Dirk Praetorius
Winfried Auzinger

Sommersemester 2018
14.03.2018

Übungen zur Vorlesung
Computermathematik

Serie 1

Aufgabe 1.1. MATLAB provides numerous useful functions. For each function, you get a detailed description by typing `help funktionsname` resp. `doc funktionsname`. Check the functionality of `diag`, `find`, and `sum`. What are possible parameters? What are possible outputs? Prepare simple examples which explain the function and implement them in an easy MATLAB-script.

Aufgabe 1.2. Let $A \in \mathbb{R}^{m_A \times n_A}$ and $B \in \mathbb{R}^{m_B \times n_B}$ be given matrices. Write a MATLAB-script which generates a block diagonal matrix C of the form

$$C = \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}.$$

Here, the 0-entries are 0-matrices of appropriate dimension. Avoid loops, and use only appropriate vector/matrix functions and indexing instead.

Aufgabe 1.3. Write a MATLAB-script which computes the expected value and the distribution function of the function `rand()`. First, use `rand()` to generate a vector of random numbers $x \in (0,1)^N$ and compute the expected value via $\mathbb{E} = \frac{1}{N}\sum_{i=1}^{N} x_i$. To get the distribution function, consider the intervals $[0, 0.1], [0, 0.2], \dots, [0, 1]$ and compute the numbers $A_a := \#\{x_i : x_i \in [0, a]\}$ for all $a = \{0.1, \dots, 1\}$. Then, the probabilty for a random number y to satisfy $y \leq a$ is given by A_a/N.
How should N be chosen in order to get a reliable estimate for the expected value. What is the probability distribution of the function `rand()`? Avoid loops! Instead, use matrix functions and matrix indexing!

Aufgabe 1.4. Schreiben Sie ein MATLAB-Skript, das für einen Vektor $x \in \mathbb{R}^n$ das Maximum $x_{\max} := \max_{1 \leq i \leq n} x_i$ bestimmt. Anschließend soll jeder Eintrag x_i mit $|x_i| \geq x_{\max}$ duch $\operatorname{sign}(x_i) x_{\max}$ ersetzt werden. Vermeiden Sie Schleifen, und verwenden Sie geeignete Matrix/Vektor-Funktionen und Indizierung an deren Stelle.

Aufgabe 1.5. Write a MATLAB-script which displays for a given vector $x \in \mathbb{R}^N$ and a given $n \leq N$ the trimmed vector y where all entries $x_j \leq x_{j-n}$ with $j - n > 0$ are cut out of x. For $j - n \leq 0$, the entry x_j is cut if $x_j \leq x_{N-n+j}$. For example, for $x = (1, 6, 5, -7, 3, 2) \in \mathbb{R}^6$ and $n = 3$ the trimmed vector is $y = (1, 6, 5) \in \mathbb{R}^3$. Avoid loops! Instead, use matrix functions and matrix/vector indexing!

Aufgabe 1.6. Write a MATLAB-script which generates, for odd $n \geq 5$ and three digits $m \in \{0, \dots, 9\}$ of your choice, matrices $A \in \mathbb{R}^{n \times n}$ with structure corresponding to the digit m. For $n = 5$ and $m = 0, 1, 2$ the matrices have the following form:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & & & & 1 \\ 1 & & & & 1 \\ 1 & & & & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad \begin{pmatrix} & & 1 & & \\ & & 1 & & \\ & & 1 & & \\ & & 1 & & \\ & & 1 & & \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ & & & & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & & & & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

where all entries which are not shown have to be initialized with 0. Avoid loops, and use only appropriate vector/matrix functions and indexing instead.

Aufgabe 1.7. Write a MATLAB script which, for given dimension n, returns the matrix $A \in \mathbb{R}^{n \times n}$ with ones on the diagonal and anti-diagonal, while all other entries are zero.

For $n = 5$, this matrix reads as

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Avoid loops, and use only appropriate vector/matrix functions and indexing instead.

Aufgabe 1.8. Write a MATLAB-script which generates, for given $n \in \mathbb{N}$, the following block diagonal matrix $A \in \mathbb{R}^{2n \times 2n}$.

$$A := \begin{pmatrix} 1 & 1 & & & & & \\ 1 & 1 & & & & & \\ & & 1 & 1 & & & \\ & & 1 & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & 1 \\ & & & & & 1 & 1 \end{pmatrix}$$

All entries which are not shown have to be initialized with 0. Avoid loops, and use only appropriate vector/matrix functions and indexing instead.