Einführung in die Grundlagen der Numerik

Winter semester 2019/2020
Prof. Dr. Marc Alexander Schweitzer
Denis Düsseldorf

universitätbonn

Übungsblatt 8.

Abgabe: 10.12.2019 vor der Vorlesung

Aufgabe 21. (Gerschgorin Kreise)

a) Betrachten Sie die Matrix

$$A = \begin{bmatrix} 2 & 9 & 0 & 2 \\ -1 & 2 & 1 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}.$$

Geben Sie eine grobe Einschätzung zur Lage der Eigenwerte von A an, indem Sie die Gerschgorin Kreise angeben und zeichnen.

b) Es sei

$$B = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 1 & 1 & 1 & 2 & 0 \\ 0 & 1 & 17 & 3 & 1 & 6 & 0 \\ 0 & 1 & 3 & 15 & 1 & 5 & 0 \\ 0 & 1 & 1 & 1 & 10 & 1 & 0 \\ 0 & 2 & 6 & 5 & 1 & 24 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 60 \end{bmatrix}$$

Zeigen Sie mit Hilfe der Gerschgorin-Kreise, dass B invertierbar ist. Berechnen Sie außerdem die Konditionszahl exakt mit Hilfe der Gerschgorin-Kreise.

(4 Punkte)

Aufgabe 22. (Eigenpaare)

Betrachten Sie die Matrix $K = K(\alpha, \beta)$ mit

$$K(\alpha, \beta) = \begin{bmatrix} \alpha & \beta \\ \beta & 0 \end{bmatrix}.$$

a) Berechnen Sie die Eigenwerte von $K(\alpha, \beta)$ in Abhängigkeit von $\alpha, \beta \in \mathbb{R}$.

b) Geben Sie falls möglich Bedingungen für α und β an, unter denen

1) $K(\alpha, \beta)$ regulär ist
2) $K(\alpha, \beta)$ nur reelle Eigenwerte hat
3) $K(\alpha, \beta)$ einen reellen und einen komplexen Eigenwert hat

c) Es sei nun

$$M_{\alpha,\beta,\varepsilon} = \begin{bmatrix} \beta & 0 \\ \alpha & \beta \end{bmatrix} + \varepsilon \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

für $\varepsilon > 0, \alpha > 0$, wobei der Parameter ε i.d.R. klein ist und eine Störung der Daten darstellt. Geben Sie die Eigenwerte und Eigenvektoren von $M_{\alpha,\beta,\varepsilon}$ in Abhängigkeit von α, β und ε an.

(2 Punkte)

Aufgabe 23. (Quadratwurzel positiv semidefiniter Matrizen)

Wir betrachten eine Matrix $A \in \mathbb{K}^{n\times n}$ mit $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$. Eine Matrix $B \in \mathbb{K}^{n\times n}$ mit

$$A = B^2$$

nennt man (Quadrat-) Wurzel der Matrix A, falls sie existiert. Es seien $A, C \in \mathbb{K}^{n\times n}$ hermitesche Matrizen und A sei positiv (semi)definit. Zeigen Sie:

a) Alle Eigenwerte von hermiteschen Matrizen sind reell.

b) Es existiert eine Quadratwurzel B von A. Die Matrix B ist selbst positiv (semi)definit.

c) Alle Eigenwerte von AC sind reell, falls A und C kommutieren.

d) Es sei $AC = CA$ und es sei A positiv-definit mit den Eigenpaaren (λ_i, v_i), $i = 1, \ldots, n$, wobei $\lambda_i \neq \lambda_j$ für $i \neq j$. Zeigen Sie: Falls $v_i \notin \ker(C)$, dann ist v_i ebenfalls ein Eigenvektor von C, von AC und von CA.

e) Es gelten die Voraussetzungen aus d) und zusätzlich habe C vollen Rang. Finden Sie eine unitäre (für $\mathbb{K} = \mathbb{R}$) / orthogonale (für $\mathbb{K} = \mathbb{C}$) Matrix W, mit

$$WCAW^H = D = \begin{bmatrix} \eta_1 & & & \\ & \eta_2 & & \\ & & \ddots & \\ & & & \eta_n \end{bmatrix}$$

wobei $\eta_1 \geq \eta_2 \geq \ldots \geq \eta_n$ die Eigenwerte von AC sind.

(6 Punkte)

Programmieraufgabe 6. (Wurzel einer Matrix)

In Programmieraufgabe 5 haben Sie mit der Power-Iteration bereits eine einfache (aber nicht sehr leistungsfähige) Methode zur Berechnung von Eigenpaaren implementiert. In dieser Aufgabe geht es nun darum, die Wurzel einer Matrix $A \in \mathbb{R}^{n\times n}$ zu berechnen. Hierfür werden wir schematisch wie folgt vorgehen:

Als erstes, benötigen Sie einen funktionierenden Code zur Power-Iteration. Sie können Ihre eigene Implementierung nutzen, oder sich die Beispielimplementierung von der Webseite herunter laden und mittels `#import eigenpairs` in Ihrem Skript einbinden. Mit Hilfe dieser Routine berechnen wir alle Eigenpaare (λ_i, v_i) der Input-Matrix A.
Im Anschluß orthogonalisieren wir die Eigenvektoren mittels Gram-Schmidt Verfahren und erhalten nach Normierung eine Orthonormalbasis. Diese Orthonormalbasis stellt die Spalten einer Matrix W dar.
Nun ist also

$$A = WDW^T, \quad D \coloneqq \operatorname{diag}\{\lambda_1, \ldots, \lambda_n\}$$

und die gesuchte Wurzel B von A erfüllt

$$B^2 = A = WD^{\frac{1}{2}}D^{\frac{1}{2}}W^T = WD^{\frac{1}{2}}\underbrace{W^TW}_{=Id}D^{\frac{1}{2}}W^T = \left[WD^{\frac{1}{2}}W^T\right]^2,$$

mit $D^{\frac{1}{2}} = \operatorname{diag}\{\lambda_1^{\frac{1}{2}}, \ldots, \lambda_n^{\frac{1}{2}}\}$, d.h. $B = WD^{\frac{1}{2}}W^T$.

a) Binden Sie Ihren Code zur Power-Iteration (bzw. die Version von der Webseite) in Ihr Skript ein.

b) Implementieren Sie das Gram-Schmidt Verfahren. Normalisieren Sie die Vektoren vor der Rückgabe, um so eine Orthonormalbasis zu erhalten. Benutzen Sie die folgende Signatur für Ihre Funktion

```
def orthonormalize_gram_schmidt(vs):
    # INPUT:
    #       vs   Liste von Vektoren
    # OUTPUT:
    #       ws   Liste der orthonormalisierten Vektoren
```

c) Schreiben Sie eine Routine zur Berechnung der Wurzel einer Matrix. Rufen Sie hierzu sowohl die Power-Iteration mit Startvektor $v_0 = \begin{bmatrix}1 & \dots & 1\end{bmatrix}$, als auch die Gram-Schmidt Routine auf.

```
def compute_root_of_matrix(A):
    # INPUT:
    #       A    reelle spd Matrix
    # OUTPUT:
    #       B    Wurzel von A
```

d) Laden Sie sich die Matrix `matrix_root.txt` von der Webseite herunter und testen Sie Ihre Routine. Berechnen Sie hierzu den Fehler in der Frobeniusnorm

```
np.linalg.norm(A-np.dot(B,B),ord='fro')
```

(5 Punkte)

Abgabe per Mail an die Tutoren.