VŠB TECHNICKÁ UNIVERZITA OSTRAVA | FAKULTA STAVEBNÍ

228-0247

Numerické metody ve stavitelství

Opora pro kombinované studium

Jiří Brožovský, Martin Krejsa

Ostrava 2025

Obsah

Kapitola 1

Informace o předmětu

Cíle předmětu vyjádřené dosaženými dovednostmi a kompetencemi

Student získá po absolvování předmětu dovednosti a znalosti v následujících oblastech:

- dovednost aplikovat základní numerické postupy v problémech stavební mechaniky,
- znalost základnách principů a postupů v metodě konečných prvků a pochopení jejich vlivu na přesnost a výstižnost řešení dosažitelných touto metodou.

Anotace

V rámci předmětu „Numerické metody ve stavitelství" se studenti seznámí se základními postupy algoritmizace, které následně využijí pro pochopení vybraných postupů numerické matematiky v aplikaci na úlohy stavební mechaniky. Zejména půjde o práci s polynomy, řešení soustav lineárních rovnic, numerické integrace a o využití těchto postupů při řešení statických úloh metodou konečných prvků.

Povinná literatura

- Krejsa, M., Algoritmizace inženýrských výpočtů, učební texty v obrazovkové verzi i ve verzi pro tisk, VŠB-TU, Ostrava, 2017.
- Brožovský, J., Materna, M. Metoda konečných prvků ve stavební mechanice, VŠB-TU, Ostrava, 2012. On-line: https://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/metoda_kone
- Klaus-Jürgen Bathe, K.-J. Finite Element Procedures (2nd ed.). Klaus-Jürgen Bathe. 2007.

Doporučená literatura

- Press, W. H., Teukolsky, S. A., Vetterling, W. T., Numerical Recipes 3rd Edition: The Art of Scientific Computing 3rd Edition, Cambridge University Press, 2007

Forma způsobu ověření studijních výsledků a další požadavky na studenta

Průběžné odevzdávání úkolů zadaných na cvičení v termínech stanovených vyučujícím, obhajoba semestrálního projektu. Písemný test, diskuse nad odbornými tématy, která byla probrána v rámci přednášek.

Osnova:

1. Úvod do práce s matematickým softwarem. Zadání proměnných, práce s vektory a maticemi, vytvoření skriptu.
2. Základy algoritmizace, vlastnosti algoritmu, elementární algoritmy.
3. Výpočet hodnot funkcí, tabelace a grafické znázornění, určení extrému funkce.
4. Řešení soustav lineárních rovnic přímými metodami, Gaussova a Gaussova-Jordanova eliminační metoda.
5. Řešení soustav lineárních rovnic s využitím iteračních metod, Jacobiova metoda, Gauss-Seidlova metoda, metoda sdružených gradientů.
6. Numerická integrace určitého integrálu, obdélníková, lichoběžníková a Gaussova metoda.
7. Energetické metody ve stavební mechanice, Ritzova metoda.
8. Základní principy metody konečných prvků, odvození konečného prvku pro řešení příhradových konstrukcí.
9. Řešení rovinného problému metodou konečných prvků.
10. Řešení tenkých desek metodou konečných prvků.
11. Řešení prostorových úloh metodou konečných prvků.
12. Izoparametrické konečné prvky, jejich výhody a nevýhody.
13. Modelování konstrukcí na pružném podloží v metodě konečných prvků.

Kapitola 2

Úvod

V této kapitole jsou uvedeny vztahy, které si studenti osvojili v předchozích předmětech. Kapitola je uvedena zejména pro zavedení jednotného označování veličin v tecxtu a jako pomůcka pro připomenutí a dohledání zejména v předchozích semestrech méně využívaných vztahů a rovnic.

2.1 Opakování: základní předpoklady v lineární mechanice

Nejrve si připomeneme základní předpoklady, které budou využity i v metodě konečných prvků:

- látka studovaného tělesa je **spojitá**
- látka je **homogenní** (ve všech místech stejné vlastnosti)
- látka je **isotropní** (ve všech směrech stejné vlastnosti)
- látka se chová **lineárně pružně** (tzv. Hookeův zákon)
- těleso je vystaveno jen **malým deformacím**

Pokud platí výše uvedené, pak lze při řešení úlohy použít:

- princip superpozice,
- princip úměrnosti.

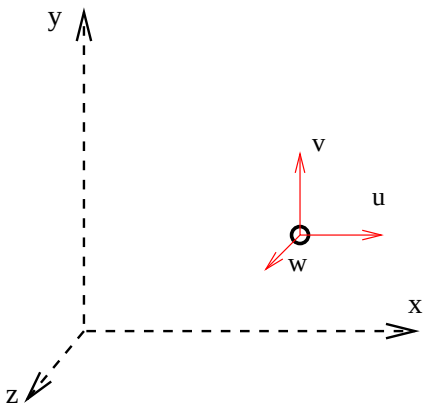
Lineárně pružný materiál může být:

- **isotropní**: ve všech směrech stejné vlastnosti,
- **anisotropní**: v různých směrech různé vlastnosti,
- **ortotropní**: různé vlastnosti ve vzájemně kolmých směrech.

2.2 Základní veličiny v teorii pružnosti

V dalším textu budeme pracovat v karzézské soustavě souřadnic s osami x, y, z.

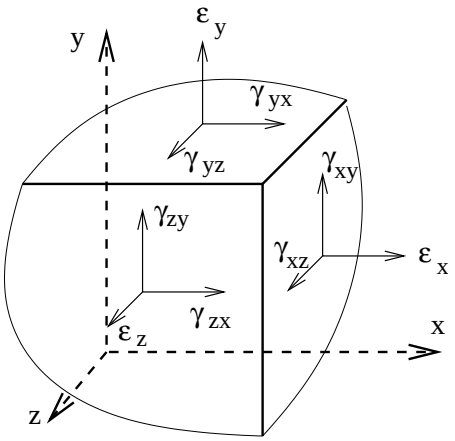
2.2.1 Vektor posunutí



Každý bodě prostoru se může posouvat ve 3 směrech.

$$\mathbf{u} = \left\{ \begin{array}{c} u \\ v \\ w \end{array} \right\} \tag{2.1}$$

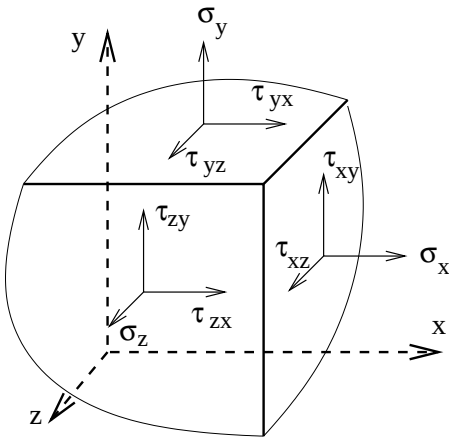
2.2.2 Vektor deformací



Předpokládáme 3 nezávislá poměrná prodloužení/zkrácení a 3 nezávislá zkosení.

$$\boldsymbol{\varepsilon} = \left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{yz} \\ \gamma_{zx} \\ \gamma_{xy} \end{array} \right\} \tag{2.2}$$

2.2.3 Vektor napětí

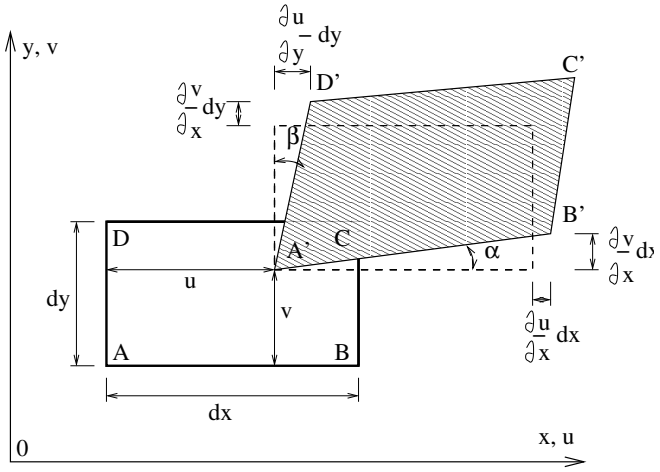


Předpokládáme 3 nezávislá normálová napětí a 3 nezávislá smyková napětí.

$$\boldsymbol{\sigma} = \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{yz} \\ \tau_{zx} \\ \tau_{xy} \end{Bmatrix} \tag{2.3}$$

2.3 Geometrické vztahy

Vyjadřují vztahy mezi posunutími a deformacemi.



S využitím obrázku můžeme napsat:

$$\varepsilon_x = \frac{A'B' - AB}{AB} = \frac{(x + dx + u + \frac{\partial u}{\partial x}dx) - (x + u) - dx}{dx} = \frac{\partial u}{\partial x} \tag{2.4}$$

Ostatní normálové deformace budou mít analogický tvar:

$$\varepsilon_x = \frac{\partial u}{\partial x} \qquad \varepsilon_y = \frac{\partial v}{\partial y} \qquad \varepsilon_z = \frac{\partial w}{\partial z}, \tag{2.5}$$

a podobně je možné získat smykové deformace:

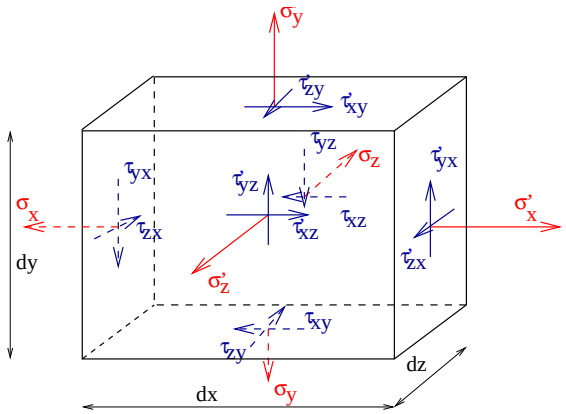
$$\gamma_{yz} = \gamma_{zy} = \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \tag{2.6}$$

$$\gamma_{zx} = \gamma_{xz} = \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \tag{2.7}$$

$$\gamma_{xy} = \gamma_{yx} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}. \tag{2.8}$$

2.4 Diferenciální podmínky rovnováhy

Diferenciální podmínky rovnováhy v metodě konečných prvků při odvozování zpravidla nevyužíváme. Tyto podmínky však musí být splněny v každém tělese, proto mohou posloužit pro kontrolu.



Na obrázku značí:

$$\sigma_x{}' = \sigma_x + \frac{\partial \sigma_x}{\partial x} dx, \quad \tau_{xy}{}' = \tau_{xy} + \frac{\partial \tau_{xy}}{\partial x} dy, ... \tag{2.9}$$

S využitím obrázku můžeme psát:

$$\sum F_{i,y} = (\sigma'_x - \sigma_x)\ dy\ dz + (\tau_{xy} - \tau'_{xy})\ dx\ dz + (\tau_{xz} - \tau_{xz}{}')\ dx\ dy = 0 \tag{2.10}$$

a

$$(\sigma_x - \sigma_x - \frac{\partial \sigma_x}{\partial x} dx)\ dy\ dz + (\tau_{xy} - \tau_{xy} - \frac{\partial \tau_{xy}}{\partial y} dy)\ dx\ dz + (\tau_{xz} - \tau_{xz} - \frac{\partial \tau_{xz}}{\partial z} dz)\ dx\ dy = 0 \tag{2.11}$$

A po úpravě:

$$\frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} = 0 \tag{2.12}$$

Finálně upravíme do podoby:

$$\begin{aligned}
\frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} + X &= 0 \\
\frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} + Y &= 0 \\
\frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \sigma_z}{\partial z} + Z &= 0
\end{aligned} \tag{2.13}$$

kde X,Y, Z jsou tzv. objemové síly (např. vlastní tíha tělesa nebo prvku).

2.5 Vzájemnost smykových napětí

Uvedené vztahy obecně neplatí:

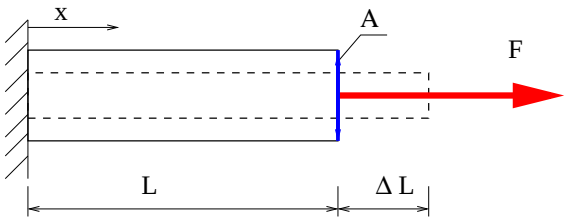
$$\begin{aligned}
\tau_{yz} &= \tau_{zy}, \\
\tau_{zx} &= \tau_{xz}, \\
\tau_{xy} &= \tau_{yx}.
\end{aligned}$$

Předpoklad o vzájemnosti smykových napětí se odvozuje z **přibližného** splnění momentových podmínek rovnováhy na elementu tělesa. Na smykové deformace se pohlíží obdobně.

2.6 Fyzikální rovnice

Vyjadřují vztahy mezi napětími a deformacemi.

2.7 Hookeův zákon v 1D (tah/tlak):



Pro úlohu prostého tahu nebo tlaku můžeme za předpokladu neměnnosti průřezové plochy **A** psát:

$$\varepsilon_x = \frac{\sigma_x}{E}$$

Připomeňme si i další související vztahy z předchozích předmětů:

$$\varepsilon_x = \frac{\Delta L}{L} = \frac{\partial L}{\partial x} \tag{2.14}$$

$$\sigma_x = \frac{F}{A} = E\ \varepsilon_x \tag{2.15}$$

2.8 Hookeův zákon v prostoru:

V prostoru musíme uvážit vliv deformace v další směrech. Míra tohoto vlivu může být popsána součinitelem příčné kontrakce ν:

$$\begin{aligned} \varepsilon_x &= \frac{1}{E}\left[\sigma_x - \nu\left(\sigma_y + \sigma_z\right)\right], \quad \gamma_{yz} = \frac{\tau_{yz}}{G} \\ \varepsilon_y &= \frac{1}{E}\left[\sigma_y - \nu\left(\sigma_x + \sigma_z\right)\right], \quad \gamma_{xz} = \frac{\tau_{xz}}{G} \\ \varepsilon_z &= \frac{1}{E}\left[\sigma_z - \nu\left(\sigma_x + \sigma_y\right)\right], \quad \gamma_{xy} = \frac{\tau_{xy}}{G} \end{aligned} \tag{2.16}$$

2.9 Shrnutí opakování

V teorii pružnosti potřebujeme nalézt **15 neznámých veličin:**
3 složky posunutí **u**
6 složek deformací $\boldsymbol{\varepsilon}$
6 složek napětí $\boldsymbol{\sigma}$

A k dispozici máme **15 rovnic:**
6 geometrických rovnic
6 fyzikálních rovnic
3 podmínky rovnováhy

2.10 Úkol k procvičení:

- Upravte geometrické a fyzikální rovnice tak, aby odpovídaly 2D úloze rovinné napjatosti ($\sigma_z = \tau_{zx} = \tau_{yz} = 0$).

Kapitola 3

Matlab

Kapitola je zaměřena na:

- seznámení s uživatelským prostředím systému **Matlab**,
- definici a správu proměnných,
- úvodní ukázku tvorby algoritmu s využitím **logického rozhodování**.

MATLAB [9] (viz obr. 3.1) je programové prostředí využívající skriptovací programovací jazyk pro vědeckotechnické numerické výpočty, modelování a návrhy algoritmů. Nástavbou systému MATLAB je Simulink — program pro simulaci a modelování dynamických systémů, který využívá algoritmy Matlabu pro numerické řešení především nelineárních diferenciálních rovnic.

Název MATLAB vznikl zkrácením slov MATrix LABoratory (volně přeloženo jako „maticová laboratoř“), což vychází ze skutečnosti, že klíčovou datovou strukturou při výpočtech v systému MATLAB jsou matice.

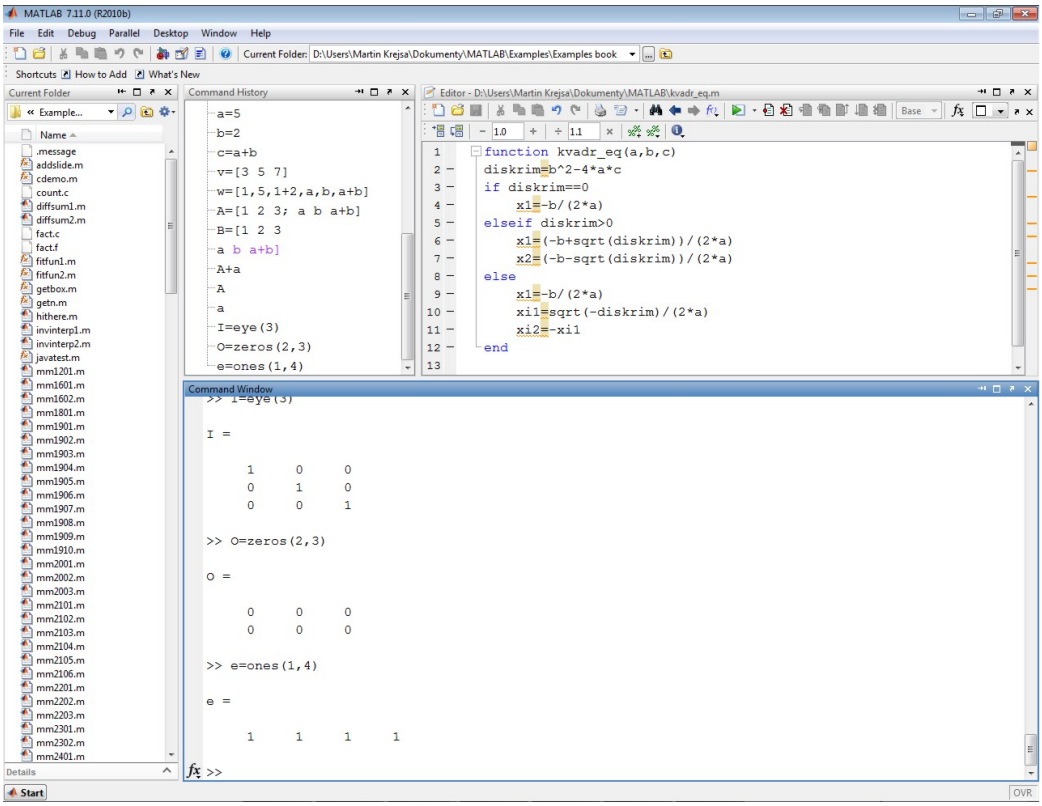
MATLAB je komerční software, takže vítanou alternativou pro tvorbu algoritmů s využitím kompatibilních příkazů a skriptů je software nazývaný Octave, který je zdarma ke stažení např. z [5]. Program Octave vznikl jako open-source a existuje tedy v mnoha mutacích pro platformy Unix (Linux), Mac OS nebo MS Windows.

Samotný systém MATLAB obsahuje řadu příkazů pro správu proměnných, základní operace algebry, výpočty s vektory i maticemi, a příkazy vyšší matematiky. Vzhledem k poněkud rozsáhlým možnostem je k dispozici rozsáhlá nápověda, se kterou lze pracovat s pomocí příkazů obsažených v tab. 3.1.

Název proměnné	*Popis*
`help`	vyvolání obsahu nápovědy
`help příkaz`	vyvolání nápovědy, vztahující se ke konkrétnímu příkazu
`lookfor výraz`	výpis všech položek nápovědy, které obsahují hledaný výraz
`info`	informace o firmě MathWorks

Tabulka 3.1: Přehled příkazů souvisejících s nápovědou

MATLAB si uchovává přesnou reprezentaci veškerých číselných hodnot, se kterými pracuje. Na uživateli však záleží, v jakém formátu jsou tyto hodnoty zobrazovány, k čemuž využívá příkaz `format` s příslušnou specifikací podle tab. 3.2.



Obrázek 3.1: Pracovní plocha programu MATLAB

Příkaz	*Popis formátu*	*Zobrazení čísla π v daném formátu*
format short	pevná desetinná čárka, 5 zobrazených číslic	3.1416
format long	pevná desetinná čárka, 15 zobrazených číslic (typ proměnné double), resp. 7 (typ single)	3.141592653589793
format shorte	pohyblivá desetinná čárka, 5 zobrazených číslic	3.1416e+000
format longe	pohyblivá desetinná čárka, 15 zobrazených číslic (typ proměnné double), resp. 7 (typ single)	3.141592653589793e+000
format shorteng	inženýrský formát, 5 zobrazených číslic a exponent	3.1416e+000
format longeng	inženýrský formát, 16 zobrazených číslic a exponent	3.14159265358979e+000
format rat	vyjádření výsledku formou zlomku (implicitní nastavení)	355/113
format hex	zobrazení v šestnáctkové soustavě	400921fb54442d18

Tabulka 3.2: Přehled možných typů formátů zobrazení číselných hodnot

3.1 Zadání proměnných

Základním typem proměnné v systému MATLAB je matice. Jednoduchá proměnná, obsahující jednu hodnotu, je interpretována jako matice typu $[1, 1]$. Proměnné se zadávají příkazy, které se definují v příkazovém řádku programu. Končí-li příkaz středníkem, výsledek příkazu se nezobrazí. Pokud je součástí zadání základní aritmetická operace, lze pro ně využít symboly z tab. 3.3. Desetinnou čárku lze zadat pomocí tečky.

Definice proměnné probíhá automaticky přiřazením její hodnoty pomocí rovnítka, např.:

```
a=5
b=2
c=a+b
d='Výsledek'
```

Operace	*Symbol*	*Příklad*
Součet	`+`	`4+11`, `a+b`
Rozdíl	`-`	`18-5`, `a-b`
Součin	`*`	`7.13*5`, `a*b`
Podíl ($\frac{1}{8}$)	`/` nebo `\`	`1/8 = 8\1`
Mocnina (2^8)	`^`	`2^8`

Tabulka 3.3: Přehled symbolů základních aritmetických operací

Pro přiřazení hodnoty proměnným lze využít i předem nadefinovaných speciálních proměnných z tab. 3.4

Název proměnné	*Popis*
`ans`	proměnná, která obsahuje výsledek aritmetické operace
`pi`	Ludolfovo číslo π
`inf`	označení pro nekonečno ∞
`nargin`	počet vstupních parametrů dané funkce
`nargout`	počet výstupních parametrů dané funkce
`eps`	nejmenší použitelné číslo v daném formátu
`realmin`	absolutně nejmenší použitelné kladné reálné číslo
`realmax`	absolutně největší použitelné kladné reálné číslo

Tabulka 3.4: Přehled předem nadefinovaných proměnných

K dispozici je rovněž množství elementárních funkcí s jedním vstupním parametrem. Přehled nejzákladnějších je uveden v tab. 3.5.

3.2 Vektory a matice

K definici vektoru se používají hranaté závorky. Mezery nebo čárky oddělují prvky v řádku, např.:

Název funkce	*Popis*
`abs(x)`	absolutní hodnota z x
`cos(x)`,`sin(x)`,`tan(x)`	goniometrické funkce (parametr x v radiánech)
`acos(x)`,`asin(x)`,`atan(x)`	inverzní goniometrické funkce
`exp(x)`	exponenciální funkce (^{x})
`log(x)`	přirozený logaritmus x
`log10(x)`	dekadický logaritmus x
`sqrt(x)`	druhá odmocnina x

Tabulka 3.5: Přehled základních elementárních funkcí

```
u=[3 5 7]
v=[1,5,1+2,a,b,a+b]
w=[0,pi,2*pi]
```

K vytvoření vektorů lze využít i pokročilejších technik, popsaných v tab. 3.6

Příkaz	*Popis*	*Výsledek*
`x=[1:5]`	vytvoří řádkový vektor `x`, začínající hodnotou 1, končící hodnotou 5, přičítá se hodnota 1	`x=[1,2,3,4,5]`
`x=[2:3:11]`	vytvoří řádkový vektor `x`, začínající hodnotou 2, končící hodnotou 11, přičítá se hodnota 3	`x=[2,5,8,11]`
`x=linspace(1,25,5)`	vytvoří řádkový vektor `x`, začínající hodnotou 1, končící hodnotou 25, vektor obsahuje 5 prvků	`x=[1,7,13,19,25]`

Tabulka 3.6: Příkazy pro konstrukci vektorů

Definice matice se provádí podobným způsobem, jak je tomu u vektorů. Řádky se ale oddělují pomocí středníku nebo klávesou `<Enter>`, např:

```
A=[1 2 3; a b a+b]
B=[1 2 3
a b a+b]
```

MATLAB rozlišuje malá a velká písmena v názvech proměnných. Příkaz

```
A+a
```

vygeneruje pro předchozí zadání matice **A** a proměnné **a**:

```
ans =
     6     7     8
    10     7    12
```

tzn., že ke každému prvku matice **A** se přičte obsah proměnné **a**, tedy hodnota 5.

K přímému generování matic nebo vektoru s předepsaným rozměrem lze využít některé ze standardních funkcí systému MATLAB, např.:

```
I=eye(3)
O=zeros(2,3)
e=ones(1,4)
```

V prvním případě je vygenerována čtvercová matice s hodnotami 1 na diagonále a 0 mimo diagonálu. Následuje vytvoření matice, resp. vektoru s hodnotou 0 resp. 1 ve všech jejích prvcích.

3.2.1 Přístup k maticím a vektorům

Po vytvoření vektoru nebo matice se lze k jednotlivým prvkům odkazovat. V případě předchozí matice A=[1 2 3; a b a+b] (konkrétní obsah je tedy [1 2 3; 5 2 7]) jsou na ukázku možné variace odkazů popsány v tab. 3.7.

Příkaz	*Popis*	*Výsledek*
`A(2,1)`	prvek z 2. řádku a 1. sloupce matice `A`	5
`A(2,[1 3])`	1. a 3. prvek z 2. řádku matice `A`	5, 7
`A(1,1:3)`	prvky 1 až 3 na 1. řádku matice `A`	1, 2, 3
`A(1,1:end)`	stejné jako předchozí	1, 2, 3
`A(1,:)`	stejné jako předchozí	1, 2, 3
`A(2,1:2:3)`	1. a 3. prvek z 2. řádku matice `A`, `a:b:c` definuje aritmetickou posloupnost s prvním prvkem rovným `a`, posledním rovným `c`, `b` je diference mezi sousedními prvky (pokud je rovno 1, nemusí se uvádět)	5, 7
`A(1:end,2:end)`	2. a 3. prvek z obou řádků (submatice)	2, 3; 2, 7

Tabulka 3.7: Příkazy pro přístup k prvkům vektorů a matic

3.2.2 Maticové operace

S vektory i maticemi lze provádět maticové operace. Transponování matic se provádí s využitím operátoru ' (apostrof), např. příkaz `A'` transponuje původní matici `A`:

```
ans =
   1   5
   2   2
   3   7
```

K provedení operací mezi jednotlivými prvky matice nebo vektoru se musí umístit znak `.` (tečka) před operátor. Například pro dříve zadaný vektor `u=[3 5 7]` je výsledkem příkazu `u*u'`:

```
ans =  83
```

kdežto po zadání příkazu `u.*u` se zobrazí vektor:

```
ans =
    9   25   49
```

Podobně lze provádět i další operace s vektory a maticemi prvek po prvku, jak je pro obecně definované vektory $a = a_1, a_2 a_n$; $b = b_1, b_2 b_n$ a skalár c uvedeno v tab. 3.8.

Operace	*Příkaz*	*Výsledek*
skalární součet	`a+c`	$a_1 + c, a_2 + c a_n + c$
skalární součin	`a*c=a.*c`	$a_1 \cdot c, a_2 \cdot c a_n \cdot c$
vektorový součet	`a+b`	$a_1 + b_1, a_2 + b_2 a_n + b_n$
vektorový součin	`a.*b`	$a_1 \cdot b_1, a_2 \cdot b_2 a_n \cdot b_n$
vektorové dělení zleva	`a./b`	$a_1/b_1, a_2/b_2 a_n/b_n$
vektorové dělení zprava	`a.\b`	$a_1 n b_1, a_2 n b_2 a_n n b_n$
skalární mocnění	`a.^c`	$a_1^c, a_2^c a_n^c$
skalární mocnění	`c.^a`	$c^{a_1}, c^{a_2} c^{a_n}$
vektorové mocnění	`a.^b`	$a_1^{b_1}, a_2^{b_2} a_n^{b_n}$

Tabulka 3.8: Operace s vektory a maticemi prvek po prvku

K dispozici je také několik maticových funkcí. Výčet nejzákladnějších obsahuje tab. 3.9.

Název funkce	*Popis*
`det(A)`	determinant matice `A`
`inv(A)`	výpočet inverzní matice k `A`
`diag(A)`	výpis prvků na diagonále matice `A`

Tabulka 3.9: Přehled základních maticových funkcí

3.3 Správa proměnných

Pro správu zadaných proměnných, vektorů a matic je v prostředí systému MATLAB zavedeno několik příkazů:

- příkazy `who` nebo `whos` vypíší seznam všech aktuálně definovaných proměnných (druhý z nich i s jejich velikostmi),
- příkaz `size(proměnná)` vrátí rozměry dané proměnné,
- příkaz `clear(proměnná)` vymaže danou proměnnou z paměti,
- příkaz `clear` bez parametru vymaže všechny zadané proměnné z paměti.

3.4 Využití grafického výstupu

Základním nástrojem v grafickém režimu je příkaz `plot`. Syntaxe tohoto příkazu je `plot(x,y,options)`, kde `x` a `y` jsou souřadnice bodů, které se mají vykreslit. Parametr `options` definuje způsob, jakým se bude grafický výstup provádět. Obsahuje znaky pro definici barvy a stylu vykreslení bodů, případně jejich spojnice:

- barva bodů a jejich spojnice: **b** (modrá), **g** (zelená), **r** (červená), **c** (modrozelená), **m** (fialová), **y** (žlutá), **k** (černá), **w** (bílá);
- styl spojnice bodů: `-` (body budou spojeny plnou čarou), `:` (body budou spojeny tečkovanou čarou), `-.` (body budou spojeny čerchovanou čarou), `--` (body budou spojeny přerušovanou čarou);
- styl vykreslení bodů: `*` (body budou vykresleny jako hvězdičky), `.` (tečky), `x` (křížky), `+` (znaky plus), `o` (kolečka), `s` (čtverce), `d` (kosočtverce).

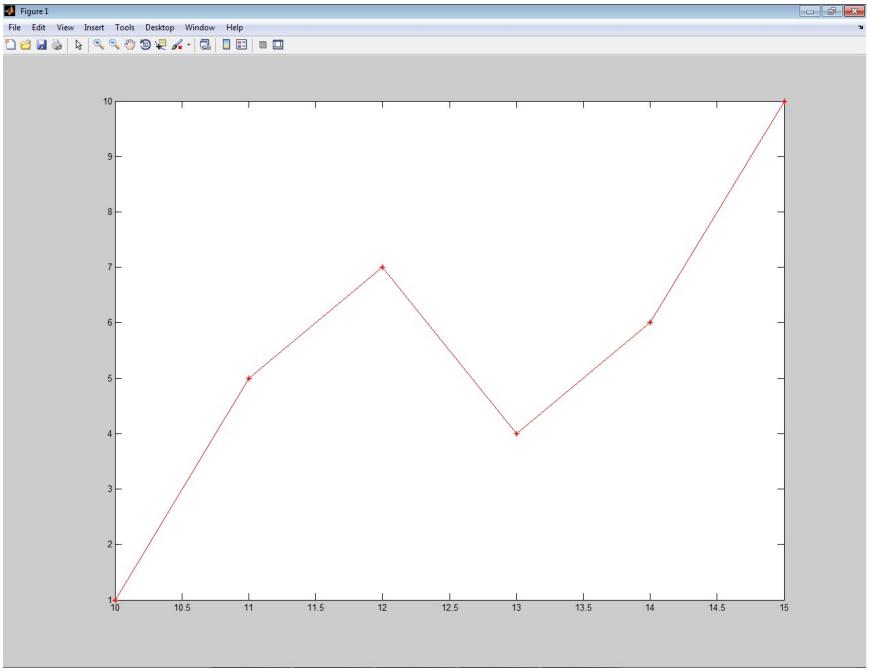
Pro vykreslení spojnice pěti bodů o souřadnicích:

x	10	11	12	13	14	15
y	1	5	7	4	6	10

poslouží příkaz:

```
plot([10:15],[1,5,7,4,6,10],'r-*')
```

Výsledný grafický výstup je zobrazen na obr .3.2.



Obrázek 3.2: Ukázka grafického výstupu z programu MATLAB

V souvislosti s grafickým prostředím je užitečný i příkaz `hold on`, který umožňuje grafický výstup více příkazů do jednoho okna (do původního stavu pak vše vrátí příkaz `hold off`).

3.4.1 Graf funkce

Při vytváření grafu funkce je nutné nejprve diskretizovat osu x. V získaných bodech je pak nutno určit odpovídající funkční hodnoty $f(x)$. Graf diskretizované funkce se pak zobrazí již známým příkazem `plot`.

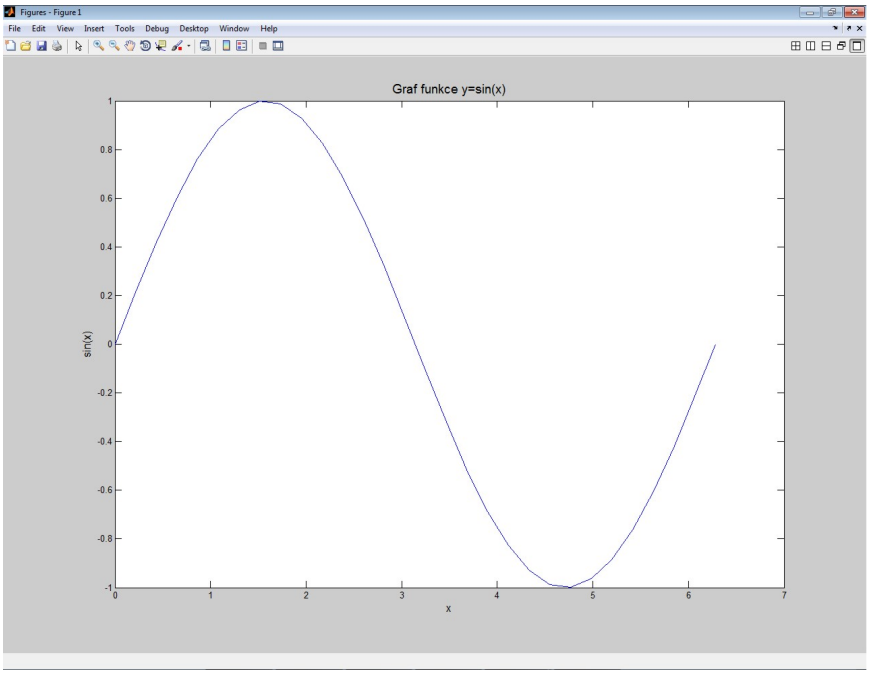
Vykreslení grafu funkce sinus lze s využitím příkazu `plot` provést následující sekvencí příkazů:

```
x=linspace(0,2*pi,30);
y=sin(x);
plot(x,y,'b-')
```

Graf lze doplnit o nadpis, případně popisem os, následujícím doplněním:

```
title('Graf funkce y=sin(x)');
xlabel('x');
ylabel('sin(x)')
```

Výsledné zobrazení grafu funkce sinus je na obr. 3.3.



Obrázek 3.3: Graf funkce sinus

3.5 Vytvoření skriptů

V programu MATLAB lze posloupnost příkazů zadávat jednoduše do příkazového řádku, systém interaktivně tyto povely postupně zpracovává. Pokud by se měl ale výpočet provádět opakovaně, musí se veškeré příkazy zadávat znovu, což je pracné a nevýhodné.

Řešením je uložení sledu matlabovských příkazů, tzv. `skriptů` do textového souboru s příponou `*.m` (tzv. m-soubor). S využitím příkazů, uložených v m-souboru, se výpočet spustí zadáním názvu souboru (bez přípony), přičemž se prohledává aktuální adresář a adresáře uvedené v seznamu systémové proměnné `path`.

Tímto způsobem lze vytvořit v samostatném souboru i speciální výpočetní funkci (tzv. „m-funkci“), kterou lze vyvolat z příkazového řádku zadáním názvu funkce, případně seznamem vstupních parametrů v závorce, oddělených čárkami. **Název souboru by měl být shodný s názvem funkce v jeho záhlaví.**

M-soubory mohou obsahovat i tzv. řídící příkazy, typické pro pokročilejší programovací platformy. Patří k nim zejména příkazy cyklu a logického rozhodování.

3.5.1 Příkazy cyklu

MATLAB umožnuje použití dvou typů programových cyklů:

- **for cyklus**, jehož syntaxe je

```
for i=počáteční hodnota:krok:koncová hodnota
    posloupnost příkazů
end
```

- **while cyklus** se syntaxí:

```
while logická podmínka
    posloupnost příkazů
end
```

Podrobnější výklad tvorby algoritmů s využitím obou typů cyklů je obsažen v následujících kapitolách.

3.5.2 Logické rozhodování

Syntaxe posloupnosti příkazů, které mají být rozděleny do tří bloků podle výsledků logického rozhodování, je následující:

```
if logická podmínka 1
    posloupnost příkazů 1
elseif logická podmínka 2
    posloupnost příkazů 2
else
    posloupnost příkazů 3
end
```

Při definování logických podmínek, jejichž výsledkem je buď „pravda“ nebo „nepravda“, je možno využít následující logické operátory: & (`and`), — (`or`), ˜ (`not`). Relačními operátory mohou být: <, <=, >=, > a == (rovnost).

K procvičení tvorby m-funkce poslouží řešení kvadratické rovnice, kterou lze s využitím logických operátorů vytvořit následujícím způsobem:

```
function kvadr_eq(a,b,c)
diskrim=b^2-4*a*c
if diskrim==0
    x1=-b/(2*a)
elseif diskrim>0
    x1=(-b+sqrt(diskrim))/(2*a)
    x2=(-b-sqrt(diskrim))/(2*a)
else
    x1=-b/(2*a)
    xi1=sqrt(-diskrim)/(2*a)
    xi2=-xi1
end
```

Funkce, kterou lze v příkazovém řádku programu MATLAB vyvolat se vstupními parametry zadáním např. `kvadr_eq(5,10,1)`, pak vrátí dva reálné kořeny kvadratické rovnice:

```
diskrim =
    80

x1 =
  -0.105572809000084

x2 =
  -1.894427190999916
```

Při vyvolání vytvořené funkce s parametry `kvadr_eq(10,5,1)` lze naopak získat výsledek:

```
diskrim =
   -15

x1 =
  -0.250000000000000

xi1 =
   0.193649167310371

xi2 =
  -0.193649167310371
```

Dosažené výsledky lze zkontrolovat speciální funkcí systému MATLAB s názvem `roots(c)`, kde `c` je vektor koeficientů polynomu seřazených sestupně podle mocnin x. Při obecném vyjádření polynomu n-tého stupně:

$$f(x) = c_n \cdot x^n + c_{n-1} \cdot x^{n-1} + \cdots + c_1 \cdot x + c_0 \,, \tag{3.1}$$

obsahuje vektor `c` prvky $c_n, c_{n-1}c_1, c_0$.

Pro první výpočet příkladu 3.5.2 je pak sled příkazů následující:

```
c=[5,10,1]
roots(c)
```

s výsledkem

```
ans =
 -1.894427190999916
  -0.105572809000084
```

Pro popis práce s programovým systémem MATLAB, resp. Octave, existuje množství publikací. Některé z nich jsou přeloženy do češtiny a v elektronické podobě jsou volně přístupné (např. [6, 16]).

Kapitola 4

Základy algoritmizace

Kapitola by měla sloužit:

- k vysvětlení pojmu algoritmus,
- k položení základů pro tvorbu jednoduchých algoritmů.

Algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy. Pojem algoritmu se nejčastěji objevuje při programování, kdy se jím myslí teoretický princip řešení problému (oproti přesnému zápisu v konkrétním programovacím jazyce). Obecně se ale algoritmus může objevit v jakémkoli jiném vědeckém odvětví. Jako jistý druh algoritmu se může chápat i např. kuchyňský recept. V užším smyslu se slovem algoritmus rozumí pouze takové postupy, které splňují požadavky, označované vlastnostmi algoritmu.

4.1 Vlastnosti algoritmu

Následující výčet vlastností algoritmu je uveden např. v [17]Wiki1:

- **Konečnost** (finitnost)

  Každý algoritmus musí skončit v konečném počtu kroků. Tento počet kroků může být libovolně velký (podle rozsahu a hodnot vstupních údajů), ale pro každý jednotlivý vstup musí být konečný. Postupy, které tuto podmínku nesplňují, se mohou nazývat výpočetní metody. Speciálním příkladem nekonečné výpočetní metody je reaktivní proces, který průběžně reaguje s okolním prostředím. Někteří autoři však mezi algoritmy zahrnují i takovéto postupy.

- **Obecnost** (hromadnost, masovost, univerzálnost)

  Algoritmus neřeší jeden konkrétní problém (např. „jak spočítat $3 \cdot 7$“), ale obecnou třídu obdobných problémů (např. „jak spočítat součin dvou celých čísel“), takže má širokou množinu možných vstupů.

- **Determinovanost**

Každý krok algoritmu musí být jednoznačně a přesně definován; v každé situaci musí být naprosto zřejmé, co a jak se má provést, jak má provádění algoritmu pokračovat, takže pro stejné vstupy lze pokaždé získat stejné výsledky. Protože běžný jazyk obvykle neposkytuje naprostou přesnost a jednoznačnost vyjadřování, byly pro zápis algoritmů navrženy programovací jazyky, ve kterých má každý příkaz jasně definovaný význam. Vyjádření výpočetní metody v programovacím jazyce se nazývá *program*. Některé algoritmy ale determinované nejsou, např. pravděpodobnostní algoritmy s (pseudo)náhodným výběrem.

- **Výstup** (resultativnost)

  Algoritmus má alespoň jeden výstup, veličinu, která je v požadovaném vztahu k zadaným vstupům, a tím tvoří odpověď na problém, který algoritmus řeší (algoritmus vede od zpracování hodnot k výstupu).

- **Elementárnost**

  Algoritmus se skládá z konečného počtu jednoduchých (elementárních) kroků.

V praxi jsou proto předmětem zájmu hlavně takové algoritmy, které jsou v nějakém smyslu kvalitní. Takové algoritmy splňují různá kritéria, měřená např. počtem kroků potřebných pro běh algoritmu, jednoduchost, efektivitu či eleganci. Problematikou efektivity algoritmů, tzn. metodami, jak z několika známých algoritmů řešících konkrétní problém vybrat ten nejlepší, se zabývají odvětví informatiky nazývané *algoritmická analýza* a *teorie složitosti*.

4.2 Elementární algoritmy

Následující výklad je zaměřen na několik elementárních algoritmů, které jsou základem mnoha výpočetních technik a postupů.

4.2.1 Záměna obsahu dvou proměnných

Za nejjednodušší algoritmus lze považovat postup popisující záměnu obsahu dvou proměnných. K této operaci je potřeba třetí, pomocné proměnné. Samotný algoritmus lze pro proměnné a a b i pro pomocnou proměnnou c popsat následovně:

```
c=a;
a=b;
b=c;
```

Byť se jedná o skutečně elementární algoritmus, lze jej společně s logickým rozhodováním (kap. 3.5.2) využít např. pro vzestupné setřídění vektoru d o 3 prvcích (c je opět pomocná proměnná):

```
function setrid(d)
  if length(d)==3
    if d(1)>d(2)
      c=d(1);
      d(1)=d(2);
```

```
      d(2)=c;
    end
    if d(2)>d(3)
      c=d(2);
      d(2)=d(3);
      d(3)=c;
    end
    if d(1)>d(2)
      c=d(1);
      d(1)=d(2);
      d(2)=c;
    end
    d
  end
end
```

Proveďte vzestupné třídění vektoru `[8 24 2]` s využitím funkce pro třídění vektoru o třech prvcích, vytvořené v předchozím oddíle.

Nejprve je potřeba přiřadit vstupní hodnoty vektoru d:

```
d=[8 24 2]
```

Pak je možno vyvolat funkci `setrid`:

```
setrid(d)
```

Výsledkem je:

```
d =
     2     8    24
```

Tento způsob třídění je samozřejmě velmi neefiktivní a v žádném případě jej nelze považovat za univerzální. Je však vhodný pro vysvětlení základů algoritmizace. Kvalitním algoritmům, které jsou zaměřeny na třídění vektorů, bude věnována jedna z následujících kapitol.

Kapitola 5

Výpočet hodnot funkcí

Kapitola má za cíl demonstrovat:

- základní přístupy k tvorbě algoritmů pro výpočet hodnot funkcí,
- odlišnosti mezi algoritmy z hlediska jejich efektivity,
- využití cyklů **for** při tvorbě algoritmů,
- provádění tzv. „tabelace funkce“.

5.1 Výpočet hodnoty polynomu

Následující ukázka jednoho z nejzákladnějších algoritmů pro výpočet polynomu byla převzata z [15]Sa06$_N$*uAn*1.*Spočívávnalezenínejlepšíhozpůsobuurčeníhodnotypolynomu* : $f(x) = 2 \cdot x$ pro konkrétně zadanou hodnotu x, např. $x = \frac{1}{2}$. Nejlepším způsobem výpočtu je chápán algoritmus s nejmenším počtem matematických operací.

Metoda 1:

První způsob vede k přímému určení požadované hodnoty:

$$f(x = \frac{1}{2}) = 2 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} + 3 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} - 3 \cdot \frac{1}{2} \cdot \frac{1}{2} + 5 \cdot \frac{1}{2} - 1 = \frac{5}{4} \,. \tag{5.2}$$

Při tomto výpočtu bylo provedeno 10 operací násobení a 4 pro součet/rozdíl.

Metoda 2:

Poněkud výhodnější řešení představuje postup, při němž jsou postupně určeny mocniny vstupního parametru x:

$$\frac{1}{2} \cdot \frac{1}{2} = \left(\frac{1}{2}\right)^2 \tag{5.3}$$

$$\left(\frac{1}{2}\right)^2 \cdot \frac{1}{2} = \left(\frac{1}{2}\right)^3 \tag{5.4}$$

$$\left(\frac{1}{2}\right)^3 \cdot \frac{1}{2} = \left(\frac{1}{2}\right)^4 \tag{5.5}$$

Nyní lze provést finální výpočet polynomu:

$$f(x = \frac{1}{2}) = 2 \cdot \left(\frac{1}{2}\right)^4 + 3 \cdot \left(\frac{1}{2}\right)^3 - 3 \cdot \left(\frac{1}{2}\right)^2 + 5 \cdot \left(\frac{1}{2}\right) - 1 = \frac{5}{4}\,. \tag{5.6}$$

Tento algoritmus je poněkud efektivnější než v prvním případě, neboť bylo provedeno celkem 7 operací násobení a stejný počet 4 operací pro součet/rozdíl.

Metoda 3:

Poslední způsob výpočtu, označovaný jako *Hornerova metoda*, předpokládá následující úpravu řešeného polynomu (5.1):

$$\begin{aligned} f(x) &= -1 + x \cdot (5 - 3 \cdot x + 3 \cdot x^2 + 2 \cdot x^3) = \\ &= -1 + x \cdot (5 + x \cdot (-3 + 3 \cdot x + 2 \cdot x^2)) = \\ &= -1 + x \cdot (5 + x \cdot (-3 + x \cdot (3 + 2 \cdot x)))\,. \end{aligned} \tag{5.7}$$

Sled výpočetních operací vypadá pro $x = \frac{1}{2}$ následovně (výpočet probíhá zevnitř směrem ven):

$$\frac{1}{2} \cdot 2 + 3 = 4 \tag{5.8}$$

$$\frac{1}{2} \cdot 4 - 3 = -1 \tag{5.9}$$

$$\frac{1}{2} \cdot (-1) + 5 = \frac{9}{2} \tag{5.10}$$

$$\frac{1}{2} \cdot \frac{9}{2} - 1 = \frac{5}{4}\,. \tag{5.11}$$

Celkem je pro určení požadované hodnoty polynomu potřeba pouze čtyř operací násobení i součtu/rozdílu. Jedná se tedy o nejvíce efektivní algoritmus, který lze pro obecně vyjádřený polynom (3.1) schématicky vyjádřit algoritmem 5.1.

[h!] InputVstup OutputVýstup n, $\mathbf{c} = \{c_1, c_2, c_3 c_n, c_{n+1}\}$, x $f(x)$ $y \leftarrow c_{n+1}$ $i \leftarrow n, n-12, 1$ $y \leftarrow y \cdot x + c_i$ $f(x) \leftarrow y$ Hornerova metoda

Algoritmus 5.1 lze zapsat prostřednictvím m-souboru i v programu MATLAB:

```
function y=horner(d,c,x)
y=c(d+1);
for i=d:-1:1
  y=y*x+c(i);
end
```

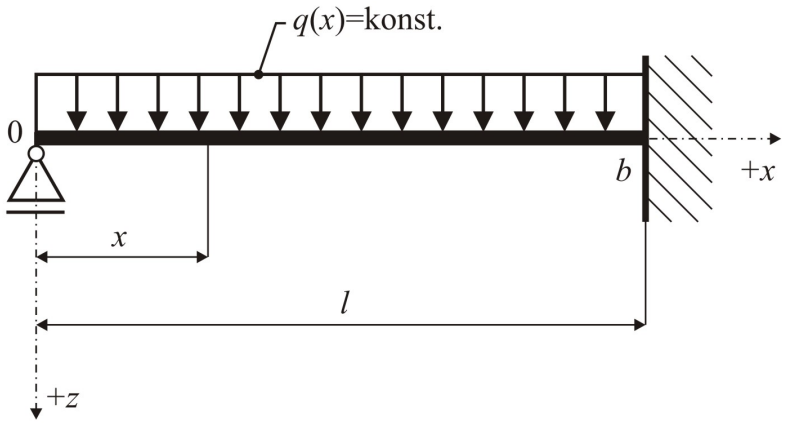
kde parametr d je stupeň zadávaného polynomu, c je vektor s $d+1$ konstantními koeficienty polynomu a x je hodnota, pro níž se polynom určuje. Funkce se pro polynom (5.1) a konkrétně zadanou hodnotu $x = \frac{1}{2}$ bude vyvolávat příkazem:

```
horner(4,[-1 5 -3 3 2],1/2)
```

Výstup z programu MATLAB pak bude:

```
ans =
    1.2500
```

S využitím předchozího postupu určete průhyb v polovině rozpětí staticky neurčitého nosníku, jehož statické schéma je zobrazeno na obr. 5.1. Pro stanovení rovnic polynomů ohybové čáry a pootočení využijte metodu přímé integrace diferenciální rovnice 4. řádu $EI_y w_z(x)'''' = q_z(x)$, kde E je modul pružnosti v tahu a tlaku [MPa], I_y příslušný moment setrvačnosti [m^4] a q_z spojité silové zatížení, působící ve svislém směru [kN/m]. Konkrétní vstupní údaje jsou uvedeny v tabulce 5.1.



Obrázek 5.1: Statické schéma řešeného staticky neurčitého nosníku

Spojité silové zatížení q_z :	4 kN/m
Rozpětí nosníku l :	6 m
Šířka obdélníkového průřezu b :	$0,02$ m
Výška obdélníkového průřezu h :	$0,15$ m
Moment setrvačnosti I_y :	$\frac{1}{12} \cdot 0,02 \cdot 0,15^3 = 5,625 \cdot 10^{-6}$ m^4
Modul pružnosti v tahu a tlaku E :	$2,1 \cdot 10^{11}$ Pa

Tabulka 5.1: Vstupní údaje příkladu 5.1

V diferenciální rovnici 4. řádu

$$EI_y w_z(x)'''' = q_z(x) \tag{5.12}$$

bude na pravé straně člen odpovídající rovnoměrnému spojitému zatížení, tedy $q(x) = q = \text{konst}$.

Vztah (5.12) pak lze postupně integrovat:

$$EI_y w_z(x)'''' = q_z \ , \tag{5.13}$$

$$EI_y w_z(x)''' = -V_z(x) = q_z \cdot x + C_1 \ , \tag{5.14}$$

$$EI_y w_z(x)'' = -M_y(x) = q_z \cdot \frac{x^2}{2} + C_1 \cdot x + C_2 \ , \tag{5.15}$$

$$EI_y w_z(x)' = q_z \cdot \frac{x^3}{6} + C_1 \cdot \frac{x^2}{2} + C_2 \cdot x + C_3 \ , \tag{5.16}$$

$$EI_y w_z(x) = q_z \cdot \frac{x^4}{24} + C_1 \cdot \frac{x^3}{6} + C_2 \cdot \frac{x^2}{2} + C_3 \cdot x + C_4 \ . \tag{5.17}$$

Integrační konstanty $C_1 C_4$ se určí z okrajových podmínek:
[alph]a)

Pro $x = 0$ je $M_y(x) = 0$ a platí tedy:

$$M_y(x = 0) = -q \cdot \frac{0^2}{2} - C_1 \cdot 0 - C_2 = 0 \ , \tag{5.18}$$

z čehož vyplývá, že $C_2 = 0$,

Pro $x = 0$ je $w_z(x) = 0$ a platí tedy:

$$w_z(x = 0) = \frac{1}{EI_y} \cdot \left(q_z \cdot \frac{0^4}{24} + C_1 \cdot \frac{0^3}{6} + 0 \cdot \frac{0^2}{2} + C_3 \cdot 0 + C_4 \right) = 0 \ , \tag{5.19}$$

z čehož plyne, že $C_4 = 0$,

Pro $x = l$ je $w_z^{'}(x) = 0$ a platí tedy:

$$w_z^{'}(x = l) = \frac{1}{EI_y} \cdot \left(q_z \cdot \frac{l^3}{6} + C_1 \cdot \frac{l^2}{2} + 0 \cdot l + C_3 \right) = 0 \ , \tag{5.20}$$

Pro $x = l$ je $w_z(x) = 0$ a platí tedy:

$$w_z(x = l) = \frac{1}{EI_y} \cdot \left(q_z \cdot \frac{l^4}{24} + C_1 \cdot \frac{l^3}{6} + 0 \cdot \frac{l^2}{2} + C_3 \cdot l + 0 \right) = 0 \ . \tag{5.21}$$

Poslední dvě rovnice představují soustavu dvou lineárních rovnic o dvou neznámých integračních konstantách C_1 a C_2. Řešením soustavy lze získat:

$$C_1 = -\frac{3}{8} q_z l \ , \tag{5.22}$$

a

$$C_3 = \frac{1}{48} q_z l^3 \ . \tag{5.23}$$

Dosazením výsledných hodnot integračních konstant $C_1 C_4$ do vztahů 5.14 až 5.17 je možno získat výsledné rovnice pro dvojici statických veličin (posouvající sílu V_z a ohybový moment M_y) i pro obě deformační veličiny - průhyb w_z a pootočení $w_z^{'} = \varphi_y$:

$$V_z(x) = -\left(q_z x - \frac{3}{8} q_z l \right) = \frac{3}{8} q_z l - q_z x \ , \tag{5.24}$$

$$M_y(x) = -\left(q_z \frac{x^2}{2} - \frac{3}{8} q_z l x + 0 \right) = \frac{3}{8} q_z l x - q_z \frac{x^2}{2} \ , \tag{5.25}$$

$$
\begin{aligned}
w_z(x)' = \varphi_y(x) &= \frac{1}{EI_y} \cdot \left(q_z \frac{x^3}{6} - \frac{3}{8} q_z l \frac{x^2}{2} + 0 \cdot x + \frac{1}{48} q_z l^3 \right) = \\
&= \frac{1}{EI_y} \cdot \left(q_z \frac{x^3}{6} - \frac{3}{16} q_z l x^2 + \frac{1}{48} q_z l^3 \right) ,
\end{aligned} \tag{5.26}
$$

$$
\begin{aligned}
w_z(x) &= \frac{1}{EI_y} \cdot \left(q_z \frac{x^4}{24} - \frac{3}{8} q_z l \frac{x^3}{6} + 0 \cdot \frac{x^2}{2} + \frac{1}{48} q_z l^3 x + 0 \right) = \\
&= \frac{1}{2EI_y} \cdot \left(q_z \frac{x^4}{12} - \frac{3}{24} q_z l x^3 + \frac{1}{24} q_z l^3 x \right) .
\end{aligned} \tag{5.27}
$$

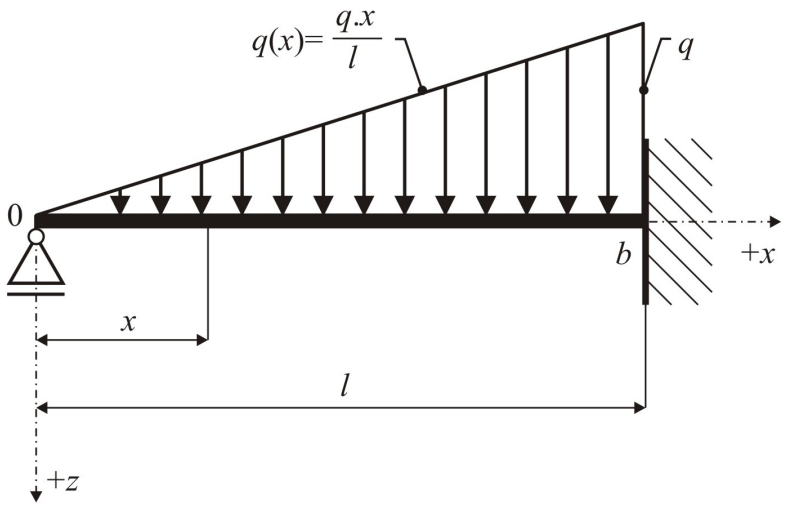
Při využití programu MATLAB i vytvořené m-funkce `horner` pro stanovení požadovaného průhybu bude sled příkazů následující:

```
qz=4000;
l=6;
b=0.02;
h=0.15;
Iy=1/12*b*h^3;
E=2.1*10^11;
c=[0 1/(48*E*Iy)*qz*l^3 0 -3/(48*E*Iy)*qz*l qz/(24*E*Iy)];
horner(4,c,l/2)*1000
```

Výsledný průhyb v milimetrech pak v polovině rozpětí vychází:

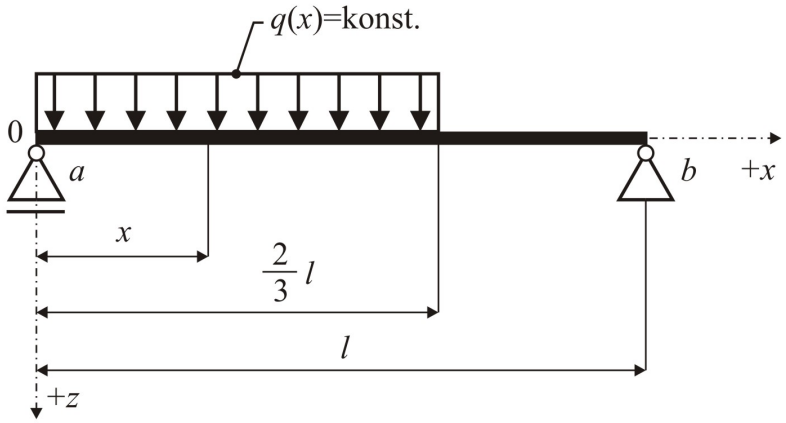
```
ans =
  22.857142857142865
```

Výše vysvětleným postupem určete průhyb v polovině rozpětí staticky neurčitého nosníku, který je schématicky zobrazen na obr. 5.2.



Obrázek 5.2: Statické schéma řešeného staticky neurčitého nosníku

Stanovte průhyb v polovině rozpětí staticky určitého nosníku, jehož schéma je zobrazeno na obr. 5.3. Pro stanovení rovnice ohybové čáry ve zkoumaném intervalu $< 0; \frac{2}{3}\, l >$ využijte Clebschovu metodu.



Obrázek 5.3: Statické schéma řešeného staticky určitého nosníku

5.1.1 Tabelace řešené funkce

Výpis hodnot funkce s parametrem x lze nejlépe provést s využitím cyklu `for` a funkcemi pro výpis na obrazovku v předepsaném formátu `disp` a `sprintf`.

Funkce `disp(x)` zobrazí obsah proměnné x typu `text`.

Funkce `sprintf` převede data do textového řetězce v požadovaném formátu pomocí tzv. „konverzních specifikátorů“, které začínají znakem %. Běžnou konverzi je možno provést specifikátorem `%f` pro převod numerické hodnoty do tvaru s pevnou desetinnou čárkou, specifikátorem `%e` pro vyjádření číselné hodnoty v exponenciálním tvaru, příp. specifikátorem `%g` pro automatickou volbu. Mezi znak `%` a specifikátor `f`, `e` nebo `g` lze navíc vložit počet znaků požadované šířky formátu, případně tečku a počet požadovaných znaků za desetinnou čárkou. Parametr `\n` zajistí přechod na nový řádek.

Proveďte tabelizaci funkce ohybové čáry nosníku z příkladu 5.1. Výsledné průhyby určete v průřezech s roztečí 10 cm.

Sled příkazů pro výpis výsledných průhybů ve sledovaných bodech x by mohl vypadat následovně:

```
format long
disp('  x [mm]    w(x) [mm]')
disp('______________________')
for x=0:.1:l
disp(sprintf('%8.1f %12.4f',x*1000,horner(4,c,x)*1000))
end
```

Výpis pak bude mít následující vzhled:

```
  x [mm]    w(x) [mm]

----------------------
     0.0       0.0000
   100.0       1.5226
   200.0       3.0377
   300.0       4.5383
   400.0       6.0176
    ...         ...
  5700.0       0.6297
  5800.0       0.2881
  5900.0       0.0741
  6000.0       0.0000
```

Výpis z předchozího příkladu doplňte i o hodnotu posouvající síly V_z, ohybového momentu M_y a pootočení φ_y.

5.1.2 Vykreslení grafu řešené funkce

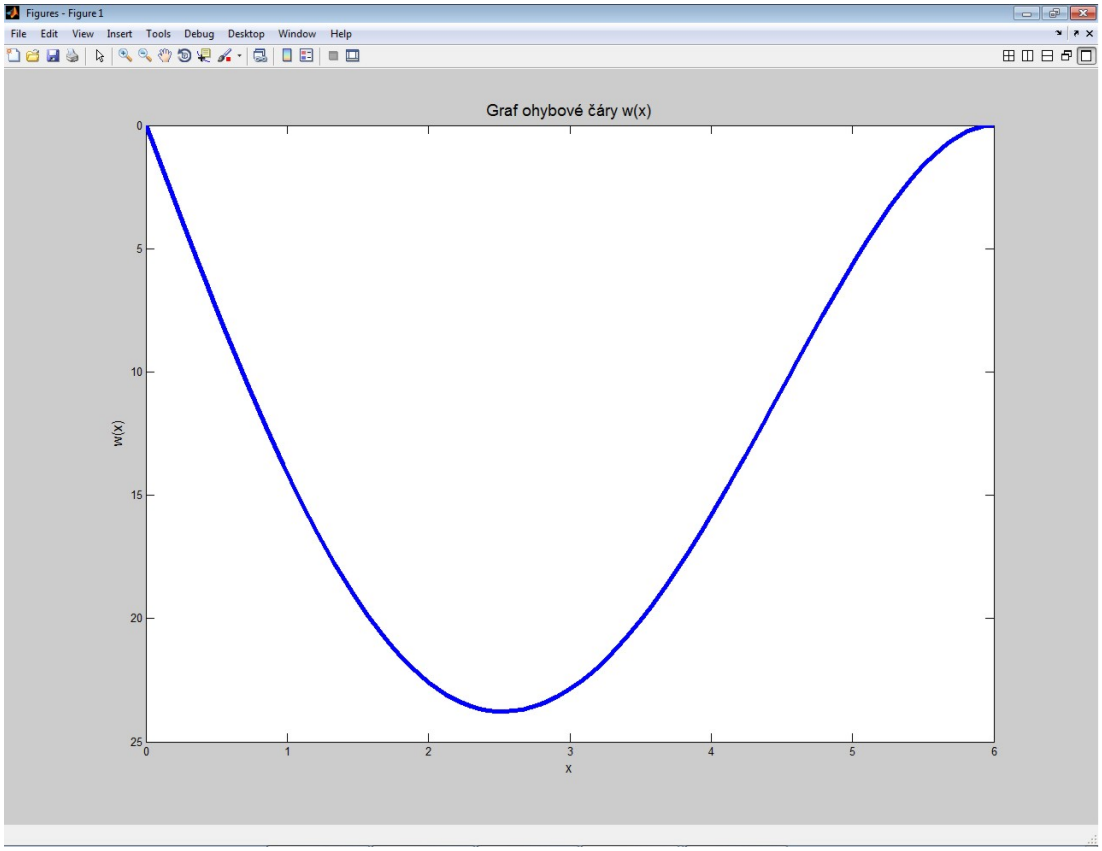
Graf řešené funkce lze vykreslit s využitím postupu, popsaném v kap. 3.4.1.

Vykreslete graf ohybové čáry nosníku z příkladu 5.1.

Sled příkazů, s využitím cyklu **for**, odvozeného vektoru c z příkladu 5.1 a m-funkce `horner` může být následující:

```
x=linspace(0,l,100);
for i=1:100, y(i)=horner(4,c,x(i))*1000; end
plot(x,y,'b-');
title('Graf ohybové čáry w(x)');
xlabel('x');
ylabel('w(x)');
```

Výsledný graf ohybové čáry je pak zobrazen na obr. 5.4.



Obrázek 5.4: Ohybová čára staticky neurčitého nosníku z příkladu 5.1

[11.]

Obdobně sestrojte graf ohybové čáry staticky neurčitém nosníku, který je schématicky zobrazen na obr. 5.2.

Vykreslete graf ohybové čáry staticky určitého nosníku, jehož schéma je zobrazeno na obr. 5.3.

5.1.3 Určení extrému diskretizované funkce

Zjednodušený výpočet největší hodnoty funkce v předem definovaném intervalu lze provést ve třech krocích nejprve diskretizací osy x, stanovením hodnot funkce pro

všechna x v požadovaném rozsahu a algoritmem 5.1.3 pro stanovení největšího čísla ve vektoru.

[h!] InputVstup OutputVýstup n, $\mathbf{b} = \{b_1, b_2 b_n\}^T$ $maximum$ $maximum \leftarrow b_1$ $i \leftarrow 2, 3n-1, n$ $b_i > maximum$ $maximum \leftarrow b_i$ Algoritmus pro stanovení největšího čísla ve vektoru

Při programování uvedeného algoritmu v systému MATLAB je možno vytvořit m-funkci `maximum` s následující posloupností příkazů:

```
function m=maximum(b)
n=length(b)
m=b(1);
if n>1
  for i=2:n
    if b(i)>m
      m=b(i);
    end
  end
end
```

Funkce `length` vrací rozměr vektoru, obsaženého ve vstupním parametru.

Určete hodnotu největšího průhybu nosníku z příkladu 5.1 s využitím tabelizovaných hodnot ohybové čáry z příkladu 5.1.1.

Nejprve je nutno vytvořit vektor $b = b_1, b_2 b_n$ s hodnotami průhybu v sledovaných průřezech ($b_i = w_z(x_i)$) o souřadnicích x_i s roztečí 10 cm (n tedy bude při rozpětí $l = 6$ m rovno hodnotě 61):

```
i=0;
for x=0:.1:l
  i=i+1;
  b(i)=horner(4,c,x)*1000;
end
```

Nyní lze vyvolat funkci pro hledání největšího čísla ve vektoru b. Po zadání příkazu `maximum(b)` bude výpis m-funkce a výsledek největšího průhybu v mm následující:

```
n =
    61


ans =
   23.7654
```

Tento způsob výpočtu největší hodnoty funkce je pouze přibližný, neboť je zatížen chybou danou diskretizací osy x. Způsob výpočtu, který povede k přesnému řešení, bude ukázán v následující kapitole.

[11.]

S využitím m-funkce `maximum` určete i největší průhyb na staticky neurčitém nosníku, který je schématicky zobrazen na obr. 5.2.

Stanovte největší průhyb na staticky určitém nosníku, jehož schéma je zobrazeno na obr. 5.3.

Kapitola 6

Soustavy lineárních rovnic

Kapitola je zaměřena na problematiku:

- algoritmů řešení soustav lineárních rovnic přímými metodami,
- iteračních metod řešení soustav lineárních rovnic,
- trojných cyklů typu `for`,
- maticového počtu.

Velké množství úloh stavební mechaniky vede k řešení soustavy lineárních rovnic. Numerické metody pro jejich řešení jsou tedy velmi častým programátorským úkolem.

Obecně může být soustava n lineárních rovnic s n proměnnými zapsána jako:

$$\begin{array}{ccccccccc} a_{1,1}\cdot x_1 & + & a_{1,2}\cdot x_2 & + & \cdots & + & a_{1,n}\cdot x_n & = & b_1 \\ a_{2,1}\cdot x_1 & + & a_{2,2}\cdot x_2 & + & \cdots & + & a_{2,n}\cdot x_n & = & b_2 \\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n,1}\cdot x_1 & + & a_{n,2}\cdot x_2 & + & \cdots & + & a_{n,n}\cdot x_n & = & b_n \end{array} , \tag{6.1}$$

kde proměnné $x_1 x_n$, obecně x_i pro $i = 1n$, jsou neznámé, $a_{i,j}$ pro $i, j = 1n$ jsou koeficienty soustavy rovnic a čísla b_i pro $i = 1n$, jsou absolutní členy soustavy (nebo také tzv. pravá strana soustavy).

K řešení kořenů soustav lineárních rovnic se využívá maticového počtu. Koeficienty soustavy lze zapsat ve tvaru matice:

$$[A] = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{bmatrix} , \tag{6.2}$$

která bývá označována jako matice soustavy.

Neznámé a pravou stranu soustavy je možno vyjádřit jako vektory:

$$\{x\} = \begin{Bmatrix} x_1 & x_2 & \cdots & x_n \end{Bmatrix}^T , \tag{6.3}$$

$$\{b\} = \begin{Bmatrix} b_1 & b_2 & \cdots & b_n \end{Bmatrix}^T . \tag{6.4}$$

Celou soustavu lineárních rovnic pak lze vyjádřit maticově:

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{bmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{Bmatrix} , \tag{6.5}$$

nebo zkráceně v maticovém tvaru:

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b} , \tag{6.6}$$

kde **A** značí matici soustavy (levých stran rovnic), **x** sloupcový vektor neznámých kořenů soustavy a **b** sloupcový vektor pravých stran rovnic.

Jednou z podmínek jednoznačného řešení soustavy lineárních rovnic je skutečnost, že matice **A** musí být **regulární**.

Regulární matice je čtvercová matice, jejíž determinant je různý od nuly. Opakem regulární matice je tzv. **singulární matice** s nulovým determinantem. Důležitou vlastností regulární matice je možnost vypočítat jednoznačně inverzní matici. Toho lze využít např při řešení soustavy lineárních rovnic.

6.1 Přímé metody řešení soustav lineárních rovnic

Metody pro řešení soustav lineárních rovnic, které vedou k přesnému řešení (pokud se neberou v úvahu chyby numerického řešení) při konečném počtu výpočetních kroků, se označují jako *metody přímé*. Jejich základním rysem je eliminace neznámých. Pro plné matice bývají tyto metody nejefektivnější, při velkém počtu rovnic však může být výpočet omezen pamětí počítače.

6.1.1 Řešení trojúhelníkové soustavy lineárních rovnic

Obecnou horní trojúhelníkovou soustavu lineárních rovnic lze zapsat ve tvaru:

$$\begin{array}{ccccccccc} a_{1,1} \cdot x_1 & + & a_{1,2} \cdot x_2 & + & \cdots & + & a_{1,n} \cdot x_n & = & b_1 \\ & & a_{2,2} \cdot x_2 & + & \cdots & + & a_{2,n} \cdot x_n & = & b_2 \\ & & & & \ddots & & \vdots & & \vdots \\ & & & & & & a_{n,n} \cdot x_n & = & b_n \end{array} , \tag{6.7}$$

nebo maticově

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ & a_{2,2} & \cdots & a_{2,n} \\ & & \ddots & \vdots \\ & & & a_{n,n} \end{bmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{Bmatrix} , \tag{6.8}$$

Řešení, označované jako zpětná substituce (zpětný chod), je možno popsat algoritmem 6.1.1.

[h!] InputVstup OutputVýstup n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$ $\mathbf{x} = \{x_1, x_2 x_n\}^T$ $i \leftarrow n, n-12, 1$ $x_i \leftarrow \dfrac{b_i - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j}{a_{i,i}}$ Algoritmus zpětné substituce

Výpočet horní trojúhelníkové soustavy lineárních rovnic lze v programu MATLAB naprogramovat například takto:

```
n=input('Zadejte pocet neznamych v soustave rovnic:\n n=');
A=zeros(n,n);
fprintf('\n Zadejte prvky matice soustavy A:');
for i=1:n
  for j=i:n
    fprintf('\n A[%d,%d]=',i,j)
    A(i,j)=input('');
  end
end
if det(A)==0
  error('Soustava rovnic je singulární! Det(A) je roven 0!')
end
fprintf('\n Zadejte prvky vektoru pravych stran b:');
for i=1:n
    fprintf('\n b[%d]=',i)
    b(i)=input('');
end
if n==1
  x(1)=b(1)/A(1,1);
else
  for i=n:-1:1
    s=0;
    if i<n
      for j=i+1:n
        s=s+A(i,j)*x(j);
      end
    end
    x(i)=(b(i)-s)/A(i,i);
  end
end
fprintf('\n')
disp('Kořeny soustavy')
disp('--------------------------------')
for i=1:n
  fprintf('x[%d]=%16.8f\n',i,x(i))
end
```

V ukázce je pro zadání vstupních údajů použito příkazu `input`, který umožňuje výpis popisu zadávané veličiny na obrazovku a přiřazení hodnoty do proměnné zadáním přímo z klávesnice.

Určete kořeny trojúhelníkové soustavy lineárních rovnic řádu 4:

$$\begin{array}{rcrcrcrcl} x_1 & + & 2\cdot x_2 & + & 3\cdot x_3 & + & 4\cdot x_4 & = & 2 \\ & & 2\cdot x_2 & + & 6\cdot x_3 & + & 12\cdot x_4 & = & 8 \\ & & & & 6\cdot x_3 & + & 24\cdot x_4 & = & 18 \\ & & & & & & 24\cdot x_4 & = & 24 \end{array} \tag{6.9}$$

Výsledný vektor neznámých kořenů má tvar $x = \begin{Bmatrix} -1 & 1 & -1 & 1 \end{Bmatrix}^T$.

Kontrolu správnosti řešení lze provést např. opětovným dosazením kořenů do jednotlivých rovnic soustavy nebo lépe odečtením levé strany soustavy od pravé, čímž je možno získat tzv. *reziduální vektor* řešení **r**, např. následujícím způsobem:

```
fprintf('\n')
disp('Reziduální vektor')
disp('---------------------------------')
for i=1:n
  r(i)=0;
  for j=i:n
    r(i)=r(i)+A(i,j)*x(j);
  end
  r(i)=r(i)-b(i);
  fprintf('r[%d]=%16.8f\n',i,r(i))
end
```

Jednotlivé prvky by se měly blížit k nule. Pro trojúhelníkovou soustavu vychází reziduální vektor:

```
r[1] = 0.00000000
r[2] = 0.00000000
r[3] = 0.00000000
r[4] = 0.00000000
```

Kontrolu přesnosti řešení lze provést rovněž pomocí *euklidovské normy reziduálního vektoru* **r**, danou výrazem $\sqrt{\sum_i |r_i|^2}$, kterou lze v programu MATLAB vyvolat např. příkazem `norm(A*x-b)`.

Při realizaci algoritmu vzniknou potíže, pokud čísla na diagonále, tj. $a_{i,i}$, budou malá. Matice soustavy **A** pak může být téměř singulární ($\det \mathbf{A} \approx 0$). Řešením je přeuspořádání soustavy lineárních rovnic tak, aby na diagonále byly největší prvky matice soustavy **A** .

[11.]

Navrhněte algoritmus pro řešení obecné trojúhelníkové soustavy lineárních rovnic, která má dolní matici soustavy **A** (nuly nad diagonálou).

6.1.2 Gaussova eliminační metoda

Gaussova eliminace je jednou z nejstarších numerických metod, při které se matice soustavy **A** převádí na horní trojúhelníkovou matici. Řádkovými úpravami s využitím tzv. *multiplikátorů* se tato matice upraví do tvaru, kdy se pod hlavní diagonálou

nachází pouze nuly. Upravená matice pak odpovídá soustavě rovnic, která je ekvivalentní s původní soustavou, a lze ji řešit podobně jako trojúhelníkovou soustavu lineárních rovnic s pomocí tzv. zpětné substituce (zpětného chodu). Celý výpočetní postup lze schématicky popsat algoritmem 6.1.2.

[h!] InputVstup OutputVýstup n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$ $\mathbf{x} = \{x_1, x_2 x_n\}^T$ $k \leftarrow 1, 2n-2, n-1$ $i \leftarrow k+1, k+2n-1, n$ $m \leftarrow -\frac{a_{i,k}}{a_{k,k}}$
$j \leftarrow k, k+1n-1, n$ $a_{i,j} \leftarrow a_{i,j} + m \cdot a_{k,j}$
$b_i \leftarrow b_i + m \cdot b_k$

$$i \leftarrow n, n-12, 1 \; x_i \leftarrow \frac{b_i - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j}{a_{i,i}}$$

Algoritmus Gaussovy eliminace

Samotný výpis m-funkce v programu MATLAB může nabývat tvaru:

```
function x=gauss(A,b)
if det(A)==0
  error('Soustava rovnic je singulární! Det(A) je roven 0!')
  return
end
n=length(A);
if n==1
  x(1)=b(1)/A(1,1);
  return
end
for k=1:n-1
  if A(k,k)==0
    error('Na diagonále je nula!')
    return
  end
  for i=k+1:n
    m=-A(i,k)/A(k,k);
    for j=k:n
      A(i,j)=A(i,j)+m*A(k,j);
    end;
    b(i)=b(i)+m*b(k);
  end
end
for i=n:-1:1
  s=0;
  if i<n
    for j=i+1:n
      s=s+A(i,j)*x(j);
    end
  end
  x(i)=(b(i)-s)/A(i,i);
end
```

S využitím vytvořené m-funkce vyřešte kořeny soustavy 4 lineárních rovnic:

$$\begin{array}{rcrcrcrcr} 2\cdot x_1 & - & x_2 & + & 3\cdot x_3 & - & x_4 & = & 7 \\ x_1 & - & x_2 & + & 4\cdot x_3 & - & 2\cdot x_4 & = & 5 \\ 3\cdot x_1 & + & 2\cdot x_2 & + & x_3 & + & 4\cdot x_4 & = & 31 \\ 4\cdot x_1 & - & 3\cdot x_2 & + & 3\cdot x_3 & - & 3\cdot x_4 & = & -5 \end{array} \tag{6.10}$$

Řešením je vektor neznámých kořenů $\{x\} = \begin{Bmatrix} 1 & 2 & 4 & 5 \end{Bmatrix}^T$. V tomto příkladě lze demonstrovat chod algoritmu Gaussovy eliminace:

```
Původní matice A              Původní vektor b
---------------------------   ---------------------------
  2.000 -1.000  3.000 -1.000    7.000
  1.000 -1.000  4.000 -2.000    5.000
  3.000  2.000  1.000  4.000   31.000
  4.000 -3.000  3.000 -3.000   -5.000

Upravená matice A             Upravený vektor b
---------------------------   ---------------------------
  2.000 -1.000  3.000 -1.000    7.000
  0.000 -0.500  2.500 -1.500    1.500
  0.000  0.000 14.000 -5.000   31.000
  0.000  0.000  0.000 -0.857   -4.286

Kořeny soustavy
-------------------
x[ 1] =    1.000
x[ 2] =    2.000
x[ 3] =    4.000
x[ 4] =    5.000

Reziduální vektor
-------------------
r[ 1] = 0.000e+000
r[ 2] = 0.000e+000
r[ 3] = -7.105e-015
r[ 4] = -1.776e-015
```

Určete kořeny soustavy 3 lineárních rovnic s maticí soustavy

$$[A] = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} \tag{6.11}$$

a vektorem pravých stran:

$$\{b\} = \begin{Bmatrix} 1 & 4 & 1 \end{Bmatrix}^T . \tag{6.12}$$

Vektor neznámých kořenů je roven $\{x\} = \begin{Bmatrix} 3 & -5 & 3 \end{Bmatrix}^T$.

Vyřešte soustavu 4 lineárních rovnic, danou maticí soustavy

$$[A] = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \tag{6.13}$$

a vektorem pravých stran:

$$\{b\} = \begin{Bmatrix} 1 & 2 & 0 & 1 \end{Bmatrix}^T . \tag{6.14}$$

Výsledný vektor neznámých kořenů je $\{x\} = \begin{Bmatrix} 0.5 & 0.75 & 0.25 & 0.5 \end{Bmatrix}^T$.

Stanovte strojový čas a přesnost řešení (normu reziduálního vektoru) náhodně vygenerované soustavy 600 lineárních rovnic.

Příklad lze vyřešit např. s využitím následujícího sledu příkazů:

```
clc;
clear;
n=600;
m=1200;
A=randn(n,m);
A=A*A';
b=randn(n,1);
tic, x=gauss(A,b); toc
norm(A*x'-b)
```

Vyřešte reakce a vnitřní síly u příhradové konstrukce z obr. 6.1 pomocí obecné styčníkové metody. Vstupní parametry jsou $b = 3$ m, $h = 1,5$ m, $F_1 = 5$ kN a $F_2 = 12$ kN. Soustavu lineárních rovnic pak vyřešte Gaussovou eliminací.

Pokud se příhradová konstrukce z obr. 6.1 interpretuje jako soustava hmotných bodů, z kinematického hlediska obsahuje $2 \cdot s$ stupňů volnosti, kde s je počet hmotných bodů, tedy styčníků. Jelikož je konstrukce tvořena 5 styčníky ($s = 5$), vychází pak $n_v = 2 \cdot s = 10$ stupňů volnosti, které jsou odebrány 3 vnějšími vazbami ($v_e = 3$) a 7 vnitřními ($v_i = 7$) (počet prutů). Vzhledem ke skutečnosti, že $n_v = v_e + v_i$, jedná se o konstrukci staticky i kinematicky určitou.

Pokud se v každém ze styčníků stanoví 2 podmínky rovnováhy, lze získat celkem 10 podmínek rovnováhy, tvořících soustavu lineárních rovnic, ze kterých lze stanovit 10 neznámých - 3 reakce ($R_{ax}, R_{az}aR_{bx}$) a 7 vnitřních sil (N_1, N_1N_7).

Jednotlivé podmínky rovnováhy nabývají tvaru:

- Styčník a:
  1. $R_x = 0 : -R_{ax} + N_1 + N_4 \cdot \cos(\alpha) = 0$
  2. $R_z = 0 : -R_{az} + N_3 + N_4 \cdot \sin(\alpha) = 0$
- Styčník b:
  3. $R_x = 0 : +R_{bx} + N_6 \cdot \cos(\alpha) = 0$
  4. $R_z = 0 : -N_3 - N_6 \cdot \sin(\alpha) = 0$
- Styčník c:
  5. $R_x = 0 : -N_1 + N_2 = 0$

Obrázek 6.1: Statické schéma řešené příhradové konstrukce

6. $R_z = 0 : +F_1 + N_5 = 0$

- Styčník d:
  7. $R_x = 0 : -N_4 \cdot \cos(\alpha) - N_6 \cdot \cos(\alpha) + N_7 \cdot \cos(\alpha) = 0$
  8. $R_z = 0 : -N_4 \cdot \sin(\alpha) - N_5 + N_6 \cdot \sin(\alpha) - N_7 \cdot \sin(\alpha) = 0$
- Styčník e:
  9. $R_x = 0 : -N_2 - N_7 \cdot \cos(\alpha) = 0$
  10. $R_z = 0 : +F_2 + N_7 \cdot \sin(\alpha) = 0$

Celou soustavu lineárních rovnic řádu 10 lze přehledně zapsat maticově:

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b} \,, \tag{6.15}$$

kde **A** značí matici levých stran rovnic, obsahující údaje geometrie konstrukce:

$$\begin{bmatrix} -1 & 0 & 0 & +1 & 0 & 0 & +\cos(\alpha) & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & +1 & +\sin(\alpha) & 0 & 0 & 0 \\ 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & +\cos(\alpha) & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -\sin(\alpha) & 0 \\ 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\cos(\alpha) & 0 & -\cos(\alpha) & +\cos(\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & -\sin(\alpha) & -1 & +\sin(\alpha) & -\sin(\alpha) \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -\cos(\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & +\sin(\alpha) \end{bmatrix} , \tag{6.16}$$

x představuje sloupcový vektor neznámých kořenů, obsahující 10 neznámých reakcí a vnitřních sil:

$$\begin{Bmatrix} R_{ax} & R_{az} & R_{bz} & N_1 & N_2 & N_3 & N_4 & N_5 & N_6 & N_7 \end{Bmatrix}^T \tag{6.17}$$

a **b** vyjadřuje sloupcový vektor pravých stran rovnic, obsahující uzlové zatížení příhradové konstrukce:

$$\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 & -F_1 & 0 & 0 & 0 & -F_2 \end{Bmatrix}^T . \tag{6.18}$$

Goniometrické funkce $\cos(\alpha)$ a $\sin(\alpha)$, obsažené v matici **A**, lze vyjádřit přímo z rozměru konstrukce:

$$\cos(\alpha) = \frac{b}{l} , \quad \sin(\alpha) = \frac{h}{l} , \tag{6.19}$$

kde l je délka prutů č.4, 6 a 7:

$$l = l_4 = l_6 = l_7 = \sqrt{b^2 + h^2} . \tag{6.20}$$

Vzhledem k podmínce řešení Gaussovy metody, že prvky na diagonále matice **A** se nesmí rovnat 0, matici **A** i vektor pravých stran **b** je nutno upravit vhodným přeuspořádáním pořadí styčníkových rovnic, a to:

- na 4.řádku bude původně 5. styčníková rovnice,
- na 5.řádku bude původně 9. styčníková rovnice,
- na 6.řádku bude původně 4. styčníková rovnice,
- na 8.řádku bude původně 6. styčníková rovnice,
- na 9.řádku bude původně 8. styčníková rovnice,

takže pozměněná matice levých stran **A** bude mít výsledný tvar:

$$\begin{bmatrix} -1 & 0 & 0 & +1 & 0 & 0 & +\cos(\alpha) & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & +1 & +\sin(\alpha) & 0 & 0 & 0 \\ 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & +\cos(\alpha) & 0 \\ 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -\cos(\alpha) \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -\sin(\alpha) & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\cos(\alpha) & 0 & -\cos(\alpha) & +\cos(\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\sin(\alpha) & -1 & +\sin(\alpha) & -\sin(\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & +\sin(\alpha) \end{bmatrix} , \tag{6.21}$$

a sloupcový vektor pravých stran rovnic **b**:

$$\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -F_1 & 0 & -F_2 \end{Bmatrix}^T . \tag{6.22}$$

Sloupcový vektor neznámých kořenů **x** zůstane nezměněn.

Takto sestavenou soustavu lineárních rovnic lze vyřešit pro konkrétně zadané vstupní veličiny s pomocí Gaussovy eliminační metody:

```
Kořeny soustavy
-------------------
x[ 1] =   29.000 kN
x[ 2] =   17.000 kN
x[ 3] =   29.000 kN
x[ 4] =   24.000 kN
x[ 5] =   24.000 kN
x[ 6] =   14.500 kN
x[ 7] =    5.590 kN
x[ 8] =   -5.000 kN
x[ 9] =  -32.423 kN
x[10] =  -26.833 kN

Norma reziduálního vektoru:
---------------------------
n =
     0
```

Vyřešte reakce a vnitřní síly u příhradové konstrukce z obr. 6.2 pomocí obecné styčníkové metody. Vstupní parametry jsou $b = 4$ m, $h = 4$ m, $F_1 = 8$ kN a $F_2 = 14$ kN. Soustavu lineárních rovnic pak vyřešte Gaussovou eliminační metodou.

Pokud se příhradová konstrukce z obr. 6.2 interpretuje podobně jako v příkladě 6.1.2, tedy jako soustava hmotných bodů, z kinematického hlediska obsahuje $2 \cdot s$ stupňů volnosti, kde s je počet hmotných bodů, tedy styčníků. Jelikož je konstrukce tvořena 7 styčníky ($s = 7$), vychází pak $n_v = 2 \cdot s = 14$ stupňů volnosti, které jsou odebrány 3 vnějšími vazbami ($v_e = 3$) a 11 vnitřními ($v_i = 11$) (počet prutů). Vzhledem ke skutečnosti, že $n_v = v_e + v_i$, jedná se o konstrukci staticky i kinematicky určitou a lze ji řešit pouze s využitím podmínek rovnováhy.

Pokud se v každém ze styčníků stanoví 2 podmínky rovnováhy, lze získat celkem 14 podmínek rovnováhy, tvořících soustavu lineárních rovnic, ze kterých lze stanovit 14 neznámých - 3 reakce ($R_{ax}, R_{az}aR_{gz}$) a 11 vnitřních sil (N_1, N_1N_{11}).

Jednotlivé podmínky rovnováhy nabývají tvaru:

- Styčník a:
  1. $R_x = 0 : -R_{ax} + N_1 \cdot \cos(\alpha) + N_2 = 0$
  2. $R_z = 0 : -R_{az} - N_1 \cdot \sin(\alpha) = 0$
- Styčník b:
  3. $R_x = 0 : +F_1 - N_1 \cdot \cos(\alpha) + N_3 \cdot \cos(\alpha) + N_4 = 0$
  4. $R_z = 0 : +N_1 \cdot \sin(\alpha) + N_3 \cdot \sin(\alpha) = 0$

Obrázek 6.2: Statické schéma řešené příhradové konstrukce

- Styčník c:
  5. $R_x = 0 : -N_2 - N_3 \cdot \cos(\alpha) + N_5 \cdot \cos(\alpha) + N_6 = 0$
  6. $R_z = 0 : +N_3 \cdot \sin(\alpha) + N_5 \cdot \sin(\alpha) = 0$
- Styčník d:
  7. $R_x = 0 : -N_4 \cdot \cos(\alpha) - N_5 \cdot \cos(\alpha) + N_7 \cdot \cos(\alpha) + N_8 = 0$
  8. $R_z = 0 : +N_5 \cdot \sin(\alpha) + N_7 \cdot \sin(\alpha) = 0$
- Styčník e:
  9. $R_x = 0 : -N_8 - N_9 \cdot \cos(\alpha) + N_{11} \cdot \cos(\alpha) + N_{12} = 0$
  10. $R_z = 0 : +N_9 \cdot \sin(\alpha) + N_{11} \cdot \sin(\alpha) = 0$
- Styčník f:
  11. $R_x = 0 : -N_{10} - N_{11} \cdot \cos(\alpha) + N_{13} \cdot \cos(\alpha) = 0$
  12. $R_z = 0 : +N_{11} \cdot \sin(\alpha) + N_{13} \cdot \sin(\alpha) = 0$

- Styčník g:

  13. $R_x = 0 : -N_{12} - N_{13} \cdot \cos(\alpha) = 0$
  14. $R_z = 0 : -N_{13} \cdot \sin(\alpha) - R_{gz} = 0$

Celou soustavu lineárních rovnic řádu 14 lze podobně jako v příkladu 6.1.2 přehledně zapsat maticově:

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b} \,, \tag{6.23}$$

kde $\mathbf{A}$ značí matici levých stran rovnic, obsahující údaje geometrie konstrukce, $\mathbf{x}$ představuje sloupcový vektor neznámých kořenů, obsahující 14 neznámých reakcí a vnitřních sil (zvoleno pořadí $\mathbf{x} = \left\{R_{ax} \quad R_{az} \quad N_1 \quad N_2 \quad \dots \quad N_{11} \quad R_{gz}\right\}^T$), a $\mathbf{b}$ vyjadřuje sloupcový vektor pravých stran rovnic, obsahující uzlové zatížení příhradové konstrukce.

Na některých řádcích matice $\mathbf{A}$ se vyskytuje nula, a proto je třeba provést vzájemné přeuspořádání pořadí styčníkových rovnic:

- na 4.řádku bude původně 5. styčníková rovnice,
- na 6.řádku bude původně 7. styčníková rovnice,
- na 8.řádku bude původně 9. styčníková rovnice,
- na 10.řádku bude původně 11. styčníková rovnice,
- na 12.řádku bude původně 13. styčníková rovnice,

Výpis výsledných hodnot kořenů soustavy pak může vypadat následovně:

```
Kořeny soustavy
-------------------
x[ 1] =    8.000 kN
x[ 2] =    2.000 kN
x[ 3] =   -2.236 kN
x[ 4] =    9.000 kN
x[ 5] =    2.236 kN
x[ 6] =  -10.000 kN
x[ 7] =   -2.236 kN
x[ 8] =   11.000 kN
x[ 9] =    2.236 kN
x[10] =  -12.000 kN
x[11] =   13.416 kN
x[12] =    6.000 kN
x[13] =  -13.416 kN
x[14] =   12.000 kN

Reziduální vektor:
-------------------
r[ 1] = 0.000e+000
r[ 2] = 0.000e+000
r[ 3] = 0.000e+000
r[ 4] = 0.000e+000
r[ 5] = 0.000e+000
```

```
r[ 6] = 0.000e+000
r[ 7] = 0.000e+000
r[ 8] = -8.882e-016
r[ 9] = -6.661e-016
r[10] = 8.882e-016
r[11] = 1.776e-015
r[12] = 0.000e+000
r[13] = -1.776e-015
r[14] = -1.776e-015

Norma reziduálního vektoru:
---------------------------
n =
  3.3894e-015
```

Matice soustavy **A** z příkladu 6.1.2 je vzhledem k vhodnému označení uzlů, a tedy i vhodném sestavení rovnic soustavy, pásová (blíže viz kap. 6.2.3). V takto vzniklé matici soustavy jsou nenulové prvky od diagonály vzdáleny maximálně o dva sloupce vlevo a o čtyři sloupce vpravo (šířka pásu je tedy 7). Pokuste se upravit algoritmus Gaussovy eliminační metody tak, aby jste zohlednili šířku pásu a snížili tak počet výpočetních operací.

6.1.3 Gauss-Jordanova metoda

Gaussovu eliminační metodu lze jednoduše modifikovat způsobem, který je označován jako Gauss-Jordanova metoda. Základní myšlenkou tohoto výpočetního postupu je úprava matice soustavy **A** na diagonální nebo dokonce jednotkovou matici. Výpočetní postup je schématicky popsán algoritmem 6.1.3 a lze jej implementovat do m-funkce programu MATLAB např. následujícím způsobem:

```
function x=gauss_jordan(A,b)
if det(A)==0
  error('Soustava rovnic je singulární! Det(A) je roven 0!')
  return
end
n=length(A);
if n==1
  x(1)=b(1)/A(1,1);
  return
end
for k=1:n
  if A(k,k)==0
    error('Na diagonále je nula!')
    return
  end
  for i=1:n
    if ~(i==k)
```

```
      m=-A(i,k)/A(k,k);
      for j=k:n
        A(i,j)=A(i,j)+m*A(k,j);
      end;
      b(i)=b(i)+m*b(k);
    end
  end
end
for i=1:n
  x(i)=b(i)/A(i,i);
end
```

[h!] InputVstup OutputVýstup n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$ $\mathbf{x} = \{x_1, x_2 x_n\}^T$ $k \leftarrow 1, 2n-2, n$ $i \leftarrow 1, 2k-1, k+1 n$ $m \leftarrow -\frac{a_{i,k}}{a_{k,k}}$
$j \leftarrow 1, 2n-1, n$ $a_{i,j} \leftarrow a_{i,j} + m \cdot a_{k,j}$
$b_i \leftarrow b_i + m \cdot b_k$
$i \leftarrow 1, 2n-1, n$ $x_i \leftarrow \frac{b_i}{a_{i,i}}$ Algoritmus Gauss-Jordanovy metody

Gauss-Jordanovu metodu je možno použít i k řešení tzv. maticových rovnic, jenž lze zkráceně vyjádřit:

$$\mathbf{A} \cdot \mathbf{X} = \mathbf{B}\,, \tag{6.24}$$

kde $\mathbf{X}$ představuje matici kořenů soustavy a $\mathbf{B}$ matici levých stran, obě s obecným rozměrem $[n, r]$. Výpočetní postup je tedy nutno upravit tak, aby pracoval s více pravými stranami, jak je např. schématicky popsánu v upraveném algoritmu 6.1.3.

[h!] InputVstup OutputVýstup n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{B} = [b_{i,j}] = [b_{1,1}b_{n,r}]$ $\mathbf{X} = [x_{i,j}] = [x_{1,1}x_{n,r}]$ $k \leftarrow 1, 2n-2, n$ $i \leftarrow 1, 2k-1, k+1n$ $m \leftarrow -\frac{a_{i,k}}{a_{k,k}}$
$j \leftarrow 1, 2n-1, n$ $a_{i,j} \leftarrow a_{i,j} + m \cdot a_{k,j}$
$j \leftarrow 1, 2r-1, r$ $b_{i,j} \leftarrow b_{i,j} + m \cdot b_{k,j}$
$j \leftarrow 1, 2r-1, r$ $i \leftarrow 1, 2n-1, n$ $x_{i,j} \leftarrow \frac{b_{i,j}}{a_{i,i}}$ Algoritmus Gauss-Jordanovy metody pro řešení tzv. maticových rovnic

Pomocí Gauss-Jordanovy metody vypočtěte maticovou rovnici:

$$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,1} & x_{3,2} & x_{3,3} \end{bmatrix}^T = \begin{bmatrix} 1 & 4 & 1 \\ 2 & 2 & 1 \\ 3 & -1 & 2 \end{bmatrix}^T . \tag{6.25}$$

Řešením je matice:

$$[X] = \begin{bmatrix} 3 & -5 & 3 \\ 2 & -2 & 1 \\ -2 & 7 & -3 \end{bmatrix}^T . \tag{6.26}$$

Řešení maticových rovnic lze využít např. k řešení **inverzní matice**, kdy je matice pravých stran tvořena jednotkovou diagonální maticí„ nebo pro výpočet příhradové konstrukce z příkladu 6.1.2 s více zatěžovacími stavy.

Pomocí Gauss-Jordanovy metody vypočtěte inverzní matici k matici:

$$\begin{bmatrix} 1 & 2 & 6 \\ 2 & 5 & 15 \\ 6 & 15 & 46 \end{bmatrix} \tag{6.27}$$

Řešením je inverzní matice:

$$[A]^{-1} = \begin{bmatrix} 5 & -2 & 0 \\ -2 & 10 & -3 \\ 0 & -3 & 1 \end{bmatrix}\,, \tag{6.28}$$

o čemž se lze přesvěčit např. výpisem:

```
A =                    B =                    A*B =
     1     2     6          5    -2     0          1     0     0
     2     5    15         -2    10    -3          0     1     0
     6    15    46          0    -3     1          0     0     1
```

[11.]

Určete reakce a vnitřní síly příhradové konstrukce z příkladu z příkladu 6.1.2 rovněž pro zatěžovací stav, tvořený dvojicí svislých uzlových zatížení $F = 10$ kN ve styčnících d a e.

6.1.4 LU rozklad

Princip metody LU rozkladu je založen na principu, kdy regulární matici $\mathbf{A}$ lze rozložit na součin dvou trojúhelníkových matic $\mathbf{L}$ a $\mathbf{U}$ tak, že platí:

$$\mathbf{A} = \mathbf{L} \cdot \mathbf{U} \,. \tag{6.29}$$

Soustava lineárních rovnic se pak řeší ve dvou výpočetních krocích, kdy v prvním se vyřeší trojúhelníková soustava lineárních rovnic:

$$\mathbf{L} \cdot \mathbf{y} = \mathbf{b} \,, \tag{6.30}$$

obdobně pak

$$\mathbf{U} \cdot \mathbf{x} = \mathbf{y} \,. \tag{6.31}$$

V dolní trojúhelníkové matici $\mathbf{L}$ se přitom volí na diagonále hodnota 1. Matice $\mathbf{U}$ je horní trojúhelníková. Výhodou metody LU rozkladu je její použití u úloh s více soustavami lineárních rovnic se stejnou maticí soustavy $\mathbf{A}$, která se rozloží pouze jednou a dále se již opakovaně řeší pouze trojúhelníkové soustavy.

Výpočetní postup řešení soustavy lineárních rovnic LU rozkladem je schématicky vyjádřen v algoritmu 6.1.4.

[h!] InputVstup OutputVýstup n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$ $\mathbf{x} = \{x_1, x_2 x_n\}^T$ $i \leftarrow 1, 2n-1, n$ $j \leftarrow 1, 2n-1, n$ $u_{i,j} \leftarrow 0$
$i \leftarrow 1, 2n-1, n$ $j \leftarrow 1, 2n-1, n$ $i = j$ $l_{i,j} \leftarrow 1$
$l_{i,j} \leftarrow 0$
$j \leftarrow 1, 2n-1, n$ $i \leftarrow 1, 2j-1, j$ $u_{i,j} \leftarrow a_{i,j} - \sum_{k=1}^{i-1} l_{i,k} \cdot u_{k,j}$

$$i \leftarrow j+1, j+2n-1, n \; l_{i,j} \leftarrow \frac{a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} \cdot u_{k,j}}{u_{j,j}}$$

$$i \leftarrow 1, 2n-1, n \; y_i \leftarrow b_i - \sum_{j=1}^{i-1} l_{i,j} \cdot y_j \quad i \leftarrow n, n-12, 1 \; x_i \leftarrow \frac{y_i - \sum_{j=i+1}^{n} u_{i,j} \cdot x_j}{u_{i,i}}$$

Algoritmus metody LU rozkladu

Výpočetní algoritmus metody LU rozkladu je možno implementovat do m-funkce programu MATLAB např. následujícím způsobem:

```
function x=lu_rozklad(A,b)
if det(A)==0
  error('Soustava rovnic je singulární! Det(A) je roven 0!')
  return
end
n=length(A);
if n==1
  x(1)=b(1)/A(1,1);
  return
end
L=eye(n,n);
U=zeros(n,n);
```

```
for j=1:n
  for i=1:j
    s=0;
      if i>1
      for k=1:i-1
        s=s+L(i,k)*U(k,j);
      end
    end
    U(i,j)=A(i,j)-s;
  end
  if U(j,j)==0
    error('Na diagonále je nula!')
    return
  end
  for i=j+1:n
    s=0;
    if j>1
      for k=1:j-1
        s=s+L(i,k)*U(k,j);
      end
    end
    L(i,j)=(A(i,j)-s)/U(j,j);
  end
end
for i=1:n
  s=0;
  if i>1
    for j=1:i-1
      s=s+L(i,j)*y(j);
    end
  end
  y(i)=(b(i)-s);
end
for i=n:-1:1
  s=0;
  if i<n
    for j=i+1:n
      s=s+U(i,j)*x(j);
    end
  end
  if U(i,i)==0
    error('Na diagonále je nula!')
    return
  end
```

```
  x(i)=(y(i)-s)/U(i,i);
end
```

Pomocí metody LU rozkladu vypočtěte kořeny soustavy lineárních rovnic z příkladu 6.1.2.

Řešení lze demonstrovat podrobným výpisem průběžných i konečných výsledků:

```
Matice soustavy A             Vektor pravých stran b
---------------------------   ---------------------------
  2.000 -1.000  3.000 -1.000    7.000
  1.000 -1.000  4.000 -2.000    5.000
  3.000  2.000  1.000  4.000   31.000
  4.000 -3.000  3.000 -3.000   -5.000


Matice L                      Matice U
---------------------------   ---------------------------
  1.000  0.000  0.000  0.000    2.000 -1.000  3.000 -1.000
  0.500  1.000  0.000  0.000    0.000 -0.500  2.500 -1.500
  1.500 -7.000  1.000  0.000    0.000  0.000 14.000 -5.000
  2.000  2.000 -0.571  1.000    0.000  0.000  0.000 -0.857

Matice L*U
---------------------------
  2.000 -1.000  3.000 -1.000
  1.000 -1.000  4.000 -2.000
  3.000  2.000  1.000  4.000
  4.000 -3.000  3.000 -3.000

Kořeny soustavy
---------------------------
x[ 1] = 1.000e+000
x[ 2] = 2.000e+000
x[ 3] = 4.000e+000
x[ 4] = 5.000e+000

Norma reziduálního vektoru:
---------------------------
n = 7.324106877635580e-015
```

6.1.5 Choleského metoda (dekompozice)

Choleského metoda (také Choleského dekompozice nebo rozklad) je modifikace metody LU rozkladu pro řešení soustav lineárních rovnic se symetrickou regulární pozitivně definitní čtvercovou maticí soustavy $\mathbf{A}$, která se upraví na součin dolní a horní trojúhelníkové matice, přičemž jedna trojúhelníková matice je transpozicí matice druhé:

$$\mathbf{A} = \mathbf{U} \cdot \mathbf{U}^T \,. \tag{6.32}$$

Dolní trojúhelníková matice $\mathbf{U}$ z tohoto rozkladu se nazývá Choleského trojúhelník matice $\mathbf{A}$.

Pozitivně definitní matice je symetrická čtvercová matice, jejíž vlastní čísla jsou větší než nula. Musí platit:

$$\mathbf{x} \cdot \mathbf{A} \cdot \mathbf{x}^T > 0 \,. \tag{6.33}$$

Výpočetní postup řešení soustavy lineárních rovnic Choleského metodou obsahuje v porovnání s metodou LU rozkladu zhruba poloviční počet výpočetních operací a je schématicky popsán algoritmem 6.1.5.

Proveďte Choleského dekompozici matice:

$$[A] = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 \\ -1 & 2 & 0 & 0 & 0 \\ -1 & 0 & 3 & 1 & 1 \\ -1 & 0 & 1 & 4 & 2 \\ -1 & 0 & 1 & 2 & 5 \end{bmatrix} \,. \tag{6.34}$$

Řešením je matice:

$$[U] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 0 & 0 \\ -1 & -1 & -1 & 1 & 0 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix} \,, \tag{6.35}$$

o čemž se lze přesvědčit:

```
A =                                    U =
     1    -1    -1    -1    -1             1     0     0     0     0
    -1     2     0     0     0            -1     1     0     0     0
    -1     0     3     1     1            -1    -1     1     0     0
    -1     0     1     4     2            -1    -1    -1     1     0
    -1     0     1     2     5            -1    -1    -1    -1     1
U*U' =
     1    -1    -1    -1    -1
    -1     2     0     0     0
    -1     0     3     1     1
    -1     0     1     4     2
    -1     0     1     2     5
```

[h!] InputVstup OutputVýstup n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$ $\mathbf{x} = \{x_1, x_2 x_n\}^T$ $i \leftarrow 1, 2n-1, n$ $j \leftarrow 1, 2n-1, n$ $i = j$ $u_{i,j} \leftarrow 1$
$u_{i,j} \leftarrow 0$
$j \leftarrow 1, 2n-1, n$ $i \leftarrow j, j+1 n-1, n$ $u_{j,j} \leftarrow \sqrt{a_{j,j} - \sum_{k=1}^{j-1} u_{j,k}^2}$

$$u_{i,j} \leftarrow \frac{a_{i,j} - \sum_{k=1}^{j-1} u_{i,k} \cdot u_{j,k}}{u_{j,j}}$$

$$i \leftarrow 1, 2n-1, n \; y_i \leftarrow \frac{b_i - \sum_{j=1}^{i-1} u_{i,j} \cdot y_j}{u_{i,i}} \quad i \leftarrow n, n-12, 1 \; x_i \leftarrow \frac{y_i - \sum_{j=1}^{i-1} u_{j,i} \cdot x_j}{u_{i,i}}$$

Algoritmus Choleského metody

M-funkce, implementující postup Choleského metody, může vypadat následovně:

```
function x=choleskeho_met(A,b)
n=length(A);
U=eye(n,n);
for j=1:n
  for i=j:n
    s=0;
    if i>1
      for k=1:j-1
        s=s+U(j,k)^2;
      end
    end
    U(j,j)=sqrt(A(j,j)-s);
    s=0;
    if i>1
      for k=1:j-1
        s=s+U(i,k)*U(j,k);
      end
    end
    U(i,j)=(A(i,j)-s)/U(j,j);
  end
end
for i=1:n
  s=0;
  if i>1
    for j=1:i-1
      s=s+U(i,j)*y(j);
    end
  end
  y(i)=(b(i)-s)/U(i,i);
end
for i=n:-1:1
  s=0;
  if i<n
    for j=i+1:n
      s=s+U(j,i)*x(j);
    end
  end
  x(i)=(y(i)-s)/U(i,i);
```

```
end
```

Vypočtěte kořeny soustavy rovnic z příkladu 6.1.2 Choleského metodou.

6.2 Iterační metody řešení soustav lineárních rovnic

Řešení soustav lineárních rovnic iteračními metodami spočívá narozdíl od přímých způsobů výpočtů v postupném přibližování se k přesnému výsledku. Existuje množství způsobů tvorby takové posloupnosti aproximací, jejíž limitou je vektor $\mathbf{x}$ přesně určených kořenů soustavy lineárních rovnic.

Soustava lineárních rovnic s reálnou maticí soustavy, vyjádřena maticově, může být vyjádřena např. vztahem (6.6). Lze předpokládat, že existuje jediné přesné řešení:

$$\mathbf{x} = \mathbf{A^{-1}} \cdot \mathbf{b} \,. \tag{6.36}$$

Rovnici (6.6) pak lze upravit do tvaru, který je vhodný k iterování:

$$\mathbf{x} = \mathbf{H} \cdot \mathbf{x} + \mathbf{g} \,, \tag{6.37}$$

kde $\mathbf{H}$ představuje *iterační matici*. Na vztahu (6.37) je založeno sestrojení posloupnosti iterací $\mathbf{x}^{(k)} = \{x_1^k, x_2^k x_n^k\}^T$ podle rekurentní formule:

$$\mathbf{x}^{(k+1)} = \mathbf{H} \cdot \mathbf{x}^{(k)} + \mathbf{g} \,, \tag{6.38}$$

pro iterační kroky $k = 0, 1, 2, \ldots$.

Kromě iterační formule je nutno definovat také volbu nulté aproximace $\mathbf{x}^{(0)} = \{x_1^0, x_2^0 x_n^0\}^T$ a způsob ukončení iteračního cyklu, který lze provést:

[alph](d)

stanovením přesného počtu iteračních cyklů, které se mají provést (např. s využitím cyklu typu `for`),

zastavovací podmínkou se zadanou tolerancí nepřesnosti $\varepsilon > 0$, např. s využitím vektorové normy:

$$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| < \varepsilon \,. \tag{6.39}$$

Iterační matice $\mathbf{H}$ i vektor $\mathbf{g}$ se může při každém iteračním kroku k měnit. Hovoří se pak o tzv. *nestacionárním iteračním procesu*, narozdíl od procesu *stacionárního*, kdy jsou matice $\mathbf{H}$ i vektor $\mathbf{g}$ nezávislými na iteračním cyklu k.

Iterační výpočet soustavy lineárních rovnic konverguje k řešení, pokud je matice soustavy $\mathbf{A}$ *diagonálně dominantní* (převažují hodnoty prvků na diagonále), což lze vyjádřit:

$$|a_{i,i}| > \sum_{j=1}^{i-1} |a_{i,j}| + \sum_{j=i+1}^{n} |a_{i,j}| \,, \quad i = 1, 2n \,. \tag{6.40}$$

Sestrojte funkci pro posouzení, zda-li je matice $\mathbf{A}$ diagonálně dominantní. Vyzkoušejte podle (6.40) např. na matici:

$$[A] = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} . \tag{6.41}$$

6.2.1 Jacobiho iterace

V případě obecné soustavy lineárních rovnic $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ jsou všechny rovnice obecně vyjádřeny:

$$a_{i,1} \cdot x_1 \quad + \quad a_{i,2} \cdot x_2 \quad + \quad \cdots \quad + \quad a_{i,n} \cdot x_n \quad = \quad b_i \ , \quad i = 1, 2n \ . \tag{6.42}$$

Pokud je $a_{i,i} \neq 0$, lze každou z rovnic upravit:

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j}{a_{i,i}} \ , \quad i = 1, 2n \ , \tag{6.43}$$

což znamená, že z i-té rovnice lze určit i-tý neznámý kořen soustavy.

Jacobiho iterační rekurentní formule pak má tvar:

$$x_i^{(k)} = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j^{(k-1)} - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j^{(k-1)}}{a_{i,i}} \ , \quad i = 1, 2n \ , \tag{6.44}$$

kde k je číslo iteračního cyklu ($k = 1, 2m$).

Výpočetní postup Jacobiho iterační metody pro m iteračních cyklů je schématicky vyjádřen algoritmem 6.2.1. Pokud se pro ukončení iteračního výpočtu použije zakončovací podmínka 6.39, algoritmus se nepatrně změní (viz algoritmus 6.2.1).

[h!] InputVstup OutputVýstup m, n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$, $\mathbf{x}^{(0)} = \{x_1^{(0)}, x_2^{(0)} x_n^{(0)}\}^T$ $\mathbf{x}^{(m)} = \{x_1^{(m)}, x_2^{(m)} x_n^{(m)}\}^T$ $k \leftarrow 1, 2m-1, m$ $i \leftarrow 1, 2n-1, n$ $x_i^{(k)} = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j^{(k-1)} - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j^{(k-1)}}{a_{i,i}}$

Algoritmus Jacobiho iterační metody

[h!] InputVstup OutputVýstup n, ε, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$, $\mathbf{x}^{(0)} = \{x_1^{(0)}, x_2^{(0)} x_n^{(0)}\}^T$ $\mathbf{x}^{(m)} = \{x_1^{(m)}, x_2^{(m)} x_n^{(m)}\}^T$ $k \leftarrow 1$

$i \leftarrow 1, 2n-1, n$ $x_i^{(1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j^{(0)} - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j^{(0)}}{a_{i,i}}$

$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| \geq \varepsilon$ $k \leftarrow k + 1$

$$i \leftarrow 1, 2n-1, n \; x_i^{(k)} = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j^{(k-1)} - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j^{(k-1)}}{a_{i,i}}$$

Algoritmus Jacobiho iterace, doplněné o zakončovací podmínku

Algoritmus Jacobiho iterace lze do programovacího jazyka programu MATLAB přepsat různými způsoby. První m-funkce obsahuje implementaci výpočetního postupu řešení systému lineárních rovnic, který pracuje Jacobiho metodou s konečným počtem cyklů, tedy s využitím cyklu typu `for`. K zápisu tohoto algoritmu se využívá možnosti maticových operací programového systému MATLAB:

```
function x=jacobi(A,b,x,m)
n=length(A);
d=diag(A);
r=A-diag(d);
for k=1:m
  x=(b-r*x)./d;
end
```

Druhá m-funkce je zapsána obecnějším způsobem. Výpočet je rovněž doplněn kontrolou, zda-li je matice soustavy **A** regulární a diagonálně dominantní:

```
function x=jacobi(A,b,x,m)
if det(A)==0
  error('Soustava rovnic je singulární! Det(A) je roven 0!')
  return
end
n=length(A);
for i=1:n
  if A(i,i)==0
    error('Na diagonále je nula!')
    return
  end
  s=0;
  for j=1:n
    if ~(i==j)
      s=s+abs(A(i,j));
    end
  end
  if abs(A(i,i))<s
    disp('Matice A není diagonálně dominantní!')
    return
  end
end
for k=1:m
  y=x;
  for i=1:n
    s1=0;
```

```
    if i>1
      for j=1:i-1
        s1=s1+A(i,j)*y(j);
      end
    end
    s2=0;
    if i<n
      for j=i+1:n
        s2=s2+A(i,j)*y(j);
      end
    end
    x(i)=(b(i)-s1-s2)/A(i,i);
  end
end
```

Třetí m-funkce řeší soustavu lineárních rovnic Jacobiho iterací cyklem typu `while`, jenž je ukončen zakončovací podmínkou 6.39:

```
function x=jacobi(A,b,x,eps)
y=x;
k=1;
for i=1:n
  s1=0;
  if i>1
    for j=1:i-1
      s1=s1+A(i,j)*y(j);
    end
  end
  s2=0;
  if i<n
    for j=i+1:n
      s2=s2+A(i,j)*y(j);
    end
  end
  if A(i,i)==0
    error('Na diagonále je nula!')
    return
  end
  x(i)=(b(i)-s1-s2)/A(i,i);
end
while ~(norm(y-x)<eps) & k<1000
  y=x;
  k=k+1;
  for i=1:n
    s1=0;
    if i>1
      for j=1:i-1
```

```
      s1=s1+A(i,j)*y(j);
    end
  end
  s2=0;
  if i<n
    for j=i+1:n
      s2=s2+A(i,j)*y(j);
    end
  end
  if A(i,i)==0
    error('Na diagonále je nula!')
    return
  end
  x(i)=(b(i)-s1-s2)/A(i,i);
 end
end
```

Všechny tři způsoby zápisu předpokládají, že musí být zadána i tzv. nultá aproximace řešených kořenů soustavy $\mathbf{x}^{(0)} = \{x_1^{(0)}, x_2^{(0)} x_n^{(0)}\}^T$.

Vyřešte kořeny soustavu lineárních rovnic Jacobiho iterací:

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 2 \\ 0 \\ 1 \end{Bmatrix} . \tag{6.45}$$

Výpočet proveďte nejprve pro konečný počet cyklů $m = 5$, 10 a 20, přičemž sledujte vzrůstající přesnost dosaženého řešení. Nakonec výpočet proveďte m-funkcí se zakončovací podmínkou 6.39 se zadanou tolerancí nepřesnosti $\varepsilon = 1 \cdot 10^{-6}$.

Správné řešení kořenů soustavy se rovná vektoru

$$\{x\} = \begin{Bmatrix} 0.5 & 0.75 & 0.25 & 0.5 \end{Bmatrix}^T . \tag{6.46}$$

Samotný průběh iteračního výpočtu může vypadat při zadání $\mathbf{x}^{(0)} = \{1, 1, 1, 1\}^T$ následujícím způsobem:

```
Průběh Jacobiho iterace
-------------------------------------------------------
 c.it.      x[1]        x[2]        x[3]        x[4]
-------------------------------------------------------
    0    1.000000    1.000000    1.000000    1.000000
    1    0.750000    1.000000    0.500000    0.750000
    2    0.625000    0.875000    0.375000    0.625000
    3    0.562500    0.812500    0.312500    0.562500
    4    0.531250    0.781250    0.281250    0.531250
    5    0.515625    0.765625    0.265625    0.515625
    6    0.507813    0.757813    0.257813    0.507813
    7    0.503906    0.753906    0.253906    0.503906
    8    0.501953    0.751953    0.251953    0.501953
    9    0.500977    0.750977    0.250977    0.500977
   10    0.500488    0.750488    0.250488    0.500488
   11    0.500244    0.750244    0.250244    0.500244
   12    0.500122    0.750122    0.250122    0.500122
   13    0.500061    0.750061    0.250061    0.500061
   14    0.500031    0.750031    0.250031    0.500031
   15    0.500015    0.750015    0.250015    0.500015
   16    0.500008    0.750008    0.250008    0.500008
   17    0.500004    0.750004    0.250004    0.500004
   18    0.500002    0.750002    0.250002    0.500002
   19    0.500001    0.750001    0.250001    0.500001
   20    0.500000    0.750000    0.250000    0.500000
```

K dosažení výsledku s tolerancí nepřesností $\varepsilon = 1 \cdot 10^{-6}$ je tedy při výpočtu Jacobiho iterací celkem 20-ti iteračních cyklů.

6.2.2 Gauss-Seidelova iterační metoda

Obecnou soustavu lineárních rovnic $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ lze opět vyjádřit vztahem (6.42), přičemž lze každou z rovnic definovat při splnění podmínky $a_{i,i} \neq 0$ jako:

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j}{a_{i,i}} , \quad i = 1, 2n . \tag{6.47}$$

Z této rovnice lze odvodit Gauss-Seidelovu iterační rekurentní formuli:

$$x_i^{(k)} = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j^{(k-1)}}{a_{i,i}} , \quad i = 1, 2n , \tag{6.48}$$

která se oproti Jacobiho iterační formule liší ve skutečnosti, že vypočtené kořeny soustavy $x_j^{(k)}$ jsou k dalšímu výpočtu použity ihned a nikoliv až v dalším iteračním cyklu. Celý výpočet pak konverguje k přesnému řešení rychleji.

Výpočet Gauss-Seidelovou iterační metodou pro konečný počet m iteračních cyklů je schématicky vyjádřen algoritmem 6.2.2. Postup lze samozřejmě opět doplnit i zakončovací podmínkou 6.39.

[h!] InputVstup OutputVýstup m, n, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$, $\mathbf{x}^{(0)} = \{x_1^{(0)}, x_2^{(0)} x_n^{(0)}\}^T$ $\mathbf{x}^{(m)} = \{x_1^{(m)}, x_2^{(m)} x_n^{(m)}\}^T$ $k \leftarrow 1, 2m-1, m$ $i \leftarrow 1, 2n-1, n$ $x_i^{(k)} = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j^{(k-1)}}{a_{i,i}}$

Algoritmus Gauss-Seidelovy iterační metody

Vyřešte soustavu lineárních rovnic z příkladu 6.2.1 Gauss-Seidelovou iterační metodou. Při výpočtu uvažujte tolerancí nepřesnosti $\varepsilon = 1 \cdot 10^{-6}$ a nultou aproximaci kořenů řešené soustavy $\mathbf{x}^{(0)} = \{1, 1, 1, 1\}^T$.

Samotný průběh iteračního výpočtu lze zobrazit např. takto:

```
Průběh Gauss-Seidelovy iterace
-------------------------------------------------------
 c.it.       x[1]         x[2]         x[3]         x[4]
-------------------------------------------------------
     0    1.000000     1.000000     1.000000     1.000000
     1    0.750000     0.937500     0.437500     0.593750
     2    0.593750     0.796875     0.296875     0.523438
     3    0.523438     0.761719     0.261719     0.505859
     4    0.505859     0.752930     0.252930     0.501465
     5    0.501465     0.750732     0.250732     0.500366
     6    0.500366     0.750183     0.250183     0.500092
     7    0.500092     0.750046     0.250046     0.500023
     8    0.500023     0.750011     0.250011     0.500006
     9    0.500006     0.750003     0.250003     0.500001
    10    0.500001     0.750001     0.250001     0.500000
    11    0.500000     0.750000     0.250000     0.500000
    12    0.500000     0.750000     0.250000     0.500000
```

K dosažení výsledku s tolerancí nepřesností $\varepsilon = 1 \cdot 10^{-6}$ je tedy při výpočtu Gauss-Seidelovou iterací celkem 12-ti iteračních cyklů.

6.2.3 Řídké a pásové matice

V numerické matematice se při řešení soustav rovnic často vyskytují matice soustavy, které jsou řídké a pásové.

Řídkou maticí se rozumí matice, která obsahuje značný počet nulových prvků.

Pásová matice pak má nenulové prvky pouze v těsné blízkosti diagonály. Šířka pásu matice pak představuje maximální počet sloupců v blízkosti diagonály matice s nenulovými prvky. Rovněž lze rozlišit i tzv. šířky polopásu p, q, kterými se rozumí maximální počet sloupců s nenulovými prvky vlevo a vpravo od diagonály.

Vyřešte kořeny soustavu lineárních rovnic s pásovou maticí soustavy:

$$\begin{bmatrix} 3 & -1 & & & \\ -1 & 3 & -1 & & \\ & & \ddots & & \\ & & -1 & 3 & -1 \\ & & & -1 & 3 \end{bmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{Bmatrix} = \begin{Bmatrix} 2 \\ 1 \\ \vdots \\ 1 \\ 2 \end{Bmatrix} \tag{6.49}$$

pro $n = 100$ Gauss-Seidelovou metodou, upravenou pro řešení pásových matic. Uvažujte tolerancí nepřesnosti hodnotou $\varepsilon = 1 \cdot 10^{-6}$ a nultou aproximaci kořenů řešené soustavy $\mathbf{x}^{(0)} = \{0, 00, 0\}^T$.

Matici soustavy $\mathbf{A}$ i vektor pravých stran lze vygenerovat příkazy programu MATLAB:

```
clc;
clear;
n=1000;
E=ones(n,1);
A=spdiags([-E 3*E -E],-1:1,n,n);
b=ones(n,1);
b(1)=2;
b(n)=2;
x=zeros(n,1);
```

Výpočet je pak možno provést m-funkcí, která implementuje výpočetní postup Gauss-Seidelovy iterační metody, upravené pro řešení pásových matic. Šířky polopásu p, q jsou v tomto případě rovny 1.

```
function x=gauss_seidel_pas(A,b,x,p,q,eps)
n=length(A);
maxkrok=1000;
y=x;
k=1;
for i=1:n
  s1=0;
  if i>1
    od=i-p;
    if od<1
      od=1;
    end
    for j=od:i-1
      s1=s1+A(i,j)*x(j);
    end
  end
  s2=0;
  if i<n
    do=i+q;
    if do>n
      do=n;
```

```
    end
    for j=i+1:do
      s2=s2+A(i,j)*y(j);
    end
  end
  if A(i,i)==0
    error('Na diagonále je nula!')
    return
  end
  x(i)=(b(i)-s1-s2)/A(i,i);
end
while ~(norm(y-x)<eps) & k<maxkrok
  y=x;
  k=k+1;
  for i=1:n
    s1=0;
    if i>1
      od=i-p;
      if od<1
        od=1;
      end
      for j=od:i-1
        s1=s1+A(i,j)*x(j);
      end
    end
    s2=0;
    if i<n
      do=i+q;
      if do>n
        do=n;
      end
      for j=i+1:do
        s2=s2+A(i,j)*y(j);
      end
    end
    if A(i,i)==0
      error('Na diagonále je nula!')
      return
    end
    x(i)=(b(i)-s1-s2)/A(i,i);
  end
end
if k==maxkrok
  disp('Výpočet nebyl ukončen ukončovací podmínkou!')
end
```

Výpočetní utilita je rovněž doplněna zakončovací podmínkou, že maximální počet

iteračních cyklů může být nejvýše 1000. Pokud bude výpočet ukončen splněním této podmínky, znamená to, že iterační výpočet nekonverguje dostatečně rychle nebo diverguje.

Proveďte výpočet soustavy lineárních rovnic s pásovou maticí soustavy z příkladu 6.2.3 Gauss-Seidelovou iterační metodou pro $n = 1000$. Uvažujte tolerancí nepřesnosti hodnotou $\varepsilon = 1 \cdot 10^{-6}$ a nultou aproximaci kořenů řešené soustavy $\mathbf{x}^{(0)} = \{0,00,0\}^T$.

Vyřešte kořeny soustavu lineárních rovnic s pásovou maticí soustavy:

$$\begin{bmatrix} 2 & 1 & & & \\ 1 & 2 & 1 & & \\ & \ddots & & & \\ & & 1 & 2 & 1 \\ & & & 1 & 2 \end{bmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{Bmatrix} = \begin{Bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{Bmatrix} \tag{6.50}$$

pro $n = 100$ Gauss-Seidelovou metodou, upravenou pro řešení pásových matic. Uvažujte tolerancí nepřesnosti hodnotou $\varepsilon = 1 \cdot 10^{-6}$ a nultou aproximaci kořenů řešené soustavy $\mathbf{x}^{(0)} = \{0,00,0\}^T$.

Správné řešení kořenů soustavy se rovná vektoru

$$\{x\} = \begin{Bmatrix} 1 & -1 & 1 & -1 & \cdots & 1 & -1 \end{Bmatrix}^T . \tag{6.51}$$

Proveďte výpočet soustavy lineárních rovnic s pásovou maticí soustavy z příkladu 6.2.3 Gauss-Seidelovou iterační metodou pro $n = 1000$. Uvažujte tolerancí nepřesnosti hodnotou $\varepsilon = 1 \cdot 10^{-6}$ a nultou aproximaci kořenů řešené soustavy $\mathbf{x}^{(0)} = \{0,00,0\}^T$.

6.2.4 Metoda sdružených gradientů

Metoda sdružených gradientů (Conjugate Gradient Method) je vhodným výpočetním postupem pro řešení soustav lineárních rovnic, u nichž je matice soustavy $\mathbf{A}$ čtvercová, symetrická a pozitivně definitní.

Princip metody sdružených gradientů spočívá ve vhodně zvolené posloupnosti vektorů $\mathbf{d}^{(k)}$, které určují směr postupu od $\mathbf{x}^{(k)}$ k další aproximaci $\mathbf{x}^{(k+1)}$. Vektory $\mathbf{d}^{(k)}$ se postupně konstruují z posloupnosti reziduálních vektorů $\mathbf{r}^{(k)}$ tak, aby pro skalární součin $(\mathbf{d}^{(k)}, \mathbf{A} \cdot \mathbf{d}^{(k-1)})$ platilo $(\mathbf{d}^{(k)}, \mathbf{A} \cdot \mathbf{d}^{(k-1)}) = 0$. Posloupnost vektoru $\mathbf{d}^{(k)}$ je pak $\mathbf{A}$-ortogonální.

Postup výpočtu je schématicky popsán algoritmem 6.2.4, ve kterém se nejdříve vytvoří počáteční aproximace $\mathbf{x}^{(0)} = 0$ a reziduální vektor $\mathbf{r}^{(0)} = \mathbf{b}$. Za první směr $\mathbf{d}^{(0)}$ se zvolí $\mathbf{r}^{(0)}$. Určí se hodnota $\alpha^{(0)}$, pro kterou funkce (kvadratická forma) $F(\mathbf{x}^{(0)} + \alpha \cdot \mathbf{r}^{(0)}) = F(\mathbf{x}) = \frac{1}{2} \cdot (\mathbf{A} \cdot \mathbf{x}, \mathbf{x}) - (\mathbf{b}, \mathbf{x})$ nabývá svého minima, takže platí:

$$\alpha^{(0)} = \frac{(\mathbf{d}^{(0)}, \mathbf{r}^{(0)})}{(\mathbf{d}^{(0)}, \mathbf{A} \cdot \mathbf{d}^{(0)})} = \frac{(\mathbf{r}^{(0)})^T \cdot \mathbf{r}^{(0)}}{(\mathbf{d}^{(0)})^T \cdot \mathbf{A} \cdot \mathbf{d}^{(0)}} . \tag{6.52}$$

Nyní se opraví aproximace řešení soustavy $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha^{(0)} \cdot \mathbf{d}^{(0)}$, vypočte se nový reziduální vektor $\mathbf{r}^{(1)} = \mathbf{r}^{(0)} - \alpha^{(0)} \cdot \mathbf{A} \cdot \mathbf{d}^{(0)}$ a určí se nový směr postupu $\mathbf{d}^{(1)}$ ve tvaru $\mathbf{d}^{(1)} = \mathbf{r}^{(1)} + \beta^{(0)} \cdot \mathbf{d}^{(0)}$ tak, aby platilo $(\mathbf{d}^{(1)}, \mathbf{A} \cdot \mathbf{d}^{(0)}) = 0$. Hodnota $\beta^{(0)}$ se přitom určí ze vztahu:

$$\beta^{(0)} = -\frac{(\mathbf{d}^{(0)}, \mathbf{A} \cdot \mathbf{r}^{(1)})}{(\mathbf{d}^{(0)}, \mathbf{A} \cdot \mathbf{d}^{(0)})} = \frac{(\mathbf{r}^{(1)})^T \cdot \mathbf{r}^{(1)}}{(\mathbf{r}^{(0)})^T \cdot \mathbf{r}^{(0)}} . \tag{6.53}$$

Dále se určí hodnota $\alpha^{(1)}$ tak, aby funkce $F(\mathbf{x}^{(1)} + \alpha \cdot \mathbf{r}^{(1)})$ nabývala minima, a celý výpočet se opakuje tak dlouho, dokud není splněna zakončovací podmínka.

[h!] InputVstup OutputVýstup m, n, ε, $\mathbf{A} = [a_{i,j}] = [a_{1,1}a_{n,n}]$, $\mathbf{b} = \{b_1, b_2 b_n\}^T$ $\mathbf{x} = \{x_1, x_2 x_n\}^T$ $\mathbf{x}^{(0)} \leftarrow 0$
$\mathbf{d}^{(0)} \leftarrow \mathbf{r}^{(0)} \leftarrow \mathbf{b}$
$k \leftarrow 1, 2m-1, m$ $\|\mathbf{r}^{(k-1)}\| < \varepsilon$ stop
$\alpha^{(k-1)} \leftarrow \dfrac{(\mathbf{r}^{(k-1)})^T \cdot \mathbf{r}^{(k-1)}}{(\mathbf{d}^{(k-1)})^T \cdot \mathbf{A} \cdot \mathbf{d}^{(k-1)}}$
$\mathbf{x}^{(k)} \leftarrow \mathbf{x}^{(k-1)} + \alpha^{(k-1)} \cdot \mathbf{d}^{(k-1)}$
$\mathbf{r}^{(k)} \leftarrow \mathbf{r}^{(k-1)} - \alpha^{(k-1)} \cdot \mathbf{A} \cdot \mathbf{d}^{(k-1)}$
$\beta^{(k-1)} \leftarrow \dfrac{(\mathbf{r}^{(k)})^T \cdot \mathbf{r}^{(k)}}{(\mathbf{r}^{(k-1)})^T \cdot \mathbf{r}^{(k-1)}}$
$\mathbf{d}^{(k)} \leftarrow \mathbf{r}^{(k)} + \beta^{(k-1)} \cdot \mathbf{d}^{(k-1)}$
Algoritmus metody sdružených gradientů

Výpočetní postup řešení soustavy rovnic metodou sdružených gradientů lze naprogramovat v programu MATLAB např. následujícím způsobem:

```
function x=sdruz_grad(A,b,eps)
n=length(A);
maxkrok=1000;
x=zeros(n,1);
d=b;
r=b;
for k=1:maxkrok
  if norm(r)<eps
    return
  end
  alfa=r'*r/(d'*A*r);
  x=x+alfa*d;
  rs=r;
  r=r-alfa*A*d;
  beta=r'*r/(rs'*rs);
  d=r+beta*d;
  if k==maxkrok
    disp('Výpočet nebyl ukončen ukončovací podmínkou!')
  end
end
```

Výpočet hodnot $\alpha^{(k-1)}$ a $\beta^{(k-1)}$ lze v m-funkci rovněž určit s využitím příkazu programu MATLAB pro skalární součin dvou vektorů s názvem `dot`, např.:

```
alfa=dot(d,r)/dot(d,A*d);
```

resp.

```
beta=-dot(d,A*r)/dot(d,A*d);
```

Proveďte výpočet soustavy lineárních rovnic s pásovou maticí soustavy z příkladu 6.2.3 metodou sdružených gradientů pro $n = 10000$ a tolerancemi nepřesnosti $\varepsilon = 1 \cdot 10^{-6}$ a $\varepsilon = 1 \cdot 10^{-12}$.

Proveďte výpočet soustavy lineárních rovnic s pásovou maticí soustavy z příkladu 6.2.3 metodou sdružených gradientů pro $n = 1000$ a tolerancemi nepřesnosti $\varepsilon = 1 \cdot 10^{-6}$ a $\varepsilon = 1 \cdot 10^{-12}$.

Vyřešte kořeny soustavu lineárních rovnic s řídkou maticí soustavy:

$$\begin{bmatrix} 3 & -1 & & & & & & & & 0.5 \\ -1 & 3 & -1 & & & & & & 0.5 & \\ & -1 & 3 & -1 & & & & 0.5 & & \\ & & & \ddots & & & & & & \\ & & & -1 & 3 & -1 & & & & \\ & & & & -1 & 3 & -1 & & & \\ & & & & & & \ddots & & & \\ & & 0.5 & & & & -1 & 3 & -1 & \\ & 0.5 & & & & & & -1 & 3 & -1 \\ 0.5 & & & & & & & & -1 & 3 \end{bmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ \\ \vdots \\ \\ \\ \vdots \\ x_{n-1} \\ x_n \end{Bmatrix} = \begin{Bmatrix} 2.5 \\ 1.5 \\ \vdots \\ 1.5 \\ 1 \\ 1 \\ 1.5 \\ \vdots \\ 1.5 \\ 2.5 \end{Bmatrix} \tag{6.54}$$

pro $n = 10000$ metodou sdružených gradientů s požadovanou tolerancí nepřesnosti $1 \cdot 10^{-6}$.

Matici soustavy **A** i vektor pravých stran lze vygenerovat příkazy programu MATLAB:

```
clc;
clear;
n=100;
E=ones(n,1);
n2=n/2;
A=spdiags([-E 3*E -E],-1:1,n,n);
C=spdiags([E/2],0,n,n);
C=fliplr(C);
A=A+C;
A(n2+1,n2)=-1;
A(n2,n2+1)=-1;
b=zeros(n,1);
b(1)=2.5;
b(n)=2.5;
b(2:n-1)=1.5;
b(n2:n2+1)=1;
```

Správné řešení kořenů soustavy se rovná vektoru

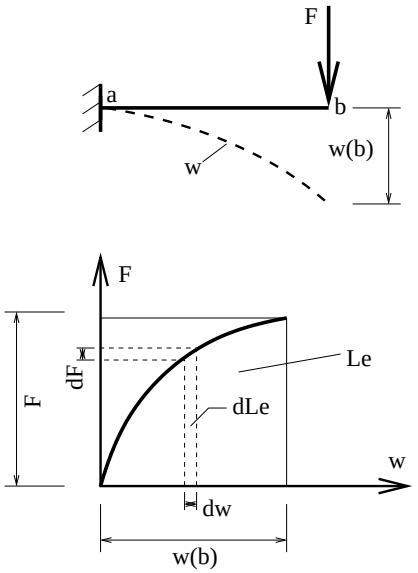
$$\{x\} = \begin{Bmatrix} 1 & 1 & \cdots & 1 & 1 \end{Bmatrix}^T . \tag{6.55}$$

Kapitola 7

Energetické principy a variační metody ve stavební mechanice

7.1 Přetvárná práce vnějších sil

V dalším textu předpokládejme působení skutečných sil na skutečných deformacích (nebudeme využívat virtuálních veličin, na které může být čtenář zvyklý např. ze silové metody).

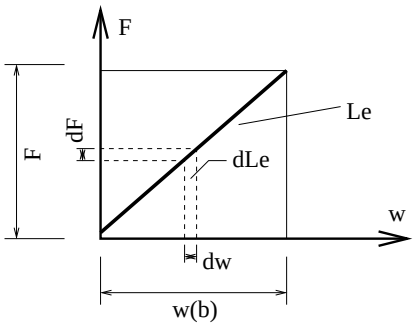


Pokud vnější síla **F** působí na dráze **w** (viz obrázek), pak je možné přetvárnou práci zapsat:

$$d\ L_e = F(w)\ d\ w, \tag{7.1}$$

$$L_e = \int_0^w F(w)\ d\ w. \tag{7.2}$$

V případě lineárně pružná odezva konstrukce bude závislost F-w lineární:



Výše zobrazenou lineární závislost popisuje Clapeyronova věta:

$$L_e = \frac{1}{2} F\ w. \tag{7.3}$$

Doplňková (komplementární) přetvárná práce vnějších sil může být zapsána:

$$d\ L_e^* = w(F)\ d\ F, \tag{7.4}$$

$$L_e^* = \int_0^F w(F)\ d\ F. \tag{7.5}$$

A pro lineárně pružnou odezvu konstrukce:

$$L_e^* = L_e = \frac{1}{2} F\ w. \tag{7.6}$$

Komplementární přetvárná práce vnějších sil L_e^* může být vykládána jako:

- práce nutná k tomu, aby působení síly F na dráze w mělo statický charakter (možno představit jako práci „brzdící“ síly působící proti F na dráze w);
- práce nutná k navrácení konstrukce do nedeformované polohy.

$$L_e + L_e^* = F\ w. \tag{7.7}$$

7.2 Potenciální energie vnějších sil

Změna potenciální energie vnějších sil Π_e odpovídá vykonané práci $L_e + L_e^*$.

Potenciální energie vnějších sil (Π_e) může být v jednorozmeěné úloze (na nosníku zatíženém osovými silami, silami kolmými k ose nosníku a momenty) vyjádřena:

$$\Pi_e = -\sum_{i=1}^{n} F_i\ u_i - \sum_{i=1}^{n} M_j\ \varphi_j - \int_c^d q(x)\ w(x)\ d\ x. \tag{7.8}$$

Pro obecný stav napjatosti tělesa pak bude mít podobu:

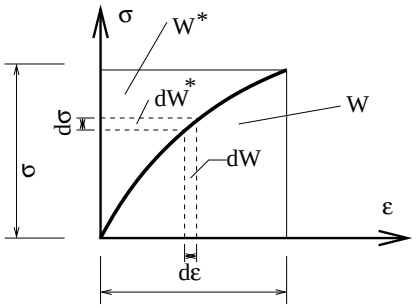
$$\Pi_e = -\int_V \mathbf{X}^T\ \mathbf{u}\ d\ V - \int_s \mathbf{p}^T\ \mathbf{u}\ d\ S. \tag{7.9}$$

7.3 Potenciální energie vnitřních sil

Dokonale pružné těleso plně akumuluje energii odpovídající vykonané přetvárné práci:

$$\Pi_i = L_e \tag{7.10}$$

Ke změně potenciální energie dochází konáním práce vnitřních sil W:



Příspěvek normálových napětí:

$$W_\sigma = \int_0^\varepsilon \sigma(\varepsilon)\ d\ \varepsilon, \tag{7.11}$$

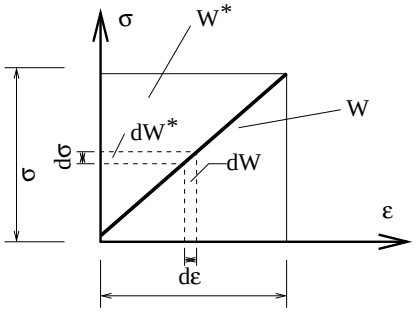
$$W_\sigma^* = \int_0^\sigma \varepsilon(\sigma)\ d\ \sigma. \tag{7.12}$$

Příspěvek smykových napětí:

$$W_\varepsilon = \int_0^\gamma \tau(\gamma)\ d\ \gamma, \tag{7.13}$$

$$W_\varepsilon^* = \int_0^\tau \gamma(\tau)\ d\ \tau. \tag{7.14}$$

Pro lineárně pružnou odezvu materiálu potom:



Příspěvek normálových napětí:

$$W_\sigma = W_\sigma^* = \frac{1}{2}\ \sigma\ \varepsilon. \tag{7.15}$$

Příspěvek smykových napětí:

$$W_\varepsilon = W_\varepsilon^* = \frac{1}{2}\ \tau\ \gamma. \tag{7.16}$$

Tedy **potenciální energie vnitřních sil** (pro lin. pružnou odezvu materiálu):

$$\begin{aligned} \Pi_i &= \\ &= \frac{1}{2} \int_V (\sigma_x \ \varepsilon_x + \sigma_y \ \varepsilon_y + \sigma_z \ \varepsilon_z + \tau_{xy} \ \gamma_{xy} + \tau_{yz} \ \gamma_{yz} + \tau_{zx} \ \gamma_{zx}) \ d \ V. \end{aligned} \tag{7.17}$$

V maticovém zápisu:

$$\Pi_i = \Pi_i^* = \frac{1}{2} \int_V \boldsymbol{\sigma}^T \ \boldsymbol{\varepsilon} \ d \ V. \tag{7.18}$$

Pro přímý prut (bez vlivu smyku) mají vztahy podobu:
Normálové síly ($\sigma = \frac{N}{A}$):

$$\Pi_{i,N} = \frac{1}{2} \int_V \frac{1}{E} \ \sigma_x^2 dV = \cdots = \frac{1}{2} \int_l \frac{N^2}{E \ A} d \ x \tag{7.19}$$

Momenty ($\sigma = \frac{M \ y}{I}$):

$$\Pi_{i,M} = \frac{1}{2} \int_V \frac{1}{E} \ \sigma_x^2 dV = \cdots = \frac{1}{2} \int_l \frac{M^2 \ y}{E \ I} d \ x \tag{7.20}$$

Tedy:

$$\Pi_i = \frac{1}{2} \int_l \frac{N^2}{E \ A} \ d \ x + \frac{1}{2} \int_l \frac{M^2 \ y}{E \ I} \ d \ x \tag{7.21}$$

7.4 Potenciální energie systému

Potenciální energie systému je součtem potenciální energie vnějších a vnitřních sil:

$$\Pi = \Pi_e + \Pi_i. \tag{7.22}$$

(Lagrangeův) **princip minima celkové potenciální energie**:

$$\Pi = \Pi_e + \Pi_i = min. \tag{7.23}$$

„Ze všech možných deformačních stavů tělesa (které neporušují jeho spojitost a respektují okrajové podmínky) nastane právě ten, při kterém je potenciální energie systému minimální.“

7.5 Variační úloha

- hledáme neznámou **funkci** (nikoli jen hodnotu),
- funkce musí splňovat určité okrajové nebo počáteční podmínky,
- hledaná funkce musí splňovat podmínku **extrému** nějaké veličiny.

7.6 Variační úlohy v teorii pružnosti

Protože platí (7.23):

$$\Pi = \Pi_i + \Pi_e = min, \tag{7.24}$$

je hodnota potenciální energie extrémní (minimální).
Z matematiky: pro extrém veličiny Π platí:

$$\partial\Pi = 0, \tag{7.25}$$

čehož využívají variační metody (např Ritzova metoda).

7.7 Ritzova metoda

7.7.1 Postup

1. Aproximace řešení volíme ve tvaru:

$$w_n(x) = \sum_{i=1}^{n} a_i \psi_i, \tag{7.26}$$

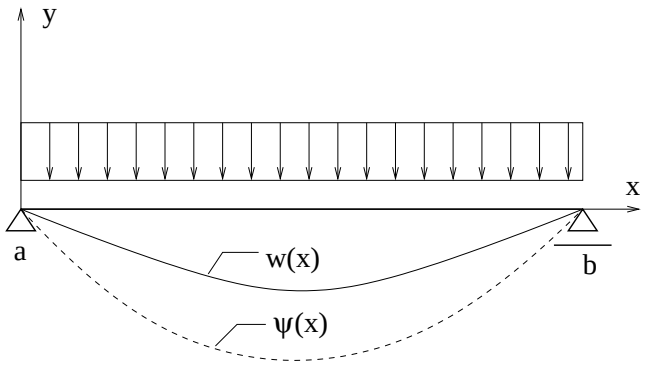
kde a_i ... neznámé konstanty, ψ_i ... aproximační funkce.
2. Vyjádříme Π pomocí $w_n(x)$.
3. Sestavení a vyřešení n rovnic:

$$\frac{\partial\Pi}{\partial a_i} = 0. \tag{7.27}$$

4. Dosazení vypočtených a_i do (7.26).

7.7.2 Bázové funkce

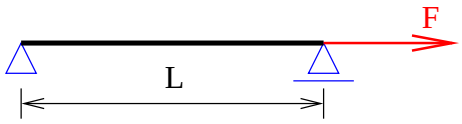
Bázové (aproximační) funkce ψ musí vyhovovat okrajovým podmínkám úlohy.



Např. při výpočtu průhybu musí platit:
$\psi(a) = 0$ (protože $w(a)$=0),
$\psi(b) = 0$ (protože $w(b)$=0).

7.8 Příklad Použití Ritzovy metody

Stanovte funkci osové deformace zadaného nosníku (viz schéma). Předpokládejte, že součin $E \times A$ je po celé délce nosníku konstantní.

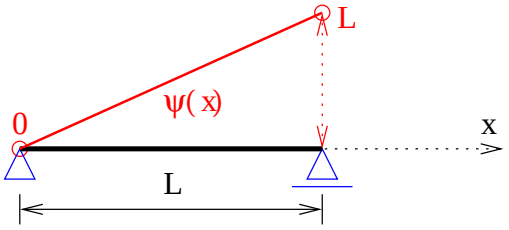


Volba aproximace:

$$u(x) = a_1\ \psi_1 = a_1\ x, \qquad tj.\ \psi_1 = x.$$

Okrajové podmínky:

$u(a) = w(x = 0) = 0 \ldots \psi_1(a) = x = 0$
$u(b) = w(x = L) \neq 0 \ldots \psi_1(b) = x = L$



Vyjádření Π_e:

$$\Pi_e = -F\ u - \int_0^L q\ u(x)dx.$$

Přitom F působí v bodě $x = L$:

$$\Pi_e = -F\ u = -F\ a_1\ \psi_1 = -F\ a_1\ x = -F\ L\ a_1.$$

Derivace funkce $u = a_1\psi_1$:

$$u' \quad = \quad [a_1\ x]' = a_1.$$

Vyjádření Π_i:

$$\Pi_i = \frac{1}{2}\int_0^L E\ A(u')^2dx = \frac{1}{2}\int_0^L E\ Aa_1^2dx = \frac{E\ A\ a1^2}{2}\ \int_0^L dx$$

$$\Pi_i = \frac{E\ A\ a_1^2}{2}\ [x]_0^L = \frac{E\ A\ L}{2}\ a_1^2.$$

Vyjádření Π:

$$\Pi = \Pi_e + \Pi_i = -F\ L\ a_1 + \frac{E\ A\ L}{2}\ a_1^2.$$

Sestavení rovnic(e) $\frac{\partial \Pi}{\partial a_i} = 0$:

$$-F\ L + E\ A\ L\ a_1 = 0$$

Výpočet a_1:

$$a_1 = \frac{F}{E\ A}$$

Výsledek (dosazením a_i do $u(x)$):

$$u(x) = a_1\ \psi_1 = \frac{F}{E\ A}\ x.$$

Protažení v $x = L$:

$$u(L) = \frac{F\ L}{E\ A}$$

.

Výpočet vnitřních sil (normálová síla):

$$N(x) = E\ A\ u' = -E\ A\ \left[\frac{F}{E\ A}\ x\right]' = F$$

Je zjevné, že výsledky (v tomto případě, kdy byly vnodně zvolena bázová funkce) odpovídají řešením známým ze základních kurzů statiky.

Kapitola 8

Metoda konečných prvků

Nevýhodou klasických variačních metod je obtížná volba (často nemožná) aproximačních funkcí φ na složitějších oblastech.

Řešení – rozdělení konstrukce na malé oblasti na n jednoduchých podoblastí a volba aproximačních funkcí na nich φ_j na nich.

Protože Π je skalární veličina, lze:

$$\Pi_{approx.} = \sum_{j=1}^{n} \Pi_{e,j}, \tag{8.1}$$

kde $\Pi_{e,j}$ je potenciální energie j-té podoblasti („konečného prvku“).

Další postup je analogický klasickým variačním metodám (např. Ritzově metodě) [1] – řeší se soustava n lineárních rovnic:

$$\frac{\partial \Pi}{\partial a_i} = 0, \qquad i = 1..n \tag{8.2}$$

8.1 Varianty MKP

- deformační (Lagrangeův variační princip) – neznámá jsou posunutí a pootočení (nejčastější, přes 90% případů),
- silová (např. Castiglianův variační princip) – neznámé jsou silové veličiny,
- smíšená (např. variační princip Hu-Washitsu).

8.2 Deformační varianta MKP

Hledané **deformační veličiny** – viz klasická teorie pružnosti (mohou být i jejich derivace!):

- rovinná napjatost s deformace (stěny, ...): u, v
- desky: w, φ_x, φ_y

[1] Pozn.: zde je použit **Lagrangeův variační princip** a jde tedy o **deformační variantu** metody konečných prvků MKP (viz dále).

- prostorové úlohy: u, v, w

Aproximační funkce se volí **zásadně** ve tvaru polynomů.

8.3 Volba náhradních polynomů

- Nejlepší konvergence při použití úplného polynomu n-tého stupně (Ženíšek et al).
- Počet konstant v polynomu $(a_1, a_2, ...)$ = počet neznámých na konečném prvku $(u_1, v_1, ...)$.
- Ne vždy je možné použít všechny členy úplného polynomu, jak bude ukázáno později.

Pro neznámou x:

1. $a_1 + a_2\ x$
2. $a_1 + a_2\ x + a_3\ x^2$
3. $a_1 + a_2\ x + a_3\ x^2 + a_4\ x^3$
4. $a_1 + a_2\ x + a_3\ x^2 + a_4\ x^3 + a_5\ x^4$
5. ...

Pro neznámé x a y:

1. $a_1 + a_2\ x + a_3\ y$
2. $a_1 + a_2\ x + a_3\ y + a_4\ xy + a_5\ x^2 + a_6\ x^2$
3. ...

8.4 Sestavení matice tuhosti konečného prvku

Potenciální energie soustavy:

$$\Pi = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V - \int_V \mathbf{X}^T\ \mathbf{r}\ d\ V. \tag{8.3}$$

Po dosazení za $\boldsymbol{\varepsilon} = \mathbf{BSr}$ a vytknutí vektoru neznámých konstant (posunutí) $\mathbf{r}$:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T \int_V \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{DB}\ \mathbf{S}^{-1}\ d\ V\ \mathbf{r}^T - \int_V \mathbf{X}^T\ d\ V\ \mathbf{r}. \tag{8.4}$$

Stručně:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T\ \mathbf{K}\ \mathbf{r}\ -\ \mathbf{F}^T\ \mathbf{r}. \tag{8.5}$$

Aplikací Lagrangeova variačního principu ($\partial\ \Pi = min.$) na (12.13):

$$\mathbf{K}\ \mathbf{r}\ =\ \mathbf{F}, \tag{8.6}$$

kde $\mathbf{K}$ ... matice tuhosti konečného prvku:

$$\mathbf{K} = \int_V\ \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}\ d\ V, \tag{8.7}$$

$\mathbf{F}$ ... zatěžovací vektor konečného prvku:

$$\mathbf{F} = -\int_V \mathbf{X}^T\ d\ V\ \ -\int_S \mathbf{p}^T\ d\ S. \tag{8.8}$$

8.5 Analýza konstrukce

Z $\mathbf{K}_e$ a $\mathbf{r}_e$ a $\mathbf{F}_e$ jednotlivých prvků (e je číslo prvku) sestavíme $\mathbf{K}$ a $\mathbf{r}$ a $\mathbf{F}$ celé konstrukce a neznámé určíme řešením soustavy rovnic:

$$\mathbf{K}\ \mathbf{r} = \mathbf{F}. \tag{8.9}$$

Poznámka: tyto sestavení matice tuhosti a zatěžovacího vektoru je zcela shodné s postupem v obecné deformační metodě.

8.6 Zatížení konstrukce

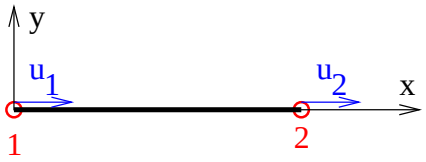
- Zatížení zavádíme výhradně v uzlech konečných prvků
- Zatížení má silový charakter:
  - **síly** pracují na posunutích
  - **momenty** pracují na pootočeních
- Zatížení deformacemi bude popsáno dále

8.7 Podepření konstrukce – okrajové podmínky

- Pružné podpory: přidáme odpovídající tuhost pružiny na diagonálu matice tuhosti
- Pevná podpora (posunutí, pootočení) $\Rightarrow$ známá hodnota $(0, 0)$ neznámého posunutí/pootočení (upravíme systém rovnic)
- Popuštění podpor: $\Rightarrow$ známá hodnota neznámého posunutí/pootočení (upravíme systém rovnic)

8.8 Odvození konečného prvku příhradoviny

Budeme předpokládat tytéž vlastnosti, jako měl prut příhradoviny v základních předmětech (klouby na koncích prutu, nenulová je jen normálová síla).



Neznámé parametry deformace: u, v v každém uzlu.
Tj. celkem má tento prvek dva neznámé uzlové parametry:

$$\{u_1, u_2\}^T.$$

Geometrická rovnice:

$$\varepsilon_x = \frac{\partial u}{\partial x} \tag{8.10}$$

Maticově ($\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T\ \mathbf{u}$):

$$\left\{\ \varepsilon_x\ \right\} = \left[\ \tfrac{\partial}{\partial x}\ \right]\ \left\{\ u\ \right\} \tag{8.11}$$

Fyzikální rovnice:

$$\sigma_x \quad = \quad E \times \varepsilon_x \tag{8.12}$$

Maticově ($\boldsymbol{\sigma} = \mathbf{D}\ \boldsymbol{\varepsilon}$):

$$\left\{\ \sigma_x\ \right\} = \left[\ E\ \right]\ \left\{\ \varepsilon_x\ \right\} \tag{8.13}$$

Aproximace neznámých uzlových posunutí:

$$u(x) \quad = \quad a_1 + a_2\ x \tag{8.14}$$

Maticově ($\mathbf{u} = \mathbf{U}\ \mathbf{a}$):

$$\left\{\ u\ \right\} = \left[\ 1 \quad x\ \right]\ \left\{\begin{array}{c} a_1 \\ a_2 \end{array}\right\} \tag{8.15}$$

Aproximace neznámých uzlových posunutí v uzlech 1, 2 ($\mathbf{r} = \mathbf{S}\ \mathbf{a}$):

$$\left\{\begin{array}{c} u_1 \\ u_2 \end{array}\right\} = \left[\begin{array}{cc} 1 & x_1 \\ 1 & x_2 \end{array}\right]\ \left\{\begin{array}{c} a_1 \\ a_2 \end{array}\right\} \tag{8.16}$$

Kombinací vztahů $\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T\ \mathbf{u}$ a $\mathbf{u} = \mathbf{U}\ \mathbf{a}$ vznikne $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{a}$, kde $\mathbf{B} = \boldsymbol{\partial}^T\ \mathbf{U}\ \mathbf{a}$:

$$\left\{\ \varepsilon_x\ \right\} = \left[\ \tfrac{\partial}{\partial x}\ \right]\ \left[\ 1 \quad x\ \right]\ \left\{\begin{array}{c} a_1 \\ a_2 \end{array}\right\} \tag{8.17}$$

Kombinací vztahů $\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T\ \mathbf{u}$ a $\mathbf{u} = \mathbf{U}\ \mathbf{a}$ vznikne $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{a}$, kde $\mathbf{B} = \boldsymbol{\partial}^T\ \mathbf{U}\ \mathbf{a}$:

$$\left\{\ \varepsilon_x\ \right\} = \left[\ 0 \quad 1\ \right]\ \left\{\begin{array}{c} a_1 \\ a_2 \end{array}\right\} \tag{8.18}$$

Z $\mathbf{r} = \mathbf{S}\ \mathbf{a}$ plyne:

$$\mathbf{a} = \mathbf{S}^{-1}\ \mathbf{r}, \tag{8.19}$$

kde:

$$\mathbf{S} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix} \Rightarrow \mathbf{S^{-1}} = \begin{bmatrix} \frac{x_2}{x_2-x_1} & \frac{-x_1}{x_2-x_1} \\ \frac{-1}{x_2-x_1} & \frac{-1}{x_2-x_1} \end{bmatrix} \tag{8.20}$$

Pak místo $\boldsymbol{\varepsilon} = \mathbf{Ba}$ lze psát $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{S}^{-1}\ \mathbf{r}$:

$$\left\{\ \varepsilon_x\ \right\} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{x_2}{x_2-x_1} & \frac{-x_1}{x_2-x_1} \\ \frac{-1}{x_2-x_1} & \frac{-1}{x_2-x_1} \end{bmatrix} \left\{ \begin{array}{c} u_1 \\ u_2 \end{array} \right\}. \tag{8.21}$$

Potenciální energie vnitřních sil:

$$\Pi_i = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T \boldsymbol{\sigma}\ d\ V = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V = \tag{8.22}$$

Potenciální energie vnějších sil:

$$\Pi_e = -\int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{8.23}$$

Potenciální energie soustavy:

$$\Pi = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V - \int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{8.24}$$

Po dosazení za $\boldsymbol{\varepsilon}$ a vytknutí $\mathbf{r}$:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T \int_V \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{DB}\ \mathbf{S}^{-1}\ d\ V\ \mathbf{r}^T - \int_V \mathbf{X}^T\ d\ V\ \mathbf{r} - \int_S \mathbf{p}^T\ d\ S\ \mathbf{r}. \tag{8.25}$$

Stručně:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T\ \mathbf{K}\ \mathbf{r}\ -\ \mathbf{F}^T\ \mathbf{r}. \tag{8.26}$$

Aplikací Lagrangeova variačního principu ($\partial\ \Pi = min.$) na (12.13):

$$\mathbf{K}\ \mathbf{r}\ =\ \mathbf{F}, \tag{8.27}$$

kde $\mathbf{K}$ ... matice tuhosti konečného prvku:

$$\mathbf{K} = \int_V\ \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}\ d\ V, \tag{8.28}$$

$\mathbf{F}$ ... zatěžovací vektor konečného prvku:

$$\mathbf{F} = -\int_V \mathbf{X}^T\ d\ V\ -\int_S \mathbf{p}^T\ d\ S. \tag{8.29}$$

Pro studovaný konečný prvek:

$$\mathbf{F} = \mathbf{X} + \mathbf{p}. \tag{8.30}$$

$$\mathbf{K} = \int_V \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1} dV = A \int_0^L, \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1} dx, \tag{8.31}$$

podrobný zápis:

$$\mathbf{K} = A\int_0^L \begin{bmatrix} \frac{x_2}{x_2-x_1} & \frac{-1}{x_2-x_1} \\ \frac{-x_1}{x_2-x_1} & \frac{-1}{x_2-x_1} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} E \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{x_2}{x_2-x_1} & \frac{-x_1}{x_2-x_1} \\ \frac{-1}{x_2-x_1} & \frac{-1}{x_2-x_1} \end{bmatrix} dx \quad (8.32)$$

Podrobný zápis (vytknutí konstant před integrál):

$$\mathbf{K} = A \begin{bmatrix} \frac{x_2}{x_2-x_1} & \frac{-1}{x_2-x_1} \\ \frac{-x_1}{x_2-x_1} & \frac{-1}{x_2-x_1} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} E \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{x_2}{x_2-x_1} & \frac{-x_1}{x_2-x_1} \\ \frac{-1}{x_2-x_1} & \frac{-1}{x_2-x_1} \end{bmatrix} \int_0^L dx \quad (8.33)$$

Po úpravě (integrace $\int_0^L dx = L$ násobení matic):

$$\mathbf{K} = EAL \begin{bmatrix} \frac{1}{(x_2-x_1)^2} & \frac{-1}{(x_2-x_1)^2} \\ \frac{-1}{(x_2-x_1)^2} & \frac{1}{(x_2-x_1)^2} \end{bmatrix}, \quad x_2 - x_1 = L \Rightarrow \mathbf{K} = \begin{bmatrix} \frac{EA}{L} & \frac{-EA}{L} \\ \frac{-EA}{L} & \frac{EA}{L} \end{bmatrix}, \quad (8.34)$$

což je matice tuhosti známá i z deformační metody.

Soustava rovnic pro jeden konečný prvek má tedy tvar:

$$\mathbf{K_e r_e} = \mathbf{F_e},$$

podrobně:

$$\begin{bmatrix} \frac{EA}{L} & \frac{-EA}{L} \\ \frac{-EA}{L} & \frac{EA}{L} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} \quad (8.35)$$

Rozšíření na proměnné u a v v každém uzlu:

$$\begin{bmatrix} \frac{EA}{L} & 0 & \frac{-EA}{L} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{-EA}{L} & 0 & \frac{EA}{L} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \end{bmatrix} = \begin{bmatrix} F_1 \\ 0 \\ F_2 \\ 0 \end{bmatrix} \quad (8.36)$$

Získaný tvar přesně odpovídá matici tuhosti příhradového prvku, která je známa z deformační metody.

8.9 Úkoly k procvičení:

- Navrhněte polynom pro aproximaci průhybu trojuzlové desky (tj. s 9 konstantami a_i).
- Na základě znalosti teorie pružnosti určete, jaký bude stupeň polynomu aproximujícího ohybové momenty.

Kapitola 9

Ilustrace programu

Dále bude nastíněna možná podoba program pro řešení příhradových konstrukcí ssetaveného v algoritmockém jazyce [1].

9.0.1 Zadání: typ úlohy, materiál

Počet stupňů volnosti (*ndof*) a počet uzlů na prvku (*puzlu*):

```
ndof=2
puzlu=2
```

Modul pružnosti (E) a plocha prvku(A):

```
E=20e9
A=0.01
```

9.0.2 Zadání: geometrie

Souřadnice uzlů (x,y, co řádek to prvek):

```
uzly=[ 0 2 ;
       3 2 ;
       3 0 ]
```

Konečné prvky (čísla uzlů na prvku – pořadová čísla řádků v matici *uzly*):

```
pruty=[ 1 2 ;
        3 2 ]
```

9.0.3 Zadání: podpory a zatížení

```
podpory=[
1 1 0 ;
1 2 0  ;
3 1 0 ;
```

[1] Viz `http://www.octave.org` nebo `http://www.matlab.com`.

```
3 2 0 ]

sily=[
2 1 -2000 ;
2 2 -3000 ]
```

Pomůcka: číslo uzlu, směr (1=u, 2=v), velikost.

9.0.4 Výpočet velikosti polí

Zadané veličiny:

```
nuzlu=size(uzly,1)
nprutu=size(pruty,1)
npodpor=size(podpory,1)
nsil=size(sily,1)
```

Velikost úlohy:

```
velikost = nuzlu*ndof
```

9.0.5 Kódová čísla

Prut 1 (uzly a=1,b=2):

$$[u1, v1, u2, v2] = [1, 2, 3, 4]$$

Prut 2 (uzly c=3,b=2):

$$[u3, v3, u2, v2] = [5, 6, 3, 4]$$

Sestavíme kódová čísla:

```
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
```

9.0.6 Výpočet: nulování K, U, F

```
K = zeros(velikost);
u = zeros(velikost,1);
F = zeros(velikost,1);
```

9.0.7 Výpočet: sestavení K

```
for i=1:nprutu
  % TADY BUDE sestaveni lokalni matice tuhosti:

  % TADY BUDE transformace do glob. souradnic:

  % TADY BUDE samotna lokalizace:
end
```

9.0.8 Výpočet: sestavení lokální Ke

```
  dx = (uzly(pruty(i,1),1)-uzly(pruty(i,2),1));
  dy = (uzly(pruty(i,1),2)-uzly(pruty(i,2),2));
  L = sqrt(dx^2 + dy^2);

  S=[1 0 ; 1 L];
  B=[0 1];
  D=[E];
  Si=inv(S);
  Kel=A*L*Si'*B'*D*B*Si;
  Keg=[Kel(1,1) 0 Kel(1,2) 0 ; 0 0 0 0 ;
      Kel(2,1) 0 Kel(2,2) 0 0 0 0 0 ];
```

9.0.9 Výpočet: transformace Ke do globálních souřadnic

```
% Transformace:
s = dy/L;
c = dx/L;

T = [ c s 0 0 ; -s c 0 0 ;
      0 0 c s ; 0 0 -s c ] ;

 Ke = T' * Keg * T;
```

9.0.10 Výpočet: lokalizace do K

```
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
```

9.0.11 Výpočet: 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 ;
end
```

Uvedený algoritmus funguje jen pro pevné podpory (s nulovým posunutím). Pro předepsaná posunutí v řádku i (popuštění podpor by bylo ještě třeba):

1. doplnit do vektoru u příslušnou hodnotu posunutí u_i,
2. hodnoty v i-tém sloupci matice $\mathbf{K}$ vynásobit hodnoutou u_i a odečíst od vektoru $\mathbf{F}$,
3. na diagonálu matice $\mathbf{K}$ v i-tém řádku dosadit 1 ,
4. do vektoru $\mathbf{F}$ v i-tém řádku dosadit u_i .

9.0.12 Výpočet: zatížení

Předpokládáme jen síly v uzlech:

```
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
```

9.0.13 Výpočet: soustava rovnic

Vyřešíme soustavu rovnic (je-li vše dobře, tak jsou lineární):

```
u=K\F
```

Vektor $\mathbf{u}$ obsahuje hledaná posunutí.

9.0.14 Výsledky: výpočet na jednotlivých prvcích

```
for i=1:nprutu
 ue  = zeros(puzlu*ndof,1);
 uel = zeros(puzlu*ndof,1);

 % TADY BUDE ziskani lokalnich vektoru deformaci:
 % TADY BUDE transformace:
 % TADY BUDE matice tuhosti (jen lokalni):
  % TADY BUDE vysledeK - sily v prutech:
end
```

9.0.15 Výsledky: lokální vektory deformací

```
 % Ziskani lokalnich vektoru deformaci:

 for j=1:(puzlu*ndof)
  ue(j) = u(kcis(i,j));
 end
```

9.0.16 Výsledky: transformace lokálních vektorů deformací

Vektory převedeme do lokálního systému souřadnic:

```
 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 = sqrt(dy2)/L;
 c = sqrt(dx2)/L;
  T = [ c s 0 0 ; -s c 0 0 ;
        0 0 c s ; 0 0 -s c ] ;
  uel = T * ue;
```

9.0.17 Výsledky: lokální matice tuhosti

```
  S=[1 0 ; 1 L];
  B=[0 1];
  D=[E];
  Si=inv(S);
  Kel=A*L*Si'*B'*D*B*Si;
  Keg=[Kel(1,1) 0 Kel(1,2) 0 ;
      0 0 0 0 ;
      Kel(2,1) 0 Kel(2,2) 0
      0 0 0 0 ];
```

9.0.18 Výsledky: koncové síly na prutech

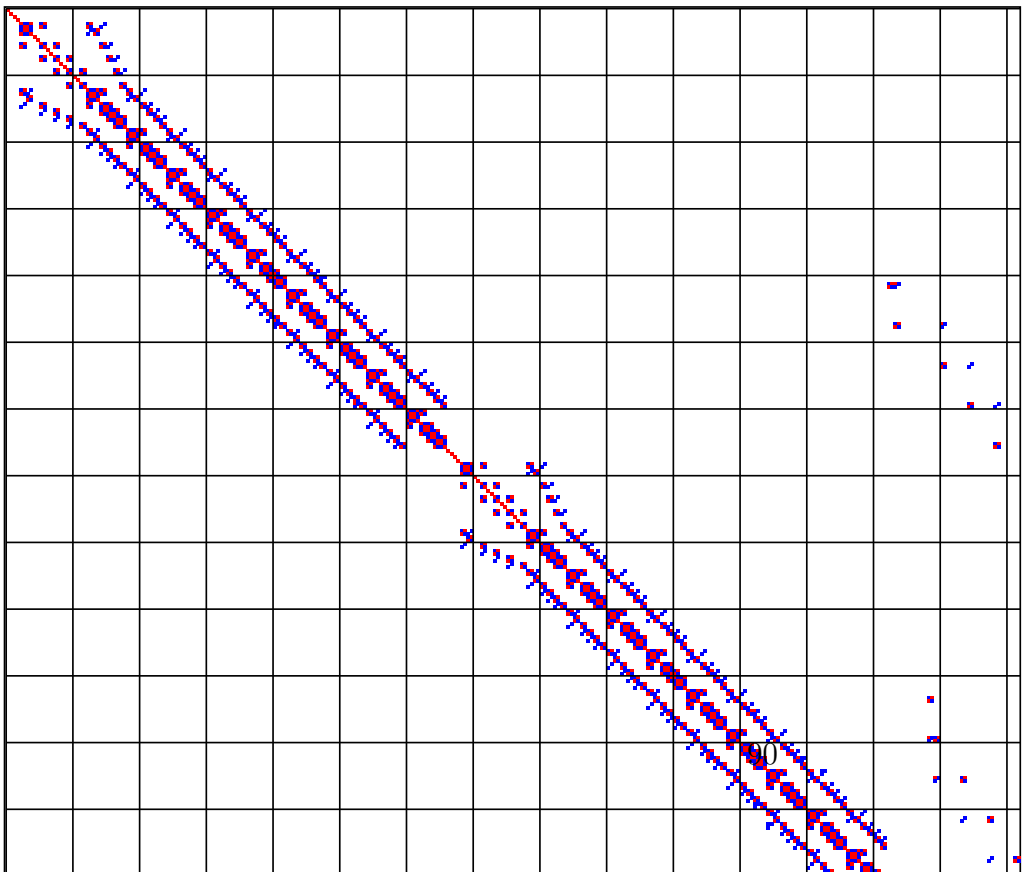
```
Fe = Keg * uel
```

9.1 Doplnění: matice tuhosti K

Vlastnosti matice tuhosti konstrukce (v úlohách lineární pružnosti):

- **pozitivně definitní** (jen pokud je konstrukce řádně podepřená!),
- **symetrická podle hlavní diagonály** (vyplývá např. z Bettiho věty),
- **řídká**, zpravidla pásová (obsahuje velmi málo nenulových členů).

Uvedených vlastností využívají efektivní řešiče soustav rovnic. Příklad rozložení nenulových prvků v matici tuhosti je uveden na následujícím obrázku:

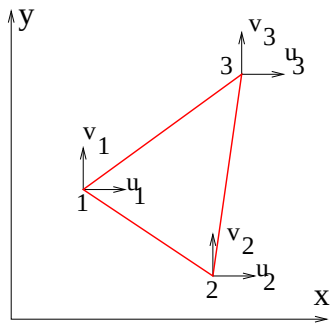


9.2 Úkoly k procvičení

- Ve vhodném software, např. GNU Octave `http://www.octave.org` nebo MATLAB `http://www.matlab.com` sestavte výpočetní program pro řešení příhradových konstrukcí.
- Ověřte, zda získané výsledky odpovídají výsledkům vypočítaným na základě zásad statiky.

Kapitola 10

Odvození konečného prvku pro rovinný problém



Neznámé parametry deformace tedy budou: u, v v každém uzlu.

Tj. celkem šest neznámých uzlových parametrů na konečném prvku:

$$\{u_1, v_1, u_2, v_2, u_3, v_3\}^T.$$

Geometrické rovnice:

$$\varepsilon_x = \frac{\partial u}{\partial x}, \quad \varepsilon_y = \frac{\partial v}{\partial y}, \quad \tau_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}. \tag{10.1}$$

Maticově ($\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T\ \mathbf{u}$):

$$\left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{array} \right\} = \left[\begin{array}{cc} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{array} \right] \left\{ \begin{array}{c} u \\ v \end{array} \right\} \tag{10.2}$$

Fyzikální rovnice (rovinná napjatost):

$$\sigma_x = \frac{E}{1-\mu^2}\,(\varepsilon_x + \mu\ \varepsilon_y) \tag{10.3}$$

$$\sigma_y = \frac{E}{1-\mu^2}\,(\varepsilon_y + \mu\ \varepsilon_x) \tag{10.4}$$

$$\tau_x = \frac{E}{2(1-\mu)}\ \gamma_{xy} \tag{10.5}$$

Maticově ($\boldsymbol{\sigma} = \mathbf{D}\ \boldsymbol{\varepsilon}$):

$$\left\{ \begin{array}{c} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{array} \right\} = \frac{E}{1-\mu^2} \left[\begin{array}{ccc} 1 & \mu & 0 \\ \mu & 1 & 0 \\ 0 & 0 & \frac{1}{2}(1-\mu) \end{array} \right] \left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{array} \right\} \tag{10.6}$$

Fyzikální rovnice (rovinná deformace):

$$\begin{aligned} \sigma_x &= \frac{E}{(1+\mu)(1-2\mu)}\ [(1-\mu)\ \varepsilon_x + \mu\ \varepsilon_y] \\ \sigma_y &= \frac{E}{(1+\mu)(1-2\mu)}\ [\mu\ \varepsilon_x + (1-\mu)\ \varepsilon_y] \\ \tau_x &= \frac{E}{(1+\mu)(1-2\mu)}\ \gamma_{xy}\ \frac{1}{2}(1-\mu) \end{aligned} \tag{10.7}$$

Maticově ($\boldsymbol{\sigma} = \mathbf{D}\ \boldsymbol{\varepsilon}$):

$$\left\{ \begin{array}{c} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{array} \right\} = \frac{E}{(1+\mu)(1-2\mu)} \left[\begin{array}{ccc} 1-\mu & \mu & 0 \\ \mu & 1-\mu & 0 \\ 0 & 0 & \frac{1}{2}(1-\mu) \end{array} \right] \left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{array} \right\}$$

Aproximace neznámých uzlových posunutí:

$$u(x,y) = a_1\ x + a_2\ y + a_3 \tag{10.8}$$

$$v(x,y) = a_4\ x + a_5\ y + a_6 \tag{10.9}$$

Maticově ($\mathbf{u} = \mathbf{U}\ \mathbf{a}$):

$$\left\{ \begin{array}{c} u \\ v \end{array} \right\} = \left[\begin{array}{cccccc} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{array} \right] \left\{ \begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{array} \right\} \tag{10.10}$$

Aproximace neznámých uzlových posunutí v uzlech 1, 2, 3 ($\mathbf{r} = \mathbf{S}\ \mathbf{a}$):

$$\left\{ \begin{array}{c} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \end{array} \right\} = \left[\begin{array}{cccccc} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{array} \right] \left\{ \begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{array} \right\} \tag{10.11}$$

Kombinací vztahů $\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T\ \mathbf{u}$ a $\mathbf{u} = \mathbf{U}\ \mathbf{a}$ vznikne $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{a}$, kde $\mathbf{B} = \boldsymbol{\partial}^T\ \mathbf{U}$:

$$\left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{array} \right\} = \left[\begin{array}{cc} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{array} \right] \left[\begin{array}{cccccc} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{array} \right] \left\{ \begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{array} \right\} \tag{10.12}$$

Kombinací vztahů $\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T\ \mathbf{u}$ a $\mathbf{u} = \mathbf{U}\ \mathbf{a}$ vznikne $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{a}$, kde $\mathbf{B} = \boldsymbol{\partial}^T\ \mathbf{U}$:

$$\left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{array} \right\} = \left[\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{array} \right] \left\{ \begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{array} \right\} \tag{10.13}$$

Z $\mathbf{r} = \mathbf{S}\ \mathbf{a}$ plyne: $\mathbf{a} = \mathbf{S}^{-1}\ \mathbf{r}$.
Pak: $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{S}^{-1}\ \mathbf{r}$.
Potenciální energie vnitřních sil:

$$\Pi_i = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T \boldsymbol{\sigma}\ d\ V = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V \tag{10.14}$$

Potenciální energie vnějších sil:

$$\Pi_e = -\int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{10.15}$$

Potenciální energie soustavy:

$$\Pi = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V - \int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{10.16}$$

Po dosazení za $\boldsymbol{\varepsilon}$ a vytknutí $\mathbf{r}$:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T \int_V \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\mathbf{B}\ \mathbf{S}^{-1}\ d\ V\ \mathbf{r}^T - \int_V \mathbf{X}^T\ d\ V\ \mathbf{r} - \int_S \mathbf{p}^T\ d\ S\ \mathbf{r}. \tag{10.17}$$

Stručně:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T\ \mathbf{K}\ \mathbf{r}\ -\ \mathbf{F}^T\ \mathbf{r}. \tag{10.18}$$

Aplikací Lagrangeova variačního principu ($\partial\ \Pi = min.$) na (12.13):

$$\mathbf{K}\ \mathbf{r}\ =\ \mathbf{F}, \tag{10.19}$$

kde $\mathbf{K}$ ... matice tuhosti konečného prvku:

$$\mathbf{K} = \int_V\ \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}\ d\ V, \tag{10.20}$$

$\mathbf{F}$ ... zatěžovací vektor konečného prvku:

$$\mathbf{F} = -\int_V \mathbf{X}^T\ d\ V\ -\int_S \mathbf{p}^T\ d\ S. \tag{10.21}$$

Pro studovaný konečný prvek:

$$\mathbf{K} = t\ A\ \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}, \tag{10.22}$$

kde t ... tloušťka konečného prvku.

$$\mathbf{F} = \mathbf{X} + \mathbf{p}. \tag{10.23}$$

10.1 Analýza konstrukce

Z $\mathbf{K}_e$ a $\mathbf{r}_e$ a $\mathbf{F}_e$ jednotlivých prvků (e je číslo prvku) sestavíme $\mathbf{K}$ a $\mathbf{r}$ a $\mathbf{F}$ celé konstrukce a neznámé určíme řešením soustavy rovnic:

$$\mathbf{K}\ \mathbf{r} = \mathbf{F}. \tag{10.24}$$

10.2 Výpočet výsledků na konečných prvcích

1. z vektoru $\mathbf{r}$ celé konstrukce sestavíme vektory $\mathbf{r}_e$ jednotliných konečných prvků
2. pro každý prvek stanovíme poměrné deformace:
   $\boldsymbol{\varepsilon}_e = \mathbf{B}\ \mathbf{S}^{-1}\ \mathbf{r}_e$
3. pro každý prvek stanovíme napětí:
   $\boldsymbol{\sigma}_e = \mathbf{D}\ \boldsymbol{\varepsilon}_e$ nebo $\boldsymbol{\sigma}_e = \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}\ \mathbf{r}_e$

10.3 Diskuse: spojitost a výstižnost výsledků

Použitá aproximace posunutí:

$$\begin{aligned} u(x,y) &= a_1\ x + a_2\ y + a_3 \\ v(x,y) &= a_4\ x + a_5\ y + a_6 \end{aligned}$$

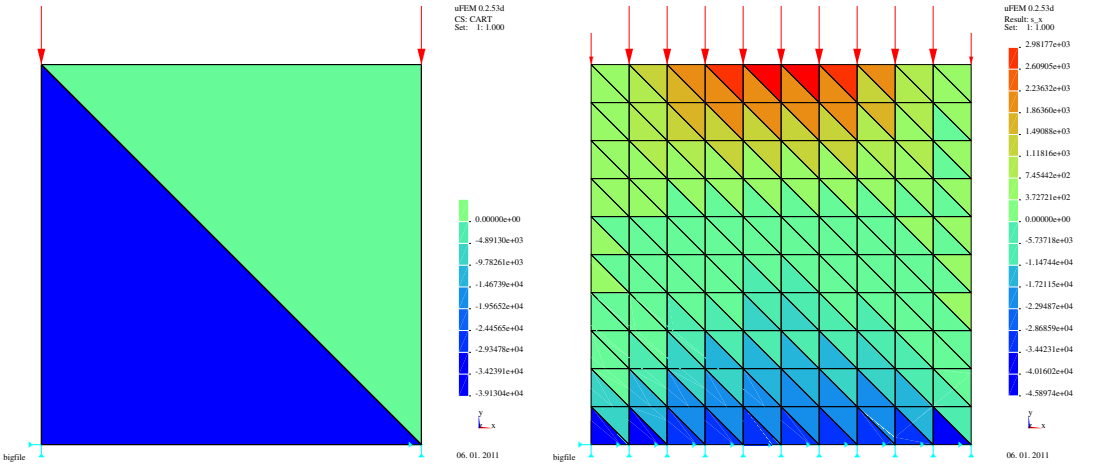
Tedy **polynom 1. stupně** pro posunutí:

- spojité deformace $\mathbf{r}$
- protože $\boldsymbol{\varepsilon} = \boldsymbol{\partial}\ \mathbf{r}$, **konstantní poměrné deformace** (po derivaci snížení na polynom 0. stupně)
- protože $\boldsymbol{\sigma} = \mathbf{D}\ \boldsymbol{\varepsilon}$, **konstantní napětí** (polynom 0. stupně)

V případě odvozeného konečného prvku tedy bude platit:

- pro uvedený prvek jsou deformace aproximovány „lineárně“,
- poměrné deformace a napětí jsou na prvku konstantní,
- pro přesnější výsledky $\Rightarrow$ hustší síť konečných prvků.

Ilustrace výsledných napětí na stěně analyzované s využitím popsaného konečného prvku pro dvě hustoty sítě:



10.4 Úloha k procvičení:

- Navrhněte úpravu výpočetního algoritmu tak, aby byl schopen řešit jednoduché stěny.

Kapitola 11

Nosné desky

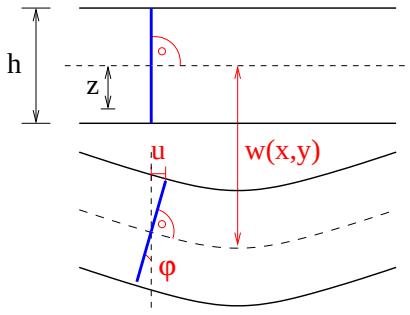
11.1 Základní předpoklady

Deska je těleso, které má jeden rozměr mnohem menší než rozměry zbývající. Zatížení desky je orientováno výhradně **kolmo k její střednicové rovině**.

V dalším textu bude předpokládána Kirchhoffova teorie ohybu tenkých desek:

- „technická teorie“ ohybu tenkých desek,
- jednotlivé vrstvy desky na sebe netlačí $\sigma_z = 0$,
- normálová napětí ve střednicové rovině jsou nulová,
- body ve střednicové rovině se mohou přemisťovat pouze ve směru osy z,
- normály střednicové roviny zůstávají i po deformaci přímé a kolmé k této rovině.

11.2 Deformace a poměrné deformace na desce



Rovnice vychází z geometrických závislosí (viz obrázek).

$$u = -z\,\frac{\partial w}{\partial x}, \quad v = -z\,\frac{\partial w}{\partial y} \tag{11.1}$$

$$\varepsilon_x = \frac{\partial u}{\partial x} = -z\,\frac{\partial^2 w}{\partial x^2} \tag{11.2}$$

$$\varepsilon_y = \frac{\partial u}{\partial y} = -z\,\frac{\partial^2 w}{\partial y^2} \tag{11.3}$$

$$\gamma_{xy} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = -2\,z\,\frac{\partial x}{\partial y} \tag{11.4}$$

11.3 Fyzikální rovnice na desce

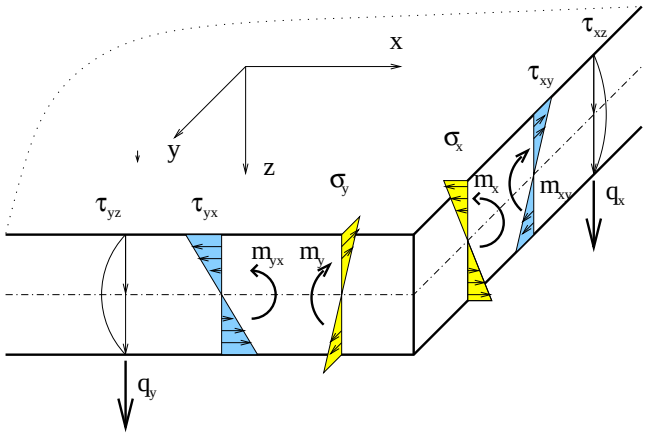
Budeme předpokládat lineární isotropní materiál:

$$\sigma_x = \frac{E}{1-\mu^2}\left(\varepsilon_x + \mu\,\varepsilon_y\right) = -\frac{E}{1-\mu^2}\left(\frac{\partial^2 w}{\partial x^2} + \mu\,\frac{\partial^2 w}{\partial y^2}\right) \tag{11.5}$$

$$\sigma_y = \frac{E}{1-\mu^2}\left(\varepsilon_y + \mu\,\varepsilon_x\right) = -\frac{E}{1-\mu^2}\left(\frac{\partial^2 w}{\partial y^2} + \mu\,\frac{\partial^2 w}{\partial x^2}\right) \tag{11.6}$$

$$\tau_{xy} = G\,\gamma = -\frac{E}{2\,(1+\mu)}\,2\,z\,\frac{\partial^2 w}{\partial x \partial y} \tag{11.7}$$

11.4 Vnitřní síly na desce



Momenty:

$$m_x = \int_{-t/2}^{t/2} \sigma_x\,z\,dx = -D\left(\frac{\partial^2 w}{\partial x^2} + \mu\frac{\partial^2 w}{\partial y^2}\right), \tag{11.8}$$

$$m_y = \int_{-t/2}^{t/2} \sigma_y\,z\,dx = -D\left(\mu\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2}\right), \tag{11.9}$$

$$m_{xy} = \int_{-t/2}^{t/2} \tau_{xy}\,z\,dx = -D\,(1-\mu)\,\frac{\partial^2 w}{\partial x\;y}, \tag{11.10}$$

kde D je **desková tuhost**:

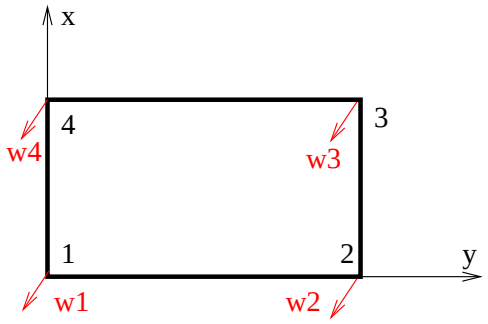
$$D = \frac{E\,t^3}{12\,(1-\mu^2)} \tag{11.11}$$

Posouvající síly:

$$q_x = \int_{-t/2}^{t/2} \tau_{xz} dz = -D \left(\frac{\partial^3 w}{\partial x^3} + \frac{\partial^3 w}{\partial x \partial y^2} \right), \tag{11.12}$$

$$q_y = \int_{-t/2}^{t/2} \tau_{yz} dz = -D \left(\frac{\partial^3 w}{\partial x^3} + \frac{\partial^3 w}{\partial x^2 \partial y} \right). \tag{11.13}$$

11.5 Odvození konečného prvku pro tenkou desku



Neznámé parametry deformace: w, φ_x, φ_y v každém uzlu.

Tj. celkem dvanáct neznámých uzlových parametrů:

$$\mathbf{r} = \{w_1, \varphi_{x,1}, \varphi_{y,1}, w_2, \varphi_{x,2}, \varphi_{y,2}, w_3, \varphi_{x,3}, \varphi_{y,3}\}^T,$$

$$\varphi_{x,i} = \frac{\partial w_i}{\partial x}, \quad \varphi_{y,i} = \frac{\partial w_i}{\partial y}.$$

Geometrické rovnice:

$$\varepsilon_x = -\frac{\partial^2 w}{\partial x^2}, \quad \varepsilon_y = -\frac{\partial^2 w}{\partial y^2}, \quad \tau_{xy} = -2\frac{\partial^2 w}{\partial x \partial y}. \tag{11.14}$$

Maticově ($\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T \mathbf{u}$):

$$\left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{array} \right\} = \left[\begin{array}{c} -\frac{\partial^2}{\partial x} \\ -\frac{\partial^2}{\partial y} \\ -2\frac{\partial^2}{\partial x \partial y} \end{array} \right] \left\{ \begin{array}{c} w \end{array} \right\} \tag{11.15}$$

Fyzikální rovnice (bez q_x, q_y!):

$$m_x = \frac{E\ h^3}{12(1-\mu^2)} (\varepsilon_x + \mu\ \varepsilon_y) \tag{11.16}$$

$$m_y = \frac{E\ h^3}{12(1-\mu^2)} (\varepsilon_y + \mu\ \varepsilon_x) \tag{11.17}$$

$$m_{xy} = \frac{E\ h^3}{24(1+\mu^2)}\ \gamma_{xy} \tag{11.18}$$

Maticově ($\boldsymbol{\sigma} = \mathbf{D}\ \boldsymbol{\varepsilon}$):

$$\left\{ \begin{array}{c} m_x \\ m_y \\ m_{xy} \end{array} \right\} = \frac{E\ h^3}{12(1-\mu^2)} \left[\begin{array}{ccc} 1 & \mu & 0 \\ \mu & 1 & 0 \\ 0 & 0 & \frac{1}{2}(1-\mu) \end{array} \right] \left\{ \begin{array}{c} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{array} \right\} \tag{11.19}$$

Neznámá uzlová posunutí:

$$\left\{ \begin{array}{c} w \\ \\ \varphi_x \\ \\ \varphi_y \end{array} \right\} = \left\{ \begin{array}{c} w \\ \\ \frac{\partial w}{\partial x} \\ \\ -\frac{\partial w}{\partial y} \end{array} \right\} \tag{11.20}$$

Aproximace neznámých uzlových posunutí:

$$\begin{aligned} w &= a_1\ + a_2\ x + a_3\ y + a_4\ x^2 + a_5\ xy + a_6\ y^2 \\ &+ a_7\ x^3 + a_8\ x^2 y + a_9\ xy^2 + a_{10} y^3 + a_{11} x^3 y + a_{12} x y^3 \end{aligned} \tag{11.21}$$

$$\varphi_x = a_2 + 2a_4 x + a_5 y + 3a_7 x^2 + 2a_8 xy + a_9 y^2 + 3a_{11} x^2 y + a_{12} y^3$$

$$\varphi_y = -a_3 - a_5 x - 2a_6 y - a_8 x^2 - 2a_9 xy - 3a_{10} y^2 - a_{11} x^3 - 3a_{12} x y^2$$

Maticově ($\mathbf{u} = \mathbf{U}\ \mathbf{a}$): $\{w, \varphi_x, \varphi_y\}^T =$

$$\left[\begin{array}{cccccccccccc} 1 & x & y & x^2 & xy & y^2 & x^3 & x^2 y & xy^2 & y^3 & x^3 y & xy^3 \\ 0 & 1 & 0 & 2x & y & 0 & 3x^2 & xy & y^2 & 0 & 3x^2 y & 3y^3 \\ 0 & 0 & -1 & 0 & -x & -2y & 0 & -x^2 & -xy & 3y^2 & -x^3 & -3xy^2 \end{array} \right] \left\{ \begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \end{array} \right\} \tag{11.22}$$

Aproximace neznámých uzlových posunutí v uzlech 1, 2, 3, 4 ($\mathbf{r} = \mathbf{S}\ \mathbf{a}$):

$$\{w_1, \varphi_{x,1}, \varphi_{y,1}, w_2, \varphi_{x,2}, \varphi_{y,2}, w_3, \varphi_{x,3}, \varphi_{y,3}, w_4, \varphi_{x,4}, \varphi_{y,4}\}^T =$$

$$\begin{bmatrix}
1 & x_1 & y_1 & x_1^2 & x_1y_1 & y_1^2 & x_1^3 & x_1^2y_1 & x_1y_1^2 & y_1^3 & x_1^3y_1 & x_1y_1^3 \\
0 & 1 & 0 & 2x_1 & y_1 & 0 & 3x_1^2 & x_1y_1 & y_1^2 & 0 & 3x_1^2y_1 & 3y_1^3 \\
0 & 0 & -1 & 0 & -x_1 & -2y_1 & 0 & -x_1^2 & -x_1y_1 & 3y_1^2 & -x_1^3 & -3x_1y_1^2 \\
1 & x_2 & y_2 & x_2^2 & x_2y_2 & y_2^2 & x_2^3 & x_2^2y_2 & x_2y_2^2 & y_2^3 & x_2^3y_2 & x_2y_2^3 \\
0 & 1 & 0 & 2x_2 & y_2 & 0 & 3x_2^2 & x_2y_2 & y_2^2 & 0 & 3x_2^2y_2 & 3y_2^3 \\
0 & 0 & -1 & 0 & -x_2 & -2y_2 & 0 & -x_2^2 & -x_2y_2 & 3y_2^2 & -x_2^3 & -3x_2y_2^2 \\
1 & x_3 & y_3 & x_3^2 & x_3y_3 & y_3^2 & x_3^3 & x_3^2y_3 & x_3y_3^2 & y_3^3 & x_3^3y_3 & x_3y_3^3 \\
0 & 1 & 0 & 2x_3 & y_3 & 0 & 3x_3^2 & x_3y_3 & y_3^2 & 0 & 3x_3^2y_3 & 3y_3^3 \\
0 & 0 & -1 & 0 & -x_3 & -2y_3 & 0 & -x_3^2 & -x_3y_3 & 3y_3^2 & -x_3^3 & -3x_3y_3^2 \\
1 & x_4 & y_4 & x_4^2 & x_4y_4 & y_4^2 & x_4^3 & x_4^2y_4 & x_4y_4^2 & y_4^3 & x_4^3y_4 & x_4y_4^3 \\
0 & 1 & 0 & 2x_4 & y_4 & 0 & 3x_4^2 & x_4y_4 & y_4^2 & 0 & 3x_4^2y_4 & 3y_4^3 \\
0 & 0 & -1 & 0 & -x_4 & -2y_4 & 0 & -x_4^2 & -x_4y_4 & 3y_4^2 & -x_4^3 & -3x_4y_4^2
\end{bmatrix}
\begin{Bmatrix}
a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12}
\end{Bmatrix} \tag{11.23}$$

Kombinací vztahů $\boldsymbol{\varepsilon} = \boldsymbol{\partial}^T\ \mathbf{u}$ a $\mathbf{u} = \mathbf{U}\ \mathbf{a}$ vznikne $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{a}$, kde $\mathbf{B} = \boldsymbol{\partial}^T\ \mathbf{U}$.
Z $\mathbf{r} = \mathbf{S}\ \mathbf{a}$ plyne: $\mathbf{a} = \mathbf{S}^{-1}\ \mathbf{r}$ a $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{S}^{-1}\mathbf{r}$.

Potenciální energie vnitřních sil:

$$\Pi_i = \frac{1}{2}\ \int_V \boldsymbol{\varepsilon}^T \boldsymbol{\sigma}\ d\ V = \frac{1}{2}\ \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V \tag{11.24}$$

Potenciální energie vnějších sil:

$$\Pi_e = -\int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{11.25}$$

Potenciální energie soustavy:

$$\Pi = \frac{1}{2}\ \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V - \int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{11.26}$$

Po dosazení za $\boldsymbol{\varepsilon}$ a vytknutí $\mathbf{r}$:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T\ \int_V \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{DB}\ \mathbf{S}^{-1}\ d\ V\ \mathbf{r}^T - \int_V \mathbf{X}^T\ d\ V\ \mathbf{r} - \int_S \mathbf{p}^T\ d\ S\ \mathbf{r}. \tag{11.27}$$

Stručně:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T\ \mathbf{K}\ \mathbf{r}\ -\ \mathbf{F}^T\ \mathbf{r}. \tag{11.28}$$

Aplikací Lagrangeova variačního principu ($\partial\ \Pi = min.$) na (12.13):

$$\mathbf{K}\ \mathbf{r}\ =\ \mathbf{F}, \tag{11.29}$$

kde $\mathbf{K}$ ... matice tuhosti konečného prvku:

$$\mathbf{K} = \int_V\ \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}\ d\ V, \tag{11.30}$$

$\mathbf{F}$ ... zatěžovací vektor konečného prvku:

$$\mathbf{F} = -\int_V \mathbf{X}^T\ d\ V\ -\int_S \mathbf{p}^T\ d\ S. \tag{11.31}$$

Pro studovaný konečný prvek je vhodné použít numerickou integraci ve 2D, protože matice $\mathbf{B}$ obsahuje proměnné x, y a její obecné vyčíslení je pracné.

$$\mathbf{K} = t \int_A \mathbf{S}^{-1T}\, \mathbf{B}^T\, \mathbf{D}\, \mathbf{B}\, \mathbf{S}^{-1}\, d\, A, \tag{11.32}$$

kde t ... tloušťka konečného prvku.

$$\mathbf{F} = \mathbf{X} + \mathbf{p}. \tag{11.33}$$

Je vhodné připomenout, že integrace na obecném čtyřúhelníku není zcela snadná, proto je vhodné buď pracovat s prvky tvaru čtverce nebo obdélníka, případně použít izoparametrické prvky, které budou uvedeny v dalším textu.

11.6 Úloha k procvičení:

- V sestevném programu pro výpočet stěn najděte části, které by vyžadovaly změnu, aby s nimi bylo možné řešit nosné desky.

Kapitola 12

Konečné prvky pro řešení 3D úloh

12.1 Geometrické vztahy ve 3D

Maticově ($\boldsymbol{\varepsilon} = \varepsilon\partial\ \mathbf{u}$):

$$\begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{yz} \\ \gamma_{zx} \\ \gamma_{xy} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial\ x} & 0 & 0 \\ 0 & \frac{\partial}{\partial\ y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial\ z} & \frac{\partial}{\partial\ y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial\ x} \\ \frac{\partial}{\partial\ y} & \frac{\partial}{\partial\ x} & 0 \end{bmatrix} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix}, \tag{12.1}$$

12.2 Fyzikální vztahy ve 3D

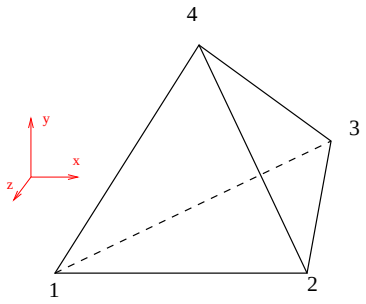
Maticově ($\boldsymbol{\sigma} = \mathbf{D}\ \boldsymbol{\varepsilon}$) pro lineárně pružný materiál:

$$\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{yz} \\ \tau_{zx} \\ \tau_{xy} \end{Bmatrix} = A \begin{bmatrix} 1 & \frac{\mu}{1-\mu} & \frac{\mu}{1-\mu} & 0 & 0 & 0 \\ \frac{\mu}{1-\mu} & 1 & \frac{\mu}{1-\mu} & 0 & 0 & 0 \\ \frac{\mu}{1-\mu} & \frac{\mu}{1-\mu} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\ \mu}{2(1-\mu)} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\ \mu}{2(1-\mu)} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\ \mu}{2(1-\mu)} \end{bmatrix} \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{yz} \\ \gamma_{zx} \\ \gamma_{xy} \end{Bmatrix},$$

$$A = \frac{E(1-\mu)}{(1+\mu)\ (1-2\ \mu)} \tag{12.2}$$

12.3 Odvození konečného prvku pro 3D úlohy

Odvození konečného prvku je analogické odvození konečného provku pro rovinný problém, hlavní rozdíl je v počtu neznámých v prostoru.



Neznámé parametry deformace: u, v, w v každém uzlu.

Tedy prvek má celkem dvanáct neznámých uzlových parametrů:

$$\{u_1, v_1, w_1, u_2, v_2, w_2, u_3, v_3, w_3, u_4, v_4, w_4\}^T.$$

Aproximace neznámých uzlových posunutí:

$$u(x,y) = a_1\ x + a_2\ y + a_3\ z + a_4 \tag{12.3}$$
$$v(x,y) = a_5\ x + a_6\ y + a_7\ z + a_8 \tag{12.4}$$
$$w(x,y) = a_9\ x + a_{10}\ y + a_{11}\ z + a_{12} \tag{12.5}$$

Maticově ($\mathbf{u} = \mathbf{U}\ \mathbf{a}$):

$$\left\{\begin{array}{c} u \\ v \\ w \end{array}\right\} = \left[\begin{array}{cccccccccccc} x & y & z & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x & y & z & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & y & z & 1 \end{array}\right] \left\{\begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \end{array}\right\} \tag{12.6}$$

Aproximace neznámých uzlových posunutí v uzlech 1, 2, 3, 4 ($\mathbf{r} = \mathbf{S}\ \mathbf{a}$):

$$\left\{\begin{array}{c} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ u_3 \\ v_3 \\ w_3 \\ u_4 \\ v_4 \\ w_4 \end{array}\right\} = \left[\begin{array}{cccccccccccc} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_2 & y_2 & z_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_3 & y_3 & z_3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_4 & y_4 & z_4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_4 & y_4 & z_4 & 1 \end{array}\right] \left\{\begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \end{array}\right\} \tag{12.7}$$

Kombinací vztahů $\boldsymbol{\varepsilon} = \boldsymbol{\partial}\ \mathbf{u}$ a $\mathbf{u} = \mathbf{U}\ \mathbf{a}$ vznikne $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{a}$, kde $\mathbf{B} = \boldsymbol{\partial}^T\ \mathbf{U}$:

$$\begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{yz} \\ \gamma_{zx} \\ \gamma_{xy} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \end{Bmatrix} \tag{12.8}$$

Z $\mathbf{r} = \mathbf{S}\ \mathbf{a}$ plyne: $\mathbf{a} = \mathbf{S}^{-1}\ \mathbf{r}$.
Pak: $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{S}^{-1}\mathbf{r}$.
Potenciální energie vnitřních sil:

$$\Pi_i = \frac{1}{2}\ \int_V \boldsymbol{\varepsilon}^T \boldsymbol{\sigma}\ d\ V = \frac{1}{2}\ \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V. \tag{12.9}$$

Potenciální energie vnějších sil:

$$\Pi_e = -\int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{12.10}$$

Potenciální energie soustavy:

$$\Pi = \frac{1}{2}\ \int_V \boldsymbol{\varepsilon}^T\ \mathbf{D}\ \boldsymbol{\varepsilon}\ d\ V - \int_V \mathbf{X}^T\ \mathbf{r}\ d\ V - \int_S \mathbf{p}^T \mathbf{r}\ d\ S. \tag{12.11}$$

Po dosazení za $\boldsymbol{\varepsilon}$ a vytknutí $\mathbf{r}$:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T\ \int_V \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{DB}\ \mathbf{S}^{-1}\ d\ V\ \mathbf{r}^T - \int_V \mathbf{X}^T\ d\ V\ \mathbf{r} - \int_S \mathbf{p}^T\ d\ S\ \mathbf{r}. \tag{12.12}$$

Stručně:

$$\Pi = \frac{1}{2}\ \mathbf{r}^T\ \mathbf{K}\ \mathbf{r}\ -\ \mathbf{F}^T\ \mathbf{r}. \tag{12.13}$$

Aplikací Lagrangeova variačního principu ($\partial\ \Pi = min.$) na (12.13):

$$\mathbf{K}\ \mathbf{r}\ =\ \mathbf{F}, \tag{12.14}$$

kde $\mathbf{K}$ ... matice tuhosti konečného prvku:

$$\mathbf{K} = \int_V\ \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}\ d\ V, \tag{12.15}$$

$\mathbf{F}$ ... zatěžovací vektor konečného prvku:

$$\mathbf{F} = -\int_V \mathbf{X}^T\ d\ V\ -\int_S \mathbf{p}^T\ d\ S. \tag{12.16}$$

Pro studovaný konečný prvek ($\mathbf{B}, \mathbf{D}, \mathbf{S^{-1}}$ obsahují jen konstanty):

$$\mathbf{K} = \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1} \int_V dV, \tag{12.17}$$

$$\mathbf{K} = V\ \mathbf{S}^{-1T}\ \mathbf{B}^T\ \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}. \tag{12.18}$$

$$\mathbf{F} = \mathbf{X}. \tag{12.19}$$

12.4 Analýza konstrukce

Z $\mathbf{K}_e$ a $\mathbf{r}_e$ a $\mathbf{F}_e$ jednotlivých prvků (e je číslo prvku) sestavíme $\mathbf{K}$ a $\mathbf{r}$ a $\mathbf{F}$ celé konstrukce a neznámé určíme řešením soustavy rovnic:

$$\mathbf{K}\ \mathbf{r} = \mathbf{F}. \tag{12.20}$$

12.5 Výpočet výsledků (napětí a deformací) na konečných prvcích

1. z vektoru $\mathbf{r}$ celé konstrukce sestavíme vektory $\mathbf{r}_e$ jednotliných konečných prvků
2. pro každý prvek stanovíme poměrné deformace:
   $\boldsymbol{\varepsilon}_e = \mathbf{B}\ \mathbf{S}^{-1}\ \mathbf{r}_e$
3. pro každý prvek stanovíme napětí:
   $\boldsymbol{\sigma}_e = \mathbf{D}\ \boldsymbol{\varepsilon}_e$ nebo $\boldsymbol{\sigma}_e = \mathbf{D}\ \mathbf{B}\ \mathbf{S}^{-1}\ \mathbf{r}_e$

12.6 Úloha k procvičení:

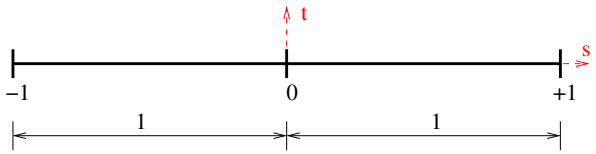
- V sestevném programu pro výpočet stěn najděte části, které by vyžadovaly změnu, aby s nimi bylo možné řešit konstrukce pomocí objemových prvků.

Kapitola 13

Izoparametrické konečné prvky

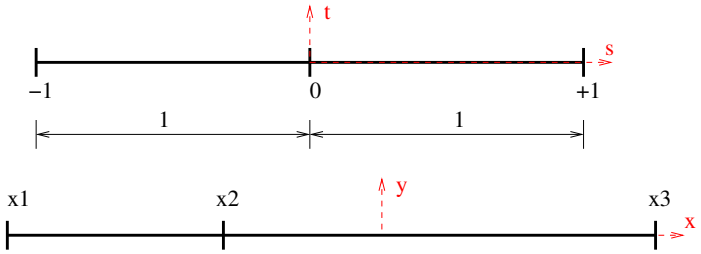
- Proces tvorby matice tuhosti konečného prvku probíhá na jednotkovém obrazci (čtverec, krychle) v jednotkových souřadnicích (η, ξ, ζ)
- Pomocí zvolených funkcí se jednotkový obrazec (η, ξ, ζ) zobrazí na skutečný tvar prvku (x, y, z)
- Funkce použité pro zobrazení se použijí i jako aproximace hledané veličiny

13.1 Jednotkový konečný prvek



- Délka strany prvku je $1 + 1 = 2$
- Použijeme přirozené souřadnice s, t (v literatuře často např. i ξ, η)

13.2 Jednotkový a skutečný prvek

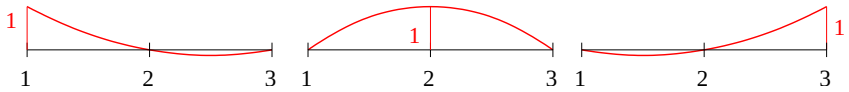


Vztah mezi x a s:

$$x = \sum_i N_i(s)\ x_i, \tag{13.1}$$

kde N_i jsou **tvarové funkce** .

13.3 Tvarové funkce



Aby bylo možné použít rovnici (13.5), musí tvarové funkce N_i nabývat:

- v bodě i hodnoty 1,
- ve všech ostatních bodech hodnoty 0.

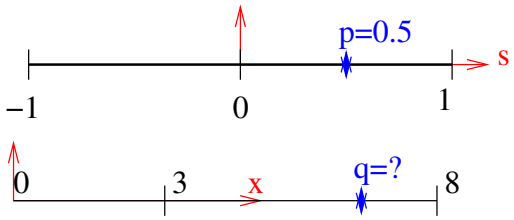
13.4 Vyjádření polohy bodů

$$x = \sum_i N_i(s)\ x_i, \tag{13.2}$$

Pro uvedený trojuzlový prvek:

$$x = N_1(s)\ x_1 + N_2(s)\ x_2 + N_3(s)\ x_3 \tag{13.3}$$

13.5 Ukázka použití



Volba tvarových funkcí pro zobrazený případ:

$$\begin{aligned} N_1 &= -\frac{s(1-s)}{2} \\ N_2 &= (1-s^2) \\ N_3 &= \frac{s(1+s)}{2} \end{aligned} \tag{13.4}$$

Tedy:

$$x = N_1\ x_1 + N_2\ x_2 + N_3\ x_3 = -\frac{s(1-s)}{2}\ x_1 + (1-s^2)\ x_2 + \frac{s(1+s)}{2}\ x_3.$$

$$p = -\frac{0.5(1-0.5)}{2}\ 0 + (1-0.5^2)\ 3 + \frac{0.5(1+0.5)}{2}\ 8 = 5.88$$

Pro kontrolu:

$$x(s=0) = -\frac{0(1-0)}{2}\ 0 + (1-0^2)\ 3 + \frac{0(1+0)}{2}\ 8 = 3$$

13.6 Vyjádření neznámých posunutí

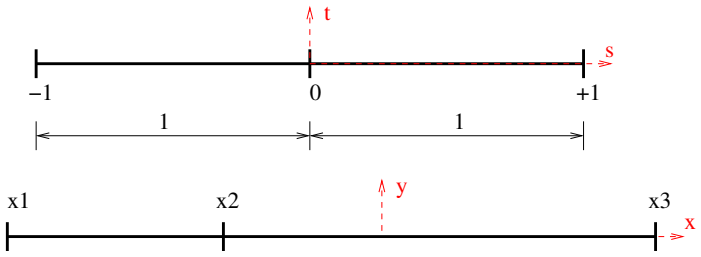
$$u = \sum_i N_i(s)\ u_i, \tag{13.5}$$

Pro uvedený trojuzlový prvek:

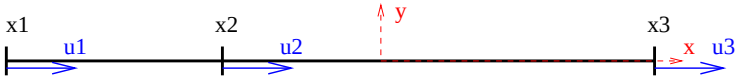
$$u = N_1(s)\ u_1 + N_2(s)\ u_2 + N_3(s)\ u_3 \tag{13.6}$$

13.7 Trojuzlový prvek příhradoviny

Jednotkový a skutečný prvek:



Neznámá posunutí:



Na prvku budeme uvažovat celkem tři neznámá posunutí $u = \{u_1, u_2, u_3\}^T$
Volba tvarových funkcí:

$$\begin{aligned} N_1 &= -\frac{s(1-s)}{2} \\ N_2 &= (1-s^2) \\ N_3 &= \frac{s(1+s)}{2} \end{aligned} \tag{13.7}$$

Tedy:

$$x = N_1\ x_1 + N_2\ x_2 + N_3\ x_3 = -\frac{s(1-s)}{2}\ x_1 + (1-s^2)\ x_2 + \frac{s(1+s)}{2}\ x_3 \tag{13.8}$$

Maticově:

$$\{x\} = \begin{bmatrix} N_1 & N_2 & N_3 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} \tag{13.9}$$

$$\{x\} = \begin{bmatrix} -\frac{s(1-s)}{2} & (1-s^2) & \frac{s(1+s)}{2} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix}$$

Aproximace posunutí u (stejnými funkcemi jako x):

$$u = \sum_i N_i(s)\ u_i, \tag{13.10}$$

$$u = N_1\ u_1 + N_2\ u_2 + N_3\ u_3 = -\frac{s(1-s)}{2}\ u_1 + (1-s^2)\ u_2 + \frac{s(1+s)}{2}\ u_3$$

Maticově:

$$\{u\} = \left[\begin{array}{ccc} N_1 & N_2 & N_3 \end{array}\right] \left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\} \tag{13.11}$$

$$\{u\} = \left[\begin{array}{ccc} -\frac{s(1-s)}{2} & (1-s^2) & \frac{s(1+s)}{2} \end{array}\right] \left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\}$$

Poměrné derivace:

$$\varepsilon = \boldsymbol{\partial}\mathbf{u} \tag{13.12}$$

tedy:

$$\varepsilon = \frac{\partial u}{\partial x} = \{\frac{\partial}{\partial x}\} \left[\begin{array}{ccc} -\frac{s(1-s)}{2} & (1-s^2) & \frac{s(1+s)}{2} \end{array}\right] \left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\} \tag{13.13}$$

a po derivaci:

$$\varepsilon = \left[\begin{array}{ccc} \frac{\partial(-\frac{s(1-s)}{2})}{\partial x} & \frac{\partial(1-s^2)}{\partial x} & \frac{\partial\frac{s(1+s)}{2}}{\partial x} \end{array}\right] \left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\}, \tag{13.14}$$

stručně $\boldsymbol{\varepsilon} = \mathbf{B}\ \mathbf{u}$:

$$\varepsilon = \left[\begin{array}{ccc} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x} \end{array}\right] \left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\} \tag{13.15}$$

Problém: Ve vztazích pro ε je $\frac{\partial N_i}{\partial x}$, ale N_i je funkcí s.
Platí (z derivačního počtu):

$$\frac{\partial N_i}{\partial x} = \frac{\partial N_i}{\partial s}\ \frac{\partial s}{\partial x} \tag{13.16}$$

Podle rovnice (13.5):

$$x = \sum_i N_i(s)\ x_i$$

Derivací vztahu (13.5):

$$\frac{dx}{ds} = \sum_i \frac{\partial N_i}{\partial s}\ x_i = J \tag{13.17}$$

Výraz (obecně matice) J je nazývá **Jakobián tranformace**.
Ze vztahu

$$\frac{dx}{ds} = J$$

vyjádříme ds:

$$ds = \frac{1}{J}\, dx \tag{13.18}$$

Poznámka: Obecně bude místo $\frac{1}{J}$ inverzní matice: $\mathbf{J^{-1}}$.

Ze vztahů

$$\frac{\partial N_i}{\partial x} = \frac{\partial N_i}{\partial s}\,\frac{\partial s}{\partial x}$$

a

$$\partial s = \frac{1}{J}\,\partial x$$

plyne:

$$\frac{\partial N_i}{\partial x} = \frac{\partial N_i}{\partial s}\frac{1}{J} \tag{13.19}$$

Poznámka: Obecně bude místo $\frac{1}{J}$ inverzní matice: $\mathbf{J^{-1}}$.

Pomocí vztahu (13.19) vyjádříme matici $\mathbf{B}$:

$$\mathbf{B} = \left[\begin{array}{ccc} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x} \end{array}\right] = \frac{1}{J}\left[\begin{array}{ccc} \frac{\partial N_1}{\partial s} & \frac{\partial N_2}{\partial s} & \frac{\partial N_3}{\partial s} \end{array}\right] \tag{13.20}$$

Tedy:

$$\{\varepsilon\} = \frac{1}{J}\left[\begin{array}{ccc} \frac{\partial N_1}{\partial s} & \frac{\partial N_2}{\partial s} & \frac{\partial N_3}{\partial s} \end{array}\right]\left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\} \tag{13.21}$$

Nebo:

$$\varepsilon = \left[\begin{array}{ccc} -\frac{(1-2s)}{2} & (-2s) & \frac{(1+2s)}{2} \end{array}\right]\left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\}$$

Známe-li $\boldsymbol{\varepsilon} = \mathbf{Dr}$:

$$\{\varepsilon\} = \frac{1}{J}\left[\begin{array}{ccc} \frac{\partial N_1}{\partial s} & \frac{\partial N_2}{\partial s} & \frac{\partial N_3}{\partial s} \end{array}\right]\left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array}\right\},$$

můžeme zapsat vztah pro výpočet potenciální energie vnitřních sil:

$$\Pi_i = \frac{1}{2}\int_V \boldsymbol{\varepsilon}^T\mathbf{D}\boldsymbol{\varepsilon} dV = \frac{1}{2}\mathbf{r}^T\int_V \mathbf{B}^T\mathbf{D}\mathbf{B}\; dV\; \mathbf{r}, \tag{13.22}$$

a pro matici tuhosti prutového prvku:

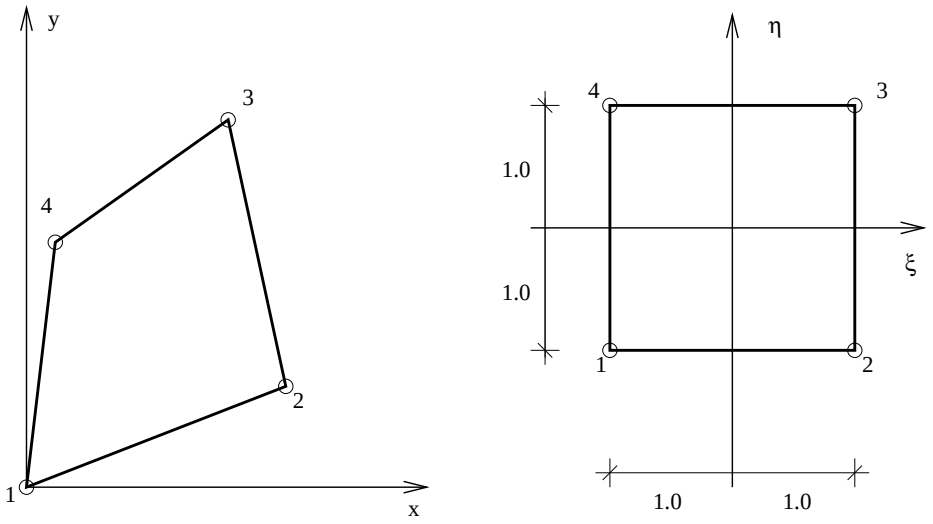
$$\mathbf{K} = A\int_{x_1}^{x_3} \mathbf{B}^T\mathbf{D}\mathbf{B} dx. \tag{13.23}$$

Rovnici (13.24) je v daném případě možné zintegrovat analyticky, u složitějších konečných prvků (prakticky použitelných - 2D, 3D) to obvykle možné není, proto se využívá **numerická integrace**, nejčastěji Gaussova integrační formule:

$$\mathbf{K} = A\sum_{i=1}^{m} \mathbf{B}^T\mathbf{D}\mathbf{B} w_i, \tag{13.24}$$

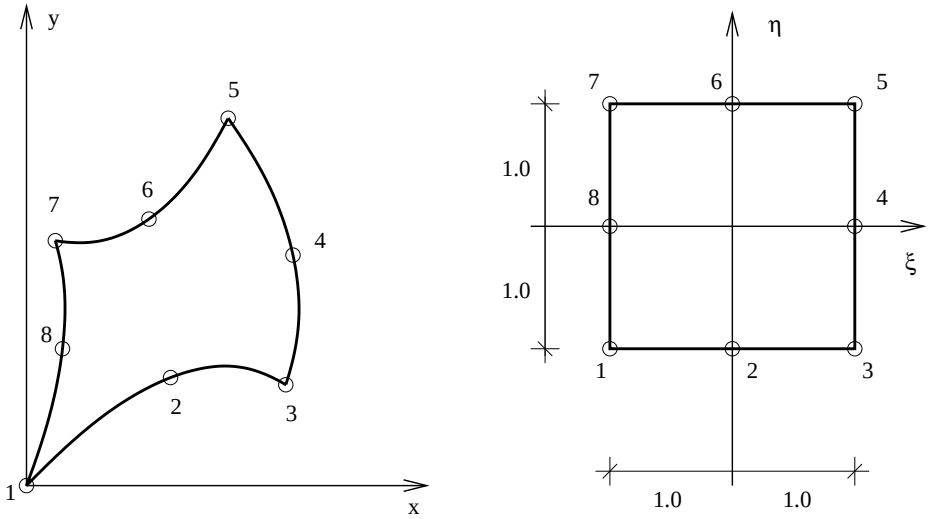
kde w_i je váha integračního bodu, m je počet integračních bodů.

13.8 Izoparametrické prvky pro rovinný problém



Tvarové funkce pro zobrazený čtyřuzlový prvek:

$$N_i(\xi,\eta) = \frac{1}{4}(1+\xi\xi_i)(1+\eta\eta_i), \tag{13.25}$$



V případě osmiuzlovho prvku budou mít tvarové funkce mírně složitější tvar. Tvarové funkce pro vrcholy:

$$N_i(\xi,\eta) = \frac{1}{4}(1+\xi\xi_i)(1+\eta\eta_i)(\xi\ \xi_i + \eta\ \eta_i - 1), \tag{13.26}$$

pro středy stran

$$N_i(\xi,\eta) = \frac{\xi_i^2}{2}(1+\xi\xi_i)(1-\eta^2) + \frac{\eta^2}{2}(1-\xi\ \xi_i)(1-\eta^2). \tag{13.27}$$

Výše uvedené izoparametrické konečné prvky se běžně používají ve výpočetních programech (např. ANSYS, ABAQUS aj.).

13.9 Úloha k procvičení

- Na Internetu vyhledejte software pro řešení úloh statiky stavebních konstrukcí, který využívá izoparametrické konečné prvky.

Kapitola 14

Konečné prvky na pružném podloží

V dalším textu uvážíme jen tyto případy:

- modely podloží,
- deska na Winklerově pružném podloží.

14.1 Obvyklé modely podloží

- pružný poloprostor
- vrstevnatý pružný poloprostor
- kontaktní modely (Winklerův, Pasternakův)

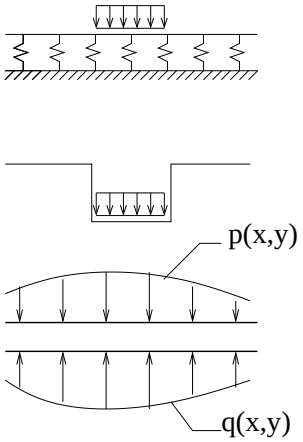
14.2 Pružný poloprostor

- vychází z předpokladů teorie pružnosti,
- pružná oblast ohraničená jen z jedné strany („povrch poloprostoru“),
- homogenní, obvykle izotropní materiál: E, ν.

Pro řešení pomocí metody konečných prvků můžeme uvážit tyto případy:

- modelování jen „dostatečně velkého“ výseku poloprostoru,
- modelování jen výseku poloroviny pro úlohy rovinné deformace,
- vrstevnatý poloprostor: různé vlastnosti konečných prvků.

14.3 Kontaktní modely podloží: Winklerův model



Winklerův model:

$$q(x,y) = C\ w(x,y) \tag{14.1}$$

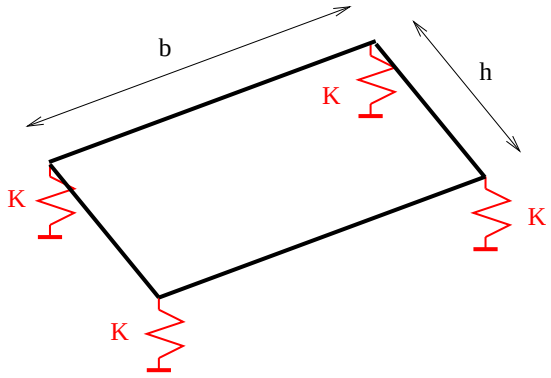
C ... součinitel stlačitelnosti podkladu $[\frac{N}{m^3}]$.

14.4 Deska na Winklerově podkladu

- v MKP se obvykle tuhost podloží rozloží do uzlů
- na obdélníkovém prvku např.:

$$K = \frac{1}{4}\ C_1\ b\ h$$

- možno použít i „ručně“ pokud to výpočetní program nepodporuje
- například program ANSYS: prvek SHELL63, „reálná“ konstanta EFS



- výpočet „matice tuhosti podloží” pomocí tvarových funkcí $\mathbf{N}$:

$$\mathbf{K_e} = C_1 \int_A \mathbf{N^T}\ \mathbf{N}\ dA,$$

kde C_1 je konstanta Winklerova podkladu

- pro čtyřuzlový obdélníkový prvek:

$$K = \frac{1}{4}\, C_1\, b\, h$$

14.5 Úloha k procvičení:

- Uvažte případ, kdy je obdélníková deska o rozměrech $4x4m$ zatížena po celé své ploše konstantním plošným zatížením $q = 12\ kN/m^2$. Konstanta $C = 2\ 000\ 000\ \frac{N}{m^3}$. Navrhněte postup, jak co nejjednodušeji vypočítat deformaci ve středu této desky.

Kapitola 15

Programy pro výpočty metodou konečných prvků

K řešení úloh metodou konečných zpravidla využíváme existující programy. Ty se dělí do několika skupin:

- Akademické programy (často dostupné včetně zdrojových kódů, zkušený uživatel je proto může měnit a vylepšovat): např **OOFEM, uFEM**.
- Obecné programy a programové balíky (obvykle komerční): např. **ABAQUS, ANSYS, COMSOL, NASTRAN** (výpočetní jádro NASTRAN je dostupné včetně zdrojových kódů, nicméně nadstavby jsou k dispozici již jako komerční programy různých výrobců)
- Specializované programy pro konkrétní oblast, často určené pro konkrétní region (optimalizované s ohledem na platné normy, návrhové zvyklosti aj.): např. pro Českou republiku a kokolí jde o programy pro navrhování stavebních konstrukcí SCIA Engineer, RFEM, pro úlohy tepelné techniky TEPLO, AREA,...

Literatura

[1] Krejsa, M., Algoritmizace inženýrských výpočtů, učební texty v obrazovkové verzi i ve verzi pro tisk, VŠB-TU, Ostrava, 2017.

[2] Brožovský, J., Materna, M. Metoda konečných prvků ve stavební mechanice, VŠB-TU, Ostrava, 2012. On-line: `https://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/metoda_konecnych_prvku_stavebni_`

[3] Klaus-Jürgen Bathe, K.-J. Finite Element Procedures (2nd ed.). Klaus-Jürgen Bathe. 2007.

[4] *Algoritmus*. Webové stránky zaměřené na tvorbu algoritmů. [on-line]. ¡http://www.algoritmy.net¿. (Citováno na s **??**.)

[5] Eaton, J.W. *Octave*. Programový systém pro provádění matematických výpočtů. Freeware, verze 4.2.1. [on-line]. ¡http://www.gnu.org/software/octave¿. University of Wisconsin, Department of Chemical Engineering, 1998-2017. (Citováno na s **??**.)

[6] Just, M. *Octave — český průvodce programem*. Elektronický manuál programového systému Octave. [on-line]. ¡http://www.octave.cz¿. Univerzita Tomáe Bati ve Zlíně, Fakulta aplikované informatiky, 2006. (Citováno na s **??**.)

[7] Kučera, R. *Numerické metody*. Učební texty. VB-TU Ostrava. (152 s). ISBN 80-248-1198-7.

[8] Materna, A. — těpánek, P. — Teplý, B. *Automatizace inenýrských úloh*. Skriptum. Vysoké učení technické v Brně, 1985. (132 s).

[9] *Matlab*. Programový systém pro provádění matematických výpočtů. Komerční software, verze R2017b. [on-line]. ¡http://www.mathworks.com¿. The MathWorks, prosinec 2017. (Citováno na s **??**.)

[10] Mika, S. *Numerické metody algebry*. Matematika pro vysoké koly technické. 2. vydání. SNTL - Nakladatelství technické literatury, Praha, 1985. (176 s). (Citováno na s **??**.)

[11] Okrouhlík, M. a kol. *Numerical methods in computational mechanics*. Elektronická publikace ve formátu PDF. [on-line]. ¡http://www.it.cas.cz/elektronicka-kniha-numerical-methods-computation-mechanics¿. Institute of Thermomechanics, Prague 2008. Last revision January 23, 2012.

[12] Olehla, M. — Tier, J. *Praktické pouití Fortranu*. 2. upravené vydání. Nakladatelství dopravy a spojů, Praha, 1979. (432 s). (Citováno na s **??** a **??**.)

[13] Ralston, A. *Základy numerické matematiky*. 1. vydání. Academia, Praha, 1973. (635 s).

[14] Rektorys, K. *Přehled uité matematiky*. 4. vydání. SNTL - Nakladatelství technické literatury, Praha, 1981. (1140 s).

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

[16] 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 přeloil Petr Kláterecký. (Citováno na s **??**.)

[17] *Wikipedia*. Otevřená internetová encyklopedie. Webové stránky. [on-line]. ¡http://cs.wikipedia.org¿. (Citováno na s **??**.)

[18] Wirth N. *Algoritmy a truktúry údajov*. 1. vydanie. Alfa, vydavatežstvo technickej a ekonomickej literatúry, Bratislava, 1988. (488 s). (Citováno na s **??**.)