Premakni in obrni (Shift-and-invert)

Če iščemo lastne vrednosti v bližini τ, potem si lahko pomagamo s podprostorom Krilova, ki ga generira matrika $(A - \tau I)^{-1}$.

Pri tej metodi je potrebno produkt z matriko $(A - \tau I)^{-1}$ izračunati točno.

Pomagamo si lahko npr. z LU razcepom. če je $P(A - \tau I) = LU$, potem moramo v vsakem koraku Arnoldija namesto $v_{j+1} = (A - \tau I)^{-1} v_j$ novi vektor izračunati kot

$$Ly = Pv_j$$

$$Uv_{j+1} = y.$$

Če je A razpršena matrika, to ponavadi velja tudi za L in U, ki pa lahko imata več neničelnih elementov kot A. Najbolj zahtevna operacija pri premakni in obrni je izračun LU razcepa matrike $A - \sigma I$. Ker pa je konvergenca potem lahko zelo hitra in porabimo veliko manj korakov Arnoldijevega algoritma, se pristop premakni in obrni večinoma splača, kadar iščemo notranje lastne vrednosti.

Če iščemo kompleksne lastne vrednosti, je najbolje vzeti kompleksni τ. Vendar, potem moramo računati v kompleksni aritmetiki. Ali je možno priti do željenih kompleksnih lastnih vrednosti samo z realno aritmetiko?

Računanje kompleksnih lastnih vrednosti

Naj bo A realna nesimetrična matrika, radi pa bi uporabili premik $\sigma = \sigma_1 + i\sigma_2$.

Namesto kompleksne aritmetike pri direktni uporabi metode premakni in obrni lahko uporabimo tudi realni operator

$$B_+ = \mathrm{re}[(A - \sigma I)^{-1}] = \frac{1}{2}\left((A - \sigma I)^{-1} + (A - \overline{\sigma} I)^{-1}\right) .$$

Velja namreč:

- Lastni vektorji matrike B_+ se ujemajo z lastnimi vektorji matrike A.
- Lastni vrednosti μ_i^+ ustreza

$$\mu_i^+ = \frac{1}{2}\left(\frac{1}{\lambda_i - \sigma_i} + \frac{1}{\lambda_i - \overline{\sigma}_i}\right) .$$

Alternativne možnosti so

1. $B_- = \mathrm{im}[(A - \sigma I)^{-1}] = \dfrac{1}{2i}\left((A - \sigma I)^{-1} - (A - \overline{\sigma} I)^{-1}\right)$,
2. $B(\alpha, \beta) = \alpha B_+ + \beta B_-$,
3. $B_* = (A - \sigma I)^{-1}(A - \overline{\sigma} I)^{-1}$.

Računanje več lastnih vrednosti

Če nas zanima več kot le ena lastna vrednost, potem je ena izmed možnosti uporaba tehnik t.i. zaklepanja (lock-and-restart).

Denimo, da smo že izračunali lastne vektorje $x_1, \ldots, x_{j-1}$. Sedaj to vzamemo za prvih $j-1$ vektorjev $v_1, \ldots, v_{j-1}$, izberemo v_j in nadaljujemo s podprostorom, ki je kombinacija že izračunanih lastnih vektorjev in podprostora Krilova $\mathcal{K}(A, v_j)$.

Po k korakih dobimo podprostor, ki ga razpenjamo vektorji

$$v_1, \ldots, v_{j-1}, v_j, Av_j, \ldots, A^{k-j}v_{k-j}.$$

Za matriko H_k velja, da je podmatrika $H_k(1:j, 1:j)$ zgornja trikotna, kot približke za lastne vrednosti pa jemljemo v poštev le Ritzeve vrednosti spodnje podmatrike $H_k(j+1:k, j+1:k)$, ki je zgornja Hessenbergova.

Alternativna možnost je, da nadaljujemo računanje od začetka z matriko

$$\widetilde{A} = (I - U_{j-1}U_{j-1}^H)A(I - U_{j-1}U_{j-1}^H),$$

kjer stolpci matrike $U_{j-1} = [\, u_1 \;\; \cdots \;\; u_{j-1} \,]$ sestavljajo ortonormirano bazo za invariantni podprostor, ki smo ga že našli.

Algoritem za Arnoldija s premakni in obrni

izberi vektor v_1, začetni premik σ in nastavi $k = 1$

while $k \leq k_{\max}$

izračunaj LU razcep $A - \sigma I$ (*)

če je $k > 1$ izračunaj $h_{ij} = v_i^T (A - \sigma I)^{-1} v_j$ za $i, j = 1, \ldots, k-1$.

$j = k, k+1, \ldots, m$ (Arnoldi) (**)

$z = (A - \sigma I)^{-1} v_j$

$h_{ij} = v_i^T z$ za $i = 1, \ldots, j$

$h_{j+1,j} = \|z\|$

$v_{j+1} = z / h_{j+1,j}$

izračunaj:

- lastno vrednost H_m z največjo absolutno vrednostjo,
- pripadajoči približek za lastni vektor w matrike $(A - \sigma I)^{-1}$,
- ostanek ρ_k

vektor w ortgonaliziraj glede na $v_1, \ldots, v_{k-1}$ in ostanek vzemi za v_k

če je ρ_k dovolj majhen, vzemi v_k kot naslednji Schurov vektor in nastavi $k = k + 1$

če je bilo že preveč korakov z istim premikom, spremeni premik σ in pojdi na (*)

izvedi ponovni zagon (vrni se na (**))

9.1 Nesimetrični Lanczos za lastne vrednosti

izberi vektorja v_1, w_1, da je $v_1^T w_1 = 1$
$\beta_0 = \delta_0 = 0$, $v_0 = w_0 = 0$
$j = 1, 2, \ldots, k$
$\alpha_j = (Av_j, w_j)$
$\widetilde{v}_{j+1} = Av_j - \alpha_j v_j - \beta_{j-1} v_{j-1}$
$\widetilde{w}_{j+1} = A^T w_j - \alpha_j w_j - \delta_{j-1} w_{j-1}$
$\delta_j = |(\widetilde{v}_{j+1}, \widetilde{w}_{j+1})|^{1/2}$
če je $\delta_j = 0$, prekini računanje
$\beta_j = (\widetilde{v}_{j+1}, \widetilde{w}_{j+1})/\delta_j$
$v_{j+1} = \widetilde{v}_{j+1}/\delta_j$
$w_{j+1} = \widetilde{w}_{j+1}/\beta_j$

Dobimo

$$AV_k = V_k T_k + \delta_k v_{k+1} e_k^T,$$

$$A^T W_k = W_k T_k^T + \beta_k w_{k+1} e_k^T,$$

$$W_k^T A V_k = T_k,$$

kjer je

$$T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \delta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \delta_{k-1} & \alpha_k \end{bmatrix}.$$

Skalarja β_j in δ_j nista enolično določena, važno je le, da velja $\beta_j \delta_j = \widetilde{v}_{j+1}^T \widetilde{w}_{j+1}$.

Izrek 1. *Če se algoritem ne zalomi do koraka k, potem velja*

- $v_i^T w_j = 0$ *za* $i \neq j$ *in* $v_i^T w_i = 1$ *za* $i, j = 1, \ldots, k$,
- *stolpci* $V_k = [v_1 \ \cdots \ v_k]$ *tvorijo bazo za* $\mathcal{K}_k(A, v_1)$,
- *stolpci* $W_k = [w_1 \ \cdots \ w_k]$ *tvorijo bazo za* $\mathcal{K}_k(A^T, w_1)$.

Kdaj se nesimetrični Lanczos zalomi

izberi vektorja v_1, w_1, da je $v_1^T w_1 = 1$
$\beta_0 = \delta_0 = 0$, $v_0 = w_0 = 0$
$j = 1, 2, \ldots, k$
$\alpha_j = (Av_j, w_j)$
$\widetilde{v}_{j+1} = Av_j - \alpha_j v_j - \beta_{j-1} v_{j-1}$
$\widetilde{w}_{j+1} = A^T w_j - \alpha_j w_j - \delta_{j-1} w_{j-1}$
$\delta_j = |(\widetilde{v}_{j+1}, \widetilde{w}_{j+1})|^{1/2}$
če je $\delta_j = 0$, prekini računanje
$\beta_j = (\widetilde{v}_{j+1}, \widetilde{w}_{j+1})/\delta_j$
$v_{j+1} = \widetilde{v}_{j+1}/\delta_j$
$w_{j+1} = \widetilde{w}_{j+1}/\beta_j$

Dobimo

$$AV_k = V_k T_k + \delta_k v_{k+1} e_k^T,$$

$$A^T W_k = W_k T_k^T + \beta_k w_{k+1} e_k^T,$$

$$W_k^T A V_k = T_k,$$

kjer je

$$T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \delta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \delta_{k-1} & \alpha_k \end{bmatrix}.$$

Metoda se zalomi, če je $\delta_j = 0$. Do tega lahko pride zaradi:

- $\widetilde{v}_{j+1} = 0$ ali $\widetilde{w}_{j+1} = 0$: To je srečni zalom, saj so potem lastne vrednosti T_k enake lastnim vrednostim matrike A.
- $\widetilde{v}_{j+1} \neq 0$, $\widetilde{w}_{j+1} \neq 0$, toda $\widetilde{v}_{j+1}^T \widetilde{w}_{j+1} = 0$: To je resni zalom.

Kaj narediti, če se nesimetrični Lanczos zalomi

Če se nesimetrični Lanczos zalomi, imamo več možnosti:

- Restart: Takoj ko se zalomi (oziroma ko je numerično δ_j blizu 0), naredimo ponovni zagon. Na ta način sicer nadaljujemo, ampak izgubimo podprostor Krilova in možnost superlinearne konvergence.
- Look-Ahead: Pokvarimo tridiagonalno strukturo, tako da uporabimo več vektorjev in pridemo do bločno biortogonalne baze in bločne tridiagonalne matrike T_k. To pomeni, da lahko npr. najdemo par (v_{j+2}, w_{j+2}), čeprav par (v_{j+1}, w_{j+1}) ne obstaja.

Izguba biortogonalnosti

Podobno kot pri Lanczosevi metodi za simetrične matrike imamo tudi pri nesimetričnem Lanczosu težave z izgubo biortogonalnosti. Posledica so lahko navidezne lastne vrednosti, ki pa se za razliko od simetričnega primera lahko pojavijo kjerkoli.

Rešitve so:

- Dodatno testiranje, ki loči navidezne lastne vrednost od resničnih.
- Ponovni zagon.
- Dodatna biortogonalizacija, najprimernejša je selektivna biortogonalizacija.

Ocene pri nesimetričnem Lanczosu

Dobimo

$$AV_k = V_k T_k + \delta_k v_{k+1} e_k^T,$$
$$A^T W_k = W_k T_k^T + \beta_k w_{k+1} e_k^T,$$
$$W_k^T A V_k = T_k,$$

kjer je

$$T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \delta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \delta_{k-1} & \alpha_k \end{bmatrix}.$$

Pri nesimetričnem Lanczosu dobimo hkrati približke za leve in desne lastne vektorje. Če je $T_k s = \theta s$, potem

- θ je vrednost Petrova,
- $x = V_k s$ je desni vektor Petrova,
- $y = W_k s$ je levi vektor Petrova.

Za ostanka velja $r_R = Ax - \theta x = \delta_{k+1} v_{k+1} (e_k^T s)$ in $r_L^H = \beta_k w_{k+1}^H (s^H e_k)$.

Upoštevati moramo, da vektorja v_{k+1} in w_{k+1} nista normirana.

Nesimetrični Lanczos in harmonične vrednosti

Če uporabimo premik σ, je θ harmonična vrednost Petrova in u harmonični vektor, če je

$$Au - \theta u \perp (A^T - \sigma I)\mathcal{K}_k(A^T, w_1).$$

Lema 1. *Po k korakih nesimetričnega Lanczosa dobimo*

$$AV_k = V_k T_k + \delta_k v_{k+1} e_k^T = V_{k+1} T_{k+1,k},$$

$$A^T W_k = W_k T_k^T + \beta_k w_{k+1} e_k^T = W_{k+1} T_{k,k+1}^T,$$

$$W_k^T A V_k = T_k,$$

kjer je

$$T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \delta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \delta_{k-1} & \alpha_k \end{bmatrix}.$$

Če je ϑ lastna vrednost posplošenega problema lastnih vrednosti

$$\vartheta(T_{k,k+1} - \sigma I_{k,k+1})(T_{k+1,k} - \sigma I_{k+1,k})y = (T_k - \sigma I_k)y,$$

potem je $\sigma + 1/\vartheta$ harmonična vrednost Petrova matrike A.

Iterativne metode v Matlabu

V Matlabu je na voljo metoda `eigs`, ki računa lastne vrednosti s pomočjo Arnoldijeve metode z implicitnim ponovnim zagonom iz paketa ARPACK.

- `d = eigs(A)` vrne 6 po absolutni vrednosti največjih lastnih vrednosti matrike A.
- `[V,D] = eigs(A)` vrne vektor lastnih vektorjev V in diagonalno matriko lastnih vrednosti D za 6 po absolutni vrednosti največjih lastnih vrednosti A.
- `[V,D,flag] = eigs(A)` deluje tako kot zgoraj, le da dodatno vrne še ali so lastne vrednosti skonvergirale (flag=0) ali ne (flag≠0).
- `eigs(A,B)` rešuje posplošeni problem lastnih vrednosti $Ax = \lambda Bx$.
- `eigs(A,k)` vrne k največjih lastnih vrednosti.
- `eigs(A,k,sigma)` vrne k največjih lastnih vrednosti, pri čemer `sigma` pomeni:
  - če je sigma skalar, iščemo lastne vrednosti v bližini skalarja (preko premakni in obrni)
  - 'lm': največja absolutna vrednost,
  - 'sm': najmanjša absolutna vrednost,
  - 'lr': največji realni del,
  - 'li': največji imaginarni del,
  - 'si': najmanjši imaginarni del.

Matriko lahko podamo kot funkcijo, ki za x vrne Ax oziroma $(A - \sigma I)^{-1}x$ v primeru shift-and-invert.