Logik mit Mathematica

ISB, WIR90

Definition der Wahrheitswerte:

```
w :=  1==1;  f:=  1==0
```

```
w
```

```
True
```

```
f
```

```
False
```

Junktoren:

Mangels der mathematischen Zeichen auf dem Textsystem (*Mathematica*) werden hier die englischen Bezeichnungen verwendet:

NOT A	(!A)	Die Klammer muss stehen, wenn der logische Ausdruck sonst mit NOT beginnen würde. "!" am Zeilenanfang hat eine andere Bedeutung.
A AND B AND C..	A && B && C ..	
A OR B OR C..	A || B || C ..	
A XOR B XOR C..	Xor[A,B,C,...]	
IF A THEN B	Implies[A,B]	
Bedingte Wahrheit:	If[P,T,F]	ergibt T falls P TRUE ist, ergibt F falls P FALSE ist.

Logisches Ausmultiplizieren:

LogicalExpand["definierter Ausdruck"]

Beispiel: Wahrheitstabelle von (X AND NOT Y) OR (X AND Y):

```
a[x_,y_] := (x&&!y)||(x&&y)
```

```
{a[w,w],a[w,f],a[f,w],a[f,f]}
```

```
{True, True, False, False}
```

```
Mathematica (MS-DOS 386/7) 1.2 (September 27, 1989) [With pre-loaded data]
by S. Wolfram, D. Grayson, R. Maeder, H. Cejtin,
   S. Omohundro, D. Ballman and J. Keiper
with I. Rivin, D. Withoff and T. Sherlock
Copyright 1988,1989 Wolfram Research Inc.

In[1]:= w:= 1==1; f:= 1==0

In[2]:= w

Out[2]= True

In[3]:= f

Out[3]= False

In[4]:= a[x_,y_]:=(x&&!y)||(x&&y)

In[5]:= {a[w,w],a[w,f],a[f,w],a[f,f]}

Out[5]= {True, True, False, False}
```

Grundoperationen mit Mathematica

Grundoperationen: ISB, WIR90

`x+y+z`	Addition
`x y z oder x*y*z`	Multiplikation

Multiplikationen können durch einen Leerschlag angegeben werden!

Achtung: xy heisst "Die Variable mit Namen "xy" und nicht x mal y!

`x/y`	Division
`-x`	Minus
`x^y`	Potenzen: "x hoch y" ("^" steht für "hoch")
	Achtung: x^3z heisst (x^3)z und nicht x^(3z) !
`Sqrt[x]`	Quadratwurzel aus x
`(x y+3.54)2+11^-3.5`	Klammerausdruck
`5.42 10^-7`	Wissenschaftliche Zahlennotation

Bemerkungen:

Mathematica rechnet "exakt", sofern man nicht einen numerischen Näherungswert verlangt oder es sich um eine Approximation handelt.

`Ausdruck // N`	`oder`	ergibt einen Näherungswert,
`N[Ausdruck]`		Genauigkeit dem System überlassen.
`N[Ausdruck, n]`		Näherung auf n Stellen genau.

Mathematica **hat auch einige Konstanten "exakt" gespeichert:**

z.B.	`Pi`	π = 3.14159....
	`E`	e = 2.71828....
	`Degree`	$\pi/180$: Konvertierung Grad in Radian
	`I`	i = Sqrt[-1]
	`Infinity`	unendlich

Mathematica merkt sich die Eingabezeilen einer Session mit Nummern. Ueber diese Nummern kann man auf die Resultate referenzieren:

`%`	Das letzte Resultat
`%%`	Das zweitletzte Resultat
`%n`	Das Resultat der Zeile Nummer n
`%n + %m`	Summe der Zeilen n und m

Mathematica rechnet komplex:

`z = x + Iy`	Komplexe Zahl x+iy
`Re[z]`	Realanteil
`Im[z]`	Imaginäranteil
`Conjugate[z]`	Konjugiert Komplexe Zahl
`Abs[z]`	Absolutbetrag
`Arg[z]`	Argument

Funktionen mit Mathematica:

- Die Argumente von Funktionen werden in [ ] gesetzt.
- Eingebaute Funktionen beginnen mit Grossbuchstaben.
- Die meisten in der Praxis gebräuchlichen Funktionen sind eingebaut.

Beispiele:

`Sqrt[x]`	Quadratwurzel
`Exp[x]`	Eulersche Exponentialfunktion
`Log[x]`	Logarithmus naturalis
`Log[b, x]`	Logarithmus zur Basis b
`Sin[x], Cos[x], Tan[x]`	Trigonometrische Funktionen (Radian)
`ArcSin[x], ArcCos[x], ArcTan[x]`	Arcus-Funktionen
`n!`	Fakultät
`Abs[x]`	Absolut-Betrag
`Round[x]`	Rundung auf ganze Zahlen
`Mod[n,m]`	n modulo m
`Random[ ]`	Pseudo-Zufallszahl zwischen 0 und 1
`Max[x,y,...]`	Maximum der gegebenen Werte
`Min[x,y,...]`	Minimum der gegebenen Werte
`FactorInteger[n]`	Primfaktor-Zerlegung

Variablen:

`Zuerst Grossbuchstabe:`	Eingebautes Objekt
`x = 3`	Der Variablen x wird der Wert 3 zugewiesen
`x = y = 5`	x und y ist 5
`x = .`	x hat keinen Wert (missing value: ".")

Definition von Funktionen:

Beispiel: `f[x_, y_, z_]:= x^2 y + (y z - x)`

Anwendung: `f[3, 4, 2] .. rechnet..3^2 4 + (4 2 - 2) = 36+6 = 42`

`?f`	Zeigt die Definition von f
Löschen: `Clear[f]`	Funktion ist nicht mehr definiert!

Operationen mit Listen und Mengen

Definition einer Menge oder einer Liste: ISB, WIR90

Formal ist in *Mathematica* eine Menge und eine Liste dasselbe.
Mengen oder Listen werden durch "{ }" abgegrenzt. Die Elemente sind durch "," getrennt.

Beispiele: `{5, 2, 4}` oder `{a, b, c, d}`

Eine Menge oder Liste kann durch eine Variable angesprochen werden:

```
var = {5, 2, 4}
```

Arithmetische Operationen mit Listen:

Beispiele:

```
var + 1   ergibt  {6, 3, 5}
var^2     ergibt  {25, 4, 16}
var/var ergibt    {1, 1, 1}     etc.
```

Arithmetische Operationen mit Listen wirken elementweise.

Einfache Listen können automatisch generiert werden:

z.B. `Range[n]` generiert `{1, 2, 3, ...., n}`
`Range[m,n]` generiert `{m, m+1, m+2, ...., n}`

Manipulation von Elementen einer Liste:

Sei `w = {..,..,..,.....}` eine Liste.

`w[[i]]` ergibt das Element mit der Nummer i der Liste

`w[[{i, j, k}]]` ergibt eine Liste mit den Elementen Nummer i, j, k...

`w[[i]] = Wert` setzt "Wert" als Element Nummer i in die Liste ein. Die Liste wird dadurch grösser.

`w[[i, j, ...]]` ergibt das .... j-te Element vom i-ten Element einer **geschachtelten** Liste.

Beispiel:

```
w={{1,2},{1,2,3,4,5,6,7,8,9,10}};
w[[2,3]]      ergibt 3
```

Bemerkung zum Gebrauch von Klammern:

`( )`	für Terme, Gruppierungen,...
`[ ]`	bei Funktionen
`{ }`	bei Listen, Mengen
`[[ ]]`	Dopperklammern bei Indices

Operationen mit Listen und Mengen *(unvollständig)*:

`Union[list1, list2,...]`	Vereinigungsmenge
`Union[list]`	Entfernt doppelte Elemente
`Intersection[list1, list2,...]`	Schnittmenge
`Complement[universal, list1,...]`	Komplement
`Join[list1, list2,...]`	Hängt die Listen aneinander (Doppelte Elemente doppelt)
`Length[list]`	Anzahl Elemente der Liste (grösser gleich Mächtigkeit)
`MemberQ[list, elem]`	Test: elem Element der Liste?
`Count[list,elem]`	Anzahl Vorkommen von elem
`Free[list,elem]`	Test auf Auftreten von elem
`Position[list,elem]`	Position von elem in list
`First[list]`	Das 1. Element der Liste
`Last[list]`	Das letzte Element der Liste
`w[[n]]`	Das Element mit Nummer n der Liste w
`w[[-n]]`	Das Element Nummer n von hinten weg
`w[[{n1, n2,..}]]`	Die Liste mit den Elementen n1, n2, ...
`{x1, x2, ...} = {y1, y2, ...}`	Weist jedem xi den Wert yi zu.
`Take[list, n]`	Teilliste der ersten n Elemente
`Take[list, -n]`	Teilliste der letzten n Elemente
`Take[list, {m, n}]`	Teilliste von Nummer m bis n
`Rest[list]`	Teilliste ohne erstes Element
`Drop[list, n]`	Teilliste ohne die ersten n Elemente
`Drop[list, -n]`	Teilliste mit den letzten n Elementen
`Drop[list, {m, n}]`	Teilliste mit den Elementen m bis n
`w[[{i1, i2, ...}, {j1, j2, ...}]]`	Bildet eine Liste aus den Unterlisten Nummer i1, i2, .. wobei jeweils nur das j1-te und das j2-te Element der Unterlisten genommen werden. (Vgl. Vorderseite.)
`Prepend[list, elem]`	Fügt elem an den Beginn einer Liste hinzu
`Append[list,elem]`	Hängt elem hinten an einer Liste an
`Insert[list, elem, n]`	Fügt elem an Position n in die Liste ein
`Insert[list, elem, -n]`	Fügt elem an Pos. n von hinten gezählt in die Liste ein
`Partition[list, n]`	Gliedert die Liste in Unterlisten mit je n Elementen
`Partition[list, n, d]`	Die jeweils folgende Unterliste beginnt nach d weiteren Elementen
`Outer[List, list1, list2, ...]`	Gibt Listen von Elementen aus den listj, kombiniert auf alle möglichen Weisen
`Flattern[list]`	Unterdrückt Gruppierung in Unterlisten
`Flattern[list, n]`	Unterdrückt das erste Unterlistenniveau
`Transpose[list]`	Vertauscht die beiden ersten Niveaus
`Transpose[list, n]`	Vertauscht das erste mit dem n-ten Niveau

Tabellen, Vektoren, Matrizen Determinanten, Eigenwerte etc.

ISB, WIR90

Beispiel: Determinante mal Inverse der Matrix, Matrixform

In[1]:=

```
M = {{1,2,-1},{2,0,3},{0,1,4}};TableForm[Det[M]Inverse[M]]
```

Out[1]//TableForm=

```
-3   -9   6

-8   4    -5

2    -1   -4
```

Mathematica-Befehle

Tabellen, d.h. Generierung von Listen:

Table[Funktion,{imax}] z.B. `Table[22,{12}]` Generiert eine Liste von imax Werten "Funktion"

Table[Funktion,{i,imax}] z.B. `Table[Sin[n Pi/12],{n,24}]` Generiert eine Liste von n Funktionswerten, n von 1 bis 24

Table[Funktion,{i,imin,imax}] Generiert Liste, i von imin bis imax

Table[Funktion,imin,imax,d}] Generiert Liste mit Schrittweite d

Table[Funktion,{i,imin,imax},{j,jmin,jmax},...] Liste multidimensional

TableForm[Liste] Gibt die Liste in Tabellenform

List[[i]] Gibt die i-te Unterliste von List

List[[{i1,i2, ...}]] Gibt eine Liste der i1-ten, i2,-ten etc. Unterliste von List

List[[i,j,...]] Gibt den Teil von List entsprechend List[[i]][[j]]...

Vektoren und Matrizen:

{a, b, c, d} ***Vektor*** mit den Komponenten a, b, c, d

{{a,b,c},{d,e,f},{g,h,i}} ***Matrix*** mit der 1. Zeile a, b, c, der 2. Zeile d, e, f...

Table[Funktion,{i,n}] Gibt einen Vektor mit den Komponenten Funktion[i], i von 1 bis n

Array[a,n] Gibt einen Vektor der Form {a[1], a[2],...} mit n Komponenten

vekt[[i]] Gibt die i-te Komponente vom Vektor "vekt"

Vektoren und Matrizen, Fortsetzung:

ColumnForm[vekt] Gibt den Vektor (die Liste) "vekt" in Spaltenform
Table[Funktion,{i,m},{j,n}] Gibt eine *m x n* Matrix, die Funktion wird berechnet für i von 1 bis m und j von 1 bis n
Array[a,{m,n}] Gibt eine *m x n* Matrix mit dem i-j-ten Element a[i,j]
IdentityMatrix[n] Gibt die *n x n* Einheitsmatrix
DiagonalMatrix[List] Quadratische Diagonalmatrix, List: Diagonale
M[[i]] Gibt die i-te Zeile der Matrix M
M[[i,j]] Gibt das i-j-te Element der Matrix M
MatrixForm[M] Erzeugt M in Matrixdarstellung auf dem Schirm

Operationen mit Vektoren und Matrizen:

const M resp. **const V** Multipliziert Matrix M resp. Vektor V mit const
M1 . M2 Matrixprodukt Matrix M1 mal Matrix M2
Inverse[M] Inverse der Matrix M
Det[M] Determinante von M
Transpose[M] Transponierte der Matrix M
{Vekt1} . Transpose[{Vekt2}] ***Skalarprodukt*** als Matrixprodukt: Vektor1 als einzeilige Matrix mal Vektor2 als einspaltige Matrix
Eigenvalues[M] Eigenwerte der Matrix M
Eigenvalues[N[M]] Numerische Werte der Eigenwerte von M
Eingenvectors[M] Eigenvektoren der Matrix M , **Eigenvectors[N[M]]...**
Dimensions[List] Dimension einer Liste als Array
VectorQ[List] Test, ob List ein Vektor ist
MatrixQ[List] Test, ob List eine Matrix ist
TensorRank[List] Rang der Liste List als Tensor

Listenoperationen und etwas Kombinatorik

Sort[List] Sortiert die Elemente von List in Standardreihenfolge
OrderedQ[List] True, falls List geordnet ist
Reverse[List] Kehrt die Reihenfolge der Elemente um
RotateLeft[List, n] Rotiert die Elemente von List n Stellen nach links
RotateRight[List,n] Rotiert die Elemente von List n Stellen nach rechts
RotateLeft[List], RotateRight[List] Rotiert um eine Position
Apply[Plus, List] Bildet die Quersumme von List
Apply[Times, List] Bildet das Querprodukt von List
Permutations[List] Gibt alle möglichen Permutationen von List
Signature[List] Gerade oder ungerade Permutation

Algebraische Operationen mit *Mathematica*

ISB, WIR90

Mathematica kann **numerisch** wie auch **symbolisch** (mit Formeln) rechnen.

Beispiele: Numerisch: 4*5+7/8

Symbolisch: 3x - 4 x y + 11 x + 2 x^2 (x + y) + 6 y x

Mit Einheiten: 2 Meter * 5 sec + 8 Meter 2 sec - 5 Meter sec

In[2]:=

```
3x - 4 x y + 11 x + 2 x^2 ( x + y) + 6 y x+ 2 Cos[x] + Cos[x]
```

Out[2]=

```
                      2
14 x + 2 x y + 2 x  (x + y) + 3 Cos[x]
```

In[4]:=

```
y = 4 x + 3;  z = y^3 + 2 x + 3
```

Out[4]=

```
                        3
3 + 2 x + (3 + 4 x)
```

Ausgabe: ***Summation gleicher Terme***, aber keine Multiplikation.
Antwort : In mathematischer **Standardnotation**.
Einer **Variablen** kann ein numerischer Wert, aber auch ein symbolischer Wert (Formel) zugeordnet werden.

Belegung von Variablen:
x = Wert ---> Variable belegt.
x = . ---> Variable wieder frei.
Expr /. x -> Wert In Expr wird x durch Wert ersetzt, x später unverändert.
Expr / . {x -> w1, y -> w2} Mehrere Werte ersetzt.

Umformung algebraischer Ausdrücke:

Expand[Ausdruck]: Multipliziert aus (Klammern verschwinden).
Factor[Ausdruck]: Faktorisiert: Ausdruck als Produkt von Termen.
Simplify[Ausdruck]: Sucht Darstellung mit kleinster Anzahl Teilen.

Weitere Operationen:

ExpandAll[Ausdruck] : Wendet Expand überall an.

FactorTerms[Ausdruck]: Zieht aus jedem Term gemeinsame Faktoren heraus.

Together[Ausdruck]: Darstellung auf einem Bruchstrich (gemeinsamer Nenner).

Apart[Ausdruck]: Zerlegt in Terme mit einfachen Nennern. (Parital.)

Cancel[Ausdruck]: Kürzt gemeinsame Faktoren in Zähler und Nenner.

Collect[Ausdruck, x]: Gruppiert Potenzen von x zusammen.

Coefficient[Ausdruck, x]: Koeffizient von x im Ausdruck.

Exponent[Ausdruck, y]: Exponent von y im Ausdruck.

Numerator[Ausdruck]: Zähler des Ausdrucks.

Denominator[Ausdruck]: Nenner des Ausdrucks.

Ausdruck // Short: Ausdruck (grosse Summe) auf einer Zeile geschrieben. Mittlere Terme unterdrückt. Nur die Anzahl Terme wird ausgegeben.

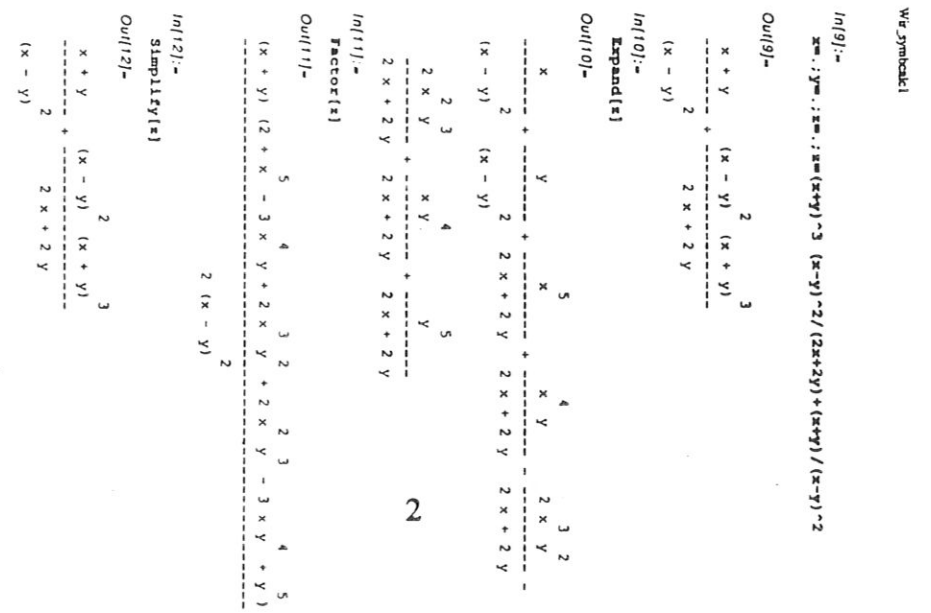
Short[Ausdruck, n]: Ausdruck abgekürzt auf n Zeilen.

Length[Ausdruck]: Anzahl Terme in der Summe.

[Ueber die Grenzen von *Mathematica*: - Speicherbedingt -

Einige *Sekunden Rechenzeit* erfordern:

Arithmetik mit Zahlen mit einigen 1000 Stellen.
Expand mit Polynomen mit einigen 100 Termen.
Faktorisieren von Polynomen mit einigen 100 Termen.
Rekursive Regeln einige 1000 Male anwenden.
Numerische Inverse einer 50 x 50 - Matrix.
Einige Seiten Output formatieren.]



Analysis und Gleichungen

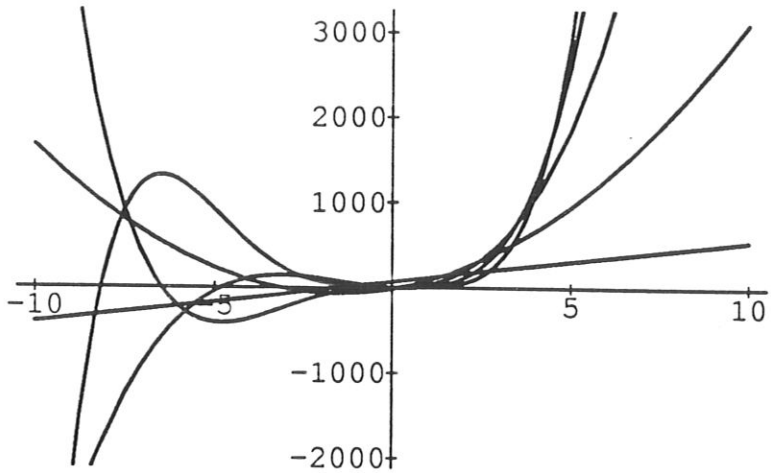
ISB, WIR90

Symbolisches und numerisches Rechnen:

Beispiel eines Plots einer Funktion zusammen mit ihren ersten vier Ableitungen:

In[1]:=

```
f[x_]:= 0.4 x^5 + 3 x^4 - 2 x^3 + x^2 - x + 4;

Plot[Release[Table[D[f[x],{x,n}],{n,0,4}]],{x,-10,10}]
```



Out[1]=

```
-Graphics-
```

Mathematica-Befehle

Ableitungen:

D[Funktion,x]	Erste Ableitung der Funktion nach x
D[Funktion,x1,x2,...]	Partielle Ableitung der Funktion nach x1, x2, ... etc.
D[Funktion,{x,n}]	n-te Ableitung der Funktion nach x
Dt[Funktion],t]	Kettenregel: Funktion mehrerer Var. abgel. nach t
Dt[Funktion]	Totales Differential

Integration:

Integrate[Funktion,x]	Stammfunktion nach der Variablen x
Integrate[Funktion,{x,xmin,xmax}]	Bestimmtes Integral
Integrate[Funkt.,{x,xmin,xmax},{y,ymin,ymax},..]	Mehrdimensional

Nintegrate[Funktion,{x,xmin,xmax}] Numerische Integration
N[Ausdruck] Numerischer Wert des Ausdrucks

Minimum einer Funktion:

FindMinimum[Funktion,{x,x1}] Sucht Minimum der Funktion von x1 an

Summen, Reihen und Produkte:

Sum[Folgenglied,[i,imin,imax}] Summe der Folgenglieder mit Index i
Sum[F_Glied,{i,imin,imax,s}] Summe mit Indexschrittweite s
Sum[F_Glied,{i,imin,imax},{j,jmin,jmax},...] Erst über.... j, dann über i
NSum[F_Glied,{i,imin,Infinity}] Numerische Approximation der Reihe
Product[F_Glied,{i,imin,imax}] Produkt der Folgenglieder mit Index i

Notation bei der Iteration:

{imax} imax Iterationsschritte, ohne eine Variable zu inkrementieren
{i,imax} i geht von 1 bis imax, Schrittweite 1
{i,imin,imax} i geht von imin bis imax
{i,imin,imax,s} i geht von imin bis imax, Schrittweite s
{i,imin,imax},{j,jmin,jmax},... i von imin bis imax, j von jmin bis jmax,...

Potenzreihen und Grenzwerte:

Series[Funktion,{x,x0,Ordnung}] Potenzreihenentwicklung der Funktion um x0 bis zur gegebenen Ordnung Ordnung = Anzahl Terme
Normal[Potenzreihe], z.B. `Normal[%3]` Formt Entwicklung in einen algebraischen verwendbaren Ausdruck um
Limit[Ausdruck,x->xo] Grenzwert des Ausdrucks für x gegen x0 (unendlich: **Infinity**)

Gleichungen und Gleichungssysteme:

Solve[links==rechts,x] Auflösen der Gleich. "links=rechts" nach x, N[%..]
Solve[{links1==rechts1,links2==rechts2,...},{x,y,...}] Mehrere Gl., Var.
Eliminate[{links1==rechts1,links2==rechts2;..},{x,y,...}] Eliminiert x, y, ... im gegebenen Gleichungssystem
Reduce[{links1==rechts1,links2==rechts2,...},{x,y,...}] Gibt eine Menge vereinfachter Gleichungen mit allen möglichen Lösungen
NRoots[Polynom==0,x] Numerische Approximation der Wurzeln v. Polyn.
FindRoots[links==rechts,{x,x0}] Numerisch Lös. der Gleich., Start mit x0

Funktionen und Funktionenscharen mit *Mathematica*

ISB, WIR90

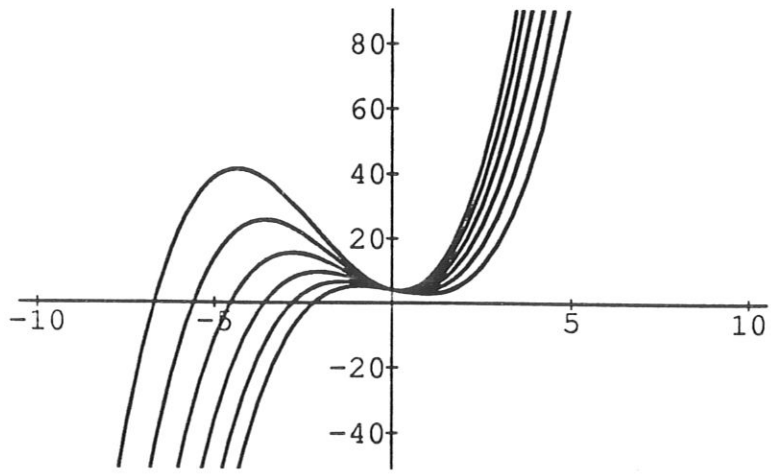
Plotten einer Funktion:

Plot[f, {x, xmin, xmax}]: Gibt den Graphen von f mit Variable x, x zwischen xmin und xmax, f = Funktion. (Wählt erst spezielle Werte für x, berechnet dann f für jedes x, etc..)

Plot[{f1, f2, ...}, {x, xmin, xmax}]: Gibt die Funktionen f1, f2, ... in demselben Diagramm.

Generierung einer Funktionenschar: Z.B. Mit "Release Table". Beispiel:

```
f[x_,a_]:= 0.8x^3+a x^2-2x+4;

Plot[Release[Table[f[x,n],{n,0,5}]],{x,-10,10}]
```




```
-Graphics-
```

Plot[Release[f],{x, xmin, xmax}]: Bestimmt erst die Funktion f, und wählt erst dann spezielle Werte für x.

Plot[Release[Table[f, ...]], {x, xmin, xmax}]: Generiert erst eine Liste von Funktionen, und plottet sie dann.

Show[%3]: Zeigt dann das Diagramm %3 nochmals.

Show[%3, %4, ...]: Zeigt die Plots von %3, %4, ... in einem Bild.

Text[Textausdruck, {x,y}]: Zentriert Textausdruck um (x,y). `Graphics[..]`

ListPlot[{y1,y2,...}]: Zeichnet die Punkte (1,y1), (2,y2) etc. in einem Bild.
ListPlot[{{x1,y1}, {x2,y2}, ...}]: Zeichnet die Punkte (x1,y1), (x2,y2),...
ListPlot[Liste, PlotJoined -> True]: Plottet die Punkte der Liste und zeichnet eine Linie durch die Punkte (Verbindung).
Point[{x,y}]: Zeichnet den Punkt (x, y). (Aktivieren mit **Graphics[...]**)
Line[{{x1,y1}, {x2,y2}, ...}]: Zieht eine Linie durch die angegebenen Punkte. (Aktivieren mit **Graphics[...]**)
Rectangle[{xmin,ymin}, {xmax,ymax}]: Gibt gefülltes Rechteck. (**Graphics[...]**)
Polygon[{{x1,y1}, {x2,y2}, ...}]: Gefülltes Polygon mit angegebenen Ecken. (Aktivieren mit **Graphics[...]**)

Optionen und vorgegbene Einstellungen (default-Werte)

Beilpiel: Plot[Cos[x^3-x+1], {x,-6,6}, **PlotRange -> {0, 1.5}, AxesLabel -> {"x-Wert", "y-Wert"}, Axes -> None**]

Im Beispiel **fett** gedruckt sind die **Optionen.** Optionen und ihre **Einstellungen** (nach "->") **stehen** nach der **Funktion** und ihrem **Wertebereich.**

PlotRange: Wertebereich **{ymin, ymax}** oder Definitions- und Wertebereich **{{xmin, xmax}, {ymin, ymax}}. Default: Automatic** (d.h. interne Algorithmen werden verwendet: "Interessante Gebiete").
All: "alle Punkte" werden gezeigt.

PlotLabel: Ausdruck, der als **Label** (hier Titel) für den Plot gesetzt wird. **Default: None** (keiner).

Framed: Setzen eines **Rahmens** um den Plot. **Default: False** (wird nicht getan).

AspectRatio: Verhältnis **Höhe:Breite. Default: 1/GoldenRatio.** (**Automatic!**)

Axes: Welche **Axen zu nehmen** sind. **Default: None**, nimmt spezifizierte {x,y}. (Gegenteil von None: **All.**)

AxesLabel: Wie die **Axen angeschrieben** sein sollen. **ylabel** für die y-Achse, **{xlabel,ylabel}** für beide Achsen. **Default: None.**

Ticks: Z.B. **{xtick,ytick}** gibt Position der x- und y-**Skalierungsmarken.** Oder **{t1, t2,...}.** Oder **{Range[0, 7, 0.4], Automatic}**: x gegeben, Sprung 0.4, y automatisch. **Default: Automatic.**

PlotColor: Ob **Farb-Graphik. Default: True** (soll gemacht werden). Farben: später.

DisplayFunction: Wie die Graphic **auszugeben** ist. **Default: $DisplayFunction.**
Identity: Keine Ausgabe.

PlotStyle: Bei verschiedenen Graphiken in einem Bild: Für jede Graphik **Stil** (Liste) definierbar (Linienart, Graustufen... wird später behandelt.).
Default: Automatic.

PlotPoints: Minimale **Anzahl von Punkten**, mit denen die Darstellung probiert wird. **Default: 25.**

MaxBend: Maximaler **Winkel** an den Knoten zwischen den **Kurvensegmenten.** **Default: 10.**

PlotDivision: Maximaler Faktor der **Unterteilung der Segmente** beim Versuch, die Funktion darzustellen. **Default: 20.**

Parametrisierte Kurven, spezielle Plots, spezielle Funktionen

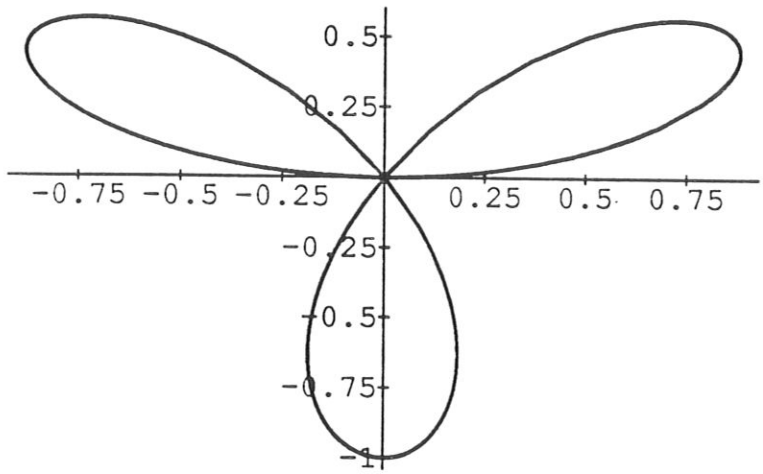
ISB, WIR90

Beispiel: **Polarkoordinaten:**

```
r(t)  = sin(3t), r = Radius, t = Winkel.
x     = r(t) cos(t) = sin(3t) cos(t)
y     = r(t) sin(t) = sin(3t) sin(t)
```

In[1]:=

```
ParametricPlot[{Sin[3 t] Cos[t],Sin[3 t] Sin[t]},{t,0,Pi}]
```



Out[1]=

```
-Graphics-
```

ParametricPlot[{fx,fy},{t,tmin,tmax}] Plot mit x-Koordinate fx und y-Koordinate fy, Variable t.

ParametricPlot[{{fx,fy},{gx,gy},{... ...}},{t,tmin,tmax}] Mehrere Funktionen f, g etc im selben Plot.

ParametricPlot[{fx,fy},...,{t,tmin,tmax},AspectRatio -> Automatic] Versucht die ***Gestalt*** der Kurve zu bewahren.

{fx,fy} = **{r[t] Cos[t], r[t] Sin[t]}** Polarkoordinaten Radius r, Winkel t

{fx,fy} = **{Re[f], Im[f]}** Plot einer komplexen Funktion

{fx,fy} = **{Log[x], Log[f]}** Logarithmische Skalen etc.

Weitere Nützliche Dinge:

Display["file", plot] Speichert den Plot in ***PostScript***-Form ins File

Options[plot] Zeigt die Options des genannten Plots (Z.B. Options[%4])

Weitere Nützliche Dinge:

InputForm[plot] Zeigt die über den Plot gespeicherte Information

Show[plot, option -> value] Zeichnet den Plot neu mit dem geänderten Wert der Option.

MathematicaBefehl[Argumente,...., OptionName -> Wert]

z.B. **Plot[....,Option ->Wert]** Aendert die default-Optionen.

Options[MathematicaBefehl], z.B. `Options[Plot]`

Zeigt die gesetzten Options

SetOptions[MathematicaBefehl, Option -> Wert,...]

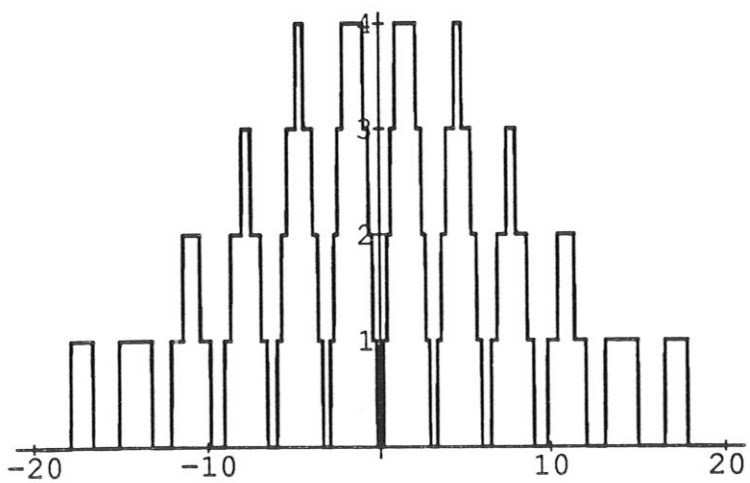
z.B. `SetOptions[Plot, PlotPoints -> 50]` Setzt Defaults anders

Spezielle Funktionen (kleine Auswahl zur Zahlenlehre):

Round[x]	Runden	**FactroInteger[n]**	bekannt
Floor[x]	Gauss-Klammer: Kleinste ganze Zahl kleiner als x		
Ceiling[x]	Grösste ganze Zahl grösse als x		
Sign[x]	Signum	**GCD[n1,n2,..]**	g.g.T.
Abs[x]	Absolut-Betrag	**LCM[n1,n2,...]**	k.g.V
PowerMod[a,b,n]	Potenz a^b modulo n	**Mod[k,n]**	k modulo n
Quotient[m,n]	Ganzzahl-Teil von m/n	**Prime[k]**	k-te Primzahl
Divisors[n]	Liste der Teiler von n	**PrimeQ[n]**	True falls n prim ist
Binomial[n,m]	Binomialkoeffizient	**n!**	Fakultät
EvenQ[x]	Test, ob x gerade	**OddQ[x]**	Test auf ungerade

In[14]:=

```
Plot[Floor[5 Abs[Sin[x] 1/((0.01 x^2) + 1)]],{x,-20,20}]
```



Out[14]=

```
-Graphics-
```

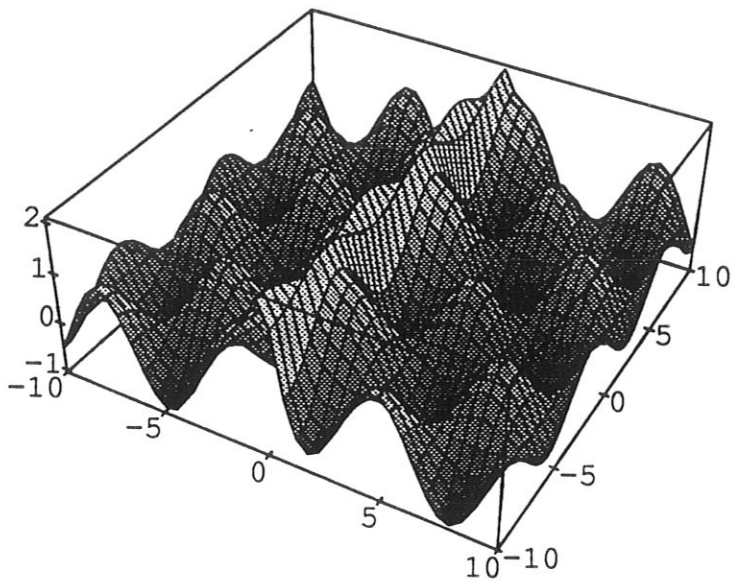
Dreidimensionale Plots mit *Mathematica*

ISB, WIR90

Generierung eines Plots, Beispiel:

(Gezeigt wird das Bild einer Funktion zweier Variablen f(x,y).)

```
Plot3D[Sin[x] Sin[0.8 y]+1/(x^2+0.5),{x,-10,10},
       {y,-10,10},PlotPoints -> 40]
```



Mathematica-Befehle:

Plot3D[f[x,y],{x,xmin,xmax},{y,ymin,ymax},*Option1,Option2,...*]
Gibt einen 3-dimensionalen Plot der Funktion f von x und y

Show[%,Option_n,...] Gibt den Plot wieder mit geänderter Option (Option_n)

Optionen und vorgegbene Einstellungen (default-Werte)

PlotRange (default **Automatic**): Wertebereich für den Plot. Möglichkeiten: All, {zmin,zmax} oder {{xmin,xmax},{ymin,ymax},{zmin,zmax}}

Optionen und vorgegebene Einstellungen (default-Werte) Fortsetzung

PlotLabel (default **None**): Label für den Plot (Titel)
Framed (default **False**): Ob Rahmen um den Plot oder nicht
AspectRatio (default **1**): Achsenverhältnis für den ganzen Plot
PlotColor (default **True**): Farbgraphik ja - nein
PlotPoints (default **15**): Anzahl Punkte in jeder Richtung in denen die Funktion berechnet wird. $\{n_x,n_y\}$: Verschiedene Werte in x- und y-Richtung
ViewPoint (default **{1.3,-2.4,2}**): Punkt von dem aus auf die Fläche geblickt wird. Möglichkeiten: {0,-2,0} -> direkt von vorn/ {0,-2,2} -> von vorne oben/ {0,-2,-2} -> von vorne unten/ {-2,-2,0} -> von der linken Ecke/ {2,-2,0} -> von der rechten Ecke/ {0,0,2} -> direkt von oben
Boxed (default **True**): Ob eine "Schachtel" um die Fläche gezeichnet werden soll
BoxRatios (default **{1,1,0.4}**): Seitenlängenverhältnisfür die "Schachtel"
HiddenSurface (default **True**): Ob die verdeckten Flächenteile unsichtbar sind
Shading (default **True**): Ob die Oberfläche schattiert oder weiss sein soll
Mesh (default **True**): Ob ein *xy* Netz auf die Oberfläche gezeichnet werden soll
Lighting (default **False**): Ob eine simulierte Beleuchtung eingeschaltet werden soll
AmbientLight (default **GrayLevel[0.]**): Umgebendes isotropes Licht
LightSources (z.B. **{{{-1,-1,1},GrayLevel[0.8]}}**): Richtung und Farbe der Punktlichtquellen z.B. **{{{x1,y1,z1},i1},{{x2,y2,z2},i2},...}**
GrayLevel[*Graunummer*]: Graunummer zwischen 0 (schwarz) und 1 (weiss)
RGBColor[*r,g,b*]: r: Zahl für rot, g: Zahl für grün, b: Zahl für blau, Intensität 0 bis 1
ClippFill (default **Automatic**): Legt fest, wie die abgeschnittenen Oberflächenteile präsentiert werden sollen. **None**: weglassen, Durchsicht/ **Automatic**: gleiche Schattierung wie eine Oberfläche an dieser Stelle haben würde/ **GrayLevel[i]**: abgeschnittener Teil mit spezieller Graustufe/ **RGBColor[*r,g,b*]**: spezielle Farbe, rot grün blau/ **{unten, oben}**: verschiedene Spezifikation für abgeschnittene Teile unten oder oben

Weitere Plot3D-Befehle:

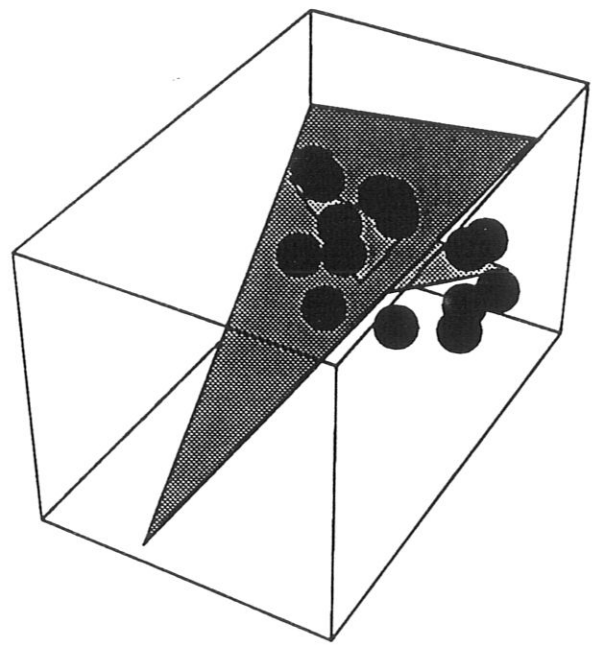
Bsp. mit Schattierungsfkt.: Plot3D[{Cos[x y], GrayLevel[x/4]},{x,0,3} ,{y,0,3 }]
Plot3D[{Funktion,Schattierungsfkt.},{x,xmin,xmax},{y,ymin,ymax}]
ListPlot[{{z_{11},z_{12},...},{z_{21},z_{22},...},....}] Gibt 3-dim. Plot vom Array "z_{xy}"
Show[Graph.,Opt.], z.B. **Show[%3,%4,%6, Mesh -> False]**: Gibt 3D-Graphiken
Graphic3D[Liste]: Mit **Graphics[Liste]** auf Spezialseite später behandelt
DensityPlot[Funkt.,{x,xmin,xmx},{y,ymin,ymax}]: Höhe als Dichte, 2D
ContourPlot[Funkt.,{x,xmin,xmax},{y,ymin,ymax}]: Höhenkurven, 2D.
Dazu **Optionen: PlotPoints** (default **15**), **PlotRange** (default **Automatic**), **ContourLevels** (default **15**, Linien zwischen zmin und zmax), **Contourspacing** (default **Automatic**, Distanz in z zwischen Linien)
ListContourPlot[*Array*]: Entsprechend ListPlot und ContourPlot
ListDensityPlot[*Array*]: Ensprechend ListPlot und DensityPlot

Zeichnen mit *Mathematica*

ISB, WIR90

Beispiel einer Zeichnung: Flächen und Kugeln, 3D:

```
rp[n_] := Polygon[Table[Random[ ],{n},{3}]];
k = Table[Point[{Random[ ],Random[ ],Random[ ]}],{20}];
l = Polygon[{{0.5,0,1},{0,-1,-0.5},{-0.5,1.5,0.5},{1,1,1}}]
Show[Graphics3D[{PointSize[0.06],rp[3],l,k}]]
```



-Graphics3D-

Mathematica-Befehle:

Show[Liste..], z.B. **Show**[%..,%...,**Optionen**], Show[**Graphics**[...],...], Show[**Graphics3D**[...],...] etc.

InputForm[...], z.B. InputForm[%]: Zeigt die interne Form der Graphik (Grundelemente, aus denen die Graphik aufgebaut ist)

Graphics[Liste] *oder* **Graphics3D[Liste]:** Zusammensetzen einer Liste von Graphik-Grundelementen zu einer Graphik, 2D resp. 3I

Die **Grundelemente** können sein (**2D** resp. **3D**):

Point[{x,y}]: Zeichnet den Punkt (x, y)

Point[{x,y,z}]: Zeichnet den Punkt (x, y, z)

Line[{{x1,y1}, {x2,y2}, ...}]: Zieht eine Linie durch die angegebenen Punkte

Weitere **Grundelemente** können sein (**2D** resp. **3D**):

Line[{{x1,y1,z1}, {x2,y2,z2}, ...}]: Linie durch die angegebenen Punkte
Rectangle[{xmin,ymin}, {xmax,ymax}]: Gibt gefülltes Rechteck
Polygon[{{x1,y1}, {x2,y2}, ...}]: Gefülltes Polygon mit angegebenen Ecken
Polygon[{{x1,y1,z1}, {x2,y2,z1}, ...}]: Gefülltes Polygon mit angegebenen Ecken
CellArray[{{a11,a12,...},{a21,...},...}]: Rechteckiches Feld von Rechtecken, unten beginnend mit a11, a12,.. , wobei die Zahlen a11 etc. als Graustufe genommen werden (Zahl muss zwischen 0 und 1 liegen. 0 :schwarz, 1: weiss.)
CellArray[Array,{{xmin,ymin}.{xmyanymax}},{zmin.zmax}]: Array von Graustufen zwischen zmin und zmax, gezeichnet in Rechtecke definiert durch xmin,ymin, xmyx, ymax
Text[Textausdruck,{x,y}]: Zentriert Textausdruck um (x,y)
Text[Textausdruck,{x,y},{xr,yr}]: xr, yr bezeichnen die Zentrierungsart des Textes um x, y: {xr,yr}={-1,0} links an {x,y}, {1,0}: rechts, {0,1}: oben, {0,-1}: unten
{x,y}: Absolute Koordinatenwerte
Scaled[{x,y}]: Von 0 bis 1 skalierte Koordinaten, in jede Richtung
GrayLevel[n]: Das **folgende Graphik-Objekt** hat die Graustufe n (0: scharz, 1: weiss, - z.B. GrayLevel[0]: schwarz, GrayLevel[0.5]: grau
RGBColor[r,g,b]: r, g, b sind die Intensitäten von rot, grün, blau. Zahl zwi. 0 und 1 - z.B. RGBColor[1,0,0]: rot, [0,1,0]: grün, [0,0,1]: blau, [1,1,0] gelb, [0,1,1]: cyan, [1,0,1]: magenta
HSBColor[f,s,h]: Farbe: Spezifikation von Färbung f, Sättigung s und Helligkeit h
PointSize[n]: Gibt dem **Punkt** eine Grösse in Relation zur Bildgrösse
Thickness[t]: Gibt der **Linie** eine Dicke in Relation zur Bildgrösse - z.B. [0.05]: dick
Dashing[{d1,d2,...}]: Linie "gestrichelt", Teillängen d1, d2,.. in Relat. zu Bildgrösse - z.B. Dashing[{0.01,0.05,0.05,0.05}]: strichpunktiert, [{0.05,0.05}]: gestrichelt
FaceForm[gvorn,ghinten]: 3D-Polygon vorn/ hinten mit angegebener Graustufe resp. RGBColor
EdgeForm[ ]: Lässt die Kanten eines Polygons weg (nur Graustufe...)
EdgeForm[Instruktion]: Kanten in spez. Graustufe/ RGBColor nach Instruktion

Optionen:

PlotRange: Wertebereich **{ymin, ymax}** oder Definitions- und Wertebereich **{{xmin, xmax}, {ymin, ymax}}**. **Default: Automatic** (d.h. interne Algorithmen werden verwendet: "Interessante Gebiete").
AspectRatio: Verhältnis **Höhe:Breite**. **Default: 1/GoldenRatio**. (**Automatic!**)
Axes: Welche **Achsen zu nehmen** sind. **Default: None**, nimmt spezifizierte {x,y}. (Gegenteil von None: **All.**)
PlotColor: Ob **Farb-Graphik** oder nicht.
Default: True (soll farbig gemacht werden).
RenderAll: Ob unsichtbare Teile auch zu zeichnen sind. **False**: Zeichnet nur die Polygone resp. Teile, die im fertigen Bild sichtbar sind.
Default: True: Zeichnet alle Bilder, angefangen hinten

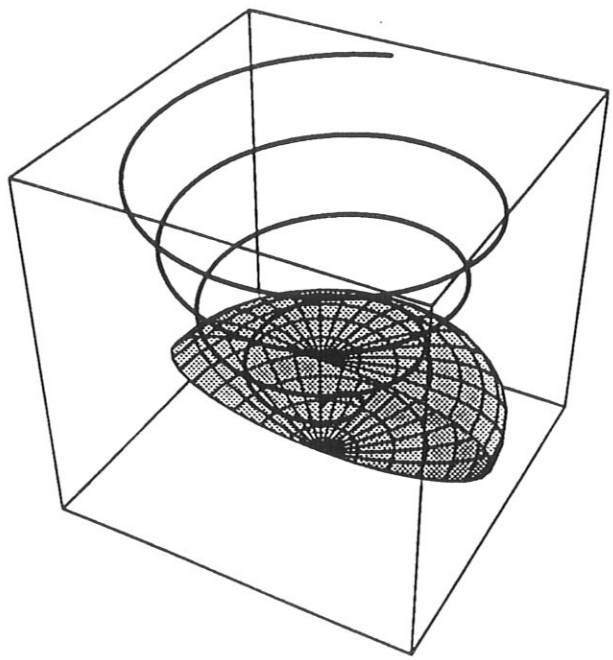
Etc. etc.

Verwendung externer Programme mit *Mathematica*

ISB, WIR90

Beispiel:

```
<<ParametricPlot3D.m;
a = ParametricPlot3D[{5 Sin[u] Cos[v],5 Sin[u] Sin[v],
          5 Cos[u]},
        {u,0,Pi,Pi/15},{v,0,Pi,Pi/15},BoxRatios -> {1,1,1}];
b = SpaceCurve[{u Sin[u^2], u Cos[u^2],u^2},{u,0,5,0.01},
BoxRatios->{1,1,1}];
Show[a,b]
```



-Graphics3D-

Allgemeines zum Arbeiten mit *Mathematica*:

Wie bei andern 4. Generations-Paketen hat man auch hier die Möglichkeit a) zum **interaktiven Arbeiten** im Foreground (bis jetzt praktiziert) *und* b) zum **Programmieren**. Die Programme stehen in speziellen Dateien, die

[Allgemeines zum Arbeiten mit *Mathematica*: (Fortsetzung)]
die Extention ".m" (filename.m) haben müssen und bei Gebrauch geöffnet werden. Sie enthalten "Prozeduren", durch die neue, selbstgemachte und aufrufbare *Mathematica*-Befehle geschaffen werden.

Vorgehen bei Verwendung eines externen Programmes:

a) **File öffnen** mit "**<<Filename.m**", im Beispiel: <<ParametricPlot3D.m .
b) Die durch das Paket **zur Verfügung gestellten Befehle verwenden.** Im Beispiel: SpaceCurve[....] etc.

In *Mathematica* stehen viele Pakete zur Verfügung.

Hier einige **Beispiele.** (Sei **data** eine Liste von Daten.)

Fourier.m : Symbolische Fourier-Transformationen und Inverse, Befehle:
Fourier[data]: Numerische Fourier-Transformation
InverseFourier[data]: Inverse Fourier-Transformation

Laplace.m : Laplace-Transformationen und Inverse

ODE.m : Funktionen zur Manipulation und zum Lösen von gewöhnlichen Differentialgleichungen

RungeKutta.m : Numerische Lösung von Differentialgleichungen

Trigonometry.m : Trigonometr. Identitäten und Vereinfachungsregeln

VectorAnalysis.m : Vektoranalysis

Statistics.m : Statistik, Befehle:
Mean[data]: Mittelwert
Median[data]: Median
Variance[data]: Varianz
StandardDeviation[data]: Standardabweichung
Fit[{y1,y2,...},{f1,f2,...},x] : Linearkombination der f1, f2,.. als Regressionskurve zu den Werten y1, y2,...
Fit[{{x1,y1},{x2,y2},...},{f1,f2,...}] : "Beste" Linearkombination der f1,f2,.. für die Punkte {x1y1},{x2,y2},... etc.

Polyhedra.m : Geradeflächig begrenzte Körper

ParametricPlot3D.m : 3-dimensionale "parametrisierte Plots"
ParametricPlot3D[...] : 3-dimensionale parametrisierte Flächen
PointParametricPlor3D[...] : Stellt Fläche durch Punkte dar
SpaceCurce[...] : 3-dimensionale Kurve

Cross.m : Vektorprodukt

Vectors.... etc.. Weitere Programme: ->Inhalt des Vezeichnisses "**Packages**"

Ein Beispiel von File-Handling mit *Mathematica* auf DOS

(Version1) WIR91

Inhalt:
Bearbeiten von *Mathematica*-Code ausserhalb von *Mathematica*, anschliessender Import ins *Mathematica* und Ausführung der bearbeiteten Befehlssequenz.

Schirm 1: ***Mathematica*** starten, Code eingeben. (Im Beispiel wird der absolute Fehler von p(r, s, q) für einen gegebenen Werte-Vektor berechnet.

Schirm 2: Abspeichern der Session mit Hilfe von F10 in ein File. (Zu beachten: ***Mathematica***-Files brauchen die Extention "m". Bei Zugriff auf dem System unbekannte Pfade muss der File-Name mit dem Pfad angegeben werden.)

Schirm 3: ***Mathematica*** ist inzwischen mit "Quit" verlassen worden, um die Session mit einem Editor eigener Wahl bearbeiten zu können. Wir haben mit "nc" den Norton-Commander gestartet und sehen unten im Directory das File "test4.m". Wir markieren das File mit der Maus und klicken unten auf "Edit", um das File zu editieren.

Schirm 4: Wir sind im Editor des Norton-Commanders und betrachten den ***Mathematica***-Code.

Schirm 5: Wir streichen alles raus, das nicht Code ist. Titel, Kommentare wie In[..], Out[..] etc.. Die Sequenz muss so verändert werden, dass sie schliesslich aus einer durch ";" getrennte Befehlsfolge besteht, die das ***Mathematica*** als eine *Zeile* interpretieren kann. Mit "Save" verlassen wir den Editor (abspeichern), steigen mit "Quit" aus dem Norton-Commander aus und starten dann von neuem ***Mathematica***.

Schirm 6: Wir sind wieder in ***Mathematica***. mit "<<" importieren wir das File (im Beispiel "<<test4.m"). Eventuell ist hier wieder der zuerst der Pfad anzugeben. ***Mathematica*** führt dann die Befehlssequenz direkt aus und gibt nach "Out" das *Resultat* aus. Mit "!!" Können wir den File-Inhalt anschauen. (Im Beispiel "!!test4.m".)

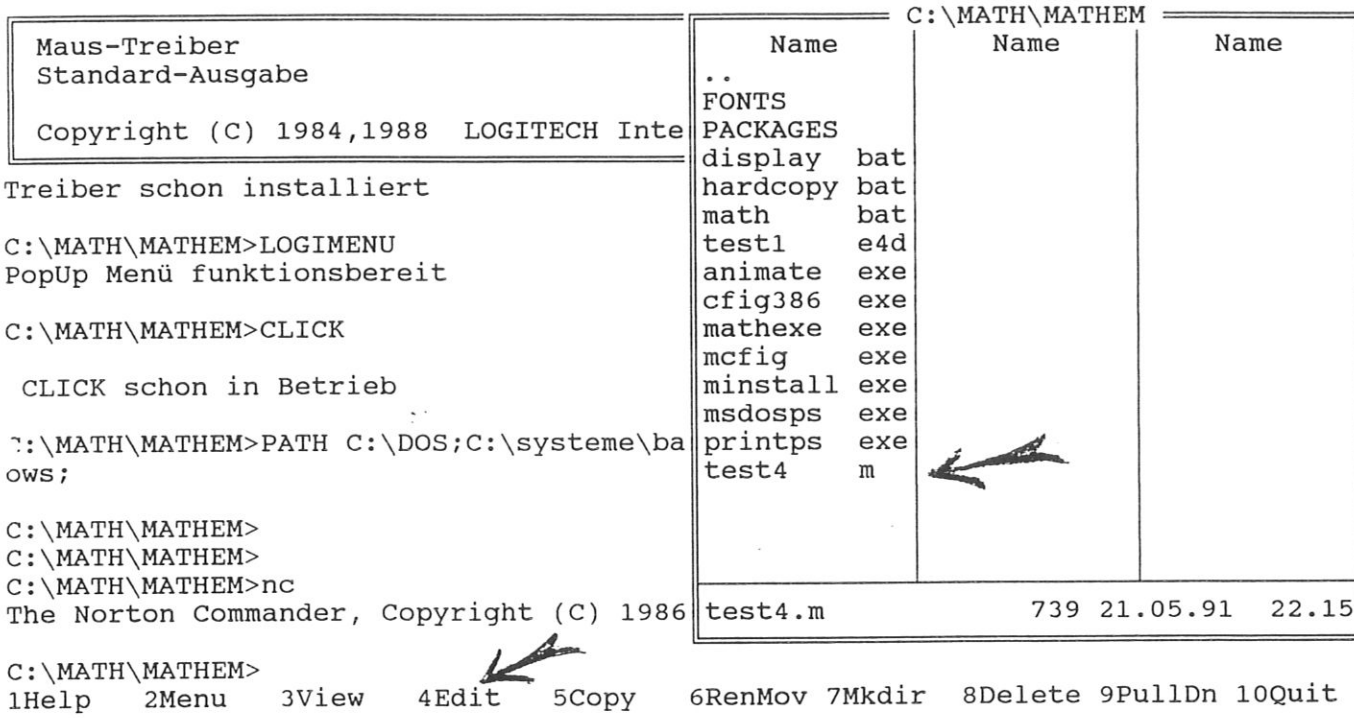
Schirm1

```
C:\MATH\MATHEM>mathem

C:\MATH\MATHEM>ECHO OFF
Mathematica (MS-DOS 386/7) 1.2 (September 27, 1989) [With pre-loaded data]
by S. Wolfram, D. Grayson, R. Maeder, H. Cejtin,
   S. Omohundro, D. Ballman and J. Keiper
with I. Rivin, D. Withoff and T. Sherlock
Copyright 1988,1989 Wolfram Research Inc.

In[1]:=
p[r_,s_,q_]:=1/2 r s^2 (r-s) + q (r+s)(r-s);
dp[dr_,ds_,dq_,r_,s_,q_]:=dr Abs[D[p[r,s,q],r]]+ds Abs[D[p[r,s,q],s]]+
         dq Abs[D[p[r,s,q],q]];
dp[dr,ds,dq,r,s,q]//.{dr->0.035,ds->0.015,dq->0.00002,r->46.245,
         s->45.015,q->1.32915}


In[2]:=
```

Schirm2

```
p[r_,s_,q_]:=1/2 r s^2 (r-s) + q (r+s)(r-s);
dp[dr_,ds_,dq_,r_,s_,q_]:=dr Abs[D[p[r,s,q],r]]+ds Abs[D[p[r,s,q],s]]+
         dq Abs[D[p[r,s,q],q]];
dp[dr,ds,dq,r,s,q]//.{dr->0.035,ds->0.015,dq->0.00002,r->46.245,
         s->45.015,q->1.32915}

      ┌──────────────────────────┐
      │ Write session log to file:
In[2] │ test4.m
In[2] └──────────────────────────┘
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:=
In[2]:= Quit
```

Schirm3



Schirm4

```
Edit: C:\math\mathem\test4.m          *  Line 15    Col 1     25'960 Free   EOF
Mathematica (MS-DOS 386/7) 1.2 (September 27, 1989) [With pre-loaded data]
by S. Wolfram, D. Grayson, R. Maeder, H. Cejtin,
   S. Omohundro, D. Ballman and J. Keiper
with I. Rivin, D. Withoff and T. Sherlock
Copyright 1988,1989 Wolfram Research Inc.

In[1]:=
p[r_,s_,q_]:=1/2 r s^2 (r-s) + q (r+s)(r-s);
dp[dr_,ds_,dq_,r_,s_,q_]:=dr Abs[D[p[r,s,q],r]]+ds Abs[D[p[r,s,q],s]]+
         dq Abs[D[p[r,s,q],q]];
dp[dr,ds,dq,r,s,q]//.{dr->0.035,ds->0.015,dq->0.00002,r->46.245,
         s->45.015,q->1.32915}
```

1Help 2Save 3 4 5 6 7Search 8 9 10Quit

Schirm5

```
Edit: C:\math\mathem\test4.m          *  Line 1     Col 1     26'227 Free   112
p[r_,s_,q_]:=1/2 r s^2 (r-s) + q (r+s)(r-s);
dp[dr_,ds_,dq_,r_,s_,q_]:=dr Abs[D[p[r,s,q],r]]+ds Abs[D[p[r,s,q],s]]+
         dq Abs[D[p[r,s,q],q]];
dp[dr,ds,dq,r,s,q]//.{dr->0.035,ds->0.015,dq->0.00002,r->46.245,
         s->45.015,q->1.32915}
```

```
1Help   2Save   3       4       5       6       7Search 8       9       10Quit
```

Schirm6

```
C:\MATH\MATHEM>mathem

C:\MATH\MATHEM>ECHO OFF
Mathematica (MS-DOS 386/7) 1.2 (September 27, 1989) [With pre-loaded data]
by S. Wolfram, D. Grayson, R. Maeder, H. Cejtin,
   S. Omohundro, D. Ballman and J. Keiper
with I. Rivin, D. Withoff and T. Sherlock
Copyright 1988,1989 Wolfram Research Inc.

In[1]:= <<test4.m

Out[1]= 2354.02

In[2]:= !!test4.m
p[r_,s_,q_]:=1/2 r s^2 (r-s) + q (r+s)(r-s);
dp[dr_,ds_,dq_,r_,s_,q_]:=dr Abs[D[p[r,s,q],r]]+ds Abs[D[p[r,s,q],s]]+
         dq Abs[D[p[r,s,q],q]];
dp[dr,ds,dq,r,s,q]//.{dr->0.035,ds->0.015,dq->0.00002,r->46.245,
         s->45.015,q->1.32915}



In[2]:=
```