MATLAB–Spicker (für Version $\geq$ 5.0 / Rel. $\geq$ 11)

Hilfefunktion

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

Allgemeines

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

Beispiele für das Arbeiten mit (Vektoren und) Matrizen

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

Grafische Ausgabe

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

`if A==B`
`  M=0`
`elseif isequal(C,[1 2])`
`  M=2`
`else`
`  M=-1`
`end`
	Bedingte Ausführung von Befehlen (selbsterklärend); die Verwendung von `isequal` ist hierbei sauberer als `==`

`for n=3:10`
`  i=i*n`
`end`
	Schleife

`switch i`
`  case 1`
`  j=0`
`  case 2`
`  j=5`
`  otherwise`
`  j=10`
`end`
	Bedingte Ausführung von Befehlen (selbsterklärend); im Gegensatz zu C/C++ kein `break` erforderlich!

Dateiein-/ausgabe

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

Weitere Informationen

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