Institut für Analysis und Numerik
FB Mathematik und Informatik der Universität Münster
Tobias Leibner, Dr. Stephan Rave

06. September 2018

Übung zum Kompaktkurs

Einführung in die Programmierung zur Numerik mit Python

Sommersemester 2018 — Blatt 4

Aufgabe 1 (Potenz-Methode)

Schreiben Sie ein Programm, das für eine gegebene positiv definite Matrix $A \in \mathbb{R}^{n\times n}$ den größten Eigenwert $\sigma > 0$ mit zugehörigem Eigenvektor $r \in \mathbb{R}^n$ näherungsweise bestimmt. Verwenden Sie dazu die iterative *Potenz-Methode* zur Bestimmung des größten Eigenvektors:

- Wähle einen zufälligen Startvektor $0 \neq r^{(0)} \in \mathbb{R}^n$.
- Für $n > 0$ ist in jedem Iterationsschritt die neue Annäherung durch $r^{(n)} := \frac{Ar^{(n-1)}}{||Ar^{(n-1)}||}$ gegeben, wobei $||v|| := \sum_{i=1}^n v_i^2$ die euklidische Norm von $v \in \mathbb{R}^n$ sei.

Eine Approximation des zugehörigen Eigenwertes ist durch den *Rayleigh*-Quotienten $\sigma(A, r) := \frac{(r,Ar)}{(r,r)}$ gegeben.

Gehen Sie dazu in folgenden Schritten vor:

- Schreiben Sie eine Funktion `rayleigh_quotient(A, r)`, welche den Rayleigh-Quotienten $\sigma(A, r)$ berechnet und zurück gibt.
- Schreiben Sie eine Funktion `find_initial_value(A)`, welche zu A einen geeigneten Startwert $r^{(0)}$ findet und zurück gibt.
  Hinweis: Suchen Sie dazu im `numpy.random` Modul eine geeignete Funktion, die einen zufälligen Vektor passender Größe erstellt.
- Schreiben Sie eine Funktion `power_method(A, n)`, welches die Approximation des Eigenvektors $r^{(n)}$ iterativ bestimmt und in jedem Schritt den Rayleigh-Quotienten $\sigma(A, r^{(n)})$ ausgibt.
- Schreiben Sie eine Funktion `test(A, r, sigma)`, welche den Approximationsfehler $Ar - \sigma r$ ausgibt.

Testen Sie ihr Programm für die folgenden Matrizen mit $n = 20$ Iterationen.

$$B = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad C = \begin{pmatrix} 4 & -1 & -1 \\ 0.5 & 2 & -1 \\ 0 & 0 & 1 \end{pmatrix}$$

Aufgabe 2 (Gauß-Algorithmus)

Sei $A \in \mathbb{R}^{n \times n}$ eine reelle Matrix und $b \in \mathbb{R}^n$ ein Vektor. Gesucht ist der Lösungsvektor $x \in \mathbb{R}^n$, für den gilt:

$$Ax = b$$

Aus der linearen Algebra kennen Sie den Gauß-Algorithmus mit Spalten-Pivot-Suche als ein Lösungsverfahren für ein solches lineares Gleichungssystem. Beim Pivoting wird im k-ten Schritt das betragsmäßig größte Element in der k-ten Spalte auf und unterhalb der Diagonalen gesucht und die zugehörige Zeile anschließend mit der k-ten Zeile vertauscht.
Schreiben Sie ein Programm zur Realisierung des Gauß-Algorithmus. Verwenden Sie dabei `numpy.array` als grundlegende Datenstruktur für Vektoren als auch Matrizen. Gehen Sie in folgenden Schritten vor:

(a) Implementieren Sie eine Funktion, die das Pivoting durchführt, d.h. zu gegebenem k das Pivot-Element (und dessen Position) bestimmt.

(b) Schreiben Sie eine Funktion, die in einer gegebenen Matrix zwei Zeilen vertauscht.

(c) Implementieren Sie, aufbauend auf diesen zwei Funktionen, den Gauß-Algorithmus als Funktion, die zu gegebenem A, b das Gleichungssystem löst.

(d) Im Falle einer nicht invertierbaren Matrix soll Ihr Programm mit einer entsprechenden Fehlermeldung abbrechen. Überlegen Sie sich hierzu, woran man im Algorithmus diesen Fall erkennt.

Testen Sie das Verfahren an folgendem Gleichungssystem:

$$\begin{pmatrix} 2 & -1 & 5 & 7 \\ -1 & 1 & 1 & 4 \\ 1 & 2 & -3 & -2 \\ 3 & -1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 43 \\ 20 \\ -12 \\ 4 \end{pmatrix}$$