MATLAB Kurzanleitung

1 System

Wir arbeiten mit MATLAB 7.1 (oder neuer) vom Netz, Aufstarten über Windows mit der entsprechenden Ikone. Dreigeteiltes MATLAB-Fenster mit Workspace, Command History und Command Window mit dem zugehörigen Prompt.

1.1 Arbeiten im Command Window

>>
MATLAB Prompt. Bei Kommandos ist Gross- und Kleinschreibung zu unterscheiden.

Mit den Tasten ↑ bzw. ↓
können abgegebene MATLAB Kommandos aus dem Kommandobuffer zurückgeholt werden.
Dies kann auch mit "copy", "paste", erledigt werden.
(Die Command History zeigt alle Kommandos an).

>> dir
Anzeige des aktuellen directory, auch im launch pad durch Wechsel zu Current Directoy möglich.

>> chdir path
Änderung des aktuellen directory, kann auch mit Mausklick bei Current Directory der button list gemacht werden.

>> quit oder exit
Ausstieg aus MATLAB (QUIT, ist ungültig).

>> help help
Informationen über die Hilfe - Möglichkeiten.

>> help Command
Informationen über ein MATLAB - Kommando, z.B. >> help zeros.
Der Menuknopf Help enthält MATLAB Help, eine **sehr gute** online Hilfe.

1.2 Erstellen eines m-Files

Effizient kann mit MATLAB nur mit sogenannten Skripts (= m-Files) gearbeitet werden. Unter dem Menuknopf File gibt es New und dann weiter zu M-File, womit ein neues Fenster für dieses m-File geöffnet wird. In diesem Fenster Editor-Untitled kann editiert und anschliessend mit Save As... gespeichert werden, cf. Abschnitt 5.

2 Daten, Variablen, Konstanten

MATLAB arbeitet mit *einem* Datentyp: *komplexwertige Matrizen*. Daten anderer Form sind als Spezialfälle dieses Typs zu betrachten. (Ausnahme: Textvariablen, sog. strings).

Variablenname:
beginnt mit einem Buchstaben, gefolgt von maximal 16 alphanumerischen Zeichen (Gross-, Kleinschreibung unterscheiden), z.B. Koeff ist *nicht* KOEFF.

=
Wertzuweisung (*keine* Gleichung), z.B. Koeff = 7.5, ordnet der Variable Koeff den Wert 7.5 zu.

`ans`
Default Variable: jeder berechneten oder eingebenen Matrix, der kein Name gegeben wurde, wird der Variablenname `ans` zugeordnet.

`>> i` imaginäre Einheit, $i = \sqrt{-1}$
`>> pi` π
`>> eps` Maschinengenauigkeit
`>> who` Zeigt alle selbstdefinierten Variablen, ist auch mit dem Fenster `Workspace` möglich.
`>> whos` Zeigt alle selbstdefinierten Variablen, inkl. Anzahl Zeilen und Anzahl Spalten.

```
>> clear a b
```
Löscht die Variablen a, b. *Ohne Liste werden alle Variablen gelöscht!*

2.1 Eingabe von Matrizen über die Tastatur

```
>> A = [1 2 -3;2 5 1.2]
>> A = [1, 2, -3;2, 5, 1.2]
>> A = [1 2 -3  return
          2  5 -1.2]
```
Alle drei Eingaben erzeugen dieselbe $2 \times 3-$ Matrix $A = \begin{pmatrix} 1 & 2 & -3 \\ 2 & 5 & 1.2 \end{pmatrix}$.

```
>> A = [1 2 -3;2 5 1.2];
```
`;` unterdrückt die Bildschirmausgabe.

```
>> x = [0:0.2:1]
```
erzeugt den Zeilenvektor `0  0.2000  0.4000  0.6000  0.8000  1.0000`, wobei `0` der Anfangswert, `0.2000` das Inkrement und `1.0000` der Endwert ist.

Zahlenformate:
`short, long, ...,` vgl. unter dem Menuknopf `File` das Menu `Preferences...` und weiter zu `Command Window -> Numeric format`.

Ausgabeformate:
analog

```
>> A(2,3)
```
Das Element der 2-ten Zeile und dritten Spalte von A, hier `>> A(2,3) = 1.2`.

```
>> A(2,3) = -7
```
Ändert das Element in der zweiten Zeile und dritten Spalte zu `-7` ab.

```
>> A(:,3)
```
Spaltenvektor col 3 (col $=$ Kolonne, bzw. Spalte), hier `>> A(:,3) =` $\begin{pmatrix} -3 \\ 1.2 \end{pmatrix}$.

```
>> A(2,:)
```
Zeilenvektor row 2 (row $=$ Zeile), hier `>> A(2,:) =` $(2 \quad 5 \quad 1.2)$.

```
>> help matlab\ops
```
Operators and special characters.

`>> help` bringt die Liste `HELP topics`. In dieser Liste ist z.B. `matlab\ops` zu finden.

Statt Zahlen können als Matrixelemente Variablen, Konstanten oder Ausdrücke stehen. Fehlermeldung, wenn die Matrix nicht rechteckig ist.

Korrektur: Eingaben aus dem Kommandobuffer holen und entsprechend abändern.

```
>> B = [A, zeros(2); zeros(size(A)), 2.0*eye(2)]


B =

    1.0000    2.0000   -3.0000         0         0
    2.0000    5.0000    1.2000         0         0
         0         0         0    2.0000         0
         0         0         0         0    2.0000
```

2.2 In MATLAB definierte Matrizen

```
>> eye(4)


ans =

     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

(4×4) - Einheitsmatrix I_4,
(1 steht auf der Hauptdiagonalen, $a_{ii} = 1$, sonst überall Null, $a_{ij} = 0$, falls $i \neq j$).

```
>> eye(2,4)


ans =

     1     0     0     0
     0     1     0     0
```

(2×4) - Einheitsmatrix.

```
>> size(A)

ans =

     2     3
```

Liefert $m =$ Anzahl Zeilen und $n =$ Anzahl Spalten von A. Hier ist $m = 2$ und $n = 3$.

```
>> eye(size(A))


ans =

     1     0     0
     0     1     0
```

Einheitsmatrix derselben Grösse wie A, (d.h. mit 2 Zeilen und 3 Spalten).

```
>> zeros(m,n)
```
Nullmatrix, Matrix bestehend aus lauter Nullen, Parameter wie für eye.

`>> ones(m,n)`
Matrix bestehend aus lauter Einsen, Parameter wie für die Matrix eye.

`>> rand(m,n)`
Matrix bestehend aus gleichverteilten Zufallszahlen zwischen 0 und 1, Parameter wie für eye.

Im online `Help` finden Sie weitere Definitionen.

3 Operationen und Funktionen

A, B: Matrizen
$k \in \mathbb{R}$, $n \in \mathbb{N}$

Es wird vorausgesetzt, dass die Anzahl Zeilen und Anzahl Spalten der Matrizen für die entsprechenden Operationen verträglich sind. Sonst folgt eine Fehlermeldung!

math. Schreibweise	MATLAB Syntax	Bemerkungen
$A + B$	`A+B`	
$k + a_{ij}$	`k+A`	elementweise Addition von k
$A - B$	`A-B`	
AB	`A*B`	Matrizenprodukt von A mit B
$a_{ij} \cdot b_{ij}$	`A.*B`	" elementweise" Multiplikation von A mit B
$k \cdot a_{ij}$	`k*A`	elementweise Multiplikation von A mit k
$A^T = (a_{ji})$	`A'`	$A \in \mathbb{R}^{m\times n}$: Transposition von A
		$A \in \mathbb{C}^{m\times n}$: Transposition von A *und* Konjugation der Elemente
	`A.'`	$A \in \mathbb{C}^{m\times n}$: *nur* Transposition von A
AB^{-1}	`A/B`	Matrizenprodukt von A mit B^{-1}
$A^{-1}B$	`A\B`	Matrizenprodukt von A^{-1} mit B
a_{ij}/b_{ij}	`A./B`	"elementweise" Division von A durch B
b_{ij}/a_{ij}	`A.\B`	"elementweise" Division von B durch A
A^n	`A^n`	$n-$te Potenz von A
a_{ij}^n	`A.^n`	"elementweises" Potenzieren
$\|A\| = det(A)$	`det(A)`	Determinante von A
$diag(A)$	`diag(A)`	Hauptdiagonale von A (Elemente a_{ii} als Spaltenvektor)
$\sin(A), ...$	`sin(A)`	Sinus wirkt elementweise auf die Matrix A

4 Graphik

`MATLAB` hat ein separates Graphikfenster, das die Darstellung von Kurven und Flächen erlaubt. Die Stützpunktkoordinaten müssen vorgängig in Vektoren oder Matrizen definiert werden.

4.1 Graphische Darstellung einer Sinuskurve

```
x = [1:0.1:pi];     % erzeugt die x−Koordinaten der Stützpunkte
y = sin(x);         % berechnet die zugehörigen y−Koordinaten
plot(x,y, '.-');    % zeichnet einen Polygonzug durch die Stützpunkte
grid                % unterlegt ein Gitter
```

Diese Befehle erzeugen die folgende graphische Darstellung.

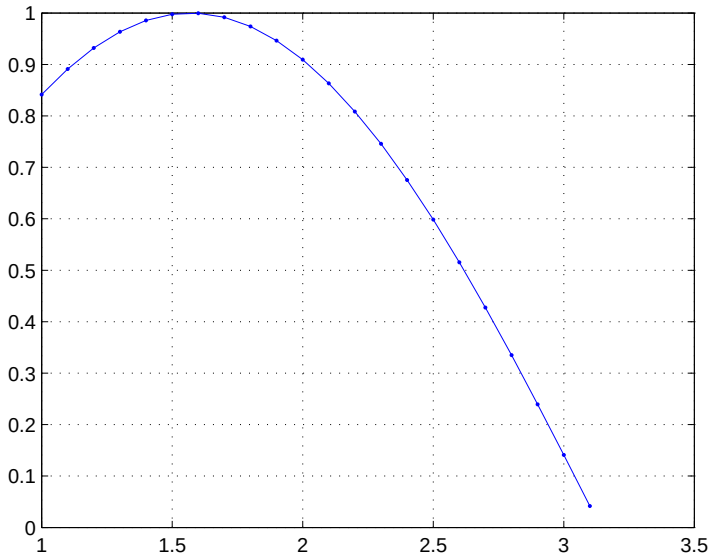

Abbildung 1: Eine erste graphische Darstellung.

4.2 Graphische Darstellung einer Sattelfläche

```
x = [-1:0.2:pi]; y = x';      % erzeugt lineare Raster in x− und y−Richtung
X = ones(size(y))*x;
Y = y*ones(size(x));          % erzeugt ein Feld mit Rasterkoordinaten
                              % (Häuschenpapier)
Z = X.^2 - Y.^2;              % z−Werte über den Rasterpunkten
mesh(Z)                       % zeichnet ein Polygonnetz durch die Stützpunkte
contour(Z)                    % zeichnet Niveaulinien
```

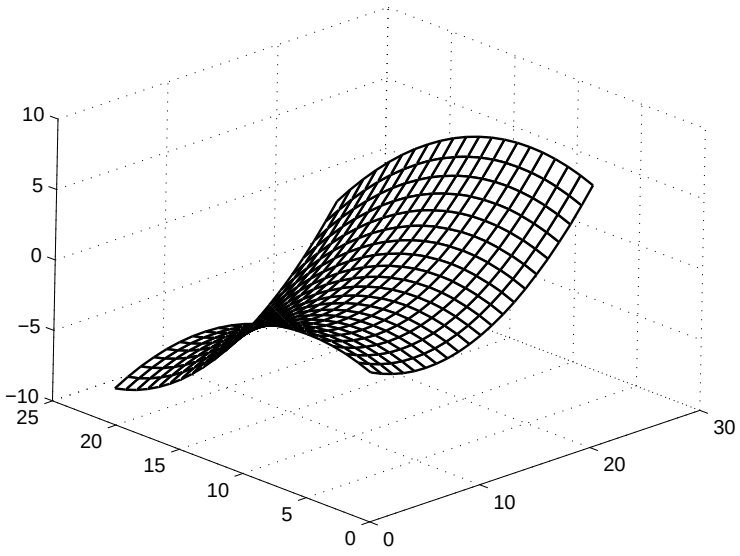

Abbildung 2: Polygonnetz

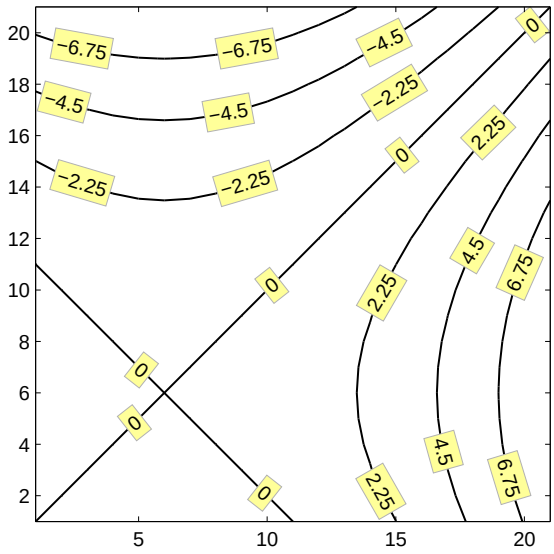

Abbildung 3: Niveaulinien

5 Funkionen- und Programmfiles

Diese Files haben die Extension `.m`

Im Beispiel wird im Funktionsfile `Funktion.m` eine Funktion $f(x)$ definiert, deren Ableitung im Programmfile `ableit.m` gebildet und anschliessend graphisch dargestellt wird.

5.1 Definition einer Funktion

```
function y = Funktion(x);     % Funktionsname muss gleich dem Namen des m-Files sein.
y = sin(x);                   % Funktionswerte werden berechnet
```

Mit dem Menuknopf `File -> New ->` weiter zu `M-File` wird ein neues `m-File` geöffnet, obige beiden Zeilen in das neue Fenster `Editor-Untitled` tippen und anschliessend mit `Save As...` als `Funktion.m`, im *richtigen* `directory` speichern.

5.2 Definition eines Programms, das eine Funktion ruft

Im File `ableit.m` steht nun das Programm, das die Funktion `Funktion` ruft.

```
a = 0.2;                                % linker Rand des Intervalls
b = 2.1;                                % rechter Rand des Intervalls
n = 15;                                 % Anzahl Punkte
dx = (b - a)/n;                         % Berechnung des Inkrements
x = [a:dx:b];                           % Stützpunkte
F = Funktion(x);                        % Stützwerte, die Funktion wird gerufen
df = diff(F);                           % Berechnung der Differenzen
n = length(x);                          % Anzahl Elemente im Vektor x, bzw. F
zdf = (df(1:n-2) + df(2:n-1))/(2*dx);   % zentraler Differenzenquotient
plot(x(2:n-1),zdf, 'r')                 % zdf wird in roter Farbe gezeichnet
```

In einem `m-File` können Kommentare mit % gemacht werden. Was in einer Zeile hinter einem % kommt wird als Kommentar interpretiert. Wo in einer Zeile % steht spielt keine Rolle.

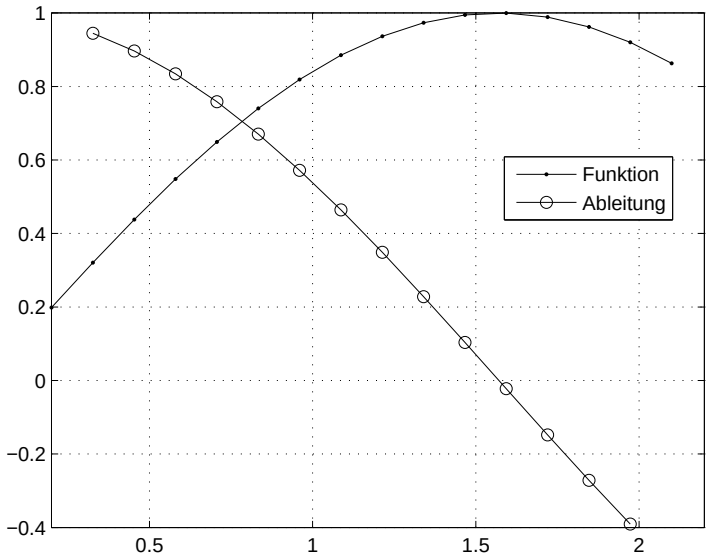

Abbildung 4: Graphische Darstellung einer Funktion samt ihrer Ableitung.

Tipp: Siehe auch

http://www.imrtweb.ethz.ch/matlab

Dies ist eine sehr gute und *umfangreiche* Einführung in `MATLAB`.