Arnoldijev algoritam za nelinearne probleme svojstvenih vrijednosti

Šain, Ivana

Master's thesis / Diplomski rad

2014

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:217:860637

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-01-16**

Repository / Repozitorij:

Repository of the Faculty of Science - University of Zagreb

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Ivana Šain

ARNOLDIJEV ALGORITAM ZA NELINEARNE PROBLEME SVOJSTVENIH VRIJEDNOSTI

Diplomski rad

Voditelj rada:
prof.dr.sc. Zlatko Drmač

Zagreb, srpanj 2014.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Sadržaj **iii**

Uvod **2**

1 Krilovljevi potprostori i Hessenbergova forma **3**
1.1 Krilovljevi potprostori 3
1.2 Hessenbergova forma 4
1.3 QR iteracije 6

2 Arnoldijev algoritam **11**
2.1 Opis Arnoldijevog algoritma 12
2.2 Implicitno restartani Arnoldijev algoritam 14
2.3 Deflacija unutar IRA iteracija 20
2.4 ARPACK 26

3 Linearizacija QEP-a **29**
3.1 Linearizacija pridruženom formom 30
3.2 Beskonačne svojstvene vrijednosti 32
3.3 Primjena pomaka 35
3.4 Rješenje kvadratičnog problema 35

4 SOAR metoda **37**
4.1 Krilovljevi potprostori drugog reda 37

5 GSOAR. Implicitno restartani GSOAR **49**
5.1 Opis GSOAR metode 49
5.2 Implicitno restartanje 52
5.3 Odabir pomaka 54
5.4 Tretiranje deflacije tijekom implicitnog restartanja 56

6 Numerički eksperimenti **59**

6.1 Slučajno generirane matrice 59
6.2 Žiroskopski sustav 61
6.3 Neprigušeni sustav 63
6.4 Prigušeni sustav 66
6.5 Zaključak 68

Bibliografija **69**

Uvod

U ovom radu se bavimo kvadratičnim problemom svojstvenih vrijednosti (QEP - Quadratic Eigenvalue Problem). Točnije, za zadane matrice M, C, $K \in \mathbb{C}^{n\times n}$ želimo izračunati skalar $\lambda \in \mathbb{C}$ i vektor $z \in C^n$, $z \neq 0$ tako da vrijedi

$$Q(\lambda)z = \left(\lambda^2 M + \lambda C + K\right) z = 0. \tag{1}$$

Ovaj problem ima primjenu u mnogim područjima, kao što su analiza i kontrola vibracija mehaničkih sustava i linearnoj stabilnosti toka u mehanici fluida.
Predložit ćemo nekoliko metoda za rješavanje ovog problema. Također ćemo proučavati na koji način možemo izračunati samo određeni dio spektra, npr. onaj sa najvećim, odnosno najmanjim realnim dijelom. Promatrat ćemo i slučaj kada nas zanimaju svojsvene vrijednosti koje su blizu nekoj unaprijed zadanoj vrijednosti σ.
Osnovna ideja za rješavanje ovakvih problema je konstrukcija potprostora u $\mathbb{C}^n$ koji sadrže željenu spektralnu informaciju. Te prostore možemo efikasno zadati ortonormiranim bazama. Jedna klasa takvih potprostora su Krilovljevi potprostori. Arnoldijeva metoda daje ortogonalnu projekciju općenite matrice na Krilovljev potprostor zadane dimenzije, koja je najčešće manja od ukupne dimenzije problema. Metoda prvotno služi za svođenje matrice u Hessenbergovu formu, no ona također daje dobru aproksimaciju za neke od svojstvenih vrijednosti zadane matrice. Stoga, u prvom poglavlju, definiramo Krilovljeve potprostore i njihova osnovna svojstva, te definiramo Hessenbergovu formu matrice.
U drugom poglavlju opisujemo Arnoldijev algoritam za linearni problem svojstvenih vrijednosti. Uvest ćemo implicitno restartani algoritam, koji na osnovu trenutnog znanja o svojstvenim vrijednostima, ažurira početni vektor u svrhu bolje aproksimacije traženih svojstvenih vrijednosti. Također se bavimo problemom deflacije, te metodama *purginga* (odbacivanje svojstvenih vrijednosti koje su iskonvergirale, no nisu nam od interesa) i *lockinga* (zaključavanje svojstvenih vrijednosti koji su iskonvergirale i pripadaju skupu traženih vrijednosti).
U trećem poglavlju želimo iskoristiti Arnoldijev algoritam za rješenje kvadratičnog problema. Ideja je početni, nelinearni problem, linearizirati te novodefinirani problem riješiti opisanim metodom. Prvo definiramo generalizirani problem tako da definiramo matrice $G, L \in \mathbb{C}^{2n\times 2n}$ za koje vrijedi

$$Gy = \lambda Ly$$

i $y = \begin{bmatrix} \lambda z^T & z^T \end{bmatrix}$. Jedan primjer takve linearizacije je

$$G = \begin{bmatrix} C & K \\ I & 0 \end{bmatrix}, \quad \begin{bmatrix} -M & 0 \\ 0 & I \end{bmatrix}.$$

Zatim iz generaliziranog problema definiramo standardni problem svojstvenih vrijednosti

$$Sy = \lambda y,$$

tako da definiramo matricu

$$S = \begin{bmatrix} -M^{-1}C & -M^{-1}K \\ I & 0 \end{bmatrix}.$$

U četvrtom poglavlju opisujemo SOAR metodu. To je metoda Arnoldijevog tipa, koja umjesto linearizacije, koristi direktno matrice M, C, K pa samim time čuva strukturu problema. Vidjet ćemo da postoji veza između Arnlodijevog algoritma i SOAR-a. Naime, dokazat ćemo da dolazi do deflacije u istom koraku i jednog i drugog algoritma. Peto poglavlje predstavlja generaliziranu SOAR metodu. Ona nam omogućava da napravimo implicitno restartanje, te da odabiremo koji dio sprektra želimo računati. I ovdje, kao i kod Arnoldijevog algoritma, moramo definirati na koji će način naša metoda tretirati deflaciju. Konačno, u zadnjem poglavlju donosimo numeričke rezultate, te primjere iz primjene našeg problema.

Poglavlje 1

Krilovljevi potprostori i Hessenbergova forma

1.1 Krilovljevi potprostori

Teorija Krilovljevih potprostora i Hessenbergove forme matrice je detaljno razrađena u [1] i sljedeće definicije i teoremi su rađeni prema toj referenci. Kako su matrice koje definiraju problem (1) velike dimenzije, reda veličine 10^6 i veće i rijetko popunjene (tj. veći dio elemenata matrice je nula) bilo kakve transformacije nad takvim matricama su skupe. Također transformacije najčešće narušavaju formu matrice, pa ona više nije rijetko popunjena. Međutim, operacija množenja matrice sa vektorom se obavlja vrlo efikasno i dozvoljava iskorištavanje strukture matrice u svrhu ubrzanja izvršavanja operacije. Množenjem matrice A s vektorom x dobivamo vektor Ax, na kojeg opet možemo efikasno primijeniti matricu A te dobiti vektor A^2x i tako redom. Prema tome, prirodno je uvesti sljedeću definiciju:

Definicija 1.1.1. *Neka je $A \in \mathbb{C}^{n\times n}$ i $b \in \mathbb{C}^n$. Krilovljeva matrica reda i je definirana s $K_i \equiv K_i(A,b) = [b, Ab, \ldots, A^{i-1}b]$, a i-ti Krilovljev potprostor $\mathcal{K}_i$ je definiran kao slika od K_i, $\mathcal{K}_i \equiv \mathcal{K}_i(A,b) = \mathcal{R}(K_i) = \text{span}\{b, Ab, \ldots, A^{i-1}b\}$.*

Svojstva Krilovljevih potprostora su:

1. *Invarijantnost na skaliranje*: $\mathcal{K}_i(A,b) = \mathcal{K}_i(\beta A, \alpha b)$, $\alpha, \beta \neq 0$,

2. *Invarijantnost na translacije*: $\mathcal{K}_i(A - \sigma I, b) = \mathcal{K}_i(A,b)$,

3. *Zamjena baze*: Ako je U unitarna matrica, tada vrijedi $U\mathcal{K}_i(U^*AU, U^*b) = \mathcal{K}_i(A,b)$.

Zaista,

$$\begin{aligned}\mathcal{K}_i(A,b) &= [x, Ax, \ldots, A^{i-1}b] \\ &= U[U^*b, (U^*AU)U^*b, \ldots, (U^*AU)^{i-1}U^*b] \\ &= U\mathcal{K}_i(U^*AU, U^*b)\end{aligned}$$

Očito je da su za $n \times n$ matricu A stupci Krilovljeve matrice K_{n+1} linearno zavisni, jer potprostor od $\mathbb{C}^n$ ne može biti dimenzije veće od n. Također postoji $m \leq n$ takav da vrijedi

$$\mathcal{K}_1 \subsetneq \mathcal{K}_2 \subsetneq \cdots \subsetneq \mathcal{K}_m = \mathcal{K}_{m+1} = \ldots.$$

Za takav m je

$$K_{m+1}(A,b) = [b, Ab, \ldots, A^m b] \in \mathbb{C}^{n\times m+1}.$$

1.2 Hessenbergova forma

Defincije i dokazi iz ove i sljedeće sekcije su rađeni prema [2].

Definicija 1.2.1. *Kažemo da je $n \times n$ matrica H u Hessenbergovoj formi ako je $H_{ij} = 0$ za $i > j + 1$. H je strogo Hessenbergova ako je $H_{j+1,j} \neq 0$ za sve $j = 1, \ldots, n-1$.*

Sljedeći teorem definira Hessenbergovu formu općenite matrice.

Teorem 1.2.2. *Neka je $A \in \mathbb{C}^{n\times n}$. Postoje $n\times n$ unitarna matrica Q i Hessenbergova matrica H, tako da je $A = QHQ^*$. Ako je A realna matrica, onda Q možemo izabrati da bude realna ortogonalna, a H realna Hessenbergova. Ako je u dekompoziciji $A = QHQ^*$ matrica H strogo Hessenbergova, onda je ta dekompozicija jedinstveno određena u sljedećem smislu: Ako je $A = \tilde{Q}\tilde{H}\tilde{Q}^*$ također dekompozicija s unitarnom $\tilde{Q}$ i Hessenbergovom $\tilde{H}$, onda $\tilde{q}_1 = e^{i\phi_1}q_1$ povlači $\tilde{Q} = Q\Phi$, gdje je $\Phi = \operatorname{diag}(e^{i\phi_k})_{k=1}^n$. U slučaju realnih dekompozicija realne matrice A su svi $e^{i\phi_k} \in \{-1, 1\}$.*

Dokaz. Hessenbergovu formu ćemo dobiti tako da poništavamo elemente na pozicijama (i, j) s $i > j + 1$ i to dobro odabranim unitarnim transformacijama sličnosti. Pri tome ćemo koristiti Householderove reflektore. U matrici A uočimo elemente $A(2 : n, 1)$ i konstruirajmo $(n-1) \times (n-1)$ reflektor $\hat{Q}_1$ tako da je

$$\hat{Q}_1 A(2:n,1) = \|A(2:n,1)\|_2 e_1 = \begin{bmatrix}\hat{a}_{11}\\ 0\\ \vdots\\ 0\end{bmatrix}$$

Odavde slijedi, da je transformacija sličnosti

$$\begin{bmatrix} 1 & 0 \\ 0 & \hat{Q}_1^* \end{bmatrix} A \begin{bmatrix} 1 & 0 \\ 0 & \hat{Q}_1 \end{bmatrix} = A_2 \tag{1.1}$$

jedan korak prema Hessenbergovoj formi. Sada se koncentriramo na elemente $A_2(3:n,2)$ i na isti način konstruiramo $(n-2)\times(n-2)$ reflektor $\hat{Q}_2$ koji će poništiti pozicije $4:n$, tj. $\hat{Q}_2 A_2(3:n,2) = \|A_2(3:n,2)\|_2 e_1$. Zajedno sa transformacijom (1.1) imamo

$$\begin{bmatrix} I_2 & 0 \\ 0 & \hat{Q}_2^* \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \hat{Q}_1^* \end{bmatrix} A \begin{bmatrix} 1 & 0 \\ 0 & \hat{Q}_1 \end{bmatrix} \begin{bmatrix} I_2 & 0 \\ 0 & \hat{Q}_2 \end{bmatrix} = A_3, \tag{1.2}$$

gdje odmah uočavamo $A_3(4:n,3)$. Shema se sada očita: odredimo $(n-3)\times(n-3)$ Householderov reflektor $\hat{Q}_3$ tako da je $\hat{Q}_3 A_3(4:n,3) = \|A_3(4:n,3)\|_2 e_1$ i primijenimo novu transformaciju sličnosti

$$\begin{bmatrix} I_3 & 0 \\ 0 & \hat{Q}_3^* \end{bmatrix} \begin{bmatrix} I_2 & 0 \\ 0 & \hat{Q}_2^* \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \hat{Q}_1^* \end{bmatrix} A \begin{bmatrix} 1 & 0 \\ 0 & \hat{Q}_1 \end{bmatrix} \begin{bmatrix} I_2 & 0 \\ 0 & \hat{Q}_2 \end{bmatrix} \begin{bmatrix} I_3 & 0 \\ 0 & \hat{Q}_3 \end{bmatrix} = A_4. \tag{1.3}$$

Općenito imamo $n-2$ koraka, i u k-tom koraku konstruiramo unitarnu matricu Q_k tako da u matrici $A_{k+1} = Q_k^* A_k Q_k$ vrijedi $A_{k+1}(k+2:n,k) = 0$. Matrica Q_k je oblika $Q_k = \begin{bmatrix} I_k & 0 \\ 0 & \hat{Q}_k \end{bmatrix}$, gdje je $\hat{Q}_k$ $(n-k)\times(n-k)$ Householderov reflektor koji zadovoljava $\hat{Q}_k A_k(k+1:n,k) = \|A_k(k+1:n,k)\|_2 e_1$. Time je završena konstrukcija Hessenbergove forme: matrica $Q = Q_1 Q_2 \dots Q_{n-2}$ ima svojstvo da je $H = Q^* A Q$ gornje Hessenbergova.

Dokažimo sada jedinstvenost. Prvo uočimo da $H = Q^* A Q$ i $\tilde{H} = \tilde{Q}^* A \tilde{Q}$, zajedno sa $q_1 = e^{i\phi_1}\tilde{q}_1$, povlače $h_{11} = \tilde{h}_{11}$. Čitanjem prvih stupaca u $AQ = QH$ i $A\tilde{Q} = \tilde{Q}\tilde{H}$ dobijemo

$$Aq_1 = q_1 h_{11} + q_2 h_{21},\ Aq_1 e^{i\phi_1} = q_1 h_{11} e^{i\phi_1} + \tilde{q}_2 \tilde{h}_{21}$$

odakle slijedi $q_2 h_{21} = e^{-i\phi_1}\tilde{q}_2\tilde{h}_{21}$. Kako je po pretpostavci $h_{21} \neq 0$, zaključujemo da je $|\tilde{h}_{21}| = |h_{21}|$ i $\tilde{q}_2 = q_2 \frac{h_{21}}{\tilde{h}_{21}} e^{i\phi_1} = q_2 e^{i\phi_2}$. Odavde je $\tilde{h}_{22} = h_{22}$, $\tilde{h}_{21} = h_{21} e^{i(\phi_1-\phi_2)}$, $\tilde{h}_{12} = h_{12} e^{i(\phi_2-\phi_1)}$. Nastavljamo induktivno: pretpostavimo da smo za $m < n$ vektora dobili $\tilde{q}_j = q_j e^{i\phi_j}$, $j = 1,\dots,m$. Sada iz relacija

$$q_{m+1} h_{m+1,m} = Aq_m - \textstyle\sum_{j=1}^m q_j h_{jm},$$

$$\tilde{q}_{m+1}\tilde{h}_{m+1,m} = A\tilde{q}_m - \textstyle\sum_{j=1}^m \tilde{q}_j \tilde{h_{jm}} = Aq_m e^{i\phi_m} - \textstyle\sum_{j=1}^m q_j e^{i\phi_j} h_{jm} e^{i(\phi_m-\phi_j)}$$

zaključujemo da je $\tilde{q}_{m+1}\tilde{h}_{m+1,m} = q_{m+1} h_{m+1,m} e^{i\phi_m}$. Ostatak slijedi kao u slučaju $\tilde{q}_2$: $h_{m+1,m} \neq 0$ povlači $\tilde{h}_{m+1,m} \neq 0$ i

$$\tilde{q}_{m+1} = q_{m+1} \frac{h_{m+1,m}}{\tilde{h}_{m+1,m}} e^{i\phi_m} = q_{m+1} e^{i\phi_{m+1}}$$

□

1.3 QR iteracije

Kod metode implicitno restartanog Arnoldijevog algoritma koristit ćemo QR iteracije, prema tome u ovom odjeljku definiramo QR iteracije, sa naglaskom na iskorištavanje Hessenbergove forme matrice pri računanju iteracija. Za matricu A QR iteracije su dane algoritmom

Algoritam 1: $[A^k, k]$ =QR iteracije(A)

1 $A^{(1)} = A$;
2 $k = 1$;
3 do konvergencije ponavljaj
4 $A^{(k)} = Q^{(k)}R^{(k)}$ (QR faktorizacija);
5 $A^{(k+1)} = R^{(k)}Q^{(k)}$;
6 $k = k + 1$;

Teorem 1.3.1. *Matrice izračunate u algoritmu (1) imaju svojstva*

- *Za svaki k je* $A^{(k+1)} = \left(Q^{(k)}\right)^* A^{(k)}Q^{(k)}$, *tj. algoritam generira niz unitarno sličnih matrica.*
- *Za svaki k je* $A^{(k+1)} = \left(Q^{(1)}Q^{(2)}\dots Q^{(k)}\right)^* A \left(Q^{(1)}Q^{(2)}\dots Q^{(k)}\right)$.
- *Ako definiramo* $Q^{[1:k]} = Q^{(1)}Q^{(2)}\dots Q^{(k)}$ *i* $R^{[1:k]} = R^{(k)}R^{(k-1)}\dots R^{(1)}$, *onda je* $A^{(k)} = Q^{[1:k]}R^{[1:k]}$ *QR faktorizacija potencije* A^k.

Dokaz.

- $A^{(k+1)} = R^{(k)}Q^{(k)} = \left(Q^{(k)}\right)^* Q^{(k)}R^{(k)}Q^{(k)} = \left(Q^{(k)}\right)^* A^{(k)}Q^{(k)}$.
- Induktivno, koristeći prethodnu tvrdnju, imamo:

$$A^{(k+1)} = \left(Q^{(k)}\right)^* A^{(k)}Q^{(k)} = \left(Q^{(k)}\right)^* \left(Q^{(k-1)}\right)^* A^{(k-1)}Q^{(k-1)}Q^{(k)}.$$

- Proučavanjem prvih nekoliko potencija vidimo da vrijedi tvrdnja

$$\begin{aligned}
A^2 &= Q^{(1)}R^{(1)}Q^{(1)}R^{(1)} = \left(R^{(1)}Q^{(1)} = A^{(2)} = Q^{(2)}R^{(2)}\right) \\
&= Q^{(1)}Q^{(2)}R^{(2)}R^{(1)} = Q^{[1:2]}R^{[1:2]}, \\
A^3 &= Q^{(1)}R^{(1)}Q^{(1)}Q^{(2)}R^{(2)}R^{(1)} \\
&= Q^{(1)}Q^{(2)}R^{(2)}Q^{(2)}R^{(2)}R^{(1)} = \left(R^{(2)}Q^{(2)} = A^{(3)} = Q^{(3)}R^{(3)}\right) \\
&= Q^{(1)}Q^{(2)}Q^{(3)}R^{(3)}R^{(2)}R^{(1)} = Q^{[1:3]}R^{[1:3]}.
\end{aligned}$$

□

Pretpostavimo da je dana Hessenbergova matrica H. Ako je $H = QR$ QR faktorizacija matrice H,tada vrijedi da su matrice Q i RQ također Hessenbergove. Pogledajmo na primjeru 5×5 matrice

$$H = \begin{bmatrix} x & x & x & x & x \\ * & x & x & x & x \\ 0 & * & x & x & x \\ 0 & 0 & * & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix},$$

gdje trebamo poništiti elemente označene sa $*$. Za poništavanje pozicije $(2,1)$ koristimo Givensovu rotaciju $G^{(1)}$

$$\begin{bmatrix} c_1 & s_1 & 0 & 0 & 0 \\ -\bar{s}_1 & c_1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x & x & x & x & x \\ * & x & x & x & x \\ 0 & * & x & x & x \\ 0 & 0 & * & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & * & x & x & x \\ 0 & 0 & * & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix} = H^{(1)},$$

$H^{(1)} = G^{(1)}H$, i odmah prelazimo na poništavanje pozicije $(3,2)$ u $H^{(1)}$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c_2 & s_2 & 0 & 0 \\ 0 & -\bar{s}_2 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & * & x & x & x \\ 0 & 0 & * & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & * & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix} = H^{(2)},$$

$H^{(2)} = G^{(2)}H^{(1)}$. Dalje računamo $H^{(3)} = G^{(3)}H^{(2)}$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & c_3 & s_3 & 0 \\ 0 & 0 & -\bar{s}_3 & c_3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & * & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix} = H^{(3)},$$

i konačno $H^{(4)} = G^{(4)}G^{(3)}G^{(2)}G^{(2)} = R$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & c_4 & s_4 \\ 0 & 0 & 0 & -\bar{s}_4 & c_4 \end{bmatrix} \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & * & x \end{bmatrix} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix} = H^{(4)}.$$

Dakle matrica Q je oblika

$$Q = \begin{bmatrix} c_1 & s_1 & 0 & 0 & 0 \\ -\bar{s}_1 & c_1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}^* \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c_2 & s_2 & 0 & 0 \\ 0 & -\bar{s}_2 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}^* \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & c_3 & s_3 & 0 \\ 0 & 0 & -\bar{s}_3 & c_3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}^* \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & c_3 & s_3 & 0 \\ 0 & 0 & -\bar{s}_3 & c_3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}^* \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & c_4 & s_4 \\ 0 & 0 & 0 & -\bar{s}_4 & c_4 \end{bmatrix}^*$$

$$= \begin{bmatrix} \bar{c}_1 & -s_1\bar{c}_2 & s_1s_2\bar{c}_3 & -s_1s_2s_3\bar{c}_4 & s_1s_2s_3s_4 \\ \bar{s}_1 & \bar{c}_1\bar{c}_2 & -s_2\bar{c}_1\bar{c}_3 & s_2s_3\bar{c}_1\bar{c}_4 & -s_2s_3s_4\bar{c}_1 \\ 0 & \bar{s}_2 & \bar{c}_2\bar{c}_3 & -s_3\bar{c}_2\bar{c}_4 & s_3s_4\bar{c}_2 \\ 0 & 0 & \bar{s}_3 & \bar{c}_3\bar{c}_4 & -s_4\bar{c}_3 \\ 0 & 0 & 0 & \bar{s}_4 & \bar{c}_4 \end{bmatrix},$$

i vidimo da je ona Hessenbergova. Produkt gornje trokutaste i Hessenbergove matrice je opet Hessenbergova matrica, pa je prema tome RQ Hessenbergova. Iz ovih razmatranja vidimo da, ako na matricu H primjenimo QR iteracije, svaka od matrica $H^{(k)}$ i $Q^{(k)}$ će biti Hessenbergova.

Korak QR iteracija $H^{(k+1)} = \left(Q^{(k)}\right)^* H^{(k)} Q^{(k)}$ je transformacija unitarne sličnosti između dvije Hessenbergove matrice. To možemo shvatiti i kao redukciju na Hessenbergovu formu matrice koja je i sama već Hessenbergova, pri čemu unitarna matrica koja realizira tu redukciju ima i svojstvo da računa QR faktorizaciju matrice $H^{(k)}$. Sada nas zanima možemo li matricu $H^{(k+1)}$ dobiti direktno kao $\left(Q^{(k)}\right)^* H^{(k)} Q^{(k)}$, a ne u dva koraka kao u linijama (4) i (5) algoritma (1). Metoda koja nam to daje je *bulge-chasing* ili naganjanje kvrge. Metodu ćemo ilustrirati na primjeru 5×5. Uzmimo

$$H = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix}, \quad \tilde{Q}_1^* = \begin{bmatrix} c_1 & s_1 & 0 & 0 & 0 \\ -\bar{s}_1 & c_1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

gdje je $\tilde{Q}_1$ proizvoljna rotacija. Nakon transformacije sličnosti

$$\tilde{Q}_1^* H \tilde{Q}_1 = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ + & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix},$$

poziciju $(3, 1)$ poništavamo pomoću

$$\tilde{Q}_2^* = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c_2 & s_2 & 0 & 0 \\ 0 & -\bar{s}_2 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \tilde{Q}_2^* \tilde{Q}_1^* H \tilde{Q}_1 \tilde{Q}_2 = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & + & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix},$$

sada poziciju $(4,2)$ poništavamo s

$$\tilde{Q}_3^* = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & c_3 & s_3 & 0 \\ 0 & 0 & -\bar{s}_3 & c_3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \tilde{Q}_3^*\tilde{Q}_2^*\tilde{Q}_1^* H \tilde{Q}_1\tilde{Q}_2\tilde{Q}_3 = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & + & x & x \end{bmatrix}$$

i konačno element na poziciji $(5,3)$ poništavamo s

$$\tilde{Q}_4^* = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & c_4 & s_4 \\ 0 & 0 & 0 & -\bar{s}_4 & c_4 \end{bmatrix}, \quad \tilde{Q}_4^*\tilde{Q}_3^*\tilde{Q}_2^*\tilde{Q}_1^* H \tilde{Q}_1\tilde{Q}_2\tilde{Q}_3\tilde{Q}_4 = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix}.$$

Dakle, prva rotacija je u Hessenbergovoj matrici stvorila izbočinu ili kvrgu na poziciji $(3,1)$ (koju smo označili s $+$) i time pokvarila Hessenbergovu formu. Nakon toga, nizom rotacija smo tu kvrgu spuštali uzduž dijagonale, dok je nismo stjerali do donjeg desnog kuta, gdje je i nestala.

Ako označimo $\tilde{Q} = \tilde{Q}_1\tilde{Q}_2\tilde{Q}_3\tilde{Q}_4$, onda je matrica $\tilde{Q}^* H \tilde{Q}$ ponovno Hessenbergova matrica i ona ima istu strukturu kao i matrica dobivena QR transformacijama. Prema teoremu (1.2.2) o jedinstvenosti Hessenbergove forme, ako se te dvije matrice razlikuju, do na množenje kompleksnim brojem modula jedan, u prvom stupcu, onda se one na isti način razlikuju i u ostalim stupcima. Dakle, jedan korak QR algoritma možemo izvesti tako da matrica $\tilde{Q}_1$ bude ista ona rotacija koja se koristi pri računaju QR faktorizacije matrice $H^{(k)}$, a ostatak je samo *bulge-chasing*, odnosno korigiranje Hessenbergove forme nizom rotacija. Teorem o jedinstvenosti garantira da ćemo dobiti esencijalno isti rezultat kao i u algoritmu (1).

Poglavlje 2

Arnoldijev algoritam

Neka je zadana matrica $A \in \mathbb{C}^{n\times n}$. Želimo naći skalar λ i vektor $x \in \mathbb{C}^n$, $x \neq 0$ tako da vrijedi

$$Ax = \lambda x. \tag{2.1}$$

U tom slučaju, λ zovemo *svojstvena vrijednost* matrice A, a vektor x zovemo *svojstveni vektor* pridružen svojstvenoj vrijednosti λ. Ako je matrica reda n, onda ona ima n svojstvenih vrijednosti, s tim da one ne moraju biti nužno različite. Ovaj problem još zovemo standardni problem svojstvenih vrijednosti (SEP - Standard Eigenvalue Problem). Možemo promatrati i takozvani generalizirani problem svojstvenih vrijednosti (GEP - Generalized Eigenvalue Problem)

$$Ax = \lambda Bx. \tag{2.2}$$

Kod ovog problema zahtijevamo da je jedna od matrica A i B regularna, te ga svodimo na SEP na sljedeći način

$$B^{-1}Ax = \lambda x. \tag{2.3}$$

Arnoldijev algoritam je metoda koja se koristi za računanje podskupa svojstvenih vrijednosti matrice A, pri čemu je A velikog reda i najčešće rijetka matrica, tj. ima puno više nul od nenul elemenata. Arnoldijev algoritam nakon k koraka formira gornje Hessenbergovu matricu H_k, čije se svojstvene vrijednosti onda koriste za aproksimaciju podskupa svojstvenih vrijednosti velike matrice A. Matrica H_k je ortogonalna projekcija matrice A na zadani Krilovljev potprostor. Svojstvene vrijednosti matrice H_k se zovu *Ritzove vrijednosti* ili *Ritzove aproksimacije*. U ovom poglavlju ćemo opisati Arnoldijev algoritam, te Implicitno restartani Arnoldijev algoritam (IRA). Također, proučavat ćemo i tipove deflacije koji se mogu desiti tijekom IRA iteracija. Naime, implicitno restartirani Arnoldijev algoritam računa podskup svojstvenih vrijednosti koje imaju određena svojstva (npr. najveće ili najmanje realne dijelove). Taj traženi podskup ćemo nazivati *tražene* svojstvene vrijednosti, dok ćemo sve ostale zvati *neželjene* svojstvene vrijednosti. Iteriranjem se može dogoditi

da neke Ritzove aproksimacije konvergiraju puno prije nego je čitav skup željenih svojstvenih vrijednosti izračunat. Ovisno o tome da li su one u skupu traženih ili neželjenih vrijednosti primijenjuje se *locking* (zaključavanje), odnosno *purging* (čišćenje). Kako sam naziv kaže, *locking* zaključava željenu svojstvenu vrijednost, ona je sadržana u narednim faktorizacijama. *Purging*, s druge strane, izbacuje neželjene svojstvene vrijednosti, koje su konvergirale, iz sljedećih faktorizacija.

2.1 Opis Arnoldijevog algoritma

Ova sekcija je rađena prema [1] gdje je detaljno opisan Arnoldijev algoritam. Ako promatramo Krilovljev potprostor $\mathcal{K}_k(A, b)$, odmah uočavamo da je prirodna baza za taj potprostor upravo $\{b, Ab, \ldots, A^{k-1}b\}$. Međutim, vektori $A^k b$ konvergiraju u smjeru svojstvenog vektora koji odogovara najvećoj svojstvenoj vrijednosti matrice A, kako k raste. Prema tome, $A^k b$ zatvara sve manje kuteve sa linearnom ljuskom vektora $A^i b$, $i = 0, 1, \ldots, k-1$ što povećava osjetljivost QR faktorizacije na pogreške računanja u konačnoj aritmetici. Ideja je ortonormirati tu bazu. Ortonormiranu bazu za Krilovljeve potprostore nam daje Arnoldijev algoritam.
Pretpostavimo da je $\{v_1, \ldots, v_i\}$ ortonormirana baza za $\mathcal{K}_i$, $i \leq k$. Gram-Schmidtovim postupkom konstruiramo vektor v_{k+1} na sljedeći način:

$$y_k = A^k b - \sum_{i=1}^{k} v_i v_i^* A^k b, \quad v_{k+1} = \frac{y_k}{\|y_k\|}.$$

Norma $\|.\|$ predstavlja Euklidsku normu $\|.\|_2$. Tada je $\{v_1, \ldots, v_{k+1}\}$ ortonormirana baza za $\mathcal{K}_{k+1}(A, b)$. Vektor v_{k+1} se može računati ekonomičnije jer vrijedi

$$\begin{aligned}
\mathcal{K}_{k+1}(A, b) &= \mathcal{R}\left(\left[b, Ab, \ldots, A^k b\right]\right) = (v_1 = b/\|b\|) \\
&= \mathcal{R}\left(\left[v_1, Av_1, \ldots, A^k v_1\right]\right) = (Av_1 = \alpha_1 v_1 + \alpha_2 v_2, \alpha_2 \neq 0) \\
&= \mathcal{R}\left(\left[v_1, \alpha_1 v_1 + \alpha_2 v_2, A\left(\alpha_1 v_1 + \alpha_2 v_2\right), \ldots, A^{k-1}\left(\alpha_1 v_1 + \alpha_2 v_2\right)\right]\right) \\
&= \mathcal{R}\left(\left[v_1, v_2, Av_2, \ldots, A^{k-1} v_2\right]\right) \\
&\vdots \\
&= \mathcal{R}\left(\left[v_1, v_2, \ldots, v_{k-1}, Av_k\right]\right).
\end{aligned}$$

Dakle, umjesto da ortogonaliziramo vektor $A^k v_1$ u odnosu na $v_1, \ldots, v_k$ da dobijemo v_{k+1}, dovoljno je ortogonalizirati Av_k u odnosu na $v_1, \ldots, v_k$. Komponenta r_k vektora Av_k koja je ortogonalna sa $v_1, \ldots, v_k$ je dana sa

$$r_k = Av_k - \sum_{i=1}^{k} v_i (v_i^* A v_k). \tag{2.4}$$

Ako je $r_k = 0$ postupak staje, što znači da smo našli invarijantni potprostor span$\{v_1, \ldots, v_k\}$. Ako je $\|r_k\| > 0$, v_{k+1} dobijemo normalizacijom

$$v_{k+1} = \frac{r_k}{\|r_k\|}. \tag{2.5}$$

Neka je

$$h_{ik} = v_i^* A v_k.$$

Tada jednakosti (2.4) i (2.5) možemo pisati u obliku

$$Av_k = \sum_{i=1}^{k+1} v_i h_{ik}. \tag{2.6}$$

Time dolazimo do algoritma

Algoritam 2: $[V, H, l]$ =Arnoldi(A, b, k)

$v_1 = b/\|b\|_2$;
for $j = 1 : k$ **do**
  $r = Av_j$;
  for $i = 1 : j$ **do**
    $h_{ij} = v_i^* r$; $r = r - v_i h_{ij}$;
  end
  $h_{j+1,j} = \|r\|$;
  if $h_{j+1,j} = 0$ **then**
    $l = j$; $V = [v_1, \ldots, v_l]$;
    $H = (h_{ij})_{(l+1)\times l}$; STOP;
  end
  $v_{i+1} = \frac{r}{h_{j+1,j}}$;
end
$l = k$;
$V = [v_1, \ldots, v_k]$; $H = (h_{ij})_{(k+1)\times k}$;

Ako definiramo $V_k = [v_1, \ldots, v_k]$, jednadžbu (2.6) za $j = 1, \ldots, k$ možemo zapisati kao

$$AV_k = V_k H_k + r_k e_k^T. \tag{2.7}$$

Matrica H_k je Hessenbergova, prema tome, Arnoldijev algoritam možemo shvatiti i kao algoritam za računanje Hessenbergove forme matrice A. Pri tome, r_k predstavlja rezidual.

2.2 Implicitno restartani Arnoldijev algoritam

Ova metoda je detaljno razrađena u [3], te je ova sekcija rađena prema navedenom članku. Neka je (y, θ) svojstveni par matrice H_k, odnosno $H_k y = \theta y$. Taj par aproksimira svojstveni par (x, θ) matrice A, gdje je $x = V_k y$. Naime, iz (2.7) vidimo da vrijedi

$$\|Ax - x\theta\| = \|(AV_k - V_k H_k)y\| = |\beta e_k^T y| \tag{2.8}$$

gdje smo sa β označili $\|r_k\|$. Dakle, nakon k koraka Arnoldijevog algoritma, imamo k aproksimacija za svojstvenu vrijednost matrice A. Za hermitske matrice A u [14] su dokazani sljedeći teoremi o aproksimacijama svojstvenih vrijednosti:

Teorem 2.2.1. *Neka je $V \in R^{n\times k}$ matrica sa ortonormiranim stupcima. Neka je $H = V^*AV$ i $R = AV - VH$ rezidual. Ako su $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ svojstvene vrijednosti matrice A i $\theta_1 \geq \theta_2 \geq \ldots \geq \theta_k$ svojstvene vrijednosti matrice H tada postoje indeksi $i_1, i_2 \ldots i_k$ takvi da*

$$\max_j \left\{|\theta_j - \lambda_{i_j}|\right\} \leq \|R\|_2 \quad i \tag{2.9}$$

$$\sqrt{\sum_j \left(\theta_j - \lambda_{i_j}\right)^2} \leq \sqrt{2}\|R\|_F. \tag{2.10}$$

Ovaj teorem nam govori da, u hermitskom slučaju, imamo aproksimaciju k različitih svojstvenih vrijednosti. Dakle, ne može nam se desiti da nam sve svojstvene vrijednosti matrice H aproksimiraju samo jednu svojstvenu vrijednost matrice A. Ako imamo višestruke vrijednosti θ_i tada matrica A također ima višestruke svojstvene vrijednosti.
Implicitno restartani algoritam želi iskoristiti vezu reziduala i početnog vektora o kojem govori sljedeći teorem:

Teorem 2.2.2. *Neka (2.7) definira Arnoldijevu faktorizaciju matrice A nakon k koraka. Tada je $r_k = 0$ ako i samo ako je $v_1 = Q_k y$, gdje je $AQ_k = R_k Q_k$, $Q_k^T Q_k = I_k$ i R_k je gornje trokutasta matrica reda k.*

U teoremu (2.2.2), linearna ljuska k stupaca od Q_k predstavlja invarijantni potprostor za A. $AQ_k = R_k Q_k$ predstavlja realnu Schurovu dekompoziciju reda k za matricu A. Dijagonalni blokovi matrice R_k sadrže svojstvene vrijednosti matrice A. Kompleksno konjugirani parovi se nalaze u 2×2 blokovima, a realne svojstvene vrijednosti se nalaze na dijagonali. Ovaj teorem zapravo govori da ako je početni vektor linearna kombinacija k linearno nezavisnih svojstvenih vektora, tada je rezidual nakon k koraka jednak nuli. Prema tome zaključujemo, da ćemo imati bolju aproksimaciju ako je početni vektor linearna kombinacija svojstvenih vektora. Međutim, kako je naš zadatak zapravo naći svojstvene vektore, ne možemo očekivati da ćemo na početku moći zadati takav vektor. Međutim, kroz iteracije,

dobivamo bolju sliku o tome kako oni izgledaju, te to želimo iskoristiti u svrhu poboljšanja algoritma. Ideja sljedećeg pristupa je nakon nekoliko koraka promijeniti vektor q_1 te ponoviti iteracije sa tim promijenjenim vektorom. Dozvolit ćemo više koraka Arnoldijevog algoritma (npr. $k + p$), od traženog broja svojstvenih vrijednosti. Izdvojiti ćemo one svojstvene vrijednosti koje nas ne zanimaju (npr. ako nas zanima k najvećih svojstvenih vrijednosti, onda su nam ostalih p svojstvenih vrijednosti nepotrebne) i na neki način prigušiti njihov doprinos u početnom vektoru q_1. Ovaj proces želimo napraviti implicitno, to jest, da ne moramo svaki put iznova raditi Arnoldijeve iteracije sa novim početnim vektorom.
Fiksirajmo broj iteracija k, i neka je p neki drugi broj, te pretpostavimo da smo izveli $k + p$ Arnoldijevih iteracija, i dobili ortogonalnu matricu V_{k+p} takvu da

$$\begin{aligned} AV_{k+p} &= V_{k+p}H_{k+p} + r_{k+p}e_{k+p}^T \\ &= [V_{k+p}\ v_{k+p+1}]\begin{bmatrix} H_{k+p} \\ \beta_{k+p}e_{k+p}^T \end{bmatrix}. \end{aligned}$$

Neka je μ svojstvena vrijednost koja nam nije od interesa i neka $(H_{k+p} - \mu I) = QR$, gdje je Q ortogonalna i R gornje trokutasta. Tada vrijedi

$$(A - \mu I)V_{k+p} - V_{k+p}(H_{k+p} - \mu I) = r_{k+p}e_{k+p}^T, \tag{2.11}$$

$$(A - \mu I)V_{k+p} - V_{k+p}QR = r_{k+p}e_{k+p}^T. \tag{2.12}$$

Ako jednadžbu (2.12) pomnožimo sa Q, dobijemo

$$(A - \mu I)(V_{k+p}Q) - (V_{k+p}Q)(RQ) = r_{k+p}e_{k+p}^T Q, \tag{2.13}$$

$$A(V_{k+p}Q) - (V_{k+p}Q)(RQ + \mu I) = r_{k+p}e_{k+p}^T Q. \tag{2.14}$$

Označimo sa $V_+ = V_{k+p}Q$ i $H_+ = RQ + \mu I$. Tada je H_+ gornje Hessenbergova. Ako na matrice (2.12) primijenimo vektor e_1, kako bismo vidjeli odnos između prvih stupaca, dobijemo

$$(A - \mu I)v_1 = v_1^+ \rho_{11},$$

gdje je $\rho_{11} = e_1^T Re_1$ i $v_1^+ = V_+ e_1$. Dakle, početni vektor v_1 zamijenili smo vektorom $(A - \mu I)v_1$.
Ova ideja se proširuje na slučaj kada imamo p pomaka $(H - \mu_i I)$, $i = 1, \ldots, p$. QR iteracije primjenjene na matricu $(H - \mu_i I)$ daju gornje Hessenbergovu ortogonalnu matricu Q_i reda $k + p$ tako da

$$AV_{k+p}Q_i = [V_{k+p}Q_i,\ v_{k+p+1}]\begin{bmatrix} Q_i^T H_{k+p}Q_i \\ \beta_{k+p}e_{k+p}^T Q_i \end{bmatrix}.$$

Nakon primjene p implicitnih pomaka, dobijemo

$$AV_{k+p}^{+} = (V_{k+p}^{+}, v_{k+p+1}) \begin{bmatrix} H_{k+p}^{+} \\ \beta_{k+p} e_{k+p}^{T} \hat{Q} \end{bmatrix}, \tag{2.15}$$

gdje su $V_{k+p}^{+} = V_{k+p}\hat{Q}$, $H_{k+p}^{+} = \hat{Q}^{T} H_{k+p} \hat{Q}$ i $\hat{Q} = Q_1 Q_2 \dots Q_p$, uz to da je Q_j ortogonalna matrica koju daje QR faktorizacija matrice $H - \mu_j I$.
Napravimo particiju

$$V_{k+p}^{+} = [V_k^{+},\ \hat{V}_p], \quad H_{k+p}^{+} = \begin{bmatrix} H_k^{+} & M \\ \hat{\beta}_k e_1 e_k^{T} & \hat{H}_p \end{bmatrix}. \tag{2.16}$$

Kako su matrice Q_i došle iz QR transformacije Hessenbergove matrice, one su također Hessenbergove. To znači da matrica $\hat{Q}$ u zadnjem retku ima prvih $k-1$ elemenata jednakih nuli, dok ostali ne moraju nužno biti nula. Iz toga slijedi sljedeće:

$$\beta_{k+p} e_{k+p}^{T} \hat{Q} = \underbrace{(0, 0, \dots, \tilde{\beta}_{k+p}}_{k}, \underbrace{b^{T})}_{p}.$$

Uvrštavajući u (2.15), dobivamo

$$A[V_k^{+},\ \hat{V}_p] = [V_k^{+},\ \hat{V}_p,\ v_{k+p+1}] \begin{bmatrix} H_k^{+} & M \\ \hat{\beta}_k e_1 e_k^{T} & \hat{H}_p \\ \tilde{\beta}_{k+p} e_k^{T} & b^{T} \end{bmatrix}. \tag{2.17}$$

Izjednačavajući prvih k stupaca u (2.17) dobivamo

$$AV_k^{+} = V_k^{+} H_k^{+} + r_k^{+} e_k^{T}, \tag{2.18}$$

pa je

$$AV_k^{+} = [V_k^{+},\ v_{k+1}^{+}] \begin{bmatrix} H_k^{+} \\ \beta_k^{+} e_k^{T} \end{bmatrix}, \tag{2.19}$$

gdje je $v_{k+1}^{+} = \frac{1}{\beta_k^{+}} r_k^{+}$, $r_k^{+} = (\hat{V}_p e_1 \hat{\beta}_k + v_{k+p+1} \tilde{\beta}_{k+p})$ i $\beta_k^{+} = \|r_k^{+}\|$. Algoritam bi izgledao ovako

Algoritam 3: $[V, H, r]$ =Implicitno restartirani Arnoldi(A, k, p, tol, v_1)

```
1  v(:,1) = v_1; H = v_1^* A v_1; r = A v_1 - v_1 H;
2  Arnoldijev algoritam nad trenutnim podacima, vraća V, H, r;
3  for m = 1, 2, ... do
4      if ||r|| < tol then
5          STOP;
6      end
7      Arnoldijev algoritam nad trenutnim podacima, vraća V, H, r;
8      u = Shifts(H, p);
9      Q = I_{k+p};
10     for j = 1, 2, ..., p do
11         H = Q_j^* H Q_j;
12         Q = Q Q_j;
13     end
14     v = (VQ)e_{k+1}; V = (VQ);
15     r = (v e_{k+1}^T H e_k + r e_{k+p}^T Q e_k);
16 end
```

Linija (11) predstavlja *bulge-chasing* koji daje isti rezultat kao QR iteracija zbog teorema o jedinstvenosti Hessenbergove forme. Kao što smo već rekli, svaka primjena implicitnog pomaka μ_j zamjenjuje početni vektor v_1 sa $(A - \mu_j I)v_1$. Prema tome, nakon završenog ciklusa petlje u koraku (3) algoritma (3) imamo

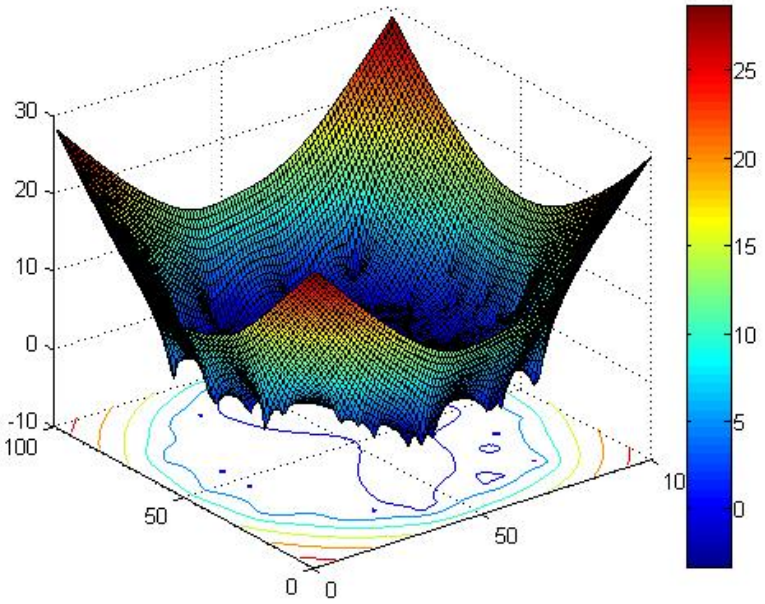
$$Ve_1 = v_1 = \psi(A)v_1,$$

gdje je $\psi(\lambda) = \frac{1}{\tau}\Pi_{j=1}^{p}(\lambda - m_j)$, a τ je normalizirajući faktor.
Postoji mnogo načina da se odabere p pomaka, jedan od njih je sljedeći

Algoritam 4: $[u]$ =Shifts(H, p)

```
1  izračunamo svojstvene vrijednosti matrice H;
2  izaberemo p svojstvenih vrijednosti koje nam nisu od interesa,
   {u(j) = μ_j : 1 ≤ j ≤ p} ⊂ σ(H);
```

Neki od kriterija za odabir $u(j)$ mogu biti da sortiramo spektar od H u odnosu na najveće realne dijelove svojstvenih vrijednosti i odaberemo p svojstvenih vrijednosti koje imaju najmanje realne dijelove za *shiftove*, ili sortiramo u odnosu na najveće po modulu, te izaberemo p najmanjih za *shiftove*.
Na sljedećoj slici je prikazan tipični polinomijalni filer koji je konstruiran tijekom IRA iteracija.



Slika 2.2.1: Polinomijalni filter

Cilj je bio izračunati 4 svojstvene vrijednosti, s najvećim realnim dijelovima, slučajno generirane matrice A reda 500. Na slici je nacrtan $log(|\psi(\lambda)|)$ iznad područja koje prestavlja spektar od A. ψ je produkt svih polinomijalnih filtera konstruiranih tijekom iteracija.

Čebiševljevi polinomi

Opisali smo kako napraviti polinomijalni filter koristeći pomake koje opisuje algoritam (3). Još jedan način konstrukcije polinomijalnih filtera su Čebiševljevi polinomi. Ta ideja je detaljno opisana u [4] i ova sekcija je rađena po tom članku. Pretpostavimo da su $\lambda_1, \lambda_2, \ldots, \lambda_n$ svojstvene vrijednosti matrice A poredane od najmanje do najveće i pretpostavimo da je λ_1 realna svojstvena vrijednost. Neka je zadan $v_k = p_k(A)v_1$, gdje je p_k polinom stupnja k. Želimo izabrati p_k tako da vektor v_k konvergira prema svojstvenom vektoru koji priprada svojstvenoj vrijednosti λ_1. Ako pretpostavimo da je A dijagonalizabilna, v_1 možemo zapisati u bazi svojstvenih vektora $\{x_i\}$

$$v_1 = \sum_{i=1}^{n} \theta_i x_i,$$

što daje sljedeći raspis za $v_k = p_k(A)v_1$

$$v_k = \sum_{i=1}^{n} \theta_i p_k(\lambda_i) x_i = \theta_1 p_k(\lambda_1) x_1 + \sum_{i=2}^{n} \theta_i p_k(\lambda_i) x_i. \tag{2.20}$$

Iz (2.20) vidimo da će v_k biti dobra aproksimacija za svojstveni vektor x_1 ako je drugi član mnogo manji u odnosu na prvi član. To znači da svaki $p_k(\lambda_j)$, $j \neq 1$ mora biti malen u odnosu na $p_k(\lambda_1)$. Dakle, moramo naći polinom koji poprima male vrijednosti na skupu

$$R = \{\lambda_2, \lambda_3, \ldots, \lambda_n\}$$

i koji zadovoljava normalizirajući uvjet

$$p_k(\lambda_1) = 1. \tag{2.21}$$

Takav polinom bi morao minimizirati sup-normu na diskretnom skupu R svih polinoma stupnja k koji zadovoljavaju uvjet (2.21). Međutim, takav polinom ne možemo izračunati bez da znamo sve svojstvene vrijednosti od A. Alternativa tome je tražiti polinom koji minimizira sup-normu na neprekidnom skupu koji sadrži R, ali ne sadrži λ_1. Označimo taj skup sa E i neka $\mathbb{P}_k$ predstavlja skup svih polinoma stupnja manjeg ili jednakog k. Tada je polinom koje tražimo zadan s

$$\min_{p \in \mathbb{P}_k, p(\lambda_1)=1} \max_{\lambda \in E} |p(\lambda)|. \tag{2.22}$$

Ovaj problem se rješava tako da se E restringira na elipsu čiji je centar na realnoj osi i sadrži neželjene svojstvene vrijednosti λ_i, $i = 2, 3, \ldots, n$. Neka je $E(c, e, a)$ elipsa koja sadrži skup $R = \{\lambda_2, \lambda_3, \ldots, \lambda_n\}$, gdje je c centar, $c + e$ i $c - e$ su fokusi, a je velika poluos elipse. Kada je A realna matrica, spektar je simetričan s obzirom na realnu os, pa možemo zahtjevati da i $E(c, e, a)$ bude simetrična. Velika poluos elipse je ili realna os ili je paralelna s imaginarnom osi. Prema tome, a i e su ili realne ili imaginarne (s realnim dijelom nula). U [4] je također pokazano da je polinom koji najbolje aproksimira rješenje problema (2.22) dan jednadžbom

$$p_k(\lambda) = \frac{C_k\left[(\lambda - c)/e\right]}{C_k\left[(\lambda_1 - c)/e\right]} \tag{2.23}$$

gdje je C_k Čebiševljev polinom stupnja k koji je dan rekurzijom

$$\begin{aligned} C_0(\lambda) &= 1, \\ C_1(\lambda) &= \lambda, \\ C_{k+1}(\lambda) &= 2\lambda C_k(\lambda) - C_{k-1}(\lambda), \quad k = 1, 2, \ldots \end{aligned}$$

Označimo $\rho_k = C_k\left[(\lambda_1 - c)/e\right]$, $k = 0, 1, \ldots$. Dobivamo

$$\rho_{k+1}p_{k+1}(\lambda) = C_{k+1}\left[\tfrac{\lambda-c}{e}\right] = 2\tfrac{\lambda-c}{e}\rho_k p_k(\lambda) - \rho_{k-1}p_{k-1}(\lambda).$$

Ako definiramo $\sigma_{k+1} = \rho_k/\rho_{k+1}$ dobivamo jednostavniju formulu

$$p_{k+1}(\lambda) = 2\sigma_{k+1}\tfrac{\lambda-c}{e}p_k(\lambda) - \sigma_k\sigma_{k+1}p_{k+1}(\lambda).$$

Sada se σ_i mogu računati rekurzijom

$$\sigma_1 = \frac{e}{\lambda_1 - c}$$
$$\sigma_{k+1} = \frac{1}{\frac{2}{\sigma_1} - \sigma_k}, \quad k = 1, 2, \ldots.$$

Time dolazimo do algoritma za računanje vektora v_k

Algoritam 5: Čebiševljeve iteracije

Izaberemo proizvoljni vektor v_0 i računamo
$\sigma_1 = \frac{e}{\lambda_1 - c}$;
$v_1 = \frac{\sigma_1}{e}(A - cI)\,v_0$;
for $i = 1 : k$ **do**
  $\sigma_{i+1} = \frac{1}{2/\sigma_1 - \sigma_i}$;
  $v_{i+1} = 2\frac{\sigma_{k+1}}{e}(A - cI)\,v_i - \sigma_i\sigma_{i+1}v_{i-1}$;
end

Kod ovog pristupa prvo pokrenemo Arnoldijev algoritam, te nakon m koraka dobijemo aproksimacije za svojstvene vrijednosti (θ_i, y_i). Izdvojimo k svojstvenih vrijednosti koje su nam od interesa, te definiramo skup R kojeg čine preostalih $m - k$ svojstvenih vrijednosti. Ako je došlo do konvergencije proces staje, a ako nije izračunamo parametre e i c elipse koristeći R. Zatim izračunamo vektor v_0 za Čebiševljeve iteracije kao lineranu kombinaciju vektora y_i, $i = 1, \ldots, k$. Provedemo k koraka Čebiševljevih iteracija da bi dobili v_k. Definiramo novi početni vektor $\tilde{v}_1 = v_k/\|v_k\|$ i pokrenemo ponovno Arnoldijev algoritam sa novim početnim vektorom koji ima bolju informaciju o traženim svojstvenim vrijednostima nego početni.

2.3 Deflacija unutar IRA iteracija

Problem deflacije, te metode *purginga* i *lockinga* su detaljno razrađeni u [5] te je nastavak ovog poglavlja rađen prema tom članku. Kao što smo rekli, definirat ćemo dvije vrste deflacije: *locking* i *purging*. Prije toga, dobro je vidjeti kako dođe do deflacije. Pretpostavimo da nakon m koraka Arnoldijevog algoritma imamo

$$A[V_1\ V_2] = [V_1\ V_2]\begin{bmatrix} H_1 & G \\ \beta e_1 e_j^T & H_2 \end{bmatrix} + re_m^T$$

gdje je $V_1 \in \mathbb{R}^{n\times j}$, $H_1 \in \mathbb{R}^{j\times j}$, $1 \leq j < m$. Ako je β dovoljno mali, faktorizacija se razdvaja u smislu da je Ritzov par (y, θ) za H_1 aproksimacija za svojstveni par $(V_1 y, \theta)$ uz ocjenu greške $|\beta e_j^T y|$. *Deflacijom* nazivamo postavljanje $\beta = 0$. Ako je β dovoljno mali, svojstvene vrijednosti od H_1 se mogu gledati kao Ritzove vrijednosti koje su iskonvergirale.
Ako je došlo do deflacije, stupci matrice V_1 se smatraju zaključanima, tj. došlo je do *lockinga*. To znači da se naredna implicitna restartanja provode na V_2, tj. podmatrice koje će se mijenjati su G, H_2 i V_2. Međutim, u fazi iteracija u kojoj se Arnoldijeva faktorizacija proširuje sa k na $k + p$ koraka, sudjeluju svi stupci matrice $[V_1\ V_2]$, kao da i nije došlo do deflacije. Ovo osigurava da su novi vektori ortogonalizirani u odnosu na Ritzove vektore koji su konvergirali, te sprečava ponovno nalaženje istih svojstvenih vrijednosti u sljedećim iteracijama.
Ako je došlo do deflacije, no neke od Ritzovih vrijednosti nisu u skupu željenih svojstvenih vrijednosi, potrebno ih je zajedno sa pripadnim svojstvenim vektorima ukloniti iz faktorizacije. To je čišćenje, odnoso *purging*.

Realizacija deflacije

Tijekom Arnoldijevih iteracija, Ritzova vrijednost može dobro aproksimirati svojsvenu vrijednost matrice A, bez da se na sporednoj dijagonali matrice H nalazi mali element (blizu nuli). Pogledamo jednadžbu (2.8). Vidimo da ako je zadnja komponenta Ritzovog vektora mala, ukupna ocjena $\|Ax - x\theta\| = \|(AV_k - V_k H_k)y\|$ će biti mala, što znači da θ dobro aproksimira neku sovjstvenu vrijednost matrice A. Međutim, ako dođe do konvergencije, uvijek možemo napraviti ortogonalnu transformaciju nakon koje će odgovarajući elementi na sporednoj dijagonali matrice H_k biti nula. U nastavku teksta ćemo ispustiti sve indekse matrica H, V i r radi jednostavnosti i preglednosti.
Neka je y Ritzov vektor, koji pripada Ritzovoj vrijednosti θ koja je iskonvergirala, tj. $Hy = \theta y$. Tada y možemo dopuniti do unitarne matrice

$$[y\ Y] = W. \tag{2.24}$$

Vrijedi

$$W^T H W = \begin{bmatrix} y^* \\ Y^* \end{bmatrix} H \begin{bmatrix} y & Y \end{bmatrix} = \begin{bmatrix} y^* \\ Y^* \end{bmatrix} \begin{bmatrix} \theta y & HY \end{bmatrix} = \begin{bmatrix} \theta & y^* HY \\ 0 & Y^* HY \end{bmatrix}.$$

Pomnožimo li Arnoldijevu rekurziju (2.6) s W zdesna, dobijemo

$$A(VW) = (VW)W^* HW + re_k^T W. \tag{2.25}$$

Sada nas zanima kako izgleda matrica W. Iz definicije (2.24) vidimo da mora vrijediti da je $We_1 = y$. Kako je W unitarna vrijedi $W = W^* = W^{-1}$ što znači da je $y = We_1$. Ovo je zapravo QR faktorizacija matrice, odnosno vektora, y. W će nam biti Householderov reflektor

$$W = I - \frac{2}{z^*z} z^*z, \tag{2.26}$$

dok ćemo z dobiti iz uvjeta $Wy = e_1$, odnosno $Wy = \tau e_1$. τ je slobodni parametar apsolutne vrijednosti jednake 1, koji je suprotnog predznaka od y_1 (prve komponente vektora y) kako bi se spriječilo katastrofalno kraćenje. Jednostavnim računom se dobije da je $z^*z = 2\,(1 - Re(\bar{\tau})y_1)$. Definiramo da je $\tau = \frac{-y_1}{|y_1|}$, odnosno $\tau = 1$ ukoliko je $y_1 = 0$, pa dobijemo $z^*z = 2 + 2|y_1|$. Sada konačno imamo da je

$$W = I - \frac{1}{1 + |y_1|} (y - \tau e_1)\,(y - \tau e_1)^*\,. \tag{2.27}$$

Primijetimo da (2.25) nije Arnoldijeva faktorizacija, naime matrica W^*HW više nije Hessenbergova, te nam je narušen zadnji stupac $re_k^T W$. Izračunajmo prvo što nam daje $re_k^T W$.

$$e_k^T W = e_k^T - \tfrac{1}{1+|y_1|} y_k\,(y - \tau e_1)^* = e_k^T + w^*.$$

Kako vrijedi

$$\|w^*\|_2 = \tfrac{|y_k|}{1+|y_1|}\|z\|_2 = \tfrac{|y_k|}{1+|y_1|}\,\sqrt{2}\,\sqrt{1 + |y_1|} = \tfrac{\sqrt{2}|y_k}{\sqrt{1+|y_1|}} \leq \sqrt{2}|y_k|,$$

vidimo da član rw^* možemo zanemariti jer je on po normi manji od $\sqrt{2}\|r\|\|e_k^T y|$, što je, po pretpostavci da je y Ritzov vektor koji je iskonvergirao, dovoljno malo.
Sada još matricu W^*HW moramo vratiti u Hessenbergovu formu. To radimo uzastopnom primjenom Householderovih transformacija, s tim da moramo paziti da ne smijemo narušiti zadnji član re_k^T. Prema tome imamo uvjet da za Householderovu matricu Y_i vrijedi $e_{k-1}^T Y_i = e_{k-1}^T$. Označimo sa Y matricu

$$Y = \begin{bmatrix} 1 & 0 \\ 0 & Y_1 Y_2 \ldots Y_{k-3} \end{bmatrix}. \tag{2.28}$$

Sada množenjem (2.25) zdesna s Y dobijemo Arnoldijevu faktorizaciju

$$AVWY = VWY(Y^*W^*HWY) + re_k^T Y + rw^*Y, \tag{2.29}$$

odnosno

$$AV = VH + re_k^T, \tag{2.30}$$

uz zamjenu $V = VWY$, $H = Y^*W^*HWY$ i odbacivanjem zadnjeg člana za kojeg smo rekli da je po normi malen.

Sada želimo ovaj koncept proširiti na slučaj kada imamo više Ritzovih vrijednosti koje su iskonvergirale, tj. koje treba zaključati.
Pretpostavimo da smo obavili m koraka Arnoldijevog algoritma, te da smo već zaključali j svojstvenih vrijednosti, odnosno prvih j stupaca matrice V razapinju invarijantan potprostor za A. Napravimo particiju

$$A\begin{bmatrix} V_j & \hat{V}_{m-j} \end{bmatrix} = \begin{bmatrix} V_j & \hat{V}_{m-j} \end{bmatrix} \begin{bmatrix} H_j & G_j \\ 0 & \hat{H}_{m-j} \end{bmatrix} + re_m^T + (rw^*)\,. \tag{2.31}$$

Pretpostavimo da imamo i novih Ritzovih vrijednosti matrice $\hat{H}_{m-j}$ koje su iskonvergirale, i označimo sa $X_i \in \mathbb{C}^{(m-j)\times i}$ matricu čiji su stupci pripadni svojstveni vektori. Tada vrijedi

$$\hat{H}_{m-j} X_i = X_i D_i,$$

gdje je D_i dijagonalna matrica, sa svojstvenim vrijednostima na dijagonali.
Napravimo li QR faktorizaciju matrice X_i

$$X_i = Q\begin{bmatrix} R_i \\ 0 \end{bmatrix} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}\begin{bmatrix} R_i \\ 0 \end{bmatrix} = Q_1 R_i,$$

možemo pisati

$$\hat{H}_{m-j} Q_1 = Q_1 \left(R_i D_i R_i^{-1} \right).$$

Zbog ortgonalnosti matrice Q je $Q_2^* \hat{H}_{m-j} Q_1 = 0$. Prema tome imamo

$$Q^* \hat{H}_{m-j} Q = \begin{bmatrix} R_i D_i R_i^{-1} & * \\ 0 & \hat{H}_{m-i-j} \end{bmatrix}.$$

Sada napravimo zamjenu

$$\hat{V}_{m-j} = \hat{V}_{m-j} Q,$$

$$\hat{H}_{m-j} = Q^* \hat{H}_{m-j} Q,$$

$$G_j = G_j Q.$$

Matrica $\hat{H}_{m-i-j}$ više nije Hessenbergova stoga je moramo vratiti u Hessenbergovu formu, što znači da moramo odrediti unitarnu matricu P takvu da je $P^* \hat{H}_{m-i-j} P$ Hessenbergova, pri čemu moramo paziti da ne pokvarimo zadnji stupac re_m^T. To znači da nam matrica P mora imati jedinicu u donjem desnom kutu, pa prema tome Householderove transformacije radimo nad retcima, umjesto nad stupcima. Napravimo zamjenu

$$\hat{H}_{m-i-j} = P^* \hat{H}_{m-i-j} P,$$

$$\hat{V}_{m-i-j} = \hat{V}_{m-i-j}P \text{ i}$$

$$G_{j+i} = G_{j+i}P.$$

Time je završen proces zaključavanja odnosno lockinga.

Čišćenje (purging)

Tijekom IRA iteracija se može desiti da nam iskonvergiraju Ritzove vrijednosti koje nam nisu od interesa. Želimo takve svojstvene vrijednosti izbaciti iz faktorizacije, odnosno želimo napraviti *purging*. Pogledajmo kako bismo napravili *purging* u slučaju da imamo samo jednu neželjenu svojstvenu vrijednost koja je iskonvergirala.
Neka smo napravili k koraka Arnoldijevog algoritma, te neka je prvi element na sporednoj dijagonali mali, tj. prvi stupac matrice V čini invarijantan potprostor za A

$$A\begin{bmatrix} v_1 & V_2 \end{bmatrix} = \begin{bmatrix} v_1 & V_2 \end{bmatrix}\begin{bmatrix} H_1 & G \\ \varepsilon e_1 e_1^T & H_2 \end{bmatrix} + re_k^T. \tag{2.32}$$

Označimo $H_1 = \alpha_1$. Izjednačavajući prve stupce dobijemo

$$Av_1 = v_1\alpha_1 + \varepsilon V_2 e_1.$$

Odavde slijedi da je

$$\|Av_1 - v_1\alpha_1\|_2 = |\varepsilon|. \tag{2.33}$$

Ako izjednačimo preostale retke dobijemo

$$AV_2 = V_1 G + V_2 H_2 + re_{m-1}^T. \tag{2.34}$$

Kada bi nam A bila simetrična matrica, čišćenje bi se jednostavno realiziralo tako da za G uzmemo $G = \varepsilon e_1^T$. No što ako A nije simetrična? Pitamo se da li postoji matrica E tako da vrijedi

$$(A + E)V_2 = V_2 H_2 + re_{m-1}^T. \tag{2.35}$$

Ako uzmemo $E = -\varepsilon v_1 (V_2 e_1)^T - \varepsilon (V_2 e_1) v_1^T$ (2.34) će biti zadovoljeno. Kako je $\|E\| = |\varepsilon|$ jednadžba (2.35) definira Arnoldijevu faktorizaciju duljine $m - 1$ za matricu koja je blizu početne matrice A. Svojstvena vrijednost α_1 se može izbaciti iz faktorizacije ako uzmemo da je $V = V_2$, $H = H_2$ i $G = 0$.
Ako imamo matricu A koja nije simetrična, onda trebamo napraviti određene transformacije kako bismo postigli da je $G = 0$. Označimo sa Z matricu

$$Z = \begin{bmatrix} 1 & -z^* \\ 0 & I \end{bmatrix}$$

i primijetimo da je njezin inverz matrica $\begin{bmatrix} 1 & z^* \\ 0 & I \end{bmatrix}$. Želimo naći vektor z tako da vrijedi

$$\begin{bmatrix} 1 & -z^* \\ 0 & I \end{bmatrix} \begin{bmatrix} \theta & \hat{h}^* \\ 0 & \hat{H} \end{bmatrix} \begin{bmatrix} 1 & z^* \\ 0 & I \end{bmatrix} = \begin{bmatrix} \theta & 0 \\ 0 & \hat{H} \end{bmatrix}.$$

Odnosno, treba vrijediti

$$\left(\hat{H}^* - \bar{\theta} I\right) z = \hat{h}.$$

Ova jednadžba će imati rješenje ukoliko je matrica $\left(\hat{H}^* - \bar{\theta}\right)$ regularna, odnosno ukoliko $\bar{\theta}$ nije svojstvena vrijednost od $\hat{H}^*$, tj. ukoliko θ nije svojstvena vrijednost od $\hat{H}$. Dakle pretpostavljamo da je svojstvena vrijednost koju želimo izbaciti jednostruka.
Pomnožimo sada relaciju (2.6) sa Z zdesna

$$AVZ = VZZ^{-1} \begin{bmatrix} \theta & \hat{h}^* \\ 0 & \hat{H} \end{bmatrix} Z + re_k^T Z + rw^* Z.$$

Primijetimo da je $e_k^T Z = e_k^T$, prema tome taj dio Arnoldijeve faktorizacije je ostao nenarušen. Izjednačavajući zadnjih $k-1$ stupaca dobivamo

$$AV \begin{bmatrix} z^* \\ I_{k-1} \end{bmatrix} = V \begin{bmatrix} z^* \\ I_{k-1} \end{bmatrix} \hat{H} + re_{k-1}^T + rw^* \begin{bmatrix} z^* \\ I_{k-1} \end{bmatrix}. \tag{2.36}$$

Matrica $\begin{bmatrix} z^* \\ I_{k-1} \end{bmatrix}$ nije ortogonalna, pa smo narušili ortogonalnost matrice V. Stoga ćemo napraviti QR faktorizaciju

$$\begin{bmatrix} z^* \\ I_{k-1} \end{bmatrix} = QR = \begin{bmatrix} q^* \\ Q_{21} \end{bmatrix} R. \tag{2.37}$$

Iz jednadžbe (2.37) odmah izlazi da je $R^{-1} = Q_{21}$. Supstitucijom u (2.36) dobivamo

$$AVQR = VQR\hat{H} + re_{k-1}^T + rw^* QR.$$

Množenjem s R^{-1} zdesna dobijemo

$$AVQ = VQR\hat{H}R^{-1} + re_{k-1}^T R^{-1} + rw^* Q. \tag{2.38}$$

Može se pokazati da je $|R_{k-1,k-1}| \geq 1$. To znači da se rezidual nije povećao, samo se mogao smanjiti.
Prema tome, čišćenje, odnosno purging završavamo supstitucijama

$$V = VQ, \quad H = R\hat{H}Q_{21}, \quad r = \frac{r}{R_{k-1,k-1}}.$$

Ova ideja se proširuje na slučaj kada imamo više svojstvenih vrijednosti (neka ih je i) koje su iskonvergirale, a nisu nam od interesa. Dakle, sada tražimo transformaciju oblika

$$\begin{bmatrix} I_i & -Z \\ 0 & I_{m-i} \end{bmatrix} H_m \begin{bmatrix} I_i & Z \\ 0 & I_{m-i} \end{bmatrix} = \begin{bmatrix} H_i & 0 \\ 0 & \hat{H}_{m-i} \end{bmatrix}.$$

Ovo se svodi na rješavanje Sylvesterove jednadžbe

$$Z\hat{H}_{m-i} - H_i Z = G_i. \tag{2.39}$$

Sada, kao i prije, napravimo QR faktorizaciju

$$\begin{bmatrix} Z \\ I_{m-i} \end{bmatrix} = QR_{m-i} = \begin{bmatrix} Q_i \\ Q_{m-i} \end{bmatrix} R_{m-i}. \tag{2.40}$$

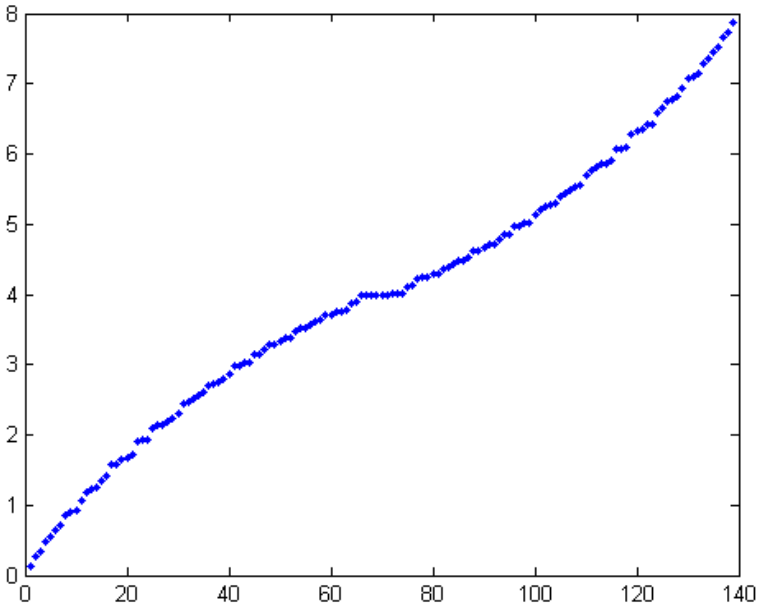
Purging završavamo nizom zamijena

$$H_{m-i} = R_{m-i}\hat{H}_{m-i}Q_{m-i} \quad V_{m-i} = V_m Q \quad r_{m-i} = \frac{1}{R_{m-i,m-i}} r_m.$$

Time smo dobili Arnoldijevu faktorizaciju reda $m - i$. Treba napomenuti da se prije ovog procesa moraju zaključati svojstvene vrijednosti koje želimo izbaciti, te se moraju nalaziti u vrhu matrice H_m.

2.4 ARPACK

ARPACK je biblioteka za MATLAB koja računa svojstvene vrijednosti i svojstvene vektore za velike matrice. U MATLABU postoji funkcija `eigs` koja računa zadani broj svojstvenih vrijednosti određenog tipa. Ona koristi implicitno restartani Arnoldijev algoritam. Više o ARPACKU se može pročitati u [6]. Jedan primjer izvedbe ove rutine je dan na [7] pod nazivom `arpack_test01.m`. On računa 5 svojstvenih vrijednosti najvećeg i 5 svojstvenih vrijednosti najmanjeg modula matrice A čiji je spektar prikazan na slici (2.4.1). Spektar je realan i svojstvene vrijednosti su prikazane na y-osi.



Slika 2.4.1: Spektar matrice A

Kao rezultat su dobivene sljedeće svojstvene vrijednosti

Najmanje po modulu	**Najveće po modulu**
5.519736907587849e-01	7.866584200423666e+00
4.787118036070203e-01	7.732433336220810e+00
3.468930344689255e-01	7.653106965531071e+00
2.675666637791856e-01	7.521288196392966e+00
1.334157995763294e-01	7.448026309241232e+00

Poglavlje 3

Linearizacija QEP-a

Nakon što smo razradili Arnoldijev algoritam za linearni problem svojstvenih vrijednosti, prirodno je pokušati kvadratni problem linearizirati, te ga riješiti sa već razvijenim algoritmom. Dakle umjesto da rješavamo problem (1) s matricama M, C i K koje su reda n, rješavamo problem oblika

$$(G - \lambda L)\begin{bmatrix}\lambda z\\ z\end{bmatrix} = 0. \tag{3.1}$$

Tri su problema koja nastaju kod korištenja Arnoldijevog algoritma. Prvo, problem nam se udupla, prema tome potrebno nam je više memorije za spremanje podataka. Drugo, tipična struktura svojstvenih vektora se gubi linearizacijom. Treće, Ritzove vrijednosti se računaju iz male Hessenbergove matrice, a ne za manji kvadratični svojstveni problem.
Te probleme ćemo pokušati riješiti SOAR metodom koju ćemo predstaviti u sljedećem poglavlju. Najveći problem kod SOAR metode je što ne znamo kako izračunati Schurovu formu lineariziranog problema.
Svojstveni vektori kvadratičnog problema se pojavljuju dva puta u svojstvenim vektorima od (3.1), kao z i kao λz. Oni su u principu različiti, stoga moramo definirati koji od njih odabrati.
Linearizacija se obavlja u dva koraka. U prvom koraku transformiramo QEP u ekvivalentni generalizirani problem svojstvenih vrijednosti

$$Gz = \lambda Lz. \tag{3.2}$$

Drugi korak reducira generalizirani svojstveni problem u linearni svojstveni problem

$$Sx = \lambda x \tag{3.3}$$

gdje je $S = G^{-1}L$ ili $S = L^{-1}G$. Nakon linearizacije, naš problem se udvostručuje, tj. matrica s kojom radimo je reda $2n \times 2n$. Ovo nam na neki način ograničava probleme koje

možemo riješiti jer nam se povećavaju zahtjevi za memorijom. Prilikom biranja linearizacije moramo paziti na to da možemo efiksano i točno izračunati S. Problem linearizacije je detaljno obrađen u [8] i prema tom članku su rađene sekcije o linearizaciji.

3.1 Linearizacija pridruženom formom

Kod ove linearizacije biramo

$$G = \begin{bmatrix} D & 0 \\ 0 & K \end{bmatrix}, \quad L = \begin{bmatrix} 0 & D \\ -M & -C \end{bmatrix}, \quad z = \begin{bmatrix} \lambda x \\ x \end{bmatrix}, \tag{3.4}$$

gdje je D proizvoljna regularna matrica. Time smo definirali generalizirani problem (3.2). Da bi prešli na linearni problem (3.3) moramo zahtijevati da je matrica K invertibilna kako bi G^{-1} imalo smisla, jer je

$$G^{-1} = \begin{bmatrix} D^{-1} & 0 \\ 0 & K^{-1} \end{bmatrix}.$$

Napomenimo da je proizvoljnost matrice D opravdana činjenicom da u konačnici S

$$S = G^{-1}L = \begin{bmatrix} 0 & I \\ -K^{-1}M & -K^{-1}C \end{bmatrix} \tag{3.5}$$

ne ovisi o D. Ovu formu zovemo *prva pridružena forma*.
Kao alternativu ovoj linearizaciji možemo promatrati

$$L = \begin{bmatrix} -M & 0 \\ 0 & D \end{bmatrix}, \quad G = \begin{bmatrix} C & K \\ D & 0 \end{bmatrix}, \quad z = \begin{bmatrix} \lambda x \\ x \end{bmatrix}, \tag{3.6}$$

s tim da u ovom slučaju S računamo kao $S = L^{-1}G$, pa prema tome moramo zahtijevati da je M regularna. Ovaj pristup daje

$$S = \begin{bmatrix} -M^{-1}C & -M^{-1}K \\ I & 0 \end{bmatrix}. \tag{3.7}$$

Ovu formu zovemo *druga pridružena forma*.

Još neke linearizacije

Promatramo vektorski prostor linearizacija oblika

$$\mathbb{L}_1(Q) = \left\{ G - \lambda L : (G - \lambda L) \begin{bmatrix} \lambda \\ 1 \end{bmatrix} = \begin{bmatrix} \eta_1 Q(\lambda) \\ \eta_2 Q(\lambda) \end{bmatrix}, \eta_{1,2} \in \mathbb{C} \right\}. \tag{3.8}$$

Napišimo G i L kao

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix},$$

gdje su matrice G_{ij} i L_{ij} reda $n \times n$. Sada izjednačavanjem faktora uz potencije od λ u

$$\begin{bmatrix} -\lambda^2 L_{11} + \lambda(G_{11} - L_{12}) + G_{12} \\ -\lambda^2 L_{21} + \lambda(G_{21} - L_{22}) + G_{22} \end{bmatrix} = \begin{bmatrix} \lambda^2(\eta_1)M + \lambda(\eta_1 C) + \eta_1 K \\ \lambda^2(\eta_2)M + \lambda(\eta_2 C) + \eta_2 K \end{bmatrix},$$

dobivamo

$$G = \begin{bmatrix} G_{11} & \eta_1 K \\ G_{21} & \eta_2 K \end{bmatrix} \text{ i } \quad L = \begin{bmatrix} -\eta_1 M & G_{11} - \eta_1 C \\ -\eta_2 M & G_{21} - \eta_2 C \end{bmatrix},$$

gdje su G_{11}, G_{21}, η_1 i η_2 proizvoljni. Uočimo da je ovo generalizacija prve pridružene forme (uz $\eta_1 = 0$, $\eta_2 = 1$ i $G_{21} = 0$ dobivamo (3.4)).
Sada promatramo prostor

$$\mathbb{L}_2(Q) = \left\{ G - \lambda L : \begin{bmatrix} \lambda I & I \end{bmatrix} (G - \lambda L) = \begin{bmatrix} \tilde{\eta}_1 Q(\lambda) & \tilde{\eta}_2 Q(\lambda) \end{bmatrix}, \tilde{\eta}_{1,2} \in \mathbb{C} \right\}. \tag{3.9}$$

Kao u prethodnom primjeru, linearizaciju

$$G = \begin{bmatrix} G_{11} & G_{12} \\ \tilde{\eta}_1 K & \tilde{\eta}_2 K \end{bmatrix} \text{ i } \quad L = \begin{bmatrix} -\tilde{\eta}_1 M & -\tilde{\eta}_2 M \\ G_{11} - \tilde{\eta}_1 C & G_{12} - \tilde{\eta}_2 C \end{bmatrix},$$

$$LG^{-1} = \begin{bmatrix} 0 & -MK^{-1} \\ I & -CK^{-1} \end{bmatrix},$$

dobivamo rješavajući

$$\begin{bmatrix} -\lambda^2 L_{11} + \lambda(G_{11} - L_{21}) + G_{21} & -\lambda^2 L_{12} + \lambda(G_{12} - L_{22}) + G_{22} \end{bmatrix} = \\ \begin{bmatrix} \lambda^2(\tilde{\eta}_1 M) + \lambda(\tilde{\eta}_1 C + \tilde{\eta}_1 K & \lambda^2(\tilde{\eta}_2 M) + \lambda(\tilde{\eta}_2 C + \tilde{\eta}_2 K \end{bmatrix}.$$

Ovo je generalizacija druge pridružene forme (3.6). Uočimo da $G^{-1}L$ ovisi o parametrima G_{11}, G_{12} i $\tilde{\eta}_{1,2}$. Presjek prostora $\mathbb{L}_1$ i $\mathbb{L}_2$ je prostor $\mathbb{DL}(Q)$ čija je općenita forma

$$G = \begin{bmatrix} \eta_1 C - \eta_2 M & \eta_1 K \\ \eta_1 K & \eta_2 K \end{bmatrix} \quad i \quad L = \begin{bmatrix} -\eta_1 M & -\eta_2 M \\ -\eta_2 M & -\eta_2 C + \eta_1 K \end{bmatrix}. \tag{3.10}$$

Kada odaberemo linearizaciju, s matricom S ulazimo u Arnoldijev algoritam (2). Dakle, dobijemo sljedeću faktorizaciju

$$S V_k = V_k H_k + r_k e_k^T \tag{3.11}$$

za k koraka algoritma.
Algoritam možemo napraviti efikasnijim za naš uduplani problem sa matricom S. Pretpostavimo da smo odabrali linearizaciju (3.5). Napravimo dekompoziciju j-tog Arnoldijevog vektora, tj. j-tog stupca matrice V_k u (3.11)

$$V(:, j) = \begin{bmatrix} v_j \\ w_j \end{bmatrix},$$

gdje su nam v_j, $w_j \in \mathbb{C}^n$.
Sada Arnoldijevu rekurziju možemo zapisati kao

$$\begin{bmatrix} 0 & I \\ -K^{-1}M & -K^{-1}C \end{bmatrix} \begin{bmatrix} V_k \\ W_k \end{bmatrix} - \begin{bmatrix} V_k \\ W_k \end{bmatrix} H_k = \beta_k \begin{bmatrix} v_{k+1} \\ w_{k+1} \end{bmatrix} e_k^T, \tag{3.12}$$

gdje nam je $\beta_k = \|r_k\|$. Izjednačavajući prvih k redaka iz gornje jednadžbe dobivamo

$$W_k = V_k H_k + \beta_k \begin{bmatrix} v_{k+1} \\ w_{k+1} \end{bmatrix} e_k^T. \tag{3.13}$$

Sada vidimo da u Arnoldijevom algoritmu trebamo spremati samo vektore V_k, v_{k+1} i w_{k+1} kako bi dobili faktorizaciju (3.11).
Ovo razmatranje sada uklopimo u algoritam (2), s tim da za ulazne podatke sada imamo dva vektora v_1 i w_1 reda n koji zadovoljavaju $\|v_1\|_2^2 + \|w_1\|_2^2 = 1$ kako bi osigurali da su vektori u V_k normirani.

3.2 Beskonačne svojstvene vrijednosti

Ova sekcija je rađena prema [9] u kojem je detaljno razrađen problem izdvajanja svojstvene vrijednosti ∞. Za kvadratični problem (1) definiramo inverzni problem

$$\text{rev}\,(Q(\lambda)) = \left(\frac{1}{\lambda}K + \frac{1}{\lambda}C + M\right) z = 0. \tag{3.14}$$

Dakle svojstvene vrijednosti prvog problema su recipročne svojstvene vrijednosti drugog problema, i obratno. Ako je 0 svojstvena vrijednost problema (3.14) onda je matrica M singularna, i obratno. Dakle, ako je matrica M singularna problem (1) će imati ∞ kao svojstvenu vrijednost (definiramo da je ∞ recipročan nuli). Također, ako problem (1) ima svojstvenu vrijednost nula, to će značiti da inverzni problem (3.14) ima svojstvenu vrijednost ∞, te da je matrica K singularna.
Jedan primjer kvadratičnog problema koji ima za svojstvenu vrijednost ∞ je

$$Q(\lambda) = \lambda^2 \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + \lambda C + \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

Dakle, matrica M je singularna, pa zaključujemo da je ∞ jedna svojstvena vrijednost. Matrica K je također singularna pa zaključujemo da je i 0 jedna svojstvena vrijednost. Ovisno o rangu matrica M i K mijenja se i kratnost svojstvene vrijednosti ∞. Ako je

$r_0 = \text{rank}(K) < n$, tada naš problem ima barem $n - r_0$ svojstvenih vrijednosti jednakih 0, a ako je $r_2 = \text{rank}(M) < n$ tada problem ima barem $n - r_2$ svojstvenih vrijednosti ∞. Neka su

$$Q_0 K P_0 = \begin{matrix} \\ r_0 \\ n-r_0 \end{matrix} \begin{matrix} \begin{matrix} r_0 & n-r_0 \end{matrix} \\ \begin{bmatrix} R_{11}^{(0)} & R_{12}^{(0)} \\ 0 & 0 \end{bmatrix} \end{matrix} = \begin{bmatrix} R^{(0)} \\ 0 \end{bmatrix}, \quad Q_2 M P_2 = \begin{matrix} \\ r_2 \\ n-r_2 \end{matrix} \begin{matrix} \begin{matrix} r_2 & n-r_2 \end{matrix} \\ \begin{bmatrix} R_{11}^{(2)} & R_{12}^{(2)} \\ 0 & 0 \end{bmatrix} \end{matrix} = \begin{bmatrix} R^{(2)} \\ 0 \end{bmatrix},$$

QR faktorizacije s pivotiranjem matrica K i M. Promatrajmo linearizaciju oblika

$$L = \begin{bmatrix} -M & 0 \\ 0 & -I \end{bmatrix}, \quad G = \begin{bmatrix} C & -I \\ K & 0 \end{bmatrix}. \tag{3.15}$$

Želimo napraviti transformaciju nad matrica G i L Tako da dobijemo matrice oblika

$$QGV = \begin{bmatrix} G_{11} & G_{12} & G_{13} \\ 0 & G_{22} & G_{23} \\ 0 & 0 & 0_{n-r_0} \end{bmatrix}, \quad QLV = \begin{bmatrix} L_{11} & L_{12} & L_{13} \\ 0 & 0_{n-r_2} & L_{23} \\ 0 & 0 & 0_{n-r_0} \end{bmatrix}. \tag{3.16}$$

Ako matrica G_{22} nije singularna iz (3.16) vidimo da naš problem ima $n - r_0$ svojstvenih vrijednosti jednakih nula, te $n - r_2$ svojstvenih vrijednosti jednakih beskonačno. Preostaje izračunati svojstvene vrijednosti generaliziranog problema $(G_{11} - \lambda L_{11})z = 0$, koje predstavljaju ostatak svojstvenih vrijednosti početnog problema (1). Dakle, na ovaj način particioniramo linearizaciju na dijelove s svojstvenim vrijednostima 0, ∞ i ostale svojstvene vrijednosti. U ovisnosti o rangu matrica M i K se za linearizaciju (3.15) definiraju matrice Q i V.

$r_0 = r_2 = n$
: U ovom slučaju nemamo ni svojstvenih vrijednosti ∞, niti svojstvenih vrijednosti nula.

$r_0 < r_2 = n$
: U ovom slučaju postoji barem $n - r_0$ svojstvenih vrijednosti jednakih nula, koje možemo particionirati tako da definiramo

$$Q = \begin{bmatrix} Q_2^* & 0 \\ 0 & Q_0^* \end{bmatrix}, \quad V = \begin{bmatrix} P_2 & 0 \\ 0 & Q_0 \end{bmatrix},$$

što daje

$$QGV = \begin{matrix} \\ n \\ r_0 \\ n-r_0 \end{matrix} \begin{matrix} \begin{matrix} n & r_0 & n-r_0 \end{matrix} \\ \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{matrix}, \quad QLV = \begin{bmatrix} -R^{(2)} & 0 & 0 \\ 0 & -I_{r_0} & 0 \\ 0 & 0 & -I_{n-r_0} \end{bmatrix},$$

gdje je $X_{11} = Q_2^* C P_2$, $\begin{bmatrix} X_{12} & X_{13} \end{bmatrix} = -Q_2^* Q_0$ i $X_{21} = R^{(0)} P_0^* P_2$. Sada ostale svojstvene vrijednosti računamo rješavajući generalizirani problem s matricama $G_{11} = \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & 0 \end{bmatrix}$ i $L_{11} = \begin{bmatrix} -R^{(2)} & 0 \\ 0 & -I_{r_0} \end{bmatrix}$.

$r_0 \leq r_2 < n$

U ovom slučaju postoji barem $n - r_0$ svojstvenih vrijednosti jednakih 0 i $n - r_2$ svojstvenih vrijednosti jednakih ∞. Definirajmo sada

$$\tilde{Q} = \begin{bmatrix} Q_2^* & 0 \\ 0 & Q_0^* \end{bmatrix}, \quad \tilde{V} = \begin{bmatrix} I_n & 0 \\ 0 & Q_0 \end{bmatrix}.$$

Time dobivamo faktorizaciju

$$\tilde{Q} G \tilde{V} = \begin{array}{c} \\ r_2 \\ n - r_2 \\ r_0 \\ n - r_0 \end{array} \begin{array}{c} \begin{array}{cccc} r_2 & n - r_2 & r_0 & n - r_0 \end{array} \\ \begin{bmatrix} X_{11} & X_{12} & X_{13} & X_{14} \\ X_{21} & X_{22} & X_{23} & X_{24} \\ X_{31} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}, \quad \tilde{Q} L \tilde{V} = \begin{bmatrix} Y_{11} & Y_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -I_{r_0} & 0 \\ 0 & 0 & 0 & -I_{n-r_0} \end{bmatrix},$$

gdje je $\begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix} = Q_2^* C$, $\begin{bmatrix} X_{13} & X_{14} \\ X_{23} & X_{24} \end{bmatrix} = -Q_2^* Q_0$, $\begin{bmatrix} X_{31} & X_{32} \end{bmatrix} = R^{(0)} P_0$ i $\begin{bmatrix} Y_{11} & Y_{12} \end{bmatrix} = -R^{(2)} P_2$. Napravimo potpunu QR faktorizaciju

$$\begin{array}{c} \\ n - r_2 \end{array} \begin{array}{c} \begin{array}{ccc} r_2 & n - r_2 & r_0 \end{array} \\ \begin{bmatrix} X_{21} & X_{22} & X_{23} \end{bmatrix} \end{array} = Q_3 \begin{array}{c} \begin{array}{cc} n - r_2 & r_0 + r_2 \end{array} \\ \begin{bmatrix} R_3 & 0 \end{bmatrix} \end{array} Z_3,$$

i konačno definirajmo

$$Q = \begin{bmatrix} I_{r_2} & 0 & 0 & 0 \\ 0 & 0 & I_{r_0} & 0 \\ 0 & Q_3^* & 0 & 0 \\ 0 & 0 & 0 & I_{n-r_0} \end{bmatrix} \tilde{Q}, \quad V = \tilde{V} \begin{array}{c} \begin{array}{cc} n + r_0 & n - r_0 \end{array} \\ \begin{bmatrix} Z_3^* & 0 \\ 0 & I_{n-r_0} \end{bmatrix} \end{array} \begin{bmatrix} 0 & I_{n-r_2} & 0 \\ I_{r_2+r_0} & 0 & 0 \\ 0 & 0 & I_{n-r_0} \end{bmatrix}.$$

Sada QGV i QLV imaju oblik (3.16).

Dakle, prije nego počnemo rješavati linearizirani problem, dobro je provjeriti rangove matrica M i K. U odnosu na njihov rang treba napraviti faktorizaciju matrica G i L kako bi izdvojili dio koji predstavlja svojstvene vrijednosti 0, odnosno ∞. Nakon toga dobivamo generalizirani problem definiran matricama G_{11} i L_{11} koji rješavamo Arnoldijevim algoritmom.

3.3 Primjena pomaka

Pretpostavimo da želimo izračunati svojstvene vrijednosti QEP-a koje su blizu neke odabrane vrijednosti σ. Ideja je početni QEP (1) transformirati, tako da novi problem ima svojstvene vrijednosti blizu vrijednosti σ. Nakon rješavanja transformiranog QEP-a napravimo transformaciju i nad izračunatim svojstvenim vrijednostima.
Pomaknuti problem bi izgledao

$$\left(\tilde{K} + (\lambda - \sigma)\,\tilde{C} + (\lambda - \sigma)^2\,\tilde{M}\right) u = 0, \tag{3.17}$$

gdje su

$$\tilde{K} = K + \sigma C + \sigma^2 M, \quad \tilde{C} = C + 2\sigma M, \quad \tilde{M} = M.$$

Bez smanjenja općenitosti, možemo pretpostaviti da je $\sigma = 0$ te zamijeniti redom K, C, M sa $\tilde{K}$, $\tilde{C}$, $\tilde{M}$ i $\lambda - \sigma$ s λ.
Nakon što izračunamo svojstvene vrijednosti pomaknutog problema λ, svojstvene vrijednosti početnog problema su $\lambda + \sigma$.

3.4 Rješenje kvadratičnog problema

Konačno, želimo vidjeti koji svojstveni vektor izabrati kao aproksimaciju za svojstveni vektor kvadratičnog problema. Radi jednostavnosti napišimo S kao

$$S = \begin{bmatrix} 0 & I \\ S_1 & S_2 \end{bmatrix}.$$

Ritzovi vektori koji pripradaju Ritzovoj vrijednosti θ su oblika

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} V_k z \\ W_k z \end{bmatrix},$$

gdje je $H_k z = \theta z$. Ako uzmemo u obzir definiciju matrice W_k (3.13) vidimo da je $x_2 = \theta x_1$. Prema tome imamo dvije opcije za odabir Ritzovog vektora, x_2/θ ili x_1.
Želimo vidjeti koji od ovih vektora je bolje izabrati. Rezidual Ritzovog para izračunatog sa Arnoldijevim algoritmom je

$$\begin{bmatrix} r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} 0 & I \\ S_1 & S_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \theta \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 - \theta x_1 \\ S_1 x_1 + S_2 x_2 - \theta x_2 \end{bmatrix}. \tag{3.18}$$

Ako odaberemo x_1 za Ritzov vektor problema (1), Ritzov vektor za linearizirani problem je tada

$$\tilde{x} = \begin{bmatrix} x_1 \\ \theta x_1 \end{bmatrix}.$$

Rezidual je

$$\begin{bmatrix} 0 & I \\ S_1 & S_2 \end{bmatrix} \begin{bmatrix} x_1 \\ \theta x_1 \end{bmatrix} - \theta \begin{bmatrix} x_1 \\ \theta x_1 \end{bmatrix} = \begin{bmatrix} 0 \\ -(S_2 - \theta I)\, r_1 + r_2 \end{bmatrix}.$$

Ako pak odaberemo x_2/θ, Ritzov vektor za linearizirani problem će biti

$$\hat{x} = \begin{bmatrix} \theta^{-1} x_2 \\ x_2 \end{bmatrix}.$$

Rezidual je u ovom slučaju

$$\begin{bmatrix} 0 & I \\ S_1 & S_2 \end{bmatrix} \begin{bmatrix} \theta^{-1} x_2 \\ x_2 \end{bmatrix} - \theta \begin{bmatrix} \theta^{-1} x_2 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \theta^{-1} S_1 r_1 + r_2 \end{bmatrix}.$$

Iz ove analize zaključujemo da za velike θ biramo vektor x_2/θ, dok za male θ biramo x_1. Primjetimo da, ako je x dovoljno blizu svojstvenog vektora, tada je $\|\tilde{x}\| \approx \|\hat{x}\|$. Također u donošenju odluke o odabiru Ritzovog vektora možemo uzeti u obzir i norme $\|S_1\|$ i $\|S_2\|$.

Poglavlje 4

SOAR metoda

Sada ćemo opisati Arnoldijev algoritam za kvadratni problem (SOAR - Second Order Arnoldi Algortihm) koji direktno koristi matrice M, C i K za rješavanje problema (1). Algoritam je detaljno razrađen u [10], te je prema tom članku rađeno ovo poglavlje. No prije nego počenom opisivati algoritam, moramo definirati Krilovljeve potprostore drugog reda.

4.1 Krilovljevi potprostori drugog reda

Definicija 4.1.1. *Neka su A i B kvadratne matrice reda n, i neka je $u \neq 0$ vektor duljine n. Niz $r_0, r_1, r_2, \ldots, r_{k-1}$, gdje je*

$$\begin{aligned} r_0 &= u, \\ r_1 &= Ar_0, \\ r_j &= Ar_{j-1} + Br_{j-2} \ \ za\ j \geq 2, \end{aligned}$$

zovemo Krilovljev niz drugog reda s obzirom na A, B i u. Prostor

$$\mathcal{G}_k(A, B; u) = \text{span}\{r_0, r_1, r_2, \ldots, r_{k-1}\}$$

se zove k-ti Krilovljev potprostor drugog reda.

Primjetimo da je ovo generalizacija strandardnog Krilovljevog potprostora $\mathcal{K}_k(A, u)$ u smislu da je

$$\mathcal{G}_k(A, 0; u) = \mathcal{K}_k(A, u).$$

Pogledajmo sada na koji način primjenjujemo ovako definiran potprostor u svrhu rješavanja QEP-a (1). U prethodnom dijelu smo razmatrali linearizaciju QEP-a (3.6). Krilovljev potprostor pridružen tom problemu bi bio

$$\mathcal{K}_k(S, v) = \text{span}\{v, Sv, S^2v, \ldots, S^{k-1}v\}$$

gdje je v početni vektor duljine $2n$ i S je kao u (3.7). Označimo s $A = -M^{-1}C$, $B = -M^{-1}K$ i $v = \begin{bmatrix} u^T & 0 \end{bmatrix}^T$. Sada vidimo da su Krilovljevi vektori drugog reda $\{r_j\}$ duljine n i standardni Krilovljevi vektori $\left\{S^j v\right\}$ duljine $2n$ vezani formulom

$$\begin{bmatrix} r_j \\ r_{j-1} \end{bmatrix} = S^j v \text{ za } j \geq 1. \tag{4.1}$$

Prema tome, generalizirani Krilovljev niz $\{r_j\}$ definira čitav standardni Krilovljev potprostor u odnosu na S i v. Jednadžba (4.1) sugerira da će nam potprostor $\mathcal{G}_j(A, B; u)$ dati dovoljno informacija da radimo direktno sa QEP-om, a ne da korisitimo potprostor $\mathcal{K}_j(S, v)$ za linearizirani problem. Primjetimo da su vektori koji čine potprostor $\mathcal{G}_j(A, B; u)$ duljine n, dok su oni koji čine $\mathcal{K}_j(S, v)$ duljine $2n$. Sada, kao kod Arnoldijevog algoritma, želimo naći ortogonalnu bazu $\{q_i\}$ za $\mathcal{G}_j(A, B; u)$, tj.

$$\text{span}\{q_1, q_2, \ldots, q_j\} = \mathcal{G}_j(A, B; u) \text{ za } j \geq 1.$$

Sljedeći algoritam pokazuje kako da iskoristimo vektor $\{r_j\}$ da generiramo ortonormiranu bazu $\{q_1, q_2, \ldots, q_j\}$. Kasnije ćemo vidjeti da je ovo algoritam Arnoldijevog tipa. Ovaj algoritam zovemo SOAR (Second order Arnoldi).

Algoritam 6: $[P, Q, T]$ =SOAR(A, B, u, m)

```
Q(:, 1) = u/||u||_2;
P(:, 1) = 0;
for j = 1, 2, ..., k do
    r = AQ(:, j) + BP(:, j);
    s = Q(:, j);
    for i = 1, 2, ..., j do
        T(i, j) = Q(:, i)^T r;
        r = r − Q(:, i)T(i, j);
        s = s − P(:, i)T(i, j);
    end
    T(j + 1, j) = ||r||_2;
    if T(j + 1, j) = 0 then
        STOP;
    end
    Q(:, j + 1) = r/T(j + 1, j);
    P(:, j + 1) = s/T(j + 1, j);
end
```

Vrlo je važno da matrice A i B sudjeluju jedino u liniji (4) algoritma (6), i to preko množenja s vektorom. Prema tome, to je pogodno za velike i rijetke matrice, jer se ta struktura može iskoristiti u svrhu efikasnijeg izvođenja operacije množenja s vektorom.
`For` petlja algoritma u linijama 6-10 predstavlja ortogonalizaciju u odnosu na vektore $\{q_i\}$. Vektori $\{p_i\}$ su pomoćni vektori. Kasnije ćemo komentirati kako se algoritam može poboljšati, u smislu da ne moramo spremati vektore $\{p_i\}$. To će nam smanjii potrebe za memorijom za barem pola.
Algoritam (6) staje kada je norma vektora r, koja se računa u liniji 12, dovoljno mala. U tom slučaju dolazi do deflacije ili prekida. Želimo naći odnos između objekata koji se računaju u algoritmu (6) sličnu onoj u (2.7).
Algoritam generira matricu Q dimenzije $n \times k$, pa ćemo je prema tome označavati s Q_k, a njene stupce $Q(:, i)$ sa q_i. Slično, matrica P je dimenzije $n \times k$ i njene stupce $P(:, i)$ označavamo sa p_i. Matrica T je $k \times k$ gornje Hessenbergova matrica, uvodimo oznaku $T_k = (t_{ij})_{i,j}$. Algoritam nam daje sljedeći odnos

$$AQ_k + BP_k = Q_k T_k + q_{k+1} e_k^T t_{k+1,k}, \tag{4.2}$$

$$Q_k = P_k T_k + p_{k+1} e_k^T t_{k+1,k}. \tag{4.3}$$

Također, skup vektora $\{q_1, \ldots, q_{k+1}\}$ je ortonormiran. Neka je $\hat{T}_k$ $(k+1) \times k$ gornje Hessenbergova matrica oblika $\hat{T}_k = \begin{bmatrix} T_k \\ e_k^T t_{k+1,k} \end{bmatrix}$. Tada relacije (4.2) i (4.3) možemo zapisati kao

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix} \begin{bmatrix} Q_k \\ P_k \end{bmatrix} = \begin{bmatrix} Q_{k+1} \\ P_{k+1} \end{bmatrix} \hat{T}_k. \tag{4.4}$$

Upravo u ovoj relaciji vidimo sličnost sa Arnoldijevim algoritmom. Zaista, ako Arnoldijev algoritam startamo sa matricom S, definiranom kao u (3.7), dobivamo matricu V_k reda $2n \times k$ i gornje Hessenbergovu matricu H_k reda $k \times k$ tako da vrijedi

$$S V_k = V_k H_k + v_{k+1} e_k^T h_{k+1,k},$$

ili još bolje

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix} V_k = V_{k+1} \hat{H}_k. \tag{4.5}$$

Sada, uspoređujući jednakosti (4.4) i (4.5) vidimo da je osnovna razlika između Arnoldijevog algorirma i SOAR-a u tome što kod SOAR-a nenegativne elemente matrice $\hat{T}_k$ biramo da osiguramo ortogonalnost vektora $\{q_j\}$ koji su dimenzije n, dok kod Arnoldija, nenegativne elemente matrice $\hat{H}_k$ biramo da osiguramo ortogonalnost vektora $\{v_i\}$ dimenzije $2n$.
Sljedeći zadatak je pokazati da opisani algoritam uistinu generira ortonormiranu bazu za Krilovljev potprostor $\mathcal{G}_j(A, B; u)$. Prvo nam je potrebna sljedeća lema:

Lema 4.1.2. *Neka je A proizvoljna $n \times n$ matrica. Neka je $V_{k+1} = \begin{bmatrix} V_k & v_{k+1} \end{bmatrix}$ matrica reda $n \times (k+1)$ koja zadovoljava*

$$AV_k = V_{k+1}\hat{H}_k,$$

gdje je $\hat{H}_k$ $(k+1)\times k$ gornje Hessenbergova matrica. Tada postoji gornje trokutasta matrica R_k takva da vrijedi

$$V_k R_k = [v_1 \; Av_1 \; \ldots \; A^{k-1}v_1]. \tag{4.6}$$

Nadalje, ako je prvih $k-1$ subdijagonalnih elemenata od $\hat{H}_k$ različito od nule, R_k je regularna i

$$\text{span}\{V_k\} = \mathcal{K}_k(A, v_1). \tag{4.7}$$

Dokaz. Prvo dokazujemo (4.6) indukcijom po k. Za $k = 1$, (4.6) vrijedi uz $R_1 = 1$. Pretpostavimo da tvrdnja vrijedi za $k-1$. Tada za k imamo

$$\begin{aligned}
\begin{bmatrix} v_1 \; Av_1 \; \ldots \; A^{k-1}v_1 \end{bmatrix} &= \begin{bmatrix} v_1 \; A \begin{bmatrix} v_1 \; Av_1 \; \ldots \; A^{k-2}v_1 \end{bmatrix} \end{bmatrix} \\
&= [v_1 \; AV_{k-1}R_{k-1}] \\
&= \begin{bmatrix} V_k e_1 \; V_k \hat{H}_{k-1} R_{k-1} \end{bmatrix} \\
&= V_k \begin{bmatrix} e_1 \; \hat{H}_{k-1} R_{k-1} \end{bmatrix} = V_k R_k.
\end{aligned}$$

Činjenica da je R_k gornje trokutasta izlazi iz same definicije $R_k = [e_1 \; \hat{H}_{k-1}R_{k-1}]$. Dijagonalni elementi od R_k su jedinica i produkti prvih $k-1$ subdijagonalnih elemenata od $\hat{H}_k$. Prema tome, ako pretpostavimo da je prvih $k-1$ subdijagonalnih elementa od $\hat{H}_k$ različito od nule, odmah slijedi da je R_k regularna.
Sada (4.7) slijedi iz (4.6) i činjenice da je R_k regularna.
Time je lema dokazana. □

Uz pomoć ove leme dokazujemo da algoritam (6) generira ortonormiranu bazu za $\mathcal{G}_j(A, B; u)$.

Teorem 4.1.3. *Ako je $t_{j+1,j} \neq 0$ za $j \geq 0$ u algoritmu (6), tada niz vektora $\{q_1, q_2, \ldots, q_j\}$ čini otronormiranu bazu za Krilovljev potprostor drugog reda $\mathcal{G}_j(A, B; u)$, tj.*

$$\text{span}\{Q_j\} = \mathcal{G}_j(A, B; u) \;\; za \;\; j \geq 1. \tag{4.8}$$

Dokaz.

$$\begin{aligned}
\mathcal{G}_j(A,B;u) &= \text{span}\{r_0, r_1, \ldots, r_{j-1}\} \\
&= \text{span}\left\{[I\ 0]\begin{bmatrix} r_0 & r_1 & \ldots & r_{j-1} \\ 0 & r_0 & \ldots & r_{j-2} \end{bmatrix}\right\} \\
&= \text{span}\{[I\ 0][v_1\ Hv_1\ \ldots\ H^{j-1}v_1]\} \text{ prema (4.1)} \\
&= \text{span}\left\{[I\ 0]\begin{bmatrix} Q_j \\ P_j \end{bmatrix} R_j\right\} \text{ prema (4.4) i lemi (4.1.2)} \\
&= \text{span}\left\{[I\ 0]\begin{bmatrix} Q_j \\ P_j \end{bmatrix}\right\} \text{ uz pretpostavku da je } R_j \text{ regularna} \\
&= \text{span}\{Q_j\}
\end{aligned}$$

Ortogonalnost vektora slijedi iz ortogonalizacije u for petlji (linije (6-10)) i normalizacije u koraku 15 u algoritmu (6). □

Postoje dva razloga zbog kojeg dolazi do prekidanja algoritma (6), to jest do toga da je $\|r\|_2 = 0$. U prvom slučaju imamo da je niz vektora $\{r_i\}$ linearno zavisan, no niz vektora $\left\{\begin{bmatrix} r_i^T & r_{i-1}^T \end{bmatrix}^T\right\}$ je linearno nezavisan. U ovom slučaju imamo da je $\mathcal{G}_{j+1}(A,B;u) = \mathcal{G}_j(A,B;u)$, no $\mathcal{K}_{j+1}(S,v) \neq \mathcal{K}_j(S,v)$ i Arnoldijev algoritam ne staje. Ovu situaciju nazivamo *deflacija*. Tada modificiramo podatke u algoritmu i proces može nastaviti.
Drugi slučaj je da su oba niza vektora $\{r_i\}$ i $\left\{\begin{bmatrix} r_i^T & r_{i-1}^T \end{bmatrix}^T\right\}$ linearno zavisni. U tom slučaju staju i Arnoldijev i SOAR algoritam. Ovu pojavu nazivamo *prekid*. Arnoldijev algoritam staje ukoliko je norma, koja se računa u liniji 7, jednaka nuli. U tom slučaju je niz vektora $\{H^i v\} = \left\{\begin{bmatrix} r_i^T & r_{i-1}^T \end{bmatrix}^T\right\}$ linearno zavisan. Dakle, došlo je do prekida. Pokazat ćemo da postoji veza između prekida u Arnoldijevom i SOAR algoritmu. Razlika je u tome da je kod Arnoldijevog algoritma prekid poželjan, jer to znači da smo našli invarijantan potprostor za mali broj iteracija, i svojstvene vrijednosti matrice H dobro aproksimiraju svojstvene vrijednosti matrice A. No kod SOAR metode to nije poželjno jer nismo dobili ortonormiranu bazu, a proces se ne može nastaviti.

Deflacija

Sljedeći algoritam pokazuje na koji način se podaci ažuriraju ukoliko dođe do deflacije.

Algoritam 7: $[P, Q, T]$ =SOAR(A, B, u, m)

```
1  Q(:, 1) = u/||u||_2;
2  P(:, 1) = 0;
3  for j = 1, 2, ..., k do
4      r = AQ(:, j) + BP(:, j);
5      s = Q(:, j);
6      for i = 1, 2, ..., j do
7          T(i, j) = Q(:, i)^T r;
8          r = r − Q(:, i)T(i, j);
9          s = s − P(:, i)T(i, j);
10     end
11     T(j + 1, j) = ||r||_2;
12     if T(j + 1, j) = 0 then
13         if s ∈ span{p_i | i : q_i = 0, 1 ≤ i ≤ j} then
14             prekid;
15         end
16         else
17             deflacija;
18             T(j + 1, j) = 1;
19             Q(:, j + 1) = 0;
20             P(:, j + 1) = s;
21         end
22     end
23     else
24         Q(:, j + 1) = r/T(j + 1, j);
25         P(:, j + 1) = s/T(j + 1, j);
26     end
27 end
```

Treba napomenuti da uz ovu modifikaciju, algoritam i dalje daje ortonormiranu bazu za Krilovljev potprostr $\mathcal{G}_k(A, B; u)$, s tim da moramo izbaciti one vektore q_i koji su u koraku 19 stavljeni na nulu.

Prekid

Kao što smo rekli, postoji veza između prekida kod Arnoldijevog i SOAR algoritma. O tome govori sljedeći teorem:

Teorem 4.1.4. *Algoritam (7) sa ulaznim matricama A i B i ulaznim vektorom u staje u koraku j ako i samo ako Arnoldijev algoritam sa ulaznom matricom S i početnim vektorom v staje u istom koraku j.*

Da bi dokazali ovaj teorem, potrebna nam je sljedeća lema:

Lema 4.1.5. *Neka za niz linearno nezavisnih vektora* $\{v_1, v_2, \ldots, v_m\}$*, sa particijom* $v_i = \begin{bmatrix} q_i^T & p_i^T \end{bmatrix}^T$*, postoji podniz* $\{q_{i_1}, q_{i_2}, \ldots, q_{i_k}\}$ *linearno nezavisnih vektora i* $q_{i_{k+1}} = q_{i_{k+2}} = \cdots = q_{i_m} = 0$*. Tada je* $v = \left\{\begin{bmatrix} 0 & p^T \end{bmatrix}^T\right\} \in \text{span}\{v_1, v_2, \ldots, v_m\}$ *ako i samo ako je* $p \in \text{span}\{p_{i_{k+1}}, p_{i_{k+2}}, \ldots, p_{i_m}\}$.

Dokaz. Ako je $v \in \text{span}\{v_1, v_2, \ldots, v_m\}$, onda postoje skalari α_i, $i = 1, \ldots, m$ takvi da je $v = \sum_{i=1}^{m} \alpha_i v_i$. Pretpostavimo da je $v = \left\{\begin{bmatrix} 0 & p^T \end{bmatrix}^T\right\} \in \text{span}\{v_1, v_2, \ldots, v_m\}$ i da postoji podniz niza $\{q_i\}$ takav da je $q_{i_{k+1}} = q_{i_{k+2}} = \cdots = q_{i_m} = 0$. Tada je $0 = \sum_{j=1}^{m} \alpha_j q_j = \sum_{j=1}^{k} \alpha_{i_j} q_{i_j}$. Sada linearna nezavisnost vektora $q_{i_1}, q_{i_2}, \ldots, q_{i_k}$ povlači da je $\alpha_{i_j} = 0$ za $j = 1, \ldots, k$. Prema tome $v = \sum_{j=k+1}^{m} \alpha_{i_j} v_{i_j}$ što znači da je $p = \sum_{j=k+1}^{m} \alpha_{i_j} p_{i_j}$ što je dalje ekvivalentno s tim da je $p \in \text{span}\{p_{i_{k+1}}, p_{i_{k+2}}, \ldots, p_{i_m}\}$. □

Sada možemo dokazati teorem (4.1.4)

Dokaz. Pretpostavimo prvo da je Arnoldijev algoritam stao u koraku j. To znači da je

$$dim(\mathcal{K}_m(S, v)) = j \text{ i } S^m v \in \mathcal{K}_j(S, v) \text{ za } m \geq j. \tag{4.9}$$

Iz (4.4) i leme (4.1.2) slijedi

$$\text{span}\left\{\begin{bmatrix} Q_j \\ P_j \end{bmatrix}\right\} = \mathcal{K}_j(S, v).$$

Kako je $dim\left(\mathcal{K}_j(S, v)\right) = j$, $\begin{bmatrix} Q_j \\ P_j \end{bmatrix}$ je punog stupčanog ranga. Iz leme (4.1.2) i jednadžbe (4.9) slijedi

$$\begin{bmatrix} r \\ s \end{bmatrix} \in \text{span}\left\{\begin{bmatrix} Q_j \\ P_j \end{bmatrix}\right\}. \tag{4.10}$$

Pokazujemo da je $r = 0$ (u liniji 11 algortima (7)). Pretpostavimo suprotno, tj. da $r \neq 0$. Kako je, po konstrukciji, $r^T q_i = 0$, za $i = 1, \ldots, j$, slijedi da

$$r \notin \text{span}\left\{q_1, q_2, \ldots, q_j\right\},$$

što povlači da

$$\begin{bmatrix} r \\ s \end{bmatrix} \notin \text{span}\left\{\begin{bmatrix} Q_j \\ P_j \end{bmatrix}\right\}.$$

Ovo je u kontradikciji s (4.10), prema tome $r = 0$. Prema tome, algoritam (7) prelazi na liniju 13. Iz (4.10) i leme (4.1.5) sada slijedi da je

$$s \in \text{span}\{p_i | i : q_i = 0, \quad 1 \leq i \leq j\},$$

što znači da dolazi do prekida i u algoritmu (7) (linija 14).
Obratno, pretpostavimo da je u nekom koraku j algoritma (7) došlo do prekida. Tada vrijedi

$$\begin{bmatrix} r \\ s \end{bmatrix} = \begin{bmatrix} 0 \\ s \end{bmatrix} \in \text{span}\left\{\begin{bmatrix} Q_j \\ P_j \end{bmatrix}\right\}. \tag{4.11}$$

Uočimo da (4.4) i dalje vrijedi uz odabir $t_{j+1,j} = 1$. Prema tome, iz leme (4.1.2) slijedi da je

$$\text{span}\left\{\begin{bmatrix} Q_j \\ P_j \end{bmatrix}\right\} = \mathcal{K}_j(S, v) \quad \text{i} \quad \text{span}\left\{\begin{bmatrix} Q_{j+1} \\ P_{j+1} \end{bmatrix}\right\} = \mathcal{K}_{j+1}(S, v).$$

Po konstrukciji matrice $\begin{bmatrix} Q_j \\ P_j \end{bmatrix}$ u algoritmu (7) vidimo da je njen rang jednak j. Ako to uzmemo u obzir, zajedno sa (4.11), imamo da je $\dim\left(\mathcal{K}_j(S, v)\right) = j$ i $\mathcal{K}_j(S, v) = \mathcal{K}_{j+1}(S, v)$. Ova dva uvjeta osiguravaju da u Arnoldijevom algoritmu dolazi do prekida u istom koraku j. □

Kada dođe do prekida u Arnoldijevom algoritmu, to znači da Krilovljev potprostor $\mathcal{K}_j(S, v)$ čini invarijantan potprostor za H i niz vektora $\left\{v_1, v_2, \ldots, v_j\right\}$ je ortonormirana baza za taj potprostor. Ovakav prekid kod Arnoldijevog algoritma je poželjan. Što se tiče SOAR-a, iz (4.4) znamo da stupčani vektori $2n \times j$ matrice $\begin{bmatrix} Q_j \\ P_j \end{bmatrix}$ također razapinju invarijantan potprostor za T, ali oni nisu ortonormirana baza. Prema tome, prekid kod SOAR-a nije poželjan.

Ušteda memorije

Ako uzmemo u obzir da je $p_1 = 0$, tada iz formule (4.3) dobivamo

$$Q_k = P_{k+1}\hat{T}_k = P_{k+1}(:, 2 : k+1)\hat{T}_k(2 : k+1, 1 : k).$$

Sada (4.2) možemo zapisati kao

$$AQ_k + BQ_kF_k = Q_kT_k + q_{k+1}e_k^T t_{k+1,k}, \tag{4.12}$$

gdje je F_k $k \times k$ gornje trokutasta matrica oblika

$$F_k = \begin{bmatrix} 0 & \hat{T}_k(2 : k, 1 : k-1)^{-1} \\ 0 & 0 \end{bmatrix}.$$

Jednadžba (4.12) nam daje metodu računanja vektora q_{j+1} preko vektora $q_1, q_2, \ldots, q_j$. To nam daje sljedeći algoritam, koji koristi upola manje memorije jer ne treba spremati matricu P.

Algoritam 8: $[P, Q, T]$ =SOAR(A, B, u, m)

```
1  Q(:, 1) = u/||u||_2;
2  f = 0;
3  for j = 1, 2, ..., m do
4      r = AQ(:, j) + Bf;
5      for i = 1, 2, ..., j do
6          T(i, j) = Q(:, i)^T r;
7          r = r − Q(:, i)T(i, j);
8      end
9      T(j + 1, j) = ||r||_2;
10     if T(j + 1, j) = 0 then
11         if f u linearnoj ljusci od prethodno izračunatih f then
12             prekid;
13         end
14         else
15             deflacija;
16             T(j + 1, j) = 1;
17             Q(:, j + 1) = 0;
18             f = Q(:, j)T̂(2 : j + 1, 1 : j)^{-1} e_j;
19         end
20     end
21     else
22         Q(:, j + 1) = r/T(j + 1, j);
23         f = Q(:, j)T̂(2 : j + 1, 1 : j)^{-1} e_j;
24     end
25 end
```

Da bi provjerili liniju 11, koristimo Gramm-Schdmitov postupak. Prije izračunate f možemo spremati u matricu Q na onim mjestima gdje je $q_j = 0$.

Rješenje QEP-a

Želimo naći aproksimativni svojstveni par (θ, z), gdje je $\theta \in \mathbb{C}$ i $z \in \mathcal{G}_m(A, B; u)$ primjenjujući sljedeći uvjet ortogonalnosti reziduala

$$(\theta^2 M + \theta C + K)z \perp \mathcal{G}_m(A, B; u)$$

što je ekvivalentno s

$$v^T(\theta^2 M + \theta C + K)z = 0 \text{ za sve } v \in \mathcal{G}_m(A, B; u). \tag{4.13}$$

Kako je $z \in \mathcal{G}_m(A, B; u)$, možemo ga zapisati kao

$$z = Q_k g, \tag{4.14}$$

gdje je $n \times m$ matrica Q_k ortonormirana baza za $\mathcal{G}_m(A, B; u)$ generirana algoritmom (8) i g je vektor duljine k, $k \leq m$. Iz (4.13) i (4.14) slijedi da θ i g moraju zadovoljavati reducirani QEP:

$$(\theta^2 M_k + \theta C_k + K_k)g = 0, \tag{4.15}$$

gdje je

$$M_k = Q_k^T M Q_k, \quad C_k = Q_k^T C Q_k, \quad K_k = Q_k^T K Q_k. \tag{4.16}$$

Svojstveni par (θ, g) problema (4.15) definira Ritzov par (θ, z). Ritzovi parovi aproksimiraju svojstvene parove QEP-a (1). Važno je napomenuti da transformirane matrice M_k, C_k, K_k čuvaju strukturu polaznih matrica M, C, K. Na primjer, Ako je M simetrična pozitivno definitna, onda je i M_k. Prema tome, esencijalna svojstva traženih svojstvenih vrijednosti ostaju sačuvana.
Time možemo definirati konačni algoritam za direktno rješavanje kvadratičnog svojstvenog problema

Algoritam 9: SOAR metoda za direktno rješavanje QEP-a

1. Pokrenuti SOAR algoritam (7) sa ulaznim podacima $A = -M^{-1}D$, $B = -M^{-1}K$ i početnim vektorom u kako bi dobili $n \times k$ ortogonalnu matricu Q_k čiji stupci razapinju ortonormiranu bazu za $\mathcal{G}_m(A, B; u)$.;
2. Izračunati matrice M_k, D_k i K_k kako je definirano u (4.16).;
3. Riješiti reducirani QEP (4.15). Za dobivene parove (θ, g) izračunati Ritzove parove (θ, z), gdje je $z = \frac{Q_k g}{\|Q_k g\|_2}$.;
4. Provjeriti točnost Ritzovih parova (θ, g) računajući relativnu normu reziduala:

$$\frac{\|\left(\theta^2 M + \theta D + K\right) z\|_2}{|\theta|^2\|M\|_1 + |\theta|\|D\|_1 + \|K\|_1} \tag{4.17}$$

.

Koja je interpretacija od (4.17)?. Pretpostavimo da smo našli svojstveni par (θ, z) koji aproksimira svojstveni par za problem (1). Tada je rezidual r jednak

$$\left(\theta^2 M + \theta C + K\right) z = r.$$

Zanima nas, da li je možda par (θ, z) egzaktni svojstveni par za neki drugi problem, koji je jako blizu našem početnom problemu, tj. ako malo promijenimo matrice M, C i K, možemo li reći da je za takve matrice ovaj svojstveni par egzaktan. Pretpostavimo da vrijedi

$$\left(\theta^2(M + \Delta M) + \theta(C + \Delta C) + (K + \Delta K)\right) z = 0. \tag{4.18}$$

Tada mora vrijediti da je $\left(\lambda^2 \Delta M + \lambda \Delta C + \Delta K\right) z = r$, odnosno $\lambda^2 \Delta M + \lambda \Delta C + \Delta K = -r z^*$. Kako vektor z uzimamo da je norme 1, vrijedi da je $\|\lambda^2 \Delta M + \lambda \Delta C + \Delta K\| = \|r\|$. Dakle, jednadžba relativna norma reziduala (4.17) nam govori da svojstveni par (θ, z) možda nije egzaktan svojstveni par za problem (1), ali je egzaktan za pomaknuti problem (4.18).

Poglavlje 5

GSOAR. Implicitno restartani GSOAR

Ovi algoritmi su detaljno razrađeni u [11], te je prema tome rađeno ovo poglavlje. Kao i kod Arnoldijevog algoritma, želimo razviti metodu koja će nam računati određen broj svojstvenih vrijednosti, koje imaju zajedničko svojstvo. Dakle, želimo algoritam restartati, te ponoviti iteracije sa početnim vektorom koji ima bolja svojstva nego onaj prije. Također, želimo da se to odvija implicitno, tj. da ne moramo iteracije obavljati od početka.
Implicitno restartanje nije moguće provesti na SOAR metodi, jer je početni vektor $p_1 = 0$ što se gubi prilikom restartanja. Prema tome, želimo razraditi metodu u kojoj početni vektor p_1 može biti različit od nule. Ovakvu metodu zovemo generalizirani SOAR (GSOAR). Ideja implicitno restartanog GSOAR-a je slična kao ona kod implicitno restartanog Arnoldija. Nađemo aproksimacije svojstvenih vrijednosti početnog problema, te uz određene pomake u Hessenbergovoj matrici, želimo prigušiti doprinos onih svojstvenih vrijednosti koje nam nisu od interesa. Međutim, kod GSOAR metode, radimo direktno sa kvadratnim problemom, što znači da nakon k koraka GSOAR algoritma, imamo aproksimaciju za $2k$ svojstvenih vrijednosti. To znači da moramo izabrati p Ritzovih vrijednosti za *shift* između njih $2k - m$. Objasnit ćemo na koji način se biraju ovi *shiftovi*. Također uvest ćemo novi Krilovljev potprostor drugog reda, koji uključuje situaciju da su oba početna vektora različita od nul-vektora.

5.1 Opis GSOAR metode

Sada ćemo proširiti definiciju (4.1.1) Krilovljevog potprostora drugog reda time što ćemo omogućiti da imamo dva početna vektora različita od nule umjesto jednog.

Definicija 5.1.1. *Neka su A i B kvadratne matrice reda n i neka su $u_1 \neq 0$ i u_2 dva vektora duljine n. Niz $r_0, r_1, r_2, \ldots, r_{k-1}$, gdje je*

$$r_0 = u_1, \quad r_1 = Au_1 + Bu_2, \quad r_j = Ar_{j-1} + Br_{j-2} \quad za\ j \geq 2$$

zovemo generalizirani Krilovljev niz drugog reda s obzirom na A, B, u_1 i u_2. Prostor

$$\mathcal{G}_k(A, B; u_1, u_2) = \text{span}\{r_0, r_1, r_2, \ldots, r_{k-1}\}$$

se zove k-ti generalizirani Krilovljev potprostor drugog reda.

Primijetimo da je generalizirani Krilovljev potprostor drugog reda $\mathcal{G}_k(A, B; u_1, u_2)$ poopćenje Krilovljevog potprostora $\mathcal{G}_k(A, B; u_1)$ u smislu da je vektor u_2 nul-vektor, tj.

$$\mathcal{G}_k(A, B; u_1, 0) = \mathcal{G}_k(A, B; u_1).$$

Sada, kao kod Krilovljevog potprostora drugog reda, ako označimo $v = \begin{bmatrix} u_1^T & u_2^T \end{bmatrix}^T$, $A = -M^{-1}D$ i $B = -M^{-1}K$, dobijemo odnos između generaliziranog Krilovljevog potprosotra drugog reda i standardnog Krilovljevog potprostora

$$\begin{bmatrix} r_j \\ r_{j-1} \end{bmatrix} = S^j v, \quad j \geq 1, \tag{5.1}$$

gdje je S kao i prije $S = \begin{bmatrix} A & B \\ I & 0 \end{bmatrix}$. Prema tome, opet zaključujemo, da ćemo dobiti dovoljno informacija ako budemo radili direktno sa QEP-om umjesto sa linearizirani problemom. GSOAR metoda je ista kao SOAR metoda, uz iznimku da je u koraku (2) algoritma (6) vektor p_1 uvijek stavljen na nulu, što nije slučaj kod GSOAR. Algoritam je sljedeći

Algoritam 10: $[P, Q, T]$ =GSOAR(A, B, u, m)

```
1  Q(:,1) = u1/||u1||2;
2  P(:,1) = u2/||u2||2;
3  for j = 1,2,...,m do
4  |   r = AQ(:,j) + BP(:,j);
5  |   s = Q(:,j);
6  |   for i = 1,2,...,j do
7  |   |   T(i,j) = Q(:,i)^T r;
8  |   |   r = r − Q(:,i)T(i,j);
9  |   |   s = s − P(:,i)T(i,j);
10 |   end
11 |   T(j+1,j) = ||r||2;
12 |   if T(j+1,j) = 0 then
13 |   |   STOP;
14 |   end
15 |   Q(:,j+1) = r/T(j+1,j);
16 |   P(:,j+1) = s/T(j+1,j);
17 end
```

Vrijede isti odnosi (4.2) i (4.3) kao kod SOAR metode.
Također, konačno rješenje kvadratičnog problema se svodi na algoritam (9). Opet uzimamo u obzir, da ako je došlo do deflacije, Q_m se sastoji od vektora koji su različiti od nule, pa prema tome imamo da je dimenzija problema (4.15) manja od m.
Pogledajmo još `For` petlju u linijama (6-10) u algoritmu (10). Taj dio predstavlja Gram – Schmidtovu ortogonalizaciju novog vektora r u odnosu na prije izračunate vektore $Q(:, i)$, $i = 1, \ldots, j$. Prilikom testiranja primjera 3 (6.poglavlje, Numerički eksperimenti) ortogonalizacija se nije provela, točnije norma $\|I_k - Q^*Q\| \approx 10^{-2}$ nije bila dovoljno mala. Zbog toga smo ortogonalizaciju proveli u jednom koraku, to jest vektore $Q(:, i)$ smo poredali u stupce matrice te ortogonalizaciju proveli na matrici, čime se izbjeglo konstantno oduzimanje u linijama (8) i (9) koje može dovesti do katastrofalnog kraćenja. Također smo uveli reortogonalizaciju (linije 10-12 u algoritmu (11)). Novi algoritam bi izgledao ovako

Algoritam 11: $[P, Q, T]$ =GSOAR(A, B, u, m)

1 $Q(:, 1) = u_1/\|u_1\|_2$;
2 $P(:, 1) = u_2/\|u_2\|_2$;
3 **for** $j = 1, 2, \ldots, m$ **do**
4 $\quad r = AQ(:, j) + BP(:, j)$;
5 $\quad s = Q(:, j)$;
6 $\quad T(1 : j, j) = Q(:, 1 : j)^T r$;
7 $\quad r = r - Q(:, 1 : j)T(1 : j, j)$;
8 $\quad s = s - P(:, 1 : j)T(1 : j, j)$;
9 $\quad$ *%Reortogonalizacija*
10 $\quad c = Q(:, 1 : j)^T * r$;
11 $\quad r = r - Q(:, 1 : j)c$;
12 $\quad s = s - P(:, 1 : j)c$;
13 $\quad T(1 : j, j) = T(1 : j, j) + c$;
14 $\quad T(j + 1, j) = \|r\|_2$;
15 $\quad$ **if** $T(j + 1, j) = 0$ **then**
16 $\quad\quad$ STOP;
17 $\quad$ **end**
18 $\quad Q(:, j + 1) = r/T(j + 1, j)$;
19 $\quad P(:, j + 1) = s/T(j + 1, j)$;
20 **end**

Nakon ove promjene, greška ortogonalizacije $\|I_k - Q^*Q\|$ je bila reda veličine 10^{-15}.

5.2 Implicitno restartanje

Pretpostavimo da smo napravili k koraka GSOAR algoritma, te da nije došlo do deflacije. Broj m će nam označavati broj svojstvenih vrijednosti koje želimo naći, dok je $p = k - m$, tj. broj dodatnih iteracija GSOAR algoritma. Sljedeći teorem opisuje proces implicitnog restartanja.

Teorem 5.2.1. *Neka je zadano p shiftova $\mu_1, \mu_2, \ldots, \mu_p$. Pretpostavimo da smo obavili p QR transformacija nad pomaknutom matricom $T_k - \mu_i$. Neka je $\psi(T_k) = V_k R_k$, $\psi(\mu) = \prod_{j=1}^{p}(\mu - \mu_j)$ QR tranformacija. Definirajmo $Q_k^+ = Q_k V_k$ i $T_k^+ = V_k^T T_k V_k$. Tada dobijemo novu dekompoziciju oblika (4.4)*

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix} \begin{bmatrix} Q_m^+ \\ P_m^+ \end{bmatrix} = \begin{bmatrix} Q_m^+ \\ P_m^+ \end{bmatrix} T_m^+ + \tilde{t}_{m+1,m} \begin{bmatrix} q_{m+1}^+ \\ p_{m+1}^+ \end{bmatrix} e_m^T, \tag{5.2}$$

koja za početni vektor ima $\begin{bmatrix} q_1^+ \\ p_1^+ \end{bmatrix}$. *Odnosi među matricama su* $Q_m^+ = Q_k V_k(:, 1:m)$, $P_m^+ = P_k V_k(:, 1:m)$, $T_m^+ = T_k^+(1:m, 1:m)$ *i*

$$\begin{aligned} \begin{bmatrix} q_{m+1}^+ \\ p_{m+1}^+ \end{bmatrix} &= \frac{1}{\tilde{t}_{m+1,m}^+} f_m^+, \\ f_m^+ &= \tilde{t}_{m+1,m}^+ \begin{bmatrix} q_{m+1} \\ p_{m+1} \end{bmatrix} + t_{k+1,k} V_k(k, m) \begin{bmatrix} q_{k+1} \\ p_{k+1} \end{bmatrix}, \\ \tilde{t}_{m+1,m}^+ &= \| t_{m+1,m}^+ q_{m+1}^+ + t_{k+1,k} V_k(k, m) q_{k+1} \|. \end{aligned}$$

Dokaz. Pretpostavimo da smo napravili $m + p = k$ koraka GSOAR algoritma, te da nije došlo do deflacije. Neka smo dobili dekompoziciju

$$S \begin{bmatrix} Q_k \\ P_k \end{bmatrix} = \begin{bmatrix} Q_k \\ P_k \end{bmatrix} T_k + t_{k+1,k} \begin{bmatrix} q_{k+1} \\ p_{k+1} \end{bmatrix} e_{k+1}^T. \tag{5.3}$$

Neka je μ_1 jedan od odabranih p shiftova. Napravimo QR dekompoziciju $T_k - \mu_1 I = V_1 R$. Iz (5.3) vidimo da vrijedi

$$(S - \mu_i I) \begin{bmatrix} Q_k \\ P_k \end{bmatrix} = \begin{bmatrix} Q_k \\ P_k \end{bmatrix} (T_k - \mu_i I) + t_{k+1,k} \begin{bmatrix} q_{k+1} \\ p_{k+1} \end{bmatrix} e_{k+1}^T. \tag{5.4}$$

Pomnožimo li tu relaciju sa V_1 dobivamo

$$(S - \mu_i I) \begin{bmatrix} Q_k \\ P_k \end{bmatrix} V_1 - \begin{bmatrix} Q_k \\ P_k \end{bmatrix} V_1 R V_1 = t_{k+1,k} \begin{bmatrix} q_{k+1} \\ p_{k+1} \end{bmatrix} e_{k+1}^T V_1, \tag{5.5}$$

što daje

$$S\begin{bmatrix}Q_k\\P_k\end{bmatrix}V_1 - \begin{bmatrix}Q_k\\P_k\end{bmatrix}V_1\,(RV_1+\mu_1 I) = t_{k+1,k}\begin{bmatrix}q_{k+1}\\p_{k+1}\end{bmatrix}e_{k+1}^T V_1. \tag{5.6}$$

Označimo $\begin{bmatrix}Q_k^+\\P_k^+\end{bmatrix} = \begin{bmatrix}Q_k\\P_k\end{bmatrix}V_1$ i $T_k^+ = RV_1 + \mu_1 I$. Kako je $V_1 R$ QR dekompozicija Hessenbergove matrice, matrica V_1 je Hessenbergova, pa je i T_k^+ Hessenbergova kao umnožak Hessenbergove i gornje trokutaste matrice (uz oduzimanje dijagonalne matrice).
Ovaj postupak se proširuje na p shiftova. Označimo sa $V_k = V_1 V_2 \dots V_p$. Ako pomnožimo (5.3) s V_k zdesna dobivamo

$$S\begin{bmatrix}Q_k^+\\P_k^+\end{bmatrix} = \begin{bmatrix}Q_k^+\\P_k^+\end{bmatrix}T_k^+ + t_{k+1,k}\begin{bmatrix}q_{k+1}\\p_{k+1}\end{bmatrix}e_{k+1}^T V_k, \tag{5.7}$$

gdje smo označili $\begin{bmatrix}Q_k^+\\P_k^+\end{bmatrix} = \begin{bmatrix}Q_k\\P_k\end{bmatrix}V_k$ i $T_k^+ = V_k^* T_k V_k$. Napravimo particiju

$$\begin{bmatrix}Q_k^+\\P_k^+\end{bmatrix} = \begin{bmatrix}Q_m^+ & \hat{Q}_p\\P_m^+ & \hat{P}_p\end{bmatrix} \quad T_k^+ = \begin{bmatrix}T_m^+ & D\\t_{m+1,m}^+ e_1 e_m^T & \hat{T}_p\end{bmatrix}$$

i primijetimo da je

$$t_{k+1,k}e_k^T V_k = \underbrace{(0,0,\dots,\tilde{t}_k}_{m},\ \underbrace{b^T)}_{p}$$

jer je matrica V_k umnožak p Hessenbergovih matrica i $\tilde{t}_k = t_k * V_k(k,m)$. Uvrstimo li ovo u (5.3) dobivamo

$$S\begin{bmatrix}Q_m^+ & \hat{Q}_p\\P_m^+ & \hat{P}_p\end{bmatrix} = \begin{bmatrix}Q_m^+ & \hat{Q}_p\\P_m^+ & \hat{P}_p\end{bmatrix}\begin{bmatrix}T_m^+ & D\\t_{m+1,m}^+ e_1 e_m^T & \hat{T}_p\end{bmatrix} + \tilde{t}_{k+1,k}\begin{bmatrix}q_{k+1}\\p_{k+1}\end{bmatrix}e_k^T. \tag{5.8}$$

Izjednačavajući prvih m stupaca dobivamo relaciju (5.2). □

Teorem (5.2.1) tvrdi da smo nakon p QR iteracija nad matricom T_k zapravo obavili m koraka GSOAR metode, uz pretpostavku da nije došlo do deflacije. Ovo je dakle implicitno restartirani GSOAR. (5.2) se dalje proširuje na k koraka metode, počevši od koraka $m+1$, a ne od početka. Sljedeći teorem nam daje formulu za novi početni vektor $\begin{bmatrix}q_1^+\\p_1^+\end{bmatrix}$.

Teorem 5.2.2. *Vrijedi*

$$\begin{bmatrix}q_1^+\\p_1^+\end{bmatrix} = \frac{1}{\tau}\psi(S)\begin{bmatrix}q_1\\p_1\end{bmatrix}, \tag{5.9}$$

gdje je $\psi(\lambda) = \prod_{j=1}^{p}(\lambda.\mu_j)$, *a* τ *je faktor normalizacije.*

Dokaz. Neka je μ shift i neka je $T_k - \mu I = QR$ QR dekompozicija. Iz (4.4) slijedi

$$(S - \mu I)\begin{bmatrix} Q_k \\ P_k \end{bmatrix} - \begin{bmatrix} Q_k \\ P_k \end{bmatrix}(T_k - \mu I) = t_{k+1,k}\begin{bmatrix} q_{k+1} \\ p_{k+1} \end{bmatrix} e_k^T,$$
$$(S - \mu I)\begin{bmatrix} Q_k \\ P_k \end{bmatrix} - \begin{bmatrix} Q_k \\ P_k \end{bmatrix} QR = t_{k+1,k}\begin{bmatrix} q_{k+1} \\ p_{k+1} \end{bmatrix} e_k^T.$$

Neka je $\begin{bmatrix} Q_k^+ \\ P_k^+ \end{bmatrix} = \begin{bmatrix} Q_k \\ P_k \end{bmatrix} Q$. Pomnožimo li drugu relaciju sa e_1 dobivamo

$$\begin{bmatrix} q_1^+ \\ p_1^+ \end{bmatrix} = \tfrac{1}{\rho_{11}}(S - \mu I)\begin{bmatrix} q_1 \\ p_1 \end{bmatrix},$$

gdje je $\rho_{11} = q_1^T R e_1$. □

5.3 Odabir pomaka

Sada nas zanima koje vrijednosti izabrati za pomake u teoremu (5.2.1). Za linearni problem svojstvenih vrijednosti smo vidjeli da što pomaci bolje aproksimiraju neke od neželjenih svojstvenih vrijednosti, novi početni vektor ima bolje informacije o traženim svojstvenim vrijednostima, pa time i rezultirajući Krilovljev potprostor ima bolje informacije o traženim svojstvenim vektorima i očekuje se da će implicitno restartani Arnoldi konvergirati brže.
Na koji način biramo pomake za GSOAR? Riješimo projicirani QEP (4.15) i izaberimo m Ritzovih vrijednosti θ_i za aproksimaciju traženih svojstvenih vrijednosti. Sada možemo iskoristiti preostale Ritzove vrijednosti kao pomake. Problem je u tome što sada imamo $2k$ Ritzovih vrijednosti. Ako bi iskoristili sve neželjene Ritzove vrijednosti za spomake, to bi značilo da smo primijenili $2k - m > p$ pomaka na $k \times k$ matricu T_k. Međutim, za k-koračnu GSOAR metodu, broj pomaka u implicitnom restartanju ne smije preći p, inače se restartanje neće moći provesti. Prema tome, moramo naći način kako odabrati $p = k - m$ pomaka, među njih $2k - m$.
Kako za kvadratični problem reda n imamo $2n$ svojstvenih vrijednosti, skup svojstvenih vektora nije linearno nezavisan, pa se prema tome može desiti da dvije različite svojstvene vrijednosti imaju isti svojstveni vektor. Dakle, ako pomak i tražena Ritzova vrijednost imaju iste svojstvene vektore, restartanjem ćemo izgubiti informaciju o traženom svojstvenom vektoru, pa se takvi pomaci ne smiju primjenjivati. Da bi to izbjegli, projiciramo QEP (1) na ortogonalni komplement od $\text{span}\{y_1, y_2, \ldots, y_m\}$ u odnosu na $\mathcal{G}_{k-1}(A, B; u_1, u_2) = \text{span}\{r_0, r_1, r_2, \ldots, \ldots, r_{k-2}\}$, gdje su $y_1, y_2, \ldots, y_m$ traženi Ritzovi vektori. Zatim izvedemo projicirani QEP reda p i izračunamo njihovih $2p$ svojstvenih vrijednosti. Sada možemo iskoristiti p svojstvenih vrijednosti za pomake. Možemo, na primjer, izabrati one koji su

najdalji od traženih svojstvenih vrijednosti θ_i, $i = 1, \ldots, m$.
Neka su $z_i = Q_k g_i$, $i = 1, \ldots, m$ aproksimacije traženih svojstvenih vektora i označimo $G_m = [g_1,\ g_2, \ldots,\ g_m]$. Ako su neka dva stupca g_i, g_{i+1} kompleksno konjugirani, zamjenimo ih sa njihovim normaliziranim realnim, odnosno imaginarnim dijelom, tako da je konačna matrica G_m realna. Napravimo QR faktorizaciju

$$G_m = [U_m,\ U_\perp] \begin{bmatrix} R_m \\ 0 \end{bmatrix},$$

gdje su U_m i $U_\perp$ $k \times m$ i $k \times p$ stupčano ortogonane matrice. Tada je $Q_k U_\perp$ ortogonalna baza ortogonalnog komplementa od $\text{span}\{g_1, \ldots, g_m\}$ u odnosu na $\mathcal{G}_{k-1}(A, B; u_1, u_2) = \text{span}\{r_0, r_1, r_2, \ldots, \ldots, r_{k-2}\}$. Projicirani QEP početnog problema (1) na $\text{span}\{Q_k U_\perp\}$ je zapravo projicirani QEP manjeg problema (4.15) na $\text{span}\{U_\perp\}$. Sada izračunamo $2p$ njegovih svojstvenih vrijednosti i izaberemo njih p za pomake.
Time smo došli do konačnog algoritma za implicitno restartani GSOAR

Algoritam 12: Implicitno restartani GSOAR

1. Za zadane početne vektore q_1 i p_1, broja željenih svojstvenih vrijednosti m i broja shiftova p, $p \leq k - m$ pokrenimo k koraka GSOAR algortma kako bi generirali matricu Q_k.;
2. Do konvergencije
3. Projiciramo QEP (1) na $\text{span}\{Q_k\}$ da dobijemo QEP (4.15), izaberemo m Ritzovih parova (θ_i, y_i) kako aproksimacije za željenih m svojstvenih vrijednosti i odredimo grešku kao u (4.17);
4. Ako nije došlo do konvergencije, izračunajmo p pomaka i implicitno restartajmo GSOAR;

Što se tiče konvergencije, nju ispitujemo na isti način na koji smo radili kod SOAR-a, dakle prema definiciji (4.17).
Ako tražimo m svojstvenih vrijednosti koje su najbliže zadanoj vrijednosti σ, koristimo transformaciju $\rho = \frac{1}{\lambda - \sigma}$, uz uvjet da je $\det(Q(\sigma)) \neq 0$, koja nam daje novi QEP

$$Q_\sigma(\rho)x = \left(\rho^2 M_\sigma + \rho C_\sigma + K_\sigma\right) x = 0, \tag{5.10}$$

gdje je

$$M_\sigma = \sigma^2 M + \sigma C + K, \tag{5.11}$$
$$C_\sigma = C + 2\sigma M, \tag{5.12}$$
$$K_\sigma = M. \tag{5.13}$$

Matrica M_σ je regularna jer je $\det(M_\sigma) = \det(Q(\sigma)) \neq 0$. Zatim primjenimo algoritam (12) na (5.10). Kod implicitnog restartanja, svojstvene vrijednosti sortiramo prema najvećoj

vrijednosti po modulu. Time izdavajamo one svojstvene vrijednosti početnog QEP-a koje su najbliže vrijednosti σ. Neka je (ρ, y) izračunati svojstveni par za $Q_\sigma(\rho)$. Tada je pripadni svojsveni par za početni problem $\left(\frac{1}{\rho} + \sigma, y\right)$.

Što se tiče odabira broja p, tj. broja pomaka u implicitno restartanom GSOAR-u, nije nužno da je on jednak $p = k - m$. Naime dovoljno je da vrijedi $p \leq k - m$. Drugačiji izbor broja p daje drugačije rezultate, međutim ne postoji određeno pravilo kako se ponaša algoritam u odnosu na odabir broja p.

5.4 Tretiranje deflacije tijekom implicitnog restartanja

Teorem (5.2.1) zahtjeva da nije došlo do deflacije tijekom implicitnog restartanja. Ako pak dođe do deflacije u koracima $m_1, m_2, \ldots, m_j \leq k$, tada je odgovarajući j-ti stupac q_{m_j} u Q_k nula. Označimo sa $\hat{Q}_k$ i $\hat{V}_k$ redom matrice koje smo dobili brisanjem nul stupaca u Q_k i odgovarajućih redaka $m_1, m_2, \ldots, m_j$ u V_k. Tada je $Q_k^+ = Q_k V_k = \hat{Q}_k \hat{V}_k$. Ako još uzmemo u obzir jednadžbu (4.4) dobivamo

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix} \begin{bmatrix} \hat{Q}_k \hat{V}_k \\ P_k V_k \end{bmatrix} = \begin{bmatrix} \hat{Q}_k \hat{V}_k \\ P_k V_k \end{bmatrix} T_k^+ + t_{k+1,k} \begin{bmatrix} q_{k+1} \\ p_{p+1} \end{bmatrix} e_k^T V_k, \tag{5.14}$$

gdje je$T_k^+ = V_k^* T_k V_k$.

Iako matrice $\hat{Q}_k$ još uvijek ima ortonormirane stupce, matrice $Q_k^+ = \hat{Q}_k \hat{V}_k$ nema, jer matrica $\hat{V}_k$ više nije ortogonalna, kako smo joj izbrisali neke retke. Prema tome, matrica $Q_m^+ = \hat{Q}_k \hat{V}_k(:, 1 : m)$ nema ortonormirane stupce, pa (5.14) više nije m-koračna GSOAR dekompozicija. To znači da implicitno restartanje nije moguće provesti ukoliko je došlo do deflacije.

Pokazat ćemo način na koji se ovo može popraviti. Primjetimo da je $\hat{V}_k$ matrica reda $(k - j) \times k$ i da je njen rang $k - j$. Bez smanjena općenitosti, pretpostavimo da je prvih $k - j$ stupaca matrice $\hat{V}_k$ linearno nezavisno, tj. matrica $\hat{V}_{k1}$ koja se sastoji od prvih $k - j$ stupaca matrice $\hat{V}_k$ nije singularna. Napravimo particiju $\hat{V}_k = \left[\hat{V}_{k1},\ \hat{V}_{k2}\right]$. Napravimo QR dekompoziciju matrice $\hat{V}_{k1}$ kako bi dobili

$$\hat{V}_k = U_k R_k = \left[U_{k-j},\ 0\right] \begin{bmatrix} R_{k-j} & R_{12} \\ 0 & I \end{bmatrix}, \tag{5.15}$$

gdje je $\hat{V}_{k1} = U_{k-j} R_{k-j}$ i $R_{12} = U_{k-j}^* \hat{V}_{k2}$. Matrica I je jedinična matrica reda j. Matrica R_k nije singularna i gornje je trokutasta.

Iz (5.15) vidimo da je $U_k = \hat{V}_k R_k^{-1}$. Uzimajući to u obzir, ako pomnožimo (5.14) sa R_k^{-1} dobivamo

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix} \begin{bmatrix} \hat{Q}_k U_k \\ P_k V_k R_k^{-1} \end{bmatrix} = \begin{bmatrix} \hat{Q}_k U_k \\ P_k V_k R_k^{-1} \end{bmatrix} R_k T_k^+ R_k^{-1} + t_{k+1,k} \begin{bmatrix} q_{k+1} \\ p_{p+1} \end{bmatrix} e_k^T V_k R_k^{-1}. \tag{5.16}$$

Kako je R_k^{-1} gornje trokutasta, matrica $R_k T_k^+ R_k^{-1}$ je Hessenbergova. Primjetimo da matrica V_k ima samo $p = k - m$ ne nul subdijagonalnih elemenata. Prema tome, prvi mogući element $\tilde{\beta}$ koji je različit od nule u $e_k^T V_k$ je na poziciji m i

$$t_{k+1,k} e_k^T V_k R_k^{-1} = (0, \ldots, 0, \tilde{\beta}, b^T),$$

gdje je $\tilde{\beta} = t_{k+1,k} V_k(k, m)/e_m^T R_k e_m$. Izjednačavanjem prvih m stupaca jednadžbe (5.16) dobivamo

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix} \begin{bmatrix} \tilde{Q}_m^+ \\ \tilde{P}_m^+ \end{bmatrix} = \begin{bmatrix} \tilde{Q}_m^+ \\ \tilde{P}_m^+ \end{bmatrix} \tilde{T}_m^+ + \beta_m^+ \begin{bmatrix} q_{k+1}^+ \\ p_{p+1}^+ \end{bmatrix} e_m^T, \tag{5.17}$$

pri čemu je $\tilde{Q}_m^+ = \hat{Q}_k U_k(:, 1 : m)$, $\tilde{P}_m^+ = P_k V_k(:, 1 : m) R_m^{-1}$, gdje smo sa R_m označili vodeću $m \times m$ podmatricu od R_k, a s $\tilde{T}_m^+$ vodeću $m \times m$ podmatricu matrice $R_k T_k^+ R_k^{-1}$. Definirajmo još i zadnji dio

$$\begin{bmatrix} q_{m+1}^+ \\ p_{m+1}^+ \end{bmatrix} = \frac{1}{\beta_m^+} f_m^+ = \tilde{t}_{m+1,m}^+ \begin{bmatrix} \hat{Q}_k U_k \\ P_k V_k R_k^{-1} \end{bmatrix} e_{m+1} + \tilde{\beta} \begin{bmatrix} q_{k+1} \\ p_{k+1} \end{bmatrix}, \tag{5.18}$$

$$\beta_m^+ = \| \tilde{t}_{m+1,m} \hat{Q}_k U_k e_{m+1} + \tilde{\beta} q_{k+1} \|. \tag{5.19}$$

Jednadžba (5.15) pokazuje da matrica $U_k(:, 1 : m)$ ima ortogonalne stupce ako je $m \leq k - j$, tj. $j \leq k - m$. To znači da $\tilde{Q}_m^+ = \hat{Q}_k U_k(:, 1 : m)$ ima ortogonalne stupce ako broj j, koji označava u koliko koraka GSOAR-a je došlo do deflacije, ne prelazi $k-m$. Ako je $m > k-j$, prvih $k - j$ stupaca matrice $\tilde{Q}_m^+$ je ortonormirano i zadnjih $m-(k-j)$ stupaca od U_k su nula, stoga su zadnjih $m - (k - j)$ stupaca od $\tilde{Q}_m^+$ nula. To znači da u (5.17) imamo $m - (k - j)$ deflacija. Kako vrijedi da je $(\tilde{Q}_m^+)^* q_{m+1}^+ = 0$, (5.17) definira dekompoziciju dobivenu nakon m koraka GSOAR-a.

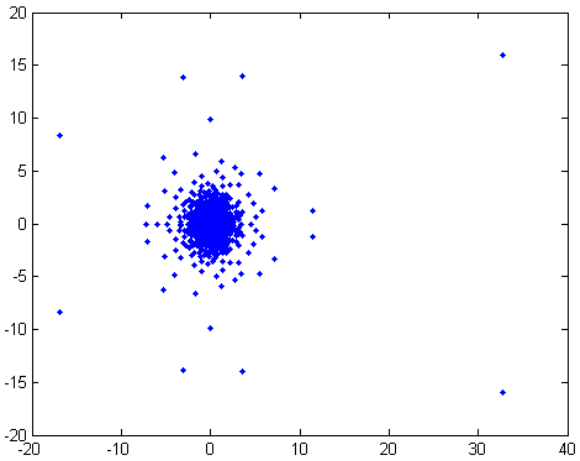
Poglavlje 6

Numerički eksperimenti

U ovom poglavlju ćemo napraviti četiri numerička eksperimenta. U svakom eksperimentu je red matrice 1000. Slike koje predstavljaju svojstvene vrijednosti problema su dobivene korištenjem MATLABove funkcije `polyeig`.

6.1 Slučajno generirane matrice

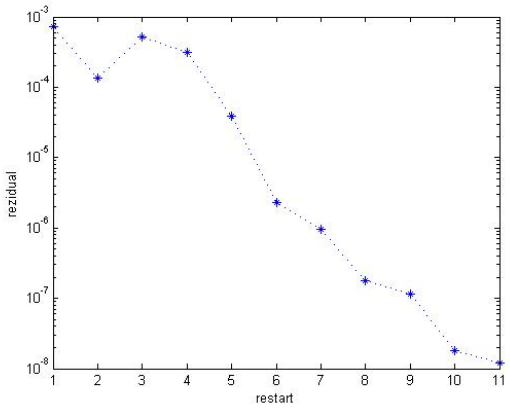
U ovom primjeru smo generirali tri slučajne matrice M, C i K. Svojstvene vrijednosti za taj problem izgledaju kao na slici (6.1.1).



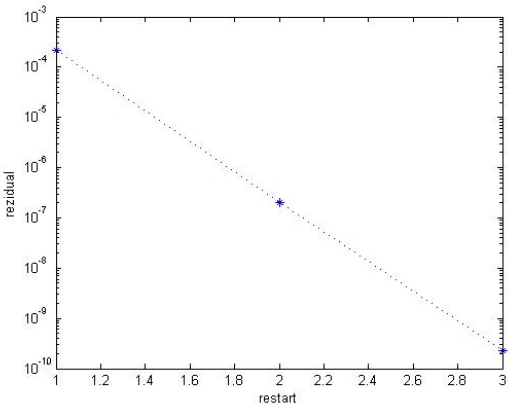
Slika 6.1.1: Svojstvene vrijednosti za slučajno generirane matrice M, C i K

Pokrenuli smo implicitno restartani GSOAR, dakle algoritam (12). Računali smo 6

svojstvenih vrijednosti s najvećom apsolutnom vrijednošću. Broj pomaka nam je bio 3, dok je ukupni broj iteracija za GSOAR metodu, tj. maksimalna dimenzija Krilovljevog potprostora $\mathcal{G}_k(A, B; u_1, u_2)$, bila 10. Tolerancija na vrijednost reziduala (4.17) je stavljena na $\sqrt{eps}$, gdje je eps strojna točnost.
Za ovaj primjer, GSOAR metodi je trebalo 11 iteracija. Prilikom izvedbe algoritma, nije došlo do deflacije.
Radi usporedbe brzine konvergencije u ovisnosti o broju pomaka, isti problem smo pokrenuli uz broj pomaka jednak 10. Maksimalna dimenzija potprostora je 16. U tom slučaju su nam trebale tri iteracije da dođemo do željene konvergencije.
Na slikama (6.1.2) i (6.1.3) su redom nacrtani reziduali ovisno o broju restartanja.

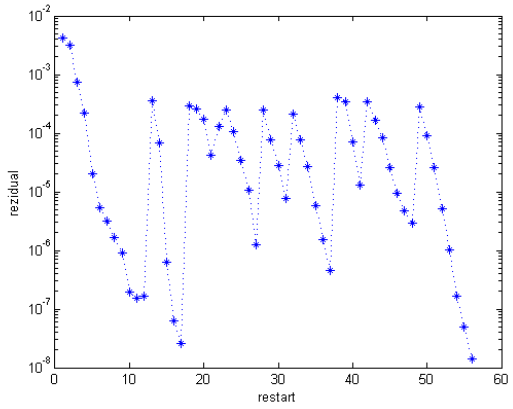


Slika 6.1.2: $m = 6$, $p = 2$

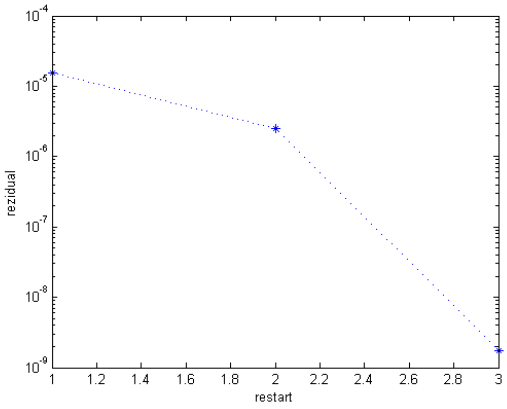


Slika 6.1.3: $m = 6$, $p = 10$

Za isti primjer smo proveli i implicitno restartani Arnoldijev algoritam. Koristili smo linearizaciju drugom pridruženom formom (3.7). Tražili smo također 6 svojstvenih vrijednosti sa najvećom apsolutnom vrijednošću. Broj pomaka nam je bio $p = k - m$, dakle maksimalni broj iteracija manje broj traženih svojstvenih vrijednosti. Za $k = 10$ nam je trebalo 56 iteracija. 4 svojstvene vrijednosti su zaključane. Tolerancija na vrijednost reziduala je također bila $\sqrt{eps}$. Isti problem smo pokrenuli uz k=20. Za konvergenciju su trebale samo 3 iteracije, a zaključane su 2 svojstve vrijednosti. Na slikama (6.1.4) i (6.1.5) vidimo kako se rezidual ponašao tijekom restartanja.



Slika 6.1.4: $k = 10$



Slika 6.1.5: $k = 20$

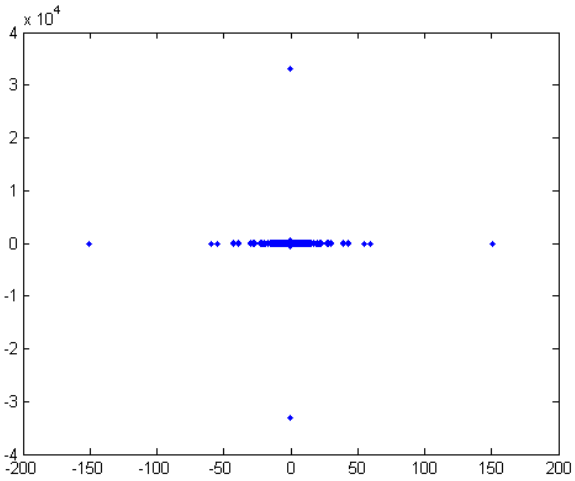
Dakle GSOAR se bolje ponaša i u slučajevima kad nam je maksimalna dimenzija potprostora manja.
U tablici (6.1) vidimo ukupne rezultate koje smo dobili

Metoda	**tol**	**m**	**p**	**k**	**br. iteracij**a
GSOAR	$\sqrt{eps}$	6	2	10	11
GSOAR	$\sqrt{eps}$	6	10	16	3
Arnoldi	$\sqrt{eps}$	6	4	10	56
Arnoldi	$\sqrt{eps}$	6	14	20	3

Tablica 6.1: Prikaz ukupnih rezultata
Slučajno generirane matrice

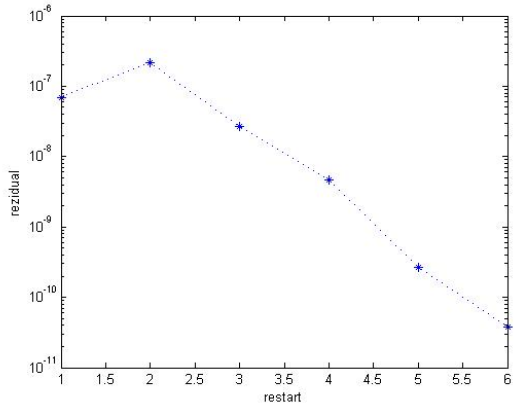
6.2 Žiroskopski sustav

Kod žiroskopskog sustava matrica M je simetrična pozitivno definitna, C je antisimetrična, dok je K simetrična negativno definitna.
Generirali smo slučajne matrice M, C i K tog oblika. U ovo primjeru želimo naći $m = 10$ svojstvenih vrijednosti sa najvećim modulom. Spektar žiroskopskog sustava izgleda kao na slici (6.2.1).

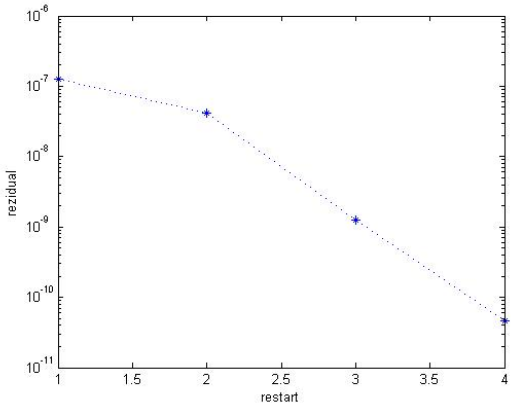


Slika 6.2.1: Svojstvene vrijednosti za žiroskopski sustav

Dakle, one su simetrične s obzirom na realnu i imaginarnu os. Ta struktura je očuvana u algoritmu GSOAR.
U prvoj primjeni implicitno restartanog GSOAR-a smo za broj pomaka uzeli $p = 6$. Maksimalna dimenzija potprostora nam je $k = 20$. Tolerancija na rezidual je 10^{-10}. Tijekom procesa nije došlo do deflacije, a ukupni broj restartanja je bio 6. Kada smo povećali broj pomaka na $p = 10$, broj restartanja se smanjio, i iznosio je 4. Na slikama (6.2.2) i (6.2.3) vidimo kako se ponašao rezidual tijekom restartanja za oba slučaja.



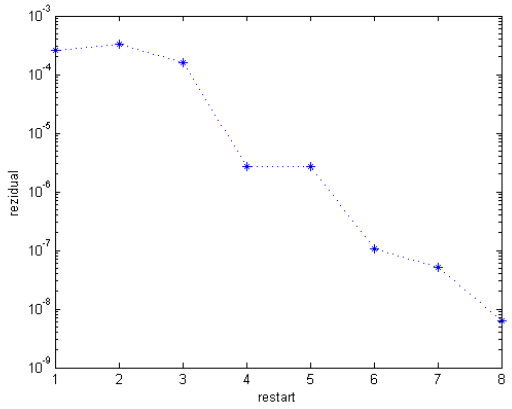
Slika 6.2.2: $m = 10$, $p = 6$



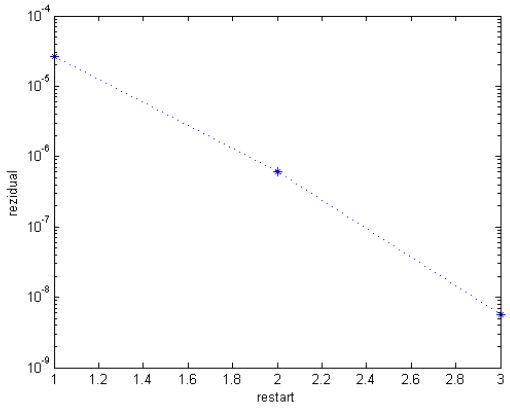
Slika 6.2.3: $m = 10$, $p = 10$

Kod Arnoldijevog algoritma, za $m = 10$ i $k = 20$ bilo je potrebno 8 iteracija, te je svih deset traženih vrijednosti zaključano. No, kada smo povećali maksimalnu dimenziju

na $k = 30$, broj iteracija se smanjio na njih 3. Također, svih deset svojstvenih vrijednosti je zaključano. Toleranciju smo zadali na $\sqrt{eps}$. Na slikama (6.2.4) i (6.2.5) su prikazani reziduali za oba slučaja.



Slika 6.2.4: $k = 20$



Slika 6.2.5: $k = 30$

Primijetimo da iako se čini da algoritmi podjednako dobro rade, na Arnoldijev algoritam smo imali veću toleranciju. Rezultati koje smo dobili su sumirani u tablici (6.2).

Metoda	**tol**	**m**	**p**	**k**	**br. iteracija**
GSOAR	10^{-10}	10	6	20	6
GSOAR	10^{-10}	10	10	20	4
Arnoldi	$\sqrt{eps}$	10	10	20	8
Arnoldi	$\sqrt{eps}$	10	20	30	3

Tablica 6.2: Prikaz ukupnih rezultata
Žiroskopski sustav

6.3 Nepriguseni sustav

Diferencijalni sustav, gdje su M, C i K realne simetrične matrice se zove *prigušen* ukoliko je zadovoljen uvjet

$$\min_{\|x\|_2=1} \left[(x^*Cx)^2 - 4\,(x^*Mx)\,(x^*Kx) \right] > 0. \tag{6.1}$$

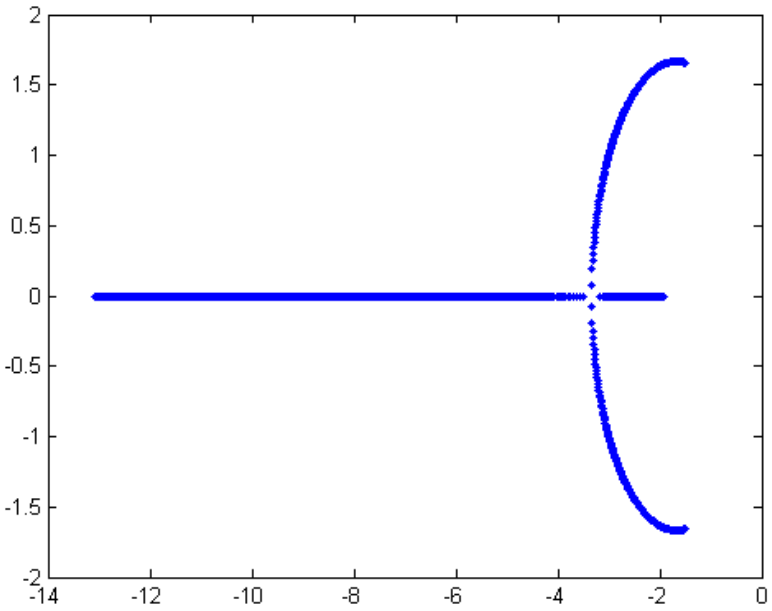
Vibracijski sustav je određen diferencijalnom jednadžbom drugog reda

$$M\ddot{q}(t) + C\dot{q}(t) + Kq(t) = f(t), \tag{6.2}$$

gdje je matrica M dijagonalna i predstavlja masu. Matrice C i K su tridijagonalne i redom predstavljaju prigušenje i krutost.
U ovom primjeru matrice su definirane kao

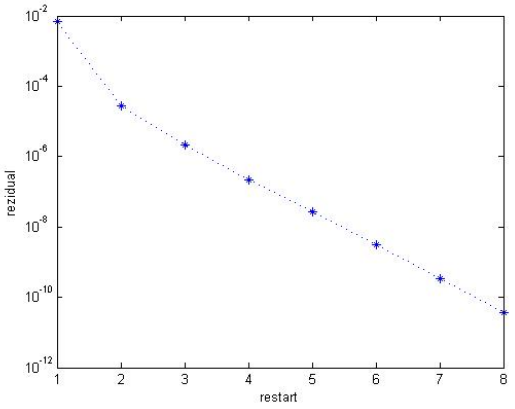
$$\begin{aligned} M &= I \\ C &= \text{tridiag}(-3, 9, -3) \\ K &= \text{tridiag}(-5, 15, -5) \end{aligned}$$

Ovo je primjer neprigušenog sustava. Svojstvene vrijednosti izgledaju kao na slici (6.3.1).

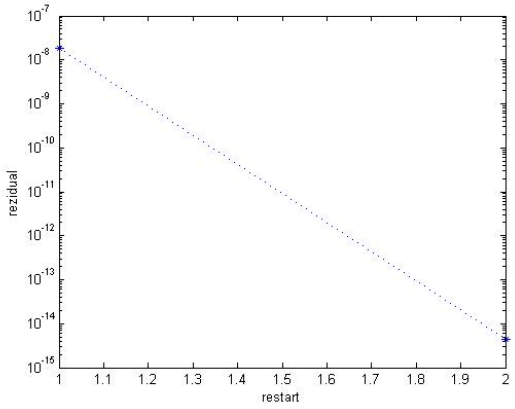


Slika 6.3.1: Svojstvene vrijednosti za neprigušeni sustav

Kod ovo primjera smo koristili pomak $\sigma = -13$, te su nam nove matrice definirane kao u (5.11). Računali smo $m = 6$ svojstvenih vrijednosti najbližih vrijednosti σ koristeći implicitno restartani GSOAR. Broj pomaka je bio $p = 3$, dok je maksimalna dimenzija potprostora bila $k = 10$. Toleranciju na rezidual smo stavili na 10^{-10}. Broj restartanja za ovako postavljen problem je bio 8, i nije došlo do deflacije. Kada smo broj pomaka povećali na $p = 10$ i $k = 20$, broj restartanja se smanjio i iznosio je 2. Na slikama (6.3.2) i (6.3.3) su prikazani reziduali tijekom restartanja.

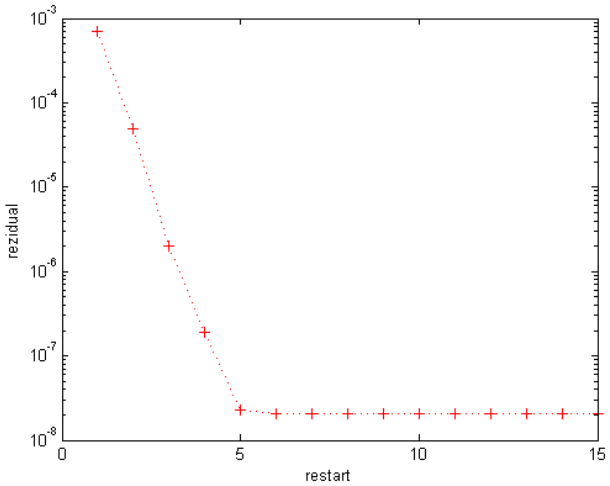


Slika 6.3.2: $m = 6$, $p = 3$



Slika 6.3.3: $m = 6$, $p = 10$

Kod Arnoldijevog algoritma, za isti problem, nam je uz maksimalnu dimenziju potprostora $k = 10$ trebalo 15 restartanja. Slika (6.3.4) prikazuje rezidual tijekom restartanja.



Slika 6.3.4: $k = 10$

Opet primijetimo da je tolerancija na rezidual kod Arnoldija bila veća, tj. $\sqrt{eps}$. Naime, za toleranciju 10^{-10} ovaj primjer nije iskonvergirao u prvih 200 restartanja. Ukupni rezultati za ovaj primjer su prikazani u tablici (6.3)

Metoda	tol	m	p	k	br. iteracija
GSOAR	10^{-10}	6	3	20	8
GSOAR	10^{-10}	6	10	20	2
Arnoldi	$\sqrt{eps}$	6	4	10	15

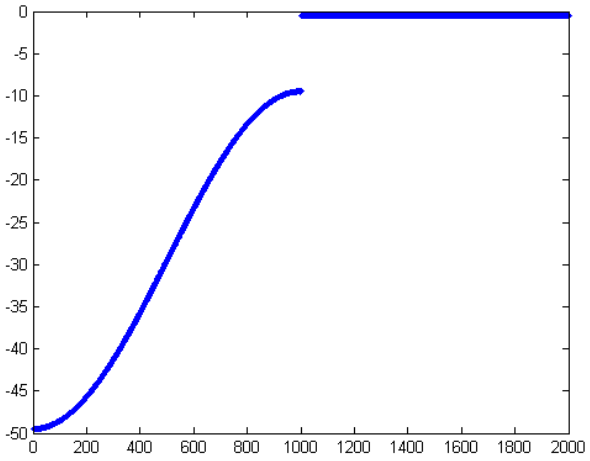
Tablica 6.3: Prikaz ukupnih rezultata
Neprigušeni sustav

6.4 Prigušeni sustav

U ovom primjeru definiramo matrice M, C i K na sljedeći način:

$$\begin{aligned} M &= I, \\ C &= \text{tridiag}(-10, 30, -10), \\ K &= \text{tridiag}(-5, 15, -5). \end{aligned}$$

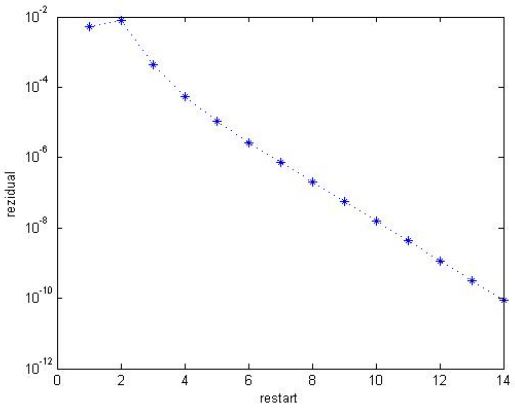
Ovo je primjer prigušenog sustava. Svojstvene vrijednosti za ovaj problem izgledaju kao na slici (6.4.1).



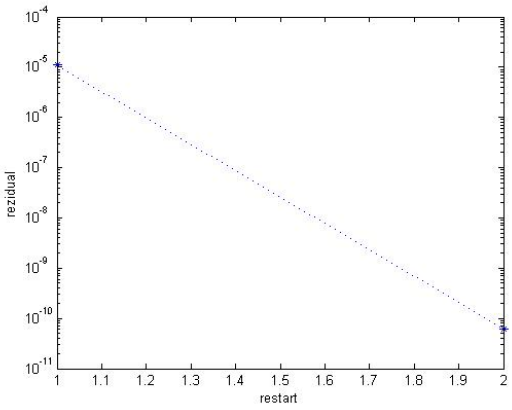
Slika 6.4.1: Svojstvene vrijednosti za prigušeni sustav

U ovom primjeru smo tražili $m = 6$ svojstvenih vrijednosti koje su najbliže vrijednosti $\sigma = -40$. U prvom slučaju smo uzeli da je broj pomaka $p = 2$ i maksimalna dimenzija potprostora $k = 10$. Do deflacije nije došlo, a bilo je potrebo 14 restartanja. U drugom slučaju smo stavili da je $p = 10$ i $k = 16$ te su nam bilo potrebo samo 2 restartanja i nije

došlo do deflacije. Tolerancija je u oba slučaja stavljena na 10^{-10}. Na slikama (6.4.2) i (6.4.3) su prikazani reziduali tijekom restartanja.

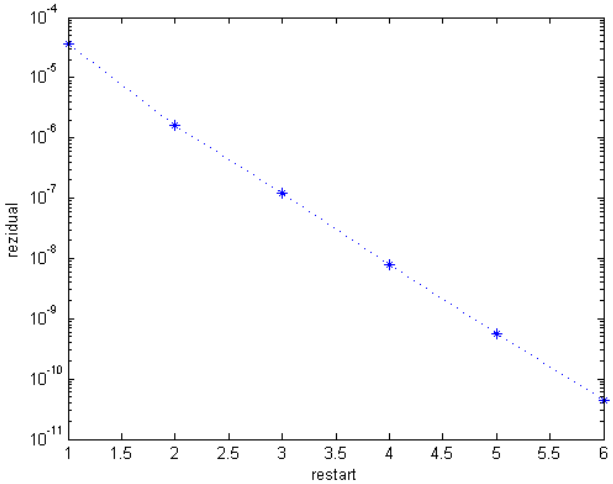


Slika 6.4.2: $m = 6$, $p = 2$



Slika 6.4.3: $m = 6$, $p = 10$

Kod Arnoldijevog algoritma, za isti problem, nam je uz maksimalnu dimenziju potprostora $k = 10$ trebalo 6 restartanja. Slika (6.4.4) prikazuje rezidual tijekom restartanja.



Slika 6.4.4: $k = 10$

Ukupni rezultati su zapisani u tablici (6.4).

Metoda	**tol**	**m**	**p**	**k**	**br. iteracija**
GSOAR	10^{-10}	6	2	10	14
GSOAR	10^{-10}	6	10	16	2
Arnoldi	10^{-10}	6	4	10	6

Tablica 6.4: Prikaz ukupnih rezultata
Prigušeni sustav

6.5 Zaključak

GSOAR metoda je pokazala jako dobre rezultate s obzirom na veličinu problem te maksimalnu dimenziju potprostora koji koristimo. Naime, za velike $n = 1000$ smo k povećavali do najviše 16, što je relativno malo. Također, za mali broj pomaka p, smo dobili jako dobre rezultate uz mali broj iteracija. Povećavanjem broja p, očekivano dobivamo bolje rezultate. Što se tiče Arnoldijevog algoritma, imao je također dobre rezultate, no pri manjoj toleranciji ne bi došlo do konvergencije u prvih 200 restartanja. Također, potreban je veći broj pomaka i veća maksimalna dimenzija potprostora nego kod GSOAR-a da dođe do konvergencije.
Uspješnost GSOAR metode možemo pripisati tome što se konačne svojstvene vrijednosti dobivaju iz manjeg kvadratnog problema, a ne kao kod Arnoldija, gdje svojstvene vrijednosti dobivamo iz manje Hessenbergove matrice. Kao što smo rekli prije, svojstva svojstvenih vrijednosti su očuvana prjektiranjem na manji potprostor (kao što radimo kod GSOAR-a), dok se kod Arnoldija ta svojstva gube linearizacijom.
Treba napomenuti da smo za zadnja dva primjera koristili pomake u početnom problemu, jer su svojstvene vrijednosti gusto raspoređene. Kako bi ubrzali konvergenciju napravili smo pomak u spektru. Novodefinirani problem ima svojstvene vrijednosti koje su blizu zadanom pomaku.

Bibliografija

[1] P. Arbenz, "Numerical methods for solving large scale eigenvalue problems, lecture notes, chapter 8," *Eidgenössische Technische Hochschule Zürich*, 2012.

[2] Z. Drmač, "Numerička analiza 1," *skripta*, 2008.

[3] D. C. Sorensen, "Implicit Application of Polynomial Filters in a k-step Arnoldi Method," *Dept. of Mathematical Sciences, Rice University*.

[4] Y. Saad, "Chebyshev Acceleration Tehniques for Solving Nonsymmetric Eigenvalue Problems," *Mathematics of Computation, Volume 42, Number 166*, 1984.

[5] L. R. B. and D. C. Sorensen, "Deflation Tehniques for an Implicitly Restarted Arnoldi Iteration."

[6] R. Lechoucq, Sorensen, and C. Yang, "Arpack User's Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods."

[7] [Online]. Available: http://people.sc.fsu.edu/~jburkardt/m_src/arpack/arpack.html

[8] K. Meerbergen, "The Quadratic Arnoldi Method for the Solution of the Quadratic Eigenvalue Problem," *Katholieke Universiteit Leuven, Department of Computer Science*, 2007.

[9] S. Hammarling, C. J. Munro, and F. Tisseur, "An Algorithm for the Complete Solution of Quadratic Eigenvalue Problem," *The Univeristy of Manchester*, 2011.

[10] Z. Bai and Y. Su, "Soar: a Second-order Arnoldi Method for the Solution of the Quadratic Eigenvalue Problem," *SIAM J. Matrix Anal. Appl. Vol. 26, No. 3, pp. 640-659*, 2005.

[11] L. Zhou, L. Bao, Y. Lin, Y. Wei, and Q. Wu, "Restarted Generalized Second-order Krylov Subspace Methods for Solving Quadratic Eigenvalue Problems," *World Academy of Science, Engineering and Technology, Vol:4*, 2010.

[12] Z. Jia and Y. Sun, "Implicitly Restarted Generalized Second-order Arnoldi Type Algorithms for the Quadratic Eigenvalue Problem," *arXiv:1005.3947v5 [math.NA]*, 2014.

[13] D. C. Sorensen, "Deflation for Implicitly Restarted Arnoldi Methods," *Center for Research on Parallel Computation, Rice University*, 1998.

[14] G. W. Stewart and J. Sun, *Matrix Perturbation Theory*, W. Reinboldt and D. Siewiorek, Eds. Academic Press, Inc., 1990.

[15] F. Tisseur and K. Meerbergen, "The Quadratic Eigenvalue Problem," *The Univeristy of Manchester*, 2001.

Sažetak

U ovo radu smo proučavali kvadratični svojstveni problem (QEP - Quadratic Eigenvalue Problem). Točnije, zanimalo nas je kako naći skalar $\lambda \in \mathbb{C}$ i vektor $z \in \mathbb{C}^n$ tako da za zadane matrice M, C i K, reda n vrijedi

$$\left(\lambda^2 M + \lambda C + K\right) z = 0.$$

U zadnjem poglavlju smo vidjeli neke primjere iz primjene koje rješava ovaj problem. Predložili smo dvije metode za rješavanje ovakvog problema. Prva metoda se svodi na to da nelinearni problem linearizirамo, tj. svedemo ga na standardni problem svojstvenih vrijednosti, te novodefinirani problem riješimo Arnoldijevim algoritmom. Ovdje nastaje problem, jer se dimenzija problem linearizacijom udvostručuje pa nam se povećavaju zahtjevi za memorijom prilikom implementacije programa.
Arnoldijev algoritam nalazi određen broj svojstvenih vrijednosti i najčešće se koristi kada imamo velike rijetke matrice S. On računa ortgonalnu bazu Q za Krilovljev potprostor zadane dimenzije, te Hessenbergovu matricu H čije svojstvene vrijednosti aproksimiraju svojstvene vrijednosti matrice S. Naveli smo teorem koji kaže da rezidual u Arnoldijevom algoritmu ovisi o početnom vektoru, u smislu da je rezidual manji što je početni vektor bliži svojstvenom vektoru tražene svojstvene vrijednosti. Tu činjenicu smo iskoristili kako bi opisali implicitno restartani Arnoldijev algoritam. U tom algoritmu dozvoljavamo više iteracija nego što nam je potrebno. Nakon k iteracija imamo k aproksimacija za svojstvene vrijednosti. Izdvojimo m svojstvenih vrijednosti koje su nam od interesa. Ostale svojstvene vrijednosti koristimo kako bi napravile pomake u matrici H, te time prigušili doprinos svojstvenih vrijednosti, koje nam nisu od interesa, u početnom vektoru. Taj postupak smo napravili implicitno, tj. Arnoldijev algoritam ne pokrećemo ponovno od prvog koraka, već od m-tog. Razradili smo metode *lockinga* odnosno zaključavanja svojstvenih vrijednosti koje su nam od interesa a iskonvergirale su ranije, te *purginga* odnosno izbacivanja svojstvenih vrijednosti koje su iskonvergirale a nisu nam od interesa.
Mana metode linearizacije je u tome što se problem udupla, te se izgubi početna struktura problema. Kako bi očuvali tu strukturu moramo raditi direktno sa matricama M, C i K. Zbog toga smo opisali SOAR (Second Order Arnoldi Method) metodu. Ta metoda projicira početni QEP velike dimenzije, na QEP puno manje zadane dimenzije koji ima istu

strukturu, a lakše je izračunati njegove svojstvene vrijednosti. Vidjeli smo da postoji veza između SOAR-a i Arnoldija. Naime, do deflacije dolazi u istom koraku algoritama. Kako bi razvili algoritam koji koristi istu ideju kao implicitno restartani Arnoldijev algoritam, bilo je potrebno modificirati SOAR metodu. Definirali smo generaliziranu SOAR metodu (GSOAR) te na njoj proveli implicitno restartanje. Problem nastaje kod odabira pomaka. Naime, kvadratični problem ima $2n$ svojstvenih vrijednosti, pa prema tome i $2n$ svojstvenih vektora. To znači da skup svih svojstvenih vektora nije linearno nezavisan. Prema tome, postoje svojstvene vrijednosti koje su možda različite, a imaju iste svojstvene vektore. To znači da nakon k koraka GSOAR algoritma imamo $2k$ aproksimacija za svojstvene vektore. Ako tražimo m svojstvenih vrijednosti imamo $2k - m$ kandidata za pomake. Među tih $2k - m$ kandidata može se desiti da postoje svojstveni vektori koji odgovaraju svojstvenim vektorima nekih od m traženih svojstvenih vrijednosti. Ako te vrijednosti iskoristimo za pomake, izgubit ćemo informaciju o željenoj svojstvenoj vrijednosti. Predložili smo način na koji se taj problem rješava.
Na kraju smo dali nekoliko primjera iz primjene i na njima proveli opisane metode. GSOAR se pokazao kao bolji rješavač, jer za mali broj pomaka daje jako dobre rezultate. Kod Arnoldijevog algoritma smo imali primjer u kojem metoda nije konvergirala za prvih 200 restartanja.

Summary

This thesis presents numerical methods for solving quadratic eigenvalue problem (QEP): find scalar $\lambda \in \mathbb{C}^n$ and vector $z \in \mathbb{C}^n$ so that, for given matrices M, C and K

$$\left(\lambda^2 M + \lambda C + K\right) z = 0.$$

In last chapter we saw few examples in which we apply this problem. Two methods are presented for solving this problem. First method finds linearization for nonlinear problem. That linearization defines standard eigenvalue problem which is solved using Arnoldi algorithm. The problem is that new linearized problem has double dimension.
Arnoldi algorithm finds a specifed number of eigenvalues and is used for large, sparse matrices S. It computes an orthogonal basis Q for Krylov subspaces $\mathcal{K}$ of given dimension, and Hessenberg matrix H which represents projection of S onto $\mathcal{K}$. The eigenvalues of H are approximate eigenvalues of S. The residual is smaller if the initial vector is closer to eigenspace for wanted eigenvalues. That fact is used to describe implicitly restarted Arnoldi algorithm. That algorithm has more iterations than needed. After k iterations we have k approximations for eigenvalues. We define m wanted eigenvalues. The remaining eigenvalues are used for shifts that define polynomial filter in implicit restart. The algorithm is implicit, which means that Arnoldi iterations start from step m and not from first step. Locking and purging are described. Linearization doubles the dimension of inital problem, and the inital structure of problem is lost. If we want to preserve the structure of the problem, we use matrices M, C and K and project them onto the search subspace, which is the key idea of the SOAR procedure. Beacuse of that, SOAR method is introduced. This method projects inital QEP (of large dimension) onto smaller QEP so we can easier find its eignvalues. It is shown that there exists a conection between Arnoldi and SOAR procedure, meaning they deflate at same step. If we want to develop idea of implicit restarting as in Arnoldi procedure we need to define generalized SOAR. The problem in restarting is choosing the shifts. Beacuse there are two eigenvalues, that can be different, and have the same eigenvector. After k steps of GSOAR we have $2k$ approximations of eigenvalues. If we need m eigenvalues, than we have $2k - m$ candidates for shifts. It is discussed how to choose the shifts and not use the one with eigenvector for one of the wanted eigenvalues.
We used selected numerical examples to illustrate preformances of the described methods.

GSOAR has good results, even for small number of shifts. There is example for wich Arnoldi iterations did not converge in first 200 restarts.

Životopis

Ivana Šain, rođena je 14. veljače 1991. godine u Mostaru. Osnovnu školu Antuna Branka Šimića završava 2005. godine u Mostaru. Gimnaziju fra Grge Martića, opći smjer završava 2009. godine u Mostaru. Školovanje je nastavila u Zagrebu, gdje je 2012. godine završila Preddiplomski sveučilišni studij matematike na Prirodoslovno - matematičkom fakultetu. Pri istom fakultetu je 2012. godine upisala Diplomski sveučilišni studij Primijenjene matematike.
Dobiva priznanje za izniman uspjeh na studiju u akademskoj godini 2013./2014. na Prirodoslovno - matematičkom fakultetu, Matematički odsjek.