marxer engineering & computing

www.mec.li

Mathematica Tour

siehe auch: *Mathematica* Help Tour, www.wolfram.com

Einleitung

Mathematica ist ein Softwarepaket, das weltweit schon über eine Million Nutzer hat. Seit Lancierung des Produkts im Jahre 1988 wurde es immer weiter entwickelt und findet immer mehr Anwendungen in immer mehr Gebieten. Auf Grund seiner Vielseitigkeit und den vielen Erweiterungsmöglichkeiten vermag es viele weitere Softwarewerkzeuge und Tools zu ersetzen. Da *Mathematica* eine interaktive Entwicklungsumgebung ist, braucht die Entwicklung von Algorithmen und die Durchführung von Berechnungen im Vergleich zu herkömmlichen Programmiersprachen wie C++ oder Java viel weniger Zeit.

Es wird in dieser Kurzübersicht nicht möglich sein, die vielen Eigenschaften - symbolische und numerische Berechnungen, graphische Darstellungen, Einlesen von Dateien vieler Formate, Kommunikation mit externen Programmen, Integration von externem Code, Präsentation der Notebooks und mathematischen Formeln etc als HTML Dateien im Internet, Lesbarkeit der entwickelten Notebooks für Jedermann mit dem freien *MathReader*, *Mathematica* als funktionale, prozedurale, object oriented Programmiersprache - umfassend darzustellen. Nur ein Kratzen an der Oberfläche und ein Andeuten von Möglichkeiten wird gelingen. Es darf jedoch darüber nicht vergessen werden, dass in *Mathematica* tausende von Funktionen eingebaut oder via Packages zuladbar sind, und damit für viele Anwendungen schon fertig entwickelt sind. Die über 1500 Seiten umfassende Dokumentation - die übrigens vollständig in der Anwendung online integriert ist - gibt übersichtlich Auskunft.

Auf den folgenden Seiten wird - auf Grund des limitierten Platzes - ein Schwerpunkt darauf gelegt, zu zeigen, wie mit wenigen Zeilen Code schon beachtliche Resultate erzeugt werden können. Auf Papier wird es jedoch nicht möglich sein, Animationen, Sound sowie die Kommunikation mit externen Programmen befriedigend darzustellen.

Ich hoffe, dass ich mit dieser kurzen Übersicht einen ersten Eindruck vermitteln kann und den Appetit wecke, sich intensiver mit den Möglichkeiten von *Mathematica* auseinanderzusetzen. Die Homepage von Wolfram Research (www.wolfram.com) gibt umfassend Auskunft.

Über diese Zusammenfassung

Wenn *Mathematica* interaktiv benutzt wird, wird standardmässig der Input in einer Input Zelle, die mit In[n] (für n-ten Input), und der von *Mathematica* berechnete Output in einer Output Zelle (Out[n]) dargestellt.

In dieser *Mathematica* Tour wurde - um auf gleichem Platz mehr Informationen unterzubringen - ein aktives Button Objekt kreiert, das es ermöglicht, den Input in blau und den Output in grün in einer Zeile und als Gleichung darzustellen.In Einzelfällen wird der Input bzw. Output doppelt in verschiedenen Formen (InputForm, StandardForm) dargestellt.

```
Sin[Pi / 2]
```

```
1
```

- **Definition der Button Funktion (nicht gezeigt)**
- **Der Input und Output sieht damit folgendermassen aus.**

$$\text{Sin}\left[\frac{\pi}{2}\right] = \text{Sin[Pi / 2]} = 1$$

Mathematica Streifzug durch grundlegende Operationen

Einfache Rechenoperationen

```
3 + 7 = 10
```

```
Sin[2.1` π] = 0.3090169943749472`
```

```
57.1`^100 = 4.609043686613955`*^175
```

Bemerkung: ` zeigt machine-precision an.

Rechnen mit komplexe Zahlen

$$\frac{(3 + i\,5)^2}{4 + 8\,i} = \frac{11}{5} + \frac{31\,i}{10}$$

Die Berechnung der Zahl Pi auf 300 Stellen genau

```
N[π, 300] = N[Pi, 300] =
 3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706798214808651328230
  64709384460955058223172535940812848111745028410270193852110555964462294895493038196442881097566593344612847564823378
  83165271201909145648566923460348610454326648213393607260249141 27
```

Berechnung des exakten Werts der Fakultät von 90

```
90! =
 14857159644817614973095227336208257378855699612846887669422168637049853930940658765455
  99213137088405964561723446997811200000000000000000000
```

Mathematica kennt Grad, griechische Buchstaben ...

$$\alpha = 30\,°;\ \text{Sin}[\alpha] = \frac{1}{2}$$

Mathematica kann mit Präfix rechnen und Einheiten umrechnen

```
Kilo = 10^3; Milli = 10^-3;
```

```
5 Kilo Meter + 3000 Milli Meter = 5003 Meter
```

Umwandeln von algebraischen Ausdrücken

$$\text{Expand}[(1 + x + 3\,y)^4] = 1 + 4\,x + 6\,x^2 + 4\,x^3 + x^4 + 12\,y + 36\,x\,y + 36\,x^2\,y + 12\,x^3\,y + 54\,y^2 + 108\,x\,y^2 + 54\,x^2\,y^2 + 108\,y^3 + 108\,x\,y^3 + 81\,y^4$$

$$\text{Factor}[x^{10} - 1] = (-1 + x)\,(1 + x)\,(1 - x + x^2 - x^3 + x^4)\,(1 + x + x^2 + x^3 + x^4)$$

$$\text{Apart}\left[\frac{(x - 1)^2\,(2 + x)}{(1 + x)\,(x - 3)^2}\right] = 1 + \frac{5}{(-3 + x)^2} + \frac{19}{4\,(-3 + x)} + \frac{1}{4\,(1 + x)}$$

Rekursive Berechnung

```
Nest[1/(1 + #1) &, x, 4] = 1/(1 + 1/(1 + 1/(1 + 1/(1 + x))))
```

Skalarprodukt und Vektorprodukt

```
a = {4, 9, 10}; b = {3, 4, 5};

a.b = 98

a × b = {5, 10, -11}
```

Marix Operationen

```
m = {{1, 2}, {3, 4}}; v = {7, 9};

Inverse[m] = {{-2, 1}, {3/2, -1/2}}

m.v = {25, 57}

Eigenvalues[m] = {1/2 (5 + Sqrt[33]), 1/2 (5 - Sqrt[33])}

SingularValueDecomposition[N[m]] // MatrixForm
```

$$\begin{pmatrix} \begin{pmatrix} -0.404554 \\ 0.914514 \end{pmatrix} & \begin{pmatrix} -0.914514 \\ -0.404554 \end{pmatrix} \\ \begin{pmatrix} 5.46499 \\ 0. \end{pmatrix} & \begin{pmatrix} 0. \\ 0.365966 \end{pmatrix} \\ \begin{pmatrix} -0.576048 \\ -0.817416 \end{pmatrix} & \begin{pmatrix} -0.817416 \\ 0.576048 \end{pmatrix} \end{pmatrix}$$

Darstellung und Berechnung der Summe

```
Sum[1/i^4, {i, 1, ∞}] = π^4/90
```

Symbolische Integration

```
∫ Sqrt[x] Sqrt[1 + x] dx = 1/4 (Sqrt[x] Sqrt[1 + x] (1 + 2 x) - ArcSinh[Sqrt[x]])
```

Numerische Integration

```
NIntegrate[Log[x + Sin[x]], {x, 0, 2}] = 0.5558893538982785
```

Ableitung einer Funktion

```
∂x (x^2 Log[x + a]) = D[x^2 * Log[x + a], x] = x^2/(a + x) + 2 x Log[a + x]
```

Berechnung der nullten bis dritten Ableitung einer Funktion

```
NestList[∂x # &, Sin[x]/x^2, 3] // ColumnForm
```

$$\frac{\sin(x)}{x^2}$$
$$\frac{\cos(x)}{x^2} - \frac{2\sin(x)}{x^3}$$
$$-\frac{4\cos(x)}{x^3} + \frac{6\sin(x)}{x^4} - \frac{\sin(x)}{x^2}$$
$$\frac{18\cos(x)}{x^4} - \frac{\cos(x)}{x^2} - \frac{24\sin(x)}{x^5} + \frac{6\sin(x)}{x^3}$$

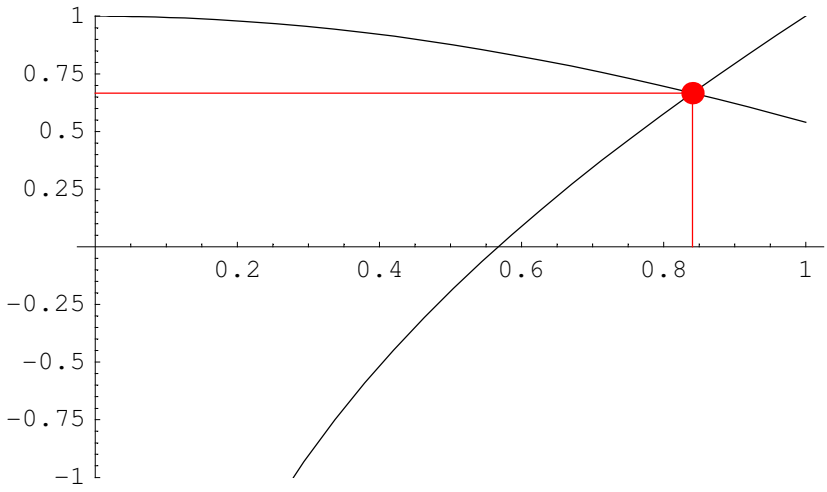
Differentialgleichungen

```
DSolve[y'[x] == a y[x] + 1, y[x], x] = {{y[x] → -1/a + e^(a x) C[1]}}
```

Berechnung der Lösungen einer Gleichung mit graphischer Darstellung im Bereich {-1,1}

```
sol = FindRoot[Cos[x] == x + Log[x], {x, 1}]
{x → 0.8406188356435845`}

xS = x /. sol; yS = Cos[xS];
Plot[{x + Log[x], Cos[x]}, {x, 0, 1}, PlotRange → {-1, 1},
  Epilog → {PointSize[0.03], Hue[1], Point[{xS, yS}],
    Line[{{0, yS}, {xS, yS}}], Line[{{xS, 0}, {xS, yS}}]}];
```



Mathematica kennt viele spezielle Funktionen

```
LegendreQ[3, x] = 2/3 - 5 x^2/2 - 1/4 x (3 - 5 x^2) Log[(1 + x)/(1 - x)]

∫ BesselJ[n, z] dz =

2^(-1-n) z^(1+n) Gamma[(1 + n)/2] HypergeometricPFQRegularized[{(1 + n)/2}, {1 + n, (3 + n)/2}, -z^2/4]
```

Mathematica weiss mit abstrakter mathematischer Notation umzugehen

```
Table[g ∘ (α_i ⊕ β_i) ⟹ ∂_i^(6-i), {i, 4}] = {g ∘ (α_1 ⊕ β_1) ⟹ ∂_1^5, g ∘ (α_2 ⊕ β_2) ⟹ ∂_2^4, g ∘ (α_3 ⊕ β_3) ⟹ ∂_3^3, g ∘ (α_4 ⊕ β_4) ⟹ ∂_4^2}
```

Listenoperationen: Bilde eine 3x3 Matrix m aus den Zahlen 1 bis 9, ...

```
(m = Partition[Range[9], 3]) // MatrixForm
```

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

... quadriere alle Matrixelemente, ...

```
(m = Map[#^2 &, m]) // MatrixForm
```

$$\begin{pmatrix} 1 & 4 & 9 \\ 16 & 25 & 36 \\ 49 & 64 & 81 \end{pmatrix}$$

... extrahiere die zweite Reihe etc.

```
Take[m, {2}, All]  =  {{16, 25, 36}}
```

Mathematica und Curve Fitting

Durch die ersten 10 Primzahlen ...

```
prims = (len = 10; Table[Prime[i], {i, len}])  =  {2, 3, 5, 7, 11, 13, 17, 19, 23, 29}
```

soll erstens ein Fit mit den drei Basisfunktionen Log[x], x, x^2...

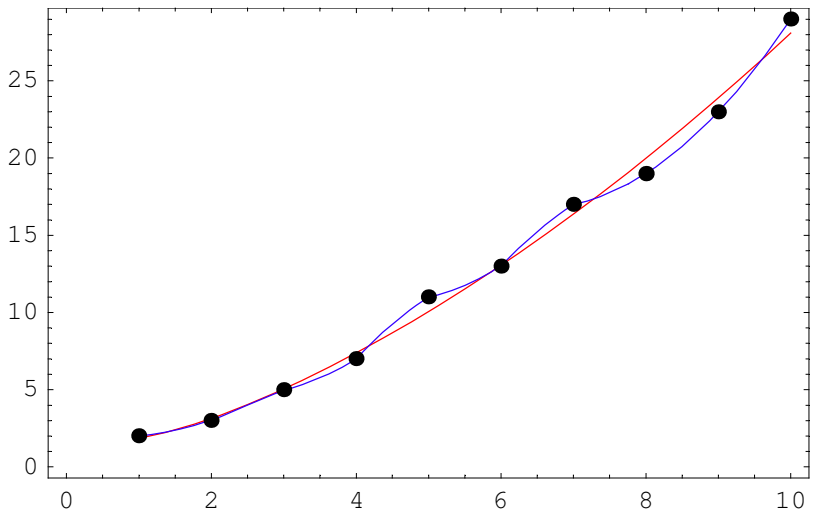
```
fitFunc = Fit[prims, {Log[x], x, x^2}, x] =
  1.7134033493543253` x + 0.13729268833721286` x^2 - 1.2046165947457952` Log[x]
```

und zweitens eine Interpolation mit einem quadratischen Polynom gelegt werden.

```
interFunc = Interpolation[prims, InterpolationOrder → 2] =
  InterpolatingFunction[{{1, 10}}, "<>"]
```

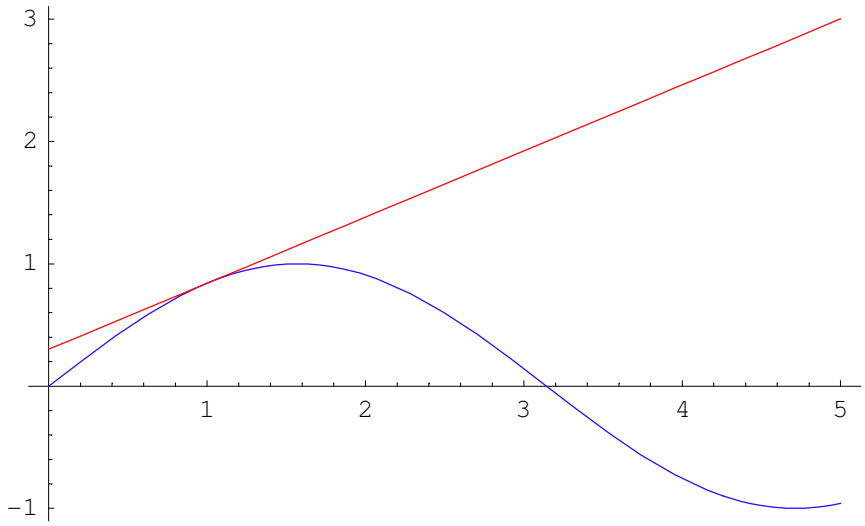
Es folgt die graphische Darstellung der Punkte und der zwei Kurven. Man sieht, dass die blaue interpolierende Kurve alle Punkte trifft, während die drei Parameter der roten Fit Kurve nur dafür sorgen können, dass alle Punkte möglichst gut getroffen werden.

```
pts = Transpose[{Range[len], prims}];
Plot[{fitFunc, interFunc[x]}, {x, 1, len}, PlotStyle -> {Hue[1], Hue[0.7]},
  Epilog -> {PointSize[0.02], Point /@ pts}, Frame -> True];
```



An die Funktion f (z.B. Sin) soll an der Stelle x0 die Tangente gezeichnet werden.

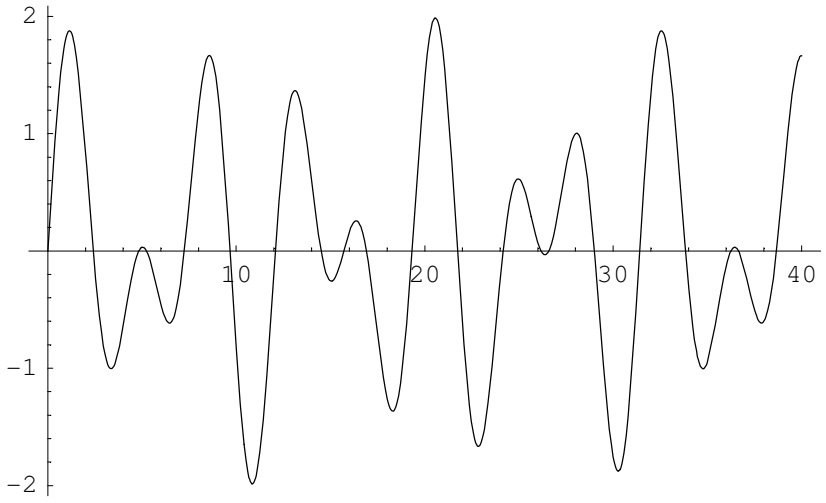
```
Clear[x, var]; x0 = 1.0; f = Sin; xRange = {x, 0., 5.};
Plot[{f[x], f[x0] + (D[f[var], var] /. var → x0) (x - x0)},
  Evaluate[xRange], PlotStyle -> {Hue[0.7], Hue[1]}];
```



Mathematica und Graphik

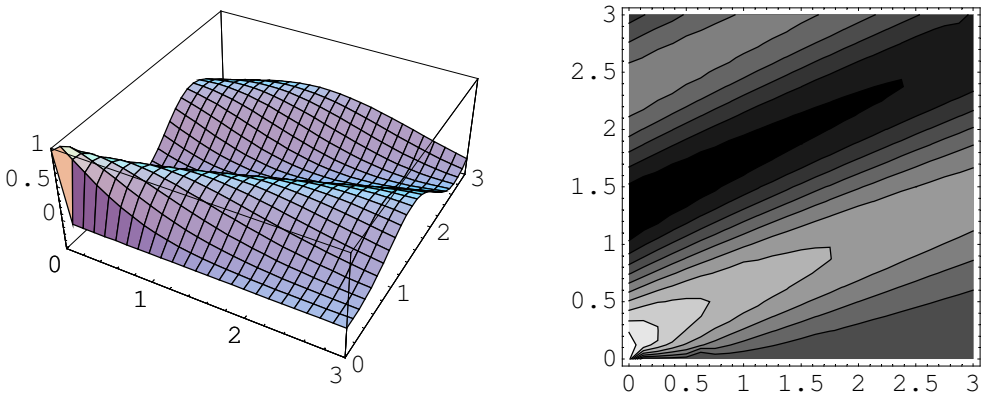
Plot einer Funktion, die von einer Variablen abhängt.

```
Plot[Sin[x] + Sin[1.6 x], {x, 0, 40}];
```



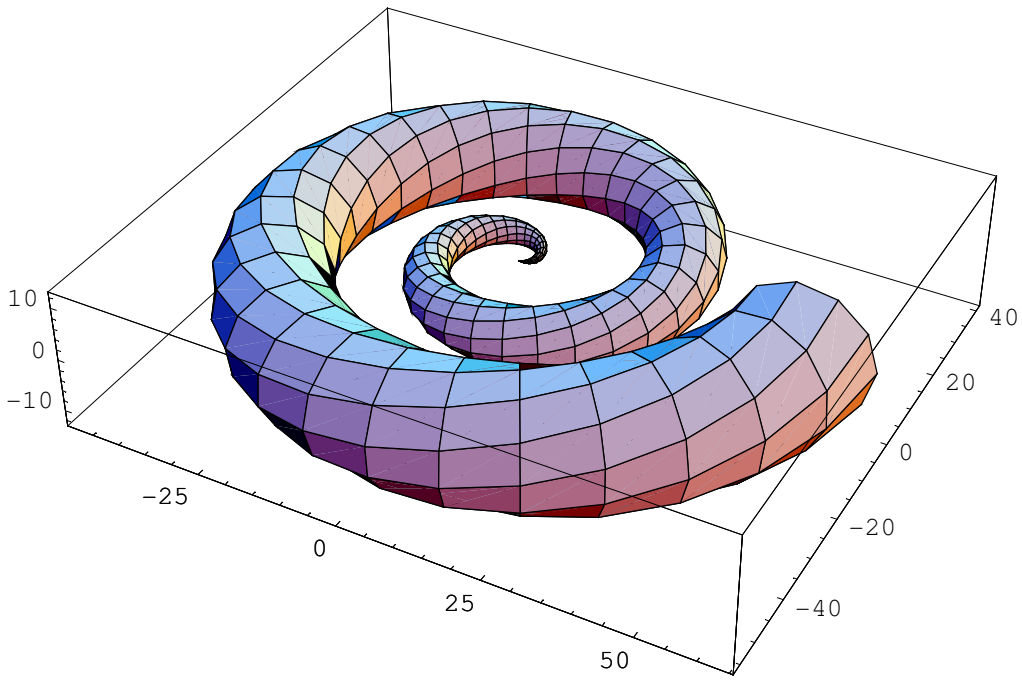
Eine Funktion, die von zwei Variablen abhängt, wird als 3D und als Contour Plot dargestellt.

```
pl=Plot3D[BesselJ[nu, 3x], {nu, 0, 3}, {x, 0, 3},DisplayFunction->Identity];
Show[ GraphicsArray[ {pl, ContourGraphics[pl]} ] ];
```



3D Darstellung eines parametrischen Plots

```
ParametricPlot3D[
  {u Cos[u] (4 + Cos[v + u]), u Sin[u] (4 + Cos[v + u]), u Sin[v + u]},
  {u, 0, 4 Pi}, {v, 0, 2 Pi}, PlotPoints → {60, 12}];
```



Mathematica und Listen

Ein wichtiger *Mathematica* Ausdruck ist eine Liste. Sie hat die Form {a1, a2, ...}. Viele Funktionen für Listen sind bereits in *Mathematica* integriert.

Beispielhaft soll im Folgenden eine Liste generiert ...

```
e1 = Table[i^2 + 3 i, {i, -4, 4}] =  {4, 0, -2, -2, 0, 4, 10, 18, 28}
```

geordnet ...

```
e2 = Sort[e1] = {-2, -2, 0, 0, 4, 4, 10, 18, 28}
```

und sowohl der Median ...

```
Median[e2] = 4
```

als auch die Summe der Listenelemente (Total) bestimmt werden.

```
Total[e2] = 60
```

Wieviele der Elemente sind gleich Null?

```
Length[Select[e1, Function[x, x == 0]]] = 2
```

Nehme die Elemente ungleich 4

```
Select[e1, Function[x, x ≠ 4]] = {0, -2, -2, 0, 10, 18, 28}
```

usw...

Die Listenfunktionen können auch bei der Analyse von Text eingesetzt werden. Es sei folgender Text gegeben.

```
t = "Die ungeheure Stärke dieser CAS Software besteht darin, dass nicht
    nur auf Zahlen, sondern auf jedem Ausdruck diese Operationen ausgeführt
    werden können. Insbesondere kann z.B. auch Text analysiert werden.";
```

Wieviele "u" enthält obenstehender Text t?

```
ttc = ToCharacterCode;
Length[Select[ttc[t], Function[x, x == ttc["u"][[1]]]]] = 9
```

Mit rot sollen sie hervorgehoben werden.

```
StringJoin[Characters[t] //. "u" -> "u"]
```

```
Die ungeheure Stärke dieser CAS Software besteht darin, dass nicht n
  ur auf Zahlen, sondern auf jedem Ausdruck diese Operationen ausgeführt
  werden können. Insbesondere kann z.B. auch Text analysiert werden.
```

Mathematica als Programmiersprache

Mathematica ist sehr flexibel und bietet umfangreiche Möglichkeiten. Oft gibt es mehrere Möglichkeiten (funktionale, prozedurale, ... Programmierung oder die Benutzung unterschiedlicher eingebauter Funktionen) zur Implementierung eines Algorithmus.

Zum Beispiel werden hier zwölf verschiedene Möglichkeiten, die Fakultät von n zu berechnen, dargestellt. Wie man sieht, liefern sie alle für x=5 den gleichen Wert.

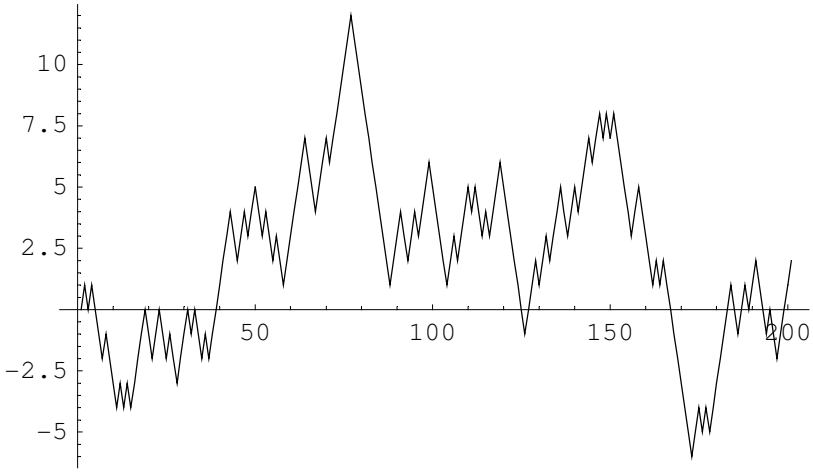
```
result = {};
Do[
  Clear[f];
  Switch[i,
   1, f := Factorial,
   2, f[n_] := n!,
   3, f[n_] := Gamma[n + 1],
   4, f[n_] := n f[n - 1] ; f[1] = 1,
   5, f[n_] := Product[i, {i, n}],
   6, f[n_] := Module[{t = 1}, Do[t = t * i, {i, n}]; t],
   7, f[n_] := Module[{t = 1, i}, For[i = 1, i <= n, i++, t *= i]; t],
   8, f[n_] := Apply[Times, Range[n]],
   9, f[n_] := Fold[Times, 1, Range[n]],
   10, f[n_] := If[n == 1, 1, n f[n - 1]],
   11, f = If[#1 == 1, 1, #1 #0[#1 - 1]] &,
   12, f[n_] := Fold[#2[#1] &, 1, Array[Function[t, # t] &, n]]
  ];
  AppendTo[result, f[5]];
  , {i, 1, 12}
 ];
Clear[f] ; result
```

```
{120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120, 120}
```

Da viele Funktionen in *Mathematica* eingebaut sind, genügen oft sehr wenige Zeilen Code, um eine Berechnung durchzuführen. Dies bedeutet erstens Effizienz und zweitens - da die Funktionen weltweit getestet und mit hoher Wahrscheinlichkeit fehlerfrei sind - grosse Zuverlässigkeit der Ergebnisse.

Ein Zweizeiler berechnet einen Random Walk mit 200 Schritten in einer Dimension und stellt ihn graphisch dar.

```
RandomWalk[n_] := NestList[(# + (-1)^Random[Integer]) &, 0, n];
ListPlot[RandomWalk[200], PlotJoined → True];
```



Mathematica und File Manipulation

Einlesen von Dateien

Mathematica kennt viele Dateiformate.

```
$ImportFormats = {"AIFF", "AU", "BMP", "CSV", "DICOM", "Dump", "DXF", "EPS", "EPSI",
  "EPSTIFF", "Expression", "ExpressionML", "FITS", "GIF", "HarwellBoeing", "HDF",
  "JPEG", "Lines", "List", "MAT", "MathML", "MGF", "MPS", "MTX", "NB", "NotebookML",
  "PBM", "PGM", "PNG", "PNM", "PPM", "SDTS", "SND", "STL", "SymbolicXML", "Table",
  "Text", "TIFF", "TSV", "UnicodeText", "WAV", "Words", "XBitmap", "XML"}
```

Deren Dateien können - wie unter Bildbearbeitung gezeigt - einfach eingelesen werden.

Finden und Kopieren von Dateien

Die Namen und Grössen der Dateien, die die Endung .jpg haben und sich in einem bestimmten Ordner befinden, sollen aufgelistet werden.

```
fn = FileNames["*.jpg", $InitialDirectory, 1];
Transpose[{Map[FileByteCount, fn], fn}] // TableForm
```

```
1089804     C:\Programme\Wolfram Research\Mathematica\5.0\DSCN1507.JPG
79180       C:\Programme\Wolfram Research\Mathematica\5.0\screenshot.jpg
31947       C:\Programme\Wolfram Research\Mathematica\5.0\Volleyball.JPG
```

Die erste Datei in obiger Liste soll unter neuem Namen in den gleichen Ordner kopiert werden.

```
CopyFile[fn[[1]], "testCopy.jpg"];
```

Man sieht, dass das File erfolgreich kopiert wurde.

```
FileNames["*.jpg", $InitialDirectory, 1] // TableForm
```

```
C:\Programme\Wolfram Research\Mathematica\5.0\DSCN1507.JPG
C:\Programme\Wolfram Research\Mathematica\5.0\screenshot.jpg
C:\Programme\Wolfram Research\Mathematica\5.0\testCopy.jpg
C:\Programme\Wolfram Research\Mathematica\5.0\Volleyball.JPG
```

Mathematica und Image Processing

Die zu bearbeitende Datei wird spezifiziert, eingelesen und dargestellt.

```
fileName =
  ToFileName[ {$InitialDirectory, "nma 5.0", "Mathematica Features"}, "volleyball.jpg"];
Show[gr1 = Import[fileName], ImageSize → {400, 400}];
```

Das Bild hat 3 Farbkanäle und mehrere hundert Zeilen und Spalten ...

```
Dimensions[gr1[[1, 1]]]
```

```
{373, 554, 3}
```

Deshalb soll nur ein Ausschnitt bearbeitet werden. Die Info Arrays enthalten Informationen für die nachfolgenden Plots.

```
xMin = 70; xMax = 220;
yMin = 70; yMax = 220;
dataArr = gr1[[1, 1]][[Range[xMin, xMax], Range[yMin, yMax]]];
infos2Arr = Sequence[gr1[[1, 2]], gr1[[1, 3]]];
infos3Arr = Sequence[gr1[[1, 2]], gr1[[1, 3]], gr1[[1, 4]]]
```

```
Sequence[{{0, 0}, {554, 373}}, {0, 255}, ColorFunction → RGBColor]
```

Auf diesem Bildausschnitt sollen der Reihe nach die folgenden Operationen durchgeführt werden:

- Original (1), Farben invertieren (2), der Rotanteil (3), der Blauanteil (4),
- Rotanteil in Grau umwandeln (5), Blauanteil in Grau umwandeln (6).

```
rG[1] = Raster[dataArr, infos3Arr];
rG[2] = Raster[255 - dataArr, infos3Arr];
rG[3] = Raster[dA = dataArr; dA[[All, All, 2]] = 0; dA[[All, All, 3]] = 0; dA, infos3Arr];
rG[4] = Raster[dA = dataArr; dA[[All, All, 1]] = 0; dA[[All, All, 2]] = 0; dA, infos3Arr];
rG[5] = Raster[dataArr[[All, All, 1]], infos2Arr, ColorFunction → GrayLevel];
rG[6] = Raster[dataArr[[All, All, 3]], infos2Arr, ColorFunction → GrayLevel];
```

Die Darstellung zeigt:

```
Show[GraphicsArray[{
    Table[Graphics[rG[i], AspectRatio → 1], {i, 1, 4}],
    Table[Graphics[rG[i], AspectRatio → 1], {i, 5, 6}]}],
  ImageSize → {450, 250}];
```

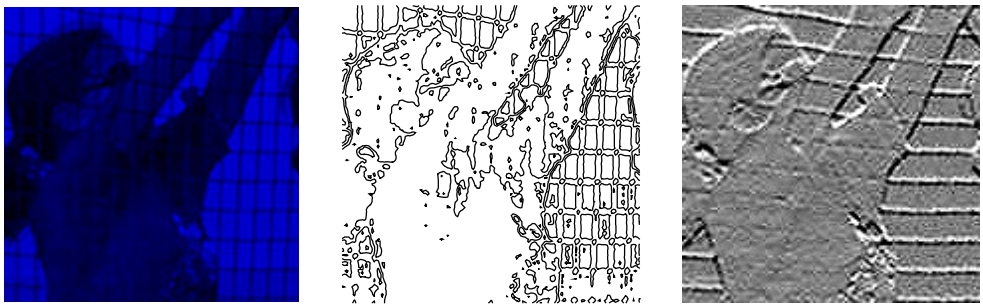
Zum Abschluss noch zwei Operationen auf dem blauen Kanal. Der Ausdruck erfolgt in Grau.

- Original (1), Contour Plot (2), Konvolution plus Density Plot (3),

```
grOption2 = Sequence[DisplayFunction → Identity, Frame → False];
gr2 = ListContourPlot[dataArr[[All, All, 3]],
   ContourShading → False, Contours → 3, grOption2];

grOption3 = Sequence[Mesh → False, Frame → False, DisplayFunction → Identity];
gr3 = ListDensityPlot[ListConvolve[{{3, -1}, {-3, 1}}, dataArr[[All, All, 3]]],
      grOption3];

Show[GraphicsArray[{Graphics[rG[4], AspectRatio → 1], gr2, gr3}]];
```

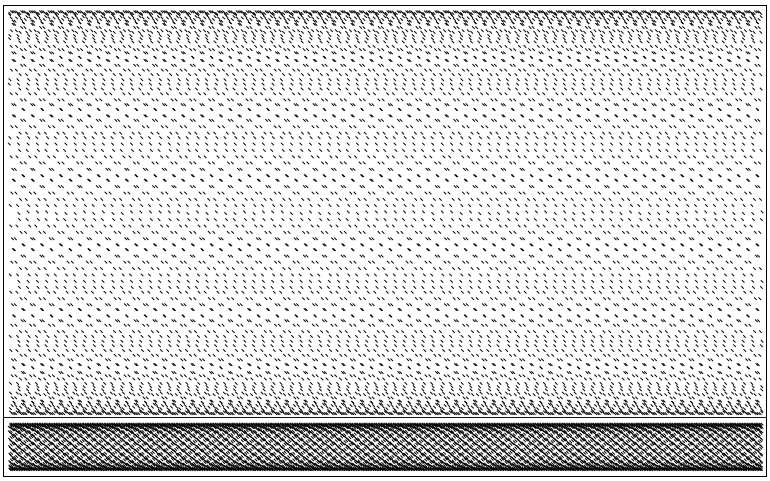


Mathematica und Sound

Generieren von Sound

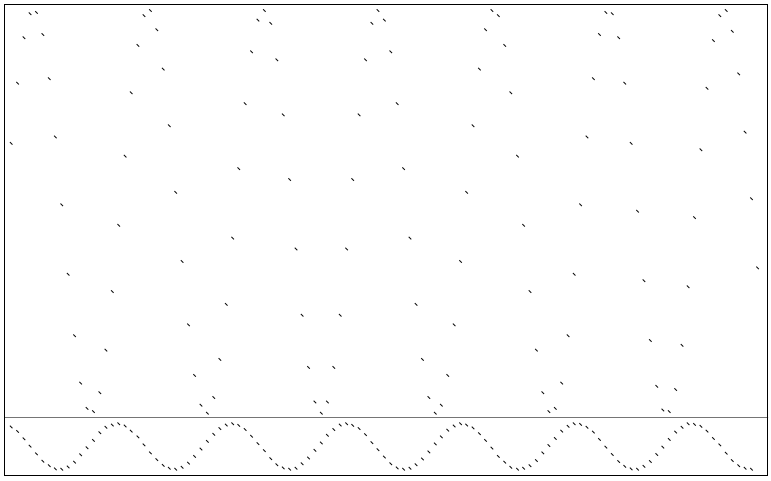
In *Mathematica* ist es ganz einfach, einen Ton zu erzeugen. Der folgende Befehl produziert einen 440Hz Ton und stellt den Amplitudenverlauf dar.

```
Play[Sin[2Pi 440 t], {t, 0, 1}];
```



Der Amplitudenverlauf ist besser sichtbar, wenn das Zeitintervall auf 15 ms gesetzt wird.

```
Play[Sin[2 Pi 440 t], {t, 0, 0.015}];
```



Aufnehmen und Bearbeiten von Sound

Mathematica ermöglicht es, über ein Mikrofon direkt Sound aufzunehmenn (Befehl: Input/Record Sound...).Der Sound kann mit Copy/Paste in *Mathematica* übernommen oder auf als wav-File abgespeichert und in *Mathematica* eingelesen werden. Die zweite Variante wird hier gewählt.

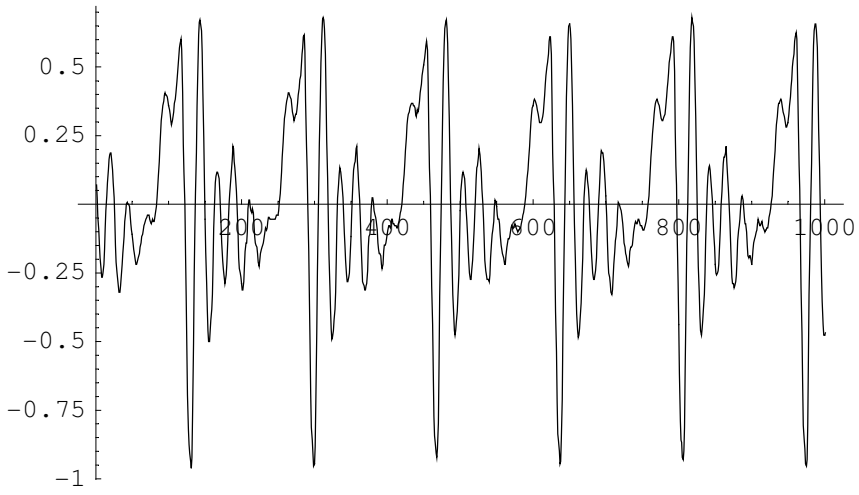
```
sA = Import[
   ToFileName[ {$InitialDirectory, "nma 5.0", "Mathematica Features" }, "soundA.wav" ] ];
```

Der Sound könnte mit Show[sA] abgespielt werden. Da die Liste der Amplitudenwerte sehr gross ist, ...

```
Length[sA[[1, 1]]]  =  112640
```

... soll nur ein Ausschnitt dargestellt werden.

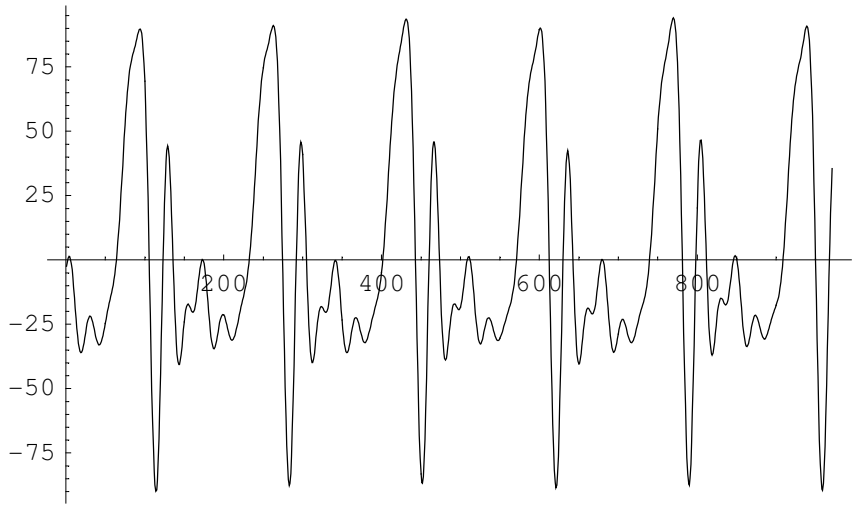
```
partList = Take[ sA[[1, 1]], {30000, 31000}];
ListPlot[ partList, PlotRange → All, PlotJoined → True];
```



Man kann die Daten noch mit einem Dreiecks Convolver glätten.

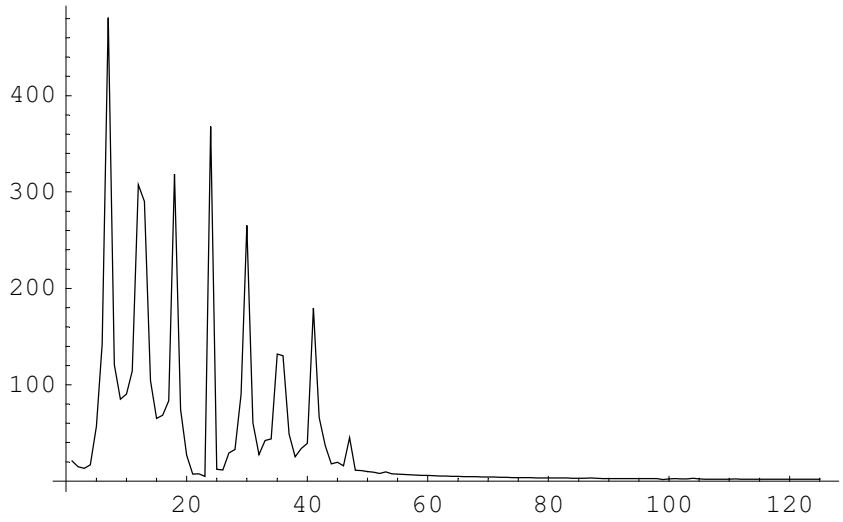
```
iMax = 15; convolver = Table[iMax - Abs[i], {i, -iMax, iMax}];
Print["Dreieck Convolver:  ", convolver];
ListPlot[convList = ListConvolve[convolver, partList],
  PlotRange -> All, PlotJoined -> True];
```

```
Dreieck Convolver:
 {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0}
```



Die Fourier Transformation des gemittelten Ausschnitts ergibt.

```
ListPlot[Take[Abs[Fourier[convList]], {1, Floor[Length[partList] / 8]}],
  PlotRange → All, PlotJoined → True];
```



Mathematica und Java

Wie unten gezeigt lassen sich Java Programme sehr einfach in *Mathematica* integrieren. Dadurch erschliessen sich weitere rund 3000 vordefinierte Funktionen (API's) zur Verwendung in *Mathematica*.

Orgel spielt eine Tonleiter über den Midi Kanal

Installieren der JLink Verbindung, Installation von Java und Laden der benötigten Java Klasse

```
Needs["JLink`"];
InstallJava[];
LoadClass["javax.sound.midi.MidiSystem"];
synth = MidiSystem`getSynthesizer[];
```

Öffnen des Midi Kanals. (Beachte die speziellen Aufrufe von Java Funktionen mit @)

```
synth @ open[] ;
channel = First[synth @ getChannels[]];
```

Setzen des Instruments (Nr. 19 entspricht einer Orgel) und Spielen der Tonleiter

```
channel@programChange[19]; vel = 80;
For[i = 60, i < 72, channel@noteOn[i, vel];
  Pause[0.3];
  channel@noteOff[i, vel];; i++];
```

Schliessen des Kanals

```
channel@allNotesOff[];
synth@close[];
```

■ Beep Function

```
javaBeep[] := JavaBlock[InstallJava[];
   LoadJavaClass["java.awt.Toolkit"];
   Toolkit`getDefaultToolkit[]@beep[];];
```

Nach einer kurzen Verzögerung (Laden der Java Class) ertönt ein kurzer Ton.

```
javaBeep[]
```

Mathematica und Turtle Graphics

Reiner *Mathematica* Code

In diesem Beispiel soll die Implementation einer virtuellen Zeichenmaschine - analog einer Turtle Graphics Machine - gezeigt werden. Diese Implementation wird zum Zeichnen der Koch Schneeflocken Kurve benutzt. *Mathematica* zeichnet die Graphik erst nach der vollständigen Berechnung.

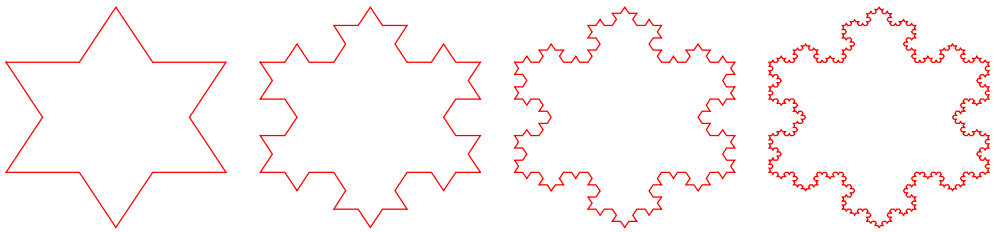
Turtle Graphics Code (pen down)

```
initialize[start_: {0, 0}] := (X = start; U = {1, 0}; path = {X});
right[a_] := U = {{Cos[aa = a Degree // N], Sin[aa]}, {-Sin[aa], Cos[aa]}}.U;
left[a_] := U = {{Cos[aa = a Degree // N], -Sin[aa]}, {Sin[aa], Cos[aa]}}.U;
forward[s_] := AppendTo[path, X += s U];
back[s_] := AppendTo[path, X -= s U];
turtlepath := Graphics[{{RGBColor[1, 0, 0], Line[path]}}, AspectRatio → 1];
```

Koch Snowflake Curve

```
flakeside[length_, 0] := forward[length];
flakeside[length_, depth_] :=
  (flakeside[length / 3, depth - 1]; left[60]; flakeside[length / 3, depth - 1]; right[120];
   flakeside[length / 3, depth - 1]; left[60]; flakeside[length / 3, depth - 1]);
KochSnowflake[n_] := (initialize[]; Do[flakeside[1., n]; right[120], {3}] );


nr = 4; Do[KochSnowflake[i]; gra[i] = turtlepath, {i, 1, nr}]
Show[GraphicsArray[Array[gra, nr]]];
```



Mathematica mit JLink und Java

Manchmal wünscht man einen graphischen Output bereits während der Berechnung. Dazu wird am einfachsten mit Hilfe von J/Link ein JavaFenster kreiert und mit Java Methoden in real time in dieses Fenster gezeichnet, wie im Folgenden am Beispiel "Random Walk in 2 Dimensionen" gezeigt wird.
Man sieht, wie einfach es ist, Java einzubinden. Damit erschliessen sich über 3000 vordefinierten Java Methoden und Klassen, auf die von *Mathematica* aus sehr einfach und ohne Kompilation zugegriffen werden kann.

Java wird installiert

```
Needs["JLink`"];
InstallJava[];
```

Ein Java Fenster zum Zeichnen wird definiert. Die Standard Java Klasse BorderLayout und die für *Mathematica* speziellen Klassen MathFrame und MathCanvas werden benutzt.

```
frame = JavaNew["com.wolfram.jlink.MathFrame"];
frame@setLayout[JavaNew["java.awt.BorderLayout"]];
mathCanvas = JavaNew["com.wolfram.jlink.MathCanvas"];
frame@add["Center", mathCanvas];
frame@setSize[400, 400];
frame@validate[];
```

Ein offscreen Buffer zum Zeichnen wird definiert, die Random Walk Schritte werden nach jeweils einer kurzen Pause mit Hilfe der Standard Java Methode "drawLine" eingezeichnet und auf die Anzeige gebracht.

```
JavaShow[frame]; (* rückt das Bild in den Vordergrund*)
offscreen = mathCanvas@
   createImage[w = mathCanvas@getSize[]@width, h = mathCanvas@getSize[]@height];
g = offscreen@getGraphics[];

(*Die Java Methode drawLine wird zum Zeichnen benutzt*)
x = Floor[w / 2]; stepX = Floor[w / 20];
y = Floor[h / 2]; setpY = Floor[h / 20];
steps = 50;
Do[g@drawLine[x, y,
    x += Random[Integer, {-stepX, stepX}], y += Random[Integer, {-stepY, stepY}]];
  mathCanvas@setImage[offscreen];
  Pause[5 / steps];, {i, 1, steps}];
```

Mathematica Grundlagen

Eine der Stärken von *Mathematica* liegt darin, dass jeder Ausdruck - sei es ein mathematischer Ausdruck, eine Graphik, die Eingabe Paletten oder ein ganzes Notebook Dokument - gleich aufgebaut ist: "Everything is an expression", nämlich head[arg1, arg2, ...].
Als weiteres Element kommt der starke Pattern Matcher hinzu. Jeder Ausdruck wird - auf Grund der internen und vom Nutzer eingegebenen Transformationsregeln - so oft umgewandelt, bis keine Änderung des Ausdrucks mehr stattfindet.

Mathematica löst auch die Schwierigkeit, mathematische Ausdrücke einerseits in der wohlbekannten mathematische Form (z.B. Integral) darzustellen und andererseits gleichzeitig die Genauigkeit und Eindeutigkeit des Ausdrucks zu erhalten (interne Darstellung, s.u. FullForm). Dadurch werden die Berechnungen plattformunabhängig und können auch über das Internet ausgetauscht werden. Mit dem gratis erhältlichen *MathReader* kann jeder die mathematischen Ausdrücke und Notebooks auch lesen.
Die Eingabe erfolgt am einfachsten in Input Form (Text) oder in Standard Form mit Hilfe der verfügbaren Paletten. Intern rechnet *Mathematica* in der eindeutigen FullForm Darstellung. In jeder Zelle des Notebook Dokuments ist sowohl der Inhalt, als auch die Form des mathematischen Ausdrucks gespeichert.

Verschiedene Darstellungen

```
InputForm (Text):          (a + c) / d
StandardForm (2dim):       (a+c)/d
FullForm (intern):         Times[Plus[a, c], Power[d, -1]]
```

Everything is an Expression

Die folgende Übersicht soll den einheitlichen Aufbau der *Mathematica* Ausdrücke demonstrieren.

Objekt	StandardForm	FullForm
Liste	{a, b, c}	List[a, b, c]
algebr. Ausdruck	$\sqrt{x} + x^2$	Plus[Power[x, Rational[1, 2]], Power[x,
Gleichung	x == Sin[x]	Equal[x, Sin[x]]
logischer Ausdruck	p && ! q	And[p, Not[q]]
Command	m[[1]] += a	AddTo[Part[m, 1], a]
Graphics	- Graphics -	Graphics[Circle[List[1, 0], 2]]
Button	Press here	Button["Press here"]
Notebook Cell	Cell[Text Cell, Text]	Cell["Text Cell", "Text"]
Function	Function[x, x^3]	Function[x, Power[x, 3]]

What else

In dieser kurzen *Mathematica* Übersicht wurden viele weitere Eigenschaften von *Mathematica* nicht gezeigt. Insbesondere möchte ich wenigstens die folgenden Eigenschaften noch erwähnen:

- *Mathematica* Funktionen und Ausdrücke können zur schnelleren Berechnung auch kompiliert werden.
- *Mathematica* hat auch einen Debugger.
- Die mit *Mathematica* entwickelten Notebooks sind portabel und plattformunabhängig. Die Notebooks und alle Ausdrücke werden in plain Text gespeichert.
- Üblicherweise wird *Mathematica* interaktiv bedient, wobei über das Front End (*Mathematica*) dem *Mathematica* Kernel via die *MathLink* Verbindung der Input zur Berechnung übergeben wird. Mit dem Front End kann der Input verändert und neu berechnet werden.
  Mit dem gratis erhältlichen *MathReader* können die *Mathematica* Notebooks von jedermann - auch ohne im Besitze der *Mathematica* Software zu sein - gelesen werden.
- Es können mit *Mathematica* einfache Benutzer Interfaces (z.B. mit Dialog[], Buttons und Paletten) geschrieben werden.
- Es können mit *Mathematica* Präsentationen (slide shows) ähnlich zu Power Point Präsentationen geschrieben werden.
- Da mit *Mathematica* sehr schnell Algorithmen entwickelt und der Output auf eine Datei geschrieben werden kann, lässt sich *Mathematica* auch zur Entwicklung von HTML Seiten, Java Benutzerinterfaces, animierten GIF's ... einsetzen.
- Mit AuthorTools können technische Bücher und Artikel geschrieben werden. Dies beinhaltet beispielsweise automatische Inhaltsverzeichnisse, Index ...
- Was *MathLink* für beliebige Programme und Network Programmierung sowie J/Link für Java Programme leistet, bietet *.NET/Link* für die .NET Platform von Microsoft.
- XML Fähigkeiten.Beliebige Dokumente im XML Format können importiert und und bearbeitet werden. *Mathematica* Ausdrücke sowie ganze Notebooks können im XML Format exportiert werden.
- Viele Methoden und Funktionen der höheren Mathematik. Viele mehr sind in zusätzlichen (Gratis) Packages verfügbar. Zum Beispiel: Algebra, Calculus, DiscreteMath, Geometry, Graphics, Linear Algebra, Miscellaneous, Number Theory, NumericalMath, Statistics, Utilities.
- Ein ausgezeichneter Help Browser, in dem auch eigene Dokumente integriert werden können.
- In vielen weiteren Gebieten sind kommerziell erhältliche Packages vorhanden.