Prof. Dr. Karsten Urban
Dipl.-Math. oec. Klaus Stolle
Institut für Numerische Mathematik
Universität Ulm

SoSe 2017
Blatt 03
05.05.2017

Angewandte Numerik 1

Besprechung in den Tutorien in der Woche vom 15.05.2017 bis 19.05.2017

Für dieses Übungsblatt gibt es 26 Theorie- und 24 Matlab-Punkte, sowie 18 Theorie- und 10 Matlab-Zusatzpunkte. Punkte, die mit einem * gekennzeichnet sind, sind Zusatzpunkte.
Die 60-Prozent-Grenzen liegen aktuell (inklusive Blatt 03) bei 42,6 Theoriepunkten und 28,2 Matlabpunkten.

Aufgabe 9 *(Aufwand zur Berechnung der Lösung eines linearen Gleichungssystems)* (2T+2T Punkte)

Nach Satz 3.1.3 im Skript beträgt der Aufwand zur Berechnung der Determinate $\det A$ einer Matrix $A \in \mathbb{R}^{n\times n}$ mit Hilfe der Leibnizschen Darstellung

$$\mathrm{FLOP}(\det A) = n\,n! - 1\,.$$

Da zur Berechnung der Lösung x^* eines linearen Gleichungssystems $Ax = b$ mit $A \in \mathbb{R}^{n\times n}$ und $b \in \mathbb{R}^n$ mit Hilfe der Cramerschen Regel die Berechnung von $n+1$ Determinanten und n Quotienten notwendig sind, ergibt sich für den Aufwand zur Berechnung der Lösung $x^* = A^{-1}b$ des linearen Gleichungssystems (Vergleiche Bemerkung 3.1.7)

$$\mathrm{FLOP}_{\mathrm{CramerLeibniz}}(A^{-1}b) = n\,(n+1)! - 1\,.$$

Der Aufwand zur Berechnung der Lösung $x^* = A^{-1}b$ mit Hilfe der Gaußschen Eliminationsmethode beträgt

$$\mathrm{FLOP}_{\mathrm{LR}}(A^{-1}b) = \frac{4n^3 + 9n^2 - n}{6}\,.$$

a) Der derzeit schnellste Supercomputer **Sunway TaihuLight** in China (Stand 11/2016) rechnet mit etwa 93,0146 PetaFLOP pro Sekunde (= Flops), d. h. rund $93,0146 \cdot 10^{15}$ flops. Berechnen Sie die Zeit (in Jahren), die dieser Supercomputer benötigen würde, um ein lineares Gleichungssystem der Dimension $n = 25$ mittels der Cramerschen Regel unter Verwendung der Leibnizschen Darstellung zu lösen.
Hinweis: Ein Jahr hat rund $365,242$ Tage.

b) Wie lange (in Sekunden) würde dieser Supercomputer (rein rechnerisch) brauchen, um das selbe Gleichungssystem mit Hilfe der Gaußschen Eliminationsmethode zu lösen?

Aufgabe 10 *(LR-Zerlegung durch Koeffizientenvergleich)* (4T+3T+3T Punkte)

In dieser Aufgabe wollen wir zu einer gegebenen Matrix $A \in \mathbb{R}^{3\times 3}$ die LR-Zerlegung durch Koeffizientenvergleich bestimmen. Gesucht sind also eine linke untere unipotente (siehe Definition 3.3.1) Dreiecksmatrix L und eine rechte obere Dreiecksmatrix R mit

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}, \quad L = \begin{pmatrix} 1 & 0 & 0 \\ l_{2,1} & 1 & 0 \\ l_{3,1} & l_{3,2} & 1 \end{pmatrix} \quad \text{und} \quad R = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ 0 & r_{2,2} & r_{2,3} \\ 0 & 0 & r_{3,3} \end{pmatrix},$$

so dass $A = LR$.

a) Wie und in welcher Reihenfolge kann man die Koeffizienten $l_{i,j}$ und $r_{i,j}$ bestimmen.

b) Verallgemeinern Sie dies auf 4×4 –Matrizen und 5×5 –Matrizen.

c) Kann man für jede reguläre Matrix A eine LR-Zerlegung finden, so dass gilt $A = LR$? Begründen Sie Ihre Antwort.

Aufgabe 11 *(Berechnung der LR-Zerlegung)* (5T*+4T*+4T* Punkte)

In dieser Aufgabe lösen wir mit Hilfe der LR-Zerlegung das lineare Gleichungssystem $Ax = b$ mit

$$A = \begin{pmatrix} 6 & -4 & 7 \\ -12 & 5 & -12 \\ 18 & 0 & 22 \end{pmatrix} \quad \text{und} \quad b = \begin{pmatrix} \frac{41}{12} \\ -\frac{22}{3} \\ \frac{29}{2} \end{pmatrix}.$$

a) Berechnen Sie die LR-Zerlegung der Matrix A durch Koeffizientenvergleich analog zu Aufgabe 10.

b) Lösen Sie damit durch Vorwärts- und Rückwärtseinsetzen das Gleichungssystem $Ax = b$.

c) Berechnen Sie die LR-Zerlegung der Matrix A nun mit Hilfe der Gaußschen Eliminationsmethode. Erhalten Sie die gleichen Matrizen L und R wie in Aufgabenteil a)? Werden die einzelnen Einträge der Matrizen L und R in der gleichen Reihenfolge berechnet wie im Aufgabenteil a)?

Verwenden Sie bei allen Rechnungen ausschließlich Brüche und keine Dezimalzahlen und geben Sie alle Zwischenschritte an.

Aufgabe 12 *(Programmieraufgabe: Berechnung der LR-Zerlegung)* (4M+2M+3M+2M Punkte)

In dieser Aufgabe implementieren wir die Gaußsche Eliminationsmethode zur Lösung eines linearen Gleichungssystems $Ax = b$.

a) Schreiben Sie eine Matlab-Funktion `[L, R] = gaussLR(A)`, die die LR-Zerlegung einer quadratischen Matrix `A` mit Hilfe der Gaußschen Eliminationsmethode berechnet.

b) Schreiben Sie ein Matlab-Skript `testGaussLR`, mit dem Sie Ihre Matlab-Funktion `[L, R] = gaussLR(A)` an verschiedenen Matrizen `A` testen. Testen Sie auch an der Matrix A aus Aufgabe 11.

c) Schreiben Sie eine Matlab-Funktion `x = solveLR(L, R, b)`, die die Lösung des linearen Gleichungssystems $LRx = b$ durch durch Vorwärts- und Rückwärtseinsetzen berechnet.

d) Schreiben Sie ein Matlab-Skript `testSolveLR`, mit dem Sie Ihre Matlab-Funktionen `[L, R] = gaussLR(A)` und `x = solveLR(L, R, b)` an verschiedenen linearen Gleichungssystemen $Ax = b$ testen. Überprüfen Sie Ihre Matlabfunktionen auch am linearen Gleichungssystem aus Aufgabe 11.

Aufgabe 13 *(LR-Zerlegung mit Spalten-Pivotisierung)* (6T+3T Punkte)

Gegeben seien

$$A = \begin{pmatrix} 2 & 3 & -1 & 0 \\ -6 & -5 & 0 & 2 \\ 2 & -5 & 6 & -6 \\ 4 & 6 & 2 & -3 \end{pmatrix} \quad \text{und} \quad b = \begin{pmatrix} 20 \\ -33 \\ -43 \\ 49 \end{pmatrix}.$$

a) Bestimmen Sie, falls existent, mit der Spaltenpivotstrategie die Matrizen P, L und R der Zerlegung $PA = LR$. Hierbei bezeichne P die Permutationsmatrix.

b) Lösen Sie mit Hilfe der Zerlegung $PA = LR$ das Gleichungssystem $Ax = b$.

Verwenden Sie bei allen Rechnungen ausschließlich Brüche und keine Dezimalzahlen und geben Sie alle Zwischenschritte an.

Aufgabe 14 *(Programmieraufgabe: LR-Zerlegung mit Spalten-Pivotisierung)*
(6M+1M+4M+(3T+2M) Punkte)

a) Erweitern Sie Ihre Matlab-Funktion `[L, R] = gaussLR(A)` aus Aufgabe 12 zu einer Matlab-Funktion `[L, R, P] = lrPivot(A)` zur Berechnung der LR-Zerlegung mit Spaltenpivotisierung. Ihre Funktion soll die Dreiecks-Matrizen L und R sowie die volle Permutationsmatrix P zurück geben.

b) Falls nötig schreiben Sie auch analog zu Ihrer Matlab-Funktion `x = solveLR(L, R, b)` aus Aufgabe 12 c) eine Matlab-Funktion `x = solveLrPivot(L, R, P, b)`, die zusammen mit Ihrer Funktion `[L, R, P] = lrPivot(A)` ein Gleichungssystem $Ax = b$ mit Spaltenpivotisierung löst.

c) Schreiben Sie ein Matlab-Skript `testSolve`, das die folgenden Gleichungssysteme $Ax = b$ numerisch jeweils ohne und mit Spaltenpivotisierung löst.

i)

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 1 \end{pmatrix},$$

ii)

$$A = \begin{pmatrix} 11 & 44 & 1 \\ 0.1 & 0.4 & 3 \\ 0 & 1 & -1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

iii)

$$A = \begin{pmatrix} 0.001 & 1 & 1 \\ -1 & 0.004 & 0.004 \\ -1000 & 0.004 & 0.000004 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

d) Vergleichen Sie ihre Ergebnisse hinsichtlich der Genauigkeit. Als Referenzlösung können Sie das Ergebnis von `A\b` betrachten.

Gibt es Unterschiede zwischen den Fällen „Gaußsches Eliminationsverfahren ohne Pivotisierung“ und „Gaußsches Eliminationsverfahren mit Spaltenpivotisierung“? Woran liegt das? Worin unterscheiden sich jeweils die Matrizen L und R in diesen Fällen?

Hinweis: Schreiben Sie sich nötigenfalls eine Matlab-Funktion `zeigeMatrix(A)`, mit der Sie diese Matrizen in einem geeigneten Format ausgeben können.

Aufgabe 15 *(Programmieraufgabe: Symmetrische Tridiagonalmatrizen)* (6T*+4M*+4M*+2M* Punkte)

In der Vorlesung haben wir gesehen, dass eine schwingende Saite auf dem Intervall $[0, 1]$ als Randwertproblem der Form

$$-u''(x) + \lambda(x)u(x) = f(x), \ x \in (0, 1) \quad \text{mit} \quad u(0) = 0 \quad \text{und} \quad u(1) = 0$$

modelliert werden kann. Gesucht ist dabei eine Funktion $u : [0, 1] \to \mathbb{R}, x \mapsto u(x)$, die die Bedingungen $u''(x)+\lambda(x)u(x) = f(x)$ für $x \in (0, 1)$ und $u(0) = 0$ sowie $u(1) = 0$ erfüllt. Dabei sind $\lambda : [0, 1] \to \mathbb{R}, x \mapsto \lambda(x)$ sowie die rechte Seite $f : [0, 1] \to \mathbb{R}, x \mapsto f(x)$ gegebene Funktionen.

Das vorgestellte numerische Verfahren zur Lösung von Randwertproblemen dieser Form teilt zunächst das Intervall $[0,1]$ in N ($N \in \mathbb{N}$) Teilintervalle $[x_{i-1}, x_i]$ ($i = 1, \ldots, N$) auf. Die $N+1$ Punkte x_i ($i = 0, \ldots, N$) können mit $x_i = ih$ äquidistant gewählt werden. Dabei ist $h := \frac{1}{N}$ die Länge der Teilintervalle und damit die Schrittweite.
Das numerische Verfahren berechnet nun durch Lösen des linearen Gleichungssystems

$$\begin{pmatrix} 2+h^2\lambda(x_1) & -1 & 0 & \ldots & 0 \\ -1 & 2+h^2\lambda(x_2) & -1 & \ldots & 0 \\ 0 & -1 & 2+h^2\lambda(x_3) & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 2+h^2\lambda(x_{N-1}) \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-1} \end{pmatrix} = h^2 \begin{pmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \\ \vdots \\ f(x_{N-1}) \end{pmatrix} \tag{1}$$

einen Vektor

$$u = \begin{pmatrix} u_1 \\ \vdots \\ u_{N-1} \end{pmatrix} \approx \begin{pmatrix} u(x_1) \\ \vdots \\ u(x_{N-1}) \end{pmatrix}$$

mit den Näherungswerten für die Funktion u an den Stellen $x_1, \ldots, x_{N-1}$. Beachte: Die Funktionswerte $u(x_0) = u(0) = 0$ und $u(x_N) = u(1) = 0$ sind vorgegeben.

a) Entwickeln Sie einen effizienten Algorithmus zur Lösung linearer Gleichungssysteme mit symmetrischen Tridiagonalmatrizen

$$\begin{pmatrix} a_{1,1} & a_{2,1} & 0 & \ldots & 0 \\ a_{2,1} & a_{2,2} & a_{3,2} & \ldots & 0 \\ 0 & a_{3,2} & a_{3,3} & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & a_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}. \tag{2}$$

Hinweis: Bringen Sie die Matrix auf Zeilenstufenform, so dass Sie ein modifiziertes Gleichungssystem der Form $Rx = L^{-1}b$ erhalten.

i) Wie viele FLOPs benötigen Sie mit Ihrem Algorithmus zur Lösung des Gleichungssystems?
ii) Welche Form hat R?
iii) Wie könnten Sie L aufstellen und welche Form hat L?

b) Schreiben Sie eine Matlab-Funktion `x = solveTD(d, nd, b)`, die ein Gleichungssystem der Form (2) löst. Dabei sollen `d` der Vektor der Diagonalelemente, $d = (a_{1,1}, a_{2,2}, \ldots, a_{n,n})$, und `nd` der Vektor der Elemente der beiden Nebendiagonalen, $nd = (a_{2,1}, a_{3,2}, \ldots, a_{n,n-1})$, sein. `b` ist die rechte Seite, $b = (b_1, b_2, \ldots, b_n)^T$. Rückgabewert soll die Lösung `x`, $x = (x_1, x_2, \ldots, x_n)^T$ des linearen Gleichungssystems sein. Belegen Sie in Ihrer Matlabfunktion abgesehen von einer Laufvariablen und einer skalaren Variablen zur Speicherung von Zwischenergebnissen keinen weiteren Speicherplatz.

c) Schreiben Sie ein Matlab-Skript `testSolveTD`, mit dem Sie Ihre Matlab-Funktion `x = solveTD(d, nd, b)` mit dem linearen Gleichungssystem (1) testen. Verwenden Sie $\lambda(x) \equiv 1$ und $f(x) \equiv 1$. Wählen Sie auch andere Werte für λ und f, beispielsweise $f(x) = 10\sin(x)$.

d) Plotten Sie die Näherungslösung u. Die Funktionswerte zwischen zwei Punkten x_i und x_{i+1} sollen dabei linear aus den Werten u_i und u_{i+1} interpoliert werden.

Hinweise:
Die Programmieraufgaben sind in Matlab zu erstellen. Der Source Code muss strukturiert und dokumentiert sein. Senden Sie **spätestens 24 Stunden vor Ihrem Tutorium** alle Matlab-Files und alle Ergebnisse in einer E-mail mit dem Betreff **Loesung-Blatt03** an **angewandte.numerik@uni-ulm.de**.