NI-MPI přednáška 10

Numerické algoritmy, řešení soustav lin. rovnic

Štěpán Starosta

11.12.2024

FIT ČVUT

20. Podmíněnost úlohy a stabilita algoritmů

21. Soustavy lineárních rovnic

- Značení
- Maticová norma
- Podmíněnost úlohy
- Popis iterační metody
- Konvergence
- Konkrétní algoritmy
- Ukázka

20. Podmíněnost úlohy a stabilita algoritmů

21. Soustavy lineárních rovnic
- Značení
- Maticová norma
- Podmíněnost úlohy
- Popis iterační metody
- Konvergence
- Konkrétní algoritmy
- Ukázka

Dopředná a zpětná chyba

Nechť V je nějaký numerický algoritmus, jehož teoretický (přesný) výstup označíme $V^*(d)$, kde d jsou vstupní data.

Dopředná a zpětná chyba

Nechť V je nějaký numerický algoritmus, jehož teoretický (přesný) výstup označíme $V^*(d)$, kde d jsou vstupní data.

Výsledek výpočtu v konečné (strojové) aritmetice označíme $V(d)$.

Označme $\Delta v = V^*(d) - V(d)$. Tato hodnota je tzv. **dopředná/přímá chyba** (*forward error*). Je to odchylka spočítaného řešení od přesného řešení.

Dopředná a zpětná chyba

Nechť V je nějaký numerický algoritmus, jehož teoretický (přesný) výstup označíme $V^*(d)$, kde d jsou vstupní data.

Výsledek výpočtu v konečné (strojové) aritmetice označíme $V(d)$.

Označme $\Delta v = V^*(d) - V(d)$. Tato hodnota je tzv. **dopředná/přímá chyba** (*forward error*). Je to odchylka spočítaného řešení od přesného řešení.

Nejmenší (v normě) číslo Δd takové, že $V^*(d + \Delta d) = V(d)$ je **zpětná chyba** (*backward error*). Jedná se promítnutí chyby algoritmu V do jeho vstupu – jaký problém byl ve skutečnosti vyřešen.

Dopředná a zpětná chyba

Nechť V je nějaký numerický algoritmus, jehož teoretický (přesný) výstup označíme $V^*(d)$, kde d jsou vstupní data.

Výsledek výpočtu v konečné (strojové) aritmetice označíme $V(d)$.

Označme $\Delta v = V^*(d) - V(d)$. Tato hodnota je tzv. **dopředná/přímá chyba** (*forward error*). Je to odchylka spočítaného řešení od přesného řešení.

Nejmenší (v normě) číslo Δd takové, že $V^*(d + \Delta d) = V(d)$ je **zpětná chyba** (*backward error*). Jedná se promítnutí chyby algoritmu V do jeho vstupu – jaký problém byl ve skutečnosti vyřešen.

$$
\begin{tikzcd}
d + \Delta d \arrow[r, "V^*"] & V(d) \\
d \arrow[u, "\Delta d"] \arrow[ur, dashed, "V"'] \arrow[r, "V^*"'] & V^*(d) \arrow[u, "\Delta v"']
\end{tikzcd}
$$

Dopředná a zpětná chyba

Nechť V je nějaký numerický algoritmus, jehož teoretický (přesný) výstup označíme $V^*(d)$, kde d jsou vstupní data.

Výsledek výpočtu v konečné (strojové) aritmetice označíme $V(d)$.

Označme $\Delta v = V^*(d) - V(d)$. Tato hodnota je tzv. **dopředná/přímá chyba** (*forward error*). Je to odchylka spočítaného řešení od přesného řešení.

Nejmenší (v normě) číslo Δd takové, že $V^*(d + \Delta d) = V(d)$ je **zpětná chyba** (*backward error*). Jedná se promítnutí chyby algoritmu V do jeho vstupu – jaký problém byl ve skutečnosti vyřešen.

$$
\begin{tikzcd}
d + \Delta d \arrow[r, "V^*"] & V(d) \\
d \arrow[u, "\Delta d"] \arrow[r, "V^*"'] \arrow[ur, dashed, "V"] & V^*(d) \arrow[u, "\Delta v"']
\end{tikzcd}
$$

Pokud je pro všechny vstupy d zpětná chyba relativně malá, řekneme, že algoritmus je **zpětně stabilní** (*backward stable*). „Malá“ závisí na kontextu.

Podmíněnost úlohy

Podmíněnost úlohy vyjadřuje závislost změny výstupu na změně vstupních dat - jejich malé perturbaci δd.

Podmíněnost úlohy

Podmíněnost úlohy vyjadřuje závislost změny výstupu na změně vstupních dat - jejich malé perturbaci δd.

Relativní číslo podmíněnosti (*relative condition number*) úlohy je

$$C_r = \lim_{\epsilon \to 0^+} \sup_{\substack{d+\delta d \in D \\ \|\delta d\| \leq \epsilon}} \frac{\frac{\|V^*(d+\delta d) - V^*(d)\|}{\|V^*(d)\|}}{\frac{\|\delta d\|}{\|d\|}},$$

kde D je zkoumaný definiční obor V potažmo V^*.

Podmíněnost úlohy

Podmíněnost úlohy vyjadřuje závislost změny výstupu na změně vstupních dat - jejich malé perturbaci δd.

Relativní číslo podmíněnosti (*relative condition number*) úlohy je

$$C_r = \lim_{\epsilon \to 0^+} \sup_{\substack{d+\delta d \in D \\ \|\delta d\| \leq \epsilon}} \frac{\frac{\|V^*(d+\delta d) - V^*(d)\|}{\|V^*(d)\|}}{\frac{\|\delta d\|}{\|d\|}},$$

kde D je zkoumaný definiční obor V potažmo V^*.

Je-li $C_r \approx 1$, řekneme, že úloha je **dobře podmíněná** (*well-conditioned*).

Je-li velké, řekneme, že úloha je **špatně podmíněná** (*ill-conditioned*).

20. Podmíněnost úlohy a stabilita algoritmů

21. Soustavy lineárních rovnic
    - Značení
    - Maticová norma
    - Podmíněnost úlohy
    - Popis iterační metody
    - Konvergence
    - Konkrétní algoritmy
    - Ukázka

Soustavy lineárních rovnic

V této přednášce se budeme soustředit na známý problém z lineární algebry:

Soustavy lineárních rovnic

V této přednášce se budeme soustředit na známý problém z lineární algebry:

Chceme řešit soustavu $n \in \mathbb{N}$ lineárních rovnic pro n neznámých. Zapíšeme ji v maticovém tvaru

$$Ax = b,$$

kde $A \in \mathbb{R}^{n,n}$ je regulární **matice soustavy**, $b \in \mathbb{R}^{n,1}$ je **vektor pravých stran** a $x \in \mathbb{R}^{n,1}$ je hledané řešení.

Soustavy lineárních rovnic

V této přednášce se budeme soustředit na známý problém z lineární algebry:

Chceme řešit soustavu $n \in \mathbb{N}$ lineárních rovnic pro n neznámých. Zapíšeme ji v maticovém tvaru

$$Ax = b,$$

kde $A \in \mathbb{R}^{n,n}$ je regulární **matice soustavy**, $b \in \mathbb{R}^{n,1}$ je **vektor pravých stran** a $x \in \mathbb{R}^{n,1}$ je hledané řešení.

Řešení takovéhoto problému je velmi často dílčím úkolem v nějaké větší úloze. V lineární algebře tuto úlohu řešíme pomocí Gaussovy eliminace (GEM).

GEM a numerické chyby

- Typickým problémem přímých metod je to, že vznikne-li v jednom kroku numerická chyba, tak se projevuje i při dalších výpočtech: přímé metody obecně nejsou „samoopravující se“, ale spíše nahrávají ke kumulování chyb.

GEM a numerické chyby

- Typickým problémem přímých metod je to, že vznikne-li v jednom kroku numerická chyba, tak se projevuje i při dalších výpočtech: přímé metody obecně nejsou „samoopravující se“, ale spíše nahrávají ke kumulování chyb.

- Pro některé úlohy se drobné chyby během výpočtů projeví pouze jako drobná odchylka ve výsledku, ovšem pro některé mohou znamenat řádovou změnu.

- Při řešení soustavy lineárních rovnic může nastat obojí. Čím to je?

Soustavy lineárních rovnic: příklad (1/2)

Uvažujme dvě soustavy dvou rovnic o dvou neznámých:

$$\begin{pmatrix} 1 & 1/2 \\ 1/2 & 1/3 \end{pmatrix} x = \begin{pmatrix} 3/2 \\ 1 \end{pmatrix} \quad \text{a} \quad \begin{pmatrix} 1 & 1/5 \\ 1/5 & -1 \end{pmatrix} x = \begin{pmatrix} 3/2 \\ 1 \end{pmatrix}.$$

Řešením těchto rovnic je

$$x = (0, 3)^T \text{ resp. } x = (85/52, -35/52)^T \approx (1.6346, -0.67308)^T.$$

Soustavy lineárních rovnic: příklad (1/2)

Uvažujme dvě soustavy dvou rovnic o dvou neznámých:

$$\begin{pmatrix} 1 & 1/2 \\ 1/2 & 1/3 \end{pmatrix} x = \begin{pmatrix} 3/2 \\ 1 \end{pmatrix} \quad \text{a} \quad \begin{pmatrix} 1 & 1/5 \\ 1/5 & -1 \end{pmatrix} x = \begin{pmatrix} 3/2 \\ 1 \end{pmatrix}.$$

Řešením těchto rovnic je

$$x = (0,3)^T \text{ resp. } x = (85/52, -35/52)^T \approx (1.6346, -0.67308)^T.$$

Zkusme simulovat chybu na vstupu či chybu vzniklou během výpočtu záměnou 1 na pravé straně rovnice za 5/6:

$$\begin{pmatrix} 1 & 1/2 \\ 1/2 & 1/3 \end{pmatrix} x = \begin{pmatrix} 3/2 \\ 5/6 \end{pmatrix} \quad \text{a} \quad \begin{pmatrix} 1 & 1/5 \\ 1/5 & -1 \end{pmatrix} x = \begin{pmatrix} 3/2 \\ 5/6 \end{pmatrix}.$$

Řešení se změní na

$$x = (1,1)^T \text{resp. } x = (125/78, -20/39)^T \approx (1.6026, -0.51282)^T.$$

Soustavy lineárních rovnic: příklad (1/2)

Pravou stranu jsme změnili o

$$\begin{pmatrix} 3/2 \\ 1 \end{pmatrix} - \begin{pmatrix} 3/2 \\ 5/6 \end{pmatrix} = \begin{pmatrix} 0 \\ 1/6 \end{pmatrix},$$

vektor euklidovské délky $\frac{1}{6}$ (norma relativní chyby je 0.09).

Soustavy lineárních rovnic: příklad (1/2)

Pravou stranu jsme změnili o

$$\begin{pmatrix} 3/2 \\ 1 \end{pmatrix} - \begin{pmatrix} 3/2 \\ 5/6 \end{pmatrix} = \begin{pmatrix} 0 \\ 1/6 \end{pmatrix},$$

vektor euklidovské délky $\frac{1}{6}$ (norma relativní chyby je 0.09).

Změna v řešení **první rovnice** byla

$$\begin{pmatrix} 0 \\ 3 \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

(norma relativní chyby je 0.75) a **druhé**

$$\begin{pmatrix} 85/52 \\ -35/52 \end{pmatrix} - \begin{pmatrix} 125/78 \\ -20/39 \end{pmatrix} = \begin{pmatrix} 5/156 \\ -25/156 \end{pmatrix}$$

(norma relativní chyby je 0.09).

Soustavy lineárních rovnic: příklad (1/2)

Pravou stranu jsme změnili o

$$\begin{pmatrix} 3/2 \\ 1 \end{pmatrix} - \begin{pmatrix} 3/2 \\ 5/6 \end{pmatrix} = \begin{pmatrix} 0 \\ 1/6 \end{pmatrix},$$

vektor euklidovské délky $\frac{1}{6}$ (norma relativní chyby je 0.09).

Změna v řešení **první rovnice** byla

$$\begin{pmatrix} 0 \\ 3 \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

(norma relativní chyby je 0.75) a **druhé**

$$\begin{pmatrix} 85/52 \\ -35/52 \end{pmatrix} - \begin{pmatrix} 125/78 \\ -20/39 \end{pmatrix} = \begin{pmatrix} 5/156 \\ -25/156 \end{pmatrix}$$

(norma relativní chyby je 0.09).

V prvním případě je relativní chyba řádově větší než relativní chyba pravé strany. U druhé rovnice jsou relativní chyby řádově stejné.

Maticová norma

Pro nějakou vektorovou normu $\|\cdot\|$ na $\mathbb{R}^n \equiv \mathbb{R}^{n,1}$ definujeme **přidruženou maticovou normu** matice $A \in \mathbb{R}^{n,n}$ následujícím způsobem

$$\|A\| := \sup\left\{\|Ax\| \colon x \in \mathbb{R}^{n,1} \text{ a } \|x\| = 1\right\}.$$

Maticová norma

Pro nějakou vektorovou normu $\|\cdot\|$ na $\mathbb{R}^n \equiv \mathbb{R}^{n,1}$ definujeme **přidruženou maticovou normu** matice $A \in \mathbb{R}^{n,n}$ následujícím způsobem

$$\|A\| := \sup\left\{\|Ax\| \colon x \in \mathbb{R}^{n,1} \text{ a } \|x\| = 1\right\}.$$

Připomenutí: **supremum** neprázdné omezené množiny $M \subset \mathbb{R}$ je číslo

$$\sup M = \min\{y \in \mathbb{R} \colon \forall x \in M : x \leq y\}.$$

Maticová norma

Pro nějakou vektorovou normu $\|\cdot\|$ na $\mathbb{R}^n \equiv \mathbb{R}^{n,1}$ definujeme **přidruženou maticovou normu** matice $A \in \mathbb{R}^{n,n}$ následujícím způsobem

$$\|A\| := \sup\left\{\|Ax\| : x \in \mathbb{R}^{n,1} \text{ a } \|x\| = 1\right\}.$$

Připomenutí: **supremum** neprázdné omezené množiny $M \subset \mathbb{R}$ je číslo

$$\sup M = \min\{y \in \mathbb{R} : \forall x \in M : x \leq y\}.$$

Takto definované zobrazení je skutečně normou (rozmyslete!) a platí pro ni $\forall A, B \in \mathbb{R}^{n,n}, \forall x \in \mathbb{R}^n$:

- $\|E\| = 1$ (zde E je jednotková matice),
- $\|Ax\| \leq \|A\| \cdot \|x\|$ (konzistence normy),
- $\|AB\| \leq \|A\| \cdot \|B\|$ (submultiplikativita).

(Obecná maticová norma je někdy definová tak, aby splňovala submultiplikativitu.)

Maticová norma – příklady

Jaké maticové normy jsou přidružené dříve uvedeným normám $\|\cdot\|_1$, $\|\cdot\|_2$ a $\|\cdot\|_\infty$?

- Pro normu $\|\cdot\|_1$ dostáváme:

$$\|A\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{n} |a_{i,j}|\,, \quad \text{tedy maximum součtu absolutních hodnot ve sloupci.}$$

- Pro normu $\|\cdot\|_\infty$ dostáváme:

$$\|A\|_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{i,j}|\,, \quad \text{tedy maximum součtu absolutních hodnot v řádku.}$$

- Pro normu $\|\cdot\|_2$ dostáváme:

$$\|A\|_2 = \text{ odmocnina z největšího vlastního čísla matice } A^T A,$$

kde $A = (a_{i,j})_{i,j=1}^{n}$ je matice z $\mathbb{R}^{n,n}$.

Podmíněnost úlohy (1/3)

Uvažujme soustavu rovnic $Ax = b \neq 0$ s regulární maticí A. Budeme zkoumat, co se stane, pokud pravou stranu b lehce změníme o *perturbaci* δb. Změnu v řešení $x = A^{-1}b$ pak označíme δx, platí tedy

$$Ax = b \quad \text{a} \quad A(x + \delta x) = Ax + A\delta x = b + \delta b.$$

Tudíž $A\delta x = \delta b$.

Podmíněnost úlohy (1/3)

Uvažujme soustavu rovnic $Ax = b \neq 0$ s regulární maticí A. Budeme zkoumat, co se stane, pokud pravou stranu b lehce změníme o *perturbaci* δb. Změnu v řešení $x = A^{-1}b$ pak označíme δx, platí tedy

$$Ax = b \quad \text{a} \quad A(x + \delta x) = Ax + A\delta x = b + \delta b.$$

Tudíž $A\delta x = \delta b$.

Platí $\|b\| = \|Ax\| \leq \|A\| \cdot \|x\|$, z čehož plyne $\frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|}$.

Podmíněnost úlohy (1/3)

Uvažujme soustavu rovnic $Ax = b \neq 0$ s regulární maticí A. Budeme zkoumat, co se stane, pokud pravou stranu b lehce změníme o *perturbaci* δb. Změnu v řešení $x = A^{-1}b$ pak označíme δx, platí tedy

$$Ax = b \quad \text{a} \quad A(x + \delta x) = Ax + A\delta x = b + \delta b.$$

Tudíž $A\delta x = \delta b$.

Platí $\|b\| = \|Ax\| \leq \|A\| \cdot \|x\|$, z čehož plyne $\frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|}$.

Dále $\|\delta x\| = \|A^{-1}\delta b\| \leq \|A^{-1}\| \cdot \|\delta b\|$.

Podmíněnost úlohy (1/3)

Uvažujme soustavu rovnic $Ax = b \neq 0$ s regulární maticí A. Budeme zkoumat, co se stane, pokud pravou stranu b lehce změníme o *perturbaci* δb. Změnu v řešení $x = A^{-1}b$ pak označíme δx, platí tedy

$$Ax = b \quad \text{a} \quad A(x + \delta x) = Ax + A\delta x = b + \delta b.$$

Tudíž $A\delta x = \delta b$.

Platí $\|b\| = \|Ax\| \leq \|A\| \cdot \|x\|$, z čehož plyne $\frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|}$.

Dále $\|\delta x\| = \|A^{-1}\delta b\| \leq \|A^{-1}\| \cdot \|\delta b\|$.

Nakonec dostaneme

$$\frac{\|\delta x\|}{\|x\|} \leq \|A\| \cdot \|A^{-1}\| \cdot \frac{\|\delta b\|}{\|b\|}.$$

Podmíněnost úlohy (2/3)

Odvodili jsme nerovnost

$$\frac{\|\delta x\|}{\|x\|} \leq \|A\| \cdot \|A^{-1}\| \cdot \frac{\|\delta b\|}{\|b\|}.$$

Číslo $\kappa(A) := \|A\| \cdot \|A^{-1}\|$ se nazývá **číslo podmíněnosti** matice A.

Nerovnost výše můžeme číst takto: relativní chyba v řešení x soustavy $Ax = b$ je menší než relativní chyba pravé strany b vynásobená číslem $\kappa(A)$.

Podmíněnost úlohy (2/3)

Odvodili jsme nerovnost

$$\frac{\|\delta x\|}{\|x\|} \leq \|A\| \cdot \|A^{-1}\| \cdot \frac{\|\delta b\|}{\|b\|}.$$

Číslo $\kappa(A) := \|A\| \cdot \|A^{-1}\|$ se nazývá **číslo podmíněnosti** matice A.

Nerovnost výše můžeme číst takto: relativní chyba v řešení x soustavy $Ax = b$ je menší než relativní chyba pravé strany b vynásobená číslem $\kappa(A)$.

Čím je $\kappa(A)$ větší, tím je úloha hůře podmíněná a při jejím numerickém řešení musíme být obezřetní, zejména je-li velká chyba při napočítávání Ax nebo b (které může být řešením nějaké jiné úlohy či výsledek měření).

Podmíněnost úlohy (2/3)

Odvodili jsme nerovnost

$$\frac{\|\delta x\|}{\|x\|} \leq \|A\| \cdot \|A^{-1}\| \cdot \frac{\|\delta b\|}{\|b\|}.$$

Číslo $\kappa(A) := \|A\| \cdot \|A^{-1}\|$ se nazývá **číslo podmíněnosti** matice A.

Nerovnost výše můžeme číst takto: relativní chyba v řešení x soustavy $Ax = b$ je menší než relativní chyba pravé strany b vynásobená číslem $\kappa(A)$.

Čím je $\kappa(A)$ větší, tím je úloha hůře podmíněná a při jejím numerickém řešení musíme být obezřetní, zejména je-li velká chyba při napočítávání Ax nebo b (které může být řešením nějaké jiné úlohy či výsledek měření).

Hodnota čísla podmíněnosti závisí na zvolené normě: závěry jsou tedy vždy vzhledem k této použité normě.

Podmíněnost úlohy (3/3)

Vraťme se ke dvou maticím z ukázkové úlohy uvedené dříve:

$$A = \begin{pmatrix} 1 & 1/2 \\ 1/2 & 1/3 \end{pmatrix} \quad \text{a} \quad B = \begin{pmatrix} 1 & 1/5 \\ 1/5 & -1 \end{pmatrix}.$$

Jejich inverze jsou

$$A^{-1} = \begin{pmatrix} 4 & -6 \\ -6 & 12 \end{pmatrix} \quad \text{resp.} \quad B^{-1} \approx \begin{pmatrix} 0.961538 & 0.192308 \\ 0.192308 & -0.961538 \end{pmatrix}.$$

Podmíněnost úlohy (3/3)

Vraťme se ke dvou maticím z ukázkové úlohy uvedené dříve:

$$A = \begin{pmatrix} 1 & 1/2 \\ 1/2 & 1/3 \end{pmatrix} \quad \text{a} \quad B = \begin{pmatrix} 1 & 1/5 \\ 1/5 & -1 \end{pmatrix}.$$

Jejich inverze jsou

$$A^{-1} = \begin{pmatrix} 4 & -6 \\ -6 & 12 \end{pmatrix} \quad \text{resp.} \quad B^{-1} \approx \begin{pmatrix} 0.961538 & 0.192308 \\ 0.192308 & -0.961538 \end{pmatrix}.$$

Pro výpočet podmíněnosti $\kappa(A) = \|A\|\|A^{-1}\|$ použijeme např. normu $\|A\|_\infty$:

$$\kappa(A) = \|A\|_\infty \|A^{-1}\|_\infty = \frac{3}{2} 18 = 27 \quad \text{a} \quad \kappa(B) = \frac{18}{13} \approx 1.3846056.$$

Úloha s maticí A je tedy výrazně **hůře podmíněná**, než ta s maticí B, což koresponduje s našimi dřívějšími výpočty.

Schéma základní iterační metody pro řešení $Ax = b$

Nejdříve si uvedeme obecný popis metody, která „spadne z nebe“, a následně si vysvětlíme, kdy a jak funguje:

Schéma základní iterační metody pro řešení $Ax = b$

Nejdříve si uvedeme obecný popis metody, která „spadne z nebe", a následně si vysvětlíme, kdy a jak funguje:

- Naším cílem je algoritmus, který konstruuje **posloupnost vektorů** $x_0, x_1, x_2, \ldots$, která se „blíží" k přesnému řešení rovnice $Ax = b$.

Schéma základní iterační metody pro řešení $Ax = b$

Nejdříve si uvedeme obecný popis metody, která „spadne z nebe“, a následně si vysvětlíme, kdy a jak funguje:

- Naším cílem je algoritmus, který konstruuje **posloupnost vektorů** $x_0, x_1, x_2, \ldots$, která se „blíží“ k přesnému řešení rovnice $Ax = b$.
- Startovací vektor x_0 zvolíme náhodně, o řešení nemáme žádnou informaci, takže chceme, aby algoritmus fungoval pro libovolnou volbu x_0.

Schéma základní iterační metody pro řešení $Ax = b$

Nejdříve si uvedeme obecný popis metody, která „spadne z nebe“, a následně si vysvětlíme, kdy a jak funguje:

- Naším cílem je algoritmus, který konstruuje **posloupnost vektorů** $x_0, x_1, x_2, \ldots$, která se „blíží“ k přesnému řešení rovnice $Ax = b$.
- Startovací vektor x_0 zvolíme náhodně, o řešení nemáme žádnou informaci, takže chceme, aby algoritmus fungoval pro libovolnou volbu x_0.
- Zvolíme si **regulární matici** Q (různé volby této matice pak povedou na různé metody).

Schéma základní iterační metody pro řešení $Ax = b$

Nejdříve si uvedeme obecný popis metody, která „spadne z nebe“, a následně si vysvětlíme, kdy a jak funguje:

- Naším cílem je algoritmus, který konstruuje **posloupnost vektorů** $x_0, x_1, x_2, \ldots$, která se „blíží“ k přesnému řešení rovnice $Ax = b$.
- Startovací vektor x_0 zvolíme náhodně, o řešení nemáme žádnou informaci, takže chceme, aby algoritmus fungoval pro libovolnou volbu x_0.
- Zvolíme si **regulární matici** Q (různé volby této matice pak povedou na různé metody).
- Členy posloupnosti $x_0, x_1, x_2, \ldots$ budeme napočítávat podle následujícího předpisu:

$$Qx_k = (Q - A)x_{k-1} + b, \quad \text{pro všechna } k > 0.$$

  resp.

$$x_k := Q^{-1}\big((Q - A)x_{k-1} + b\big), \quad \text{pro všechna } k > 0.$$

Základní iterační metody pro řešení $Ax = b$: myšlenka

Kdyby byla posloupnost $(x_k)_{k=0}^{\infty}$ konvergentní s limitou x^*, potom je toto x^* hledané řešení!

Základní iterační metody pro řešení $Ax = b$: myšlenka

Kdyby byla posloupnost $(x_k)_{k=0}^{\infty}$ konvergentní s limitou x^*, potom je toto x^* hledané řešení!

Pošleme-li totiž v rovnici

$$Qx_k = (Q - A)x_{k-1} + b,$$

k do nekonečna, dostaneme

$$Qx^* = (Q - A)x^* + b,$$

a tedy $Ax^* = b$.

Základní iterační metody pro řešení $Ax = b$: myšlenka

Kdyby byla posloupnost $(x_k)_{k=0}^{\infty}$ konvergentní s limitou x^*, potom je toto x^* hledané řešení!

Pošleme-li totiž v rovnici

$$Qx_k = (Q - A)x_{k-1} + b,$$

k do nekonečna, dostaneme

$$Qx^* = (Q - A)x^* + b,$$

a tedy $Ax^* = b$.

Poznámka: Skutečně. Ze základních vlastností normy plyne implikace: pokud $\lim_{k\to\infty} x_k = x^*$, potom $\lim_{k\to\infty} Mx_k = Mx^*$.

Základní iterační metody pro řešení $Ax = b$: myšlenka

Kdyby byla posloupnost $(x_k)_{k=0}^{\infty}$ konvergentní s limitou x^*, potom je toto x^* hledané řešení!

Pošleme-li totiž v rovnici

$$Qx_k = (Q - A)x_{k-1} + b,$$

k do nekonečna, dostaneme

$$Qx^* = (Q - A)x^* + b,$$

a tedy $Ax^* = b$.

Poznámka: Skutečně. Ze základních vlastností normy plyne implikace: pokud $\lim_{k\to\infty} x_k = x^*$, potom $\lim_{k\to\infty} Mx_k = Mx^*$.

Myšlenka: budeme volit Q tak, aby výše definovaná posloupnost $(x_k)_{k=0}^{\infty}$ konvergovala k nějakému x^*.

Konvergence – volba Q

Rovnost $x_k = Q^{-1}\big((Q-A)x_{k-1} + b\big)$ dosadíme do

$$\begin{aligned} x_k - x &= Q^{-1}\big((Q-A)x_{k-1} + b\big) - x \\ &= (E - Q^{-1}A)x_{k-1} - x + Q^{-1}b \\ &= (E - Q^{-1}A)x_{k-1} - (E - Q^{-1}A)x \\ &= (E - Q^{-1}A)(x_{k-1} - x), \end{aligned}$$

kde x je vektor splňující $Ax = b$ a E jednotková matice.

Konvergence – volba Q

Rovnost $x_k = Q^{-1}\big((Q - A)x_{k-1} + b\big)$ dosadíme do

$$\begin{aligned} x_k - x &= Q^{-1}\big((Q - A)x_{k-1} + b\big) - x \\ &= (E - Q^{-1}A)x_{k-1} - x + Q^{-1}b \\ &= (E - Q^{-1}A)x_{k-1} - (E - Q^{-1}A)x \\ &= (E - Q^{-1}A)(x_{k-1} - x), \end{aligned}$$

kde x je vektor splňující $Ax = b$ a E jednotková matice.

Označme $W := E - Q^{-1}A$. Dále označme **vektor chyby** $e_k := x_k - x$, pak platí

$$e_k = We_{k-1} = W^2 e_{k-2} = \cdots = W^k e_0.$$

Konvergence – volba Q

Rovnost $x_k = Q^{-1}\big((Q-A)x_{k-1}+b\big)$ dosadíme do

$$\begin{aligned} x_k - x &= Q^{-1}\big((Q-A)x_{k-1}+b\big) - x \\ &= (E - Q^{-1}A)x_{k-1} - x + Q^{-1}b \\ &= (E - Q^{-1}A)x_{k-1} - (E - Q^{-1}A)x \\ &= (E - Q^{-1}A)(x_{k-1} - x), \end{aligned}$$

kde x je vektor splňující $Ax = b$ a E jednotková matice.

Označme $W := E - Q^{-1}A$. Dále označme **vektor chyby** $e_k := x_k - x$, pak platí

$$e_k = We_{k-1} = W^2 e_{k-2} = \cdots = W^k e_0.$$

Naším cílem je, aby se e_k pro rostoucí k zmenšovalo a blížilo se k nule, vágně řečeno platí: e_k bude „menší“ než e_{k-1} pokud bude W „malé“.

Konvergence – volba Q

Rovnost $x_k = Q^{-1}\big((Q-A)x_{k-1} + b\big)$ dosadíme do

$$\begin{aligned} x_k - x &= Q^{-1}\big((Q-A)x_{k-1} + b\big) - x \\ &= (E - Q^{-1}A)x_{k-1} - x + Q^{-1}b \\ &= (E - Q^{-1}A)x_{k-1} - (E - Q^{-1}A)x \\ &= (E - Q^{-1}A)(x_{k-1} - x), \end{aligned}$$

kde x je vektor splňující $Ax = b$ a E jednotková matice.

Označme $W := E - Q^{-1}A$. Dále označme **vektor chyby** $e_k := x_k - x$, pak platí

$$e_k = We_{k-1} = W^2 e_{k-2} = \cdots = W^k e_0.$$

Naším cílem je, aby se e_k pro rostoucí k zmenšovalo a blížilo se k nule, vágně řečeno platí: e_k bude „menší“ než e_{k-1} pokud bude W „malé“.

Tj. potřebujeme W pro které $\lim_{k\to\infty} W^k = 0$.

Konvergence – Spektrální poloměr

Spektrální poloměr matice M je číslo $\rho(M) \geq 0$ definované jako absolutní hodnota největšího (v absolutní hodnotě) vlastního čísla:

$$\rho(M) := \max\{|\lambda| \colon \lambda \text{ je vlastním číslem } M\}.$$

Konvergence – Spektrální poloměr

Spektrální poloměr matice M je číslo $\rho(M) \geq 0$ definované jako absolutní hodnota největšího (v absolutní hodnotě) vlastního čísla:

$$\rho(M) := \max\{|\lambda| : \lambda \text{ je vlastním číslem } M\}.$$

Věta 22.1

Nechť $M \in \mathbb{C}^{n,n}$. Potom platí

$$\lim_{k \to +\infty} M^k = 0 \quad \Leftrightarrow \quad \rho(M) < 1.$$

Konvergence – Spektrální poloměr

Spektrální poloměr matice M je číslo $\rho(M) \geq 0$ definované jako absolutní hodnota největšího (v absolutní hodnotě) vlastního čísla:

$$\rho(M) := \max\{|\lambda| \colon \lambda \text{ je vlastním číslem } M\}.$$

Věta 22.1

Nechť $M \in \mathbb{C}^{n,n}$. Potom platí

$$\lim_{k \to +\infty} M^k = 0 \quad \Leftrightarrow \quad \rho(M) < 1.$$

Tedy v našem případě máme zajištěnou konvergenci iterační metody **právě tehdy, když**

$$\rho(W) < 1,$$

neboli všechna vlastní čísla matice $W = E - Q^{-1}A$ jsou v absolutní hodnotě menší než 1.

Důkaz věty (1/3)

Ukážeme si implikaci

$$\rho(M) < 1 \;\Rightarrow\; \lim_{k\to\infty} M^k = 0$$

pro speciální případ.

Důkaz věty (1/3)

Ukážeme si implikaci

$$\rho(M) < 1 \;\Rightarrow\; \lim_{k\to\infty} M^k = 0$$

pro speciální případ.

Předpokládejme, že matice M je diagonalizovatelná, neboli že existuje regulární matice P taková, že $M = PDP^{-1}$, kde

$$D = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix}$$

a $\lambda_1, \ldots, \lambda_n$ jsou vlastní čísla M.

Důkaz věty (1/3)

Ukážeme si implikaci

$$\rho(M) < 1 \;\Rightarrow\; \lim_{k\to\infty} M^k = 0$$

pro speciální případ.

Předpokládejme, že matice M je diagonalizovatelná, neboli že existuje regulární matice P taková, že $M = PDP^{-1}$, kde

$$D = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix}$$

a $\lambda_1, \ldots, \lambda_n$ jsou vlastní čísla M.

Platí $M^k = PDP^{-1}PD^{k-1}P^{-1} = \cdots = PD^kP^{-1}$.

Důkaz věty (2/3)

Zřejmě platí

$$D^k = \begin{pmatrix} \lambda_1^k & 0 & \cdots & 0 \\ 0 & \lambda_2^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n^k \end{pmatrix}.$$

Protože $\rho(M) < 1$ jistě pro všechna $i = 1, \ldots, n$ platí $|\lambda_i| < 1$. Tudíž $\lim_{k\to\infty} D^k = 0$.

Důkaz věty (2/3)

Zřejmě platí

$$D^k = \begin{pmatrix} \lambda_1^k & 0 & \cdots & 0 \\ 0 & \lambda_2^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n^k \end{pmatrix}.$$

Protože $\rho(M) < 1$ jistě pro všechna $i = 1, \ldots, n$ platí $|\lambda_i| < 1$. Tudíž $\lim_{k\to\infty} D^k = 0$.

Celkem tedy

$$\lim_{k\to\infty} M^k = P\Big(\lim_{k\to+\infty} D^k\Big)P^{-1} = 0.$$

Důkaz věty (2/3)

Zřejmě platí

$$D^k = \begin{pmatrix} \lambda_1^k & 0 & \cdots & 0 \\ 0 & \lambda_2^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n^k \end{pmatrix}.$$

Protože $\rho(M) < 1$ jistě pro všechna $i = 1, \ldots, n$ platí $|\lambda_i| < 1$. Tudíž $\lim_{k\to\infty} D^k = 0$.

Celkem tedy

$$\lim_{k\to\infty} M^k = P\Big(\lim_{k\to+\infty} D^k\Big)P^{-1} = 0.$$

Pro nediagonalizovatelnou matici M se postupuje velice podobně, jen se použije Jordanův normální tvar matice.

Důkaz věty (3/3)

Dokažme druhou implikaci

$$\lim_{k\to\infty} M^k = 0 \;\Rightarrow\; \rho(M) < 1$$

sporem.

Důkaz věty (3/3)

Dokažme druhou implikaci

$$\lim_{k\to\infty} M^k = 0 \;\Rightarrow\; \rho(M) < 1$$

sporem.

Mějme vlastní číslo λ matice M, $|\lambda| \geq 1$, s vlastním vektorem $v \neq 0$.

Důkaz věty (3/3)

Dokažme druhou implikaci

$$\lim_{k\to\infty} M^k = 0 \;\Rightarrow\; \rho(M) < 1$$

sporem.

Mějme vlastní číslo λ matice M, $|\lambda| \geq 1$, s vlastním vektorem $v \neq 0$.

Potom

$$0 = \lim_{k\to\infty} \|M^k v\| = \lim_{k\to\infty} |\lambda|^k \cdot \|v\| \neq 0.$$

Což je spor.

Rychlost konvergence

Jak rychle se vektor chyby e_k blíží k nule?

Rychlost konvergence

Jak rychle se vektor chyby e_k blíží k nule?

Máme

$$e_k = W^k e_0.$$

Rychlost konvergence

Jak rychle se vektor chyby e_k blíží k nule?

Máme

$$e_k = W^k e_0.$$

Odhadneme normu (vizte vlastnosti maticové normy!)

$$\|e_k\| = \|W^k e_0\| \leq \|W^k\| \cdot \|e_0\| \leq \|W\|^k \cdot \|e_0\|.$$

Rychlost konvergence

Jak rychle se vektor chyby e_k blíží k nule?

Máme

$$e_k = W^k e_0.$$

Odhadneme normu (vizte vlastnosti maticové normy!)

$$\|e_k\| = \|W^k e_0\| \leq \|W^k\| \cdot \|e_0\| \leq \|W\|^k \cdot \|e_0\|.$$

Odhad vpravo bude ostře klesající pokud $\|W\| < 1$.

Rychlost konvergence

Jak rychle se vektor chyby e_k blíží k nule?

Máme

$$e_k = W^k e_0.$$

Odhadneme normu (vizte vlastnosti maticové normy!)

$$\|e_k\| = \|W^k e_0\| \leq \|W^k\| \cdot \|e_0\| \leq \|W\|^k \cdot \|e_0\|.$$

Odhad vpravo bude ostře klesající pokud $\|W\| < 1$.

Pro tento odhad tedy záleží na volbě normy. Jak jsme viděli pro symetrické matice je chování W^k ovlivněno hodnotou $\rho(W)$. Obecně platí jen $\rho(M) \leq \|M\|$ (pro všechny přidružené maticové normy). Jelikož pro symetrickou W platí $\|W\|_2 = \rho(W)$, je volba normy $\|\cdot\|_2$ pro tento účel ta nejlepší, ačkoliv v praxi se jedná o výpočetně náročnou normu a spíše se nepoužívá. Pro nesymetrickou W je situace obdobná, jen je nutné se opřít o pojem singulární hodnoty. Obecně lze učinit závěr, že čím menší hodnota $\|W\|$ (a menší než 1), tím rychlejší konvergenci můžeme očekávat.

Kdy iterování ukončit? (1/2)

Iterační metodu ukončíme v kroku k, dosáhne-li x_k požadované přesnosti.

Tady vzniká **chyba algoritmu**.

Požadovaná přesnost a podmínka na ni většinou plyne z úlohy.

Kdy iterování ukončit? (1/2)

Iterační metodu ukončíme v kroku k, dosáhne-li x_k požadované přesnosti.

Tady vzniká **chyba algoritmu**.

Požadovaná přesnost a podmínka na ni většinou plyne z úlohy.

Pro případ $\|W\| < 1$ máme zajištěno, že posloupnost $(\|e_k\|)$ je ostře klesající a iterace lze zastavit, když nastane

$$\|e_k\| = \|x_k - x\| < \epsilon,$$

kde konstanta ϵ je uživatelem zadaný parametr. To je samozřejmě nepraktické, protože nemáme přesné řešení x.

Kdy iterování ukončit? (1/2)

Iterační metodu ukončíme v kroku k, dosáhne-li x_k požadované přesnosti.

Tady vzniká **chyba algoritmu**.

Požadovaná přesnost a podmínka na ni většinou plyne z úlohy.

Pro případ $\|W\| < 1$ máme zajištěno, že posloupnost $(\|e_k\|)$ je ostře klesající a iterace lze zastavit, když nastane

$$\|e_k\| = \|x_k - x\| < \epsilon,$$

kde konstanta ϵ je uživatelem zadaný parametr. To je samozřejmě nepraktické, protože nemáme přesné řešení x.

Napočítáme v kroku k tzv. **reziduum** $Ax_k - b$ a tzv. **kritérium konvergence** bude

$$\|Ax_k - b\| < \epsilon.$$

Kdy iterování ukončit? (2/2)

Někdy se místo počítání rezidua volí výpočetně méně náročné kritérium

$$\|x_{k+1} - x_k\| < \epsilon.$$

Kdy iterování ukončit? (2/2)

Někdy se místo počítání rezidua volí výpočetně méně náročné kritérium

$$\|x_{k+1} - x_k\| < \epsilon.$$

Platí totiž

$$\begin{aligned}
\|e_k\| = \|x_k - x\| &= \|x_k - x_{k+1} + x_{k+1} - x\| \\
&\leq \|x_k - x_{k+1}\| + \|\underbrace{x_{k+1} - x}_{=e_{k+1}}\| \\
&< \epsilon + \|W\| \cdot \|e_k\|,
\end{aligned}$$

Kdy iterování ukončit? (2/2)

Někdy se místo počítání rezidua volí výpočetně méně náročné kritérium

$$\|x_{k+1} - x_k\| < \epsilon.$$

Platí totiž

$$\begin{aligned} \|e_k\| = \|x_k - x\| &= \|x_k - x_{k+1} + x_{k+1} - x\| \\ &\leq \|x_k - x_{k+1}\| + \|\underbrace{x_{k+1} - x}_{=e_{k+1}}\| \\ &< \epsilon + \|W\| \cdot \|e_k\|, \end{aligned}$$

kde za předpokladu $\|W\| < 1$ z poslední nerovnosti plyne

$$\|e_k\| < \frac{\epsilon}{1 - \|W\|}.$$

Kdy iterování ukončit? (2/2)

Někdy se místo počítání rezidua volí výpočetně méně náročné kritérium

$$\|x_{k+1} - x_k\| < \epsilon.$$

Platí totiž

$$\begin{aligned}\|e_k\| = \|x_k - x\| &= \|x_k - x_{k+1} + x_{k+1} - x\| \\ &\leq \|x_k - x_{k+1}\| + \|\underbrace{x_{k+1} - x}_{=e_{k+1}}\| \\ &< \epsilon + \|W\| \cdot \|e_k\|,\end{aligned}$$

kde za předpokladu $\|W\| < 1$ z poslední nerovnosti plyne

$$\|e_k\| < \frac{\epsilon}{1 - \|W\|}.$$

Tedy kritérium lze výhodně použít je-li $\|W\|$ menší než 1, ale nepříliš blízko 1.

Poznámka: Výpočty v konečné přesnosti

Všechny dosud uvedené úvahy byly v teoretické (absolutní) přesnosti.

Při počítání v nepřesné aritmetice samozřejmě nemusí metoda konvergovat k přesnému řešení, i když máme $\|W\| < 1$.

Poznámka: Výpočty v konečné přesnosti

Všechny dosud uvedené úvahy byly v teoretické (absolutní) přesnosti.

Při počítání v nepřesné aritmetice samozřejmě nemusí metoda konvergovat k přesnému řešení, i když máme $\|W\| < 1$.

Pokud ovšem v nepřesné aritmetice posloupnost konverguje, pak mají iterační metody tu výhodu, že si „nepamatují“ chyby z předchozích iterací – v každém kroku se z aproximace vyrobí „lepší“ řešení úlohy a začíná se znovu.

Poznámka: Výpočty v konečné přesnosti

Všechny dosud uvedené úvahy byly v teoretické (absolutní) přesnosti.

Při počítání v nepřesné aritmetice samozřejmě nemusí metoda konvergovat k přesnému řešení, i když máme $\|W\| < 1$.

Pokud ovšem v nepřesné aritmetice posloupnost konverguje, pak mají iterační metody tu výhodu, že si „nepamatují“ chyby z předchozích iterací – v každém kroku se z aproximace vyrobí „lepší“ řešení úlohy a začíná se znovu.

V nepřesné aritmetice metoda nemusí konvergovat i v případě, kdy úloha není špatně podmíněná.

Poznámka: Výpočty v konečné přesnosti

Všechny dosud uvedené úvahy byly v teoretické (absolutní) přesnosti.

Při počítání v nepřesné aritmetice samozřejmě nemusí metoda konvergovat k přesnému řešení, i když máme $\|W\| < 1$.

Pokud ovšem v nepřesné aritmetice posloupnost konverguje, pak mají iterační metody tu výhodu, že si „nepamatují“ chyby z předchozích iterací – v každém kroku se z aproximace vyrobí „lepší“ řešení úlohy a začíná se znovu.

V nepřesné aritmetice metoda nemusí konvergovat i v případě, kdy úloha není špatně podmíněná.

V praxi je tedy ještě dalším parametrem tohoto algoritmu **maximální počet iterací**. Při jeho překročení metoda selhala – po daném maximálním počtu iterací nebylo nalezeno přibližné řešení, které by splnilo podmínku konvergence dané konstantou ϵ.

Poznámka 2: iterační zpřesnění

Základní metodu lze dále různě vylepšovat: například použitím tzv. iteračního zpřesnění (*iterative refinement*):

1. V každém kroku napočteme reziduum: $r_k = Ax_k - b$.
2. Vyřešíme systém $Ay_k = r_k$ (přímou metodou).
3. Vypočtené řešení y_k použijeme k vylepšení x_k:

$$x'_k = x_k + y_k.$$

Konkrétní volby Q

Vraťme se zpět k základnímu algoritmu $x_k := Q^{-1}\big((Q-A)x_{k-1} + b\big)$.

Označme $a_{i,j}$ prvky matice A a položme

$$L := \begin{pmatrix} 0 & 0 & \cdots & 0 \\ a_{2,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n-1} & 0 \end{pmatrix} \quad \text{a} \quad D := \begin{pmatrix} a_{1,1} & 0 & \cdots & 0 \\ 0 & a_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_{n,n} \end{pmatrix}.$$

a $U := A - L - D$, tj. platí

$$A = L + D + U.$$

Konkrétní volby Q

Vraťme se zpět k základnímu algoritmu $x_k := Q^{-1}\big((Q - A)x_{k-1} + b\big)$.

Označme $a_{i,j}$ prvky matice A a položme

$$L := \begin{pmatrix} 0 & 0 & \cdots & 0 \\ a_{2,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n-1} & 0 \end{pmatrix} \quad \text{a} \quad D := \begin{pmatrix} a_{1,1} & 0 & \cdots & 0 \\ 0 & a_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_{n,n} \end{pmatrix}.$$

a $U := A - L - D$, tj. platí

$$A = L + D + U.$$

Zmíníme následující volby Q:

- Richardsonova metoda $Q = E$ (E je jednotková matice),
- Jacobiho metoda $Q = D$,
- Gaussova-Seidlova metoda $Q = D + L$ (obecně superrelaxační (*successive overrelaxation*) metoda / SOR metoda $Q = \frac{1}{\omega}D + L$).

Richardsonova metoda

Richardsonova metoda odpovídá volbě $Q = E$.

Tedy iterace jsou jednoduše dány

$$x_k = Q^{-1}\big((Q - A)x_{k-1} + b\big) = (E - A)x_{k-1} + b.$$

Richardsonova metoda

Richardsonova metoda odpovídá volbě $Q = E$.

Tedy iterace jsou jednoduše dány

$$x_k = Q^{-1}\big((Q - A)x_{k-1} + b\big) = (E - A)x_{k-1} + b.$$

Konvergenci kontroluje matice $W = E - Q^{-1}A = E - A$.

Richardsonova metoda

Richardsonova metoda odpovídá volbě $Q = E$.

Tedy iterace jsou jednoduše dány

$$x_k = Q^{-1}\big((Q - A)x_{k-1} + b\big) = (E - A)x_{k-1} + b.$$

Konvergenci kontroluje matice $W = E - Q^{-1}A = E - A$.

Konvergenci proto máme zajištěnou pro velmi úzkou třídu matic: A musí být blízko E tak, aby například platilo

$$\|E - A\| < 1.$$

Jacobiho metoda

Jacobiho metoda odpovídá volbě $Q = D$.

Iterace se napočítávají takto

$$x_k = Q^{-1}\big((Q - A)x_{k-1} + b\big) = D^{-1}(-L - U)x_{k-1} + D^{-1}b.$$

Konvergence je kontrolována maticí $W = E - Q^{-1}A = E - D^{-1}A$. Dále máme následující postačující podmínku.

Jacobiho metoda

Jacobiho metoda odpovídá volbě $Q = D$.

Iterace se napočítávají takto

$$x_k = Q^{-1}\big((Q - A)x_{k-1} + b\big) = D^{-1}(-L - U)x_{k-1} + D^{-1}b.$$

Konvergence je kontrolována maticí $W = E - Q^{-1}A = E - D^{-1}A$. Dále máme následující postačující podmínku.

Tvrzení 22.2
Pokud je matice A ostře diagonálně dominantní, pak Jacobiho metoda konverguje pro všechny volby x_0.

Poznámka: Matice je **ostře diagonálně dominantní** tehdy, pokud pro každý její řádek platí, že součet absolutních hodnot prvků vyjma diagonálního je menší než absolutní hodnota diagonálního prvku. Důkaz vynecháváme.

SOR metoda

SOR metoda odpovídá volbě $Q = \frac{1}{\omega}D + L$, kde $\omega \in \mathbb{R} \setminus \{0\}$.

Iterace se napočítávají takto

$$\left(\frac{1}{\omega}D + L\right)x_k = \left(\frac{1}{\omega}D + L - A\right)x_{k-1} + b = \left(\left(\frac{1}{\omega} - 1\right)D - U\right)x_{k-1} + b.$$

SOR metoda

SOR metoda odpovídá volbě $Q = \frac{1}{\omega}D + L$, kde $\omega \in \mathbb{R} \setminus \{0\}$.

Iterace se napočítávají takto

$$\left(\frac{1}{\omega}D + L\right)x_k = \left(\frac{1}{\omega}D + L - A\right)x_{k-1} + b = \left(\left(\frac{1}{\omega} - 1\right)D - U\right)x_{k-1} + b.$$

Tvrzení 22.3
SOR metoda konverguje pokud $0 < \omega < 2$ a A je symetrická a pozitivně definitní s kladnými prvky na diagonále.

SOR metoda

SOR metoda odpovídá volbě $Q = \frac{1}{\omega} D + L$, kde $\omega \in \mathbb{R} \setminus \{0\}$.

Iterace se napočítávají takto

$$\left(\frac{1}{\omega} D + L\right) x_k = \left(\frac{1}{\omega} D + L - A\right) x_{k-1} + b = \left(\left(\frac{1}{\omega} - 1\right) D - U\right) x_{k-1} + b.$$

Tvrzení 22.3
SOR metoda konverguje pokud $0 < \omega < 2$ a A je symetrická a pozitivně definitní s kladnými prvky na diagonále.

Parametr ω se používá k urychlení konvergence.

Závěr: Algoritmus

Vstup: matice A, Q, vektor b, požadovaná přesnost ϵ, maximální počet iterací K

1. zvol náhodně x_0
2. pro k od 1 do K prováděj
    1. $x_k = Q^{-1}(Q - A)x_{k-1} + Q^{-1}b$
    2. pokud $\|Ax_k - b\| < \epsilon$, vrať x_k (nebo obecně je-li splněno jiné kritérium konvergence)
3. vrať „řešení nebylo nalezeno po K iteracích"

Ukázka – Jacobiho algoritmus (1/2)

Máme matici $A = \begin{pmatrix} 2 & 1 \\ 1 & 4 \end{pmatrix}$.

$$\|E - D^{-1}A\| = \frac{1}{2}.$$

Použijeme Jacobiho algoritmu pro výpočet řešení pro pravou stranu rovnou $b = (3,5)^T$. Přesné řešení je $(1,1)^T$.

Kritérium konvergence použijeme $\varepsilon := \|Ax_k - b\| < 10^{-2}$.

k	x_k	$\|Ax_k - b\|$
0	$(0.5, 1.5)$	1.58113883008
1	$(0.75, 1.125)$	0.450693909433
2	$(0.9375, 1.0625)$	0.197642353761
3	$(0.96875, 1.015625)$	0.0563367386791
4	$(0.9921875, 1.0078125)$	0.0247052942201
5	$(0.99609375, 1.001953125)$	0.00704209233489

Ukázka – Jacobiho algoritmus (2/2)

...stejná úloha, jen začneme vektorem x_0, který je dále od přesného řešení.

k	x_k	$\|Ax_k - b\|$
0	$(-10, 10)$	28.1780056072
1	$(-3.5, 3.75)$	9.01734439844
2	$(-0.375, 2.125)$	3.5222507009
3	$(0.4375, 1.34375)$	1.1271680498
4	$(0.828125, 1.140625)$	0.440281337613
5	$(0.9296875, 1.04296875)$	0.140896006226
6	$(0.978515625, 1.017578125)$	0.0550351672016
7	$(0.9912109375, 1.00537109375)$	0.0176120007782
8	$(0.997314453125, 1.002197265625)$	0.0068793959002