Contents

problema 1 punctul a tema 3

```
A=[0 1 1;2 1 5; 4 2 1]

b=[3;5;1]

fprintf('solutia este\n')
gaussPivTot(A,b);
type('gaussPivTot');
```

```
A =


     0     1     1

     2     1     5

     4     2     1




b =


     3

     5

     1


solutia este

    -1


     2


     1



function [x] = gaussPivTot(A,b)

%GaussPivPart rezolva sisteme patratice


%Synopsis:      x=gaussPivTot(A,b)
```

```
%Input:         A=matricea asociata sistemului
%               b=vectorul termenilor liberi

%Output:        x=solutia sistemului

%-----Verificare matrice patratica
[n,m]=size(A);
if n~=m
    error('Matricea nu este patratica');
    x=[];
    return
end
%-----Verificare daca e vector coloana si daca nu e, il traspunem ca sa fie
[n1,m1]=size(b);
if m1~=1
    b=transpose(b);
end

nr=length(b);

if n~=nr
    warning('Vectorul nu are aceeasi dimensiune cu matricea A')
    x=[];
    return;
end

%construim matricea extinsa

A=[A,b];
for i=1:n
index(i)=i;
end
for k=1:n-1
```

```
    max=-1;
    for i=k:n
        for j=k:n
            if abs(A(i,j))>max;
                max=abs(A(i,j));
                p=i;m=j;
            end
        end
    end
    if A(p,m)==0
        warning('Sist incomp. sau comp. nedet')
        x=[];
        return
    end
    if p~=k
      A([p,k],:)=A([k,p],:);
    end
      if m~=k
          A(:,[m,k])=A(:,[k,m]);
          index([m,k])=index([k,m]);
      end

      for l=k+1:n
          M(l,k)=A(l,k)/A(k,k);
          A(l,:)=A(l,:)-M(l,k)*A(k,:);
      end
end

    if A(n,n)==0
        warning('Sist. incomp. sau comp. nedet.');
        return
    end
    y=subsDesc(A(:,1:n),A(:,n+1));
    for j=1:n
```

```
            x(index(j))=y(j);

    end

    for i=1:n

        disp(x(i));

    end


end
```

problema 1 punctul b tema 3

```
A=[0 1 -2;1 -1 1; 1 0 -1]

b=[4;6;2]

fprintf('solutia este\n')
gaussPivTot(A,b);
type('gaussPivTot')
```

```
A =


     0     1    -2

     1    -1     1

     1     0    -1




b =


     4

     6

     2


solutia este

Warning: Sist. incomp. sau comp. nedet.


function [x] = gaussPivTot(A,b)

%GaussPivPart rezolva sisteme patratice


%Synopsis:      x=gaussPivTot(A,b)


%Input:         A=matricea asociata sistemului
```

```
%               b=vectorul termenilor liberi

%Output:        x=solutia sistemului

%-----Verificare matrice patratica
[n,m]=size(A);
if n~=m
    error('Matricea nu este patratica');
    x=[];
    return
end
%-----Verificare daca e vector coloana si daca nu e, il traspunem ca sa fie
[n1,m1]=size(b);
if m1~=1
    b=transpose(b);
end

nr=length(b);

if n~=nr
    warning('Vectorul nu are aceeasi dimensiune cu matricea A')
    x=[];
    return;
end

%construim matricea extinsa

A=[A,b];
for i=1:n
index(i)=i;
end
for k=1:n-1
    max=-1;
    for i=k:n
```

```
        for j=k:n
            if abs(A(i,j))>max;
                max=abs(A(i,j));
                p=i;m=j;
            end
        end
    end
    if A(p,m)==0
        warning('Sist incomp. sau comp. nedet')
        x=[];
        return
    end
    if p~=k
      A([p,k],:)=A([k,p],:);
    end
      if m~=k
          A(:,[m,k])=A(:,[k,m]);
          index([m,k])=index([k,m]);
      end

      for l=k+1:n
          M(l,k)=A(l,k)/A(k,k);
          A(l,:)=A(l,:)-M(l,k)*A(k,:);
      end
end

    if A(n,n)==0
        warning('Sist. incomp. sau comp. nedet.');
        return
    end
    y=subsDesc(A(:,1:n),A(:,n+1));
    for j=1:n
            x(index(j))=y(j);
    end
```

```
    for i=1:n
        disp(x(i));
    end


end
```

problema 2 varianta 5 tema 3

```
A=matrice(10)

b=solutie(10)

fprintf('solutia este\n')
gaussPivTot(A,b);
type('gaussPivTot')
```

```
A =


    21    -7     0     0     0     0     0     0     0     0
    -7    21    -7     0     0     0     0     0     0     0
     0    -7    21    -7     0     0     0     0     0     0
     0     0    -7    21    -7     0     0     0     0     0
     0     0     0    -7    21    -7     0     0     0     0
     0     0     0     0    -7    21    -7     0     0     0
     0     0     0     0     0    -7    21    -7     0     0
     0     0     0     0     0     0    -7    21    -7     0
     0     0     0     0     0     0     0    -7    21    -7
     0     0     0     0     0     0     0     0    -7    21



b =


     2
     1
     1
     1
     1
     1
     1
     1
```

```
     1

     2


solutia este

    0.1429


    0.1429


    0.1429


    0.1429


    0.1429


    0.1429


    0.1429


    0.1429


    0.1429


    0.1429
```

```
function [x] = gaussPivTot(A,b)

%GaussPivPart rezolva sisteme patratice


%Synopsis:      x=gaussPivTot(A,b)


%Input:         A=matricea asociata sistemului

%               b=vectorul termenilor liberi


%Output:        x=solutia sistemului
```

```
%-----Verificare matrice patratica
[n,m]=size(A);
if n~=m
    error('Matricea nu este patratica');
    x=[];
    return
end
%-----Verificare daca e vector coloana si daca nu e, il traspunem ca sa fie
[n1,m1]=size(b);
if m1~=1
    b=transpose(b);
end

nr=length(b);

if n~=nr
    warning('Vectorul nu are aceeasi dimensiune cu matricea A')
    x=[];
    return;
end

%construim matricea extinsa

A=[A,b];
for i=1:n
index(i)=i;
end
for k=1:n-1
    max=-1;
    for i=k:n
        for j=k:n
            if abs(A(i,j))>max;
                max=abs(A(i,j));
```

```
                p=i;m=j;
            end
        end
    end
    if A(p,m)==0
        warning('Sist incomp. sau comp. nedet')
        x=[];
        return
    end
    if p~=k
      A([p,k],:)=A([k,p],:);
    end
      if m~=k
          A(:,[m,k])=A(:,[k,m]);
          index([m,k])=index([k,m]);
      end

      for l=k+1:n
          M(l,k)=A(l,k)/A(k,k);
          A(l,:)=A(l,:)-M(l,k)*A(k,:);
      end
end

    if A(n,n)==0
        warning('Sist. incomp. sau comp. nedet.');
        return
    end
    y=subsDesc(A(:,1:n),A(:,n+1));
    for j=1:n
            x(index(j))=y(j);
    end
    for i=1:n
        disp(x(i));
    end
```

```
end
```

problema 3 tema 3

```
A=matrice(10)

b=solutie(10)

InvDet(A,b);

type('InvDet')
```

```
A =


    21    -7     0     0     0     0     0     0     0     0
    -7    21    -7     0     0     0     0     0     0     0
     0    -7    21    -7     0     0     0     0     0     0
     0     0    -7    21    -7     0     0     0     0     0
     0     0     0    -7    21    -7     0     0     0     0
     0     0     0     0    -7    21    -7     0     0     0
     0     0     0     0     0    -7    21    -7     0     0
     0     0     0     0     0     0    -7    21    -7     0
     0     0     0     0     0     0     0    -7    21    -7
     0     0     0     0     0     0     0     0    -7    21


b =


     2
     1
     1
     1
     1
     1
     1
     1
     1
     2
```

inversa matricei este

: Columns 1 through 7

0.0546	0.0208	0.0080	0.0030	0.0012	0.0004	0.0002
0.0208	0.0625	0.0239	0.0091	0.0035	0.0013	0.0005
0.0080	0.0239	0.0637	0.0243	0.0093	0.0035	0.0014
0.0030	0.0091	0.0243	0.0639	0.0244	0.0093	0.0036
0.0012	0.0035	0.0093	0.0244	0.0639	0.0244	0.0093
0.0004	0.0013	0.0035	0.0093	0.0244	0.0639	0.0244
0.0002	0.0005	0.0014	0.0036	0.0093	0.0244	0.0639
0.0001	0.0002	0.0005	0.0014	0.0035	0.0093	0.0243
0.0000	0.0001	0.0002	0.0005	0.0013	0.0035	0.0091
0.0000	0.0000	0.0001	0.0002	0.0004	0.0012	0.0030

  Columns 8 through 10

0.0001	0.0000	0.0000
0.0002	0.0001	0.0000
0.0005	0.0002	0.0001
0.0014	0.0005	0.0002
0.0035	0.0013	0.0004
0.0093	0.0035	0.0012
0.0243	0.0091	0.0030
0.0637	0.0239	0.0080
0.0239	0.0625	0.0208
0.0080	0.0208	0.0546

solutia sistemului este

    0.1429

    0.1429

    0.1429

    0.1429

    0.1429

```
    0.1429

    0.1429

    0.1429

    0.1429

    0.1429


determinantul este:   5.0029e+12
```

```
function [Det,x] = InvDet(A,b)
%Inv det cu Gauss Piv Totala calc inv

%Synopsis:      Det,x=InvDet(A,b)

%Input:         A=matricea asociata sistemului
%
%Output:        x=matricea inversa cu ajutorul careia rezolvam
%               sistemul Ax=b
%               Det=determinantul matricei asociate


%-----Verificare matrice patratica
[n,m]=size(A);
if n~=m
    warning('Matricea nu este patratica');
    x=[];
    return
end


%construim matricea extinsa
A=[A,eye(n)];
Det=1;
for i=1:n
```

```
index(i)=i;
end
for k=1:n-1
    max=-1;
    for i=k:n
        for j=k:n
            if abs(A(i,j))>max;
                max=abs(A(i,j));
                p=i;m=j;
            end
        end
    end
    if A(p,m)==0
        warning('Sist incomp. sau comp. nedet')
        x=[];
        return
    end
    if p~=k
      A([p,k],:)=A([k,p],:);
      Det=Det*(-1);
    end
      if m~=k
          A(:,[m,k])=A(:,[k,m]);
          index([m,k])=index([k,m]);
          Det=Det*(-1);
      end

      for l=k+1:n
          M(l,k)=A(l,k)/A(k,k);
          A(l,:)=A(l,:)-M(l,k)*A(k,:);
      end

end
 for i=1:n
```

```
     Det=Det*A(i,i);
 end
    if A(n,n)==0
        warning('Sist. incomp. sau comp. nedet.');
        return
    end
    for i=1:n
        y=subsDesc(A(:,1:n),A(:,n+i));
        for j=1:n
            x(index(j))=y(j);
        end
        A=[A,transpose(x)];
    end
    fprintf('inversa matricei este\n:');
    x=A(:,(2*n+1):end);
    disp(x);
    fprintf('solutia sistemului este\n');
    disp(A(:,(2*n+1):end)*b);
    fprintf('determinantul este:');
    disp(Det);



end
```

Published with MATLAB® R2017b