Fachbereich Mathematik & Informatik
Freie Universität Berlin
Prof. Dr. V. John (john@wias-berlin.de), H. Hardering (harderin@zedat.fu-berlin.de)

5. Übung zur Vorlesung

Numerik 1

SS 2012

Abgabe: Mi., 25.04.2012, 8:30 Uhr, am Beginn der Vorlesung

Es werden nur Lösungen bewertet, deren Lösungsweg klar erkennbar ist. Alle Aussagen sind zu begründen. Aus der Vorlesung bekannte Sachverhalte können vorausgesetzt werden.

Für die Programmieraufgaben sind ein lauffähiges Programm in Matlab, welches ohne weitere Eingaben auskommt, und alle dafür notwendigen Teilprogramme an den jeweiligen Tutor per Email zu schicken. Alle Programme müssen *sinnvoll* kommentiert werden. Zudem müssen Ausdrucke der etwaigen Ausgaben sowie der Programme selbst mit den Theorieaufgaben abgegeben werden.

1. Aufgabe (4 Theoriepunkte)

Eine Matrix $A \in \mathbb{R}^{n\times n}$ welche nur im oberen Dreieck sowie in der ersten unteren Diagonalen von Null verschiedene Einträge besitzt wird (obere) Hessenberg–Matrix genannt. Diese Matrizen kann man mit Givens–Drehungen effizient auf Dreiecksform bringen. Man bringe die obere Hessenberg–Matrix

$$\begin{pmatrix} 3 & 5 & 1 & 7 \\ 4 & 2 & 0 & -4 \\ 0 & 1 & 6 & 3 \\ 0 & 0 & 2 & -2 \end{pmatrix}$$

mittels Givens–Drehungen auf Dreiecksgestalt.

2. Aufgabe (4 Theoriepunkte)

Betrachten Sie die in der Vorlesung besprochenen Algorithmen zu *Givens-Rotation* und *Householder-Reflexion*. Verifizieren Sie folgende Abschätzungen bezüglich des Aufwandes unter der Voraussetzung, dass die Anzahl der zu approximierenden Punkte ungefähr mit der Anzahl der Fitting-Parameter übereinstimmt; also $m \approx n$.

a) Für die Givens-Rotation werden

$$\mathcal{O}(\tfrac{4}{3}n^3) \text{ Punktoperationen und } \mathcal{O}(\tfrac{1}{2}n^2) \text{ Quadratwurzeln}$$

benötigt.

b) Für die Householder-Reflexion werden

$$\mathcal{O}(\tfrac{2}{3}n^3) \text{ Punktoperationen}$$

benötigt.

3. Aufgabe (6 Programmierpunkte)
Schreiben Sie ein Programm `interpolation(x,fx,z)`, das das Interpolationspolynom einer Funktion $f : [a, b] \to \mathbb{R}$ an mehreren Stellen z_j auswertet. Dabei enthalten die Einträge von `x` die Stellen x_i und die von `fx` die Werte $f(x_i)$, an denen das Polynom mit f übereinstimmen soll. Die Einträge des Vektors `z` sind die Stellen z_j, an denen das Polynom ausgewertet werden soll. Ihre `matlab`-Funktion soll die Werte des Polynoms an diesen Stellen zurückgeben. Testen Sie Ihr Programm mit den Funktionen

$$\begin{aligned} f_1 &: [0, 2\pi] \to \mathbb{R}, & x &\mapsto \sin(x) \\ f_2 &: [0, 1] \to \mathbb{R}, & x &\mapsto x^{\frac{3}{2}} \\ f_3 &: [-1, 1] \to \mathbb{R}, & x &\mapsto \frac{1}{\gamma}\arctan(\gamma x), \gamma = 10. \end{aligned}$$

Gehen Sie dazu wie folgt vor:

a) Werten Sie für jedes f_i jeweils den Interpolationsfehler für $n = 5, 10, 15, 20, ..., 70, 75$ äquidistant verteilte Stützstellen näherungsweise durch

$$\|f_i - p_n\|_\infty = \max_{x\in[a,b]} |f_i(x) - p_n(x)| \approx \max_{x\in M} |f_i(x) - p_n(x)|$$

mit $M = \{x \in [a, b] : x = a + \frac{m}{10^4}(b - a), m \in \mathbb{N}\}$ aus. Plotten Sie die Näherung des Interpolationsfehlers für jedes f_i in Abhängigkeit vom Polynomgrad in einer geeigneten Skalierung (`plot, semilogx, semilogy, loglog`?).

b) Gehen Sie nun für f_1 und die Stützstellen $0 = x_0 < ... < x_{n-1} = \pi < x_n = 2\pi$ mit $n = 1, 2, 3, ..., 15$ analog zu Teil a) vor. Dabei sollen nur die ersten n Stützstellen äquidistant in $[0, \pi]$ verteilt sein. Plotten Sie für $n = 5, 10, 15$ das Interpolationspolynom im Vergleich zur Funktion f_1 indem Sie die Polynome und f_1 auf M auswerten. Im Plot sollen außerdem die gegebenen Stützstellen sichtbar sein.

c) Kommentieren Sie Ihre Ergebnisse. Wie erklären Sie sich das Verhalten des Fehlers für die Funktion f_1?