Karlsruher Institut für Technologie

Prof. Dr. Andreas Rieder
Dipl.-Math. techn. T. Kreutzmann
Institut für Angewandte und Numerische Mathematik
Fakultät für Mathematik

Numerische Mathematik 1
Wintersemester 2010/2011

6. Programmierübungsblatt vom 02. Februar 2011

Aufgabe 1: (Polynom-Interpolation)
Schreiben Sie ein Programm, dass die Funktion

$$f\colon [-1,1] \to \mathbb{R}, \quad x \mapsto \frac{1}{25x^2+1},$$

wahlweise in äquidistanten oder Tschebyscheff-Knoten mit einem Polynom vom Grad kleiner gleich n interpoliert.
Wählen Sie $n = 10$ und plotten Sie für beide Fälle den Funktionsverlauf.

Aufgabe 2: (Audiokompression)
Zur Aufnahme eines Audiosignals wird ein Schallsignal f in gewissen, meist gleichen Zeitabständen gemessen ("abgetastet") und dann abgespeichert. Seien $t_j = 2\pi j/n$, $j = 0, \dots, n-1$, die skalierten Abtastzeiten mit zugehörigen Amplitudenwerten $f_j = f\big(t_j\, t_{\max}/(2\pi)\big)$ und der Aufnahmedauer $t_{\max}$. Dann wird das Audiosignal durch das trigonometrische Polynom

$$\Psi(t) = \begin{cases} \dfrac{A_0}{2} + \displaystyle\sum_{j=1}^{(n-1)/2} \big(A_j \cos(jt) + B_j \sin(jt)\big), & \text{falls } n \text{ ungerade,} \\ \dfrac{A_0}{2} + \displaystyle\sum_{j=1}^{n/2-1} \big(A_j \cos(jt) + B_j \sin(jt)\big) + \dfrac{A_{n/2}}{2}\cos(nt/2), & \text{falls } n \text{ gerade,} \end{cases}$$

interpoliert, wobei die Koeffizienten durch

$$A_j = \frac{2}{n}\sum_{k=0}^{n-1} f_k \cos(kt_j) \quad \text{und} \quad B_j = \frac{2}{n}\sum_{k=0}^{n-1} f_k \sin(kt_j)$$

gegeben sind. Da hochfrequente Anteile des Signals vom menschlichen Gehör nur schwer wahrzunehmen sind, können diese zwecks Datenkompression weggelassen werden.

Laden Sie die Datei "handel.mat" in den Workspace (Sie können auch eine beliebige .wav-Datei mit dem Befehl `wavread` einlesen). Mit dem Befehl `sound` können Sie sich die Datei anhören. Berechnen Sie die diskrete Fourier-Transformierte des Audiosignals, d.h. die Koeffizienten A und B. Eliminieren Sie anschließend 30% der höchsten Frequenzen und führen eine inverse diskrete Fourier-Transformation aus, d.h. berechnen Sie Ψ ohne die hochfrequenten Anteile an den Stützstellen. Vergleichen Sie auditiv die Originaldatei mit der komprimierten Datei.

Anmerkung: Die Berechnung der Koeffizientenvektoren in obiger Form benötigt einen Aufwand von der Ordnung n^2. Daher wird dieser Schritt, ebenso wie die Inversion, in Ihrem Programm einige Minuten beanspruchen. Durch die schnelle diskrete Fourier-Transformation lässt sich der Aufwand auf die Ordnung $n \log n$ reduzieren. Sie können daher die MATLAB-Methoden `fft` und

`ifft` verwenden. Diese berechnen allerdings die im Allgemeinen komplexwertige diskrete Fourier-Transformierte

$$c_j = \sum_{k=0}^{n-1} f_k \mathrm{e}^{ikt_j} \quad \text{für } j = 0, \ldots, n-1$$

bzw. deren Inverse

$$\Phi(t) = \frac{1}{n} \sum_{j=0}^{n-1} c_j \mathrm{e}^{-ijt} \quad \text{für } t \in [0, 2\pi].$$

Durch die Zusammenhänge

$$c_j = \frac{n}{2}(A_j + iB_j), \quad c_{n-j} = \frac{n}{2}(A_j - iB_j), j = 0, \ldots, n-1$$

lassen sich die beiden Vorgehensweise jedoch ineinander überführen.

Die bearbeiteten Programmieraufgaben geben Sie in der Programmierbetreuung (mittwochs von 14:00 bis 17:00 Uhr im A-Pool des Rechenzentrums) ab, führen Ihr Programm vor und erläutern dies. Die Abgabe ist bis spätestens **Mittwoch, den 09. Februar 2011** möglich.