TU Chemnitz
Professur Numerische Mathematik
Prof. Dr. Oliver Ernst
Dipl.-Math. Ingolf Busch
Dipl.-Math. techn. Tommy Etling

5. Februar 2014

Numerische Mathematik

Sommersemester 2013

5. Übungsblatt

Aufgabe 19:

Ist $\hat{\boldsymbol{x}}$ eine Näherung an die Lösung $\boldsymbol{x}$ des linearen Gleichungssystems $A\boldsymbol{x} = \boldsymbol{b}$, $A \in \mathbb{C}^{n\times n}$ nichtsingulär, $\boldsymbol{b} \in \mathbb{C}^n$, so kann der Vorwärtsfehler $\Delta\boldsymbol{x} = \boldsymbol{x} - \tilde{\boldsymbol{x}}$ auch mit Hilfe des Residuums $\boldsymbol{r} = \boldsymbol{b} - A\tilde{\boldsymbol{x}}$ abgeschätzt werden. Man zeige:

(a) Es gilt $\|\Delta\boldsymbol{x}\| \leq \|A^{-1}\|\|\boldsymbol{r}\|$. (Was sind Vor- und Nachteile dieser oberen Schranke?)

(b) Zu $\boldsymbol{r} = \boldsymbol{b} - A\tilde{\boldsymbol{x}}$ existiert $\Delta A \in \mathbb{C}^{n\times n}$ so dass $\|\Delta A\|_2 = \|\boldsymbol{r}\|_2/\|\tilde{\boldsymbol{x}}\|_2$ und $(A + \Delta A)\tilde{\boldsymbol{x}} = \boldsymbol{b}$. Ferner gibt es keine Matrix ΔA mit kleinerer Norm, welche $(A+\Delta A)\tilde{\boldsymbol{x}} = \boldsymbol{b}$ erfüllt. (Dieser Satz gilt für jede Vektornorm mit einer verträglichen Matrixnorm.)

Aufgabe 20:

Schreiben Sie ein MATLAB-Programm, welches die LR-Zerlegung einer quadratischen Matrix berechnet. Versuchen Sie, möglichst effizienten MATLAB-Code zu schreiben, d.h. insbesondere for-Schleifen zu vermeiden. (Stattdessen können oft mehrere Operationen als Vektoroperationen zusammengefasst werden. Implementieren Sie dabei (mindestens) zwei verschiedene algorithmische Varianten der Gauß-Elimination. (Hier bietet sich eine Gegenüberstellung von zeilen- und spaltenorientierten Varianten an.)

Testen Sie Ihr Programm an Zufallsmatrizen verschiedener Größe. Führen Sie systematische Zeitmessungen aus. Plotten Sie die Laufzeiten der verschiedenen Varianten in Abhängigkeit von der Dimension n.

Hinweis: Für Zeitmessungen existieren in MATLABdie Befehle `tic` und `toc`.

Aufgabe 21:

Bei der Approximation einer stetigen Funktion $f : [0, 1] \to \mathbb{R}$ durch ein Polynom $p(x) = a_n x^n + \cdots + a_1 x + a_0$ werde der Approximationsfehler E in der L^2-Norm gemessen, d.h.

$$E^2 := \|p - f\|_{L^2}^2 = \int_0^1 [p(x) - f(x)]^2 \, dx.$$

(a) Zeigen Sie, dass die Minimierung des Fehlers $E = E(a_0, \ldots, a_n)$ auf ein lineares Gleichungssystem

$$H_n \boldsymbol{a} = \boldsymbol{b} \tag{1}$$

führt, wobei

$$\boldsymbol{b} = [b_0, \ldots, b_n]^\top \in \mathbb{R}^{n+1}, \qquad b_i = \int_0^1 f(x) x^i \, dx, \qquad i = 0, \ldots, n,$$

H_n die n-te Hilbert-Matrix mit

$$(H_n)_{i,j} = \frac{1}{i+j+1}, \quad i, j = 0, \ldots, n$$

und $\boldsymbol{a}$ der Vektor der Koeffizienten von p sind.

(b) Zeige Sie, dass H_n symmetrisch und positiv definit ist.

(c) Verifizieren Sie, dass die Komponenten der Inversen von H_n gegeben sind durch

$$(H_n^{-1})_{i,j} = (-1)^{i+j}(i+j+1)\binom{n+i}{n-j-1}\binom{n+j}{n-i-1}\binom{i+j}{i}^2$$

und dass H_n die Cholesky-Zerlegung $H_n = L_n L_n^T$ besitzt mit

$$(L_n)_{i,j} = \frac{\sqrt{2j+1}\,(i!)^2}{(i+j+1)!(i-j)!}, \qquad i \geq j.$$

Aufgabe 22:

Da Schranken für den Vorwärtsfehler oft die Norm von A^{-1} enthalten, benötigt man hinreichend genaue Schätzungen für diese Größe. Der Aufwand bei der Berechnung dieser Schätzer sollte neben dem für die Lösung des Gleichungssystems nicht zu sehr ins Gewicht fallen. (Insbesondere scheidet die Berechnung von A^{-1} und deren Norm aus, da hierbei neben den $2n^3/3$ Operationen für das Gleichungslösen zusätzlich noch $2n^3$ Operationen anfallen würden.) Man begnügt sich mit Fehlerschätzern, welche ausgehend von der bereits berechneten LR-Zerlegung $\mathscr{O}(n^2)$ Aufwand erfordern und eine Schätzung von $\|A^{-1}\|$ bis auf einen Faktor zehn genau liefern. Im folgenden betrachten wir solch einen Konditionszahl-Schätzer, welcher auf Hager zurückgeht und mit der 1-Norm arbeitet.

Die 1-Norm einer Matrix $B \in \mathbb{C}^{n\times n}$ is gegeben durch

$$\|B\|_1 = \max_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|B\boldsymbol{x}\|_1}{\|\boldsymbol{x}\|_1} = \max_j \sum_{i=1}^n |b_{i,j}| = \|B\boldsymbol{e}_k\|_1,$$

wobei $\boldsymbol{e}_k$ den k-ten Einheitsvektor bezeichnet und k der Index ist, für den das Maximum der Spaltensummen auftritt. Alle Einheitsvektoren für $B = A^{-1}$ durchzuprobieren läuft wieder auf die (zu teure) Berechnung von A^{-1} hinaus. Stattdessen wird ein *Gradienten-Aufstiegsverfahren* auf die konvexe Funktion

$$f(\boldsymbol{x}) = \|B\boldsymbol{x}\|_1 = \sum_{i=1}^n \left| \sum_{j=1}^n b_{i,j} x_j \right|, \quad \|\boldsymbol{x}\|_1 \leq 1, \tag{$\star$}$$

angewandt. Dabei wird $f(\boldsymbol{x})$ schrittweise maximiert, indem $\boldsymbol{x}$ in Richtung des steilsten Aufstiegs - also des Gradienten $\nabla f(\boldsymbol{x})$ - korrigiert wird. Unter der Annahme $\sum_j b_{i,j}x_j \neq 0$ (für alle i) ist der Gradient gegeben durch

$$\nabla f(\boldsymbol{x}) = \boldsymbol{z}^T B = (B^T \boldsymbol{z})^T, \quad z_i = \text{sign}(\sum_{j=1}^n b_{i,j}x_j), i = 1, \ldots, n.$$

Es ergibt sich folgender Algorithmus:

wähle $\boldsymbol{x}$ beliebig mit $\|\boldsymbol{x}\|_1 = 1$
repeat
  $\boldsymbol{w} = B\boldsymbol{x}$, $\boldsymbol{z} = \text{sign}(\boldsymbol{w})$, $\boldsymbol{y} = B^T\boldsymbol{z}$
  if $\|\boldsymbol{y}\|_\infty \leq \boldsymbol{y}^T\boldsymbol{x}$ **then**
    return $\|\boldsymbol{w}\|_1$
  else
    $\boldsymbol{x} = \boldsymbol{e}_j \operatorname{sign}(y_j)$ mit $|y_j| = \|\boldsymbol{y}\|_\infty$
  end if
until Maximale Iterationszahl erreicht

Zeigen Sie:

(a) Die Funktion f aus ($\star$) ist konvex.

(b) Terminiert der Algorithmus mit der Rückgabe von $\|\boldsymbol{w}\|_1$, so ist $\|\boldsymbol{w}\|_1 = \|B\boldsymbol{x}\|_1$ lokales Maximum von $f(\boldsymbol{x}) = \|B\boldsymbol{x}\|_1$.

(c) Andernfalls ist $\|B\boldsymbol{e}_j\|$ am Ende der Schleife größer als $\|B\boldsymbol{x}\|_1$ am Beginn der Schleife, d.h. der Algorithmus hat in diesem Schritt $f(\boldsymbol{x})$ noch vergrößern können.