NUMERICKÉ METÓDY LINEÁRNEJ ALGEBRY

08. Iteračné metódy na báze Krylovových podpriestorov.

Ing. Marek Macák, PhD.

Konzultácie: podľa potreby/dohody

3. Apríl 2024

Všeobecná projekčná metóda

- Úloha: nájst riešenie sústavy lineárnych rovníc:

$$Ax = b, \quad A \in \mathbb{C}^{n \times n} \quad x, b \in \mathbb{C}^{n \times 1} \tag{1}$$

- Všeobecná projekcná metóda hladá aproximáciu $\tilde{x}$ z afinného podpriestoru $x_0 + \mathcal{K}_m$ s dimenziou m (x_0 je lubovolný pociatocný odhad), pricom má byt splnená Petrovova-Galerkinova podmienka

$$b - Ax_m \perp \mathcal{L}_m \tag{2}$$

kde $\mathcal{L}$ je další tzv. testovací podpriestor s dimenziou m.

Krylovov podpriestor

- Metoda Krylovovych podpriestorov je metoda kde podpriestor $\mathcal{K}_m$ je Krylovov podpriestor:
$$\mathcal{K}_m(A, r_0) = span\{r_0, Ar_0, A^2 r_0, ..., A^{m-1} r_0\} \tag{3}$$
kde $r_0 = b - Ax_0$.
- Aproximacie v Krylovovho podpriestoru mozeme vyjadrit vo forme
$$A^{-1} b \approx x_m = x_0 + q_{m-1}(A) r_0, \tag{4}$$
kde q_{m-1} je polynom stupna $m+1$.
- Rôzne verzie Krylovovho podpriestoru vznikajú z voľby podpriestoru $\mathcal{L}_m$. Napr.: $\mathcal{L}_m = \mathcal{K}_m(A, r_0)$, $\mathcal{L}_m = A\mathcal{K}_m(A, r_0)$, alebo $\mathcal{L}_m = \mathcal{K}_m(A^T, r_0)$

Krylovov podpriestor

- Pre všeobecny vektor v je príslušný Krylovov podpriestor:

$$\mathcal{K}_m(A, v) = span\{v, Av, A^2v, ..., A^{m-1}v\} \tag{5}$$

- Jeden iteracný krok znamená nárast dimenzie $\mathcal{K}_m$ o jednotku.
- $\mathcal{K}_m$ je podpriestorom vsetkych vektorov v $\mathbb{R}^n$ co mozeme vyjadrit v tvare $x = p(A)v$, kde p je polynom neprekracujuci stupen $m - 1$

Arnoldiho metóda

- Je orhogonalna projekcna metoda na vytvorenie ortogonálnej bázy Krylovovho podpriestoru $\mathcal{K}_m$ pre vseobecne ne-Hermitovske matice.
- Predstavena v roku 1951 ako prostriedok redukcie riednych matic na matice Hessenbergoveho tvaru.
- Arnoldi prezentoval svoju metódu a naznačil, že vlastné hodnoty Hessenbergovej matice by mohli poskytnúť presné aproximácie k vlastným číslam pôvodnej matice. Neskôr sa zistilo, že táto metóda vedie k efektívnej technike aproximácie vlastné čísel veľkých riedkych matíc.

Arnoldiho metód

- *Pseudo algoritmus modifikovanej Arnoldiho metódy:*

Zvoľte vektor $v^{(1)}$ s normou 1.
for $j = 1, 2, \ldots, m$
  $w^{(j)} = Av^{(j)}$ **for** $i = 1, 2, \ldots, j$
    $h_{ij} = (w^{(j)}, v^{(i)})$
    $w^{(j)} = w^{(i)} - h_{ij} v^{(i)}$
    end
  $h_{i+1,j} = ||w^{(j)}||_2$ **if** $h_{i+1,j} = 0$; **STOP**
  $v^{(j+1)} = w^{(j)} / h_{i+1,j}$
  end

- Iné varianty ortogonalizácie: klasický GS, Householder Arnoldi, ...

Plne ortogonalizacna metoda (FOM)

- Metóda hľadá aproximáciu riešenia x_m z podpriestoru $x_0 + \mathcal{K}_m$ splnajúcu podmienku $b - Ax_m \perp \mathcal{K}_m$.
- Ak v Arnoldiho metode zvolime $v^{(1)} = r^{(0)}/||r^{(0)}||_2$ a oznacime $\beta = ||r^{(0)}||_2$
- Vysledny m-dimenzionalny podpriestor je dany

$$
\begin{aligned}
x_m &= x_0 + V_m y_m, \quad (6)\\
y_m &= H_m^{-1}(\beta e_1). \quad (7)
\end{aligned}
$$

- Metoda zalozena na tomto pristupe sa nazyva plne ortogonalizacna metoda (Full Orthogonalization Method (FOM)).

Plne ortogonalizacna metoda (FOM)

- *Pseudo algoritmus FOM metódy:*

Spočítajte $r^{(0)} = b - Ax^{(0)}$, $\beta = ||r^{(0)}||_2$ a $v^{(1)} = r^{(0)}/\beta$
Inizializujte maticu $m \times m$, $H_m = \{h_{ij}\}_{i,j,1,\dots,m} = 0$ **for** $j = 1, 2, \dots, m$
$w^{(j)} = Av^{(j)}$ **for** $i = 1, 2, \dots, j$
$h_{ij} = (w^{(j)}, v^{(i)})$
$w^{(j)} = w^{(j)} - h_{ij}v^{(i)}$
end
$h_{i+1,j} = ||w_j||_2$ **if** $h_{i+1,j} = 0$, $m := j$; **BREAK**
$v^{(j+1)} = w^{(j)}/h_{i+1,j}$
end
Spočítajte $y_m = H_m^{-1}(\beta e_1)$ a $x_m = x_0 + V_m y_m$

Generalized Minimal Residual (GMRES)

- Zovšeobecnená metóda minimalných reziduí - Generalized Minimal Residual (GMRES)
- Každý vektor x_m z podpriestoru $x_0 + \mathcal{K}_m$ môžeme napísať $x = x_0 + V_m y_m$ a metóda potom minimalizuje normu reziduu nad všetkými vektormi z afinného podpriestoru $x_0 + \mathcal{K}_m$, t.j. minimalizuje funkcionál

$$\begin{aligned} J(y) &= ||b - Ax||_2 = ||b - A(x_0 + V_m y)||_2 && (8) \\ &= ||\beta e_1 - \tilde{H}_m y||_2 && (9) \end{aligned}$$

- Minimalizácia $J(y)$ je lacná na výpočet, pretože vyžaduje riešenie $(m+1) \times m$ problému najmenších štvorcov, kde m je zvyčajne malé.

Generalized Minimal Residual (GMRES)

- *Pseudo algoritmus GMRES metódy:*

Spočítajte $r^{(0)} = b - Ax^{(0)}$, $\beta = ||r^{(0)}||_2$ a $v^{(1)} = r^{(0)}/\beta$

Inizializujte maticu $(m+1) \times m$, $\tilde{H}_m = \{h_{ij}\}_{1 \leq i \leq m+1, 1 \leq \leq m} = 0$ **for** $j = 1, 2, \dots, m$

$w^{(j)} = Av^{(j)}$ **for** $i = 1, 2, \dots, j$

$h_{ij} = (w^{(j)}, v^{(i)})$

$w^{(j)} = w^{(j)} - h_{ij} v^{(i)}$

end

$h_{i+1,j} = ||w_j||_2$ **if** $h_{i+1,j} = 0$, $m := j$; **BREAK**

$v^{(j+1)} = w^{(j)}/h_{i+1,j}$

end

Spočítajte y_m ako minimum z $||\beta e_1 - \tilde{H} y||_2$ a $x_m = x_0 + V_m y_m$

Generalized Minimal Residual (GMRES)

- Spočítajte y_m ako minimum z $||\beta e_1 - \tilde{H}y||_2$??
  - Vypocítaj QR dekompozíciu: $\tilde{H}_m = Q_{m+1} \begin{pmatrix} R_m \\ 0 \end{pmatrix}$
  - $\tilde{g}_m = Q_{m+1}^T(\beta e_1) = \begin{pmatrix} g_m \\ \gamma_{m+1} \end{pmatrix}$
  - $y_m = R_m^{-1} g_m$
- Konvergenciu možno testovat podla normy rezídua, t.j. podla

$$||b - Ax_m||_2 = |\gamma_{m+1}| \tag{10}$$

Restarted Generalized Minimal Residual (GMRES)

- Algoritmus sa stáva nepraktický, keď m je veľké kvôli rastu pamäte a výpočtovým požiadavkám.
- Jeden zo sposobov je reštart a druhý je zalozeny na skrátení Arnoldiho ortogonalizácie.
- *Pseudo algoritmus restartovanej GMRES metódy:*

1. Spočítajte $r^{(0)} = b - Ax^{(0)}$, $\beta = ||r^{(0)}||_2$ a $v^{(1)} = r^{(0)}/\beta$
2. Inizializujte maticu $\tilde{H}_m$ použitím Arnoldiho algoritmu
3. Spočítajte y_m ako minimum z $||\beta e_1 - \tilde{H}y||_2$ a $x_m = x_0 + V_m y_m$
4. Ak je splnená zastavovanie podmienka skonči, inak $x_0 = x_m$ a pokračuj bodom 1

BiConjugate Gradient (BiCG)

- Metóda CG nie je vhodná pre nesymetrické systémy, pretože rezidu vektory nemôžu byť ortogonálne.
- Metóda GMRES zachováva ortogonalitu rezidui pomocou použitia dlhých opakovaní za cenu pamatovych narokov.
- Metóda BiConjugate Gradient (BiCG) má iný prístup a nahrádza ortogonálnu postupnosť rezíduí dvomi navzájom ortogonálnymi sekvenciami.
- vzťahy pre rezíduá v metóde CG sú rozšírené vzťahmi, ktoré sú podobné, ale založené na A^T namiesto A.

BiConjugate Gradient (BiCG)

- Rezidua potom updatujeme

$$r^{(i)} = r^{(i-1)} - \alpha_i A p^{(i)}, \qquad \tilde{r}^{(i)} = \tilde{r}^{(i-1)} - \alpha_i A^T \tilde{p}^{(i)}. \tag{11}$$

- smery hladame

$$p^{(i)} = r^{(i-1)} + \beta_{i-1} p^{(i-1)}, \qquad \tilde{p}^{(i)} = \tilde{r}^{(i-1)} + \beta_{i-1} \tilde{p}^{(i-1)}. \tag{12}$$

- a koeficienty vypocitame

$$\alpha_i = \frac{\tilde{r}^{(i-1)T} r^{(i-1)}}{\tilde{p}^{(i)T} A p^{(i)}} \qquad \beta_i = \frac{\tilde{r}^{(i)T} r^{(i)}}{\tilde{r}^{(i-1)T} r^{(i-1)}}. \tag{13}$$

- Biortogonalitu zabezpecuju vztahy

$$\tilde{r}^{(i-1)T} r^{(j)} = \tilde{p}^{(i)T} A p^{(j)} = 0 \qquad pre \quad i \neq j. \tag{14}$$

BiConjugate Gradient (BiCG)

1. *Compute* $r_0 := b - Ax_0$. *Choose* r_0^* *such that* $(r_0, r_0^*) \neq 0$.
2. *Set*, $p_0 := r_0$, $p_0^* := r_0^*$
3. *For* $j = 0, 1, \ldots,$ *until convergence Do:*
4. $\quad \alpha_j := (r_j, r_j^*)/(Ap_j, p_j^*)$
5. $\quad x_{j+1} := x_j + \alpha_j p_j$
6. $\quad r_{j+1} := r_j - \alpha_j A p_j$
7. $\quad r_{j+1}^* := r_j^* - \alpha_j A^T p_j^*$
8. $\quad \beta_j := (r_{j+1}, r_{j+1}^*)/(r_j, r_j^*)$
9. $\quad p_{j+1} := r_{j+1} + \beta_j p_j$
10. $\quad p_{j+1}^* := r_{j+1}^* + \beta_j p_j^*$
11. *EndDo*

Conjugate Gradient Squared(CGS)

- Metóda CGS je vhodná pre nesymetrické systémy.
- Pri rieseni nepouziva A^T .
- Konverguje rychlejsie ako BiCG pri rovnakych vypoctovych narokoch.

Conjugate Gradient Squared(CGS)

- V BiCG môžeme rezidualny vektor a smer gradientu v kroku i vyjadrit ako

$$r^{(i)} = \phi_i(A)r^{(0)}, \qquad p^{(i)} = \pi_i(A)r^{(0)}, \tag{15}$$

$$\tilde{r}^{(i)} = \phi_i(A^T)\tilde{r}^{(0)}, \qquad \tilde{p}^{(i)} = \pi_i(A^T)\tilde{r}^{(0)}, \tag{16}$$

kde ϕ_i a π_i su polynomy

$$\phi_{i+1}(t) = \phi_i(t) - \alpha_j t\pi_j(t) \qquad \pi_{i+1}(t) = \phi_{i+1}(t) + \beta_j\pi_j(t) \tag{17}$$

- Stalar α je potom definovany

$$\alpha_i = \frac{(\phi_i(A^T)\tilde{r}^{(0)})^T\phi_i(A)r^{(0)}}{(\pi_i(A^T)\tilde{r}^{(0)})^T A\pi_i(A)r^{(0)}} = \frac{\tilde{r}^{(0)T}\phi_i^2(A)r^{(0)}}{\tilde{r}^{(0)T}A\pi_i^2(A)r^{(0)}} \tag{18}$$

Conjugate Gradient Squared(CGS)

- Mozeme napisat

$$\begin{aligned} r^{(i)} &= \phi_i^2(A) r^{(0)}, && (19)\\ p^{(i)} &= \pi_i^2(A) r^{(0)}, && (20)\\ q^{(i)} &= \phi_{i+1}(A)\pi_i(A) r^{(0)}. && (21) \end{aligned}$$

- a v polynomialnom tvare

$$\begin{aligned} r^{(i+1)} &= r^{(i)} - \alpha_i A(2r^{(i)} + 2\beta_{i-1} q^{(i-1)} - \alpha_i A p^{(i)}), && (22)\\ q^{(i+1)} &= r^{(i)} + \beta_{i-1} q^{(i-1)} - \alpha_i A p^{(i)}, && (23)\\ p^{(i+1)} &= r^{(i+1)} + 2\beta_i q^{(i)} + \beta_i^2 p^{(i)}. && (24) \end{aligned}$$

- V kode oznacme $u^{(i)} = r^{(i)} + \beta_{i-1} q^{(1)}$

Conjugate Gradient Squared(CGS)

1. Compute $r_0 := b - Ax_0$; r_0^* arbitrary.
2. Set $p_0 := u_0 := r_0$.
3. For $j = 0, 1, 2 \ldots$, until convergence Do:
4. $\quad \alpha_j = (r_j, r_0^*)/(Ap_j, r_0^*)$
5. $\quad q_j = u_j - \alpha_j Ap_j$
6. $\quad x_{j+1} = x_j + \alpha_j(u_j + q_j)$
7. $\quad r_{j+1} = r_j - \alpha_j A(u_j + q_j)$
8. $\quad \beta_j = (r_{j+1}, r_0^*)/(r_j, r_0^*)$
9. $\quad u_{j+1} = r_{j+1} + \beta_j q_j$
10. $\quad p_{j+1} = u_{j+1} + \beta_j(q_j + \beta_j p_j)$
11. EndDo

BiConjugate Gradient Stabilized(BiCGSTAB)

- V BiCGSTAB pocita reziduá podla

$$r^{(i)} = \psi_i(A)\phi_i(A)r^{(0)}, \tag{25}$$

kde ψ je novy polynom ktoreho cielom je stabilizovat (vyhladit) konvergenciu algoritmu.

- Konkretne $\psi_i(t)$ je definovany

$$\psi_{i+1} = (1 - \omega_i t)\psi_i(t). \tag{26}$$

kde ω je skalar ktory treba vypocitat.

BiConjugate Gradient Stabilized(BiCGSTAB)

- Mozeme napisat

$$r^{(i)} = \phi_i(A)\psi_i(A)r^{(0)}, \tag{27}$$

$$p^{(i)} = \psi_i(A)\pi_i(A)r^{(0)}. \tag{28}$$

- a v polynomialnom tvare

$$r^{(i+1)} = (I - \omega_i A)(r^{(i)} - \alpha_i A p^{(i)}), \tag{29}$$

$$p^{(i+1)} = r^{(i)} + \beta_i (I - \omega_i A) p^{(i)}. \tag{30}$$

BiConjugate Gradient Stabilized(BiCGSTAB)

- Nakoniec mozeme napisat

$$r^{(i+1)} = s^{(i)} - \omega_i A s^{(i)} = r^{(i)} - \alpha_i A p^{(i)} - \omega_i A s^{(i)}, \quad (31)$$

$$x^{(i+1)} = x^{(i)} + \alpha_i p^{(i)} + \omega_i s^{(i)}. \quad (32)$$

  kde $s^{(i)} = r^{(i)} - \alpha_i A p^{(i)}$.

- ω_i je volana aby minimalizovala normu vektora $(I - \omega_i A)\psi_i(A)\phi_{i+1}(A)r^{(0)}$ v tvare

$$\omega_i = \frac{s^{(i)T} A s^{(i)}}{(As^{(i)})^T A s^{(i)}} \quad (33)$$

- Konstanty su pre BiCGSTAB vyjadrene v tvare

$$\alpha_i = \frac{r^{(i)T} \tilde{r}^{(0)}}{Ap^{(i)T} \tilde{r}^{(0)}}, \qquad \beta_i = \frac{r^{(i+1)T} \tilde{r}^{(0)}}{r^{(i)T} \tilde{r}^{(0)}} \times \frac{\alpha_i}{\omega_i}. \quad (34)$$

BiConjugate Gradient Stabilized(BiCGSTAB)

1. Compute $r_0 := b - Ax_0$; r_0^* arbitrary;
2. $p_0 := r_0$.
3. For $j = 0, 1, \ldots$, until convergence Do:
4. $\quad \alpha_j := (r_j, r_0^*)/(Ap_j, r_0^*)$
5. $\quad s_j := r_j - \alpha_j A p_j$
6. $\quad \omega_j := (As_j, s_j)/(As_j, As_j)$
7. $\quad x_{j+1} := x_j + \alpha_j p_j + \omega_j s_j$
8. $\quad r_{j+1} := s_j - \omega_j A s_j$
9. $\quad \beta_j := \frac{(r_{j+1}, r_0^*)}{(r_j, r_0^*)} \times \frac{\alpha_j}{\omega_j}$
10. $\quad p_{j+1} := r_{j+1} + \beta_j (p_j - \omega_j A p_j)$
11. EndDo

Zhrnutie

- Jakobiho metóda
  - Veľmi jednoduché použitie, ale ak matica nie je 'silne' diagonálne dominantná,
  - Metóda je pravdepodobne najlepšia len ako prepodmienovač.
  - Triviálne paralelizovateľná.

Zhrnutie

- Gauss-Seidelová metóda
  - Zvyčajne rýchlejšia konvergencia ako Jacobiho, ale vo všeobecnosti nie je konkurencieschopná s nestacionárnymi metódami.
  - Použiteľné pre striktne diagonálne dominantné, alebo symetrické pozitívne definitné matice. - Vlastnosti paralelizácie závisia od štruktúry matice koeficientov.
  - Rôzne usporiadania matice majú rôzny stupeň paralelizmu. Napr.: viacfarebné usporiadania .
  - Ide o špeciálny prípad metódy SOR, ktorú získame voľbou $\omega = 1$.

Zhrnutie

- Gauss Seidelová metóda
  - Zrýchľuje konvergenciu Gauss-Seidelovej metódy ($\omega > 1$, môže priniesť konvergenciu keď Gauss-Seidel zlyhá).
  - Rýchlosť konvergencie závisí v rozhodujúcej miere od ω. Optimálnu hodnotu pre ω možno odhadnúť zo spektrálneho polomeru Jacobiho iteračnej matice za určitých podmienok.
  - Vlastnosti paralelizácie sú rovnaké ako pri Gaussovej-Seidelovej metóde.

Zhrnutie

- Metóda združených gradientov (CG)
  - Použiteľná iba pre symetrické pozitívne definitné systémy.
  - Rýchlosť konvergencie závisí od čísla podmienenosti matice.
  - V prípade paralelizácie sú skalárne produkty synchronizačné body v paralelizácii.
  - Ďalšie paralelné vlastnosti sú do značnej miery nezávislé od štruktúry matice.

Zhrnutie

- Metóda zovšeobecnenéj minimálizácii reziduí (GMRES)
  - Použiteľná na nesymetrické matice.
  - GMRES vedie k najmenšiemu rezíduu pre pevný počet iteračných krokov, ale tieto kroky sa stávajú časovo náročné.
  - S cieľom obmedziť rastúce nároky na úložisko a prácu na jeden iteračný krok je potrebné reštartovanie.
  - GMRES vyžaduje iba maticovo-vektorové súčiny a nasobenie s maticou.
  - Počet vnútorných produktov rastie lineárne s počtom iterácií až do reštartu bodu.
  - Paralelizácia je komplikovanejšie.

Zhrnutie

- Metóda Bi (dvoj) združených gradientov (BiCG)
  - Použiteľné pre nesymetrické matice.
  - Vyžaduje maticovo-vektorové súčiny s maticou a jej transpozíciou. Toto diskvalifikuje metódu pre prípady, keď je matica daná len implicitne.
  - Vlastnosti paralelizácie sú podobné ako v prípade CG.

Zhrnutie

- Metóda združených gradientov na štvorec (CGS)
    - Použiteľná pre nesymetrické matice.
    - Konverguje (diverguje) zvyčajne približne dvakrát rýchlejšie ako BiCG.
    - Má tendenciu divergenovať, ak je počiatočný odhad blízko riešenia.
    - Výpočtové nároky na iteráciu sú podobné ako pri BiCG, ale metóda nevyžaduje transpoziciu maticu.

Zhrnutie

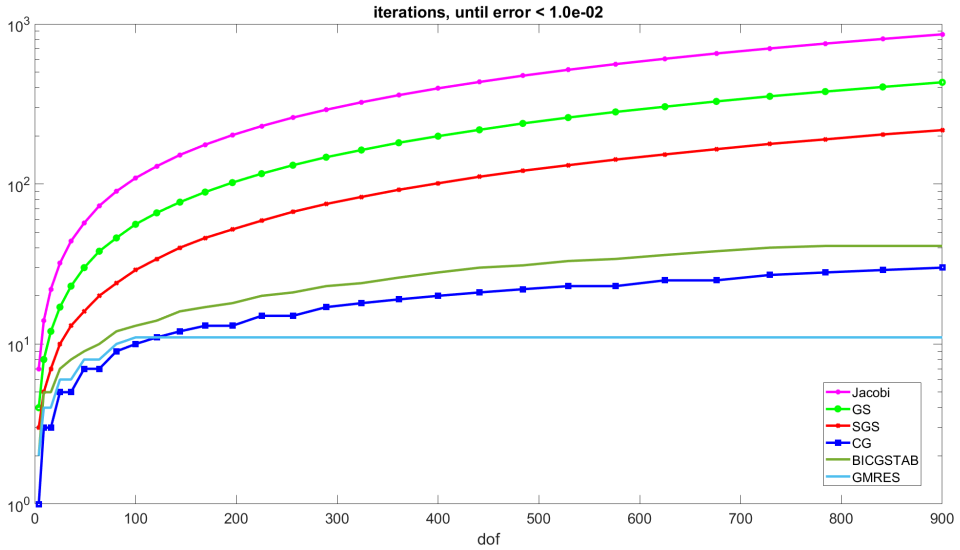
- Stabilizovaná Metóda Bi (dvoj) združených gradientov (BiCGSTAB)
  - Použiteľná na nesymetrické matice. - Výpočtové nároky na jednu iteráciu sú podobné ako pri BiCG a CGS, ale metóda nevyžaduje transponovanú maticu.
  - Alternatíva pre CGS, pričom zachováva približne rovnakú rýchlosť konvergencie.Pozorujeme menšiu stratu presnosti rezídua.

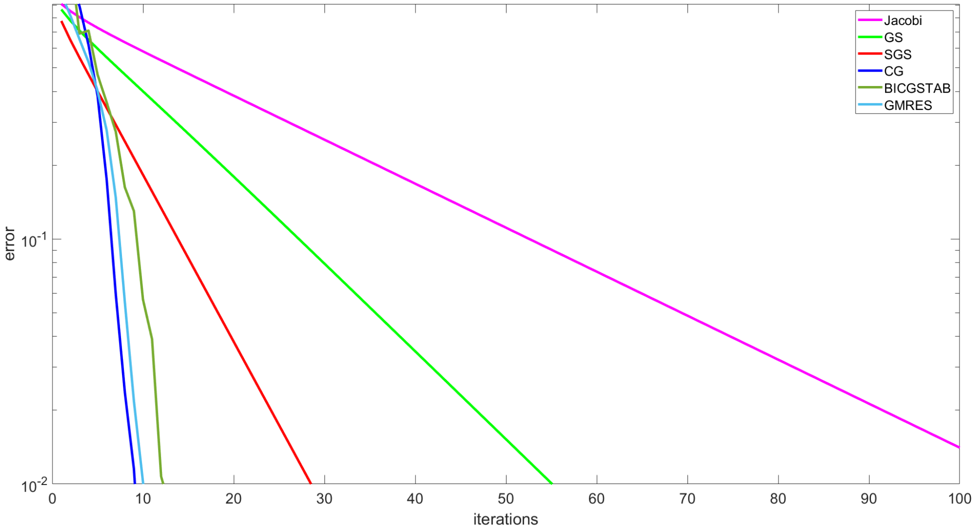
Zhrnutie

- Sumár operácii

Metóda	$\alpha = (a, b)$	$u = v + \alpha w$	$y = A \cdot x$	Pamäťové nároky
Jacobi			1	$A + 3n$
GS		1	1	$A + 3n$
SOR		1	1	$A + 2n$
CG	2	3	1	$A + 6n$
GMRES	i+1	i+1	1	$A + (i + 5)n$
BiCG	2	5	1	$A + 10n$
CGS	2	6	2	$A + 11n$
BiCGSTAB	4	6	2	$A + 10n$

kde A je matica rozmeru $n \times n$.


iterations, until error < 1.0e-02
10^3
10^2
10^1
10^0
0
100
200
300
400
500
600
700
800
900
dof
Jacobi
GS
SGS
CG
BICGSTAB
GMRES


Jacobi
GS
SGS
CG
BICGSTAB
GMRES
error
10^{-1}
10^{-2}
0
10
20
30
40
50
60
70
80
90
100
iterations