Kapitel 10

Übungsbeispiele

Achtung: Der Inhalt der Übungen kann sich im Laufe des Semesters ändern. Überprüfen sie immer vor Beginn der Übung den jeweiligen Inhalt.

Die Programme zu den einzelnen Übungen werden mit Hilfe eines bereitgestellten Scripts abgegeben. Sie werden direkt an einem für sie vorbestimmten Platz gespeichert, wo sie beurteilt werden können.

Der Name des Scripts ist:

- `uebungsabgabe file1 file2 ...`
- `uebungsabgabe` allein zeigt Ihnen die Liste der schon abgegebenen Files.
- `uebungsstatus` zeigt Ihnen den jeweiligen Status (abgegeben, korrigiert, mangelhaft, ...) an.

In MATLAB können Sie die Skripts mit vorangestellten Rufzeichen aufrufen. Falls Sie einen Fehler vermuten, können Sie Ihr Beispiel auch ändern und wieder übermitteln.

10.1 Funktionen, Input, Output

Ziel: Ein erster Einstieg in Matlab unter Linux soll geschafft werden. Editieren, Speichern und Ausführen von MATLAB-skripten. Aufruf einfacher eingebauter Funktionen, Ein- und Ausgabe.

Vorraussetzung: Grundlagen von Linux, Gnome und Matlab wie in der ersten Vorlesungsstunde besprochen.

10.1.1 Eine Formel

Schreiben Sie ein MATLAB-skript `lnexp.m`, das für Sie folgende Dinge tut:

- Lesen Sie drei Zahlen β, ε und μ von der Tastatur ein
- Berechnen Sie die y gemäß der Formel

$$y = \log\left(1 + e^{-\beta(\varepsilon-\mu)}\right) .$$

- Geben Sie das Ergebnis formatiert aus.

Kommentieren Sie Ihr skript (Autor, Datum, Zweck), sodass mit dem Befehl `help lnexp` die Kommentarzeilen erscheinen. Speichern Sie das erstellte skript mit dem Namen `lnexp.m` in ihrem MATLAB-Verzeichnis. Kopieren Sie es von dort auf eine Diskette.

10.1.2 Mathematische Identitäten

Schreiben Sie ein MATLAB-skript `idents.m` und speichern Sie es in Ihrem MATLAB-Verzeichnis ab. Das skript soll folgende Dinge tun:

- Lesen Sie zwei Zahlen a und b von der Tastatur ein.

- Prüfen Sie durch Ausrechnen der linken und rechten Seite der unten angeführten Identitäten die Richtigkeit der Formeln nach.

$$
\begin{aligned}
e^{ia} &= \cos a + i\ \sin a \\
\log(a\,b) &= \log a + \log b \\
e^{a+b} &= e^a\, e^b \\
\cos(a+b) &= \cos(a)\cos(b) - \sin(a)\sin(b) \\
\sin(a+b) &= \sin(a)\cos(b) + \cos(a)\sin(b) \\
\sin(a) &= \left(e^{ia} - e^{-ia}\right)/(2i) \\
\cos(a) &= \left(e^{ia} + e^{-ia}\right)/2 \\
\sin(a/2) &= \ldots \\
\cos(a/2) &= \ldots
\end{aligned}
$$

  Geben Sie dazu das Ergebnis der linken und der rechten Seite der Gleichungen formatiert aus.

10.2 Felder

Ziel: Ziel der Übung ist die Verwendung grundlegender Befehle zum Erstellen und Verändern von Arrays und das Üben der Doppelpunkt Notation für den Zugriff auf Arrays.

Anzufertigen ist ein MATLAB Script-File `uebarray.m`, der die Lösungen für die Beispiele enthalten soll.

Voraussetzung: Speichern Sie den File `uebarray.m` in Ihrem MATLAB Directory, welches in der Regel `matlab` heißt. Dies kann direkt aus NETSCAPE mit dem Befehl `save as` erfolgen. Speichern Sie auch den Datenfile `liste.dat`.

Alternativ dazu können Sie mit dem Skript `uebungsdaten` (Übung 1) die Files automatisiert bei Ihnen speichern. Der Aufruf erfolgt einfach in einem Terminalfenster. Dieses Skript kann aber auch direkt aus MATLAB gestartet werden. Dabei muss man wie bei allen externen Programmen ein Rufzeichen voranstellen (`!uebungsdaten`).

Starten Sie MATLAB und geben Sie im Command Fenster den Befehl `uebarray` ein, damit sollte die Vorlage ablaufen, ein Resultat anzeigen, und auf einen Tastendruck warten. Funktioniert das nicht, stellen sie mit der Eingabe `dir` sicher, dass sich der File `uebarray.m` in Ihrem Matlab Directory befindet. Ist das nicht der Fall, dann müssen Sie den File erst mit NETSCAPE oder dem oben angegebenen Skript speichern.

Mit dem Befehl `edit uebarray` öffnet sich ein Editor Fenster mit diesem File. Der Umgang mit dem Editor sollte selbsterklärend sein.

Sie können nun Befehle zum Ausprobieren direkt im MATLAB Command Fenster eingeben, oder im Editor Befehle dazuschreiben, speichern und mit dem Befehl `uebarray` ausfuehren.

Mit dem Befehl `help uebarray` sehen sie wie bei jedem MATLAB Befehl die Hilfe zu diesem Kommando. Dazu wird der erste Block von Kommentaren (gekennzeichnet mit %) verwendet. Füllen Sie dort ihren Namen und das Datum aus.

1. Probieren Sie die Befehle `zeros`, `ones`, `eye`, `linspace`, `logspace` und die Operatoren `+` und `-` aus. Erzeugen Sie z.B. ein 3×4 Array mit lauter Dreiern. Die explizite Eingabe sollte dabei nicht verwendet werden.

2. Erzeugen Sie eine schachbrettartige Anordnung von Einsern und Nullen als Symbole für schwarz und weiß. Die Dimension der Matrix sollte 8×8 sein und sie sollte links oben mit einer Eins beginnen.

   Denken Sie dabei an die Verwendung der Befehle `eye` und `repmat`.

3. Studieren Sie den Befehl `diag` und erzeugen Sie dann mit Hilfe der Befehls `diag` die nachfolgende Matrix. Die Verwendung von `ones` und `eye` ist möglich aber nicht notwendig.

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 3 & 1 & 1 \\ 1 & 1 & 1 & 4 & 1 \\ 1 & 1 & 1 & 1 & 5 \end{bmatrix}.$$

   Verwenden Sie dabei `n = 5` und danach nur mehr `n`.

4. Erstellen Sie eine 5×5 Matrix mit Zufallszahlen im Intervall $[0,\ 1]$, die zusätzlich symmetrisch ist. Als symmetrische Matrix bezeichnet man eine Matrix für die $a_{ij} = a_{ji}$. Verwenden Sie dazu die Befehle `rand`, `triu` und `transpose`.

   Die Lösung dieses Problems ist ein wenig schwieriger und erfolgt in zwei Stufen. Die erste Stufe erzeugt mit `rand` und `triu` eine Matrix, bei der nur der Teil über und inklusive der Hauptdiagonale besetzt ist. In einer zweiten Stufe wird dann die zweite Form des Befehls `triu(A,k)` verwendet, das Ergebnis transponiert und zum ersten Teil addiert.

5. Erzeugen Sie mit dem Befehl `magic` ein $n \times n$ magisches Quadrat `M`, wobei $n = 5$ ist. Beweisen Sie durch Anwendung der Befehle `sum`, `diag` und `fliplr`, dass alle Summen über die Reihen, die Spalten und die beiden Diagonalen gleich groß sind.

   Verwenden Sie dazu die verschiedenen Formen von `sum(A,k)` und die Befehle `diag` und `fliplr`.

   Verbinden Sie die vier Summenvektoren zu einem Zeilenvektor und bedenken Sie dabei, dass die Ausrichtung der resultierenden Vektoren von `sum(A,1)` und `sum(A,2)` unterschiedlich ist.

   Das Zusammenhängen von Arrays erfolgt mit dem Befehl `cat` bzw. mit seiner Kurzform `[A,B]` oder `[A;B]`. Berechnen Sie anschließend die Länge dieses Vektors.

   Stehen im ganzen Vektor die gleichen Zahlen und ist seine Länge $2(n+1)$?

6. Erzeugen Sie folgende Matrix

$$V = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix},$$

   definieren Sie zuerst `n=4` und verwenden Sie dann die Zahl vier nicht mehr. Erzeugen Sie dabei einen Vektor und verwenden Sie dann die Befehle `reshape` und `transpose`.

Die Erzeugung des Vektors $1 \ldots n^2$ kann mit der `Doppelpunkt` Notation erfolgen. Für das Quadrieren kann man `n*n` oder `n^2` verwenden.

7. Speichern Sie in einer Matrix das mittlere 2×2 Quadrat der Matrix V. Verwenden Sie dabei das Keyword `end` in der Form `end-1`.

   Setzen Sie in der ursprünglichen Matrix die vier Eckpunkte auf Null. Verwenden Sie wieder die `Doppelpunkt` Notation und beachten Sie dabei, dass das Keyword `end` auch in der Schrittweite verwendet werden kann.

8. Lesen sie vom File `liste.dat` alle Werte in die Matrix D. Bilden Sie dann die Summe über alle Werte. Dabei können Sie im Summenbefehl `sum` die Form `D(:)` verwenden.

   Ist das Ergebnis Null?