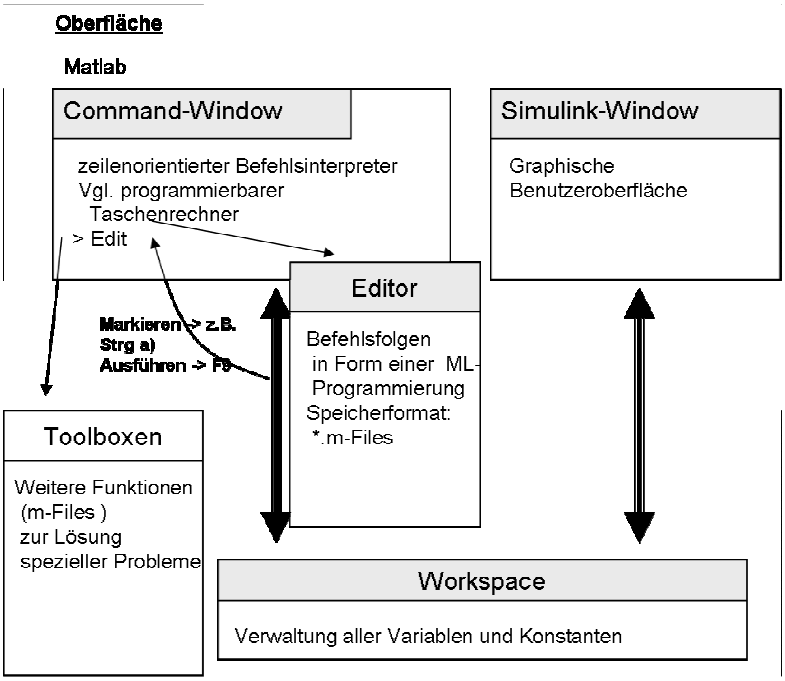
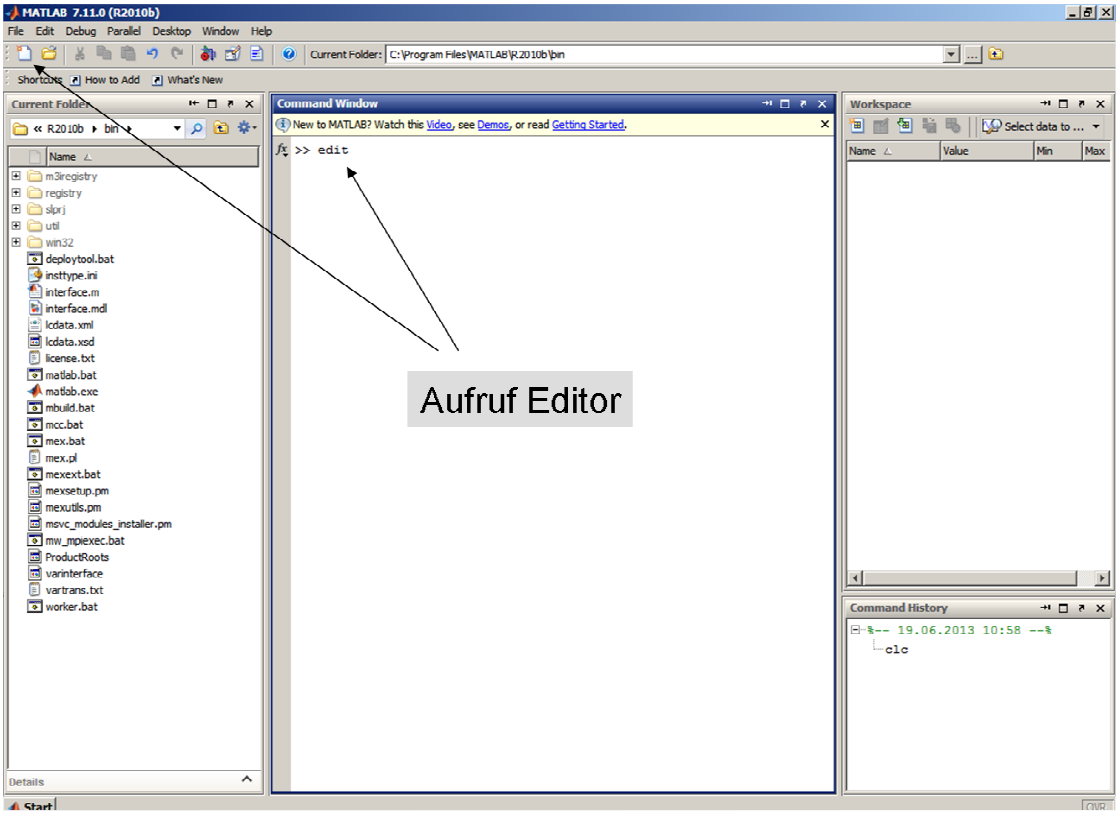
A. Einführung Matlab

A.1. Einführung in das Arbeiten mit Matlab

Beim Aufruf von Matlab-Simulink erscheint zunächst eine Oberfläche, die in das Command-Window, den Workspace und die Command-History eingeteilt ist. Im Command-Window können Befehle eingegeben werden, die Matlab nach Bestätigung mit der Return-Taste interpretiert und ausführt. Variablen und Parameter lassen sich ebenfalls über Eingabe im Command-Window anlegen und definieren. Im Workspace lassen sich Variablen und Parameter verwalten und es stehen alle Informationen zu den jeweiligen Größen zur Verfügung. Das Fenster „Command- History“ zeigt die bislang eingegebenen Befehle, Befehlsfolgen, Programme und Programmteile an.

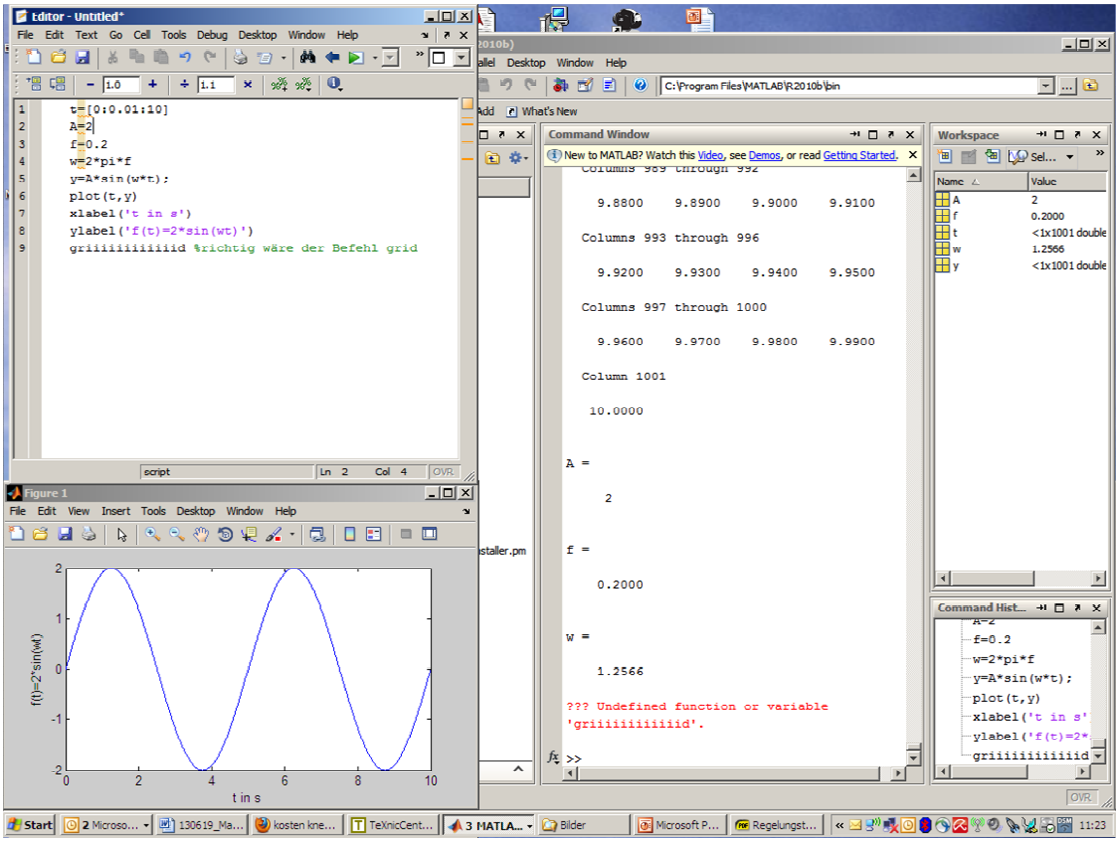


Beim Arbeiten mit Matlab sollten Sie ein ausschließliches Arbeiten mit dem Command-Window (bis auf wenige Ausnahmen) vermeiden. Matlab stellt einen Editor zur Verfügung, in dem Berechnungsfolgen und Programme formuliert, abgespeichert und anschließend in Matlab ausgeführt werden können. Den Editor können Sie entweder durch die Eingabe des Befehls „Edit“ im Command-Window oder durch einen Mausklick auf den Button „Save &Run“ aufrufen.



.

Im Editor beschriebene Anweisungen und Programmteile lassen sich durch Markieren mit dem Mauszeiger und der Tastenkombination „STRG-F9“ oder durch den Icon „Save&Run“ ausgeführen. Beim Arbeiten mit Matlab-Simulink ist es hilfreich, das Editorfenster und das Matlabfenster so anzuordnen, daß ausgeführte Befehle und Rückmeldungen im Command-Window direkt erkennbar sind, ohne die Oberflächen verschieben oder aus dem Hintergrund hervor holen zu müssen.



Bitte achten Sie auf eine vorteilhafte Anordnung der Fenster, so daß Rückmeldungen von Matlab wie z.B. Fehlermeldungen direkt erkennbar sind.

A.1.1. Befehlsübersicht

Matlab stellt eine Vielzahl von Befehlen zur Verfügung. Eine Übersicht über die wichtigsten Befehlskategorien liefert die folgende Aufstellung:

- o Hilfsbefehle
- o Workspace-Befehle
- o Systembefehle
- o Zeichen und Operatoren
- o log. Operatoren
- o exp- und log-Funktionen
- o komplexe Zahlen
- o Matrizen und Vektoren
- o Lineare Algebra
- o Graphiken

Hilfsbefehle und Hilfestellungen

- o help Befehl ... Liefert einen ausführlichen Hilfetext zu dem angegebenen Befehl (z.B.: help plot)
- o lookfor Stichwort ... Durchsucht *.m-files nach dem angegebenen Stichwort

Systembefehle

cd	...	„change directory“ → wechseln eines Ordners
pwd	...	Zeigt den aktuellen Pfad
path	...	Einrichten von Pfaden, auf die Matlab zugreifen kann
dir	...	„directory“ → zeigt den Inhalt des aktuellen Ordner
	dir *.	Zeigt weite Verzeichnisse auf
	dir *.m	Listet Dateien mit der Endung .m auf
delete	...	Löscht eine Datei oder eine Graphik

Workspace-Befehle

who	... Variablen werden aufgelistet
whos	... Ausführliche Auflistung der ML-Variablen (mit Zusatzinfo)
clear	... Variablen werden gelöscht
save	... Variablen werden in einer Datei abgespeichert
load	... Variablen werden in aus einer Datei geladen

Zeichen und Operatoren

()	... Klammern für Argumente
[]	... Klammern für Matrizen
.	... Punkt für Dezimalstelle
,	... Trennzeichen für Spalten einer Matrix
;	... Trennzeichen für eine Zeile
’ ’	... Anführungszeichen für einen Text (Stringsequenz)
%	... Kommentartrennzeichen

Grundrechenarten

+	... Addition
-	... Subtraktion
\	... linke Matrixdivision
/	... rechte Matrixdivision
^	... Exponentialfunktion
*	... Multiplikation
.*	... Elementeweise Multiplikation

Logische Operatoren

&	... Und
|	... Oder
==	... Vergleich auf Gleichheit
<=	... kleiner gleich
>=	... größer gleich
<	... kleiner
.>	... größer

Exponential- und Logarithmusfunktionen

exp	... e-Funktion
log	... natürlicher Logarithmus
sqrt	... Quadratwurzel

Komplexe Zahlen

abs	... Absolutwert einer Zahl
angle	... Phasenwinkel
conj	... konjugiert komplexe Zahl
imag	... Imaginärteil
real	... Realteil

Runden

fix	... Runden in Richtung 0
floor	... Abrunden
ceil	... Aufrunden
round	... Runden
sign	... Vorzeichen bestimmen

Matrixbefehle

zeros(n,m)	... Matrix mit Nullen gefüllt
ones(n,m)	... Matrix mit Inhalt 1 auf jeder Stelle
eye(n)	... Identitätsmatrix
diag([1,2,..,n],0)	... Diagonalmatrix
rand(n,m)	... Zufallszahlen in einer Matrix
linspace(anf,ende,N)	... Linear unterteilter Vektor
meshgrid	... Matrix aus zwei Vektoren zusammengesetzt

Infos über Matrizen

size	... Liefert die Dimension einer Matrix
length	... Liefert die Länge eines Vektors

Lineare Algebra

,	... Transponieren einer Matrix − > Zeilen und Spalten werden vertauscht
inv	... Inverse einer Matrix
cond	... Zustand einer Matrix (Verhältnis zwischen größter und kleinster Singularität)
rank	... Rang einer Matrix
norm	... Norm einer Matrix
det	... Determinante einer Matrix
trace	... Spur
orth	... orthonormierter Basisvektor
eig	... Eigenvektor
svd	... Singularwert

2D-Graphikbefehle

plot	... 2-D-Graphik
subplot	... mehrere 2-D-Graphiken in einem Fenster
semilogx	... 2D-Graphik (einfach logarithmisch auf der x-Achse)
loglog	... 2D-Graphik doppellogarithmisch
polar	... 2D-Graphik in Polarkoordinaten
plotyy	... 2D-Graphik mit zwei Ordinaten
axis	... legt Achsenskalierung fest
zoom	... Vergrößert einen Ausschnitt des Graphikfensters
grid	... Legt ein Liniennetz über das Graphikfenster
box	... Graphik wird in einen Rahmen eingebunden
hold	... Aktuelle Graphik bleibt erhalten und es kann eine zweite Graphik „darüber“ gelegt werden

Graphikbeschriftung

legend(´Text’,’Text’,’...’)	... Legt Legende eines Graphen fest
title(’Bildtitel’)	... Bildüberschrift
xlabel(´...’)	... Bezeichnung der x-Achse
ylabel(‘yy’)	... Bezeichnung der y-Achse
text(x,y,´lkajflsj’)	... Beliebiger auf den xy-KoordinatenText in einer Graphik
gtext(´Text´)	... Beliebiger Text über die Maus platzierbar

3D-Graphik

plot3	... Liniengraphik
mesh	... Netzgraphik
surf	... Oberflächengraphik
fill3	... Polygonfläche

Programmieren

if	... logische Bedingung
for	... Schleife mit einer Vorgegebenen Anzahl an Durchläufen
while	... Schleife mit einem logischen Abbruchkriterium
switch case	... Bedingte Verzweigung

Beispiel:

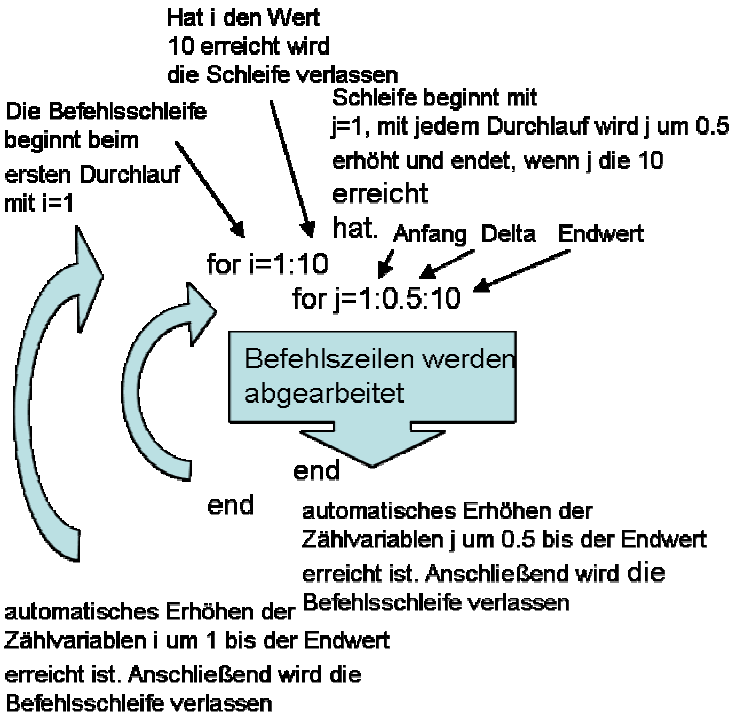
```
t=[0:0.1:10]          % Definition eines Zeitvektors
T=2                   % Periodendauer
a=5*sin(2*pi/T*t);    % Berechnung der Sinusfunktion an den Zeitwerten des Vektors t
b=a;
for i=1:length(t)
if a(i)>0
    a(i)=1;
elseif a(i)< 0
    a(i)=-1;
elseif a(i)==0
    a(i)=0;
end
end
plot(t,a,t,b)

a=b;
i=1
while i< length(a)      % Schleife wird so lange durchlaufen,
    if a(i)<-1               % wie i kleiner ist als der Vorgabewert
        a(i)=-1
    end
    i=i+1
end
    plot(t,a,t,b,'d')
```

Verschachtelte Schleifen

Zur Verarbeitung und Berechnung aller Matrixelemente müssen wir eine Schleifenkonstruktion erzeugen, die es uns erlaubt, Elemente aller Zeilen und aller Spalten anzusprechen.

Beispiel:



A.1.2. Definieren von Skalare, Vektoren und Matrizen

Die Festlegung von Variablen erfolgt durch Zuweisen eines Skalars, eines Vektors, einer Matrix oder einer Rechenoperation zu einem Variablennamen.
Beispiel:
$A = 3$
$B = 2$
$C = A + B$

Bei der Definition der Vektoren oder Matrizen sind die Zahlenwerte in eckigen Klammern zu schreiben. Runde Klammern enthalten nur Argumente von Funktionen oder eine Positionsangabe bei Vektoren oder Matrizen.

Beispiel: Definition einer Matrix

$M = [1, \quad 2, \quad 3; \quad 4 \quad 5 \quad 6; \quad 7 \quad 8 \quad 9]$
$V = [1 \quad 2 \quad 3 \quad 4, \quad 5,6 : 1 : 10 \quad 11 \quad 12 \quad 13]$
Beachte:
Die Trennung von Werten in einer Zeile erfolgt über ein Leerzeichen oder ein Komma. Das Einleiten einer neuen Zeile bewirkt ein Semikolon. Aufeinander folgende Werte mit gleichem Abstand können durch folgende Darstellung kurz und übersichtlich beschrieben werden:
$[Anfangswert : Schrittweite : Endwert]$

Beispiel:
Eingabe von:
$V = \begin{bmatrix} 1, & 2:1:8 & 9 & 10 \end{bmatrix}$
liefert:
$V = \begin{bmatrix} 1 & 2 & 3 & 4, & 5, & 6 & 7 & 8 & 9 & 10 \end{bmatrix}$

Der Aufruf einzelner Elemente oder Bereiche eines Vektors oder einer Matrix lässt sich durch Positionsangabe in runden Klammern realisieren.
Beispiel:
In der Matrix $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$ läßt sich der Wert a_{32} in der dritten Zeile und der zweiten Spalte über die Zuordnung $B = A(3, 2)$ ansprechen. Bei der Reihenfolge ist zu beachten, daß immer zuerst der Zeilenindex und dann die Spaltennummer angegeben werden. Vollständige Zeilen, Spalten oder Teile einer Matrix lassen sich wie folgt zuordnen:

Spaltenzuordnung:
$B = A(:, 3) = \begin{bmatrix} 3 \\ 6 \\ 7 \end{bmatrix}$

Zeilenzuordnung:
$B = A(2, :) = \begin{bmatrix} 4 & 5 & 6 \end{bmatrix}$

Zuordnung von Untermatrizen:
$B = A(2:3, 1:2) = \begin{bmatrix} 4 & 5 \\ 7 & 8 \end{bmatrix}$

A.1.3. Spezielle Vektoren- und Matrixbefehle

Die Funktionen ones(n,m) und zeros(n,m) liefern jeweils eine Matrix mit der Dimension (nxm) gefüllt mit Einsen und Nullen, während die Funktion eye(n) eine Einheitsmatrix der Dimension (nxn) definiert. Achten Sie bei Vektor- und Matrixoperationen darauf, daß die Dimensionen und mathematischen Operationen Sinn ergeben.
Beispiel: Multiplikation eines Zeilenvektors mit einem Zeilenvektor hat mathematisch keine Bedeutung.
$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} * \begin{bmatrix} 4 & 5 & 6 \end{bmatrix}$
Bitte achten Sie bei Rechnungen mit Matlab im Falle von Fehlermeldungen immer auf die Dimensionen der Vektoren und Matrizen, die im Workspace verzeichnet sind.

Transponieren von Matrizen:
$M_T = M'$

Invertieren einer Matrix:
M_{inv} =inv(M)

A.1.4. Graphik-Befehle

plot

Ähnlich wie beim Darstellen eines Funktionsverlaufs mit Bleistift und Papier arbeitet der Plot-Befehl in Matlab. Für die graphische Darstellung einer beliebigen Funktion, z.B. $f(t) = t \cdot 2$ benötigen wir zunächst diskrete Werte für die Zeit t, an denen wir die Funktionswerte für $f(t_i)$ berechnen. Sowohl Zeitwerte als auch Funktionswerte tragen wir anschließend in eine Tabelle ein und entwickeln dann den Graphen, in dem wir über jeden Wert t_i den entsprechenden Wert für $f(t_i)$ auftragen. Abschließend verbinden wir die einzelnen Werte von $f(t)$ miteinander.

Bei der Berechnung und anschließender Darstellung einer Funktion in Matlab arbeiten wir exakt in der gleichen Art und Weise:

1. Festlegung der Zeitwerte:
   $t = [-2 : 1 : 5]$ % Definieren eines Vektors mit den Zeitwerten, an denen die Funktion berechnet werden soll
   $\Rightarrow t = [-2, -1, 0, 1, 2, 3, 4, 5]$

2. Berechnung des Funktionswertes
   $y = 2 \cdot t$ %Multiplikation Skalar * Vektor $\rightarrow$ Vektor
   Ergebnis:$\Rightarrow y =$[-4 -2 0 2 4 6 8 10]

3. Graphische Ausgabe des Ergebnisses
   plot(t,y)

4. Beschriften der Graphik mit den Befehlen xlabel(), ylabel(), zlabel() (nur für 3-D-Grafik), title() und axis([Xmin Xmax Ymin Ymax]).

3D-Graphik

Zur Erstellung von 3D-Graphiken stehen folgende Befehle zur Verfügung:

- mesh ... Graphik mit Gitternetzcharakter
- surf ... Graphik mit geschlossener Hüllkurve
- plot3 ... Darstellung von Punkten die mit x,y,z- Koordinaten angegeben werden