Priedai

B priedas. Rėmo optimizavimo programinis kodas

3.5 poskyrio, pirmojo skaitinio eksperimento programinis kodas parašytas MATLAB programavimo terpėje

REMO RIBINIU IRAZ OPT PASISKIRSTYMAS

```
%{
       min L'S0

    kai:
nt l      {lambda_cr_}.*(sigmay[G1]{A}-sigmay[G1](chi){A}) = 0
<<<NAUJA
nt l      {lambda}'([Fi][G]{lambda}-sigmay[Gama](chi){A}+[Fi]{Ne}) = 0
t nl      [Fi][G]{lambda}-sigmay[Gama](chi){A}<=-[Fi]{Ne1}
t nl      -lambda <= 0
t nl      -So <= -S0_min
t nl      [H]{lambda} <= u_sup - ue_max
t nl      -[H]{lambda} <= - u_inf + ue_min
```

```
        cia:    lambda = [lambda_max; lambda_cr]
 nt l- netiesinie lygybes; t nl - tiesines nelygybes
============================================================
%}
```

Pradiniai duomenys apie medziagas ir skerspjuvius

```
clc;
% clear;
clearvars -except REZULTATAI
close all;
syms clear;

fy = 235e3;     % kPa takumo itempis
E = 205e6;      % kPa Tamprumo modulis

% Skerspjuviu pradines charakteristikos
% I1 yra kolonos; I2 - sijos
d1 = 0.14; d2 = 0.14;   % Kvadratinio skerspj. krastine [m]
t1 = 0.01; t2 = 0.01;   % Sienutes storiai [m]

% Plastinis skerspjuvio atsparumo momentas Mpl (supaprastintas mathcad)
Wpl = @(b,t) (3/2)*b^2*t + ((-44*b+5*pi*b)/8)*t^2 + ((142-33*pi)/24)*t^3;
% Triju skirtingu skerspjuviu plastiniai atparumo momentai:
Wpl1 = Wpl(d1,t1); Wpl2 = Wpl(d2,t2);
% plastiniai lenkimo momentai
Mpl1 = Wpl1*fy; Mpl2 = Wpl2*fy;
Mpl = [Mpl1; Mpl2; Mpl2; Mpl1];% Plastiniai momentai atitinkamiem strypam

% Inercijos momentu bendra formule. Cia supaprastinta su mathcadu.
% Tikroji yra mathcad faile...
I = @(b,t) ((79*b*t^3)/12) + ((2*b^3*t)/3) - ((83*t^4)/12) - ...
    ((13*b^2*t^2)/4) + ((133*pi*t^4)/64) - ((11*pi*b*t^3)/8) + ...
    ((5*pi*b^2*t^2)/16);
% Inercijos momentai skirtingiems stypams:
I1 = I(d1,t1); I2 = I(d2,t2);
I_skerspjuviu = [I1 I2 I2 I1]';

% Skerpsjuviu plotai. Bendra formule:
As = @(b,t) 2*b*t + 2*(b-2*t)*t - (((1.5*t)^2)*(4-pi)) + t^2*(4-pi);
% Plotai visiem strypam:
```

```
A1 = As(d1,t1); A2 = As(d2,t2);
A_skerspjuviu = [A1 A2 A2 A1]';
```

Geometrijos ivedimas (keiciamas blokas)

```
L = 2.25; % m  puse remo tarpatramio
H = 3*L;  % m  Remo aukstis
elementu_mazgai = [ 1 2; 2 3; 3 4; 4 5]; % pirma mazesnis numeris!!!!
mazgu_koordinates = [0 H; 0 0; L 0; 2*L 0; 2*L H]; % Mazgu koordinates x y
metrais
                    % (paveiksle mazgai nurodyti staciakampiuose)

atramos = [1 2 3 13 14 15];   % suvarzyti poslinkiai is visu galimu

lankstai = [];

mastelis = 30;     % mastelis kiek kartu padidinsime poslinkius brezinyje
M_mastelis = 50;   % momentu diagramos mastelis tiek kNm atitiks 1 metra
elementu_skaicius = size(elementu_mazgai,1);

% Ilgiu vektorius optimizacijai
global IL; IL = [2*H; 2*L ];
```

Apkrovos (keiciamas blokas)

```
% Apkrovu kombinacijos:
F1 = [ 15 100 0 0 60 0 0 100 0 ]';
F2 = [-10 100 0 0 60 0 0 100 0 ]';
F3 = [ 15 100 0 0 0  0 0 100 0 ]';
F4 = [-10 100 0 0 0  0 0 100 0 ]';
```

Tamprus skaiciavimas ir braizymas

```
% Pusiausvyros salygu koeficientu matrica:
[A] = A_matrica(elementu_mazgai,mazgu_koordinates,atramos,lankstai);
% Pasiduodamumu matrica:
[D] = D_matrica(elementu_mazgai,mazgu_koordinates,...
                E,A_skerspjuviu,I_skerspjuviu);

% Tamprus skaiciavimas:
```

```
alfa = D\A'/(A/D*A');       % Irazu influentine matrica
beta = inv(A/D*A');         % Poslinkiu influentine matrica

global Se1 Se2 Se3 Se4
% Tampraus skaiciavimo irazos
Se1 = alfa*F1; %Se2 = alfa*F2; Se3 = alfa*F3; Se4 = alfa*F4;
 % Tampraus skaiciavimo poslinkiai
% ue1 = beta*F1; %ue2 = beta*F2; ue3 = beta*F3; ue4 = beta*F4;


% APRIBOJIMAI
S0_min = [0; 0];
u_sup1 = 0.1; % m Horizontalus remo virsaus i desine
u_sup5 = 0.05; % m sijos vidurio ilinkis zemyn

% Isrenku asines jegas
for ii = 1:elementu_skaicius
    k = ii*3;
    Ned(ii,:) = Se1(k); % Asiniu jegu vektorius:
end

% Konfiguracijos matrica:
Gama = [ 1 0; 0 1; 0 1; 1 0];

% ++++++++++++++++++% optimizacijos ciklo pradzia++++++++++++++
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

x0 = zeros(130,1); % 32*4 + 2
iter = 10;
for i=1:iter


I1 = I(d1,t1); I2 = I(d2,t2);   % Inercijos momentai
A1 = As(d1,t1); A2 = As(d2,t2); % Skerpsjuviu plotai
Npl1 = A1*fy; Npl2 = A2*fy;     % Plastines asines jegos
Npl = Gama*[Npl1; Npl2];        % Plastines asines atitinkamiem strypam

% koeficientai del STIPRUMO salygu (EC3 6.39 ir greta)
a1 = (A1-2*t1*d1)/A1;       a2 = (A2-2*t2*d2)/A2;

if a1 > 0.5
    a1 = 0.5;
end
```

```
if a2 > 0.5
    a2 = 0.5;
end

a_vek = Gama*[a1; a2];

n = abs(Ned)./Npl; % Koeficientu n vektorius
% galutiniu koef prie M0 vektorius (st koeficentuas - stiprumo)
k = (ones(elementu_skaicius,1)-n)./(ones(elementu_skaicius,1)-0.5*a_vek);
% SUKEICIAU VIETOM

for o = 1:elementu_skaicius;
    if k(o)>1;
        k(o)=1;
    elseif k(o)<0;
        k(o)=0;
    else k(o)=k(o);
    end
end

%  Nauja konfiguracijos matrica su vienu koef. STIPRUMUI
G1 = [ones(2,1) zeros(2,1); zeros(4,1) ones(4,1); ones(2,1) zeros(2,1)];

kk = zeros(elementu_skaicius*2,1);
kk(1:2,:)=k(1); kk(3:6,:)=k(2); kk(7:8,:)=k(1);
G2 = G1.*repmat(kk,1,2);
G = [G2;G2];

% Nauja takumo matrica STIPRUMUI su vienu koeficientu Fi = zeros(8,12);:
fik = [1 0 0; 0 1 0]; % momentas momentas, asine ismetam.
Fi = blkdiag(fik,fik,fik,fik);
FI = [Fi;-Fi];

% ==== cia to paprasto klupumo koef skaiciavimas (vektoriais)
alfakoef = 0.49; % 0.49 - saltai formuoti (atrodo buna dazniau)
VI = Gama*[I1; I2];  % visai konstrukcijai inerc. momentu vektorius
VA = Gama*[A1; A2]; % Elementu plotu vektorius
%     ======= skaiciuojamuju ilgius nustatymas START
%     % Virsutinis aukstas:
%     Iru = I4; Irl = I4; Ic = I3; lr = 2*L4; lc = L3;
%     n_virs = (Iru*lc)/(Ic*lr); p_virs = (Irl*lc)/(2*Ic*lr);
%     % Vidurinis aukstas
%     Iru = I4; Irl = I4; Ic = I2; lr = 2*L4; lc = L2;
```

```
%     n_vid = (Iru*lc)/(2*Ic*lr); p_vid = (Irl*lc)/(2*Ic*lr);
    % Apatinis aukstas
    Iru = I2; Ic = I1; lr = 2*L; lc = H;
    n_apat = (Iru*lc)/(Ic*lr);
    % Bendros formules  STR 7.16 lentele
%     miu_02 = @(p,n)
((p+0.68)*(n+0.22)^(1/2))/((0.68*p*(p+0.9)*(n+0.08)+0.1*n)^(1/2));
%     miu_03 = @(p,n)
((p+0.63)*(n+0.28)^(1/2))/((p*n*(p+0.9)+0.1*n)^(1/2));
    miu_ap = @(n) ((n+0.56)/(n+0.14))^(1/2); % vieno auksto remui su st.
itvirtinimu

    % Skaiciuojam koeficientus
%     if n_virs <= 0.2;
%         miu_virs = miu_02(p_virs,n_virs);
%     else miu_virs = miu_03(p_virs,n_virs);
%     end;
%     if n_vid <= 0.2;
%         miu_vid = miu_02(p_vid,n_vid);
%     else miu_vid = miu_03(p_vid,n_vid);
%     end;
    miu_apat = miu_ap(n_apat);
    Vmiu = [miu_apat 1] % visi koef vienam vektoriuje

    VLcr = Gama*(Vmiu'.*[H; L]); % L.*Vmiu;
%     ======== skaiciuojamuju ilgius nustatymas END

VN_cry = (pi^2*E*VI)./(VLcr.^2);  % Pagal Euleri!? cia jau vektoriais V
Vlambda = ((fy*VA)./VN_cry).^(1/2);
Vfi = 0.5*(1+alfakoef*(Vlambda-0.2)+(Vlambda.^2));
Vchi = 1./(Vfi+(((Vfi.^2)-(Vlambda.^2)).^(1/2)));

l = length(Vchi); % Ivykdom salyga, kad chi<=1:
 for j=1:l
    if Vchi(j,1)>1;
        Vchi(j,1)=1;
    else
        Vchi(j,1)=Vchi(j,1);
    end
 end
% ====================================================

% Konstrukcijos konfiguracijos matrica G KLUPUMUI:
```

```
G1_kl = [ones(4,1) zeros(4,1)]; % Kolona
G2_kl = [zeros(4,1) ones(4,1)]; % Sija
G_klup = [G1_kl; G2_kl; G2_kl; G1_kl];


I_skerspjuviu = Gama*[I1 I2]';
A_skerspjuviu = Gama*[A1 A2]';
% Konstrukcijos pasiduodamomu matrica:
[D] = D_matrica(elementu_mazgai,mazgu_koordinates,...
    E,A_skerspjuviu,I_skerspjuviu);

% Tamprus skaiciavimas:
alfa = D^-1*A'*(A/D*A')^-1;   % Irazu influentine matrica
beta = (A/D*A')^-1;           % Poslinkiu influentine matrica


% Tampraus skaiciavimo irazos
Se1 = alfa*F1; Se2 = alfa*F2; Se3 = alfa*F3; Se4 = alfa*F4;
 % Tampraus skaiciavimo poslinkiai charakteristine apkrova
ue1 = beta*(1/1.3)*F1; ue2 = beta*(1/1.3)*F2;
ue3 = beta*(1/1.3)*F3; ue4 = beta*(1/1.3)*F4;

c = Mpl./Npl; % Mpl/Npl  koef vektorius

VCmy = 0.4*ones(elementu_skaicius,1);
k_yy = VCmy.*(ones(elementu_skaicius,1)+(Vlambda-
0.2.*ones(elementu_skaicius,1)).*abs(Ned./(Npl.*Vchi)));
k_yy_lim = VCmy.*(ones(elementu_skaicius,1)+
0.8.*ones(elementu_skaicius,1).*abs(Ned./(Npl.*Vchi)));
kl = length(k_yy); % Ivykdom salyga, kad chi<=1:
 for j=1:kl
    if k_yy (j,1) > k_yy_lim(j,1);
        k_yy (j,1) = k_yy_lim(j,1);
    else
        k_yy (j,1)=k_yy (j,1);
    end
 end

Fi_kl = @(t) [ k_yy(t)  0   -c(t)/Vchi(t)
              -k_yy(t)  0   -c(t)/Vchi(t)
               0    k_yy(t) -c(t)/Vchi(t)
               0   -k_yy(t) -c(t)/Vchi(t)]; %  Betkuriam elementui t
% Visai konstrukcijai dydis:
```

```
Fi_klup = blkdiag(Fi_kl(1),Fi_kl(2),Fi_kl(3),Fi_kl(4));

Hsubr = (A/D*A')^-1*A/D; % liekamuju poslinkiu infl matrica
Gsubr = D\A'*Hsubr-inv(D); % liekamuju irazu infl matrica

global Fi_sum
Fi_sum = [FI; Fi_klup]; % abi ir stiprumo ir klupumo salygos
H_liek = Hsubr*Fi_sum'; G_liek = Gsubr*Fi_sum';

global G_sum
G_sum = [-G; -G_klup]; % abi ir stiprumo ir klupumo konfiguracija

global Z % pagalbine del apribojimu failo
Z = [Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek
     Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek
     Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek
     Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek];

% Aeq = []; beq = []; %  tiesiniai apribojimai lygybemis
% va cia ta nauja salyga!!!!!!
N0 = zeros(16); V1 = eye(16); M00 = zeros(16,2);
Aeq = [N0 V1 N0 N0 N0 N0 N0 N0 M00
       N0 N0 N0 V1 N0 N0 N0 N0 M00
       N0 N0 N0 N0 N0 V1 N0 N0 M00
       N0 N0 N0 N0 N0 N0 N0 V1 M00];
beq = zeros(64,1);
%

Anq = zeros(266,130); % TIESINIAI APRIBOJIMAI NELYGYBEMIS
Anq(1:128,1:128)=[Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek
Fi_sum*G_liek
                  Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek
Fi_sum*G_liek
                  Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek
Fi_sum*G_liek
                  Fi_sum*G_liek  Fi_sum*G_liek  Fi_sum*G_liek
Fi_sum*G_liek];

Anq(1:128,129:130) = [G_sum; G_sum; G_sum; G_sum];

Anq(129:256,1:128) = -eye(128); % -lambda <= 0
Anq(257:258,129:130) = -eye(2); % -S0 <= -S0_min
Anq(259:266,1:128) = [ H_liek(1,:) H_liek(1,:) H_liek(1,:) H_liek(1,:)
```

```
                        H_liek(1,:) H_liek(1,:) H_liek(1,:) H_liek(1,:)
                        H_liek(1,:) H_liek(1,:) H_liek(1,:) H_liek(1,:)
                        H_liek(1,:) H_liek(1,:) H_liek(1,:) H_liek(1,:)
                        H_liek(5,:) H_liek(5,:) H_liek(5,:) H_liek(5,:)
                        H_liek(5,:) H_liek(5,:) H_liek(5,:) H_liek(5,:)
                        H_liek(5,:) H_liek(5,:) H_liek(5,:) H_liek(5,:)
                        H_liek(5,:) H_liek(5,:) H_liek(5,:) H_liek(5,:)];
                      % poslinkiu ribojimas i viena puse

bnq = zeros(266,1); % TIESINIAI APRIBOJIMAI NELYGYBEMIS (LAISVIEJI NARIAI)
bnq(1:128) = [ -Fi_sum*Se1
               -Fi_sum*Se2
               -Fi_sum*Se3
               -Fi_sum*Se4] ;% Stiprumo salygos

bnq(257:258) = -S0_min; % -S0 <= -S0_min
bnq(259:266) = [u_sup1 - ue1(1)
                u_sup1 - ue2(1)
                u_sup1 - ue3(1)
                u_sup1 - ue4(1)
                u_sup5 - ue1(5)
                u_sup5 - ue2(5)
                u_sup5 - ue3(5)
                u_sup5 - ue4(5)]; % poslinkiu ribojimas i viena puse

lb = []; ub = []; % Apatines ir virs kintamuju reiksmes


%==============OPTIMIZACIJOS PAPROGRAME====START==================
options = optimset('TolCon',1e-8,'TolFun',1e-8, 'TolX',1e-8, ...
    'Algorithm','sqp') %,'MaxIter',20000,'MaxFunEvals',20000 );

% ,'MaxIter',400 'trust-region-reflective''active-set' 'interior-point'
[x,fval,exitquad,output,lambda] = fmincon(@funkcija,x0,Anq,bnq,Aeq,beq,...
    lb,ub,@apribojimas,options);
%===============OPTIMIZACIJOS PAPROGRAME====END==================

% liekamosios irazos:
Sr = [G_liek G_liek G_liek G_liek]*x(1:128);
for ii = 1:elementu_skaicius
    ko = ii*3;
    Ne_tamprus(ii,:) = Se1(ko);     % tampriu asiniu jegu vektorius
    Ne_liekamasis(ii,:) = Sr(ko);  % liekamuju asiniu jegu vektorius
```

```
end

Ned = Ne_tamprus + Ne_liekamasis; % bendros asines jegos

Mpl1 = x(129);
Mpl2 = x(130); % Plastiniai momentai

Wpl1=Mpl1/fy; Wpl2=Mpl2/fy; % Plasti atsparumo momentai
Mpl = Gama*[Mpl1; Mpl2];

% ============ Pagal kubines priklausomybes randame naujus vamzdziu
storius:
 syms x1 x2 positive

a = solve(((142-33*pi)/24)*x1^3+((-44*d1+5*pi*d1)/8)*(x1^2)+(3/2)*d1^2*x1-
Wpl1);
h1 = double(a); % pakeiciam is simbolinio i skaitini dydi
int1 = size(h1,1); % pagalbinis kintamasis kiek teigiamu realiu saknu yra
for y=1:int1
    delta1(y)=h1(y)-t1;  % surenkam skirtumus tarp gautos saknies ir t1
end
[C,m] = min(delta1); % randam maziausia skirtuma ir jo indeksa
t1 = h1(m); % to indekso sakni duodam kaip sprendini - nauja stori t1
t1 = real(t1)
% ============
aa = solve(((142-33*pi)/24)*x2^3+((-
44*d2+5*pi*d2)/8)*(x2^2)+(3/2)*d2^2*x2-Wpl2);
h2 = double(aa); % pakeiciam is simbolinio i skaitini dydi
int2 = size(h2,1); % pagalbinis kintamasis kiek teigiamu realiu saknu yra
for y=1:int2
    delta2(y)=h2(y)-t2;  % surenkam skirtumus tarp gautos saknies ir t1
end
[CC,mm] = min(delta2); % randam maziausia skirtuma ir jo indeksa
t2 = h2(mm);% to indekso sakni duodam kaip sprendini - nauja stori t2
t2 = real(t2)
clear x1 x2 a aa delta1 delta2 h1 h2

x0 = x;
fval
exitquad
%Spausdiname iteracijos rezultatus lenteleje
for j=1:2
    rez(i,j)= x(128+j);
```

```
    rez(i,3)= fval;
    rez(i,4)= exitquad;
%     rez(i,6:8) = [t1 t2 t3];
end;

plot(rez(:,3));
pause(.0001);
end

rez
% plot(rez(:,5));
disp('sieneliu storiai');
T = [ t1 t2 ];
disp(T);

pavv = {'M1','M2','N12','M3','M4','N34','M5','M6','N56','M7','M8',...
    'N78'};

% Sr = G_liek*x(1:312);
S = Se1 + Sr;
S_v = [Se1 Sr S];
Irazos = dataset({S_v 'Se','Sr','S'}, ...
                'obsnames',  cellstr(pavv));
disp(Irazos)

ur = [H_liek H_liek H_liek H_liek]*x(1:128);

% Atskiriam takumo salygas i klupumo ir stiprumo, kad matytusi is kur
% lambdos
% Pagalbine nuline matrica
Nul = zeros(16,12);
M_stipr = [FI; Nul; FI; Nul; FI; Nul; FI; Nul;]; % Stiprumas fi matrica
M_klup = [ Nul; Fi_klup; Nul; Fi_klup; Nul; Fi_klup; Nul; Fi_klup;];%
Stiprumas fi
lambda_st = M_stipr'*x(1:128); % Stiprumas daugikliai lambda
lambda_kl = M_klup'*x(1:128); % Klupumas daugikliai lambda
lambda_bendr = [Fi_sum; Fi_sum; Fi_sum; Fi_sum]'*x(1:128);

format bank
daugikliai = 1e6*[lambda_bendr lambda_st lambda_kl];
% display('Plastiniai daugikliai');
isvados =dataset({daugikliai 'Visi','Stiprumo','Klupumo'}, ...
                'obsnames',  pavv');
```

```
disp(isvados)

format short
U1= [ue1(1) ue2(1) ue3(1) ue4(1)];
U1_sum = U1 + ur(1)

U5 = [ue1(5) ue2(5) ue3(5) ue4(5)];
U5_sum = U5 + ur(5)

fval

format bank
REZULTATAI(:,4) = [fval; Mpl1; Mpl2; t1*1000; t2*1000];
Rez_pav = {'Fval','Mpl_1','Mpl_2','t_1, mm','t_2, mm'};
REZULT = dataset({REZULTATAI
'A','B','C','D'},'obsnames',cellstr(Rez_pav));
disp(REZULT)
```

```
function [A] = A_matrica(elementu_mazgai,mazgu_koordinates,...
    atramos,lankstai)

%A_MATRICA failas sudarantis pusiausvyros lygciu koeficientu matrica
%   bet kokiai lenkiamai, gniuzdomai strypinei konstrukcijai plokstumoje
%   (remui)

elementu_skaicius = size(elementu_mazgai,1);  % Elementu skaicius
mazgu_skaicius = size(mazgu_koordinates,1);   % Mazgu skaicius

xx = mazgu_koordinates(:,1);      % mazgu koordinates x
yy = mazgu_koordinates(:,2);      % mazgu koordinates y

LL = 3*mazgu_skaicius;            % Laisvumo laipsnis (bendras su atramom)

for k = 1:elementu_skaicius;
    indice = elementu_mazgai(k,:);
    elemento_LL = [ indice(1)*3-2, indice(1)*3-1, indice(1)*3,...
                    indice(2)*3-2, indice(2)*3-1, indice(2)*3];

    xa = xx(indice(2))-xx(indice(1));
```

```
    ya = yy(indice(2))-yy(indice(1));
    elemento_ilgis = sqrt(xa*xa+ya*ya);
    cosa = xa/elemento_ilgis;
    sina = ya/elemento_ilgis;

    % k-ojo elemento transformacijos matricos pamatrice Tk:
    Tk = [ [cosa sina; -sina cosa], zeros(2,4); zeros(1,2) 1 zeros(1,3);
          zeros(2,3), [cosa sina; -sina cosa], zeros(2,1);
          zeros(1,5) 1];
    % k-ojo elemento pusiausvyros lygciu lokali matrica Ak_strich
    Ak_strich = [ 0 0 -1; 1/elemento_ilgis -1/elemento_ilgis 0; 1 0 0;
                  0 0 1; -1/elemento_ilgis 1/elemento_ilgis 0; 0 -1 0];

    % Atskiro k-ojo elemento transformuota pusiausvyros pamatrice:
    Ak_be_c = Tk'*Ak_strich;

    % Visos konstrukcijos situ pagalbiniu Ak_be_c sujungtine matrica:
    A_subr(6*k-5:6*k,3*k-2:3*k) = Ak_be_c;


    [aktyvus_poslinkiai, ia] = setdiff(elemento_LL, atramos);
    % k-ojo elemento poslinkiu darnos matrica Ck
    Ck1 = zeros(6,LL);   % cia yra visi poslinkiai - kartu su atramom!!
        for j=1:size(ia);
        Ck1(ia(j),aktyvus_poslinkiai(j)) = 1;
        end

    % Visi tariami konstrukcijos poslinkiai (su atraminiais):
    Visi_LL = (1:LL);
    % Tik galimi poslinkiai (atmetame itvirtinimus):
    Galimi_LL = setdiff(Visi_LL, atramos);

        for u = 1:size(Galimi_LL,2); % atmetam nereikalingas Ck eilutes
            Ck(:,u) = Ck1(:,Galimi_LL(u));
        end

    % Visos konstrukcijos darnos matrica C:
    C_be_lankstu(6*k-5:6*k,:) = Ck;

end

% Pridemam lankstus jeigu tokiu yra
if size(lankstai,1) > 0
```

```
    C = [C_be_lankstu, zeros(size(C_be_lankstu,1),size(lankstai,1))];

    for b = 1:size(lankstai,1)
        el_numeris = lankstai(b,1);
        galas = lankstai(b,2);
        if galas == 1
            index = 3;
        elseif galas == 2
            index = 6;
        else disp('Neteisingai ivesti lankstai');
        end
        C(6*el_numeris-6+index,size(C_be_lankstu,2)+b) = 1;
    end

else
    C = C_be_lankstu;
end

% Konstrukcijos pusiausyros matrica:
    A = C'*A_subr;
end
```

```
% cia aprasomi netiesiniai apribojimai

function [Cnq,Ceq] = apribojimas(x)
global Z G_sum Fi_sum Se1 Se2 Se3 Se4
% kadangi netiesiniu apribojimu nelygybemis nera, tai :
Cnq = [];
Ceq = [(x(1:128,1))'*(Z*x(1:128,1)+...
    [G_sum; G_sum; G_sum;
G_sum]*x(129:130,1)+[Fi_sum*Se1;Fi_sum*Se2;Fi_sum*Se3;Fi_sum*Se4])];
```

```
function [D] = D_matrica(elementu_mazgai,mazgu_koordinates,E,As,I)

%D_MATRICA failas sudarantis pasiduodamumu kvazidiagonaline matrica
%   bet kokiai lenkiamai, gniuzdomai strypinei konstrukcijai plokstumoje
```

```
%   (remui)

elementu_skaicius = size(elementu_mazgai,1);  % Elementu skaicius

xx = mazgu_koordinates(:,1);      % mazgu koordinates x
yy = mazgu_koordinates(:,2);      % mazgu koordinates y

for k = 1:elementu_skaicius;
    indice = elementu_mazgai(k,:);
    xa = xx(indice(2))-xx(indice(1));
    ya = yy(indice(2))-yy(indice(1));
    elemento_ilgis = sqrt(xa*xa+ya*ya);

    % k-ojo elemento pasiduodamumu matrica:
    dk = elemento_ilgis*[1/(3*E*I(k)) 1/(6*E*I(k)) 0;
                         1/(6*E*I(k)) 1/(3*E*I(k)) 0;
                         0 0 1/(E*As(k))];

    % Konstrukcijos pasiduodamumu matrica:
    D(3*k-2:3*k,3*k-2:3*k) = dk;
end

end
```

```
% Tikslo funkcija

function [F] = funkcija(x)
% ilgiu vektorius:
global IL
F = IL'*x(129:130);
```