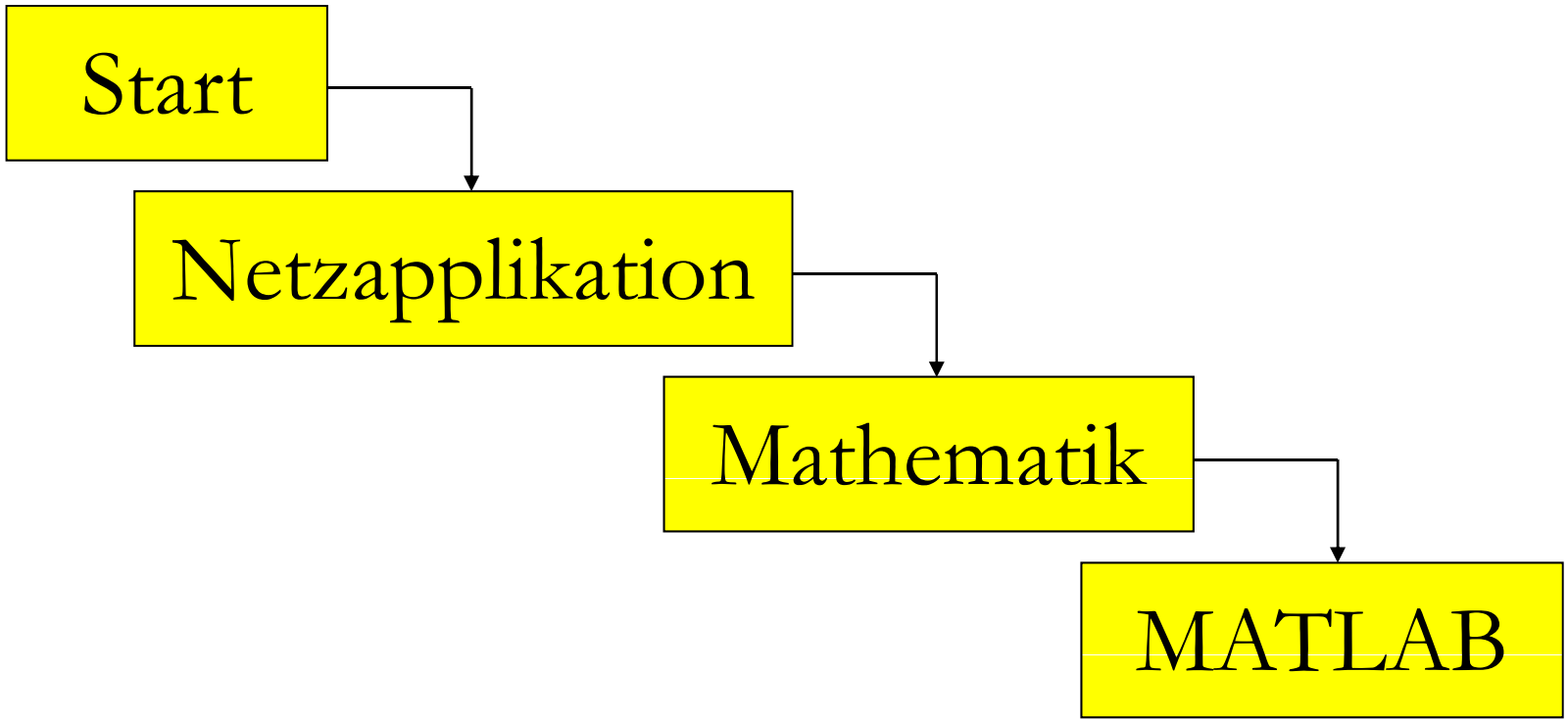
Einführung in MATLAB

MATLAB = Matrizenlaboratorium

MATLAB = Programmsprache +Computeralgebra
+ Datenvisualisierung

Alle Daten werden in MATLAB als Felder (Vektoren, Matrizen) abgelegt und bearbeitet.

Aufruf von MATLAB



startup.m – File

- Vor dem ersten Aufruf von MATLAB muss im Verzeichnis I: ein Ordner mit dem Namen „matlab“ angelegt werden, der das File startup.m enthält.
- In diesem Verzeichnis wird alles abgelegt, was mit MATLAB im Zusammenhang steht.
- Im File startup.m sind alle Pfade anzugeben, die zu persönlichen Verzeichnissen führen und selbstgeschriebene Skript- oder m-Files enthalten. Deshalb sollte das startup.m – File auf keinen Fall gelöscht werden.

Command – Window

- Einfache Berechnungen können im Command - Window ausgeführt werden.
- Rechenoperationen

  + ; - ; * ; / ; \ ; ^ Klammern ( ) und [ ]
- Berechnung von links nach rechts mit üblichen Prioritäten
- Ergebnisausschrift mit Variabler ans (answer)

Beispiel:

```
>> (5 - 3*6)/(7 + 35 - 18*3)
ans =
    1.0833
>>
```

Anlegen eines Protokollfiles

- >> diary <filename>

  Schreibt alles in ein File mit dem Namen

  <filename>, der im Verzeichnis I:\matlab angelegt wird.
- >> diary on weiterschreiben in ein File
- >> diary off unterbrechen

Variable

- Ausdrücke können mit Variablen bezeichnet werden

Beispiel:

```
>> a = 3 * 5;
>> b = ((a - 1)*6 -3);
>> c = a - b
c = -66
```

- Display der definierten Variablen mit *who* bzw. *whos*
- Variablennamen sind „case-sensitiv“

  (Länge von Variablennamen bis zu 31 characters)
- Löschen einer Variablen mit *clear* <variablenname>

  Löschen aller Variablen mit *clear*.

Vordefinierte Variable

- pi = π
- i = j = 0 + i imaginäre Einheit
- eps = 2.2204 e-016 Maschinengenauigkeit
- NaN - not a number (Antwort z.B. bei 0/0)
- Inf - „unendlich“-Symbol
- Matlabprocessing wird unterbrochen mit
  Ctrl – C
- Semikolon am Ende eines Ausdruckes
  verhindert das Ausdrucken seines Wertes

Ausgabeformat

- Intern wird mit 16 Mantissenstellen gerechnet
- Ausgabeformat kann festgelegt werden

  z.B.: >> format short (ist Standard)

  >> format long
- Format besser im Menü „Preferences“ im

  im „File“-Menü einstellen
- Löschen des Bildschirminhaltes mit „clc“
- Fortsetzen eines Ausdruckes auf der

  nächsten Zeile durch „ …“

Komplexe Zahlen

- c = 1 – 2 i d = 3 + 5 i

- aber e = 1 + cos(pi/4) * i

Spezielle Funktionen für komplexe Zahlen

real(c) =1 imag(c) = -2
angle(c) = -1.1071 % im Bogenmaß
Berechnung im Gradmaß:
angle(c)*180/pi
abs(c) = 2.2361 % Betrag von c

Array - Processing

```
>> A = [1, 2, 3; 4, 5, 6; 7, 8, 9]
```

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

Ansprechen des Feldelementes A(1,3):

```
>> A(1,3)  % gibt Element A(1,3) auf dem Bildschirm aus
```

Matrixoperationen

```
>> A = [2, 1, 1; 1, 2, 1; 1, 1, 2];        % A =
```

$$\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$$

```
>> B = [4, 1, 0; 1, 4, 1; 0, 1, 4];        % B =
```

$$\begin{pmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{pmatrix}$$

```
>> A + B          % Summe A + B

>> A – B          % Differenz A – B

>> A * B          % Produkt A * B
```

Matrixfunktionen

```
>> det(A)          % Determinante von A
>> norm(A)         % 2 – Norm von A;  ||A||_2
>> A‘              % Transponierte Matrix A^T
>> size(A)         % Zeilen- u. Spaltenanzahl von A
      >> size(A,1) % Zeilenanzahl von A
      >> size(A,2) % Spaltenanzahl von A
>> cond(A)         % Konditionszahl: ||A||_2||A^-1||_2
>> rank(A)         % Rang von A
>> poly(A)         % Koeff. des charkteristischen Polynoms
>> inv(A)          % Inverse Matrix zu A
```

Lineare Gleichungssysteme

```
>> [L, U] = lu(A)      % LU – Dekomposition von A: A = LU

>> [U, D] = eig(A)     % U – Matrix der Eigenvektoren von A
                       % D – Diagonalmatrix mit EW von A
```

Lösen des linearen Gleichungssystems Ax = b:

$$\mathbf{x = A^{-1} b}$$

>> x = A\b

Beispiel:

```
>> A = [2, 1, 1; 1, 2, 1; 1, 1, 2];
>> b = [4, 4, 4]';
>> x = A\b
```

$$x = \begin{pmatrix} 0.3333 \\ 0.3333 \\ 0.3333 \end{pmatrix}$$

Spezielle Matrizen (1)

Setzen im Weiteren:

```
>> n = 5;
>> m = 3;
>> eye(n)          % Einheitsmatrix
>> zeros(n,m)      % Nullmatrix (speziell: >> zeros(n)  )
>> ones(n,m)       % Matrix ausschließlich mit „Einsen“
                   % besetzt
>> diag(b)         % Matrix mit Diagonalenvektor b
```

Beispiel:

```
>> b = [1, 2, 3];
>> B = diag(b)
```

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

Spezielle Matrizen (2)

```
>> triu(A)      % obere (upper) Dreiecksmatrix von A
>> tril(A)      % untere (lower) Dreiecksmatrix von A
>> rand(n)      % Zufallsmatrix (Elemente gleichverteilt)
>> randn(n)     % Zufallsmatrix (Elemente normalverteilt)
>> hilb(n)      % Hilbertmatrix
>> magic(n)     % magisches Quadrat
```

Felderzeugung

```
>> x = 0 : 0.1 : 1        % x = [0, 0.1, 0.2,…,0.9, 1.0]

>> x = π*(0:0.1:1)        % x = π*[0, 0.1, 0.2,…,0.9, 1.0]

>> a = 1 : 5              % a = [1, 2, 3, 4, 5]
>> a = 1: 0.5 : 9.7       % a = [1, 1.5, 2.0,…,9.0, 9.5]

>> a = 1 : 5;
>> b = 3 : 0.5 : 7;
>> x = [a, b]             % Zeilenvektor
>> y = x‘                 % Spaltenvektor
```

Teilfelder

$$A([1,2],2) = \begin{pmatrix} 2 \\ 5 \end{pmatrix}$$

$$A(1:2:3,:) = \begin{pmatrix} 1 & 2 & 3 \\ 7 & 8 & 9 \end{pmatrix}$$

$$A(1:2:3,1:2:3) = \begin{pmatrix} 1 & 3 \\ 7 & 9 \end{pmatrix}$$

$$A([1,2],[1,3]) = \begin{pmatrix} 1 & 3 \\ 4 & 6 \end{pmatrix}$$

Skalare Funktionen

```
>> sin()
>> sind()
>> cos()
>> tan()
>> asin()
>> acos()
>> atan()
>> exp()
>> abs()
>> sqrt()
```

```
>> log()      % natürlicher
              % Logarithmus
>> rem()      % Divisionsrest
>> sign()     % Vorzeichen
>> round()    % Runden zur nächst-
              %liegenden ganzen Zahl
>> floor()     % Runden zur nächst-
              % kleineren ganzen Zahl
>> ceil()     % Runden zur nächst-
              % größeren ganzen Zahl
```

Beispiele

```
>> exp([0:0.5:1])      %  (1.0000, 1.6487, 2.7183)
>> abs([-5,pi,i])      %  (5.0000, 3.1416, 1.0000)
>> asin([-1:0.5:1])    %  (-1.5708, -0.5236, 0, 0.5236, 1.5708)
>> sqrt([0:5:10])      %  (0, 2.2361, 3.1623)
>> sign([-3:3])        %  (-1   -1   -1    0    1    1    1)

>> round([pi,exp(1)])        %  (3,   3)
>> floor([pi,exp(1)])        %  (3,   2)
>> ceil([pi,exp(1)])         %  (4,   3)
>> rem(pi,2)                 %  1.1416
```

Vektorfunktionen

>> **a** = [a_1,…,a_n]

>> max(**a**)
>> sum(**a**)
>> sort(**a**)

>> min(**a**)
>> median(**a**)

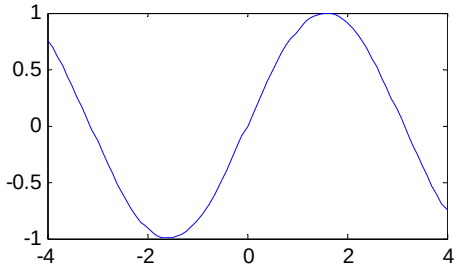
Beispiel:

>> **a** = [3, 5, 1, 9, 7];
>> max(**a**) = 9
>> min(**a**) = 1
>> sum(**a**) = 25
>> median(**a**) = 5
>> sort(**a**) = (1, 3, 5, 7, 9)

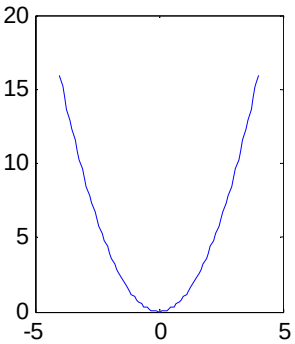
Plotten von Funktionen (1)

Funktionsdefinierte Kurven

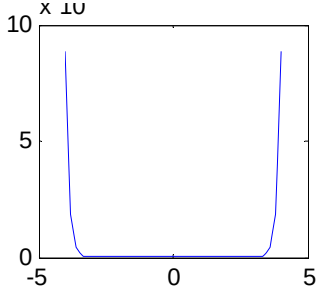
```
>> x = -4 : 0.1 : 4;
>> y = sin(x);

>> plot(x,y)
```




```
>> y = x.^2

>> plot(x,y)
```



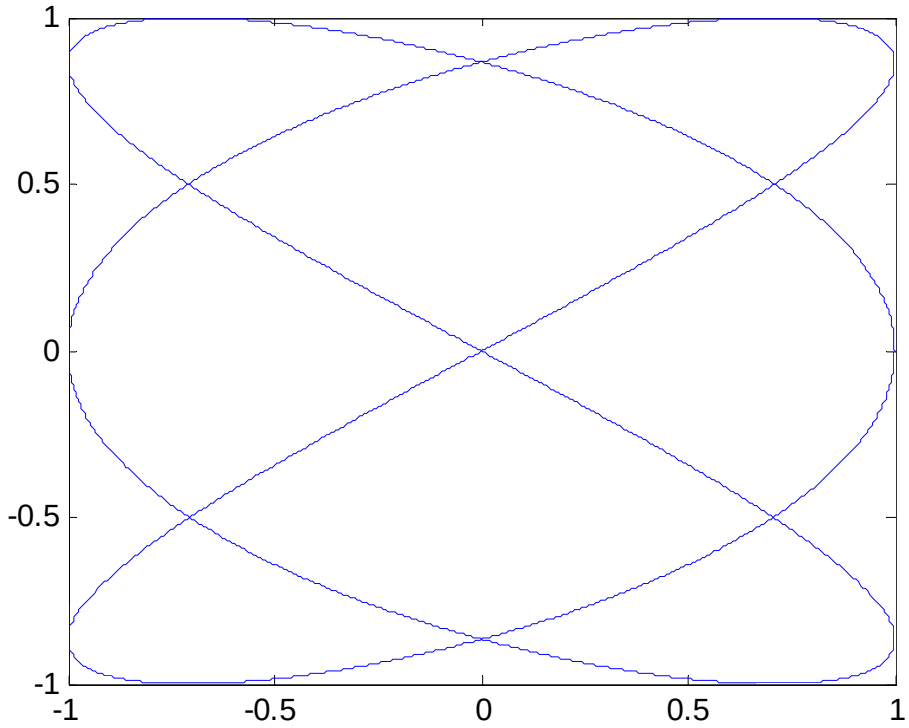

```
>> plot(x,exp(x.^2))
```



Plotten von Funktionen (2)

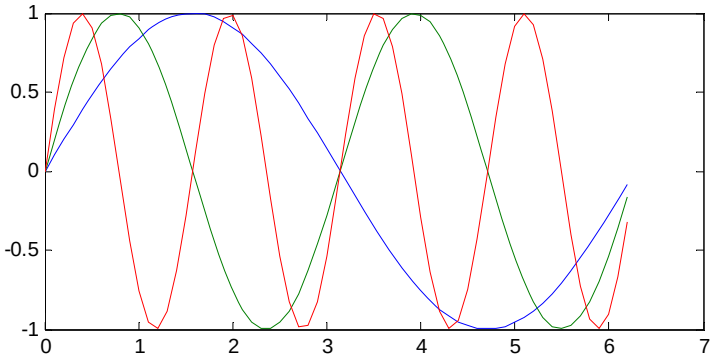
Parametrisch definierte Kurven

```
>> t = 0 : 0.001 : 2*pi;

>> x = cos(3*t);

>> y = sin(2*t);

>> plot(x,y)
```

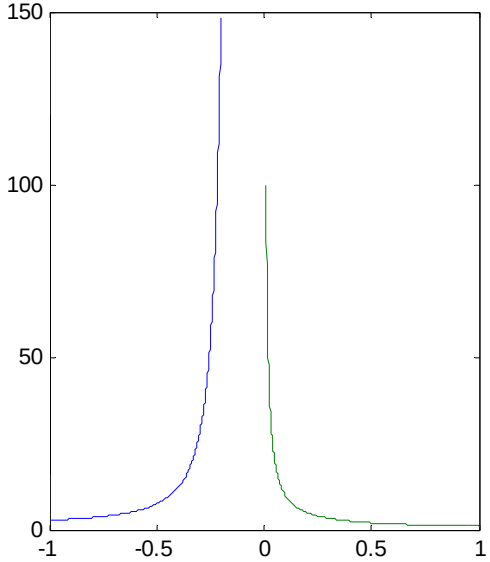


Multiple Plots

```
>> x   = 0 : 0.1 : 2*pi;
>> y1 = sin(x);
>> y2 = sin(2*x);
>> y3 = sin(4*x);
>> plot(x,y1,x,y2,x,y3)
```




```
>> x = -1 : 0.001 : -0.2;
>> y = 0.01 : 0.001 : 1;
>> plot(x,exp(-1./x),y,1./y)
```



Polynome

- Darstellung durch den Koeffizientenvektor
- Beginn bei der höchsten Potenz
- Wertbelegung mittels polyval(name,wert)

Beispiele:

```
>>p=[1,0,1-i];          %   p = x^2 + 1 - i
>>q=[1,0,-2,-5]         %   q = x^3 – 2x - 5
>>a=polyval(q,5)        %   a =110
```

Funktionen von Polynomen

```
>>r=roots(p)              % Nullstellen von p(x)
>>poly(r)                 % berechnet p(x) aus den Nullstellen
>>conv(p,q)               % Produkt der Polynome p(x) und q(x)
>>[q,rest]=deconv(c,p)    %  Polynomdivision
                                c(x)/p(x)=q(x)+rest/p(x)
>>polyder(p)              % Ableitung des Polynoms p(x)
>>polyfit(x,y,n)          % Polynomapproximation der Ordnung
                             n zu den Datensätzen x und y
```

Beispielfunktion (1)

- mittels m-files können Funktionen selbst definiert werden.
- Aufruf erfolgt durch name(wert)
- Weitergabe des Namens in einer Parameterliste muss als character-Variable erfolgen

Beispiel: Funktion humps(x) ist vordefiniert

$$humps(x) = \frac{1}{(x-0.3)^2 + 0.01} + \frac{1}{(x-0.9)^2 + 0.04} - 6$$

Beispielfunktion (2)

```
>>humps(1)                  % 16
>>humps(-1)                 % -5.1378
>>fplot('humps',[-5,5])     % zeichnet humps(x) in [-5,5]
>>fminbnd ('humps',0.3,1)   % sucht Minimum in [0.3,1]
>>fzero('humps',-0.2)       % sucht Nullstelle in der
                              Umgebung von -0.2
>>fzero('humps',[-1,1])     % sucht Nullstelle im Intervall
                              [-1,1] (sinnvoll, wenn Vorzei-
                              chenwechsel dort bekannt ist)
```

Die Nullstellenbestimmung erfolgt numerisch mittels Iteration

Symbolische Berechnungen

- Erfolgen in der Symbolischen Toolbox, die dem Computeralgebrasystems MuPAD entspricht (direkter Zugriff möglich ➔ >> mupadwelcome)
- Enthält über 100 Funktionen, z.B. für Differentiation, Integration, Grenzwerte, Summen, Reihen, Lösung von algebraischen und Differentialgleichungen, Lineare Algebra, Fourier- und Laplacetransformation,….
- Variablen müssen dafür deklariert werden:
  syms variablenliste

Kurvendiskussion (1)

- Beispiel: $f(x)=\frac{3x^2+6x-1}{x^2+x-3}$

  Zu bestimmen sind:
  - Symmetrie
  - Verhalten an den Rändern von D_f
  - Achsenschnittpunkte
  - Unstetigkeitsstellen
  - Extremwerte
  - Skizze

Kurvendiskussion (2)

```
>>syms x                          % Deklaration von x
>>zae=3*x^2+6*x-1;
>>nen=x^2+x-3;f=zae/nen;          % Definition von f
>>subs(f,-x)                      % ersetze x durch –x
                                  % zur Symmetrieprüfung
>>ezplot(f)                       % erstelle Skizze
>>limit(f,inf)                    % berechne
>>limit(f,-inf)                   lim_{x→∞} f(x), lim_{x→-∞} f(x)
>>subs(f,0)                       %Schnittpunkt mit y-Achse
>>nstx=solve(zae)                 %Nullstellen des Zaehlers
```

Kurvendiskussion (3)

```
>>nst=solve(nen)                          %Nullstellen des Nenners
>>limit(f,x,nst(1),‘left‘)                % Einseitige Grenzwerte
>>limit(f,x,nst(1),‘right‘)               % an den Nennernullstellen
>>limit(f,x,nst(2),‘left‘)                % zur Identifizierung der
>>limit(f,x,nst(2),‘right‘)               % Polstellen
>>f1=diff(f),f2=diff(f1)                  % 1. und 2. Ableitung von f
>>simplify(f1), simplify(f2)              % Vereinfachen der Terme
>>pretty(f1)                              % Schreibmaschinenausgabe
>>kritpkt=solve(f1)                       % Nullstellen der 1. Ableitung
>>w1=double(subst(f2,kritpkt(1)))         % Werte der 2. Ableitung zum
>>w2=double(subst(f2,kritpkt(2)))         % Test auf Max. / Minimum
```

Hilfemöglichkeiten

- help name: Kommentare aus m-File name.m erscheinen im Command Window
- helpwin name: Kurze Erklärung zum m-File name.m in der Hilfedatei mit Verweisen auf ähnliche Befehle, funktioniert auch bei eigenen m-files!
- doc name: ausführliche Erklärung zum m-File name.m in der Hilfedatei mit Beispielen
- who: listet alle Variablen auf, die in Benutzung sind
- whos: listet alle Variablen mit Größe und Typ auf
- Weiter gibt es: exist name, what name, which name (s. Hilfedatei, s.o.)