Grundlegende Operationen

x = 3.4	Zuweisung Variabel
;	unterdrückt Ausgabe
,	Trennt Anweisungen/Ergebnis
V(2,:)=[]	Löschen einer Zeile
V(:,3)=[]	Löschen einer Spalte
y = [1:0.5:100.5]	$\vec{y} = (1 \quad 1.5 \quad 2 \quad 2.5 \quad \ldots \quad 100.5)$

Mathematische Funktionen & Operationen

+-*/^	Rechenoperationen
…	nächste Zeile
mod(x,y)	x Modulo y
sqrt(x)	Quadratwurzel
abs(x)	Betrag (absolute value)
sign(x)	Signum (Vorzeichen)
sin(x)	Sinus
cos(x)	Cosinus
tan(x)	Tangens
atan(x)	Arctan

round(x,n)	Runden (n optional)
ceil(x)	Runden nach oben
floor(x)	Runden nach unten
exp(x)	Exponentialfunktion
log(x)	Natürlicher Logarithmus
log10(x)	Zehner-Logarithmus
rad2deg(x)	Umwandlung von Rad nach Grad
deg2rad(x)	Umwandlung von Grad nach Rad
cart2pol	Cartesian coordinates to polar
[rho,theta] = cart2pol(x,y)	Aufpassen Reihenfolge! zuerst φ dann radius r

size(vec)	Grösse (Anzahl Zeilen und Spalten)
help elfun	Liste alle elementaren math. Funktionen
help specfun	Liste spezieller math. Funktionen
help ops	Liste spezieller Operatoren & Zeichen

Konstanten

pi	Kreiszahl π
eps	Fliesskomma-Genauigkeit
inf	unendlich ∞
NaN	Not-a-Number
ans	Standard-Ausgabevariabel
i, j	Imaginäre Einheit $\sqrt{-1}$

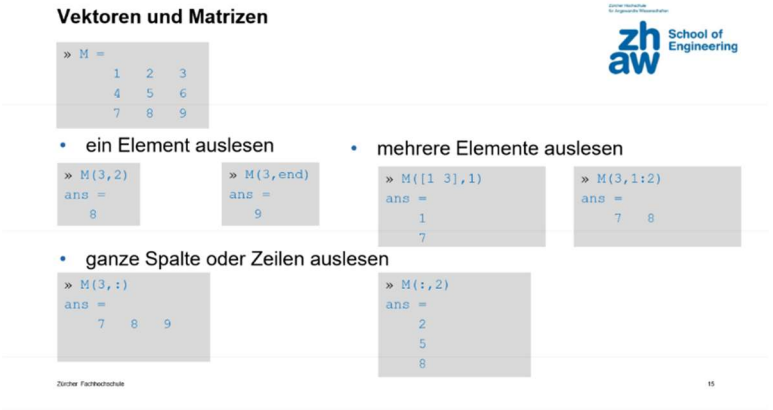
Datentypen und Darstellung

double(x)	Fliesskommazahl (Standard)
single(x)	Fliesskommazahl kompakt
0x9B9A	Zahl in hexadezimal
0b1010	Zahl als binär
Logical(x)	Resultat bei Vergleichsoperatoren
format long	Ausgabeformat lang
format short	Ausgabeformat kurz
format longE	Exponentialformat lang
format shortE	Exponentialformat kurz

Vektoren und Matrizen

[]	Begrenzung
,«SPACE »	Trennt Anweisungen
;	neue Zeile
:	Ausgabe einzelner S/Z
end	Zugriff auf letztes Element

Beispiel

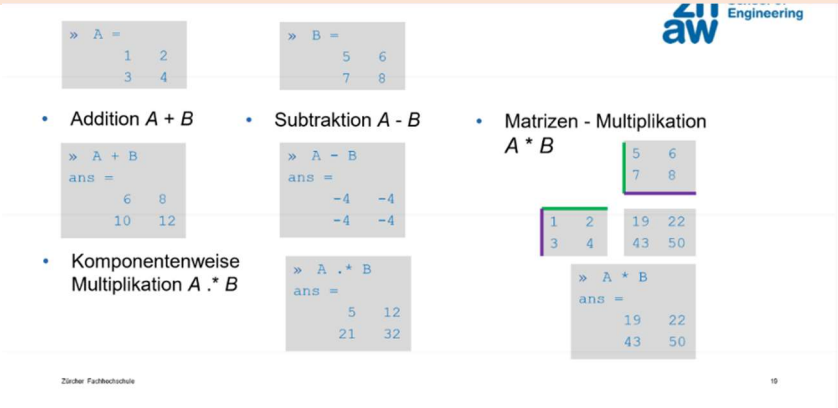


[x1 x2; x3 x4]	Eingabe von Vektoren und Matrizen
start:[schrittweite:]ziel	Doppelpunkt-Operator(erzeugt Zeilenvektor)
linspace(start,ziel,anzahl)	Erzeugung linearer Vektoren
logspace(start,ziel,anzahl)	Erzeugung logarithmischer Vektoren
eye(zeilen)	Einheitsmatrix
ones(zeilen,spalten[,typ])	Matrix mit Einträgen 1
zeros(zeilen,spalten[,typ])	Matrix mit Einträgen 0
rand(zeilen,spalten)	Matrix mit Zufallswerten zwischen 0 und 1
randn(zeilen,spalten)	Matrix mit normalverteilten Zufallswerten

Funktionen & Operatoren für Vektoren und Matrizen

*^\	Vektor/Matrixoperatoren, Linksdivision
.*.^./	Elementweise Operatoren
x.*y	Standardskalarprodukt
matrix.', transpose(matrix)	Transponierte
sortiert = sort(vector)	Sortieren in aufsteigender Reihenfolge
min(vec)	kleinstes Vektorelement
max(vec)	grösstes Vektorelement
mean(vec)	Mittelwert
sum(vec)	Summe der Vektorelemente
inv(m)	Inverse einer Matrix
$\frac{1}{\det(A)}$	Inverse mathematisch
$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - cb$	det(A)
$A * A^{-1} = E_x$	Matrix * Inverse der Matrix = Einheitsmatrix

Beispiel



A = C/B	$C \cdot B^{-1}$	Rechtsdivision
A = C\B	$C^{-1} \cdot B$	Linksdivision
length(vec)	Länge eines Vektors	

Beispiel:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

$$C = A.* Einheitsmatrix$$

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.* \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 9 \end{pmatrix}$$

Vergleichsoperatoren (elementweise)

A < B	kleiner
A >= B	grösser gleich
A == B	gleich
A ~= B	**nicht gleich**
A&B	und
A|B	oder
\A	nicht
xor(A,B)	exklusiv oder
logical	Datentyp vom Resultat bei VO, logisch, class
double	"normal", class

Beispiel:

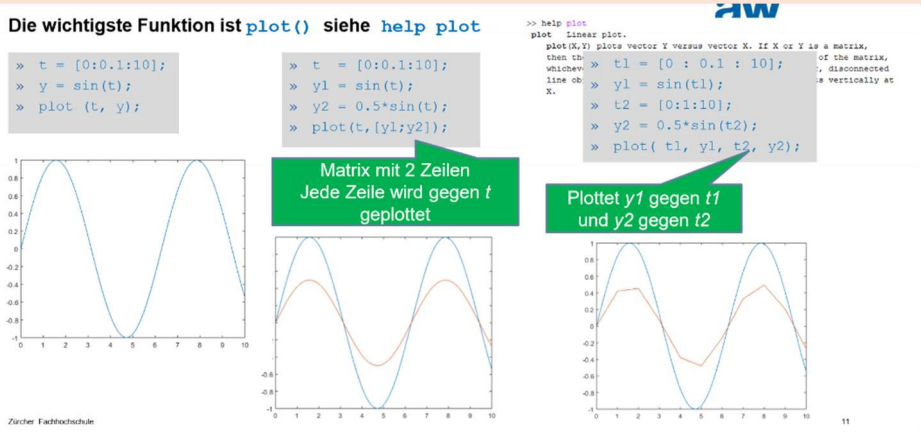
$$C(i,j) = \begin{cases} A(i,j), \text{wenn } A(i,j) \leq 5 \\ -1, \text{ sonst} \end{cases} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{pmatrix}$$

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix}$$

$$C = A$$

$$C(C > 5) = -1$$

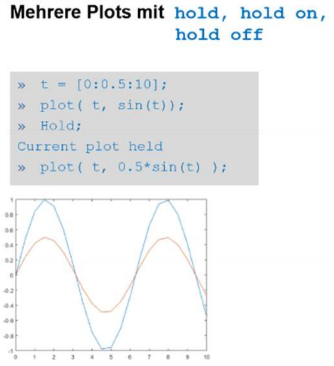
Grafische Funktionen	
help graph2d	Liste aller 2-D Grafikfunktionen
help graph3d	Liste aller 3-D Grafikfunktionen
figure(nummer)	Erzeugt Fenster oder existierendes Fenster an
figure(1)	
close nummer	schliesst jeweilige Figur
close all	schliesst alle Figures
set(gcf,'Color',[1 1 1]);	setzt Hintergrundfarbe der aktiven Figur auf Weiss
hold on	behält Funktion im gleichen Fenster
plot(x,y,s)	plottiert eine Funktion, Kurzbefehl
plot(t, v)	x-Achse: t, y-Achse: v
help plot	Beschreibung der Funktion plot
Beispiel	

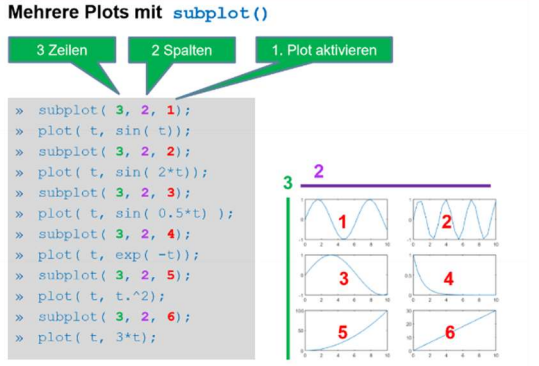


plot(x,y,'ParameterName1', ParameterWert1, 'ParameterName2', ParameterWert2)	
'LineWidth'	0.5 (Default), 1, 1.5
'Color'	[0, 0.5, 0.5] (RGB Triplet)
'LineStyle'	'-', '-.', ':', '-.o'
plot(t3,y3,'-.o','Color', [0, 0.39, 0.65],'LineWidth', 1.5);	

Beschriften von Plots	
title('…')	Titel
title('Diagramm 1', 'FontSize', 14);	
axis(x-Limite y-Limite)	Achsenbeschriftung
axis([0 15 13 25]);	Achsenbeschriftung Limits
xlabel('…')	
xlabel('Zeit', 'FontSize', 12);	
ylabel('…')	
legend('..')	
legend('y3 Signal','Location','SouthWest');	
grid on/off	
if plotting time t Intervall	t = [:0.1:2]

Mehrere plots	
hold	
hold on	behält Funktion im gleichen Fenster
hold off	
subplot()	(Zeilen, Spalten, Nummer subplot)
subplot(2, 1, 1)	2 Zeilen/Graphen, 1 Spalte, (Graph)Nummer 1
Beispiel	





I/O Operationen

help iofun	Liste aller I/O Operationen
save meineDaten	gesamter Workspace im Current Folder speichern
save meineDaten Var1	bestimmte variabel im Current Folder speichern
load meineDaten	Daten im Current Folder laden
save 'Pfad'	
load 'Pfad'	
uiimport	ruft das Importtool auf (Geht auch über das Menü)
xlswrite	Schreiben von Exceldaten
xlsread	Exceldaten laden

MATLAB Scripts & Functions

Scripts

clear	Workspace wird gelöscht
close	Figures werden geschlossen
clc	Command Window wird geleert
% text	Kommentare
%% Titel	Für Titel mit 2 %
help Skriptname	Zeigt ersten Kommentarblock an im Command Win.

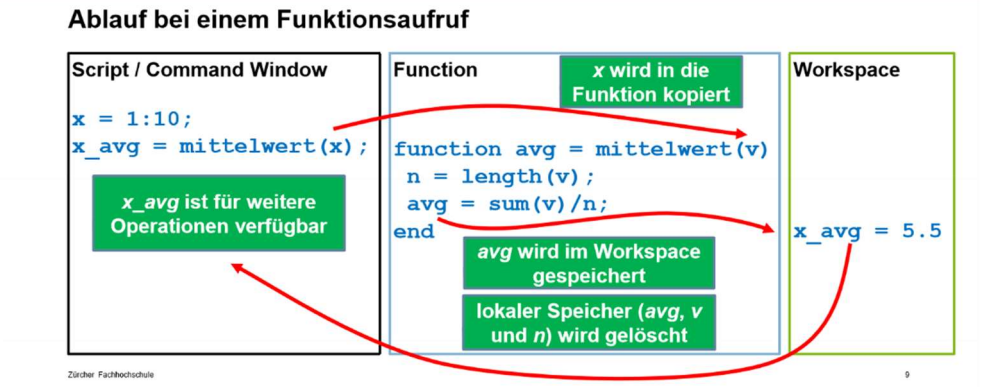
Functions

Lokale function	**Ende des Skripts** definiert, über Funktionsnamen aufgerufen, Variabeln in Funktion bleiben lokal
Separate function	- Separates Skript - **Funktionsname MUSS mit Dateiname übereinstimmen** - im Command Win über ihren Namen aufgerufen

Aufbau Funktion

```
function [outputArg1,outputArg2] = untitled(inputArg1,inputArg2)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
outputArg1 = inputArg1;
outputArg2 = inputArg2;
end
```

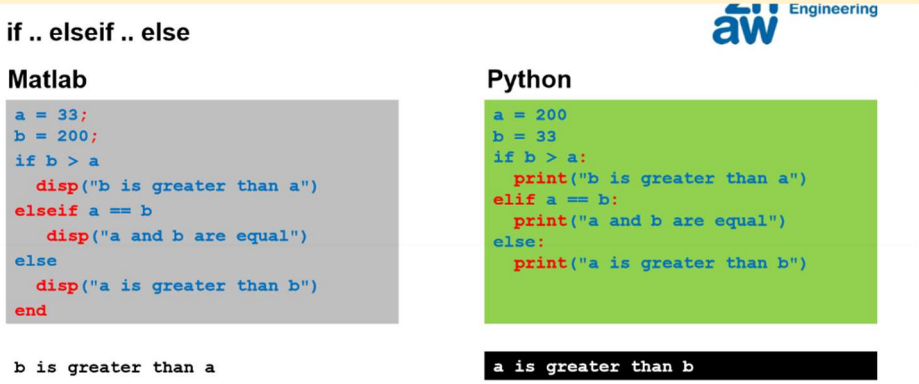
Beispiel Ablauf



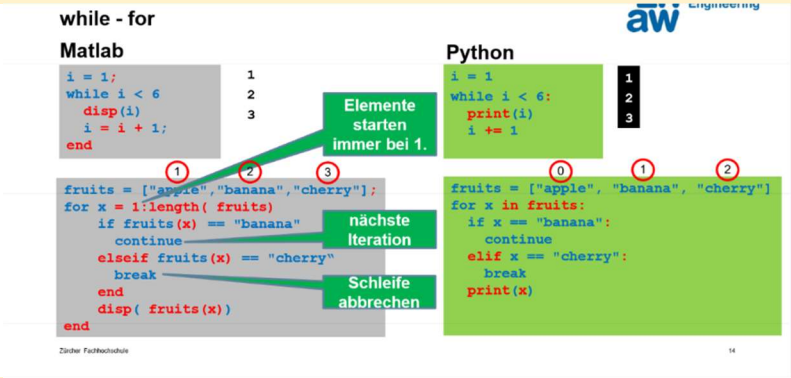
Sprachkonstrukte

help lang Liste aller Sprachkonstrukte

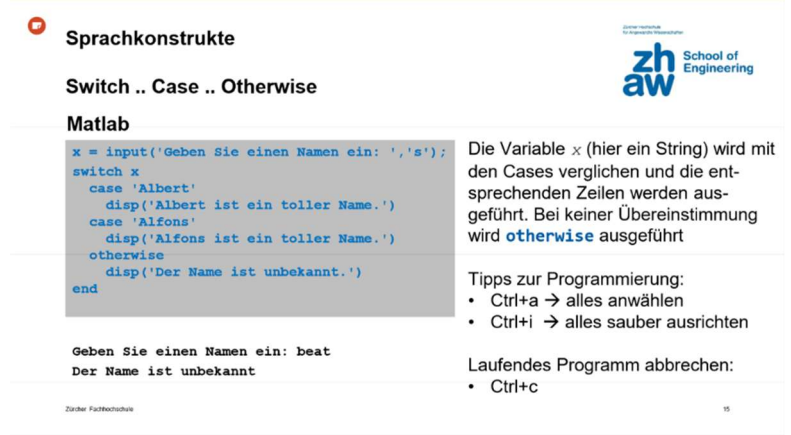
if..else



while loop/for loop



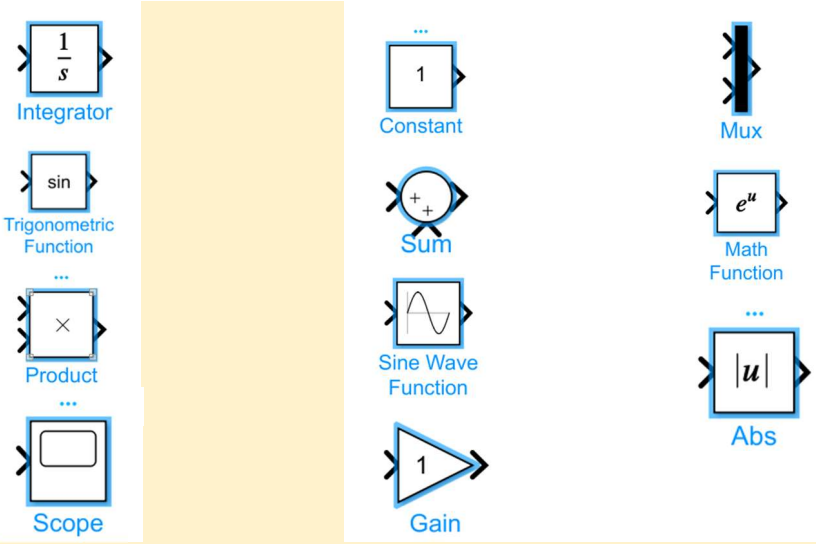
switch case



Beispiel

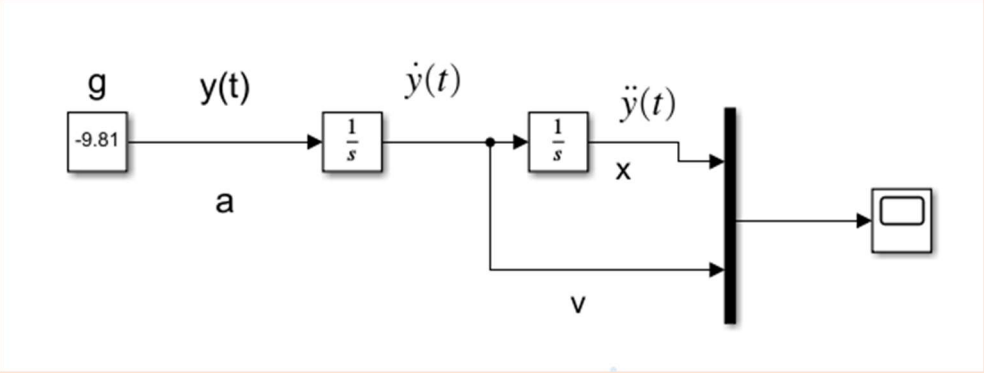
```
function [] = LsgVorzeichen(y)
% Liest ob eine eingegebene Zahl positiv, negativ oder 0 ist

x = input('Geben Sie eine Zahl ein:');

% Vorzeichen bestimmen
v = sign(x);
if floor(x) ~= ceil(x)
    v = 2;
end

switch v
    case 0
        disp('Zahl = 0')
    case -1
        disp('negative Zahl')
    case 1
        disp('postive Zahl')
    otherwise
        disp('keine ganze Zahl')
end
end
```

MATLAB SIMULINK



Sources	für den Anfang/Anfangswert/-block
Sinks	für die Ausgabe/-wert
Math operations	verschiedene math. Operationen
Most commonly used	die meist gebrauchten

Beispiel senkrechter Wurf



Differentialgleichungen

Lösen von gewöhnliche Differenzialgleichungen mit MATLAB

School of Engineering

Beispiel ode23-Funktion Syntax (siehe `doc ode23`) :

```
>> [t, y] = ode23(odefun, tspan, y0)
```

- `odefun`: «function handle» für die ODE $y' = f(t, y)$
- `tspan`: Zeitspanne $[t_0, t_1]$ oder Zeitvektor `t0:dt:t1`
- `y0`: Anfangswert $y_0 = y(t_0)$

Der Function Handle hat im einfachsten Fall die Form:

```
function dy = funktionsname(t, y)
```

Die Funktion muss ein Skalar `t` und einen Spaltenvektor `y` entgegen nehmen und einen Spaltenvektor `dy` zurück geben (mit gleicher Länge wie `y`)

Lösen von gewöhnliche Differenzialgleichungen mit MATLAB: Beispiel 1

School of Engineering

Gegeben: $\dot{y} = 2yt$ mit dem Anfangswert $y(t = -1) = e$
Gesucht: Lösung für $y(t = [-1,1])$

Lösung:

Funktion (Datei)

Aufruf

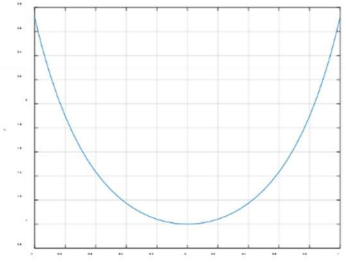
```
function [dy] = dlg1(t, y)
dy = 2 * t * y;
```

Aufruf

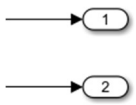
```
>> [t, y] = ode45(@dlg1, [-1, 1], exp(1));
>> plot(t, y);
```

Interval

Anfangswert



simOut = sim('filename') gets output from Simulink file
t = simOut.get('variable name') gets data from Simulink variable
pos = yout{1}.Values.Data; gets data and correlates it with signal
When using: pos = yout{1}.Values.Data; make sure to put these
in the Simulink model, otherwise it won't work:



set_param()
fixed-step gleiche Länge der Vektoren in Modeling – Setup
variable step opposite of fixed-step

Beispiel ODE:

```
t = 0:0.1:25; % Zeitvektor in Sekunden

% Magic Number
MagicNo = 56061

% Lösen der Differentialgleichungen
[~, alpha] = ode45(@pendgl_lin, t, [(19*pi)/20,0]);

function[alphadot]= pendgl_lin(t, alpha)
 % Linearisierte DLG für Pendel
 l=10;              % Pendellänge in m
 g=9.81;            % Erdbeschleunigung in m/(s^2)
alphadot=[0;0];     % Initialisierung

alphadot(1) = alpha(2);
alphadot(2) = -(g/l)*(alpha(1));
end
```

m = [m m*2 m*4]; Modell simulieren mit 3 verschiedenen Massen

```
% iteriere über alle Massen und simuliere das System
for k=1:length(m)
    % ändern der Masse im Modell
    open('SW7_Example3');
    set_param('SW7_Example3/mass', 'Value', num2str(m(k)));

    % Resultate auspacken
    t = simOut.get('tout');
    yout = simOut.get('yout');
    vel = yout{2}.Values.Data;

    % anhängen des neuen Resultats
    y = [y, vel];
end
```

```
clear
clc

t1 = (0:0.1:2.5)';
u1 = 2*t1;
u1(t1>1) = 2;
R = 10*10^3;
C = 47e-6;

C = [C, C*2, C*4];

u = [];

for k=1:length(C)
    open('SW7_Example6');
    set_param('SW7_Example6/Gain', 'Gain', num2str(1/(R*C(k))));
    set_param('SW7_Example6/Gain1', 'Gain', num2str(1/(R*C(k))));

    % Resultate auspacken
    simOut = sim('SW7_Example6');
    t = simOut.get('tout');
    uout = simOut.get('uout');

    % anhängen des neuen Resultats
    u = [t, uout];
end


plot(t1, u1, t, u);
legend('u with C', 'u with 2*C', 'u with 4*C', 'u1(t1)');
xlabel('Zeit in sec');
ylabel('Ausgangsspannung in V');
```

Frage 4

Repetition: Lösen von Differentialgleichungen.

DGL 1. Ordnung:

1. DGL auflösen nach Ableitung: $\dot{x} = -\frac{a}{2}x + \frac{25}{2}t$
2. MATLAB Function erstellen mit der DGL
   Muss folgende Form haben:

```
function dx = funktionsname(t, x)
   Berechnen von dx anhand von t und x
end
```

3. Diese Funktion mit dem geeigneten Solver aufrufen:

```
[t, x] = ode23(@funktionsname, intervall, startwert)
```

$2\dot{x} + ax = 25t$

$mit\ x(0) = 7\ und\ a = 6$

Lösen im Bereich t = 0 … 10s

Frage 4

DGL 2. Ordnung:

1. DGL 2. Ordnung muss in ein DGL-System von 1. Ordnung gewandelt werden:
   Ansatz:
   $y_1 = x\ und\ y_2 = \dot{x}$

   In Form $\dot{y}_1 = f(y_1, y_2)\ und\ \dot{y}_2 = f(y_1, y_2)$ bringen:
   $\dot{y}_1 = \dot{x} = y_2$
   $\dot{y}_2 = \ddot{x} = \frac{8}{6}x + \frac{2}{6}\dot{x} = \frac{8}{6}y_1 + \frac{2}{6}y_2$

   Das DGL-System lautet dann:

$$\begin{bmatrix}\dot{y}_1\\ \dot{y}_2\end{bmatrix} = \begin{bmatrix}y_2\\ {}^{8}/_{6}\,y_1 + {}^{2}/_{6}\,y_2\end{bmatrix}$$

2. MATLAB Function erstellen mit dem DGL-System
   Muss folgende Form haben:

```
function dy = funktionsname(t, y)
   Berechnen von dy anhand von t und y
end
```

Sind jetzt Vektoren:

dy = $\begin{bmatrix}\dot{y}_1\\ \dot{y}_2\end{bmatrix}$ d.h. dy(1) = $\dot{y}_1$ und dy(2) = $\dot{y}_2$

y = $\begin{bmatrix}y_1\\ y_2\end{bmatrix}$ d.h. y(1) = y_1 und y(2) = y_2

$6\ddot{x} - 2\dot{x} = 8x$

$mit\ x(0) = 7\ und\ \dot{x}(0) = -2$

Lösen für Bereich t = 0:0.1:5

Frage 4

DGL 2. Ordnung:

3. Diese Funktion mit dem geeigneten Solver aufrufen:

```
[t, y] = ode23(@funktionsname, intervall, startwerte)
```

Ist jetzt ein Vektor mit y1(0) und y2(0)

$6\ddot{x} - 2\dot{x} = 8x$

$mit\ x(0) = 7\ und\ \dot{x}(0) = -2$

Lösen für Bereich t = 0:0.1:5

Frage 4

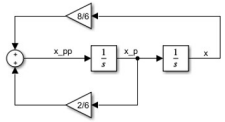
DGL in Simulink:

1. DGL auflösen nach höchster Ableitung:

$\ddot{x} = \frac{8}{6}x + \frac{2}{6}\dot{x}$

2. Integratorblöcke einfügen, verbinden, Initialwerte setzten und evtl. beschriften:

3. Restliche Blöcke einfügen:



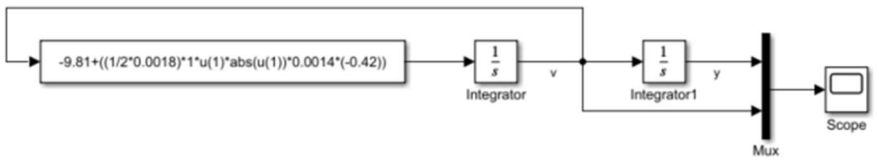
$6\ddot{x} - 2\dot{x} = 8x$

$mit\ x(0) = 7\ und\ \dot{x}(0) = -2$

Simulink – Fcn und Matlab Function Block

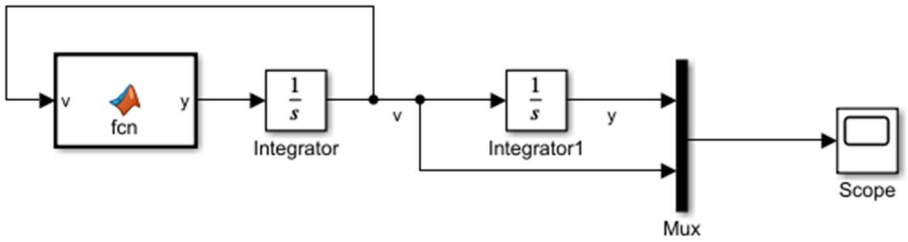
Fcn Block Beispiel:

IMPORTANT: use variable u! (here instead of v)

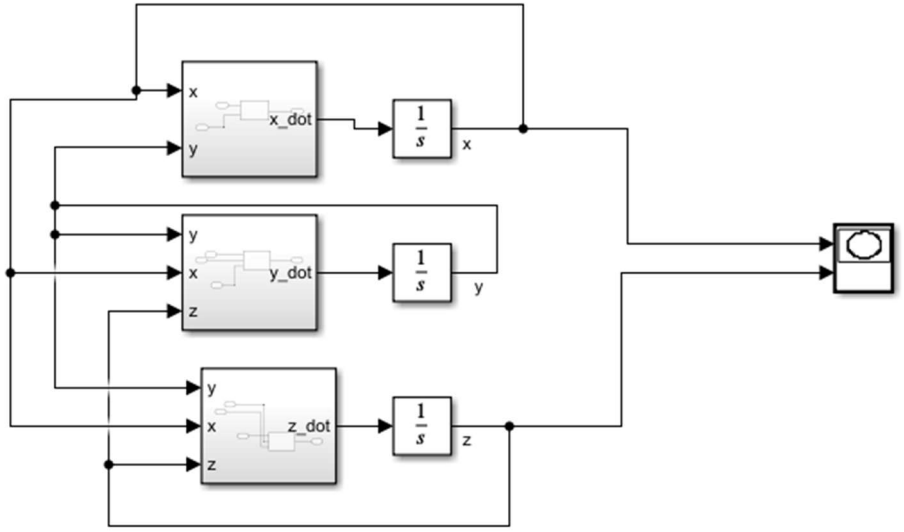


Matlab Function Beispiel

v = input argument, y = output argument



Subsystem Beispiel Lorenz attractor



Symbolic Math Toolbox

help symbolic

syms	creates symbolic variable
syms x y z	creates x y z as symbolic variable
pretty	pretty print a symbolic expression, visualisierung
pretty(f)	visualiesiert die function f
whos	zeigt variable list aufg
diff	differentiate, ableiten, differentiates auto for x
diff(f,y)	differentiates to y
dfg = diff(f,y)	correlate variable !
dfg2 = diff(f,y,2)	2. derivative, zweite Ableitung

help diff

int	integrate, integrieren
simplify()	algebraic simplification

simplify(a | (a & b),'steps', 10)

expand	expand
solve	solve equation
dsolve	differential equations solver
collect	collect coefficients
combine	combine terms of identical tructure
factor	factorization
simple	search for shortest form
isolate	isolate variable or expression in equation
subs	macht symbolische werte zu numerischen

Beispiel

```
»   syms x y v
»   f = sin ( x * y ^2)* cos ( v * x * y );
»   wert = subs( f, {x, y, v}, {1, 2, -1})
cos(2)*sin(4)                    % symbolisches Ergebnis
»   eval( wert)                  % Auswertung der symbolischen Darstellung auch mit double möglich
0.3149
```

Algebraische Gleichungen mit mehreren symbolischen Variablen lösen

```
»   syms x y
»   solve(6*x^2 - 6*x^2*y + x*y^2 - x*y + y^3 - y^2 == 0, y)
1
2*x
-3*x
```

```
»   syms x y z
»   [x, y, z] = solve(z == 4*x, x == y, z == x^2 + y^2)
x =
 0
 2
y =
...
```

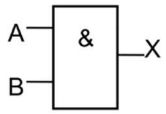
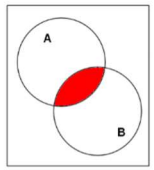
Boolsche Algebra

n — Anzahl Eingangssignale

2^n — Anzahl Ausgangssignale, Zeilen Wahrheitstabelle

& — und/ and

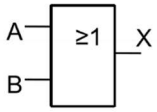
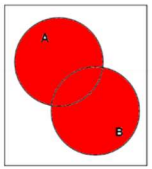
wenn alle Eingang = 1 → Ausgang =1

Symbol	Formel	Wahrheitstabelle	Mengenlehre
	$X = A \wedge B$ $X = A \cdot B$ $X = AB$ $X = A \,\&\, B$	A B X 0 0 0 0 1 0 1 0 0 1 1 1	

Operation gleichbedeutend mit der Standardmultiplikation

| — oder / or

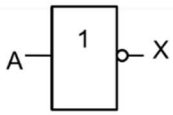
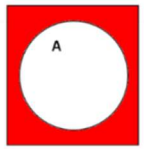
wenn min. 1 Wert = 1 → Ausgang = 1

Symbol	Formel	Wahrheitstabelle	Mengenlehre
	$X = A \vee B$ $X = A + B$ $X = A \mid B$	A B X 0 0 0 0 1 1 1 0 1 1 1 1	

Operation gleichbedeutend mit der Standardsumme

~ — nicht/ not

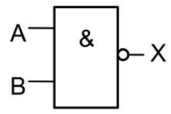
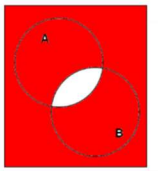
wenn Eingang = 0 → Ausgang =1

Symbol	Formel	Wahrheitstabelle	Mengenlehre
	$X = \bar{A}$ $X = \sim B$	A X 0 1 1 0	

Logic and Bit operations — Simulink Blockbereich

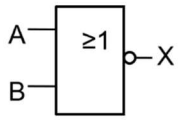
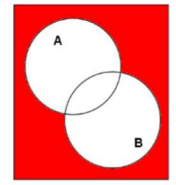
~(A&B) — nicht und / nand/ not and

wenn alle Eingang = 1 → Ausgang =1

Symbol	Formel	Wahrheitstabelle	Mengenlehre
	$X = \overline{A \wedge B}$ $X = \overline{A \cdot B}$ $X = \overline{AB}$ $X = \sim(A\&B)$	A B X 0 0 1 0 1 1 1 0 1 1 1 0	

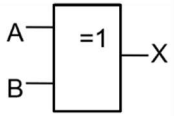
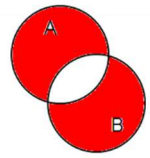
~(A|B) — nicht oder /nor / not or

wenn alle Eingang = 0 → Ausgang = 1

Symbol	Formel	Wahrheitstabelle	Mengenlehre
	$X = \overline{A \vee B}$ $X = \overline{A + B}$ $X = \sim(A\|B)$	A B X 0 0 1 0 1 0 1 0 0 1 1 0	

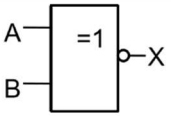
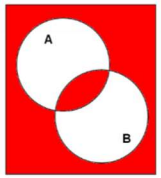
xor(A,B) — nur oder/ xor/ exclusive or

wenn alle Eingang = unterschiedlich → Ausgang = 1

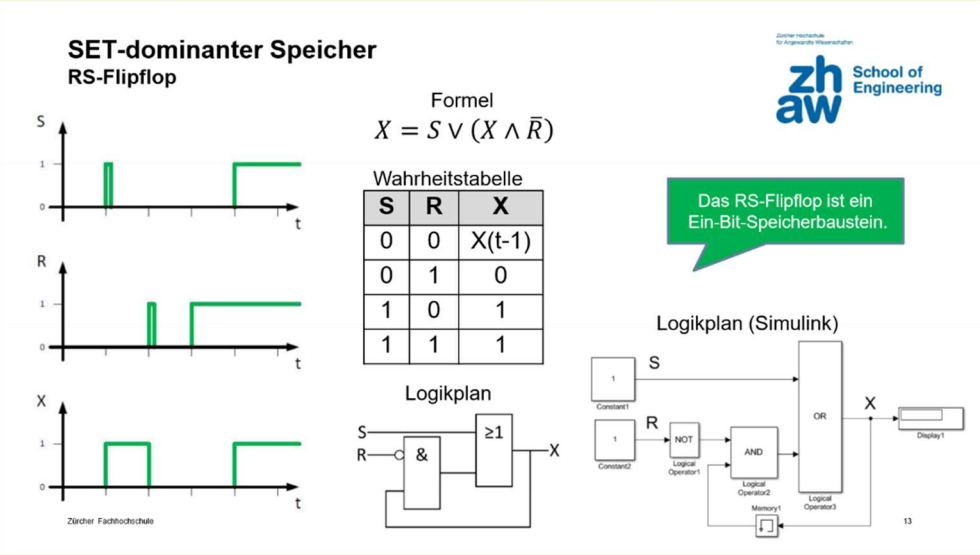
Symbol	Formel	Wahrheitstabelle	Mengenlehre
	$X = A \oplus B$ $X = xor(A,B)$	A B X 0 0 0 0 1 1 1 0 1 1 1 0	

~xor(A,B) — nicht nur oder/ xnor/ exclusive not or

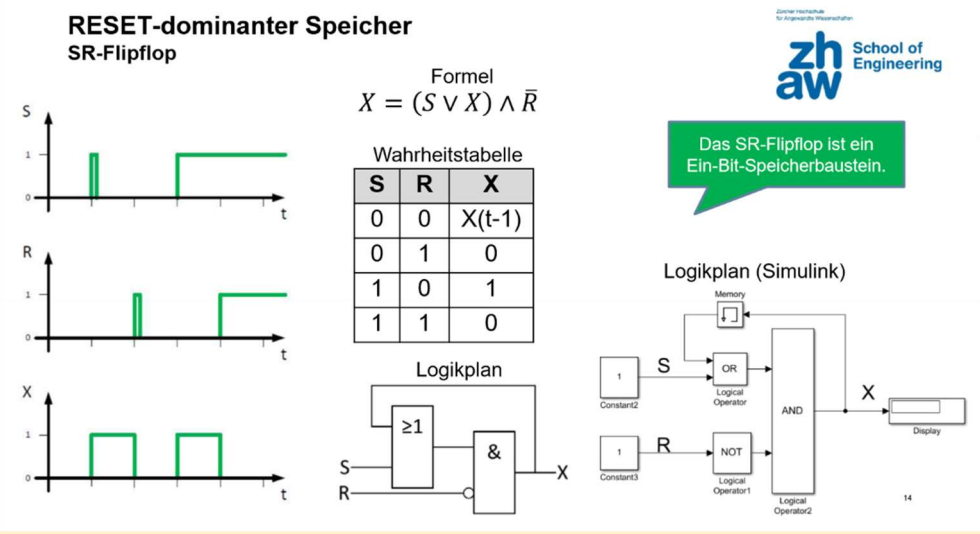
wenn alle Eingang = gleich → Ausgang =1

Symbol	Formel	Wahrheitstabelle	Mengenlehre
	$X = \overline{A \oplus B}$ $X = \sim xor(A,B)$	A B X 0 0 1 0 1 0 1 0 0 1 1 1	

SET-dominanter Speicher



RESET-dominanter Speicher



Logische Rechenregeln

Nr	Gesetz	Rechenregel
1	Identitätsgesetz	$A \cdot 0 = 0$
2		$A \cdot 1 = A$
3	Komplementgesetz	$A \cdot A = A$
4		$A \cdot \bar{A} = 0$
5	Identitätsgesetz	$A + 0 = A$
6		$A + 1 = 1$
7	Komplementgesetz	$A + A = A$
8		$A + \bar{A} = 1$
9	Kommutativgesetz	$\bar{\bar{A}} = A$
10		$A \cdot B = B \cdot A$
11		$A + B = B + A$

Nr	Gesetz	Rechenregel
12	Assoziativgesetz	$(A \cdot B) \cdot C = A \cdot (B \cdot C) = A \cdot B \cdot C$
13		$(A + B) + C = A + (B + C) = A + B + C$
14	Distributivgesetz	$(A \cdot B) + (A \cdot C) = A \cdot (B + C)$
15		$(A + B) \cdot (A + C) = A + (B \cdot C)$
16	Allg. Distributivgesetz	$(A + B)(C + D) = AC + AD + BC + BD$
17		$AB + CD = (A + C)(A + D)(B + C)(B + D)$
18	Absorbtionsgesetz	$A \cdot (A + B + C) = A$
19		$A + (A \cdot B \cdot C) = A$
20	De Morgan	$\overline{A \cdot B} = \bar{A} + \bar{B}$
21		$\overline{A + B} = \bar{A} \cdot \bar{B}$

Vereinfachung logischer Ausdrücke

syms S1 S2 S3

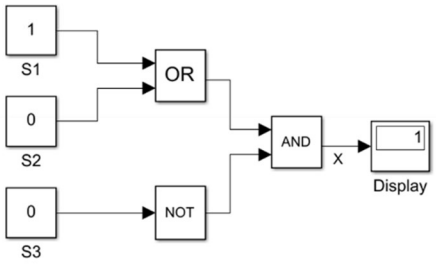
X = simplify((~S1 & S2 & ~S3)| (S1 & ~S2 & ~S3)|(S1 & S2 & ~S3))

Logik mit Simulink

Einfaches Beispiel einer Logik mit Simulink

Umsetzung mit Simulink eines logischen Ausdrucks:

```
X = ~S3 & (S1 | S2)
```

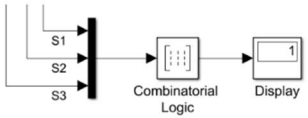


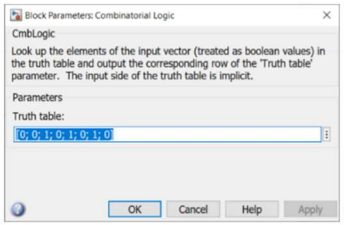
Einfaches Beispiel einer Logik mit Simulink

Combinatorial Logic Block für `X = ~S3 & (S1 | S2)`

Wahrheitstabelle in Matlab:
[0 0 0 **0**; 0 0 1 **0**; 0 1 0 **1**; 0 1 1 **0**; 1 0 0 **1**; 1 0 1 **0**; 1 1 0 **1**; 1 1 1 **0**]

Der *Combinatorial Logic Block* liest S1, S2 und S3 ein und gibt anhand der letzten Spalte den Ausgabewert.



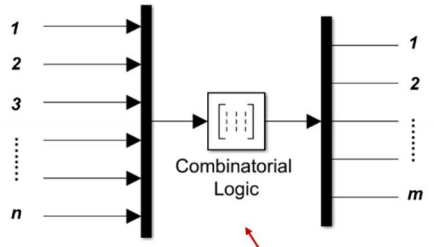


S1	S2	S3	X
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Combinatorial Logic allgemein

Die Wahrheitstabelle muss zwingend diese Struktur haben!

Die allgemeine Implementation der Wahrheitstabelle:



n	…	*3*	*2*	*1*	*m*	…	*2*	*1*
…	…	0	0	0	…	…	…	…
…	…	0	0	1	…	…	…	…
…	…	0	1	0	…	…	…	…
…	…	0	1	1	…	…	…	…
…	…	⋮	⋮	⋮	⋮	⋮	⋮	⋮

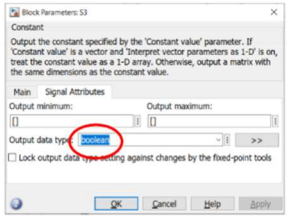
Eingänge sind fix in dieser Struktur gegeben, müssen nicht programmiert werden.

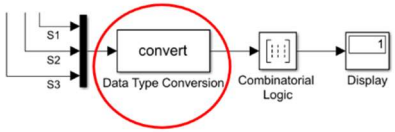
Ausgänge müssen im Combinatorial Logic Block manuell eingegeben werden.

Datentypen in Simulink

Grundsätzlich funktionieren die Logikblöcke von Simulink, auch wenn die Quellen, z.B. Konstanten, mit dem Datenformat double sind.

Eine Ausnahme hierbei ist der Combinatorial Logic Block. Sie haben hierbei zwei Möglichkeiten:
1. Sie definieren Ihre Quellen explizit als boolean.
2. Sie verwenden einen Data Type Conversion Block, der ihre Signale automatisch umwandelt.





Open-Source Alternativen

Octave	interaktive Skriptsprache, “kompatibel“ mit MATLAB für numerische Berechnungen, „kompatibel“ zu MATLAB (Xcos alternative zu Simulink)
Scilab	