Vysoká škola báňská – Technická univerzita Ostrava
Fakulta stavební

Přednáška z předmětu: Speciální numerické metody

Téma č.4: Vlastní čísla a vlastní vektory

doc. Ing. Martin Krejsa, Ph.D.

Obsah

Obsah
3. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

Kapitola 1

Vlastní čísla a vlastní vektory

Cíle

Kapitola je zaměřena na:

- seznámení s pojmy vlastní čísla a vlastní vektory,
- teoretické pozadí některých způsobů jejich výpočtu,
- ukázku využití vlastních čísel a vektorů v inženýrské praxi.

1.1. Úvod

Vlastní vektor matice $[A]$ je takový nenulový vektor $\{u\}$, pro který platí:

$$[A] \cdot \{u\} = \lambda \cdot \{u\} \,, \tag{1.1}$$

Obsah
4. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

kde λ je vlastní (charakteristické) číslo matice $[A]$ a $\{u\}$ vlastní (charakteristický) vektor matice $[A]$ příslušný vlastnímu číslu λ.

Poznámka 1.1. Množina všech vlastních čísel matice $[A]$ se označuje jako spektrum matice $[A]$.

Příklad 1.2. Určete vlastní číslo λ matice $[A] = \begin{bmatrix} 4 & -2 \\ 1 & 1 \end{bmatrix}$ a vektoru $\{u\} = \begin{Bmatrix} 2 & 1 \end{Bmatrix}^T$.

Řešení. Řešení vychází z rovnice (1.1), kdy po dosazení lze získat:

$$[A] \cdot \{u\} = \begin{bmatrix} 4 & -2 \\ 1 & 1 \end{bmatrix} \cdot \begin{Bmatrix} 2 \\ 1 \end{Bmatrix} = \begin{Bmatrix} 6 \\ 3 \end{Bmatrix} = 3 \cdot \begin{Bmatrix} 2 \\ 1 \end{Bmatrix} = 3 \cdot \{u\} \,. \tag{1.2}$$

Vlastním číslem matice $[A]$ je tedy $\lambda = 3$. Vektor $\{u\} = \begin{Bmatrix} 2 & 1 \end{Bmatrix}^T$ je pak vlastní vektor matice $[A]$ příslušný k vlastnímu číslu $\lambda = 3$. ▲

Rovnici (1.1) lze zapsat rovněž ve tvaru:

$$([A] - \lambda \cdot [E]) \cdot \{u\} = 0 \,, \tag{1.3}$$

kde $[E]$ je jednotková matice velikosti n (čtvercová matice typu $n \times n$, která má na hlavní diagonále jedničky a na ostatních místech nuly).

Rovnice (1.3) představuje soustavu homogenních rovnic:

Obsah
5. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

$$\begin{array}{ccccccccc} (a_{1,1}-\lambda)\cdot x_1 & + & a_{1,2}\cdot x_2 & + & \cdots & + & a_{1,n}\cdot x_n & = & 0 \\ a_{2,1}\cdot x_1 & + & (a_{2,2}-\lambda)\cdot x_2 & + & \cdots & + & a_{2,n}\cdot x_n & = & 0 \\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n,1}\cdot x_1 & + & a_{n,2}\cdot x_2 & + & \cdots & + & (a_{n,n}-\lambda)\cdot x_n & = & 0 \end{array}, \tag{1.4}$$

která má netriviální řešení pro

$$\det\left([A]-\lambda\cdot[E]\right)=0\,, \tag{1.5}$$

což představuje tzv. *charakteristickou rovnici* matice $[A]$. Určením determinantu lze získat tzv. *charakteristický polynom* $p(\lambda)$ stupně n. Řešením rovnice $p(\lambda)=0$ je pak celkem n vlastních čísel matice $[A]$.

Poznámka 1.3. Čtvercová diagonální matice $\begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ \vdots & & \ddots & \\ 0 & 0 & 0 & \lambda_n \end{bmatrix}$ se označuje jako tzv. *spektrální matice* matice $[A]$.

Příklad 1.4. Určete pro matici matice $[A]=\begin{bmatrix} 3 & -1 \\ 2 & 0 \end{bmatrix}$ vlastní čísla $\lambda_{1,2}$ a vlastní vektory $\{u_{1,2}\}$.

Řešení. Řešení vychází z charakteristické rovnice (1.5), kdy po dosazení lze získat:

$$\det\left([A]-\lambda\cdot[E]\right)=\begin{bmatrix} 3-\lambda & -1 \\ 2 & -\lambda \end{bmatrix}=0\,. \tag{1.6}$$

Obsah
6. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

Výpočet determinantu této matice vede ke kvadratické rovnici:

$$(3-\lambda)\cdot(-\lambda)-(-1)\cdot 2=\lambda^2-3\cdot\lambda+2=0\,, \tag{1.7}$$

jejíž řešením jsou dva kořeny - vlastní čísla $\lambda_1=2$ a $\lambda_2=1$.

Vlastní vektor u_1 příslušný hodnotě vlastního čísla $\lambda_1=2$ lze určit řešením soustavy lineárních rovnic:

$$\begin{bmatrix} 3-\lambda_1 & -1 \\ 2 & -\lambda_1 \end{bmatrix}\cdot u_1=\begin{bmatrix} 1 & -1 \\ 2 & -2 \end{bmatrix}\cdot u_1=0\,. \tag{1.8}$$

Řešením soustavy lineárních rovnic (1.8) je např. vektor $u_1=\left\{\begin{array}{cc} 2 & 2 \end{array}\right\}^T$.

Vlastní vektor u_2 příslušný hodnotě vlastního čísla $\lambda_2=1$ pak lze určit obdobně řešením soustavy lineárních rovnic:

$$\begin{bmatrix} 3-\lambda_2 & -1 \\ 2 & -\lambda_2 \end{bmatrix}\cdot u_2=\begin{bmatrix} 2 & -1 \\ 2 & -1 \end{bmatrix}\cdot u_2=0\,. \tag{1.9}$$

Řešením soustavy lineárních rovnic (1.9) je např. vektor $u_2=\left\{\begin{array}{cc} 1 & 2 \end{array}\right\}^T$.

▲

Příklad 1.5. Určete vlastní čísla a vlastní vektory pro matici $[A]$ z příkladu 1.4 s využitím příkazů systému Matlab.

Řešení. V případě potřeby lze určit vlastní čísla a vlastní vektory matice pomocí funkce `eig`:

Obsah
7. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
[V,L]=eig(A);
diag(L)
V
```

kde L je spektrální matice, ve které jsou vlastní čísla na diagonále, příkaz `diag(L)` vypíše vektor s vlastními čísly a `V` vypíše matici, kde jsou jednotlivé vlastní vektory uloženy po sloupcích. Správnost výpočtu vlastních čísel a vlastních vektorů lze zkontrolovat na základě vztahu (1.3) např. příkazem:

```
A*V-V*L
```

V programovém systému Matlab lze také určit koeficienty charakteristického polynomu a následně i jeho řešení následující posloupností příkazů:

```
c=poly(A)
l=roots(c)
```

Řešení lze opět zkontrolovat např.:

```
trace(A)-sum(l)
det(A)-l(1)*l(2)
```

nebo na základě vztahu (1.3):

```
det(A-l(1)*eye(size(A)))
det(A-l(2)*eye(size(A)))
```

Obsah
8. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

Příklad 1.6. Určete vlastní čísla a vlastní vektory pro následující matice: $[A_1] = \begin{bmatrix} 3 & 2 \\ 4 & 1 \end{bmatrix}$, $[A_2] = \begin{bmatrix} 2 & 0 & 0 \\ 2 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$, $[A_3] = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$ a $[A_4] = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix}$.

1.2. Numerické metody pro řešení vlastních čísel

Matematické úlohy související s nalezením vlastních čísel lze rozdělit do dvou skupin:

- *částečný problém vlastních čísel*, kdy se v úloze hledají pouze některá vlastní čísla - obvykle s nejmenší nebo největší absolutní hodnotou,
- *úplný problém vlastních čísel*, kdy se v úloze určují všechna vlastní čísla,

1.2.1. Částečný problém vlastních čísel

V případě částečného problému vlastních čísel se určuje pouze některé vlastní číslo. Většinou je řešením tzv. dominantní vlastní číslo s největší hodnotou.

1.2.1.1. Mocninná metoda

Zde je uveden skript, který využívá Mocninnou metodu (originální označení je *Power Iteration*) pro určení dominantního vlastního čísla čtvercové matice:

Obsah
9. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
function [x,lam]=powerit(A,x,k)
% Vstup: matice A, počáteční (nenulový) vektor x, počet kroků cyklu k
% Výstup: dominantní vlastní číslo lam, příslušný vlastní vektor x
for j=1:k
   u=x/norm(x);              % normalizace vektoru
   x=A*u;                    % mocninný krok
   lam=u'*x;                 % Rayleighův aproximační koeficient
end
```

Příklad 1.7. Určete Mocninnou metodou hodnotu dominantního vlastního čísla matice $[A] = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 3 & 6 & 2 \end{bmatrix}$. Počáteční vektor x zvolte $\{x\} = \{1, 1, 1\}^T$ a $k = 10$. Výsledek zkontrolujte postupem z příkladu 1.5.

1.2.1.2. Inverzní mocninná metoda

Modifikovaná Mocninná metoda pro určení vlastního čísla, které je nejblíže zadané hodnotě s, se nazývá *Inverzní mocninná metoda* (originální označení je *Inverse Power Iteration*). Následuje skript pro řešení touto metodou:

```
function [x,lam]=invpowerit(A,x,s,k)
% Vstup: matice A, počáteční (nenulový) vektor x
% tzv. posun s, počet kroků cyklu k
% Výstup: vlastní číslo lam, dominantní vlastní vektor inv(A-sI)
```

Obsah
10. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
As=A-s*eye(size(A));
for j=1:k
  u=x/norm(x);                % normalizace
  x=As\u;                     % inverzní mocninný krok
  lam=u'*x;                   % Rayleighův aproximační koeficient
end
lam=1/lam+s;
```

Příklad 1.8. Určete Inverzní mocninnou metodou hodnotu vlastního čísla matice $[A]$ z příkladu 1.7, které leží nejblíže hodnotě $s = -2$. Výsledek zkontrolujte postupem z příkladu 1.5.

1.2.1.3. Metoda Rayleighova podílu

Modifikací mocninné metody je metoda Rayleighova podílu (Rayleigh Quotient Iteration), která je implementována v následujícím skriptu:

```
function [x,lam]=rqi(A,x,k)
% Vstup: matice A, počáteční (nenulový) vektor x, počet kroků cyklu k
% Výstup: dominantní vlastní číslo lam, příslušný vlastní vektor x
for j=1:k
  u=x/norm(x);                % normalizace
  lam=u'*A*u;                 % Rayleighův koeficient
  x=(A-lam*eye(size(A)))\u; % inverzní mocninná iterace
end
x=x/norm(x);
lam=x'*A*x;                   % Rayleighův koeficient
```

Obsah
11. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

Příklad 1.9. Určete metodou Rayleighova podílu hodnotu dominantního vlastního čísla matice $[A] = \begin{bmatrix} 4 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \end{bmatrix}$. Počáteční vektor x zvolte $\{x\} = \{1, 1, 1\}^T$ a $k = 10$. Výsledek zkontrolujte postupem z příkladu 1.5.

1.2.2. Úplný problém vlastních čísel

Metody na řešení úplného problému se dělí do tří základních skupin:

1. Metody založené na výpočtech vlastních čísel pomocí charakteristického polynomu. Jsou nevýhodné pro řešení matic vyššího řádu n, protože v takových případech je obtížné určit kořeny charakteristického polynomu pomocí determinantu.

2. Metody využívající podobnosti matic, které jsou založeny na principu stejných vlastních čísel pro podobné matice. Patří zde např. *metoda LU-rozkladu* nebo *metoda ortogonálních transformací*.

3. Smíšené metody založené na převodu obecné matice na matici třídiagonální (patří zde např. (např. metoda Givensova, Householderova nebo Lanczosova) a následný efektivní výpočet kořenů charakteristického polynomu této upravené matice.

1.2.2.1. Metoda simultánní iterace

Metoda simultánní iterace (v originále Normalized Simultaneous Iteration) je implementována v následujícím skriptu:

Obsah
12. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
function [Q,lam]=nsi(A,k)
% Vstup: matice A, počet iteračních kroků k
% Výstup: vlastní čísla ve vektoru lam a matice vlastních vektorů Q
[m,n]=size(A);
Q = eye(m,m);
for j = 1:k
    [Q,R] = qr(A*Q); % QR faktorizace
end
lam=diag(Q'*A*Q);    % Rayleighův koeficient
```

Příklad 1.10. Určete metodou simultánní iterace hodnoty všech vlastních čísel matice $[A] = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$ a příslušné vlastní vektory. Výsledek zkontrolujte postupem z příkladu 1.5.

Obsah
13. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

1.2.2.2. Neposuvný QR algoritmus

Neposuvný QR algoritmus (v originále Unshifted QR Algorithm) je implementován v následujícím skriptu:

```
function [Qbar,lam]=unshiftedqr(A,k)
% Vstup: matice A, počet iteračních kroků k
% Výstup: vlastní čísla ve vektoru lam a
% matice vlastních vektorů Qbar
[m,n]=size(A);
Q=eye(m,m);
Qbar=Q; R=A;
for j = 1:k
  [Q,R] = qr(R*Q); % QR factorizace
  Qbar=Qbar*Q;     % akumulace Q'
end
lam=diag(R*Q);     % diagonalní konvergence k vlastním číslům
```

Příklad 1.11. Určete s pomocí neposuvného QR algoritmu hodnoty všech vlastních čísel matice $[A]$ z příkladu 1.10 a příslušné vlastní vektory. Výsledek zkontrolujte postupem z příkladu 1.5.

Příklad 1.12. Všechny popisované algoritmy pro výpočet vlastních čísel a vlastních vektorů pracovaly s využitím cyklu typu `for`. Upravte tyto algoritmy tak, aby využívaly cyklus typu `while` s vhodnou zakončovací podmínkou se zadanou tolerancí nepřesnosti ε.

Obsah
14. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

1.2.2.3. Posuvný QR algoritmus

Posuvný QR algoritmus (v originále Shifted QR Algorithm) určuje reálná a komplexní vlastní čísla čtvercové matice a je implementován v následujícím skriptu s využitím cyklu typu `while`:

```
% Vstup: matice A
% Výstup: vlastní čísla ve vektoru lam
function lam=shiftedqr(a)
tol=1e-14;
kounttol=500;
m=size(a,1);
lam=zeros(m,1);
n=m;
while n>1
  kount=0;
  while max(abs(a(n,1:n-1)))>tol&kount<kounttol
    kount=kount+1;   % sleduje počet qr's
    mu=a(n,n);       % posun je mu
    [q,r]=qr(a-mu*eye(n));
    a=r*q+mu*eye(n);
  end
  if kount<kounttol  % byl izolován blok 1x1
    lam(n)=a(n,n);   % určení vlastního čísla
    n=n-1;
```

Obsah
15. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
    a=a(1:n,1:n);    % zmenšení řádu matice o 1
  else               % byl izolován blok 2x2
    disc=(a(n-1,n-1)-a(n,n))^2+4*a(n,n-1)*a(n-1,n);
    lam(n)=(a(n-1,n-1)+a(n,n)+sqrt(disc))/2;
    lam(n-1)=(a(n-1,n-1)+a(n,n)-sqrt(disc))/2;
    n=n-2;
    a=a(1:n,1:n);    % zmenšení řádu matice o 2
  end
end
if n>0;lam(1)=a(1,1);end % zbyl jen blok 1x1
```

Příklad 1.13. Určete s pomocí posuvného QR algoritmu hodnoty všech vlastních čísel matice $[A]$ z příkladu 1.10. Výsledek zkontrolujte postupem z příkladu 1.5.

1.3. Praktické využití vlastních čísel a příslušných vlastních vektorů v úlohách stavební mechaniky

Vlastní čísla a jejich příslušné vektory mají své uplatnění při řešení technických problémů včetně úloh stavební mechaniky. V následujícím výkladu bude stručně popsána aplikace metody inverzních iterací při řešení stability prutových konstrukcí a naznačeno obdobné uplatnění v dynamice stavebních konstrukcí při řešení vlastních frekvencí prutové konstrukce.

Obsah
16. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

1.3.1. Metoda inverzních iterací

Metoda slouží k výpočtu vlastních čísel a příslušných vlastních vektorů svazku matic $\mathbf{K}$ a $\mathbf{M}$. V úlohách stavební mechaniky matice $\mathbf{K}$ obvykle představuje matici tuhosti konstrukce, zatímco matice $\mathbf{M}$ může být buď geometrickou maticí (v úlohách stability konstrukcí) nebo maticí hmotnosti v dynamice (při výpočtu vlastních frekvencí).

V případě netlumeného kmitání platí:

$$\mathbf{M}\,\ddot{\mathbf{u}} + \mathbf{K}\,\mathbf{u} = \mathbf{0}\,. \tag{1.10}$$

Řešení rovnice (1.10) lze hledat ve tvaru:

$$\mathbf{u(t)} = \mathbf{y}\sin\omega t\,. \tag{1.11}$$

Určením $\ddot{\mathbf{u}}$ a dosazením do (1.10) lze získat rovnici pro výpočet netlumeného kmitání:

$$(-\omega^2\,\mathbf{M} + \mathbf{K})\,\mathbf{y} = \mathbf{0}\,. \tag{1.12}$$

Rovnice (1.12) se může přepsat do tvaru:

$$\mathbf{K}\,\mathbf{y} = \omega^2\,\mathbf{M}\,\mathbf{y}\,. \tag{1.13}$$

Rovnice (1.13) je splněna pokud ω je vlastní frekvencí konstrukce a $\mathbf{y}$ je příslušným vlastním tvarem kostrukce. Protože tyto veličiny nejsou předem známy a je potřebné je vypočítat, místo $\mathbf{y}$ se může použít aproximace $\mathbf{x_i}$.

Poznámka 1.14. Při volbě aproximací $\mathbf{x_i}$ je nutno dbát v úvahu také skutečnost, že zvolené aproximace musí rovněž splňovat okrajové podmínky úlohy.

Obsah
17. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

Nyní je možné určit amplitudy setrvačných sil:

$$\mathbf{S} = \mathbf{M}\ \mathbf{x_i}\ . \tag{1.14}$$

Následuje výpočet vektoru posunutí $\mathbf{x_{i+1}}$ pomocí:

$$\mathbf{K}\ \mathbf{x_{i+1}} = \mathbf{S} = \mathbf{M}\ \mathbf{x_i}\ . \tag{1.15}$$

Proces (1.15) je možné opakovat tak dlouho, až je rozdíl mezi $\mathbf{x_i}$ a $\mathbf{x_{i+1}}$ dostatečně malý. Pak je možné předpokládat, že byla nalezena aproximace vektoru výchylek $\mathbf{y}$ s požadovanou přesností.

Samotnou hodnotu vlastní kruhové frekvence ω je možné najít pomocí vztahu pro Rayleighův koeficient:

$$\omega_2 = \rho(i+1) = \frac{\mathbf{x_{i+1}^T}\ \mathbf{K}\ \mathbf{x_{i+1}}}{\mathbf{x_{i+1}^T}\ \mathbf{M}\ \mathbf{x_{i+1}}}\ . \tag{1.16}$$

1.3.2. Stabilita konstrukcí

Stejným způsobem je možné řešit úlohy stability konstrukcí – jen matice $\mathbf{M}$ bude nahrazena geometrickou maticí tuhosti $\mathbf{K_G}$ podle [1] a koeficient ρ bude mít význam násobitele zatížení. Velikost kritických zatížení $\mathbf{F_{cr}}$ lze získat výpočtem:

$$\mathbf{F_{cr}} = \omega\ \mathbf{F}\ , \tag{1.17}$$

kde $\mathbf{F}$ je vektor zadaných zatížení.

Obsah
18. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

1.3.3. Postup výpočtu

Pro praktický výpočet je potřebné hodnoty $\mathbf{x_i}$ normovat, aby nebyly členy tohoto vektoru příliš malé. Nejvhodnější je normování pomocí matice tuhosti:

$$\mathbf{x_i} = \frac{\mathbf{x_i}}{\sqrt{\mathbf{x_i}^T \mathbf{M}\, \mathbf{x_i}}} \,. \tag{1.18}$$

1.3.4. Počáteční aproximace

Počáteční vektor neznámých výchylek $\mathbf{x_1}$ lze zvolil libovolně. Musí však plnit stejné okrajové podmínky, jako pro vektor výchylek konstrukce.

V dalším výpočtu se výpočet opakuje pro $i = 1...n$:

$$\mathbf{K}\, \mathbf{x_{i+1}} = \mathbf{M}\, \mathbf{x_i}, \tag{1.19}$$

$$\mathbf{x_{i+1}} = \frac{\mathbf{x_{i+1}}}{\sqrt{\mathbf{x_{i+1}}^T \mathbf{M}\, \mathbf{x_{i+1}}}}. \tag{1.20}$$

$$\rho(i+1) = \frac{\mathbf{x_{i+1}^T}\, \mathbf{K}\, \mathbf{x_{i+1}}}{\mathbf{x_{i+1}^T}\, \mathbf{M}\, \mathbf{x_{i+1}}} \,. \tag{1.21}$$

Nakonec se provede test konvergence:

$$\frac{|\rho_{k+1} - \rho_k|}{|\rho_{k+1}|} \leqq 10^{-2\ s} \,, \tag{1.22}$$

kde hodnota $-2\ s$ vyjadřuje míru požadované přesnosti (doporučuje se volit $s \leqq 4$).

Obsah
19. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

Pokud je rovnice (1.22) splněna, může být výpočet ukončen a ρ_{i+1} je hledaným vlastním číslem svazku matic (a tedy $\omega = \sqrt{rho_{i+1}}$ je hledanou vlastní kruhovou frekvencí). Potom je možno také vypočítat vektor výchylek $\mathbf{y_{i+1}}$ odpovídající vlastnímu číslu ρ_{i+1}:

$$\mathbf{y_{k+1}} = \mathbf{M}\ \mathbf{x_{i+1}}\ . \tag{1.23}$$

1.3.5. Aplikace řešené úlohy

Celý problém lze aplikovat prostřednictvím následujícího skriptu:

```
% Lineární stabilita: Eulerův prostě uložený nosník
% Autor: doc. Ing. Jiří Brožovský, Ph.D.
clear; clc;
% Dimenze úlohy (2 stupně volnosti: u,v; 2 uzly na prutu):
ndof=3;
puzlu=2;
% Vstupní údaje: materiálové a průřezové charakteristiky
E=2.10e11;
% Průměr kruhového průřezu d [m]
d=0.1;
% Moment setrvačnosti kruhového průřezu
I=pi/64*d^4;
A=pi/4*d^2;
% Vstupní údaje: geometrie konstrukce
L=6;
uzly=[0 0; 0.25 0; 0.5 0; 0.75 0;
```

Obsah
20. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
1.0  0; 1.25 0; 1.5 0; 1.75 0; 2.0 0];
uzly=L*uzly;
pruty=[1 2; 2 3; 3 4; 4 5; 5 6; 6 7; 7 8; 8 9];
nuzlu=size(uzly,1);
nprutu=size(pruty,1);
% Vstupní údaje: podpory (uzel, směr, velikost):
podpory=[1 1 0; 1 2 0; 9 2 0];
npodpor=size(podpory,1);
% Vstupní údaje: zatížení - síly (uzel, směr, velikost):
sily=[9 1 -1000];
nsil=size(sily,1);
% Výpočet
% Výpočet kódových čísel
kcis=zeros(nprutu,ndof*puzlu);
for i=1:nprutu
  for j=1:puzlu
    for k=1:ndof
      kcis(i,((j-1)*ndof+k))=(pruty(i,j)-1)*ndof+k;
    end
  end
end
% Nulování matic a vektorů
velikost = nuzlu*ndof;
K = zeros(velikost);
M = zeros(velikost);
u = zeros(velikost,1);
```

Obsah
21. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
F = zeros(velikost,1);
% Sestavení a lokalizace matic tuhosti
for i=1:nprutu
  Keg=zeros(puzlu*ndof);
  % Délka prutu
  dx2 = (uzly(pruty(i,1),1)-uzly(pruty(i,2),1))^2;
  dy2 = (uzly(pruty(i,1),2)-uzly(pruty(i,2),2))^2;
  L = sqrt(dx2 + dy2);
  % Matice tuhosti
  Keg=[E*A/L 0 0 -E*A/L 0 0;
  0 12*E*I/L^3 6*E*I/L^2 0 -12*E*I/L^3 6*E*I/L^2;
  0 6*E*I/L^2 4*E*I/L 0 -6*E*I/L^2 2*E*I/L;
  -E*A/L 0  0 E*A/L 0 0;
  0 -12*E*I/L^3 -6*E*I/L^2 0 12*E*I/L^3 -6*E*I/L^2;
  0 6*E*I/L^2 2*E*I/L 0 -6*E*I/L^2 4*E*I/L];
  % Transformace
  s=(uzly(pruty(i,2),2)-uzly(pruty(i,1),2))/L;
  c=(uzly(pruty(i,2),1)-uzly(pruty(i,1),1))/L;
  T=[ c  s  0  0  0  0;
     -s  c  0  0  0  0;
     0  0  1  0  0  0;
     0  0  0  c  s  0;
     0  0  0 -s  c  0;
     0  0  0  0  0  1];
  Ke=T'*Keg*T;
```

Obsah
22. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
  % Lokalizace
  for j=1:(puzlu*ndof)
    for k=1:(puzlu*ndof)
      K(kcis(i,j),kcis(i,k))=K(kcis(i,j),kcis(i,k))+Ke(j,k);
    end
  end
end
% Zatížení
for i=1:nsil
  iuz=sily(i,1);
  ismer=sily(i,2);
  pos=(ndof*(iuz-1))+ismer;
  F(pos) = F(pos)+sily(i,3);
end
% Podpory
for i=1:npodpor
  iuz=podpory(i,1);
  ismer=podpory(i,2);
  pos=(ndof*(iuz-1))+ismer;
  u(pos)=u(pos)+podpory(i,3);
  for j=1:velikost
    K(pos,j)=0.0;
    K(j,pos)=0.0;
  end
  K(pos,pos)=1.0;
```

Obsah
23. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
end
% Řešení soustavy rovnic
u=K\F;
% Výpočet výsledků na prutech
for i=1:nprutu
  ue=zeros(puzlu*ndof,1);
  uel=zeros(puzlu*ndof,1);
  % Získání lokálních vektorů deformací
  for j=1:(puzlu*ndof)
ue(j)=u(kcis(i,j));
  end
  % Transformace
  dx2=(uzly(pruty(i,1),1)-uzly(pruty(i,2),1))^2;
  dy2=(uzly(pruty(i,1),2)-uzly(pruty(i,2),2))^2;
  L=sqrt(dx2 + dy2);
  s=(uzly(pruty(i,2),2)-uzly(pruty(i,1),2))/L;
  c=(uzly(pruty(i,2),1)-uzly(pruty(i,1),1))/L;
  T=[c  s  0  0  0  0;
  -s  c  0  0  0  0;
  0  0  1  0  0  0;
  0  0  0  c  s  0;
  0  0  0 -s  c  0;
  0  0  0  0  0  1];
  uel=T*ue;
  % Matice tuhosti (jen lokální)
```

Obsah
24. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
  Kelb=[E*A/L 0 0 -E*A/L 0 0;
        0 12*E*I/L^3 6*E*I/L^2 0 -12*E*I/L^3 6*E*I/L^2;
        0 6*E*I/L^2 4*E*I/L 0 -6*E*I/L^2 2*E*I/L;
        -E*A/L 0  0 E*A/L 0 0;
        0 -12*E*I/L^3 -6*E*I/L^2 0 12*E*I/L^3 -6*E*I/L^2;
        0 6*E*I/L^2 2*E*I/L 0 -6*E*I/L^2 4*E*I/L];
  % síly v prutech
  Fe=Kelb*uel;
  N=Fe(1);
  % geometrická matice
  Me= N/L * [0 0 0 0 0 0;
      0 6/5 L/10 0 -6/5 L/10;
      0 L/10 2*L^2/15 0 -L/10 -L^2/30;
      0 0 0 0 0 0;
      0 -6/5 -L/10 0 6/5 -L/10;
      0 L/10 -L^2/30 0 -L/10 2*L^2/15];
  % Transformace
  Mg=T'*Me*T;
  for j=1:(puzlu*ndof)
    for k=1:(puzlu*ndof)
      M(kcis(i,j),kcis(i,k)) = M(kcis(i,j),kcis(i,k))+Mg(j,k);
    end
  end
end
% Podpory na matici hmotnosti
```

Obsah
25. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
for i=1:npodpor
  iuz=podpory(i,1);
  ismer=podpory(i,2);
  pos=(ndof*(iuz-1))+ismer;
  for j=1:velikost
    M(pos,j)=0.0;
    M(j,pos)=0.0;
  end
  M(pos,pos)=1.0;
end
% Metoda inverzních iterací
xk=zeros(velikost,1);
for i=1:velikost
  xk(i)=1;
end
% v aproximaci musí být splněny okrajové podmínky
for i=1:npodpor
  iuz=podpory(i,1);
  ismer=podpory(i,2);
  pos=(ndof*(iuz-1))+ismer;
  xk(pos)=0.0;
end
% Iterační cyklus
for i=1:10
  xk1 = K\(M*xk);
```

Obsah
26. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
  xk=xk1/sqrt(xk1'*M*xk1);  % "nové" zk
  % Násobitel - počítá se jako Rayleighuv koeficient
  ro=(xk'*K*xk)/(xk'*M*xk);
  % podmínka konvergence
  if (i>1)
    if (((abs(ro-ro0))/abs(ro))<(1e-8))
      break
    end
  end
  ro0=ro;
end
% Výchylky odpovídající ro
y=M*xk;
% ------------------------------------------------
% Následující výpočet může stanovit
% vlastní frekvenci a tvar pomocí funkce EIG
% ------------------------------------------------
[v,d]=eig(K, M);
D=diag(d);
disp('Násobitel zatížení je: ')
dd=D(1);
di=1;
for iy=1:size(D,1)
  if D(iy)>1.00001
    dd=D(iy);
```

Obsah
27. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
    di=iy;
    break;
  end
end
for iy=di:size(D,1)
  if D(iy)<dd
    if D(iy)>0
      if D(iy)~=1.0
        dd=D(iy);
        di=iy;
      end
    end
  end
end
dd
% Deformovaný tvar (pro přímý nosník)
a=zeros(nuzlu,1);
b=zeros(nuzlu,1);
for i=1:nuzlu
  a(i,1)=i;
  pos=3*(i-1)+2;
  b(i,1)=y(pos);
end
plot(a,b);
disp('Inverzními iteracemi spočteno [kN]: ')
```

Obsah
28. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

```
Fnu=ro
disp('Eulerovo řešení [kN]: ')
L=uzly(nuzlu,1)-uzly(1,1);
Frc=(((pi^2)*E*I)/(L^2))/1000
```

Obsah
29. strana ze 30
Zavřít dokument
Konec
Celá obrazovka/Okno

Literatura

[1] Cook, R.D. – Malkus, D.S. – Plesha, M.E. – Witt, R.J. *Concepts and Applications of Finite Element Analysis*. 4. vydání. Wiley, 2007. (736 s).

[2] MATLAB. Programový systém pro provádění matematických výpočtů. Komerční software, verze R2014b. [on-line]. <http://www.mathworks.com>. The MathWorks, únor 2015.

[3] Sauer T. *Numerical Analysis*. George Mason University. Pearson Education, Inc., 2006. (669 s). ISBN 0-321-26898-9.

[4] Sigmon K. *MATLAB Primer CZ*. Elektronický manuál programového systému MATLAB. Druhé vydání. [on-line]. <https://artax.karlin.mff.cuni.cz/~beda/cz/matlab/primercz/matlab-primer.html>. Department of Mathematics, University of Florida, 1989, 1992. Z anglického originálu přeložil Petr Klášterecký.

Obsah

30. strana ze 30

Zavřít dokument

Konec

Celá obrazovka/Okno