QR razcep zgornje Hessenbergove matrike

Aleš Kert, 63170010

27.3.2023

Abstract

1 Uvod

Hitrost QR razcepa je odvisna od oblike podane matrike. V primeru, da je podana matrika zgornja Hessenbergova, je možno pohitriti QR razcep iz $O(n^3)$ na $O(n^2)$, z uporabo n Givensovih rotacij. To lahko dodatno izkoristimo pri numeričnem računanju lastnih vrednosti, pri QR iteraciji, kjer osrednji korak, QR razcep, ni več ozko grlo časovne kompleksnosti.

Naše algoritme smo implementirali v programskem jeziku Python, uporabljali pa knjižnico NumPy [1].

2 Metode

Za naš algoritem so ključni štiri pojmi - zgornja Hessenbergova matrika, Givensove rotacije, QR razcep in QR iteracija.

2.1 Zgornja Hessenbergova matrika

Zgornja Hessenbergova matrika je posebna oblika kvadratne matrike. Po strukturi lahko rečemo, da je "skoraj" diagonalna, saj so vsi elementi pod prvo poddiagonalo ničelni. Formalno je matrika zgornja Hessenbergova, če velja $\forall i, j, i > j + 1 : a_{i,j} = 0$. Primer takšne matrike je viden v Enačbi 1.

$$H = \begin{bmatrix} 1 & 2 & 1 & 4 \\ 3 & 2 & 4 & 3 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 5 & 1 \end{bmatrix} \tag{1}$$

2.2 Givensove rotacije

Givensova rotacija je zrcaljenje čez ravnino, ki jo razpenjata dve koordinatni osi. Rezultat rotacije je izničen spodnji element, kot je vidno na Enačbi 2, kjer sta c in s kosinus in sinus kota rotacije, $r = \sqrt{a^2 + b^2}$. Ti vrednosti lahko zapišemo tudi kot $c = \frac{a}{r}$ in $s = -\frac{b}{r}$. Primer rotacije v matriki je viden v Enačbi 3. Givensovo rotacijo je možno izvesti v linearnem času, saj se spremenita zgolj dve vrstici, kar zahteva $6 * O(n)$ operacij, vse ostale vrednosti pa ostanejo iste.

$$\begin{bmatrix} c & -s \\ s & c \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 6 & 5 & 0 \\ 5 & 1 & 4 \\ 0 & 4 & 3 \end{bmatrix} \approx \begin{bmatrix} 7.8102 & 4.4813 & 2.5607 \\ 0 & -2.4327 & 3.0729 \\ 0 & 4 & 3 \end{bmatrix} \tag{3}$$

2.3 QR razcep

QR razcep je razcep matrike, ki generira ortogonalno matriko Q in zgornjetrikotno matriko R, da $A = QR$. V našem primeru lahko iz zgornje Hessenbergove matrike H pridobimo zgornjetrikotno R z uporabo $n-1$ Givensovih rotacij. Ortogonalno matriko Q pa je možno pridobiti z zmnožkom vseh uporabljenih Givensovih rotacij, oz. $Q = G_1^T G_2^T ... G_n^T$.

V splošnem takšen algoritem zahteva $\frac{10}{3}n^3$ operacij [2], vendar ga je za to posebno matriko možno izvesti z $6n^2$ operacij.

2.4 QR iteracija

QR iteracija je algoritem za izračun lastnih vrednosti matrike. Kot hrbtenico uporablja QR razcep. Potek algoritma je viden v spodnji psevdokodi, kjer je A_i trenutni približek matrike, L_i spodnjetrikotni del matrike A_i. Končna matrika A_i je zelo blizu zgornjetrikotni matriki, njene lastne vrednosti pa so njeni diagonalni elementi.

$A_0 \leftarrow A$
$i \leftarrow 0$
while $\frac{||L_i||_F}{||A_i||_F}$ **do**
    $Q_i, R_i \leftarrow qr(A_i)$
    $A_{i+1} \leftarrow R_i Q_i$
    $i \leftarrow i + 1$
end while

3 Rezultati

Implementirane algoritme smo primerjali z že implementiranimi metodami v knjižnici NumPy. Testiranje QR razcepa je bilo opravljeno na kvadratnih matrikah dimenzij $n \in [3, 1000]$, kjer se je za vsako velikost opravilo 10 ponovitev. Testiranje QR iteracije je bilo opravljeno na kvadratnih matrikah dimenzij $n \in [3, 50]$, kjer se je za vsako velikost opravilo 5 ponovitev. Pri testiranju QR iteracije smo se omejili na tridiagonalne simetrične matrike, saj so to matrike, ki imajo zagotovljeno realne lastne vrednosti, ter so še vedno zgornje Hessenbergove. Ponovitve eksperimentov so bile opravljene za pridobitev bolj gladke krivulje.

Rezultati evalvacije so vidni na Sliki 1 in Sliki 2. Pri QR razcepu je takoj opazno, da je prilagojena verzija QR razcepa bistveno hitrejša od privzete. Vendar pa je pri QR iteraciji naša implementacija občutno počasnejša. To bi lahko dodatno omejili z večjo toleranco do napake. Rezultat časovne kompleksnosti QR iteracije ni toliko presenetljiv, saj je naša implementacija zelo naivna, že vgrajene funkcije pa so bistveno bolje optimizirane.

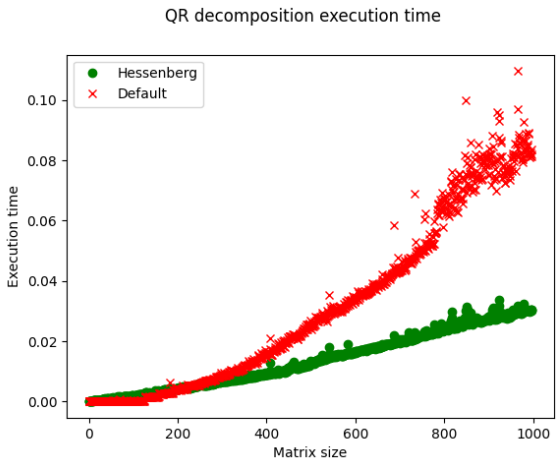

Figure 1: Časovna kompleksnost QR razcepa.

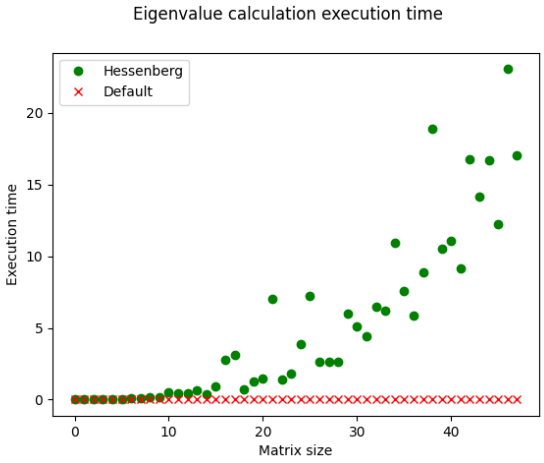

Figure 2: figure
Časovna kompleksnost QR iteracije.

4 Zaključek

Naša implementacija QR razcepa, prirejena za zgornje Hessenbergove matrike je veliko hitrejša od privzete, pri matrikah velikosti 1000×1000 je skoraj štirikrat hitrejša, ta razlika pa se zgolj veča ($O(n^2)$ vs. $O(n^3)$). Vendar pa je naša QR iteracija počasnejša, saj je popolnoma neoptimizirana verzija algoritma. Naš algoritem bi bilo možno izboljšati z implementacijo QR iteracije z enojnim premikom, vendar pa je to izven obsega naše naloge.

References

[1] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, Sept. 2020.

[2] J. Demmel, *Applied Numerical Linear Algebra*. Other Titles in Applied Mathematics, Society for Industrial and Applied Mathematics, 1997.