

MATLAB-Spicker (für Version ≥ 5.0 / Rel. ≥ 11)

v1.0, © Thorsten Raasch, 2001

Hilfefunktion

Befehl	Aktion
<code>help</code>	Übersicht zur Online-Hilfefunktion
<code>help demos</code>	Übersicht der mitgelieferten Demonstrationsdateien
<code>help det</code>	Hilfe zu eingebauten Befehlen bzw. zu m-Files, hier zum Befehl <code>det</code>

Allgemeines

Befehl	Aktion
<code>x='Hallo'</code>	Zuweisung einer Variablen, hier einer Zeichenkette
<code>x=1.5+3i;</code>	stumme Zuweisung, d.h. ohne Ausgabe von <code>x</code>
<code>x</code>	Anzeigen einer Variablen
<code>whos</code>	liefert Informationen über alle definierten Variablen
<code>who a</code>	liefert Informationen über die Variable <code>a</code>
<code>clear</code>	Löschen aller definierten Variablen
<code>clc</code>	Löschen des Kommandofensters
<code>nargin</code>	enthält Anzahl der übergebenen Variablen einer Funktion
<code>[x y]=size(M)</code>	gleichzeitiges Verarbeiten von <i>zwei</i> Rückgabewerten einer Funktion (zu <code>size</code> s.u.)

Beispiele für das Arbeiten mit (Vektoren und) Matrizen

Befehl	Aktion
<code>y=[1 4e-1 -2.7]</code>	Zuweisung eines (Zeilen-)Vektors
<code>sin(y)</code>	liefert Vektor von Funktionswerten an den Stellen <code>y</code>
<code>y(3)</code>	Zugriff auf einzelne Komponenten eines Vektors, Index ≥ 1
<code>length(y)</code>	Länge eines Vektors
<code>0:pi/10:pi</code>	liefert Zeilenvektor mit den Zahlen von 0 bis π in Schritten von $\frac{\pi}{10}$
<code>z=[1 2 3;4 5 6]</code>	Zuweisung einer Matrix, hier 2 Zeilen und 3 Spalten
<code>size(z)</code>	liefert Anzahl Zeilen bzw. Spalten einer Matrix; 2 Rückgabewerte auf einmal! (s.o.)
<code>ones(4)</code>	Einheitsmatrix der entsprechenden Größe
<code>zeros(3)</code>	quadratische Nullmatrix der entsprechenden Größe
<code>zeros(2,3)</code>	Nullmatrix, hier mit 2 Zeilen und 3 Spalten; erzeugt auch Nullvektoren (sind ja auch Matrizen)
<code>sparse(7)</code>	dünn besetzte Matrix (anderes, sparsames Speicherformat); Syntax wie <code>zeros</code>
<code>z(2,3)=-0.7^3</code>	Zugriff auf einzelne Komponenten einer Matrix (Zeile,Spalte)
<code>z(2,:)</code>	Zugriff auf ganze Zeilen-/Spaltenvektoren einer Matrix (hier 2. Zeile)
<code>z(:,1)=[]</code>	Löschen einer Matrixspalte (Zeile analog)
<code>M(2:4,6:10)</code>	Zugriff auf Matrixbereich (Teilmatrix); auch mit beliebigen Indexvektoren: <code>M(1:3,[1 4:6 8])</code>
<code>x'</code>	Transponieren
<code>sum(x)</code>	Addition aller Einträge eines Vektors, bei Matrizen spaltenweise
<code>det(M)</code>	Determinante von <code>M</code>
<code>diag(A)</code>	liefert alle Elemente auf der Hauptdiagonalen von <code>A</code>
<code>A*B</code>	Matrixmultiplikation
<code>[1 2 3].*5</code>	Multiplikation aller Vektoreinträge mit 5, auch für Division, Potenzieren
<code>[1 2].*[3 4]</code>	elementweise Multiplikation von Vektoren (Matrizen)
<code>A\b</code>	liefert Lösung x der Gleichung $Ax = b$ („Division von links“), auch für über-/unterbestimmte Systeme!
<code>inv(M)</code>	Inverse von <code>M</code>
<code>eig(M)</code>	Eigenwerte von <code>M</code> als Spaltenvektor
<code>poly(M)</code>	charakteristisches Polynom (Koeffizienten als Zeilenvektor)
<code>any(x)</code>	liefert 1 (wahr), wenn <code>x</code> mindestens einen Eintrag $\neq 0$ hat, sonst 0 (falsch)
<code>all(x)</code>	liefert 1, wenn alle Einträge von <code>x</code> von 0 verschieden sind, sonst 0
<code>find(x==1)</code>	liefert Indizes aller <code>x(n)</code> , die gleich 1 sind, als Vektor

Grafische Ausgabe

Befehl	Aktion
<code>plot(x,y)</code>	Plottet Vektor y gegenüber Vektor x (2D)
<code>xlabel('x-Achse')</code>	Achsenbezeichnung des aktuellen Plots setzen
<code>hold on</code>	Ermöglicht 2 Plots in einem, danach <code>hold off</code> setzen!
<code>mesh(x,y,z)</code>	Plottet Werte der Matrix z mit Achsen x bzw. y (3D), einfarbig
<code>surf(x,y,z)</code>	wie <code>mesh</code> , nur mit Oberflächenfarbe (3D); Beleuchtung möglich

Zu allen Plot-Befehlen sind viele weitere Optionen möglich, wie z.B. `colormap` (Farbpalette), `hidden on` bzw. `off` (verdeckte Linien) oder zum Ändern von Linienfarbe/-dicke/-typ bzw. Markertyp (siehe MATLAB-Hilfe!).

Ablaufsteuerung

Befehle	Aktion
<code>if A==B</code> <code>M=0</code> <code>elseif isequal(C,[1 2])</code> <code>M=2</code> <code>else</code> <code>M=-1</code> <code>end</code>	Bedingte Ausführung von Befehlen (selbsterklärend); die Verwendung von <code>isequal</code> ist hierbei sauberer als <code>==</code>
<code>for n=3:10</code> <code>i=i*n</code> <code>end</code>	Schleife
<code>switch i</code> <code>case 1</code> <code>j=0</code> <code>case 2</code> <code>j=5</code> <code>otherwise</code> <code>j=10</code> <code>end</code>	Bedingte Ausführung von Befehlen (selbsterklärend); im Gegensatz zu C/C++ kein <code>break</code> erforderlich!

Dateiein-/ausgabe

Befehl	Aktion
<code>save myfile.mat</code>	komplette Sitzung in Datei speichern (MATLAB-Format)
<code>save('c:\temp\myfile2.mat','M')</code>	Variable M in Datei speichern (MATLAB-Format)
<code>save myfile3.mat p -ascii</code>	Variable p in Datei speichern (ASCII-Format), geht <i>nicht</i> für komplexe Zahlen p !
<code>load myfile3.mat</code>	Einlesen (default: MATLAB-Format)
<code>print -depsc myplot.eps</code>	Aktuellen Plot speichern als Farb-EPS; Syntax auch mit <code>()</code> möglich
<code>print -djpeg -f4 myplot.eps</code>	Plot Nr. 4 speichern als 24bit-JPEG; Syntax auch mit <code>()</code> möglich

Weitere Informationen

siehe z.B. `help pde` (PDE-Toolbox)