# Priedai

## D priedas. Gelžbetoninio rėmo optimizavimo programinis kodas

### D1. 3.2 poskyrio skaitinis eksperimentas. Uždavinio (3.40)–(3.46) sprendimo programini kodas.

```
clc
clear all

%%% GB remo optimizavimas%%%

% Columns length
L1=4;

% Beams length
B1=7;

% Equilibrium matrix (m,n)
A=zeros(18,24);
A(1,2)=1; A(1,4)=1; A(1,19)=1;
A(2,5)=1; A(2,7)=1;
```

```
A(3,8)=1; A(3,10)=1;
A(4,11)=1; A(4,13)=1;
A(5,14)=1; A(5,16)=1; A(5,23)=1;
A(6,20)=1; A(6,22)=1;
A(7,1)=-1/(L1); A(7,2)=-1/(L1); A(7,4)=1/(L1);
A(7,5)=1/(L1); A(7,21)=-1;
A(8,19)=1/B1; A(8,20)=1/B1; A(8,3)=-1; A(8,6)=1;
A(9,4)=-1/L1; A(9,5)=-1/L1; A(9,9)=-1;
A(10,7)=1/B1; A(10,8)=1/B1; A(10,6)=-1;
A(11,9)=1; A(11,12)=-1;
A(12,7)=-1/B1; A(12,8)=-1/B1; A(12,10)=1/B1; A(12,11)=1/B1;
A(13,13)=-1/L1; A(13,14)=-1/L1; A(13,12)=1;
A(14,10)=-1/B1; A(14,11)=-1/B1; A(14,15)=-1;
A(15,13)=1/L1; A(15,14)=1/L1; A(15,16)=-1/L1; A(15,17)=-
1/L1; A(15,24)=1;
A(16,22)=-1/B1; A(16,23)=-1/B1; A(16,15)=1; A(16,18)=-1;
A(17,21)=1; A(17,24)=-1;
A(18,19)=-1/B1; A(18,20)=-1/B1; A(18,22)=1/B1;
A(18,23)=1/B1;

global ilg ilgg Fi Gama G Z W BV;
%Elements lenght
J1=4; J2=4; J3=7; J4=7; J5=4; J6=4; J7=7; J8=7;
ilgg=[8 8 14 14]; % length of the elements with the same
limit force [M01 M02 M03 M04]
ilg=[4 4 7 7 4 4 7 7];

%% 1. Innitial section properties / design according to the
EC2
M01=195;
M02=79;
M03=79;
M04=144;

N01=64;
N02=22;
N03=23;
N04=1;

%Lover bound of the limit bending moment
M01lb=5;
M02lb=5;
M03lb=5;
M04lb=5;
```

```
%Loads

F1sup=160; F1inf=0;
F2sup=100; F2inf=-120;

F1=[0 0 0 0 0 0 F2sup 0 1.1*F2sup 0 0 0.6*F1sup 0.55*F2sup
0 0.5*F2sup 0 0 F1sup]';
F2=[0 0 0 0 0 0 0.5*F2inf 0 0.55*F2inf 0 0 0.6*F1sup
1.2*F2inf 0 F2inf 0 0 F1sup]';
F3=[0 0 0 0 0 0 F2sup 0 1.1*F2sup 0 0 0 0.55*F2sup 0
0.5*F2sup 0 0 0]';
F4=[0 0 0 0 0 0 0.5*F2inf 0 0.55*F2inf 0 0 0 1.2*F2inf 0
F2inf 0 0 0]';



% Starting point for nonlinear problem
x0=zeros(644,1); %(yielding condition equations number *
sections number * load locus appexes nr) + nr of limit
bending moments = = = e.g. (10*16*4)+4=644

% Section properties
b=[0.4 0.4 0.4 0.4]; % Width of the section, [m]
h=[0.5 0.5 0.6 0.4]; % Height of the section, [m]

a1=0.04; % Safety zone of the As1 reinforcement, [m]
a2=0.04; % Safety zone of the As2 reinforcement, [m]
d=h-a1;

% Material properties
 Ecm=29e6; %Concrete Young modulus GPa->kN/m2
 Es=200e6; %Steel Young modulus GPa(e6)->[kN/m2]
 nsc=Es/Ecm; %The modular ratio nsc = Es /Ec. Multiplying
the area of the reinforcement bars by the modular ratio re-
sults in the area concrete being effectively transformed
into an equivalent area of reinforcement

fcd=20; % C30/37, [MPa]
fyd=350; % B400, [MPa]

xiu=0.534; % Limit value of the relative height of the com-
pression zone. "According to the table [ ]"
xu=xiu*d;

ro_min=0.0005 % Minimal reinforcing coeficient
```

```
ro_max=0.04 % Maximal reinforcing coeficient

figure;
%% START of the CYCLE
iteration=9;
results=zeros(iteration,7);
for iter = 1:iteration
% Each iteration with a new M0 and N0
iter

M0=[M01 M02 M03 M04]
N0=[N01 N02 N03 N04]
%%%%%%%%%%%%%%%%%%%%%%%%%

% DESIGN of reinforcement
for i=1:4;
e(i)=M0(i)/N0(i); % Excentricity, [m]
e1(i)=e(i)+(h(i)/2)-a1;
end;

eta=1; % Coefficient, usually is taken =1.0

for i=1:4;
x(i)=(N0(i)*1e-3)/(eta*fcd*b(i)); % [m]
end;



ks=1; % kapa_s

%% BIG EXCENTRICITY acc. to EC2 requirements %%
for i=1:4;
if x(i)<xu(i) % Big excentricity

    z(i)=d(i)-(x(i)/2); % Symetric reinforcement
    disp('Big excentricity case, x<xu, As1=As2')

    %for i=1:4;
    As(i)=((N0(i)*1e-3)*(e1(i)-z(i)))/(fyd*(d(i)-a2)); %
As1=As2, [m2]
    %end
    As1=As(i)
    As2=As(i)
    disp('Reinforcing coefficient')
```

```
    ro(i)=(As1+As2)/b(i)*h(i)

    if ro(i)<ro_min
        ro(i)=ro_min
        disp('ro=ro_min')

    elseif ro(i)>ro_max
        disp('Check As, ro_max exceeded')
    end

elseif x(i)==xu(i) % Nonsymetric reinforcement
    disp('Big excentricity case, simetric reinforcing,
x=xu')
    As2=((N0(i)*1e-3)*e1(i)-eta*fcd*b(i)*x(i)*(d(i)-
x(i)/2))/(fyd*(d(i)-a2)) % [m2]
    As1=(eta*fcd*b(i)*x(i)+fyd*As2-(N0(i)*1e-3))/(ks*fyd) %
[m2]
    ro(i)=(As1+As2)/b(i)*h(i)

         if ro(i)<ro_min
            ro(i)=ro_min
            disp('ro=ro_min')
        elseif ro(i)>ro_max
            disp('Check As1 or As2, ro_max exceeded')
        end;

    if As1>0;
        disp('Big excentricity case, x=xu, As1>0')


    else % As1<=0, reinforcement need to be design accord-
ing to SMALL EXCENTRICITY rooles
        disp('Big excentricity case, but As1<=0, Need to
design according to SMALL excentricity rooles, then
As1=As2')
        x_st(i)=a2+sqrt(a2^2+(2*(N0(i)*1e-3)*(d(i)-a1-
e1(i)))/(eta*fcd*b(i)))
        As(i)=((N0(i)*1e-3)-eta*fcd*x_st(i)*b(i)*(d(i)-
x_st(i)/2))/(fyd*(d(i)-a2));
        As1=As(i)
        As2=As(i)
        ro(i)=(As1+As2)/b(i)*h(i)

        if ro(i)<ro_min
```

```
            ro(i)=ro_min
            disp('ro=ro_min')
            disp('Because As1<=0, design has been performed
according to SMALL excentricity rooles')
       elseif ro(i)>ro_max
            disp('Check As1 or As2, ro_max exceeded')
       end;

    end;

else %small exc

    x_st(i)=a2+sqrt(a2^2+(2*(N0(i)*1e-3)*(d(i)-a1-
e1(i)))/(eta*fcd*b(i)))

     if x(i)<h(i)
         disp('Small excentricity case, x>xu, x<h')
         As2=((N0(i)*1e-3)*e1(i)-
eta*fcd*b(i)*x_st(i)*(d(i)-x_st(i)/2))/(fyd*(d(i)-a2))
         As1=(eta*fcd*b(i)*x_st(i)+fyd*As2-(N0(i)*1e-
3))/(ks*fyd)
        ro(i)=(As1+As2)/b(i)*h(i)
     else
         disp('Small excentricity case, x>xu, x>=h')
         As2=((N0(i)*1e-3)*e1(i)-eta*fcd*b(i)*h(i)*(d(i)-
h(i)/2))/(fyd*(d(i)-a2))
         As1=(eta*fcd*b(i)*d(i)+fyd*As2-(N0(i)*1e-3))/fyd
         ro(i)=(As1+As2)/b(i)*h(i)
     end;

        if ro(i)<ro_min
            ro(i)=ro_min
            disp('ro=ro_min, Need to recalculate As1 ac-
cording to ro_min')
        elseif ro(i)>ro_max
            disp('Check As1 or As2, ro_max exceeded')
        end;

end;
Ass(i,:)=[As1 As2]
rez(i,:)=[i As1*1e4 As2*1e4 ro(i)]

end;
```

```
 %% Rule of mixtures



 Vf1=(Ass(1,1)+Ass(1,2))*ilg(1); %volume of fibers (rein-
forcement) in each element
 Vf2=(Ass(2,1)+Ass(2,2))*ilg(2);
 Vf3=(Ass(3,1)+Ass(3,2))*ilg(3);
 Vf4=(Ass(3,1)+Ass(3,2))*ilg(4);
 Vf5=(Ass(2,1)+Ass(2,2))*ilg(5);
 Vf6=(Ass(1,1)+Ass(1,2))*ilg(6);
 Vf7=(Ass(4,1)+Ass(4,2))*ilg(7);
 Vf8=(Ass(4,1)+Ass(4,2))*ilg(8);

 Vf=[Vf1 Vf2 Vf3 Vf4 Vf5 Vf6 Vf7 Vf8];

 Vm1=(b(1)*h(1)*ilg(1))-Vf1; %volume of matrix (w/o rein-
forcement) in each element
 Vm2=(b(2)*h(2)*ilg(2))-Vf2;
 Vm3=(b(3)*h(3)*ilg(3))-Vf3;
 Vm4=(b(3)*h(3)*ilg(4))-Vf4;
 Vm5=(b(2)*h(2)*ilg(5))-Vf5;
 Vm6=(b(1)*h(1)*ilg(6))-Vf6;
 Vm7=(b(4)*h(4)*ilg(7))-Vf7;
 Vm8=(b(4)*h(4)*ilg(8))-Vf8;

 Vm=[Vm1 Vm2 Vm3 Vm4 Vm5 Vm6 Vm7 Vm8]

 for i=1:8;
     f(i)=Vf(i)/(Vf(i)+Vm(i)); %Volume fraction
 end;



 for i=1:8
 Ec(i)=f(i)*Es+(1-f(i))*Ecm;
 end;
disp('Young modulus of composite, [kN/m2]')
disp(Ec)

%The second moment of area of the transformed section
Ic1=((b(1)*x(1)^3)/3)+(nsc-1)*Ass(1,2)*(x(1)-
a2)^2+nsc*Ass(1,1)*(d(1)-x(1))^2;
```

```
Ic2=((b(2)*x(2)^3)/3)+(nsc-1)*Ass(2,2)*(x(2)-
a2)^2+nsc*Ass(2,1)*(d(2)-x(2))^2;
Ic3=((b(3)*x(3)^3)/3)+(nsc-1)*Ass(3,2)*(x(3)-
a2)^2+nsc*Ass(3,1)*(d(3)-x(3))^2;
Ic4=((b(4)*x(4)^3)/3)+(nsc-1)*Ass(4,2)*(x(4)-
a2)^2+nsc*Ass(4,1)*(d(4)-x(4))^2;

Ic=[Ic1 Ic2 Ic3 Ic4]

%The area of the transformed section
Aa1=b(1)*x(1)-Ass(1,2)+nsc*Ass(1,2)+nsc*Ass(1,1);
Aa2=b(2)*x(2)-Ass(2,2)+nsc*Ass(2,2)+nsc*Ass(2,1);
Aa3=b(3)*x(3)-Ass(3,2)+nsc*Ass(3,2)+nsc*Ass(3,1);
Aa4=b(4)*x(4)-Ass(4,2)+nsc*Ass(4,2)+nsc*Ass(4,1);

Aa=[Aa1 Aa2 Aa3 Aa4]

% Felxability matrix
dd=@(L,I,Ak,E) [L/(3*E*I) L/(6*E*I) 0; L/(6*E*I) L/(3*E*I)
0; 0 0 L/(E*Ak)];
D=blkdiag(dd(J1,Ic1,Aa1,Ec(1)), dd(J2,Ic2,Aa2,Ec(2)),
dd(J3,Ic3,Aa3,Ec(3)), dd(J4,Ic3,Aa3,Ec(3)),
dd(J5,Ic2,Aa2,Ec(2)), dd(J6,Ic1,Aa1,Ec(1)),
dd(J7,Ic4,Aa4,Ec(4)), dd(J8,Ic4,Aa4,Ec(4)));

% Elastic response influence matrixes
alfa=inv(D)*A'*inv(A*inv(D)*A');
beta=inv(A*inv(D)*A');

%Elastic response
Se1=alfa*F1;
Se2=alfa*F2;
Se3=alfa*F3;
Se4=alfa*F4;

ue1=beta*F1;
ue2=beta*F2;
ue3=beta*F3;
ue4=beta*F4;

%Yielding conditions matrix for a single element (10 equa-
tions for a single section, total 20 per element)
fi=@(As1, As2, b, d) [1 0 (a1/2 - d/2);
    1 0  (a1/2 - d/2 + (d*xiu)/2);
```

```
    1 0  ((d*xiu)/2 - d/4 - a1/4);
    1 0 (-((a1 - d)*(8*As1*fyd - a1*b*eta*fcd +
b*d*eta*fcd))/(4*(4*As1*fyd - a1*b*eta*fcd +
b*d*eta*fcd)));
    1 0 (d/2);
    -1 0 (a2/2 - d/2);
    -1 0  (a2/2 - d/2 + (d*xiu)/2);
    -1 0  ((d*xiu)/2 - d/4 - a1/4);
    -1 0 (-((a2 - d)*(8*As2*fyd - a2*b*eta*fcd +
b*d*eta*fcd))/(4*(4*As2*fyd - a2*b*eta*fcd +
b*d*eta*fcd)));
    -1 0 (d/2); % for the 1st section

    0 1 (a1/2 - d/2);
    0 1  (a1/2 - d/2 + (d*xiu)/2);
    0 1  ((d*xiu)/2 - d/4 - a1/4);
    0 1 (-((a1 - d)*(8*As1*fyd - a1*b*eta*fcd +
b*d*eta*fcd))/(4*(4*As1*fyd - a1*b*eta*fcd +
b*d*eta*fcd)));
    0 1 (d/2);
    0 -1 (a2/2 - d/2);
    0 -1  (a2/2 - d/2 + (d*xiu)/2);
    0 -1  ((d*xiu)/2 - d/4 - a2/4);
    0 -1 (-((a2 - d)*(8*As2*fyd - a2*b*eta*fcd +
b*d*eta*fcd))/(4*(4*As2*fyd - a2*b*eta*fcd +
b*d*eta*fcd)));
    0 -1 (d/2);];%for the 2nd section

%Yielding condition matrix of whole structure
Fi = blkdiag(fi(Ass(1,1), Ass(1,2), b(1), d(1)),
fi(Ass(2,1), Ass(2,2), b(2), d(2)), fi(Ass(3,1), Ass(3,2),
b(3), d(3)), fi(Ass(3,1), Ass(3,2), b(3), d(3)),
fi(Ass(2,1), Ass(2,2), b(2), d(2)), fi(Ass(1,1), Ass(1,2),
b(1), d(1)), fi(Ass(4,1), Ass(4,2), b(4), d(4)),
fi(Ass(4,1), Ass(4,2), b(4), d(4)));

%Configuration matrix

%Vieno elemento (su dviem pjuviais) takumo salygu laisvuju
nariu vektorius
Bv=@(As1,As2,b,d)[As1*fyd*(a1 - d);
      As1*a1*fyd - As1*d*fyd - a1*b*d*eta*fcd*xiu;
       As1*a1*fyd - As1*d*fyd - (b*d^2*eta*fcd*xiu)/4 -
(a1*b*d*eta*fcd*xiu)/4;
```

```
       As1*a1*fyd - As1*d*fyd + (((a1*b*eta*fcd)/2 +
(b*d*eta*fcd)/2)*(a1 - d)*(8*As1*fyd - a1*b*eta*fcd +
b*d*eta*fcd))/(4*(4*As1*fyd - a1*b*eta*fcd + b*d*eta*fcd))
- (a1^2*b*eta*fcd)/8 - (b*d^2*eta*fcd)/8 -
(a1*b*d*eta*fcd)/4;
      -(d*(2*As1*fyd + b*d*eta*fcd))/2 -
(a1*b*d*eta*fcd)/2;
      As2*fyd*(a2 - d);
      As2*a2*fyd - As2*d*fyd - a2*b*d*eta*fcd*xiu;
       As2*a2*fyd - As2*d*fyd - (b*d^2*eta*fcd*xiu)/4 -
(a2*b*d*eta*fcd*xiu)/4;
       As2*a2*fyd - As2*d*fyd + (((a2*b*eta*fcd)/2 +
(b*d*eta*fcd)/2)*(a1 - d)*(8*As2*fyd - a2*b*eta*fcd +
b*d*eta*fcd))/(4*(4*As2*fyd - a2*b*eta*fcd + b*d*eta*fcd))
- (a2^2*b*eta*fcd)/8 - (b*d^2*eta*fcd)/8 -
(a2*b*d*eta*fcd)/4;
      -(d*(2*As2*fyd + b*d*eta*fcd))/2 -
(a2*b*d*eta*fcd)/2];



 BV=zeros(160,1); %visos konstrukcijos takumo salygu lais-
vuju nariu vektorius, vienai F virsunei (takumo sal. sk. *
pjuviu sk. === e.g. 16*10=160)

 BV(1:10,1)=Bv(Ass(1,1),Ass(1,2),b(1),d(1));
 BV(11:20,1)=Bv(Ass(1,1),Ass(1,2),b(1),d(1));

 BV(21:30,1)=Bv(Ass(2,1),Ass(2,2),b(2),d(2));
 BV(31:40,1)=Bv(Ass(2,1),Ass(2,2),b(2),d(2));

 BV(41:50,1)=Bv(Ass(3,1),Ass(3,2),b(3),d(3));
 BV(51:60,1)=Bv(Ass(3,1),Ass(3,2),b(3),d(3));
 BV(61:70,1)=Bv(Ass(3,1),Ass(3,2),b(3),d(3));
 BV(71:80,1)=Bv(Ass(3,1),Ass(3,2),b(3),d(3));

 BV(81:90,1)=Bv(Ass(2,1),Ass(2,2),b(2),d(2));
 BV(91:100,1)=Bv(Ass(2,1),Ass(2,2),b(2),d(2));

 BV(101:110,1)=Bv(Ass(1,1),Ass(1,2),b(1),d(1));
 BV(111:120,1)=Bv(Ass(1,1),Ass(1,2),b(1),d(1));

 BV(121:130,1)=Bv(Ass(4,1),Ass(4,2),b(4),d(4));
 BV(131:140,1)=Bv(Ass(4,1),Ass(4,2),b(4),d(4));
 BV(141:150,1)=Bv(Ass(4,1),Ass(4,2),b(4),d(4));
```

```
 BV(151:160,1)=Bv(Ass(4,1),Ass(4,2),b(4),d(4));

Gama=[ones(20,1) zeros(20,3); zeros(20,1) ones(20,1)
zeros(20,2); zeros(40,2) ones(40,1) zeros(40,1);
zeros(20,1) ones(20,1) zeros(20,2); ones(20,1) zeros(20,3);
zeros(40,3) ones(40,1)];
% Bv=

%Influence matrixes of residual values
Hsubr=inv(A*inv(D)*A')*A*inv(D); %influence matrix of re-
sidual displacements
Gsubr=inv(D)*A'*Hsubr-inv(D); %influence matrix of residual
internal forces

H=Hsubr*Fi';
G=Gsubr*Fi';

%Linear constrains inequalities
% Anq=zeros(1124, 564);
Anq=zeros(1284, 644); %(640*2+4,644)

%Yielding condtions goes to Anq
Anq(1:640,1:640)=[Fi*G Fi*G Fi*G Fi*G; Fi*G Fi*G Fi*G Fi*G;
Fi*G Fi*G Fi*G Fi*G; Fi*G Fi*G Fi*G Fi*G];

Anq(1:640, 641:644)= - [Gama; Gama; Gama; Gama];

%Sign limitation of platic multipliers
Anq(641:1280,1:640)= - eye(640);

%Limitation of limit bending moments
Anq(1281:1284,641:644)=- eye(4);

%Free units of the linear constrains inequalities
bnq=zeros(1284,1);
% bnq(1:560,1)= - [Fi*Se1+BV; Fi*Se2+BV; Fi*Se3+BV;
Fi*Se4+BV];
bnq(1:640,1)=  [-Fi*Se1-BV; -Fi*Se2-BV; -Fi*Se3-BV; -
Fi*Se4-BV];
bnq(1281:1284,1)= - [M01lb; M02lb; M03lb; M04lb];

%Supporting matrixes for linear constrains
Z=[Fi*G Fi*G Fi*G Fi*G; Fi*G Fi*G Fi*G Fi*G; Fi*G Fi*G Fi*G
Fi*G; Fi*G Fi*G Fi*G Fi*G];
```

```
W=[Fi*Se1+BV; Fi*Se2+BV; Fi*Se3+BV; Fi*Se4+BV];

%Solving nonlinear optimization problem
lb=[]; ub=[]; Aeq=[]; beq=[];
[xx,fval,exitflag]=fmincon(@my-
fun1,x0,Anq,bnq,Aeq,beq,lb,ub,@mycon1);

%Data preparation for the new iteration
x0=xx;
M01x=xx(641); M02x=xx(642); M03x=xx(643); M04x=xx(644);




% Residual internal forces and residual displacements
lambda=repmat(eye(160),1,4)*xx(1:640);
Sr=G*lambda;
ur=H*lambda;

 See1=Se1+Sr;
 See2=Se2+Sr;
 See3=Se3+Sr;
 See4=Se4+Sr;



% %M rinkiniai pagal ribine iraza
SM01F1=[See1(1) See1(2) See1(16) See1(17)];
SM01F2=[See2(1) See2(2) See2(16) See2(17)];
SM01F3=[See3(1) See3(2) See3(16) See3(17)];
SM01F4=[See4(1) See4(2) See4(16) See4(17)];

SM01=[SM01F1 SM01F2 SM01F3 SM01F4];
maxSM01=max(abs(SM01));
M01=maxSM01;

SM02F1=[See1(4) See1(5) See1(13) See1(14)];
SM02F2=[See2(4) See2(5) See2(13) See2(14)];
SM02F3=[See3(4) See3(5) See3(13) See3(14)];
SM02F4=[See4(4) See4(5) See4(13) See4(14)];

SM02=[SM02F1 SM02F2 SM02F3 SM02F4];
maxSM02=max(abs(SM02));
M02=maxSM02;
```

```
SM03F1=[See1(7) See1(8) See1(10) See1(11)];
SM03F2=[See2(7) See2(8) See2(10) See2(11)];
SM03F3=[See3(7) See3(8) See3(10) See3(11)];
SM03F4=[See4(7) See4(8) See4(10) See4(11)];

SM03=[SM03F1 SM03F2 SM03F3 SM03F4];
maxSM03=max(abs(SM03));
M03=maxSM03;

SM04F1=[See1(19) See1(20) See1(22) See1(23)];
SM04F2=[See2(19) See2(20) See2(22) See2(23)];
SM04F3=[See3(19) See3(20) See3(22) See3(23)];
SM04F4=[See4(19) See4(20) See4(22) See4(23)];

SM04=[SM04F1 SM04F2 SM04F3 SM04F4];
maxSM04=max(abs(SM04));
M04=maxSM04;



% N rinkiniai pagal ribine iraza
SN01F1=[See1(3) See1(18)];
SN01F2=[See2(3) See2(18)];
SN01F3=[See3(3) See3(18)];
SN01F4=[See4(3) See4(18)];

SN01=[SN01F1 SN01F2 SN01F3 SN01F4];
maxSN01=max(abs(SN01));
N01=maxSN01;

SN02F1=[See1(6) See1(15)];
SN02F2=[See2(6) See2(15)];
SN02F3=[See3(6) See3(15)];
SN02F4=[See4(6) See4(15)];

SN02=[SN02F1 SN02F2 SN02F3 SN02F4];
maxSN02=max(abs(SN02));
N02=maxSN02;

SN03F1=[See1(9) See1(12)];
SN03F2=[See2(9) See2(12)];
SN03F3=[See3(9) See3(12)];
SN03F4=[See4(9) See4(12)];
```

```
SN03=[SN03F1 SN03F2 SN03F3 SN03F4];
maxSN03=max(abs(SN03));
N03=maxSN03;

SN04F1=[See1(21) See1(24)];
SN04F2=[See2(21) See2(24)];
SN04F3=[See3(21) See3(24)];
SN04F4=[See4(21) See4(24)];

SN04=[SN04F1 SN04F2 SN04F3 SN04F4];
maxSN04=max(abs(SN04));
N04=maxSN04;

%Result of each iteration
results(iter,:)=[iter fval M01 M02 M03 M04 exitflag];


%% Recalculating points of interaction curves, after each
iteration

for i=1;
k1=eta*fcd*b(i)*d(i);
k2=k1*d(i);
beta1=a1/d(i); %%%%
beta2=a2/d(i);
aa1=fyd*Ass(i,1)/k1; %%%% 2 lape As2
aa2=fyd*Ass(i,2)/k1;
end;

% Main points of the interaction curves
n1=-2*aa1;
n2=2*beta1;
n3=xiu;
n4=1+2*aa1;
n5=1+2*aa1+beta1;
n6=0.5*(1+beta1);
n7=-2*aa2;
n8=2*beta2;
n9=xiu;
n10=1+2*aa2;
n11=1+2*aa2+beta2;
n12=0.5*(1+beta2);
```

```
% n1=k1*[n1 n2 n6 n3 n4 n5 n7 n8 n12 n9 n10 n11]';
n1=k1*[n1 n2 n3 n6 n4 n5 n7 n8 n9 n12 n10 n11]';

m1=0;
m2=(aa1+beta1)*(1-beta1);
m3=-0.5*(xiu^2)+0.5*xiu*(1+beta1)+aa1*(1-beta1);
m4=0.5*beta1;
m5=0;
m6=aa1*(1-beta1)+0.125*((1+beta1)^2);
m7=0;
m8=(aa2+beta2)*(1-beta2);
m9=-0.5*(xiu^2)+0.5*xiu*(1+beta2)+aa2*(1-beta2);
m10=0.5*beta2;
m11=0;
m12=aa2*(1-beta2)+0.125*((1+beta2)^2);
%
% m1=k2*[m1 m2 m6 m3 m4 m5 -m7 -m8 -m12 -m9 -m10 -m11]';

m1=k2*[m1 m2 m3 m6 m4 m5 -m7 -m8 -m9 -m12 -m10 -m11]';

hold all
subplot(2,2,1)
plot(n1, m1)
title('Subplot 1: M01')

%# vertical line
hx = graph2d.constantline(0, 'LineStyle',':', 'Color',[.7
.7 .7]);
changedependvar(hx,'x');
%# horizontal line
hy = graph2d.constantline(0, 'Color',[.7 .7 .7]);
changedependvar(hy,'y');

for i=2;
k1=eta*fcd*b(i)*d(i);
k2=k1*d(i);
beta1=a1/d(i); %%%% gali buti a1/k1
beta2=a2/d(i);
aa1=fyd*Ass(i,1)/k1; %%%% 2 lape As2
aa2=fyd*Ass(i,2)/k1;
end;

% Main points of the interaction curves
n1=-2*aa1;
n2=2*beta1;
```

```
n3=xiu;
n4=1+2*aa1;
n5=1+2*aa1+beta1;
n6=0.5*(1+beta1);
n7=-2*aa2;
n8=2*beta2;
n9=xiu;
n10=1+2*aa2;
n11=1+2*aa2+beta2;
n12=0.5*(1+beta2);

n2=k1*[n1 n2 n3 n6 n4 n5 n7 n8 n9 n12 n10 n11]';

m1=0;
m2=(aa1+beta1)*(1-beta1);
m3=-0.5*(xiu^2)+0.5*xiu*(1+beta1)+aa1*(1-beta1);
m4=0.5*beta1;
m5=0;
m6=aa1*(1-beta1)+0.125*((1+beta1)^2);
m7=0;
m8=(aa2+beta2)*(1-beta2);
m9=-0.5*(xiu^2)+0.5*xiu*(1+beta2)+aa2*(1-beta2);
m10=0.5*beta2;
m11=0;
m12=aa2*(1-beta2)+0.125*((1+beta2)^2);

m2=k2*[m1 m2 m3 m6 m4 m5 -m7 -m8 -m9 -m12 -m10 -m11]';

hold all
subplot(2,2,2)
plot(n2, m2)
title('Subplot 2: M02')

%# vertical line
hx = graph2d.constantline(0, 'LineStyle',':', 'Color',[.7
.7 .7]);
changedependvar(hx,'x');
%# horizontal line
hy = graph2d.constantline(0, 'Color',[.7 .7 .7]);
changedependvar(hy,'y');

for i=3;
k1=eta*fcd*b(i)*d(i);
k2=k1*d(i);
beta1=a1/d(i); %%%% gali buti a1/k1
```

```
beta2=a2/d(i);
aa1=fyd*Ass(i,1)/k1; %%%% 2 lape As2
aa2=fyd*Ass(i,2)/k1;
end;

% Main points of the interaction curves
n1=-2*aa1;
n2=2*beta1;
n3=xiu;
n4=1+2*aa1;
n5=1+2*aa1+beta1;
n6=0.5*(1+beta1);
n7=-2*aa2;
n8=2*beta2;
n9=xiu;
n10=1+2*aa2;
n11=1+2*aa2+beta2;
n12=0.5*(1+beta2);

n3=k1*[n1 n2 n3 n6 n4 n5 n7 n8 n9 n12 n10 n11]';

m1=0;
m2=(aa1+beta1)*(1-beta1);
m3=-0.5*(xiu^2)+0.5*xiu*(1+beta1)+aa1*(1-beta1);
m4=0.5*beta1;
m5=0;
m6=aa1*(1-beta1)+0.125*((1+beta1)^2);
m7=0;
m8=(aa2+beta2)*(1-beta2);
m9=-0.5*(xiu^2)+0.5*xiu*(1+beta2)+aa2*(1-beta2);
m10=0.5*beta2;
m11=0;
m12=aa2*(1-beta2)+0.125*((1+beta2)^2);

m3=k2*[m1 m2 m3 m6 m4 m5 -m7 -m8 -m9 -m12 -m10 -m11]';

hold all
subplot(2,2,3)
plot(n3, m3)
title('Subplot 3: M03')

%# vertical line
hx = graph2d.constantline(0, 'LineStyle',':', 'Color',[.7
.7 .7]);
changedependvar(hx,'x');
```

```
%# horizontal line
hy = graph2d.constantline(0, 'Color',[.7 .7 .7]);
changedependvar(hy,'y');

for i=4;
k1=eta*fcd*b(i)*d(i);
k2=k1*d(i);
beta1=a1/d(i); %%%% gali buti a1/k1
beta2=a2/d(i);
aa1=fyd*Ass(i,1)/k1; %%%% 2 lape As2
aa2=fyd*Ass(i,2)/k1;
end;

% Main points of the interaction curves
n1=-2*aa1;
n2=2*beta1;
n3=xiu;
n4=1+2*aa1;
n5=1+2*aa1+beta1;
n6=0.5*(1+beta1);
n7=-2*aa2;
n8=2*beta2;
n9=xiu;
n10=1+2*aa2;
n11=1+2*aa2+beta2;
n12=0.5*(1+beta2);

n4=k1*[n1 n2 n3 n6 n4 n5 n7 n8 n9 n12 n10 n11]';

m1=0;
m2=(aa1+beta1)*(1-beta1);
m3=-0.5*(xiu^2)+0.5*xiu*(1+beta1)+aa1*(1-beta1);
m4=0.5*beta1;
m5=0;
m6=aa1*(1-beta1)+0.125*((1+beta1)^2);
m7=0;
m8=(aa2+beta2)*(1-beta2);
m9=-0.5*(xiu^2)+0.5*xiu*(1+beta2)+aa2*(1-beta2);
m10=0.5*beta2;
m11=0;
m12=aa2*(1-beta2)+0.125*((1+beta2)^2);

m4=k2*[m1 m2 m3 m6 m4 m5 -m7 -m8 -m9 -m12 -m10 -m11]';

hold all
```

```
subplot(2,2,4)
plot(n4, m4)
title('Subplot 4: M04')

%# vertical line
hx = graph2d.constantline(0, 'LineStyle',':', 'Color',[.7
.7 .7]);
changedependvar(hx,'x');
%# horizontal line
hy = graph2d.constantline(0, 'Color',[.7 .7 .7]);
changedependvar(hy,'y');

end;
    results

   %Ploting convergence graph
   figure
   plot(results(:,2))

%Plastic deformations
plast_def=Fi'*lambda;
beep
```

## D2. Funkcinio failo „myfun1.m” programinis kodas

```
function[F]=myfun1(xx);
global ilgg
F=ilgg*xx(641:644);
```

## D3. Apribojimų failo „mycon1.m” programinis kodas

```
function [Cnq, Ceq] = mycon1(xx);
global ilgg Fi Gama G Z W;
Cnq=[];
Ceq=xx(1:640)'*([Gama;Gama;Gama;Gama]*xx(641:644)-
Z*xx(1:640)-W);
```