Institut für Numerische und Angewandte Mathematik 04.12.2014
FB Mathematik und Informatik der Universität Münster
Prof. Dr. M. Ohlberger, Dipl.-Math. C. Himpe

Übung zur Vorlesung

Numerische Lineare Algebra

Wintersemester 2014/2015 — Blatt 8

11.12.2014

Aufgabe 1 (Gram-Schmidt Orthogonalisierung) (4 Punkte)
Eine Menge von linear unabhängigen Vektoren $\{u_1, ..., u_n\} \in \mathbb{R}^n$ lässt sich mit Hilfe des Gram-Schmidt Verfahrens in eine orthogonale Basis $\{v_1, ..., v_n\} \in \mathbb{R}^n$ überführen. Das Verfahren lautet wie folgt:

- $v_1 = u_1$
- $v_k = u_k - \sum_{j=1}^{k-1} \frac{<v_j, u_k>}{<v_j, v_j>} v_j$

Berechnen Sie eine orthogonale Basis zu$\{ \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 2 \end{pmatrix} \}$.

Aufgabe 2 (Relaxationsverfahren) (4 Punkte)
Sei $A = L+D+R \in \mathbb{R}^{n\times n}$ mit $D = I_n$, wobei I_n die Einheitsmatrix in $\mathbb{R}^{n\times n}$ bezeichnet. Sei T die zugehörige Iterationsmatrix des Gesamtschrittverfahrens zur Lösung von $Ax = b$. Die Eigenwerte λ_i der Matrix T seien reell und erfüllen $-1 < \lambda_1 \leq \ldots \leq \lambda_n < 1$. Für $\omega \in \mathbb{R}$ sei das Relaxationsverfahren durch die Iterationsmatrix $T(\omega) = (1-\omega)I_n + \omega T$ und die Iterationsvorschrift $x^{(k+1)} = T(\omega)x^{(k)} + D^{-1}b,\ x^{(0)} \in \mathbb{R}^n$ definiert.

a) Zeigen Sie, dass $T(\omega)$ die Eigenwerte $\mu_i = 1 - \omega + \omega\lambda_i$, $i = 1, \ldots, n$ besitzt.

b) Bestimmen Sie ω_0 so, dass der Spektralradius von $T(\omega_0)$ minimal wird.

c) Zeigen Sie, dass der Spektralradius von $T(\omega_0)$ für $\lambda_1 \neq -\lambda_n$ kleiner als der Spektralradius von $T(1) = T$ ist.

Aufgabe 3 (Vorkonditionierung von LGS) (4 Punkte)
Zu einem Gleichungssystem $Ax = b$ mit $A = (a_{ij})_{i,j=1}^n \in \mathbb{R}^{n\times n}, a_{ii} \neq 0, b \in \mathbb{R}^n$ besteht die *Zeilenäquilibrierung* darin, das System $CAx = Cb$ zu lösen, wobei $C = \mathrm{diag}(c_1, \ldots, c_n) \in \mathbb{R}^{n\times n}$ mit $c_i := \left(\sum_{k=1}^n |a_{ik}|\right)^{-1}$. Die *Diagonalvorkonditionierung* verwendet stattdessen die Matrix $C' = \mathrm{diag}(a_{11}^{-1}, \ldots, a_{nn}^{-1})$. Gegeben seien für $a \neq 0$ die folgende Matrix und ihre Inverse:

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 1 & a & a^2 \\ 1 & 2a & 4a^2 \end{pmatrix} \quad \text{mit} \quad A^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ -\frac{3}{2a} & \frac{2}{a} & -\frac{1}{2a} \\ \frac{1}{2a^2} & -\frac{1}{a^2} & \frac{1}{2a^2} \end{pmatrix}.$$

Berechnen Sie für $a > 4$ die Konditionen $\mathrm{cond}_\infty(A)$, $\mathrm{cond}_\infty(CA)$ und $\mathrm{cond}_\infty(C'A)$ (d.h. bzgl. der induzierten $\|\cdot\|_\infty$-Norm). Welches der Verfahren ist daher für große a zu bevorzugen?

Aufgabe 4 (Programmieraufgabe: cg-Verfahren) (4 Punkte)
Implementieren Sie das cg-Verfahren zur Lösung eines linearen Gleichungssystems

$$Ax = b$$

mit $A \in \mathbb{R}^{m\times m}$, $b \in \mathbb{R}^m$ gegeben und A positiv definit und symmetrisch.
Das cg-Verfahren zur Lösung von $Ax = b$ startet mit einem Vektor $z_1 \in \mathbb{R}^m$ und $t_1 = r_1 = b - Az_1$. Die Funktion zur Lösung des LGS sollte von der Form

$$cg(A, b, z_1, tol)$$

sein und neben der Näherungslösung auch die Anzahl der durchgeführten Iterationen zurückgeben. Der Parameter tol sollte für das Abbruchkriterium $< r_n, r_n > \leq tol$ verwendet werden.
Die **Hilbert Matrix** is ein notorisch schlecht konditioniertes Problem für $Ax = b$ mit

$$a_{ij} = (i + j - 1)^{-1},\ b_i = \sum_{j=1}^{m} (-1)^{j-1}(i + j - 1)^{-1},\ (i, j = 1, \dots, m)$$

und exakter Lösung $x_i = (-1)^{i-1}$ $i = 1, \dots, m$. Testen Sie das cg-Verfahren mit $m = 5$ und $m = 10$. Wählen Sie für tol zum einen $1e - 13$ und zum anderen $1e - 8$. Verwenden Sie als Startvektor den Nullvektor. Geben Sie jeweils den maximalen Fehler zur exakten Lösung sowie die Anzahl der Iterationen im cg-Verfahren an.

Aufgabe 5 (Bonus-Nikolausaufgabe) (4 Punkte)
In der folgenden Matrix versteckt sich eine saisonale Botschaft,

$$A = \begin{pmatrix} 0 & 0 & 11 & 9,7 & 9,9 \\ 0 & 8,7 & 10,1 & 10,5 & 10,4 \\ 0 & 0 & 0 & 0 & 11 \\ 7 & 11,4 & 11,1 & 10,4 & 10,1 \\ 0 & 0 & 0 & 10,4 & 11,6 \end{pmatrix},$$

Mit der Verkettung welcher sechs NumPy-Operation lässt diese sich extrahieren?