Einführung in die Grundlagen der Numerik

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

universitätbonn

Übungsblatt 6.

Abgabe: 26.11.2019 vor der Vorlesung

Aufgabe 15. (Numerische Eigenwertberechnung)

Es sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix. Des Weiteren sei $B \in \mathbb{R}^{n\times n}$ eine symmetrische, positiv-definite Matrix. Die Lösungen des generalisierten Eigenwertproblems

$$Ax = \lambda Bx, \tag{gEVP}$$

sind Paare (λ, x), bestehend aus einem Eigenwert $\lambda \in \mathbb{R}$ und einem Eigenvektor $x \in \mathbb{R}^n$.

a) Zeigen Sie, dass das generalisierte Eigenwertproblem (gEVP) als Standard-Eigenwertproblem $Cx = \lambda x$ geschrieben werden kann.

b) Das Arnoldi-Verfahren zur Lösung von $Cx = \lambda x$ entspricht einer Orthogonaltransformation $H_n = V_n^T C V_n$ von $C \in \mathbb{R}^{n\times n}$ in eine obere Hessenbergmatrix $H_m \in \mathbb{R}^{n\times n}$. Die Matrix V_n besteht aus einer Orthonormalbasis $\{q_0, \dots, q_{n-1}\}$ des Krylovraumes

$$K_n(C, q_0) = \{q_0, Cq_0, \dots, C^{n-1}q_0\}$$

zu einem beliebigen Startvektor $q_0 \in \mathbb{R}^n$. Zeigen Sie, welche Vereinfachungen sich aus einer Schiefsymmetrie von C, d.h. $C^T = -C$, für das Arnoldi Verfahren ergeben.

(6 Punkte)

Aufgabe 16. (Erweiterung des CG Verfahrens)

Das CG-Verfahren ist definiert für die Lösung von Gleichungssystemen $Ax = b$ mit symmetrischer, positiv-definiter Matrix $A \in \mathbb{R}^{n\times n}$. Wir zeigen nun, dass diese Voraussetzungen abgeschwächt werden können.

a) Angenommen $b \in \mathrm{Im}(A)$ und A ist symmetrisch, positiv-semidefinit. Zeigen Sie, dass die Suchrichtungen $q_0, \dots, q_{n-1}$ ebenfalls im Bild von A liegen. Folgern Sie daraus, dass für $Aq_j \neq 0$, $j = 0, \dots, n-1$ die Ungleichung $\langle Aq_j, q_j\rangle > 0$ gilt.

Aus a) folgt, dass der Nenner in der Berechnung von α_k und β_k nicht null ist, daher kann das Verfahren auch auf Systeme mit symmetrischer, positiv-semidefiniter Matrix angewendet werden.

b) Es sei nun $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix, die nicht notwendigerweise symmetrisch oder positiv-definit ist. Wie kann man in diesem Fall das CG-Verfahren Trotzdem anwenden?

(6 Punkte)

Programmieraufgabe 5. (Power-Iteration)

Es sei $A \in \mathbb{R}^{n \times n}$ eine diagonalisierbare Matrix und $v_0 \in \mathbb{R}^n$ ein Startvektor. Die Power-Iteration ist

$$v_{k+1} = \frac{Av_k}{\|Av_k\|}.$$

Wenn A einen betragsgrößten Eigenwert hat, so konvergiert die Power-Iteration gegen den zugehörigen Eigenvektor. Den betragsgrößten Eigenwert erhält man über die Iterationsvorschrift

$$\lambda_k = \frac{\langle v_k, Av_k \rangle}{\|v_k\|}.$$

a) Implementieren Sie die Power-Iteration. Die Berechnung kann abgebrochen werden, wenn $\|v_{k+1} - v_k\| < 1e-8$.

b) Testen Sie Ihre Methode an der Matrix `matrix_power.txt`, die Sie als Textdatei auf der Webseite finden.

Um auch die anderen Eigenpaare berechnen zu können, kann man auf so genannte 'Deflation'-Techniken zurückgreifen. Es seien im Folgenden $\lambda_1^A > \lambda_2^A > \ldots > \lambda_n^A$ die Eigenwerte von A und v der Eigenvektor zu λ_1^A. Ziel ist es, die Matrix so zu modifizieren, dass ein anderer Eigenwert betragsmäßig am größten wird. Eine Möglichkeit solch eine Modifikation vorzunehmen ist es, ein so genanntes 'Rang-1-Update' durchzuführen, d.h. konkret

$$B \coloneqq A - \lambda v v^T,$$

wobei v als Spaltenvektor aufzufassen ist. Die Matrix B hat die Eigenwerte $0, \lambda_2^A, \ldots, \lambda_n^A$, d.h. der ursprünglich betragsmäßig größte Eigenwert wurde durch 0 ersetzt und alle anderen Eigenwerte bleiben unangetastet.

c) Schreiben Sie eine Funktion, die ein Rang-1-Update durchführt

```
def rank_1_update(A,lam,ev):
    # Input:
    #       A       Reelle n x n Matrix
    #       lam     Eigenwert von A
    #       ev      Eigenvektor von A zum Eigenwert lam
    # Output:
    #       B       Matrix die durch Rang-1-Update von A entsteht
    #               B hat die gleichen Eigenwerte wie A, ausser dass
    #               der Eigenwert lam auf 0 gesetzt wurde

    # Ihr Code hier
```

d) Wir fügen die einzelnen Bauteile nun zusammen. Schreiben Sie eine Funktion, die die ersten m Eigenwerte einer Matrix A zu einem beliebigen Startvektor und mit einer beliebigen maixmalen Iterationszahl *max_it* berechnet

```
def compute_first_eigenpairs(A,v_0,m,max_it):
    # Input:
    #       A       Reelle n x n Matrix mit betragsmaessig strikt
    #               geordneten Eigenwerten
    #       v_0     Startvektor der Power-Iterationen
    #       m       Anzahl der Eigenpaare die berechnet werden
    #               (maximal len(A) Stueck)
    #       max_it  Maximale Iterationszahl pro Power-Iteration
    # Output:
    #       lams    Liste mit den Eigenwerten
    #       evs     Liste mit den Eigenvektoren

    # Hier Ihr Code
```

Sie müssen also in jedem Schritt das Eigenpaar zum betrasgsmäßig größten Eigenwert berechnen und anschließend das richtige Rang-1-Update durchführen.

a) Testen Sie Ihr Verfahren erneut mit der Matrix `matrix_power.txt`, die Sie als Textdatei auf der Webseite finden. Berechnen Sie alle Eigenwerte der Matrix.

(5 Punkte)

Abgabe per Mail an Ihren Tutor.