Abteilung für
Angewandte Mathematik

WiSe 2023/2024
Dr. Alexei Gazca, Tatjana Stiefken

Praktische Übungen zu Numerik I

Projekt 2 – 30.10.2023

Abgabe: per E-Mail bis Freitag, den 10.11.2022, 10:00 Uhr

Homepage zur Vorlesung:

`https://aam.uni-freiburg.de/agru/lehre/ws23/num/index.html`

Projekt 1 (8 Punkte)**.** Schreiben Sie ein Programm mit Funktionen `solve_upper` und `solve_lower` zur Lösung linearer Gleichungssysteme mit regulärer oberer beziehungsweise unterer Dreiecksmatrix. Die Lösungen sind mit rückwärts beziehungsweise vorwärts laufenden Schleifen gegeben durch

$$x_j = \left(b_j - \sum_{k=j+1}^{n} u_{jk} x_k \right) / u_{jj}, \qquad x_j = \left(b_j - \sum_{k=1}^{j-1} \ell_{jk} x_k \right) / \ell_{jj},$$

wobei die leere Summe den Wert Null habe. Testen Sie die Routinen für die Gleichungssysteme $A_l x = b_l$ mit

$$A_1 = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 4 & 5 \\ 0 & 0 & 6 \end{bmatrix}, \quad b_1 = \begin{bmatrix} 6 \\ 9 \\ 6 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 3 & 0 \\ 4 & 5 & 6 \end{bmatrix}, \quad b_2 = \begin{bmatrix} 3 \\ 12 \\ 28 \end{bmatrix}.$$

Projekt 2 (4 Punkte)**.** Schreiben Sie ein Programm, das für eine LU-zerlegbare Matrix $A \in \mathbb{R}^{n \times n}$ ihre LU-Zerlegung bestimmt. Unter welchen Umständen sollte man die Berechnung von L abbrechen?
Lösen Sie mit Hilfe ihres Programms die Gleichungssysteme $A_i x = b_i$ für

$$A_1 = \begin{bmatrix} 4 & 2 & 3 \\ 2 & 4 & 2 \\ 3 & 2 & 4 \end{bmatrix}, \quad b_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 2 & -1 & 0 & & \cdots & 0 \\ -1 & 2 & -1 & & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & & 0 \\ \vdots & & & -1 & 2 & -1 \\ 0 & & \cdots & 0 & -1 & 2 \end{bmatrix}, \quad b_2 = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}.$$

Hierbei sei $A_2 \in \mathbb{R}^{n \times n}$ und $b_2 \in \mathbb{R}^n$.Verdoppeln Sie sukzessive die Dimension des Problems und beobachten Sie die Laufzeit des Programms. Überprüfen Sie Ihre Ergebnisse mit Hilfe der Matlab-Befehle `[L,U] = lu(A)` und `x = A\b`. In Python kann `scipy` sehr nützlich sein:

```
from scipy import linalg
_,L,U = linalg.lu(A)
x = linalg.solve(A,b)
```

Hinweis: Mit den Befehlen **tic** und **t = toc** können Sie in Matlab die zwischen **tic** und **toc** vergangene Zeit messen. In Python können Sie `import time; start = time.time(); "code"; end = time.time()` verwenden.