Dirk Praetorius
Michele Ruggeri

Wintersemester 2014/15
02.12.2014

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 10

Aufgabe 10.1. *Bubble-Sort* ist ein ineffizienter, aber kurzer Sortier-Algorithmus: Man vergleicht aufsteigend jedes Element eines Arrays x_j mit seinem Nachfolger x_{j+1} und - falls notwendig - vertauscht die beiden. Nach dem ersten Durchlauf muß zumindest das letzte Element bereits am richtigen Platz sein. Der nächste Durchlauf muß also nur noch bis zur vorletzten Stelle gehen, usw. Wie viele geschachtelte Schleifen braucht dieses Vorgehen? Schreiben Sie eine Funktion `bubblesort`, die ein gegebenes Array $x \in \mathbb{R}^n$ mittels Bubble-Sort aufsteigend sortiert, d.h. $x_1 \leq x_2 \leq \cdots \leq x_n$, und zurückgibt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, das den Vektor x einliest und in sortierter Reihenfolge ausgibt. Die Länge des Vektors soll eine Konstante im Hauptprogramm sein, die Funktion `bubblesort` ist für beliebige Länge n zu programmieren. Speichern Sie den Source-Code unter `bubblesort.c` in das Verzeichnis `serie10`.

Aufgabe 10.2. Eine obere Dreiecksmatrix $U \in \mathbb{R}^{n\times n}$ hat höchstens $\frac{n(n+1)}{2} = \sum_{j=1}^{n} j$ nicht-triviale Einträge. Schreiben Sie eine Struktur `matrixU`, in der neben der Dimension $n \in \mathbb{N}$ die Koeffizienten U_{ij} in einem dynamischen Vektor der Länge $\frac{n(n+1)}{2}$ gespeichert werden. Schreiben Sie die entsprechenden Zugriffsfunktionen (`newMatrixU`, `delMatrixU`, `getMatrixUDimension`, `getMatrixUij`, `setMatrixUij`), und überlegen Sie sich zuvor, an welcher Stelle u_ℓ im dynamischen Vektor ein Eintrag U_{ij} gespeichert werden soll. Tipp: Speichern Sie U spaltenweise.

Aufgabe 10.3. Schreiben Sie eine Funktion `mvmU`, die die Matrix-Vektor-Multiplikation mit einer oberen Dreiecksmatrix $U \in \mathbb{R}^{n\times n}$ realisiert. Die Matrix U sei dabei in der Struktur aus Aufgabe 10.2 gespeichert, der Vektor in der Struktur aus der Vorlesung. Überflüssige Multiplikationen mit Nulleinträgen der Matrix U sollen aus Effizienzgründen vermieden werden.

Aufgabe 10.4. Es sei $U \in \mathbb{R}^{n\times n}$ eine obere Dreiecksmatrix mit $U_{jj} \neq 0$ für alle $j = 1, \ldots, n$. Zu gegebener rechter Seite $b \in \mathbb{R}^n$ existiert dann ein eindeutiger Vektor $x \in \mathbb{R}^n$ mit $Ux = b$. Leiten Sie, ausgehend von der Formel für die Matrix-Vektor-Multiplikation, eine Formel für x her (Hinweis: Schreiben Sie sich dazu das Matrix-Vektor-Produkt $b = Ux$ komponentenweise für b_j mit $j = 1, \ldots, n$ als Summe, und überlegen Sie, wie die spezielle Gestalt von U die Laufindizes der Summe vereinfacht). Schreiben Sie eine Funktion `solveU`, die für gegebenes U und b den Vektor x berechnet und zurückgibt. Die Matrix U soll dabei in der Struktur aus Aufgabe 10.2 gespeichert werden, die Vektoren $b, x \in \mathbb{R}^n$ in der Struktur aus der Vorlesung. Testen Sie Ihren Code, indem Sie im Hauptprogramm die Matrix U und den Vektor x einlesen und das Matrix-Vektor-Produkt $b = Ux$ mit Hilfe der Funktion aus Aufgabe 10.3 berechnen. Löst man nun mit dem berechneten b das Gleichungssystem $U\widetilde{x} = b$, muss die Lösung $\widetilde{x}$ mit dem ursprünglichen x übereinstimmen.

Aufgabe 10.5. Nicht jede Matrix $A \in \mathbb{R}^{n\times n}$ hat eine normalisierte LU-Zerlegung $A = LU$, d.h.

$$\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ \ell_{21} & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \ell_{n1} & \ldots & \ell_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \ldots & u_{1n} \\ 0 & u_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & u_{n-1,n} \\ 0 & \ldots & 0 & u_{nn} \end{pmatrix}.$$

Wenn aber A eine normalisierte LU-Zerlegung besitzt, so gilt

$$u_{ik} = a_{ik} - \sum_{j=1}^{i-1} \ell_{ij} u_{jk} \quad \text{für } i = 1, \ldots, n, \quad k = i, \ldots, n,$$

$$\ell_{ki} = \frac{1}{u_{ii}} \left(a_{ki} - \sum_{j=1}^{i-1} \ell_{kj} u_{ji} \right) \quad \text{für } i = 1, \ldots, n, \quad k = i+1, \ldots, n,$$

$$\ell_{ii} = 1 \quad \text{für } i = 1, \ldots, n,$$

wie man leicht über die Formel für die Matrix-Matrix-Multiplikation zeigen kann. Alle übrigen Einträge von $L, U \in \mathbb{R}^{n \times n}$ sind Null. Schreiben Sie eine Funktion `computeLU`, die die LU-Zerlegung von A berechnet und zurückgibt. Dazu überlege man, in welcher Reihenfolge man die Einträge von L und U berechnen muss, damit die angegebenen Formeln wohldefiniert sind (d.h. alles was benötigt wird, ist bereits zuvor berechnet worden). Speichern Sie den Source-Code unter `computeLU.c` in das Verzeichnis `serie10`.

Aufgabe 10.6. Die Matrix $A \in \mathbb{R}^{n \times n}$ sei eine Tridiagonalmatrix, d.h.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & & & \\ a_{2,1} & a_{2,2} & a_{2,3} & & \\ & a_{3,2} & a_{3,3} & \ddots & \\ & & \ddots & \ddots & a_{n-1,n} \\ & & & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

wobei alle anderen Einträge von A Null sind, und besitze eine LU-Zerlegung. Mit Hilfe der Formeln in Aufgabe 10.5 überlege man sich Formeln für die Einträge von L und U in diesem speziellen Fall. Dann schreibe man eine Funktion `computeLU3` bei der keine unnötigen Rechenoperationen (d.h. Additionen/Multiplikationen von Null) durchgeführt werden und nur Einträge von L und U berechnet werden, die nicht Null sind.

Aufgabe 10.7. Der Laplacesche Entwicklungssatz für Determinanten besagt, dass für ein beliebiges $j \in \{1, \ldots, n\}$ gilt, dass

$$\det A = \sum_{i=1}^{n} (-1)^{i+j} \cdot a_{ij} \cdot \det A_{ij} \tag{1}$$

wobei A_{ij} die $(n-1) \times (n-1)$-Untermatrix von A ist, die durch Streichen der i-ten Zeile und j-ten Spalte entsteht. Schreiben Sie eine Funktion `detlaplace`, die die Determinante $\det(A)$ einer Matrix $A \in \mathbb{R}^{n \times n}$ mit Hilfe des Laplaceschen Entwicklungssatzes berechnet. Speichern Sie den Source-Code unter `detlaplace.c` in das Verzeichnis `serie10`.

Aufgabe 10.8. Um die Determinante einer Matrix $A \in \mathbb{R}^{n \times n}$ zu berechnen, ist der Laplacesche Entwicklungssatz aus Aufgabe 10.7 kein geeignetes Mittel (Sie sind eingeladen, dies praktisch zu erfahren!). Es ist besser, man berechnet die normalisierte LU-Zerlegung aus Aufgabe 10.5. Es gilt nämlich $\det(A) = \det(L) \det(U) = \det(U) = \prod_{j=1}^{n} u_{jj}$. Schreiben Sie eine Funktion `det(A)`, die die Determinante einer Matrix A mittels der normalisierten LU-Zerlegung berechnet.