

Matlab: Grundlagen und Hilfen

Jan Marc Stockschläder

Fachbereich Physik
TU Kaiserslautern

16.08.2012

- 1 Grundlagen
 - Hilfreiche Funktionen
 - Grundrechenarten
 - Matrizen
 - Graphen
 - Statistik
 - Funktionen
 - Schleifen & Abfragen
 - Speichern & Laden
- 2 Erweiterte Themen
 - Komplexere Rechnungen
 - Daten umschreiben
 - Symbolisches rechnen
 - Matlab-Datenbank

MATLAB 7.12.0 (R2011a)

File Edit Debug Desktop Window Help

Current Folder: /home/maitre/

Shortcuts How to Add What's New

Current Folder

- home
- maitre
 - meinsinus.m
 - Vortrag.mat
 - meinsinus.m-

Details

Select a file to view details

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

```
% >>
```

Workspace

Name	Value	Min	Max

Command History

Start Ready

clear all	löscht alle Variablen
clc	löscht das „Command Window“
help	Hilfe zu einem Befehl

$a+b$	$a + b$	$a-b$	$a - b$
$a*b$	$a * b$	a/b	$\frac{a}{b}$
a^b	a^b	$\text{sqrt}(a)$	\sqrt{a}
$\text{mod}(a,b)$	$a \bmod b$	$\text{div}(a,b)$	$a \div b$
$\text{exp}(b)$	e^b		
$\text{log}(a)$	$\ln(a)$	$\text{log}_{10}(a)$	$\log_{10}(a)$
$\text{sin}(a)$	$\sin(a)$	$\text{asin}(a)$	$\sin^{-1}(a)$
pi	π	i	i

$$A=[1 \ 2 \ 3]$$

$$B=[1 \ 2 \ 3].'$$

$$x=\text{linspace}(-\pi,\pi,1001)$$

$$k=1:3$$

$$l=1:.25:2$$

$$m=2:-.5:1$$

$$A = (1 \ 2 \ 3)$$

$$B = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

$$x = \left(-\pi \quad -\pi \cdot \frac{499}{500} \quad \dots \quad -\pi \cdot \frac{499}{500} \quad \pi \right)$$

$$k = (1 \ 2 \ 3)$$

$$l = (1 \ 1,25 \ 1,5 \ 1,75 \ 2)$$

$$m = (2 \ 1,5 \ 1)$$

$$\begin{array}{l}
 C=[1 \ 2 \ 3;4 \ 5 \ 6;7 \ 8 \ 9] \quad C = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \\
 D=\text{ones}(m,n) \quad D = \begin{pmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{pmatrix} \\
 E=\text{zeros}(m,n) \quad D = \begin{pmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{pmatrix} \\
 F=\text{eye}(m,n) \quad F = \begin{pmatrix} 1 & 0 & \dots \\ 0 & 1 & 0 \\ \vdots & 0 & \ddots \end{pmatrix}
 \end{array}$$

$$A \cdot B \quad (1 \ 2 \ 3) * \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = 14$$

$$B \cdot A \quad \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} * (1 \ 2 \ 3) = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{pmatrix}$$

$$C(i,j) \quad c_{ij}$$

$$A \cdot A \quad a_i * a_i = (1 \ 4 \ 9)$$

$$A \cdot A^2 \quad (1^2 \ 2^2 \ 3^2) = (1 \ 4 \ 9)$$

$$[B \ B] \quad \begin{pmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{pmatrix}$$

$$\text{eig}(C) \quad \text{Eigenwerte} \begin{pmatrix} 16.1168 \\ -1.1168 \\ -0.0000 \end{pmatrix}$$

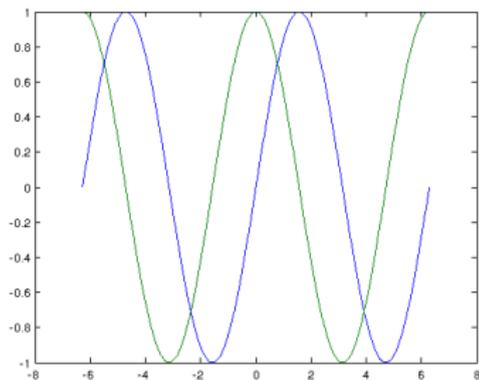
$$[V, D] = \text{eig}(C) \quad V: \text{Matrix aus den Eigenvektoren}$$

$$V = \begin{pmatrix} -0.2320 & -0.7858 & 0.4082 \\ -0.5253 & -0.0868 & -0.8165 \\ -0.8187 & 0.6123 & 0.4082 \end{pmatrix}$$

D: Diagonalmatrix mit Eigenwerten

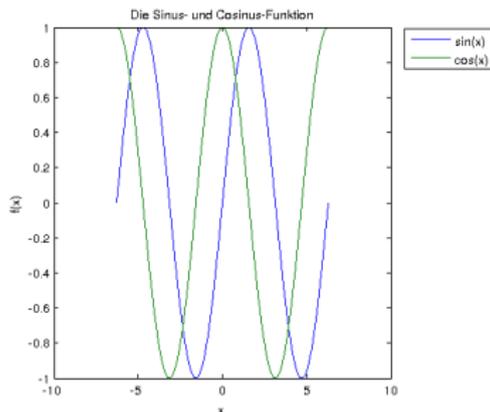
$$D = \begin{pmatrix} 16.1168 & 0 & 0 \\ 0 & -1.1168 & 0 \\ 0 & 0 & -0.0000 \end{pmatrix}$$

```
1 x=linspace(-2*pi,2*pi,1000);  
2 y=sin(x);  
3 y2=cos(x);  
4 plot(x,y,x,y2)
```

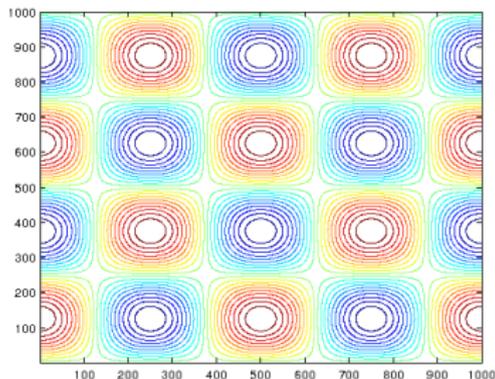


```

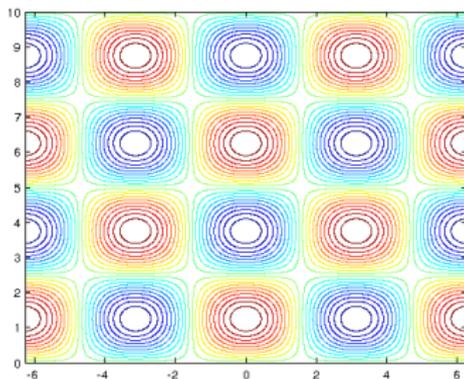
1 x=linspace(-2*pi,2*pi,1000);
2 y=sin(x);
3 y2=cos(x);
4 plot(x,y,x,y2)
5 xlabel('x')
6 ylabel('f(x)')
7 title('Die Sinus- und Cosinus-
      Funktion')
8 legend('sin(x)', 'cos(x)',
      Location, 'NorthEastOutside')
  
```



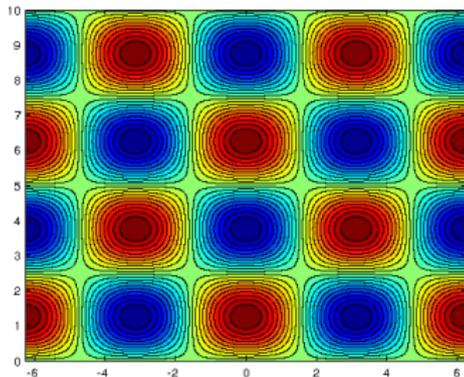
```
1 x=linspace(-2*pi,2*pi,1000);  
2 x2=linspace(0,10,1000);  
3 y=sin(x);  
4 y2=cos(x);  
5 z=y'*y2;  
6 contour(z,20)
```



```
1 x=linspace(-2*pi,2*pi,1000);  
2 x2=linspace(0,10,1000);  
3 y=sin(x);  
4 y2=cos(x);  
5 z=y'*y2;  
6 contour(x,x2,z,20)
```

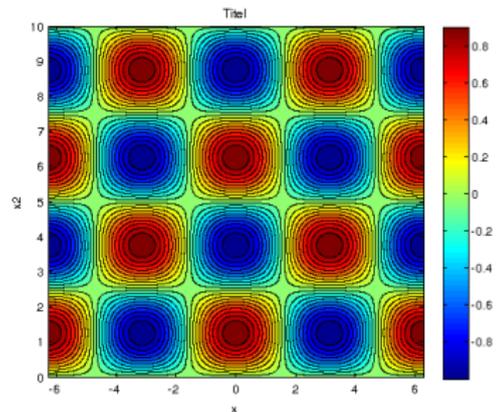


```
1 x=linspace(-2*pi,2*pi,1000);  
2 x2=linspace(0,10,1000);  
3 y=sin(x);  
4 y2=cos(x);  
5 z=y'*y2;  
6 contourf(x,x2,z,20)
```



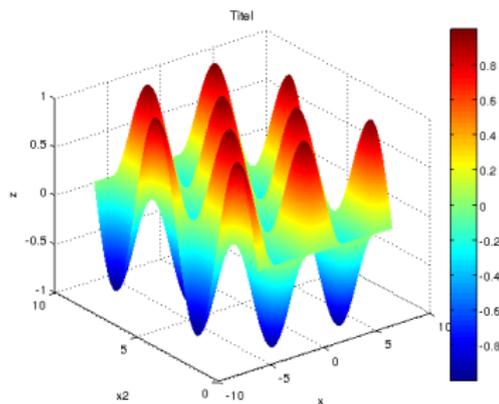
```

1 x=linspace(-2*pi,2*pi,1000);
2 x2=linspace(0,10,1000);
3 y=sin(x);
4 y2=cos(x);
5 z=y'*y2;
6 contourf(x,x2,z,20)
7 xlabel('x')
8 ylabel('x2')
9 title('Titel')
0 colorbar
  
```



```

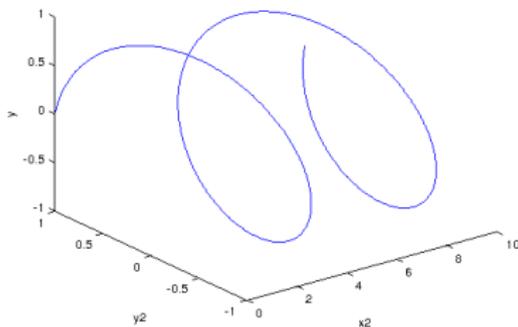
1 x=linspace(-2*pi,2*pi,1000);
2 x2=linspace(0,10,1000);
3 y=sin(x);
4 y2=cos(x);
5 z=y'*y2;
6 mesh(x,x2,z)
7 xlabel('x')
8 ylabel('x2')
9 zlabel('z')
0 title('Titel')
1 colorbar
  
```



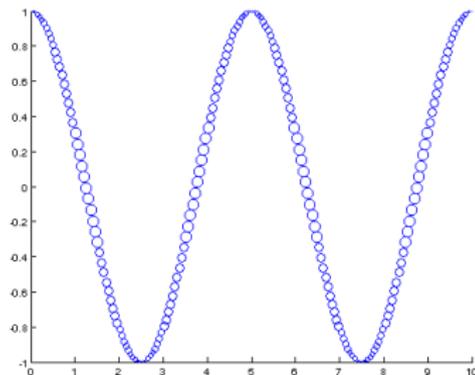
```

1 x=linspace(-2*pi,2*pi,1000);
2 x2=linspace(0,10,1000);
3 y=sin(x);
4 y2=cos(x);
5 plot3(x2,y2,y)
6 xlabel('x2')
7 ylabel('y2')
8 zlabel('y')

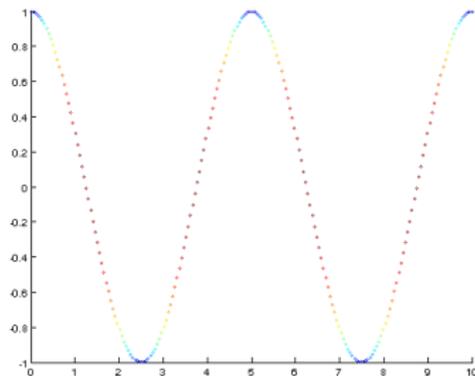
```



```
1 x=linspace(-2*pi,2*pi,200);  
2 x2=linspace(0,10,200);  
3 y=sin(x);  
4 y2=cos(x);  
5 scatter(x2,y2,100*abs(y))  
6 xlabel('x2')  
7 ylabel('y2')
```



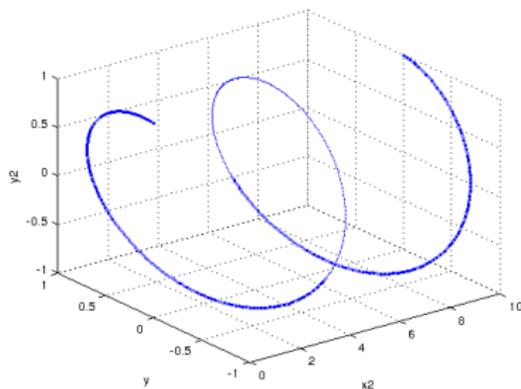
```
1 x=linspace(-2*pi,2*pi,200);  
2 x2=linspace(0,10,200);  
3 y=sin(x);  
4 y2=cos(x);  
5 scatter(x2,y2,5,y)  
6 colorbar  
7 xlabel('x2')  
8 ylabel('y2')
```



```

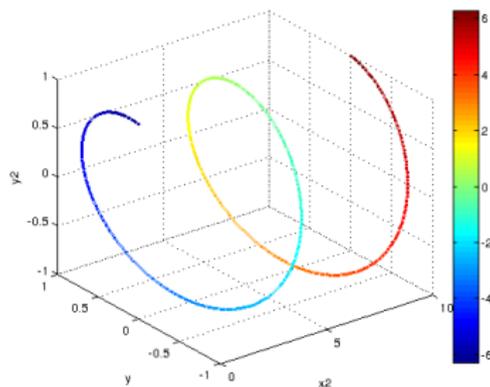
1 x=linspace(-2*pi,2*pi,1000);
2 x2=linspace(0,10,1000);
3 y=sin(x);
4 y2=cos(x);
5 scatter3(x2,y,y2,abs(x))
6 xlabel('x2')
7 ylabel('y')
8 zlabel('y2')

```



```

1 x=linspace(-2*pi,2*pi,1000);
2 x2=linspace(0,10,1000);
3 y=sin(x);
4 y2=cos(x);
5 scatter3(x2,y,y2,5,x)
6 xlabel('x2')
7 ylabel('y')
8 zlabel('y2')
9 colorbar
    
```



hold on / off

figure

figure(#)

clf

axis([xmin xmax ymin ymax])

Graph überschreiben aus / an
neuer Graph

auf Graph # zugreifen

Graph löschen

Achsenlimit setzen

<code>max()</code>	Maximum
<code>min()</code>	Minimum
<code>mean()</code>	Mittelwert
<code>std()</code>	Standardabweichung
<code>var()</code>	Varianz
<code>hist()</code>	Histogramm

MATLAB 7.12.0 (R2011a)

File Edit Debug Desktop Window Help

Current Folder: /home/maitre/

Shortcuts How to Add What's New

Current Folder

- home
- maitre
 - Desktop
 - profile
 - Vortrag
 - mein_sinus.m
 - Vortrag.mat
 - mein_sinus.m-

Details

Select a file to view details

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

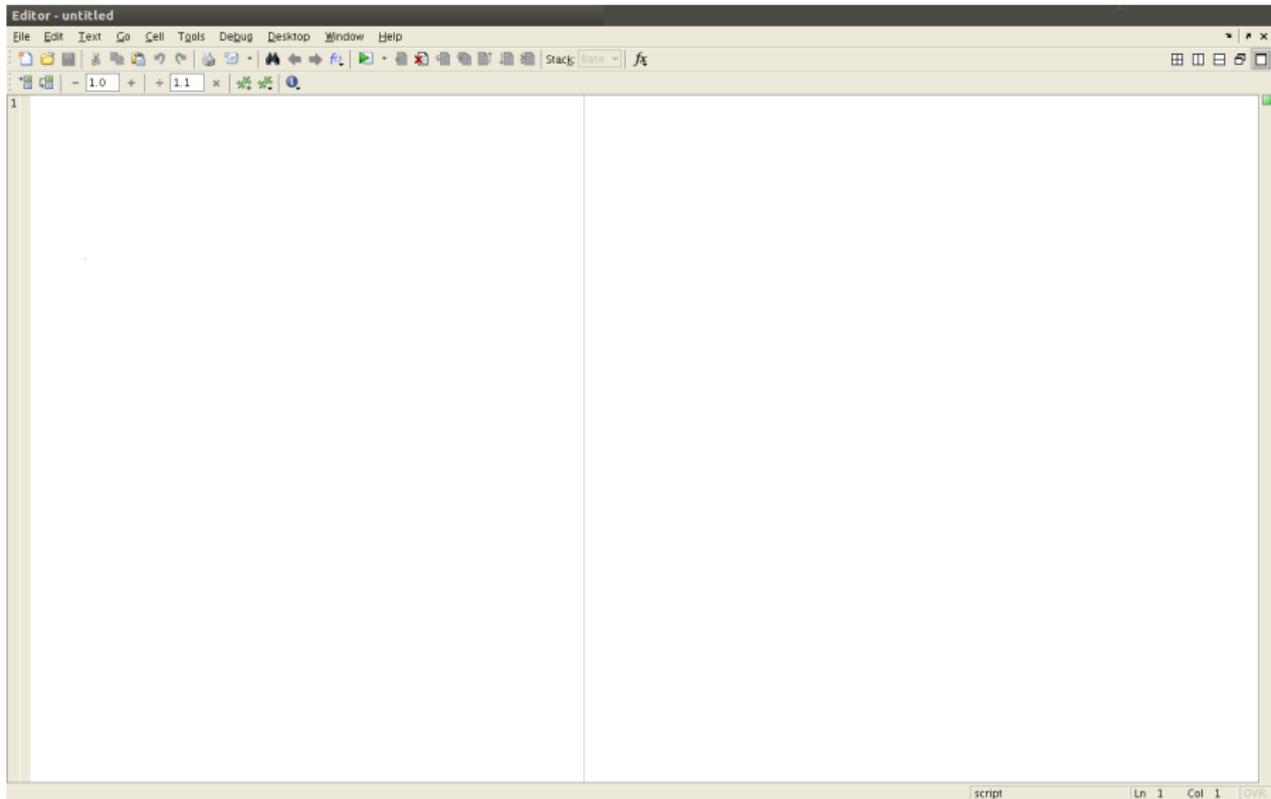
```
% >>
```

Workspace

Name	Value	Min	Max

Command History

Start Ready



```
1 function mein_sinus(x)  
2 sin(x)
```

- Berechnet den Sinus von x
- Kein „;“ gibt Wert aus
- Kein Rückgabewert

```
1 function [y]=mein_sinus(x)  
2 y=sin(x);
```

- Berechnet $\sin(x)$
- Gibt y aus
- Kann eine Variable beschreiben
„ $y=\text{mein_sinus}(\pi)$ “

```
1 function [y,y2]=mein_sinus(x)
2 y=sin(x);
3 y2=cos(x);
4 figure
5 plot(y,y2)
```

- Berechnet $\sin(x)$ und $\cos(x)$
- Gibt y aus
- Kann eine Variable beschreiben
„ $y=\text{mein_sinus}(\pi)$ “
- Kann zwei Variablen beschreiben „ $[y,y2]=\text{mein_sinus}(\pi)$ “
- Plottet den Punkt mit skalarem x
- Plottet einen Kreis mit vektorielltem x

```
1 function Kreis
2 global y y2;
3 x=linspace(-pi,pi,1000);
4 mein_sinus(x);
5 figure
6 plot(y,y2)
```

```
1 function mein_sinus(x)
2 global y y2;
3 y=sin(x);
4 y2=cos(x);
```

- Aufruf über „Kreis“
- Definiert x, y und y2 global
- Führt Funktion mein_sinus aus
- Plottet den Kreis

```
1 function Kreis
2 %x y y2 koennen ueberall
   genutzt werden wo sie
   global sind
3 global x y y2;
4 %% Hier beginnt die Rechnung
5 x=linspace(-pi,pi,1000);
6 mein_sinus(x);
7 %% Hier wird geplottet
8 figure
9 plot(y,y2)
```

- % ist das Kommentarzeichen in Matlab
- %% gefolgt von min. einem Leerzeichen definiert einen neuen Block

```
1 for n=1:1:10  
2 Anweisungen  
3 end;
```

```
1 while n<10  
2 Anweisungen  
3 end;
```

- Schleifen werden mit einem „end;“geschlossen

```
1 if x<10
2   Anweisung 1
3 elseif x<=20
4   Anweisung 2
5 else
6   Anweisung 3
7 end;
```

- if-Abfragen werden mit einem „end;“ geschlossen
- Schachtelung mit „elseif“
- < ist kleiner
- > ist größer
- <= ist kleiner oder gleich
- == ist gleich
- ~= ist ungleich

```
1 switch x
2 case 1
3   Anweisung 1
4 case 2
5   Anweisung 2
6 otherwise
7   Anweisung 3
8 end;
```

- Case-Abfragen werden mit einem „switch“ gefolgt von der Variablen begonnen
- Enden mit „end;“
- Strings als Variable möglich, dann „case 'ja'“

```
1 data=num2str(x);  
2 x=str2num(data);  
3 name=strcat('Teil1 ','Teil2');
```

- Wandelt Zahlen in Strings um
- Wandelt Strings in Zahlen um
- Verbindet Strings

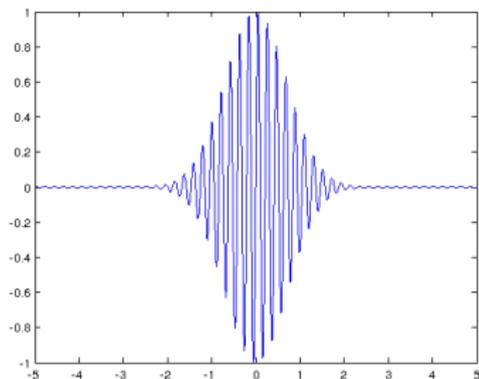
```
1 save ( 'Dateiname' , 'x' ) ;  
2 save ( 'Dateiname' , 'x' , ' - MAT' ) ;  
3 save ( 'Dateiname' , 'x' , ' - ASCII' )  
  ;  
4 save ( 'Dateiname' , 'x' , ' - ASCII  
  ' , ' - TABS' ) ;  
5 save ( 'Dateiname' , 'x' , ' - ASCII  
  ' , ' - DOUBLE' , ' - TABS' ) ;
```

- Binäres Speichern volle Genauigkeit
- Speichern in Text-Datei 8bit oder 16bit
- Tabulatorgetrennt
- Strings werden als Zahlen gespeichert

```
1 load ( 'Dateiname' );  
2 load ( 'Dateiname', 'x' );  
3 importdata ( 'Dateiname', '\t' );  
4 f=imread ( 'Dateinamen', 'Format'  
5         ' );  
6 f=imread ( 'http://...', 'Format'  
7         ' );
```

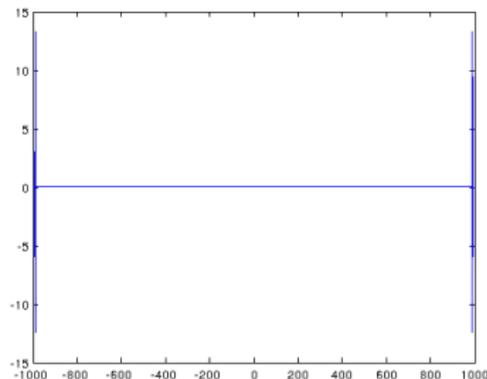
- Lädt alle Variablen einer m-Datei
- Lädt Variable „x“ einer m-Datei
- Lädt Text-Datei, tabulatorgetrennt
- Lädt ein Bild im „Format“ in Variable „f“

```
1 x=linspace(-5,5,10000);  
2 f=exp(-x.^2).*sin(30.*x);  
3 plot(x,f);  
4 g=fft(f);  
5 x2=(-1/(x(2)-x(1)),1/(x(2)-x  
    (1)),10000)
```



```

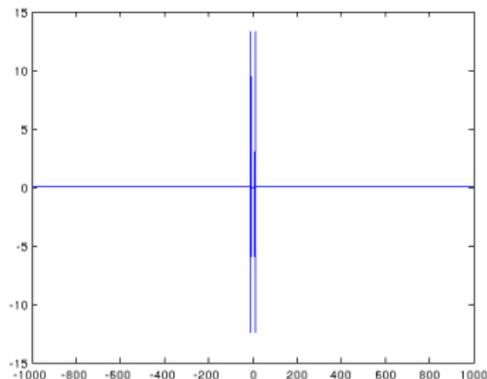
1 x=linspace(-5,5,10000);
2 f=exp(-x.^2).*sin(30.*x);
3 g=fft(f);
4 x2=(-1/(x(2)-x(1)),1/(x(2)-x
   (1)),10000)
5 plot(x2,g)
  
```



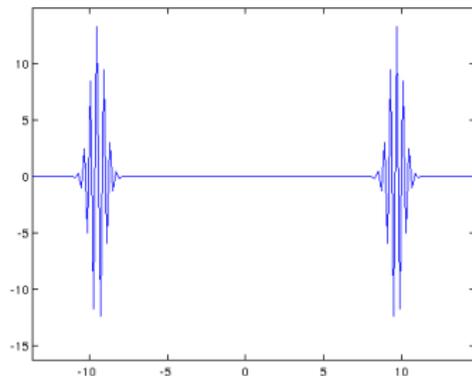
```

1 x=linspace(-5,5,10000);
2 f=exp(-x.^2).*sin(30.*x);
3 g=fftshift(fft(f));
4 x2=(-1/(x(2)-x(1)),1/(x(2)-x
   (1)),10000)
5 plot(x2,g)

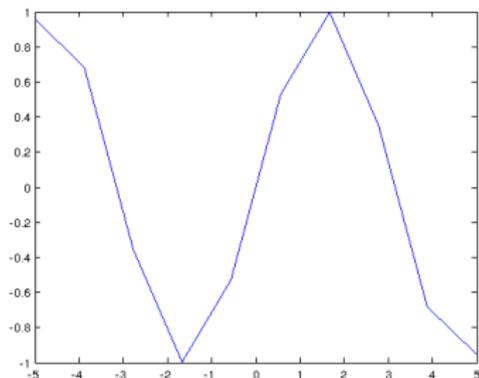
```



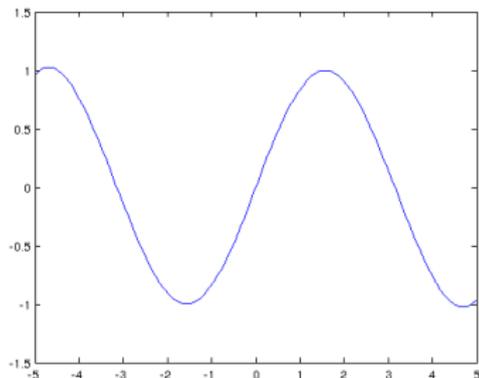
```
1 x=linspace(-5,5,10000);  
2 f=exp(-x.^2).*sin(30.*x);  
3 g=fftshift(fft(f));  
4 x2=(-1/(x(2)-x(1)),1/(x(2)-x  
   (1)),10000)  
5 plot(x2,g)
```



```
1 x=linspace(-5,5,10);  
2 y=sin(x);  
3 plot(x,y)  
4 x2=linspace(-5,5,100);  
5 y2=spline(x,y,x2);  
6 plot(x2,y2)
```



```
1 x=linspace(-5,5,10);  
2 y=sin(x);  
3 plot(x,y)  
4 x2=linspace(-5,5,100);  
5 y2=spline(x,y,x2);  
6 plot(x2,y2)
```



```
[t_out , y_out]=ode45('Funktion', [0 20], [2 0]);
```

- Löst die Differentialgleichung $y' = \text{„Funktion“}$ von y und t
- Zeitbereich t von 0 bis 20
- Anfangsbedingungen $y=2$
 $y'=0$
- t_out und y_out sind die Funktionspaare zu verschiedenen Zeiten

```
1 daten=zeros(length(data(1,:))*  
    length(data(:,1)),3);  
2 for j=1:length(data(1,:))  
3 for k=1:length(data(:,1))  
4 daten(j*(k-1)+k,1)=x(j);  
5 daten(j*(k-1)+k,2)=y(k);  
6 daten(j*(k-1)+k,3)=data(j,k);  
7 end;  
8 end;
```

- Daten liegen als x- & y-Vektoren und 2D-z-Matrix vor
- Hinterher eine Datei, das in gnuplot genutzt werden kann

```
1 syms a b c x f g h;  
2 f=a*x^2+b*x+c;  
3 g=diff(f,x)  
4 h=int(f,x)  
5 simplify(f)  
6 expand(f)  
7 factor(f)  
8 taylor(f)
```

- benötigt die „Symbolic Toolbox“
- „syms“ definiert symbolisch Variablen
- Symbolische Ableitung
- Symbolische Integration
- Vereinfachen
- Ausmultiplizieren
- Faktorisieren
- Taylorentwicklung

- <http://www.mathworks.de/matlabcentral/fileexchange/>
- Datenbank von User-Programmen