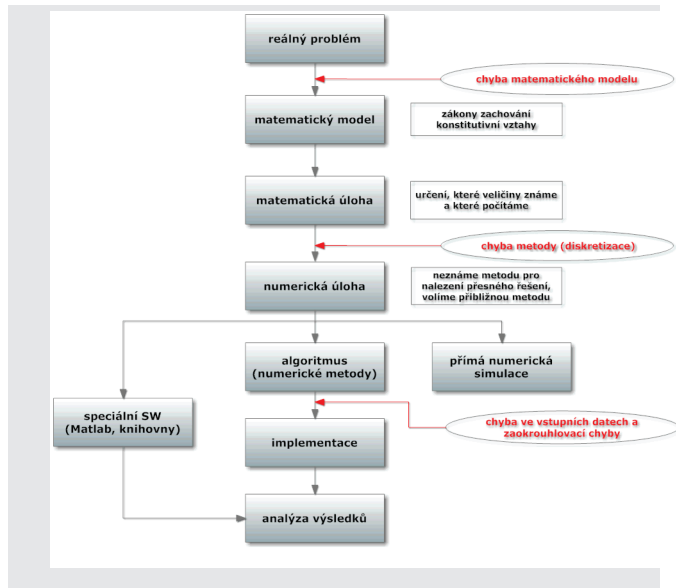


# Kapitola 1. Úvod do numerické matematiky

**Numerická matematika** = věda, která se zabývá řešením matematicky formulovaných úloh pomocí logických operací a aritmetických operací s číslly o konečné délce.



**Příklad**  
 Reálný problém ... intravenózní dávkování léku  
 Matematický model

- nezávisle proměnná je pouze čas  $t$
- šíření látky není závislé na prostorových proměnných

- popis pomocí diferenciální rovnice

$$\frac{dC}{dt} = -k \cdot C$$

kde  $C$  je koncentrace látky v krvi a  $k > 0$  je absorpční koeficient

- počáteční podmínka

$$C(0) = C_0$$

chyba matematického modelu odpovídá zjednodušujícím předpokladům

Matematická úloha

- chceme vypočítat hodnotu koncentrace látky v čase  $t \ll 0, T >$

Numerická úloha

- řešení hledáme pouze v konečně mnoha bodech (diskretizujeme čas,  $t_0 = 0, t_n = n \cdot \frac{T}{N}, t_N = T$ )  
 $N$  je počet dělení intervalu  $< 0, T >$

chyba diskretizace (metody)

Numerická metoda

- derivaci  $\frac{dC}{dt}$  aproximujeme poměrnou diferencí

$$\frac{C_{n+1} - C_n}{\frac{T}{N}} = -k \cdot C_n$$

chyba diskretizace (metody)

Výpočet

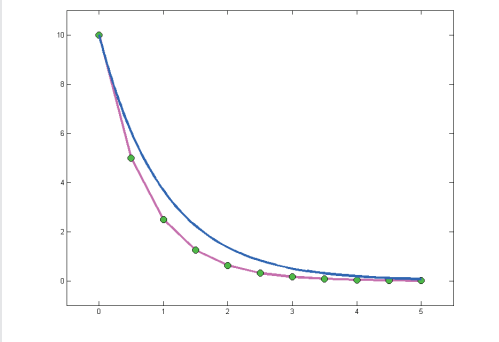
$$C_{n+1} = (1 - \frac{T}{N} \cdot k) \cdot C_n, \quad C_0 \text{ dáno}$$

zaokrouhlovací chyby

Analytické řešení

$$C(t) = C_0 \cdot e^{-kT}$$

např:  $C(0) = 10, k = 1, T = 5, N = 10$



Reálné hodnoty ???

## CHYBY

$x$  ... přesná hodnota  
 $\tilde{x}$  ... přibližná hodnota

**absolutní chyba** ...  $A(x) = |x - \tilde{x}| \leq a(x)$   
odhad

**relativní chyba** ...  $R(x) = \frac{A(x)}{|x|} \leq r(x)$   
odhad

**Pozn.:** Při odečítání „blízkých“ čísel roste relativní chyba (ztráta platných číslic)

$$a(x \pm y) = a(x) + a(y)$$

$$|(x \pm y) - (\tilde{x} \pm \tilde{y})| \leq |x - \tilde{x}| + |y - \tilde{y}|$$

$$r(x \pm y) = \frac{a(x) + a(y)}{|x \pm y|} \quad |x \pm y| \rightarrow 0, \quad !!!$$

**Pozn.:** Násobení a dělení nemohou podstatně zvětšit relativní chybu

$$a(x \cdot y) = |x| \cdot a(y) + |y| \cdot a(x)$$

$$|xy - \tilde{x}\tilde{y}| = |xy - \tilde{x}y + \tilde{x}y - \tilde{x}\tilde{y}| = |y(x - \tilde{x}) + \tilde{x}(y - \tilde{y})| \leq |y| \cdot |x - \tilde{x}| + |\tilde{x}| \cdot |y - \tilde{y}|$$

$$r(x \cdot y) = r(x) + r(y)$$

$$\frac{|x|a(y) + |y|a(x)}{|xy|} = \frac{a(y)}{|y|} + \frac{a(x)}{|x|}$$

$$a\left(\frac{x}{y}\right) = \frac{|x| \cdot a(y) + |y| \cdot a(x)}{y^2}$$

$$\left| \frac{x}{y} - \frac{\tilde{x}}{\tilde{y}} \right| = \left| \frac{1}{y\tilde{y}}(x\tilde{y} - \tilde{x}y) \right| = \left| \frac{1}{y\tilde{y}}(x\tilde{y} - xy + xy - \tilde{x}y) \right| = \left| \frac{1}{y\tilde{y}}(x(\tilde{y} - y) + y(x - \tilde{x})) \right| \leq \frac{1}{y^2}(|x| \cdot |y - \tilde{y}| + |y| \cdot |x - \tilde{x}|)$$

$$r\left(\frac{x}{y}\right) = r(x) + r(y)$$

$$\frac{|x|a(y) + |y|a(x)}{\frac{|x|}{|y|}} = \frac{a(y)}{|y|} + \frac{a(x)}{|x|}$$

**Definice:** Mějme dány dvě množiny  $X$  (vstupní data) a  $Y$  (výstupní data). Předpokládejme, že  $X, Y$  jsou Banachovy prostory. **Úlohou** rozumíme relaci

$$y = U(x), \quad x \in X, \quad y \in Y.$$

**Definice:** Řekneme, že úloha je **korektní** na dvojici prostorů  $(X, Y)$ , když

- $\forall x \in X \exists! y \in Y : y = U(x)$  (zobrazení),
- řešení  $y$  spojitě závisí na vstupních datech

$$\forall \{x_n\} : x_n \rightarrow x, \quad U(x_n) = y_n : y_n \rightarrow y = U(x).$$

**Poznámka:** Banachův prostor = úplný + normovaný

**úplný prostor:** metrický prostor, kde  $\forall$  Cauchyovská posl.  $u_n \subset X$  má limitu  $u \in X$

**normovaný prostor** = množina  $X$ :

a)  $X$  je lineární;

b)  $\forall u \in X \rightarrow \|u\|$ :

$$\|u\| \geq 0, \quad \|u\| = 0 \Leftrightarrow u = 0;$$

$$\|au\| = |a| \cdot \|u\| \quad \forall a \in \mathbb{R};$$

$$\|u+v\| \leq \|u\| + \|v\|;$$

c)  $d(u, v) = \|u - v\|$

**Poznámka:** Protože  $X, Y$  jsou Banachovy prostory, lze spojitost zaručit podmínkou

$$\|y_n - y\|_Y \leq L \|x_n - x\|_X.$$

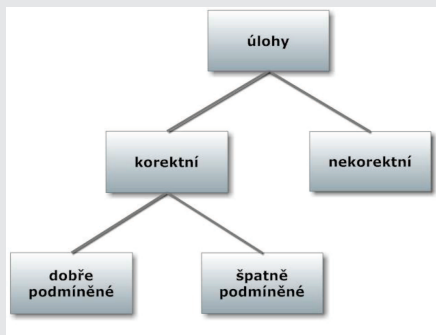
**Poznámka:** **Nekorektní** úlohy jsou úlohy, které nejsou korektní. Někdy je nekorektnost způsobena pouze nevhodnou formulací.

**Definice:** Úloha je **dobře podmíněná**, jestliže malá relativní změna ve vstupních datech vyvolá malou relativní změnu řešení.

**Číslo podmíněnosti** úlohy  $y = U(x)$

$$C_p = \frac{\left| \frac{\Delta y}{y} \right|}{\left| \frac{\Delta x}{x} \right|}$$

**Poznámka:** Je-li  $C_p \approx 1$  je úloha velmi dobře podmíněná. V praxi hovoříme o špatně podmíněné úloze pro  $C_p \gtrsim 100$ .



**Příklad 1**

Posuďte podmíněnost úlohy určit hodnotu funkce  $y = \sin(x)$

- a) v bodě 3,14;
- b) v bodě -0,01.

a) Volíme  $x = 3,14$ ,  $\Delta x = 0,01$  ← *malá změna na vstupu*

$$(y =) \sin x = \sin 3,14 = 0,0015926$$

$$(y + \Delta y =) \sin(x + \Delta x) = \sin 3,15 = -0,0084072$$

$$\Delta y = \sin(x + \Delta x) - \sin x = -0,0099998$$
 ← *změna na výstupu*

Relativní chyba na vstupu:  $\frac{|\Delta x|}{|x|} \approx 0,0031847$

Relativní chyba na výstupu:  $\frac{|\Delta y|}{|y|} \approx 6,2789149$

$$C_p \approx 1971,6 \rightarrow \text{špatně podmíněná úloha}$$

b) Volíme  $x = -0,01$ ,  $\Delta x = 0,01$

$$\sin x = -0,0099998$$

$$\sin(x + \Delta x) = \sin 0 = 0$$

$$\Delta y = 0,0999998$$

Relativní chyba na vstupu:  $\frac{|\Delta x|}{|x|} \approx 1$

Relativní chyba na výstupu:  $\frac{|\Delta y|}{|y|} \approx 1$

$$C_p \approx 1 \rightarrow \text{velmi dobře podmíněná úloha}$$

**Poznámka:** Podívejme se na předchozí příklad obecněji. Úloha má tvar  $y = f(x)$ .

Podle věty o střední hodnotě platí:

$$|\Delta y| \approx |f'(x)| \cdot |\Delta x|$$

odtud:

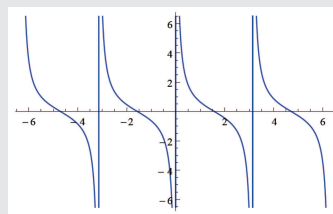
$$\left| \frac{\Delta y}{y} \right| \approx \frac{|f'(x)| \cdot |\Delta x|}{|f(x)|} = \left| \frac{x \cdot f'(x)}{f(x)} \right| \cdot \left| \frac{\Delta x}{x} \right|$$

Tedy

$$C_p \approx \left| \frac{x \cdot f'(x)}{f(x)} \right|$$

v našem případě:  $y = \sin x \Rightarrow y' = \cos x$

$$C_p \approx \left| \frac{x \cos x}{\sin x} \right| = |x \cotg x|$$



$$\lim_{x \rightarrow \pm\pi/2} x \cotg x = \pm\infty$$

$$\lim_{x \rightarrow 0} x \frac{\cos x}{\sin x} = \cos 0 \cdot \lim_{x \rightarrow 0} \frac{x}{\sin x} = 1 \cdot 1 = 1$$

**Poznámka:** Podobné příklady (posuďte podmíněnost úlohy určit hodnotu):

a)  $f(x) = x^\alpha, x \rightarrow 0, x > 0$

$$C_p = \left| \frac{x f'(x)}{f(x)} \right| = \left| \frac{x \alpha x^{\alpha-1}}{x^\alpha} \right| = \alpha$$

b)  $f(x) = \arcsin x, x \rightarrow 1, x < 1$

$$C_p = \left| \frac{x f'(x)}{f(x)} \right| = \left| \frac{x \frac{1}{\sqrt{1-x^2}}}{\arcsin x} \right| = \left| \frac{1}{\sqrt{1-x^2}} \left( \frac{x}{\arcsin x} \right)^{-1} \right| \xrightarrow{x \rightarrow 1} \infty$$

c)  $f(x) = x-1, x \rightarrow 1$

$$C_p = \left| \frac{x \cdot 1}{x-1} \right| \rightarrow \infty$$

**Příklad 2**

Posuďte podmíněnost úlohy řešit soustavu lineárních algebraických rovnic (pro  $\alpha \neq \pm 1$ )

$$x + \alpha y = 1$$

$$\alpha x + y = 0$$

$$x(1 - \alpha^2) = 1$$

$$x = \frac{1}{1 - \alpha^2}$$

$$y = -\frac{\alpha}{1 - \alpha^2}$$

Nechť vstup je hodnota  $\alpha$  a výstup hodnota  $x$ .

Pak

$$C_p = \frac{\left| \frac{\Delta x}{x} \right|}{\left| \frac{\Delta \alpha}{\alpha} \right|} \approx \frac{\left| \frac{\alpha \frac{dx}{d\alpha}}{x} \right|}{\left| \frac{\alpha \frac{dx}{d\alpha}}{x} \right|} = \frac{\alpha \frac{2\alpha}{(1-\alpha^2)^2}}{\frac{1}{1-\alpha^2}} = \frac{2\alpha^2}{1-\alpha^2}$$

$\Rightarrow$  pro  $\alpha^2 \rightarrow 1$  je tato úloha **špatně podmíněná!**

\* viz předchozí poznámka

\*\*  $\frac{dx}{d\alpha} = \frac{d}{d\alpha} \left( \frac{1}{1-\alpha^2} \right) = - \left( \frac{1}{(1-\alpha^2)^2} (-2\alpha) \right)$

**Pozn.:** Matice výše uvedené soustavy je pro hodnoty  $\alpha$  blízké  $\pm 1$  skoro singulární.

„U nestabilní metody (algoritmu) se relativně malé chyby v jednotlivých krocích výpočtu postupně akumulují tak, že dojde ke katastrofální ztrátě přesnosti numerického řešení úlohy.“  
- Při výpočtu dochází k zaokrouhlovacím chybám. Je proto vhodné vybírat algoritmy málo citlivé na zaokrouhlovací chyby.

**Stabilní algoritmus**

- **dobře podmíněný** - málo citlivý na poruchy ve vstupních datech
- **numericky stabilní** - málo citlivý na vliv zaokrouhlovacích chyb

**Poznámka:**

U stabilních metod roste chyba výsledku s počtem kroků  $N$  nejvýše lineárně (v ideálním případě, kdy je znaménko chyby náhodné, zaokrouhlovací chyba roste  $\sim \sqrt{N}$ ).

U nestabilních metod roste zaokrouhlovací chyba rychleji, např. geometrickou řadou  $\sim q^N$ , kde  $|q| > 1$ .

**Příklad 3**

Řešte diferenční rovnici (rekurentní formule, nestabilní rekurze)

$$x_{n+1} = \frac{13}{3}x_n - \frac{4}{3}x_{n-1}, \quad x_0 = 1, \quad x_1 = \frac{1}{3}$$

Snadno se ukáže, že řešení je  $x_n = \frac{1}{3^n}$  (dosazením).

Při numerickém výpočtu dojdeme k problémům (viz obr). Hodnoty  $x_n$  začínou velmi rychle klesat. Pro vysvětlení ukážeme obecné řešení zadané diferenční rovnice.

- charakteristický polynom

$$\lambda^2 = \frac{13}{3}\lambda - \frac{4}{3}$$

(předpokládáme řešení  $\lambda^n$ :  $\lambda^{n+1} = \frac{13}{3}\lambda^n - \frac{4}{3}\lambda^{n-1}$ )

- kořeny

$$\lambda_{1,2} = \frac{\frac{13}{3} \pm \sqrt{\left(\frac{13}{3}\right)^2 - 4 \cdot \frac{4}{3}}}{2} = \frac{\frac{13}{3} \pm \sqrt{\frac{121}{9}}}{2}, \quad \text{tj. } \lambda_1 = \frac{1}{3}, \lambda_2 = 4$$

- obecné řešení

$$x_n = A \cdot \left(\frac{1}{3}\right)^n + B \cdot 4^n$$

$$x_0 = 1 = A \cdot \left(\frac{1}{3}\right)^0 + B \cdot 4^0 = A + B = 1$$

$$x_1 = \frac{1}{3} = A \cdot \left(\frac{1}{3}\right)^1 + B \cdot 4^1 = \frac{1}{3}A + 4B = \frac{1}{3}$$

$$\Rightarrow A = 1, B = 0$$

Přes počáteční podmínku  $B = 0$  vzniknou vlivem zaokrouhlovacích chyb malé druhé komponenty řešení

Výsledky z MATLABU, FORMAT SHORT, pevná čárka na 5 číslic

```

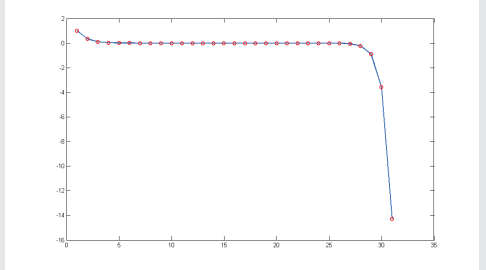
clc;
clear;
format short;

n=30;

x(1)=1;
x(2)=1/3;

for i=2:n
    x(i+1)=13/3*x(i)-4/3*x(i-1);
end

plot(1:n+1,x,'b-',1:n+1,x,'ro');
    
```



**Příklad 4**

Vypočítejte přibližně hodnotu

$$J_n = \int_0^1 \frac{x^n}{x+5} dx$$

Platí:

$$\int_0^1 x^{n-1} dx = \int_0^1 \frac{x^{n-1}(x+5)}{x+5} dx = \int_0^1 \frac{x^n}{x+5} dx + 5 \int_0^1 \frac{x^{n-1}}{x+5} dx$$

$$\left[ \frac{1}{n} x^n \right]_0^1 = \frac{1}{n}$$

Dále:

$$J_0 = \int_0^1 \frac{1}{x+5} dx = \ln|x+5| \Big|_0^1 = \ln \frac{6}{5}$$

Rekurentní formule:

$$J_0 = \ln \frac{6}{5}$$

$$J_n = -5 \cdot J_{n-1} + \frac{1}{n}$$

Nestabilní algoritmus! ... vždy  $\exists n_0: J_{n_0} < 0!$

Proto je lépe postupovat odzadu:

• dokážeme, že  $\lim_{n \rightarrow \infty} J_n = 0$

$$|J_n| = \left| \int_0^1 \frac{x^n}{x+5} dx \right| \leq \int_0^1 \left| \frac{x^n}{x+5} \right| dx \leq \frac{1}{5} \int_0^1 x^n dx = \frac{1}{5(n+1)} \rightarrow_{n \rightarrow \infty} 0$$

• např. zvolíme  $J_{100} = 0$  a počítáme  $J_{n-1} = -\frac{1}{5}(J_n - \frac{1}{n})$

$$J_{100} = 0$$

$$J_{n-1} = -\frac{1}{5} \cdot J_n + \frac{1}{5n}$$

Výsledky z MATLABu, FORMAT SHORT E

```

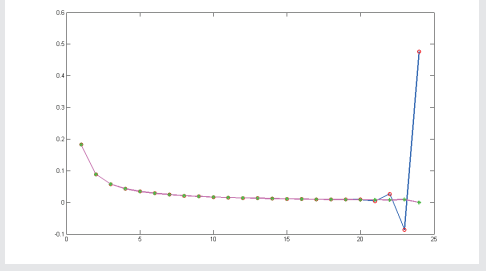
clc;
clear;
format short e;

n=24;

J(1)=log(6/5);
JJ(n)=0;

for i=1:n-1
    J(i+1) = -5*J(i) + 1/i;
end
for i=n-1:-1:1
    JJ(i) = (1/i - JJ(i+1)) / 5;
end

[J' JJ']
plot(1:n,J,'b-',1:n,J,'ro');
hold on
plot(1:n,JJ,'m-',1:n,JJ,'g*');
    
```



J	JJ
1.8232e-001	1.8232e-001
8.8392e-002	8.8392e-002
5.8039e-002	5.8039e-002
4.3139e-002	4.3139e-002
3.4306e-002	3.4306e-002
2.8468e-002	2.8468e-002
2.4325e-002	2.4325e-002
2.1233e-002	2.1233e-002
1.8837e-002	1.8837e-002
1.6926e-002	1.6926e-002
1.5368e-002	1.5368e-002
1.4071e-002	1.4071e-002
1.2977e-002	1.2977e-002
1.2040e-002	1.2040e-002
1.1229e-002	1.1229e-002
1.0522e-002	1.0521e-002
9.8903e-003	9.8964e-003
9.3719e-003	9.3414e-003
8.6960e-003	8.8485e-003
9.1515e-003	8.3893e-003
4.2426e-003	8.0535e-003
2.6406e-002	7.3518e-003
-8.6575e-002	8.6957e-003
4.7635e-001	0

**Zobrazení čísel**

Motivace:  $\sum_{k=1}^{10000} \frac{1}{10^k} = 9998,55664$

- Lidé používají desítkovou soustavu.
- Počítače dvojkovou.

Komunikace s počítačem

- Zadáni v 10-soustavě.
- Převod do 2-soustavy (počítač).
- Výpočet (počítač).
- Zpětný převod do 10-soustavy (počítač).
- Výsledek v 10-soustavě.

Soustavy

• desítková

$$1563 = (1 \cdot 10^3) + (5 \cdot 10^2) + (6 \cdot 10^1) + (3 \cdot 10^0)$$

obecně

$$N = (a_k \cdot 10^k) + (a_{k-1} \cdot 10^{k-1}) + \dots + (a_1 \cdot 10^1) + (a_0 \cdot 10^0)$$

$$(N \in \mathbb{N}, a_k \in \{0, 1, 2, \dots, 9\})$$

značení

$$N = a_k a_{k-1} a_{k-2} \dots a_1 a_0$$

• dvojková

$$1563 = (1 \cdot 2^{10}) + (1 \cdot 2^9) + (0 \cdot 2^8) + (0 \cdot 2^7) + (0 \cdot 2^6) + (0 \cdot 2^5) + (1 \cdot 2^4) + (1 \cdot 2^3) + (0 \cdot 2^2) + (1 \cdot 2^1) + (1 \cdot 2^0)$$

$$(1563)_{10} = (11000011011)_2$$

**Binární zlomky**

Ize vyjádřit jako sumu se zápornými mocninami dvou

$$R \in \mathbb{R} \quad 0 < R < 1 \quad d_j \in \{0, 1\}$$

$$R = (d_1 \cdot 2^{-1}) + (d_2 \cdot 2^{-2}) + \dots + (d_n \cdot 2^{-n}) + \dots$$

$$R = (0, d_1 d_2 \dots d_n \dots)_2$$

Zápis čísel

- V desítkové soustavě (vědecká notace)

$$0,000747 = 7,47 \cdot 10^{-4}$$

$$313,815 = 3,13815 \cdot 10^2$$

- Strojová čísla

normalizovaná pohyblivá řádová čárka (REAL)

$$x = \pm q \cdot 2^n \quad \frac{1}{2} \leq q < 1 \dots \text{mantisa}, \quad n \dots \text{exponent}$$

Poznámka: Mnoho reálných čísel, které lze v desítkové soustavě zapsat pomocí konečného počtu cifer, pro zápis ve dvojkové soustavě vyžaduje nekonečně mnoho cifer.

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$$(0,7)_{10} = (0,10110)_{2} = 1 \cdot 2^{-1} + \sum_{k=0}^{\infty} 1 \cdot 2^{-(3+4k)} + \sum_{k=0}^{\infty} 1 \cdot 2^{-(4+4k)} =$$

$$= 2^{-1} + 2^{-3} \cdot \sum_{k=0}^{\infty} (2^{-4})^k + 2^{-4} \cdot \sum_{k=0}^{\infty} (2^{-4})^k = \frac{1}{2} + \frac{1}{8} \cdot \frac{1}{1-\frac{1}{16}} + \frac{1}{16} \cdot \frac{1}{1-\frac{1}{16}} =$$

$$= \frac{1}{2} + \frac{2}{15} + \frac{1}{15} = \frac{15+4+2}{30} = \frac{21}{30} = \frac{7}{10}$$

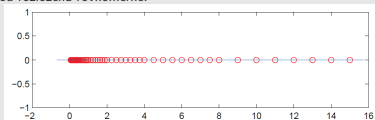
**Příklad:**

Sestrojte všechna strojová čísla s mantisou délky 4 a exponentem v rozsahu od -3 do 4, tj.  
 $x = q \cdot 2^n$ , kde  $q = 0, d_1 d_2 d_3 d_4$ ,  $n \in \{-3, -2, -1, 0, 1, 2, 3, 4\}$

Abychom si lépe uvědomili jakou mantisou a jakým exponentem je určeno získané číslo, uvedeme si je v následující tabulce.

q \ n	-3	-2	-1	0	1	2	3	4
0.1000 <sub>2</sub>	0,0625	0,125	0,25	0,5	1	2	4	8
0.1001 <sub>2</sub>	0,0703125	0,140625	0,28125	0,5625	1,125	2,25	4,5	9
0.1010 <sub>2</sub>	0,078125	0,15625	0,3125	0,625	1,25	2,5	5	10
0.1011 <sub>2</sub>	0,0859375	0,171875	0,34375	0,6875	1,375	2,75	5,5	11
0.1100 <sub>2</sub>	0,09375	0,1875	0,375	0,75	1,5	3	6	12
0.1101 <sub>2</sub>	0,1015625	0,203125	0,40625	0,8125	1,625	3,25	6,5	13
0.1110 <sub>2</sub>	0,109375	0,21875	0,4375	0,875	1,75	3,5	7	14
0.1111 <sub>2</sub>	0,1171875	0,234375	0,46875	0,9375	1,875	3,75	7,5	15

Získaná čísla si je také vhodné vykreslit na číselnou osu, získáme tak přehled o jejich rozložení. Snadno zjistíme, že čísla nejsou rozložena rovnoměrně.



pomocná funkce v MATLABu

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

```
function [A,P]=stroj_cisla(cisel_mantis,exponent,zobraz);
%
% [A,P]=stroj_cisla(4,-3:4,1);

for i=1:length(exponent)
for j=0:2^(cisel_mantis)-1
zaklad=dec2bin(j);
zakladstr=num2str(zaklad);
for k=1:cisel_mantis-length(zakladstr)-1
zakladstr=strcat('0',zakladstr);
end;
zakladstr=strcat('1',zakladstr);
zaklad=bin2dec(zakladstr)*2^(-cisel_mantis);
A(j+1,i)=zaklad*2^exponent(i);
end;
end;

[k,1]=size(A);
P=sort(reshape(A,1,k*1));

if zobraz==1
figure(1);
plot(P,zeros(size(P),'ro');
pr=(P(k*1)-P(1))/20;
hold on;
plot([P(1)-pr,P(k*1)+pr],[0 0] ,'b-');
end;

format short g;
```

**Příklad 5:**

Uvažujme množinu strojových čísel vygenerovanou v předchozím příkladu (tj. strojová čísla s mantisou délky 4 a exponentem v rozsahu od -3 do 4). Předpokládáme, že počítač zobrazí číslo na nejbližší číslo, které lze zobrazit, v případě shody na větší.

Ukažme si, jak se v tomto stroji sečtou čísla  $\frac{1}{10}$  a  $\frac{1}{5}$ .

Výsledky získané z MATLABu

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

Zobrazení součtu čísel A a B v dané množině strojových čísel s mantisou délky M a exponentem v rozsahu od Exp\_min do Exp\_max

Cislo A = 0.100000  
Cislo B = 0.200000  
Pocet cisel mantisy M = 4  
Rozsah pro exponent: od -3 do 4

cislo	zapis obrazu	obraz
A = 0.100000	0.1101 x 2 <sup>-3</sup>	0.1015625
B = 0.200000	0.1101 x 2 <sup>-2</sup>	0.203125
obrazA+obrazB=		
0.3046875	0.1010 x 2 <sup>-1</sup>	0.3125
A+B= 0.300000	0.1010 x 2 <sup>-1</sup>	0.3125

**Poznámka:**  
V tomto příkladě se shodoval obraz přesného výsledku s obrazem součtu obrazů jednotlivých sčítanců.

**Příklad 6:**  
Uvažujme množinu strojových čísel vygenerovanou v předchozím příkladu (tj. strojová čísla s mantisou délky 4 a exponentem v rozsahu od -3 do 4). Předpokládáme, že počítač zobrazí číslo na nejbližší číslo, které lze zobrazit, v případě shody na větší.  
Ukažme si, jak se v tomto stroji sečtou čísla  $\frac{3}{10}$  a  $\frac{1}{6}$ .

Výsledky získané z MATLABu

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

Zobrazení součtu čísel A a B v dané množině strojových čísel s mantisou délky M a exponentem v rozsahu od Exp\_min do Exp\_max

Cislo A = 0.300000  
Cislo B = 0.166667  
Pocet cisel mantisy M = 4  
Rozsah pro exponent: od -3 do 4

cislo	zapis obrazu	obraz
A = 0.300000	0.1010 x 2 <sup>-1</sup>	0.3125
B = 0.166667	0.1011 x 2 <sup>-2</sup>	0.171875
obrazA+obrazB=		
0.484375	0.1000 x 2 <sup>0</sup>	0.5
A+B= 0.466667	0.1011 x 2 <sup>0</sup>	0.46875

**Poznámka:**  
V tomto příkladě se obraz přesného výsledku s obrazem součtu obrazů jednotlivých sčítanců neshodoval!

Chyba výpočtu:  
 $\frac{7}{15} - 0,1000_2 \cdot 2^0 = \frac{14-15}{30} = -\frac{1}{30} = -0,0\bar{3}$

Relativně:  
 $\frac{\frac{1}{30}}{\frac{7}{15}} = \frac{1}{14} = 7,14\% \quad !!!$

**Přesnost počítače**

- Vymežeme-li pro mantisu 24 bitů, získáme 7 desetinných míst ( $2^{24} = 16777216$ ).
- Vymežeme-li pro mantisu 32 bitů, získáme 9 desetinných míst ( $2^{32} = 4294967296$ ).

Základní formáty:

Formát	Bytes	Bitů pro mantisu	Bitů pro exponent
Single	4	24	8
Double	8	53	11

**Příklad:**

- Uvažujme formát SINGLE, tj. 24 bitů pro mantisu.

$$\frac{1}{10} = 0,00011_2 \approx 0,11001100110011001100_2 \cdot 2^{-3}$$

$$\text{Chyba zobrazení je } 0,1100_2 \cdot 2^{-27} (= \frac{1}{10} \cdot 2^{-24}) \approx 5,96 \cdot 10^{-9}$$

- Máme-li počítat  $\sum_{k=1}^{10000} \frac{1}{10^k}$  dostaneme ve formátu SINGLE 9,998,55664.

Chyba musí být větší než  $100000 \cdot 5,96 \cdot 10^{-9} = 5,96 \cdot 10^{-4}$ .

Ve skutečnosti je chyba ještě větší, neboť se v průběhu výpočtu musí částečné suma zaokrouhlovat dolů nebo nahoru, jak suma roste, později přičítaná čísla  $\frac{1}{10^k}$  jsou oproti sumě menší a jsou tedy počítány s menší přesností (viz následující příklad).

Příklad: Ve formátu SINGLE sečtete čísla 10000 a 0,1.

```
-----
Prevod čísla 10000 z 10-soustavy do 2-soustavy na 0 desetinných míst
Čela část ..... 10000
Desetinná část ..... 0.000000
-----
prevod_cele_casti =

10000 : 2 = 5000 : 2 = 2500 : 2 = 1250 : 2 = 625 : 2 = 312 : 2 =
 0         0         0         0         1         0

= 156 : 2 = 78 : 2 = 39 : 2 = 19 : 2 = 9 : 2 = 4 : 2 = 2 : 2 = 1
 0         0         1         1         1         0         0

Číslo 10000 v 10-soustave prevedeno do 2-soustavy je 10011100010000.
```

$$(10000)_{10} = (10011100010000)_2 = 0,100111001 \cdot 2^{14}$$

$$(2^{14} = 16384)$$

$$10000 \dots 0,100111001000000000000000 \cdot 2^{14}$$

$$0,1 \dots 0,110011001100110011001100 \cdot 2^{-3}$$

$$0,1 \text{ po SHIFTu} \dots 0,000000000000000011100110 \cdot 2^{14}$$

$$= (01100110)_2 \cdot 2^{-24} \cdot 2^{14} = (64 + 32 + 4 + 2) \cdot 2^{-10} =$$

$$= \frac{102}{1024} = 0,099609375$$

$$10000 + 0,1 \dots 0,100111001000000011100110 \cdot 2^{14}$$

Číslo 10000 je zobrazeno přesně.  
Chyba zobrazení 0,1 po SHIFTu je  $\frac{1}{10} - \frac{102}{1024} = 0,1 - 0,099609375 = 3,90625 \cdot 10^{-4}$

Shrnutí:  
 $10000 + 0,1 \rightarrow$  výsledek s chybou  $3,90625 \cdot 10^{-4}$   
(v sumě z motivačního příkladu jde o jeden krok)

skript v MATLABu

```
s=0;
h=single(1/10);

for i=1:100000
    s=s+h;
end;

s
```

## Kapitola 2. Nelineární rovnice

### Formulace:

Je dána funkce  $f: \mathbb{R} \rightarrow \mathbb{R}$  definovaná na intervalu  $\langle a, b \rangle$ . Hledáme  $x \in \langle a, b \rangle$  tak, aby  $f(x) = 0$ .  
( $x \dots$  kořen rovnice)

### Poznámka:

Najít přesné řešení analyticky je možné jen ve velmi jednoduchých případech, např. při řešení lineární rovnice  $12x - 3 = 0$ , při řešení kvadratické rovnice  $4x^2 - 5x + 8 = 0$  nebo např. při řešení rovnice  $\sin 5x = \pi$ . Proto je nutné pro nalezení kořenů použít nějakou numerickou metodu.

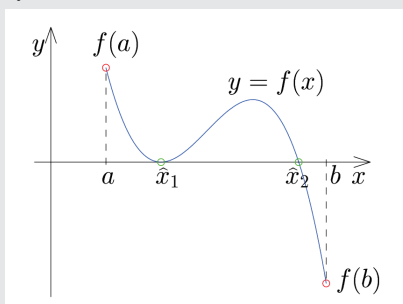
Numerické metody, kterými se budeme zabývat jsou založeny na **iteračních principech**. Pro každou iterační metodu nás budou zajímat odpovědi na dvě otázky:

- Konverguje posloupnost iterací ke hledanému kořenu?
- Jestliže ano, jak rychle?

### Věta:

Předpokládejme, že  
(i) reálná funkce  $f$  je spojitá pro  $x \in \langle a, b \rangle$ ,  
(ii)  $f(a), f(b) < 0$ .  
Potom existuje aspoň jedno řešení  $x$  rovnice  $f(x) = 0$  na  $\langle a, b \rangle$ .

Větu ilustruje následující obrázek.



### Startovací metody

- metoda půlení intervalu

- regula falsi

- metoda prosté iterace

### Zpřesňující metody

- Newtonova metoda
- metoda sečen
- Mullerova metoda

### Metoda prosté iterace

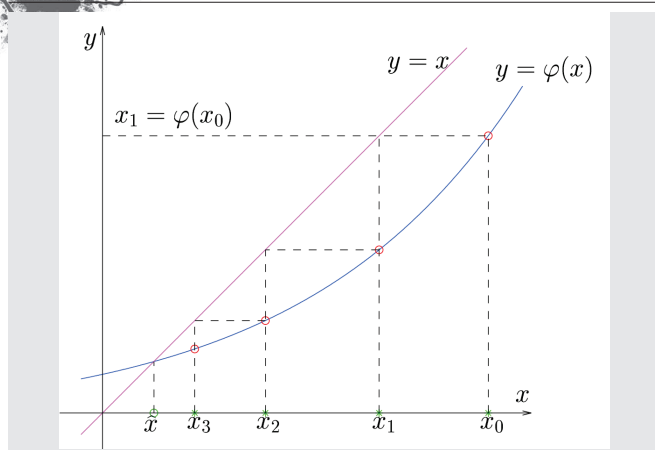
Všechny (jednobodové) iterační metody lze pokládat za speciální případ této metody.

Princip:

- původní rovnici  $f(x) = 0$  přepíšeme na tvar  $x = \varphi(x)$
- existuje celá řada možností, jak to udělat!
- na konkrétní volbě funkce  $\varphi$  závisí *konvergence metody* a *rychlost konvergence*

Algoritmus:

- Zadáme  $x_0 \in \langle a, b \rangle$ ,  $\varepsilon > 0$
- $x_{k+1} = \varphi(x_k)$
- Je-li  $|x_{k+1} - x_k| < \varepsilon$ , pak  $x = x_{k+1}$ , KONEC  
jinak jdi na 2)



**Příklad 1**

Metodou prosté iterace najděte na intervalu (1,4) řešení rovnice

$$x^2 + \ln x - \frac{10}{x} = 0.$$

Za počáteční iteraci volte střed zadaného intervalu, tj.  $x_0 = 2,5$ .

**Řešení**

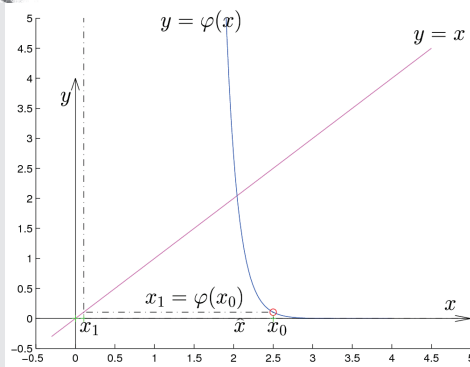
Ukážeme si 4 způsoby přepisu rovnice  $f(x) = 0$  na tvar  $x = \varphi(x)$ .

1. způsob:

$$\ln x = \frac{10}{x} - x^2 \Rightarrow x = e^{\left(\frac{10}{x} - x^2\right)} \text{ tj. } \varphi(x) = e^{\left(\frac{10}{x} - x^2\right)}$$

k	$x_k$
0	2.5
1	0.1054
2	$1.5845 \cdot 10^{41}$

Již první iterace  $x_1$  je mimo zadaný interval, navíc druhá iterace  $x_2$  je velmi velké číslo a proto metoda prosté iterace nekonverguje.

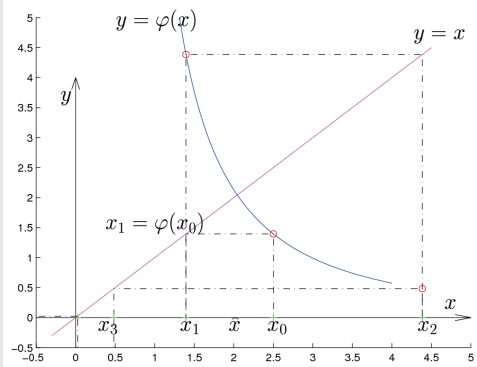


2. způsob:

$$x^2 + \ln x = \frac{10}{x} \Rightarrow x = \frac{10}{x^2 + \ln x} \text{ tj. } \varphi(x) = \frac{10}{x^2 + \ln x}$$

k	$x_k$
0	2.5
1	1.3954
2	4.3852
3	0.4829
4	-20.2122

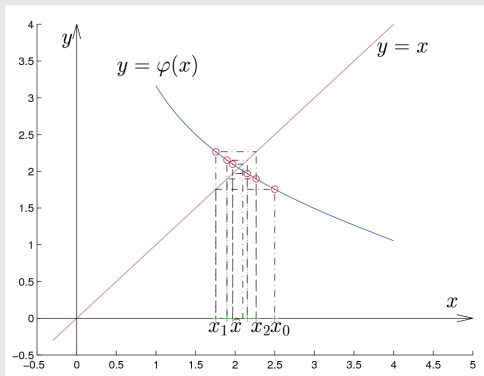
Podobně jako v předchozím případě, zde je 2. iterace  $x_2$  mimo zadaný interval a metoda prosté iterace opět nekonverguje.



3. způsob:

$$x^2 + x \ln x - 10 = 0 \Rightarrow x = \sqrt{\frac{10}{x} - \ln x} \text{ tj. } \varphi(x) = \sqrt{\frac{10}{x} - \ln x}$$

k	$x_k$
0	2.5
1	1.7560
2	2.2653
3	1.8965
4	2.1524
5	1.9696
6	2.0974
7	2.0067
8	2.0704
9	2.0254
10	2.0571
11	2.0347
12	2.0505
13	2.0393
14	2.0472
15	2.0416
16	2.0455
17	2.0428
18	2.0447
19	2.0434

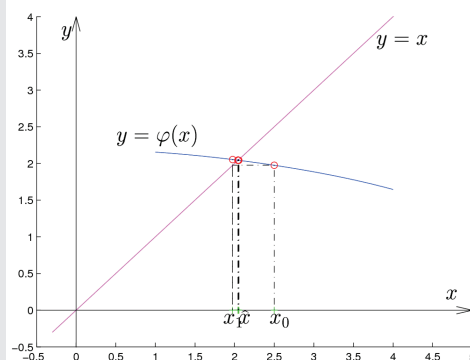


V tomto případě metoda prosté iterace konvergovala k výsledku  $\hat{x}$ , rychlost "zahušování" byla ovšem malá.

4. způsob:

$$x^2 + x \ln x - 10 = 0 \Rightarrow x = \sqrt[3]{10 - x \ln x} \text{ tj. } \varphi(x) = \sqrt[3]{10 - x \ln x}$$

k	$x_k$
0	2.5
1	1.9755
2	2.0532
3	2.0427
4	2.0441
5	2.0439



V tomto posledním případě metoda prosté iterace konvergovala k výsledku  $\hat{x}$  velmi rychle. To dokazuje ten fakt, že kdybychom použili pro zastavení podmínku, aby absolutní hodnota rozdílu dvou po sobě jdoucích iterací byla menší než  $10^{-10}$  potřebovali bychom k tomu pouze 10 iterací.

**Poznámka:**

Chování metody prosté iterace je závislé na zvoleném předpisu pro funkci  $\varphi = \varphi(x)$ . Porovnáním grafů z předchozího příkladu lze usoudit, že je vhodné, aby se funkce  $\varphi(x)$  co nejvíce blížila konstantní funkci.

**Věta** (Postačující podmínky konvergence metody prosté iterace.)

Předpokládejme, že je funkce  $\varphi$  na intervalu  $I = (a, b)$  spojitá a platí:

- (a)  $\forall x \in I : \varphi(x) \in I$  (funkce  $\varphi$  zobrazuje  $I$  do sebe),
- (b)  $\exists q \in (0, 1) : |\varphi(x) - \varphi(y)| \leq q|x - y| \quad \forall x, y \in I$  (funkce  $\varphi$  je kontrakce).

Potom

- 1) v intervalu  $I$  existuje právě jeden kořen  $\alpha$  rovnice  $x = \varphi(x)$ ,
- 2) posloupnost  $\{x_k\}_{k=1}^{\infty}$  určená formulí  $x_k = \varphi(x_{k-1})$  konverguje pro každé  $x_0 \in I$  a  $\lim_{k \rightarrow \infty} x_k = \alpha$ .

**Důkaz**

- 1) **existence** je důsledkem podmínky (a) a spojitosti  $\varphi$

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

(a)

$\Rightarrow \varphi(x) \geq a, \varphi(x) \leq b \quad \forall x \in (a, b)$   
 $\Rightarrow$  musí platit  
 $\varphi(a) \geq a, \varphi(b) \leq b$   
 $\varphi(a) \leq b, \varphi(b) \geq a$

**Jednoznačnost** plyne z vlastnosti (b)

**DK sporem:** Předpokládejme, že existují 2 různé hodnoty  $\alpha_1 \neq \alpha_2$  takové, že  
 $\alpha_1 = \varphi(\alpha_1), \alpha_2 = \varphi(\alpha_2)$

Potom platí:

$$|\alpha_2 - \alpha_1| = |\varphi(\alpha_2) - \varphi(\alpha_1)| \stackrel{(b)}{\leq} q \cdot |\alpha_2 - \alpha_1| \stackrel{\text{spor}}{\leq} |\alpha_2 - \alpha_1|$$


---

2) Platí:

$$\frac{x_k = \varphi(x_{k-1})}{\alpha = \varphi(\alpha)}$$

po odečtení:

$$x_k - \alpha = \varphi(x_{k-1}) - \varphi(\alpha)$$

$$|x_k - \alpha| = |\varphi(x_{k-1}) - \varphi(\alpha)| \stackrel{(b)}{\leq} q \cdot |x_{k-1} - \alpha| \quad (*)$$

(\*)  $\Rightarrow$

$$|x_1 - \alpha| \leq q \cdot |x_0 - \alpha|$$

$$|x_2 - \alpha| \leq q \cdot |x_1 - \alpha| \leq q^2 \cdot |x_0 - \alpha|$$

$$\dots$$

$|x_k - \alpha| \leq \underbrace{q^k}_{(**)} \underbrace{|x_0 - \alpha|}_{\text{konst}} \quad \forall x_0 \in (a, b)$

(\*\*)  $q^k \rightarrow 0$  pro  $k \rightarrow \infty$  ( $|q| < 1$ )

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

**Poznámka:**  
Podívejte se na souvislost předpokladů předchozí věty a volby funkce  $\varphi$  v jednotlivých případech příkladu 1.

**Poznámka:**  
Pro diferencovatelnou funkci  $\varphi$  lze podmínku (b) nahradit podmínkou  
 $(b') \exists q \in (0, 1) : |\varphi'(x)| \leq q \quad \forall x \in I$

**Poznámka:**  
Rychlost konvergence metody prosté iterace je charakterizována  $|\varphi'(x_k)|$ , jelikož lze psát

$$\varphi'(x_k) \approx \frac{\varphi(x_{k+1}) - \varphi(x_k)}{x_{k+1} - x_k} = \frac{x_{k+2} - x_{k+1}}{x_{k+1} - x_k}$$

**Poznámka:**  
Souvislost "zahušťování iterací" a hodnotě  $q$

a)  $q \approx 1 \Rightarrow |x_k - x_{k-1}| \approx |x_{k-1} - x_{k-2}|$

b)  $q \approx 0 \Rightarrow |x_k - x_{k-1}| \ll |x_{k-1} - x_{k-2}|$

**Poznámka:**  
Jak bylo řečeno hodnota rozdílu dvou po sobě jdoucích iterací neodpovídá obecně chybě přibližného řešení. Geometricky si to lze představit takto:

**Odhad chyby metody prosté iterace**

- Máme konvergentní proces  $x_k = \varphi(x_{k-1}), k = 1, 2, \dots$
- Přesné řešení  $\alpha$  splňuje vztah  $\alpha = \varphi(\alpha), \lim_{k \rightarrow \infty} x_k = \alpha$

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

- Po odečtení dostaneme  $x_k - \alpha = \varphi(x_{k-1}) - \varphi(\alpha)$   
tj.  $|x_k - \alpha| = |\varphi(x_{k-1}) - \varphi(\alpha)| \leftarrow$
- Předpokládáme, že  $\varphi$  je lipchitzovská s konstantou  $q \in (0, 1)$ , tj. musí platit:  
 $|\varphi(x_{k-1}) - \varphi(\alpha)| \leq q \cdot |x_{k-1} - \alpha| \leftarrow$
- Dále použijeme  $\Delta$  nerovnost:  
 $|x_{k-1} - \alpha| = |x_{k-1} - x_k + x_k - \alpha| \leq |x_{k-1} - x_k| + |x_k - \alpha| \leftarrow$
- Z posledních 3 vztahů dostaneme:  
 $|x_k - \alpha| \leq q \cdot |x_{k-1} - \alpha| + q \cdot |x_k - \alpha|$   
 $(1 - q) \cdot |x_k - \alpha| \leq q \cdot |x_{k-1} - \alpha| \quad / \cdot \frac{1}{1 - q} > 0$   

$|x_k - \alpha| \leq \frac{q}{1 - q} \cdot |x_{k-1} - \alpha|$
- Použijeme-li zastavovací podmínku  
 $|x_k - x_{k-1}| < \varepsilon$   
potom platí odhad chyby  

$|x_k - \alpha| \leq \frac{q}{1 - q} \varepsilon$

**Příklad 2**

Pomocí metody prosté iterace řešte na intervalu  $(0, 4)$  rovnici  
 $x - \sqrt{x+4} = 0$ .

---

*presné řešení:*

$$x = \sqrt{x+4} \quad /^2$$

$$x^2 = x+4$$

$$x^2 - x - 4 = 0$$

$$x_{1,2} = \frac{1 \pm \sqrt{17}}{2} \Rightarrow x_1 \approx 2,5615, x_2 \approx -1,5615 \notin (0,4)$$


---

- Rovnici přepíšeme na tvar:  $x = \sqrt{x+4} = \varphi(x)$
- Ověříme splnění předpokladů věty o postačujících podmínkách konvergence metody prosté iterace:  
(a)  $\forall x \in (0, 4) :$   
 $0 \leq \sqrt{x+4} \leq 4$   
 $0 \leq x+4 \leq 16$   
 $-4 \leq x \leq 12$

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

(b')  $\forall x \in (0, 4) :$

$$|\varphi'(x)| < 1$$

$$\left| \frac{1}{2\sqrt{x+4}} \right| < 1$$

$$\frac{1}{2\sqrt{x+4}} < 1$$

$$1 < 2\sqrt{x+4}$$

$$1 < 4x + 16$$

$$-15 < 4x$$

- Vlastní výpočet:  
volíme  $x_0 = 2$  a pro zastavovací podmínku hodnotu  $\varepsilon = 0,001$ .

k	$x_k$
0	2
1	2,4494
2	2,5395
3	2,5572
4	2,5607
5	2,5613

$\Rightarrow \tilde{x} = x_5 = 2,5613$ .

- Odhadněme velikost chyby přibližného řešení předchozího příkladu.  
Graf funkce  $\varphi(x)$ :

Platí:

$$\varphi' = \frac{1}{2\sqrt{x+4}} \dots \text{kladná klesající funkce}$$

$$(\varphi' = (\frac{1}{2}(x+4)^{-1/2})y = \frac{1}{4}(x+4)^{-3/2} = -\frac{1}{4(x+4)\sqrt{x+4}} < 0)$$

↓

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

$\max_{x \in I} |\varphi'(x)| = |\varphi'(0)| = \frac{1}{4} = q \dots$  podmínka (b')

Zvolili jsme  $\epsilon = 0,001$  a proto platí odhad chyby:

$|x_5 - \alpha| \leq \frac{1}{1 - \frac{1}{4}} \cdot 0,001 = 0,000333$

**Definice:** Říkáme, že posloupnost  $x_k$  **konverguje** k číslu  $\alpha$  rychlostí  $r$ , jestliže pro  $k \rightarrow \infty$

$$|x_{k+1} - \alpha| = c|x_k - \alpha|^r + O(|x_k - \alpha|^{r+1}),$$

Mluvíme o asymptotické rychlosti konvergence ( $k \rightarrow \infty$ ).

**Poznámka:**  
 $f(x) = O(g(x))$  pro  $x \rightarrow a \Leftrightarrow \left| \frac{f(x)}{g(x)} \right|$  je omezená (a nenulová) pro  $x \rightarrow a$ .

- Příklady:**
- a)  $x^5 \cdot \sin x = O(x^5)$  pro  $x \rightarrow \infty \Leftrightarrow \left| \frac{x^5 \cdot \sin x}{x^5} \right| = |\sin x| \leq 1$  pro  $x \rightarrow \infty$
- b)  $x^5 \cdot \sin x = O(x^6)$  pro  $x \rightarrow 0 \Leftrightarrow \left| \frac{x^5 \cdot \sin x}{x^6} \right| = \left| \frac{\sin x}{x} \right| \rightarrow 1$  pro  $x \rightarrow 0$

**Rychlost konvergence metody prosté iterace**

Je-li funkce  $\varphi$  dostatečně hladká, můžeme napsat její Taylorův rozvoj v bodě  $\alpha$  a potom pro  $x = x_{k-1}$  platí:

$$\varphi(x_{k-1}) = \varphi(\alpha) + \varphi'(\alpha)(x_{k-1} - \alpha) + \frac{\varphi''(\alpha)}{2}(x_{k-1} - \alpha)^2 + \frac{\varphi'''(\xi)}{6}(x_{k-1} - \alpha)^3$$

$$x_k - \alpha = \varphi'(\alpha)(x_{k-1} - \alpha) + \frac{\varphi''(\alpha)}{2}(x_{k-1} - \alpha)^2 + \frac{\varphi'''(\xi)}{6}(x_{k-1} - \alpha)^3$$

- je-li  $\varphi'(\alpha) \neq 0$ , potom  $x_k - \alpha = \varphi'(\alpha)(x_{k-1} - \alpha) + O((x_{k-1} - \alpha)^2)$   
 $\Rightarrow$  rychlost konvergence je řádu 1
- je-li  $\varphi'(\alpha) = 0$  a  $\varphi''(\alpha) \neq 0$ , potom  $x_k - \alpha = \frac{\varphi''(\alpha)}{2}(x_{k-1} - \alpha)^2 + O((x_{k-1} - \alpha)^3)$   
 $\Rightarrow$  rychlost konvergence je řádu 2

**Newtonova metoda**

**Předpoklady:**  
 Necht' v intervalu  $I = (a, b)$  leží jediný jednoduchý kořen  $\tilde{x}$  rovnice  $f(x) = 0$ . Jelikož mluvíme o zpřesňující

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

metodě, předpokládáme, že máme zadánu nultou iteraci  $x_0 \in I$ , která je relativně blízko hledanému řešení. Vyjádříme Taylorův rozvoj funkce  $f$  v bodě  $x_0$ . Přitom předpokládáme, že existují příslušné derivace funkce  $f$ .

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(\xi)(x - x_0)^2$$

Rovnici  $f(x) = 0$  nahradíme lineární rovnicí

$$f(x_0) + f'(x_0)(x - x_0) = 0$$

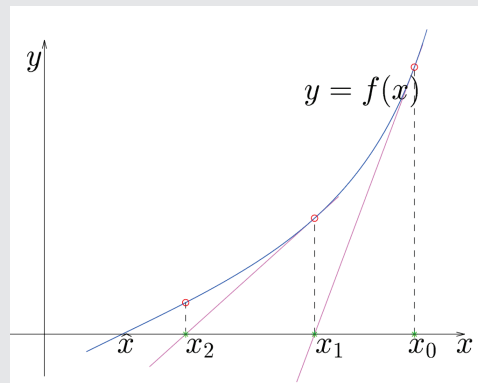
Ta má kořen

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Celý postup opakujeme a dostáváme iterační formuli

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

**Geometrický význam Newtonovy metody:**  
 Křivku  $y = f(x)$  nahradíme tečnou ke grafu v bodě  $x_k$  a hodnotu  $x_{k+1}$  získáme jako průsečík tečny s osou  $x$ . Proto se také Newtonova metoda nazývá **metoda tečen** nebo **metoda linearizace**.



**Poznámka:**  
 Jako zastavovací podmínku lze např. volit  $|x_{k+1} - x_k| < \epsilon$  nebo  $|f(x_k)| < \delta$ .

**Poznámka:**  
 Algoritmus Newtonovy metody je speciálním případem metody prosté iterace. Za funkci  $\varphi$  jsme volili funkci

$$\varphi(x) = x - \frac{f(x)}{f'(x)}$$

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

**Rychlost konvergence Newtonovy metody**

1. **způsob odvození** (Newtonova metoda jako speciální případ metody prosté iterace)

Rychlost konvergence závisí na  $\varphi'(\alpha)$ , resp.  $\varphi''(\alpha) \dots$  (viz dříve).

Platí:

$$\varphi(x) = x - \frac{f(x)}{f'(x)}$$

$$\varphi' = 1 - \frac{f' \cdot f' - f \cdot f''}{(f')^2} = 1 - 1 + \frac{f \cdot f''}{(f')^2} = \frac{f \cdot f''}{(f')^2}$$

$$\varphi'(\alpha) = 0, \text{ protože } f(\alpha) = 0$$

Platí:

$$\varphi'' = \frac{(f' \cdot f'' + f \cdot f''')(f')^2 - f \cdot f'' \cdot 2 \cdot f' \cdot f''}{(f')^4}$$

$$\varphi''(\alpha) = \frac{(f')^3 \cdot f'''}{(f')^4} = \frac{f'''(\alpha)}{f'(\alpha)}, \text{ protože } f(\alpha) = 0$$

Platí tedy  $\varphi'(\alpha) = 0$  a obecně  $\varphi''(\alpha) \neq 0 \Rightarrow$  rychlost konvergence je řádu 2.

2. **způsob odvození**

Pomocí Taylorova rozvoje funkce  $f$  v bodě  $x_0$  (necht' existují  $f'(x)$  a  $f''(x)$  v  $I$ ):

$$f(x) = f(x_0) + f'(x_0) \cdot (x - x_0) + \frac{1}{2} \cdot f''(\xi_0) \cdot (x - x_0)^2$$

Dosadíme za  $x$  přesné řešení  $\alpha$  (tj.  $f(\alpha) = 0$ )

$$\frac{f(\alpha)}{\alpha - x_0} = f'(x_0) + f''(\xi_0) \cdot \frac{1}{2} \cdot (\alpha - x_0)$$

Vydělíme  $f'(x_0) \neq 0$ :

$$0 = \frac{f(x_0)}{f'(x_0)} + \alpha - x_0 + \frac{1}{2} \frac{f''(\xi_0)}{f'(x_0)} (\alpha - x_0)^2$$

$$= -x_1$$

$$x_1 - \alpha = \frac{1}{2} \frac{f''(\xi_0)}{f'(x_0)} (\alpha - x_0)^2$$

Proces opakujeme:

$$\underbrace{x_{k+1} - \alpha}_{\text{chyba } k+1 \text{ iterace}} = \frac{1}{2} \frac{f''(\xi_k)}{f'(x_k)} \underbrace{(\alpha - x_k)^2}_{\text{kvadrát chyby } k\text{-té iterace}}$$

$\Rightarrow$  rychlost konvergence = 2

Necht' platí

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

$$\frac{1}{2} \left| \frac{f''(\xi)}{f'(x)} \right| \leq C \quad \forall \xi, \eta \in I$$

(Určit  $C$  může být obecně problém.)

Označíme-li  $\epsilon_k = x_k - \alpha$  chybu  $k$ -té iterace, potom z (\*) plyne

$$|\epsilon_{k+1}| \leq C|\epsilon_k|^2, \text{ tj. } |C\epsilon_{k+1}| \leq |C\epsilon_k|^2$$

$$|C\epsilon_1| \leq |C\epsilon_0|^2$$

$$|C\epsilon_2| \leq |C\epsilon_1|^2 \leq (|C\epsilon_0|^2)^2$$

$$|C\epsilon_3| \leq |C\epsilon_2|^2 \leq (|C\epsilon_0|^2)^4$$

$$\vdots$$

$$|C\epsilon_k| \leq |C\epsilon_0|^{2^k}$$

Dostáváme odhad chyby:

$$|\epsilon_k| \leq \frac{1}{C} |C\epsilon_0|^{2^k}$$

**Postačující podmínka konvergence:**  
 Platí  $|\epsilon_1| \leq C|\epsilon_0| = \underbrace{C}_{\neq} |\epsilon_0| |\epsilon_0|$

Pokud bude  $\neq < 1$ , dostaneme kontrakci.

$$|C\epsilon_0| = \underbrace{|C(\alpha - x_0)|}_{\neq} < 1$$

$$|C(\alpha - x_1)| \leq |C(\alpha - x_0)| < 1, \text{ tj. } |C\epsilon_1| < 1$$

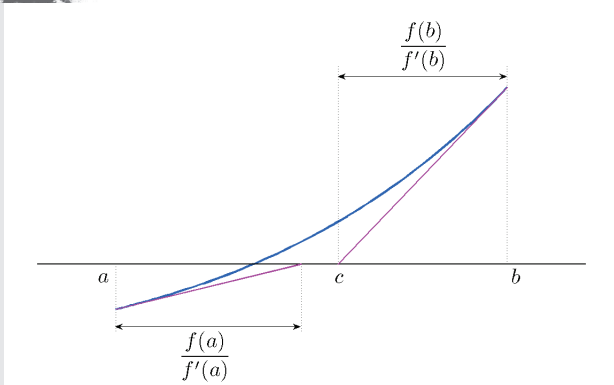
$$\vdots$$

Pro "velkou" hodnotu  $C$  musí být počáteční iterace  $x_0$  "velmi přesná".

**Věta** (Postačující podmínky konvergence Newtonovy metody.)  
 Je-li  $f'(a) \neq 0$ ,  $f''$  nemění znaménko v  $I = (a, b)$ , platí-li  $f(a) \cdot f(b) < 0$  a  $\left| \frac{f(a)}{f'(a)} \right| < b - a$ ,  $\left| \frac{f(b)}{f'(b)} \right| < b - a$ , potom Newtonova metoda konverguje  $\forall x_0 \in I$

Platnost tvrzení lze ověřit pomocí následujícího obrázku.





např.  $\frac{f(b)-f(c)}{b-c} = f'(b) \Rightarrow b-c = \frac{f(b)}{f'(b)}$

Praktické pravidlo pro odhad přesnosti:

Je-li  $|\alpha - x_k| < 10^{-d}$  potom  $|\alpha - x_{k+1}| < 10^{-2d}$ .

(pokud jsou splněny předpoklady pro odvození metody)

**Poznámka:**

Dosud jsme řešili nelineární rovnici pouze v  $\mathbb{R}$ . Algoritmus Newtonovy metody můžeme však použít i pro řešení dané rovnice v oboru **komplexních čísel**.

**Příklad 1**

Newtonovou metodou řešte v komplexním oboru rovnici

$$z^4 + z = 0, \quad z = x + iy, \quad x, y \in \mathbb{R}$$

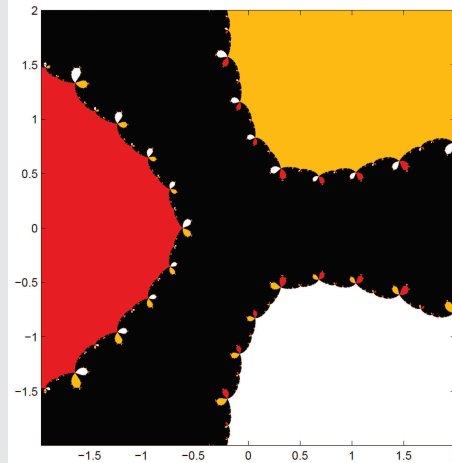
Iterační formule bude mít tvar

$$z_{k+1} = z_k - \frac{z_k^4 + z_k}{4z_k^3 + 1}$$

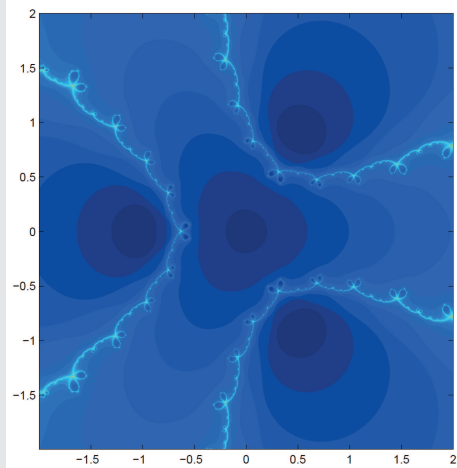
Je zřejmé, že daná rovnice bude mít 4 řešení:

$$0, \quad -1, \quad \frac{1}{2} + \frac{\sqrt{3}}{2}i, \quad \frac{1}{2} - \frac{\sqrt{3}}{2}i$$

Řešíme-li danou rovnici Newtonovou metodou pro konkrétní počáteční aproximaci ze čtverce  $(-2;2) \times (-2;2)$ , dostaneme jedno ze čtyř uvedených řešení. Obarvíme-li bod představující počáteční aproximaci různou barvou, podle toho k jakému řešení dospějeme, získáme fraktálovou strukturu.



Pokud vykreslíme pro každý počáteční bod počet iterací nutných k rozhodnutí, ke kterému z možných kořenů metoda konverguje, dostaneme následující obrázek (tmavé odstíny znamenají malý počet iterací, světlé odstíny velký počet iterací).



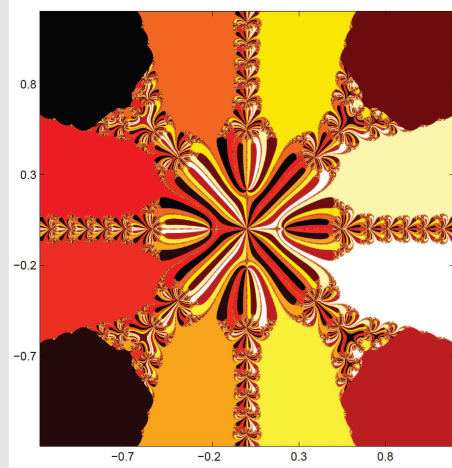
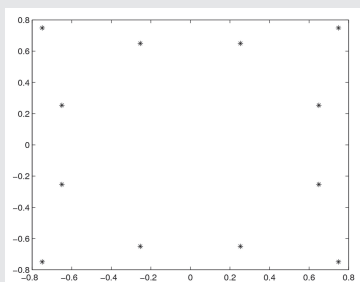
**Příklad 2**

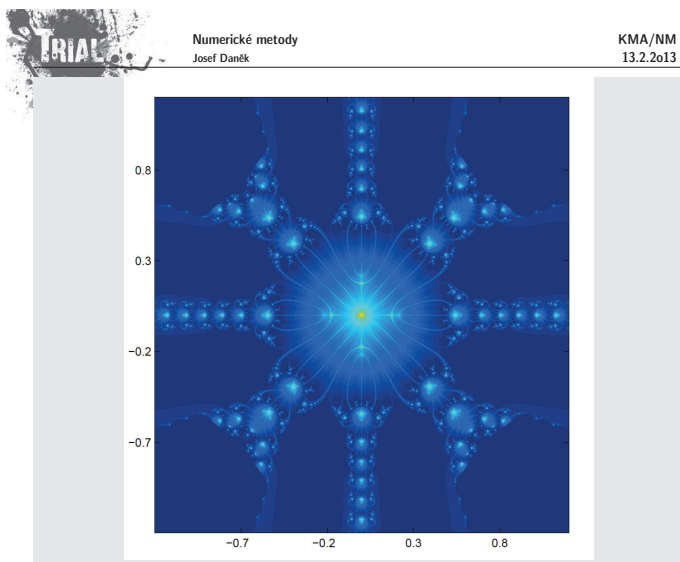
Newtonovou metodou řešte v komplexním oboru na čtverci  $(-1.2;1.2) \times (-1.2;1.2)$  rovnici

$$z^{12} + 744/611z^8 - 86/16057z^4 + 25/357 = 0$$

Kořeny:

- 0.75 + 0.75i
- 0.75 - 0.75i
- 0.75 + 0.75i
- 0.75 - 0.75i
- 0.65 + 0.25i
- 0.65 - 0.25i
- 0.25 + 0.65i
- 0.25 - 0.65i
- 0.25 + 0.65i
- 0.25 - 0.65i
- 0.65 + 0.25i
- 0.65 - 0.25i



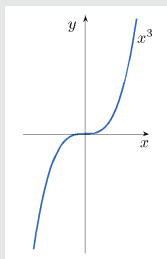


**Poznámka:**

Při odvozování Newtonovy metody jsme předpokládali, že  $f'(\alpha) \neq 0$ , tj.  $\alpha$  je jednoduchý kořen.

**Příklad:**

Pomocí Newtonovy metody najděte kořen rovnice  $x^3 = 0$ .



$\alpha = 0$  ... trojnásobný kořen

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

$$x_{k+1} = x_k - \frac{x_k^3}{3x_k^2}$$

$$x_{k+1} = \frac{2}{3}x_k \Rightarrow \text{rychlost konvergence je 1 !!!}$$

$$x_{k+1} = \frac{2}{3}x_k \wedge \alpha = \frac{2}{3}\alpha \Rightarrow (x_{k+1} - \alpha) = \frac{2}{3}(x_k - \alpha)^1$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

**Definice:** Kořen  $\alpha$  rovnice  $f(x) = 0$  má **násobnost**  $s$ , jestliže  $0 \neq g(\alpha) < \infty$ , kde  $g(x) = \frac{f(x)}{(x-\alpha)^s}$

**Modifikovaná iterační formule**

$$x_{k+1} = x_k - s \frac{f(x_k)}{f'(x_k)} \dots \text{již opět kvadratický iterační proces}$$

pro předchozí příklad:  $x_{k+1} = x_k - 3 \frac{x_k^3}{3x_k^2} = x_k - x_k = 0$   
nevýhoda - musíme znát násobnost  $s$

**Jiný přístup pro hledání násobných kořenů**  
Je-li  $\alpha$   $s$ -násobný kořen rovnice  $f(x) = 0$ , potom je  $\alpha$   $(s-1)$ -násobným kořenem rovnice  $f'(x) = 0$  a tedy jednoduchým kořenem rovnice

$$g(x) = \frac{f(x)}{f'(x)} = 0.$$

D.cv:  $g'(x) = ?$

**Aitkenův proces**

Konverguje-li iterační metoda lineárně, lze pomocí Aitkenova procesu urychlit konvergenci.

Platí:

$$\alpha - x_{k+1} = C_{k+1}(\alpha - x_k), \quad |C_k| < 1$$

kde  $|C_k| \rightarrow C$  je asymptotická konstanta chyby.

Jsmo-li blízko limity, jsou čísla  $C_k$  přibližně stejná a lze psát

$$\alpha - x_{k+1} \approx \bar{C}(\alpha - x_k), \quad |\bar{C}| < 1$$

Pro další iteraci

$$\alpha - x_{k+2} \approx \bar{C}(\alpha - x_{k+1})$$

Po vyloučení  $\bar{C}$ :

$$\frac{\alpha - x_{k+1}}{\alpha - x_k} \approx \frac{\alpha - x_{k+2}}{\alpha - x_{k+1}}$$

$$(\alpha - x_{k+2})(\alpha - x_k) \approx (\alpha - x_{k+1})^2$$

$$\alpha^2 - \alpha(x_k + x_{k+2}) + x_k x_{k+2} \approx \alpha^2 - 2\alpha x_{k+1} + x_{k+1}^2$$

$$x_k x_{k+2} - x_{k+1}^2 \approx \alpha(x_k - 2x_{k+1} + x_{k+2})$$

$$\alpha \approx \frac{x_k x_{k+2} - x_{k+1}^2}{x_k - 2x_{k+1} + x_{k+2}}$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

**Prakticky:**

$x_0, x_1, x_2 \rightarrow \alpha \approx x_3$   
 $x_3, x_4, x_5 \rightarrow \alpha \approx x_6$   
...

**Příklad 3**  
Pomocí **metody prosté iterace** řešte rovnici  $x^2 - x = 0$ . Použijte přepis  $x = \sqrt{x}$ , počáteční iteraci  $x_0 = 3$  a zastavovací podmínku  $|x_k - x_{k-1}| < 10^{-5}$ .

*výsledky získané v MATLABu*

krok	x(k)	dx(k)=x(k)-x(k-1)	dx(k)/dx(k-1)
0	3.000000		
1	1.732051	-1.267949	
2	1.316074	-0.415977	0.328071
3	1.147203	-0.168871	0.405963
4	1.071075	-0.076127	0.450800
5	1.034928	-0.036148	0.474833
6	1.017314	-0.017614	0.487272
7	1.008620	-0.008694	0.493600
8	1.004301	-0.004319	0.496791
9	1.002148	-0.002153	0.498393
10	1.001073	-0.001075	0.499196
11	1.000537	-0.000537	0.499598
12	1.000268	-0.000268	0.499799
13	1.000134	-0.000134	0.499899
14	1.000067	-0.000067	0.499950
15	1.000034	-0.000034	0.499975
16	1.000017	-0.000017	0.499987
17	1.000008	-0.000008	0.499994

Předchozí výpočet urychlete použitím **Aitkenova procesu**.

*výsledky získané v MATLABu*

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

krok	x(k)	dx(k)=x(k)-x(k-1)	dx(k)/dx(k-1)
0	3.000000		
1	1.732051	-1.267949	
2	1.316074	-0.415977	0.328071
Zpsneni pomoci Aitkenovy formule			
3	1.112973	-0.203101	0.488251
4	1.054975	-0.057997	0.285559
5	1.027120	-0.027855	0.480285
Zpsneni pomoci Aitkenovy formule			
6	1.001378	-0.025742	0.924133
7	1.000689	-0.000689	0.026771
8	1.000344	-0.000344	0.499742
Zpsneni pomoci Aitkenovy formule			
9	1.000000	-0.000344	0.998968
10	1.000000	-0.000000	0.000344

**Příklad 4**  
Pomocí **Newtonovy metody** řešte rovnici  $x^2 - x = 0$ . Použijte počáteční iteraci  $x_0 = 3$  a zastavovací podmínku  $|x_k - x_{k-1}| < 10^{-5}$ .

*výsledky získané v MATLABu*

krok	x(k)	dx(k)=x(k)-x(k-1)	dx(k)/dx(k-1)
0	3.000000		
1	1.800000	-1.200000	
2	1.246154	-0.553846	0.461538
3	1.040603	-0.205551	0.371134
4	1.001525	-0.039078	0.190113
5	1.000002	-0.001522	0.038959
6	1.000000	-0.000002	0.001522

Rychlost konvergence Newtonovy metody je 2, tj. pro urychlení nelze použít Aitkenův proces. Pokud bychom jej použili, výpočet se naopak zpomalí.

*výsledky získané v MATLABu*

krok	x(k)	dx(k)=x(k)-x(k-1)	dx(k)/dx(k-1)
0	3.000000		
1	1.800000	-1.200000	
2	1.246154	-0.553846	0.461538
Zpřesnění pomocí Aitkenovy formule			
3	0.771429	-0.474725	0.857143
4	1.096241	0.324812	-0.684211
5	1.007767	-0.088473	-0.272383
Zpřesnění pomocí Aitkenovy formule			
6	1.026707	0.018940	-0.214073
7	1.000677	-0.026030	-1.374350
8	1.000000	-0.000677	0.025995
Zpřesnění pomocí Aitkenovy formule			
9	0.999982	-0.000018	0.026688
10	1.000000	0.000018	-0.974664
11	1.000000	-0.000000	-0.000018

### Nevýhody Newtonovy metody

- zadaná funkce  $f$  musí být diferencovatelná
- derivace se přímo vyskytuje v iterační formuli
- v každé iteraci musíme kromě funkční hodnoty počítat také hodnotu derivace

Pro odbourání poslední vlastnosti můžeme za předpokladu, že se derivace  $f'$  na okolí kořene příliš nemění, Newtonovu metodu modifikovat tak, že hodnotu derivace vypočteme pouze jednou, tj. v bodě  $x_0$  a položíme

$$f'(x_k) \approx f'(x_0).$$

Dostaneme iterační formuli **modifikované Newtonovy metody**

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_0)}$$

Chceme-li modifikovat Newtonovu metodu pro funkce, které nejsou diferencovatelné, nahradíme v iterační formuli derivaci  $f'(x_k)$  diferencčním podílem

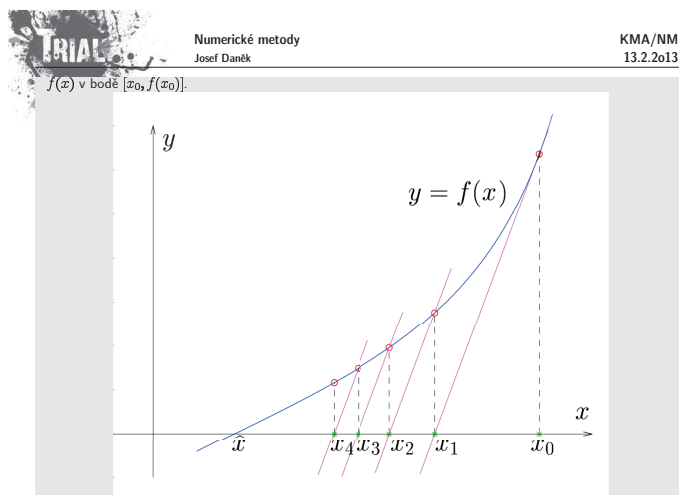
$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

Dostaneme iterační formuli **metody sečen**

$$x_{k+1} = x_k - f(x_k) \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}$$

### Geometrický význam modifikované Newtonovy metody

Tečny ke grafu v bodech  $[x_k, f(x_k)]$  nahrazujeme přímkami rovnoběžnými s tečnou ke grafu funkce  $y =$

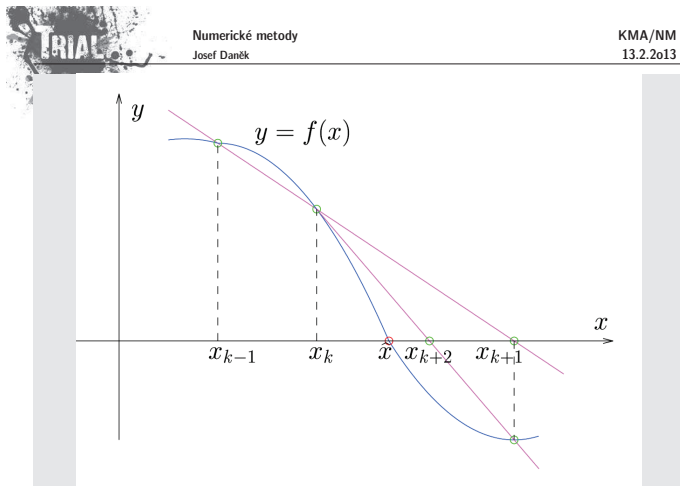


### Poznámka:

V této modifikaci počítáme pouze jednu hodnotu derivace  $f'(x_0)$ , a proto je tento postup vhodný je-li derivace  $f'(x)$  složitá. Nemění-li  $f'(x)$  a  $f''(x)$  znaménko, je možné dokázat konvergenci této metody.

### Geometrický význam metody sečen

Mějme dvě dobré aproximace  $x_{k-1}$  a  $x_k$  kořene  $\hat{x}$  rovnice  $f(x) = 0$ . Křivku  $y = f(x)$  nahradíme přímkou (sečnou), která prochází body  $[x_{k-1}, f(x_{k-1})]$  a  $[x_k, f(x_k)]$ . Další iteraci  $x_{k+1}$  získáme jako průsečík sečny s osou  $x$ .



### Poznámka:

Pro zahájení výpočtu potřebujeme znát 2 počáteční aproximace, ale na rozdíl od Newtonovy metody počítáme v každém kroku pouze jednu novou funkční hodnotu, což je úspora času.

### Poznámka:

Metoda sečen má obdobný algoritmus jako metoda regula falsi, nepožadujeme však splnění podmínky  $f(x_{k-1}) \cdot f(x_k) < 0$ .

### Rychlost konvergence metody sečen

Odvzuje se podobně jako u Newtonovy metody. Necht' platí

$$\left| \frac{1}{2} \frac{f''(\eta)}{f'(\eta)} \right| \leq C \quad \forall \eta, \eta \in I$$

Potom

$$|e_{k+1}| \leq C \cdot |e_k| \cdot |e_{k-1}|$$

Náznak odvození:

Označme  $d_k = C|e_k|$ , potom

$$d_{k+1} \leq d_k d_{k-1}$$

Nechť čísla  $d_0$  a  $d_1$  jsou rovna číslu  $d < 1$  nebo menší, potom dosazením dostáváme

$$d_2 \leq d_1 d_0 \leq d^2 d = d^3$$

$$d_3 \leq d_2 d_1 \leq d^3 d = d^6$$

$$d_4 \leq d_3 d_2 \leq d^6 d^2 = d^8$$

$$d_5 \leq d_4 d_3 \leq d^8 d^3 = d^{11}$$

⋮

$$d_{k+1} \leq d^{n_{k+1}}, \text{ kde } n_{k+1} = n_k + n_{k-1}, n_1 = 1, n_0 = 1.$$

Rekurentně daná posloupnost  $n_k$  definuje tzv. **Fibonacciho čísla**. Odpovídající charakteristická rovnice je

$$\rho^2 = \rho + 1.$$

Pro její kořeny platí:

$$\rho_{1,2} = \frac{1 \pm \sqrt{5}}{2} = \begin{cases} 1,618 \\ -0,618 \end{cases}$$

Snadno se ověří, že explicitní vyjádření pro  $n_k$  je následující

$$n_k = \frac{1}{\sqrt{5}} \left[ \left( \frac{1+\sqrt{5}}{2} \right)^{k+1} - \left( \frac{1-\sqrt{5}}{2} \right)^{k+1} \right].$$

Hledaný řád metody  $r$  potom získáme ze vztahu

$$d_{k+1} \approx d_k^r,$$

tj. po zlogaritmování

$$r = \frac{\log d_{k+1}}{\log d_k} = \frac{\log d^{n_{k+1}}}{\log d^{n_k}} = \frac{n_{k+1}}{n_k}, \text{ pro } k \rightarrow \infty.$$

$$\lim_{k \rightarrow \infty} \frac{n_{k+1}}{n_k} = \lim_{k \rightarrow \infty} \frac{\frac{1}{\sqrt{5}} \left[ \left( \frac{1+\sqrt{5}}{2} \right)^{k+2} - \left( \frac{1-\sqrt{5}}{2} \right)^{k+2} \right]}{\frac{1}{\sqrt{5}} \left[ \left( \frac{1+\sqrt{5}}{2} \right)^{k+1} - \left( \frac{1-\sqrt{5}}{2} \right)^{k+1} \right]} = \frac{1+\sqrt{5}}{2}$$

Pro rychlost konvergence tedy dostáváme

$$r = \frac{1}{2}(1 + \sqrt{5}) \doteq 1,618 \quad (\text{pro } k \rightarrow \infty).$$

**Poznámka:**  
Metoda sečen je tzv. dvoukroková interpolační metoda, analogicky lze odvodit tříkrokovou interpolační metodu, kterou nazýváme **Mullerova metoda**.

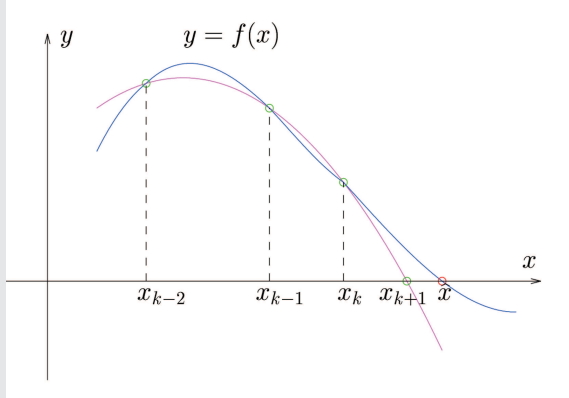
**Geometrický význam Mullerovy metody**

Mějme tři dobré aproximace  $x_{k-2}$ ,  $x_{k-1}$  a  $x_k$  kořene  $x$  rovnice

$$f(x) = 0.$$

Křivku  $y = f(x)$  nahradíme parabolou (kvadratickou funkcí), která prochází body  $[x_{k-2}, f(x_{k-2})]$ ,  $[x_{k-1}, f(x_{k-1})]$  a  $[x_k, f(x_k)]$ .

Další iteraci  $x_{k+1}$  získáme jako průsečík paraboly s osou  $x$ . (Ten, který je blíže k  $x_k$ .)



Prostředky MATLABu pro řešení nelineárních rovnic

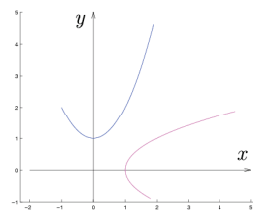
- fzero pro obecnou nelineární rovnici
- roots pro kořeny polynomu

**Příklad:**

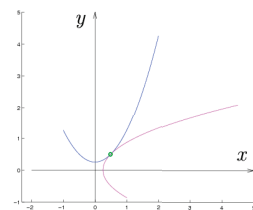
Řešte soustavu dvou rovnic pro dvě neznámé ( $a \in \mathbb{R} \dots$  parametr)

$$\begin{aligned} x^2 - y + a &= 0 \\ -x + y^2 + a &= 0 \end{aligned}$$

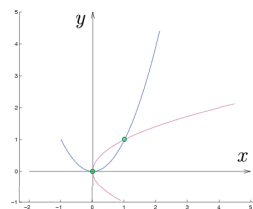
1)  $a = 1$



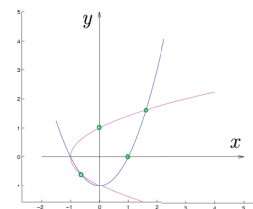
2)  $a = \frac{1}{4}$



3)  $a = 0$



4)  $a = -1$



**Formule:**

Jsou dány funkce  $F_i: \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $i = 1, \dots, n$  definované na  $(a_i, b_i)$ .  
Označme  $I = (a_1, b_1) \times (a_2, b_2) \times \dots \times (a_n, b_n)$ .

Hledáme  $x = (x_1, x_2, \dots, x_n)^T \in I$  tak, aby

$$\begin{aligned} F_1(x_1, x_2, \dots, x_n) &= 0 \\ F_2(x_1, x_2, \dots, x_n) &= 0 \\ &\vdots \\ F_n(x_1, x_2, \dots, x_n) &= 0 \end{aligned}$$

Vektorově:

$$F(x) = 0, \text{ kde } F = (F_1, F_2, \dots, F_n)^T.$$

**Věta** (Postačující podmínky konvergence metody prosté iterace.)

Předpokládejme, že je funkce  $\Phi$  na  $I$  spojitá a platí:

- (a)  $\forall x \in I: \Phi(x) \in I$  (funkce  $\Phi$  zobrazuje  $I$  do sebe).
- (b)  $\exists q \in (0, 1): \|\Phi(x) - \Phi(y)\| \leq q \|x - y\| \quad \forall x, y \in I$  (funkce  $\Phi$  je kontrakce).

Potom

- 1. v množině  $I$  existuje právě jedno řešení  $x$  soustavy rovnic  $x = \Phi(x)$ .
- 2. posloupnost  $\{x^k\}_{k=1}^{\infty}$  určená formulí  $x^k = \Phi(x^{k-1})$  konverguje pro každé  $x^0 \in I$  a  $\lim_{k \rightarrow \infty} x^k = x$ .

**Metoda prosté iterace**

Soustavu rovnic  $F(x) = 0$  nahradíme soustavou rovnic  $x = \Phi(x)$  (více možností).

Algoritmus:

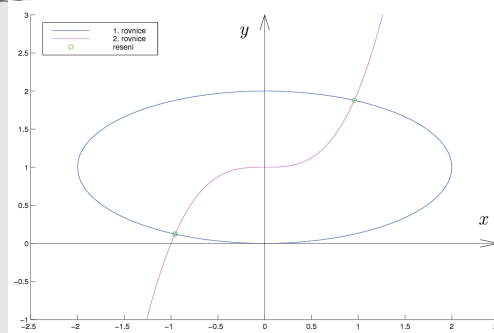
- 1) Zadáme  $x^0 \in I$ ,  $\epsilon > 0$
- 2)  $x^{k+1} = \Phi(x^k)$
- 3) Je-li  $\|x^{k+1} - x^k\| < \epsilon$ , pak  $x = x^{k+1}$ , KONEC  
jinak jdi na 2)

**Příklad 5**

Řešte metodou prosté iterace soustavu dvou rovnic pro dvě neznámé

$$\begin{aligned} x^2 + 4y^2 - 8y &= 0 \\ x^3 - y + 1 &= 0 \end{aligned}$$

- 1. rovnice je rovnicí elipsy  $x^2 + 4(y-1)^2 = 4$   
 $(\frac{x}{2})^2 + (y-1)^2 = 1$
- 2. rovnici upravíme na tvar  $y = x^3 + 1$



Z 2. rovnice vyjádříme  $x$ :

$$\begin{aligned} x^3 &= y - 1 \\ x &= \sqrt[3]{y-1} \end{aligned}$$

Z 1. rovnice vyjádříme  $y$ :

$$\begin{aligned} 4y^2 &= 8y - x^2 \\ y &= \frac{1}{2}\sqrt{8y - x^2} \end{aligned}$$

Rekurentní formule:

$$x_{k+1} = \sqrt[3]{y_k - 1} \quad y_{k+1} = \frac{1}{2}\sqrt{8y_k - x_{k+1}^2}$$

výsledky získané v MATLABu

krok	x_1(k)	x_2(k)	x(k)-x(k-1)
0	1.000000	1.000000	
1	0.000000	1.322876	1.050832
2	0.686033	1.626577	0.750250
3	0.855706	1.770732	0.222643
4	0.916856	1.832595	0.086985
5	0.940758	1.858772	0.035448
6	0.950516	1.869836	0.014752
7	0.954580	1.874514	0.006197
8	0.956288	1.876492	0.002613
9	0.957009	1.877328	0.001104
10	0.957313	1.877682	0.000467

výsledky získané v MATLABu

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

krok	x_1(k)	x_2(k)	x(k)-x(k-1)
0	-1.000000	0.000000	
1	0.500000	0.000000	1.802776
2	0.651123	0.677644	2.108913
3	0.670383	1.241743	1.749676
4	0.716783	1.673937	1.015477
5	0.838816	1.743370	0.454007
6	0.906819	1.820267	0.181812
7	0.936234	1.853465	0.074113
8	0.948577	1.867581	0.030794
9	0.953759	1.873559	0.012921
10	0.955941	1.876088	0.005446
11	0.956862	1.877158	0.002300
12	0.957251	1.877610	0.000972

**Poznámka:**  
Pozor na definiční obory funkcí.  
Pozor na zápis funkcí v MATLABu.

```

>> (-8)^(1/3)
ans =
    1.0000 + 1.7321i

```

**Poznámka:** Pokud chceme v předchozím příkladě najít druhé řešení, je třeba zvolit jiný předpis pro funkci  $\Phi$ .

*výsledky získané v MATLABu*

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

Metoda prosté iterace pro řešení soustavy nelineárních rovnic  $F(x)=0$  s využitím prepisu na tvar  $x=\Phi(x)$  pro počáteční aproximaci  $x_0=[1,1]$  a zastavovací podmínku  $||x(k)-x(k-1)|| < 0.001$

Funkce  $\Phi(x)$  je zadána takto:

```

function out=Phi(in);
x=in(1);
y=in(2);
if (y-1)>=0
    out(1)=(y-1)^(1/3);
else
    out(1)=-(1-y)^(1/3);
end;
out(2)=(x^2+4*y^2)/8;
out=out';

```

krok	x_1(k)	x_2(k)	x(k)-x(k-1)
0	1.000000	1.000000	
1	0.000000	0.625000	1.068000
2	-0.721125	0.195312	0.839436
3	-0.930127	0.084076	0.236761
4	-0.971150	0.111677	0.049444
5	-0.961296	0.124127	0.015879
6	-0.956783	0.123215	0.004604
7	-0.957116	0.122020	0.001240
8	-0.957550	0.121953	0.000440

### Newtonova metoda

Odvození je opět analogické případu funkce jedné reálné proměnné. Vyjádříme si Taylorův rozvoj funkce  $F$  v bodě  $x^k$ . (Předpokládáme, že existují derivace !)  
Soustavu rovnic  $F(x)=0$  nahradíme soustavou lineárních rovnic

$$F(x^k) + F'(x^k)(x - x^k) = 0$$

Její řešení označíme  $x^{k+1}$ , tj.

$$F(x^k) + F'(x^k) \frac{(x^{k+1} - x^k)}{h^k} = 0$$

Dostáváme soustavu

$$F'(x^k)h^k = -F(x^k),$$

kteřá má řešení

$$h^k = -[F'(x^k)]^{-1}F(x^k)$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

Novou iteraci  $x^{k+1}$  získáme ze vztahu  $x^{k+1} = x^k + h^k$

**Poznámka:**  
 $F'(x^k)$  je Jacobiho matice funkce  $F(x)$  v bodě  $x^k$ .  
Je zřejmé, že musí být regulární (musí existovat matice k ní inverzní).

**Příklad 6**  
Řešte Newtonovou metodou soustavu dvou rovnic pro dvě neznámé

$$\begin{aligned} x^2 + 4y^2 - 8y &= 0 \\ x^3 - y + 1 &= 0 \end{aligned}$$


---


$$F(x, y) = \begin{bmatrix} x^2 + 4y^2 - 8y \\ x^3 - y + 1 \end{bmatrix}$$

$$\frac{\partial F_1(x, y)}{\partial x} = 2x \quad \frac{\partial F_1(x, y)}{\partial y} = 8y - 8$$

$$\frac{\partial F_2(x, y)}{\partial x} = 3x^2 \quad \frac{\partial F_2(x, y)}{\partial y} = -1$$

$$F'(x, y) = \begin{bmatrix} 2x & 8y - 8 \\ 3x^2 & -1 \end{bmatrix}$$


---

**1. iterace**

$$\begin{bmatrix} x^1 \\ y^1 \end{bmatrix} = \begin{bmatrix} x^0 \\ y^0 \end{bmatrix} + \begin{bmatrix} h_1^0 \\ h_2^0 \end{bmatrix}$$

Platí  $h^0 = -[F'(x^0, y^0)]^{-1}F(x^0, y^0)$

Abychom nemuseli počítat inverzní matici, vypočteme  $h^0$  jako řešení soustavy

$$F'(x^0, y^0)h^0 = -F(x^0, y^0)$$

$$F(x^0, y^0) = \begin{bmatrix} 4 \\ 7 \end{bmatrix} \quad F'(x^0, y^0) = \begin{bmatrix} 4 & 8 \\ 12 & -1 \end{bmatrix}$$

Dostaneme

$$h^0 = \begin{bmatrix} -0.6 \\ -0.2 \end{bmatrix} \Rightarrow \begin{bmatrix} x^1 \\ y^1 \end{bmatrix} = \begin{bmatrix} x^0 \\ y^0 \end{bmatrix} + \begin{bmatrix} h_1^0 \\ h_2^0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix} + \begin{bmatrix} -0.6 \\ -0.2 \end{bmatrix} = \begin{bmatrix} 1.4 \\ 1.8 \end{bmatrix}$$


---

**2. iterace**

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

$$\begin{bmatrix} x^2 \\ y^2 \end{bmatrix} = \begin{bmatrix} x^1 \\ y^1 \end{bmatrix} + \begin{bmatrix} h_1^1 \\ h_2^1 \end{bmatrix}$$

Platí  $h^1 = -[F'(x^1, y^1)]^{-1}F(x^1, y^1)$

Abychom nemuseli počítat inverzní matici, vypočteme  $h^1$  jako řešení soustavy

$$F'(x^1, y^1)h^1 = -F(x^1, y^1)$$

$$F(x^1, y^1) = \begin{bmatrix} 0,52 \\ 1,944 \end{bmatrix} \quad F'(x^1, y^1) = \begin{bmatrix} 2,8 & 6,4 \\ 5,88 & -1 \end{bmatrix}$$

Dostaneme

$$h^1 = \begin{bmatrix} -0,3206 \\ 0,0590 \end{bmatrix}$$

$$\begin{bmatrix} x^2 \\ y^2 \end{bmatrix} = \begin{bmatrix} x^1 \\ y^1 \end{bmatrix} + \begin{bmatrix} h_1^1 \\ h_2^1 \end{bmatrix} = \begin{bmatrix} 1,4 \\ 1,8 \end{bmatrix} + \begin{bmatrix} -0,3206 \\ 0,0590 \end{bmatrix} = \begin{bmatrix} 1,0794 \\ 1,8590 \end{bmatrix}$$


---

Další iterace bychom počítali podobně. V následující tabulce jsou shrnuty 4. iterace Newtonovy metody.

k	x <sub>k</sub>	y <sub>k</sub>	x <sub>k</sub> - x <sub>k-1</sub>
0	2.0000	2.0000	
1	1.4000	1.8000	0.6325
2	1.0794	1.8590	0.3260
3	0.9703	1.8763	0.1105
4	0.9577	1.8779	0.0127

---

**Normy vektorů a matic**

Připomenutí:  
**Norma** je zobrazení  $n$  lineárního vektorového prostoru  $\mathcal{L}$  do  $\mathbb{R}_0^+$ :

- $n(x) = 0 \Leftrightarrow x = 0$  (definitnost)
- $n(\lambda x) = |\lambda|n(x)$  (homogenita)
- $n(x + y) \leq n(x) + n(y)$  ( $\Delta$  nerovnost)

**VEKTORY:**

$\|x\|_p = (\sum_i |x_i|^p)^{\frac{1}{p}}$   
 ... p-tá vektorová norma

$\|x\|_1 = \sum_i |x_i|$  ... první vektorová norma  $\leftrightarrow$   $\|A\|_S = \max_k \{\sum_i |a_{ik}|\}$  ... sloupcová maticová norma

$\|x\|_2 = \sqrt{\sum_i x_i^2}$  ... euklidovská vektorová norma  $\leftrightarrow$   $\|A\|_{SP} = \max_k \{\lambda_k^{\frac{1}{2}}(A^H A)\}$  ... spektrální norma

$\|x\|_\infty = \max_i |x_i|$  ... maximová norma  $\leftrightarrow$   $\|A\|_R = \max_k \{\sum_i |a_{ik}|\}$  ... řádková maticová norma

**Poznámky:**

$A^H$  ... hermitovský transponovaná matice;  $A^H = [a_{ji}^H] = [\bar{a}_{ji}]$ ;  $\alpha$  je komplexně sdružené číslo k číslu  $\alpha$

Symbol  $\leftrightarrow$  značí vazbu mezi vektorovou a maticovou normou. Příslušná maticová norma je **generovaná** příslušnou vektorovou normou.

**Příklad**

Pro zadanou matici A a vektor x určete výše uvedené normy.

$A = \begin{bmatrix} 2 & -1 \\ 0 & 3 \end{bmatrix}, \quad x = \begin{bmatrix} 6 \\ -1 \end{bmatrix}$

$\|A\|_S = \max\{|2| + |0|, |-1| + |3|\} = \max\{2, 4\} = 4$   $\|x\|_1 = |6| + |-1| = 7$   
 $\|A\|_R = \max\{|2| + |-1|, |0| + |3|\} = \max\{3, 3\} = 3$   $\|x\|_\infty = \max\{|6|, |-1|\} = 6$

$\|A\|_{SP}$ :

$A^H A = \begin{bmatrix} 2 & 0 \\ -1 & 3 \end{bmatrix} \cdot \begin{bmatrix} 2 & -1 \\ 0 & 3 \end{bmatrix} = \begin{bmatrix} 4 & -2 \\ -2 & 10 \end{bmatrix}$

$\det(A^H A - \lambda I) = \begin{vmatrix} 4 - \lambda & -2 \\ -2 & 10 - \lambda \end{vmatrix} =$   
 $= (4 - \lambda)(10 - \lambda) - 4 = \lambda^2 - 14\lambda + 36$

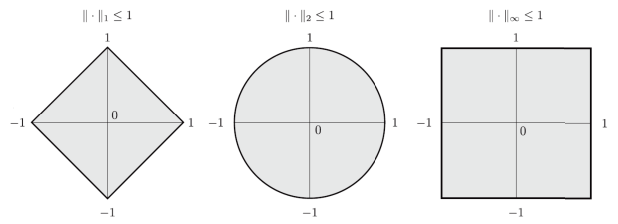
$\lambda_{1,2}(A^H A) = \frac{14 \pm \sqrt{14^2 - 4 \cdot 36}}{2} =$   
 $= 7 \pm \sqrt{7^2 - 36} = 7 \pm \sqrt{13}$

$\max |\lambda_{i,2}| = 7 + \sqrt{13}$

$\|A\|_{SP} = \sqrt{7 + \sqrt{13}} \approx 3,2566$   $\|x\|_2 = \sqrt{6^2 + (-1)^2} = \sqrt{37} \approx 6,0827$

**Geometrický význam vektorových norem**

jednotkové koule v  $\mathbb{R}^2$  ... množina (bodů) prvků s normou  $\leq 1$ :



- $\|x\|_1 = \sum_i |x_i| = |x| + |y| \leq 1$
- $\|x\|_2 = \sqrt{\sum_i x_i^2} = \sqrt{x^2 + y^2} \leq 1$
- $\|x\|_\infty = \max_i |x_i| = \max\{|x|, |y|\} \leq 1$

**Kapitola 3. SLAR - přímé metody**

**Formule:**

Je dána čtvercová matice  $A = [a_{ij}]_{i,j=1}^n$  a vektor pravé strany  $b = [b_1, b_2, \dots, b_n]^T$ .

Hledáme vektor  $x = [x_1, x_2, \dots, x_n]^T$  tak, aby platilo

$Ax = b$ ,

rozepsáno po složkách

$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1$

$a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2$

...

$a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n$

Předpokládáme, že je matice A regulární (tj. soustava má právě jedno řešení).

Máme dva základní typy soustav:

- soustavy s obecnou maticí
- soustavy se speciální maticí (symetrická, pozitivně definitní, řídká, pásová apod.)

Pro první skupinu se většinou používají přímé metody, pro druhou skupinu metody iterační nebo speciální modifikace přímých metod.

**Cramerovo pravidlo**

neznamá složka řešení  $x_i = \frac{\det A_i}{\det A}$

počet operací:

Je nutné vypočítat  $(n + 1)$  determinantů.

Pro výpočet determinantu je třeba  $n!$  sčítání a v každém sčítání je  $(n - 1)$  násobení.

Dostáváme:

$(n + 1)(n - 1)n! + n! = n(n + 1)!$

např: pro  $n = 30$ ,  $10^6$  operací za sekundu  $\rightarrow$  výpočet trvá **7,82 · 10<sup>21</sup>** let

Idea dalších přímých metod vychází z faktu, že soustavy

$Ax = b$  a  $TAx = Tb$ ,

kde T je regulární matice, mají totéž řešení, tj. jsou ekvivalentní.

Touto transformací lze získat trojúhelníkovou soustavu

$Ux = y; \quad U = TA, \quad y = Tb$

př:

$\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$

Trojúhelníkovou soustavu lze velmi snadno řešit zpětnou substitucí. Realizovaný proces se nazývá zpětný chod.

**Gaussova eliminační metoda**

$Ax = b$  rozepsáno po složkách  $\begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \end{bmatrix}$

Definujeme multiplikátory  $m_{21} = -\frac{a_{21}}{a_{11}}$ ,  $m_{31} = -\frac{a_{31}}{a_{11}}$

$r_1^{(1)} = r_1$   $b_1^{(1)} = b_1$

$r_2^{(1)} = r_2 + m_{21}r_1$   $b_2^{(1)} = b_2 + m_{21}b_1$

$r_3^{(1)} = r_3 + m_{31}r_1$   $b_3^{(1)} = b_3 + m_{31}b_1$

Získáme novou soustavu ... **1. fáze eliminace**

$A^{(1)}x = b^{(1)} \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & b_2^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(1)} & b_3^{(1)} \end{bmatrix}$

Definujeme multiplikátory  $m_{32} = -\frac{a_{32}^{(1)}}{a_{22}^{(1)}}$

$r_1^{(2)} = r_1^{(1)}$   $b_1^{(2)} = b_1^{(1)}$

$r_2^{(2)} = r_2^{(1)}$   $b_2^{(2)} = b_2^{(1)}$

$r_3^{(2)} = r_3^{(1)} + m_{32}r_2^{(1)}$   $b_3^{(2)} = b_3^{(1)} + m_{32}b_2^{(1)}$

Získáme novou soustavu ... **2. fáze eliminace**

$A^{(2)}x = b^{(2)} \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & b_2^{(2)} \\ 0 & 0 & a_{33}^{(2)} & b_3^{(2)} \end{bmatrix}$

Celý tento postup nazýváme přímý chod. Trojúhelníkovou soustavu řešíme zpětným chodem.

Efektivnost algoritmu GEM

Bereme v úvahu pouze operace násobení a dělení (počet operací sčítání je přibližně stejný).  
( $N \dots$  je řád matice  $A$ )  
• Celkem je  $N - 1$  fázi eliminace. V  $K$ -té fázi počítáme  $N - K$  multiplikátorů (tj.  $N - K$  dělení)

$$\sum_{K=1}^{N-1} (N - K) = (N - 1)N - \sum_{K=1}^{N-1} K = (N - 1)N - \frac{1}{2}(N - 1)N = \frac{1}{2}(N - 1)N$$

• Každým multiplikátorem vynásobíme  $(N - K + 1)$  prvků rozšířené matice (jeden rozšířený řádek), tj.  $(N - K)(N - K + 1)$  v  $K$ -té fázi

$$\begin{aligned} \sum_{K=1}^{N-1} (N - K)(N - K + 1) &= \sum_{K=1}^{N-1} [(N^2 + N) - K(2N + 1) + K^2] = \\ &= (N - 1)(N^2 + N) - \frac{1}{2}N(N - 1)(2N + 1) + \frac{1}{6}(N - 1)N(2N - 1) = \\ &= N^3 - N^2 + N^2 - N - N^3 + \frac{1}{2}N^2 + \frac{1}{2}N + \frac{1}{3}N^3 - \frac{1}{2}N^2 + \frac{1}{6}N = \\ &= \frac{1}{3}N^3 - \frac{1}{3}N \end{aligned}$$

• Zpětný chod

$$1 + (\underbrace{1}_{\text{dělení}} + \underbrace{1}_{\text{násobení}}) + (1 + 2) + \dots + (1 + N - 1) = \sum_{K=1}^N K = \frac{1}{2}N(N + 1)$$

Celkem

$$\frac{1}{2}N(N - 1) \quad + \quad \frac{1}{3}N^3 - \frac{1}{3}N \quad + \quad \frac{1}{2}N(N + 1) = \frac{1}{3}N^3 + N^2 - \frac{1}{3}N$$

počet multiplikátorů     přímý chod     zpětný chod

Příklad 1

Řešte soustavu rovnic

$$\begin{aligned} x + 2y + 3z &= 14 \\ 2x + 4y + 5z &= 25 \\ 7x + 8y + 9z &= 50 \end{aligned} \quad \text{tj.} \quad \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 7 & 8 & 9 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 14 \\ 25 \\ 50 \end{bmatrix}$$

Řešení

Pro zápis budeme používat tvar **matice rozšířené**:

$$\left[ \begin{array}{ccc|c} 1 & 2 & 3 & 14 \\ 2 & 4 & 5 & 25 \\ 7 & 8 & 9 & 50 \end{array} \right] \quad / \cdot (-2) \quad / \cdot (-7)$$

$$\left[ \begin{array}{ccc|c} 1 & 2 & 3 & 14 \\ 0 & 0 & -1 & -3 \\ 0 & -6 & -12 & -48 \end{array} \right] \quad / \cdot (-\frac{1}{3}) \quad !!! \text{ dělíme } 0$$

- Algoritmus Gaussovy eliminační metody pro tento příklad **není realizovatelný**.
- Snadno se přesvědčíme, že má daná soustava řešení  $x = 1, y = 2$  a  $z = 3$ .  
Gaussova eliminační metoda ale selhala.

Otázka 1: Pro jaké matice  $A$  má soustava  $Ax = b$  právě jedno řešení?

$\rightarrow$  matice  $A$  musí být **regulární**, tj.  
všechna vlastní čísla musí být různá od nuly  
jinak řečeno řádky matice  $A$  musí být lineárně nezávislé  
jinak řečeno sloupce matice  $A$  musí být lineárně nezávislé  
jinak řečeno  $\det A \neq 0$

**Poznámka:** Vlastní číslo matice  $A$  je číslo  $\lambda$  splňující rovnici  $Av = \lambda v$ , kde  $v$  je vlastní vektor odpovídající vlastnímu číslu  $\lambda$ . Číslo  $\lambda$  tedy určitým způsobem charakterizuje matici  $A$ .

Otázka 2: Pro jaké matice  $A$  je algoritmus Gaussovy eliminační metody realizovatelný?

$\rightarrow$  **Věta:** Je-li matice  $A$  **ostře diagonálně dominantní**, pak je algoritmus GEM realizovatelný.

**Poznámka:** Matice  $A = [a_{ij}]_{i,j=1,\dots,n}$  je **ostře diagonálně dominantní**, platí-li

$$|a_{ii}| > \sum_{j=1, j \neq i}^n |a_{ij}|$$

tj. **absolutní hodnota diagonálního prvku je větší než součet absolutních hodnot ostatních prvků v řádku.**

**Např.:**

$$A = \begin{bmatrix} 7 & 1 & 2 \\ 1 & 4 & 1 \\ 2 & 4 & 9 \end{bmatrix}$$

$\rightarrow$  **Věta:** Je-li matice  $A$  **symetrická a pozitivně definitní**, pak je algoritmus GEM realizovatelný.

**Poznámka:** Matice  $A$  je **symetrická**, platí-li pro její prvky

$$a_{ij} = a_{ji} \quad \forall i, j = 1, 2, \dots, n.$$

**Poznámka:** Matice  $A$  je **pozitivně definitní**, má-li všechna vlastní čísla kladná nebo jinak řečeno  $\forall x \neq 0: x^T Ax > 0$ .

**Poznámka:** Pro soustavu s maticí, která splňuje předpoklady některé z uvedených vět, je možné dopředu říci, že půjde řešit pomocí Gaussovy eliminační metody. Obráceně to ovšem neplatí, tj. není-li např. matice soustavy **ostře diagonálně dominantní**, ještě to obecně neznamená, že nepůjde pomocí Gaussovy eliminační metody řešit.

**Poznámka:** Abychom zaručili, že soustava půjde vyřešit pro libovolnou regulární matici, musíme algoritmus Gaussovy eliminační metody upravit. Zavedeme tzv. **výběr hlavního prvku (pivotaci)**.

**Poznámka:** **Pivot (hlavní prvek)** ... první nenulový prvek v daném řádku matice.

Příklad 2

Pomocí **GEM se sloupcovou pivotací** vyřešte soustavu rovnic z Příkladu 1, kde selhala klasická GEM, tj. řešíme soustavu  $Ax = b$ , kde

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 7 & 8 & 9 \end{bmatrix} \quad a \quad b = \begin{bmatrix} 14 \\ 25 \\ 50 \end{bmatrix}$$

Řešení

1. sloupec  
↓  
vyměň  $\left[ \begin{array}{ccc|c} 1 & 2 & 3 & 14 \\ 2 & 4 & 5 & 25 \\ 7 & 8 & 9 & 50 \end{array} \right] \sim \left[ \begin{array}{ccc|c} 7 & 8 & 9 & 50 \\ 2 & 4 & 5 & 25 \\ 1 & 2 & 3 & 14 \end{array} \right] \quad / \cdot (-\frac{7}{3}) \quad / \cdot (-\frac{1}{3})$   
2. sloupec  
↓  
není třeba měnit  $\rightarrow \left[ \begin{array}{ccc|c} 7 & 8 & 9 & 50 \\ 0 & \frac{12}{3} & \frac{17}{3} & \frac{25}{3} \\ 0 & \frac{8}{3} & \frac{10}{3} & \frac{48}{3} \end{array} \right] \quad / \cdot (-\frac{8}{12}) = -\frac{1}{3}$

$$\begin{pmatrix} 7 & 8 & 9 & 50 \\ 0 & \frac{12}{3} & \frac{17}{3} & \frac{25}{3} \\ 0 & 0 & \frac{1}{3} & \frac{3}{3} \end{pmatrix} \Rightarrow x = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

**Poznámky:**

- Při sloupcové pivotaci jsme postupně v každém sloupci (resp. jeho části pod diagonálou včetně) vybírali číslo, které bylo maximální v absolutní hodnotě a v případě, že toto číslo neleželo na diagonále, vyměnili jsme příslušné 2 rovnice. Dále jsme pokračovali jako v GEM bez pivotace, tj. nulovali jsme koeficienty pod diagonálou.
- Sloupcová pivotace není jediná možnost. Podobně můžeme vybírat i maximální prvek v absolutní hodnotě z příslušného řádku (resp. jeho části) a poté vyměnit příslušné sloupce. Pozor! Je ovšem třeba zaměřit i příslušné složky řešení  $x$ . V tomto případě hovoříme o **řádkové pivotaci**.
- Další možností je vybírat maximální prvek v absolutní hodnotě z celé matice  $A$  (resp. příslušné podmatice). V tomto případě hovoříme o **úplně pivotaci**. Opět je třeba mít na paměti, že je třeba zaměřit složky ve vektoru řešení. Nevýhodou úplné pivotace je pomalejší výpočet neboť hlavní prvek vyhledáváme z celé dosud neupravené části.
- Libovolnou pivotací dosáhneme realizovatelnosti GEM pro libovolnou regulární matici  $A$ .

Metoda LU-rozkladu

Opět uvažujeme regulární matici  $A$  řádu  $N$ . Matici  $A$  lze rozložit na součin  **$A = LU$** , kde  $L$  je dolní trojúhelníková matice řádu  $N$  a  $U$  je horní trojúhelníková matice řádu  $N$ .

**Např.:**

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \cdot \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

Tento rozklad není dán jednoznačně (12 neznámých a 9 podmínek), jednoznačnosti dosáhneme např. tím, že položíme  **$l_{ii} = 1, i = 1, 2, \dots, N$** .

**Algoritmus:** (viz skriptu)

je odvozen z postupného násobení řádků matice  $L$  a sloupců matice  $U$

$$\begin{aligned} (1, 1) \quad u_{11} &= a_{11} & (2, 1) \quad a_{21} &= l_{21} u_{11} & (3, 1) \quad a_{31} &= l_{31} u_{11} \\ (1, 2) \quad u_{12} &= a_{12} & (2, 2) \quad a_{22} &= l_{21} u_{12} + u_{22} & (3, 2) \quad a_{32} &= l_{31} u_{12} + l_{32} u_{22} \\ (1, 3) \quad u_{13} &= a_{13} & (2, 3) \quad a_{23} &= l_{21} u_{13} + u_{23} & (3, 3) \quad a_{33} &= l_{31} u_{13} + l_{32} u_{23} + u_{33} \end{aligned}$$

**Řešení soustavy  $Ax = b$  metodou LU-rozkladu:**

1. Realizace LU-rozkladu:  **$A = LU$**

2. Řešení trojúhelníkové soustavy:  **$Ly = b$**

3. Řešení trojúhelníkové soustavy:  **$Ux = y$**

$$\boxed{L U x = b}$$

Souvislost GEM a metody LU-rozkladu

Gaussovu eliminaci lze popsat pomocí násobení regulárními maticemi.

První fázi popíšeme takto ... A^(1) = M1\*A, kde M1 = [1 m21 1 m31 1 m41 1 ... mn1 1]

Matricové vyjádření první fáze: [a11 a12 a13 ... a1n; m21\*a11+a21 m21\*a12+a22 ... m21\*a1n+a2n] = [1 m21 1 ... mn1 1] \* [a11 a12 a13 ... a1n; a21 a22 a23 ... a2n; ...]

Druhou fázi popíšeme takto ... A^(2) = M2\*A^(1), kde M2 = [1 m32 1 ... mn2 1]

Nakonec (n-1) fázi popíšeme ... A^(n-1) = Mn-1\*A^(n-2), kde Mn-1 = [1 1 1 ... mn,n-1 1]

Dostali jsme horní trojúhelníkovou matici, označíme ji např. V

V = A^(n-1) = Mn-1 \* Mn-2 \* Mn-3 \* ... \* M2 \* M1 \* A => A = M^-1 \* V

- (\*) dolní trojúhelníková s 1 na diagonále
(\*\*) horní trojúhelníková

=> jedná se o LU-rozklad (rozklad je jednoznačný) L = M^-1 a U = V

Výpočet determinantů

1. Užití GEM

U = Mn-1 \* Mn-2 \* ... \* M2 \* M1 \* A

det U = det Mn-1 \* det Mn-2 \* ... \* det M2 \* det M1 \* det A

det A = det U = prod\_{i=1}^n u\_ii

2. Užití LU-rozkladu

det A = det L \* det U = prod\_{i=1}^n u\_ii

Výpočet inverzní matice

1. Užití GEM

A \* X = I (maticová soustava)
X = A^-1

2. Užití LU-rozkladu

A = L \* U
A^-1 = U^-1 \* L^-1

Numerické aspekty GEM a metody LU-rozkladu

Při numerické realizaci nevypočteme přesné matice L a U, ale přibližné matice L-tilde a U-tilde. Teoreticky platí A = LU. Označíme A-tilde = L-tilde \* U-tilde ... dopočteno pro získané matice L-tilde a U-tilde. Budeme zkoumat rozdíl A-tilde - A.

Označme E a F matice chyb takové, že platí:

L-tilde = L + E, U-tilde = U + F

Potom:

A-tilde - A = L-tilde \* U-tilde - LU = (L + E)(U + F) - LU = EU + LF + EF

M^-1 = M1^-1 \* M2^-1 \* ... \* Mn-1^-1

Jak vypadá např. M2^-1 ?

M2 = [1 m32 1 ... mn2 1] -> M2^-1 = [1 0 0 ... 0; -m32 1 0 ... 0; -m42 0 1 ... 0; ...; -mn2 0 0 ... 1]

protože po vynásobení:

- z-tý řádek x j-tý sloupec (j != i)

- buď m42 \* 1 + 1 \* (-m42) = 0

- nebo m42 \* 0 + 0 \* 1 + 1 \* 0 = 0

- z-tý řádek x i-tý sloupec

- m42 \* 0 + 1 \* 1 = 1

tj. M2 \* M2^-1 = I

Jak vypadá M^-1 ?

M^-1 = [1 -m21 1 -m31 -m32 1 ... -mn1 -mn2 -mn3 ... 1]

Př.:

[1 0 0 | 1 0 0] [-m21 1 0 | 0 1 0] [-m31 0 1 | 0 -m32 1] = [1 0 0 | 1 0 0] [-m21 1 0 | 0 1 0] [-m31 -m32 1 | -m31 -m32 1]

Platí

A = M^-1 \* V (\*\*) (\*\*) (\*)

- (\*) dolní trojúhelníková s 1 na diagonále
(\*\*) horní trojúhelníková

Ódtud plyne závěr: Pokud jsou multiplikátory v absolutní hodnotě velké, pak prvky L jsou v absolutní hodnotě velké => chyba může být velká. Toto je jeden z důvodů realizace pivotace.

Přímé metody pro soustavy se speciální matricí

Uvažujeme matice:

- symetrická
symetrická a pozitivně definitní
diagonálně dominantní
pásová

Platí: Je-li matice A symetrická a A^(k) jsou matice získané GEM v základní verzi, pak podmatice A^(k) jsou také symetrické.

př.:

[a b c; b d e; c e f] / (-b/a) ... / (-e/a)

[a b c; 0 d - e/a e - b/a; 0 e - b/a f - c/a]

Lze pak použít symetrickou verzi GEM a LU-rozkladu.

Je-li matice A navíc pozitivně definitní, pak lze realizovat Choleského metodu rozkladu

A = U^T \* U

Poznámka: V algoritmu je potřeba realizovat výpočet odmocnin a to lze pouze pro pozitivně definitní matice.

př.:

[a11 a12 a13; a21 a22 a23; a31 a32 a33] = [u11 0 0; u21 u22 0; u31 u32 u33] \* [u11 u12 u13; 0 u22 u23; 0 0 u33]

(1,1): a11 = u11^2

(2,1): a21 = u12 \* u11

(3,1): a31 = u13 \* u11

(1,2): a12 = u12 \* u11

(2,2): a22 = u22^2 + u23^2

(3,2): a32 = u13 \* u12 + u23 \* u22

(1,3): a13 = u13 \* u11

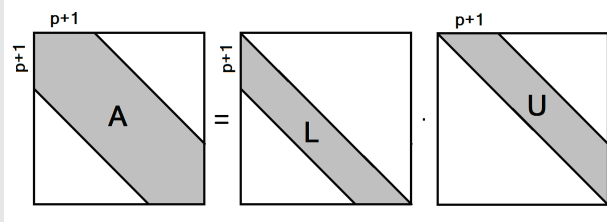
(2,3): a23 = u12 \* u13 + u22 \* u23

(3,3): a33 = u13^2 + u23^2 + u33^2

Metoda LU-rozkladu pro pásové matice

Uvažujeme matice A takovou, že [a\_ij = 0], když |i - j| > p (šířka pásu je 2p + 1).





Pokud lze realizovat LU-rozklad, pak

$$l_{ij} = 0, \text{ když } j > i \text{ a } j < i - p, \\ u_{ij} = 0, \text{ když } j < i \text{ a } j > i + p.$$

**Poznámka:** V případě obecné matice však nelze čekat, že matice L a U bude mít nulový prvek v téže pozici jako její matice A.

Pro pásové, symetrické, pozitivně definitní matice používáme **speciální verzi Choleského rozkladu**.

### Metoda faktorizace pro třídiagonální matici

Uvažujeme soustavu  $n + 1$  lineárních algebraických rovnic  $A \cdot Y = F$  ve tvaru:

$$\begin{pmatrix} c_0 & -b_0 & & & & & & & & & f_0 \\ -a_1 & c_1 & -b_1 & & & & & & & & f_1 \\ & -a_2 & c_2 & -b_2 & & & & & & & f_2 \\ & & -a_3 & c_3 & -b_3 & & & & & & f_3 \\ & & & & & \ddots & & & & & \vdots \\ & & & & & & -a_{n-1} & c_{n-1} & -b_{n-1} & & f_{n-1} \\ & & & & & & & -a_n & c_n & & f_n \end{pmatrix}$$

Označíme

$$\alpha_1 = \frac{b_0}{c_0}, \quad \beta_1 = \frac{f_0}{c_0}$$

První dvě rovnice (první rovnice je vydělena  $c_0$ ) lze psát ve tvaru:

$$\begin{aligned} y_0 - \alpha_1 y_1 &= \beta_1 \quad / \cdot a_1 \\ -a_1 y_0 + c_1 y_1 - b_1 y_2 &= f_1 \end{aligned} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} + \\ (c_1 - a_1 \alpha_1) y_1 - b_1 y_2 &= f_1 + a_1 \beta_1$$

Přepíšeme (rovnice vydělená koeficientem u  $y_1$ ) na tvar:

$$y_1 - \alpha_2 y_2 = \beta_2$$

kde

$$\alpha_2 = \frac{b_1}{c_1 - a_1 \alpha_1}, \quad \beta_2 = \frac{f_1 + a_1 \beta_1}{c_1 - a_1 \alpha_1}$$

Po zobecnění:

• **PŘÍMÝ CHOD**

$$\alpha_1 = \frac{b_0}{c_0}$$

$$\beta_1 = \frac{f_0}{c_0}$$

$$\alpha_{i+1} = \frac{b_i}{c_i - a_i \alpha_i} \quad i = 1, 2, \dots, n-1$$

$$\beta_{i+1} = \frac{f_i + a_i \beta_i}{c_i - a_i \alpha_i} \quad i = 1, 2, \dots, n$$

• **ZPĚTNÝ CHOD**

Pro poslední dvě rovnice získáme:

$$\begin{aligned} y_{n-1} - \alpha_n y_n &= \beta_n \quad / \cdot a_n \\ -a_n y_{n-1} + c_n y_n &= f_n \end{aligned} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} + \\ (c_n - a_n \alpha_n) y_n &= f_n + a_n \beta_n$$

♣ ve druhé rovnici již není člen  $-b_n y_{n+1}$

Vyjádříme poslední složku řešení:

$$y_n = \frac{f_n + a_n \beta_n}{c_n - a_n \alpha_n} =: \beta_{n+1}$$

Zpětně dosazujeme:

$$y_{i-1} = \beta_i + \alpha_i y_i \quad i = n, n-1, \dots, 1 \quad (*)$$

**Efektivnost algoritmu**

dělení	$2n+1$	$(1+n-1+1+n)$
násobení	$3n$	$(n+n+n)$
sčítání a odčítání $\pm$	$3n$	$(n+n+n)$
<b>celkem</b>	<b><math>(8n+1)</math> operací</b>	

**Poznámka:** Pokud budeme řešit tuto soustavu pro různé pravé strany F, nemusíme již znovu vyjadřovat koeficienty  $\alpha_i$ , protože nezávisí na F. Stačí tedy přepočítat  $\beta_i$ .

Zatím jsme neuvedli předpoklady pro metodu faktorizace

- je třeba zajistit, aby jmenovatel  $c_i - a_i \alpha_i$  byl nenulový pro  $i = 1, 2, \dots, n$
- $y_i$  se určuje z rekurentní formule (\*), přitom může dojít k akumulaci zaokrouhlovacích chyb.

Nechť  $\alpha_i, \beta_i$  jsou dokonce přesně vypočítané a nechť máme  $\tilde{y}_n = y_n + \varepsilon_n$  (s chybou  $\varepsilon_n$ ).

Potom postupně vypočítáme podle (\*)

$$\tilde{y}_{i-1} = \beta_i + \alpha_i \tilde{y}_i, \quad i = n, n-1, \dots, 1$$

Označíme-li  $\varepsilon_i = \tilde{y}_i - y_i$  chybu, bude jistě splňovat homogenní rovnici

$$\varepsilon_{i-1} = \alpha_i \varepsilon_i, \quad i = n, n-1, \dots, 1$$

(protože přesné hodnoty  $y_i$  splňují  $y_{i-1} = \beta_i + \alpha_i y_i$ )

⇒ Pokud by byly koeficienty  $|\alpha_i| > 1$ , dojde k velkému nárůstu chyby  $\varepsilon_0$  !!!

Pro  $|\alpha_i| \leq 1, i = 1, 2, \dots, n$  je algoritmus **stabilní**.

Postažující podmínky pro zajištění (●): Matice soustavy je **ostře diagonálně dominantní**.

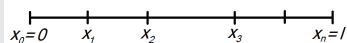
**Důkaz** viz literatura.

### Příklady aplikací, které vedou na soustavu s třídiagonální maticí

1. Řešení okrajové úlohy

$$\begin{aligned} (k(x)u'(x))' - q(x)u(x) &= -f(x) \quad x \in (0, l) \\ u(0) &= 0 \\ u(l) &= 0 \\ k(x) &> 0 \\ q(x) &\geq 0 \end{aligned}$$

$x$  na  $(0, l)$  diskretizujeme ... síť  $x_i$

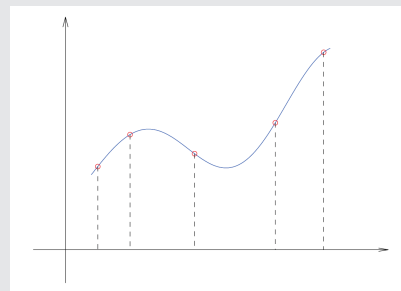


původní úlohu nahradíme úlohou s **diferenční** rovnicí a použijeme vzorec pro poměrnou diferenci

2. Diferenční schémata pro rovnici vedené tepla

$$\begin{aligned} \frac{\partial u}{\partial t} &= \frac{\partial^2 u}{\partial x^2} \quad x \in (0, l), \quad t > 0 \\ u(0, t) &= \mu_1(t) \\ u(l, t) &= \mu_2(t) \\ u(x, 0) &= u_0(x) \end{aligned}$$

3. Soustava pro výpočet koeficientů kubického spline (aproximace funkce) ... budeme probírat



### Podmíněnost úlohy řešit SLAR

Uvažujeme opět soustavu  $Ax = b$ ,  $A \dots n \times n$  regulární,  $b \in \mathbb{R}^n$ .

**Označení:**

$\Delta A \dots$  malá změna matice A

$\Delta b \dots$  malá změna vektoru b

$\Delta x \dots$  odpovídající změna vektoru neznámých

$x^* \dots$  přesné řešení soustavy  $Ax = b$

Platí:

$$(A + \Delta A)(x^* + \Delta x) = b + \Delta b$$

1. Uvažujme situaci  $\Delta A = 0$ , tj. A je zadána přesně

**Otázka:** Jakou změnu řešení vyvolá změna pravé strany?

$$\begin{aligned} Ax^* + A\Delta x &= b + \Delta b \\ A\Delta x &= \Delta b \end{aligned}$$

$$\Delta x = A^{-1} \Delta b$$

Z vlastností maticové normy plyne:

$$Ax^* = b \Rightarrow \|b\| \leq \|A\| \cdot \|x^*\| \Rightarrow \frac{1}{\|x^*\|} \leq \frac{\|A\|}{\|b\|}$$

$$\Delta x = A^{-1} \Delta b \Rightarrow \|\Delta x\| \leq \|A^{-1}\| \cdot \|\Delta b\|$$

$$\frac{\|\Delta x\|}{\|x^*\|} \leq \frac{\|A^{-1}\| \cdot \|\Delta b\| \cdot \|A\|}{\|b\|}$$

$$C_p = \frac{\|\Delta x\|}{\|x^*\|} \leq \|A^{-1}\| \cdot \|A\|$$

2. Příklad, kdy  $\Delta b = 0$ , tj.  $b$  je zadána přesně

$$(A + \Delta A)(x^* + \Delta x) = b$$

$$Ax^* + \Delta Ax^* + A\Delta x + \Delta A\Delta x = b$$

$$A\Delta x = -\Delta A(x^* + \Delta x)$$

$$\Delta x = -A^{-1} \Delta A(x^* + \Delta x)$$

$$\|\Delta x\| \leq \|A^{-1}\| \cdot \|\Delta A\| \cdot \|x^* + \Delta x\| \cdot \frac{\|A\|}{\|A\|}$$

$$\frac{\|\Delta x\|}{\|x^* + \Delta x\|} \leq \frac{\|A^{-1}\| \cdot \|\Delta A\|}{\|A\|} \geq C_p$$

$$C_p = \frac{\|\Delta x\|}{\|x^* + \Delta x\|} \leq \|A^{-1}\| \cdot \|A\|$$

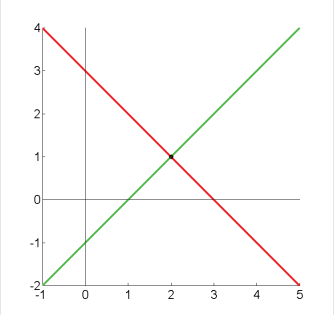
3. Rozmyslete obecný případ  $\Delta b \neq 0$ ,  $\Delta A \neq 0$  viz skripta (D.cv.)

**Poznámka:** Pro symetrické matice je číslo podmíněnosti podíl největší a nejmenší absolutní hodnoty vlastního čísla.

$$C_p = \frac{\|A^{-1}\| \cdot \|A\|}{\lambda_{\min}} = \frac{\lambda_{\max}}{\lambda_{\min}}$$

Geometrická interpretace - dobře podmíněná úloha (2D)

$$\begin{cases} x + y = 3 \\ x - y = 1 \end{cases} \Rightarrow \begin{cases} y = 3 - x \\ y = x - 1 \end{cases} \text{ řešení } \begin{cases} x^* = 2 \\ y^* = 1 \end{cases}$$



$$A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

Dobře podmíněná úloha - malá relativní změna vstupních dat vyvolá malou relativní změnu výstupních dat.

$$C_p = \|A^{-1}\| \cdot \|A\|$$

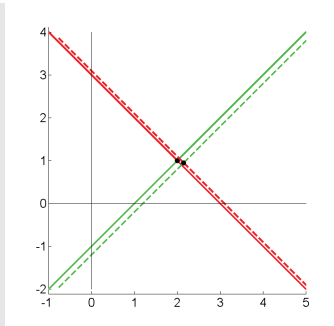
$$A^{-1} = \begin{bmatrix} 0,5 & 0,5 \\ 0,5 & -0,5 \end{bmatrix} = 0,5 A$$

$$C_p = 1 \cdot 2 = 2 \text{ (řádková, sloupcová norma); } C_p = \frac{1}{\sqrt{2}} \sqrt{2} = 1 \text{ (spektrální norma)}$$

$$(A^T A = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \quad \|A\|_{SP} = \sqrt{2}, \quad \|A^{-1}\|_{SP} = \frac{1}{\sqrt{2}})$$

1. změna pravé strany (změna matice  $\Delta A = 0$ )

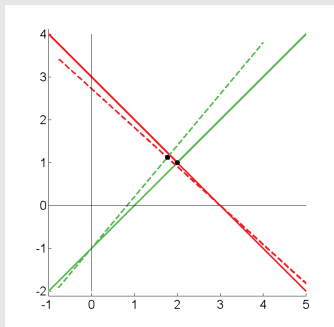
$$\Delta b = \begin{bmatrix} 0,1 \\ 0,2 \end{bmatrix}, \quad b + \Delta b = \begin{bmatrix} 3,1 \\ 1,2 \end{bmatrix}$$



změněná soustava  $y = 3,1 - x; y = x - 1,2$

2. změna matice soustavy (změna pravé strany  $\Delta b = 0$ )

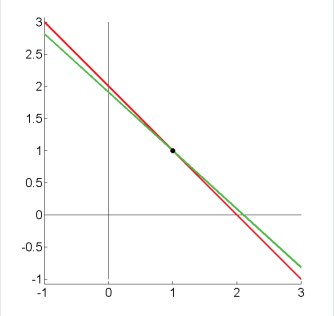
$$\Delta A = \begin{bmatrix} 0 & 0,1 \\ 0,2 & 0 \end{bmatrix}, \quad A + \Delta A = \begin{bmatrix} 1 & 1,1 \\ 1,2 & -1 \end{bmatrix}$$



změněná soustava  $y = \frac{1}{1,1}(3 - x); y = 1,2x - 1$

Geometrická interpretace - špatně podmíněná úloha (2D)

$$\begin{cases} x + y = 2 \\ x + 1,1y = 2,1 \end{cases} \Rightarrow \begin{cases} y = 2 - x \\ y = \frac{1}{1,1}(2,1 - x) \end{cases} \text{ řešení } \begin{cases} x^* = 1 \\ y^* = 1 \end{cases}$$



$$A = \begin{bmatrix} 1 & 1 \\ 1,1 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 2,1 \end{bmatrix}$$

Špatně podmíněná úloha - malá relativní změna vstupních dat vyvolá velkou relativní změnu výstupních dat.

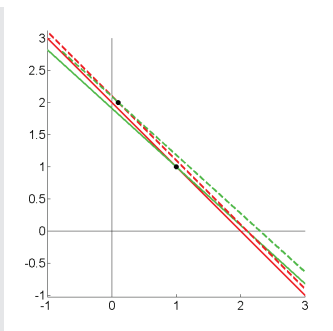
$$C_p = \|A^{-1}\| \cdot \|A\|$$

$$A^{-1} = \begin{bmatrix} 11 & -10 \\ -10 & 10 \end{bmatrix}$$

$$C_p = 2,1 \cdot 21 = 44,1 \text{ (řádková, sloupcová norma); } C_p = 2,0512 \cdot 20,5125 = 42,07 \text{ (spektrální norma)}$$

1. změna pravé strany (změna matice  $\Delta A = 0$ )

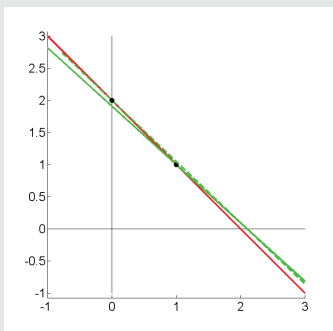
$$\Delta b = \begin{bmatrix} 0,1 \\ 0,2 \end{bmatrix}, \quad b + \Delta b = \begin{bmatrix} 2,1 \\ 2,3 \end{bmatrix}$$



změněná soustava  $y = 2,1 - x; y = \frac{1}{13}(2,3 - x)$

2. změna matice soustavy (změna pravé strany  $\Delta b = 0$ )

$$\Delta A = \begin{bmatrix} 0 & 0 \\ 0 & -0,05 \end{bmatrix}, \quad A + \Delta A = \begin{bmatrix} 1 & 1 \\ 1 & 1,05 \end{bmatrix}$$



změněná soustava  $y = 2 - x; y = \frac{1}{13}(2,1 - x)$

Poznámka: Jiný výpočet čísla podmíněnosti (prakticky)

$$Ax = b$$

změna na vstupu ...  $\Delta A$

vyvolá změnu na výstupu ...  $\Delta x$

$$(A + \Delta A)(x + \Delta x) = b$$

$$C_p = \frac{\frac{\|\Delta x\|}{\|x\|}}{\frac{\|\Delta A\|}{\|A\|}}$$

Příklad

$$A = \begin{bmatrix} 50 & -100 \\ 50 & -101 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \quad x = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

změna matice soustavy

$$\Delta A = \begin{bmatrix} 0 & 0,1 \\ 0,2 & 0 \end{bmatrix} \Rightarrow \tilde{A} = A + \Delta A = \begin{bmatrix} 50 & -99,9 \\ 50,2 & -101 \end{bmatrix}$$

$$\tilde{x} = \tilde{A}^{-1}b = \begin{bmatrix} 2,8527 \\ 1,4278 \end{bmatrix}$$

$$\Delta x = x - \tilde{x} = \begin{bmatrix} 2 \\ 1 \end{bmatrix} - \begin{bmatrix} 2,8527 \\ 1,4278 \end{bmatrix} = \begin{bmatrix} -0,8527 \\ -0,4278 \end{bmatrix}$$

$\ \cdot\ $	1.vektorová / sloupcová	maximová / řádková	euklidovská / spektrální
$\Delta x = \begin{bmatrix} -0,8527 \\ -0,4278 \end{bmatrix}$	1,2805	0,8527	0,9539
$x = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$	3	2	2,2361
$\Delta A = \begin{bmatrix} 0 & 0,1 \\ 0,2 & 0 \end{bmatrix}$	0,2	0,2	0,2
$A = \begin{bmatrix} 50 & -100 \\ 50 & -101 \end{bmatrix}$	201	151	158,7479
$C_p$	428,97	321,89	338,6

$$\|x\|_1 = \sum_i |x_i|, \quad \|x\|_\infty = \max_i |x_i|, \quad \|x\|_2 = \sqrt{\sum_i x_i^2}$$

$$\|A\|_s = \max_k \sum_i |a_{ik}|, \quad \|A\|_R = \max_k \sum_i |a_{ik}|, \quad \|A\|_{SP} = \max_k \lambda_k^2(A^H A)$$

(Při použití různých norem dostáváme obecně různá čísla podmíněnosti.)

$$C_p = \|A^{-1}\| \cdot \|A\|$$

$$A^{-1} = \begin{bmatrix} 2,02 & -2 \\ 1 & -1 \end{bmatrix}$$

$$C_p = 151 \cdot 4,02 = 607,02 \quad (\text{řádková } \|\cdot\|);$$

$$C_p = 201 \cdot 3,02 = 607,02 \quad (\text{sloupcová } \|\cdot\|);$$

$$C_p = 158,7479 \cdot 3,1750 \doteq 504,03 \quad (\text{spektrální } \|\cdot\|)$$

Pro špatně podmíněnou úlohu obsahuje  $A^{-1}$  velké prvky, které i pro malé hodnoty  $r$  mohou znamenat velkou chybu.

př:

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0,99999 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 1,99999 \end{bmatrix}$$

přesné řešení:  $x^* = [1, 1]^T$

vypočtené může být:  $x_c = [-98, 100]^T$

$$r = Ax_c - b = \begin{bmatrix} 2 \\ 1,999 \end{bmatrix} - \begin{bmatrix} 2 \\ 1,99999 \end{bmatrix} = \begin{bmatrix} 0 \\ -0,00099 \end{bmatrix}$$

$\Rightarrow$  je lepší pokoušet se odhadovat  $\|x_c - x^*\|$ .

Bohužel se v odhadech vždy vyskytuje norma  $\|A^{-1}\|$  !!! Podrobněji viz literatura.

Poznámky k podmíněnosti

- Stále předpokládáme, že  $A$  je regulární matice. Soustava  $Ax = b$  má potom právě jedno řešení. Předpokládáme, že je matice  $A$  normalizována tak, že její max. prvek v absolutní hodnotě je roven 1. Je-li soustava špatně podmíněná, potom matice  $A^{-1}$  musí obsahovat velké prvky.

např:

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0,99999 \end{bmatrix} \Rightarrow A^{-1} = \begin{bmatrix} -99999 & 100000 \\ 100000 & -100000 \end{bmatrix}$$

To odpovídá faktu, že číslo podmíněnosti je velké  $C_p = \|A\| \cdot \|A^{-1}\|$  (norma  $\|A^{-1}\|$  je velká).

Rozbor chyb

$$Ax = b, \quad x^* \dots \text{přesné}, \quad x_c \dots \text{vypočtené}$$

$\Rightarrow$  chybu můžeme měřit pomocí rezidia  $r = Ax_c - b$  (pro přesné řešení je  $r = Ax^* - b = 0$ )

Platí: Je-li  $x_c$  blízko  $x^* \Rightarrow$  reziduum je malé. Bohužel to neplatí obráceně !!!

$$r = A(x_c - x^*)$$

$$x_c - x^* = A^{-1}r$$



## Kapitola 4. SLAR - iterační metody

### Metody na řešení SLAR

- přímé (GEM, metoda LU-rozkladu) ✓
- iterační
- gradientní

**Iterační metody** najdou přesné řešení teoreticky až po nekonečně mnoha krocích.

Pamatujeme si, že v numerické praxi používáme pro řešení soustav s plnou maticí **přímé metody**, zatímco pro speciální (řídké) matice používáme **iterační metody**.

Toto rozdělení je dáno **výpočetní složitostí** těchto metod, tj. počtem matematických operací sčítání, odčítání, násobení a dělení nutných k získání výsledku.

**Poznámka:** V případě plné matice je výpočetní cena v každé iteraci řádu  $n^2$ , srovnáme-li toto s celkovou výpočetní cenou přímých metod, tj. řádově  $2/3 n^3$ , vidíme, že má-li být výpočetní složitost iterační metody stejná jako u přímé metody, musela by iterační metoda najít řešení (s předem zadanou přesností) řádově po  $n$  iteracích. Na druhou stranu v případě speciální (řídké) matice je výhodné použít iterační metodu.

### Příklad

Uvažujme rovnici

$$9x = 9$$

Přesné řešení je

$$x^* = 1$$

Rovnici lze přepsat např. na tvar

$$10x - x = 9$$

$$x = \frac{9 + x}{10}$$

- viz metoda prosté iterace pro nelineární rovnice
- nyní uvažujeme lineární rovnice, proto předpis funkce  $\varphi(x)$  může být lineární
- řešení hledáme pomocí rekurentní formule (volíme např.  $x^{(0)} = 0$ )

$$x^{(k+1)} = \frac{9 + x^{(k)}}{10}$$

Dostáváme

$$x^{(1)} = 0,9$$



$$x^{(2)} = 0,99$$

$$x^{(3)} = 0,999$$

$$x^{(4)} = 0,9999$$

Zastavíme např. pomocí podmínky na rozdíl dvou po sobě jdoucích iterací  $|x^{(4)} - x^{(3)}| < \varepsilon = 0,001$

Uvedený postup realizujeme pro soustavy.

Podobně jako v metodě prosté iterace pro nelineární soustavy přepíšeme soustavu

$$Ax - b = 0 \iff F(x) = 0$$

na tvar

$$x = Hx + g \iff x = \Phi(x)$$

Uvažujme-li soustavu lineárních algebraických rovnic, tj. funkce  $F$  je lineární, můžeme potom najít lineární předpis pro funkci  $\Phi$ .

Všechny iterační metody pro řešení soustavy lineárních algebraických rovnic budou používat iterační formuli

$$x^{(k+1)} = H \cdot x^{(k)} + g$$

samozejmě s různou iterační maticí  $H$  a vektorem  $g$  a je zřejmé, že o kvalitě metody rozhodují právě vlastnosti matice  $H$ .

Počáteční aproximací  $x^{(0)}$  zvolíme a výpočet ukončíme pomocí zastavovací podmínky

$$\|x^{(k)} - x^{(k-1)}\| < \varepsilon$$

### Jacobiova metoda

Princip:

Z  $i$ -té rovnice vyjádříme  $i$ -tou složku vektoru  $x$

$$i\text{-tá rovnice: } a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n = b_i$$

$$\text{pro } a_{ii} \neq 0: \quad x_i = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1, j \neq i}^n a_{ij}x_j \right)$$

Iterační formule:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1, j \neq i}^n a_{ij}x_j^{(k)} \right)$$



### Gausova-Seidelova metoda

Princip:

Stejný jako u Jacobiovy metody s tím rozdílem, že jestliže při výpočtu  $(k+1)$ -iterace již známe  $(k+1)$ -iteraci některých složek, tak ji použijeme.

Z  $i$ -té rovnice vyjádříme  $i$ -tou složku vektoru  $x$

$$i\text{-tá rovnice: } a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n = b_i$$

$$\text{pro } a_{ii} \neq 0: \quad x_i = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1, j \neq i}^n a_{ij}x_j \right)$$

Iterační formule:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right)$$



$$x_i^{(k+1)} = (1 - \omega)x_i^{(k)} + \omega \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right)$$

$x_i^{(k+1)}$  z G-S

**Poznámka:**  $(k+1)$ -iterace metody SOR je lineární kombinací  $(k+1)$ -iterace získané Gauss-Seidelovou metodou a předchozí  $k$ -té iterace metody SOR

$$x_i^{(k+1)} = \omega x_{iGS}^{(k+1)} + (1 - \omega)x_i^{(k)}$$

### Maticový zápis iteračních metod

Nejprve rozložíme matici  $A$ :

$$A = L + D + U$$

kde  $L$  je dolní trojúhelníková část matice  $A$  s nulami na diagonále,  $D$  je diagonální matice a  $U$  je horní trojúhelníková část matice  $A$  s nulami na diagonále.

#### Jacobiova metoda

$$\begin{aligned} Ax &= b \\ (L + D + U)x &= b \\ Dx + (L + U)x &= b \\ Dx &= b - (L + U)x \\ x &= \underbrace{-D^{-1}(L + U)}_{H_J} x + \underbrace{D^{-1}b}_{g_J} \end{aligned}$$

#### Gauss-Seidelova metoda

$$\begin{aligned} Ax &= b \\ (L + D + U)x &= b \\ (L + D)x + Ux &= b \\ (L + D)x &= b - Ux \\ x &= \underbrace{-(L + D)^{-1}U}_{H_{GS}} x + \underbrace{(L + D)^{-1}b}_{g_{GS}} \end{aligned}$$

#### Relaxační metoda SOR

### Relaxační metoda SOR

Princip:

Vyděme z Gausovy-Seidelovy metody její iterační formuli je

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right),$$

dále vyjádříme  $(k+1)$ -ní iteraci pomocí  $k$ -té iterace a příslušné změny (tj. přičteme a odečteme  $x_i^{(k)}$ )

$$x_i^{(k+1)} = x_i^{(k)} + \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right) - x_i^{(k)}$$

$= r_i^{(k)}$

k urychlení výpočtu použijeme ideu, že nepřičteme k předchozí iteraci změnu  $r_i^{(k)}$ , ale její násobek  $\omega r_i^{(k)}$

$$x_i^{(k+1)} = x_i^{(k)} + \omega \cdot \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right) - a_{ii}x_i^{(k)} = \sum_{j=1}^n a_{ij}x_j^{(k)}$$

Iterační formule:

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

$$Ax = b$$

$$\omega Ax = \omega b \quad / + Dx$$

$$(\omega A + D)x = \omega b + Dx$$

$$[\omega(L + D + U) + D]x = \omega b + Dx$$

$$(\omega L + D)x = \omega b + Dx - \omega Dx - \omega Ux$$

$$(\omega L + D)x = [(1 - \omega)D - \omega U]x + \omega b$$

$$x = \underbrace{(\omega L + D)^{-1}[(1 - \omega)D - \omega U]}_{\text{HSOR}} x + \underbrace{(\omega L + D)^{-1}\omega b}_{\text{GSOR}}$$

Iterační metoda je dána formulí

$$x^{(k+1)} = Hx^{(k)} + g$$

Pro přesné řešení  $x^*$  musí platit

- $x^* = Hx^* + g$
- $x^* = A^{-1}b$

$$\Rightarrow A^{-1}b = HA^{-1}b + g \quad (*)$$

**Definice:** Iterační metodu  $x^{(k+1)} = Hx^{(k)} + g$  nazveme **konzistentní**, pokud platí (\*).

**Poznámka:** Uvedené metody jsou konzistentní.

- např. pro Jacobiovu metodu musí platit:

$$A^{-1}b = \underbrace{-D^{-1}(L+U)}_{H_J} A^{-1}b + \underbrace{D^{-1}b}_{g_J}$$

$$A^{-1}b = D^{-1} \left( \underbrace{-(L+U)}_D + A \right) A^{-1}b$$

- D.cv. Ukažte, že Gauss-Seidelova metoda a metoda SOR jsou konzistentní.

**Definice:** Iterační metoda  $x^{(k+1)} = Hx^{(k)} + g$  se nazývá **konvergentní**, jestliže pro každou počáteční aproximaci  $x^{(0)} \in \mathbb{R}^n$  platí:

$$\lim_{k \rightarrow \infty} x^{(k)} = x^* \quad (= A^{-1}b)$$

Chyba  $k$ -té iterace:

$$e^{(k)} = x^{(k)} - x^*$$

### Nutná a postačující podmínka konvergence metody

iterační předpis

$$x^{(k)} = Hx^{(k-1)} + g$$

konzistentní metoda

$$x^* = Hx^* + g$$

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

po odečtení dostáváme

$$x^{(k)} - x^* = H(x^{(k-1)} - x^*)$$

$$e^{(k)} = He^{(k-1)}$$

Platí:

$$e^{(k)} = He^{(k-1)} = H^2e^{(k-2)} = \dots = H^k e^{(0)}$$

$$\lim_{k \rightarrow \infty} x^{(k)} = x^* \Leftrightarrow \lim_{k \rightarrow \infty} e^{(k)} = 0$$

**Věta:** Daná konzistentní iterační metoda  $x^{(k+1)} = Hx^{(k)} + g$  konverguje pro libovolné  $x^{(0)} \in \mathbb{R}^n$  právě tehdy, když je **stabilní**, tj.

$$\rho(H) = \max_i |\lambda_i(H)| < 1$$

kde číslo  $\rho(H)$  nazýváme **spektrální poloměr** matice  $H$  a  $\lambda_i(H)$  jsou vlastní čísla matice  $H$ .

**Poznámka:** Připomeňme souvislost s metodou prosté iterace pro řešení soustav nelineárních rovnic. Funkce  $\Phi(x)$  z přepisu  $x = \Phi(x)$  musela splňovat podmínku ( $\Phi'$ ) ... (pokud byla diferencovatelná)

$$\exists q \in (0, 1): \|\Phi'(x)\| \leq q \quad \forall x$$

V našem případě je

$$\Phi(x) = Hx + g \Rightarrow \Phi'(x) = H.$$

Tj.

$$\|H\| < 1.$$

Spektrální poloměr  $\rho(H)$  je také normou matice  $H$ , tj.  $\rho(H) < 1$

Předchozí věta je silnější (kritérium)

$\Leftrightarrow$

Věta pro prostou iteraci uváděla postačující podmínky

$\Rightarrow$

**Poznámka:** Určovat  $\rho(H)$  je celkem drahé, proto za chvíli uvedeme větu (postačující podmínky) jejíž předpoklady se ověří snadněji.

**Definice:**

Maticovou normu  $\|\cdot\|$  nazveme **multiplicativní**, splňuje-li pro všechny čtvercové matice  $A, B$  řádu  $n$  vztah

$$\|A \cdot B\| \leq \|A\| \cdot \|B\|$$

### Příklad 2

Řešme soustavu  $Ax = b$  Jacobiovou metodou.

$$A = \begin{bmatrix} 1 & 0,9 & 0,9 \\ 0,9 & 1 & 0,9 \\ 0,9 & 0,9 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2,8 \\ 2,8 \\ 2,8 \end{bmatrix}, \quad \text{přesné řešení } x^* = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

$$H = \begin{bmatrix} 0 & -0,9 & -0,9 \\ -0,9 & 0 & -0,9 \\ -0,9 & -0,9 & 0 \end{bmatrix}$$

... vlastní čísla jsou  $-1, 8; 0,9; 0,9 \Rightarrow \rho(H) = 1,8 > 1 \Rightarrow$  metoda diverguje !!!

Uvažujme normu  $\|H\|_M = \max_{i,j} |h_{ij}|$

D.cv. Ukažte, že  $\|\cdot\|_M$  splňuje vlastnosti normy.

$$\|H\|_M = 0,9 < 1 \quad !!!$$

$\|\cdot\|_M$  není multiplicativní:

např:

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}, \quad A \cdot B = \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix}$$

$$\|A\|_M = 1, \quad \|B\|_M = 2, \quad \|A \cdot B\|_M = 4$$

$$\|A\|_M \cdot \|B\|_M = 2 < \|A \cdot B\|_M = 4$$

**Věta:** Pro každou multiplicativní maticovou normu  $\|\cdot\|$  a čtvercovou matici  $A$  platí:

$$\rho(A) \leq \|A\|$$

**Důkaz:**

$$\rho(A) = \max_i |\lambda_i| = |\lambda_p|$$

- necht čísla  $\lambda_p$  odpovídá normovaný vlastní vektor  $v_p$  ( $v(v_p) = 1$ )
- potom

$$\rho(A) = |\lambda_p| \cdot v(v_p) = v(\lambda_p v_p) = v(A v_p)$$

vlastnost kompatibilní maticové a vektorové normy:

$$v(A v_p) \leq \|A\| \cdot v(v_p) = \|A\| \cdot 1 = \|A\|$$

**Pomocná věta:** Ke každé multiplicativní maticové normě  $\mu$  existuje kompatibilní vektorová norma  $\nu$ .

**Důkaz:** Je dána maticová norma  $\mu$ .

Definujeme  $\nu(x) = \mu([x, 0, 0, \dots, 0])$  ... splňuje vlastnosti normy

? kompatibilita:

**Numerické metody** KMA/NM  
13.2.2013  
Josef Daněk

$$\nu(Ax) = \mu([Ax, 0, 0, \dots, 0]) = \mu(A[x, 0, 0, \dots, 0]) \stackrel{(*)}{\leq} \mu(A)\mu([x, 0, 0, \dots, 0]) = \mu(A) \cdot \nu(x)$$

(\*)  $\mu$  je multiplicativní

**Věta** (Postačující podmínka konvergence)

Je-li pro multiplicativní normu splněna podmínka  $\|H\| \leq q < 1$ , potom posloupnost  $\{x^{(k)}\}$  určená konzistentní formulí

$$x^{(k)} = Hx^{(k-1)} + g$$

konverguje při libovolné volbě vektoru  $x^{(0)} \in \mathbb{R}^n$  a platí

$$\lim_{k \rightarrow \infty} x^{(k)} = (I - H)^{-1}g = x^*$$

**Důkaz:** Důsledek kritéria a předchozí věty (jiný důkaz viz skripta).

### Odhad chyby

Předpokládáme, že je splněna postačující podmínka konvergence

$$\|H\| \leq q < 1$$

$$\begin{aligned} x^{(k)} - x^* &= H(x^{(k-1)} - x^*) = \\ &= H(x^{(k-1)} - x^{(k)} + x^{(k)} - x^*) = \\ &= H(x^{(k-1)} - x^{(k)}) + H(x^{(k)} - x^*) \end{aligned}$$

Odtud dostáváme

$$\|x^{(k)} - x^*\| \leq q \|x^{(k-1)} - x^{(k)}\| + q \|x^{(k)} - x^*\|$$

tj.

$$\underbrace{(1-q)}_{>0} \|x^{(k)} - x^*\| \leq q \|x^{(k)} - x^{(k-1)}\|$$

a po vydělení

$$\|x^{(k)} - x^*\| \leq \frac{q}{1-q} \|x^{(k)} - x^{(k-1)}\|$$

Jestliže  $\|x^{(k)} - x^{(k-1)}\| < \varepsilon$ , potom

$$\|x^{(k)} - x^*\| \leq \frac{q}{1-q} \varepsilon$$

**Příklad 3**

Jacobiovou metodou řešte soustavu  $Ax = b$ .

$$A = \begin{bmatrix} 8 & 4 & 2 \\ 1 & 10 & 1 \\ 0 & 0 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 14 \\ 12 \\ 2 \end{bmatrix}, \quad x^{(0)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$A = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_L + \underbrace{\begin{bmatrix} 8 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 2 \end{bmatrix}}_D + \underbrace{\begin{bmatrix} 0 & 4 & 2 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_U$$

$$H_J = -D^{-1}(L + U) = \begin{bmatrix} 0 & -0,5 & -0,25 \\ -0,1 & 0 & -0,1 \\ 0 & 0 & 0 \end{bmatrix}, \quad g_J = D^{-1}b = \begin{bmatrix} 1,75 \\ 1,2 \\ 1 \end{bmatrix}$$

Provedeme 5 iterací:

k	$x_1^k$	$x_2^k$	$x_3^k$
0	0	0	0
1	1,75	1,2	1
2	0,9	0,925	1
3	1,0375	1,01	1
4	0,995	0,99625	1
5	1,001875	1,0005	1

$$\underbrace{x^{(5)} - x^{(4)}}_{= x^{(6)}} = \begin{bmatrix} 0,006875 \\ 0,004250 \\ 0,000000 \end{bmatrix}$$

Odhadněme chybu  $x^{(6)}$ , tj.

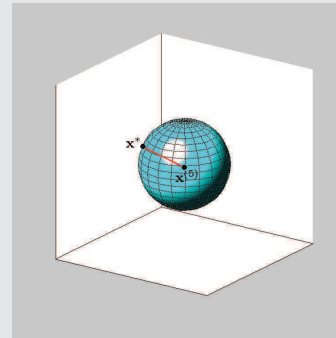
$$\|x^{(6)} - x^*\| \leq \frac{\|H\|}{1 - \|H\|} \cdot \|x^{(5)} - x^{(4)}\|$$

maticová norma	vektorová norma	odhad chyby
$\ H\ _S = \max_k(\sum_i h_{ik}) = 0,5$	$\ r^{(6)}\ _1 = \sum_i  r_i  = 0,011125$	$\frac{0,5}{1 - 0,5} \cdot 0,011125 = \mathbf{0,011125!}$
$\ H\ _R = \max_k(\sum_i h_{ik}) = 0,75$	$\ r^{(6)}\ _\infty = \max_k  r_k  = 0,006875$	$\frac{0,75}{0,25} \cdot 0,006875 = \mathbf{0,0206}$
$\ H\ _{SP} = \max_k \lambda_k^2(H^H H) \approx 0,56$	$\ r^{(6)}\ _2 = \sum_i r_i^2 \approx 0,0081$	$\frac{0,56}{0,44} \cdot 0,0081 = \mathbf{0,0103}$

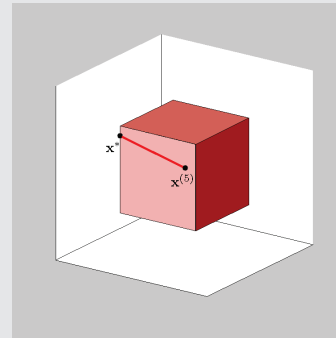
**Geometrický význam**

$$\|x^{(5)} - x^*\|_2 \leq 0,0103$$

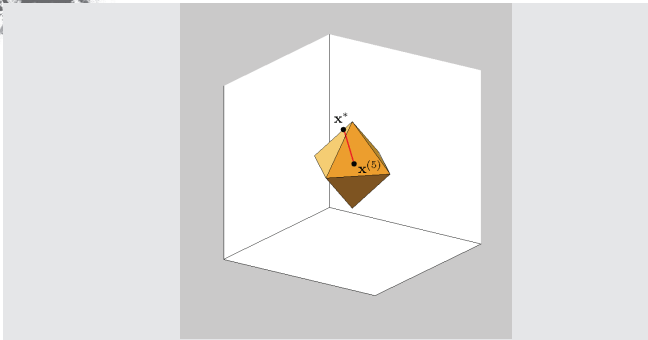
vzdálenost  $x^{(5)}$  a  $x^*$  je menší než vypočtená hodnota



$$\|x^{(5)} - x^*\|_\infty \leq 0,0206$$



$$\|x^{(5)} - x^*\|_1 \leq 0,011125$$



**Rychlost konvergence**

- Lineární rychlost konvergence

$$\exists q \in (0,1) \exists k_0 \geq 0 \forall k > k_0 : \|x^{(k+1)} - x^*\| \leq q \|x^{(k)} - x^*\|$$

- Superlineární rychlost konvergence

$$\|x^{(k+1)} - x^*\| \leq q_k \|x^{(k)} - x^*\|, \quad \text{kde } q_k \rightarrow 0 \text{ (} k \rightarrow \infty \text{)}$$

- Konvergence řádu r

$$\|x^{(k+1)} - x^*\| \leq q \|x^{(k)} - x^*\|^r$$

Poznámka:

- Jacobiova, Gauss-Seidelova i SOR metoda mají lineární rychlost konvergence

$$x^{(k+1)} - x^* = H(x^{(k)} - x^*)$$

během výpočtu se nemění iterační matice H, jedná se o **stacionární metody**

$$\|H\| \leq q, \quad \|H\| \dots \text{pevné číslo}$$

- Metody se superlineární rychlostí konvergence patří mezi **nestacionární procesy**

$$x^{(k+1)} = H_k x^{(k)} + g_k$$

V každém kroku se mohou měnit  $H_k, g_k$ .

Potom  $\|H_k\| \leq q_k$ , platí-li  $q_k \rightarrow 0$  pak jde o superlineární metodu.

Definujeme asymptotickou rychlost konvergence

$$R = -\log \frac{\|e^{(k)}\|}{\|e^{(k-1)}\|} \geq -\log \|H\|$$

ta určuje počet platných desetinných míst získaných v jednom iteračním kroku.

Prakticky:

$$\frac{\|e^{(k)}\|}{\|e^{(k-1)}\|} = \frac{\|x^{(k)} - x^*\|}{\|x^{(k-1)} - x^*\|} \approx \frac{\frac{q}{1-q} \|x^{(k)} - x^{(k-1)}\|}{\frac{q}{1-q} \|x^{(k-1)} - x^{(k-2)}\|}$$

**Poznámka:** Pro metody s lineární rychlostí konvergence (Jacobiova, Gauss-Seidelova, SOR metoda) lze pro urychlení použít **Aitkenovu extrapolaci formulí** (viz dříve).

Posloupnost chyb je geometrická

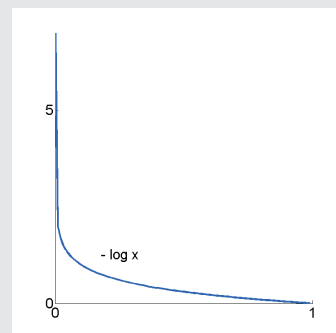
$$e^{(k)} = H e^{(k-1)}$$

$$\frac{e_2^{(k+1)}}{e_2^{(k)}} = \frac{e_2^{(k)}}{e_2^{(k-1)}} \approx \frac{e_1^{(k)}}{e_1^{(k-1)}} = \frac{e_1^{(k)}}{e_1^{(k-1)}}$$

po úpravě:

$$x_2^* \approx x_2^{(k+1)} - \frac{(x_2^{(k+1)} - x_2^{(k)})^2}{x_2^{(k+1)} - 2x_2^{(k)} + x_2^{(k-1)}}$$

Při odvození metody SOR jsme se pokusili urychlit výpočet změnou iterační matice H tak, aby měla menší spektrální poloměr  $\rho(H)$ . Čím menší je  $\rho(H)$ , tím větší asymptotická rychlost konvergence.



**Věta:** Spektrální poloměr  $\rho(\mathbf{H}_{SOR})$  splňuje podmínku

$$\rho(\mathbf{H}_{SOR}) \geq |\omega - 1| \quad \forall \omega \in \mathbb{R}$$

**Důkaz:**

$$\mathbf{H}_{SOR} = (\omega \mathbf{L} + \mathbf{D})^{-1} [(1 - \omega)\mathbf{D} - \omega \mathbf{U}]$$

Je známo, že součin vlastních čísel je roven determinantu

$$\prod_{i=1}^n \lambda_i = \det(\mathbf{H}_{SOR})$$

$$\det(\mathbf{H}_{SOR}) = \det[(\omega \mathbf{L} + \mathbf{D})^{-1} [(1 - \omega)\mathbf{D} - \omega \mathbf{U}]] = \det[(\mathbf{D}(\omega \mathbf{D}^{-1} \mathbf{L} + \mathbf{I}))^{-1} \mathbf{D} [(1 - \omega)\mathbf{I} - \omega \mathbf{D}^{-1} \mathbf{U}]] =$$

$$= \det[(\omega \mathbf{D}^{-1} \mathbf{L} + \mathbf{I})^{-1} [(1 - \omega)\mathbf{I} - \omega \mathbf{D}^{-1} \mathbf{U}]] = \det(\underbrace{(\omega \mathbf{D}^{-1} \mathbf{L} + \mathbf{I})^{-1}}_{(*)}) \cdot \det(\underbrace{[(1 - \omega)\mathbf{I} - \omega \mathbf{D}^{-1} \mathbf{U}]}_{(**)}) = (1 - \omega)^n$$

(\*) dolní trojúhelníková matice s 1 na diagonále

(\*\*) horní trojúhelníková matice a prvky  $(1 - \omega)$  na diagonále

$$\prod_{i=1}^n \lambda_i = (1 - \omega)^n \quad \Leftrightarrow \quad \max_i |\lambda_i| \geq |1 - \omega|$$

(\*\*\*) Důkaz sporem:  $\forall \lambda_i: |\lambda_i| < |1 - \omega| \quad / \prod_{i=1}^n$

$$\prod_{i=1}^n |\lambda_i| < |1 - \omega|^n$$

**Důsledek:** Aby SOR konvergovala, musí platit:

$$|\omega - 1| \leq \rho(\mathbf{H}_{SOR}) < 1$$

$$|\omega - 1| < 1 \Rightarrow \omega \in (0, 2)$$

**Poznámky:**

Parametr  $\omega$  v relaxační metodě SOR volíme z intervalu  $(0, 2)$ .

Pro  $\omega = 1$  přejde relaxační metoda na Gauss-Seidelovu metodu.

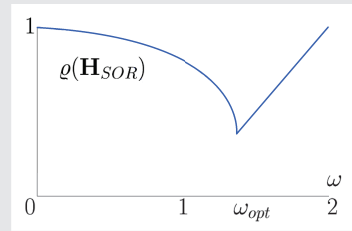
Volba parametru  $\omega$  samozřejmě ovlivní rychlost konvergence iteračního procesu metody SOR. Lze ukázat, že existuje optimální hodnota parametru omega

$$\omega_{opt} = \frac{2}{1 + \sqrt{1 - \rho^2(\mathbf{H}_J)}}$$

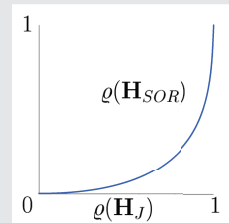
kde  $\rho(\mathbf{H}_J)$  je spektrální poloměr Jacobiovy iterační matice  $\mathbf{H}_J$ .

Pro spektrální poloměr iterační matice  $\mathbf{H}_{SOR}$  relaxační metody lze odvodit následující závislosti:

- závislost spektrálního poloměru iterační matice metody SOR na relaxačním parametru  $\omega$



- závislost spektrálního poloměru iterační matice metody SOR na spektrálním poloměru iterační matice Jacobiovy metody



**Konvergenční věty**

Dosud jsme udávali podmínky pro iterační matici  $\mathbf{H}$ . To je ovšem nepraktické. Uvedeme několik snadnější ověřitelných podmínek.

**Věta 1** Je-li matice  $\mathbf{A}$  ostře diagonálně-dominantní, potom konverguje Jacobiova i Gauss-Seidelova metoda pro libovolnou volbu  $\mathbf{x}^{(0)}$ .

**Věta 2** Je-li matice  $\mathbf{A}$  symetrická a pozitivně-definitní, potom Gauss-Seidelova metoda konverguje pro libovolnou volbu  $\mathbf{x}^{(0)}$ .

**Věta 3** Nutnou podmínkou konvergence metody SOR je  $0 < \omega < 2$ . Přidáme-li symetrii a pozitivní definitnost matice  $\mathbf{A}$ , dostaneme postačující podmínky konvergence.

Důkaz Věty 1 pro Jacobiovu metodu:

$\mathbf{Ax} = \mathbf{b}$ , rozklad matice  $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$   
označíme-li  $\mathbf{C} = \mathbf{L} + \mathbf{U}$ , potom  $\mathbf{A} = \mathbf{C} + \mathbf{D}$

- Jacobiova metoda

$$\mathbf{x}^{(k+1)} = \mathbf{H}_J \mathbf{x}^{(k)} + \mathbf{g}_J$$

$$\mathbf{H}_J = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U}) = -\mathbf{D}^{-1}\mathbf{C} \quad \mathbf{g}_J = \mathbf{D}^{-1}\mathbf{b}$$

- Matice  $\mathbf{A}$  je ostře diagonálně-dominantní, tj. platí

$$|a_{ii}| > \sum_{j=1, j \neq i}^n |a_{ij}| \quad i = 1, 2, \dots, n$$

- Pro náš rozklad  $\mathbf{A} = \mathbf{C} + \mathbf{D}$  tedy platí:

$$|d_{ii}| > \sum_{j=1}^n |c_{ij}| \quad i = 1, 2, \dots, n \quad /: |d_{ii}| \neq 0$$

(kdyby  $|d_{ii}| = 0$ , potom by byl celý řádek nulový ...  $\mathbf{A}$  je ale regulární)

- Platí:

$$\sum_{j=1}^n \frac{|c_{ij}|}{|d_{ii}|} < 1 \quad \text{pro } i = 1, 2, \dots, n \quad (*)$$

- Pro iterační matici platí:

$$\mathbf{H}_J = -\mathbf{D}^{-1}\mathbf{C} = - \begin{bmatrix} \frac{1}{d_{11}} & & \\ & \frac{1}{d_{22}} & \\ & & \ddots \\ & & & \frac{1}{d_{nn}} \end{bmatrix} \cdot \begin{bmatrix} c_{1j} \\ \vdots \\ c_{ij} \\ \vdots \\ c_{nj} \end{bmatrix} = - \begin{bmatrix} \frac{c_{1j}}{d_{11}} \\ \vdots \\ \frac{c_{ij}}{d_{ii}} \\ \vdots \\ \frac{c_{nj}}{d_{nn}} \end{bmatrix}$$

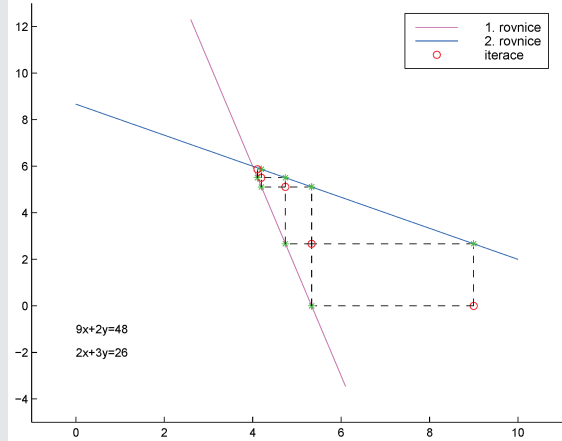
- Řádková norma matice  $\mathbf{H}_J$ :

$$\|\mathbf{H}_J\| = \max_i \sum_{j=1}^n \frac{|c_{ij}|}{|d_{ii}|} < 1 \quad \text{plyne z } (*)$$

**Geometrický význam Jacobiovy metody**

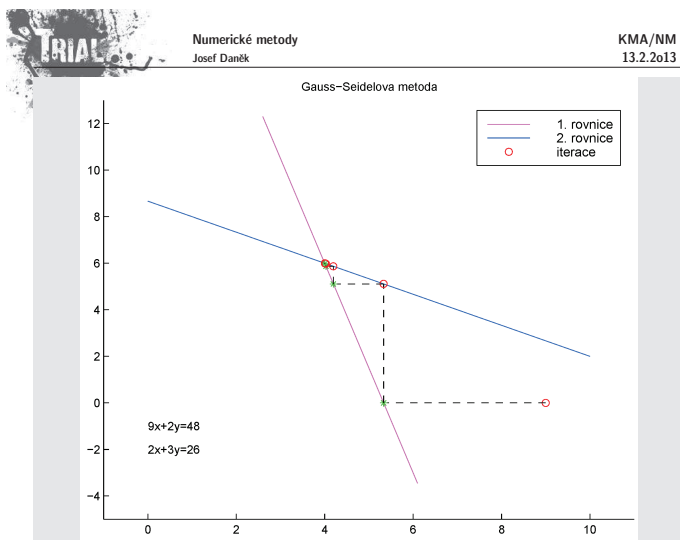
k	$x^{(k)}$	$y^{(k)}$
0	9.0000	0
1	5.3333	2.6667
2	4.7407	5.1111
3	4.1975	5.5062
4	4.1097	5.8683
5	4.0293	5.9268

**Jacobiova metoda**



**Geometrický význam Gauss-Seidelovy metody**

k	$x^{(k)}$	$y^{(k)}$
0	9.0000	0
1	5.3333	5.1111
2	4.1975	5.8683
3	4.0293	5.9805
4	4.0043	5.9971
5	4.0006	5.9996



#### Geometrický význam metody SOR ( $\omega < 1$ )

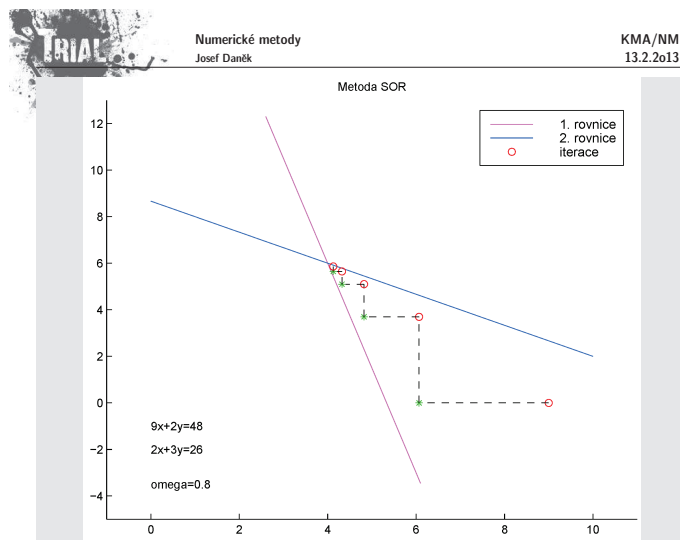
$$9x + 2y = 48 \quad x_{GS}^{(k+1)} = \frac{1}{9}(48 - 2y^{(k)})$$

$$2x + 3y = 26 \quad y_{GS}^{(k+1)} = \frac{1}{3}(26 - 2x^{(k+1)})$$

$$x^{(k+1)} = \omega \frac{1}{9}(48 - 2y^{(k)}) + (1 - \omega)x^{(k)}$$

$$y^{(k+1)} = \omega \frac{1}{3}(26 - 2x^{(k+1)}) + (1 - \omega)y^{(k)}$$

k	$x^{(k)}$	$y^{(k)}$
0	9.0000	0
1	6.0667	3.6978
2	4.8226	5.1008
3	4.3244	5.6472
4	4.1276	5.8614
5	4.0502	5.9455



#### Geometrický význam metody SOR ( $\omega > 1$ )

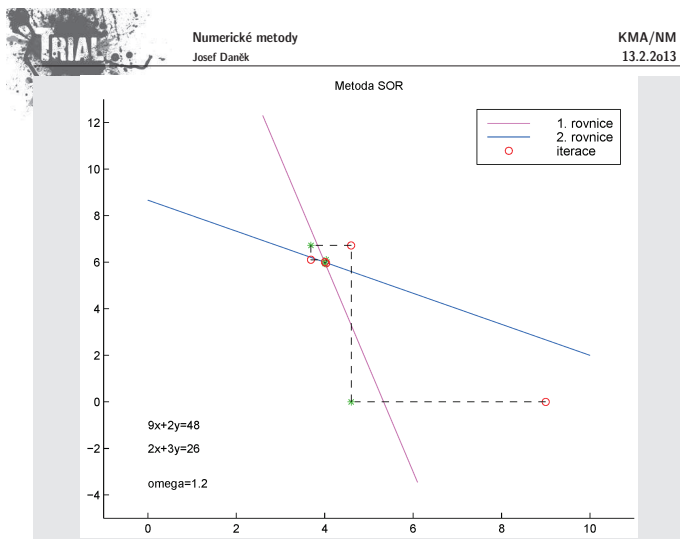
$$9x + 2y = 48 \quad x_{GS}^{(k+1)} = \frac{1}{9}(48 - 2y^{(k)})$$

$$2x + 3y = 26 \quad y_{GS}^{(k+1)} = \frac{1}{3}(26 - 2x^{(k+1)})$$

$$x^{(k+1)} = \omega \frac{1}{9}(48 - 2y^{(k)}) + (1 - \omega)x^{(k)}$$

$$y^{(k+1)} = \omega \frac{1}{3}(26 - 2x^{(k+1)}) + (1 - \omega)y^{(k)}$$

k	$x^{(k)}$	$y^{(k)}$
0	9.0000	0
1	4.6000	6.7200
2	3.6880	6.1056
3	4.0342	5.9515
4	4.0061	6.0048
5	3.9975	6.0010



## Kapitola 5. SLAR - gradientní metody

### Metody na řešení SLAR

- přímé (GEM, metoda LU-rozkladu) ✓
- iterační (Jacobiho m., Gauss-Seidelova m., metoda SOR) ✓
- gradientní

### Motivace

Uvažujme kvadratickou funkci reálné proměnné  $x$ :

$$f(x) = \frac{1}{2}ax^2 - bx + c, \quad a > 0.$$

Nutná a postačující podmínka minima funkce ( $f'(x) = 0$ ) má tvar  
 $ax = b$ .

To znamená, že místo řešení lineární rovnice můžeme řešit úlohu najít minimum konvexní kvadratické funkce  $f(x)$  (obě úlohy mají stejná řešení).

Uvědomme si, že v případě funkce více proměnných je třeba splnit další podmínky kladené na matici soustavy  $A$ , abychom zaručili konvexnost příslušné kvadratické funkce.

Uvažujme soustavu (kde matice  $A$  je symetrická, pozitivně definitní)

$$Ax = b$$

Dále uvažujme kvadratickou formu, tzv. **energetický funkcionál**

$$F(x) = \frac{1}{2}x^T Ax - b^T x.$$

Platí

$$\text{grad} F(x) = Ax - b.$$

Funkce  $F(x)$  je konvexní a kvadratická  $\Rightarrow F(x)$  má globální minimum a pro bod minima  $\bar{x}$  platí

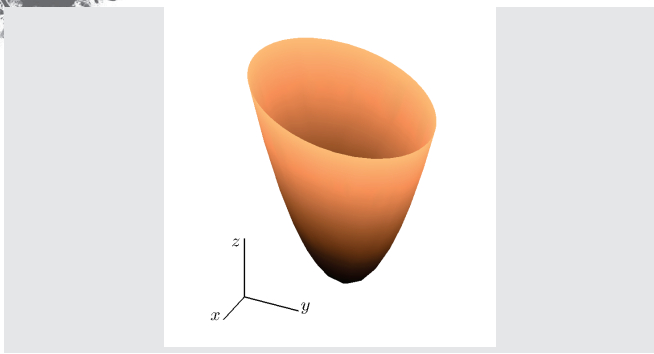
$$\text{grad} F(\bar{x}) = A\bar{x} - b = 0.$$

Bod minima  $\bar{x}$  je tedy řešením soustavy  $Ax = b$ .

**Poznámka:** Úlohy najít bod minima funkce  $F$  a řešit soustavu  $Ax = b$  jsou ekvivalentní.

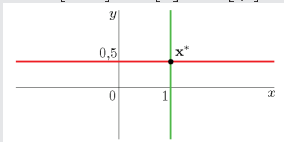
**Poznámka:** V případě soustavy 2 rovnic si lze udělat geometrickou představu, neboť pro  $x \in \mathbb{R}^2$  je grafem funkce  $F(x)$  eliptický paraboloid, jehož vrstevnice jsou elipsy. Minima  $F(x)$  se nabývá v vrcholu paraboloidu.





**Příklad 1**

Uvažujme velmi jednoduchou soustavu  $Ax = b$ , kde  $A = \begin{bmatrix} 25 & 0 \\ 0 & 16 \end{bmatrix}$ ,  $b = \begin{bmatrix} 25 \\ 8 \end{bmatrix}$ ,  $x^* = \begin{bmatrix} 1 \\ 0,5 \end{bmatrix}$ .



Odpovídající kvadratická funkce je

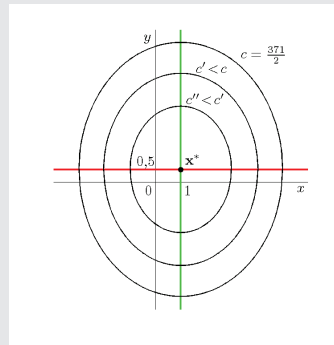
$$F(x) = \frac{1}{2}x^T Ax - b^T x = \frac{1}{2} \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 25 & 0 \\ 0 & 16 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} 25 & 8 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{2}(25x^2 + 16y^2) - 25x - 8y.$$

Vrstevnice (hladiny):

$$\begin{aligned} \boxed{F(x) = c} \\ \frac{1}{2}(25x^2 + 16y^2) - 25x - 8y = c \\ 25x^2 + 16y^2 - 50x - 16y = 2c \\ 25(x-1)^2 - 25 + 16(y-\frac{1}{2})^2 - 4 = 2c \\ 25(x-1)^2 + 16(y-\frac{1}{2})^2 = 2c + 29 \end{aligned}$$

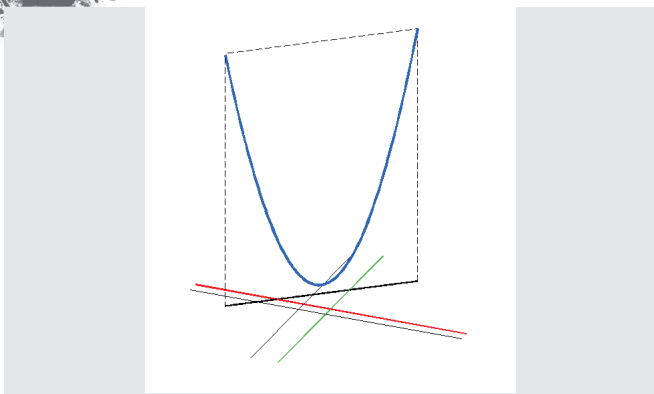
např. pro  $c = \frac{371}{2}$ :

$$\frac{(x-1)^2}{4^2} + \frac{(y-\frac{1}{2})^2}{5^2} = \frac{2c+29}{400} = 1$$



Řezy svislou rovinou  $y = px + q$

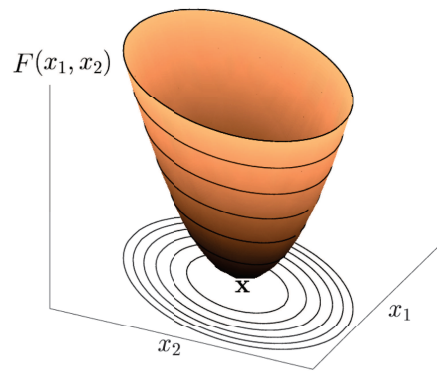
$$\begin{aligned} F(x) &= \frac{1}{2}(25x^2 + 16y^2) - 25x - 8y = \frac{1}{2}(25x^2 + 16(px+q)^2) - 25x - 8(px+q) = \\ &= \frac{1}{2}(25x^2 + 16(p^2x^2 + 2pqx + q^2)) - 25x - 8(px+q) = \\ &= \underbrace{\left(\frac{25}{2} + \frac{16}{2}p^2\right)}_{>0} x^2 + (16pq - 25 - 8p)x + 8q^2 - 8q \end{aligned}$$



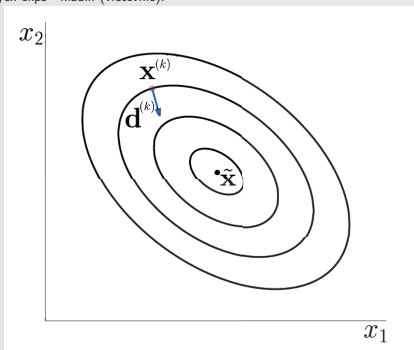
**Princip**

Stejně jako u každé iterační metody nejprve zvolíme počáteční aproximaci řešení  $x^{(0)}$ .  
Princip gradientních metod spočívá v tom, že zvolíme směr a v tomto směru se budeme chtít co nejvíce přiblížit k přesnému řešení. Gradientní metoda je tedy určena volbou směrů, ve kterých minimalizujeme funkci  $F$ .

Během jedné iterace se pohybujeme po povrchu grafu funkce  $F(x)$  tak, abychom se dostali na nižší vrstevnici.



V případě soustavy dvou rovnic získáme promítnutím grafu funkce  $F(x)$  do roviny proměnných  $x_1, x_2$  systém soustředných elips - hladin (vrstevnic).



**Metoda největšího spádu**

Metodu největšího spádu získáme, pokud budeme za směrové vektory volit směry největšího spádu, tj. vektory

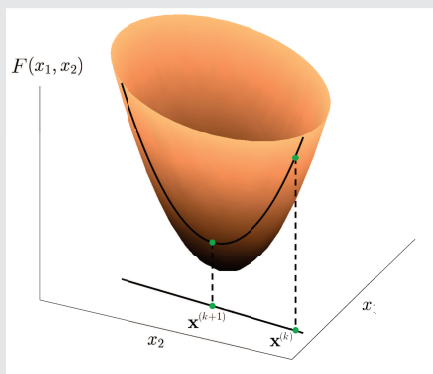
$$d^{(k)} = -\text{grad } F(x^{(k)}) = b - Ax^{(k)}$$

Iterační formulí volíme ve tvaru

$$x^{(k+1)} = x^{(k)} + \xi^{(k)} \cdot d^{(k)}$$

v každém kroku metody určíme směr největšího spádu  $d^{(k)}$  a provedeme jednorozměrnou minimalizaci v tomto směru, tj.

$$\min_{t \geq 0} F(x^{(k)} + t d^{(k)}).$$



Minimalizovanou funkci proměnné  $t$  označíme  $\Psi(t)$ .

Potom platí:

$$\begin{aligned} \Psi(t) &= F(x^{(k)} + t d^{(k)}) = \frac{1}{2} (x^{(k)} + t d^{(k)})^T A (x^{(k)} + t d^{(k)}) - b^T (x^{(k)} + t d^{(k)}) \\ &= \frac{1}{2} x^{(k)T} A x^{(k)} + \frac{1}{2} t x^{(k)T} A d^{(k)} + \frac{1}{2} t^2 d^{(k)T} A d^{(k)} + \frac{1}{2} t^2 d^{(k)T} A d^{(k)} - b^T x^{(k)} - t b^T d^{(k)} \end{aligned}$$

$$\frac{d\Psi(t)}{dt} = \frac{1}{2} x^{(k)T} A d^{(k)} + \frac{1}{2} d^{(k)T} A x^{(k)} + t d^{(k)T} A d^{(k)} - b^T d^{(k)}$$

**Poznámka:** První 2 členy, tj.  $x^{(k)T} A d^{(k)}$  a  $d^{(k)T} A x^{(k)}$  jsou skaláry a jsou si pro symetrickou matici  $A$  rovny.

$$d^{(k)T} A x^{(k)} = (d^{(k)T} A x^{(k)})^T = (A x^{(k)})^T d^{(k)} = x^{(k)T} A^T d^{(k)}$$

$$\begin{aligned} \frac{d\Psi(t)}{dt} &= t d^{(k)T} A d^{(k)} + \frac{x^{(k)T} A d^{(k)} - b^T d^{(k)}}{-d^{(k)T} d^{(k)}} \\ \frac{d\Psi(t)}{dt} &= t d^{(k)T} A d^{(k)} + \frac{x^{(k)T} A d^{(k)} - b^T d^{(k)}}{-d^{(k)T} d^{(k)}} \\ \frac{d\Psi(t)}{dt} &= t d^{(k)T} A d^{(k)} - d^{(k)T} d^{(k)} = 0 \\ \xi^{(k)} &= \frac{d^{(k)T} d^{(k)}}{d^{(k)T} A d^{(k)}} \end{aligned}$$

**Poznámka:**

Pokud by matice  $A$  nespĺovala podmínku symetrie, jaký výsledek by nám dala metoda největšího spádu?

$$\text{grad } F(x) = 0$$

$$\begin{aligned} \text{grad } F(x) &= \text{grad} \left( \frac{1}{2} x^T A x - b^T x \right) = \frac{1}{2} (x^T A)^T + \frac{1}{2} A x - b = \frac{1}{2} A^T x + \frac{1}{2} A x - b = 0 \\ \Rightarrow \frac{1}{2} (A^T + A) x &= b \end{aligned}$$

**Algoritmus metody největšího spádu**

1. volba  $x^{(0)}, \epsilon$
2. výpočet směru spádu  $d^{(k)} = b - Ax^{(k)}$
3. výpočet koeficientu  $\xi^{(k)} = \frac{d^{(k)T} d^{(k)}}{d^{(k)T} A d^{(k)}}$
4. výpočet nové iterace  $x^{(k+1)} = x^{(k)} + \xi^{(k)} d^{(k)}$
5.  $k = k + 1$  a zpět na 2) pokud  $\|x^{(k+1)} - x^{(k)}\| > \epsilon$

**Poznámka:** Abychom ušetřili operace násobení matice a vektoru, určíme  $d^{(k+1)}$  takto:

$$d^{(k+1)} = b - Ax^{(k+1)} = b - A(x^{(k)} + \xi^{(k)} d^{(k)}) = d^{(k)} - \xi^{(k)} \underbrace{A d^{(k)}}_{(*)}$$

iteraci

**Věta:** Metoda největšího spádu konverguje (pro symetrickou, pozitivně definitní matici  $A$ ) pro libovolnou volbu počáteční aproximace  $x^{(0)}$  k přesnému řešení soustavy  $Ax = b$ .

**Důkaz:**

Konvergenzi dokážeme v normě  $\|\cdot\|_A = \sqrt{x^T A x}$  (tzv. energetická norma).

$\|\cdot\|_A$  je s euklidovskou normou  $\|\cdot\|_2$  ekvivalentní, tj. z toho již plyne i konvergence v  $\|\cdot\|_2 = \sqrt{x^T x}$

(Definice:  $X$  ... lineární prostor,  $\|\cdot\|_1$  a  $\|\cdot\|_2$  ... normy na  $X$ ;

$\|\cdot\|_1$  a  $\|\cdot\|_2$  jsou ekvivalentní, existují-li čísla  $c_1, C_1 > 0: \forall x \in X \quad c_1 \|x\|_1 \leq \|x\|_2 \leq C_1 \|x\|_1$ )

tj. má platit

$$c^2 x^T x \leq x^T A x \leq C^2 x^T x$$

$$x^T (C^2 I) x \leq x^T A x \leq x^T (c^2 I) x$$

$$\text{platí pro } c = |\lambda_{\min}|, C = |\lambda_{\max}|$$

$x^*$  ... přesné řešení  $Ax = b$

$e^{(k)} = x^{(k)} - x^*$  ... chyba  $k$ -té iterace

Odvodíme nejprve vztah pro energetickou normu chyby  $k$ -té iterace.

$$F(x^{(k)}) - F(x^*) = F(x^* + e^{(k)}) - F(x^*) = \dots \quad (*)$$

Obecně pro 2 body  $x, x + td$  platí:

$$\begin{aligned} F(x + td) - F(x) &= \frac{1}{2} (x + td)^T A (x + td) - b^T (x + td) - \frac{1}{2} x^T A x + b^T x \\ &= t x^T A d + \frac{1}{2} t^2 d^T A d - t b^T d \\ &= t d^T (A x - b) + \frac{1}{2} t^2 d^T A d \end{aligned}$$

Pro náš případ  $x = x^*, t = 1, d = e^{(k)}$

$$\dots = F(x^* + e^{(k)}) - F(x^*) = \frac{1}{2} e^{(k)T} A e^{(k)} = \|e^{(k)}\|_A^2 \quad (**)$$

(\*) + (\*\*)  $\Rightarrow$

$$F(x^{(k)}) - F(x^*) = \frac{1}{2} e^{(k)T} A e^{(k)} \quad (***)$$

(\*\*\*)  $\Rightarrow$

$$F(x^{(k+1)}) - F(x^*) = \frac{1}{2} e^{(k+1)T} A e^{(k+1)} \quad (***)$$

Odečtením (\*\*\*) a (\*\*\*) dostaneme

$$F(x^{(k+1)}) - F(x^{(k)}) = \frac{1}{2} e^{(k+1)T} A e^{(k+1)} - \frac{1}{2} e^{(k)T} A e^{(k)} \quad (\clubsuit)$$

kde iterace  $x^{(k+1)}$  je vypočtena metodou největšího spádu, tj.

$$x^{(k+1)} = x^{(k)} + \xi^{(k)} r^{(k)}$$

Pro výraz na levé straně opět použijeme **zvýrazněný vztah** pro hodnoty  $x = x^{(k)}, t = \xi^{(k)}, d = r^{(k)}$

$$F(x^{(k)} + \xi^{(k)} r^{(k)}) - F(x^{(k)}) = \xi^{(k)} r^{(k)T} (A x^{(k)} - b) + \frac{1}{2} \xi^{(k)2} r^{(k)T} A r^{(k)} =$$

( $\xi^{(k)}$  jsme počítali podle vztahu  $\xi^{(k)} = \frac{r^{(k)T} r^{(k)}}{r^{(k)T} A r^{(k)}}$ )

$$= -\frac{r^{(k)T} r^{(k)}}{r^{(k)T} A r^{(k)}} + \frac{1}{2} \frac{r^{(k)T} r^{(k)}^2}{r^{(k)T} A r^{(k)}^2} - \frac{1}{2} \frac{r^{(k)T} r^{(k)}^2}{r^{(k)T} A r^{(k)}} = -\frac{1}{2} \frac{r^{(k)T} r^{(k)}^2}{r^{(k)T} A r^{(k)}} \quad (\heartsuit)$$

Porovnáním ( $\heartsuit$ ) a ( $\clubsuit$ ) dostaneme

$$\frac{1}{2} \frac{r^{(k)T} r^{(k)}^2}{r^{(k)T} A r^{(k)}} - \frac{1}{2} e^{(k+1)T} A e^{(k+1)} - \frac{1}{2} e^{(k)T} A e^{(k)} \quad (\diamond)$$

Nyní poslední rovnici

- a) vynásobíme 2
- b) poslední člen převedeme na druhou stranu
- c) a vydělíme jím rovnici

Dostaneme

$$\frac{e^{(k+1)T} A e^{(k+1)}}{e^{(k)T} A e^{(k)}} = 1 - \frac{r^{(k)T} r^{(k)}^2}{r^{(k)T} A r^{(k)} e^{(k)T} A e^{(k)}} = \frac{r^{(k)}}{r^{(k)}} \quad (\diamond)$$

Platí  $\frac{A e^{(k)}}{r^{(k)}} = \frac{r^{(k)}}{r^{(k)}}$  protože  $A x^* - b = 0$  a  $r^{(k)} = b - A x^{(k)}$

$$r^{(k)} = b - A x^{(k)} + A x^* - b = A (x^* - x^{(k)}) = A e^{(k)}$$

Dále z  $\frac{A e^{(k)}}{r^{(k)}} = \frac{r^{(k)}}{r^{(k)}}$  plyne  $e^{(k)} = A^{-1} r^{(k)}$  a tedy odhad

$$\|e^{(k)}\| \leq \|A^{-1}\| \cdot \|r^{(k)}\|$$

Pro ( $\diamond$ ) dostáváme odhad:

$$\frac{e^{(k+1)T} A e^{(k+1)}}{e^{(k)T} A e^{(k)}} \leq 1 - \frac{\|r^{(k)}\|^4}{\|A\| \cdot \|r^{(k)}\|^2 \cdot \|A^{-1}\| \cdot \|r^{(k)}\|^2} = 1 - \frac{1}{\|A\| \cdot \|A^{-1}\|} = q < 1$$

Tj.

$$\begin{aligned} e^{(k+1)T} A e^{(k+1)} &\leq q e^{(k)T} A e^{(k)} \quad \forall k \\ e^{(k+1)T} A e^{(k+1)} &\leq q e^{(k)T} A e^{(k)} \quad \forall k \end{aligned} \quad (\blacksquare)$$

Důkaz:

Jde o to odhadnout  $q$  v (■).

$$\|e^{(k+1)}\|_A^2 \leq q \|e^{(k)}\|_A^2 \Rightarrow \|e^{(k)}\|_A^2 \leq q^k \|e^{(0)}\|_A^2$$

$$q = 1 - \frac{1}{\|A\| \cdot \|A^{-1}\|} = 1 - \frac{1}{\kappa(A)} = \frac{\kappa(A) - 1}{\kappa(A) + 1} \leq \sqrt{\frac{\kappa(A) - 1}{\kappa(A) + 1}}$$

$$= \lambda_{\max} = \frac{1}{\lambda_{\min}}$$

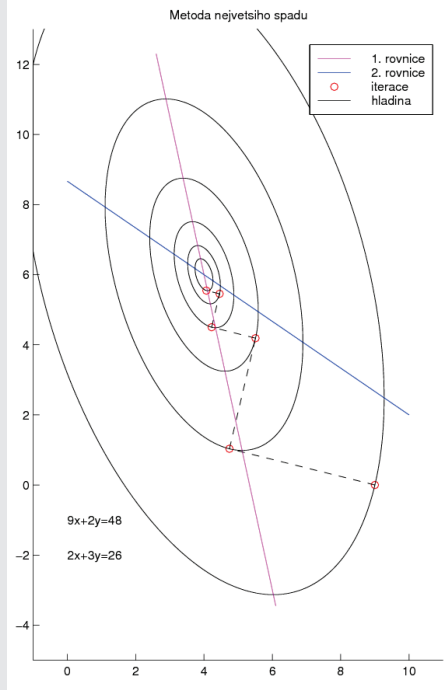
Důkaz poslední nerovnosti:

$$\frac{(\kappa(A) - 1)^2}{\kappa^2(A)} \leq \frac{\kappa(A) - 1}{\kappa(A) + 1} \quad /: (\kappa(A) - 1)$$

$$\frac{\kappa(A) - 1}{\kappa^2(A)} \leq \frac{1}{\kappa(A) + 1} \quad / \cdot \kappa^2(A)(\kappa(A) + 1)$$

$$\kappa^2(A) - 1 \leq \kappa^2(A) \quad \dots OK$$

Geometrický význam metody největšího spádu



k	x <sup>(k)</sup>	y <sup>(k)</sup>
0	9.0000	0
1	4.7425	1.0321
2	5.5081	4.1902
3	4.2240	4.5015
4	4.4549	5.4541
5	4.0676	5.5480



Vlastnost reziduí

Všimněme si faktu, že vždy po sobě jdoucí iterace směru spádu, tj.  $d^{(k)}$  a  $d^{(k+1)}$  jsou na sebe kolmé.

Cvičení: Ukažte, že platí  $d^{(k)T} d^{(k+1)} = 0$ .

$$d^{(k)T} d^{(k+1)} = d^{(k)T} (b - Ax^{(k+1)}) =$$

$$= d^{(k)T} (b - A(x^{(k)} + t^{(k)} d^{(k)})) =$$

$$= d^{(k)T} (b - Ax^{(k)} - t^{(k)} A d^{(k)}) =$$

$$= d^{(k)T} (d^{(k)} - t^{(k)} A d^{(k)}) =$$

$$= d^{(k)T} d^{(k)} - t^{(k)} d^{(k)T} A d^{(k)} =$$

$$= d^{(k)T} d^{(k)} - \frac{d^{(k)T} A d^{(k)}}{d^{(k)T} A d^{(k)}} d^{(k)T} A d^{(k)} =$$

$$= d^{(k)T} d^{(k)} - d^{(k)T} d^{(k)} = 0$$

Poznámka:

V případě, že budou hladiny (elipsy) „velmi protáhlé“, bude obecně metoda největšího spádu konvergovat velmi pomalu, nastane tzv. cik-cak efekt.

Na druhou stranu, pokud budou hladiny (elipsy) „skoro kružnice“, bude metoda největšího spádu konvergovat velmi rychle.

Nevýhodu cik-cak efektu odstraní nová metoda, tzv. **metoda sružených gradientů**, která využívá důmyslnější volby směrů minimalizace, a sice tak, aby se neopakovali, jak k tomu docházelo u metody největšího spádu.

Příklad 1 - pokračování

Uvažovali jsme jednoduchou soustavu  $Ax = b$ , kde  $A = \begin{bmatrix} 25 & 0 \\ 0 & 16 \end{bmatrix}$ ,  $b = \begin{bmatrix} 25 \\ 8 \end{bmatrix}$ ,  $x^* = \begin{bmatrix} 1 \\ 0,5 \end{bmatrix}$ .

Jedna z vrstevnic měla tvar

$$\frac{(x-1)^2}{4^2} + \frac{(y-\frac{1}{2})^2}{5^2} = 1$$

poměr poloos:

$$\sqrt{\lambda_2} = \sqrt{16} \rightarrow 4 : 5 \leftarrow \sqrt{25} = \sqrt{\lambda_1}$$

$$\sqrt{\lambda_2} : \sqrt{\lambda_1}$$

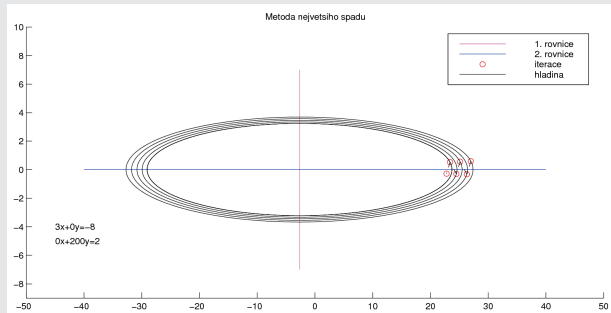
Poznámka:

- Pro případ  $\lambda_2 \gg \lambda_1$  získáme protáhlé elipsy
- Pro případ  $\lambda_2 \approx \lambda_1$  získáme skoro kružnice

Příklad 2

Pomocí metody největšího spádu řešte soustavu  $Ax = b$ , kde

$$A = \begin{bmatrix} 3 & 0 \\ 0 & 200 \end{bmatrix}, \quad b = \begin{bmatrix} -8 \\ 2 \end{bmatrix}, \quad \text{počáteční iterace } x^{(0)} = \begin{bmatrix} 27 \\ 0,6 \end{bmatrix}$$



k	x <sup>(k)</sup>	y <sup>(k)</sup>
0	27.000000	0.600000
1	26.307758	-0.317804
2	25.139940	0.563008
3	24.491101	-0.297251
4	23.396504	0.528335
5	22.788346	-0.277987
⋮	⋮	⋮
250	-2.657606	0.010180

vlastní čísla matice A:

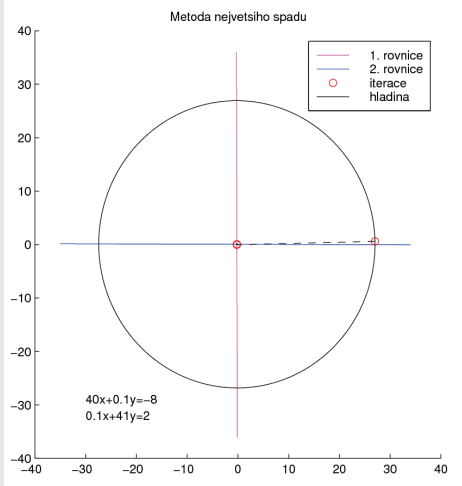
$$\lambda_1 = 3 \quad \text{a} \quad \lambda_2 = 200$$

presné řešení soustavy je  $x^* = \begin{bmatrix} -\frac{8}{3} \\ \frac{1}{100} \end{bmatrix}$

Příklad 3

Pomocí metody největšího spádu řešte soustavu  $Ax = b$ , kde

$$A = \begin{bmatrix} 40 & 0,1 \\ 0,1 & 41 \end{bmatrix}, \quad b = \begin{bmatrix} -8 \\ 2 \end{bmatrix}, \quad \text{počáteční iterace } x^{(0)} = \begin{bmatrix} 27 \\ 0,6 \end{bmatrix}$$



k	x <sup>(k)</sup>	y <sup>(k)</sup>
0	27.000000	0.600000
1	-0.197972	-0.032418
2	-0.199872	0.049274
3	-0.200123	0.049268

vlastní čísla matice A:  $\lambda_1 \doteq 39,99$  a  $\lambda_2 \doteq 41,01$   
 přesné řešení soustavy  $x^*$  se  $s(x^{(3)})$  shoduje na 6 desetných míst

Poznámky k rychlosti konvergence:

$$\|x^{(k)} - x^*\|_A = \left( \frac{\kappa(A) - 1}{\kappa(A) + 1} \right)^k \|x^{(0)} - x^*\|_A$$

- Je-li  $\kappa(A) \gg 1$ , tj.  $\lambda_{max} \gg \lambda_{min}$ , pak metoda největšího spádu konverguje pomalu

$$\frac{\kappa(A) - 1}{\kappa(A) + 1} = \frac{\kappa(A) - 1 + 1 - 1}{\kappa(A) + 1} = 1 - \frac{2}{\kappa(A) + 1} \approx 1 - \frac{2}{100 \text{ pro } \kappa(A) = 100}$$

- Je-li  $\kappa(A) \lesssim 1$ , tj.  $\lambda_{max} \approx \lambda_{min}$ , pak metoda největšího spádu konverguje rychle

$$\frac{\kappa(A) - 1}{\kappa(A) + 1} = 1 - \frac{2}{\kappa(A) + 1} \approx 0$$

- Pokud jsou vrstevnice sféry (v  $\mathbb{R}^2$  kružnice), potom metoda největšího spádu nalezne řešení (přesné) v jednom kroku.

Poznámka:

Směr, ve kterém provádíme minimalizaci v rámci jednoho kroku metody, můžeme volit i jinak než směr největšího spádu.

Obecně označme používané směry  $s^{(k)}$ .

Novou iteraci hledáme ve tvaru

$$x^{(k+1)} = x^{(k)} + t s^{(k)}$$

Koeficient  $t$  získáme z jednorozměrné minimalizace

$$\min_{t \geq 0} F(x^{(k)} + t s^{(k)})$$

$$\Phi(t) = \frac{1}{2} (x^{(k)} + t s^{(k)})^T A (x^{(k)} + t s^{(k)}) - b^T (x^{(k)} + t s^{(k)})$$

$$\frac{d\Phi(t)}{dt} = t s^{(k)T} A s^{(k)} + \frac{x^{(k)T} A s^{(k)} - b^T s^{(k)}}{1} = 0$$

$$\frac{d\Phi(t)}{dt} = \underbrace{t s^{(k)T} A s^{(k)}}_{(Ax^{(k)} - b)^T s^{(k)}} = 0$$

$$= -r^{(k)} \text{ (reziduum)}$$

$$t^{(k)} = \frac{r^{(k)T} s^{(k)}}{s^{(k)T} A s^{(k)}}$$

Volíme-li za vektory  $s^{(k)}$  postupně jednotkové vektory souřadných os, získáme Gauss-Seidelovu metodu !!!

Pokud na vektor  $x$  aplikujeme 1 iteraci gradientní metody se směrovým vektorem  $e_i = [0, \dots, 0, 1, 0, \dots, 0]^T$  (na  $i$ -té pozici 1, jinak 0), dostaneme:

$$x := x + \frac{r^T e_i}{e_i^T A e_i} e_i$$

Platí:  $e_i^T A$  ...  $i$ -tý řádek matice A

$e_i^T A e_i$  ... diagonální prvek  $a_{ii}$  matice A

$r = b - Ax$  ... reziduum

$r^T e_i = r_i$  ...  $i$ -tá složka vektoru  $r$

$r_i = b_i - \sum_{j=1}^n a_{ij} x_j$

Vztah (\*) zvětší  $i$ -tou složku vektoru  $x$  o hodnotu  $r_i$ , tj.

$$x_i := x_i + \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^n a_{ij} x_j \right) = r_i$$

$$x_i := \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j - \sum_{j=i+1}^n a_{ij} x_j \right)$$

Script v MATLABu

```
function [vysledky_gs,vysledky_gm]=gs_gm(A,b,x0,iteraci);
%*****
% Porovnání Gauss-Seidelovy metody a
% gradientní metody, kde za směry volíme
% jednotkové vektory souřadných os
%*****
n=size(A,1);
%*****
% Gauss-Seidelova metoda
%*****
x=x0;
vysledky_gs=x';
D=diag(diag(A)); L=tril(A)-D; U=triu(A)-D;
H=-(L+D)\U;
g=(L+D)\b;
for i=1:iteraci
    x=H*x+g;
    vysledky_gs=[vysledky_gs;x'];
end
%*****
% Gradientní metoda
%*****
x=x0;
vysledky_gm=x';
for i=1:iteraci
    for j=1:n
        s=zeros(n,1);
        s(j)=1;
        r=A*x+b;
        t=(r'*s)/(s'*A*s);
        x=x+t*s;
    end;
    vysledky_gm=[vysledky_gm;x'];
end
```

Úvaha

Při vhodné volbě směrových vektorů  $s^{(k)}$  je možné dojít do přesného řešení za konečný počet kroků  $\leq n$ . Musí existovat  $n$  vektorů  $s^{(k)}$  tak, že

$$x^* - x^{(0)} = \sum_{k=1}^n t^{(k)} s^{(k)} \quad (*)$$

Jak volit směry  $s^{(k)}$ ?

- Zkusíme takto: nechť  $s^{(k)}$  tvoří bázi (ortogonální)  $n$ -rozměrného euklidovského prostoru, potom vynásobením (\*) skalárně  $s^{(k)}$  a úpravou získáme

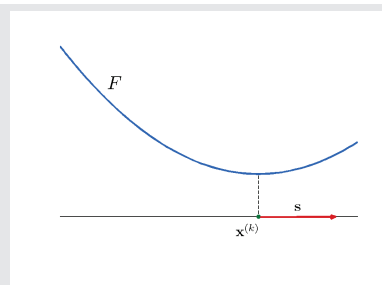
$$s^{(k)T} (x^* - x^{(0)}) = t^{(k)} s^{(k)T} s^{(k)}$$

$$t^{(k)} = \frac{s^{(k)T} (x^* - x^{(0)})}{s^{(k)T} s^{(k)}} \quad \dots \text{ nešikovní } t, \text{ obsahuje přesné řešení}$$

- je třeba zvolit lepší strategii volby vektorů  $s^{(k)}$

Definice  $x^{(k)}$  je optimální vzhledem ke směru  $s \neq 0$ , jestliže

$$F(x^{(k)}) \leq F(x^{(k)} + ts) \quad \forall t \in \mathbb{R} \quad (*)$$



Poznámka: Je-li  $x^{(k)}$  optimální vzhledem k libovolnému směru  $s$  z vektorového prostoru  $V$ , říkáme, že je  $x^{(k)}$  optimální vzhledem k  $V$ .

Podle (\*) se minima nabývá pro  $t = 0$ , tzn. že derivace  $F$  podle  $t$  je v minimu ( $t = 0$ ) rovna 0:

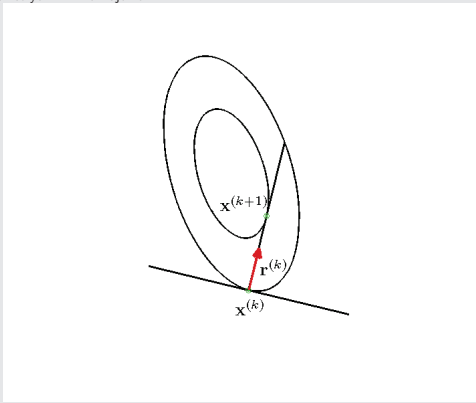
$$\frac{\partial F}{\partial t} (x^{(k)} + ts) = t s^T A s + s^T (A x^{(k)} - b)$$

$$\frac{\partial F}{\partial t}(x^{(k)}) = s^T (Ax^{(k)} - b) = 0$$

$$= r^{(k)}$$

$$s \perp r^{(k)}$$

**Poznámka:** Iterace  $x^{(k+1)}$  metody největšího spádu je optimální vzhledem k reziduu  $r^{(k)} = b - Ax^{(k)}$  ... směry, ve kterých minimalizujeme.



Naším cílem je, aby se i v dalších iteracích zachovávala optimalita k již použitým směrům.

To pro metodu největšího spádu bohužel neplatí.

Např. pro soustavu ve 2D jsme ukazovali, že směry největšího spádu (reziduí) jsou na sebe kolmé,

$$\text{tj. } r^{(k)} \perp r^{(k+1)} \text{ a } r^{(k+1)} \perp r^{(k+2)} \Rightarrow \boxed{r^{(k)} \parallel r^{(k+2)}} !!!$$

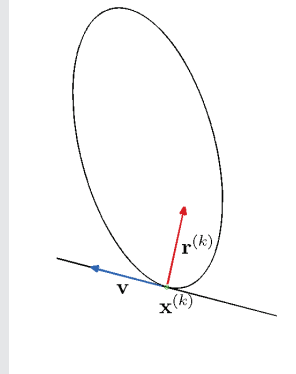
$\Rightarrow x^{(k+2)}$  je optimální vzhledem k  $r^{(k+1)}$ , ale již není optimální vzhledem k  $r^{(k)}$

Existují směry, které udržují optimalitu k předchozím?

Nechť

$$\boxed{x^{(k+1)} = x^{(k)} + s}$$

Předpokládáme, že  $x^{(k)}$  je optimální vzhledem k  $v$  (tj.  $r^{(k)} \perp v$ ).



Chceme-li, aby bylo  $x^{(k+1)}$  optimální vzhledem k  $v$ , (tj.  $r^{(k+1)} \perp v$ ), musí platit:

$$0 = v^T r^{(k+1)} = v^T (b - Ax^{(k+1)}) = v^T (b - A(x^{(k)} + s)) = v^T \left( \frac{b - Ax^{(k)}}{r^{(k)}} - As \right) = v^T (r^{(k)} - As) = -v^T As$$

**Závěr**

Chceme-li zachovat optimalitu vzhledem ke všem použitým směrům, musí tyto směry splňovat podmínky tzv. **A-ortogonalit**y, tj. pro 2 různé směry  $s$  a  $v$  musí platit:

$$\boxed{v^T As = 0}$$

**Poznámka:** Vektorům které jsou **A-ortogonální** se také říká **A-sdružené**.

**Metoda sdružených gradientů**

Za směry, ve kterých minimalizujeme, budeme brát A-ortogonální vektory  $s^{(k)}$ .

Platí tedy:

$$\boxed{s^{(k)T} A s^{(l)} = 0, \quad k \neq l}$$

Chceme, aby platilo:

$$s^{(k)T} A \cdot / \quad \boxed{x^* - x^{(0)} = \sum_{k=1}^n \alpha^{(k)} s^{(k)}} \quad (*)$$

$$s^{(k)T} A(x^* - x^{(0)}) = \alpha^{(k)} s^{(k)T} A s^{(k)}$$

$$Ax^* - Ax^{(0)} = \underbrace{Ax^* - b}_{=0} - \underbrace{Ax^{(0)} - b}_{r^{(0)}}$$

$$\alpha^{(k)} = \frac{s^{(k)T} r^{(0)}}{s^{(k)T} A s^{(k)}}$$

**Strategie volby směrů**

• Máme-li ortogonální bázi  $\mathbb{R}^n$ , lze z ní procesem A-ortogonalizace získat A-ortogonální bázi.

• Za ortogonální bázi budeme volit reziduové vektory.

Abý proces ortogonalizace vedl k cíli, musíme zaručit, že reziduové vektory tvoří bázi. Ortogonalitu ukážeme vzápětí; může se stát, že se některé reziduum anulují. Potom ovšem iterační proces končí - dosáhli jsme přesného řešení.

• Provádíme tedy současně 2 procesy!

- iterační proces
- proces A-ortogonalizace

• Vektory reziduí budeme značit  $r^{(k)}$ , získané sdružené směry označíme  $s^{(k)}$

- pro zadané  $x^{(0)}$  určíme  $r^{(0)} = b - Ax^{(0)}$
- $s^{(0)}$  položíme rovno  $r^{(0)}$
- určíme  $x^{(1)}$  optimální vzhledem k  $s^{(0)}$
- určíme  $r^{(1)}$
- $s^{(1)}$  určíme z  $r^{(1)}$  tak, aby  $s^{(1)T} A s^{(0)} = 0$
- atd.

**Proces A-ortogonalizace**

$$\boxed{s^{(k)} = r^{(k)} + \sum_{i=1}^{k-1} \beta_{ki} s^{(i)}} \quad (**)$$

(Při určení  $s^{(k)}$  vyjeme z  $r^{(k)}$ . Přičítáme násobky předchozích  $s^{(i)}$  tak, abychom zaručili A-ortogonalitu.)

Koeficienty  $\beta_{ki}$  volíme tak, aby

$$\boxed{s^{(k)T} A s^{(i)} = 0, \quad (i < k)}$$

(\*\*) vynásobíme  $s^{(i)T} A \cdot /$

$$\frac{s^{(i)T} A s^{(k)}}{=0} = s^{(i)T} A r^{(k)} + \beta_{ki} s^{(i)T} A s^{(i)}$$

$$\Rightarrow \boxed{\beta_{ki} = -\frac{s^{(i)T} A r^{(k)}}{s^{(i)T} A s^{(i)}}$$

Z vlastností A-ortogonalit vyplývá řada skutečností.

**Věta 1** Platí

$$\boxed{\begin{aligned} r^{(k)T} \cdot s^{(j)} &= r^{(0)T} \cdot s^{(j)} & k \leq j \\ r^{(k)T} \cdot s^{(j)} &= 0 & k > j \end{aligned}}$$

**Důkaz:**

$$\begin{aligned} -b \cdot / \quad A \cdot / \quad x^{(k+1)} &= x^{(k)} + \alpha^{(k)} s^{(k)} \\ \Rightarrow -r^{(k+1)} &= -r^{(k)} + \alpha^{(k)} A s^{(k)} \\ \Rightarrow r^{(k)} &= r^{(0)} - \sum_{j=1}^{k-1} \alpha^{(j)} A s^{(j)} \end{aligned}$$

vynásobíme skalárně s  $s^{(j)}$

$$\boxed{r^{(k)T} s^{(j)} = r^{(0)T} s^{(j)} - \sum_{i=1}^{k-1} \alpha^{(i)} s^{(i)T} A s^{(j)}} \quad (*)$$

(\*) počítáme (viz dříve) takto

$$\alpha^{(j)} = \frac{r^{(0)T} s^{(j)}}{s^{(j)T} A s^{(j)}}$$

**Důkaz:**

vztah (\*\*) vynásobíme skalárně s  $A s^{(j)}$

$$s^{(k)} = r^{(k)} + \sum_{i=1}^{k-1} \beta_{ki} s^{(i)} \quad (**)$$

$$\begin{aligned} \frac{s^{(j)T} A s^{(k)}}{=0} &= s^{(j)T} A r^{(k)} + \sum_{i=1}^{k-1} \beta_{ki} s^{(j)T} A s^{(i)} \\ &= 0 \quad (k < j) \\ &\neq 0 \quad (k = j) \end{aligned}$$

**Důkaz:**

Úplnou matematickou indukci ukážeme, že  $\boxed{r^{(j)T} r^{(k)} = 0 \quad \text{pro } j > k}$

Platí

$$r^{(k+1)} = b - Ax^{(k+1)} = b - A(x^{(k)} + \alpha^{(k)} s^{(k)}) = r^{(k)} - \alpha^{(k)} A r^{(k)}$$

1.  $j=1 \Rightarrow k=0$

$$r^{(1)T} r^{(0)} = (r^{(0)} + \alpha^{(0)} A s^{(0)})^T r^{(0)} = r^{(0)T} r^{(0)} + \alpha^{(0)} s^{(0)T} A r^{(0)} = r^{(0)T} r^{(0)} - \frac{r^{(0)T} s^{(0)}}{s^{(0)T} A s^{(0)}} s^{(0)T} A r^{(0)} = 0$$

( $r^{(0)} = s^{(0)}$ )

2. a)  $\forall k < j$  platí:  $r^{(j+1)T} r^{(k)} = 0$

$$r^{(j+1)T} r^{(k)} = (r^{(j)} + \theta^{(j)} A s^{(j)})^T r^{(k)} = \underbrace{r^{(j)T} r^{(k)}}_{=0 \text{ (předpoklad)}} + \theta^{(j)} \underbrace{s^{(j)T} A r^{(k)}}_{=0 \text{ (Věta 2)}}$$

b)  $r^{(j+1)T} r^{(j)} = 0$

$$r^{(j+1)T} r^{(j)} = (r^{(j)} + \theta^{(j)} A s^{(j)})^T r^{(j)} = r^{(j)T} r^{(j)} + \theta^{(j)} s^{(j)T} A r^{(j)} = \underbrace{r^{(j)T} r^{(j)}}_{=0} - \underbrace{\frac{r^{(j)T} s^{(j)}}{s^{(j)T} A s^{(j)}} s^{(j)T} A r^{(j)}}_{=0} \quad (\text{Věta 2})$$

$$r^{(j)T} r^{(j)} = r^{(j)T} A r^{(j)} \quad (\text{Věta 1})$$

Důkaz: Platí:

$$\beta_{ki} = -\frac{s^{(i)T} A r^{(k)}}{s^{(i)T} A s^{(i)}}$$

Pro čítel platí:

$$s^{(i)T} A r^{(k)} = r^{(k)T} A s^{(i)} = r^{(k)T} \frac{1}{r^{(i)}} (r^{(i+1)} - r^{(i)}),$$

kde se použil vztah

$$r^{(i+1)} = r^{(i)} + \theta^{(i)} A s^{(i)}, \quad \theta^{(i)} \neq 0 \text{ pro } r^{(i)} \neq 0$$

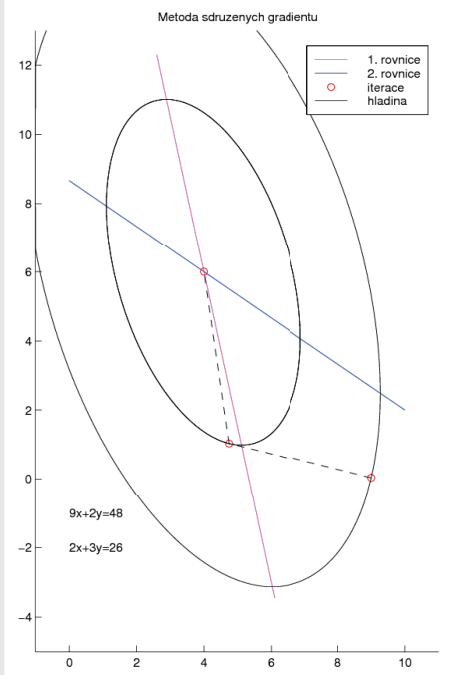
Platí

- pro  $i < k - 1$ : čítel  $\beta_{ki} = r^{(k)T} (r^{(i+1)} - r^{(i)}) \frac{1}{r^{(i)}} = 0$  (Tvzení)
- pro  $i = k - 1$ : čítel  $\beta_{k,k-1} = r^{(k)T} (r^{(k)} - r^{(k-1)}) \frac{1}{r^{(k-1)}} = r^{(k)T} r^{(k)} \frac{1}{r^{(k-1)}} \neq 0$  (pro  $r^{(k)} \neq 0$ )

Algoritmus (Metoda sdružených gradientů)

- $x^{(0)}, \epsilon$
- $r^{(0)} = b - A x^{(0)}, s^{(0)} = r^{(0)}$
- $\theta^{(k)} = \frac{s^{(k)T} r^{(k)}}{s^{(k)T} A s^{(k)}}$
- $x^{(k+1)} = x^{(k)} + \theta^{(k)} s^{(k)}$
- $r^{(k+1)} = r^{(k)} + \theta^{(k)} A s^{(k)}$
- $\beta_k = -\frac{s^{(k)T} A r^{(k+1)}}{s^{(k)T} A s^{(k)}}$
- $s^{(k+1)} = r^{(k+1)} + \beta_k s^{(k)}$
- If  $\|x^{(k+1)} - x^{(k)}\| < \epsilon$  then **konec**, else  $\rightarrow$  add 3

Geometrický význam metody sdružených gradientů



k	$x^{(k)}$	$y^{(k)}$
0	9.0000	0
1	4.7425	1.0321
2	4.0000	6.0000



Poznámka: Gradientní metody patří mezi **nestacionární metody**.

např. pro metodu největšího spádu platí

$$x^{(k+1)} = x^{(k)} + \theta^{(k)} d^{(k)} = x^{(k)} + \theta^{(k)} (b - A x^{(k)}) = (I - \theta^{(k)} A) x^{(k)} + \theta^{(k)} b$$

V každém kroku se mění matice  $H^{(k)}$ .

Platí-li  $\|H^{(k)}\| \rightarrow 0$  (pro  $k \rightarrow \infty$ ), dostaneme metody se **superlineární rychlostí konvergence**.

Věta Nechť  $A$  je symetrická pozitivně definitní. Potom metoda sdružených gradientů konverguje nejvýše po  $n$  krocích. Navíc chyba  $k$ -té iterace ( $k < n$ ) je ortogonální na směry  $s^{(j)}, j = 0, 1, \dots, k-1$  a platí:

$$\|x^{(k)} - x^*\|_A \leq \frac{2C^k}{1 + C^{2k}} \|x^{(0)} - x^*\|_A$$

kde  $C = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}, \kappa(A) = \frac{\lambda_{max}}{\lambda_{min}}$ .

Poznámka: V metodě největšího spádu vystupuje ve vztahu pro chybu  $k$ -té iterace koeficient

$$\left( \frac{\kappa(A) - 1}{\kappa(A) + 1} \right)^k$$

Je zřejmé, že na rychlost konvergence má vliv číslo  $\kappa(A)$ , tj.  $\lambda_{max}$  a  $\lambda_{min}$ . Čím blíže je  $\lambda_{max}$  a  $\lambda_{min}$ , tím rychleji metody konvergují.

Příklad 4

Řešte soustavu  $Ax = b$ , kde  $A = \begin{bmatrix} 1 & 0 \\ 0 & 10000 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 10000 \end{bmatrix}$ , přesné řešení  $x^* = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ .



- poměr poloos elips je  $\sqrt{10000} : \sqrt{1} = 100 : 1$  !!!
  - $\kappa(A) = \frac{10000}{1} = 10000 \Rightarrow$  **pomalá konvergence!**
- Vezměme si matici  $P^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{10000} \end{bmatrix}$  ( $\det(P^{-1}) \neq 0$ ) a řešme soustavu

$$P^{-1}Ax = P^{-1}b$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

- $\kappa(P^{-1}A) = \frac{1}{1} = 1 \Rightarrow$  rychlá konvergence (1. iterace). Mluvíme o tzv. **předpodmínování**.

**Poznámka:**

Chceme-li i novou (předpodmíněnou) soustavu řešit metodou sdružených gradientů, musí být její matice symetrická pozitivně definitní.

Místo matice  $P^{-1}A$  vezmeme matici (podobnou  $A$ )  $P^{-\frac{1}{2}}AP^{-\frac{1}{2}}$  ( $P \dots$  symetrická pozitivně definitní) a řešíme soustavu

$$\tilde{A}\tilde{x} = \tilde{b}$$

$$\tilde{A} = P^{-\frac{1}{2}}AP^{-\frac{1}{2}}, \quad \tilde{x} = P^{\frac{1}{2}}x, \quad \tilde{b} = P^{-\frac{1}{2}}b$$

Jak volit matice předpodmínění  $P$  ?  
... řada možností, např.  $P = \text{diag}(A)$

**Příklad 5**

Porovnejte vlastní čísla zadané matice  $A$  a matice získané pomocí diagonálního předpodmínění.

$$A = \begin{bmatrix} 1000000 & 200 & 30 & 0 \\ 200 & 10000 & 40 & 0 \\ 30 & 40 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P = \begin{bmatrix} 1000000 & 0 & 0 & 0 \\ 0 & 10000 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P^{-\frac{1}{2}} = \begin{bmatrix} \frac{1}{\sqrt{1000000}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{10000}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{100}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{1}} \end{bmatrix} = \begin{bmatrix} \frac{1}{1000} & 0 & 0 & 0 \\ 0 & \frac{1}{100} & 0 & 0 \\ 0 & 0 & \frac{1}{10} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\tilde{A} = P^{-\frac{1}{2}}AP^{-\frac{1}{2}} = \begin{bmatrix} \frac{1}{1000} & 0 & 0 & 0 \\ 0 & \frac{1}{100} & 0 & 0 \\ 0 & 0 & \frac{1}{10} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1000000 & 200 & 30 & 0 \\ 200 & 10000 & 40 & 0 \\ 30 & 40 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{1000} & 0 & 0 & 0 \\ 0 & \frac{1}{100} & 0 & 0 \\ 0 & 0 & \frac{1}{10} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0.002 & 0.003 & 0 \\ 0.002 & 1 & 0.04 & 0 \\ 0.003 & 0.04 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

vlastní čísla matice  $\tilde{A}$ :

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

$$\lambda_1 = 1; \quad \lambda_2 = 99,837534233; \quad \lambda_3 = 10000,121161153; \quad \lambda_4 = 1000000,041304614$$

$$\kappa(A) = \frac{1000000,041304614}{1} = 1000000,041304614$$

vlastní čísla matice  $\tilde{A} = P^{-\frac{1}{2}}AP^{-\frac{1}{2}}$ :

$$\tilde{\lambda}_1 = 0,959987454937999; \quad \tilde{\lambda}_2 = 0,999702401514713; \quad \tilde{\lambda}_3 = 1; \quad \tilde{\lambda}_4 = 1,040310143547287$$

$$\kappa(\tilde{A}) = \frac{1,040310143547287}{0,959987454937999} = 1,083670560689229$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

## Kapitola 6. Vlastní čísla a vlastní vektory

**Výpočet vlastních čísel a vlastních vektorů**

S pojmem **vlastního čísla** jsme se již setkali například u iteračních metod pro řešení soustav lineárních algebraických rovnic. Velikosti vlastních čísel iterační matice rozhodovaly o konvergenci příslušné iterační metody. S úlohou na vlastní čísla se setkáme i v aplikacích při řešení řady technických a fyzikálních problémů.

**Definice:** Je dána čtvercová matice  $A$  řádu  $n$ . Číslo  $\lambda$ , pro které má soustava

$$Av = \lambda v \quad \text{resp.} \quad (A - \lambda I)v = 0$$

nenulové řešení, se nazývá **vlastní číslo** matice  $A$ , jemu odpovídající nenulové řešení v **vlastní vektor** matice  $A$ .

Homogenní soustava má nenulové řešení  $\Leftrightarrow$  matice soustavy je singularní, tj. její determinant je nulový.

Vlastní čísla  $\lambda_1, \lambda_2, \dots, \lambda_n$  jsou kořeny **charakteristické rovnice**

$$p_A(\lambda) = \det(A - \lambda I) = 0.$$

Ke každému vlastnímu číslu  $\lambda_i$  existuje alespoň jeden vlastní vektor  $v_i$ .

**Poznámka:** Charakteristický polynom je stupně  $n \Rightarrow \exists n$  vlastních čísel.

**Definice:** Matici  $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$  nazýváme **spektrální maticí** matice  $A$ .

Úlohy na nalezení vlastních čísel rozdělíme do dvou skupin:

- Úplný problém** – úloha najít všechna vlastní čísla
- Částečný problém** – úloha najít pouze některá vl. čísla (obvykle s největší absolutní hodnotou)

Úlohu na vlastní čísla si připomeneme na příkladu.

**Příklad 1**

Stanovte taková čísla  $\lambda$ , pro která má homogenní soustava  $Av = \lambda v$  nenulové řešení, dále určete toto řešení, pro matici

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 2 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}.$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

Řešíme tedy soustavu

$$(A - \lambda I)v = \begin{bmatrix} 2-\lambda & 0 & 0 \\ 2 & 2-\lambda & 1 \\ 1 & 1 & 2-\lambda \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

Abý homogenní soustava měla nenulové řešení, musí být determinant soustavy nulový. Hledáme proto taková  $\lambda$ , aby

$$\det(A - \lambda I) = (2 - \lambda)^3 - (2 - \lambda) = (2 - \lambda)[(2 - \lambda)^2 - 1] = (2 - \lambda)(1 - \lambda)(3 - \lambda) = 0.$$

Dostali jsme algebraickou rovnici stupně 3 a pouze pro její kořeny

$$\lambda_1 = 3, \quad \lambda_2 = 2, \quad \lambda_3 = 1$$

bude mít uvažovaná soustava nenulové řešení.

Ke každému vlastnímu číslu  $\lambda_i$  můžeme najít nenulové řešení homogenní soustavy

$$(A - \lambda_i I)v = 0.$$

Např. pro  $\lambda_1 = 3$  řešíme soustavu

$$\begin{bmatrix} -1 & 0 & 0 \\ 2 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

Matice soustavy je samozřejmě singularní a proto bude existovat celý systém řešení v závislosti na parametru  $r \in \mathbb{R}$ . Každý vektor  $[0, r, r]^T$  řeší danou soustavu. Ze systému vybereme jednoho zástupce, např.  $v^{(1)} = [0, 1, 1]^T$ , a říkáme, že  $v^{(1)}$  je **vlastní vektor** odpovídající vlastnímu číslu  $\lambda_1$ . Podobně bychom našli vlastní vektory odpovídající vlastním číslům  $\lambda_2$  a  $\lambda_3$ .

**Poznámka:**

Vlastní čísla (horní) trojúhelníkové matice jsou rovna jejím diagonálním prvkům, neboť charakteristický polynom má tvar:

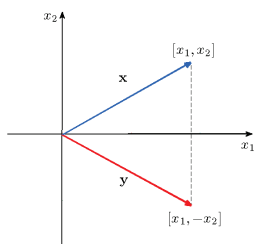
$$p_A(\lambda) = (a_{11} - \lambda)(a_{22} - \lambda) \dots (a_{nn} - \lambda).$$

**Motivace**

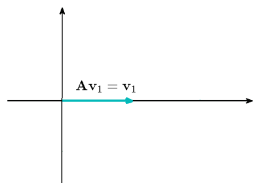
Vlastní vektor je takový vektor, pro který platí, že vynásobíme-li matici  $A$  s tímto vektorem, získáme násobek původního vektoru. Mluvíme o **samodružných prvcích**.

**Příklad:** Osová souměrnost = zobrazení  $y = Ax$ .

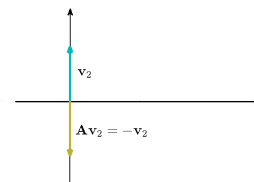
$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\lambda_1} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$



$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{-1}{\lambda_1} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



**Příklad:** Určete vlastní čísla a vlastní vektory těchto matic:

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix},$$

$$C = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}, \quad D = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}.$$

**Řešení:** Všechny zadané matice mají stejný charakteristický polynom

$$p_A(\lambda) = p_B(\lambda) = p_C(\lambda) = p_D(\lambda) = (2 - \lambda)^3,$$

Vidíme, že  $\lambda = 2$  je trojnásobné vl. číslo všech čtyř matic.

Vlastní vektory:

$$A: \begin{aligned} v^{(1)} &= [1, 0, 0]^T \\ v^{(2)} &= [0, 1, 0]^T \\ v^{(3)} &= [0, 0, 1]^T \end{aligned}$$

**Pozn.:** matice  $A - \lambda I$  je nulová, tj. systém všech řešení rovnice  $A - \lambda I = 0$  je lin. kombinací  $v^{(1)}, v^{(2)}, v^{(3)}$ .

$$B: \begin{aligned} v^{(1)} &= [1, 0, 0]^T \\ v^{(3)} &= [0, 0, 1]^T \end{aligned}$$

$$\text{Pozn.: } B - \lambda I = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$C: \begin{aligned} v^{(1)} &= [1, 0, 0]^T \\ v^{(2)} &= [0, 1, 0]^T \end{aligned}$$

$$\text{Pozn.: } C - \lambda I = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$D: v^{(1)} = [1, 0, 0]^T$$

$$\text{Pozn.: } D - \lambda I = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

**Poznámka:** Počet lineárně nezávislých vlastních vektorů může být menší než je řád matice.

Připomeňme si některé poznatky z lineární algebry.

**Definice:** Říkáme, že matice  $A$  a  $B$  jsou podobné, existuje-li regulární matice  $P$  taková, že  $P^{-1}AP = B$ , resp.  $A = PBP^{-1}$ .

**Věta** Podobné matice mají stejná vlastní čísla.

**Věta:** Nechť  $A$  je reálná symetrická matice. Potom existuje ortogonální matice  $Q$  taková, že pro spektrální matici platí

$$A = Q^T \Lambda Q$$

**Důkaz:**

$$\det(A - \lambda I) = \det(A - \lambda I) \frac{\det(P)}{\det(P)} = \det(P^{-1}) \cdot \det(A - \lambda I) \cdot \det(P) = \det[P^{-1}(A - \lambda I)P] = \det\left(\frac{P^{-1}AP - \lambda I}{B}\right)$$

**Věta** Je-li  $v$  vlastní vektor matice  $A$ , potom  $P^{-1}v$  je vlastní vektor matice  $B = P^{-1}AP$ .

**Důkaz:**

$$P^{-1} \cdot \begin{aligned} Av &= \lambda v \\ PBP^{-1}v &= \lambda v \\ B \underbrace{P^{-1}v}_w &= \lambda \underbrace{P^{-1}v}_w \end{aligned}$$

**Poznámka:** Pokud jsou vlastní vektory  $v_1, v_2, \dots, v_n$  lineárně nezávislé, potom platí:

$$X^{-1}AX = \Lambda \quad (\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n) \dots \text{spektrální matice})$$

Matice  $A$  je tedy podobná diagonální matici. Matice  $X$  je matice, jejíž sloupce tvoří vlastní vektory

$$AX = X\Lambda$$

$$A \cdot \begin{bmatrix} v_1 \\ v_2 \\ \dots \\ v_n \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ \dots \\ v_n \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \dots \\ & & & \lambda_n \end{bmatrix}$$

**Věta** Nechť  $A$  je čtvercová matice řádu  $n$ ,  $\lambda$  její vlastní číslo a  $v$  její vlastní vektor, tj.  $Av = \lambda v$ . Potom platí:

(i)  $k \in \mathbb{N} \quad \lambda(A^k) = [\lambda(A)]^k$

(ii)  $A \dots$  regulární  $\Rightarrow \lambda(A^{-1}) = [\lambda(A)]^{-1}$

(iii)  $\lambda(A^H) = \overline{\lambda(A)}$

(iv) vlastní čísla symetrické (hermitovské) matice jsou reálná

(v) vlastní vektory symetrické matice odpovídající různým vlastním číslům jsou ortogonální

(vi) symetrická pozitivně definitní matice má všechna vlastní čísla kladná

**Důkaz:**

(i)

$$A \cdot Av = \lambda v \Rightarrow A^2v = \lambda Av = \lambda^2v$$

$$A \cdot A^k v = \lambda^k v \Rightarrow A^{k+1}v = \lambda^k Av = \lambda^{k+1}v$$

(ii)

$$Av = \lambda v \Rightarrow A^{-1}Av = \lambda A^{-1}v$$

$$v = \lambda A^{-1}v \quad / \cdot \frac{1}{\lambda}$$

$$\frac{1}{\lambda}v = A^{-1}v$$

(iii) Označme  $B = A - \lambda I$ . Platí

$$\det B^H = \overline{\det B} = \overline{\det B}$$

$$\det(A^H - \overline{\lambda I}) = \overline{\det(A - \lambda I)} = 0$$



Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

(iv)  $A = A^H, Av = \lambda v$

$$\lambda v^H v = v^H (\lambda v) = v^H A v = \underbrace{v^H A^H v}_{\substack{\text{číslo} \\ \text{přidáme } A^H}} = \overline{(v^H A^H v)^H} = \overline{v^H A v} = \overline{\lambda v^H v} = \overline{\lambda} \overline{v^H v} \stackrel{(*)}{=} \overline{\lambda} v^H v$$

(\*)  $\overline{(v^H v)^H} = v^H v \Rightarrow \overline{v^H v} = v^H v$

(v)

$$\left. \begin{array}{l} u^H \cdot / \\ v^H \cdot / \end{array} \right\} \begin{array}{l} Av = \lambda v \\ Au = \mu u \end{array} \quad \lambda \neq \mu; \lambda = \overline{\lambda}, \mu = \overline{\mu}, A = A^H$$

$$\begin{array}{l} u^H A v = \lambda u^H v \\ v^H A u = \mu v^H u \\ u^H A^H v = \mu u^H v \end{array} \quad /^H$$

$$0 = \underbrace{(\lambda - \mu)}_{\neq 0} u^H v \Rightarrow \boxed{u^H v = 0}$$

(vi)

$$\begin{array}{l} Av = \lambda v \\ v^T A v = \lambda v^T v \end{array}$$

Platí (pro pozitivně definitní matici A):

$$\forall v \neq 0: v^T A v > 0 \Rightarrow \lambda \underbrace{v^T v}_{>0} > 0 \Rightarrow \boxed{\lambda > 0}$$

Poznámka: Ortoğonální matice Q:  $\boxed{Q^T Q = I} / \cdot Q^{-1}$   $\boxed{Q^T = Q^{-1}}$

### Podmíněnost úlohy na vlastní čísla

Omezíme se na případ, kdy matice A má n lineárně nezávislých vlastních vektorů  $v_1, v_2, \dots, v_n$  odpovídajících vlastním číslům  $\lambda_1, \lambda_2, \dots, \lambda_n$ .

- $\Delta a_{ij}$  ... malé změny v prvcích  $a_{ij}$   $|\Delta a_{ij}| \leq \epsilon$
- porušená matice  $A(\epsilon) = A + \Delta A$  má vlastní čísla  $\lambda_k(\epsilon) = \lambda_k + \Delta \lambda_k$
- dále platí (viz literatura):

$$|\lambda_k(\epsilon) - \lambda_k| \lesssim \alpha_k \epsilon, \quad \text{kde } \alpha_k = \frac{1}{|\cos \alpha_k|}$$

kde  $\alpha_k$  je úhel  $v_k$  a vlastního vektoru  $A^H$  odpovídajícím vlastnímu číslu  $\overline{\lambda_k}$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

- Pro symetrickou matici je
 
$$|\lambda_k(\epsilon) - \lambda_k| \leq \epsilon \quad \alpha_k = 0 \Rightarrow \alpha_k = 1$$

**dobře podmíněná úloha**
- Pro nesymetrickou matici je
 
$$\alpha_k \neq 0 \Rightarrow \alpha_k \text{ může být velmi velké}$$

**špatně podmíněná úloha**

**Příklad**

script v MATLABu

```
A=[1 5 0; 0 3 1; 0 0 2]
AH=ctranspose(A)

[v,c]=eig(A,'nobalance')
[vH,cH]=eig(AH,'nobalance')

disp('-----')
disp(' Vlastni vektory A a AH odpovidajici vlastnimu cislu lambda,')
disp(' cos uhlu, ktery sviraji a tento uhel')
for j=1:length(A)
    disp('-----')
    lambda=c(j,j)
    vlastni_vektor_A=v(:,j)
    vlastni_vektor_AH=vH(:,j)
    cosinus_uhlu=vlastni_vektor_A*vlastni_vektor_AH'...
        /norm(vlastni_vektor_A)/norm(vlastni_vektor_AH)
    uhel=acos(cosinus_uhlu);
    uhel=uhel*180/pi
    pause
end;
```

výsledky v MATLABu

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

```
A =
-1  5  0
 0  3  1
 0  0  2

AH =
-1  0  0
 5  3  0
 0  1  2

v =
1.0000  1.0000 -1.0000
 0  0.8000 -0.6000
 0  0  0.6000

c =
-1  0  0
 0  3  0
 0  0  2

vH =
-0.8000  0.0000 -0.0000
1.0000 -1.0000  0.0000
-0.3333 -1.0000  1.0000

cH =
-1.0000  0  0
 0  3.0000  0
 0  0  2.0000
```

Vlastni vektory A a AH odpovidajici vlastnimu cislu lambda, cos uhlu, ktery sviraji a tento uhel

```
lambda =
-1
vlastni_vektor_A =
 1  0  0
vlastni_vektor_AH =
-0.8000  1.0000
cosinus_uhlu =
-0.6046
uhel =
127.1966
```

-----

```
lambda =
 3
vlastni_vektor_A =
 1.0000  0.8000  0
vlastni_vektor_AH =
 0.0000 -1.0000 -1.0000
cosinus_uhlu =
-0.4417
uhel =
116.2141
```

-----

```
lambda =
 2
vlastni_vektor_A =
-1.0000 -0.6000  0.6000
```

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

**Příklad 2**

$$A = \begin{bmatrix} 20 & 20 & & & \\ & 19 & 20 & & \\ & & 18 & 20 & \\ & & & \dots & \\ & & & & 2 & 20 \\ \epsilon & & & & & 1 \end{bmatrix}$$

Výpočet determinantu  $(A - \lambda I)$  pomocí rozvoje podle posledního řádku:

$$p_A(\lambda) = \det(A - \lambda I) = (20 - \lambda)(19 - \lambda) \dots (1 - \lambda) - 20^{19} \epsilon$$

- pro  $\epsilon = 0 \Rightarrow \boxed{\lambda_{\min} = 1}$
- pro  $\epsilon = 20! \cdot 20^{-19} \doteq 4,64 \cdot 10^{-7}$

$$p_A(\lambda) = (20 - \lambda)(19 - \lambda) \dots (1 - \lambda) - 20^{19} \cdot 20! \cdot 20^{-19} = \lambda \cdot (\dots) = 0 \Rightarrow \boxed{\lambda_{\min} = 0}$$

konst. člen = 20!

- Malé změně  $\epsilon$  odpovídá velká změna vlastního čísla  $\lambda_{\min}$ .
- Vlastní čísla nesymetrických matic jsou citlivá na změnu prvků (citlivost roste s rostoucí vzdáleností od diagonály).

**Mocnná metoda**

Chceme určit vlastní číslo matice A s největší absolutní hodnotou (dominantní vlastní číslo).

Předpoklady:

- A má n lineárně nezávislých vlastních vektorů
- existuje jediné dominantní vlastní číslo
- vlastní čísla lze seřadit:  $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|$ .

Odvození:

- Zvolíme  $y^{(0)}$  jako lineární kombinaci vlastních vektorů
 
$$y^{(0)} = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n.$$
- Sestrojíme posloupnost
 
$$y^{(k)} = A y^{(k-1)}, \quad \text{tj. } y^{(k)} = A^k y^{(0)},$$

$$y^{(k)} = \alpha_1 A^k v_1 + \alpha_2 A^k v_2 + \dots + \alpha_n A^k v_n,$$

3. Platí  $Av_j = \lambda_j v_j$ , proto

$$y^{(k)} = \alpha_1 \lambda_1^k v_1 + \alpha_2 \lambda_2^k v_2 + \dots + \alpha_n \lambda_n^k v_n.$$

4. Vytkneme dominantní vlastní číslo (viz \*)

$$y^{(k)} = \lambda_1^k \left[ \alpha_1 v_1 + \sum_{i=2}^n \alpha_i \left( \frac{\lambda_i}{\lambda_1} \right)^k v_i \right].$$

5. Analogicky vyjádříme  $y^{(k+1)}$ .

6. Vybereme  $j$ -tou složku  $y^{(k+1)}$  a  $y^{(k)}$ , vydělíme je a provedeme limitní přechod

$$\lim_{k \rightarrow \infty} \frac{y_j^{(k+1)}}{y_j^{(k)}} = \lim_{k \rightarrow \infty} \frac{\lambda_1^{k+1} (\alpha_1 v_{1j} + \epsilon_{k+1,j})}{\lambda_1^k (\alpha_1 v_{1j} + \epsilon_{kj})} = \lambda_1$$

**Příklad**

Mocninnou metodou stanovte dominantní vlastní číslo matice A, kde

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad a \quad y^{(0)} = [1, 1, 1]^T.$$

**Řešení:** Použijeme iterační formuli

$$y^{(k+1)} = Ay^{(k)}, \quad \text{pro } k = 0, 1, \dots$$

$$y^{(1)} = [2; 3; 2]^T \quad \lambda_1^{(1)} = \frac{y_2^{(1)}}{y_2^{(0)}} = 3,$$

$$y^{(2)} = [5; 7; 5]^T \quad \lambda_1^{(2)} = \frac{7}{3} \approx 2,3333,$$

$$y^{(3)} = [12; 17; 12]^T \quad \lambda_1^{(3)} = \frac{17}{7} \approx 2,4285,$$

$$y^{(4)} = [29; 41; 29]^T \quad \lambda_1^{(4)} = \frac{41}{17} \approx 2,4117,$$

$$y^{(5)} = [70; 99; 70]^T \quad \lambda_1^{(5)} = \frac{99}{41} \approx 2,4146.$$

**Poznámka:**

Abychom zamezili přetečení, resp. podtečení při zobrazení čísel v počítači je vhodné v každém kroku

normovat vektor  $y^{(k)}$  (norma  $y^{(k)}$  roste, resp. klesá pro vlastní číslo v absolutní hodnotě větší, resp. menší než 1).

$$y^{(k)} := \frac{y^{(k)}}{\|y^{(k)}\|}$$

výsledky v MATLABu

```
Mocninná metoda pro vypočet dominantního vlastního čísla matice A
A =
  900    20     1
    20   500    30
     1    30   100

| k | y(1)_k | y(2)_k | y(3)_k || lambda_k |
-----|-----|-----|-----|-----|
| 0 | 1.000000e+000 | 1.000000e+000 | 1.000000e+000 || |
| 1 | 9.2100000e+002 | 5.500000e+002 | 1.310000e+002 || 921.000000 |
| 2 | 8.400310e+005 | 2.973500e+005 | 3.052100e+004 || 912.0857763 |
| 3 | 7.620054e+008 | 1.663913e+008 | 1.281263e+007 || 907.1158338 |
| 4 | 6.891455e+011 | 9.882011e+010 | 7.035006e+009 || 904.3840077 |
| 5 | 6.222144e+014 | 6.340402e+013 | 4.357249e+012 || 902.8781109 |
| 6 | 5.612654e+017 | 4.427701e+016 | 2.960060e+015 || 902.0450147 |
| 7 | 5.060274e+020 | 3.345262e+019 | 2.185582e+018 || 901.5830307 |
| 8 | 4.560959e+023 | 2.691242e+022 | 1.728164e+021 || 901.3264854 |
| 9 | 4.110263e+026 | 2.262997e+025 | 1.436285e+024 || 901.1839104 |
|10 | 3.703777e+029 | 1.957860e+028 | 1.233554e+027 || 901.1046393 |

>> eig(A)
ans =
 1.0e+002 *
 0.977622158203950
 5.012324900167472
 9.010052941628578
```

výsledky v MATLABu

Mocninná metoda pro vypočet dominantního vlastního čísla matice A s normováním vlastního vektoru v každé iteraci

```
A =
  900    20     1
    20   500    30
     1    30   100

| k | y(1)_k | y(2)_k | y(3)_k || lambda_k |
-----|-----|-----|-----|
| 0 | 1.0000000 | 1.0000000 | 1.0000000 || |
| 1 | 921.0000000 | 550.0000000 | 131.0000000 || 921.000000 |
| 1 | 1.0000000 | 0.5971770 | 0.1422367 || |
| 2 | 912.0857763 | 322.8555917 | 33.1389794 || 912.085776 |
| 1 | 1.0000000 | 0.3539750 | 0.0363332 || |
| 3 | 907.1158338 | 198.0775114 | 15.2525693 || 907.115834 |
| 1 | 1.0000000 | 0.2183597 | 0.0168144 || |
| 4 | 904.3840077 | 129.6842642 | 9.2322257 || 904.384008 |
| 1 | 1.0000000 | 0.1433951 | 0.0102083 || |
| 5 | 902.8781109 | 92.0038151 | 6.3226842 || 902.878111 |
| 1 | 1.0000000 | 0.1019006 | 0.0070028 || |
| 6 | 902.0450147 | 71.1603809 | 4.7572988 || 902.045015 |
| 1 | 1.0000000 | 0.0788878 | 0.0052739 || |
| 7 | 901.5830307 | 59.6021361 | 3.8940255 || 901.583031 |
| 1 | 1.0000000 | 0.0661083 | 0.0043191 || |
| 8 | 901.3264854 | 53.1837310 | 3.4151593 || 901.326485 |
| 1 | 1.0000000 | 0.0590061 | 0.0037890 || |
| 9 | 901.1839104 | 49.6167046 | 3.1490857 || 901.183910 |
| 1 | 1.0000000 | 0.0550572 | 0.0034944 || |
|10 | 901.1046393 | 47.6334548 | 3.0011561 || 901.104639 |
| 1 | 1.0000000 | 0.0528612 | 0.0033305 || |
```

výsledky v MATLABu

Mocninná metoda pro vypočet dominantního vlastního čísla matice A

```
A =
  1/1000    -3/10000    -1/2000
  1/5000     1/200     -1/10000
 -1/1000     1/500     3/1000

| k | y(1)_k | y(2)_k | y(3)_k || lambda_k |
-----|-----|-----|-----|
| 0 | 1.000000e+000 | 1.000000e+000 | 1.000000e+000 || |
| 1 | 2.000000e-004 | 5.100000e-003 | 4.000000e-003 || 0.0051000 |
| 2 | -3.330000e-006 | 2.514000e-005 | 2.200000e-005 || 0.0049294 |
| 3 | -2.187200e-008 | 1.228340e-007 | 1.196100e-007 || 0.0048860 |
| 4 | -1.185272e-010 | 5.978346e-010 | 6.263700e-010 || 0.0052368 |
| 5 | -6.110626e-013 | 2.902831e-012 | 3.193306e-012 || 0.0050981 |
| 6 | -3.078566e-015 | 1.407261e-014 | 1.599664e-014 || 0.0050094 |
| 7 | -1.529867e-017 | 6.814767e-017 | 7.921371e-017 || 0.0049519 |
| 8 | -7.534983e-020 | 3.297572e-019 | 3.892351e-019 || 0.0049137 |
| 9 | -3.688946e-022 | 1.594793e-021 | 1.902570e-021 || 0.0048880 |
|10 | -1.798617e-024 | 7.709928e-024 | 9.266189e-024 || 0.0048704 |

>> eig(A)
ans =
 0.000770409049726
 0.003399468175334
 0.004830122774940
```

výsledky v MATLABu

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

Mocinnina metoda pro vypocet dominantního vlastního čísla matice A s normovaním vlastního vektoru v každé iteraci

```
A =
    1/1000    -3/10000    -1/2000
    1/5000     1/200     -1/10000
   -1/1000     1/500      3/1000
```

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	1.0000000	1.0000000	1.0000000	
1	0.0002000	0.0051000	0.0040000	0.005100
2	0.0392157	1.0000000	0.7843137	
3	-0.0006529	0.0049294	0.0043137	0.004929
4	-0.1324582	1.0000000	0.8750994	
5	-0.0008700	0.0048860	0.0047578	0.004886
6	-0.1780614	1.0000000	0.9737532	
7	-0.0009649	0.0048670	0.0050993	0.005237
8	-0.1892287	0.9544432	1.0000000	
9	-0.0009756	0.0046344	0.0050981	0.005098
10	-0.1913573	0.9090360	1.0000000	
11	-0.0009641	0.0044069	0.0050094	0.005009
12	-0.1924507	0.8797227	1.0000000	
13	-0.0009564	0.0042601	0.0049519	0.004952
14	-0.1931316	0.8603014	1.0000000	
15	-0.0009512	0.0041629	0.0049137	0.004914
16	-0.1935843	0.8471929	1.0000000	
17	-0.0009477	0.0040972	0.0048880	0.004888
18	-0.1938928	0.8382309	1.0000000	
19	-0.0009454	0.0040524	0.0048704	0.004870
20	-0.1941054	0.8320495	1.0000000	

**Poznámka:**

Nejlepší aproximaci dostaneme, dělíme-li složky, které mají největší absolutní hodnotu. Obecně nelze použít libovolnou složku vektoru  $y^{(k)}$  neboť odpovídající vlastní vektor jí může mít nulovku.

výsledky v MATLABu

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

```
A =
    1    1    0
    0    2    0
    0    0    3
```

```
>> [v,c]=eig(A,'nobalance')
```

```
v =
    1    1    0
    0    1    0
    0    0    1
```

```
c =
    1    0    0
    0    2    0
    0    0    3
```

Mocinnina metoda pro vypocet dominantního vlastního čísla matice A

k	y(1)_k	y(2)_k	y(3)_k	lambda_k	1s	2s	3s
0	2.000000e+00	1.000000e+00	1.000000e+00				
1	3.000000e+00	2.000000e+00	3.000000e+00	1.5000000	2	3	
2	5.000000e+00	4.000000e+00	9.000000e+00	1.6666667	2	3	
3	9.000000e+00	8.000000e+00	2.700000e+01	1.8000000	2	3	
4	1.700000e+01	1.600000e+01	8.100000e+01	1.8888889	2	3	
5	3.300000e+01	3.200000e+01	2.430000e+02	1.9411765	2	3	
6	6.500000e+01	6.400000e+01	7.290000e+02	1.9696970	2	3	
7	1.290000e+02	1.280000e+02	2.187000e+03	1.9846154	2	3	
8	2.570000e+02	2.560000e+02	6.561000e+03	1.9922481	2	3	
9	5.130000e+02	5.120000e+02	1.968300e+04	1.9961089	2	3	
10	1.025000e+03	1.024000e+03	5.904900e+04	1.9980507	2	3	

**Poznámka:**

Při praktickém použití mocinné metody neověřujeme, zda jsou splněny předpoklady odvozené. Zadaná matice nemusí mít jediné dominantní vlastní číslo nebo počet lineárně nezávislých vlastních vektorů může být menší než řád matice. Při nesplnění předpokladů odvození může být konvergence pomalá. Další nevýhodou mocinné metody je potom odhad chyby získané aproximace.

výsledky v MATLABu

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

```
A =
    3    1    1
   -1    1    0
    0    0    1
```

```
>> [v,c]=eig(A,'nobalance')
```

```
v =
    1.0000    -1.0000    0.0000
   -1.0000    1.0000   -1.0000
    0         0         1.0000
```

```
c =
    2.0000         0         0
    0    2.0000         0
    0         0    1.0000
```

Mocinnina metoda pro vypocet dominantního vlastního čísla matice A

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	1.000000e+000	1.000000e+000	1.000000e+000	
1	5.000000e+000	0.000000e+000	1.000000e+000	5.0000000
2	1.600000e+001	-5.000000e+000	1.000000e+000	3.2000000
3	4.400000e+001	-2.100000e+001	1.000000e+000	2.7500000
4	1.120000e+002	-6.500000e+001	1.000000e+000	2.5454545
5	2.720000e+002	-1.770000e+002	1.000000e+000	2.4285714
6	6.400000e+002	-4.490000e+002	1.000000e+000	2.3529412
7	1.472000e+003	-1.089000e+003	1.000000e+000	2.3000000
8	3.328000e+003	-2.561000e+003	1.000000e+000	2.2608696
9	7.424000e+003	-5.889000e+003	1.000000e+000	2.2307692
10	1.638400e+004	-1.331300e+004	1.000000e+000	2.2068966
150	8.556839e+016	-8.219069e+016	1.000000e+000	2.0402685
1000	1.914152e+032	-1.876123e+032	1.000000e+000	2.0200669
1500	3.225580e+047	-3.182762e+047	1.000000e+000	2.0133630
2000	4.836884e+062	-4.788675e+062	1.000000e+000	2.0100167
2500	6.802785e+077	-6.748508e+077	1.000000e+000	2.0080107
3000	9.187032e+092	-9.125921e+092	1.000000e+000	2.0066741
10000	1.608334e+304	-1.605120e+304	1.000000e+000	2.0020007

KMA/NM  
13.2.2013

Numerické metody  
Josef Daněk

Z uvedeného příkladu je dále vidět, že je opravdu vhodné normovat vektor  $y^{(k)}$  a zabránit tak přetečení.

**Poznámka:**

Při praktickém použití mocinné metody použijeme například počáteční volbu vektoru  $y^{(0)} = [1, 1, \dots, 1]^T$ . Je-li ovšem vektor  $y^{(0)}$  takovou lineární kombinací vlastních vektorů, že koeficient u vlastního vektoru odpovídajícího dominantnímu vlastnímu číslu bude roven 0, potom mocinná metoda nevypočte dominantní vlastní číslo, ale nejbližší nižší, u kterého u odpovídajícího vlastního vektoru bude nenulový koeficient.

Pokud bychom prováděli výpočet dostatečně dlouho, dojde vlivem zaokrouhlovacích chyb k tomu, že u příslušné iterace  $y^{(k)}$  bude již zmiňovaný koeficient u vlastního vektoru odpovídajícího dominantnímu vlastnímu číslu již nenulový a metoda nakonec dominantní vlastní číslo naleznе.

výsledky v MATLABu

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

```

A =
  4   -3   2
  0   2   0
  0  -4   6

>> [v,c]=eig(A,'noblance')

v =
  1.0000   1.0000  -0.5000
         0         0  -1.0000
         0   1.0000  -1.0000

c =
  4   0   0
  0   6   0
  0   0   2

>> y=[1 1 1]'
>> alpha=(v\y)'

alpha =
  0.5000         0  -1.0000
  
```

Mocninná metoda pro výpočet dominantního vlastního čísla matice A s normováním vlastního vektoru v každé iteraci

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	1.0000000	1.0000000	1.0000000	
1	3.0000000	2.0000000	2.0000000	3.0000000
1	1.0000000	0.6666667	0.6666667	
2	3.3333333	1.3333333	1.3333333	3.3333333
1	1.0000000	0.4000000	0.4000000	
3	3.6000000	0.8000000	0.8000000	3.6000000
1	1.0000000	0.2222222	0.2222222	
4	3.7777778	0.4444444	0.4444444	3.7777778
1	1.0000000	0.1176471	0.1176471	
5	3.8823529	0.2352941	0.2352941	3.8823529
1	1.0000000	0.0606061	0.0606061	
6	3.9393939	0.1212121	0.1212121	3.9393939
1	1.0000000	0.0307692	0.0307692	
7	3.9692308	0.0615385	0.0615385	3.9692308
1	1.0000000	0.0155039	0.0155039	
8	3.9844961	0.0310078	0.0310078	3.9844961
1	1.0000000	0.0077821	0.0077821	
9	3.9922179	0.0155642	0.0155642	3.9922179
1	1.0000000	0.0038986	0.0038986	
10	3.9961014	0.0077973	0.0077973	3.9961014
1	1.0000000	0.0019512	0.0019512	
11	3.9980488	0.0039024	0.0039024	3.9980488
1	1.0000000	0.0009761	0.0009761	
12	3.9990239	0.0019522	0.0019522	3.9990239
1	1.0000000	0.0004882	0.0004882	
13	3.9995118	0.0009763	0.0009763	3.9995118
1	1.0000000	0.0002441	0.0002441	

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

Poznámka:  
Iterační proces ukončujeme použitím zastavovací podmínky ve tvaru  $|\lambda_1^{(k+1)} - \lambda_1^{(k)}| < \epsilon$   
Posuďte výsledky získané pro následující příklad. Kde je problém?  
výsledky v MATLABu

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

```

A =
  3   3   0
  0   4   2
  0   0   1

>> [v,c]=eig(A,'noblance')

v =
  1.0000   1.0000   1.0000
         0   0.3333  -0.6667
         0         0   1.0000

c =
  3   0   0
  0   4   0
  0   0   1

>> y=[1 1 1]'
>> alpha=(v\y)'

alpha =
  -5   5   1
  
```

Mocninná metoda pro výpočet dominantního vlastního čísla matice A

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	1.000000e+000	1.000000e+000	1.000000e+000	
1	6.000000e+000	6.000000e+000	1.000000e+000	6.0000000
2	3.600000e+001	2.600000e+001	1.000000e+000	6.0000000

```

>> y=[1 0 1]'
>> alpha=(v\y)'

alpha =
  -2   2   1
  
```

Mocninná metoda pro výpočet dominantního vlastního čísla matice A

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	1.000000e+000	0.000000e+000	1.000000e+000	
1	3.000000e+000	2.000000e+000	1.000000e+000	3.0000000
2	1.500000e+001	1.000000e+001	1.000000e+000	5.0000000
3	7.500000e+001	4.200000e+001	1.000000e+000	5.0000000

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

```

>> y=[3 2 1]'
>> alpha=(v\y)'

alpha =
  -6   8   1
  
```

Mocninná metoda pro výpočet dominantního vlastního čísla matice A

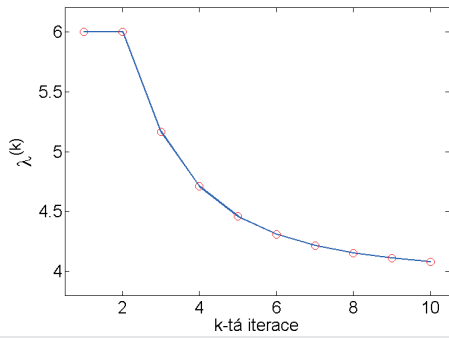
k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	3.000000e+000	2.000000e+000	1.000000e+000	
1	1.500000e+001	1.000000e+001	1.000000e+000	5.0000000
2	7.500000e+001	4.200000e+001	1.000000e+000	5.0000000

Všechny předpoklady byly splněny, byla použita i vhodná počáteční volba vektoru  $y^{(0)}$ .  
Jediné, co se stalo je skutečnost, že v posloupnosti přibližných řešení generovaných mocninnou metodou se objevily dva po sobě jdoucí stejné členy, které zdaleka nebyly limitou této posloupnosti.

výsledky v MATLABu

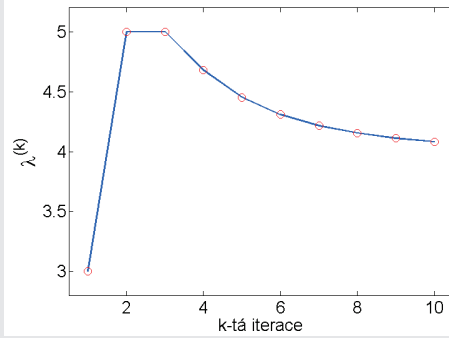
Mocninná metoda pro výpočet dominantního vlastního čísla matice A

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	1.000000e+000	1.000000e+000	1.000000e+000	
1	6.000000e+000	6.000000e+000	1.000000e+000	6.0000000
2	3.600000e+001	2.600000e+001	1.000000e+000	6.0000000
3	1.860000e+002	1.060000e+002	1.000000e+000	5.1666667
4	8.760000e+002	4.260000e+002	1.000000e+000	4.7096774
5	3.906000e+003	1.706000e+003	1.000000e+000	4.4589041
6	1.683600e+004	6.826000e+003	1.000000e+000	4.3102919
7	7.098600e+004	2.730600e+004	1.000000e+000	4.2163222
8	2.948760e+005	1.092260e+005	1.000000e+000	4.1540022
9	1.212306e+006	4.369060e+005	1.000000e+000	4.1112400
10	4.947636e+006	1.747626e+006	1.000000e+000	4.0811775



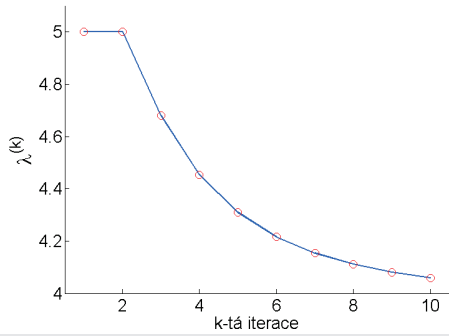
Mocninna metoda pro vypocet dominantniho vlastniho cisla matice A

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	1.000000e+000	0.000000e+000	1.000000e+000	
1	3.000000e+000	2.000000e+000	1.000000e+000	3.0000000
2	1.500000e+001	1.000000e+001	1.000000e+000	5.0000000
3	7.500000e+001	4.200000e+001	1.000000e+000	5.0000000
4	3.510000e+002	1.700000e+002	1.000000e+000	4.6800000
5	1.563000e+003	6.820000e+002	1.000000e+000	4.4529915
6	6.735000e+003	2.730000e+003	1.000000e+000	4.3090211
7	2.839500e+004	1.092200e+004	1.000000e+000	4.2160356
8	1.179510e+005	4.369000e+004	1.000000e+000	4.1539356
9	4.849230e+005	1.747620e+005	1.000000e+000	4.1112242
10	1.979055e+006	6.990500e+005	1.000000e+000	4.0811737



Mocninna metoda pro vypocet dominantniho vlastniho cisla matice A

k	y(1)_k	y(2)_k	y(3)_k	lambda_k
0	3.000000e+000	2.000000e+000	1.000000e+000	
1	1.500000e+001	1.000000e+001	1.000000e+000	5.0000000
2	7.500000e+001	4.200000e+001	1.000000e+000	5.0000000
3	3.510000e+002	1.700000e+002	1.000000e+000	4.6800000
4	1.563000e+003	6.820000e+002	1.000000e+000	4.4529915
5	6.735000e+003	2.730000e+003	1.000000e+000	4.3090211
6	2.839500e+004	1.092200e+004	1.000000e+000	4.2160356
7	1.179510e+005	4.369000e+004	1.000000e+000	4.1539356
8	4.849230e+005	1.747620e+005	1.000000e+000	4.1112242
9	1.979055e+006	6.990500e+005	1.000000e+000	4.0811737
10	8.034315e+006	2.796202e+006	1.000000e+000	4.0596724



**Poznámka:**

**Pro urychlování konvergence metody**

- lze použít např. Aitkenův proces,
- pokud platí, že  $\lambda_1$  a  $\lambda_2$  jsou si velmi blízka, rychlost konvergence mocninné metody bude malá; předpokládáme-li např., že jsou všechna vlastní čísla reálná, lze použít Wilkinsonovu metodu:  
A má vlastní čísla  $\lambda_1, \lambda_2, \dots, \lambda_n$

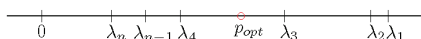
$\hat{A} = A - pI$  má vlastní čísla  $\lambda_1 - p, \lambda_2 - p, \dots, \lambda_n - p$

Uvažujme pro jednoduchost, že jsou všechna  $\lambda_i > 0$ .

Pomalou konvergenci způsobuje podíl  $\left| \frac{\lambda_2}{\lambda_1} \right| \lesssim 1$ .

Chceme tento podíl co nejvíce zmenšit:  $\frac{\lambda_2 - p}{\lambda_1 - p} < \frac{\lambda_2}{\lambda_1}$

Jak musíme volit  $p$ ?  $p_{opt} = \frac{\lambda_2 + \lambda_1}{2}$  ... představuje posunutý počátek



**Příklad:**

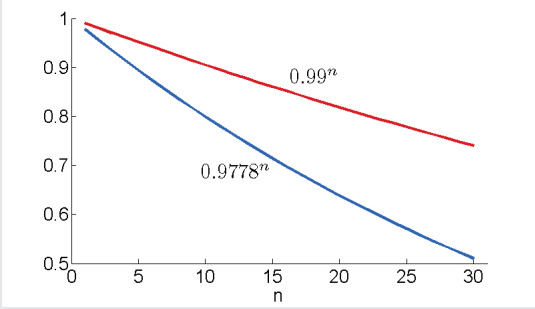
$A = \begin{bmatrix} 100 & & \\ & 99 & \\ & & 11 \end{bmatrix}$  ... vlastní čísla  $\lambda_1 = 100, \lambda_2 = 99, \lambda_3 = 11 \Rightarrow p_{opt} = \frac{99+11}{2} = 55$

$\hat{A} = A - 55I = \begin{bmatrix} 45 & & \\ & 44 & \\ & & -44 \end{bmatrix}$  ... vlastní čísla  $\hat{\lambda}_1 = 45, \hat{\lambda}_2 = 44, \hat{\lambda}_3 = -44$

$\frac{\lambda_2}{\lambda_1} = \frac{99}{100} = 0,99 \quad \frac{\hat{\lambda}_2}{\hat{\lambda}_1} = \frac{44}{45} \approx 0,9778$

**výsledky v MATLABu**

n	$0,99^n$	$0,9778^n$
1	0.9900	0.9778
2	0.9801	0.9561
3	0.9703	0.9349
4	0.9606	0.9141
5	0.9510	0.8938
6	0.9415	0.8740
7	0.9321	0.8546
8	0.9227	0.8356
9	0.9135	0.8171
10	0.9044	0.7989
11	0.8953	0.7812
12	0.8864	0.7638
13	0.8775	0.7469
14	0.8687	0.7303
15	0.8601	0.7141
16	0.8515	0.6982
17	0.8429	0.6827
18	0.8345	0.6676
19	0.8262	0.6528
20	0.8179	0.6383
21	0.8097	0.6241
22	0.8016	0.6102
23	0.7936	0.5967
24	0.7857	0.5834
25	0.7778	0.5705
26	0.7700	0.5578
27	0.7623	0.5454
28	0.7547	0.5333
29	0.7472	0.5215
30	0.7397	0.5099



Metoda Rayleighova podílu

Chceme určit vlastní číslo matice A s největší absolutní hodnotou (dominantní vlastní číslo). Při odvození metody Rayleighova podílu budeme navíc (oproti mocninné metodě) předpokládat, že matice A je symetrická (reálná). Potom musí být vlastní vektory ortonormální (v\_i^T v\_j = 0 pro i != j a v\_i^T v\_i = 1).

Odvození:

6. krok z odvození mocninné metody nahradíme vyjádřením součinu y^{(k)T} y^{(k)}

$$y^{(k)T} y^{(k)} = \lambda_1^k \left[ \alpha_1 v_1^T + \sum_{i=2}^n \alpha_i \left( \frac{\lambda_i}{\lambda_1} \right)^k v_i^T \right] \cdot \lambda_1^k \left[ \alpha_1 v_1 + \sum_{i=2}^n \alpha_i \left( \frac{\lambda_i}{\lambda_1} \right)^k v_i \right] = \lambda_1^{2k} \left[ \alpha_1^2 + \sum_{i=2}^n \alpha_i^2 \left( \frac{\lambda_i}{\lambda_1} \right)^{2k} \right] = \lambda_1^{2k} \varepsilon_k$$

a součinu y^{(k)T} y^{(k+1)T}

$$y^{(k)T} y^{(k+1)} = \lambda_1^k \left[ \alpha_1 v_1^T + \sum_{i=2}^n \alpha_i \left( \frac{\lambda_i}{\lambda_1} \right)^k v_i^T \right] \cdot \lambda_1^{k+1} \left[ \alpha_1 v_1 + \sum_{i=2}^n \alpha_i \left( \frac{\lambda_i}{\lambda_1} \right)^{k+1} v_i \right] = \lambda_1^{2k+1} \left[ \alpha_1^2 + \sum_{i=2}^n \alpha_i^2 \left( \frac{\lambda_i}{\lambda_1} \right)^{2k+1} \right] = \lambda_1^{2k+1} \varepsilon_{k+1}$$

Dostáváme:

$$\lim_{k \rightarrow \infty} \frac{y^{(k)T} A y^{(k)}}{y^{(k)T} y^{(k)}} = \lim_{k \rightarrow \infty} \frac{y^{(k)T} y^{(k+1)}}{y^{(k)T} y^{(k)}} = \frac{\lambda_1^{2k+1} (\alpha_1^2 + \sum_{i=2}^n \alpha_i^2 \frac{\varepsilon_{k+1}}{\varepsilon_k})}{\lambda_1^{2k} (\alpha_1^2 + \sum_{i=2}^n \alpha_i^2 \frac{\varepsilon_k}{\varepsilon_{k-1}})} = \lambda_1$$

Poznámka: Součin \varepsilon\_k^T \varepsilon\_k konverguje k nule (pro k \to \infty) zhruba dvakrát rychleji než \varepsilon\_k k nulovému vektoru \implies metoda Rayleighova podílu bude rychlejší než mocninná metoda.

Příklad

Metodou Rayleighova podílu určete dominantní vlastní číslo matice A, kde

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad a \quad y^{(0)} = [1; 1; 1]^T$$

Řešení:

$$y^{(1)} = [2; 3; 2]^T \quad \lambda_1^{(1)} = \frac{y^{(0)T} y^{(1)}}{y^{(0)T} y^{(0)}} = \frac{4}{3} \approx 2,3333,$$

$$y^{(2)} = [5; 7; 5]^T \quad \lambda_1^{(2)} = \frac{14}{5} \approx 2,8, \quad \lambda_1^{(3)} = \frac{60+119+60}{25+49+25} = \frac{239}{99} \approx 2,41417.$$

$$y^{(3)} = [12; 17; 12]^T$$

Příklad 3

Pro stejné zadání symetrické matice A porovnejme rychlost konvergence mocninné metody a metody Rayleighova podílu.

$$A = \begin{bmatrix} 60 & 20 & 10 & 1 \\ 20 & 50 & 10 & 2 \\ 10 & 10 & 30 & 5 \\ 1 & 2 & 5 & 10 \end{bmatrix}, \quad y^{(0)} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \varepsilon = 10^{-5}$$

výsledky v MATLABu

```
A =
    60    20    10     1
    20    50    10     2
    10    10    30     5
     1     2     5    10

>> [v,c]=eig(A,'nobalance')

v =
    0.029201136324116    0.070393944798935   -0.657594927428575   -0.749507103097250
   -0.002755406652908    0.347503151150767    0.720730957555407   -0.599817350945878
   -0.241058474917619   -0.907169537175591    0.206770509289230   -0.276007606741715
    0.970067272431118   -0.226560551059880    0.073224003781179   -0.047728910858600

c =
    8.781938031360916     0              0              0
     0    26.642118061325501     0              0
     0              0    34.824093743363321     0
     0              0              0    79.751850163950266

>> y=[1 1 1 1]'
>> alpha=(v\y)

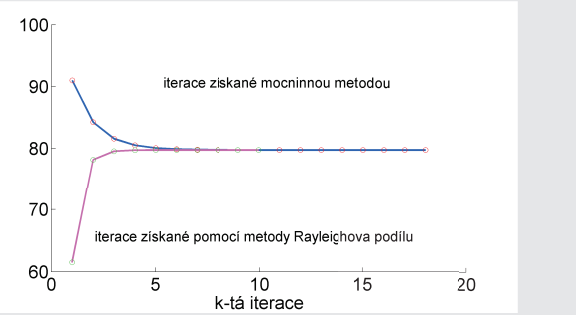
alpha =
    0.755454527184707   -0.715832992285768    0.343130543197241   -1.673060971643443
```

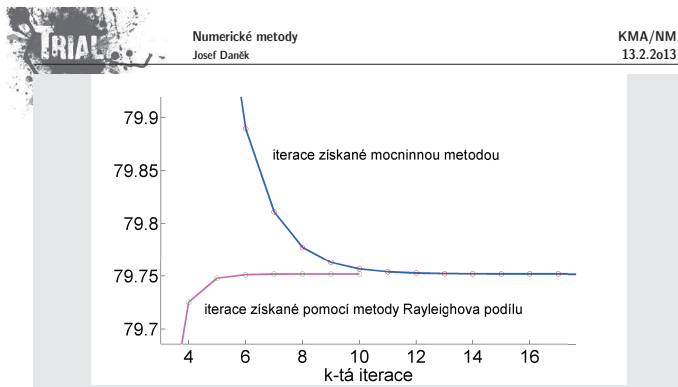
```
Mocninná metoda pro výpočet dominantního vlastního čísla matice A

| k | y(1)_k | y(2)_k | y(3)_k | y(4)_k | lambda_k
---|---|---|---|---|---
0 | 1.000000e+000 | 1.000000e+000 | 1.000000e+000 | 1.000000e+000 || 1.000000
1 | 9.100000e+001 | 8.200000e+001 | 5.500000e+001 | 1.800000e+001 || 91.000000
2 | 7.668000e+003 | 6.506000e+003 | 3.470000e+003 | 7.100000e+002 || 76.680000
3 | 6.256100e+005 | 5.147800e+005 | 2.493900e+005 | 4.513000e+004 || 6.256100
4 | 5.037123e+007 | 4.083536e+007 | 1.911125e+007 | 3.353420e+006 || 5.037123
5 | 4.033447e+009 | 3.247012e+009 | 1.502171e+009 | 2.611324e+008 || 4.033447
6 | 3.222990e+011 | 2.585635e+011 | 1.191754e+011 | 2.064965e+010 || 3.222990
7 | 2.571747e+013 | 2.060583e+013 | 9.486443e+012 | 1.641730e+012 || 2.571747
8 | 2.051671e+015 | 1.642789e+015 | 7.560349e+014 | 1.307786e+014 || 2.051671
9 | 1.636471e+017 | 1.309947e+017 | 6.027953e+016 | 1.042521e+016 || 1.636471
10 | 1.305194e+019 | 1.044633e+019 | 4.806931e+018 | 8.312864e+017 || 1.305194
11 | 1.040944e+021 | 8.330871e+020 | 3.833471e+020 | 6.29211e+019 || 1.040944
12 | 8.301813e+022 | 6.643928e+022 | 3.057218e+022 | 5.286774e+021 || 8.301813
13 | 6.620882e+024 | 5.298622e+024 | 2.438173e+024 | 4.216253e+023 || 6.620882
14 | 5.280287e+026 | 4.225737e+026 | 1.944484e+026 | 3.362525e+025 || 5.280287
15 | 4.211131e+028 | 3.370099e+028 | 1.550760e+028 | 2.681670e+027 || 4.211131
16 | 3.358456e+030 | 2.687715e+030 | 1.236759e+030 | 2.138680e+029 || 3.358456
17 | 2.678431e+032 | 2.143502e+032 | 9.863383e+031 | 1.705636e+031 || 2.678431
18 | 2.136099e+034 | 1.709482e+034 | 7.866230e+033 | 1.360276e+033 || 2.136099
```

Metoda Rayleighova podílu pro výpočet dominantního vlastního čísla matice A

k	y(1)_k	y(2)_k	y(3)_k	y(4)_k	lambda_k
0	1.000000e+000	1.000000e+000	1.000000e+000	1.000000e+000	1.000000
1	9.100000e+001	8.200000e+001	5.500000e+001	1.800000e+001	61.500000
2	7.668000e+003	6.506000e+003	3.470000e+003	7.100000e+002	76.680000
3	6.256100e+005	5.147800e+005	2.493900e+005	4.513000e+004	76.6806714
4	5.037123e+007	4.083536e+007	1.911125e+007	3.353420e+006	79.7252270
5	4.033447e+009	3.247012e+009	1.502171e+009	2.611324e+008	79.7478446
6	3.222990e+011	2.585635e+011	1.191754e+011	2.064965e+010	79.7512057
7	2.571747e+013	2.060583e+013	9.486443e+012	1.641730e+012	79.7517406
8	2.051671e+015	1.642789e+015	7.560349e+014	1.307786e+014	79.7518308
9	1.636471e+017	1.309947e+017	6.027953e+016	1.042521e+016	79.7518466
10	1.305194e+019	1.044633e+019	4.806931e+018	8.312864e+017	79.7518495





**Poznámka:**

Pokud jsme vypočítali  $\lambda_1, v_1$  a chceme určit další vlastní čísla, resp. vlastní vektory  $\lambda_2, v_2, \lambda_3, v_3, \dots$  (ovšem ne všechny), můžeme použít metody využívající znalosti  $\lambda_1, v_1$  atd.

• **Maticová redukce**

**Věta:** Nechť  $\lambda_1$  je vlastní číslo matice  $A$  a  $v_1$  jemu odpovídající vlastní vektor. Nechť  $w$  je libovolný vektor, pro který  $w^T v_1 = 1$ . Pak matice

$$W_1 = A - \lambda_1 v_1 w^T$$

má stejná vlastní čísla jako matice  $A$ , s výjimkou vlastního čísla  $\lambda_1$ , které je nahrazeno číslem 0 ( $W_1 \dots$  redukovaná matice).

**Otázka:** Jak volit vektor  $w$  ?

1. **Hotellingova redukce**

$w \dots$  levý vlastní vektor vlastního čísla  $\lambda_1$  (je normalizován:  $w^T v_1 = 1$ )

obvykle levý vlastní vektor neznáme a může být  $w^T v_1 = 0$

užijeme tuto metodu pro symetrické matice, protože potom  $w = v_1$

(tj. pravý a levý vlastní vektor odpovídající stejnému vlastnímu číslu je stejný)

2. **Wielandtova redukce**

(viz literatura)

3. **podobnostní redukce**

(viz literatura)

• **Anihilační postupy**

Je-li  $w$  libovolný vektor a  $\lambda_1, v_1$  vlastní číslo a vektor matice  $A$ , pak vektor

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$u = (A - \lambda_1 I)w$

nemá složku ve směru vektoru  $v_1$

Cv. vyjádříme vektor  $w$  jako lineární kombinaci vlastních vektorů  $v_i$  a ověříme

$$w = \sum_{i=1}^n \beta_i v_i$$

$$u = (A - \lambda_1 I) \left( \sum_{i=1}^n \beta_i v_i \right) = \sum_{i=1}^n \beta_i \frac{A v_i}{\lambda_i v_i} - \beta_i \lambda_1 v_i = \beta_1 \lambda_1 v_1 - \beta_1 \lambda_1 v_1 + \sum_{i=2}^n (\beta_i \lambda_i v_i - \lambda_1 \beta_i v_i) =$$

$$= 0 \cdot v_1 + \sum_{i=2}^n \beta_i (\lambda_i - \lambda_1) v_i$$

□

- Použijeme-li  $u$  jako vstup do mocninné metody, získáme  $\lambda_2, v_2$  (pozor na problém se zaokrouhlovacími chybami).
- Abychom odstranili tento problém, odbouráváme stále složku ve směru  $v_1$

$$u = (A - \lambda_1 I)u$$

**Charakteristika metod na řešení úplného problému:**

1) metody založené na výpočtech vlastních čísel **pomocí charakteristického polynomu**

Nevýhodné pro velká  $n$  (řád matice  $A$ ), protože je obtížné vypočítat  $p_A(\lambda) = \det(A - \lambda I)$  z definice determinantu.

2) **metody využívající podobnosti matic**

Tato kategorie metod využívá faktu, že podobné matice mají stejná vlastní čísla.

**Princip:** konstruujeme posloupnost navzájem podobných matic, která konverguje k matici, jejíž vlastní čísla se dají jednoduchým způsobem určit.

3) **smíšené metody**

založené na převodu obecné matice na matici trojúhelníkovou (např. Givensova, Householderova a Lanczosova metoda) a následný efektivní výpočet kořenů charakteristického polynomu této upravené matice.

**Metoda LU-rozkladu (LR-transformace, LR-algoritmus)**

(Lower-Upper, Left-Right)

$A = LU \dots$  rozklad matice  $A$  na dolní trojúhelníkovou matici  $L$  a horní trojúhelníkovou matici  $U$ , kde na diagonále matice  $L$  jsou pro jednoznačnost rozkladu jednotky.

Sestrojíme matici  $B$ , která bude podobná matici  $A$ .

$$B = UL \quad (U = L^{-1}A \Rightarrow B = UL = L^{-1}AL)$$

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

**Postup:**

Sestrojíme posloupnost matic  $A_k$ :

- $A_0 = A, k = 0$
- provedeme LU rozklad matice  $A_k = L_k U_k$
- sestrojíme matici  $A_{k+1} = U_k L_k$
- je-li matice  $A_{k+1}$  horní trojúhelníková  $\Rightarrow$  konec, jinak  $k = k + 1$  a jdi na (ii)

**Poznámka:**

Dá se ukázat, že když matice  $B_k = L_0 L_1 \dots L_k$  konvergují k regulární matici, potom matice  $A_k$  také konvergují, a to k horní trojúhelníkové matici s vlastními čísly na diagonále. Platí

$$A_{k+1} = \underbrace{L_k^{-1} A_k L_k}_{U_k}$$

a tedy

$$A_{k+1} = \underbrace{L_k^{-1} L_{k-1}^{-1} \dots L_0^{-1} A_0}_{B_{k+1}^{-1}} \underbrace{L_0 L_1 \dots L_k}_{B_{k+1}}$$

**Poznámka:**

Matice  $B_k$  konvergují k matici, jejíž sloupce tvoří vlastní vektory matice  $A$ . Pro symetrickou matici  $A$  je důkaz zřejmý

$$B_{k+1} A_{k+1} \rightarrow A$$

**Poznámka:**

Je-li matice  $A$  symetrická a pozitivně definitní, provádíme LU-rozklad ve smyslu Choleského rozkladu ( $A = LL^T$ ). Potom lze ukázat, že  $A_k$  konverguje k diagonální matici.

**Nevýhody:**

- pomalá konvergence posloupnosti  $A_k$
- velký počet operací pro matice větších řádů
- nelze realizovat pro obecné matice  $A$

**Metody ortogonálních transformací**

Použijeme podobný princip jako v předchozím případě, tj. sestrojíme posloupnost navzájem podobných matic  $A_0, A_1, A_2, \dots$  tak, že

$$A_{k+1} = Q_k^T A_k Q_k, \quad k = 0, 1, 2, \dots$$

Požadujeme, aby posloupnost  $A_k$  konvergovala k matici, jejíž vlastní čísla lehce určíme. Ortogonální matici  $Q_k$  vybíráme speciálním postupem. Výhodou tohoto algoritmu je **numerická stabilita**.

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

**Poznámka:** Pro obecnou matici používáme metodu **QU-rozkladu (QR-transformace)**.

$A = QU$   $Q \dots$  ortogonální matice ( $QQ^T = I$ , tj.  $Q^T = Q^{-1}$ )  
 $U \dots$  horní trojúhelníková matice

$B = UQ$  ( $U = Q^{-1}A \Rightarrow B = Q^{-1}AQ \Rightarrow B = Q^T A Q$ )

**Motivační příklad:**

Příkladem ortogonální matice je matice rovinné rotace o úhel  $\alpha$ :

$$Q(\alpha) = \begin{bmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{bmatrix} \text{ otaž. } \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$$

Pro matici

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$$

stanovte matici  $B = Q^T(\alpha) A Q(\alpha)$  tak, aby  $b_{12} = 0$ .

**Řešení:**

Rozepíšeme si prvky matice  $B$ :

$$\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \cdot \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \cdot \begin{bmatrix} c & -s \\ s & c \end{bmatrix} =$$

$$= \begin{bmatrix} 2c + s & c + 3s \\ -2s + c & -s + 3c \end{bmatrix} \cdot \begin{bmatrix} c & -s \\ s & c \end{bmatrix} =$$

$$= \begin{bmatrix} 2c^2 + cs + cs + 3s^2 & -2cs - s^2 + c^2 + 3cs \\ -2cs + c^2 - s^2 + 3cs & 2s^2 - cs - cs + 3c^2 \end{bmatrix}$$

Pro splnění podmínky  $b_{12} = 0$  musí platit

$$-2cs - s^2 + c^2 + 3cs = cs - s^2 + c^2 = 0,$$

tj.

$$\frac{\cos \alpha \sin \alpha - \sin^2 \alpha + \cos^2 \alpha}{\frac{1}{2} \sin 2\alpha + \cos 2\alpha} = 0$$

$$\cos 2\alpha = -\frac{1}{2} \sin 2\alpha$$

$$-2 = \tan 2\alpha$$

$$\alpha \doteq -0,5535$$

Po dosazení dostaneme, že

$$B = \begin{bmatrix} 3,6180 & 0 \\ 0 & 1,3819 \end{bmatrix}$$

$B$  je diagonální matice s vlastními čísly na diagonále a stejná vlastní čísla má i matice  $A$ . □

**Poznámka:**

Podobně jako v předchozí metodě, pro dostatečně velké  $k$  je  $A_k$  horní trojúhelníková matice a vlastní vektory jsou (přibližně) sloupce matice  $Q_0Q_1 \dots Q_{k-1}$ .

**Poznámka:**

Pro symetrickou matici  $A$  vede uvedený postup na tzv. **metodu Jacobiovy diagonalizace**.

**Jacobiova diagonalizace** (speciální případ QR-transformace)

**Věta:** Je-li  $A$  reálná symetrická matice, potom existuje ortogonální matice  $Q$  tak, že

$$Q^T A Q = \Lambda$$

( $\Lambda$  ... spektrální matice = diagonální matice s vlastními čísly na diagonále).

**Princip:** Matici  $Q$  získáme součinem matic  $Q_{pq}(\alpha)$ , kde

$$Q_{pq}(\alpha) = \begin{bmatrix} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ & & \cos \alpha & & & & & & -\sin \alpha \\ & & \vdots & 1 & & & & & \vdots \\ & & \vdots & & \ddots & & & & \vdots \\ & & \vdots & & & 1 & & & \vdots \\ & & \sin \alpha & & & & & & \cos \alpha \\ & & & & & & & & 1 \\ & & & & & & & & \ddots \\ & & & & & & & & & 1 \end{bmatrix}$$

$\uparrow$   $p$ -tý sloupec                       $\uparrow$   $q$ -tý sloupec

←  $p$ -tý řádek  
 ←  $q$ -tý řádek

a parametr  $\alpha$  volíme tak, abychom vynulovali prvek v pozici  $(p, q)$  a tedy i v pozici  $(q, p)$ .

(D.c.v.  $Q_{pq}^T(\alpha)Q_{pq}(\alpha) = I$ .)

$$B \stackrel{ozna}{=} [A_1 = Q_{pq}^T(\alpha) A Q_{pq}(\alpha)]$$

$$b_{pq} = \begin{bmatrix} \dots & \cos \alpha & \dots & \sin \alpha & \dots \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \end{bmatrix} A \begin{bmatrix} \dots & -\sin \alpha & \dots & \cos \alpha & \dots \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \end{bmatrix}^T$$

$p$ -tý řádek  $Q_{pq}^T$                                        $q$ -tý sloupec  $Q_{pq}$

$$b_{pq} = a_{pq}(\cos^2 \alpha - \sin^2 \alpha) + (a_{qq} - a_{pp}) \cos \alpha \sin \alpha$$

požadujeme, aby  $b_{pq} = 0$ :

$$a_{pq} \cos 2\alpha + (a_{qq} - a_{pp}) \frac{1}{2} \sin 2\alpha = 0 \quad / \cdot 2$$

$$2a_{pq} \cos 2\alpha = -(a_{qq} - a_{pp}) \sin 2\alpha \quad / : \cos 2\alpha \quad / : (a_{qq} - a_{pp})$$

$$\frac{2a_{pq}}{a_{qq} - a_{pp}} = \tan 2\alpha \Rightarrow \alpha = \dots$$

**Poznámka:**

Při výpočtech nemusíme určovat úhel  $\alpha$ , ale stačí nám vyjádřit  $\sin \alpha$  a  $\cos \alpha$ . Lze odvodit vzorce pro

$$\sin \alpha = \dots \qquad \cos \alpha = \dots$$

Celkovou matici získáme takto  $Q = \prod_{p,q} Q_{pq}(\alpha)$  - postupně nulujeme všechny nediagonální prvky.

**Poznámka:**

Zbývá zvolit strategii na volbu indexů  $p$  a  $q$ . Nejjednodušší je postupně nulovat všechny mimodiagonální prvky (podobně jako v Gaussově eliminační metodě pro řešení soustavy lineárních rovnic). Uvědomme si ale, že se získané nuly z předchozího kroku obecně nezachovají. Další možnost je nulovat vždy mimodiagonální prvek, který je největší v absolutní hodnotě (zde je třeba v každé iteraci vyhledat tento prvek, což zpomalí výpočet). Iterační proces zastavíme, je-li norma trojúhelníkové matice pod diagonálou menší než zadaná tolerance.

**1. varianta** - postupné nulování

**2. varianta** - nulování největšího prvku (v abs. hodnotě)

**Vlastní vektory:**

$$A_1 = Q_1^T A Q_1$$

$$A_2 = Q_2^T A_1 Q_2$$

$\vdots$

$$A_k = Q_k^T A_{k-1} Q_k$$

$$\Rightarrow A_k = \underbrace{Q_k^T Q_{k-1}^T \dots Q_1^T}_{P_k^T} A \underbrace{Q_1 Q_2 \dots Q_k}_{P_k}$$

$$P_k A_k = A \underbrace{P_k}_{(**)}$$

(\*)  $A_k \rightarrow \Lambda (k \rightarrow \infty)$       (\*\*)  $P_k \rightarrow X \dots$  jejíž sloupce jsou vlastní vektory matice  $A$

**Givensova transformace**

- Slouží pro převod matice  $A$  na třídíagonální tvar (předpokládáme, že  $A$  je symetrická)
- Opět používáme podobnostní transformace

$$A_{k+1} = Q_k^T A_k Q_k$$

matice  $Q_k$  jsou opět maticemi rovinné rotace, ovšem jsou voleny tak, abychom zachovali již anulované prvky

$$Q_1 = \begin{bmatrix} 1 & & & & & & & \\ & \frac{a_{21}}{d} & & -\frac{a_{31}}{d} & & & & \\ & \frac{a_{31}}{d} & & \frac{a_{21}}{d} & & & & \\ & & & & 1 & & & \\ & & & & & \ddots & & \\ & & & & & & & \\ & & & & & & & 1 \end{bmatrix} \qquad d = \sqrt{a_{21}^2 + a_{31}^2}$$

$A_2 = Q_1^T A_1 Q_1$  bude mít 0 v pozici (3,1) a (1,3)

$$Q_2 = \begin{bmatrix} 1 & & & & & & & \\ & \frac{a_{21}}{d} & & & -\frac{a_{41}}{d} & & & \\ & & 1 & & & & & \\ & \frac{a_{41}}{d} & & & \frac{a_{21}}{d} & & & \\ & & & & & 1 & & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{bmatrix} \qquad d = \sqrt{a_{21}^2 + a_{41}^2}$$

$A_3 = Q_2^T A_2 Q_2$  bude mít 0 v pozici (4,1) a (1,4)

atd.

**Příklad:** Převeďte matici  $A$  na třídíagonální tvar.

$$A = \begin{bmatrix} 1 & 4 & 3 \\ 4 & 2 & 6 \\ 3 & 6 & 5 \end{bmatrix}$$

$$a_{21} = 4, \quad a_{31} = 3, \quad d = \sqrt{4^2 + 3^2} = 5$$

$$Q_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{4}{5} & -\frac{3}{5} \\ 0 & \frac{3}{5} & \frac{4}{5} \end{bmatrix}$$

**$Q_1^T Q_1 = I$  ?**

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{4}{5} & -\frac{3}{5} \\ 0 & \frac{3}{5} & \frac{4}{5} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{4}{5} & \frac{3}{5} \\ 0 & -\frac{3}{5} & \frac{4}{5} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Q_1^T A Q_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{4}{5} & -\frac{3}{5} \\ 0 & \frac{3}{5} & \frac{4}{5} \end{bmatrix} \begin{bmatrix} 1 & 4 & 3 \\ 4 & 2 & 6 \\ 3 & 6 & 5 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{4}{5} & \frac{3}{5} \\ 0 & -\frac{3}{5} & \frac{4}{5} \end{bmatrix} =$$

$$= \begin{bmatrix} 1 & 4 & 3 \\ 5 & \frac{26}{5} & \frac{39}{5} \\ 0 & \frac{18}{5} & \frac{2}{5} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{4}{5} & \frac{3}{5} \\ 0 & -\frac{3}{5} & \frac{4}{5} \end{bmatrix} = \begin{bmatrix} 1 & 5 & 0 \\ 5 & \frac{221}{25} & \frac{78}{25} \\ 0 & \frac{78}{25} & -\frac{46}{25} \end{bmatrix}$$

**Efektivní výpočet hodnoty charakteristického polynomu pro třídíagonální matici**

$$A = \begin{bmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & b_3 & a_3 & c_3 & \\ & & \ddots & \ddots & \ddots \\ & & & b_n & a_n \end{bmatrix}$$

$$f_{-1}(\lambda) = 0$$

$$f_0(\lambda) = 1$$

$$f_k(\lambda) = (a_k - \lambda) f_{k-1}(\lambda) - b_k c_{k-1} f_{k-2}(\lambda) \quad k = 1, 2, \dots, n$$

$$f_n(\lambda) = p_A(\lambda)$$



Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

$$A - \lambda I = \begin{pmatrix} a_1 - \lambda & c_1 & & & \\ b_2 & a_2 - \lambda & c_2 & & \\ & b_3 & a_3 - \lambda & c_3 & \\ & & & \ddots & \\ & & & & b_{n-1} & a_{n-1} - \lambda & c_{n-1} \\ & & & & & b_n & a_n - \lambda \end{pmatrix}$$

rozvoj podle posledního řádku:

$$\det M = (a_n - \lambda) \det(M_{n-1}) - b_n c_{n-1} \det(M_{n-2})$$

( $M_{n-1}$  ... prvních  $n-1$  řádků a sloupců z  $M$ )

$$M_1 = a_1 - \lambda = \det(M_1)$$

$$M_0 = 1$$

$$M_{-1} = 0$$

Podstata výpočtu vlastních čísel třídiagonální matice pomocí jednoduchého vyjadřování hodnoty charakteristického polynomu metodou bisekce:

**Příklad:**

$$A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}$$

Zvolíme interval  $(0, 5)$  ... v něm očekáváme všechna vlastní čísla

Vypočtem snadno určité

$$f_3(0) = p_A(0) = 4$$

$$f_{-1}(0) = 0$$

$$f_0(0) = 1$$

$$f_1(0) = \underbrace{(a_1 - 0)}_{=2} \underbrace{f_0(0)}_{=1} - b_1 c_0 \underbrace{f_{-1}(0)}_{=0} = 2$$

$$f_2(0) = \underbrace{(a_2 - 0)}_{=2} \underbrace{f_1(0)}_{=2} - \underbrace{b_2}_{=-1} \underbrace{c_1}_{=-1} \underbrace{f_0(0)}_{=1} = 3$$

$$f_3(0) = \underbrace{(a_3 - 0)}_{=2} \underbrace{f_2(0)}_{=3} - \underbrace{b_3}_{=-1} \underbrace{c_2}_{=-1} \underbrace{f_1(0)}_{=2} = 4$$

$$f_3(5) = p_A(5) = -21$$

$$f_{-1}(5) = 0$$

$$f_0(5) = 1$$

$$f_1(5) = \underbrace{(a_1 - 5)}_{=2} \underbrace{f_0(5)}_{=1} - b_1 c_0 \underbrace{f_{-1}(5)}_{=0} = -3$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

$$f_2(5) = \underbrace{(a_2 - 5)}_{=2} \underbrace{f_1(5)}_{=-3} - \underbrace{b_2}_{=-1} \underbrace{c_1}_{=-1} \underbrace{f_0(5)}_{=1} = 8$$

$$f_3(5) = \underbrace{(a_3 - 5)}_{=2} \underbrace{f_2(5)}_{=8} - \underbrace{b_3}_{=-1} \underbrace{c_2}_{=-1} \underbrace{f_1(5)}_{=-3} = -21$$

Vypočteme střed intervalu,  $s = \frac{0+5}{2} = 2,5$  a určíme  $f_3(2,5)$  ...

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

## Kapitola 7. Aproximace funkcí - I

**Motivace**

Při numerickém řešení úloh matematické analýzy často nahrazujeme danou funkci  $f$ , vystupující v řešené matematické úloze, jinou funkcí  $\varphi$ , která v nějakém vhodném smyslu napodobuje funkci  $f$  a snadno se přitom matematicky zpracovává či modeluje na počítači. Tuto funkci  $\varphi$  nazýváme **aproximací funkce**  $f$ .

**Poznámka:** Aproximací funkce jsme již používali u řešení nelineární rovnice. Například u Newtonovy metody jsme danou funkci  $f$  z řešené rovnice  $f(x) = 0$  aproximovali lineární funkcí (tečnou ke grafu funkce  $f$ ); podobně tak tomu bylo u metody sečen.

**Poznámka:** Již pouhý výpočet funkčních hodnot některých základních funkcí ( $\sin x$ ,  $e^x$ ,  $\ln x$ , ...) v počítači či na kalkulačce se provádí užitím aproximace těchto funkcí. Tyto aproximace jsou ovšem zabudovány do výpočetního systému a uživatel si často ani neuvědomuje, že přišli-li v programu např.  $y = \sin(x)$ , nahrazuje výpočet hodnoty funkce  $\sin x$  výpočtem hodnoty jistého polynomu.

**Příklady užití:**

- numerické metody pro výpočet určitého integrálu
- zpracování výsledků měření

**Formulace** (základní aproximační úloha)

Je dána funkce  $f = f(x)$ ,  $x \in (a, b)$ . Zvolíme  $(n+1)$  lineárně nezávislých funkcí  $\varphi_0, \varphi_1, \dots, \varphi_n$  a hledáme funkci  $\varphi$  definovanou na  $(a, b)$ , kterou lze vyjádřit ve tvaru lineární kombinace

$$\varphi(x) = c_0 \varphi_0(x) + c_1 \varphi_1(x) + \dots + c_n \varphi_n(x)$$

a která je v nějakém smyslu blízká funkci  $f$ .

- Tento typ aproximace se nazývá **lineární aproximace**
- Pokud za funkce  $\varphi_i(x)$  volíme polynomy, mluvíme o **polynomiální aproximaci**
- Příkladem **nelineární aproximace** je funkce  $\varphi(x)$ :

$$\varphi(x) = \frac{a_0 + a_1 x + \dots + a_n x^n}{1 + b_1 x + \dots + b_m x^m}$$

V tomto případě mluvíme o **racionální aproximaci**

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

měření), kde nutně nevyžadujeme, aby aproximace danými body procházela. Důvodem můžou být např. chyby, se kterými jsme hodnoty naměřili.

**Aproximace na okolí bodu**

– mluvíme o **aproximaci Taylorovým polynomem**

Předpokládáme, že daná funkce  $f$  má v daném bodě  $x_0$  a jeho okolí spojité derivace až do řádu  $n$ . Podmínky pro funkci, která co nejlépe napodobuje chování funkce  $f$  matematicky запиšeme takto:

$$\varphi^{(j)}(x_0) = f^{(j)}(x_0), \quad j = 0, 1, \dots, n$$

(Hodnoty derivací funkcí  $f$ ,  $\varphi$  v bodě  $x_0$  jsou stejné až do řádu  $n$ .)

Tuto podmínku samozřejmě splňuje Taylorův polynom

$$T_n(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \dots + \frac{f^{(n)}(x_0)}{n!}(x-x_0)^n$$

Pro chybu aproximace Taylorovým polynomem platí

$$e(x) = f(x) - T_n(x) = f^{(n+1)}(\xi) \frac{(x-x_0)^{n+1}}{(n+1)!}, \quad \xi \in U(x_0)$$

umíme-li odhadnout  $(n+1)$ -ní derivaci funkce  $f$  na daném okolí bodu  $x_0$ , můžeme provést následující odhad chyby aproximace:

$$\text{Platí-li } |f^{(n+1)}(x)| \leq M \quad \forall x \in U(x_0), \quad \text{potom } |e(x)| \leq \frac{M}{(n+1)!} |x-x_0|^{n+1}.$$

**Příklad** Stanovte odhad chyby aproximace Taylorovým polynomem 10. stupně funkce  $f(x) = e^x$  v bodě  $x_0 = 0$  na intervalu  $(-1, 1)$ .

$$f^{(j)}(x) = e^x, \quad j = 0, 1, \dots; \quad f^{(j)}(0) = 1, \quad j = 0, 1, \dots$$

$$T_n(x) = 1 + x + \frac{x^2}{2} + \dots + \frac{x^{10}}{10!}$$

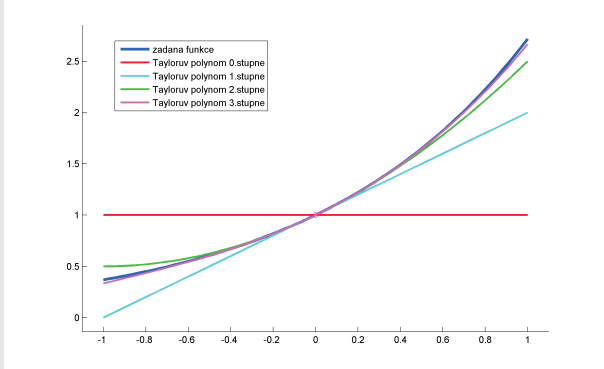
$$e(x) = e^x - T^{10}(x) = \frac{e^\xi}{11!} x^{11} \Rightarrow |e(x)| \leq \frac{e}{11!} \leq 7 \cdot 10^{-8}$$

### Přístupy k aproximaci

**Aproximace na okolí bodu** - Použijeme, chceme-li aproximovat chování funkce v malém okolí bodu. Příkladem může být např. vyčíslení hodnoty  $\sin \frac{\pi}{4}$  na kalkulačce.

**Interpolace** - Použijeme, chceme-li tabulkou danými body proložit polynom, tj. požadujeme-li, aby aproximace přesně procházela zadanými body.

**L<sub>2</sub>-aproximace** - Použijeme, hledáme-li funkční závislost mezi tabulkou danými body (získaných například



**Příklad** Určete stupeň Taylorova polynomu funkce  $f(x) = \sin x$  v bodě  $x_0 = 0$  tak, aby jeho chyba na intervalu  $(-\pi, \pi)$  byla nejvýše  $10^{-5}$ , resp.  $(10^{-12})$ .

Pro chybu platí

$$e(x) = f(x) - T_n(x) = \frac{(x-x_0)^{n+1}}{(n+1)!} f^{(n+1)}(\xi)$$

Zajímá nás chyba na intervalu délky  $2\pi \Rightarrow$  odhad pro  $f^{(n+1)}(x)$  je  $\left| \frac{f^{(n+1)}(x)}{(n+1)!} \right| \leq 1 \quad \forall x \in (-\pi, \pi)$

(\*) buď  $\pm \sin x$  nebo  $\pm \cos x$  - tento odhad je vždy nejménší možný

$$\left| \frac{x^{n+1}}{(n+1)!} \right| \leq 10^{-5} \quad \forall x \in (-\pi, \pi)$$

$$\frac{\pi^{n+1}}{(n+1)!} \leq 10^{-5} \quad (\text{resp. } 10^{-12})$$

$n$	$\frac{\pi^{n+1}}{(n+1)!}$
0	3.141592653589793
1	4.934802200544679
2	5.167712780049969
3	4.058712126416768
4	2.550164039877345
5	1.335262768854589
6	0.599264529320792
7	0.235330630358893
8	0.08214588661128
9	0.025806891390014
10	0.007370430945714
11	0.001929574309404
12	0.000466302805768
13	0.000104638104925
14	0.000021915353448
15	<b>0.000004303069587</b> $< 10^{-5}$
16	0.000000795205400
17	0.000000138789525
18	0.000000022948429
19	0.000000003604731
20	0.000000000539266
21	0.000000000077007
22	0.000000000010518
23	0.000000000001377
24	<b>0.000000000000173</b> $< 10^{-12}$

Jak vypadá Taylorův polynom  $T_n(x)$ ?

$$\begin{aligned} f(x) &= \sin x & f(0) &= 0 \\ f'(x) &= \cos x & f'(0) &= 1 \\ f''(x) &= -\sin x & f''(0) &= 0 \\ f'''(x) &= -\cos x & f'''(0) &= -1 \\ f^{(4)}(x) &= \sin x & f^{(4)}(0) &= 0 \\ & \vdots & & \vdots \end{aligned}$$

$$T_n(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2!} f''(x_0)(x-x_0)^2 + \dots + \frac{1}{n!} f^{(n)}(x_0)(x-x_0)^n$$

$$T_{15}(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} + \frac{x^{13}}{13!} - \frac{x^{15}}{15!}$$

Má-li kalkulačka vypočítat např.  $\sin 0,4$  a má povolenou chybu  $10^{-5}$ , určí  $\sin 0,4 \doteq T_{15}(0,4)$ .

Pro dodržení chyby  $10^{-12}$  stačí určit  $\sin 0,4 \doteq T_{24}(0,4)$ , tj. přičíst další 4 členy.

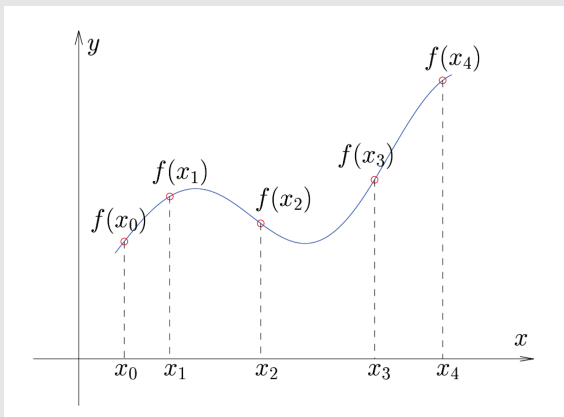
**Poznámka:**

Stejný postup lze užít i pro výpočet v bodech mimo interval  $(-\pi, \pi)$ . Stačí využít periodičnosti funkce  $\sin x$ .

**Aproximace interpolačním polynomem**

Aproximujeme funkci, která je dána svými hodnotami v  $n+1$  bodech  $x_i, i = 0, 1, \dots, n$  (body  $x_i$  nazýváme uzly interpolace), a požadujeme, aby aproximace (interpolačním polynomem) procházela zadanými body.

Aproximace nám potom poslouží k získání přibližné hodnoty zadané funkce v libovolném bodě intervalu  $(x_0, x_n)$ .



Interpolační podmínky

$$P_n(x_i) = f(x_i), \quad i = 0, 1, \dots, n$$

Chyba  $e(x) = f(x) - P_n(x)$  pak nabývá nulových hodnot v uzlech interpolace.

**Věta** Interpolační úloha má jediné řešení, pokud jsou uzly  $x_0, x_1, \dots, x_n$  navzájem různé.

**Důkaz:** Interpolační polynom uvažujeme ve tvaru

$$P_n(x) = \sum_{k=0}^n a_k x^k$$

Dosažením do interpolačních podmínek získáme soustavu  $(n+1)$  lineárních rovnic pro koeficienty  $a_0, a_1, \dots, a_n$ .

$$\sum_{k=0}^n a_k x_i^k = f(x_i) \quad i = 0, 1, \dots, n$$

Maticové soustavy (Vandermondova matice):

$$V = \begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix}$$

Soustava má právě jedno řešení, pokud  $\det V \neq 0$ .

Maticové soustavy převedeme na trojúhelníkový tvar

- přičtením násobku řádku k jinému řádku se determinant nezmění
- vynásobíme-li řádek číslem  $\alpha$ , pak je determinant  $\alpha$  násobkem původního

Od 2. až  $(n+1)$ -ního řádku odečteme 1. řádek.

$$V \sim \begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 & \dots & x_0^n \\ 0 & x_1 - x_0 & x_1^2 - x_0^2 & x_1^3 - x_0^3 & \dots & x_1^n - x_0^n \\ 0 & x_2 - x_0 & x_2^2 - x_0^2 & x_2^3 - x_0^3 & \dots & x_2^n - x_0^n \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & x_n - x_0 & x_n^2 - x_0^2 & x_n^3 - x_0^3 & \dots & x_n^n - x_0^n \end{bmatrix}$$

Vynormujeme  $-(j+1)$ -ní řádek vydělíme  $(x_j - x_0), j = 1, 2, \dots, n$ .

$$V \sim \begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 & \dots & x_0^n \\ 0 & 1 & \frac{x_1^2 - x_0^2}{x_1 - x_0} & \frac{x_1^3 - x_0^3}{x_1 - x_0} & \dots & \frac{x_1^n - x_0^n}{x_1 - x_0} \\ 0 & 1 & \frac{x_2^2 - x_0^2}{x_2 - x_0} & \frac{x_2^3 - x_0^3}{x_2 - x_0} & \dots & \frac{x_2^n - x_0^n}{x_2 - x_0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & \frac{x_n^2 - x_0^2}{x_n - x_0} & \frac{x_n^3 - x_0^3}{x_n - x_0} & \dots & \frac{x_n^n - x_0^n}{x_n - x_0} \end{bmatrix}$$

od 3. až  $(n+1)$ -ního řádku odečteme 2. řádek

$$V \sim \begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 & \dots & x_0^n \\ 0 & 1 & x_1 + x_0 & x_1^2 + x_1 x_0 + x_0^2 & \dots & \dots \\ 0 & 0 & x_2 - x_1 & (*) & \dots & \dots \\ 0 & 0 & x_3 - x_1 & (**) & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & x_n - x_1 & \dots & \dots & \dots \end{bmatrix}$$

$$(*) = \frac{x_2^3 - x_0^3}{x_2 - x_0} - \frac{x_1^3 - x_0^3}{x_1 - x_0} = (x_2^2 + x_2x_0 + x_0^2) - (x_1^2 + x_1x_0 + x_0^2) = x_2^2 - x_1^2 + x_0(x_2 - x_1) = (x_2 - x_1)(x_2 + x_1 + x_0)$$

$$(**) = \frac{x_3^3 - x_0^3}{x_3 - x_0} - \frac{x_1^3 - x_0^3}{x_1 - x_0} = (x_3^2 + x_3x_0 + x_0^2) - (x_1^2 + x_1x_0 + x_0^2) = x_3^2 - x_1^2 + x_0(x_3 - x_1) = (x_3 - x_1)(x_3 + x_1 + x_0)$$

Vynormujeme -  $(j+1)$ -ní řádek vydělíme  $(x_j - x_1)$ ,  $j = 2, 3, \dots, n$ .

$$V \sim \begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 & \dots & x_0^n \\ 0 & 1 & x_1 + x_0 & x_1^2 + x_1x_0 + x_0^2 & \dots & \dots \\ 0 & 0 & 1 & x_2 + x_1 + x_0 & \dots & \dots \\ 0 & 0 & 1 & x_3 + x_1 + x_0 & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 1 & x_n + x_1 + x_0 & \dots & \dots \end{bmatrix}$$

Získáme trojúhelníkovou matici s jedničkami na diagonále, tj. výsledná matice má determinant roven 1.

Při úpravách jsme dělili  $(x_j - x_1)$ ,  $j > 1$

$$\Rightarrow \det V = \prod_{j>1}^n (x_j - x_1)$$

$$\det V \neq 0 \Leftrightarrow x_i \neq x_j \quad \forall i \neq j$$

### Lagrangeův interpolační polynom

Označme si hledaný polynom  $n$ -tého stupně  $L_n(x)$ .

Víme, že musí být splněny interpolační podmínky

$$L_n(x_i) = f(x_i), \quad i = 0, 1, \dots, n.$$

Lagrangeův interpolační polynom hledáme ve tvaru

$$L_n(x) = \sum_{i=0}^n f(x_i) \cdot l_i(x)$$

kde  $l_i(x)$  jsou polynomy  $n$ -tého stupně takové, že platí

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

(snadno se přesvědčíte, že dosadíte-li do předpisu pro  $L_n(x)$  uzly interpolace, získáte zadané interpolační podmínky).

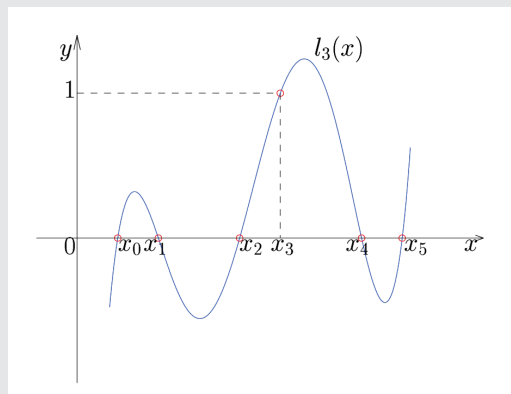
Konkretizujme nyní dílčí polynomy  $l_i(x)$ .

Víme, že  $l_i(x)$  má kořeny  $x_0, x_1, x_{i-1}, x_{i+1}, \dots, x_n$  a nabývá hodnoty 1 v bodě  $x_i$ .

Můžeme jej tedy zapsat ve tvaru

$$l_i(x) = \frac{(x-x_0)(x-x_1)\dots(x-x_{i-1})(x-x_{i+1})\dots(x-x_n)}{(x_i-x_0)(x_i-x_1)\dots(x_i-x_{i-1})(x_i-x_{i+1})\dots(x_i-x_n)}$$

Na obrázku je ukázan příklad dílčího polynomu  $l_3(x)$ .



### Newtonův interpolační polynom

Označme si hledaný polynom  $n$ -tého stupně  $N_n(x)$ .

Pro jeho odvození použijeme jinou konstrukci. Polynom volíme ve tvaru

$$N_n(x) = a_0 + a_1(x-x_0) + a_2(x-x_0)(x-x_1) + \dots + a_n(x-x_0)(x-x_1)\dots(x-x_{n-1})$$

Opět požadujeme splnění interpolačních podmínek

$$N_n(x_i) = f(x_i), \quad i = 0, 1, \dots, n.$$

**Poznámka:** Výhodou volby tohoto zdánlivě složitějšího předpisu je fakt, že přidáme-li další bod interpolace  $[x_{n+1}, f(x_{n+1})]$ , nemusíme celý výpočet opakovat, ale stačí dopočítat příslušný koeficient  $a_{n+1}$  (ostatní koeficienty  $a_i$  zůstávají beze změny). Při použití Lagrangeova polynomu bychom museli celý výpočet provést znovu.

Ukažme si, co dostaneme dosazováním interpolačních podmínek do předpisu polynomu:

$$N_n(x_0) = a_0 = f(x_0)$$

$$N_n(x_1) = a_0 + a_1(x_1 - x_0) = f(x_1) \Rightarrow a_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

$$N_n(x_2) = a_0 + a_1(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) = f(x_2)$$

$$\Rightarrow a_2 = \frac{f(x_2) - f(x_0) - a_1(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)} = \frac{f(x_2) - f(x_0) - \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)} = \frac{f(x_2) - f(x_0) - \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)}$$

$$\Rightarrow a_2 = \frac{f(x_2) - f(x_1) - \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x_2 - x_1)}{x_2 - x_0}$$

**Poznámka:**

Počítat koeficienty  $a_i$  přímo ze soustavy není praktické, budeme je počítat pomocí tzv. **poměrných diferencí**.

**Algoritmus** (koeficienty Newtonova polynomu)

Pro  $i = 0, 1, \dots, n$

$$A_{i0} = f(x_i)$$

Pro  $k = 1, 2, \dots, i$

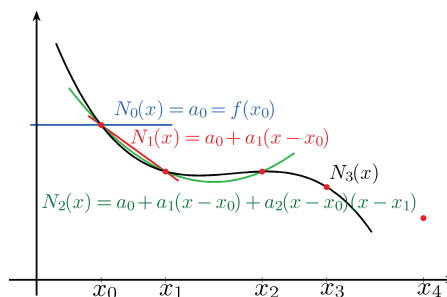
$$A_{ik} = \frac{A_{i,k-1} - A_{i-1,k-1}}{x_i - x_{i-k}}$$

Výstup:  $a_i = A_{ii}$

**Schéma algoritmu**

$x_0$	$f(x_0) = A_{00} = a_0$		
$x_1$	$f(x_1) = A_{10}$	$A_{11} = a_1$	
$x_2$	$f(x_2) = A_{20}$	$A_{21}$	$A_{22} = a_2$
$\vdots$	$\vdots$	$\vdots$	$\ddots$
$x_n$	$f(x_n) = A_{n0}$	$A_{n1}$	$A_{n2} \dots A_{nn} = a_n$

**Poznámka:** Vezmu-li z matice  $A$  pouze prvních  $M$  řádků znamená to, že jsem sestavil Newtonův polynom pro pouze prvních  $M$  zadaných tabulkových bodů.



$N_0(x) = a_0$  prochází bodem  $[x_0, f(x_0)]$

$N_1(x) = a_0 + a_1(x - x_0)$  prochází bodem  $[x_0, f(x_0)]$  a směrnici má takovou, aby procházel také bodem  $[x_1, f(x_1)]$

$N_2(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1)$  prochází bodem  $[x_0, f(x_0)]$ , dále jelikož platí  $N_2(x_1) = N_1(x_1)$ , prochází i bodem  $[x_1, f(x_1)]$  a navíc

prochází bodem  $[x_2, f(x_2)]$ .

$$N_2(x) = \underbrace{a_0}_{(*)} + \underbrace{a_1(x-x_0)}_{(**)} + \underbrace{a_2(x-x_0)(x-x_1)}_{(***)}$$

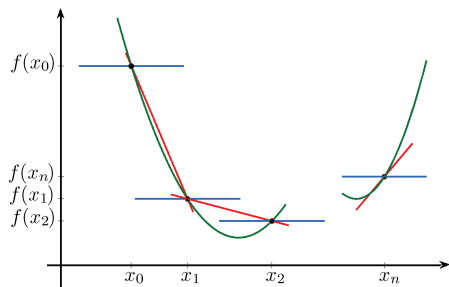
(\*) prochází  $[x_0, f(x_0)]$

(\*\*) přidáme tento člen tak, aby se zachoval průchod  $[x_0, f(x_0)]$  (hodnota pro  $x = x_0$  je 0),  $a_1$  se určí tak, aby procházel  $[x_1, f(x_1)]$

(\*\*\*) přidáme tento člen tak, aby se zachoval průchod  $[x_0, f(x_0)]$  a  $[x_1, f(x_1)]$  (hodnota členu pro  $x = x_0$  a  $x = x_1$  je nulová),  $a_2$  se určí tak, aby procházel bodem  $[x_2, f(x_2)]$

Co znamenají jednotlivá čísla v tabulce?

$x_0$	$f(x_0) = A_{00}$		
$x_1$	$f(x_1) = A_{10}$	$A_{11}$	
$x_2$	$f(x_2) = A_{20}$	$A_{21}$	$A_{22}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x_n$	$f(x_n) = A_{n0}$	$A_{n1}$	$A_{n2}$



**Věta** Má-li funkce  $f$ , k níž přísluší data  $f(x_i)$ ,  $i = 0, 1, \dots, n$ , spojité derivace v nějakém intervalu  $(a, b) \supset \text{int}(x_0, x_1, \dots, x_n)$  do řádu  $n$  a derivaci  $f^{(n+1)}(x)$  v  $(a, b)$ , potom  $\forall x \in (a, b) \exists \xi = \xi(x) \in (a, b)$  tak, že pro chybu interpolačního polynomu  $P_n(x)$  platí:

$$e(x) = f(x) - P_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \underbrace{(x-x_0)(x-x_1)\dots(x-x_n)}_{\text{polynom } (n+1) \text{ stupně}} \quad \textcircled{B}$$

(Důkaz pomocí věty o střední hodnotě, viz dále.)

**Odhad chyby interpolace**

Umíme-li stanovit číslo  $M$  takové, že  $\forall x \in (a, b) |f^{(n+1)}(x)| \leq M$ , pak

$$|e(x)| \leq \frac{M}{(n+1)!} |w(x)| \leq \frac{M}{(n+1)!} \max_{x \in (a,b)} |w(x)|$$

kde jsme označili  $w(x) = (x-x_0)(x-x_1)\dots(x-x_n)$ .

$\Rightarrow$  Průběh chyby nezávisí jen na  $f^{(n+1)}(x)$ , ale i na  $w(x)$ !

**Důkaz:** Zvolme bod  $x \neq x_i$ ,  $i = 0, 1, \dots, n$  libovolně. Definujme funkci

$$F(t) = f(t) - P_n(t) - \frac{f(x) - P_n(x)}{w_{n+1}(x)} w_{n+1}(t) \quad \textcircled{B}$$

Tato funkce proměnné  $t$  má  $n+2$  nulových bodů:

- $x_i$ ,  $i = 0, 1, \dots, n$ :

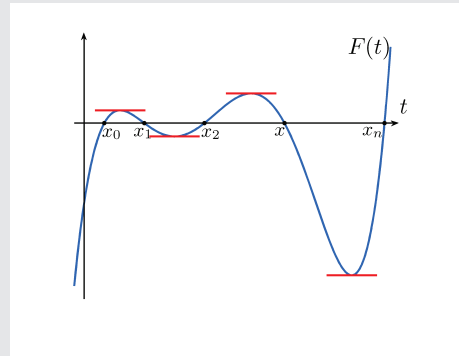
$$F(x_i) = \underbrace{f(x_i) - P_n(x_i)}_{=0} - \frac{f(x) - P_n(x)}{w_{n+1}(x)} \underbrace{w_{n+1}(x_i)}_{=0} = 0$$

- $x$ :

$$F(x) = f(x) - P_n(x) - \frac{f(x) - P_n(x)}{w_{n+1}(x)} w_{n+1}(x) = 0$$

Použijeme-li  $(n+1)$ krát **Rollovu větu** zjistíme, že první derivace  $F'(t)$  má v  $(a, b)$  alespoň  $n+1$  nulových bodů.

**Rollova věta:** Nechť  $f(x)$  je v  $(a, b)$  spojitá a má v  $(a, b)$  derivaci. Nechť  $f(a) = f(b)$ . Pak existuje alespoň 1 bod  $c \in (a, b)$  tak, že  $f'(c) = 0$ .



Pokračujeme dál a aplikujeme nyní **Rollovu větu** na funkci  $F'(t)$  a zjistíme, že  $F''(t)$  má v  $(a, b)$  alespoň  $n$  nulových bodů, atd.

$\vdots$

$F^{(n+1)}(t)$  má v  $(a, b)$  alespoň jeden nulový bod a ten označíme  $\xi = \xi(x)$ .

Vztah  $\textcircled{B}$  zderivujeme  $(n+1)$  krát podle  $t$ :

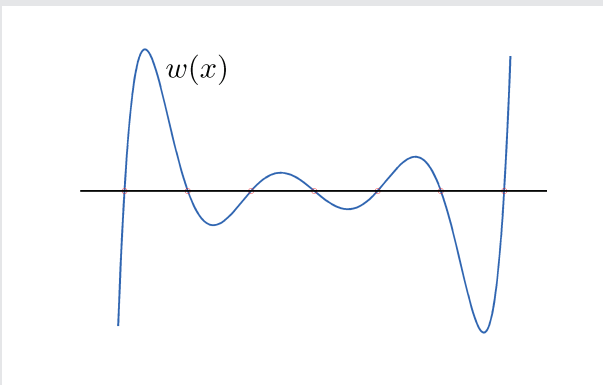
$$\underbrace{F^{(n+1)}(t)}_{\textcircled{B}} = \underbrace{f^{(n+1)}(t)}_{\textcircled{A}} - \underbrace{P_n^{(n+1)}(t)}_{=0 \textcircled{A}} - \frac{f(x) - P_n(x)}{w_{n+1}(x)} \underbrace{w_{n+1}^{(n+1)}(t)}_{=(n+1)! \textcircled{A}}$$

- $\textcircled{A}$  Víme, že existuje  $\xi$  tak, že  $F^{(n+1)}(\xi) = 0$
- $\textcircled{B}$   $P_n \dots$  polynom  $n$ -tého stupně
- $\textcircled{C}$   $w_{n+1}(t)$  je polynom  $(n+1)$  stupně a u  $t^{n+1}$  je koeficient 1

$$0 = f^{(n+1)}(\xi) - \frac{f(x) - P_n(x)}{w_{n+1}(x)} (n+1)! \Rightarrow f(x) - P_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} w_{n+1}(x) \quad \square$$

- Pokud interpolujeme funkci zadanou v **ekvidistančních uzlech**, mohou mít nepřesnosti ve vstupních datech silný vliv na hodnotu výsledku  $\rightarrow$  úloha je špatně podmíněná

**Poznámka:** Špatná podmíněnost platí tím více i pro extrapolaci.



- Dobrou strategií je volit  $x_i$  tak, aby byly rozloženy stejně jako kořeny **Čebyševových polynomů**  $\rightarrow$  minimalizuje se tak hodnota  $\max |w(x)|$ .

**Definice (Čebyševova aproximace)**

K dané spojitě funkci  $f(x)$ ,  $x \in (a, b)$ , chceme najít mezi všemi polynomy  $P_n(x)$  stupně nejvýše  $n$  takový polynom  $P_n^*(x)$ , který splňuje:

$$\max_{x \in (a,b)} |f(x) - P_n^*(x)| = \min_{P_n(x)} \max_{x \in (a,b)} |f(x) - P_n(x)|$$

**Poznámka:**

Při volbě ekvidistančních uzlů byla úloha špatně podmíněná.

Při vhodné volbě uzlů (kořeny tzv. **Čebyševových polynomů** - viz dále  $\heartsuit$ ) má interpolační proces pro  $n \rightarrow \infty$  tu vlastnost, že interpolační polynomy **konvergují** na  $(a, b)$  **stejněměrně** k aproximované funkci např. v případě, když existuje spojitá první derivace  $f'$  na  $(a, b)$ .

**Stejněměrná konvergence** funkce  $f_n$  definované na  $(a, b)$

- varianta 1:** posloupnost  $\{f_n\}$  je na  $(a, b)$  stejnoměrně konvergentní

$$\Leftrightarrow \forall \epsilon > 0 \exists n_0 \in \mathbb{N}: \text{stejně } \forall x \in (a, b) \text{ tak, že}$$

$$\forall n > n_0 \text{ a } \forall x \in (a, b) : |f_n(x) - f(x)| < \epsilon$$

- varianta 2:** posloupnost  $\{f_n\}$  je na  $(a, b)$  stejnoměrně konvergentní

$$\Leftrightarrow \lim_{n \rightarrow \infty} \sup_{x \in (a,b)} |f_n(x) - f(x)| = 0$$

**Čebyševovy polynomy**

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1} &= 2x T_n(x) - T_{n-1}(x) \quad \heartsuit \end{aligned}$$

Užijeme-li substituci  $x = \cos \alpha$ ,  $\alpha = \arccos x$  a goniometrické vzorce, dostaneme

$$T_n(x) = \cos(n \arccos x) \quad x \in (-1, 1) \quad \clubsuit$$

**Důkaz:**

$$T_0(x) = \cos(0 \arccos x) = 1 \quad \checkmark$$

$$T_1(x) = \cos(1 \arccos x) = x \quad \checkmark$$

$$T_n = \cos(n \arccos x) \quad \dots \text{ dosadíme do vztahu } \heartsuit$$

$$\cos \left( \frac{(n+1) \arccos x}{\alpha} \right) = 2x \cos \left( \frac{n \arccos x}{\alpha} \right) - \cos \left( \frac{(n-1) \arccos x}{\beta} \right)$$

Platí:

$$\cos \alpha + \cos \beta = 2 \cos \frac{\alpha + \beta}{2} \cos \frac{\alpha - \beta}{2} \quad \checkmark$$

Kořeny polynomů jsou

$$\begin{aligned} \cos(n \arccos x) &= 0 \\ n \arccos x_k &= \frac{\pi}{2} + k\pi, \quad k \in \mathbb{Z} \end{aligned}$$

$$\arccos x_k = \frac{2k+1}{n} \cdot \frac{\pi}{2}$$

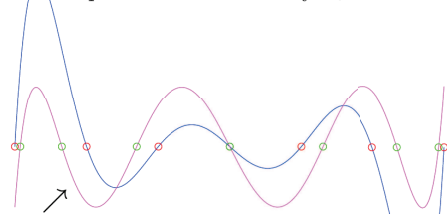
$$x_k = \cos\left(\frac{2k+1}{n} \cdot \frac{\pi}{2}\right), \quad k = 0, 1, \dots, n-1$$

Poznámka: Pro obecný interval  $(a, b)$  použijeme transformaci

$$x_k = \frac{b-a}{2} x_k + \frac{a+b}{2}$$

$$w(x) = (x - x_0)(x - x_1) \dots (x - x_N)$$

← pro ekvidistanční uzly  $x_i$



pro neekvidistanční uzly  $x_i$  (kořeny Čebyševových polynomů)

**Příklad**

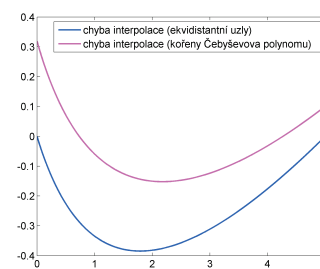
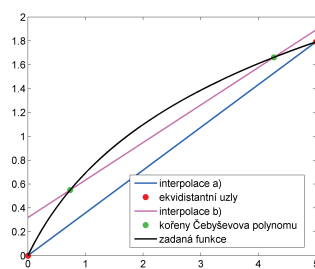
Uvažujme funkci  $f(x) = \ln(x+1)$  zadanou na intervalu  $(0, 5)$ .

Pro zvolený počet  $n$  uzlových bodů proveďte interpolaci zadané funkce pro uzlové body, které jsou

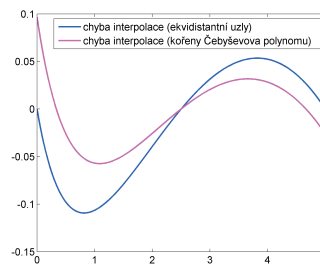
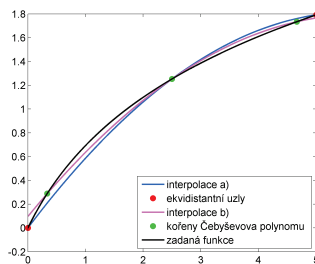
- a) ekvidistanční,
- b) kořeny Čebyševových polynomů.

Porovnejte chybu získaných interpolačních polynomů.

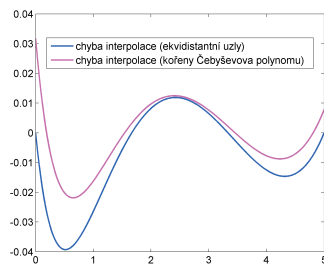
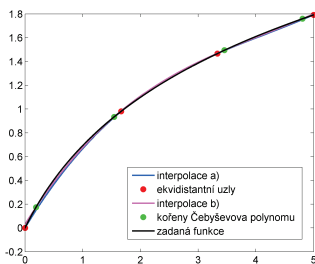
1)  $n = 2$



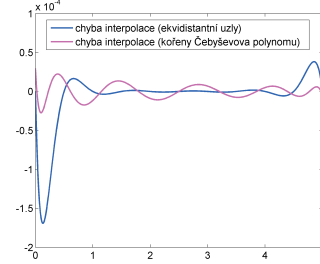
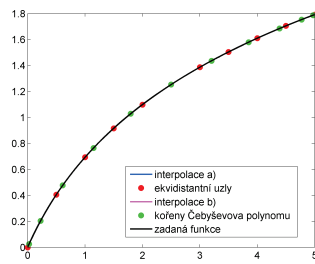
2)  $n = 3$



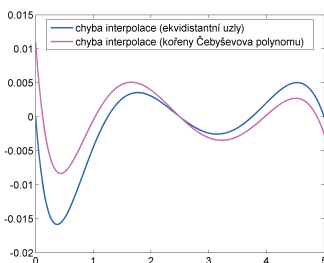
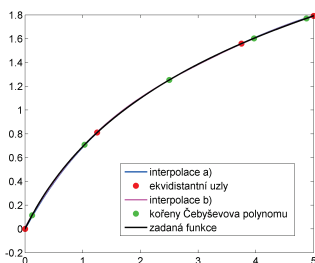
3)  $n = 4$



4)  $n = 5$

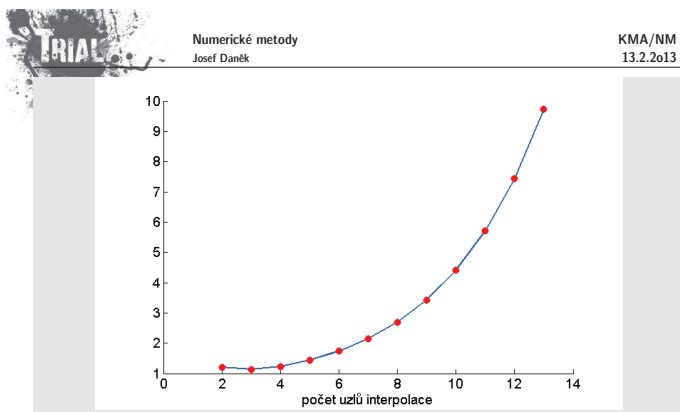


6)  $n = 13$



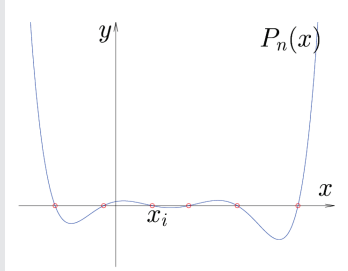
5)  $n = 11$

Následující obrázek ukazuje poměr maximové normy chyby interpolačního polynomu vypočteného pro hodnoty v ekvidistančních uzlech ku maximové normě chyby interpolačního polynomu vypočteného pro hodnoty v uzlech určených jako kořeny Čebyševových polynomů.



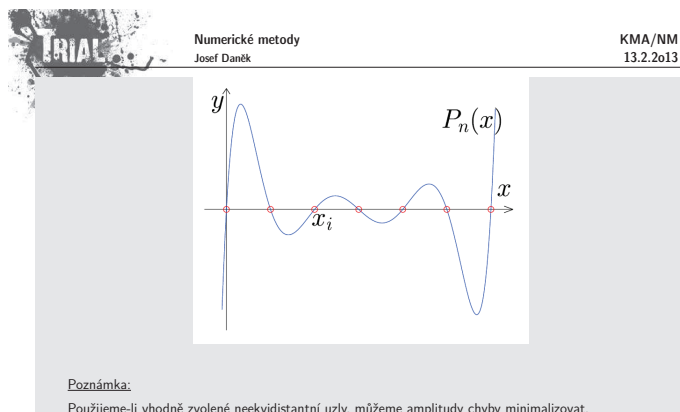
**Poznámka:**

Interpoláční polynomy vyšších řádů není vhodné užívat pro aproximaci hodnot funkce mimo interval obsahující uzly interpolace (tzv. extrapolaci), protože absolutní hodnota polynomu nabývá velkých hodnot.



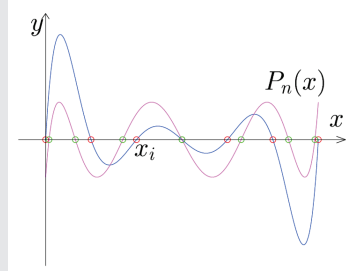
**Poznámka:**

Není obecně vhodné interpolovat polynomem funkci, která je dána velkým počtem svých hodnot. Stupeň interpolačního polynomu by potom byl velký.



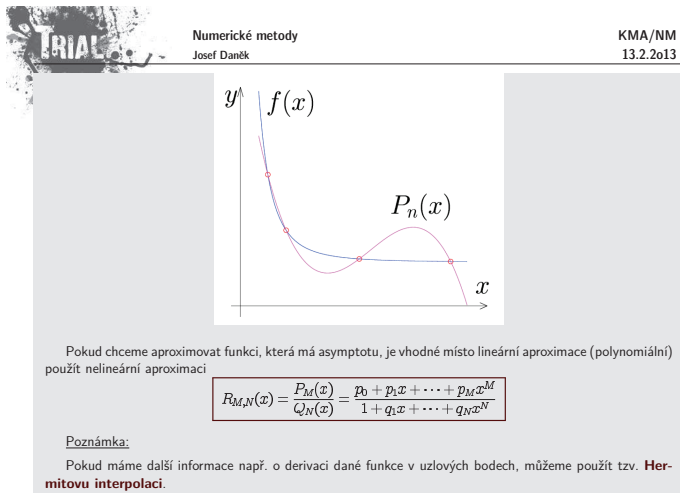
**Poznámka:**

Použijeme-li vhodné zvolené neekvidistantní uzly, můžeme amplitudy chyby minimalizovat. (Vhodnou volbou jsou uzly zvolené jako kořeny tzv. Čebyševových polynomů.)



**Poznámka:**

Interpolace polynomem není obecně vhodná např. pro funkce, které mají asymptotu.



Pokud chceme aproximovat funkci, která má asymptotu, je vhodné místo lineární aproximace (polynomiální) použít nelineární aproximaci

$$P_{M,N}(x) = \frac{P_M(x)}{Q_N(x)} = \frac{p_0 + p_1x + \dots + p_Mx^M}{1 + q_1x + \dots + q_Nx^N}$$

**Poznámka:**

Pokud máme další informace např. o derivaci dané funkce v uzlových bodech, můžeme použít tzv. **Hermitovu interpolaci**.

**Poznámka:**

Všimněme si, že v konstrukci interpolačního polynomu nezáleží na pořadí zadaných tabulkových bodů.

V řadě případů potřebujeme kromě  $a_i$  vypočítat hodnotu polynomu v daném bodě  $\alpha$ , tj.

$$N_n(\alpha) = a_0 + a_1(\alpha - x_0) + a_2(\alpha - x_0)(\alpha - x_1) + \dots + a_n(\alpha - x_0)(\alpha - x_1) \dots (\alpha - x_{n-1})$$

Při vhodném uzávkování můžeme vypočít zefektivnit (zmenšíme počet operací sčítání a násobení):

$$N_n(\alpha) = a_0 + (\alpha - x_0)[a_1 + (\alpha - x_1)[a_2 + (\alpha - x_2)[a_3 + \dots [a_n]]]]$$

Tento postup můžeme samozřejmě použít jen tehdy, když už známe koeficienty  $a_i$ .

Chceme-li vypočítat pouze hodnotu polynomu  $N_n(\alpha)$  v bodě  $\alpha$  za co nejmenšího počtu operací a nepotřebujeme-li koeficienty  $a_i$ , použijeme tzv. **Nevilleův algoritmus**.

Princip je podobný jako v algoritmu pro určení koeficientů Newtonova polynomu.

**Nevilleův algoritmus**

1.  $P_{i,0} = f(x_i), \quad i = 0, 1, \dots, n$

2.  $P_{i,k} = P_{i,k-1} + (\alpha - x_i) \frac{P_{i,k-1} - P_{i-1,k-1}}{x_i - x_{i-1}}$

KMA/NM  
13.2.2013

3.  $N_n(\alpha) = P_{nn}$

Princip Nevilleova algoritmu je ukázan v následujícím příkladu.

**Příklad** Vypočítejte  $f(3,5)$ , kde funkce  $f(x)$  je dána tabulkou:

$x_i$	1	2	4	5
$f(x_i)$	1	8	64	125

**Řešení:**

Uzly  $x_i$  je výhodné uspořádat podle rostoucí vzdálenosti od bodu  $\alpha$ , v němž chceme stanovit přibližnou hodnotu funkce  $f(x)$ . Podle rozdílů hodnot  $P_{i,i}$  a  $P_{i-1,i-1}$  ( $i = 1, \dots, n$ ) lze rozhodnout o předčasném ukončení Nevilleova algoritmu, popř. o vhodnosti interpolace pomocí  $N_n(x)$ .

$\alpha - x_i$	$x_i$	$f(x_i)$			
-0,5	4	<b>64</b>			
1,5	2	8	$8 + 1,5 \frac{8 - 64}{2 - 4}$		
			<b>= 50</b>		
-1,5	5	125	$125 - 1,5 \frac{125 - 8}{5 - 2}$	$66,5 - 1,5 \frac{66,5 - 50}{5 - 4}$	
			<b>= 66,5</b>	<b>= 41,75</b>	
2,5	1	1	$1 + 2,5 \frac{1 - 125}{1 - 5}$	$78,5 + 2,5 \frac{78,5 - 66,5}{1 - 2}$	$48,5 + 2,5 \frac{48,5 - 78,5}{1 - 4}$
			<b>= 78,5</b>	<b>= 48,5</b>	<b>= 42,875</b>

Z následujících obrázků je patrný význam hodnot, které dostáváme na diagonále.

**Poznámka:**

Vyděme z předpokladu, že máme přibližně interpolovat hodnotu tabulkou dané funkce v libovolném bodě. Pokud nutně netrváme na tom, že v tomto bodě chceme přesně určit hodnotu interpolačního polynomu procházejícími všemi tabulkovými body, přistupujeme k Nevilleovu algoritmu iteračně, tj. pokud bude rozdíl po sobě jdoucích diagonálních prvků dostatečně malý, ukončíme výpočet.

V tomto případě je ovšem rozumné seřadit uzlové body podle rostoucí vzdálenosti od zadaného bodu, ve kterém interpolujeme hodnotu funkce.

V následujících příkladech jsou demonstrovány výsledky pro stejné zadání, ovšem při použití různých seřazení uzlových bodů:

- 1) od nejbližšího po nejvzdálenější,
- 2) od nejvzdálenějšího po nejbližší.

**Příklad 1** Interpolujte hodnotu zadané funkce  $f$  v bodě  $\alpha = 3,6$ .

$x_i$	1	2	3	4	5	6	7
$f(x_i)$	-5	14	19	16	12	14	35

Výsledky Nevilleova algoritmu, když uvažujeme uzly seřazené podle vzdálenosti od  $\alpha$  **vzestupně**:

výsledky získané v MATLABu

x(k)	f(k)	Aproximace f(alfa)
4.0000	16.0000	
3.0000	19.0000	17.2000
5.0000	12.0000	16.9000   17.3200
2.0000	14.0000	12.9333   19.2800   17.7120
6.0000	14.0000	14.0000   11.4400   17.7120   17.7120
1.0000	-5.0000	4.8800   28.5920   17.4432   17.7926   17.7228
7.0000	35.0000	12.3333   -13.0080   15.2800   18.9574   17.9674   17.6901

**Příklad 2** Interpolujte hodnotu zadané funkce  $f$  v bodě  $\alpha = 3,6$ .

$x_i$	1	2	3	4	5	6	7
$f(x_i)$	-5	14	19	16	12	14	35

Výsledky Nevilleova algoritmu, když uvažujeme uzly seřazené podle vzdálenosti od  $\alpha$  **sestupně**:

výsledky získané v MATLABu

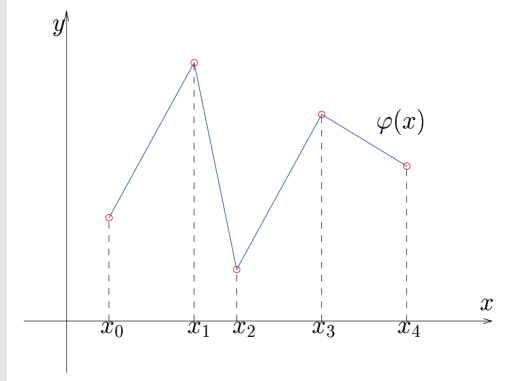
x(k)	f(k)	Aproximace f(alfa)
7.0000	35.0000	
1.0000	-5.0000	12.3333
6.0000	14.0000	4.8800   -13.0080
2.0000	14.0000	14.0000   28.5920   15.2800
5.0000	12.0000	12.9333   11.4400   17.4432   18.9574
3.0000	19.0000	16.9000   19.2800   17.7120   17.7926   17.9674
4.0000	16.0000	17.2000   17.3200   17.7120   17.7120   17.7228   17.6901

**Poznámka**

Jak se dá interpretovat libovolná (tj. i nedиаagonalní) hodnota v trojúhelníkové matici, kterou získáváme Nevilleovým algoritmem?

**Interpolace spline funkcemi**

Nejjednodušší spline funkcí je tzv. **lineární spline funkce**; jde vlastně o lomenou čáru spojující zadané interpolované body.



Máme danou funkci  $f$  tabulkou hodnot  $\{x_i, f_i\}$   $i = 0, 1, \dots, n$ .

Funkci  $s(x)$  definovanou na intervalu  $\langle x_0, x_n \rangle$  nazýváme **lineární spline interpolací** funkce  $f(x)$ , má-li následující vlastnosti:

(i) na každém intervalu  $\langle x_i, x_{i+1} \rangle$ ,  $i = 0, 1, \dots, n-1$  je **polynom prvního stupně**, tj.

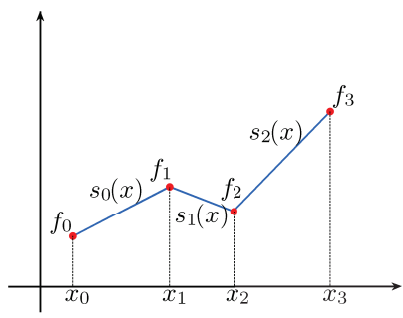
$$s(x) = s_i(x), \quad x \in \langle x_i, x_{i+1} \rangle, \quad \text{kde } s_i(x) = a_i + b_i(x - x_i)$$

(ii) **splňuje interpolační podmínky**  $s(x_i) = f(x_i)$ , tj.

$$\begin{aligned} s_i(x_i) &= f_i, \quad i = 0, 1, \dots, n-1 \\ s_{n-1}(x_n) &= f_n \end{aligned}$$

(iii) je **spojitá** na  $\langle x_0, x_n \rangle$ , tj. i v uzlech  $x_i$

$$s_i(x_{i+1}) = s_{i+1}(x_{i+1}), \quad i = 0, 1, \dots, n-2$$



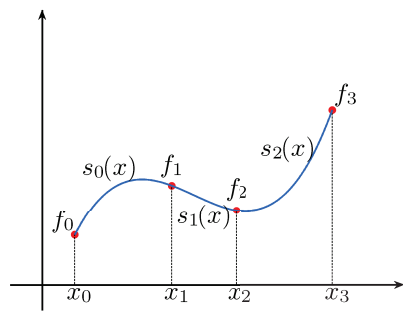
**Poznámka:** Těmito požadavky je funkce  $s(x)$  určena jednoznačně.

- (i) ... hledáme  $2n$  koeficientů  $a_i$  a  $b_i$
- (ii) představuje  $(n+1)$  podmínek
- (iii) představuje  $(n-1)$  podmínek

Platí: 
$$s_i(x) = f_i + \frac{f_{i+1} - f_i}{h_i}(x - x_i), \quad h_i = x_{i+1} - x_i, \quad i = 0, 1, \dots, n-1$$

Pokud bychom chtěli, aby byla aproximace spline funkcí hladká, musíme použít polynomy vyššího stupně než 1.

Nejvíce používanou je tzv. **kubická spline interpolace**, která používá polynomy 3. stupně.



Poznamenejme zde, že ostatní volby stupně polynomů nepřinášejí lepší výsledky a výpočty jsou v případě vyšších stupňů složitější.

**Kubická spline interpolace**

Funkce  $f$  je dána tabulkou  $\{x_i, f_i\}$ ,  $i = 0, 1, \dots, n$

Funkci  $s(x)$  definovanou na intervalu  $\langle x_0, x_n \rangle$  nazýváme **kubickou spline interpolací** funkce  $f$ , má-li následující vlastnosti:

(i) je na každém intervalu  $\langle x_i, x_{i+1} \rangle$ ,  $i = 0, 1, \dots, n-1$  **polynomem 3. stupně** ve tvaru

$$s_i(x) = a_i + b_i(x - x_i) + \frac{c_i}{2}(x - x_i)^2 + \frac{d_i}{6}(x - x_i)^3$$

(ii) **splňuje interpolační podmínky**  $s(x_i) = f(x_i)$ , tj.

$$\begin{aligned} s_i(x_i) &= f_i, \quad i = 0, 1, \dots, n-1 \\ s_{n-1}(x_n) &= f_n \end{aligned}$$

(iii) je **spojitá** na  $\langle x_0, x_n \rangle$ , tj. v uzlech  $x_i$  platí

$$s_i(x_{i+1}) = s_{i+1}(x_{i+1}), \quad i = 0, 1, \dots, n-2$$

(iv) má **spojitou první derivaci** na  $\langle x_0, x_n \rangle$

$$s'_i(x_{i+1}) = s'_{i+1}(x_{i+1}), \quad i = 0, 1, \dots, n-2$$

(v) má **spojitou druhou derivaci** na  $\langle x_0, x_n \rangle$

$$s''_i(x_{i+1}) = s''_{i+1}(x_{i+1}), \quad i = 0, 1, \dots, n-2$$

Funkce  $s(x)$  není podmínkami (ii) – (v) určena jednoznačně:

- (ii) ...  $(n+1)$  podmínek
- (iii) ...  $(n-1)$  podmínek
- (iv) ...  $(n-1)$  podmínek
- (v) ...  $(n-1)$  podmínek
- celkem ...  $4n-2$  podmínek; (počet koeficientů je ale  $4n$ )

2 podmínky je nutno dodat:

(A) přirozené podmínky

$$s'(x_0) = s'(x_n) = 0$$

(B) podmínky periodicity (s periodou  $T = x_n - x_0$ )

$$s(x_0) = s(x_n) \dots \text{splněna automaticky (} f_0 = f_n \text{)}$$

$$s'(x_0) = s'(x_n), \quad s''(x_0) = s''(x_n)$$

(C) podmínky tečen

$$s'(x_0) = y'_0, \quad s'(x_n) = y'_n, \quad \text{kde } y'_0, y'_n \text{ jsou daná čísla}$$

(D) viz MATLAB

Konstrukce kubické spline funkce

Funkci  $s(x)$  lze psát ve tvaru

$$s(x) = \eta_0 s_0(x) + \eta_1 s_1(x) + \dots + \eta_{n-1} s_{n-1}(x)$$

kde  $\eta_i = \eta_i(x)$  jsou charakteristické funkce intervalu, tj.

$$\eta_i = 1 \text{ na } \langle x_i, x_{i+1} \rangle, \quad \eta_{i-1} = 1 \text{ na } \langle x_{i-1}, x_i \rangle, \quad \eta_i = 0 \text{ jinde}$$

Snadno lze odvodit:

$$s(x_i) = \alpha_i, \quad s'(x_i) = \beta_i, \quad s''(x_i) = c_i, \quad s'''(x_i+) = d_i, \quad s'''(x_i-) = d_{i-1}$$

Pomocí těchto vztahů přepíšeme podmínky (ii) až (v) (označíme  $h_i = x_{i+1} - x_i$ ,  $i = 0, 1, \dots, n-1$ )

(ii) interpolační podmínky

$$f_i = \alpha_i, \quad i = 0, 1, \dots, n-1$$

$$f_n = \alpha_{n-1} + \beta_{n-1} h_{n-1} + \frac{c_{n-1}}{2} h_{n-1}^2 + \frac{d_{n-1}}{6} h_{n-1}^3$$

(iii) spojitost

$$f_{i+1} = \alpha_i + \beta_i h_i + \frac{c_i}{2} h_i^2 + \frac{d_i}{6} h_i^3, \quad i = 0, 1, \dots, n-2$$

$$[s_{i+1}(x_{i+1}) = s_i(x_{i+1})]$$

(iv) spojitost derivace

$$b_i + c_i h_i + \frac{d_i}{2} h_i^2 = b_{i+1}, \quad i = 0, 1, \dots, n-2$$

$$[s'_i(x_{i+1}) = s'_{i+1}(x_{i+1})]$$

(v) spojitost 2. derivace

$$c_i + d_i h_i = c_{i+1}, \quad i = 0, 1, \dots, n-2$$

$$[s''_i(x_{i+1}) = s''_{i+1}(x_{i+1})]$$

Z těchto podmínek lze sestavit soustavu lineárních algebraických rovnic pro neznámé  $c_0, \dots, c_n$  a jejich prostřednictvím vypočítat  $b_0, \dots, b_n$  a  $d_0, \dots, d_n$ . Po úpravách dostaneme:

$$\alpha_k c_{k-1} + 2c_k + \beta_k c_{k+1} = g_k, \quad k = 1, 2, \dots, n-1$$

kde

$$\beta_k = \frac{h_k}{h_k + h_{k-1}}$$

$$\alpha_k = 1 - \beta_k$$

$$g_k = \frac{6}{h_k + h_{k-1}} \left[ \frac{f_{k+1} - f_k}{h_k} - \frac{f_k - f_{k-1}}{h_{k-1}} \right]$$

Vztahy představují soustavu  $n-1$  rovnic pro  $n+1$  neznámých. Pro jednoznačnost je třeba přidat jednu z podmínek (A), (B), (C). Soustava bude mít tvar:

$$\begin{bmatrix} 2 & \beta_0 & & & & \\ \alpha_1 & 2 & \beta_1 & & & \\ & \alpha_2 & 2 & \beta_2 & & \\ & & & & \ddots & \\ & & & & & \alpha_{n-1} & 2 & \beta_{n-1} \\ & & & & & & \alpha_n & 2 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{n-1} \\ g_n \end{bmatrix} \quad (*)$$

(\*) ... první a poslední řádek soustavy představuje podmínka (A) nebo (B).  
 ... použití podmínky (C) znamená vyškrtnout 1. a poslední řádek a sloupec

Poznámka:

Matice soustavy je tridiagonální a ostře diagonálně dominantní ( $\alpha_k + \beta_k = 1$ ).  $\Rightarrow$  je regulární  $\Rightarrow$  řešení.

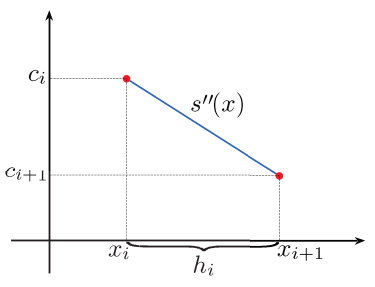
Soustavu řešíme GEM pro tridiagonální matici.

**Odvození tridiagonální matice pro výpočet koeficientů  $c_i$**

Druhá derivace  $s''_i(x) = c_i + d_i(x - x_i)$  je lineární funkcí, tj. známe-li  $s''(x_i) = c_i$ , pak na  $\langle x_i, x_{i+1} \rangle$  můžeme psát

$$s''_i(x) = c_i \frac{x_{i+1} - x}{h_i} + c_{i+1} \frac{x - x_i}{h_i} \quad \textcircled{*}$$

(Ověření:  $s''_i(x_i) = c_i$ ,  $s''_i(x_{i+1}) = c_{i+1}$ ,  $s''_i(x)$  je lineární.)



Integraci  $\textcircled{*}$  dostaneme

$$s'_i(x) = -c_i \frac{(x_{i+1} - x)^2}{2h_i} + c_{i+1} \frac{(x - x_i)^2}{2h_i} + A_i \quad \textcircled{**}$$

$$s_i(x) = c_i \frac{(x_{i+1} - x)^3}{6h_i} + c_{i+1} \frac{(x - x_i)^3}{6h_i} + A_i(x - x_i) + B_i$$

Z interpolačních podmínek plyne ( $s(x_i) = f_i$ ):

$$B_i = f_i - \frac{c_i h_i^2}{6}$$

Pro  $x = x_{i+1}$

$$f_{i+1} = c_{i+1} \frac{h_i^2}{6} + A_i h_i + f_i - \frac{c_i h_i^2}{6}$$

$$\Rightarrow A_i = \frac{f_{i+1} - f_i}{h_i} - \frac{(c_{i+1} - c_i) h_i}{6} \quad \textcircled{***}$$

Ze spojitosti první derivace  $s'_i(x_{i+1}) = s'_{i+1}(x_{i+1})$ ,  $i = 0, 1, \dots, n-2$  s užitím  $\textcircled{**}$  a  $\textcircled{***}$  plyne

$$\frac{s'_i(x_{i+1})}{c_{i+1} \frac{h_i}{2} + \frac{f_{i+1} - f_i}{h_i} - \frac{(c_{i+1} - c_i) h_i}{6}} = \frac{s'_{i+1}(x_{i+1})}{-c_{i+1} \frac{h_{i+1}^2}{2h_{i+1}} + \frac{f_{i+2} - f_{i+1}}{h_{i+1}} - \frac{(c_{i+2} - c_{i+1}) h_{i+1}}{6}}$$

$$= \frac{s'_{i+1}(x_{i+1})}{s'_{i+1}(x_{i+1})}$$

$$c_i \frac{h_i}{6} + c_{i+1} \left( \frac{h_i}{2} - \frac{h_i}{6} + \frac{h_{i+1}}{2} - \frac{h_{i+1}}{6} \right) + c_{i+2} \frac{h_{i+1}}{6} = \frac{f_{i+2} - f_{i+1}}{h_{i+1}} - \frac{f_{i+1} - f_i}{h_i}$$

$$= \frac{h_i}{3} = \frac{h_{i+1}}{3}$$

Vynásobíme  $\frac{6}{h_i + h_{i+1}}$  a dostaneme

$$\frac{h_i}{h_i + h_{i+1}} c_i + 2c_{i+1} + \frac{h_{i+1}}{h_i + h_{i+1}} c_{i+2} = \frac{6}{h_i + h_{i+1}} \left( \frac{f_{i+2} - f_{i+1}}{h_{i+1}} - \frac{f_{i+1} - f_i}{h_i} \right) = g_i$$

**Minimální vlastnost a odhad chyby**

Označme  $S_1(a, b)$  množinu funkcí  $f$ , které splňují podmínky (ii) až (v) a podmínku (A) a jsou navíc na  $\langle a, b \rangle$  integrovatelné s kvadrátem. Mezi všemi funkcemi  $f \in S_1(a, b)$  právě **přirozený kubický spline** udílí nejmenší hodnotu integrálu

$$J(f) = \int_a^b |f''(x)|^2 dx$$

$J(f)$  ... míra celkové křivosti křivky  $y = f(x)$ .

**Věta**

Nechť funkce  $f$  má spojité derivace až do řádu 4 a má omezenou 4. derivaci pro  $x \in \langle a, b \rangle$ . Nechť dále platí:

$$\frac{h_i}{h_i} \leq K, \quad i = 0, 1, \dots, n-1, \quad h = \max_i |x_{i+1} - x_i|$$

Když  $s(x)$  je spline interpolace funkce  $f$  v bodech  $x_i$  a splňuje podmínky  $s'(a) = f'(a), s'(b) = f'(b)$ , potom pro  $x \in \langle a, b \rangle$  platí:

$$|f(x) - s(x)| \leq c_1 K h^4$$

$$|f'(x) - s'(x)| \leq c_2 K h^3$$

$$|f''(x) - s''(x)| \leq c_3 K h^2.$$

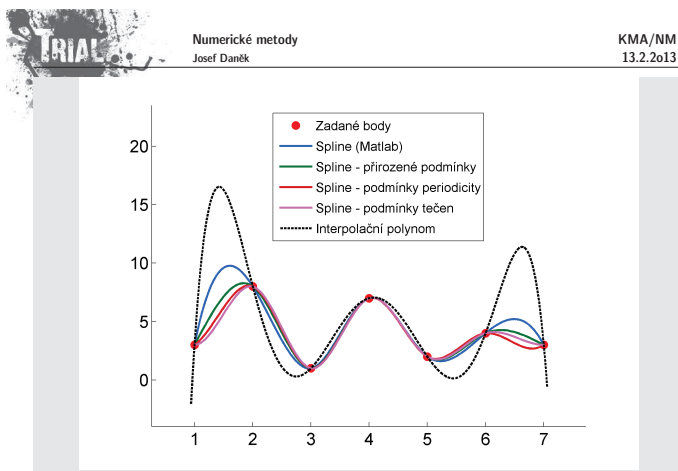
**Příklad**

Určete kubický spline pro funkci zadanou tabulkou

$x_i$	1	2	3	4	5	6	7
$f(x_i)$	3	8	1	7	2	4	3

Použijte různé dodatečné podmínky. Pro spline s podmínkami tečen použijte  $f'(1) = 0$  a  $f'(7) = 0$ .





Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

## Kapitola 8. Aproximace funkcí - II

### Aproximace funkcí

- Aproximace na okolí bodu** - aproximujeme chování funkce „v malém okolí bodu“ ✓
- Interpolace** - tabulkou danými body prokládáme polynom, tj. požadujeme-li, aby aproximace přesně procházela zadanými body. ✓
- $L_2$ -aproximace** - použijeme, hledáme-li funkční závislost mezi tabulkou danými body (získaných například měřeními), kde nutně nevyžadujeme, aby aproximace danými body procházela. Důvodem můžou být např. chyby, se kterými jsme hodnoty naměřili.
- určíme systém jednoduchých **základních (bázových) funkcí** (ne nutně polynomů)  $\varphi_0, \varphi_1, \varphi_2, \dots, \varphi_n$  a funkci  $f$  aproximujeme lineární kombinací základních funkcí
 
$$\varphi(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + \dots + c_n\varphi_n(x)$$
- Otázka výběru aproximace se tedy převede na určení hodnot parametrů  $c_0, c_1, \dots, c_n$  podle nějakého kritéria vhodného pro konkrétní úlohu.

**Poznámka:** Velmi často budeme za základní funkce volit funkce  $1, x, x^2, \dots, x^n$ , tj. aproximaci  $\varphi$  budeme hledat ve třídě polynomů nejvýše  $n$ -tého stupně.

### Úvod d diskrétní $L_2$ -aproximace

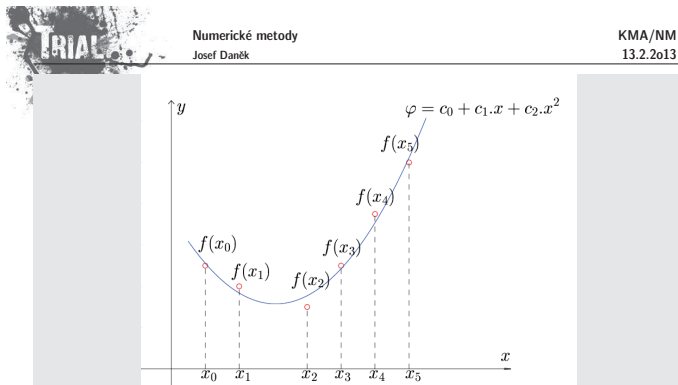
**Mýšlenka**  
Chceme aproximovat funkci, která je dána tabulkou  $\{[x_i, f(x_i)], i = 0, 1, \dots, n\}$ .  
V případě, kdy jsou  $f(x_i)$  zatíženy chybou (např. výsledky měření) nebo pokud je bodů „mnoho“, není vhodné provádět interpolaci.  
Aproximaci  $\varphi$  hledáme ve tvaru

$$\varphi(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + \dots + c_m\varphi_m(x),$$

kde  $\varphi_i$  jsou zadané funkce a  $c_i$  hledané parametry.

- počet bázových funkcí  $\varphi_i$  je menší než počet zadaných bodů ( $m < n$ )
- v případě rovnosti se může jednat o interpolaci (záleží na zvolených bázových funkcích)
- o interpolaci se může jednat i pokud je  $m < n$

Naším cílem je minimalizovat „odchylku“ funkce  $\varphi$  od zadaných dat.



U interpolace jsme požadovali, aby aproximace přímo procházela zadanými body, tj. chyba  $e_i = f(x_i) - \varphi(x_i) = 0$ .  
Nyní na tomto netrváme, pouze chceme tuto chybu v nějakém smyslu minimalizovat.

Jakou použít normu pro měření chyby  $e$  ?

- $\max_{0 \leq i \leq n} \{ |f(x_i) - \varphi(x_i)| \}$
- $\frac{1}{n+1} \sum_{i=0}^n |f(x_i) - \varphi(x_i)|$
- $\sqrt{\frac{1}{n+1} \sum_{i=0}^n [f(x_i) - \varphi(x_i)]^2}$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

Cílem je chybu minimalizovat  $\Rightarrow$  vybereme tu normu, která umožní nejsnázi postup.

**Uvažujme příklad:**

$x_i$	1	2	3
$f(x_i)$	1	2	2

$$\varphi(x) = c_0 + c_1x$$

Jak by vypadala minimalizace s užitím předchozích norem?

- $\min_{c_0, c_1} \max \{ |1 - c_0 - c_1|, |2 - c_0 - 2c_1|, |2 - c_0 - 3c_1| \}$  ... pro počítání nevhodné
- $\min_{c_0, c_1} \frac{1}{3} (|1 - c_0 - c_1| + |2 - c_0 - 2c_1| + |2 - c_0 - 3c_1|)$  ... opět nevhodné
- $\min_{c_0, c_1} \sqrt{\frac{1}{3} [(1 - c_0 - c_1)^2 + (2 - c_0 - 2c_1)^2 + (2 - c_0 - 3c_1)^2]}$  zjednodušíme  
(Nezáporná funkce  $f$  nabývá svého minima ve stejném bodě jako nabývá minima funkce  $\sqrt{f}$ )

$$\min_{c_0, c_1} \underbrace{[(1 - c_0 - c_1)^2 + (2 - c_0 - 2c_1)^2 + (2 - c_0 - 3c_1)^2]}_{(*)}$$

(\*) ... kvadratická funkce proměnných  $c_0, c_1$  (konvexní)  $\Rightarrow$  je hladká, snadno se derivuje

### Formulace

Je dána funkce  $f$  tabulkou hodnot v  $n + 1$  bodech  $x_0, x_1, \dots, x_n$   $\frac{x_i}{f(x_i)} \left| \dots \right.$ .  
Zvolíme tvar aproximující funkce  $\varphi(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + \dots + c_m\varphi_m(x)$   
s počtem parametrů  $c_i$  nejvýše rovným  $n + 1$ .  
**Diskrétní  $L_2$ -aproximaci** funkce  $f$  je potom taková lineární kombinace bázových funkcí  $\varphi_i(x)$ , jejíž koeficienty splňují podmínku, že  $L_2$  norma chyby je minimální, tj.

$$R(f, \varphi) = \sum_{i=0}^n [f(x_i) - \varphi(x_i)]^2 = \sum_{i=0}^n \left[ f(x_i) - \sum_{j=0}^m c_j \varphi_j(x_i) \right]^2$$

je minimální.

**Poznámka:**  
Tato nejlepší aproximace má velmi dobré statistické vlastnosti a vyrovnává vliv náhodných chyb v zadaných (naměřených) funkčních hodnotách.

### Diskrétní $L_2$ -aproximace lineárním polynomem

Úkolem je stanovit diskrétní  $L_2$ -aproximaci funkce  $f$  dané tabulkou  $\{x_i, f_i\}, i = 0, 1, \dots, n$  lineárním polynomem, tj. zvolíme např.  $\varphi_0(x) = 1, \varphi_1(x) = x$ .  
Tedy  $\varphi(x) = c_0 + c_1x$

Minimalizujeme funkci  $R = \sum_{i=0}^n [f_i - c_0 - c_1x_i]^2$

Nutná a postačující podmínka minima je

$$\frac{\partial R}{\partial c_0} = -2 \sum_{i=0}^n [f_i - c_0 - c_1 x_i] = 0$$

$$\frac{\partial R}{\partial c_1} = -2 \sum_{i=0}^n [f_i - c_0 - c_1 x_i] x_i = 0$$

Koeficienty  $c_0$  a  $c_1$  nalezneme jako řešení soustavy

$$(n+1)c_0 + \left(\sum_{i=0}^n x_i\right) c_1 = \sum_{i=0}^n f_i$$

$$\left(\sum_{i=0}^n x_i\right) c_0 + \left(\sum_{i=0}^n x_i^2\right) c_1 = \sum_{i=0}^n f_i x_i$$

Maticově zapsáno

$$Pc = q, \text{ kde } P \text{ je symetrická, pozitivně definitní.}$$

Jiný postup:

Užijeme metodu pro řešení „neřešitelných soustav“.

Platí (mělo by):

$$c_0 + c_1 x_i = f_i, \quad i = 0, 1, \dots, n$$

$$\begin{bmatrix} 1 & x_0 \\ 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}$$

Maticově zapsáno

$$Qc = F$$

Soustava  $Qc = F$  je neřešitelná. Provádíme minimalizaci  $r^T r$ , kde  $r = F - Qc$  je residuum soustavy. Dosadíme-li, pak platí:

$$r^T r = (F - Qc)^T (F - Qc) = F^T F - c^T Q^T F - F^T Qc + c^T Q^T Qc = F^T F - 2c^T Q^T F + c^T Q^T Qc,$$

protože  $\underbrace{(c^T Q^T F)^T}_{\text{číslo}} = \underbrace{(F^T Qc)}_{\text{číslo}}$

Matice  $Q^T Q$  je symetrická, pozitivně definitní. Nutná a postačující podmínka minima:

$$Q^T Qc - Q^T F = 0 \Rightarrow Q^T Qc = Q^T F$$

tzv. **soustava normálních rovnic**.

Platí:

$$P = Q^T Q \quad \text{a} \quad q = Q^T F$$

**Diskrétní  $L_2$ -aproximace kvadratickým polynomem**

Funkci  $f$  aproximujeme kvadratickým polynomem

$$\varphi(x) = c_0 + c_1 x + c_2 x^2$$

Minimalizujeme veličinu

$$R = \sum_{i=0}^n [f_i - c_0 - c_1 x_i - c_2 x_i^2]^2$$

Z nutných a postačujících podmínek minima

$$\frac{\partial R}{\partial c_0} = 0, \quad \frac{\partial R}{\partial c_1} = 0, \quad \frac{\partial R}{\partial c_2} = 0$$

dostaneme soustavu ve tvaru

$$(n+1)c_0 + \left(\sum x_i\right) c_1 + \left(\sum x_i^2\right) c_2 = \sum f_i$$

$$\left(\sum x_i\right) c_0 + \left(\sum x_i^2\right) c_1 + \left(\sum x_i^3\right) c_2 = \sum f_i x_i$$

$$\left(\sum x_i^2\right) c_0 + \left(\sum x_i^3\right) c_1 + \left(\sum x_i^4\right) c_2 = \sum f_i x_i^2$$

Stejnou soustavu dostaneme i postupem, kdy řešíme neřešitelnou soustavu pomocí minimalizace kvadrátu rezidua.

**Poznámka:** V případě, že některé hodnoty chceme eliminovat, např. důsledkem špatného měření, je vhodné použít váhy, tj. minimalizujeme

$$R(f, \varphi, w_i) = \sum_{i=0}^n [f(x_i) - \varphi(x_i)]^2 w_i,$$

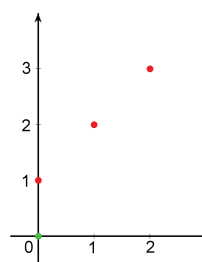
kde  $w_i \dots$  váha uzlu  $x_i$ .

**Příklad**

Aproximujte funkci  $f$ , která je dána tabulkou

$x_i$	0	1	2
$f(x_i)$	1	2	3

pomocí funkce  $\varphi(x) = c_0 x + c_1 x^2 + c_2 x^3$



$Qc = F, \quad Q \dots$  singulární matice

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 4 & 8 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

• první rovnici nelze splnit  $\Rightarrow$  **soustava nemá řešení**

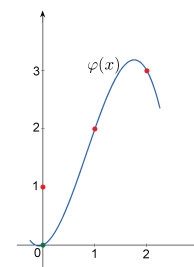
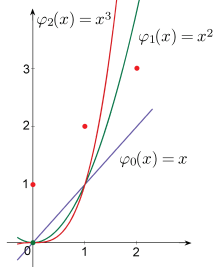
• řešíme metodou nejmenších čtverců

$$Q^T Qc = Q^T F \Rightarrow Q^T Qc = Q^T F$$

$$\begin{bmatrix} 5 & 9 & 17 \\ 9 & 17 & 33 \\ 17 & 33 & 65 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 8 \\ 14 \\ 26 \end{bmatrix}$$

$$\Rightarrow c_0 = 0,4, \quad c_1 = 2,65, \quad c_2 = -1,05$$

$$\varphi(x) = 0,4x + 2,65x^2 - 1,05x^3$$



$$\varphi_0(0) = \varphi_1(0) = \varphi_2(0) = 0 \Rightarrow \varphi(0) = 0$$

**Řešitelnost úlohy diskrétní  $L_2$ -aproximace**

**Definice:** Řekneme, že systém funkcí  $\varphi_j(x), j = 0, 1, \dots, m$ , definovaných na  $(a, b) \supset \text{int}(x_0, x_1, \dots, x_n)$  je **diskrétně lineárně nezávislý**, jsou-li vektory

$$\begin{bmatrix} \varphi_0(x_0) \\ \varphi_1(x_0) \\ \vdots \\ \varphi_m(x_0) \end{bmatrix}, \begin{bmatrix} \varphi_0(x_1) \\ \varphi_1(x_1) \\ \vdots \\ \varphi_m(x_1) \end{bmatrix}, \dots, \begin{bmatrix} \varphi_0(x_n) \\ \varphi_1(x_n) \\ \vdots \\ \varphi_m(x_n) \end{bmatrix}$$

lineárně nezávislé.

**Poznámka:**

Tato definice říká, že hodnost matice  $\Phi = [\varphi_j(x_i)]_{i=0,1,\dots,n}^{j=0,1,\dots,m}$  je rovna  $(m+1)$ .

Platí, že  $m \leq n$ .

**Podmíněnost úlohy diskrétní  $L_2$ -aproximace**

Budeme-li aproximovat na intervalu  $(0,1)$  funkci  $f$  a zvolíme-li ekvidistantní dělení a bázové funkce budeme volit  $\varphi_j = x^j$ , bude matice  $P = Q^T Q$  soustavy normálních rovnic blízká **Hilbertově matici**, která je velmi špatně podmíněná.

**Řešení:** Za funkce  $\varphi_j(x)$  volíme **ortogonální polynomy** (např. Gramovy polynomy).

**Poznámka:** Ze systému  $n$ -lineárně nezávislých funkcí  $g_i$  lze pomocí Gram-Schmidtova ortogonalizačního

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

procesu zkonstruovat systém ortogonálních funkcí.

**Spojité  $L_2$ -aproximace**

**Definice:**  
Mějme funkci  $w = w(x)$ , která je definována na  $(a, b)$  a je kladná a omezená. Množinu reálných funkcí  $f = f(x)$  definovaných na  $(a, b)$  takových, že

$$\int_a^b w(x)[f(x)]^2 dx < \infty$$

označíme  $L_2(a, b)$ . Skalárním součinem dvou funkcí  $f, g \in L_2(a, b)$  nazýváme číslo

$$(f, g) = \int_a^b w(x)f(x)g(x) dx$$

Číslo

$$\|f\| = \sqrt{(f, f)} = \left( \int_a^b w(x)[f(x)]^2 dx \right)^{\frac{1}{2}}$$

nazýváme **normou funkce**  $f$  v  $L_2(a, b)$ .  
Funkce  $f, g$  se nazývají **ortogonální**, platí-li

$$(f, g) = 0$$

Chceme tedy minimalizovat veličinu

$$R(f, \varphi) = \left( f - \sum_{j=0}^m c_j \varphi_j, f - \sum_{j=0}^m c_j \varphi_j \right)$$

Nutné a postačující podmínky minima mají tvar

$$\frac{\partial R}{\partial c_k} = 0, \quad k = 0, 1, \dots, m$$

Derivováním a jednoduchými úpravami dostaneme

$$R(f, \varphi) = (f, f) - 2\left(f, \sum_{j=0}^m c_j \varphi_j\right) + \left(\sum_{j=0}^m c_j \varphi_j, \sum_{j=0}^m c_j \varphi_j\right)$$

$$\frac{\partial R}{\partial c_k} = 0 - 2(f, \varphi_k) + 2\left(\varphi_k, \sum_{j=0}^m c_j \varphi_j\right) = 0$$

$$\sum_{j=0}^m (\varphi_k, c_j \varphi_j) = (f, \varphi_k)$$

$$\sum_{j=0}^m c_j (\varphi_k, \varphi_j) = (f, \varphi_k)$$

Zapsáním všech podmínek dostaneme soustavu

$$\begin{aligned} (\varphi_0, \varphi_0)c_0 + (\varphi_0, \varphi_1)c_1 + \dots + (\varphi_0, \varphi_m)c_m &= (f, \varphi_0) \\ (\varphi_1, \varphi_0)c_0 + (\varphi_1, \varphi_1)c_1 + \dots + (\varphi_1, \varphi_m)c_m &= (f, \varphi_1) \end{aligned}$$

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

Minimalizujeme funkci

$$r(c_0, c_1) = \int_1^e |f(x) - \varphi(x)|^2 dx = \int_1^e (\ln x - c_0 - c_1 x)^2 dx$$

Podmínky minima

$$\frac{\partial r}{\partial c_0} = -2 \int_1^e (\ln x - c_0 - c_1 x) dx = 0$$

$$\frac{\partial r}{\partial c_1} = -2 \int_1^e (\ln x - c_0 - c_1 x)x dx = 0$$

$$c_0 \int_1^e 1 dx + c_1 \int_1^e x dx = \int_1^e \ln x dx$$

$$c_0 \int_1^e x dx + c_1 \int_1^e x^2 dx = \int_1^e x \ln x dx$$

$$\begin{bmatrix} e-1 & \frac{1}{2}(e^2-1) \\ \frac{1}{3}(e^2-1) & \frac{1}{3}(e^3-1) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{e^2+1}{4} \end{bmatrix}$$

$$\int_1^e \frac{\ln x}{v} \frac{1}{u} dx = \left| \begin{array}{l} v = \ln x \quad v' = \frac{1}{x} \\ u' = 1 \quad u = x \end{array} \right| = [x \ln x]_1^e - \int_1^e 1 dx = e - e + 1 = 1$$

$$\int_1^e \frac{x \ln x}{u} \frac{1}{v} dx = \left| \begin{array}{l} v = \ln x \quad v' = \frac{1}{x} \\ u' = x \quad u = \frac{x^2}{2} \end{array} \right| = \left[ \frac{x^2}{2} \ln x \right]_1^e - \int_1^e \frac{x^2}{2} \frac{1}{x} dx = \frac{e^2}{2} - \frac{1}{4}(e^2-1) = \frac{e^2}{4} + \frac{1}{4}$$

Řešení:  $c_0 \doteq -0,46518$ ,  $c_1 \doteq 0,56325$   
 $\varphi(x) = 0,56325x - 0,46518$

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$$(\varphi_m, \varphi_0)c_0 + (\varphi_m, \varphi_1)c_1 + \dots + (\varphi_m, \varphi_m)c_m = (\varphi_m, f)$$

**Věta**  
Jsou-li funkce  $\varphi_0, \varphi_1, \dots, \varphi_m$  lineárně nezávislé, má úloha spojitě  $L_2$ -aproximace jediné řešení. Koeficienty  $c_j^*$  jsou řešením normální soustavy a platí:

$$(f - \varphi^*, \varphi_j) = 0, \quad j = 0, 1, \dots, m$$

tj. funkce  $f - \varphi^*$  je ortogonální ke všem funkcím  $\varphi_j$ .

**Geometrická interpretace**

$\varphi(x) = \sum_{j=0}^m c_j \varphi_j(x)$

Pokud leží funkce  $f$  v množině funkcí  $\varphi(x) = c_0 \varphi_0(x) + c_1 \varphi_1(x) + \dots + c_m \varphi_m(x)$ , potom

$\varphi(x) = \sum_{j=0}^m c_j \varphi_j(x)$

**Příklad**  
Stanovte spojitou  $L_2$ -aproximaci funkce  $f(x) = \ln x$  na  $(1, e)$  lineární funkcí  $\varphi(x) = c_1 x + c_0$ .

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$f(x) = \ln x$   
 $\varphi = 0,56325x - 0,46518$

**Podmíněnost úlohy spojitě  $L_2$ -aproximace**  
**Příklad:** Volíme-li váhu  $w(x) \equiv 1$  a aproximujeme-li funkci  $f = f(x)$  na intervalu  $(0, 1)$  funkcí  $\varphi = \varphi(x)$  ve tvaru

$$\varphi(x) = c_0 + c_1 x + c_2 x^2 + \dots + c_m x^m,$$

tj.  $\varphi_j(x) = x^j$ , platí

$$(\varphi_i, \varphi_j) = \int_0^1 x^i x^j dx = \frac{1}{i+j+1}.$$

Soustava normálních rovnic  $Pc = g$  je opět špatně podmíněná, protože  $P$  je **Hilbertova matice**.

**Řešení problému:** Volíme funkce  $\varphi_j(x)$ ,  $j = 0, 1, \dots, m$ , ortogonální ve smyslu skalárního součinu

$$(\varphi_i, \varphi_j) = \int_a^b w(x)\varphi_i(x)\varphi_j(x) dx.$$

Potom platí:  $(\varphi_i, \varphi_j) = 0$  pro  $i \neq j$  a soustava normálních rovnic má **diagonální matici**. Pak lze psát:

$$c_j^* = \frac{(f, \varphi_j)}{(\varphi_j, \varphi_j)}, \quad j = 0, 1, \dots, m$$

$c_j^* \dots$  **Fourierovy koeficienty**.

**Gram-Schmidtův ortogonalizační proces**

Jsou dány lineárně nezávislé funkce  $g_1, g_2, \dots, g_n$  (prvky jistého prostoru). Hledáme funkce (prvky téhož prostoru), které jsou navzájem po dvou ortogonální.

$$f_1 = g_1$$

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$f_2$  hledáme ve tvaru  $f_2 = g_2 + \kappa_{21}f_1$  a použijeme  $(f_1, f_2) = 0$

$$\frac{(f_2, f_1)}{0} = \frac{(g_2, f_1)}{0} + \kappa_{21} \frac{(f_1, f_1)}{0} \Rightarrow \kappa_{21} = -\frac{(g_2, f_1)}{(f_1, f_1)}$$

$f_3$  hledáme ve tvaru  $f_3 = g_3 + \kappa_{31}f_1 + \kappa_{32}f_2$  a použijeme  $(f_3, f_1) = 0$   
a  $(f_3, f_2) = 0$

$$\frac{(f_3, f_1)}{0} = \frac{(g_3, f_1)}{0} + \kappa_{31} \frac{(f_1, f_1)}{0} + \kappa_{32} \frac{(f_2, f_1)}{0}$$

$$\Rightarrow \kappa_{31} = -\frac{(g_3, f_1)}{(f_1, f_1)}$$

$$\frac{(f_3, f_2)}{0} = \frac{(g_3, f_2)}{0} + \kappa_{31} \frac{(f_1, f_2)}{0} + \kappa_{32} \frac{(f_2, f_2)}{0}$$

$$\Rightarrow \kappa_{32} = -\frac{(g_3, f_2)}{(f_2, f_2)}$$

Obecně  $f_k$  hledáme ve tvaru  $f_k = g_k + \kappa_{k1}f_1 + \kappa_{k2}f_2 + \dots + \kappa_{k,k-1}f_{k-1}$

$$\kappa_{kj} = -\frac{(g_k, f_j)}{(f_j, f_j)}$$

$j = 1, 2, \dots, k-1$

#### Příklad

Najděte ortogonální bázi prostoru polynomů do stupně 2 na  $(0, 10)$ .

Vyjďeme z báze  $g_0 = 1, g_1 = x, g_2 = x^2$ .

$$f_0 = 1$$

$$f_1 = x + \kappa_{10}f_0$$

$$\kappa_{10} = -\frac{(x, 1)}{(1, 1)} = -\frac{\int_0^{10} x dx}{\int_0^{10} 1 dx} = -\frac{50}{10}$$

$$f_1 = x - 5$$

$$f_2 = x^2 + \kappa_{20}f_0 + \kappa_{21}f_1$$

$$\kappa_{20} = -\frac{(x^2, 1)}{(1, 1)} = -\frac{\int_0^{10} x^2 dx}{\int_0^{10} 1 dx} = -\frac{1000}{10} = -33,3$$

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$$\kappa_{21} = -\frac{(x^2, x-5)}{(x-5, x-5)} = -\frac{\int_0^{10} x^3 - 5x^2 dx}{\int_0^{10} x^2 - 5x dx} = -\frac{10000 - 5 \cdot \frac{1000}{3}}{\frac{30000}{3} - 20000} = -10$$

$$f_2 = x^2 - 33,3 - 10(x-5)$$

$$f_2 = x^2 - 10x + 16,6$$

#### Příklad

Určete ortogonální bázi prostoru polynomů do stupně 2 pro uzlové body  $x_0 = 0; x_1 = 0,2; x_2 = 0,4; x_3 = 0,6; x_4 = 0,8; x_5 = 1$

Opět použijeme jako výchozí bázi

$$g_0 = 1 \quad \text{tj.} \quad [1; 1; 1; 1; 1; 1]$$

$$g_1 = x \quad \text{tj.} \quad [0; 0,2; 0,4; 0,6; 0,8; 1]$$

$$g_2 = x^2 \quad \text{tj.} \quad [0; 0,04; 0,16; 0,36; 0,64; 1]$$

$$f_0 = g_0 = 1$$

$$f_1 = g_1 + \kappa_{10}f_0$$

$$\kappa_{10} = -\frac{(g_1, f_0)}{(f_0, f_0)} = -\frac{0 + 0,2 + 0,4 + 0,6 + 0,8 + 1}{6} = -\frac{3}{6} = -\frac{1}{2}$$

$$f_1 = x - \frac{1}{2}$$

$$f_2 = g_2 + \kappa_{20}f_0 + \kappa_{21}f_1$$

$$\kappa_{20} = -\frac{(g_2, f_0)}{(f_0, f_0)} = -\frac{0 + 0,04 + 0,16 + 0,36 + 0,64 + 1}{6} = -\frac{2,2}{6} = -\frac{1,1}{3}$$

$$\kappa_{21} = -\frac{(g_2, f_1)}{(f_1, f_1)} = -\frac{[0; 0,04; 0,16; 0,36; 0,64; 1]^T [-0,5; -0,3; -0,1; 0,1; 0,3; 0,5]}{0,25 + 0,09 + 0,01 + 0,01 + 0,09 + 0,25} = -\frac{0 - 0,012 - 0,016 + 0,036 + 0,192 + 0,5}{0,7} = -\frac{0,7}{0,7} = -1$$

$$f_2 = x^2 - \frac{1,1}{3} - x + \frac{1}{2}$$

$$f_2 = x^2 - x + \frac{2}{15} = x^2 - x + 0,13$$

#### Poznámka

Pokud bychom zvolili jiné uzlové body  $x_i$ , dostali bychom i obecně jiný systém ortogonálních bázových funkcí.

Např. pro  $x_0 = 0; x_1 = 0,125; x_2 = 0,25; x_3 = 0,375; x_4 = 0,5; x_5 = 0,625; x_6 = 0,75; x_7 = 0,875; x_8 = 1$

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

bychom získali následující ortogonální bázi prostoru polynomů do stupně 2:

$$f_0 = 1 \quad f_1 = x - \frac{1}{2} \quad f_2 = x^2 - x + 0,1458 \quad \dots \quad \text{Ověřte (D.cv.).}$$

#### Poznámka:

Uvažujme úlohu (spojité)  $L_2$ -aproximace, kde za aproximující funkci volíme polynom stupně  $n$ .

#### Jak máme volit stupeň polynomu ?

Pokud nemáme další informace, je vhodné řešit normální rovnice postupně pro  $m = 0, 1, 2, \dots$  a sledovat hodnotu

$$\sigma_m^2 = \frac{\sum_{i=0}^n (f(x_i) - \varphi(x_i))^2}{n - m}$$

kde  $\varphi \dots$  polynom stupně  $m$ .

Pokud  $\sigma_m^2$  s rostoucí  $m$  významně klesá, pokračujeme, jinak hodnota po níž nenásleduje výrazný pokles  $\sigma_m^2$  je ze statistických důvodů vhodným stupněm polynomu.

Volíme-li za  $\varphi_j = x^j$

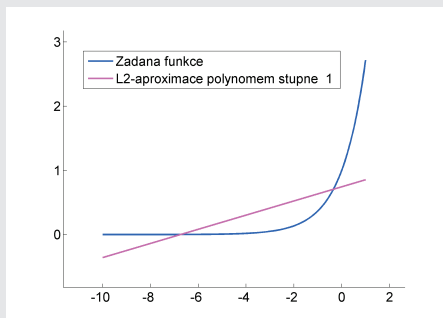
1. musíme řešit soustavu normálních rovnic pro každý stupeň  $m$  znovu.

2. špatná podmíněnost

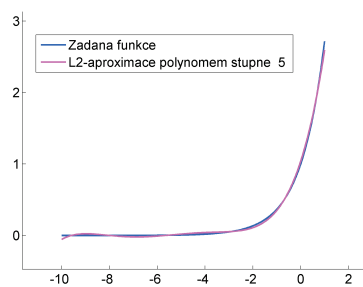
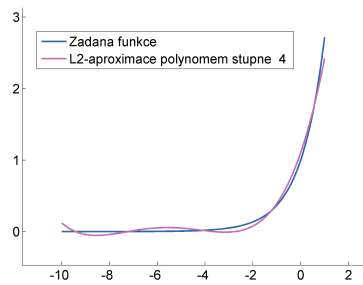
Řešení: Použijeme ortogonální polynomy  $\rightarrow$  potom stačí vždy dopočítat pouze 1 koeficient.

#### Příklad

Určete spojitou  $L_2$ -aproximaci funkce  $f(x) = e^x$  na  $(-10; 1)$  pomocí polynomů stupně nejvýše 1, 2, 3, 4 a 5.



Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013



**Nelineární aproximace metodou nejmenších čtverců**

Např. volíme-li

- 1. přístup je metoda linearizace dat; (\*) zlogaritmuje

$$\varphi(x) = Ce^{Ax}$$

(\*)

$$\ln \varphi = \ln C + Ax$$

$$\Phi = Ax + B$$

(původní body  $[x_i, f_i]$  je třeba transformovat na body  $[x_i, \ln f_i]$ )

získáme  $A, B$  a z  $B$  vypočteme  $C = e^B$

- 2. přístup minimalizujeme  $L_2$  normu chyby přímo

$$R(A, C) = \sum_{i=0}^n (f_i - Ce^{Ax_i})^2$$

parciální derivace:

$$\frac{\partial R}{\partial A} = 2 \sum_{i=0}^n (f_i - Ce^{Ax_i}) (Cx_i e^{Ax_i}) = 0$$

$$\frac{\partial R}{\partial C} = 2 \sum_{i=0}^n (f_i - Ce^{Ax_i}) (e^{Ax_i}) = 0$$

soustava normálních rovnic:

1. rovnice:

$$\sum_{i=0}^n (f_i - Ce^{Ax_i}) (Cx_i e^{Ax_i}) = 0$$

$$C \sum_{i=0}^n f_i x_i e^{Ax_i} - C^2 \sum_{i=0}^n x_i e^{2Ax_i} = 0 \quad / \cdot \frac{1}{C} \neq 0$$

$$\sum_{i=0}^n f_i x_i e^{Ax_i} - C \sum_{i=0}^n x_i e^{2Ax_i} = 0$$

2. rovnice:

$$\sum_{i=0}^n (f_i - Ce^{Ax_i}) (e^{Ax_i}) = 0$$

$$\sum_{i=0}^n f_i e^{Ax_i} - C \sum_{i=0}^n e^{2Ax_i} = 0$$

... soustava nelineárních rovnic, pro řešení lze použít např. Newtonovu metodu.

**Příklad**

Určete diskétní  $L_2$ -aproximaci funkce  $f$  zadané tabulkou

$x_i$	0	1	2	3	4
$f(x_i)$	1.5	2.5	3.5	5	7.5

funkcí ve tvaru

$$\varphi(x) = Ce^{Ax}$$

Pro řešení použijeme oba předchozí přístupy.

**1. přístup**

script v MATLABu

```
x=0:4
f=[1.5 2.5 3.5 5 7.5]
F=log(f)
Q=[x.' 0' x.^1']
P=Q'*Q
g=Q'*F
koef=P\g
A=koef(2)
C=exp(koef(1))
```

výsledky v MATLABu

```
x =
    0    1    2    3    4
f =
 1.5000  2.5000  3.5000  5.0000  7.5000
F =
 0.4055
 0.9163
 1.2528
 1.6094
 2.0149
Q =
 1  0
 1  1
 1  2
 1  3
 1  4
P =
 5  10
 10  30
g =
 6.1989
 16.3097
koef =
 0.4574
 0.3912
A =
 0.3912
C =
 1.5799
```

$$\Rightarrow \varphi_1(x) \doteq 1,5799 e^{0,3912x}$$

**2. přístup**

script v MATLABu

```
koef=fminsearch('R',[1 1]);
AA=koef(1)
CC=koef(2)

%-----
function out=R(koef);
A=koef(1);
C=koef(2);
out=(C-1.5).^2+(C.*exp(A)-2.5).^2+(C.*exp(2*A)-3.5).^2+...
(C.*exp(3*A)-5).^2+(C.*exp(4*A)-7.5).^2;
```

výsledky v MATLABu

```
AA =
 0.3836
CC =
 1.6109
```

$$\Rightarrow \varphi_2(x) \doteq 1,6109 e^{0,3836x}$$

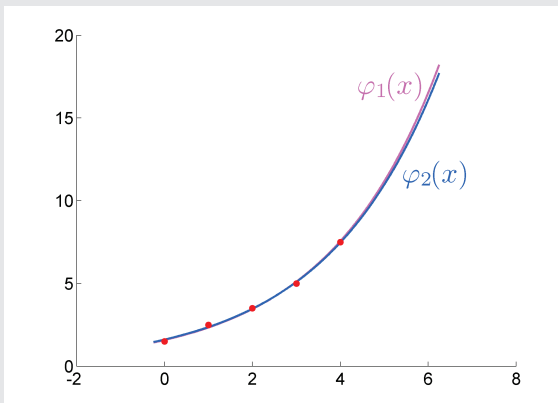


Table 5.6 Change of Variable(s) for Data Linearization

Function, $y = f(x)$	Linearized form, $Y = Ax + B$	Change of variable(s) and constants
$y = \frac{A}{x} + B$	$y = \frac{A}{x} + B$	$X = \frac{1}{x}, Y = y$
$y = \frac{D}{x+C}$	$y + \frac{-1}{C}(xy) + \frac{D}{C}$	$X = xy, Y = y$ $C = \frac{-1}{A}, D = \frac{-B}{A}$
$y = \frac{1}{Ax+B}$	$\frac{1}{y} = Ax + B$	$X = x, Y = \frac{1}{y}$
$y = \frac{x}{Ax+B}$	$\frac{1}{y} = A \frac{1}{x} + B$	$X = \frac{1}{x}, Y = \frac{1}{y}$
$y = A \ln(x) + B$	$y = A \ln(x) + B$	$X = \ln(x), Y = y$
$y = Ce^{Ax}$	$\ln(y) = Ax + \ln(C)$	$X = x, Y = \ln(y)$ $C = e^B$
$y = Cx^A$	$\ln(y) = A \ln(x) + \ln(C)$	$X = \ln(x), Y = \ln(y)$ $C = e^B$
$y = (Ax+B)^{-2}$	$y^{-1/2} = Ax + B$	$X = x, Y = y^{-1/2}$
$y = Cxe^{-Dx}$	$\ln\left(\frac{y}{x}\right) = -Dx + \ln(C)$	$X = x, Y = \ln\left(\frac{y}{x}\right)$ $C = e^B, D = -A$
$y = \frac{L}{1 + Ce^{Ax}}$	$\ln\left(\frac{L}{y} - 1\right) = Ax + \ln(C)$	$X = x, Y = \ln\left(\frac{L}{y} - 1\right)$ $C = e^B$ and $L$ is a constant that must be given

Fourierova analýza

Do této chvíle jsme se zabývali aproximací funkce hlavně pomocí polynomů. V úvodu jsme uvedli, že za bázevých funkce můžeme volit libovolné funkce. Například pro aproximaci periodických funkcí není vhodné použít polynomy (a to jak ve smyslu interpolace tak ve smyslu  $L_2$ -aproximace). Pro aproximaci periodických funkcí je vhodné použít nějaký systém periodických bázevých funkcí, např. systém tzv. trigonometrických polynomů:

$$\begin{aligned} \varphi_0(x) &= 1 \quad (\text{nebo } \frac{1}{2}) \\ \varphi_{2k-1}(x) &= \cos\frac{2\pi kx}{T} \quad k = 1, 2, \dots \\ \varphi_{2k}(x) &= \sin\frac{2\pi kx}{T} \quad k = 1, 2, \dots \end{aligned}$$

kde  $T$  představuje periodu zadané funkce (vzdálenost prvního a posledního uzlu v diskrétním případě, resp. délku zadaného intervalu ve spojitém případě).

Pro jednoduchost uvažujeme ekvidistantní uzly (v diskrétním případě).

Počet uvažovaných bázevých funkcí volíme buď menší než je počet zadaných bodů (ve smyslu  $L_2$ -aproximace), nebo roven počtu zadaných bodů (ve smyslu interpolace).

Jednoduchým cvičením je ukázat, že systém trigonometrických polynomů je ortogonální jak v diskrétním (pozor na počet) tak ve spojitém případě. Ověřte!

Úlohu najít koeficienty  $c_i$  u bázevých funkcí  $\varphi_i$  z vyjádření

$$\varphi(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + \dots + c_n\varphi_n(x),$$

nazýváme v tomto případě **Fourierovou analýzou**.

Formálně pouze přeznačíme koeficienty  $c_i$ , tj.

u bázevých funkce  $\varphi_0(x) = 1$  použijeme koeficient  $A_0$ ,

u bázevých funkcí  $\varphi_{2k-1}(x) = \cos(2\pi kx)/T$  použijeme  $\dots A_k$

u bázevých funkcí  $\varphi_{2k}(x) = \sin(2\pi kx)/T$  použijeme  $\dots B_k$

Následující jednoduchý příklad ukáže princip Fourierovy analýzy.

Příklad

Aproximujte  $2\pi$ -periodickou funkci zadanou tabulkou za použití maximálního počtu bázevých funkcí (tj. ve smyslu interpolace).

$x_j$	0	$\pi/2$	$\pi$	$3\pi/2$
$f(x_j)$	12	-4	0	4

Řešení

Ze zadaní je zřejmé, že perioda zadané funkce je  $2\pi$ . Aproximující trigonometrický polynom budeme tedy volit ve tvaru

$$\varphi(x) = A_0 + A_1 \cos x + B_1 \sin x + A_2 \cos 2x.$$

Zapišeme interpolační podmínky

$$\varphi(x_j) = f(x_j), \quad j = 0, 1, 2, 3,$$

tj.

$$\begin{bmatrix} 1 & \cos x_0 & \sin x_0 & \cos 2x_0 \\ 1 & \cos x_1 & \sin x_1 & \cos 2x_1 \\ 1 & \cos x_2 & \sin x_2 & \cos 2x_2 \\ 1 & \cos x_3 & \sin x_3 & \cos 2x_3 \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ B_1 \\ A_2 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ f(x_3) \end{bmatrix},$$

tj.

$$\begin{bmatrix} 1 & \cos 0 & \sin 0 & \cos 0 \\ 1 & \cos \pi/2 & \sin \pi/2 & \cos \pi \\ 1 & \cos \pi & \sin \pi & \cos 2\pi \\ 1 & \cos 3\pi/2 & \sin 3\pi/2 & \cos 3\pi \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ B_1 \\ A_2 \end{bmatrix} = \begin{bmatrix} 12 \\ -4 \\ 0 \\ 4 \end{bmatrix},$$

tj.

$$\underbrace{\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & -1 \\ 1 & -1 & 0 & 1 \\ 1 & 0 & -1 & -1 \end{bmatrix}}_Q \underbrace{\begin{bmatrix} A_0 \\ A_1 \\ B_1 \\ A_2 \end{bmatrix}}_c = \underbrace{\begin{bmatrix} 12 \\ -4 \\ 0 \\ 4 \end{bmatrix}}_F.$$

tj.

$$\underbrace{\begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}}_{Q^T Q} \underbrace{\begin{bmatrix} A_0 \\ A_1 \\ B_1 \\ A_2 \end{bmatrix}}_c = \underbrace{\begin{bmatrix} 12 \\ -8 \\ 0 \\ 12 \end{bmatrix}}_{Q^T F}.$$

$Q^T Q$  je diagonální, protože funkce  $\varphi_0(x) = \frac{1}{2}$ ,  $\varphi_1(x) = \cos x$ ,  $\varphi_2(x) = \sin x$  jsou diskrétně ortogonální ve smyslu skalárního součinu

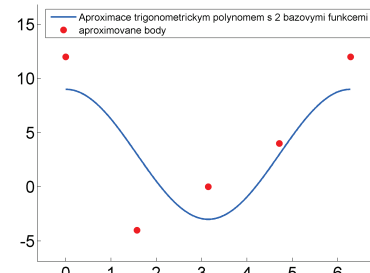
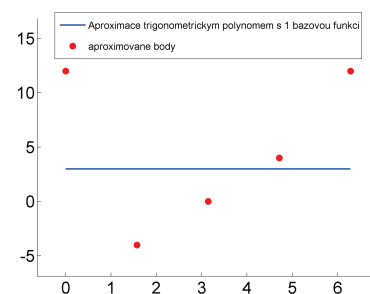
$$(\varphi, \psi) = \sum_{j=0}^3 \varphi(x_j)\psi(x_j) \quad x_j = \frac{2\pi j}{N}, \quad N = 3$$

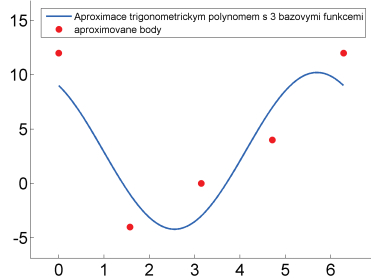
D.cv.

$$\begin{aligned} \left(\frac{1}{2}, \cos x\right) &= \dots = 0 \\ \left(\frac{1}{2}, \sin x\right) &= \dots = 0 \\ (\cos x, \sin x) &= \dots = 0 \end{aligned}$$

Vyřešením soustavy získáme hledané koeficienty  $A_0 = 3$ ,  $A_1 = 6$ ,  $B_1 = -4$ ,  $A_2 = 3$  a tím i aproximující trigonometrický polynom

$$\varphi(x) = 3 + 6 \cos x - 4 \sin x + 3 \cos 2x.$$





**Úloha diskrétní Fourierovy analýzy**

Řadu  $T$  periodických funkcí (integrovatelných) lze vyjádřit ve tvaru **Fourierovy řady**

$$f(x) = \sum_{k=0}^{\infty} \left( a_k \cos \frac{2\pi kx}{T} + b_k \sin \frac{2\pi kx}{T} \right)$$

nebo

$$f(x) = \sum_{k=0}^{\infty} r_k \sin \left( \frac{2\pi kx}{T} + v_k \right), \text{ kde } r_k^2 = a_k^2 + b_k^2, \quad v_k = \arctan \frac{a_k}{b_k}$$

Položili jsme  $a_k = r_k \sin v_k, \quad b_k = r_k \cos v_k.$

$$a \cos \alpha + b \sin \alpha = r \sin v \cos \alpha + r \cos v \sin \alpha = r (\sin v \cos \alpha + \cos v \sin \alpha) = r \sin (\alpha + v).$$

**Fourierovou (harmonickou) analýzou** rozumíme úlohu určit amplitudy  $r_k$  a fáze  $v_k$  tzv. harmonických složek  $r_k \sin \frac{2\pi kx}{T} + v_k$ , je-li dána funkce  $f(x)$ .

**Fourierovou (harmonickou) syntézou** rozumíme úlohu určit funkci  $f$ , jsou-li dány fáze  $v_k$  a amplitudy  $r_k$ .

Tuto úlohu můžeme řešit několika způsoby:

(ž) Vyjdeme z úlohy spojité Fourierovy analýzy a numericky vypočteme Fourierovy koeficienty, které jsou dány:

$$a_k = \frac{2}{T} \int_0^T f(x) \cos \frac{2\pi kx}{T} dx$$

$$b_k = \frac{2}{T} \int_0^T f(x) \sin \frac{2\pi kx}{T} dx$$

Užijeme např. **lichoběžníkové pravidlo** (viz další přednáška). Pozor, **pro velká  $k$  integrandy oscilují!**

(žž) Funkci  $f$  aproximujeme (interpolace, diskrétní  $L_2$ -aproximace) přímo vhodnou funkcí  $\varphi$ , která má tvar **trigonometrického polynomu**.

Použijeme přístup (žž) realizovaný v příkladu.

**Diskrétní Fourierova analýza - ve smyslu interpolace**

**Věta** Trigonometrický polynom

$$\varphi(x) = \frac{A_0}{2} + \sum_{k=1}^L (A_k \cos kx + B_k \sin kx), \quad N = 2L + 1 \quad (N \text{ liché})$$

resp.

$$\varphi(x) = \frac{A_0}{2} + \sum_{k=1}^{L-1} (A_k \cos kx + B_k \sin kx) + \frac{A_L}{2} \cos Lx, \quad N = 2L \quad (N \text{ sudé})$$

splňuje **interpoloční podmínky**

$$\varphi(x_j) = f(x_j), \quad x_j = \frac{2\pi j}{N}, \quad j = 0, 1, \dots, N-1,$$

právě když koeficienty polynomu  $\varphi(x)$  jsou dány pomocí vzorců

$$A_k = \frac{2}{N} \sum_{j=0}^{N-1} f_j \cos kx_j, \quad k = 0, 1, \dots, L$$

$$B_k = \frac{2}{N} \sum_{j=0}^{N-1} f_j \sin kx_j, \quad k = (0, 1, 2, \dots, L$$

$$(A_0 = \frac{2}{N} \sum_{j=0}^{N-1} f_j)$$

**Důkaz:**

$(\varphi(x), \cos kx) \rightarrow A_k, \quad (\varphi(x), \sin kx) \rightarrow B_k$  a využijeme ortogonality.

Z těchto vzorců vychází **algoritmus diskrétní Fourierovy analýzy**.

Úloha a řešení Fourierovy analýzy lze formulovat elegantně použitím **komplexní proměnné**.

Uvažujme pro jednoduchost lichý počet bazových funkcí ( $N = 2L + 1$ ) a periodu dané funkce  $2\pi$ . Potom má aproximující funkce tvar

$$\varphi(x) = A_0 + \sum_{k=1}^L (A_k \cos kx + B_k \sin kx). \quad (*)$$

Pomocí Eulerova vzorce

$$e^{ix} = \cos x + i \sin x$$

lze pro funkce  $\sin x$  a  $\cos x$  odvodit vztahy

$$\cos x = \frac{e^{ix} + e^{-ix}}{2}, \quad \sin x = \frac{e^{ix} - e^{-ix}}{2i} = -\frac{1}{2} i (e^{ix} - e^{-ix})$$

a tedy

$$\begin{aligned} \varphi(x) &= A_0 + \sum_{k=1}^L \left( \frac{1}{2} A_k (e^{ikx} + e^{-ikx}) - \frac{1}{2} i B_k (e^{ikx} - e^{-ikx}) \right) = \\ &= A_0 + \sum_{k=1}^L \left( \frac{1}{2} (A_k - i B_k) e^{ikx} + \frac{1}{2} (A_k + i B_k) e^{-ikx} \right). \end{aligned}$$

Označíme-li

$$C_0 = A_0, \quad C_k = \frac{1}{2} (A_k - i B_k), \quad C_{-k} = \frac{1}{2} (A_k + i B_k)$$

dostaneme

$$\varphi(x) = \sum_{k=-L}^L C_k e^{ikx}$$

Pro koeficienty dostaneme vynásobením (\*) jednotlivými bazovými funkcemi, využitím jejich ortogonality a interpolačních podmínek předpisy:

$$A_0 = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j)$$

$$A_k = \frac{2}{N} \sum_{j=0}^{N-1} f(x_j) \cos kx_j$$

$$B_k = \frac{2}{N} \sum_{j=0}^{N-1} f(x_j) \sin kx_j$$

$$\begin{aligned} C_{\pm k} &= \frac{1}{2} (A_k \mp i B_k) = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) \cos kx_j \mp i \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) \sin kx_j = \\ &= \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) \underbrace{(\cos kx_j \mp i \sin kx_j)}_{e^{\mp ikx_j}} \end{aligned}$$

$$C_k = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) e^{-ikx_j}, \quad k = -L, \dots, L$$

**Poznámka**

Vezmeme-li aproximující polynom o menším počtu bazových funkcí než je počet zadaných bodů, jedná se o aproximaci ve smyslu metody nejmenších čtverců, tj. diskrétní  $L_2$ -aproximaci. Potom obecně nemohou být splněny interpolační podmínky přesně (jen ve speciálních případech).

Výpočet koeficientů  $C_k$  představuje sčítání konečné řady.

$$C_k = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) e^{-ikx_j}, \quad k = -L, \dots, L$$

Uvažujeme-li počet aproximujících bazových funkcí  $N$  jako mocninu čísla 2 (tj.  $N = 2^M$ ), lze odvodit velmi rychlý a efektivní algoritmus pro výpočet koeficientů  $C_k$ .

Tento algoritmus se potom nazývá **rychlá Fourierova transformace (Fast Fourier transform - FFT)**.

Princip metody si ukážeme na následujícím příkladě.

**Příklad**

Uvažujme následující zadání funkce  $f$  pro  $N = 2^2 = 4$  ekvidistantní uzlové body.

$x_i$	$x_0 = 0$	$x_1 = \frac{\pi}{2}$	$x_2 = \pi$	$x_3 = \frac{3\pi}{2}$
$f(x_i)$	$f_0$	$f_1$	$f_2$	$f_3$

Počítáme koeficienty  $C_k$ .

Platí:

$$C_k = \frac{1}{4} (f_0 + f_1 e^{-i2k\frac{\pi}{2}} + f_2 e^{-i4k\frac{\pi}{2}} + f_3 e^{-i6k\frac{\pi}{2}}), \quad k = 0, 1, 2, 3$$

Označme

$$w = e^{-i\frac{\pi}{2}}, \quad F_k = \frac{1}{4} f_k \quad k = 0, 1, 2, 3$$

Potom

$$C_k = F_0 + F_1 w^k + F_2 w^{2k} + F_3 w^{3k}, \quad k = 0, 1, 2, 3$$

Uvědomme si, že platí

$$w^4 = 1 \quad (\text{obecně } w^N = 1).$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

$$C_0 = \frac{(F_0 + F_2) + (F_1 + F_3)}{S_0}$$

$$C_1 = \frac{(F_0 + w^2 F_2) + w(F_1 + w^2 F_3)}{S_1}$$

$$C_2 = (F_0 + F_2) + w^2(F_1 + F_3)$$

$$C_3 = (F_0 + w^2 F_2) + w^3(F_1 + w^2 F_3)$$

Výpočetní náročnost:

na  $R_k, S_k \dots 4S + 2N$

na  $C_k \dots 4S + 3N$

$\Sigma \dots 8S + 5N$

**Příklad**

Aproximujte periodickou funkci  $f$  (perioda  $T = 31$ ) zadanou tabulkou pomocí trigonometrického polynomu (ve smyslu  $L_2$ -aproximace, až při použití plného počtu bázeových funkcí ve smyslu interpolace).

$x_k = 1, 2, \dots, 32$	
$f(x_k) = 1$	pro $k = 1, 2, \dots, 15$
$= 0$	pro $k = 16$
$= -1$	pro $k = 17, 18, \dots, 31$
$= 1$	pro $k = 32$

**Řešení**

výsledky v MATLABu

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

koeficient A(0) = 0.000000	u bazove funkce phi(0) = 1
koeficient A(1) = 0.128701	u bazove funkce phi(1) = cos(2*pi*1*x/31-1)
koeficient B(1) = 1.265623	u bazove funkce phi(2) = sin(2*pi*1*x/31-1)
koeficient A(2) = 0.001321	u bazove funkce phi(3) = cos(2*pi*2*x/31-1)
koeficient B(2) = 0.006426	u bazove funkce phi(4) = sin(2*pi*2*x/31-1)
koeficient A(3) = 0.126074	u bazove funkce phi(5) = cos(2*pi*3*x/31-1)
koeficient B(3) = 0.401825	u bazove funkce phi(6) = sin(2*pi*3*x/31-1)
koeficient A(4) = 0.005229	u bazove funkce phi(7) = cos(2*pi*4*x/31-1)
koeficient B(4) = 0.012184	u bazove funkce phi(8) = sin(2*pi*4*x/31-1)
koeficient A(5) = 0.120926	u bazove funkce phi(9) = cos(2*pi*5*x/31-1)
koeficient B(5) = 0.217866	u bazove funkce phi(10) = sin(2*pi*5*x/31-1)
koeficient A(6) = 0.011564	u bazove funkce phi(11) = cos(2*pi*6*x/31-1)
koeficient B(6) = 0.016614	u bazove funkce phi(12) = sin(2*pi*6*x/31-1)
koeficient A(7) = 0.113468	u bazove funkce phi(13) = cos(2*pi*7*x/31-1)
koeficient B(7) = 0.132175	u bazove funkce phi(14) = sin(2*pi*7*x/31-1)
koeficient A(8) = 0.020067	u bazove funkce phi(15) = cos(2*pi*8*x/31-1)
koeficient B(8) = 0.019075	u bazove funkce phi(16) = sin(2*pi*8*x/31-1)
koeficient A(9) = 0.104007	u bazove funkce phi(17) = cos(2*pi*9*x/31-1)
koeficient B(9) = 0.080507	u bazove funkce phi(18) = sin(2*pi*9*x/31-1)
koeficient A(10) = 0.030389	u bazove funkce phi(19) = cos(2*pi*10*x/31-1)
koeficient B(10) = 0.018942	u bazove funkce phi(20) = sin(2*pi*10*x/31-1)
koeficient A(11) = 0.092929	u bazove funkce phi(21) = cos(2*pi*11*x/31-1)
koeficient B(11) = 0.045584	u bazove funkce phi(22) = sin(2*pi*11*x/31-1)
koeficient A(12) = 0.042109	u bazove funkce phi(23) = cos(2*pi*12*x/31-1)
koeficient B(12) = 0.015596	u bazove funkce phi(24) = sin(2*pi*12*x/31-1)
koeficient A(13) = 0.080687	u bazove funkce phi(25) = cos(2*pi*13*x/31-1)
koeficient B(13) = 0.020891	u bazove funkce phi(26) = sin(2*pi*13*x/31-1)
koeficient A(14) = 0.054747	u bazove funkce phi(27) = cos(2*pi*14*x/31-1)
koeficient B(14) = 0.008387	u bazove funkce phi(28) = sin(2*pi*14*x/31-1)
koeficient A(15) = 0.067784	u bazove funkce phi(29) = cos(2*pi*15*x/31-1)
koeficient B(15) = 0.003438	u bazove funkce phi(30) = sin(2*pi*15*x/31-1)

Aproximace je dana predpisem :

$$\phi = A(0) + \sum_{k=1}^L [ A(k)*\phi(2k-1) + B(k)*\phi(2k) ]$$

pro pocet bazovych funkcí N=2L+1

$$\phi = A(0) + \sum_{k=1}^{L-1} [ A(k)*\phi(2k-1) + B(k)*\phi(2k) ] + A(L)*\phi(2L-1)$$

pro pocet bazovych funkcí N=2L

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

## Kapitola 9. Numerické derivování

**Definice:** Existuje-li pro danou funkci  $f: \mathbb{R} \rightarrow \mathbb{R}$  vlastní (tj. konečná) limita

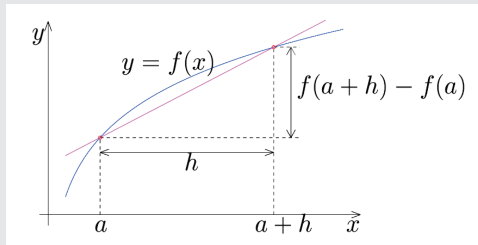
$$\lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$

říkáme, že funkce  $f(x)$  má v bodě  $a$  derivaci. Příslušnou limitu značíme  $f'(a)$ .

**Poznámka:**

Geometrický význam derivace  $f'(a)$  je směrnice tečny křivky dané rovnicí  $y = f(x)$  v bodě  $a$  (neboť tečna v bodě  $a$  je limitní polohou sečny pro  $h \rightarrow 0$ ).

Fyzikálně značí derivace funkce  $y = f(x)$ , kde  $x$  je čas a  $y$  dráha pohybu, limitu z průměrné rychlosti, tedy okamžitou rychlost v čase  $a$ .



**Poznámka:**

Pro danou funkci  $f(x)$  vyjadřuje derivace  $f'(x_0)$  míru „stoupání“, resp. „klesání“ v bodě  $x_0$ .

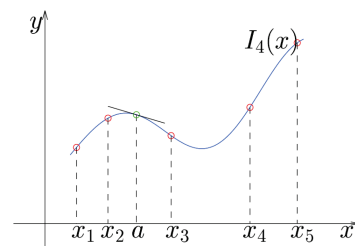
Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

**Způsoby odvození vzorců pro výpočet derivace**

1. Odvození pomocí interpolačního polynomu

Pro funkci  $f$ , která je zadána tabulkou, sestojíme interpolační polynom a derivaci funkce  $f$  v bodě  $a$  ztotožníme s derivací tohoto interpolačního polynomu v bodě  $a$ .



$$f'(a) \approx I'_n(a)$$

$$f^{(k)}(a) \approx I_n^{(k)}(a)$$

**Poznámky:**

- Stupeň polynomu nemůže být nižší než řád počítané derivace.
- Pro jednoduchost hledáme hodnotu derivace v uzlovém bodě a navíc uvažujeme ekvidistantní uzly s krokem  $h$ .

2. Odvození pomocí Taylorova rozvoje

Pro dostatečně hladkou funkci  $f$  platí (pro  $h > 0$ ):



$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(\xi_1), \quad \xi_1 \in (x_0, x_0 + h)$$

$$f(x_0 - h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(\xi_2), \quad \xi_2 \in (x_0 - h, x_0)$$

Z první rovnice potom plyne vztah

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{1}{2}hf''(\xi_1)$$

$$= D_P f(x_0, h)$$

Podobně ze druhé rovnice

$$f'(x_0) = \frac{f(x_0) - f(x_0 - h)}{h} + \frac{1}{2}hf''(\xi_2)$$

$$= D_L f(x_0, h)$$

Obdrželi jsme dva základní **dvoubodové** vzorce  $D_P f(x_0, h)$  a  $D_L f(x_0, h)$ , tzv. pravou a levou poměrnou diferencí.

Podobně odvodíme další vzorce pomocí Taylorova rozvoje vyšších řádů. Platí:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(\xi_1), \quad \xi_1 \in (x_0, x_0 + h)$$

$$f(x_0 - h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) - \frac{h^3}{6}f'''(\xi_2), \quad \xi_2 \in (x_0 - h, x_0)$$

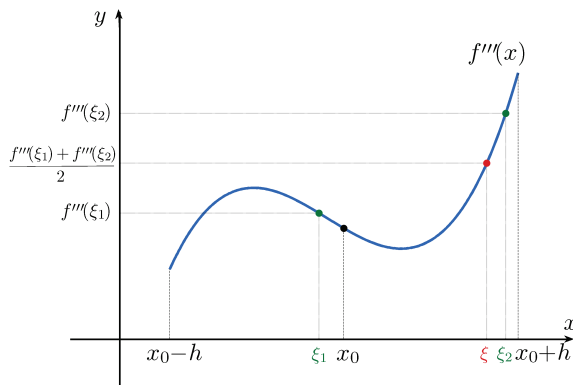
Po odečtení obdržíme:

$$f(x_0 + h) - f(x_0 - h) = 2hf'(x_0) + \frac{h^3}{6}(f'''(\xi_1) + f'''(\xi_2))$$

Odtud vyjádříme první derivaci a získáme **tříbodový** vzorec  $D_C f(x_0, h)$ , tzv. centrální poměrnou diferencí

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} - \frac{h^2}{12}(f'''(\xi_1) + f'''(\xi_2))$$

$$= \frac{D_C f(x_0, h)}{\frac{h^2}{6}f'''(\xi)}$$



Uvedené vzorce jsou pro výpočet první derivace  $f'(x_0)$ .

Pro výpočet druhé derivace  $f''(x_0)$  lze použít například vzorec, který dostaneme po sečtení vztahů:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(x_0) + \frac{h^4}{24}f^{(4)}(\xi_1), \quad \xi_1 \in (x_0, x_0 + h)$$

$$f(x_0 - h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) - \frac{h^3}{6}f'''(x_0) + \frac{h^4}{24}f^{(4)}(\xi_2), \quad \xi_2 \in (x_0 - h, x_0)$$

$$f(x_0 + h) + f(x_0 - h) = 2f(x_0) + h^2 f''(x_0) + \frac{h^4}{24}(f^{(4)}(\xi_1) + f^{(4)}(\xi_2))$$

Odtud vyjádříme druhou derivaci a získáme **tříbodový** vzorec pro druhou derivaci

$$f''(x_0) = \frac{f(x_0 + h) - 2f(x_0) + f(x_0 - h))}{h^2} - \frac{h^2}{24}(f^{(4)}(\xi_1) + f^{(4)}(\xi_2))$$

$$= \frac{D_C^2 f(x_0, h)}{\frac{h^2}{12}f^{(4)}(\xi)}$$

**Poznámka:**

Samozřejmě lze odvodit řadu dalších vzorců, přičemž platí, že čím více bodů použijeme, tím bude řád chyby vyšší.

**Příklad:** Pomocí uvedených tří vzorců vypočítejte přibližnou hodnotu první derivace funkce  $f(x) = e^x(1-x)$  v bodě  $x_0 = 1$ . Použijte krok  $h = 0,1$ .

**Řešení:**

Nejprve si pro kontrolu analyticky zjistíme přesnou hodnotu první derivace funkce  $f$  bodě  $x_0$ .

$$f'(x) = e^x(1-x) + e^x(-1) = -xe^x, \quad \text{tj. } f'(1) = -1e^1 = -e \approx -2,7182$$

Nyní použijeme pravou, levou a centrální poměrnou diferencí:

$$1. \quad D_P f(x_0, h) = \frac{f(x_0 + h) - f(x_0)}{h} = \frac{e^{1,1}(1-1,1) - e^1(1-1)}{0,1} = \frac{-0,1e^{1,1}}{0,1} = -e^{1,1} \approx -3,0041 \quad (\text{chyba } 0,2858)$$

$$2. \quad D_L f(x_0, h) = \frac{f(x_0) - f(x_0 - h)}{h} = \frac{e^1(1-1) - e^{0,9}(1-0,9)}{0,1} = \frac{-0,1e^{0,9}}{0,1} = -e^{0,9} \approx -2,4596 \quad (\text{chyba } 0,2586)$$

$$3. \quad D_C f(x_0, h) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} = \frac{e^{1,1}(1-1,1) - e^{0,9}(1-0,9)}{0,2} = \frac{-0,1e^{1,1} - 0,1e^{0,9}}{0,2} = -\frac{e^{1,1} + e^{0,9}}{2} \approx -2,7318 \quad (\text{chyba } 0,0136)$$

Všimněme si velikosti chyb v jednotlivých případech. Potvrzuje se fakt, že chyba prvních dvou (dvoubodových) vzorců je řádu  $h$ , tj. v řádu desetin a chyba posledního (tříbodového) vzorce je řádu  $h^2$ , tj. v řádu setin.

**Podmíněnost úlohy numerického derivování**

Uvažujme nyní např. vzorec s pravou diferencí  $D_P f(x_0, h)$ , tj. platí

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{1}{2}hf''(\xi)$$

$$= D_P f(x_0, h) + \underbrace{\frac{1}{2}hf''(\xi)}_{\text{chyba metody}}$$

Chybu metody označme  $r_1$ .

Platí-li  $|f''(x)| < M$  pro  $x \in (x_0, x_0 + h)$ , potom  $|r_1| \leq \frac{M}{2}h$ .

Musíme uvážit **chyby měření** (zaokrouhlovací chyby) - označme  $r_2$ .

Označíme-li

$f(x_0), f(x_0 + h)$  ..... přesné hodnoty  
 $f^*(x_0), f^*(x_0 + h)$  ..... vstupní hodnoty

Potom pro  $r_2$  platí

$$r_2 = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{f^*(x_0 + h) - f^*(x_0)}{h}$$

přesná hodnota vzorce      vypočtená hodnota vzorce

A dále

$$|r_2| = \left| \frac{f(x_0 + h) - f^*(x_0 + h)}{h} + \frac{f^*(x_0) - f(x_0)}{h} \right| \leq$$

$$\leq \frac{|f(x_0 + h) - f^*(x_0 + h)|}{h} + \frac{|f^*(x_0) - f(x_0)|}{h} \leq$$

$$\leq \frac{\epsilon}{h} + \frac{\epsilon}{h} = \frac{2\epsilon}{h}$$

Využili jsme zde odhady

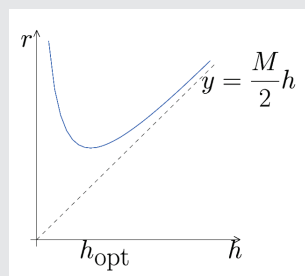
$$|f^*(x_0 + h) - f(x_0 + h)| \leq \epsilon$$

$$|f^*(x_0) - f(x_0)| \leq \epsilon$$

Číslo  $\epsilon$  může představovat např. strojovou přesnost.

Pro celkovou chybu  $r$  potom platí

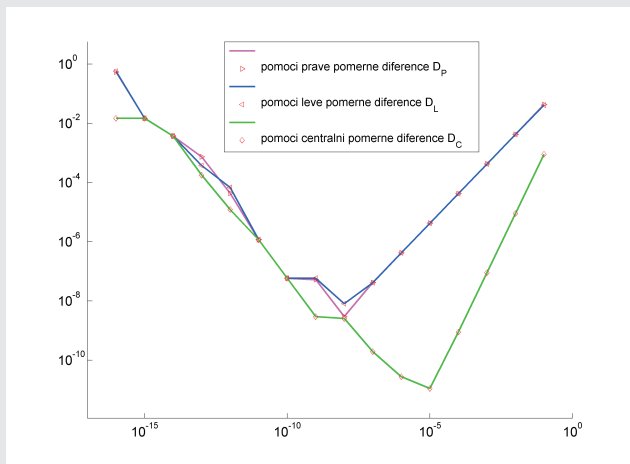
$$|r| \leq |r_1| + |r_2| \leq \frac{M}{2}h + \frac{2\epsilon}{h}$$



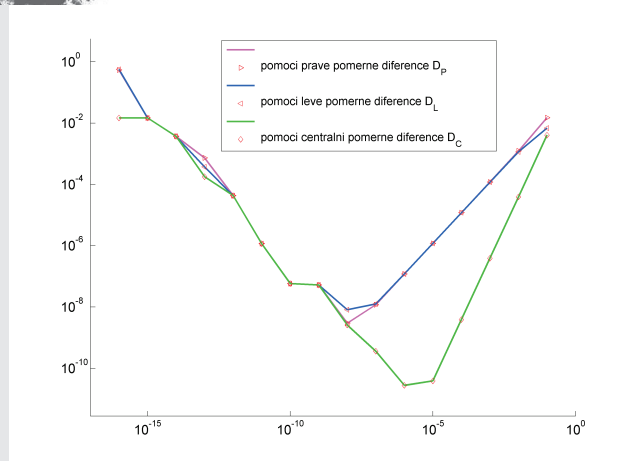
- Úloha numerického derivování je **špatně podmíněná** ! (pro zmenšující se  $h$  roste chyba)
- Lze najít optimální krok  $h_{opt}$

V následujících příkladech ukažte chybu 3 odvozených vzorců pro výpočet první derivace funkce  $f(x)$  v bodě  $x_0$  při použití kroků  $h = 10^{-16}, 10^{-15}, \dots, 10^{-1}$ .

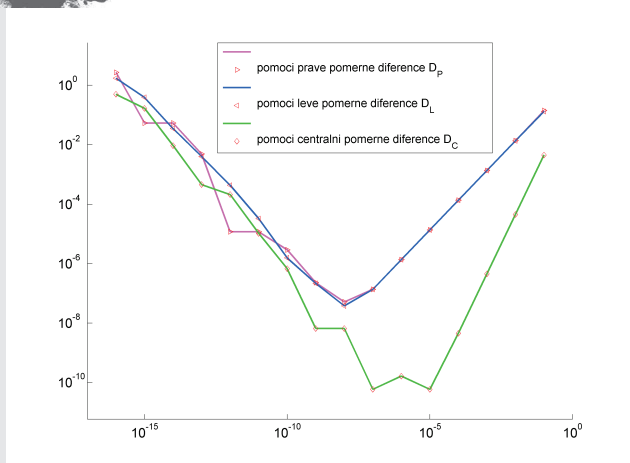
**Příklad 1**  $f(x) = \sin x, x_0 = 1$ .



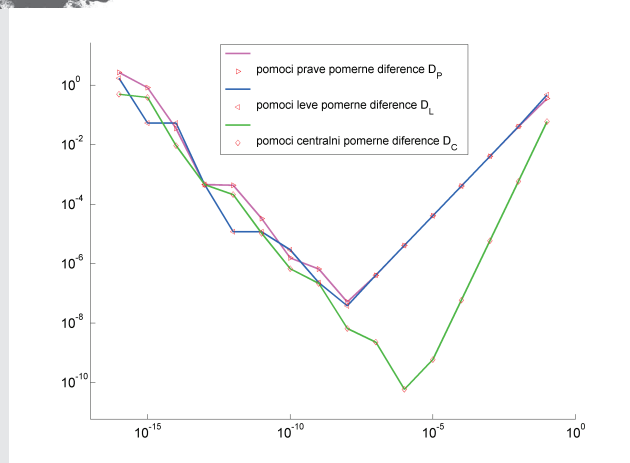
**Příklad 2**  $f(x) = \sin \frac{1}{x}, x_0 = 1$ .



**Příklad 3**  $f(x) = e^x, x_0 = 1$ .



**Příklad 4**  $f(x) = e^{\frac{1}{2}}, x_0 = 1$ .



**Poznámka:**

Na základě špatné podmíněnosti se zdá, že nebude možné při výpočtu derivace dosáhnout libovolné přesnosti.

Zvýšení přesnosti ale můžeme dosáhnout

- 1) použitím vzorce s chybou vyššího řádu
- 2) použitím tzv. Richardsonovy extrapolace

**Richardsonova extrapolace**

Jde o obecný princip, který se používá nejen u numerického derivování.

Myšlenka vychází z toho, že na základě znalosti výrazu pro rozvoj chyby využijeme dvou přibližných výsledků k získání třetího, který bude přesnější.

Tento proces eliminace chyb budeme demonstrovat např. na poměrně centrální diferenci  $D_C f(x_0, h)$ .

Vyjdeme z Taylorova rozvoje:

$$(1) \quad f(x_0+h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(x_0) + \frac{h^4}{24}f^{(4)}(x_0) + \frac{h^5}{120}f^{(5)}(\xi_1)$$

$$(2) \quad f(x_0-h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) - \frac{h^3}{6}f'''(x_0) + \frac{h^4}{24}f^{(4)}(x_0) - \frac{h^5}{120}f^{(5)}(\xi_2)$$





$$\frac{1}{h^2} \left[ \left( 1 - \frac{\alpha_0 h}{\beta_0} \right) U_0 - U_1 \right] = \frac{g_0}{h \beta_0}$$

tuto úpravu děláme proto, aby výsledná matice soustavy sítových rovnic byla symetrická.

Předchozí rovnici přidáme k získaným diferenciálním rovnicím a dosadíme  $g_1$  za  $U_N$ . Dostaneme opět soustavu lineárních algebraických rovnic se symetrickou třídiagonální maticí (pozor:  $U = [U_0, U_1, \dots, U_{N-1}]^T$ ).

$$\frac{1}{h^2} \begin{bmatrix} 1 + \frac{\alpha_0 h}{\beta_0} & -1 & 0 & \dots & 0 \\ -1 & 2 + h^2 q_1 & -1 & 0 & \dots & 0 \\ 0 & -1 & 2 + h^2 q_2 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & 0 & -1 & 2 + h^2 q_{N-2} & -1 \\ 0 & 0 & \dots & 0 & -1 & 2 + h^2 q_{N-1} \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ \vdots \\ \vdots \\ U_{N-1} \end{bmatrix} = \begin{bmatrix} \frac{g_0}{h \beta_0} \\ f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_{N-2} \\ f_{N-1} - \frac{g_1}{\beta_1} \end{bmatrix}$$

Matice soustavy je řádu  $N$  a lze dokázat, že je regulární. Přibližné řešení lze proto jednoznačně stanovit a dá se ukázat, že jeho **chyba je velikosti  $O(h)$** .

Tato skutečnost, tj. snížení řádu chyby metody, možná překvapí, neboť sítová rovnice pro všechny uzly s výjimkou  $x_0$  aproximuje diferenciální rovnici s diskretizační chybou  $O(h^2)$ . Přesto však okolnost, že jsme se v jediné rovnici dopustili diskretizační chyby velikosti  $O(h)$ , ovlivní velikost chyby metody nejen v blízkosti bodu  $x_0$ , ale ve všech bodech sítě. Jde tu o jev, který je pro užití diferenciální metody typický a ukazuje, že chceme-li využít přesnosti, s níž jsme aproximovali diferenciální rovnici samotnou, musíme stejně přesně aproximovat i okrajové podmínky.

### 2.způsob

Bezprostřední možnost přesnější náhrady hodnoty derivace v okrajové podmínce spočívá v užití vzorců 4 a 5 z tabulky. Tyto aproximace mají pro  $u \in C^2$  diskretizační chybu  $O(h^2)$ .

Vzniklá soustava diferenciálních rovnic však již není třídiagonální, není symetrická a navíc se při jejím řešení standardními metodami mohou vyskytnout numerické problémy. Proto tento postup nedoporučujeme.

### 3.způsob

Hodnotu  $u'(0)$  aproximujeme pomocí vzorce 3 (centrální poměrná diference).

$$u'(0) \approx \frac{1}{2h} [u(x_1) - u(x_{-1})],$$

kde  $x_{-1} = -h$ .

Chyba aproximace je druhého řádu. Okrajovou podmínku  $\alpha_0 u(0) - \beta_0 u'(0) = g_0$  tak nahradíme diferenciální rovnicí

$$\alpha_0 U_0 - \beta_0 \frac{U_1 - U_{-1}}{2h} = g_0 \quad (\heartsuit)$$

Je zde však navíc další neznámá  $U_{-1}$  a potřebujeme proto připojit ještě jednu rovnici.

Nejjednodušší to provedeme tak, že žádáme platnost rovnice pro vnitřní uzly i pro hraniční uzel  $x_0$ , tj. platnost rovnice

$$-\frac{1}{h^2} (U_{-1} - 2U_0 + U_1) + q(x_0)U_0 = f(x_0). \quad (\diamondsuit)$$

(Jde vlastně o aproximaci diferenciální rovnice v bodě  $x_0 = 0$ ).

Fiktivní hodnotu  $U_{-1}$ , která nemá význam aproximace přesného řešení naší okrajové úlohy (neboť toto řešení uvažujeme pouze na intervalu  $(0, 1)$ ), z rovnic  $(\heartsuit)$ ,  $(\diamondsuit)$  vyloučíme

$$U_{-1} = h^2 (q_0 U_0 - f_0) + 2U_0 - U_1,$$

## Kapitola 10. Numerické integrování

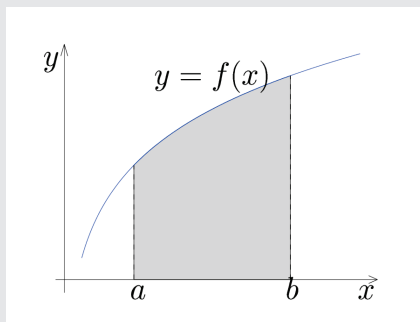
### Numerický výpočet hodnoty určitého integrálu

**Formule:** Mějme na  $(a, b)$  dānu integrovatelnou funkci  $f = f(x)$ . Naším cílem je určit přibližnou hodnotu určitého integrálu

$$I(f) = \int_a^b f(x) dx.$$

### Poznámka:

Geometrický význam integrálu  $I(f)$  (viz obrázek) je obsah plochy mezi grafem funkce  $f$  a osou  $x$  na intervalu  $(a, b)$ .



Numerické metody výpočtu integrálu užíváme zejména tehdy, když  $I(f)$  není možno spočítat analyticky (velmi častý případ) nebo je sice analyticky řešení možné, ale je velmi pracné. V případě, že máme zadānu funkci  $f$  tabulkou, není ani jiný přístup možný.

Přirozený princip numerických metod pro výpočet integrálu vychází z aproximace funkce. Danou funkci  $f$  nahradíme její vhodnou aproximací  $\varphi$  a jako aproximaci integrálu  $I(f)$  prohlásíme hodnotu integrálu  $I(\varphi)$ , tj.

$$I(f) \approx I(\varphi) = \int_a^b \varphi(x) dx.$$

### Poznámka:

Narozdíl od výpočtu derivace je výpočet integrálu stabilní, protože je-li  $\varphi$  dobrou aproximací funkce  $f$  na intervalu  $(a, b)$ , je integrál  $I(\varphi)$  dobrou aproximací  $I(f)$ .

dosadíme

$$\alpha_0 U_0 - \beta_0 \frac{U_1 - h^2 (q_0 U_0 - f_0) - 2U_0 + U_1}{2h} = g_0$$

a dospějeme k diferenciální rovnici

$$\left( \alpha_0 + \frac{1}{2} \beta_0 h q_0 \right) U_0 - \beta_0 \frac{U_1 - U_0}{h} = g_0 + \frac{1}{2} \beta_0 h f_0,$$

která aproximuje okrajovou podmínku v bodě  $x_0 = 0$  a kterou dostatečně hladké přesné řešení naší okrajové úlohy splňuje s chybou řádové velikosti  $O(h^2)$ . Rovnici upravíme na tvar

$$\frac{1}{h^2} \left[ \left( 1 + \frac{\alpha_0 h}{\beta_0} + \frac{1}{2} h^2 q_0 \right) U_0 - U_1 \right] = \frac{g_0}{h \beta_0} + \frac{1}{2} f_0$$

a připojíme ji k diferenciálním rovnicím pro vnitřní uzly.

Dá se ukázat, že **chyba přibližného řešení je** v tomto případě velikosti  $O(h^2)$ .

$$\left| \int_a^b f(x) dx - \int_a^b \varphi(x) dx \right| \leq \int_a^b |f(x) - \varphi(x)| dx \leq (b-a) \sup_{x \in (a,b)} |f(x) - \varphi(x)| \leq \varepsilon$$

**Princip** většiny metod na výpočet určitého integrálu

$$\int_a^b f(x) dx$$

je založen na tom, že interval  $(a, b)$  rozdělíme na  $N$  podintervalů  $\langle x_k, x_{k+1} \rangle$  tak, že

$$a = x_0 < x_1 < x_2 < \dots < x_{N-1} < x_N = b.$$

Na těchto podintervalech nahradíme funkci  $f$  polynomem a integrujeme tento polynom.

Vzorec pro výpočet určitého integrálu (tzv. **kvadraturní vzorec**) dělíme na:

na intervalech  $\langle x_k, x_{k+1} \rangle$  ... **základní**  
přes celý interval  $(a, b)$  ... **složený** (složený kv. vzorec je součtem základních kv. vzorců)

Pro jednoduchost předpokládáme, že jsou všechny podintervaly  $\langle x_k, x_{k+1} \rangle$  stejně velké.

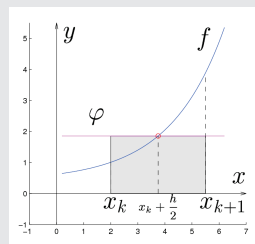
Ekvidistantní uzly potom vyjádříme takto

$$x_k = a + kh, \quad \text{kde } k = 0, 1, \dots, N-1 \quad \text{a } h = \frac{b-a}{N}.$$

### Newtonovy-Cotesovy základní kvadraturní vzorec

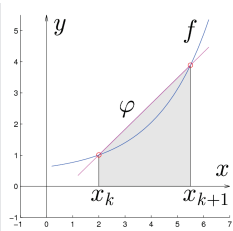
1) **Obdélníkové pravidlo** ( $f$  nahrazujeme konstantní funkcí  $\varphi$ )

$$\int_{x_k}^{x_{k+1}} f(x) dx \approx \underbrace{h \cdot f\left(x_k + \frac{h}{2}\right)}_{\equiv R_{\mathcal{O}}(f, h)}$$



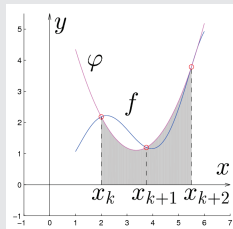
2) **Lichoběžníkové pravidlo** ( $f$  nahrazujeme lineární funkcí  $\varphi$ )

$$\int_{x_k}^{x_{k+1}} f(x) dx \approx \frac{h}{2} [f(x_k) + f(x_{k+1})] \equiv T_2(f, h)$$



3) Simpsonovo pravidlo (f nahrazujeme kvadratickou funkcí φ)

$$\int_{x_k}^{x_{k+2}} f(x) dx \approx \frac{h}{3} [f(x_k) + 4f(x_{k+1}) + f(x_{k+2})] \equiv S_2(f, h)$$



Odvození Simpsonova pravidla

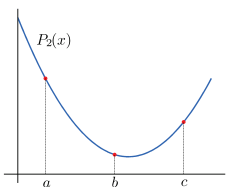
Např. pomocí Lagrangeova interpolačního polynomu:

$$P_2(x) = f(a)l_a(x) + f(b)l_b(x) + f(c)l_c(x)$$

$$l_a(x) = \frac{(x-b)(x-c)}{(a-b)(a-c)} = \frac{(x-b)(x-c)}{2h^2}$$

$$l_b(x) = \frac{(x-a)(x-c)}{(b-a)(b-c)} = \frac{(x-a)(x-c)}{-h^2}$$

$$l_c(x) = \frac{(x-a)(x-b)}{(c-a)(c-b)} = \frac{(x-a)(x-b)}{2h^2}$$



$$\begin{aligned} \int_a^c P_2(x) dx &= \frac{f(a)}{2h^2} \int_a^c (x-b)(x-c) dx - \frac{f(b)}{h^2} \int_a^c (x-a)(x-c) dx + \frac{f(c)}{2h^2} \int_a^c (x-a)(x-b) dx = \\ &= \frac{f(a)}{2h^2} \left[ \frac{x^3}{3} - \frac{x^2}{2}(b+c) + xbc \right]_a^c - \frac{f(b)}{h^2} \left[ \frac{x^3}{3} - \frac{x^2}{2}(a+c) + xac \right]_a^c + \frac{f(c)}{2h^2} \left[ \frac{x^3}{3} - \frac{x^2}{2}(a+b) + xab \right]_a^c = \\ &= \frac{f(a)}{2h^2} \left[ \frac{c^3}{3} - \frac{a^3}{3} - \left( \frac{c^2}{2} - \frac{a^2}{2} \right) (b+c) + (c-a)bc \right] - \frac{f(b)}{h^2} \left[ \frac{c^3}{3} - \frac{a^3}{3} - \left( \frac{c^2}{2} - \frac{a^2}{2} \right) (a+c) + (c-a)ac \right] + \\ &\quad + \frac{f(c)}{2h^2} \left[ \frac{c^3}{3} - \frac{a^3}{3} - \left( \frac{c^2}{2} - \frac{a^2}{2} \right) (a+b) + (c-a)ab \right] \end{aligned}$$

$$(*) \quad \frac{1}{6}(c-a)[2c^2 + ac + 2a^2 - 3(a+c)(b+c) + 6bc] =$$

$$= \frac{2h}{6} [2c^2 + 2ac + 2a^2 - 3(a+b)c - 3c^2 - 3ab + 6bc] =$$

$$= \frac{2h}{6} [-c^2 - ac + 3bc + 2a^2 - 3ab] =$$

$$= \frac{2h}{6} \left[ \frac{a^2 - c^2}{(a-c)(a+c)} + \frac{3b(c-a)}{2h} + \frac{a(a-c)}{-2h} \right] =$$

$$= \frac{2h}{6} [-(a+c)2h + 2h3b - 2ha] =$$

$$= -\frac{4h^2}{6} [a + c - 3b + a] =$$

$$= -\frac{4h^2}{6} \left( \frac{2a-2b+c-b}{-2h} \right) =$$

$$= \frac{4h^3}{6} = \frac{2}{3}h^3$$

$$(**) \quad \dots = \frac{1}{6}(2h)^3 \quad \text{viz pomocný výpočet pro odvození lichoběžníkového pravidla (slide 10.5.)}$$

$$(***) \quad \dots = \frac{2}{3}h^3 \quad \text{stejně jako (*) - plyne ze symetrie}$$

$$\int_a^c P_2(x) dx = \frac{f(a)}{2h^2} \frac{2}{3}h^3 + \frac{f(b)}{h^2} \frac{1}{6}(2h)^3 + \frac{f(c)}{2h^2} \frac{2}{3}h^3 = \frac{h}{3} [f(a) + 4f(b) + f(c)] = T_2(f, h)$$

Příklad:

Pomocí základních Newtonových-Cotesových vzorců vypočítejte integrál

$$\int_1^{1.2} e^{2x} dx.$$

Řešení:

(Přesné řešení je  $[e^{2x}]_1^{1.2} = e^{2.4} - e^2 \approx 0,601835$ .)

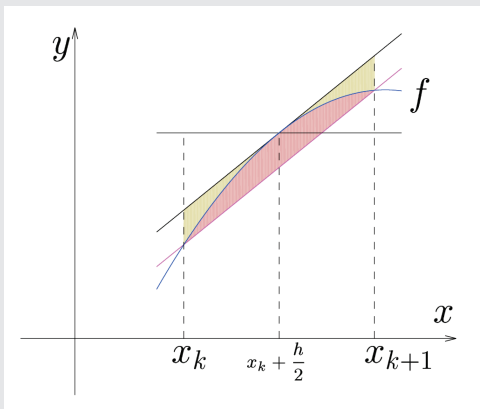
$$R_2(e^{2x}; 0, 2) = 0, 2e^{2.1} \approx 0,600833 \quad \text{chyba: } 0,001002$$

$$T_2(e^{2x}; 0, 2) = \frac{9}{25}(e^{4.0} + e^{4.2}) \approx 0,603839 \quad \text{chyba: } 0,002003$$

$$S_2(e^{2x}; 0, 1) = \frac{9}{8}(e + 4e^{1.1} + e^{1.2}) \approx 0,601835 \quad \text{chyba: } 0,000000$$

Poznámka:

Všimněme si chyb. U obdélníkového pravidla vyšla chyba menší než u lichoběžníkového, přestože u lichoběžníkového pravidla jsme funkci  $f$  aproximovali „lepší“ funkcí  $\varphi$  (lineární). Chyba u Simpsonova pravidla vyšla menší než u ostatních. Tyto výsledky potvrzují vztahy pro chyby jednotlivých vzorců. Fakt, že obdélníkové pravidlo je přesnější než lichoběžníkové můžeme demonstrovat na obrázku:



Základní vzorce se odvodí snadno na základě geometrické interpretace.

Pokud chceme vyjádřit současně i vztahy pro chyby těchto vzorců, musíme použít k odvození Taylorův rozvoj.

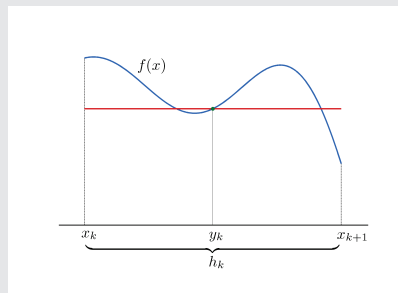
$$\int_{x_k}^{x_{k+1}} f(x) dx = R_2(f, h) + \frac{h^3}{24} f''(\xi)$$

$$\int_{x_k}^{x_{k+1}} f(x) dx = T_2(f, h) - \frac{h^3}{12} f''(\xi)$$

$$\int_{x_k}^{x_{k+2}} f(x) dx = S_2(f, h) - \frac{h^5}{90} f^{(4)}(\xi)$$

Odvození pro obdélníkové pravidlo

Předpokládáme, že je integrovaná funkce  $f$  dostatečně hladká a použijeme Taylorův polynom.



Označíme

$$h_k = x_{k+1} - x_k, \quad y_k = \frac{x_k + x_{k+1}}{2}$$

$$f(x) = f(y_k) + (x - y_k)f'(y_k) + \frac{1}{2}(x - y_k)^2 f''(\xi_k), \quad \xi_k \in \text{int}\{y_k, x\}$$

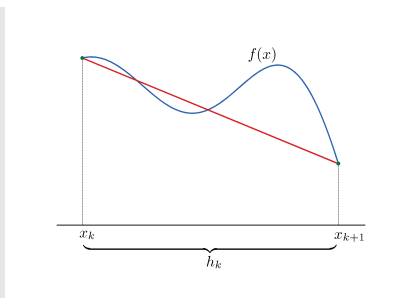
Potom platí:

$$\int_{x_k}^{x_{k+1}} f(x) dx = h_k f\left(\frac{x_k + x_{k+1}}{2}\right) + \frac{1}{2} \left[ \frac{h_k^3}{(x_{k+1} - y_k)^2} - \frac{h_k}{(x_k - y_k)^2} \right] f''\left(\frac{x_k + x_{k+1}}{2}\right) + f''(\xi_k) \frac{1}{6} \left[ \frac{h_k^3}{(x_{k+1} - y_k)^3} - \frac{h_k^3}{(x_k - y_k)^3} \right]$$

$$\int_{x_k}^{x_{k+1}} f(x) dx = h_k f\left(\frac{x_k + x_{k+1}}{2}\right) + \frac{h_k^3}{24} f''(\xi_k)$$

Odvození pro lichoběžníkové pravidlo

Funkci  $f$  aproximujeme na  $\langle x_k, x_{k+1} \rangle$  lineární funkcí, tj. interpolačním polynomem 1. stupně.



Z aproximací funkce známe:

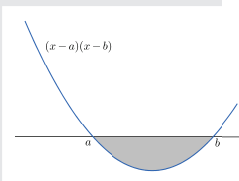
$$f(x) = P_2(x) + \frac{f''(\xi_k)}{2}(x-x_k)(x-x_{k+1}), \quad \xi_k \in (x_k, x_{k+1})$$

Potom platí:

$$\int_{x_k}^{x_{k+1}} f(x) dx = \frac{h_k}{2}(f(x_k) + f(x_{k+1})) + \frac{f''(\xi_k)}{2} \int_{x_k}^{x_{k+1}} (x-x_k)(x-x_{k+1}) dx$$

pomocný výpočet

$$\begin{aligned} \int_a^b (x-a)(x-b) dx &= \int_a^b (x^2 - x(a+b) + ab) dx = \\ &= \frac{b^3}{3} - \frac{a^3}{3} - \left(\frac{b^2}{2} - \frac{a^2}{2}\right)(a+b) + (b-a)ab = \\ &= \frac{1}{3}(b-a)(b^2 + ab + a^2) - \frac{1}{2}(b-a)(a+b)^2 + (b-a)ab = \\ &= \frac{1}{6}(b-a)[2b^2 + 2ab + 2a^2 - 3a^2 - 6ab - 3b^2 + 6ab] = \\ &= \frac{1}{6}(b-a)[-a^2 + 2ab - b^2] = -\frac{1}{6}(b-a)^3 \end{aligned}$$



$$\int_{x_k}^{x_{k+1}} f(x) dx = \frac{h_k}{2}(f(x_k) + f(x_{k+1})) - \frac{h_k^3}{12} f''(\xi_k)$$

chyba metody

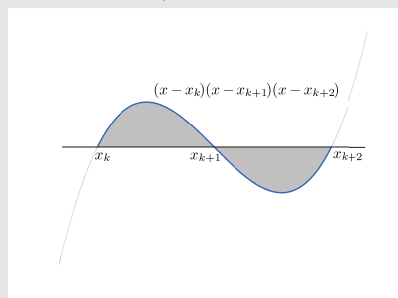
Komentář pro Simpsonovo pravidlo

Funkci  $f$  aproximujeme na  $\langle x_k, x_{k+2} \rangle$  kvadratickou funkcí, tj. interpolačním polynomem 2. stupně.

$$f(x) = P_2(x) + \frac{f''(\xi)}{6}(x-x_k)(x-x_{k+1})(x-x_{k+2})$$

$$\int_{x_k}^{x_{k+2}} f(x) dx = S_2(f, h) + \frac{f''(\xi)}{6} \int_{x_k}^{x_{k+2}} (x-x_k)(x-x_{k+1})(x-x_{k+2}) dx \dots$$

Ačkoliv z uvedeného vychází, že chyba by řádově měla být  $h^4$ , je chyba o jeden řád vyšší. Důvod je podobný jako u odvození chyby obdélníkového pravidla (integrujeme funkci symetrickou podle středu intervalu).



Jelikož výraz pro chybu základního Simpsonova pravidla obsahuje 4-tou derivaci, je zřejmé, že Simpsonovo pravidlo bude přesně integrovat polynomy až do stupně 3, protože pro ně je 4-tá derivace identicky nulová.

**Newton-Cotesovy složené kvadraturní vzorce**

Složené kvadraturní vzorce získáme sečtením základních kvadraturních vzorců:

$$\int_a^b f(x) dx = \sum_{k=0}^{N-1} \int_{x_k}^{x_{k+1}} f(x) dx \approx \sum_{k=0}^{N-1} \int_{x_k}^{x_{k+1}} \varphi(x) dx$$

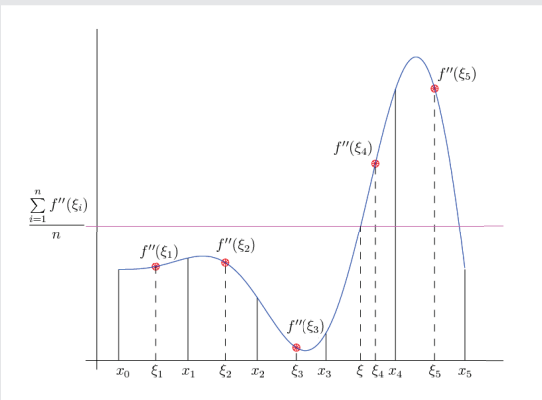
$$\begin{aligned} R(f, h) &\equiv h \cdot \sum_{k=0}^{N-1} f\left(x_k + \frac{h}{2}\right) \\ T(f, h) &\equiv \frac{h}{2} [f(x_0) + 2f(x_1) + 2f(x_2) + \dots + 2f(x_{N-1}) + f(x_N)] = \\ &= h \left[ \frac{1}{2} f(x_0) + \sum_{k=1}^{N-1} f(x_k) + \frac{1}{2} f(x_N) \right] \\ S(f, h) &\equiv \frac{h}{3} [f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \\ &\quad + \dots + 2f(x_{N-2}) + 4f(x_{N-1}) + f(x_N)] \end{aligned}$$

Pro chyby složených vzorců potom platí:

$$\begin{aligned} I(f) &= R(f, h) + (b-a) \frac{h^2}{24} f''(\xi) \\ I(f) &= T(f, h) - (b-a) \frac{h^2}{12} f''(\xi) \\ I(f) &= S(f, h) - (b-a) \frac{h^4}{180} f^{(4)}(\xi) \end{aligned}$$

$$\sum_{k=1}^N \frac{h^3}{24} f''(\xi_k) = \frac{h^3}{24} \sum_{k=1}^N f''(\xi_k) = \frac{h^3}{24} N f''(\xi) = \frac{h^3}{24} \frac{b-a}{h} f''(\xi)$$

chyba základního vzorce na  $\langle x_{k-1}, x_k \rangle$



Průměr hodnot leží mezi minimální a maximální hodnotou:

$$\min_k f''(\xi_k) \leq \frac{1}{N} \sum_k f''(\xi_k) \leq \max_k f''(\xi_k)$$

Ze spojitosti funkce  $f''(x)$  vyplývá:

$$\exists \xi \in (x_0, x_N): f''(\xi) = \frac{1}{N} \sum_k f''(\xi_k)$$

**Jak dosáhnout požadovanou přesnost ?**

Ze vzorců lze odhadnout velikost chyby, případně určit krok  $h$  tak, aby chyba byla menší než předem zadaná tolerance.

**Příklad** Určete  $h$  tak, aby chyba složeného lichoběžníkového pravidla pro výpočet

$$I = \int_2^3 \frac{1}{(x-1)^2} dx$$

byla nejvýše  $10^{-3}$ .

Musí platit:

$$\frac{(b-a)}{12} h^2 \max_{x \in (2,3)} |f''(x)| \leq 10^{-3}$$

$\Rightarrow$  je nutné odhadnout  $f''$ :

$$f' = -\frac{1}{(x-1)^2}$$

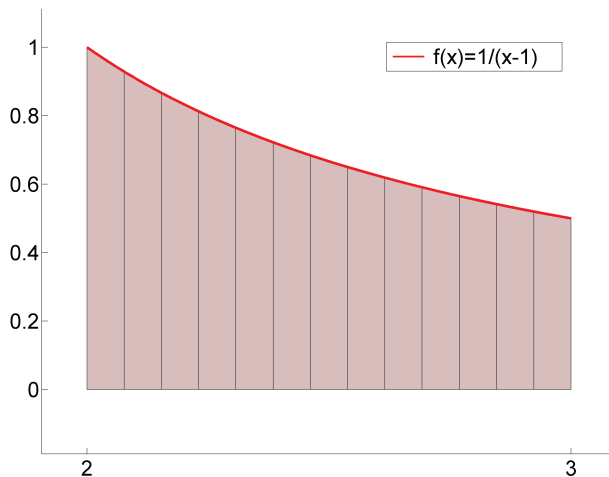
$$f'' = \frac{2}{(x-1)^3} \text{ na } (2,3) \text{ je } f'' > 0 \text{ (kladná)}$$

$$f''' = -\frac{3}{(x-1)^4} < 0 \Rightarrow f'' \text{ je klesající}$$

$$\Rightarrow \max_{x \in (2,3)} |f''(x)| = f''(2) = \frac{2}{(2-1)^3} = 2$$

$$\frac{1}{12} h^2 \cdot 2 \leq 10^{-3} \Rightarrow h^2 \leq 6 \cdot 10^{-3}$$

$$\Rightarrow N = \frac{b-a}{h} = \frac{1}{\sqrt{6 \cdot 10^{-3}}} \div 12,9 \Rightarrow \text{nejbližší vyšší } N=13 \Rightarrow h = \frac{1}{13}$$



$$T\left(\frac{1}{x-1}, \frac{1}{13}\right) = \dots \pm 0,69352$$

Přesná hodnota:  $I = [\ln|x-1|]_2^3 = \ln 2 - \ln 1 = \ln 2 \pm 0,69315$   
 Skutečná chyba:  $3,7 \cdot 10^{-4} \leq 10^{-3}$

**Nevhody tohoto postupu:**

- výrazy pro chybu obsahují derivace (často vysokého řádu), které není lehké odhadnout
- výsledné odhady jsou většinou velmi pesimistické
- Newton-Cotesovy vzorce nejsou konvergentní (zvýšujeme-li řád vzorce, nemusí konvergovat aproximace integrálů k teoretické hodnotě)
- pro odhad chyby je vhodné užít metodu polovičního kroku (Richardsonova extrapolace)

**Richardsonova extrapolace**

Stručně si připomeňme princip Richardsonovy extrapolace, kterou jsme již používali pro zpřesňování při výpočtu hodnoty derivace funkce.

Předpokládejme, že výraz pro chybu má tvar

$$e(f) = h^k M_1, \quad h = \frac{b-a}{N}$$

Přesná hodnota integrálu je potom

$$I = K(h) + h^k M_1 \quad (*)$$

Integrál vypočteme stejným vzorcem, ale s krokem  $\frac{h}{2}$ . Dostaneme

$$I = K\left(\frac{h}{2}\right) + \left(\frac{h}{2}\right)^k M_1 \Rightarrow h^k = \frac{\varepsilon 2^k}{M_1} \quad (**)$$

Dosadíme-li  $h^k$  do (\*), získáme

$$I = K(h) + \frac{\varepsilon 2^k M}{M_1} \quad (***)$$

Předpokládáme-li, že se hodnota derivace ve výrazu  $e(f)$  pro chybu příliš nemění (tj.  $M \approx M_1$ ), potom  $\frac{M}{M_1} \approx 1$  a pro (\*\*\*) a (\*\*\*) musí platit

$$K\left(\frac{h}{2}\right) + \varepsilon \approx K(h) + 2^k \varepsilon$$

Odtud plyne odhad chyby  $\varepsilon$

$$\varepsilon \approx \frac{1}{2^k - 1} \left[ K\left(\frac{h}{2}\right) - K(h) \right]$$

a přesnější hodnota integrálu je potom

$$I = K\left(\frac{h}{2}\right) + \frac{1}{2^k - 1} \left[ K\left(\frac{h}{2}\right) - K(h) \right]$$

k ... řád eliminované chyby

**Algoritmus** (Pro složené lichoběžníkové pravidlo)

Pro  $s = 0, 1, 2, \dots, S$   $h_0 = b - a$   
 $T_{s,0} = T(f, h_s)$   $h_1 = \frac{1}{2} h_0$   
 Pro  $k = 1, 2, \dots, s$   $\vdots$   
 $T_{s,k} = T_{s,k-1} + \frac{T_{s,k-1} - T_{s-1,k-1}}{4^k - 1}$   $h_s = \frac{1}{2^s} h_0$

**Schéma**

$h$	$T_{10}$		
$\frac{h}{2}$	$T_{10}$	$T_{11}$	
$\frac{h}{4}$	$T_{20}$	$T_{21}$	$T_{22}$
$\frac{h}{8}$	$T_{30}$	$T_{31}$	$T_{32}$ $T_{33}$

Všechny hodnoty  $T_{s,k}$  jsou aproximacemi původního integrálu.

Pro funkci  $f$  integrovatelnou v Riemannově smyslu platí

$$T_{s,k} \rightarrow I(f) \quad \text{pro } s \rightarrow \infty, k = 0, 1, \dots$$

a také

$$T_{1,k} \rightarrow I(f) \quad \text{pro } k \rightarrow \infty.$$

Dále se dá ukázat, že celá procedura je numericky stabilní.

Pro kontrolu uveďme přesnou hodnotu integrálu:

$$\int_1^5 \ln x \, dx = \left| u = \ln x \quad v' = 1 \right| = [x \ln x]_1^5 - \int_1^5 1 \, dx = 5 \ln 5 - 4 \approx 4,04719$$

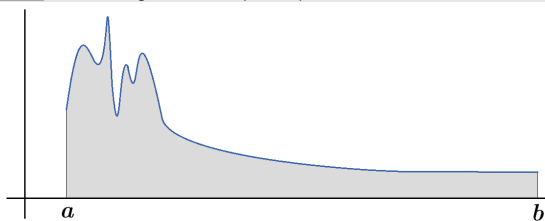
**Poznámka**

Metoda Richardsonovy extrapolace pro lichoběžníkové pravidlo se nazývá **Rombergova metoda**.

**Adaptivní integrování**

- intervaly integrace nejsou dány dopředu
- určují se na základě splnění testu chyby založeném na odhadu pomocí metody polovičního kroku

**Motivace:** Pokud má integrovaná funkce např. tento průběh



je zřejmé, že na druhé části intervalu stačí pro splnění zadané tolerance uvažovat větší kroky, než v první části.

**Stavy**

- S ... interval, na kterém je zajištěno splnění chybového testu
- A ... aktivní interval integrace
- N ... interval, přes který se ještě nezapočítal dílčí integrál

**Příklad**

Pomocí lichoběžníkového pravidla vypočítejte  $\int_1^5 \ln x \, dx$ . Ke zpřesnění použijte Richardsonovu extrapolaci.

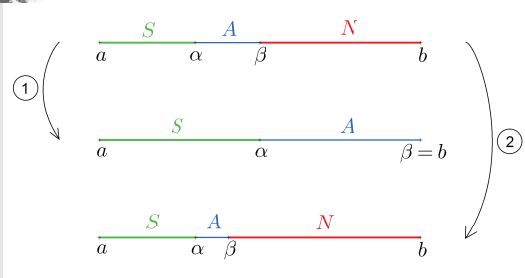
**Řešení:** Pro rozvoj chyby lichoběžníkového pravidla platí

$$I = T(f, h) + \underbrace{a_2 h^2}_{\text{tab. k=2}} + \underbrace{a_4 h^4}_{\text{tab. k=4}} + a_6 h^6 + \dots$$

Výsledky opět zapíšeme do tabulky

$h$	$T(f, h)$	1. zpřesnění ( $k=2$ )	2. zpřesnění ( $k=4$ )
4	$\frac{4}{2}(\ln 1 + \ln 5) = 3,2188$		
2	$\frac{2}{2}(\ln 1 + 2 \ln 3 + \ln 5) = 3,8066$	$\frac{3,8066 - 3,2188}{3} + 3,8066 = 4,0025$	
1	$\frac{1}{2}(\ln 1 + 2 \ln 2 + 2 \ln 3 + 2 \ln 4 + \ln 5) = 3,9827$	$\frac{3,9827 - 3,8066}{3} + 3,9827 = 4,0414$	$\frac{4,0414 - 4,0025}{15} + 4,0414 = 4,04399$





Změny stavu:

- (1) Je splněna podmínka na velikost chyby na intervalu A
- (2) Není splněna podmínka na velikost chyby na intervalu A

**Test chyby:** (pomocí metody polovičního kroku) – interval  $\langle \alpha, \beta \rangle$  rozpůlíme a použijeme stejný vzorec

$$\varepsilon_f(\alpha, \beta) \approx \frac{1}{2^k - 1} \left[ I_{(\alpha, \beta)}\left(\frac{h}{2}\right) - I_{(\alpha, \beta)}(h) \right], \quad k - \text{řád chyby vzorce}$$

$$\varepsilon_f(\alpha, \beta) \leq \varepsilon \frac{\beta - \alpha}{b - a}$$

$\varepsilon$  – celková požadovaná přesnost

**Algoritmus**

na začátku:

$$\begin{aligned} A &= \langle a, b \rangle \\ N &= \emptyset \\ S &= \emptyset \\ I_S &= 0 \quad (I_S \approx \int_a^b f(x) dx) \end{aligned}$$

(1) je splněn TEST CHYBY:

$$(i) I_S := I_S + I_A$$

$$(ii) S = S \cup A; \quad A = N$$

(2) není splněn TEST CHYBY:

$$(z) A \text{ rozpůlíme, tj. } A = \langle \alpha, \frac{\alpha + \beta}{2} \rangle$$

$$(zz) N := N \cup \langle \frac{\alpha + \beta}{2}, \beta \rangle$$

(zzz) nový TEST CHYBY

(1) a (2) opakujeme dokud  $S \neq \langle a, b \rangle$

**Příklad**

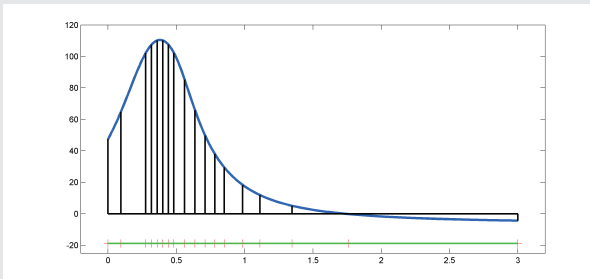
Použijte adaptivní přístup pro výpočet

$$\int_0^3 \frac{1}{(0,3x - 0,1)^2 + 0,01} + \frac{1}{(x - 0,5)^2 + 0,04} - 6 dx$$

tak, aby výsledná chyba aproximace integrálu byla menší než 0,25.

Pro výpočet použijte obdélníkové, lichoběžníkové i Simpsonovo pravidlo.

**Obdélníkové pravidlo**



výsledky v MATLABu

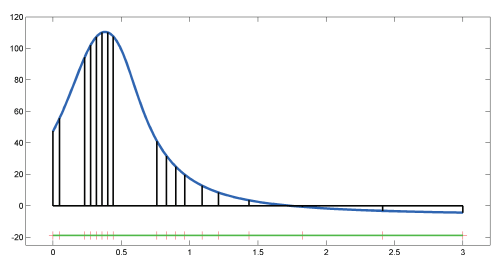
Adaptivní numerický výpočet určitého integrálu funkce  $f(x) = 1 / ((0.3x - 0.1)^2 + 0.01) + 1 / ((x - 0.5)^2 + 0.04) - 6$  na intervalu  $\langle 0.000000, 3.000000 \rangle$  se zadanou přesností 0.250000  
Pro výpočet se použije obdélníkové pravidlo I\_0.

Přesná hodnota integrálu ..... 69.800931  
Vypočtená hodnota integrálu .... 69.784747  
Skutečná chyba ..... 0.016184  
Odhadnutá chyba ..... 0.110713  
Počet podintervalů ..... 17  
Celkový počet dělení intervalu pro dodržení odhadu chyby .... 94

Adaptivní numerický výpočet určitého integrálu funkce  $f(x) = 1 / ((0.3x - 0.1)^2 + 0.01) + 1 / ((x - 0.5)^2 + 0.04) - 6$  na intervalu  $\langle 0.000000, 3.000000 \rangle$  se zadanou přesností 0.250000  
Pro výpočet se použije lichoběžníkové pravidlo I\_L.

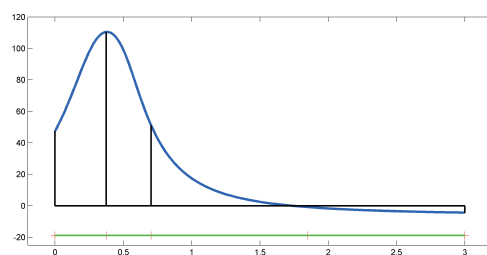
Přesná hodnota integrálu ..... 69.800931  
Vypočtená hodnota integrálu .... 69.686611  
Skutečná chyba ..... 0.114320  
Odhadnutá chyba ..... -0.084305  
Počet podintervalů ..... 17  
Celkový počet dělení intervalu pro dodržení odhadu chyby ... 89

**Lichoběžníkové pravidlo**



výsledky v MATLABu

**Simpsonovo pravidlo**



výsledky v MATLABu

Adaptivní numerický výpočet určitého integrálu funkce  
 $f(x) = 1 / ((0.3 + x - 1)^2 + 0.1) + 1 / ((x - 5)^2 + 0.4) - 6$   
 na intervalu  $\langle 0, 0.000000, 3, 0.000000 \rangle$   
 se zadanou přesností 0.250000  
 Pro výpočet se použije Simpsonovo pravidlo I.S.

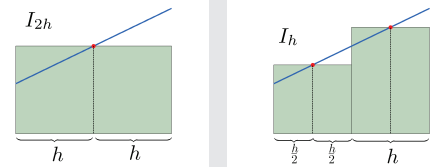
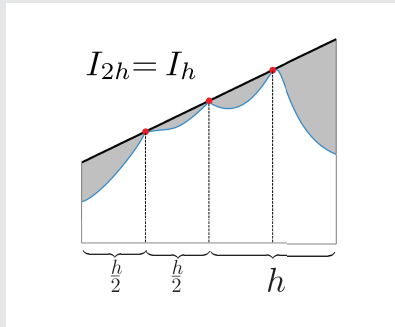
-----  
 Přesná hodnota integrálu ..... 69.800931  
 Vypočtená hodnota integrálu .... 69.849993  
 Skutečná chyba ..... -0.049061  
 Odhadnutá chyba ..... -0.073144  
 Počet podintervalů ..... 4  
 Celkový počet dělení intervalu  
 pro dodržení odhadu chyby .... 11  
 -----

**Poznámka**

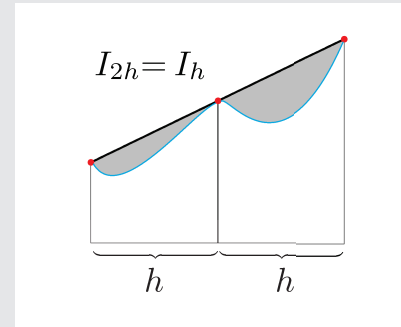
Odhadujeme-li chybu pomocí metody polovičního kroku, nemusí být skutečná chyba menší než zadaná tolerance.

**Příklady** v nichž je splněn TEST CHYBY, ale chyba je ve skutečnosti větší než zadaná tolerance

- **Obdélníkové pravidlo:**

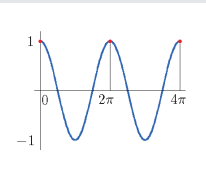


• **Lichoběžníkové pravidlo:**



Např.:

$$\int_0^{4\pi} \cos x \, dx, \quad \text{tolerance } \varepsilon = 10^{-5}$$



$$I_{4\pi} = 1 \cdot 4\pi \quad I_{2\pi} = 1 \cdot 2\pi + 1 \cdot 2\pi = 4\pi$$

Odhad chyby je

$$(I_{2\pi} - I_{4\pi}) \cdot \frac{1}{3} = 0$$

Přesná hodnota je

$$\int_0^{4\pi} \cos x \, dx = 0$$

Chyba skutečná je

$$4\pi$$

**Poznámka:**

**Newtonovy-Cotesovy** vzorce používají  $(m + 1)$  ekvidistantních uzlů a integrují přesně polynomy až do  $m$ -tého, případně  $(m + 1)$ -ního stupně (máme na mysli základní vzorec na intervalu  $(x_{i+1}, x_i)$ ).

Pro zvýšení přesnosti by se mohlo zdát výhodné použít více uzlů a funkci  $f$  aproximovat polynomem vyššího řádu. Ze zkušenosti z aproximace funkce polynomem ovšem víme, že limitní případ polynomu stupně  $m \rightarrow \infty$  nemusí odpovídat původní funkci (říkáme, že Newton-Cotesovy vzorce **nejsou konvergentní**).

**Gaussovy kvadraturní vzorce**

**Princip:** Snažíme se, aby kvadraturní vzorec integroval přesně polynomy co možná nejvyššího řádu.

Obecně kvadraturní vzorec (základní) uvažujeme ve tvaru

$$K(f) = \sum_{i=0}^m w_i f(x_i),$$

kde  $w_i$  jsou tzv. **váhy** a  $x_i$  jsou **uzly**.

Máme-li na základním intervalu  $m + 1$  bodů, potom nejvyšší možný stupeň polynomu, který ještě kvadraturní vzorec integruje přesně, je  $2m + 1$  (mluvíme o tzv. **algebraickém řádu přesnosti**).

Počet parametrů kvadraturního vzorce je  $2m + 2$

– polovina pro váhy  $w_i$

– polovina pro uzly  $x_i$

(Newton-Cotesovy vzorce integrovaly přesně polynomy do stupně  $\sim m$ .)

Čenou za vyšší přesnost budou ovšem **neekvidistantní uzly**.

**Příklad:** Odvoďte pro interval  $\langle -1, 1 \rangle$  základní Gaussův kvadraturní vzorec pro  $m = 0$  (tj. v intervalu uvažujeme pouze jeden uzel).

Řešení:

Kvadraturní vzorec pro  $m = 0$  má tvar

$$K(f) = w_0 f(x_0),$$

kde vystupují 2 neznámé  $w_0$  a  $x_0$ .

Vzorec musí přesně integrovat:

1) konstantu

$$\int_{-1}^1 b \, dx = 2b \stackrel{\text{poz.}}{=} w_0 \cdot \overbrace{f(x_0)}^b \Rightarrow w_0 = 2.$$

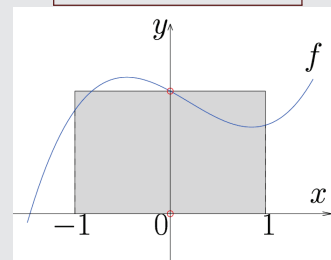
2) lineární funkci

$$\int_{-1}^1 (ax + b) \, dx = \left[ \frac{a}{2}x^2 + bx \right]_{-1}^1 = \frac{a}{2} - \frac{a}{2} + 2b \stackrel{\text{poz.}}{=} w_0 \cdot \overbrace{f(x_0)}^{ax_0 + b}$$

$$\Rightarrow 2b = 2(ax_0 + b) \Rightarrow x_0 = 0.$$

Jednodobový základní Gaussův kvadraturní vzorec je

$$K(f) = \int_{-1}^1 f(x) \, dx = 2f(0) + \underbrace{\frac{1}{3}f''(\xi)}_{\text{chyba}}$$



**Příklad:** Odvoďte pro interval  $\langle -1, 1 \rangle$  základní Gaussův kvadraturní vzorec pro  $m = 1$  (tj. v intervalu uvažujeme 2 uzly).

Řešení:

Kvadraturní vzorec pro  $m = 1$  má tvar

$$K(f) = w_0 f(x_0) + w_1 f(x_1),$$

kde vystupují 4 neznámé  $w_0, w_1, x_0$  a  $x_1$ .

Vzorec musí přesně integrovat polynom až 3 stupně:

$$\int_{-1}^1 (ax^3 + bx^2 + cx + d) dx = \left[ \frac{a}{4}x^4 + b\frac{x^3}{3} + c\frac{x^2}{2} + dx \right]_{-1}^1 = 0 \cdot a + \frac{2}{3} \cdot b + 0 \cdot c + 2 \cdot d \stackrel{\text{pož.}}{=} 0$$

$$\underbrace{w_0}_{f(x_0)} (ax_0^3 + bx_0^2 + cx_0 + d) + \underbrace{w_1}_{f(x_1)} (ax_1^3 + bx_1^2 + cx_1 + d) = K(f)$$

Soustava nelineárních rovnic pro 4 neznámé:

$$a: w_0 x_0^3 + w_1 x_1^3 = 0 \quad (1)$$

$$b: w_0 x_0^2 + w_1 x_1^2 = \frac{2}{3} \quad (2)$$

$$c: w_0 x_0 + w_1 x_1 = 0 \quad (3)$$

$$d: w_0 + w_1 = 2 \quad (4)$$

$$(1) - (3): w_0(x_0^2 - 1) + w_1(x_1^2 - 1) = 0$$

$$(2) - (4): w_0(x_0^2 - 1) + w_1(x_1^2 - 1) = -\frac{4}{3} \quad / \cdot (-x_1) \quad / \cdot (-x_0)^{\dagger}$$

$$\left. \begin{array}{l} \dagger \quad w_0(x_0 - x_1)(x_0^2 - 1) = \frac{4}{3}x_1 \\ \dagger \quad w_1(x_1 - x_0)(x_1^2 - 1) = \frac{4}{3}x_0 \\ (3) \text{ a } (4) \Rightarrow \begin{cases} w_1 = 2 - w_0 \\ w_0 x_0 + (2 - w_0)x_1 = 0 \\ w_0(x_0 - x_1) = -2x_1 \end{cases} \Rightarrow \end{array} \right\}$$

$$-2x_1(x_0^2 - 1) = \frac{4}{3}x_1 \Rightarrow -2(x_0^2 - 1) = \frac{4}{3} \Rightarrow x_0^2 - 1 = -\frac{2}{3} \Rightarrow x_0^2 = \frac{1}{3} \Rightarrow x_0 = -\sqrt{\frac{1}{3}}$$

analogicky:

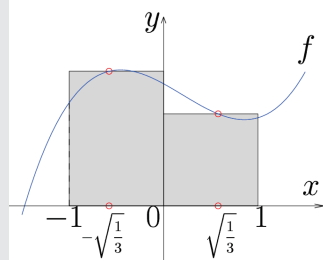
$$(3) \text{ a } (4) \Rightarrow \begin{cases} w_0 = 2 - w_1 \\ (2 - w_1)x_0 + w_1 x_1 = 0 \\ w_1(x_1 - x_0) = -2x_0 \end{cases} \Rightarrow$$

$$-2x_0(x_1^2 - 1) = \frac{4}{3}x_0 \Rightarrow -2(x_1^2 - 1) = \frac{4}{3} \Rightarrow x_1^2 - 1 = -\frac{2}{3} \Rightarrow x_1^2 = \frac{1}{3} \Rightarrow x_1 = \sqrt{\frac{1}{3}}$$

$$(3) \text{ a } (4): \begin{cases} w_0 + w_1 = 2 \\ \sqrt{\frac{1}{3}}w_0 - \sqrt{\frac{1}{3}}w_1 = 0 \Rightarrow w_0 = w_1 \Rightarrow w_0 = w_1 = 1 \end{cases}$$

Dvoubodový základní Gaussův kvadraturní vzorec je

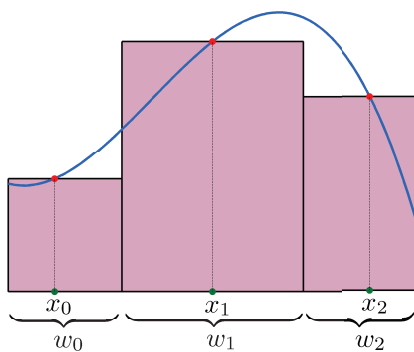
$$K(f) = \int_{-1}^1 f(x) dx = f\left(-\frac{\sqrt{3}}{3}\right) + f\left(\frac{\sqrt{3}}{3}\right) + \frac{1}{135} f^{(4)}(\xi) \quad \text{chyba}$$



Poznámka:

Další základní Gaussův kvadraturní vzorec (třibodový, tj. pro  $m = 2$ ) vypadá na intervalu  $(-1, 1)$  takto:

$$K(f) = \int_{-1}^1 f(x) dx = \frac{5}{9} f\left(-\frac{\sqrt{3}}{5}\right) + \frac{8}{9} f(0) + \frac{5}{9} f\left(\frac{\sqrt{3}}{5}\right) + \frac{1}{15750} f^{(6)}(\xi) \quad \text{chyba}$$



Poznámka: Koeficienty a uzly vzorců vyšších řádů jsou uvedeny v tabulkách.

Poznámka: To, že jsme vyjádřili  $\int_{-1}^1 f(x) dx$  neubírá nic na obecnosti, můžeme totiž libovolný interval  $(a, b)$  transformovat na  $(-1, 1)$  a použít odvozené vztahy.

Poznámka: Gaussovy kvadraturní vzorce jsou konvergentní.

Příklad

Vypočítejte  $\int_1^{1.2} e^x dx$  použitím jedno- a dvoubodového základního Gaussova kvadraturního vzorce.

Řešení:



$$x_i = 1,1 + 0,1 \cdot x_i^{(0)},$$

$$w_i = \frac{1}{2}(1,2 - 1)w_i^{(0)} = 0,1w_i^{(0)}$$

$n = 0$

$$\int_1^{1.2} f(x) dx \approx 0,2 \cdot f(1,1) = 0,2 \cdot e^{1,1} = 0,600833$$

$$x_0 = 1,1 + 0,1 \cdot 0 = 1,1$$

$$w_0 = 0,1 \cdot 2 = 0,2$$

$n = 1$

$$\int_1^{1.2} f(x) dx \approx 0,1 \left[ f\left(1,1 - 0,1\sqrt{\frac{1}{3}}\right) + f\left(1,1 + 0,1\sqrt{\frac{1}{3}}\right) \right] = 0,1 \left[ 2,835632 + 3,182716 \right] = 0,601834$$

$$x_0 = 1,1 + 0,1 \cdot \left(-\sqrt{\frac{1}{3}}\right) = 1,1 - 0,1\sqrt{\frac{1}{3}}$$

$$x_1 = 1,1 + 0,1\sqrt{\frac{1}{3}}$$

$$w_0 = 0,1 \cdot 1 = 0,1$$

$$w_1 = 0,1$$

Přesný výsledek:  $e^{1.2} - e \approx 0,601835$

Poznámka:

Podobně jako u Newton-Cotesových vzorců můžeme definovat složené Gaussovy kvadraturní vzorce

Příklad

Vypočítejte  $\int_0^\pi x^2 \sin 3x dx$  použitím jedno-, dvou- a třibodového složeného Gaussova kvadraturního vzorce. Počet dělení intervalu  $(0, \pi)$  volte  $N = 10$ , resp.  $N = 20$ .

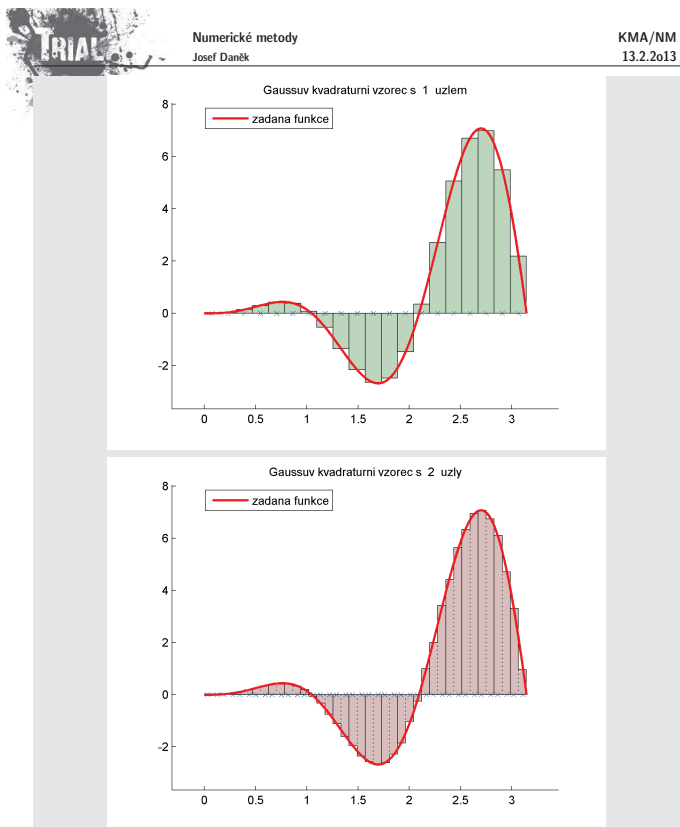
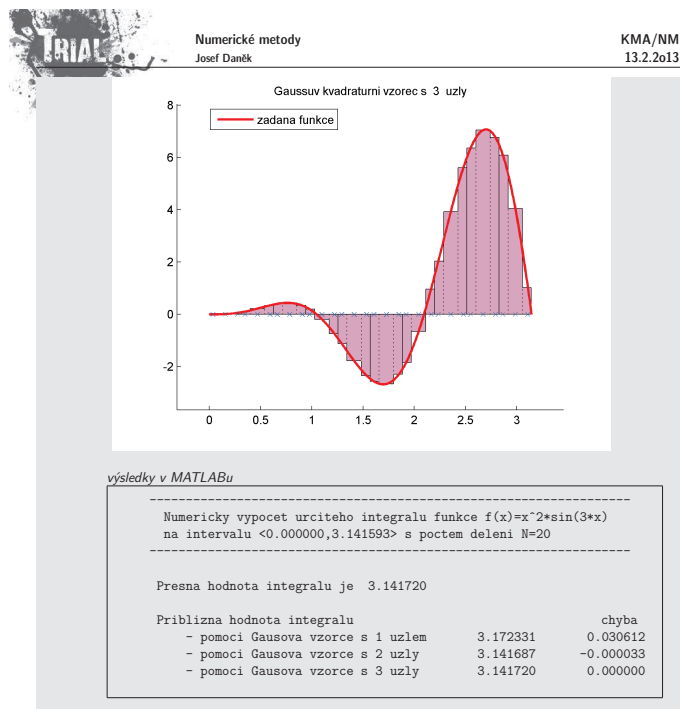
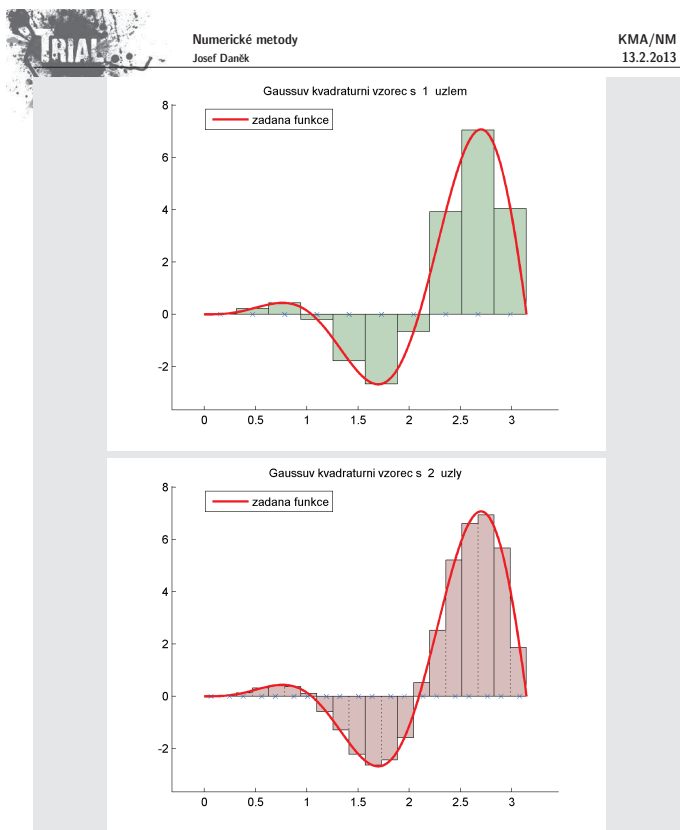
Řešení:

výsledky v MATLABu

Numericky vypočet určitého integrálu funkce  $f(x) = x^2 \sin(3x)$  na intervalu  $\langle 0, \pi \rangle$  s počtem dělení  $N=10$

Přesná hodnota integrálu je 3.141720

Přibližná hodnota integrálu		chyba
- pomocí Gaussova vzorce s 1 uzlem	3.266250	0.124530
- pomocí Gaussova vzorce s 2 uzly	3.141191	-0.000529
- pomocí Gaussova vzorce s 3 uzly	3.141721	0.000001



Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

Gaussův kvadraturní vzorec s 3 uzly

— zadana funkce

**Integrovaní periodické funkce přes periodu**

Pro lichoběžníkové pravidlo platí:

$$T(f, h) = \int_a^b f(x) dx + \frac{h^2}{12} [f''(b) - f''(a)] - \frac{h^4}{720} [f^{(3)}(b) - f^{(3)}(a)] + \dots$$

(tzv. Eulerův - Maclaurinův vzorec)

Souvislost s rozvojem chyby:

$$T(f, h) = \int_a^b f(x) dx + \frac{h^2}{12} \frac{f''(b) - f''(a)}{f''(b) - f''(a)} (b-a) - \frac{h^4}{720} \frac{f^{(3)}(b) - f^{(3)}(a)}{f^{(3)}(b) - f^{(3)}(a)} (b-a) + \dots$$

Pro kladnou periodickou funkci s periodou na intervalu  $(a, b)$  platí:

$$f'(a) = f'(b)$$

$$f^{(3)}(a) = f^{(3)}(b)$$

⋮

**Pozor:** Obecně nemusí platit, že  $T(f, h)$  je přesná hodnota integrálu  $\int_a^b f(x) dx$ , protože zbytek má tvar

$$(b-a) c_{2m} h^{2m} f^{(2m)}(\xi)$$

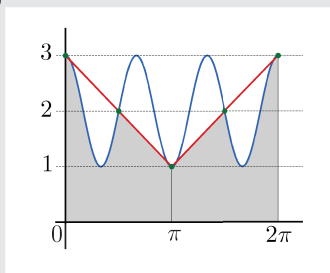
a  $\xi$  neznáme.

Platí však, že chyba je velikosti  $O(h^{2m})$  pro libovolné  $m$  takové, že  $f$  má spojitou  $2m$ -tou derivaci. Proto není nutné používat Rombergovu metodu.

**Příklad:** Vypočítejte složeným lichoběžníkovým pravidlem

$$\int_0^{2\pi} (2 + \cos 3x) dx.$$

(Přesná hodnota je  $4\pi$ .)



Zvolíme krok  $h = \pi$ , tj.  $N = 2$ :

$$T(f, \pi) = \frac{\pi}{2} [f(0) + 2f(\pi) + f(2\pi)] = \frac{\pi}{2} (3 + 2 \cdot 1 + 3) = 4\pi.$$

**Platí:** Složené lichoběžníkové pravidlo s  $(k+2)$  uzly integruje přesně trigonometrické polynomy  $k$ -tého stupně a stupňů menších (tj. obsahující členy  $\cos kx, \sin kx$ ) přes periodu  $2\pi$ .

### Nevlastní integrály

Při výpočtu integrálu

$$\int_{-\infty}^{\infty} f(x) dx$$

lze většinou předpokládat, že hodnoty funkce  $f$  a nižší derivace  $f$  jsou vně nějakého intervalu  $(-R, R)$  dostatečně malé. Proto je vhodné použít lichoběžníkové pravidlo pro integrál

$$\int_{-R}^R f(x) dx.$$

### Příklad

Vypočítejte integrál  $I = \int_{-\infty}^{\infty} e^{-x^2} dx$ .

Pro  $|z| \geq 4$  je integrand menší než  $0,5 \cdot 10^{-6}$  a jeho první derivace menší než  $10^{-6}$ . Použijeme-li lichoběžníkové pravidlo pro  $(-4, 4)$ , dostaneme:

$$T(f, 1) = 1,772636$$

$$T(f; 0, 5) = 1,772453$$

Přesná hodnota

$$I = \sqrt{\pi} \doteq 1,7724538$$

Poznamenejme, že

$$\left| \int_{-\infty}^{\infty} e^{-x^2} dx - \int_{-4}^4 e^{-x^2} dx \right| < 10^{-7}.$$

Při výpočtu integrálu  $\int_0^{\infty} f(x) dx$  můžeme použít transformaci  $x = f(t)$ .

a)  $x = -\ln t, \quad dx = -\frac{dt}{t}, \quad \left. \frac{x}{t} \right|_1^0 = \left. \frac{-\ln t}{t} \right|_1^0$

$$\int_0^{\infty} f(x) dx = - \int_1^0 f(-\ln t) \frac{dt}{t} = \int_0^1 \frac{f(-\ln t)}{t} dt$$

b)  $x = \frac{t}{1-t}, \quad dx = (1-t)^{-2} dt, \quad \left. \frac{x}{t} \right|_0^{\infty} = \left. \frac{1}{1-t} \right|_0^{\infty}$

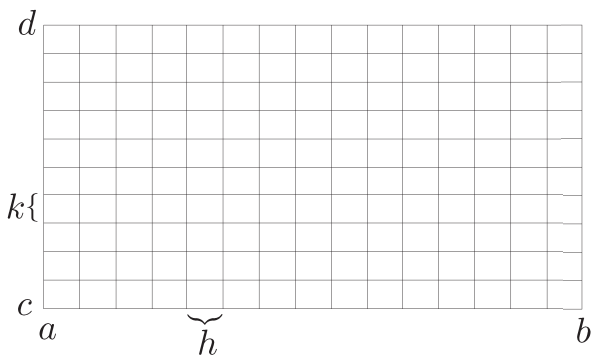
$$\int_0^{\infty} f(x) dx = \int_0^1 f\left(\frac{t}{1-t}\right) \frac{dt}{(1-t)^2}$$

### Integrovaní funkce 2 proměnných

Odvodte obdélníkové a lichoběžníkové pravidlo pro integrování funkce 2 proměnných na obdélníku  $(a, b) \times (c, d)$ , tj.

$$\int_a^b \left( \int_c^d f(x, y) dy \right) dx$$

**Řešení:**

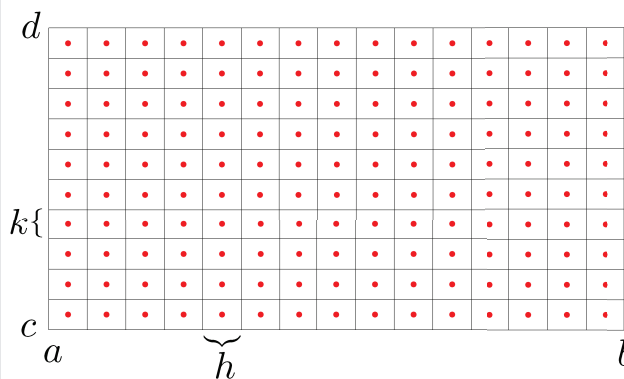


$$h = \frac{b-a}{N}, \quad k = \frac{d-c}{M}, \quad x_i = a + i \cdot h, \quad y_j = c + j \cdot k$$

$$\int_a^b \left( \int_c^d f(x, y) dy \right) dx = \dots$$

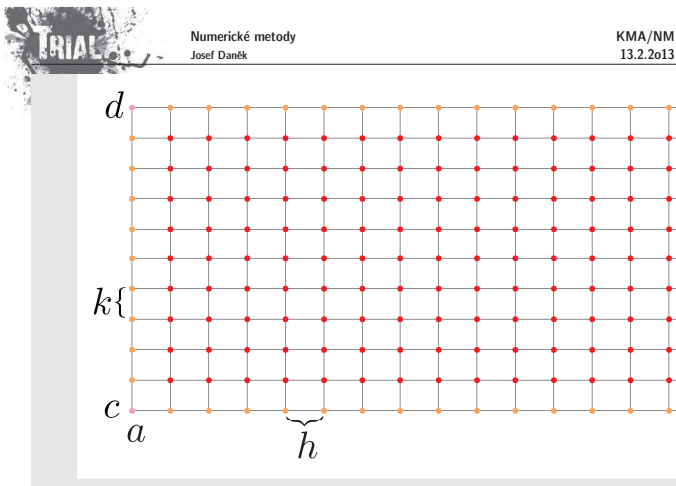
• obdélníkové pravidlo:

$$\dots = h \sum_{i=0}^{N-1} \left( \int_c^d f\left(x_i + \frac{h}{2}, y\right) dy \right) = h \sum_{i=0}^{N-1} \left( k \sum_{j=0}^{M-1} f\left(x_i + \frac{h}{2}, y_j + \frac{k}{2}\right) \right) = \underline{hk \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} f\left(x_i + \frac{h}{2}, y_j + \frac{k}{2}\right)}$$



• lichoběžníkové pravidlo:

$$\begin{aligned} \dots &= \frac{h}{2} \sum_{i=0}^{N-1} \left( \int_c^d [f(x_i, y) + f(x_{i+1}, y)] dy \right) = \\ &= \frac{h}{2} \sum_{i=0}^{N-1} \left( \frac{k}{2} \sum_{j=0}^{M-1} [f(x_i, y_j) + f(x_{i+1}, y_{j+1})] + \frac{k}{2} \sum_{j=0}^{M-1} [f(x_{i+1}, y_j) + f(x_i, y_{j+1})] \right) = \\ &= \underline{\frac{hk}{2} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} [f(x_i, y_j) + f(x_{i+1}, y_{j+1}) + f(x_{i+1}, y_j) + f(x_i, y_{j+1})]} \end{aligned}$$



Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

### Kapitola 11. Počáteční úlohy pro ODR - I

**Počáteční úlohy pro obyčejné diferenciální rovnice 1. řádu**

**Formulace:**  
Je dána funkce dvou proměnných  $f = f(x, y)$ ,  $y \in \mathbb{R}$ ,  $x \in (a, b)$  a čísla  $x_0 \in (a, b)$  a  $y_0 \in \mathbb{R}$ . Chceme najít takovou funkci  $y = y(x)$ ,  $x \in (x_0, b)$ , která na intervalu  $(x_0, b)$  vyhovuje rovnici

$$y' = f(x, y)$$

a splňuje počáteční podmínku

$$y(x_0) = y_0$$

Funkci  $y = y(x)$ , která splňuje počáteční podmínku a rovnost  $y' = f(x, y(x))$  na příslušném intervalu, nazýváme řešením úlohy.

V některých případech budeme uvažovat speciální úlohu s rovnicí

$$y' = a(x)y + b(x)$$

nebo s rovnicí

$$y' = \lambda y$$

Velmi podstatnou úlohu v našich dalších úvahách bude hrát předpoklad, že funkce  $f$  je na nějakém intervalu **lipschitzovsky spojitá** (v druhé proměnné), tj. platí:

$$|f(x, y_1) - f(x, y_2)| \leq L|y_1 - y_2| \quad \forall x \in (a, b) \quad \forall y_1, y_2 \in \mathbb{R}$$

**Příklad 1**  
Uvažujme úlohu

$$y' = y^2$$

$$y(0) = \eta > 0$$

Funkce  $f(x, y) = y^2$  je lipschitzovsky spojitá na libovolném konečném intervalu  $(\eta - K, \eta + K)$ . Konstanta  $L = 2(\eta + K)$ .

$$|y_1^2 - y_2^2| \leq 2(\eta + K)|y_1 - y_2|$$

$$|y_1 + y_2| \leq 2(\eta + K) \quad \forall x \in (a, b), \quad \forall y_1, y_2 \in (\eta - K, \eta + K)$$


---


$$|y_1 + y_2| \leq |y_1| + |y_2| \leq \eta + K + \eta + K = 2(\eta + K) \quad \forall x \in (a, b), \quad \forall y_1, y_2 \in (\eta - K, \eta + K)$$

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

Řešíme pomocí separace proměnných

$$\frac{dy}{dx} = y^2$$

$$\frac{dy}{y^2} = dx$$

$$-\frac{1}{y} = x + c$$

$$y = -\frac{1}{x + c}$$

$$y(0) = \eta$$

$$-\frac{1}{c} = \eta$$

$$c = -\frac{1}{\eta}$$

Řešení této úlohy má tvar

$$y(x) = \frac{1}{\frac{1}{\eta} - x}$$

Když  $x \rightarrow (\frac{1}{\eta})_-$ , pak  $y \rightarrow \infty$ .  
 $\Rightarrow$  Pro všechna  $y_1, y_2 \in \mathbb{R}$  nelze najít jednu konstantu  $L$  a řešení neexistuje pro libovolné  $x$  (řešení existuje pouze do určitého času).

**Příklad 2**  
Uvažujme úlohu

$$y' = \sqrt{y}$$

$$y(0) = 0$$

Funkce  $f(x, y) = \sqrt{y}$  není lipschitzovsky spojitá v okolí 0, protože  $\frac{\partial f(x, y)}{\partial y} = \frac{1}{2\sqrt{y}} \rightarrow \infty$  pro  $y \rightarrow 0_+$ .  
 Z věty o střední hodnotě plyne:

$$f(x, y_1) - f(x, y_2) = \frac{\partial f(x, \xi)}{\partial y}(y_1 - y_2)$$

Řešení této úlohy není jednoznačné:

Numerické metody  
Josef Daněk

KMA/NM  
13.2.2013

$$y_1(x) = 0$$

$$y_2(x) = \frac{1}{4}x^2$$

$$\frac{dy}{dx} = \sqrt{y}$$

1)  $y \equiv 0$  OK    2)  $y \neq 0$

$$\frac{dy}{\sqrt{y}} = dx$$

$$2\sqrt{y} = x + c$$

$$\sqrt{y} = \frac{x}{2} + c$$

$$y = \left(\frac{x}{2} + c\right)^2$$

$$y(0) = 0$$

$$c^2 = 0$$

$$c = 0$$

$$y = \frac{x^2}{4}$$

**Příklad 3**  
Uvažujme úlohu

$$y' = \lambda y$$

$$y(0) = 1$$

Funkce  $f(x, y) = \lambda y$  je lipschitzovsky spojitá pro všechna  $y \in \mathbb{R}$  s konstantou  $L = |\lambda|$ .

$$|\lambda y_1 - \lambda y_2| = |\lambda||y_1 - y_2| \leq |\lambda||y_1 - y_2|$$

Tato úloha má právě jedno řešení pro všechna  $x \in (0, \infty)$  ve tvaru

$$y(x) = e^{\lambda x}$$

$$\frac{dy}{dx} = \lambda y$$

$$\frac{dy}{y} = \lambda dx$$

$$\ln|y| = \lambda x + c$$

$$\ln|y| = \ln e^{\lambda x} + \ln c$$

$$y = c e^{\lambda x}$$

$$y(0) = 1$$

$$c = 1$$

$$y = e^{\lambda x}$$

Následující příklady ukazují význam Lipschitzovy konstanty.

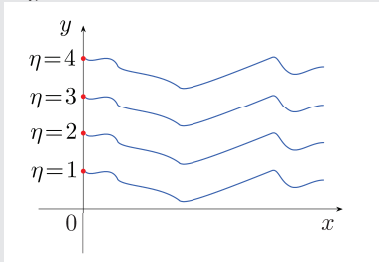
**Příklad 4**

Uvažujme úlohu ve tvaru

$$\begin{cases} y' = g(x) \\ y(0) = \eta \end{cases}$$

Řešení má tvar  $y(x) = \eta + \int_0^x g(\xi) d\xi$  a Lipschitzova konstanta  $L=0$

Křivky řešení mohou vypadat třeba takto:



Změníme-li počáteční podmínku  $\eta$ , potom nové řešení je pouhé posunutí původního do hodnoty  $\eta$ .

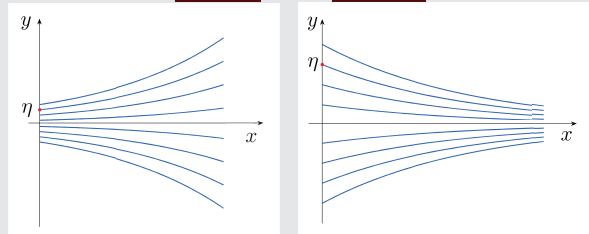
**Příklad 5**

Uvažujme úlohu

$$\begin{cases} y' = 3y \\ y(0) = \eta \end{cases} \quad \text{a} \quad \begin{cases} y' = -3y \\ y(0) = \eta \end{cases}$$

$$\bar{y}(x) = \eta e^{3x} \quad \text{a} \quad \bar{y}(x) = \eta e^{-3x}$$

Dostaneme řešení



V obou případech je Lipschitzova konstanta  $L=3$ . Její velikost však může ovlivňovat chování konkrétní numerické metody pro konkrétní úlohu.

**Věta** Necht funkce  $f(x, y)$  má následující vlastnosti:

(i) je definována v pásu  $S = (a, b) \times \mathbb{R}$  ( $a, b \dots$  konečné),

(ii) je spojitá v proměnné  $x \in (a, b)$  pro každé  $y \in \mathbb{R}$ ,

(iii) splňuje Lipschitzovu podmínku v proměnné  $y$ , tj. existuje číslo  $L$  takové, že platí nerovnost

$$|f(x, y_1) - f(x, y_2)| \leq L |y_1 - y_2| \quad \forall x \in (a, b) \quad \forall y_1, y_2 \in \mathbb{R}$$

Potom pro každé  $x_0 \in (a, b)$  a libovolné  $y_0 \in \mathbb{R}$  existuje právě jedna funkce  $y = y(x)$  s vlastnostmi:

- a)  $y(x)$  je spojitá a spojitě diferencovatelná pro  $x \in (a, b)$ ,
- b) platí rovnost  $y'(x) = f(x, y(x))$  pro  $\forall x \in (a, b)$ ,
- c)  $y(x_0) = y_0$ .

Numerické metody lze dělit podle různých kritérií:

- A) metody založené na numerické derivaci **X** na numerické integraci
- B) jedнокrokové metody **X** vícečrokové metody
- C) explicitní metody **X** implicitní metody
- D) metody s konstantním krokem **X** metody s proměnným krokem

Princip:

Základem metod je diskretizace proměnných.

Přibližné řešení nehledáme jako spojitou funkci, ale nageneme body  $x_0, x_1, x_2, \dots$  a určujeme čísla  $y_0, y_1, y_2, \dots$  která aproximují  $y(x_0), y(x_1), y(x_2), \dots$

Pro jednoduchost můžeme uvažovat ekvidistantní dělení, tj.  $h = x_{k+1} - x_k, \forall k$ .

**Eulerova metoda**

- nejjednodušší jedнокroková explicitní metoda; lze odvodit řadou postupů

• 1. odvození

$y_0 \dots$  dáno

$y_1 \dots$  počítáme extrapolací z hodnoty  $y_0$ , přičemž se na intervalu  $(x_0, x_1)$  řešení aproximuje přímkou,

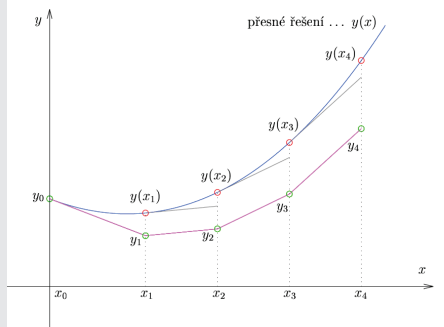
kteřá prochází bodem  $[x_0, y_0]$  a má směrnici  $y' = f(x_0, y_0)$ .

Ta má rovnici  $y - y_0 = (x - x_0)f(x_0, y_0)$ . Tj. pro  $x_1$  dostáváme:

$$y_1 = y_0 + \frac{(x_1 - x_0)}{h} f(x_0, y_0)$$

Obecně dostaneme rekurentní vztah

$$y_{k+1} = y_k + h_k f(x_k, y_k), \quad k = 0, 1, 2, \dots$$



• 2. odvození Pomocí Taylorova rozvoje.

$$y(x_{k+1}) = y(x_k) + h_k \underbrace{y'(x_k)}_{=f(x_k, y(x_k))} + \frac{1}{2} h_k^2 \underbrace{y''(\xi_k)}_{(*)}, \quad \xi_k \in (x_k, x_{k+1})$$

(\*) zanedbáme a dostaneme vztah pro přibližné řešení

$$y_{k+1} = y_k + h_k f(x_k, y_k), \quad k = 0, 1, 2, \dots$$

$y_0 \dots$  počáteční podmínka

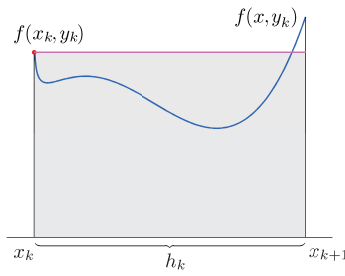
• 3. odvození Původní diferenciální rovnici nahradíme diferenciální rovnicí (aproximujeme derivaci).

$$y' = f(x, y) \rightarrow \frac{y_{k+1} - y_k}{h_k} = f(x_k, y_k), \quad k = 0, 1, 2, \dots$$

• 4. odvození Původní diferenciální rovnici zintegrujeme a aproximujeme určitý integrál.

$$y' = f(x, y) \rightarrow y(x_{k+1}) - y(x_k) = \int_{x_k}^{x_{k+1}} f(x, y(x)) dx$$

(\*)  $y(x)$  na  $(x_k, x_{k+1})$  aproximujeme konstantou  $y_k$



$$y_{k+1} - y_k = h_k f(x_k, y_k), \quad k = 0, 1, 2, \dots$$

Poznámka:

Eulerova metoda je

• jedнокroková metoda ( $y_k \rightarrow y_{k+1}$ )

K výpočtu  $y_{k+1}$  použijeme pouze předchozí hodnotu  $y_k$ .

• explicitní metoda (na pravé straně není  $y_{k+1}$ )

V získané formuli je explicitně vyjádřena hodnota  $y_{k+1}$ .

**Příklad**

Pomocí Eulerovy metody řešte následující úlohu na intervalu  $(0, 0.6)$  s konstantními kroky  $h = 0, 2$  a  $h = 0, 1$ .

$$\begin{cases} y' = x - y \\ y(0) = 1 \end{cases}$$

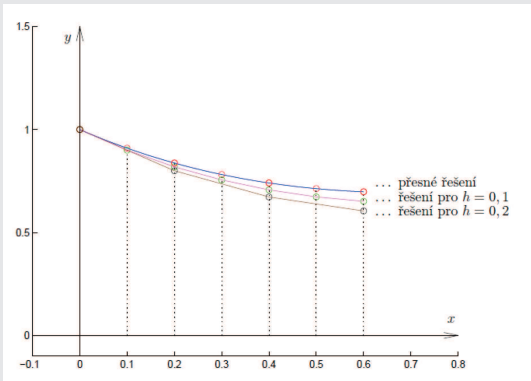
Řešení:

(Přesné řešení:  $y(x) = 2e^{-x} + x - 1$ ).

Eulerova metoda je dána rekurentním vztahem:

$$y_{k+1} = y_k + h \cdot f(x_k, y_k).$$

$x_k$	$y(x_k)$	$h = 0,2$		$h = 0,1$	
		$y_k$	$e_k$	$y_k$	$e_k$
0	1,000	1,000	0,000	1,000	0,000
0,1	0,910			0,900	0,010
0,2	0,837	0,800	0,037	0,820	0,017
0,3	0,782			0,758	0,024
0,4	0,741	0,680	0,061	0,712	0,029
0,5	0,713			0,681	0,032
0,6	0,698	0,624	0,074	0,663	0,035



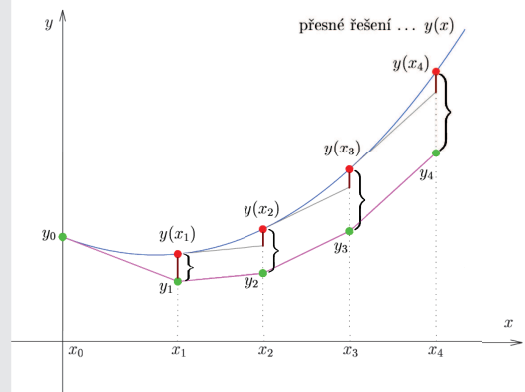
- Poznámky:**
- Vidíme, že je chyba úměrná  $h$ .
  - Chyba s rostoucím  $x$  vzrůstá.

**Definice: Lokální diskretizační chyba  $d_k$**  na intervalu  $(x_k, x_{k+1})$  je nepřesnost, s níž hodnoty teoretického řešení dané úlohy splňují rekurentní vztah, ze kterého se počítá hodnota  $y_{k+1}$ .

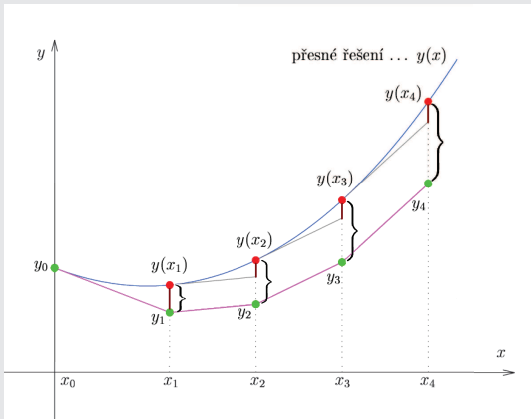
Pro Eulerovu metodu je lokální diskretizační chyba  $d_k$ :

$$y(x_{k+1}) = y(x_k) + h_k f(x_k, y(x_k)) + d_k$$

**Poznámka:** Lokální diskretizační chyba se nazývá lokální proto, že  $d_k$  lze interpretovat také jako chybu jednoho kroku metody (při výpočtu  $y_{k+1}$ ) za předpokladu, že všechny hodnoty  $y_k, y_{k-1}, \dots$  potřebné k výpočtu  $y_{k+1}$  jsou přesné.



**Definice: Globální diskretizační chyba** je  $e_k = y(x_k) - y_k$ , tj. rozdíl teoretické hodnoty řešení a vypočtené hodnoty řešení v daném bodě  $x_k$ .



**Globální diskretizační chyba Eulerovy metody** (pro konstantní krok  $h$ )

**Přibližné řešení:**

$$y_0 = y(x_0)$$

$$y_{k+1} = y_k + h f(x_k, y_k) \quad k = 0, 1, \dots$$

**Přesné řešení:**

$$y(x_{k+1}) = y(x_k) + h f(x_k, y(x_k)) + d_k \quad k = 0, 1, \dots$$

Po odečtení:

$$e_0 = 0$$

$$e_{k+1} = e_k + h (f(x_{k+1}, y(x_{k+1})) - f(x_k, y_k)) + d_k$$

tj. v každém kroku se ke globální chybě  $e_k$  připočítá lokální chyba  $d_k$  a člen  $h \cdot (\dots)$ , který představuje nepřesnosti z minulých kroků.

**Příklad:**

Speciální případ, kdy  $f$  nezávisí na  $y$ :

$$\begin{aligned} y' &= f(x) \\ y(x_0) &= y_0 \end{aligned} \Rightarrow e_{k+1} = \sum_{m=0}^k d_m \quad \textcircled{a}$$

tj. globální chyba je součtem lokálních chyb.

**Poznámka:**

Lokální chyba Eulerovy metody je  $O(h^2)$  (viz další slide).

Protože  $\textcircled{a}$  má  $k$  sčítanců a protože pro pevné  $x$  je  $k = \frac{x - a}{h}$ , plyne z  $\textcircled{a}$

$$e(x, h) = \frac{\text{const}}{h} \cdot O(h^2) = O(h)$$

... podobně jako u základních a složených kvadraturních vzorců.

**Poznámka:**

Lokální i globální diskretizační chyby jsou chyby aproximace, tj. neuvažovali jsme zaokrouhlovací chyby.

**Definice: Řád diferenciální metody** je největší přirozené číslo  $p$  takové, že pro danou metodu aplikovanou na libovolnou počáteční úlohu s dostatečně hladkým řešením platí při každém pevném  $k$  a  $h_k \rightarrow 0$  odhad

$$d_k = O(h_k^{p+1}).$$

**Řád Eulerovy metody**

Ze vztahu pro lokální diskretizační chybu  $d_k$  plyne:

$$d_k = y(x_{k+1}) - y(x_k) - h_k \cdot \frac{y'(x_k)}{f(x_k, y(x_k))}$$

$y(x_{k+1})$  vyjádříme pomocí Taylorova rozvoje (předpokládáme, že  $y$  má 2. derivaci)

$$y(x_{k+1}) = y(x_k) + h_k y'(x_k) + \frac{1}{2} h_k^2 y''(\xi) \quad \xi \in (x_k, x_{k+1})$$

Po dosazení:

$$d_k = \frac{1}{2} h_k^2 y''(\xi) = O(h_k^2)$$

$2 = p + 1 \Rightarrow$  **řád Eulerovy metody** je  $p = 1$ .



Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

**Obecná jednkroková metoda**

Eulerova metoda je sice velmi jednoduchá (řád je 1), ale k dosažení určité přesnosti musíme používat velmi malé kroky  $h_k$ . Chceme-li jednkrokovou metodu vyššího řádu, musíme se zříci linearity

$$y_{k+1} = y_k + h_n f(x_k, y_k) \quad k = 0, 1, 2, \dots$$

$$\bar{y}_{k+1} = y_k + \bar{\Phi}(x_k, y_k, h_k, f) \quad k = 0, 1, 2, \dots$$

**Metody Taylorova typu**

Hodnotu  $y(x_{k+1})$  budeme aproximovat pomocí Taylorova rozvoje vyššího řádu  $p$  (v Eulerově metodě byl použit řád 1), tj.

$$y(x_{k+1}) = y(x_k + h_k) = y(x_k) + h_k y'(x_k) + \frac{h_k^2}{2!} y''(x_k) + \dots + \frac{h_k^p}{p!} y^{(p)}(x_k) + \frac{h_k^{p+1}}{(p+1)!} y^{(p+1)}(\xi_k) \quad \xi_k \in (x_k, x_{k+1})$$

Je třeba dosadit za derivace  $y$  v bodě  $x_k$ . Derivace určíme postupným derivováním funkce  $f$ .

$$y' = f(x, y(x))$$

$$y'' = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} \cdot \frac{dy}{dx} = f_x + f_y \cdot f \stackrel{\text{ozn.}}{=} f^{[1]}(x, y)$$

$$y''' = \frac{\partial f^{[1]}}{\partial x} + \frac{\partial f^{[1]}}{\partial y} \cdot \frac{dy}{dx} = f_x^{[1]} + f_y^{[1]} \cdot f \stackrel{\text{ozn.}}{=} f^{[2]}(x, y)$$

Obecně lze odvodit rekurenci

$$y^{(r+1)} = f^{[r+1]}(x, y(x)) = f_x^{[r+1]}(x, y(x)) + f_y^{[r+1]}(x, y(x)) \cdot f(x, y(x)) \quad r = 1, 2, \dots$$

Po dosazení (uvažujeme konstantní krok  $h$ ) dostáváme

$$y_{k+1} = y_k + h f(x_k, y_k) + \frac{h^2}{2} f^{[1]}(x_k, y_k) + \dots + \frac{h^p}{p!} f^{[p-1]}(x_k, y_k)$$

**Poznámka:**  
Metody Taylorova typu se v praxi nepoužívají právě z důvodu nutnosti vyjadřovat derivace  $y', y'', \dots$

**Příklad** Odvoďte metodu Taylorova typu 2.řádu pro řešení následující úlohy na intervalu  $(0; 0,6)$  s konstantním krokem  $h = 0,2$

$$y' = x - y, \quad y(0) = 1$$

**Řešení:**  
(Přesné řešení:  $y(x) = 2e^{-x} + x - 1$ ).

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$$\bar{y}_{k+1} = y_k + h f(x_k, y_k)$$

$$y_{k+1} = y_k + \frac{h}{2} [f(x_k, y_k) + f(x_{k+1}, \bar{y}_{k+1})]$$

**Poznámka:** Lokální diskretizační chyba, tj. chyba jednoho kroku metody, je  $d_k = O(h^3)$ . Globální chyba je potom o řád nižší, tj.  $e_k = O(h^2)$ , protože chyba metody se zvětšuje lineárně s počtem kroků  $k \sim \frac{1}{h}$ .

**Modifikovaná Eulerova metoda** (Runge-Kuttova metoda 2. řádu)

vztah  $y' = f(x, y(x))$  opět zintegrujeme přes interval  $(x_k, x_{k+1})$

$$\int_{x_k}^{x_{k+1}} y'(x) dx = \int_{x_k}^{x_{k+1}} f(x, y(x)) dx$$

$$y(x_{k+1}) - y(x_k) = \int_{x_k}^{x_{k+1}} f(x, y(x)) dx$$

použijeme **obdélníkové pravidlo**

$$y(x_{k+1}) - y(x_k) = h \cdot f\left(x_k + \frac{h}{2}, y\left(x_k + \frac{h}{2}\right)\right) + O(h^3)$$

viz chyba obd. pr.

hodnotu  $y\left(x_k + \frac{h}{2}\right)$  určíme pomocí Eulerovy metody

$$y\left(x_k + \frac{h}{2}\right) = y(x_k) + \frac{h}{2} f(x_k, y(x_k)) + O(h^2)$$

dostáváme metodu ve tvaru

$$y_{k+\frac{1}{2}} = y_k + \frac{h}{2} f(x_k, y_k)$$

$$y_{k+1} = y_k + h f\left(x_k + \frac{h}{2}, y_{k+\frac{1}{2}}\right)$$

**Poznámka:** Lokální diskretizační chyba je opět  $d_k = O(h^3)$ . Globální chyba je potom o řád nižší, tj.  $e_k = O(h^2)$ , protože chyba metody se zvětšuje lineárně s počtem kroků  $k \sim \frac{1}{h}$ .

Ukažme si jiné odvození předchozích dvou Runge-Kuttových metod 2. řádu. Odvození vychází z geometrické interpretace.

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

$$f(x, y) = x - y$$

$$f^{[1]}(x, y) = f_x + f_y \cdot f = 1 + (-1) \cdot f(x, y) = 1 - x + y.$$

Dostáváme rekurentní vztah:

$$y_{k+1} = y_k + h(x_k - y_k) + \frac{1}{2} h^2 (1 - x_k + y_k)$$

$x_k$	$y(x_k)$	$y_k$	$h(x_k - y_k)$	$\frac{h^2}{2}(1 - x_k + y_k)$	$e_k$
0	1,000	1,000	-0,200	0,040	0,000
0,2	0,837	0,840	-0,128	0,033	-0,003
0,4	0,741	0,745	-0,069	0,027	-0,004
0,6	0,698	0,703			-0,005

**Poznámka:**  
Vidíme, že metoda Taylorova typu 2. řádu pro  $h = 0,2$  dává přesnější výsledky než Eulerova metoda s  $h = 0,1$ .

**Metody Runge-Kuttova typu**

- Univerzálnější metody než metody Taylorova typu.
- Vychází také z Taylorova polynomu, ale nepoužívá se ho přímo, aby nebylo nutné explicitně vyjadřovat derivace funkce  $f = f(x, y(x))$  a počítat jejich hodnoty. Hledaná aproximace je kombinací několika hodnot funkce  $f$  vypočítaných v několika strategicky volených bodech  $(x, y)$  na intervalu  $(x_k, x_{k+1})$ .

**Poznámka:** Těchto metod je velké množství!

**Heunova metoda** (Runge-Kuttova metoda 2. řádu)

vztah  $y' = f(x, y(x))$  zintegrujeme přes interval  $(x_k, x_{k+1})$

$$\int_{x_k}^{x_{k+1}} y'(x) dx = \int_{x_k}^{x_{k+1}} f(x, y(x)) dx$$

$$y(x_{k+1}) - y(x_k) = \int_{x_k}^{x_{k+1}} f(x, y(x)) dx$$

použijeme **lichoběžníkové pravidlo**

$$y(x_{k+1}) - y(x_k) = \frac{h}{2} [f(x_k, y(x_k)) + f(x_{k+1}, y(x_{k+1}))] + O(h^3)$$

viz chyba lich. pr.

na pravé straně vystupuje hodnota  $y(x_{k+1})$ , její aproximaci určíme pomocí **Eulerovy metody**

$$\bar{y}(x_{k+1}) = y(x_k) + h f(x_k, y(x_k)) + O(h^2)$$

dostáváme metodu ve tvaru

Numerické metody  
Josef Daněk  
KMA/NM  
13.2.2013

**Věta**  
Nechť oblouk  $M_0 M_1$  je částí paraboly. Potom platí:  
1. Tečna v bodě  $P$  je rovnoběžná s tětivou  $M_0 M_1$ .  
2. Směrnice tětivy  $M_0 M_1$  je aritmetickým průměrem směrnic tečen v  $M_1$  a  $M_2$ .

**Důkaz:**  
Rovnice paraboly (polynomu 2.stupně):  $y - b = c(x - a)^2$

$$y = c(x - a)^2 + b$$

$$\Rightarrow$$

$$y' = 2c(x - a)$$

1. Směrnice tečny v bodě  $P$ :

$$y'\left(\frac{x_0 + x_1}{2}\right) = 2c\left(\frac{x_0 + x_1}{2} - a\right) = c(x_0 + x_1 - 2a)$$

Směrnice tětivy  $M_0 M_1$  je:

$$\frac{y(x_1) - y(x_0)}{x_1 - x_0} = \frac{c(x_1 - a)^2 + b - c(x_0 - a)^2 - b}{x_1 - x_0}$$

$$= \frac{cx_1^2 - 2acx_1 + a^2c + b - cx_0^2 + 2acx_0 - a^2c - b}{x_1 - x_0}$$

$$= c \left( \frac{x_1^2 - x_0^2 - 2a(x_1 - x_0)}{x_1 - x_0} \right) = \underline{c(x_1 + x_0 - 2a)}$$

2. Směrnice tečny v bodě  $M_0$  je:

$$y'(x_0) = 2c(x_0 - a)$$

Směrnice tečny v bodě  $M_1$  je:

$$y'(x_1) = 2c(x_1 - a)$$

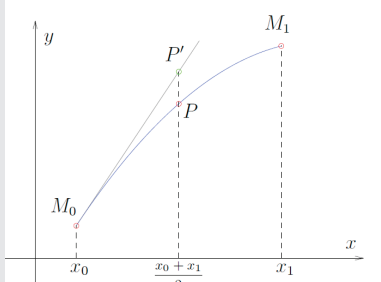
Jejich aritmetický průměr:

$$\frac{y'(x_0) + y'(x_1)}{2} = \frac{2c(x_0 - a) + 2c(x_1 - a)}{2}$$

$$= c(x_0 - a + x_1 - a) = \underline{c(x_0 + x_1 - 2a)}$$

Nyní použijeme vlastnost 1.

Známe souřadnice bodu  $M_0$ . Jestliže bychom znali  $y$ -souřadnici bodu  $P$ , pak stačí udělat tečnu a bodem  $M_0$  vést rovnoběžku a dostaneme  $y$ -souřadnici bodu  $M_1$ . My ale  $y$ -souřadnici bodu  $P$  neznáme, takže ji vyjádříme přibližně. Bod  $P$  nahradíme bodem  $P'$ , který má stejnou  $x$ -ovou souřadnici a leží na tečně k  $M_0$ .



$P'$  má souřadnice:

$$\left[ x_0 + \frac{h_0}{2}, y_0 + \frac{h_0}{2} \cdot \frac{f(x_0, y_0)}{y_0(x_0)} \right]$$

Tečna v bodě  $P'$  má směrnici:

$$y'(x_0 + \frac{h_0}{2}), \text{ tj.}$$

$$y'(x_0 + \frac{h_0}{2}) =$$

$$f(x_0 + \frac{h_0}{2}, y_0 + \frac{h_0}{2} \cdot \frac{k_1}{f(x_0, y_0)})$$

Stejnou směrnici by však měla mít i tětiva  $M_0M_1 \Rightarrow$  souřadnice bodu  $M_1$  jsou:

$$x_1 = x_0 + h_0$$

$$y_1 = y_0 + h_0 \cdot \frac{k_2}{2}$$

Tyto vztahy lze přepsat do tvaru (obecně)

$$k_1 = f(x_k, y_k)$$

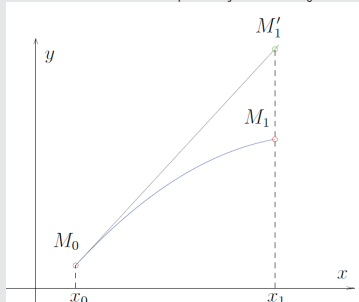
$$k_2 = f(x_k + \frac{h_k}{2}, y_k + \frac{h_k}{2} \cdot k_1)$$

$$y_{k+1} = y_k + h_k \cdot k_2$$

modifikovaná Eulerova metoda

Nyní použijeme vlastnost 2.

Známe souřadnice bodu  $M_0$ . Protože neznáme  $y$ -souřadnici bodu  $M_1$ , nahradíme ho bodem  $M'_1$ , který má stejnou  $x$ -souřadnici a leží na tečně procházející bodem  $M_0$ .



$M'_1$  má souřadnice:

$$\left[ x_0 + h_0, y_0 + h_0 \cdot \frac{f(x_0, y_0)}{y_0(x_0)} \right]$$

Směrnice tečny v  $M'_1$  je:

$$f(x_0 + h_0, y_0 + h_0 \cdot \frac{k_2}{f(x_0, y_0)})$$

Bod  $M'_1$  dostaneme z podmínky, že směrnice tečny  $M_0M_1$  je aritmetickým průměrem směrnic tečen v  $M_0$  a  $M'_1$ , tj.

$M'_1$  má souřadnice:

$$x_1 = x_0 + h_0$$

$$y_1 = y_0 + h_0 \cdot \frac{1}{2}(k_1 + k_2)$$

Obecně:

$$k_1 = f(x_k, y_k)$$

$$k_2 = f(x_k + \frac{h_k}{2}, y_k + \frac{h_k}{2} \cdot k_1)$$

$$y_{k+1} = y_k + h_k \cdot \frac{(k_1 + k_2)}{2}$$

Heunova metoda

Poznámka: Obě tyto metody jsou 2.řádu (aproximovali jsme parabolou).

#### Klasická Runge-Kuttova metoda 4. řádu

- jedna z nejvíce používaných metod tohoto typu
- předpis metody:

$$k_1 = f(x_k, y_k)$$

$$k_2 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2} \cdot k_1)$$

$$k_3 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2} \cdot k_2)$$

$$k_4 = f(x_k + h, y_k + h \cdot k_3)$$

$$y_{k+1} = y_k + \frac{h}{6} \cdot (k_1 + 2k_2 + 2k_3 + k_4)$$

Poznámka: Lokální diskretizační chyba, tj. chyba jednoho kroku metody, je  $d_k = O(h^5)$ . Globální chyba je potom o řád nižší, tj.  $e_k = O(h^4)$ , protože chyba metody se zvětšuje lineárně s počtem kroků  $k \sim \frac{1}{h}$ .

#### Příklad

Pomocí Heunovy metody, modifikované Eulerovy metody a klasické Runge-Kuttovy metody 4. řádu řešte následující úlohu na intervalu  $(0; 0,6)$  s konstantním krokem  $h = 0,2$

$$y' = x - y,$$

$$y(0) = 1$$

Řešení:

(Přesné řešení:  $y(x) = 2e^{-x} + x - 1$ ).

Předpis pro Heunovu metodu:

$$k_1 = f(x_k, y_k)$$

$$k_2 = f(x_k + h, y_k + h \cdot k_1)$$

$$y_{k+1} = y_k + h \cdot \frac{k_1 + k_2}{2}$$

$x_k$	$y(x_k)$	$y_k$	$k_1$	$k_2$	$h \cdot \frac{k_1 + k_2}{2}$	$e_k$
0	1,000	1,000	-1,000	-0,600	-0,160	0,000
0,2	0,837	0,840	-0,640	-0,312	-0,095	-0,003
0,4	0,741	0,745	-0,345	-0,076	-0,042	-0,004
0,6	0,698	0,703				-0,005

Předpis pro modifikovanou Eulerovu metodu:

$$k_1 = f(x_k, y_k)$$

$$k_2 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2} \cdot k_1)$$

$$y_{k+1} = y_k + h \cdot k_2$$

$x_k$	$y(x_k)$	$y_k$	$k_1$	$k_2$	$h \cdot k_2$	$e_k$
0	1,000	1,000	-1,000	-0,800	-0,160	0,000
0,2	0,837	0,840	-0,640	-0,476	-0,095	-0,003
0,4	0,741	0,745	-0,345	-0,210	-0,042	-0,004
0,6	0,698	0,703				-0,005

Poznámka:

Vidíme, že výsledky Heunovy i modifikované Eulerovy metody odpovídají výsledkům získaným metodou Taylorova typu 2. řádu (uvedené metody jsou 2. řádu).

Předpis pro klasickou Runge-Kuttovou metodu 4. řádu:

$$k_1 = f(x_k, y_k)$$

$$k_2 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2} \cdot k_1)$$

$$k_3 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2} \cdot k_2)$$

$$k_4 = f(x_k + h, y_k + h \cdot k_3)$$

$$y_{k+1} = y_k + \frac{h}{6} \cdot (k_1 + 2k_2 + 2k_3 + k_4)$$

$x_k$	$y(x_k)$	$y_k$	$k_1$	$k_2$	$k_3$	$k_4$	$e_k$
0	1,00000000	1,00000000	-1,000000	-0,800000	-0,820000	-0,636000	0,00000000
0,2	0,83746150	0,83746666	-0,637466	-0,437320	-0,490094	-0,339447	0,00000516
0,4	0,74064009	0,74064854	-0,340648	-0,206583	-0,219990	-0,096650	0,00000845
0,6	0,69762327	0,69763364					0,00001037

Několik otázek k zamýšlení:

1. Uveďte příklad počáteční úlohy pro obyčejnou diferenciální rovnici 1. řádu (s nenulovým řešením), pro kterou budou výsledky Eulerovy metody totožné s výsledky metody Taylorova typu 2. řádu.
2. Uveďte příklad počáteční úlohy pro obyčejnou diferenciální rovnici 1. řádu (s nenulovým řešením), pro kterou bude metoda Taylorova typu 2. řádu totožná s metodou Taylorova typu 3. řádu, ale různá od Eulerovy metody.
3. Uveďte příklad počáteční úlohy pro obyčejnou diferenciální rovnici 1. řádu (s nenulovým řešením), pro kterou bude modifikovaná Eulerova metoda totožná s Heunovou metodou, ale různá od Eulerovy metody.

## Kapitola 12. Počáteční úlohy pro ODR - II

### Víceokrové metody

V případě **jednokrokových** metod vystupovaly ve formuli pouze hodnoty  $y_k, y_{k+1}$ .

V případě **víceokrových** metod vypočítáváme hodnotu  $y_{k+1}$  pomocí hodnot

$$y_{k-2}, y_{k-1}, \dots, y_{k-1}, y_k, (y_{k+1})$$

**Poznámka:** Pokud nepoužijeme hodnotu  $y_{k+1}$ , jedná se o **explicitní metody**, v opačném případě mluvíme

o **implicitních** metodách.

Opět vyjďeme z rovnosti

$$y' = f(x, y(x))$$

Musí tedy platit i rovnost integrálů

$$\int_{x_k}^{x_{k+1}} y'(x) dx = \int_{x_k}^{x_{k+1}} f(x, y(x)) dx$$

Tedy

$$y(x_{k+1}) = y(x_k) + \int_{x_k}^{x_{k+1}} f(x, y(x)) dx \quad (*)$$

Dále postupujeme tak, že funkci  $g(x)$  aproximujeme interpolačním polynomem  $G_n(x)$ , který zintegrujeme přesně.

**Poznámka:** Připomeňme si odvození jednokrokové Eulerovy metody.

Funkci  $g(x)$  ze vztahu (\*) aproximujeme konstantní funkcí  $G_0(x)$

a)

$$G_0(x) = g(x_k), \quad x \in (x_k, x_{k+1})$$

Dostáváme:

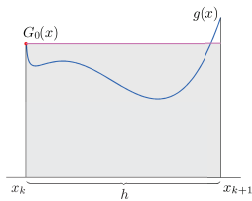
$$y_{k+1} = y_k + \int_{x_k}^{x_{k+1}} G_0(x) dx$$

$$y_{k+1} = y_k + hg(x_k)$$

$$y_{k+1} = y_k + hf(x_k, y_k)$$

### Eulerova metoda

... explicitní jednokroková metoda, řád 1



b)

$$G_0(x) = g(x_{k+1}), \quad x \in (x_k, x_{k+1})$$

Dostáváme:

$$y_{k+1} = y_k + \int_{x_k}^{x_{k+1}} G_0(x) dx$$

$$y_{k+1} = y_k + hg(x_{k+1})$$

$$y_{k+1} = y_k + hf(x_{k+1}, y_{k+1})$$

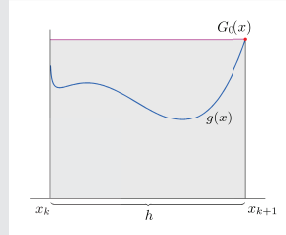
### Implicitní Eulerova metoda

... implicitní jednokroková metoda, řád 1

**Poznámka:**

Při použití implicitní metody je třeba zvolit počáteční aproximaci  $y_{k+1}^0$  a dále realizovat iterační proces

$$y_{k+1}^{i+1} = y_k + hf(x_{k+1}, y_{k+1}^i)$$



### Odvození dvoukrokových metod

Funkci  $g(x)$  ze vztahu (\*) aproximujeme **lineární** funkcí  $G_1(x)$

a)

$$G_1(x) = g(x_k) + \frac{g(x_k) - g(x_{k-1})}{h}(x - x_k), \quad x \in (x_k, x_{k+1})$$

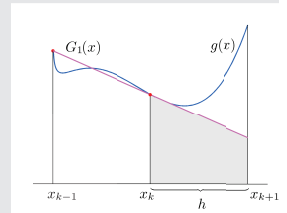
$$\int_{x_k}^{x_{k+1}} G_1(x) dx = g(x_k)h + \frac{h}{2}[g(x_k) - g(x_{k-1})] = \frac{h}{2}[3g(x_k) - g(x_{k-1})]$$

$$y_{k+1} = y_k + \frac{h}{2}[3f(x_k, y_k) - f(x_{k-1}, y_{k-1})]$$

### Adams-Bashforthova metoda

... explicitní dvoukroková metoda, řád 2

b)



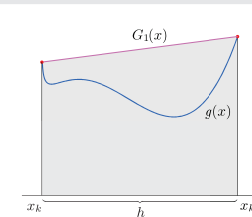
$$G_1(x) = g(x_k) + \frac{g(x_{k+1}) - g(x_k)}{h}(x - x_k), \quad x \in (x_k, x_{k+1})$$

$$\int_{x_k}^{x_{k+1}} G_1(x) dx = g(x_k)h + \frac{h}{2}[g(x_{k+1}) - g(x_k)] = \frac{h}{2}[g(x_k) + g(x_{k+1})]$$

$$y_{k+1} = y_k + \frac{h}{2}[f(x_k, y_k) + f(x_{k+1}, y_{k+1})]$$

### Adams-Moultonova metoda

... implicitní dvoukroková metoda, řád 2



### Odvození tříkrokových metod

Funkci  $g(x)$  ze vztahu (\*) aproximujeme kvadratickou funkcí  $G_2(x)$

a)

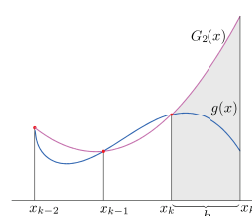
$$G_2(x) = \dots \text{polynom procházející body } [x_{k-2}, g(x_{k-2})], [x_{k-1}, g(x_{k-1})], [x_k, g(x_k)]$$

$$\int_{x_k}^{x_{k+1}} G_2(x) dx = \dots \text{D.cv.}$$

$$y_{k+1} = y_k + \frac{h}{12}[23f(x_k, y_k) - 16f(x_{k-1}, y_{k-1}) + 5f(x_{k-2}, y_{k-2})]$$

### Adams-Bashforthova metoda

... explicitní tříkroková metoda, řád 3



b)

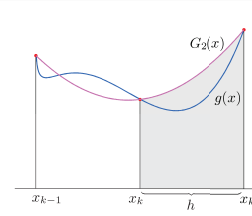
$$G_2(x) = \dots \text{polynom procházející body } [x_{k-1}, g(x_{k-1})], [x_k, g(x_k)], [x_{k+1}, g(x_{k+1})]$$

$$\int_{x_k}^{x_{k+1}} G_2(x) dx = \dots \text{D.cv.}$$

$$y_{k+1} = y_k + \frac{h}{12}[5f(x_{k+1}, y_{k+1}) + 8f(x_k, y_k) - f(x_{k-1}, y_{k-1})]$$

### Adams-Moultonova metoda

... implicitní tříkroková metoda, řád 3



**Poznámky:**

(i) U  $n$ -krokových metod je třeba znát  $n$  hodnot

$$y_{k-n+1}, y_{k-n}, \dots, y_k$$

Na začátku výpočtu však tyto hodnoty, tj.  $Y_1, \dots, Y_{n-1}$ , nejsou známy.

Pro jejich výpočet je třeba užít explicitní jednokrokové metody odpovídajícího řádu.

(ii) U implicitních metod je třeba určit aproximaci  $y_{k+1}^0$  a realizovat metodu prosté iterace

$$y_{k+1}^{i+1} = y_k + \dots y_{k+1}^i$$

### Obecný zápis metod

Víceokrovou (i jednokrokovou) metodu lze obecně zapsat ve tvaru

$$\sum_{j=0}^r \alpha_j y_{k+j} = h \sum_{j=0}^r \beta_j f(x_{k+j}, y_{k+j}) \approx y'(x_{k+1})$$

• **Explicitní Eulerova metoda** ( $\alpha_0 = -1, \alpha_1 = 1, \beta_0 = 1, \beta_1 = 0$ )

$$y_{k+1} - y_k = hf(x_k, y_k)$$

• **Implicitní Eulerova metoda** ( $\alpha_0 = -1, \alpha_1 = 1, \beta_0 = 0, \beta_1 = 1$ )

$$y_{k+1} - y_k = hf(x_{k+1}, y_{k+1})$$

• **Adams-Bashforthova metoda - dvoukroková**

$$(\alpha_0 = 0, \alpha_1 = -1, \alpha_2 = 1, \beta_0 = \frac{1}{2}, \beta_1 = \frac{3}{2}, \beta_2 = 0)$$

$$y_{k+2} - y_{k+1} = \frac{h}{2}[3f(x_{k+1}, y_{k+1}) - f(x_k, y_k)] \quad (k := k+1)$$

• **Adams-Moultonova metoda - dvoukroková**

$$(\alpha_0 = -1, \alpha_1 = 1, \beta_0 = \frac{1}{2}, \beta_1 = \frac{1}{2})$$

$$y_{k+1} - y_k = \frac{h}{2}[f(x_k, y_k) + f(x_{k+1}, y_{k+1})]$$

• **Adams-Bashforthova metoda - tříkroková**

$$(\alpha_0 = 0, \alpha_1 = 0, \alpha_2 = -1, \alpha_3 = 1, \beta_0 = \frac{5}{12}, \beta_1 = -\frac{4}{3}, \beta_2 = \frac{23}{12})$$

$$y_{k+3} - y_{k+2} = \frac{h}{12} [23f(x_{k+2}, y_{k+2}) - 16f(x_{k+1}, y_{k+1}) + 5f(x_k, y_k)] \quad (k := k+2)$$

• Adams-Moultonova metoda - tříčlenná

$$(\alpha_0 = 0, \alpha_1 = -1, \alpha_2 = 1, \beta_0 = -\frac{1}{12}, \beta_1 = \frac{2}{3}, \beta_2 = \frac{5}{12})$$

$$y_{k+2} - y_{k+1} = \frac{h}{12} [5f(x_{k+2}, y_{k+2}) + 8f(x_{k+1}, y_{k+1}) - f(x_k, y_k)] \quad (k := k+1)$$

Definice: Lokální diskretizační chybou metody rozumíme

$$\tau_k = \frac{1}{h} \left[ \sum_{j=0}^r \alpha_j y(x_{k+j}) - h \sum_{j=0}^r \beta_j y'(x_{k+j}) \right]$$

Definice: Řekneme, že metoda je **konzistentní**, je-li splněna podmínka

$$\tau_k(h) \rightarrow 0 \quad \text{pro } h \rightarrow 0$$

Použijeme-li Taylorův rozvoj, získáme:

$$\begin{aligned} y(x_k) &= y(x_k) \\ y(x_{k+1}) &= y(x_k) + hy'(x_k) + \frac{1}{2}h^2 y''(x_k) + \dots \\ y(x_{k+2}) &= y(x_k) + 2hy'(x_k) + \frac{1}{2}(2h)^2 y''(x_k) + \dots \\ y(x_{k+3}) &= y(x_k) + 3hy'(x_k) + \frac{1}{2}(3h)^2 y''(x_k) + \dots \\ &\vdots \\ y(x_{k+j}) &= y(x_k) + jhy'(x_k) + \frac{1}{2}(jh)^2 y''(x_k) + \dots \end{aligned}$$

a analogicky pro derivaci (formálně připsáme '):

$$\begin{aligned} y'(x_k) &= y'(x_k) \\ y'(x_{k+1}) &= y'(x_k) + hy''(x_k) + \frac{1}{2}h^2 y'''(x_k) + \dots \\ y'(x_{k+2}) &= y'(x_k) + 2hy''(x_k) + \frac{1}{2}(2h)^2 y'''(x_k) + \dots \\ y'(x_{k+3}) &= y'(x_k) + 3hy''(x_k) + \frac{1}{2}(3h)^2 y'''(x_k) + \dots \\ &\vdots \\ y'(x_{k+j}) &= y'(x_k) + jhy''(x_k) + \frac{1}{2}(jh)^2 y'''(x_k) + \dots \end{aligned}$$

Tj. jde vždy polynom v proměnné  $h$  ( $\Rightarrow i \tau_k$ ).

Dosažením do vztahu pro lokální diskretizační chybu

$$\tau_k = \frac{1}{h} \left[ \sum_{j=0}^r \alpha_j y(x_{k+j}) - h \sum_{j=0}^r \beta_j y'(x_{k+j}) \right]$$

tj.

$$\tau_k = \frac{1}{h} \left[ (\alpha_0 y(x_k) + \alpha_1 y(x_{k+1}) + \alpha_2 y(x_{k+2}) + \dots + \alpha_r y(x_{k+r})) - h (\beta_0 y'(x_k) + \beta_1 y'(x_{k+1}) + \dots + \beta_r y'(x_{k+r})) \right],$$

dostaneme

$$\tau_k = \frac{1}{h} \left[ \underbrace{y(x_k) \left( \sum_{j=0}^r \alpha_j \right)}_{\text{konstantní členy}} + \underbrace{hy'(x_k) \left( \sum_{j=0}^r (j\alpha_j - \beta_j) \right)}_{\text{lineární členy}} + \underbrace{h^2 y''(x_k) \left( \frac{1}{2} \sum_{j=0}^r j^2 \alpha_j - \sum_{j=0}^r j \beta_j \right)}_{\text{kvadratické členy}} + \underbrace{h^3 y'''(x_k) (\dots)}_{\text{kubické členy}} + \dots \right]$$

Po roznásobení:

$$\tau_k = \frac{1}{h} y(x_k) \left( \sum_{j=0}^r \alpha_j \right) + y'(x_k) \left( \sum_{j=0}^r (j\alpha_j - \beta_j) \right) + h(\dots) + h^2(\dots) + \dots$$

D.c.v. Ukažte, že dříve odvozené metody jsou konzistentní.

Definice: Polynomy v proměnných  $v$  a  $w$  ve tvaru

$$\rho(v) = \sum_{j=0}^r \alpha_j v^j \quad \text{a} \quad \sigma(w) = \sum_{j=0}^r \beta_j w^j$$

nazýváme **charakteristické polynomy** metody.

Poznámka: Metoda je konzistentní, platí-li

$$\rho(1) = 0 \quad \text{a} \quad \rho'(1) = \sigma(1)$$

Poznámka:

Nestabilita - do výsledku je vnášena chyba, jejíž vliv zesiluje až celý výpočet znehodnotí.

Příčiny - špatná podmíněnost úlohy (nezávisí na volbě metody); nevhodná metoda nebo příliš velký krok.

Definice: Mějme dánu počáteční úlohu s lipschitzovskou funkcí  $f$ :

$$\begin{aligned} y' &= f(x, y), \quad x \in (0, T) \\ y(0) &= \eta \end{aligned}$$

Řekneme, že metoda je **konvergentní**, když platí:

$$\lim_{\substack{h \rightarrow 0 \\ Nh = T}} y_N = y(T)$$

a

$$\lim_{h \rightarrow 0} y_k = \eta \quad \text{pro } k = 0, 1, \dots, r-1$$

pro každé pevné  $T$  (uvažujeme  $r$ -krokovou metodu).

Poznámka: Je zřejmé, že nutnou podmínkou konvergence je konzistence metody. Je i podmínkou postačující?

Věta Uvažujme úlohu

$$\begin{aligned} y' &= \lambda y, \quad x \in (0, T) \\ y(0) &= \eta \end{aligned}$$

Eulerova metoda je pro tuto úlohu konvergentní.

Důkaz:

$$\begin{aligned} y(x_{k+1}) &= y(x_k) + h \frac{y'(x_k)}{\lambda y(x_k)} + h\tau_k \\ y_{k+1} &= y_k + h \lambda y_k \end{aligned}$$

$$\frac{y(x_{k+1}) - y_{k+1}}{E_{k+1}} = \frac{y(x_k) - y_k}{E_k} + h \lambda \frac{(y(x_k) - y_k)}{E_k} + h\tau_k$$

$$E_{k+1} = E_k(1 + h\lambda) + h\tau_k$$

Tj.

$$\begin{aligned} E_1 &= E_0(1 + h\lambda) + h\tau_0 \\ E_2 &= E_1(1 + h\lambda) + h\tau_1 \\ &\vdots \end{aligned}$$

Obecně lze psát:

$$E_k = (1 + h\lambda)^k E_0 + h \sum_{m=1}^k (1 + h\lambda)^{k-m} \tau_{m-1}$$

Dále platí (z Taylorova rozvoje  $e^x$ ):

$$|1 + h\lambda| \leq e^{h|\lambda|}$$

a tedy

$$(1 + h\lambda)^{k-m} \leq e^{(k-m)h|\lambda|} \leq e^{kh|\lambda|} \leq e^{T|\lambda|}$$

Potom

$$|E_k| \leq e^{k|\lambda|} \left[ |E_0| + h \max_{1 \leq m \leq k} |\tau_{m-1}| \right]$$

$$(*) \quad |E_1| = |h\tau_0| \rightarrow 0 \quad \text{pro } h \rightarrow 0$$

$$(**) \rightarrow 0 \quad \text{pro } h \rightarrow 0, \text{ protože je Eulerova metoda konzistentní.}$$

Důkaz:

Platí

$$\begin{aligned} y(x_{k+1}) &= y(x_k) + hf(x_k, y(x_k)) + h\tau_k \\ y_{k+1} &= y_k + hf(x_k, y_k) \end{aligned}$$

$$\frac{y(x_{k+1}) - y_{k+1}}{E_{k+1}} = \frac{y(x_k) - y_k}{E_k} + h [f(x_k, y(x_k)) - f(x_k, y_k)] + h\tau_k$$

Funkce  $f$  je lipschitzovskou spojitá:

$$|f(x_k, y(x_k)) - f(x_k, y_k)| \leq L \cdot |y(x_k) - y_k| \quad \forall x_k \in (0, T)$$

Pak lze psát:

$$|E_{k+1}| \leq |E_k| + hL|E_k| + h|\tau_k|$$

Dále je důkaz stejný ( $|\lambda| = L$ )

$$|E_k| \leq e^{kL} \left[ |E_0| + h \max_{1 \leq m \leq k} |\tau_{m-1}| \right]$$

□

Konvergence vícečlenných metod

Příklad: Uvažujme vícečlennou metodu ve tvaru

$$y_{k+2} = 3y_{k+1} - 2y_k - hf(x_k, y_k)$$

Obecný zápis byl

$$\sum_{j=0}^r \alpha_j y_{k+j} = h \sum_{j=0}^r \beta_j f(x_{k+j}, y_{k+j})$$

Platí:

$$\alpha_0 = 2, \alpha_1 = -3, \alpha_2 = 1, \sum_{j=0}^2 \alpha_j = 0$$

$$\beta_0 = -1, \beta_1 = 0, \beta_2 = 0, \sum_{j=0}^2 \beta_j = -1 \stackrel{?}{=} -1 \stackrel{?}{=} \sum_{j=0}^2 j\alpha_j = 0 \cdot \alpha_0 + 1 \cdot \alpha_1 + 2 \cdot \alpha_2 = 0 + (-3) + 2 = -1$$

$\Rightarrow$  metoda je konzistentní.

Touto metodou budeme řešit počáteční úlohu

$$\begin{aligned} y' &= 0, \quad x \in (0, T) \\ y(0) &= 0 \end{aligned}$$

Pro tuto úlohu má metoda tvar

$$y_{k+2} = 3y_{k+1} - 2y_k$$

Obecně lze psát:

$$y_k = 2y_0 - y_1 + 2^k(y_1 - y_0)$$

Důkaz: (pomocí úplné matematické indukce)

1.  $k = 2, k = 3$

$$y_2 = 2y_0 - y_1 + 2^2(y_1 - y_0) = 3y_1 - 2y_0 \quad \checkmark$$

$$y_3 = 2y_0 - y_1 + 2^3(y_1 - y_0) = 7y_1 - 6y_0 \stackrel{?}{=} 3y_2 - 2y_1 = 3(3y_1 - 2y_0) - 2y_1 \quad \checkmark$$

$$\left. \begin{aligned} y_k &= 2y_0 - y_1 + 2^k(y_1 - y_0) \\ y_{k+1} &= 2y_0 - y_1 + 2^{k+1}(y_1 - y_0) \end{aligned} \right\} \Rightarrow y_{k+2} = 2y_0 - y_1 + 2^{k+2}(y_1 - y_0)$$

$$y_{k+2} = 3y_{k+1} - 2y_k = 3(2y_0 - y_1 + 2^{k+1}(y_1 - y_0)) - 2(2y_0 - y_1 + 2^k(y_1 - y_0)) =$$

$$= 2y_0 - y_1 + \underbrace{(6-2)2^k}_{=2^2}(y_1 - y_0) \quad \checkmark$$

**Problém:**

Pokud  $y_1 = y_0 = y(0) = 0 \Rightarrow y_k = 0 \forall k$ .  
 Pokud se  $y_1$  bude lišit (i když velmi málo) od 0, pak pro  $k \rightarrow \infty: y_k \rightarrow 0$ .

Považujeme-li rovnost  $y_{k+2} = 3y_{k+1} - 2y_k$  za diferenciální rovnici, můžeme ji řešit. Předpokládáme, že  $y_k = C \cdot v^k$ . Pak lze psát

$$\begin{aligned} C v^{k+2} &= 3C v^{k+1} - 2C v^k \\ C v^2 &= 3C v - 2C \\ v^2 - 3v + 2 &= 0 \\ v_{1,2} &= \frac{3 \pm \sqrt{9-8}}{2} \\ v_1 &= 2, \quad v_2 = 1 \\ y_k &= C_1 2^k - C_2 \end{aligned} \quad (*)$$

Dále víme, že pro

$$\begin{aligned} k=0: \quad y_0 &= C_1 + C_2 \\ k=1: \quad y_1 &= 2C_1 + C_2 \\ \Rightarrow C_1 &= y_1 - y_0 \\ C_2 &= y_0 - C_1 = y_0 - y_1 + y_0 = 2y_0 - y_1 \\ y_k &= (y_1 - y_0) 2^k + (2y_0 - y_1) \end{aligned}$$

Připomeňme, že víceokrovou metodu jsme zapisovali ve tvaru

$$\sum_{j=0}^r \alpha_j y_{k+j} = h \sum_{j=0}^r \beta_j f(x_{k+j}, y_{k+j})$$

a charakteristické polynomy jsme definovali

$$\rho(v) = \sum_{j=0}^r \alpha_j v^j \quad \text{a} \quad \sigma(w) = \sum_{j=0}^r \beta_j w^j$$

O stabilitě výpočtu rozhodují kořeny polynomu  $\rho(v)$  (viz (\*)).

Pro kořeny  $\vartheta_j$  polynomu  $\rho(v)$  musí platit  $|\vartheta_j| \leq 1$ .

Předpokládáme, že máme dostatečně přesně vypočítané hodnoty  $y_0, y_1, \dots, y_{k-1}$  nějakou explicitní jedнокrokovou metodou.

Nyní chceme počítat  $y_k$ .

- Nejprve nějakou explicitní metodou určíme nultou iteraci  $y_k^{[0]}$  jako vstupní hodnotu pro další výpočet (PREDIKTOR).
- Vypočteme hodnotu pravé strany  $F_k^{[s]} = f(x_k, y_k^{[s]})$ .
- Vypočteme lepší aproximaci  $y_k^{[s+1]}$  pomocí nějaké implicitní metody s využitím  $F_k^{[s]} := f_k$  (KOREKTOR).

Pomocí kroků 2) a 3) určíme  $N$  iterací  $y_k^{[1]}, y_k^{[2]}, \dots, y_k^{[N]}$  ( $N$  - dáno).

Na závěr přiřadíme  $y_k = y_k^{[N]}$ .

Stejný postup opakujeme pro  $y_{k+1}, y_{k+2}, \dots$ .

**Poznámka:** Dané schéma lze použít na různé metody. Je žádoucí použít explicitní a implicitní metodu stejného řádu (pro zachování přesnosti). Volba konkrétních metod je na nás.

**Poznámka:** Označíme-li operaci:

- P ... prediktor
- E ... vyčíslení (evaluation)
- C ... korektor

Můžeme toto schéma zapsat ve tvaru:

$F(EC)^N$  případně  $P(EC)^N E$ , vyčísľujeme-li ještě  $F_k = f(x_k, y_k^{[N]})$  (což je lepší). Dostaneme pak různé varianty tohoto schématu:

$$\begin{aligned} PEC &, PECE \\ P(EC)^2 &, P(EC)^2 E \\ P(EC)^3 &, P(EC)^3 E \\ &\vdots \end{aligned}$$

**Příklad:** Řešte algoritmem prediktor-korektor založeném na Adamsových metodách druhého řádu na intervalu  $(0; 0,6)$  počáteční úlohu:

$$\begin{aligned} y' &= y + e^x, \quad \text{tj. } f(x, y(x)) = y + e^x \\ y(0) &= -1 \end{aligned}$$

**Přesné řešení:**  $y = e^x(x-1)$ .

Použijeme algoritmus typu PEC.

Vzorec prediktoru má tvar:

$$y_{n+1}^{[0]} = y_n + \frac{h}{2}(3F_n - F_{n-1})$$

Korektor:

$$y_{n+1} = y_n + \frac{h}{2}(F_{n+1}^{[0]} + F_n)$$

Volte krok  $h = 0,2$

**Definice:** Řekneme, že metoda je  $D$ -stabilní, pokud kořeny charakteristického polynomu  $\rho(v)$  splňují podmínky:  
 (i)  $|\vartheta_j| \leq 1$  pro  $j = 1, 2, \dots, r$ ,  
 (ii) je-li  $\vartheta_j$  násobný kořen, potom  $|\vartheta_j| < 1$ .

**Poznámky:**

- Je-li metoda  $D$ -stabilní, nebude v průběhu výpočtu radikálně zvětšovat jedнокrokovou chybu.
- Uvažujme Eulerovu metodu  
 $y_{k+1} - y_k = h f(x_k, y_k)$   
 $\rho(v) = v - 1 = 0 \Rightarrow \vartheta = 1$   
 $\Rightarrow$  Eulerova metoda je  $D$ -stabilní.

**Odhad chyby metodou polovičního kroku**

Pro globální chybu metodu lze psát

$$y(x) = \underbrace{y(x, h)}_{\text{přesné}} + \underbrace{E_h}_{O(h^r)} + \underbrace{F_h}_{O(h^r)}, \quad r > p \quad (*)$$

(\*)  $y(x, h) = y_k(x, h)$ ,  $x \in (x_k, x_{k+1})$ ,  $k = 0, 1, \dots, N-1$   
 pro poloviční krok

$$y(x) = y(x, \frac{h}{2}) + E_{\frac{h}{2}} + F_{\frac{h}{2}} \quad (**)$$

Po odečtení (\*) - (\*\*):

$$\begin{aligned} 0 &= y(x, h) - y(x, \frac{h}{2}) + E_h - E_{\frac{h}{2}} + \dots \\ 0 &\approx y(x, h) - y(x, \frac{h}{2}) + (2^p - 1)E_{\frac{h}{2}} \\ E_{\frac{h}{2}} &= \frac{y(x, \frac{h}{2}) - y(x, h)}{2^p - 1} \\ \Rightarrow y(x) &\approx y(x, \frac{h}{2}) + E_{\frac{h}{2}} \end{aligned}$$

**Poznámka:** Opět lze použít Richardsonovu extrapolaci

- aktivní extrapolace  
 extrapolaci provádíme v každém kroku (extrapolované  $y_k$  použijeme pro výpočet  $y_{k+1}$ )
- pasivní extrapolace  
 vypočteme  $y_k$ ,  $k = 0, 1, \dots, N-1$  s různými parametry  $h$  potom provedeme extrapolaci

**Algoritmus prediktor-korektor**

**Poznámka:** Jde o obecné schéma výpočtu.

**Princip:**

$n$	$x_n$	$y(x_n)$	$y_n^{[0]}$	$F_n^{[0]}$	$y_n$	$e_n$
0	0	-1		0	-1	0
1	0,2	-0,9771		0,2425	-0,9789	0,0018
2	0,4	-0,8950	$P - 0,9061 \rightarrow E$	0,5857	$C - 0,8960$	0,0010
3	0,6	-0,7288	$P - 0,7445 \rightarrow E$	1,0776	$C - 0,7296$	0,0008

Pro určení hodnoty  $y_1$  použijeme např. jedнокrokovou modifikovanou Eulerovu metodu (2. řádu):

$$\begin{aligned} k_1 &= f(x_0, y_0) = y_0 + e^{x_0} = -1 + 1 = 0 \\ k_2 &= f(x_0 + h/2, y_0 + h/2 \cdot k_1) = -1 + e^{0,1} \approx 0,1051 \\ y_1 &= y_0 + h \cdot k_2 \approx -1 + 0,2 \cdot 0,1051 = -0,9789 \end{aligned}$$

Určíme hodnoty  $F_0$  a  $F_1$ .

**Odhad chyby pomocí algoritmu prediktor-korektor**

Za předpokladu, že se hodnota derivace  $y^{(p+1)}$ , kde  $p$  je řád metody, příliš nemění, lze odvodit odhad pro lokální chybu algoritmu

$$d_k \approx \frac{C_{p+1}^C}{C_{p+1}^P - C_{p+1}^E} (y_{k+1}^C - y_{k+1}^P)$$

kde

$C_{p+1}^P, C_{p+1}^E$  ... konstanty v lokální chybě metody, tj.  $d_k = c_{p+1} h^{p+1} y^{(p+1)}(x_k)$

$y_{k+1}^C$  ... vypočteno korektorem

$y_{k+1}^P$  ... vypočteno prediktorem

**Podmíněnost úlohy a stabilita metody**

**Příklad** Řešte počáteční úlohu

$$\begin{aligned} y' &= y - \frac{x}{3} - \frac{2}{3}, \quad x \in (0, T) \\ y(0) &= 1 \end{aligned}$$

řešení této počáteční úlohy má tvar  $y = \frac{x}{3} + 1$

obecné řešení dané rovnice je  $y = Ae^x + \frac{x}{3} + 1$

⇒ úloha je špatně podmíněná!

$$(y(0) = 1 + \varepsilon \rightarrow y = \varepsilon e^x + \frac{1}{3} + 1)$$

pro řešení je třeba použít metodu vyššího řádu a dostatečně přesnou aritmetiku

**Příklad** Pomocí Eulerovy metody řešíme počáteční úlohu

$$\begin{cases} y' = \lambda y, & x \in (0, T) \\ y(0) = 1 \end{cases}$$

(\*)

přesné řešení úlohy je  $y(x) = e^{\lambda x}$

v tomto případě má Eulerova metoda tvar

$$y_{k+1} = y_k + h\lambda y_k$$

$$y_{k+1} = \left(1 + \frac{h\lambda}{1}\right) y_k$$

Je-li  $|1 + h\lambda| < 1$ , pak je posloupnost  $y_k$  omezená a klesající.

Je-li  $|1 + h\lambda| > 1$ , pak posloupnost  $y_k$  neomezeně roste (osciluje).

$$|1 + h\lambda| < 1 \Leftrightarrow \bar{h} = h\lambda \in (-2, 0)$$

**Pro konkrétní úlohu:**

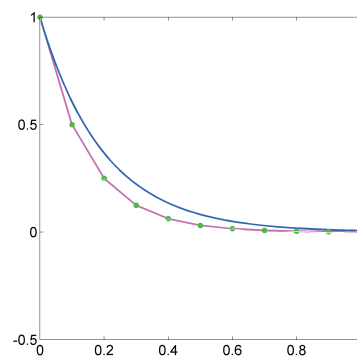
$$\begin{cases} y' = -5y, & x \in (0, 1) \\ y(0) = 1 \end{cases}$$

a krok  $h$  pomocí Eulerovy metody dostaneme:

1)  $h = 0,1$

$$y_{k+1} = \underbrace{(1 - 0,1 \cdot 5)}_{= 0,5} y_k$$

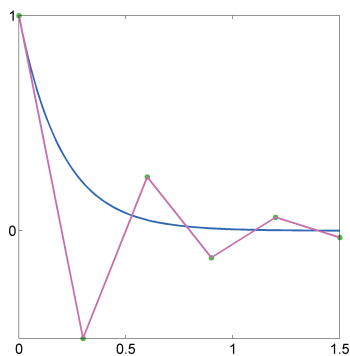
$x_k$	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
$y_k$	1,0000	0,5000	0,2500	0,1250	0,0625	0,0313	0,0156	0,0078	0,0039	0,0020	0,0010



2)  $h = 0,3$

$$y_{k+1} = \underbrace{(1 - 0,3 \cdot 5)}_{= -0,5} y_k$$

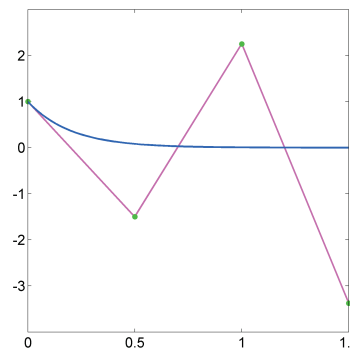
$x_k$	0	0,3	0,6	0,9	1,2	1,5
$y_k$	1,0000	-0,5000	0,2500	-0,1250	0,0625	-0,0313



3)  $h = 0,5$

$$y_{k+1} = \underbrace{(1 - 0,5 \cdot 5)}_{= -1,5} y_k$$

$x_k$	0	0,5	1,0	1,5
$y_k$	1,0000	-1,5000	2,2500	-3,3750



Tj. v případě velké záporné hodnoty  $\lambda = -2000 \rightarrow h < \frac{2}{2000} = 0,0001$

řešení  $y(x) = e^{-2000x} \rightarrow 0$  pro  $x \rightarrow \infty$

$y_k = (1 + h\lambda)^k y_0 = (1 - 2000h)^k \rightarrow 0$  pro  $k \rightarrow \infty$

**Poznámka:**

Řekneme, že metoda je pro  $\bar{h}$  **absolutně stabilní**, jestliže při  $h = \bar{h}$ , všechna přibližná řešení mají pro  $k \rightarrow \infty$  limitu rovnou 0 ( $y_k \rightarrow 0$ ).

Úlohu (\*) uvažujeme proto, že rovnice  $y' = f(x, y)$  po linearizování přejde na tvar  $y' = \frac{\partial f}{\partial y} y = \lambda$

⇒ Stabilita závisí jak na metodě, tak na úloze.

Připomeňme, že platí:

$$y(x_{k+1}) = y(x_k) + h\lambda y(x_k) + h\tau_k$$

$$y_{k+1} = y_k + h\lambda y_k$$

$$E_{k+1} = E_k + h\lambda E_k + h\tau_k$$

$$|E_{k+1}| \leq |1 + h\lambda| \cdot |E_k| + h|\tau_k|$$

Chceme-li, aby  $|E_k| \rightarrow 0$  pro  $k \rightarrow \infty$ , musíme požadovat

$$|1 + h\lambda| < 1$$

Tuto úvahu můžeme učinit i pro obecnou víceřadkovou metodu

$$\sum_{j=0}^r \alpha_j y_{k+j} = h \sum_{j=0}^r \beta_j \lambda y_{k+j}$$

$$\sum_{j=0}^r (\alpha_j - \bar{h} \beta_j) y_{k+j} = 0$$

Definujeme **polynom stability**:

$$\Pi(u, \bar{h}) = \sum_{j=0}^r (\alpha_j - \bar{h} \beta_j) u^j$$

**Definice:** **Oblastí absolutní stability metody** nazýváme množinu

$$\mathcal{A} = \{ \bar{h} \in \mathbb{C} : |a_j| \leq 1 \quad \forall a_j : \Pi(u, \bar{h}) = 0 \\ |a_j| < 1 \text{ pro násobné kořeny} \}$$

tj. „množina hodnot  $\bar{h}$  v komplexní rovině, pro které kořeny polynomu  $\Pi(u, \bar{h})$  splňují podmínku  $|a_j| < 1$ “

**Příklady**

1. explicitní Eulerova metoda

$$y_{k+1} - y_k = hf(x_k, y_k)$$

Koeficienty metody

$$\alpha_0 = -1, \alpha_1 = 1, \beta_0 = 1, \beta_1 = 0$$

Polynom stability

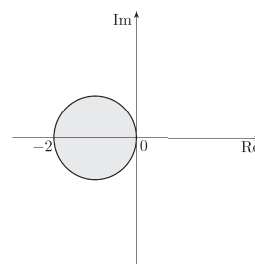
$$\Pi(u, \bar{h}) = \sum_{j=0}^1 (\alpha_j - \bar{h} \beta_j) u^j$$

$$\Pi(u, \bar{h}) = (-1 - \bar{h}) + u = u - 1 - \bar{h}$$

Kořen

$$u = 1 + \bar{h}; \quad |a| = |1 + \bar{h}| \leq 1$$

Oblast absolutní stability metody



2. implicitní Eulerova metoda

$$y_{k+1} - y_k = hf(x_{k+1}, y_{k+1})$$

Koeficienty metody

$$\alpha_0 = -1, \alpha_1 = 1, \beta_0 = 0, \beta_1 = 1$$

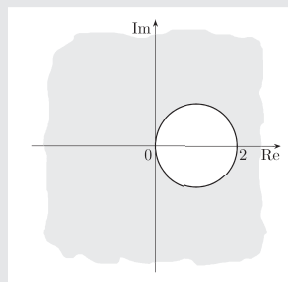
Polynom stability

$$\Pi(u, \bar{h}) = -1 + (1 - \bar{h})u = (1 - \bar{h})u - 1$$

Kořen

$$|a| = \left| \frac{1}{1 - \bar{h}} \right| \leq 1; \quad |1 - \bar{h}| \geq 1$$

Oblast absolutní stability metody



Intervaly absolutní stability

Eulerova metoda  $(-2, 0)$

Implicitní Eulerova metoda  $(-\infty, 0) \cup (2, \infty)$

**Příklad** Stanovte oblast absolutní stability pro tzv. obdélníkové pravidlo, tj. metodu s předpisem

$$y_{k+1} = y_{k-1} + 2hf(x_k, y_k)$$

Koeficienty metody

$$\alpha_0 = -1, \alpha_1 = 0, \alpha_2 = 1, \beta_0 = 0, \beta_1 = 2, \beta_2 = 0$$

Polynom stability

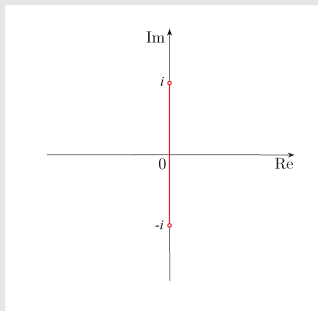
$$\Pi(u, \bar{h}) = -1 - 2\bar{h}u + u^2$$

Kořeny

$$u_{1,2} = \frac{2\bar{h} \pm \sqrt{4\bar{h}^2 + 4}}{2} = \bar{h} \pm \sqrt{\bar{h}^2 + 1}$$

Pro oblast absolutní stability musí platit (D.cv.)

$$|u_1| < 1 \quad \wedge \quad |u_2| < 1$$



**Příklad** Stanovte oblast absolutní stability pro tzv. lichoběžníkové pravidlo, tj. metodu s předpisem

$$y_{k+1} = y_k + \frac{h}{2} [f(x_k, y_k) + f(x_{k+1}, y_{k+1})]$$

Koeficienty metody

$$\alpha_0 = -1, \alpha_1 = 1, \beta_0 = \frac{1}{2}, \beta_1 = \frac{1}{2}$$

Polynom stability

$$\Pi(u, \bar{h}) = (-1 - \frac{1}{2}\bar{h}) + (1 - \frac{1}{2}\bar{h})u$$

Kořen

$$u = \frac{2 + \bar{h}}{2 - \bar{h}}$$

Platí

$$|u| < 1 \quad \text{pro} \quad \text{Re} \bar{h} < 0$$

Neboť

$$\left| \frac{2 + \bar{h}}{2 - \bar{h}} \right| < 1 \Leftrightarrow |2 + \bar{h}| < |2 - \bar{h}| \Leftrightarrow |\bar{h} - (-2)| < |\bar{h} - 2|,$$

tj. vzdálenost  $\bar{h}$  od  $-2$  je menší než vzdálenost od  $2$ .

Oblast absolutní stability metody

