Seminární práce

Mezní analýza rámových konstrukcí

Autor:
Michal Šmejkal

Vedoucí práce:
Prof. Ing. Milan Jirásek, DrSc.

24.4.2017

ČVUT Fakulta stavební v Praze

Obsah

I. Úvod

V seminární práci se zabývám plastickou analýzou konstrukcí, především tedy mezní analýzou pomocí kinematické metody. Hlavní náplní práce je vytvoření programu v jazyce MATLAB, který ze zadaných hodnot týkajících se geometrie konstrukce a zatížení určí body na konstrukci, kde dojde k vytvoření plastického kloubu. Pro zjednodušení uvažujeme rámovou konstrukci zatíženou pouze bodovými silami a osamělými momenty.

Pod pojmem plastická analýza konstrukce si lze představit vyšetřování stavu, kdy nastane kolaps v důsledku vytvoření dostatečného počtu plastických kloubů. Obecně je potřeba *n+1* plastických kloubů, kde *n* je číslo značící, kolikrát je konstrukce staticky neurčitá. Nemusí tomu však vždy tak být, například u spojitého nosníku, který není namáhán normálovou silou, může stačit menší počet kloubů.

Plastickou analýzu lze rozdělit na několik základních podskupin. Přírůstková analýza vyšetřuje postupné tvoření plastických kloubů. Jednoduše řečeno, na staticky neurčité konstrukci se vyšetří průběh vnitřních sil, určí se nejnamáhanější průřez, ve kterém při následném zvyšování zatížení vznikne plastický kloub. V místě tohoto průřezu vytvoříme skutečný kloub, který bude přenášet plastický moment, který se už ale nemůže zvětšovat. Tímto způsobem se bude postupovat, dokud nevznikne dostatečný počet plastických kloubů.

V některých případech nás však nezajímá historie vývinu plastických kloubů, ale pouze mezní stav, kdy dojde ke kolapsu konstrukce. Tímto se zabývá mezní plastická analýza. Tu lze rozdělit na statickou metodu a kinematickou metodu, kterou se v této práci zabývám. Na rozdíl od přírůstkové analýzy, která je obecnou metodou, lze mezní analýzu využít pouze, pokud se jedná o ideálně pružnoplastický model.

II. Princip maxima plastické disipace

Pro zjednodušení vysvětlení se při teoretickém odvození omezím na případ jednoosé napjatosti. Ta nastává například u příhradových konstrukcí, které jsou namáhané pouze osovou silou. Nicméně také u ohýbaných nosníků či trámů, v případě zanedbání vlivu smykového napětí.

Teoretickým základem mezních metod je princip maxima plastické disipace. Nejprve na úrovni materiálového bodu zavedeme množinu všech plasticky přípustných stavů.
V případě ideálně pružnoplastického modelu platí, že napětí σ je přípustné, pokud jeho absolutní hodnota je menší nebo rovna mezi kluzu, tj. pokud platí

$$\sigma \in S_{ep} = [-\sigma_0, \sigma_0]$$

kde S_{ep} je množina všech plasticky přípustných stavů. Pro vysvětlení podstaty principu maxima plastické disipace nám pomůže zavedení napětí σ^*, které může být libovolné hodnoty, avšak plasticky přípustné. Dále budeme předpokládat, že známe skutečné napětí σ a skutečnou rychlost plastické deformace $\dot{\varepsilon_p}$. Podle znaménka $\dot{\varepsilon_p}$ mohou nastat tři případy.

Buďto je $\dot{\varepsilon_p} > 0$ a dochází k přetváření v tahu a napětí tudíž musí být na kladné mezi kluzu, značeno $\sigma = \sigma_0$. Jakékoli napětí σ^* je vždy menší nebo rovno napětí σ_0, tedy platí, že $\sigma^* \leq \sigma_0 = \sigma$ a tím pádem i

$$\sigma^* \dot{\varepsilon_p} \leq \sigma \dot{\varepsilon_p}$$

protože $\dot{\varepsilon_p}$ je kladné a směr nerovnosti se nezmění.

I pro další dva stavy, tedy i pro $\dot{\varepsilon_p} < 0$ a $\dot{\varepsilon_p} = 0$, se dá ukázat, že nerovnost bude splněna. Tato podmínka nám také říká, že ze všech plasticky přípustných napětí σ^* je součin $\sigma^* \dot{\varepsilon_p}$ největší v případě skutečného napětí. Zapsáno

$$\max_{\sigma^* \in S_{ep}} \sigma^* \dot{\varepsilon_p} = \sigma \dot{\varepsilon_p}$$

Je jednoduché ukázat, že součin napětí a deformace značí práci vztaženou na jednotku objemu. Necháme-li působit napětí σ na nekonečně malé těleso o objemu dV, jeho

deformace se zvětší o infinitezimální přírustek dε. Napětí potom vykoná práci σ dε dV. V případě, kdy je přírůstek deformace pružný, práce vykonaná napětím vede ke zvýšení potenciální energie. Pokud je však přírůstek deformace plastický, vykonaná práce je disipována v plastických přetvárných procesech a potenciální energie zůstává nezměněna. Disipační výkon vztažený na jednotku objemu nebo také hustotu plastické disipace potom vyjádříme jako $D = \sigma \frac{\mathrm{d}\varepsilon_p}{\mathrm{d}t} = \sigma\dot{\varepsilon_p}$.

Jak je výše zmíněno, nerovnost $\sigma^*\dot{\varepsilon_p} \leq \sigma\dot{\varepsilon_p}$ platí i v případě, kdy $\dot{\varepsilon_p} < 0$ nebo $\dot{\varepsilon_p} = 0$, a znaménkem plastické deformace je i jednoznačně určeno, zdali se jedná o σ_0 nebo $-\sigma_0$. Proto můžeme hustotu plastické disipace vyjádřit jako $D = \sigma\dot{\varepsilon_p} = \sigma_0 \left|\dot{\varepsilon_p}\right|$. Z toho také vyplývá, že hustota plastické disipace je funkcí $\dot{\varepsilon_p}$, jelikož σ_0 je materiálová konstanta. Vzorec tedy můžeme rozšířit na

$$\max_{\sigma^* \in S_{ep}} \sigma^*\dot{\varepsilon_p} = \sigma\dot{\varepsilon_p} \doteq \sigma_0 \left|\dot{\varepsilon_p}\right| = D(\dot{\varepsilon_p})$$

Z toho plyne následující věta: Plastická disipace, tedy výkon skutečného napětí na skutečné rychlosti plastické deformace, je největší ze všech myšlených výkonů, které by libovolné plasticky přípustné napětí podávalo na skutečné rychlosti plastické deformace.

III. Věty mezní plastické analýzy

Pro následující odvození se nám bude hodit zavedení tzv. součinitele zatížení μ. Vektor vnějšího zatížení zapíšeme potom jako násobek takzvaného referenčního zatížení,

$$\mathrm{f} = \mu\,\bar{\mathrm{f}}$$

Princip mezní analýzy spočívá v hledání mezní hodnoty μ_0, při které dojde ke kolapsu konstrukce. Na základě zmíněných metod jsme schopni hledat horní a dolní odhady tohoto součinitele. μ_s názýváme *staticky přípustným součinitelem zatížení*, v případě existuje-li takový stav vnitřních sil, který je v rovnováze s vnějšími silami $\mu_s\ \bar{\mathrm{f}}$ a splňuje podmínky plastické přípustnosti. Obdobně pro každý kinematicky přípustný proces, který je popsán rychlostmi styčníkových posunů (a pootoční) a rychlostmi plastické deformace, zavedeme *kinematicky přípustný součinitel zatížení* μ_k, který splňuje rovnici

$$\mu_k \overline{f}^T \dot{d}_k = D(\dot{\varepsilon_p})$$

Ta nám říká, že výkon vnějších sil na rychlostech styčníkových posunů (a pootočení) je roven plastické disipaci pro příslušné rychlosti plastických deformací.

Nyní už můžeme zmínit základní větu mezní plastické analýzy: **Pro libovolný staticky přípustný součinitel zatížení $\boldsymbol{\mu_s}$ a libovolný kinematicky přípustný součinitel zatížení $\boldsymbol{\mu_k}$ platí nerovnost $\boldsymbol{\mu_s \leq \mu_k}$.**

Skutečný součinitel zatížení μ_0 musí být zároveň staticky i kinematicky přípustný. Při jeho dosažení je splněna podmínka rovnováhy s vnějšími silami a právě akorát dochází ke kolapsu, kdy se konstrukce stává mechanismem. Z tohoto plynou dvě další věty mezní analýzy:

Statická věta mezní plastické analýzy: Součinitel zatížení v mezním plastickém stavu je největší ze všech staticky přípustných součinitelů. Libovolný staticky přípustný součinitel je tedy jeho dolním odhadem.

Kinematická věta mezní plastické analýzy: Součinitel zatížení v mezním plastickém stavu je nejmenší ze všech kinematicky přípustných součinitelů. Libovolný kinematicky přípustný součinitel je jeho horním odhadem.

A. Důkaz základní věty mezní plastické analýzy

V případě, že μ_s je staticky přípustný součinitel zatížení, existují vnitřní síly $\mathbf{s}_s$, které jsou plasticky přípustné a v rovnováze s vnějšími silami $\mu_s\,\bar{\mathrm{f}}$, zapsáno jako

$$s_s \in S_{ep}, \qquad \mathrm{B}^T \mathrm{s}_s = \mu_s \bar{\mathrm{f}}$$

Jestliže je μ_k kinematicky přípustný součinitel zatížení, pak pro něj platí rovnice

$$\mu_k \bar{\mathrm{f}}^T \dot{d}_k = D(\dot{e}_{pk})$$

ve které se rychlosti plastických protažení

$$\dot{e}_{pk} = B\dot{d}_k$$

vypočítají ze styčníkových posunů a pootočení $\dot{d}_k$, které splňují podmínku

$$\bar{f}^T \dot{d}_k > 0$$

Podle principu maxima plastické disipace platí

$$D(\dot{e}_{pk}) \geq {s_s}^T \dot{e}_{pk}$$

Nyní lze výše zmíněné vztahy substituovat a upravit

$$\mu_k \bar{f}^T \dot{d}_k = D\left(\dot{e}_{pk}\right) \geq {s_s}^T \dot{e}_{pk} = {s_s}^T B\dot{d}_k = (B^T s_s)^T \dot{d}_k = \mu_s \bar{f}^T \dot{d}_k$$

z čehož nám vyplývá, že

$$\mu_k \bar{f}^T \dot{d}_k \geq \mu_s \bar{f}^T \dot{d}_k$$

Tím jsme dokázali, že platí výše zmíněný vztah $\mu_s \leq \mu_k$.

IV. Lineární programování

Na základě výše odvozené teorie jsme schopni se zapojením intuice ručně vypočítat a určit kritické průřezy, ve kterých dojde ke vzniku plastických kloubů. Tento ruční výpočet probíhá následovně.

Podle stupně statické neurčitosti zjistíme, kolik musí vzniknout plastických kloubů. Následně sestavíme všechny možné kombinace, podle toho, v jakých průřezech klouby vzniknou. Poté na základě úvahy vyjádříme rychlosti plastické deformace v závislosti na rychlosti posunu a pootočení styčníků. Zapsáno jako

$$B\dot{d} = \dot{e_p}$$

kde $\dot{d}$ je vektor rychlostí styčníkových posunů a potočení a $\dot{e_p}$ je vektor rychlostí plastických deformací. Z této rovnice vyjádříme posuny a pootočení $\dot{d}$ v závislosti na $\dot{e_p}$. Následně můžeme už podle vztahu

$$\mu_k \bar{\mathrm{f}}^T \dot{d} = {s_0}^T \left|\dot{e_p}\right|$$

hledat součinitel zatížení μ_0, který bude ze všech součinitelů μ_k ten nejmenší.

Takto lze vypočítat jednoduché konstrukce, avšak u složitějších by tento intuitivní přístup pravděpodobně selhal. Naštěstí lze tento typ úlohy převést na problém lineárního programování, což je optimalizační metoda, kdy se hledá minimum lineární funkce na určité oblasti popsané lineárními rovnicemi a lineárními nerovnostmi. Základní forma pro řešení problému simplexovou metodou vypadá následovně:

$$\text{Minimize } c^T x$$

$$Ax = b$$

$$x \geq 0$$

Po zavedení jistých předpokladů se pokusím ukázat postup, pomocí kterého i náš problém převedeme do tzv. standartní formy. Z výše zmíněného vzorce můžeme vyjádřit součinitel μ_k jako

$$\mu_k = \frac{{s_0}^T \left|\dot{e_p}\right|}{\bar{\mathrm{f}}^T \dot{d}}$$

Když vynásobíme rychlosti posunů $\dot{d}$ a rychlosti deformací $\dot{e_p}$ stejným číslem, dostaneme další kinematicky přípustný stav, kterému však odpovídá stejný součinitel zatížení μ_k.

V seminární práci jsem pro výpočet využil vestavěnou funkci v programu MATLAB, která pracuje na principu simplexové metody. Na základě tohoto tvrzení můžeme zavést normalizační podmínku, kdy

$$\bar{\mathrm{f}}^{T}\dot{d} = 1.$$

Nyní hledáme μ_k podle vztahu $\mu_k = {s_0}^T\left|\dot{e_p}\right|$. V úloze lineárního programování musí být však všechny vztahy lineární, což absolutní hodnota nesplňuje. Můžeme však rychlost deformace vyjádřit jako

$$\dot{e} = \dot{e}^+ - \dot{e}^-$$

kde platí, že

$$\dot{e}^+ = (|\dot{e}| + \dot{e})/2 \geq 0$$

$$\dot{e}^- = (|\dot{e}| - \dot{e})/2 \geq 0$$

$$|\dot{e}| = \dot{e}^+ + \dot{e}^-$$

Teď už můžeme napsat úlohu ve tvaru lineárního programování jako

$$\text{Minimize } \mu_k(\dot{e}^+, \dot{e}^-, \dot{d}) = {s_0}^T\dot{e}^+ + {s_0}^T\dot{e}^-$$

$$\dot{e}^+ - \dot{e}^- - B\dot{d} = 0$$

$$\bar{\mathrm{f}}^{T}\dot{d} = 1$$

$$\dot{e}^+ \geq 0$$

$$\dot{e}^- \geq 0$$

Jak je vidět, funkce μ_k závisí na kladných rychlostech deformacích $\dot{e}^+$ a $\dot{e}^-$, ale také na rychlostech posunutí $\dot{d}$, které nemusí být nutně nezáporné. Jedna z možností, jak je možné tento problém vyšešit, je vyjádření rychlostí posunutí pomocí rychlostí deformací, zapsáno jako

$$\dot{d} = P^T\dot{e}$$

a

$$0 = S^T\dot{e}$$

Nyní je už problém převeden na sandartní formu lineárního programování. Výsledná verze vypadá tedy takto:

$$\text{Minimize } \mu_k(\dot{e}^+, \dot{e}^-) = {s_0}^T\dot{e}^+ + {s_0}^T\dot{e}^-$$

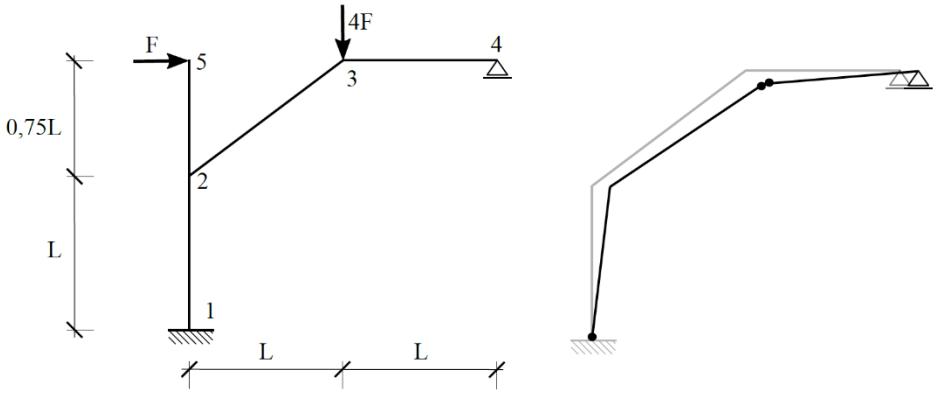
$$S^T\dot{e}^+ - S^T\dot{e}^+ = 0$$

$$\bar{\mathrm{f}}^{T}P^T\dot{e}^+ - \bar{\mathrm{f}}^{T}P^T\dot{e}^- = 1$$

$$\dot{e}^+ \geq 0$$

$$\dot{e}^- \geq 0$$

V. Příklad



Rychlosti posunů a pootočení	
$\dot{u}_2$	0.1739
$\dot{w}_2$	0.0000
$\dot{\varphi}_2$	-0.1739
$\dot{u}_3$	0.3043
$\dot{w}_3$	0.1739
$\dot{\varphi}_3$	0.0000
$\dot{u}_4$	0.3043

Rychlosti deformací	
$\dot{\theta}_{12}$	0.1739
$\dot{\theta}_{21}$	0.0000
$\dot{\theta}_{23}$	0.0000
$\dot{\theta}_{32}$	0.1739
$\dot{\theta}_{34}$	-0.1739

$\mu_0 = 0{,}1739$

VI. Závěr

Hlavní náplní seminární práce je vytvoření jednoduchého programu pro určení kritických průřezů, kde dojde ke vzniku plastických kloubů. Nejnáročnější část výpočtu je zpracování geometrie konstrukce a vytvoření matice $\boldsymbol{B}$, figurující ve vztahu

$$B\dot{d} = \dot{e_p}$$

Tato část je stále ještě nekompletní, a proto program zatím zvládne jen specifické případy podepření konstrukce. V budoucnu by bylo možné práci rozšířit, například pro výpočet spojitého zatížení, a pokusit se program zobecnit. Další možnost vylepšení by mohla proběhnout v grafickém zadávání dat, týkajících se geometrie a zatížení konstrukce.

Reference

[1] JIRÁSEK, Milan a Jan ZEMAN. *Přetváření a porušování materiálů: dotvarování, plasticita, lom a poškození*. 2. vyd. V Praze: České vysoké učení technické, 2012. ISBN 978-80-01-05064-4.

[2] JIRÁSEK, Milan a Zdeněk BAŽANT. *Inelastic Analysis of Structures*. 1. John Wiley, 2002. ISBN 9780471987161.

VII. Příloha - script

```
syms  F
% načtení dat
filename = 'data.txt';
data = dlmread(filename);
n_st = data(1,1);
sour_styc = data(2:n_st+1,1:2);
pos = data(2:n_st+1,3:5);
pos_tran = pos';
n_pr = data(2+n_st,1);
s_leve = data(n_st+3:n_st+2+n_pr,1);
s_prave = data(n_st+3:n_st+2+n_pr,2);
F_pom = data(n_st+3+n_pr:end,1);

[poc_nezn,~]=size(find(not(pos)));
[l,~]=find(pos==2);
l1 = length(l);

L_pom = sqrt((sour_styc(s_prave,2)-sour_styc(s_leve,2)).^2
+(sour_styc(s_prave,1)-sour_styc(s_leve,1)).^2);
L_pom1 = zeros(2*n_pr-l1,1);
for k = 1:n_pr
L_pom1(2*k,1) = L_pom(k,1);

L_pom1(2*k-1,1) = L_pom(k,1);
end

z=[];
z1 =[];
for h = 1:l1

Q = find(s_leve==l(h,1));
Q1= find(s_prave==l(h,1));

z=[z;Q];
z1=[z1;Q1];
end

if l1>0
  if 2*z-1 > 2*z1
    L_pom1(2*z-1)=[];
    L_pom1(2*z1)=[];
  else
    L_pom1(2*z1)=[];
    L_pom1(2*z-1)=[];
  end
end
sinus = (sour_styc(s_leve,2)-sour_styc(s_prave,2))./L_pom;
```

```
cosinus = (sour_styc(s_prave,1)-
sour_styc(s_leve,1))./L_pom;

sin_pom1 = zeros(2*n_pr-l1,1);
cos_pom1 = zeros(2*n_pr-l1,1);
for k = 1:n_pr
sin_pom1(2*k,1) = sinus(k,1);
sin_pom1(2*k-1,1) = sinus(k,1);

cos_pom1(2*k,1) = cosinus(k,1);
cos_pom1(2*k-1,1) = cosinus(k,1);
end

z=[];
z1 =[];
for h = 1:l1

Q = find(s_leve==l(h,1));
Q1= find(s_prave==l(h,1));

z=[z;Q];
z1=[z1;Q1];
end

if l1>0
  if 2*z-1 > 2*z1
    sin_pom1(2*z-1)=[];
    sin_pom1(2*z1)=[];

    cos_pom1(2*z-1)=[];
    cos_pom1(2*z1)=[];
  else
    sin_pom1(2*z1)=[];
    sin_pom1(2*z-1)=[];
    cos_pom1(2*z1)=[];
    cos_pom1(2*z-1)=[];
  end
end
n=n_pr;
L = 1;
% délky jednotlivých částí
L_m = L_pom1;
x = poc_nezn;
% siny a cosiny jednotlivých pruů
s = sin_pom1;
c = cos_pom1;
% vytvoření matice B
s1 = sinus;
c1 = cosinus;

u_prave = pos(s_prave,1);
```

```
u_leve = pos(s_leve,1);
s = find(u_prave==0);
u_prave(u_prave==1,1)=zeros(length(find(u_prave==1)),1);
u_prave(s,1)=ones(length(s),1);
s = find(u_leve==0);
u_leve(u_leve==1,1)=zeros(length(find(u_leve==1)),1);
u_leve(s,1)=ones(length(s),1);

w_prave = pos(s_prave,2);
w_leve = pos(s_leve,2);
s = find(w_prave==0);
w_prave(w_prave==1,1)=zeros(length(find(w_prave==1)),1);
w_prave(s,1)=ones(length(s),1);
s = find(w_leve==0);
w_leve(w_leve==1,1)=zeros(length(find(w_leve==1)),1);
w_leve(s,1)=ones(length(s),1);

 T=  [];
 S = [];

 for k = 1:n_pr
     kk = s_leve(k);
A(k,3*kk-2)=1;
A(k,3*kk-1)=1;
if u_leve(k)==0;
 T = [T,3*kk-2];
end
if w_leve(k)==0;
   S = [S,3*kk-1];
end
 end
 T2=  [];
 S2 = [];

 for k = 1:n_pr
     kk = s_prave(k);
A2(k,3*kk-2)=1;
A2(k,3*kk-1)=1;
if u_prave(k)==0;
 T2 = [T2,3*kk-2];
end
if w_prave(k)==0;
   S2 = [S2,3*kk-1];
end
 end

fi_leve = pos(s_leve,3);
   s = find(fi_leve==0);
fi_leve(fi_leve==1,1)=zeros(length(find(fi_leve==1)),1);
fi_leve(s,1)=ones(length(s),1);
```

```
fi_prave = pos(s_prave,3);
   s = find(fi_prave==0);
fi_prave(fi_prave==1,1)=zeros(length(find(fi_prave==1)),1);
fi_prave(s,1)=ones(length(s),1);

 T_fi=  [];
 S_fi = [];
 for k = 1:n_pr
     kk = s_leve(k);
if fi_leve(k)==0;
 T_fi = [T_fi,3*kk];
end
if fi_prave(k)==0;
    S_fi = [S_fi,3*kk-1];
end
 end
TS_fi =  [T_fi,S_fi];
TS2 =  [T2,S2];
TS =  [T,S];

A2(:,[TS2,TS_fi])=[];
A2(:,1:2)=[];% Pozor
ZZZ = A;
A(:,[TS,TS_fi])=[];
[c,d]=size(A);
A(:,d+1:poc_nezn)=zeros(c,poc_nezn-d);

A_leve = A;
A_prave = A2;
for k = 1:n_pr
A_pom1(2*k,:) = A_leve(k,:);
A_pom1(2*k-1,:) = A_leve(k,:);

A_pom2(2*k,:) = A_prave(k,:);
A_pom2(2*k-1,:) = A_prave(k,:);
end

 z=[];
z1 =[];
for h = 1:l1
Q = find(s_leve==l(h,1));
Q1= find(s_prave==l(h,1));

z=[z;Q];
z1=[z1;Q1];
end

if l1>0
  if 2*z-1 > 2*z1
     A_pom1(2*z-1,:)=[];
     A_pom1(2*z1,:)=[];
```

```
    A_pom2(2*z-1,:)=[];
    A_pom2(2*z1,:)=[];

  else
    A_pom1(2*z1,:)=[];
    A_pom1(2*z-1,:)=[];

    A_pom2(2*z1,:)=[];
    A_pom2(2*z-1,:)=[];

  end
end
 B_new = A_pom2-A_pom1;
   fi = pos(s_leve,3);
   s = find(fi==0);
fi(fi==1,1)=zeros(length(find(fi==1)),1);
fi(s,1)=ones(length(s),1);
   fi(fi==2)=[];

   for k = 1:n_pr
if s_prave(k,1)-1 < 1
    continue
end
 if  (s_prave(k,1)-1)*3 >poc_nezn
     break
 end
 if 2*k < length(sin_pom1)+1
      B_new(2*k,(s_prave(k,1)-1)*3)= fi(s_prave(k,1),1);
 end
end
   for k = 1:n_pr
if s_leve(k,1)-1 < 1
    continue
end
 if  (s_leve(k,1)-1)*3 >poc_nezn
     break
 end
 if 2*k-1 < length(sin_pom1)+1
      B_new(2*k-1,(s_leve(k,1)-1)*3)= fi(s_leve(k,1),1);
 end
   end
B = B_new;

for k = 1:3:poc_nezn
    B(:,k)=  -sin_pom1./L_m.*B(:,k);
    if k+1>poc_nezn
        break
    end
    B(:,k+1)=  cos_pom1./L_m.*B(:,k+1);
    if k+2>poc_nezn
```

```
        break
    end
end
% e - matice protazeni
e = zeros(n,x);

for k = 0:n_pr-1
    e(k+1,:)= (A_pom2(2*k+1,:) - A_pom1(2*k+1,:));
end


for k = 1:n
    if 3*k-2>poc_nezn
        break
    end
e(:,3*k-2)= c1.*e(:,3*k-2);

end
for k = 0:(n-2)
e(:,3*k+2)= s1.*e(:,3*k+2);
e(:,3*k+3)= zeros(n,1);
end
% spojeni matic dohromady, příprava pro gaussovu eliminaci
B = [e;B];
B_prava = [zeros(n,2*n-1);eye(2*n-1)];
B = [B,B_prava];
a1 = gau_elim(B,x);

[m,n] = size(a1);
P_t = a1(1:x,x+1:n);

% vnější síly
M0=1;
mi = F*L/M0;
F = solve(mi==1,F);
F_ex = F.*F_pom';

% normalizing condition
Norm_cond = F_ex*P_t;
[~,x1] = size(Norm_cond);
% compatibility condition
com_con = a1(x+1:m,x+1:n);

% simplex tabulka
M_0 = ones(1,length(cos_pom1));
    M_0 = [1,1,1,1,1];
%   M_0 = [1,1,1];%test
% M_0 = [1,1,1,1,1,1,1]; %test2
    SIM_TAB = zeros(1+m-x+1,2*x1+2);
    [q,r] = size(SIM_TAB);
    SIM_TAB(q,1)=1;
```

```
    %první řádky - compatibility condition
        SIM_TAB(1:m-x,2:1+x1)=com_con;
        SIM_TAB(1:m-x,2+x1:r-1)=-com_con;
    %předposlední řádek - normalizing condition
        SIM_TAB(q-1,2:1+x1)=Norm_cond;
        SIM_TAB(q-1,2+x1:r-1)=-Norm_cond;

    SIM_TAB(q-1,r)=1;
    SIM_TAB(q,2:r-1)= [M_0,M_0]; %objective function

theta = linprog(SIM_TAB(q,2:r-1),[],[],SIM_TAB(1:q-1,2:r-
1),SIM_TAB(1:q-1,r),zeros(r-2,1));
theta = theta(1:(r-2)/2,1)-theta((r-2)/2+1:end,1)
zplastizovane_prurezy = find(abs(theta)>1e-8)
posuny = P_t*theta
mi = M_0*theta
```