**Matlab-Syntax in a nutshell:**

Version: 6.9.2009

Zu folgenden Themen gibt es hier eine kurze Matlab-Befehlsübersicht:

\* Variablen

\* Vektoren

\* Matrizen

\* Datentypen

\* Zeichenketten (string)

\* Operatoren

\* Mathematische Funktionen

\* Zufall und beschreibende Statistik

\* Graphik

\* Fallunterscheidungen

\* Schleifen

\* Suchen und sortieren

\* Datenverwaltung

\* Bildschirmausgaben und Kommentare

\* Zeitmessung

**Variablen:**

Variablennamen bestehen aus Buchstaben, Ziffern und Unterstrichen. Sie beginnen grundsätzlich mit einem Buchstaben. Groß- und Kleinbuchstaben werden unterschieden.

a=7

Generiert eine Variable a mit Wert 7, oder ändert den Wert der Variablen a in 7, falls a bereits eine bekannte Variable ist. Ein einfaches Gleichheitszeichen bezeichnet in Matlab eine Definition, "a gesetzt gleich 7".

clear a

Löscht die Variable a aus dem aktuellen Workspace (der Kommandozeile oder einer Funktion)

clear all

Löscht alle Variablen des aktuellen Workspace

**Vektoren:**

v=[1 4 78]

Generiert einen Zeilenvektor v mit drei Komponenten (die Komponenten innerhalb einer Zeile können entweder durch Leerzeichen oder durch Kommas voneinander getrennt werden.) Eckige Klammern umschließen generell die zusammengehörigen Komponenten eines Vektors oder einer Matrix.

v=[1; 4; 78; 15.8; 0]

Generiert einen Spaltenvektor v mit drei Komponenten (Zeilen werden durch Semikolon getrennt)

l=[]

Generiert einen leeren Vektor l

zaehl=1:10

Generiert einen Vektor zaehl mit 10 Komponenten, die mit Werten von 1 bis 10 belegt werden

rueck=10:-2:1

Generiert einen Vektor mit 5 Komponenten, der von 10 bis 1 in Zweierschritten rückwärts zählt

neu=[zaehl rueck]

Generiert einen Vektor mit 15 Komponenten, indem die Vektoren zaehl und rueck zu einem langen Vektor aneinandergehängt werden

l=linspace(a,b,n)

Generiert einen Vektor l mit n Komponenten, die mit jeweils gleichem Abstand zwischen a und b liegen. l=linspace(a,b) generiert einen Vektor mit 100 Komponenten zwischen a und b.

b=v(2)

Generiert eine Variable b, der der Wert der zweiten Komponente von v zugewiesen wird

v(2)

weist der Standard-Variablen "ans" den Wert der zweiten Komponente von v zu

c=v(1:3)

generiert einen Vektor mit drei Komponenten, denen die ersten drei Werte von v zugewiesen werden

d=v(end)

Generiert eine Variable d, der der letzte Wert des Vektors v zugewiesen wird

alles=v(:)

Setzt die Variable alles mit Vektor v gleich. Andere Schreibweisen sind alles=v oder alles=v(1:end)

zeil=v'

Transponisert v: generiert einen Zeilenvektor zeil, dem alle Werte des Spaltenvektors v zugewiesen werden. (Funktioniert genauso für Spalte nach Zeile).

l=lenght(v)

weist der Variablen l die Anzahl der Komponenten von v zu

**Matrizen:**

m=[1 5; 79 0.5; 17 0]

Generiert eine Matrix mit drei Zeilen und zwei Spalten (Zeilen sind durch Semikolon getrennt, die Werte innerhalb einer Zeile durch Leerzeichen oder Komma)

z=zeros(3,4)

Generiert eine Matrix mit drei Zeilen und vier Spalten, deren Werte alle 0 sind

o=ones(3,4)

Entsprechend, aber alle Werte sind 1

e=m(2,1)

Weist der Variablen e den Wert der zweiten Zeile und ersten Spalte von m zu

e=m(2)

Weist ebenfalls der Variablen e den Wert der zweiten Zeile und ersten Spalte zu. Bei der linearen Indizierung wird entlang der Spalten gezählt. (e=m(4) wäre in diesem Beispiel der Wert der ersten Zeile und zweiten Spalte.)

zeile1=m(1,:)

Erzeugt einen Vektor zeile1 mit allen Werten der ersten Zeile von m

[zeilen,spalten]=size(m)

Weist der Variable Zeilen die Anzahl der Zeilen, der Variablen spalten die Anzahl der Spalten in Matrix m zu

zeilen=size(m,1)

Weist der Variable Zeilen die Größe der ersten Dimension zu, also die Anzahl der Zeilen. Entsprechend für die anderen Dimensionen einer Matrix (spalten=size(m,2), tiefe=size(m,3) etc)

l=length(m(:))

Weist der Variablen l die Anzahl aller Elemente der Matrix m zu.

mt=m'

Transponiert die Matrix m. Die erste Zeile von m wird zur ersten Spalte von mt etc.

B = repmat(A,m,n)

Repliziert die Matrix A. B ist eine Matrix, in der A m-mal "untereinander" und n-mal "nebeneinander" enthalten ist.

B = reshape(A,m,n)

Sortiert die Einträge von Matrix A in die aus m Zeilen und n Spalten bestehende Matrix B. (Spaltenweise durchgezählt)

**Datentypen**

Standardtyp für Zahlen ist in Matlab der Datentyp double (Fließkommazahlen mit doppelter Präzision)

realmax

größte positive Fließkommazahl, mit der Matlab rechnen kann

eps

kleinste positive Fließkommazahl, mit der Matlab rechnen kann

**Umwandlung von Datentypen:**

double(v)

Wandelt eine Variable (Vektor oder Matrix) in Fließkommazahlen um.

int8(v), int16(v), int32(v), int64(v)

Wandelt eine Variable (Vektor oder Matrix) in ganze Zahlen um, die mit jeweils maximal 8, 16, 32 oder 64 bit dargestellt werden.

logical(v)

Wandelt eine Variable (Vektor oder Matrix) in Wahrheitswerte um.

char(v)

Wandelt eine Variable (Vektor) in eine Zeichenkette um.

**Tests auf Datentypen:**

b=islogical(a);

gibt den Wahrheitswert 1 zurück, wenn a aus Wahrheitswerten besteht

b=isnumeric(a);

gibt den Wahrheitswert 1 zurück, wenn a aus Zahlen besteht (Fließkommazahlen, ganze Zahlen, komplexe Zahlen)

b=isa(a,'class-name');

gibt den Wahrheitswert 1 zurück, wenn a dem mit der Option class-name angegebenen Datenyp entspricht, z.B. logical, char, integer, int8, struct

b=isnan(a);

gibt eine Matrix aus Wahrheitswerten mit den gleichen Dimensionen wie a zurück, für jeden Eintrag wird 1 zurückgegeben, wenn er NaN (not a number) - also keine darstellbare Zahl ist (z.B. nachdem man durch 0 geteilt hat).

**Zeichenketten (string)**

s='bla'

Erzeugt einen string s mit Wert bla

b=blanks(10)

Erzeugt einen String aus 10 Leerzeichen

s(2)

Zweites Element (Buchstabe) des strings s

s=sprintf(format, A);

Schreibt den Inhalt von Matrix A formatiert in die Zeichenkette s.

wichtige Formatierungsparameter (Werte für format) sind:

%g Kompakte Darstellung von Dezimalzahlen

%s Darstellung von Strings

(mehr Formatierungsmöglichkeiten in der Hilfe)

strcmp(s,'bla')

vergleicht string s mit string 'bla' auf Gleichheit

strncmp(s,'bla',2)

vergleicht erste 2 Buchstaben der beiden Strings auf Gleichheit

ind=strfind(str,pattern)

sucht string pattern in string str und gibt Startindex zurück

**Operatoren**

**Arithmetische Operatoren** +, - \*, /, ^

werden auf ganze Vektoren bzw Matrizen angewandt,

für punktweise Berechnung

verwendet man .\*, ./ und .^

**Vergleichsoperatoren** <, >, <=, >=, ==

geben Wahrheitswerte zurück, 1 für wahr, 0 für falsch. == ist der Test auf Gleichheit (im Gegensatz zu =, was "gesetzt gleich" bedeutet).

Bei Vektoren und Matrizen erfolgt der Vergleich Punktweise

z.B.: a=(2<3); b=([0.5 7]==[0.5 8]);

l=isequal(a,b)

Weist der Variablen l den Wahrheitswert für den Test auf Gleichheit von a und b zu. a und b müssen keine Zahlen, Vektoren oder Matrizen sein, isequal funktioniert für alle Datentypen (z.B. auch strings und Strukturen)

**Logische Operatoren:**

& logisches und

| logisches Oder

~ logische Negation

Punktweiser Vergleich, der Wahrheitswerte liefert

**Mathematische Funktionen**

a=abs(m)

Absolutbetrag, komponentenweise berechnet von Matrix m

a=sqrt(m)

Quadratwurzel, komponentenweise berechnet von Matrix m

a=exp(m)

Exponentialfunktion, komponentenweise berechnet von Matrix m

a=log(m)

Natürlicher Logarithmus, komponentenweise berechnet von Matrix m

a=log10(m)

Logarithmus zur Basis 10, komponentenweise berechnet von Matrix m

a=sin(m), a=cos(m)

Sinus und Cosinus, komponentenweise berechnet von Matrix m

a=sum(m)

a ist die spaltenweise aufsummierte Matrix m (a ist ein Zeilenvektor)

a=sum(m,dim)

a ist die entlang der Dimension dim aufsummierte Matrix m

a=rem(m,n)

Restbeträge wenn die Matrix m komponentenweise ganzzahlig durch Matrix n geteilt wird

**Zufall und beschreibende Statistik**

r=rand(2,5)

Generiert eine 2x5-Matrix mit gleich verteilten Zufallszahlen zwischen 0 und 1

r=randn(2,5)

Generiert eine 2x5-Matrix mit normal verteilten Zufallszahlen mit Mittelwert 0 und Standardabweichung 1

p=randperm(n)

Gibt einen Vektor p zurück, in dem die natürlichen Zahlen 1 bis n in zufälliger Reihenfolge stehen

mean(m,dim)

Mittelwert von m entlang Dimension dim (wenn keine Dimension angegeben wird: spaltenweise)

std(m,dim)

Standardabweichung von m entlang Dimension dim (wenn keine Dimension angegeben wird: spaltenweise)

var(m,dim)

Varianz von m entlang Dimension dim (wenn keine Dimension angegeben wird: spaltenweise)

median(m,dim)

Median von m entlang Dimension dim (wenn keine Dimension angegeben wird: spaltenweise)

min(m,dim , max(m,dim)

Minimum und Maximum entlang Dimension dim (wenn keine Dimension angegeben wird: spaltenweise)

h=hist(v,nbins)

Histogramm der Häufigkeit des Auftretetens von Werten im Vektor v wobei nbins die Anzahl Klassen angibt, in die der Bereich zwischen dem minimalen und dem Maximalen Wert von v aufgeteilt wird. Ohne Angabe des Ausgabearguments h wird das Histogramm auf dem Bildschirm dargestellt. Es gibt noch mehr Möglichkeiten, hist zu benutzen, s. Hilfe.

[p,S]=polyfit(x,y,n)

Findet das Polynom n-ten Grades, das die durch die Vektoren x und y gegebenen Messwerte optimal beschreibt, indem der quadratische Abstand zwischen Messwerten und Funktionswerten des Polynoms minimiert werden. p gibt die Koeffizienten des Polynoms p1xn+p2xn-1+...pnx+pn+1 an. S ist eine Struktur zur Verwendung mit polyval um die Güte der Kurvenanpassung abzuschätzen.

y=polyval(p,x)

Bestimmt die Funktionswerte für ein Polynom p1xn+p2xn-1+...pnx+pn+1 dessen Koeffizienten im Vektor p angegeben sind für die im Vektor x angegebenen Argumente.

[y,delta]=polyval(p,x,S)

Bestimmt die Funktionswerte für ein Polynom p1xn+p2xn-1+...pnx+pn+1 dessen Koeffizienten im Vektor p angegeben sind für die im Vektor x angegebenen Argumente. Zusätzlich wird mit Hilfe der von polyfit zurückgegebenen Struktur S im Vektor delta ein Mass für die Zuverlässigkeit der Schätzung angegeben. Der Bereich von y-delta bis y+delta enthält mindestens 50% der Schätzungen (bei Normalverteilteilung).

**Grafik**

plot(v)

trägt die Werte des Vektors v gegen ihre Indizes auf

plot(x,y)

trägt die Werte des Vektors y gegen die des Vektors x auf

plot(x1,y1,x2,y2)

kombiniert die Auftragungen von y1 gegen x1 und y2 gegen x2 in einem Bild

plot(x,y,'ro-')

benutzt rote, mit Linien verbundene Kreise, um y gegen x aufzutragen (Reihenfolge der Formatierungsparameter ist egal - Parameter bitte in der Hilfe nachgucken.)

loglog(x,y)

Doppellogarithmische Auftragung von y gegen x

semilogx(x,y)

Auftragung mit logarithmischer x- und linearer y-Achse (semilogy entsprechend)

errorbar(x,y,e)

Trägt y gegen x auf und versieht jeden Datenpunkt mit symmetrischen Fehlerbalken der Laenge 2e (jeweils e nach oben und unten)

pie(x), pie3(x)

Darstellung des Vektors x als Kuchengraphik bzw als 3-dimensionale Kuchengrafik

mesh(x,y,z)

3-dimensionale Darstellung der Matrix z als Maschendrahtmodell mit Vektoren x und y als Achsen. z muss die Dimension y\*x haben. (Mehr Möglichkeiten in der Hilfe.)

surf(x,y,z)

3-dimensionale Darstellung der Matrix z als farbige Oberfläche mit Vektoren x und y als Achsen. z muss die Dimension y\*x haben. (Mehr Möglichkeiten in der Hilfe.)

hist(v,nbins)

Stellt das Histogramm der Häufigkeit des Auftretens von Werten im Vektor v grafisch dar, wobei nbins die Anzahl Klassen angibt, in die der Bereich zwischen dem minimalen und dem Maximalen Wert von v aufgeteilt wird (siehe Zufall und beschreibende Statistik).

imagesc(m)

Zeigt die Elemente einer 2D-Matrix farbkodiert an, das kleinste Element ist dunkelblau, das größte rot.

figure(2)

öffnet ein neues Graphikfenster und gibt ihm die Nummer 2

close(2)

schließt das zweite Graphikfenster

clf

Inhalt des aktuellen Graphikfensters löschen

subplot(m,n,p)

Teilt das aktuelle Graphikfenster in m\*n Unterabbildungen (m nebeneinander, n untereinander) und bewirkt, dass beim nächsten plot-Befehl die p-te Unterabbildung verwendet wird (zeilenweise durchgezählt).

xlabel('bla bla')

Beschriftung der x-Achse (y entsprechend)

title('Tolles Bild')

Bildüberschrift

axis([xmin xmax ymin ymax])

setzt die Granzwerte fest, zwischen denen Werte graphisch aufgetragen werden sollen

bild=imread(datei.jpg)

lädt eine jpg-Datei als 3D-Matrix mit dem Namen bild in den Workspace. Werte zwischen 0 und 255

image(bild)

stellt eine 3D-Matrix als farbiges Bild dar.

**Fallunterscheidungen:**

"if" für wenige unterschiedene Fälle

if Bedingung

Befehle

end

if Bedingung

Befehle1

else

Befehle2

end

if Bedingung1

Befehle1

elseif Bedingung2

Befehle2

else

Befehle3

end

z.B.:

if a==b

c=1;

elseif a>b

c=2;

else

c=0;

end

switch für viele Möglichkeiten, oder wenn strings getestet werden sollen

switch Ausdruck

case Fall1

Befehle1

case {Fall2, Fall3}

Befehle2

otherwise

Befehle3

end

z.B.

str='a'

switch str

case 'a'

x=1

case 'b'

x=2

otherwise

x=0

end

**Schleifen**

**Zählschleife:**

for variable=Ausdruck

Befehle

end

z.B.

o=ones(1,10)

for a=2:10

o(a)=2\*o(a-1);

end

**Bedingte Schleife:**

while Bedingung

Befehle

end

z.B.

o=1;

a=1;

while o(a)<1000

a=a+1;

o=[o 2\*o(a-1)];

end

**Doppelpunktoperator als implizite Schleife:**

Folgende Möglichkeiten erzeugen den gleichen Vektor a:

a=[1:0.1:2]

oder

a=1;

for b=1:10

a=[a a+0.1\*b];

end

oder

a=1

while a(end)<2

a=[a a(end)+0.1];

end

**Suchen und Sortieren**

B=A>0

Erzeugt eine Matrix B aus Wahrheitswerten, in der für jede Komponente der Matrix A angegeben wird, ob sie positiv ist.

[i,j]=find(X)

Gibt die Indizes der Matrixelemente X zurück, die ungleich 0 sind

[i,j]=find(X>10)

Gibt die Indizes der Matrixelemente X zurück, die >10 sind. (Find ist ein mächtiger Befehl, in der Bedingung kann jede beliebige Art von Abfrage stehen, auch für andere Datentypen.)

m2=sortrows(m1,n)

Sortiert die Zeilen der Matrix m1 so in Matrix m2 ein, dass die n-te Zeile aufsteigend sortiert ist.

[m2,index]=sortrows(m1,n)

Sortiert die Zeilen der Matrix m1 so in Matrix m2 ein, dass die n-te Zeile aufsteigend sortiert ist. index ist ein Vektor, der die Abfolge der Sortierung angibt, so dass gilt B=A(index,:)

**Datenverwaltung:**

save filename

sichert alle aktuellen Variablen im file filename.mat im aktuellen Verzeichnis

save('filename','var1','var2') alternativ: save filename var1 var2

sichert nur Variablen var1 und var2 im file filename.mat

save -ascii filename.txt

sichert alle aktuellen Variablen als Textdatei filename.txt, die von einem beliebigen Editor eingeladen und weiterverarbeitet werden kann.

load('filename')

holt alle in filename.mat gespeicherten Variablen in den Workspace

cd subfolder

wechselt das aktuelle Verzeichnis zum Unterordner subfolder

cd ..

wechselt das aktuelle Verzeichnis zum übergeordneten Verzeichnis

fscanf

Mächtige Möglichkeit jegliche formatierte Daten einzulesen (viele Optionen, bei Bedarf bitte in der Hilfe nachsehen)

fprintf

Mächtige Möglichkeit Daten formatiert abzuspeichern (viele Optionen, bei Bedarf bitte in der Hilfe nachsehen)

xlswrite('dateiname.xls',variable)

Speichert eine Matlab-Variable in einer xls-Datei ab, um sie mit Excel weiter zu verarbeiten.

variable=xlsread('dateinme.xls')

lädt eine mit Excel erzeugte Datei in den Matlab Workspace ein.

antwort=input('bitte Zahl eingeben')

Liest eine Benutzereingabe von der Tastatur in Variable antwort ein

**Bildschirmausgaben und Kommentare**

;

hinter einem Befehl unterdrückt die Bildschirmausgabe

...

unterbricht eine Programmzeile. Dadurch lassen sich sehr umfangreiche Befehle lesbarer schreiben (ansonsten führt Matlab eine Programmzeile aus, sobald sie mit Return abgeschlossen ist).

echo on

wiederholt den jeweils ausgeführten Befehl auf dem Bildschirm

echo off

stellt echo wieder ab

pause

wartet mit der Ausführung des nächsten Befehls auf einen Tastendruck

pause(10)

wartet mit der Ausführung des nächsten Befehls 10sec

disp('hello world'), disp(v)

Stellt eine Matrix oder Zeichenkette auf dem Bildschirm dar, ohne den Arraynamen darzustellen.

warning('Achtung, Achtung!')

Stellt eine Warnung auf dem Bildschirm dar, der Ablauf des Programms wird dadurch nicht beeinflusst.

error('Achtung, Fehler!')

Stellt eine Fehlermeldung rot geschrieben auf dem Bildschirm dar und bricht das Programm ab.

type('filename')

Stellt den Inhalt der Datei auf dem Bildschirm dar.

lookfor Begriff

Durchsucht alle Hilfetexte nach dem angegebenen (englischen) Begriff und stellt jeweils die erste Zeile des Textes auf dem Bildschirm dar.

format long, format long e, format short, format short e

Umstellung, wie viele Nachkommastellen bei Fließkommazahlen angezeigt werden

%

Schlüsselzeichen, dass alles weitere in einer Programmzeile als Kommentar gewertet wird

%%

Schlüsselzeichen für den Beginn eines neuen Programmteils (cell)

**Zeitmessung**

tic; toc

Stoppuhr:

tic %startet die Stoppuhr

Befehle

toc %gibt die Zeit seit dem Aufruf von tic auf dem Bildschirm aus.

Alternativ kann die Zeit mit t=toc; in die Variable t geschrieben werden.

str=date

gibt eine Zeichenkette mit dem aktuellen Datum zurück

str=clock

gibt eine Zeichenkette mit dem aktuellen Datum und Uhrzeit zurück