Praktische Mathematik II

Sommersemester 2005

Prof. Dr. Karl Scherer, Dr. Marcel Arndt

Aufgabenblatt 6

Ausgabe: 2.6.2005, Abgabe der Lösungen: 9.6.2005, 10:10
Abgabe der Programmieraufgabe: 20.6.-24.6.2005, genauer Termin nach Vereinbarung.

Aufgabe 22:
Auf dem Raum $C([a,b])$ definiere das Skalarprodukt

$$(f,g) = \int_a^b f(x)g(x)\omega(x)\mathrm{d}x$$

zu einem positiven Gewicht $\omega \in L^1((a,b))$. Definiere die Folge $(q_n)_{n\in\mathbb{N}}$ von Polynomen über die Dreitermrekursion

$$q_0(x) = 1, \qquad q_1(x) = x - a_1, \qquad q_n(x) = (x - a_n)q_{n-1}(x) - b_n q_{n-2}(x),$$

wobei

$$a_n = \frac{(xq_{n-1}, q_{n-1})}{(q_{n-1}, q_{n-1})} \qquad \text{und} \qquad b_n = \frac{(xq_{n-1}, q_{n-2})}{(q_{n-2}, q_{n-2})}.$$

Zeige, dass die Polynome orthogonal bzgl. des Skalarprodukts $(\cdot,\cdot)$ sind.[1] (10 Punkte)

Aufgabe 23:
Seien $q_n \in C([-1,1])$ die Polynome aus Aufgabe 22 bezüglich des Skalarprodukts

$$(f,g) = \int_{-1}^1 f(x)g(x)(1-x)^\alpha(1+x)^\beta\mathrm{d}x,$$

wobei $\alpha, \beta > -1$. Zeige, dass q_n bis auf einen Normierungsfaktor genau dem Jacobi-Polynom $P_n^{(\alpha,\beta)}$ entspricht, das durch die Rodriguessche Formel

$$(1-x)^\alpha(1+x)^\beta P_n^{(\alpha,\beta)}(x) = \frac{(-1)^n}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[(1-x)^{n+\alpha}(1+x)^{n+\beta}\right]$$

gegeben ist.
Hinweis: Weise die Orthogonalität der Jacobi-Polynome mittels partieller Integration nach und nutze diese Eigenschaft für die weitere Argumentation. (10 Punkte)

Aufgabe 24:
Es bezeichne $F_N f$ die diskrete Fouriertransformierte von f, wobei $N = 2^M$ und $M \in \mathbb{N}$. Zur Vereinfachung der Notation wird $F_N f$ per $(F_N f)_k = (F_N f)_{k+N}$ periodisch fortgesetzt.
a) Für $f \in \mathbb{C}^N$ gilt $(\overline{F_N f})_k = (F_N \overline{f})_{N-k}$, für $f \in \mathbb{R}^N$ folgt insbesondere $(\overline{F_N f})_k = (F_N f)_{N-k}$.
b) Für $f \in \mathbb{R}^N$ ist die direkte Berechnung von $F_N f$ ineffizient, da die verschwindenden Imaginärteile mitgeführt werden. Besser ist es, die Fouriertransformation für den Vektor

$$g \in \mathbb{C}^{N/2}, \qquad g_k = f_{2k} + i f_{2k+1}$$

der halben Länge durchzuführen. Zeige, dass man dann $F_N f$ über

$$(F_N f)_k = \frac{(F_{N/2}g)_k + (\overline{F_{N/2}g})_{N/2-k}}{4} + e^{2\pi i k/N}\frac{(F_{N/2}g)_k - (\overline{F_{N/2}g})_{N/2-k}}{4i}, \qquad k = 0, 1, \ldots, N-1$$

erhält. (Wegen a) genügt es sogar, nur die Hälfte von $(F_N f)$ abzuspeichern.) (10 Punkte)

[1]Bemerkung: Mit Hilfe der Gram-Schmidt-Orthogonalisierung lässt sich immer rekursiv eine Folge orthogonaler Polynome bestimmen, wobei für das Polynom q_n alle Polynome $q_0, q_1, \ldots, q_{n-1}$ benötigt werden. Die Besonderheit hier liegt darin, dass nur die letzten beiden Polynome q_{n-2} und q_{n-1} in die Berechnung eingehen.

Aufgabe 25:
Zu einem Vektor $f \in \mathbb{R}^N$ ist die diskrete Kosinustransformierte $C_N f$ durch

$$(C_N f)_k = \frac{\alpha_k}{N} \sum_{j=0}^{N-1} f_j \cos \frac{\pi \left(j + \frac{1}{2}\right) k}{N}, \qquad \alpha_k = \begin{cases} 1 & \text{für } k = 0 \\ 2 & \text{für } k = 1, \ldots, N-1 \end{cases}$$

definiert.
a) Zeige, dass die inverse diskrete Kosinustransformierte durch

$$(C_N^{-1} g)_j = \sum_{k=0}^{N-1} g_k \cos \frac{\pi \left(j + \frac{1}{2}\right) k}{N}$$

gegeben ist.
b) Führe die Berechnung der diskreten Kosinustransformierten auf die Berechnung der diskreten Fouriertransformierten eines geeignet gewählten Vektors $\tilde{f} \in \mathbb{R}^{2N}$ zurück. (10 Punkte)

Programmieraufgabe 3:
Schreibe ein C-, C++- oder Javaprogramm, das zu gegebenen Vektoren $f \in \mathbb{C}^N$ bzw. $g \in \mathbb{C}^N$ mit $N = 2^M$ und $M \in \mathbb{N}$ die diskrete Fouriertransformation (DFT)

$$g = F_N f \in \mathbb{C}^N \qquad \text{mit} \qquad g_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-2\pi i jk/N}, \qquad k = 0, 1, \ldots, N-1$$

und die inverse diskrete Fouriertransformation

$$f = F_N^{-1} g \in \mathbb{C}^N \qquad \text{mit} \qquad f_j = \sum_{k=0}^{N-1} g_k e^{2\pi i kj/N}, \qquad j = 0, 1, \ldots, N-1$$

mit Hilfe des FFT-Algorithmus berechnet.[2]
a) Bestimme die Fouriertransformation zu folgenden Funktionswerten:

$$f_j = \begin{cases} 0 & \text{falls } j < \frac{N}{2} \\ 1 & \text{falls } j \geq \frac{N}{2} \end{cases} \qquad \text{und} \qquad f_j = \frac{j(N-j)}{N^2}.$$

Verifiziere numerisch, dass $F_N^{-1} F_N f = f$ ist, also F_N und F_N^{-1} tatsächlich invers zueinander sind.
b) Entferne nun den hochfrequenten Anteil des Signals aus a) durch Setzen der Fourierkoeffizienten g_k für $k = \frac{1-c}{2} N, \ldots, \frac{1+c}{2} N$ auf null, wobei $c \in (0, 1)$ den auszufilternden Anteil bezeichnet.[3] Transformiere mit Hilfe der inversen Fouriertransformation zurück. Plotte die ursprüngliche Funktion und die so gefilterte Funktion und vergleiche für verschiedene Werte von c.
Bemerkung: Ein solcher Tiefpassfilter wird zum Beispiel zur Datenkompression beim JPEG-Format für Bilddateien verwendet. Dort wird lediglich die diskrete Kosinustransformation (DCT) anstelle der diskreten Fouriertransformation verwendet. (10 Punkte)

[2] Hinweis: Hierbei kann in jedem der $M = \log_2 N$ Schritte des FFT-Algorithmus ein Hilfsvektor zum Abspeichern der neu berechneten Teil-Fouriertransformationen verwendet werden. Alternativ kann mittels „bit reversal“ zuerst eine Umsortierung der Eingabedaten vorgenommen werden, um danach „in place“, also ohne Hilfsvektor, die Teil-Fouriertransformationen berechnen zu können.

[3] Die hochfrequenten Anteile entsprechen den mittleren Koeffizienten von g und nicht denen am Ende des Vektors, da letztere aufgrund der Periodizität im Frequenzraum wieder niederfrequent sind, lediglich mit negativer Frequenz.