Learning MATLAB by doing MATLAB

Dieses kleine MATLAB-Tutorial[1] setzt auf "Learning by Doing". Geben Sie jeweils die hinter `>` angegebenen Befehle im MATLAB Command Window ein und beobachten Sie, was bei der Ausgabe passiert. Arbeiten Sie am Anfang zunächst die Abschnitte 1 bis 4 durch. Abschnitt 5 sollten Sie erst nach der Einführung von Matrizen und den entsprechenden Matrixoperationen durcharbeiten.

1. Variablen

`>a=22`	die Variable a wird angelegt und mit 22 initialisiert
`>b=20`	
`>a+b`	
`>c=a+b;`	Semikolon am Ende unterdrückt Ausgabe
`>pi`	
`>format long; pi`	
`>format short`	
`>sin(pi)`	Wieso nicht Null?
`>whos`	gibt aus, welche Variablen belegt sind
`>clear a b`	löscht die Variablen a und b
`>whos`	
`>clear`	löscht alle Variablen
`>help clear`	gibt Hilfe zum Befehl `clear`

2. Schleifen

```
>for j=1:10,
disp(sprintf('%i. Schleife',j));
end
```

j läuft von 1 bis 10. (MATLAB macht erst etwas, wenn `end` kommt!)

```
>for j=1:2:10,
disp(sprintf('%i. Schleife',j));
end
```

j läuft in Zweierschritten von 1 bis 10.

Neben der 'for'-Schleife gibt es auch eine 'while'-Schleife. Siehe `help while` für Informationen.

3. Graphiken

```
>for i=1:10, x(i)=i/10; y(i)=x(i)^2; z(i)=sqrt(x(i)); end
>x,y,z,
>plot(x,y)
>plot(x,z)
>clf
>plot(x,y)
>hold on
>plot(x,z)
>plot(x,2*z,'r')
>plot(x,y+z,'g*')
>hold off
>plot(x,y-z,'k+')
>help plot
>title('Meine Grafik')
>xlabel('x-Achse')
>ylabel('y-Achse')
>axis([0,20,-5,50])
>box
>grid
```

[1] Basierend auf einem von Christian Mehl und Andreas Steinbrecher an der TU Berlin entwickelten Tutorial

```
>clf
>subplot(3,2,1)
>plot(x,y)
>subplot(3,2,2)
>plot(x,z,'k')
>subplot(3,2,5)
>plot(x,z+y,'mo')
>hold on
>plot(x,z,'k')
>subplot(3,2,1)
>plot(x,z,'k')
>subplot(3,2,4)
>title('leer')
>subplot(3,1,2)
>plot(y)
```

Der Plot hat $3 \cdot 2 = 6$ Subplots. Der 1. Subplot wird angesprochen.

Jetzt gibt es nur noch drei Subplots, einen pro Zeile. Der 2. wird angesprochen.

4. Unterprogramme, m-Files

Zur effizienten Nutzung von MATLAB ist die Verwendung von Skripten und Funktionen unerlässlich. Für die folgenden Unterprogramme müssen Sie Dateien mit dem jeweiligen Namen `foo.m` anlegen (z.B. mit dem MATLAB-internen Editor) und abspeichern. Doch Achtung: Diese m-files müssen in dem Verzeichnis liegen, in dem sich MATLAB aktuell befindet. Das aktuelle Verzeichnis kann im Befehlsfenster unter "Current Directory" oder mittels dem Befehl `cd` geändert werden.

```
% Das ist mein erstes Skript.
s = 0;
for i = 1:10,
  s = s + i;
end
```

Bitte unter `test1.m` abspeichern.

Im Befehlsfenster rufen Sie dieses Skript einfach durch Eingabe des Befehls `test1` auf:

```
>s,
>test1
>s,
```

Im Gegensatz zu einem Skript akzeptiert eine MATLAB-Funktion Eingabeparameter. Ausserdem sind die innerhalb einer Funktion angelegten Variablen nicht vom Befehlsfenster aus sichtbar (und umgekehrt).

```
function [s,p]=test2(n)
% Das ist meine erste MATLAB-Funktion. Sie heißt test2.m.
% Eingabe: natürliche Zahl n
% Ausgabe: Summe und Produkt der Zahlen von 1 bis n
s = 0; p = 1;
for i = 1:n
  s = s + i; p = p * i;
end
```

Bitte unter `test2.m` abspeichern.

Im Befehlsfenster:

```
>help test2,
>[s,p] = test2(4),
>[s,p] = test2(10),
```

Unterprogramme können auch mehrere Variablen zur Eingabe (z.B. `function ausgabe=test3(A,B,C)`) oder weder Eingabe noch Ausgabe erfordern (z.B. `function []=test4()`, Aufruf mit `test4`).

5. Vektoren, Matrizen

`>a=7`	a wird als Skalar interpretiert (oder 1×1-Matrix)
`>b=[1,2,3]`	nach Komma: neues Element in derselben Zeile, also hier $b \in \mathbb{R}^{1,3}$
`>c=[1+2,3,3]`	
`>d=[7 7 2]`	Leerzeichen haben gleiche Bedeutung wie Kommata
`>e=[7 a 2]`	
`>f=[1;2;3;4]`	Semikolon: Beginn einer neuen Zeile, also hier $f \in \mathbb{R}^{4,1}$
`>g=f(2)`	fragt das 2. Element des Spaltenvektors f ab
`>E=[1 2 3;2 1 3]`	MATLAB unterscheidet zwischen Groß- und Kleinschreibung
`>h=E(1,2)`	fragt das $(1,2)$-Element von E an
`>E`	
`>F(3,4)=7`	MATLAB interpretiert F zunächst als 3×4-Matrix. Nicht benannte Elemente werden auf Null gesetzt.
`>F(4,3)=2`	Jetzt brauchen wir eine 4. Zeile!
`>F(1,2)=3`	Belegt das $(1,2)$-Element von F mit 3.
`>F(1:2,3:4)=[1 3;2 7]`	$1:2$ bedeutet Zeile 1 bis Zeile 2; $3:4$ bedeutet Spalte 3 bis Spalte 4
`>A=[1 2];`	Semikolon am Ende unterdrückt die Ausgabe
`>A`	
`>pi`	
`>A=eye(3)`	3×3-Einheitsmatrix
`>b=[1 2 3]`	
`>B=diag(b)`	Diagonalmatrix
`>C=diag([1 7 8])`	
`>D=diag([1;7;8])`	geht auch mit Spaltenvektoren
`>E=ones(4)`	4×4-Matrix mit Einsen
`>F=ones(2,3)`	2×3-Matrix mit Einsen
`>G=zeros(4)`	
`>H=zeros(2,3)`	
`>I=[A B;zeros(3) A]`	Blockmatrix
`>I`	
`>I'`	Transponierte von C
`>w(3)=5`	MATLAB interpretiert w als Zeilenvektor. Das 3. Element wird gleich 5 gesetzt.
`>x=0:1/3:2`	Zeilenvektor mit Einträgen von 0 bis 2 in 1/3-Schritten

5.1. Einfache Operationen

`>clear`	
`>A=[1 2 3;2 1 0]`	
`>B=[2 2;1 0;0 1]`	
`>C=[0 1 0;5 1 3]`	
`>size(A)`	Gibt Anzahl der Zeilen und Spalten von A als Zeilenvektor aus.
`>[m,n]=size(A)`	So bekomme ich sie einzeln.
`>b=[2 1 3]`	
`>x=[2;1;3]`	
`>A*B`	Matrixmultiplikation
`>A*C`	Fehler, da Dimensionen nicht passen!
`>A`	
`>C`	
`>A*C'`	$A \cdot C^T$
`>diag(A*C')`	liefert die Diagonale der Matrix
`>whos`	`ans` (für "answer") ist die unbenannte Ausgabevariable
`>D=A+C`	Matrixaddition
`>E=A+B`	Fehler, da Dimensionen nicht passen!
`>E=A-B'`	$E = A - B^T$
`>g=A*x`	Matrix mal Vektor
`>g=A*b`	Fehler!
`>A`	
`>b`	
`>B`	
`>f=b*B`	Zeilenvektor mal Matrix
`>C=[1 2 3]'`	

5.2. Matrixmanipulationen

`>clear`	
`>A=[1 2;3 4]`	
`>A(3,2)=7`	Hinzunahme einer dritten Zeile!
`>A(1:2,2)`	`(1:2,2)`: 1. bis 2. Element der 2. Spalte
`>A(3,1:2)`	`(3,1:2)`: 1. bis 2. Element der 3. Zeile
`>B(3:4,3)=[5;6]`	MATLAB erzeugt eine Matrix B bei der das 3. und 4. Element der 3. Spalte 5 bzw 6 ist. Alle anderen Einträge sind Null.
`>C(4:5,4)=A(1:2,2)`	
`>B(:,3)`	3. Spalte von B
`>d=C(1,:)`	1. Zeile von C
`>E=[1 2 3;4 5 6;7 8 9;10 11 12]`	
`>E(1:2:4,3)`	`(1:2:4,3)`: Sucht in der 3. Spalte vom 1. bis 4. Element jedes 2. Element heraus.
`>F=[1 2 3 4 5;6 7 8 9 10;11 12 13 14 15]`	
`>G=F(1:2:3,1:2:5)`	Sucht in der 1., 3. und 5. Spalte jeweils vom 1. bis 3. Element jedes 2. Element heraus.
`>H=[1 3;9 11]`	
`>H^(-1)`	Die Inverse.
`>inv(H)`	Ebenfalls die Inverse.
`>det(H)`	Die Determinante.
`>b=[99 100 101]`	
`>F(1,1:3)=b`	
`>A`	
`>A(1,1:3)=b`	