

Übungen zur Vorlesung Einführung in das Programmieren für TM

Serie 9

Die Aufgaben mit Stern (*) sind bis zur Übung in der kommenden Woche vorzubereiten. Kopieren Sie die Source-Codes vor der Übung auf Ihren Account auf der `lva.student.tuwien.ac.at` und überprüfen Sie, ob diese von `matlab` auf der `lva.student.tuwien.ac.at` interpretiert werden können. In den folgenden Aufgaben sollen **Arithmetik**, **Verzweigungen** und **Zählschleifen** geübt werden.

Aufgabe 9.1*. Was macht die folgende Matlab-Funktion?

```
function [y,j] = f(x)
j = 1;
absy = abs(x(1));
for k = 2:length(x)
    absx = abs(x(k));
    if absx > absy
        absy = absx;
        j = k;
    end
end
y = x(j);
```

Welchen Wert haben die Variablen `j`, `k`, `absx`, `absy` und `x` jeweils vor dem `end` in der vorletzten Zeile, wenn man die Funktion mittels

```
[y,j] = f([1,-3,3,2,4,-4])
```

aufruft? Schreiben Sie einen alternativen Source-Code, der anstelle der Schleife geeignete Matlab-Funktionen verwendet.

Aufgabe 9.2*. Schreiben Sie eine Funktion `maxcount`, die von einem Vektor $x \in \mathbb{R}^n$ das Maximum zurückliefert und die Anzahl, wie oft dieses im Vektor vorkommt. Realisieren Sie die Funktion mit und ohne Schleifen. Speichern Sie die Source-Codes im Verzeichnis `serie09` unter den Namen `maxcount.m` bzw. `maxcountfor.m`.

Aufgabe 9.3*. Gegeben sei eine obere Dreiecksmatrix $U \in \mathbb{R}^{n \times n}$ mit $u_{jj} \neq 0$ für alle $j = 1, \dots, n$. Für gegebenes $b \in \mathbb{R}^n$ existiert dann eine eindeutige Lösung $x \in \mathbb{R}^n$ von $Ux = b$. Schreiben Sie eine Funktion `solveU`, die die Lösung berechnet. Dabei sollen lediglich Schleifen und elementare Matlab-Arithmetik verwendet werden. Versuchen Sie, möglichst viele Schleifen über die Vektor-Arithmetik zu realisieren. Speichern Sie den Source-Code im Verzeichnis `serie09` unter `solveU.m`.

Aufgabe 9.4*. Gegeben seien eine Matrix $A \in \mathbb{R}^{n \times n}$ und eine rechte Seite $b \in \mathbb{R}^n$. Lösen Sie das Gleichungssystem $Ax = b$ mit dem *Gauß'schen Eliminationsverfahren*. Dies ist gerade das Vorgehen, wenn man ein lineares Gleichungssystem händisch löst:

- Zunächst bringt man die Matrix A auf obere Dreiecksform, in dem man die Unbekannten eliminiert. Gleichzeitig modifiziert man die rechte Seite b .
- Das entstandene Gleichungssystem mit oberer Dreiecksmatrix A löst man mit Aufgabe 9.3.

Im ersten Eliminationsschritt zieht man geeignete Vielfache der ersten Zeile von den übrigen Zeilen ab und erhält dadurch eine Matrix der Form

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ 0 & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2} & \dots & a_{nn} \end{pmatrix},$$

Im zweiten Eliminationsschritt zieht man nun geeignete Vielfache der zweiten Zeile von den übrigen Zeilen ab und erhält eine Matrix der Form

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22} & a_{23} & \dots & a_{2n} \\ 0 & 0 & a_{33} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & a_{n3} & \dots & a_{nn} \end{pmatrix},$$

Nach $n - 1$ Eliminationsschritten erhält man also eine obere Dreiecksmatrix A . Berücksichtigen Sie, dass auch die rechte Seite $b \in \mathbb{R}^n$ geeignet modifiziert werden muss und machen Sie sich das Vorgehen zunächst an einem Beispiel mit $A \in \mathbb{R}^{2 \times 2}$ sowie $A \in \mathbb{R}^{3 \times 3}$ klar. Schreiben Sie eine Matlab-Funktion `gauss`, die die Lösung von $Ax = b$ berechnet. Speichern Sie den Source-Code im Verzeichnis `serie09` unter `gauss.m`.

Aufgabe 9.5. Das Gauß'sche Eliminationsverfahren scheitert, falls im k -ten Schritt $a_{kk} = 0$ gilt, auch wenn das Gleichungssystem $Ax = b$ eine eindeutige Lösung x besitzt. Deshalb kann man das Verfahren um eine sogenannte *Pivot-Suche* erweitern:

- Im k -ten Schritt wählt man aus a_{kk}, \dots, a_{nk} das betragsgrößte Element a_{pk} .
- Dann vertauscht man die k -te und die p -te Zeile von A (und b).
- Schließlich führt man den Eliminationsschritt aus wie zuvor.

Schreiben Sie eine Matlab-Funktion `gausspivot`, die die Lösung von $Ax = b$ wie angegeben berechnet. (Man kann übrigens mathematisch beweisen, dass das Gauss-Verfahren mit Pivot-Suche genau dann durchführbar ist, wenn das Gleichungssystem $Ax = b$ eine eindeutige Lösung besitzt. Einen Beweis dazu sehen Sie in der Vorlesung zur Numerischen Mathematik.)

Aufgabe 9.6. Wenn man im Gauß-Verfahren mit Pivot-Suche die Zeilen im Eliminationsschritt *wirklich* vertauscht (d.h. Speicher kopiert), führt dies auf unnötig viele Operationen und entsprechend lange Laufzeit des Programms. Es empfiehlt sich daher, die Vertauschung nur *virtuell* durchzuführen: Man startet mit einem Buchhaltervektor $\pi = (1, \dots, n)$. Im Vertauschungsschritt vertauscht man lediglich $\pi(p)$ mit $\pi(k)$. Im Source-Code (sowohl zu Aufgabe 9.3 als auch zu Aufgabe 9.4) sind jetzt die Zeilenindizes, d.h. der erste Index von a_{jk} sowie der Index von b_j geeignet zu modifizieren.

Aufgabe 9.7. Schreiben Sie eine Funktion `spaltennorm`, die die Spaltensummennorm

$$\|A\| = \max_{k=1, \dots, n} \sum_{j=1}^m |A_{jk}|$$

einer Matrix $A \in \mathbb{R}^{m \times n}$ berechnet. Realisieren Sie die Funktion mit und ohne Schleifen.

Aufgabe 9.8. Nicht jede Matrix $A \in \mathbb{R}^{n \times n}$ hat eine normalisierte LU-Zerlegung $A = LU$, d.h.

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

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

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

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

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

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. Man schreibe 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). Verwenden Sie bei der Implementierung lediglich die MATLAB-Arithmetik sowie geeignete Schleifen. Sie können ihren Funktion in MATLAB mittels `lu` verifizieren.

Aufgabe 9.9. Man schreibe eine Funktion `solveLU`, die die Lösung x des linearen Gleichungssystem $Ax = b$ berechnet. Dabei soll folgende Lösungsstrategie verwendet werden:

- (1) Berechne die LU-Zerlegung von A .
- (2) Löse $Ly = b$ nach y .
- (3) Löse $Ux = y$ nach x .

Es gilt dann nämlich $Ax = LUx = Ly = b$. Verwenden Sie bei der Implementierung lediglich die MATLAB-Arithmetik sowie geeignete Schleifen. Sie können Ihre Funktion in MATLAB mit Hilfe des Backslash-Operators `A\b` oder mittels `linsolve` verifizieren.

Aufgabe 9.10. Man kann den Speicheraufwand in Aufgabe 9.9 minimieren, indem man die Einträge der Matrix A geeignet durch die Einträge der Matrizen L und U überschreibt. Ferner kann man den Vektor b bei geeignetem Vorgehen, zunächst durch y und schließlich durch x überschreiben. Dadurch wird insgesamt kein zusätzlicher Speicher benötigt. Realisieren Sie dieses Vorgehen.