Výpočty a zaokrouhlovací chyby aneb „Jáchyme, hoď ho do stroje“

Petr Tichý

19. března 2014

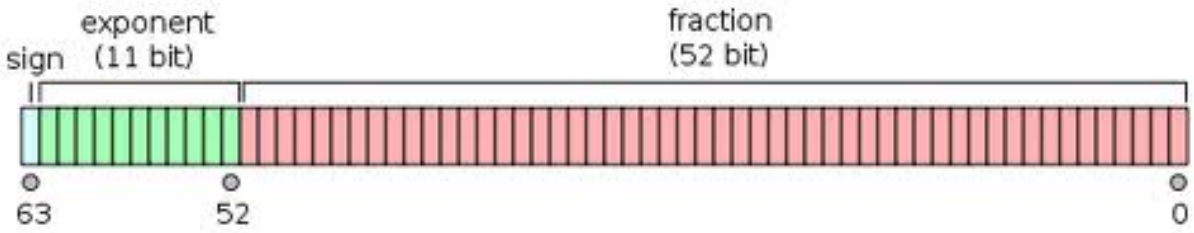
Přehled

Matematický popis a výpočet v počítači

Matematika, spojitost, ∞.	$\leftrightarrow$	Počítač, diskrétní hodnoty, konečno.

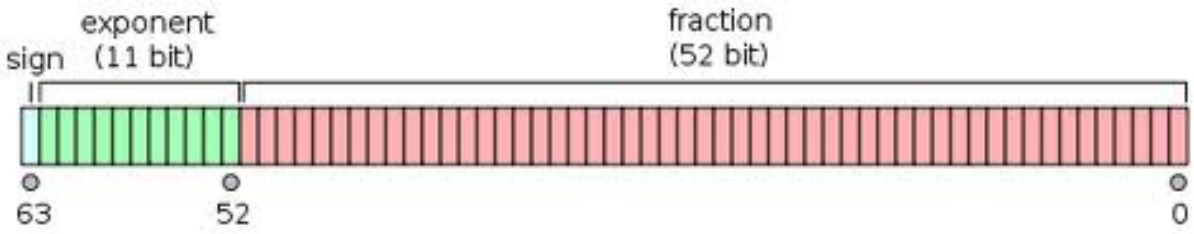
Čísla v počítači, pravidla

- **Reprezentace** čísla v počítači



Čísla v počítači, pravidla

- **Reprezentace** čísla v počítači



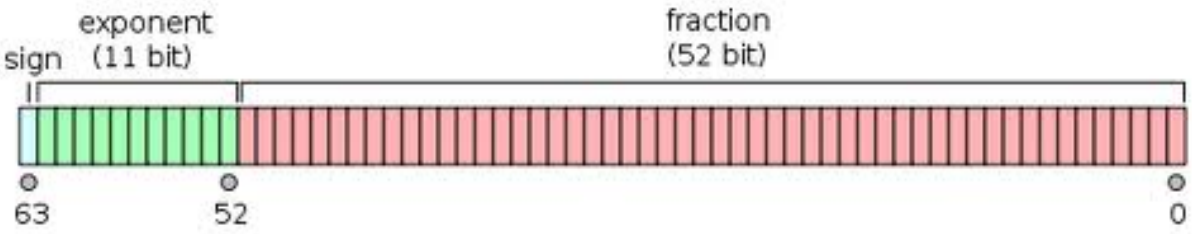
- **Standardní model konečné aritmetiky** předpokládá

$$\mathrm{fl}(x \circ y) = (x \circ y)(1+\epsilon), \qquad |\epsilon| \leq \mathbf{u},$$

kde $\circ$ zastupuje operace $+, -, *, /$ a $\mathrm{fl}(\cdot)$ označuje, že výpočet byl proveden v konečné aritmetice počítače.

Čísla v počítači, pravidla

- **Reprezentace** čísla v počítači



- **Standardní model konečné aritmetiky** předpokládá

$$\mathrm{fl}(x \circ y) = (x \circ y)(1+\epsilon), \qquad |\epsilon| \leq \mathbf{u},$$

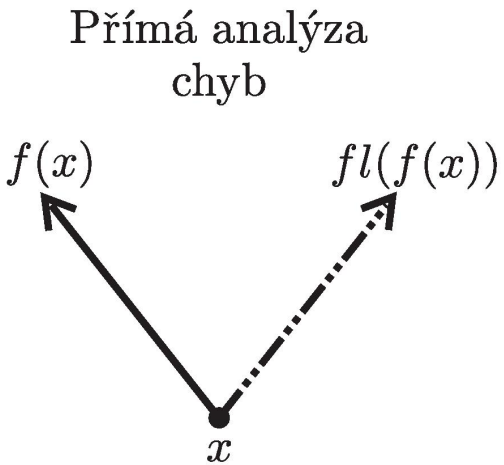
kde $\circ$ zastupuje operace $+, -, *, /$ a $\mathrm{fl}(\cdot)$ označuje, že výpočet byl proveden v konečné aritmetice počítače.

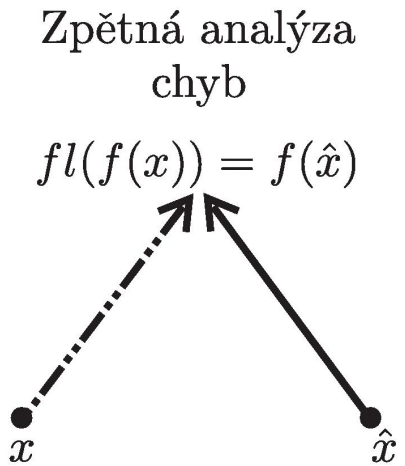
- Snaha o pochopení chování algoritmů v počítači

  numerická (numerus = číslo, vyčíslení pomocí čísel)
  analýza (analyzovat, pochopit, rozebrat do detailů).

Přímá a zpětná analýza chyb

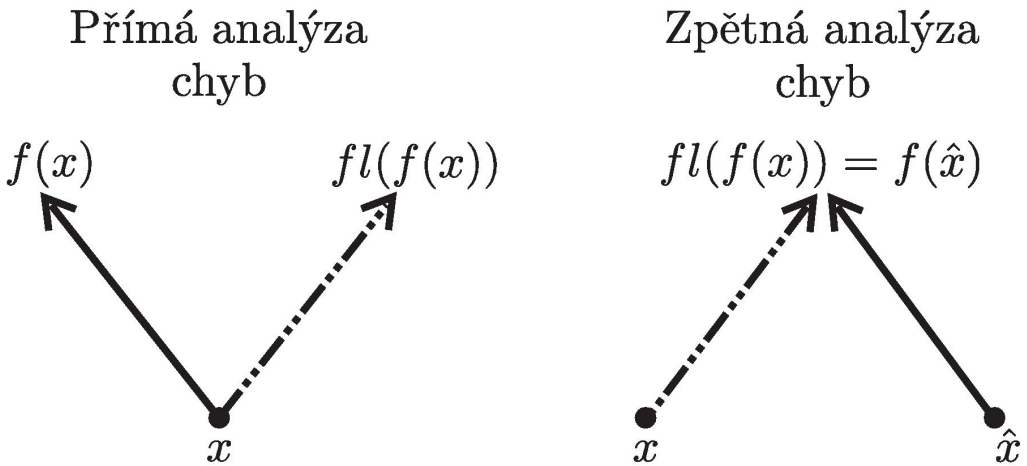
Numerická stabilita algoritmů





Přímá a zpětná analýza chyb

Numerická stabilita algoritmů

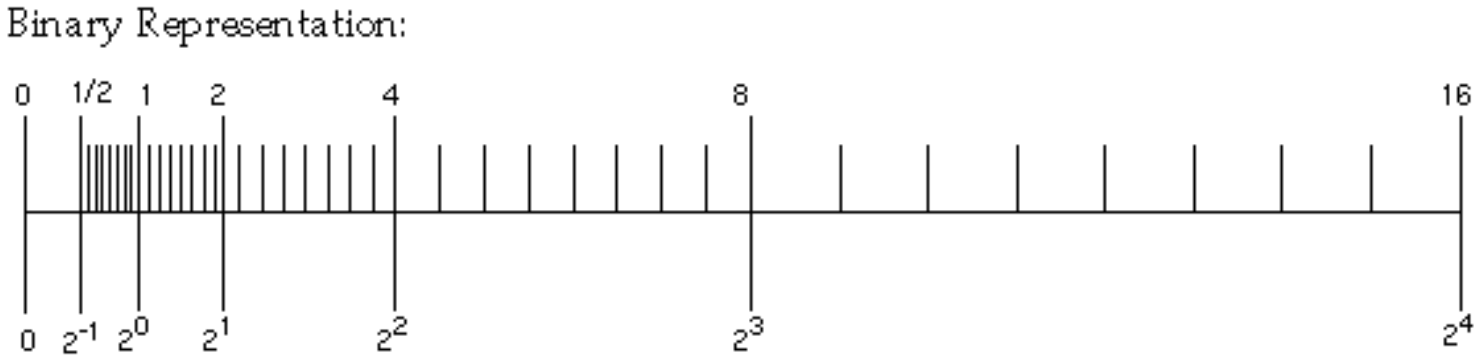


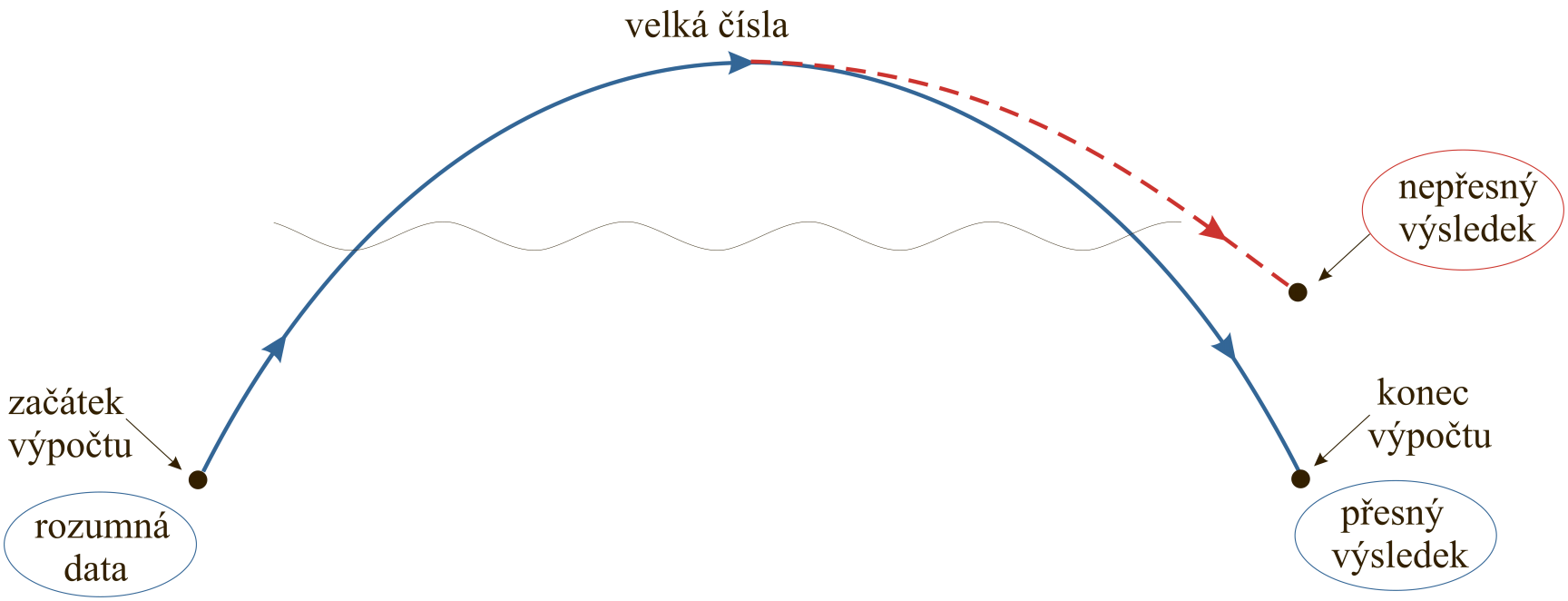
Algoritmus $f(x)$ je **zpětně stabilní**, pokud platí

$$\mathrm{fl}(f(x)) = f(\widehat{x}) \quad \text{pro nějaké } \widehat{x} \text{ splňující } \quad \frac{\|x - \widehat{x}\|}{\|x\|} = \mathcal{O}(\mathbf{u}),$$

Kde se ztrácí přesnost?

Příklad 1: Velká čísla





Kde se ztrácí přesnost?

Příklad 2: Odečítání dvou blízkých čísel

Výpočet

$$\frac{a-b}{c},$$

kde $a \approx b$, $|c| \approx |a-b| \rightarrow$ vyrušení cifer, **ztráta přesnosti**.

Kde se ztrácí přesnost?

Příklad 2: Odečítání dvou blízkých čísel

Výpočet

$$\frac{a-b}{c},$$

kde $a \approx b$, $|c| \approx |a-b| \rightarrow$ vyrušení cifer, **ztráta přesnosti**.

Příklad: Vyhodnocení funkce

$$f(x) \equiv \frac{1-\cos(x)}{x^2} = \frac{1}{2}\left(\frac{\sin\frac{x}{2}}{\frac{x}{2}}\right)^2 \qquad \text{v} \qquad x = 1.2 * 10^{-8}.$$

Kde se ztrácí přesnost?

Příklad 2: Odečítání dvou blízkých čísel

Výpočet

$$\frac{a-b}{c},$$

kde $a \approx b$, $|c| \approx |a-b| \rightarrow$ vyrušení cifer, **ztráta přesnosti**.

Příklad: Vyhodnocení funkce

$$f(x) \equiv \frac{1-\cos(x)}{x^2} = \frac{1}{2}\left(\frac{\sin\frac{x}{2}}{\frac{x}{2}}\right)^2 \qquad \text{v} \qquad x = 1.2 * 10^{-8}.$$

$$\begin{aligned}
\cos(x) &= 0.99999999999999928 \\
\text{fl}\,(\cos(x)) &= 0.99999999999999989 \\
1-\cos(x) &= 0.72000000000000000 * 10^{-16} \\
\text{fl}\,(1-\cos(x)) &= 1.1102230246251565 * 10^{-16} \\
\text{fl}\left((1-\cos(x))/x^2\right) &= 0.77098821154524766 \\
\text{fl}\,(f(x)) &= 0.5\,.
\end{aligned}$$

Přehled

Numerický výpočet derivace (gradientu)

Nechť $|f''(x)| \leq L$ na dané oblasti, potom

$$\frac{\partial f}{\partial x_i} = \frac{f(x + h\, e_i) - f(x)}{h} + \delta_h, \quad |\delta_h| \leq \frac{L}{2}\, h\,.$$

Numerický výpočet derivace (gradientu)

Nechť $|f''(x)| \leq L$ na dané oblasti, potom

$$\frac{\partial f}{\partial x_i} = \frac{f(x + h\,e_i) - f(x)}{h} + \delta_h, \quad |\delta_h| \leq \frac{L}{2}\,h\,.$$

Platí

$$\frac{\partial f}{\partial x_i} = \frac{\mathrm{fl}(f(x + h\,e_i)) - \mathrm{fl}(f(x))}{h} + \delta_{\mathbf{u}} + \delta_h,$$

$$\begin{aligned} \delta_{\mathbf{u}} &= \frac{f(x + h\,e_i) - f(x)}{h} - \frac{\mathrm{fl}(f(x + h\,e_i)) - \mathrm{fl}(f(x))}{h} \\ &= \frac{1}{h}\left(f(x + h\,e_i) - \mathrm{fl}(f(x + h\,e_i)) - f(x) + \mathrm{fl}(f(x))\right). \end{aligned}$$

Numerický výpočet derivace (gradientu)

aneb „Kolik třešní, tolik višní“

Předpokládejme, že v konečné aritmetice

$$|\mathrm{fl}(f(y)) - f(y)| \;\leq\; \mathbf{u}\,\|f\|_\infty\,.$$

Numerický výpočet derivace (gradientu)

aneb „Kolik třešní, tolik višní“

Předpokládejme, že v konečné aritmetice

$$|\mathrm{fl}(f(y)) - f(y)| \;\leq\; \mathbf{u}\,\|f\|_\infty\,.$$

Potom

$$\frac{\partial f}{\partial x_i} \;=\; \frac{\mathrm{fl}(f(x + h\,e_i)) - \mathrm{fl}(f(x))}{h} \;+\; \delta_{\mathbf{u}} \;+\; \delta_h,$$

$$|\delta_{\mathbf{u}} \;+\; \delta_h| \;\leq |\delta_{\mathbf{u}}| \;+\; |\delta_h| \;\leq\; \frac{2\,\mathbf{u}\,\|f\|_\infty}{h} + \frac{L\,\mathbf{u}}{2}.$$

Numerický výpočet derivace (gradientu)

aneb „Kolik třešní, tolik višní“

Předpokládejme, že v konečné aritmetice

$$|\mathrm{fl}(f(y)) - f(y)| \;\leq\; \mathbf{u}\,\|f\|_\infty\,.$$

Potom

$$\frac{\partial f}{\partial x_i} \;=\; \frac{\mathrm{fl}(f(x + h\,e_i)) - \mathrm{fl}(f(x))}{h} \;+\; \delta_{\mathbf{u}} \;+\; \delta_h,$$

$$|\delta_{\mathbf{u}} \;+\; \delta_h| \;\leq |\delta_{\mathbf{u}}| \;+\; |\delta_h| \;\leq\; \frac{2\,\mathbf{u}\,\|f\|_\infty}{h} + \frac{L\,\mathbf{u}}{2}.$$

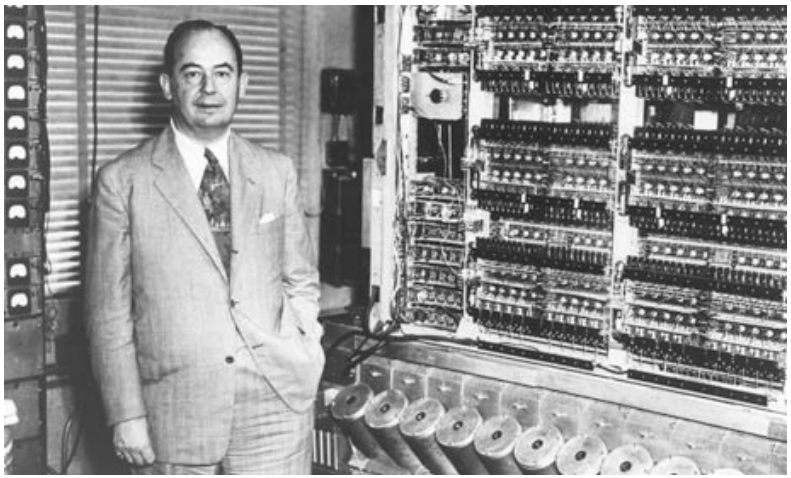
Minimalizujeme horní mez přes h,

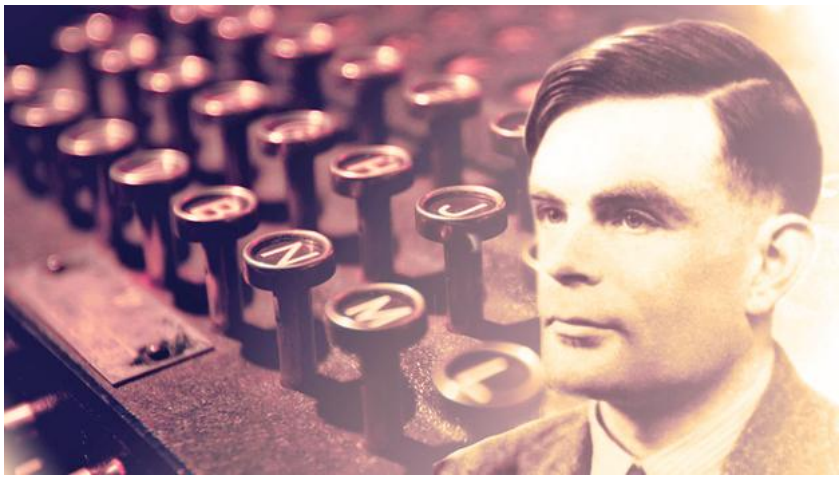
$$h_* \;=\; \sqrt{\mathbf{u}}\;\left(2\sqrt{\frac{\|f\|_\infty}{L}}\right)\,.$$

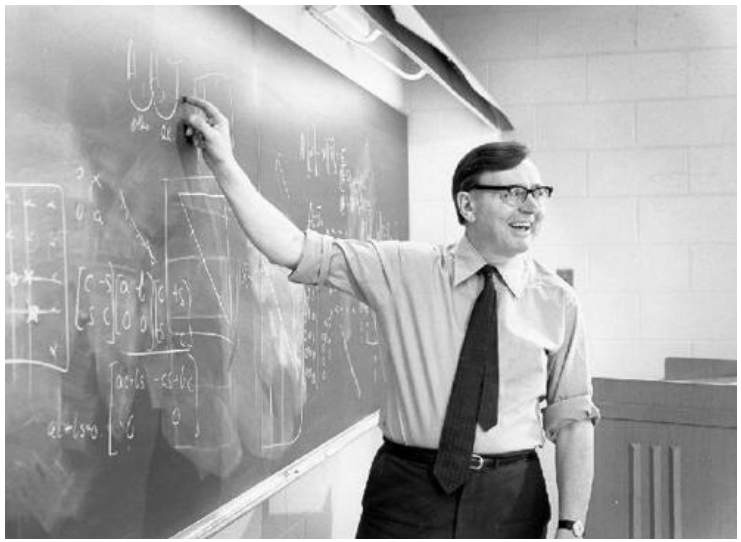
Přehled

Numerická analýza Gaussovy eliminace

- 40 léta minulého století poprvé implementována na počítači,
- John von Neumann, Alan Turing, James H. Wilkinson.







Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{rrrrr|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ -1 & 1 & 0 & 0 & 1 & \bullet \\ -1 & -1 & 1 & 0 & 1 & \bullet \\ -1 & -1 & -1 & 1 & 1 & \bullet \\ -1 & -1 & -1 & -1 & 1 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{rrrrr|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ -1 & -1 & 1 & 0 & 1 & \bullet \\ -1 & -1 & -1 & 1 & 1 & \bullet \\ -1 & -1 & -1 & -1 & 1 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{rrrrr|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & -1 & 1 & 0 & 2 & \bullet \\ -1 & -1 & -1 & 1 & 1 & \bullet \\ -1 & -1 & -1 & -1 & 1 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{rrrrr|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & -1 & 1 & 0 & 2 & \bullet \\ 0 & -1 & -1 & 1 & 2 & \bullet \\ -1 & -1 & -1 & -1 & 1 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & -1 & 1 & 0 & 2 & \bullet \\ 0 & -1 & -1 & 1 & 2 & \bullet \\ 0 & -1 & -1 & -1 & 2 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & 0 & 1 & 0 & 4 & \bullet \\ 0 & -1 & -1 & 1 & 2 & \bullet \\ 0 & -1 & -1 & -1 & 2 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & 0 & 1 & 0 & 4 & \bullet \\ 0 & 0 & -1 & 1 & 4 & \bullet \\ 0 & -1 & -1 & -1 & 2 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & 0 & 1 & 0 & 4 & \bullet \\ 0 & 0 & -1 & 1 & 4 & \bullet \\ 0 & 0 & -1 & -1 & 4 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. × za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & 0 & 1 & 0 & 4 & \bullet \\ 0 & 0 & 0 & 1 & 8 & \bullet \\ 0 & 0 & -1 & -1 & 4 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. $\times$ za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & 0 & 1 & 0 & 4 & \bullet \\ 0 & 0 & 0 & 1 & 8 & \bullet \\ 0 & 0 & 0 & -1 & 8 & \bullet \end{array}\right].$$

Gaussova eliminace může dávat nepřesné výsledky

„Politováníhodné nedopatření, ke kterému docházím maximálně .. × za 10 let“

- Velikost prvků může při eliminaci růst

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 1 & \bullet \\ 0 & 1 & 0 & 0 & 2 & \bullet \\ 0 & 0 & 1 & 0 & 4 & \bullet \\ 0 & 0 & 0 & 1 & 8 & \bullet \\ 0 & 0 & 0 & 0 & 16 & \bullet \end{array}\right].$$

- n rovnic o n neznámých $\rightarrow$ velikost prvku je

$$2^{n-1}.$$

Při **praktických výpočtech** dochází k velkému růstu velikosti prvků **zřídka**, nutné ale kontrolovat během výpočtu.

Příklad řešený na počítači, 56 neznámých

Zvolme hodnoty proměnných (řešení), například

$$[1, -1, 1, \ldots, -1, 1, -1]$$

a **dopočtěme** pravou stranu

$$[0, -3, 0, -3, \ldots, 0, -3, 0, -2].$$

Řešíme soustavu tvaru

$$\left[\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 1 & -3 \\ -1 & -1 & 1 & 0 & 0 & 1 & 0 \\ -1 & -1 & -1 & 1 & 0 & 1 & -3 \\ -1 & -1 & -1 & -1 & 1 & 1 & 0 \\ -1 & -1 & -1 & -1 & -1 & 1 & -2 \end{array}\right],$$

ale pro 56 neznámých.

Příklad řešený na počítači, 56 neznámých

Zvolme hodnoty proměnných (řešení), například

$$[1, -1, 1, \ldots, -1, 1, -1]$$

a **dopočtěme** pravou stranu

$$[0, -3, 0, -3, \ldots, 0, -3, 0, -2].$$

Řešíme soustavu tvaru

$$\left[\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 1 & -3 \\ -1 & -1 & 1 & 0 & 0 & 1 & 0 \\ -1 & -1 & -1 & 1 & 0 & 1 & -3 \\ -1 & -1 & -1 & -1 & 1 & 1 & 0 \\ -1 & -1 & -1 & -1 & -1 & 1 & -2 \end{array}\right],$$

ale pro 56 neznámých. Počítač spočte **nepřesné řešení**

$$[1, -1, 1, \ldots, -1, 1, -1, 1, 0, 2, -1].$$

Přehled

Gram-Schmidtův proces

- Dána báze $\{a_1, \ldots, a_m\}$ podprostoru $\mathbb{C}^n$, $n > m$.

  Gram-Schmidtův ortogonalizační proces spočte ortonormální bázi $\{q_1, \ldots, q_m\}$ takovou, že

$$\operatorname{span}\{a_1, \ldots, a_k\} = \operatorname{span}\{q_1, \ldots, q_k\}, \qquad k = 1, \ldots, m.$$

Gram-Schmidtův proces

- Dána báze $\{a_1, \ldots, a_m\}$ podprostoru $\mathbb{C}^n$, $n > m$.

  Gram-Schmidtův ortogonalizační proces spočte ortonormální bázi $\{q_1, \ldots, q_m\}$ takovou, že

  $$\operatorname{span}\{a_1, \ldots, a_k\} = \operatorname{span}\{q_1, \ldots, q_k\}, \qquad k = 1, \ldots, m.$$

- odečteme od a_k jeho projekci na prostor generovaný sloupci $[q_1, \ldots, q_{k-1}] = Q_{k-1}$

  $$z \quad = \quad (I - Q_{k-1} Q_{k-1}^*)\, a_k,$$

  ekvivalentně,

  $$z = (I - q_{k-1} q_{k-1}^*) \ldots (I - q_2 q_2^*)(I - q_1 q_1^*)\, a_k\,,$$

  a normalizujeme

  $$q_k = z / \|z\|\,.$$

Gram-Schmidtův proces

- Dána báze $\{a_1, \ldots, a_m\}$ podprostoru $\mathbb{C}^n$, $n > m$.

  Gram-Schmidtův ortogonalizační proces spočte ortonormální bázi $\{q_1, \ldots, q_m\}$ takovou, že

  $$\operatorname{span}\{a_1, \ldots, a_k\} = \operatorname{span}\{q_1, \ldots, q_k\}, \qquad k = 1, \ldots, m.$$

- odečteme od a_k jeho projekci na prostor generovaný sloupci $[q_1, \ldots, q_{k-1}] = Q_{k-1}$

  $$z \quad = \quad (I - Q_{k-1}Q_{k-1}^*)\, a_k,$$

  ekvivalentně,

  $$z = (I - q_{k-1}q_{k-1}^*) \ldots (I - q_2 q_2^*)(I - q_1 q_1^*)\, a_k\,,$$

  a normalizujeme

  $$q_k = z/\|z\|\,.$$

- Vede na **klasický** a **modifikovaný Gram-Schmidtův proces**.

Klasický (CGS) a modifikovaný (MGS) Gram-Schmidtův proces v konečné aritmetice

- **Ztrátu ortogonality** vektorů vypočtených v aritmetice s konečnou přesností budeme měřit pomocí 2-normy matice

$$\widehat{Q}^T\widehat{Q} - I.$$

Klasický (CGS) a modifikovaný (MGS) Gram-Schmidtův proces v konečné aritmetice

- **Ztrátu ortogonality** vektorů vypočtených v aritmetice s konečnou přesností budeme měřit pomocí 2-normy matice

$$\widehat{Q}^T\widehat{Q} - I.$$

- $A = [a_1, \ldots, a_m]$. Lze matematicky dokázat:

Algoritmus	$\|\widehat{Q}^T\widehat{Q} - I\|$
CGS	$\kappa^2(A)\,\mathbf{u}$
MGS	$\kappa(A)\,\mathbf{u}$

Výsledek pro CGS: [L. Giraud, J. Langou, M. Rozložník, 2005].

Klasický (CGS) a modifikovaný (MGS) Gram-Schmidtův proces v konečné aritmetice

- **Ztrátu ortogonality** vektorů vypočtených v aritmetice s konečnou přesností budeme měřit pomocí 2-normy matice

$$\widehat{Q}^T\widehat{Q} - I.$$

- $A = [a_1, \ldots, a_m]$. Lze matematicky dokázat:

Algoritmus	$\|\widehat{Q}^T\widehat{Q} - I\|$
CGS	$\kappa^2(A)\,\mathbf{u}$
MGS	$\kappa(A)\,\mathbf{u}$

  Výsledek pro CGS: [L. Giraud, J. Langou, M. Rozložník, 2005].

- Lze **potlačit vliv zaokrouhlovacích chyb** a sestrojit téměř perfektně ortogonální bázi?

CGS algoritmus s opakovanou ortogonalizací (ICGS)

aneb „Já ovšem musím trvat na opakování ceremoniálu“

- První ortogonalizace: **projektujeme** a_k

$$z = a_k - \left[(q_{k-1}^T a_k)\, q_{k-1} + \ldots + (q_1^T a_k)\, q_1 \right].$$

CGS algoritmus s opakovanou ortogonalizací (ICGS)

aneb „Já ovšem musím trvat na opakování ceremoniálu“

- První ortogonalizace: **projektujeme** a_k

$$z = a_k - \left[(q_{k-1}^T a_k)\, q_{k-1} + \ldots + (q_1^T a_k)\, q_1 \right].$$

- Opakovaná ortogonalizace: **projektujeme** z

$$w = z - \left[(q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T z)\, q_1 \right].$$

CGS algoritmus s opakovanou ortogonalizací (ICGS)

aneb „Já ovšem musím trvat na opakování ceremoniálu“

- První ortogonalizace: **projektujeme** a_k

$$z = a_k - \left[(q_{k-1}^T a_k)\, q_{k-1} + \ldots + (q_1^T a_k)\, q_1 \right].$$

- Opakovaná ortogonalizace: **projektujeme** z

$$w = z - \left[(q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T z)\, q_1 \right].$$

- Dosadíme-li za z do druhého vztahu,

$$w \quad = \quad a_k - \left[(q_{k-1}^T a_k + q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T a_k + q_1^T z)\, q_1 \right].$$

CGS algoritmus s opakovanou ortogonalizací (ICGS)

aneb „Já ovšem musím trvat na opakování ceremoniálu“

- První ortogonalizace: **projektujeme** a_k

$$z = a_k - [(q_{k-1}^T a_k)\, q_{k-1} + \ldots + (q_1^T a_k)\, q_1]\,.$$

- Opakovaná ortogonalizace: **projektujeme** z

$$w = z - [(q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T z)\, q_1]\,.$$

- Dosadíme-li za z do druhého vztahu,

$$w \;=\; a_k - [(q_{k-1}^T a_k + q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T a_k + q_1^T z)\, q_1]\,.$$

- **Stačí jedno opakování**, potom

$$\|\widehat{Q}^T\widehat{Q} - I\| \;\approx\; \mathbf{u}.$$

Výsledek [L. Giraud, J. Langou, M. Rozložník, and J. van den Eshof, 2005].

CGS algoritmus s opakovanou ortogonalizací (ICGS)

aneb „Já ovšem musím trvat na opakování ceremoniálu“

- První ortogonalizace: **projektujeme** a_k

$$z = a_k - [(q_{k-1}^T a_k)\, q_{k-1} + \ldots + (q_1^T a_k)\, q_1]\,.$$

- Opakovaná ortogonalizace: **projektujeme** z

$$w = z - [(q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T z)\, q_1]\,.$$

- Dosadíme-li za z do druhého vztahu,

$$w \;=\; a_k - [(q_{k-1}^T a_k + q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T a_k + q_1^T z)\, q_1]\,.$$

- **Stačí jedno opakování**, potom

$$\|\widehat{Q}^T\widehat{Q} - I\| \;\approx\; \mathbf{u}.$$

Výsledek [L. Giraud, J. Langou, M. Rozložník, and J. van den Eshof, 2005].

CGS algoritmus s opakovanou ortogonalizací (ICGS)

aneb „Já ovšem musím trvat na opakování ceremoniálu“

- První ortogonalizace: **projektujeme** a_k

$$z = a_k - [(q_{k-1}^T a_k)\, q_{k-1} + \ldots + (q_1^T a_k)\, q_1]\,.$$

- Opakovaná ortogonalizace: **projektujeme** z

$$w = z - [(q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T z)\, q_1]\,.$$

- Dosadíme-li za z do druhého vztahu,

$$w \;=\; a_k - [(q_{k-1}^T a_k + q_{k-1}^T z)\, q_{k-1} + \ldots + (q_1^T a_k + q_1^T z)\, q_1]\,.$$

- **Stačí jedno opakování**, potom

$$\|\widehat{Q}^T\widehat{Q} - I\| \;\approx\; \mathbf{u}.$$

Výsledek [L. Giraud, J. Langou, M. Rozložník, and J. van den Eshof, 2005].
„No, zřejmě slušnej oddíl, no.“

Přehled

Formulace problému

Uvažujme systém

$$Ax = b$$

kde $A \in \mathbb{R}^{n \times n}$ je **symetrická, pozitivně definitní**.

Formulace problému

Uvažujme systém

$$Ax = b$$

kde $A \in \mathbb{R}^{n \times n}$ je **symetrická, pozitivně definitní**.

- A je velká a řídká ,
- nepotřebujeme znát přesné řešení ,
- jsme schopni realizovat Av (v je vektor) .

Bez újmy na obecnosti, $\|b\| = 1$, $x_0 = 0$.

Metoda sdružených gradientů

input A, b

$x_0 = 0$, $r_0 = p_0 = b$

for $k = 0, 1, 2, \ldots$ **do**

$$
\begin{aligned}
\gamma_k &= \frac{r_k^T r_k}{p_k^T A\, p_k} \\
x_{k+1} &= x_k + \gamma_k p_k \\
r_{k+1} &= r_k - \gamma_k A\, p_k \\
\delta_{k+1} &= \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k} \\
p_{k+1} &= r_{k+1} + \delta_{k+1} p_k
\end{aligned}
$$

test kvality x_{k+1}

end for

Matematické vlastnosti CG

optimální vlastnost

$k$tý **Krylovův podprostor**,

$$\mathcal{K}_k(A, b) \equiv \mathrm{span}\{b, Ab, \ldots, A^{k-1}b\}\,.$$

CG $\rightarrow$ x_k, r_k, p_k

Matematické vlastnosti CG

optimální vlastnost

$k$tý **Krylovův podprostor**,

$$\mathcal{K}_k(A, b) \equiv \text{span}\{b, Ab, \ldots, A^{k-1}b\}\,.$$

CG $\rightarrow x_k$, r_k, p_k

- rezidua $r_0, \ldots, r_{k-1}$ tvoří **ortogonální** bázi $\mathcal{K}_k(A, b)$,
- vektory $p_0, \ldots, p_{k-1}$ tvoří A-**ortogonální** bázi $\mathcal{K}_k(A, b)$,

Matematické vlastnosti CG

optimální vlastnost

$k$tý **Krylovův podprostor**,

$$\mathcal{K}_k(A, b) \equiv \text{span}\{b, Ab, \ldots, A^{k-1}b\}\,.$$

CG $\rightarrow x_k$, r_k, p_k

- rezidua $r_0, \ldots, r_{k-1}$ tvoří **ortogonální** bázi $\mathcal{K}_k(A, b)$,
- vektory $p_0, \ldots, p_{k-1}$ tvoří A-**ortogonální** bázi $\mathcal{K}_k(A, b)$,
- CG nalezne řešení $Ax = b$ nejvýše po n krocích.

Matematické vlastnosti CG

optimální vlastnost

$k$tý **Krylovův podprostor**,

$$\mathcal{K}_k(A, b) \equiv \mathrm{span}\{b, Ab, \ldots, A^{k-1}b\}\,.$$

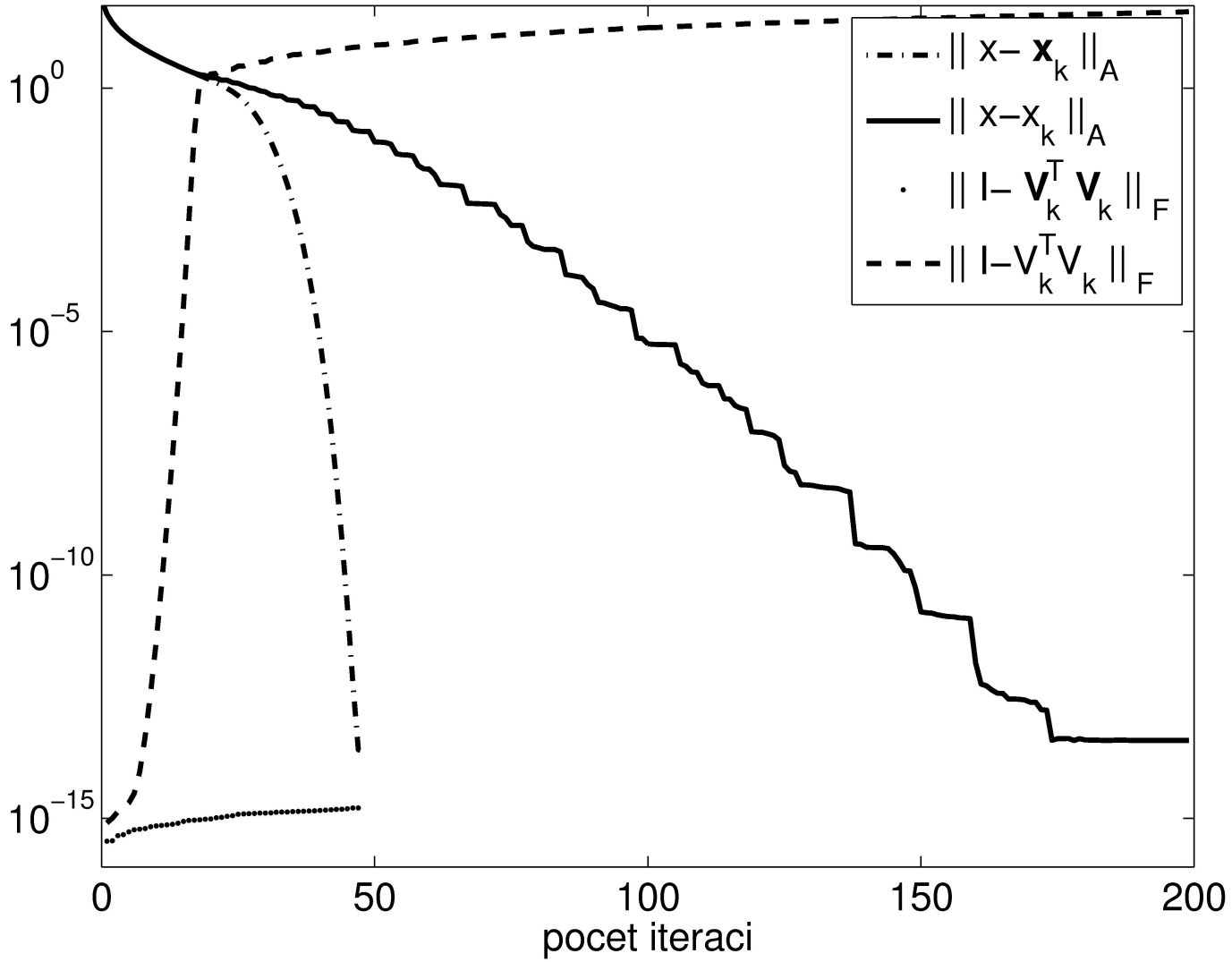
CG $\rightarrow x_k$, r_k, p_k

- rezidua $r_0, \ldots, r_{k-1}$ tvoří **ortogonální** bázi $\mathcal{K}_k(A, b)$,
- vektory $p_0, \ldots, p_{k-1}$ tvoří A-**ortogonální** bázi $\mathcal{K}_k(A, b)$,
- CG nalezne řešení $Ax = b$ nejvýše po n krocích.
- Aproximace řešení x_k **je optimální**

$$\|x - x_k\|_A = \min_{y \in \mathcal{K}_k} \|x - y\|_A\,.$$

Vliv konečné aritmetiky

Zpoždění konvergence, ztráta ortogonality, hladina limitní přesnosti



Analýza chování CG v konečné aritmetice

aneb „Nemusí pršet, jen když kape“

- [Paige 1972] ztráta ortogonality v Lanczosově algoritmu.

Analýza chování CG v konečné aritmetice

aneb „Nemusí pršet, jen když kape“

- [Paige 1972] ztráta ortogonality v Lanczosově algoritmu.
- [Greenbaum 1989]
  - na **výpočty FP CG** aplikované na $Ax = b$ lze hledět jako
  - na **výpočty přesné CG** aplikované na jiný systém $\tilde{A}\tilde{x} = \tilde{b}$, kde $\tilde{A}$ je symetrická pozitivně definitní matice.

Analýza chování CG v konečné aritmetice

aneb „Nemusí pršet, jen když kape“

- [Paige 1972] ztráta ortogonality v Lanczosově algoritmu.
- [Greenbaum 1989]
  - na **výpočty FP CG** aplikované na $Ax = b$ lze hledět jako
  - na **výpočty přesné CG** aplikované na jiný systém $\tilde{A}\tilde{x} = \tilde{b}$, kde $\tilde{A}$ je symetrická pozitivně definitní matice.
- Některé **teoretické vlastnosti CG** zůstávají **zachovány** i v konečné aritmetice.

Analýza chování CG v konečné aritmetice

aneb „Nemusí pršet, jen když kape“

- [Paige 1972] ztráta ortogonality v Lanczosově algoritmu.
- [Greenbaum 1989]
  - na **výpočty FP CG** aplikované na $Ax = b$ lze hledět jako
  - na **výpočty přesné CG** aplikované na jiný systém $\tilde{A}\tilde{x} = \tilde{b}$, kde $\tilde{A}$ je symetrická pozitivně definitní matice.
- Některé **teoretické vlastnosti CG** zůstávají **zachovány** i v konečné aritmetice.
- Zaokrouhlovací chyby „jsou pod kontrolou“, ale dochází ke
  - **zpoždění konvergence**,
  - **zastavení na hladině** maximálně dosažitelné přesnosti.

Přehled

Formulace problému

Uvažujme systém

$$Ax = b$$

kde $A \in \mathbb{R}^{n \times n}$ je regulární, obecně **nesymetrická**.

- A je velká a řídká ,
- nepotřebujeme znát přesné řešení ,
- jsme schopni realizovat Av (v je vektor) ,
- potřebujeme metodu **s nízkými paměťovými nároky** .

Metoda bikonjugovaných gradientů (BiCG)

Současné řešení systémů

$$Ax = b\,, \qquad A^T y = c\,.$$

input A, b, c

$$x_0 = y_0 = 0$$
$$r_0 = p_0 = b,\ s_0 = q_0 = c$$

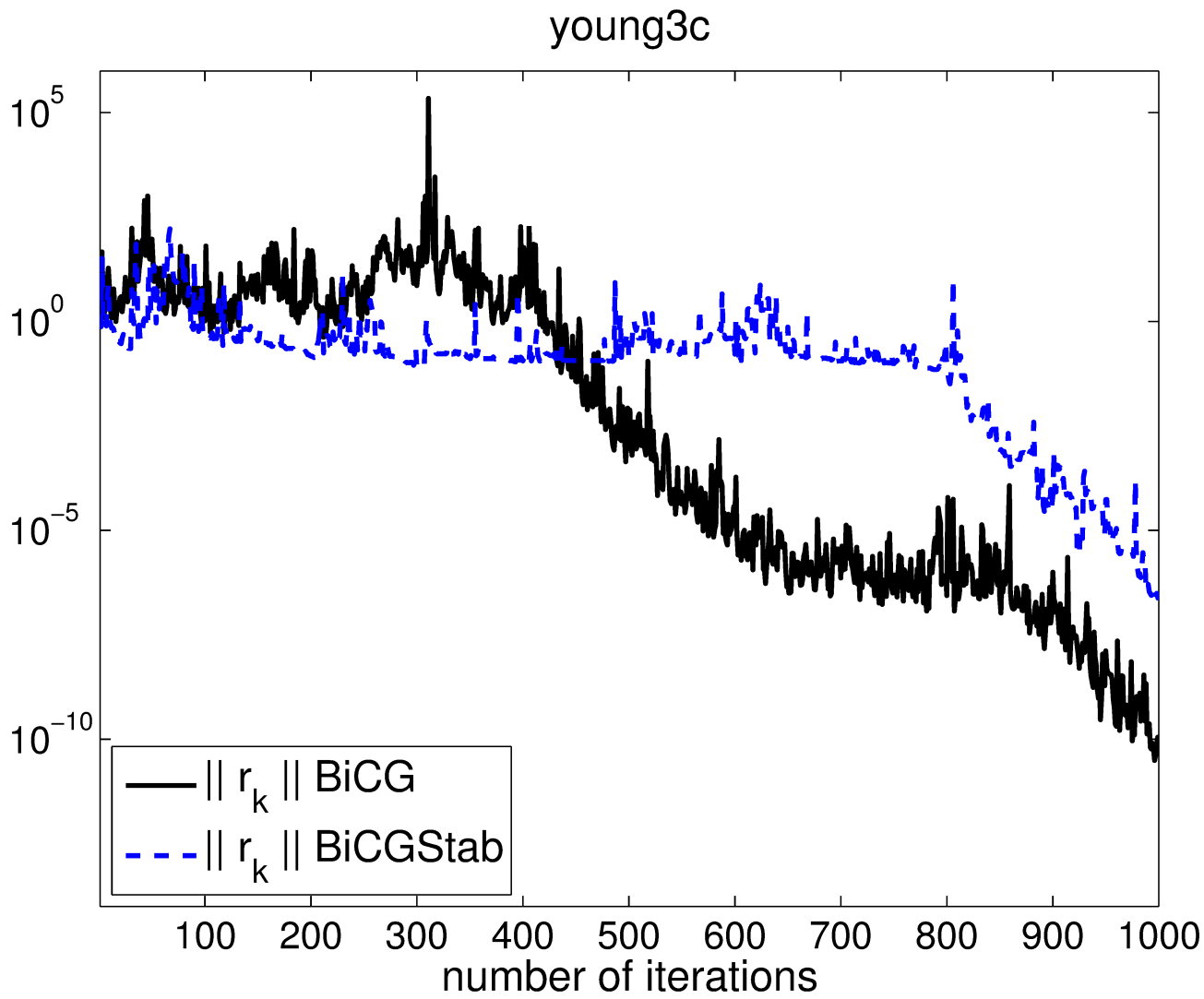
for $k = 0, 1, \ldots$

$$\gamma_k = \frac{s_k^T r_k}{q_k^T A p_k}\,,$$
$$x_{k+1} = x_k + \gamma_k\, p_k\,, \qquad y_{k+1} = y_k + \gamma_k\, q_k\,,$$
$$r_{k+1} = r_k - \alpha_k\, A p_k\,, \qquad s_{k+1} = s_k - \gamma_k\, A^T q_n\,,$$
$$\delta_{k+1} = \frac{s_{k+1}^T r_{k+1}}{s_k^T r_k}\,,$$
$$p_{k+1} = r_{k+1} + \delta_{k+1}\, p_k\,, \qquad q_{k+1} = s_{k+1} + \delta_{k+1}\, q_k$$

end

BiCG v konečné aritmetice

„Nebudem se pouštět do žádných větších akcí.“



Závěr

- **Zaokrouhlovací chyby je nutné brát v úvahu**, často podstatně ovlivňují chování algoritmů.

Závěr

- **Zaokrouhlovací chyby je nutné brát v úvahu**, často podstatně ovlivňují chování algoritmů.
- **Jeden z úkolů numerické analýzy** - snaha pochopit vliv zaokrouhlovacích chyb na chování algoritmu a kvalitu spočteného výsledku.

Závěr

- **Zaokrouhlovací chyby je nutné brát v úvahu**, často podstatně ovlivňují chování algoritmů.
- **Jeden z úkolů numerické analýzy** - snaha pochopit vliv zaokrouhlovacích chyb na chování algoritmu a kvalitu spočteného výsledku.

[Nicholas J. Higham, Accuracy and Stability of Numerical Algorithms, SIAM, 2002]