COMPUTERPRAKTIKUM

EINFÜHRUNG - MATHEMATICA

1 Einleitung

1.1 Was ist Mathematica?

Mathematica gehört zu den weltweit am intensivsten entwickelten Computeralgebrasystemen (CAS), die heute verfügbar sind. Mit Mathematica haben Sie ein mächtiges und sehr komplexes Werkzeug zur Ausführung von mathematischen Aufgaben zur Hand.

- **Mathematica als Taschenrechner**
  Für numerische Berechnungen stehen mehrere tausend eingebaute Funktionen zur Verfügung, wobei der Benutzer die Genauigkeit wählen kann.
- **Mathematica für symbolische Berechnungen**
  Mathematica beherrscht u.a. die Differential- und Integralrechnung. Es kann also z.B. Stammfunktionen symbolisch angeben.
- **Mathematica als Visualisierungswerkzeug**
  Mathematica stellt in zwei- oder dreidimensionalen Grafiken mathematische Funktionen oder Datenlisten dar.

1.2 Bestandteile von Mathematica

Die beiden wichtigsten Bestandteile von Mathematica sind

- **Benutzeroberfläche** (engl.: **Frontend**)
  Das Frontend nimmt Anweisungen des Benutzers entgegen, übergibt die auszuwertenden Mathematica-Ausdrücke (Input) an den Kernel und bereitet dessen Output für den Benutzer auf.
- **Kern** (engl.: **Kernel**)
  Er ist plattformunabhängig und führt im Hintergrund alle mathematischen Operationen aus. Der Kernprozess wird normalerweise erst bei Anforderung der ersten Berechnung gestartet.

Die mit der Benutzeroberfläche von Mathematica erstellten Dokumente werden als **Notebooks** bezeichnet. In der Regel wird die Namenserweiterung `.nb` verwendet. Die Notebookinhalte sind in **Zellen** organisiert, deren Erstreckung am rechten Rand des Notebookfensters durch Klammern angezeigt wird.
Ein Notebook kann neben den Input- und Output-Zellen sowie Grafiken auch erklärenden

Text und Überschriften zur Gliederung enthalten (Menüpunkt **Format** > **Style**).
Am Anfang eines neuen Notebooks befindet sich eine waagerechte Linie als Zelleinfügemarke. Sobald Sie Text eintragen, wird für selbigen eine **Input**-Zelle eingerichtet. Die Einfügemarke verschwindet. Eine Zelle darf mehrere Zeilen umfassen, die jeweils mehrere Ausdrücke enthalten dürfen (Zeilenumbruch mit <**Return**>).

Um einen Ausdruck in einer Zelle als Mathematica-Input auswerten zu lassen, muss die Zelle entweder markiert sein oder die Schreibmarke (Cursor) muss sich darin befinden. Die Auswertung erfolgt mit der Tastenkombination <**Shift**> + <**Return**> oder mit der **Enter**-Taste des Ziffernblockes.

Alle **Input**-Zellen erhalten erst bei der Auswertung die Beschriftung `In[]:=` und werden **automatisch numeriert**. Die entsprechende **Output**-Zelle erhält die Beschriftung `Out[]=` und die gleiche Nummer.

1.3 Die Notation von Mathematica

Alle Mathematica-Anweisungen werden in einer einheitlichen Syntax formuliert. Zum Verfassen gibt es zwei Möglichkeiten:

- Die **klassische Mathematica-Syntax** wird mit der **Standardtastatur** eingegeben. Das ist universell und in allen Mathematica-Frontends gleich. Allerdings müssen dazu die Mathematica-Kommandos erlernt werden.
- Eine Eingabe in **mathematischer Notation** kann mit Hilfe von **Paletten** erfolgen. Diese können nötigenfalls über den Menüpunkt **Palettes** aktiviert werden kann. Mit ihrer Hilfe lässt sich eine Formel analog zur Arbeitsweise mit den Formeleditoren von Textverarbeitungsprogrammen aufbauen.

Argumente und Parameter von Funktionen bzw. Kommandos werden stets in **eckige Klammern** gesetzt.

1.4 Pakete

Viele Kommandos sind integraler Bestandteil von Mathematica. Darüberhinaus gibt es spezielle Kommandos, die durch **Pakete** (engl.: **packages**) bei Bedarf geladen werden können. Dafür gibt es das Kommando `Needs` oder den Operator `<<`. Nach dem Paketnamen muss ein nach rechts gerichteter Apostroph stehen. `Needs` möchte überdies eine Zeichenkette (String) als Argument übergeben bekommen.

```
In[1]:= Needs["NumericalCalculus`"]

In[2]:= ND[Log[x],x,3]
Out[2]= 0.333333
```

Beachten Sie, dass ein Paket geladen werden muss, bevor das erste Kommando daraus verwendet wird. Die im Moment aktiven Pakete können jederzeit mit folgendem Befehl eingesehen werden:

```
In[3]:= $Packages
Out[3]= {NumericalCalculus`,...,System`,Global`}
```

1.5 Befehle finden

Bei der Arbeit mit Mathematica sollten Sie unbedingt die sehr umfassende Online-Hilfe nutzen. Über den Menüpunkt **Help** > **Wolfram Documentation** oder durch Drücken der Taste **F1** können Sie das Hilfesystem auch direkt starten. Sie finden dort umfangreiche Materialien zu allen Aspekten von Mathematica. Hilfetexte zu einzelnen Funktionen können Sie sich direkt im Notebook-Fenster anzeigen lassen. Geben Sie dazu `?Funktion` ein. Sie erhalten dann eine kurze Information über Aufrufsyntax und Bedeutung der entsprechenden Funktion.

```
In[4]:= ?Log
        Log[z] gives the natural logarithm of z (logarithm to base e).
        Log[b,z] gives the logarithm to base b.  >>
```

Der Hilfetext im Notebook endet mit einem Querverweis-Symbol >>, dem Sie durch Anklicken folgen können.

Die Schriftfarbe der Ausdrücke gibt zudem darüber Aufschluss, ob eine aufgerufene Funktion integriert und deren Syntax korrekt ist.

Lg	**sin**	**Falsch**
Log	**Sin**	**Richtig**

2 Arbeiten mit Mathematica

2.1 Mathematica als Taschenrechner

Es stehen die üblichen arithmetischen Operationen zur Verfügung:

Operation	Eingabe
Addition	`x+y`
Minus	`-x`
Subtraktion	`x-y`
Multiplikation	`x y` oder `x*y`
Division	`x/y`
Potenzieren	`x^y`

```
In[1]:= 2(12-2)+(2+1)^2
Out[1]= 29
```

Die Auswertungsreihenfolge entspricht den üblichen Konventionen und kann - wenn notwendig - mit **runden Klammern** gesteuert werden. Als **Dezimaltrennzeichen ist der Punkt** zu verwenden. Ein Beispiel für die Exponentialschreibweise:

```
In[2]:= 5.43 10^-45
Out[2]= 5.43 × 10^-45
```

Während Taschenrechner mit begrenzter Genauigkeit arbeiten, liefert Mathematica nach Möglichkeit exakte Ergebnisse:

```
In[3]:= 3^68
Out[3]= 278 128 389 443 693 511 257 285 776 231 761
```

Wollen Sie ein Näherungsergebnis erhalten, müssen Sie die Funktion `N[Ausdruck]` (für: **Numerical value**) auf den auszuwertenden Ausdruck anwenden. `N[x]` gibt den numerischen Wert des Argumentes x aus:

```
In[4]:= N[3^68]
Out[4]= 2.78128 × 10^32
```

Optional kann dabei die Genauigkeit festgelegt werden:

```
In[5]:= N[Pi,30]
Out[5]= 3.14159265358979323846264338328
```

In Mathematica-Ausdrücken können Sie einige **Konstanten** über Namen oder Sonderzeichen ansprechen. Die Kreiszahl π hat den Mathematica-Namen `Pi`. Die Eulersche Zahl e wird über `E` angesprochen. Bei Integrationen beispielsweise kann die Integrationsgrenze unendlich (∞) mit `Infinity` aufgerufen werden. Weitere mathematische Konstanten sind im Hilfesystem über den Menüpunkt **Help** > **Wolfram Documentation** zu finden.

Wenn eine ganze Zahl ohne Dezimaltrennzeichen eingegeben wird, hält Mathematica diese für exakt. Enthält ein Ausdruck ausschließlich exakte Werte, dann beschränkt sich Mathematica bei der Auswertung auf exakte Operationen:

```
In[6]:= 1/2 + 3/7
Out[6]= 13/14
```

Wenn Sie eine Zahl mit Dezimalpunkt eingeben, hält Mathematica diese für gerundet. Enthält in einem Ausdruck irgendeine Zahl einen Dezimalpunkt, wird das Ergebnis des Ausdruckes numerisch ermittelt:

```
In[7]:= 1/2 + 3.0/7
Out[7]= 0.928571
```

2.2 Funktionen

2.2.1 Mathematica-Funktionen

In Mathematica-Ausdrücken können fest eingebaute und per Paket ergänzte mathematische Funktionen verwendet werden. Bei der Schreibweise von Funktionsaufrufen ist auf die **Groß/Klein-Schreibung** zu achten. Die Namen der in Mathematica integrierten Funktionen beginnen dabei stets mit einem Großbuchstaben. Die Schriftfarbe der Ausdrücke gibt Aufschluss darüber, ob eine aufgerufene Funktion integriert und deren Syntax korrekt ist (siehe Abschnitt 1.5). Argumente und Parameter von Funktionen werden in **eckige Klammern** gesetzt. Bei der Anwendung von **trigonometrischen Funktionen** ist das **Argument im Bogenmaß** einzusetzen:

```
In[8]:= Sin[45.]
Out[8]= 0.850904

In[9]:= Sin[Pi/4.]
Out[9]= 0.707107
```

Wie bei den arithmetischen Operationen beschränkt sich Mathematica auch bei der Auswertung von Funktionen mit exakten Argumenten auf exakte Operationen:

```
In[10]:= Log[1/5]
Out[10]= -Log[5]
```

Eine Näherungslösung kann erzwungen werden, indem die Funktion `N[Ausdruck]` (für: **Numerical value**) auf den Ausdruck anwendet wird:

```
In[11]:= N[%]
Out[11]= -1.60944
```

Beim Arbeiten mit Mathematica ist es oft bequem, in der aktuellen Input-Zelle eine frühere Ausgabe anzusprechen.

Eingabe	Bedeutung
`%`	letzte Ausgabe
`%%`	vorletzte Ausgabe
`%n`	Inhalt der Ausgabe `Out[n]`

```
In[12]:= %1/14 + N[%6]
Out[12]= 3.
```

> Für die Erstellung von Notebooks ist jedoch die Verwendung von Variablen unbedingt empfehlenswert (siehe Abschnitt 2.2.2).

Komplexe Zahlen schreibt Mathematica mit Hilfe der imaginären Einheit, die im Inputformat durch den Namen `I` dargestellt wird:

```
In[13]:= Sqrt[-4]
Out[13]= 2 i

In[14]:= Sin[I]
Out[14]= i Sinh[1]
```

Mathematica kennt die üblichen Operationen mit komplexen Zahlen. Die Funktionen `Re[z]` und `Im[z]` geben den Real- bzw. Imaginärteil des Argumentes z aus:

```
In[15]:= Re[%%]
Out[15]= 0

In[16]:= Im[%14]
Out[16]= Sinh[1]
```

2.2.2 Benutzerdefinierte Funktionen / Variablen

Man kann natürlich Funktionen auch selbst definieren. Eine solche Variable besitzt einen Namen, über den man diese ansprechen kann. Um Verwechslungen mit Befehlen von Mathematica zu vermeiden, sollte die benutzerdefinierte Funktionsbezeichnung mit einem Kleinbuchstaben beginnen. Bei der Syntax ist zu beachten, dass auf der **linken** Seite der Definitionsgleichung an das Argumentensymbol ein **Unterstrich** angehängt wird. Es ist das in der Mathematik übliche **Definitionszeichen** `:=` zu verwenden (siehe Abschnitt 6). Die Belegung einer Variablen löscht man mit dem Befehl `Clear`.

```
In[17]:= f[x_,y_]:=x^2 + 5y
```

Mathematica erzeugt keine Ausgabezelle zur Bestätigung der Definition. Selbst definierte Funktionen verarbeiten sowohl numerische als auch symbolische Argumente, wie das folgende Beispiel zeigt:

```
In[18]:= f[3,7]
Out[18]= 44

In[19]:= f[a,b]^3
Out[19]= (a^2 + 5b)^3
```

2.3 Algebraische Umformungen von Ausdrücken

Bisher hat Mathematica für uns numerische Lösungen (exakt oder angenähert) ermittelt. Nun lernen wir seine Fähigkeiten kennen, algebraische bzw. arithmetische Ausdrücke mit **Symbolen** umzuformen:

```
In[20]:= Expand[%]
Out[20]= a^6 + 15a^4b + 75a^2b^2 + 125b^3
```

`Simplify[`*`Ausdruck`*`]` und `FullSimplify[`*`Ausdruck`*`]` vereinfachen den Term:

```
In[21]:= Simplify[x^2 + 23x - 9x - x^3 + 18 + 2x/3 + 3x^2]
Out[21]= 18 + 44x/3 + 4x^2 - x^3

In[22]:= Simplify[x^2 + 2x^3/(3x) + 2x/(3x^2 + 5)]
Out[22]= 5x^2/3 + 2x/(5+3x^2)
```

2.4 Differenzieren und Integrieren

Mathematica kann alle elementaren mathematischen Funktionen ableiten. Mit dem Befehl `D[`f`,`x`]` wird die Funktion f partiell nach der Variablen x abgeleitet. Hierbei wird für alle anderen Symbole im Funktionsausdruck angenommen, dass diese nicht von x abhängig sind.

```
In[23]:= D[Cos[3*x^2],x]
Out[23]= -6 x Sin[3 x^2]
```

Mathematica kann auch Ausdrücke mit Symbolen für unbekannte Funktionen ableiten:

```
In[24]:= D[f[x]^2,x]
Out[24]= 2 f[x] f'[x]
```

Mit dem Kommando `Integrate[f,x]` kann mit Mathematica die Stammfunktion F zur Funktion f bestimmt werden:

```
In[25]:= Integrate[Cos[3*x],x]
Out[25]= 1/3 Sin[3 x]
```

Auch bei bestimmten Integralen mit symbolischen Grenzen versucht Mathematica, einen geschlossenen Ausdruck als Lösung zu finden:

```
In[26]:= Integrate[Exp[x]x,{x,0,a}]
Out[26]= 1 + (-1 + a) e^a
```

3 Vektoren und Matrizen

3.1 Vektoren und Matrizen als (formatierte) Listen

Vektoren und Matrizen werden in Mathematica durch Listen repräsentiert. Mit Hilfe der **geschweiften Klammern** werden die Elemente - jeweils durch ein Komma getrennt - zu einer Liste zusammengefasst. Eine Matrix wird als „Liste von Listen“ dargestellt, wobei eine Liste einer Zeile der Matrix entspricht:

```
In[1]:= vektor1:={x,y,z}
        vektor2:={u,v,w}
        vektor3:={s,t}
        matrix1:={{a,b},{c,d}}
```

Matrizen können auch über den Menüpunkt **Insert** > **Table/Matrix** > **New** eingegeben werden. Die Zeilen- und Spaltenanzahl kann im Dialogfenster festgelegt werden.
Per `MatrixForm` erhält man die vertrauten Darstellungen:

```
In[5]:= MatrixForm[vektor1]
Out[5]//MatrixForm=
        ( x )
        ( y )
        ( z )
In[6]:= MatrixForm[matrix1]
Out[6]//MatrixForm=
        ( a  b )
        ( c  d )
```

Um einzelne Elemente einer Liste anzusprechen, sind die Indizes in **doppelte eckige Klammern** einzuschließen. Bei einer zweidimensionalen Liste (bzw. Matrix) müssen entsprechend zwei Indizes angegeben werden, um auf ein einzelnes Element der Liste zuzugreifen:

```
In[7]:= matrix1[[2]]
Out[7]= {c,d}

In[8]:= matrix1[[2,2]]
Out[8]= d
```

3.2 Rechnen mit Matrizen

Mathematica berechnet das Produkt zweier Matrizen nach den üblichen Regeln. Als Operatorzeichen für das Matrixprodukt ist ein Punkt „.“ oder `Dot[a,b]` zu verwenden. Setzt man den Multiplikations-Stern „*“ oder gar kein Operatorzeichen, führt Mathematica eine skalare (elementweise) Multiplikation aus. Das Kreuzprodukt erhält man durch das Kommando `Cross[a,b]`:

```
In[9]:=  vektor1.vektor2
Out[9]=  {ux + vy + wz}

In[10]:= MatrixForm[Cross[vektor1,vektor2]]
Out[10]//MatrixForm=
          ( wy - vz  )
          ( -wx + uz )
          ( vx - uy  )
```

Beim Produkt eines Vektors mit einer Matrix wird der Vektor situationsabhängig als Zeile oder Spalte behandelt:

```
In[11]:= MatrixForm[matrix1.vektor3]
Out[11]//MatrixForm=
          ( as + bt )
          ( cs + dt )

In[12]:= vektor3.matrix1
Out[12]//MatrixForm=
          ( as + ct )
          ( bs + dt )
```

Die Ausdrücke `Transpose[]`, `Det[]` und `Inverse[]` sollten selbsterklärend sein.

Weiterhin können von Mathematica mit Hilfe der Kommandos `Eigenvalues[]` und `Eigenvectors[]` Eigenwertprobleme von Matrizen gelöst werden:

```
In[13]:= Eigenvalues[matrix1]
```

$$\text{Out[13]}= \left\{ \frac{1}{2}\left(\texttt{a + d - }\sqrt{\texttt{a}^2\texttt{ + 4bc - 2ad + d}^2}\right),\ \frac{1}{2}\left(\texttt{a + d + }\sqrt{\texttt{a}^2\texttt{ + 4bc - 2ad + d}^2}\right)\right\}$$

```
In[14]:= Eigenvectors[matrix1]
```

$$\text{Out[14]}= \left\{ \left\{ -\frac{\texttt{-a + d + }\sqrt{\texttt{a}^2\texttt{ + 4bc - 2ad + d}^2}}{\texttt{2c}},\ 1 \right\},\ \left\{ -\frac{\texttt{-a + d - }\sqrt{\texttt{a}^2\texttt{ + 4bc - 2ad + d}^2}}{\texttt{2c}},\ 1 \right\} \right\}$$

`Eigensystem[]` gibt sowohl die Eigenwerte, als auch die zugehörigen Eigenvektoren aus.

4 Lösen von Gleichungen

4.1 Polynomgleichungen mit einer Variablen

Mit der Mathematica-Funktion `Solve` kann man Polynomgleichungen lösen. Man beachte, dass in Mathematica das einfache Gleichheitszeichen verwendet wird, um einer Variablen einen Ausdruck zuzuordnen. Das doppelte Gleichheitszeichen steht für den Identitätsoperator, mit dem getestet wird, ob die Ausdrücke auf beiden Seiten des Operatorzeichens übereinstimmen.

```
In[1]:= gl1 = a*x^2 + b*x + c == 0
Out[1]= c + bx + ax^2 == 0
```

`Solve` liefert nun als Ergebnis eine Liste von **Ersetzungsregeln** für die Variablenwerte, welche die Gleichung erfüllen. Wendet man diese Ersetzungsregeln auf den einfachen Ausdruck x an (siehe Abschnitt 6.2), resultiert eine Liste mit den Lösungswerten:

```
In[2]:= sol1 = Solve[gl1,x]
```

$$\text{Out[2]}= \left\{\left\{x \to \frac{-b - \sqrt{b^2 - 4ac}}{2a}\right\}, \left\{x \to \frac{-b + \sqrt{b^2 - 4ac}}{2a}\right\}\right\}$$

```
In[3]:= x/.sol1
```

$$\text{Out[3]}= \left\{\frac{-b - \sqrt{b^2 - 4ac}}{2a}, \frac{-b + \sqrt{b^2 - 4ac}}{2a}\right\}$$

Um numerische Lösungen zu bekommen, nutzt man den Befehl `NSolve`.

4.2 Transzendente Gleichungen in einer Variablen

Mathematica kann auch manche transzendente Gleichung lösen, wobei in der Regel eine Warnung darauf hinweist, dass vermutlich nicht alle Lösungen gefunden wurden.
In der Regel können transzendente Gleichungen jedoch nicht in symbolischer Form gelöst werden. Mit der Funktion `FindRoot` und einem geeigneten Startwert lässt sich aber eine numerische Näherungslösung ermitteln:

```
In[4]:= FindRoot[Cos[x]==x, {x,1}]
Out[4]= {x → 0.739085}
```

4.3 Gleichungssysteme mit mehreren Variablen

Mit der Mathematica-Funktion `Solve` lassen sich alle linearen Gleichungssysteme und zahlreiche Systeme mit Polynomgleichungen explizit lösen.

4.3.1 Zwei Gleichungen mit einer Variablen

Die zu lösenden Gleichungen sind in Form einer Liste dem Kommando zu übergeben:

```
In[5]:= Solve[{x^2==1, x^3==1},x]
Out[5]= {{x → 1}}
```

4.3.2 Zwei Gleichungen in zwei Variablen

Hier wird man in der Regel nach x und y auflösen wollen, weil eine Auflösung nach x allein der Suche nach denjenigen x-Werten entspräche, welche die Gleichungen für beliebige Werte des symbolischen Parameters y erfüllen:

```
In[6]:= Solve[{x + y == 5, 5*x - a*y == 0},{x,y}]
```

$$\text{Out[6]= } \left\{\left\{\text{x} \to \frac{5\text{a}}{5 + \text{a}}\text{ , y} \to \frac{25}{5 + \text{a}}\right\}\right\}$$

Mit Hilfe der Mathematica-Funktion `LinearSolve` lässt sich bequem der Lösungsvektor x zu einer Matrix-Gleichung der Form

$$m.x == b$$

ermitteln. Das eben gezeigte Beispiel wird als inhomogenes Gleichungssystem gelöst. Zunächst wird die Koeffizientenmatrix `km` definiert.

```
In[7]:= km = {{1,1},{5,-a}}
        b = {5,0}
        LinearSolve[km,b]
```

$$\text{Out[7]= } \{\{1,1\},\{5,-\text{a}\}\}$$

$$\text{Out[8]= } \{5,0\}$$

$$\text{Out[9]= } \left\{\frac{5\text{a}}{5 + \text{a}}\text{ , }\frac{25}{5 + \text{a}}\right\}$$

4.3.3 Variablen eliminieren

Wenn beispielsweise zwei Gleichungen in den drei Veränderlichen x,y und a nach x und y aufgelöst werden sollen, erhalten wir Ergebnisterme mit dem symbolischen Parameter a:

```
In[10]:= Solve[{x==y-3a, y==2x-a},{x,y}]
Out[10]= {{x → 4 a , y → 7 a}}
```

In dieser Situation kann man aber auch eine Veränderliche, z.B. a, eliminieren. Dann bleibt noch eine Gleichung mit den beiden Veränderlichen x und y übrig. Mit folgender Syntax erhält man sofort die Auflösung nach x bei Elimination von a:

```
In[11]:= Solve[{x==y-3a, y==2x-a},{x},{a}]
```

$$\text{Out[11]= } \left\{\left\{\text{x} \to \frac{4\ \text{y}}{7}\right\}\right\}$$

4.4 Differentialgleichungen

Die Funktion `DSolve` löst lineare und nichtlineare gewöhnliche Differentialgleichungen sowie Differentialgleichungssysteme. Als Ergebnisse liefert diese Funktion - genau wie `Solve` - Ersetzungsregeln (siehe Abschnitt 6.2).

```
In[12]:= DSolve[y''[x] == 3*y[x],y[x],x]
```

$$\text{Out[12]= } \left\{\left\{\text{y[x]} \to e^{\sqrt{3}\ \text{x}}\ \text{C[1]} + e^{-\sqrt{3}\ \text{x}}\ \text{C[2]}\right\}\right\}$$

Die unbestimmten Konstanten `C[1]` und `C[2]` in der Lösungsfunktion können über die Anfangsbedingungen festgelegt werden:

```
In[13]:= DSolve[{x''[t] + ω^2*x[t]==0, x'[0]==0, x[0]==1},x[t],t]
Out[13]= {{x[t] → Cos[tω]}}
```

Leider sind nicht alle Differentialgleichungen geschlossen lösbar. In diesen Fällen liefert `NDSolve` zumindest eine numerische Lösung. Damit lässt sich die DG dann auch grafisch darstellen!

```
In[14]:= NDSolve[{x''[t] + Sin[x[t]]==0, x[0]==1, x'[0]==0},x,{t,0,10}]
Out[14]= {{x → InterpolatingFunction[{{0., 10.}},<>]}}
```

Dieses numerische Resultat kann nun noch ausgewertet werden:

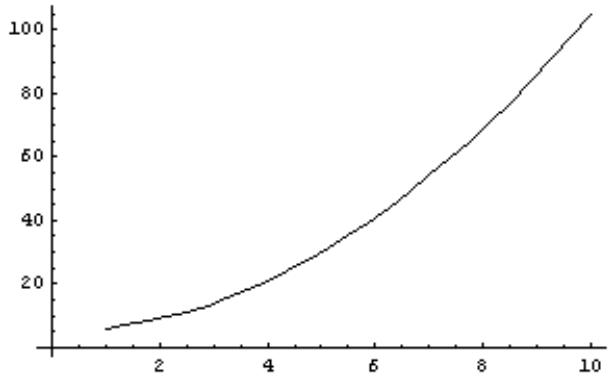
```
In[15]:= {x[1.5],x'[3.0]}/.%14[[1]]
Out[15]= {0.166939, -0.29119}
```

5 Grafiken

Um sich den Funktionsverlauf einer reellwertigen Funktion darstellen zu lassen, können Sie das Kommando `Plot` benutzen:

```
In[1]:= Plot[x^2 + 5, {x,1,10}]
```

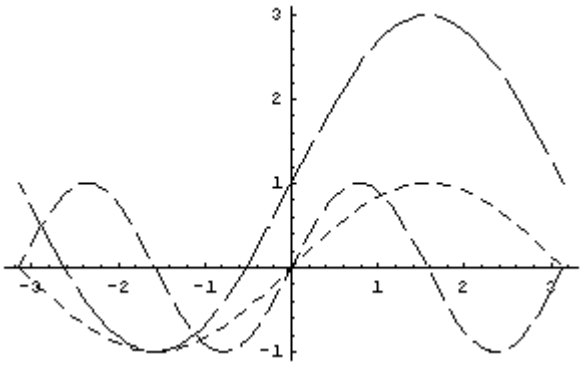
Out[1]=



Es sind auch simultane Darstellungen von mehreren Funktionen möglich. Die Unterscheidung erfolgt in den Beispielen zum einen durch verschiedene Linienarten und zum anderen durch verschiedene Farbgebung der Funktionsverläufe:

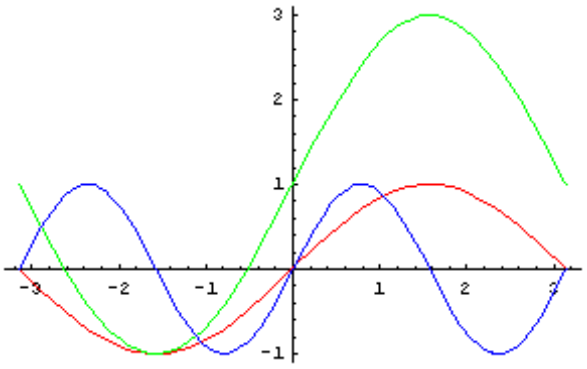
```
In[2]:= Plot[{Sin[x], 2*Sin[x] + 1, Sin[2x]}, {x,-Pi,Pi},
        PlotStyle→{Dashing[{.02,.02}],Dashing[{.15,.02}],
        Dashing[{.05,.02}]}]
```

Out[2]=



```
In[3]:= Plot[{Sin[x], 2*Sin[x] + 1, Sin[2x]}, {x,-Pi,Pi},
        PlotStyle→{RGBColor[1,0,0],RGBColor[0,1,0],
        RGBColor[0,0,1]}]
```

Out[3]=



Viele Eigenschaften einer Grafik können über **Optionen** beeinflusst werden. Nach der Mathematica-Syntax werden Optionen generell am Ende eines Funktionsaufrufes als Folge von **Regeln** mit der Syntax

$$Option \rightarrow Wert$$

angegeben. Für die `Plot`-Funktion lautet also die erweiterte Syntax:

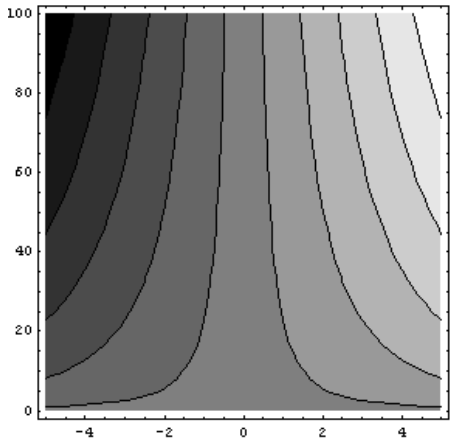
```
Plot[{f1,f2, ...},{x,xmin,xmax},Option1->Wert1, Option2->Wert2, ...]
```

Um eine bestimmte Grafik zu formatieren, experimentiert man am Besten, indem man eine Reihe von verschiedenen Einstellungen für unterschiedliche Optionen ausprobiert. Die Optionen und deren genaue Syntax findet man im Hilfesystem, welches über den Menüpunkt **Help** > **Wolfram Documentation** oder durch Drücken der Taste **F1** gestartet werden kann.

Konturdiagramme eignen sich beispielsweise um Skalarfelder darzustellen. Dabei werden Argumentbereiche mit Funktionswerten innerhalb bestimmter Intervalle durch Grenzlinien und Grau-/Farbstufen (je größer der Funktionswert, desto heller) markiert:

```
In[4]:= ContourPlot[x*Sqrt[y], {x,-5,5},{y,0,100}]
```

Out[4]=

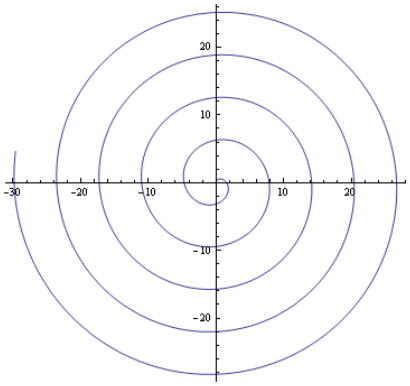


Auch hier stehen eine Reihe von Optionen zur Gestaltung der Grafik zur Verfügung, die man in der Hilfe leicht findet.

Mit Hilfe des Kommandos `ParametricPlot` erhält man parametrische Diagramme. Bei zweidimensionalen parametrischen Diagrammen wird eine Kurve in der Ebene über zwei von derselben Variablen abhängige Funktionen für x- und y-Koordinate beschrieben:

```
In[6]:= ParametricPlot[{t*Sin[t], t*Cos[t]}, {t,0,30}]
```

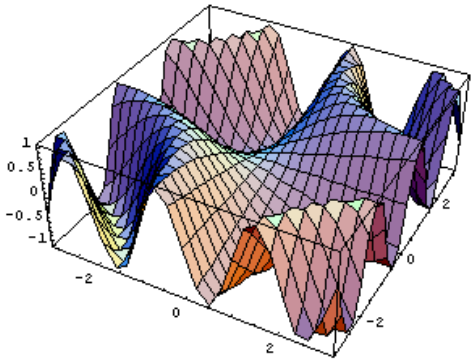
Out[6]=



Mit Mathematica haben Sie die Möglichkeit neben 2D-Darstellungen auch räumliche Darstellungen zu erzeugen. Die meisten 2D-Plot-Befehle gibt es auch als 3D-Befehle. Für eine räumliche Darstellung einer Funktionsoberfläche verwendet man das Kommando `Plot3D`:

```
In[7]:= Plot3D[Sin[x*y], {x,-Pi,Pi}, {y,-Pi,Pi}]
```

Out[7]=



6 Wertzuweisungen

6.1 Zuweisung mit und ohne Auswertung

Mathematica kennt zwei Formen der Zuweisung von Ausdrücken an Bezeichner. Unter dem Begriff Bezeichner versteht man hierbei korrekt geformte Zeichenketten, die als Namen von Variablen, Symbolen, Funktionen, Konstanten, etc. verwendet werden.

Die Kommandozeile `y=A` wertet den Ausdruck A auf der rechten Seite vor der Zuweisung aus. Die Syntax `x:=A` dagegen weist x den Ausdruck A zu, ohne dass dieser vorher ausgewertet wird.

Wenn Bezeichnern konstante (nicht symbolische) Werte zugewiesen werden, besteht kein Unterschied zwischen beiden Zuweisungsarten.

```
In[1]:= x:=7; y=7; {x,y}
Out[3]= {7,7}
```

Das folgende Beispiel sieht auf den ersten Blick auch so aus, als wären die Zuweisungen gleichwertig. Das ist jedoch ein Irrtum. Der Variable `y` wird der Wert von `f`, also 4, zugewiesen. Der Variable `x` wird dagegen das Symbol `f` zugewiesen. Das merken Sie spätestens, wenn Sie den Wert von `f` ändern. Während das in diesem Beispiel auf `y` keinen Einfluss hat, liefert die Auswertung von `x` nun 9 zurück.

```
In[4]:= f=4;
In[5]:= x:=f; y=f; {x,y}
Out[7]= {4,4}
In[8]:= f=9; {x,y}
Out[9]= {9,4}
```

Deshalb folgender **Ratschlag**:

> Für Wertvariablen sollten Sie in der Regel die Zuweisung `=` benutzen. Dagegen sollten Sie für Funktionsdefinitionen (siehe Abschnitt 2.2.2) normalerweise `:=` verwenden.

6.2 Lokale Wertzuweisung

Um Variablen lokal (also nur vorübergehend) Werte zuzuweisen, kann in Mathematica mit Substitutionen gearbeitet werden. Mit der Kommandozeile `5z - 3/.z→2` wird der Ausdruck mit `z=2` ausgewertet. `z` wird aber nicht bleibend verändert.

```
In[10]:= 5z-3/.z→2
Out[10]= 7
In[11]:= z
Out[11]= z
```

Die Verwendung von **Ersetzungsregeln** läuft auch auf ein lokale Wertzuweisung hinaus. Die allgemeine Syntax lautet:

Ausdruck/.Ersetzungsregel