi Nasıl Bulunur

Bir matrisin tersini alabilmek için:

- **Matrisin determinantını ve matrisin ekini (adjoint) bulmak gerekir.**

Bir matrisin ekinin bulunması için:

1. **Matrisin bütün elemanlarının eş-çarpanları bulunur.**
2. **Bulunan eş-çarpanlardan yeni bir matris oluşturulur.**
3. **Bu matrisin transpozesi alınır.**

Örnek

$$A=\begin{bmatrix}1&4&2\\3&3&1\\2&0&1\end{bmatrix}$$ matrisinin tersini bulunuz.

1) Matrisin determinantı bulunur.

$$|A|=-13$$

2) Matrisin eş-çarpanları bulunur.

$$A_{11}=(-1)^{1+1}.\begin{vmatrix}3&1\\0&1\end{vmatrix}=1\times3=3$$

$$A_{12}=(-1)^{1+2}.\begin{vmatrix}3&1\\2&1\end{vmatrix}=(-1)(3-2)=-1$$

$$A_{13}=(-1)^{1+3}.\begin{vmatrix}3&3\\2&0\end{vmatrix}=0-6=-6$$

$$A_{21}=(-1)^{2+1}.\begin{vmatrix}4&2\\0&1\end{vmatrix}=(-1)(4-0)=-4$$

$$A_{22}=(-1)^{2+2}.\begin{vmatrix}1&2\\2&1\end{vmatrix}=1-4=-3$$

$$A_{23}=(-1)^{2+3}.\begin{vmatrix}1&4\\2&0\end{vmatrix}=(-1)(0-8)=8$$

$$A_{31}=(-1)^{3+1}.\begin{vmatrix}4&2\\3&1\end{vmatrix}=4-6=-2$$

$$A_{32}=(-1)^{3+2}.\begin{vmatrix}1&2\\3&1\end{vmatrix}=(-1)(1-6)=5$$

$$A_{33}=(-1)^{3+3}.\begin{vmatrix}1&4\\3&3\end{vmatrix}=3-12=-9$$

Örnek-Devam

3) Ek-matris bulunur.

$$Ek(A)=\begin{bmatrix}3 & -1 & -6\\ -4 & -3 & 8\\ -2 & 5 & -9\end{bmatrix}^{T}=\begin{bmatrix}3 & -4 & -2\\ -1 & -3 & 5\\ -6 & 8 & -9\end{bmatrix}$$

4) Matrisin tersi bulunur.

$$A^{-1}=\frac{1}{|A|}.Ek(A)=\frac{1}{-13}\begin{bmatrix}3 & -4 & -2\\ -1 & -3 & 5\\ -6 & 8 & -9\end{bmatrix}$$

MATLAB’da Matrisin Tersini Alma

Matrisin tersini verir.

inv (matris)

tersi hesaplanacak matris

Örnek:

$$A=\begin{bmatrix}1 & 4 & 2\\ 3 & 3 & 1\\ 2 & 0 & 1\end{bmatrix}$$

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

A =

     1     4     2
     3     3     1
     2     0     1

>> inv(A)

ans =

   -0.2308    0.3077    0.1538
    0.0769    0.2308   -0.3846
    0.4615   -0.6154    0.6923

>>
```

Lineer Denklem Sistemlerinin Çözümü

Lineer denklem sistemlerinin Matlab Programı ile çözümü 3 ana başlıkta incelenebilir.

1. Cramer Metodu
2. Matris Tersi Yöntemi
3. “\” Operatörü Yöntemi

Denklem Sayısı ve Bilinmeyen Sayısı Eşit Olan Denklemler

- Bu tip lineer denklemlerin oluşturdukları katsayılar matrisi KARE MATRİS olacaktır.
- Bu tip Denklem sayısı ve bilinmeyen sayısı eşit olan denklem takımlarının çözümün de Cramer Metodu, Matris İnversi yöntemi yada "\" operatörü ile çözüm yöntemi kullanılabilir.
- Cramer yöntemi, denklem sayısı ile bilinmeyen sayısının eşit olması durumunda, katsayılar matrisinin determinantı sıfırdan farklı ise uygulanır.

Cramer Yöntemi ile Çözümleme

Klasik yöntemlerden biri olup, çözüm iki matrisin determinantları oranı olarak elde edilir. Bu yöntemde, *n* tane bilinmeyen içeren

$$A.\vec{x} = \vec{b}$$

şeklindeki lineer denklem sisteminin çözümü;

$$x_i = \frac{|D_i|}{|A|} \qquad (i = 1,2,3,\ldots,n)$$

$|D_i|$: Katsayılar matrisinde (A), *i.* *s*ütun atılıp yerine b vektörünün konması ile elde edilen matrisin determinantıdır.

Bu yöntemde, her biri (n×n) boyutundaki (n+1) adet matrisin determinantının bulunup, bunların oranlanması gerekir. Bundan dolayı işlem sayısı fazla ve çözüm süresi uzundur.

Matrislerde Determinant (2×2)

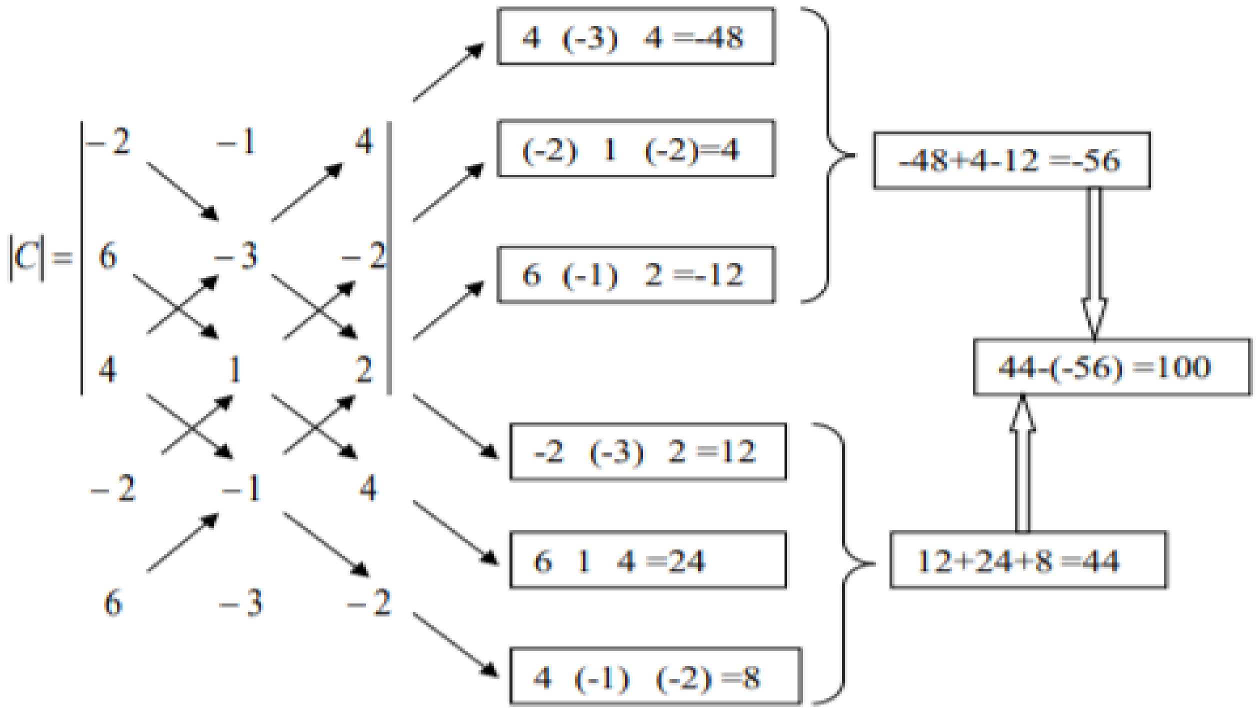
Matrisin köşegenindeki elemanların çarpımından ters köşegenindeki elemanlarının çarpımı birbirinden çıkarılarak bulunur.

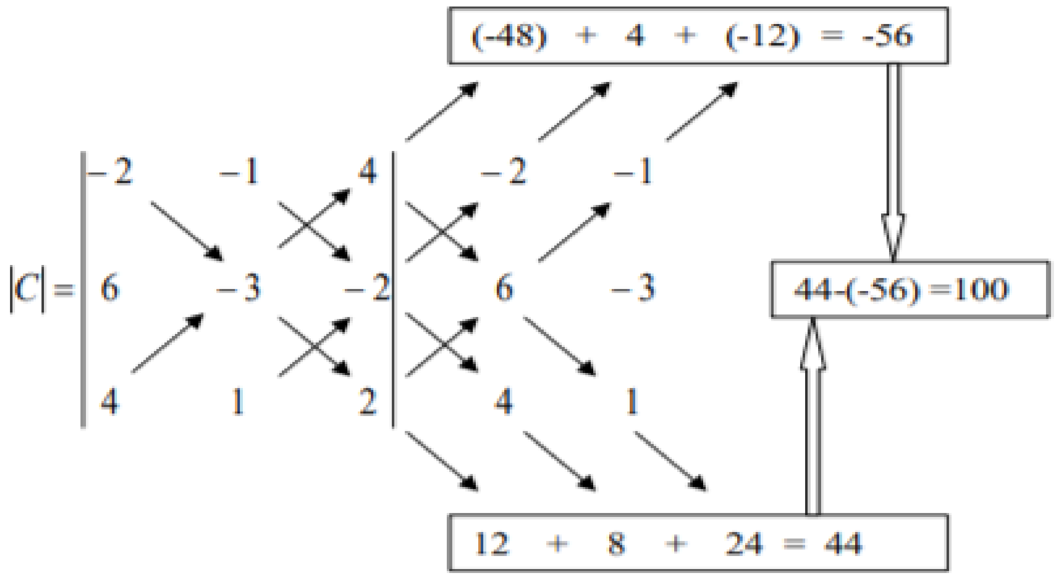
$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}_{2\times 2}$$

$$|A| = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = (a_{11} \times a_{22}) - (a_{12} \times a_{21})$$

Matrislerde Determinant (3×3)

1. ve 2. satırlar/sütunlar, aynı matrise 4. ve 5. satır/sütun olarak eklenir ve oluşan matriste sol köşegen çarpımlarının toplamından sağ köşegen çarpımlarının toplamı çıkarılarak determinant bulunur.





Seçilen Satır/Sütuna göre Determinant Bulma

Minör ve Kofaktör kullanılarak determinant hesaplanır.

- **Bir kare matrisin bulunduğu a_{ij} elemanının i'nci satır ve j'nci sütunu atıldığında geriye kalan M_{ij} matrisinin determinantına a_{ij} elemanının küçüğü (minörü) denir.**
- **$A_{ij} = (-1)^{i+j} |M_{ij}|$ sayısına a_{ij} elemanının eş-çarpanı (kofaktörü) denir.**

Örnek: 3x3 matrisin determinantı,

❶ **Bir satır yada sütun seçilir**

$$A = \begin{bmatrix} \boxed{a_{11} \quad a_{12} \quad a_{13}} \\ a_{21} \quad a_{22} \quad a_{23} \\ a_{31} \quad a_{32} \quad a_{33} \end{bmatrix}$$

❷ **Determinant ifadesi yazılır**

$$|A| = a_{11}\ M_{11} + a_{12}\ M_{12} + a_{13}\ M_{13}$$

❸ **Kofaktörleri hesaplanır**

$$M_{11} = (-1)^{1+1} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix}$$

$$M_{12} = (-1)^{1+2} \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix}$$

$$M_{13} = (-1)^{1+3} \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

MATLAB’da Matrisin Determinantı

Matrisin determinantını verir.
det (matris)

determinantı hesaplanacak matris

Örnek:

$$A = \begin{bmatrix} -2 & -1 & 4 \\ 6 & -3 & -2 \\ 4 & 1 & 2 \end{bmatrix}$$

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

A =

    -2    -1     4
     6    -3    -2
     4     1     2

>> det(A)

ans =

   100
```

$$2x + y - 2z = 0$$

$$x - 2y + z = 5$$

$$x + 3y - 2z = -3$$

Lineer denklemi verilmiş olsun. Burada x, y, z değişkenlerini bulmak için önce değişken katsayılarının matrisi ve eşitliğin sağındaki sayılar sütun vektörü biçiminde yazılmalıdır.

Cramer Metodu

```
>>A=[2 1 -2;1 -2 1;1 3 -2]     %değişken katsayıları A matrisi olarak girildi

>> B=[0;5;-3]                  %eşitliğin sağındaki sayılar sütun vektörü olarak girildi

>>m1=A;                        %A matrisi m1 değişkenine atandı
>>m1(:,1)=B                    %m1 matrisinin birinci sütununa B vektörü yazdırıldı
>>m2=A;                        %A matrisi m2 değişkenine atandı
>>m2(:,2)=B                    %m2 matrisinin ikinci sütununa B vektörü yazdırıldı
>>m3=A;                        %A matrisi m3 değişkenine atandı
>>m3(:,3)=B                    %m3 matrisinin üçüncü sütununa B vektörü yazdırıldı
```

```
>>x_y_z=[det(m1);det(m2);det(m3)] / det(A)  %klasik
çözüm Cramer Metodu

x_y_z=

     2.2

    -0.4

     2.0
```

Matris Tersi Yöntemi

```
>>x_y_z=inv(A) * B          %matris tersini kullanarak çözüm
x_y_z=
      2.2
     -0.4
      2.0
```

NOT: Sadece kare matrislerin tersleri bulunmaktadır.

Gauss Eliminasyon Yöntemi(\)

```
>>x_y_z=A \ B          %\ operatörü kullanarak gauss eliminasyon tekniği ile çözüm

x_y_z=
       2.2
      -0.4
       2.0
```

GAUSS-JORDAN ELİMİNASYONU (GAUSS-JORDAN ELIMINATION)

Gauss-Jordan Eliminasyonunda A matrisi Birim matrise dönüştürülecek şekilde işlemler uygulanılır. 3x3 matris için Gauss eliminasyonu uygulanarak aşağıdaki matris elde edilir:

$$\begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & a_{13}^{(0)} & b_1^{(0)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & b_2^{(1)} \\ 0 & 0 & a_{33}^{(2)} & b_3^{(2)} \end{bmatrix}$$

Son satırda $a_{33}^{(2)}$ ye bölersek

$$\begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & a_{13}^{(0)} & b_1^{(0)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & b_2^{(1)} \\ 0 & 0 & a_{33}^{[1]} = 1 & b_3^{[1]} = b_3^{(2)}/a_{33}^{(2)} \end{bmatrix}$$

elde ederiz. Sonrasında 3. satırı $a_{m3}^{(m-1)} (m = 1, 2)$ ile çarparak 1.ve 2. Satırlardan çıkartırsak

$$\begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & a_{13}^{[1]} = 0 & b_1^{[1]} = b_1^{(0)} - a_{13}^{(0)} b_3^{[1]} \\ 0 & a_{22}^{(1)} & a_{23}^{[1]} = 0 & b_2^{[1]} = b_2^{(1)} - a_{23}^{(1)} b_3^{[1]} \\ 0 & 0 & a_{33}^{[1]} = 1 & b_3^{[1]} \end{bmatrix}$$

sonucunu elde ederiz. Tekrar 2.satırı $a_{22}^{(1)}$ ile bölersek sonucunu elde ederiz.

2.satırı $a_{22}^{(1)}$ ile bölersek

$$\begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & 0 & b_1^{[1]} \\ 0 & a_{22}^{[2]} = 1 & 0 & b_2^{[2]} = b_2^{[1]}/a_{22}^{[1]} \\ 0 & 0 & a_{33}^{[1]} = 1 & b_3^{[1]} \end{bmatrix}$$

sonucunu elde ederiz. 2.satırı $a_{m2}^{(m-1)}(m=1)$ ile çarpıp ilk satırdan çıkartırsak

$$\begin{bmatrix} a_{11}^{(0)} & 0 & 0 & b_1^{[2]} = b_1^{[1]} - a_{12}^{(0)} b_2^{[2]} \\ 0 & 1 & 0 & b_2^{[2]} \\ 0 & 0 & 1 & b_3^{[1]} \end{bmatrix}$$

sonucunu elde ederiz. Burada tekrar ilk satırı $a_{11}^{(0)}$ ile bölersek

$$\begin{bmatrix} 1 & 0 & 0 & b_1^{[3]} = b_1^{[2]}/a_{11}^{(0)} \\ 0 & 1 & 0 & b_2^{[2]} \\ 0 & 0 & 1 & b_3^{[1]} \end{bmatrix}$$

ifadesinde A matrisini birim matrise döndürmüş oluruz. Sağ taraftaki vektörler de çözüm olur.

Editor - C:\Matlab_Dersler\gauss_jordan_elimination.m

gauss_jordan_elimination.m

```
function gauss_jordan_elimination
clc;clear all;warning off;
A=[1 2 5;-1 0 2; 2 1 3];b=[2;0;1];[n m]=size(A);
A=[A b];%genişletilmiş matris

for j=1:n-1 % sütunlar
    for i=j+1:n % satırlar
        if A(i,j)~=0
            lambda=A(i,j)/A(j,j);
            A(i,j:n+1)=A(i,j:n+1)-lambda*A(j,j:n+1);
        end
    end
    A(j,1:n+1)=A(j,1:n+1)/A(j,j);
end
A(n,n:n+1)=A(n,n:n+1)/A(n,n);

for j=n:-1:1
    for i=j-1:-1:1
        lambda=A(i,j);
        A(i,i:n+1)=A(i,i:n+1)-lambda*A(j,i:n+1);
    end
end
x=A(:,n+1);
fprintf('%5s %3s\n','Çözüm','-x-');
fprintf('%12.8f\n',x)
```

VARIABLE

Editor - C:\Matlab_Dersler\gauss_jordan_elim_2.m

gauss_jordan_elimination.m | gauss_jordan_elim_2.m | gaus_gaus_jordan.m | gauss_jordan_elimination.asv | +

```
function [x,err]=gauss_jordan_elim(A,b)
A = [1 1 1;2 3 5; 4 0 5] % input for augmented matrix A
b = [5 ; 8; 2] % intput for matrix B
[n,m]=size(A); % finding the size of matrix A
err =0; % calculation of error
x=zeros(n,1); % calling fuction zero
if n ~= m
    disp('error: n~=m'); % displaying error if found
    err = 1;
end % end of the scope of if
if length(b) ~= n  % finding the legth of matrix B
    disp('error: wrong size of b'); % displaying error, if found
    err = 2;
else
    if size(b,2) ~= 1
        b=b';
    end % end of the scope of if-else
    if size(b,2) ~= 1
        disp('error: b is a matrix'); % displaying erron in matrix B
        err = 3;
    end
end
if err == 0
    Aa=[A,b];
    for i=1:n
```

Editor - C:\Matlab_Dersler\gauss_jordan_elim_2.m

gauss_jordan_elimination.m | gauss_jordan_elim_2.m | gaus_gaus_jordan.m | gauss_jordan_elimination.asv | +

```
        for i=1:n
            [Aa(i:n,i:n+1),err]=gauss_pivot(Aa(i:n,i:n+1));
            if err == 0
                Aa(1:n,i:n+1)=gauss_jordan_step(Aa(1:n,i:n+1),i);
            end
        end
        x=Aa(:,n+1);
    end
    A=0;
    function A1=gauss_jordan_step(A,i)  % calling of fuction function

     [n,m]=size(A); % determination of size of matrix A
    A1=A; % assigning A to A1
    s=A1(i,1);
    A1(i,:) = A(i,:)/s;
    k=[[1:i-1],[i+1:n]];
    for j=k
        s=A1(j,1);
        A1(j,:)=A1(j,:)-A1(i,:)*s;
    end % end of for loop
    function [A1,err]=gauss_pivot(A)  % calling of fucntion
     [n,m]=size(A); % finding the size of matrix A
    A1=A; % process of assigning
    err = 0; % error flag
    if A1(1,1) == 0
        check = logical(1); % logical(1) - TRUE
```

SIMULINK ENVIRONMENT

Editor - C:\Matlab_Dersler\gauss_jordan_elim_2.m

gauss_jordan_elimination.m | gauss_jordan_elim_2.m | gaus_gaus_jordan.m | gauss_jordan_elimination.asv

```
        s=A1(j,1);
        A1(j,:)=A1(j,:)-A1(i,:)*s;
    end % end of for loop
    function [A1,err]=gauss_pivot(A) % calling of fucntion
    [n,m]=size(A); % finding the size of matrix A
    A1=A; % process of assigning
    err = 0; % error flag
    if A1(1,1) == 0
        check = logical(1); % logical(1) - TRUE
        i = 1;
        while check
            i = i + 1;
            if i > n
                disp('error: matrix is singular');
                err = 1;
                check = logical(0);
            else
                if A(i,1) ~= 0 & check
                    check = logical(0);
                    b=A1(i,:);        % process to change row 1 to i
                    A1(i,:)=A1(1,:);
                    A1(1,:)=b;
                end
            end
        end
    end
```

Editor - C:\Matlab_Dersler\gaus_gaus_jordan.m

gauss_jordan_elimination.m | gauss_jordan_elim_2.m | gaus_gaus_jordan.m | gauss_jordan_elimination.asv | +

```
% Code from "Gauss elimination and Gauss Jordan methods using MATLAB"
% https://www.youtube.com/watch?v=kMApKEKisKE

a = [3 4 -2 2 2
     4 9 -3 5 8
     -2 -3 7 6 10
     1 4 6 7 2];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Gauss elimination method [m,n)=size(a);
[m,n]=size(a);
for j=1:m-1
    for z=2:m
        if a(j,j)==0
            t=a(j,:);a(j,:)=a(z,:);
            a(z,:)=t;
        end
    end
    for i=j+1:m
        a(i,:)=a(i,:)-a(j,:)*(a(i,j)/a(j,j));
    end
end
x=zeros(1,m);
for s=m:-1:1
```

Editor - C:\Matlab_Dersler\gaus_gaus_jordan.m

gauss_jordan_elimination.m | gauss_jordan_elim_2.m | gaus_gaus_jordan.m | gauss_jordan_elimination.asv

```
for s=m:-1:1
    c=0;
    for k=2:m
        c=c+a(s,k)*x(k);
    end
    x(s)=(a(s,n)-c)/a(s,s);
end
disp('Gauss elimination method:');
a
x'
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Gauss-Jordan method
[m,n]=size(a);
for j=1:m-1
    for z=2:m
        if a(j,j)==0
            t=a(1,:);a(1,:)=a(z,:);
            a(z,:)=t;
        end
    end
    for i=j+1:m
        a(i,:)=a(i,:)-a(j,:)*(a(i,j)/a(j,j));
    end
end
```

```
            if a(j,j)==0
                t=a(1,:);a(1,:)=a(z,:);
                a(z,:)=t;
            end
        end
        for i=j+1:m
            a(i,:)=a(i,:)-a(j,:)*(a(i,j)/a(j,j));
        end
    end

    for j=m:-1:2
        for i=j-1:-1:1
            a(i,:)=a(i,:)-a(j,:)*(a(i,j)/a(j,j));
        end
    end

    for s=1:m
        a(s,:)=a(s,:)/a(s,s);
        x(s)=a(s,n);
    end
    disp('Gauss-Jordan method:');
    a
    x'
```