Funktionsplotter für xfig

1. August 2006 Version 2.5

Zusammenfassung

Dieses Programm ist freie Software. Sie können es unter den Bedingungen der GNU General Public License, wie von der Free Software Foundation veröffentlicht, weitergeben und/oder modifizieren, entweder gemäß Version 2 der Lizenz oder (nach Ihrer Option) jeder späteren Version.

Die Veröffentlichung dieses Programms erfolgt in der Hoffnung, daß es Ihnen von Nutzen sein wird, aber OHNE IRGENDEINE GARANTIE, sogar ohne die implizite Garantie der MARKTREIFE oder der VERWENDBARKEIT FÜR EINEN BESTIMMTEN ZWECK. Details finden Sie in der GNU General Public License.

Sie sollten ein Exemplar der GNU General Public License zusammen mit diesem Programm erhalten haben. Falls nicht, schreiben Sie an die Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110, USA.

Bei Fehlern schreiben Sie diese einfach an webmaster@rendtel.de

Inhaltsverzeichnis

1 Funktionsplotter für xfig

Um mir das Leben als Mathematiklehrerin einfacher zu machen habe ich mir das Perl-Skript `plotter.pl` geschrieben, welches Graphen und Vektoren in xfig automatisch zeichnet. Dies geschah in den Winterferien 2005/2006. Seit der letzten Veröffentlichung gibt es gleich einen Sprung auf die Version 2.5. Es gibt einige Neuerungen und Erweiterungen der alten Funktionen.
Dazu müssen in einer Datei mit der Extension ptxt verschiedene Objekte angegeben, z.B. Vektoren, und diese ptxt-Datei wird dann durch das Perl-Skript auf der Kommandozeile mit dem Aufruf

```
./plotter.pl datei.ptxt
```

in eine datei.fig verwandelt.
Im folgenden erläutere ich zunächst die möglichen Befehle und darauf folgt dann der Quellcode.

1.1 Mögliche Befehle

Um eine xfig-Objekte zu erzeugen gibt es einige Befehle die in eine `ptxt-Datei` geschrieben werden muss. In der `befehle.ptxt`-Datei stehen diese alle zusammengefasst. Hier werden einmal alle Befehle mit Beispielen angesprochen. Es gibt mehrere optionale Parameter, die in den eckigen Klammern gesetzt sind.

1.1.1 Skalierung

Als erstes sollte in der `ptxt-Datei` enthalten sein, wie groß man die Datei skaliert haben möchte. Dazu gibt es zwei Möglichkeiten

```
skalierung <faktor>
skalierung x=<faktor> y=<faktor>
```

Wenn man nur `skalierung 3` setzt, so wird der Skalierungsfaktor für die x- und y-Achse auf den gleichen Wert gesetzt.

1.1.2 Achsenschnittpunkte

Da sich die Achsen des Koordinatensystems nicht nur immer im Punkt $(0|0)$ schneiden sollen, kann mit der Funktion

```
achsenschnitt x=<xwert> y=<ywert>
```

der Schnittpunkt verschoben werden. Als Standard wird der Punkt $(0|0)$ genutzt. Dieser Aufruf muss direkt nach der Skalierung erfolgen.

1.1.3 Gitter und andere Raster

Mit dem Befehl

```
kariert [layer=<n>] abstandx=<schritt1>,<schritt2>,<schritt3>
abstandy=<schritt1>,<schritt2>,<schritt3>
bereich=<xmin>,<ymin>,<xmax>,<ymax>
```

kann ein Gitter im Koordinatensystem gezeichnet werden. Dabei gibt `abstandx` und `abstandy` an, in welchem Abstand die Striche in der $x-$ und $y-$Ebene gezeichnet werden sollen und nach wie vielen Strichen eine dicke und besonders dicke Linie gezeichnet werden soll. Es gibt auch die Befehle `millimeterpapier` und `matheheft`, die den `kariert` Befehl aufrufen.

```
millimeterpapier [layer=<n>] bereich=<xmin>,<ymin>,<xmax>,<ymax>
matheheft [layer=<n>] bereich=<xmin>,<ymin>,<xmax>,<ymax>
```

Analog zur `gitter`-Funktion gibt es die `grid`-Funktion, die kleine Punkte in ein Koordinatensystem einzeichnet.

```
grid [layer=<n>] [farbe=<n>]
abstandx=<schritt1>,<schritt2>,<schritt3>
abstandy=<schritt1>,<schritt2>,<schritt3>
bereich=<xmin>,<ymin>,<xmax>,<ymax>
```

Hierbei gibt `schritt1` jeweils an, wie viele Punkte innerhalb des Abstandes `schritt2` angelegt werden sollen. `schritt3` gibt an, in welchem Abstand die Linien liegen sollen.

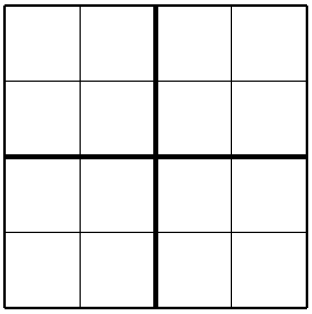
Zusätzlich können anstelle von kariertem Papier und gepunkteten Papier auch nur kleine Kreuze in ein Koordinatensystem mit dem Befehl `punkte` gezeichnet werden. Hier müssen wieder nur die Minima und Maxima für die x- und y-Achsen angegeben werden und in welchem Abstand sich diese Punkte befinden sollen.

```
punkte [layer=<n>] abstand=<schritt1> bereich=<xmin>,<ymin>,<xmax>,<ymax>
```

Beispiele

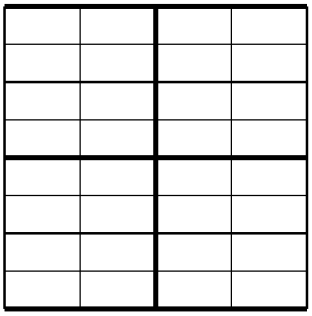
1. (a) Kariertes Papier im Abstand von 1 in $x-$ und $y-$Richtung:

```
skalierung 1
kariert layer=50 abstandx=1,2,4 abstandy=1,2,4 bereich=-2,-2,2,2
```



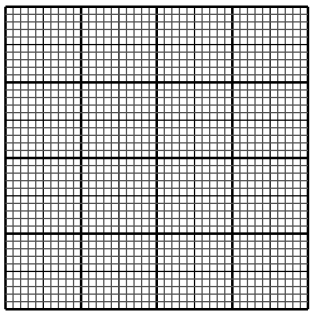
(b) Kariertes Papier im Abstand von 1 in $x-$ und 0,5 in $y-$Richtung:

```
skalierung 1
kariert layer=50 abstandx=1,2,4 abstandy=0.5,2,4 bereich=-2,-2,2,2
```



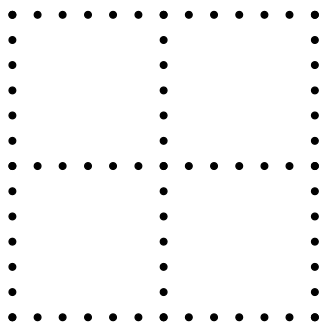
2. Millimeterpapier

```
skalierung 1
millimeterpapier layer=50 bereich=-2,-2,2,2
```



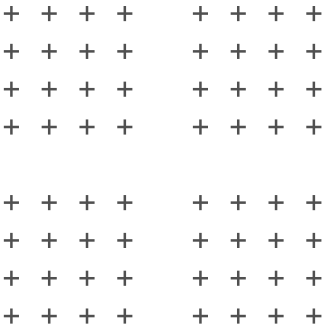
3. (a) Gepunktetes Papier. Dabei liegen innerhalb eines Zentimeters drei Punkte. Die Linien haben immer einen Abstand von 2 cm.

```
skalierung 1
grid layer=50 abstandx=3,1,2 abstandy=3,1,2 bereich=1,1,5,5
```



4. Punkte im Koordinatensystem

```
skalierung 1
punkte layer=50 abstand=0.5 bereich=-2,-2,2,2
```



1.1.4 Koordinatensystem

Es kann ein Koordinatensystem gezeichnet werden, indem man einzeln die Befehle `xachse` und `yachse` aufruft. Zusammen wird dann ein Koordinatensystem gezeichnet mit dem Schnittpunkt im Punkte (xwert; ywert) aus der Achsenschnittpunktsfunktion.

```
xachse layer=<n> bereich=<xmin>,<xmax>,<schrittweite>
skala=<keine|nurstriche|nureins|normal|strichundeins> text="
    achsenbeschriftung">
yachse layer=<n> bereich=<ymin>,<ymax>,<schrittweite>
skala=<keine|nurstriche|nureins|normal|strichundeins> text="
    achsenbeschriftung">
```

Die `schrittweite` gibt an, in welchem Abstand die Achsenbeschriftung und kleine Linien gezeichnet werden sollen. Diese kann man über `skala` einstellen und `text` gibt an, welche Beschriftung an der Achse stehen soll. Bei der Achsenbeschriftung ist LaTeX-Code zulässig.
Zusätzlich gibt es noch den Befehl

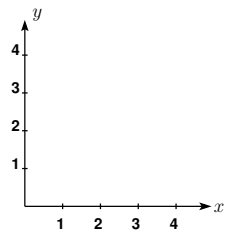
```
xachserad layer=<n> bereich=<xmin>,<xmax>,<schrittweite>
```

mit dem man an die Achsen Beschriftung im Bogenmaß für trigonometrische Funktionen zeichnen kann.

Beispiel

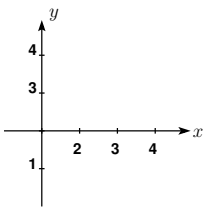
1. Achsen:

```
skalierung 1
xachse layer=10 bereich=0,5,1 skala=normal text="$x$"
yachse layer=10 bereich=0,5,1 skala=normal text="$y$"
```



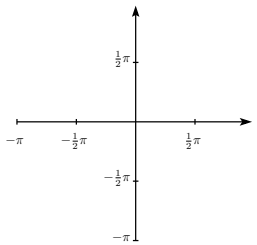
oder

```
skalierung 1
achsenschnitt x=1 y=2
xachse layer=10 bereich=0,5,1 skala=normal text="$x$"
yachse layer=10 bereich=0,5,1 skala=normal text="$y$"
```



2. Achsen in Bogenmaß:

```
skalierung 1
xachserad layer=10 bereich=-180,180,90
yachserad layer=10 bereich=-180,180,90
```



1.1.5 Zeichnen von Funktionen

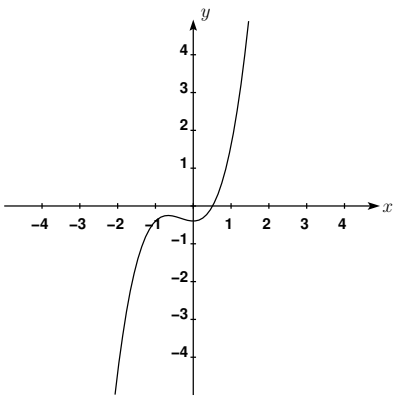
Mit dem Befehl `plot` können Funktionen in ein Koordinatensystem gezeichnet werden. Dabei dürfen allerdings zwischen den Operatoren und Parametern keine Leerzeichen geschrieben werden. Der Befehl lautet:

```
plot [layer=<n>] [farbe=<n>] bereich=<xmin>,<xmax> [grenze=<ymin>,<ymax>]
    function
```

Gibt man keine `grenze` an, so wird der Wertebereich vollständig in dem angegebenen Definitionsbereich gezeichnet.

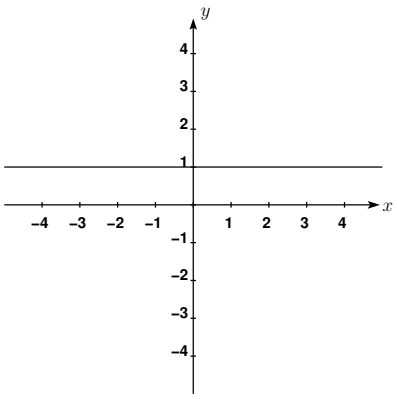
Beispiel

```
skalierung 1
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=-5,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5  x^3+x^2-0.4
```



oder auch

```
skalierung 1
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=-5,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 1
```



1.1.6 Zeichnen der Ableitungs- und Stammfunktion

Zusätzlich zur `plot` Funktion gibt es auch die `plotableitung` Funktion, die die Ableitung der gegebenen Funktion zeichnet. Der Aufruf ist genau wie bei der `plot` Funktion:

```
plotableitung [layer=<n>] [farbe=<n>] bereich=<xmin>,<xmax> [grenze=<ymin>,<
    ymax>] function
```

Desweiteren kann auch die Stammfunktion gezeichnet werden über den Befehl `plotstammfunktion`.

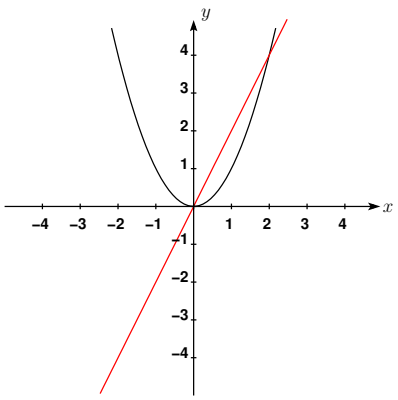
```
plotstammfunktion [layer=<n>] [farbe=<n>] bereich=<xmin>,<xmax> [grenze=<ymin
    >,<ymax>] [nullwert=<n>] function
```

Diese Funktion wird ähnlich zu den letzten beiden aufgerufen. Zusätzlich gibt `nullwert` an, wo der Schnittpunkt $(0, \text{nullwert})$ mit der $y-$Achse liegt. Gibt man nichts an, so liegt er in $(0;0)$.

Beispiele

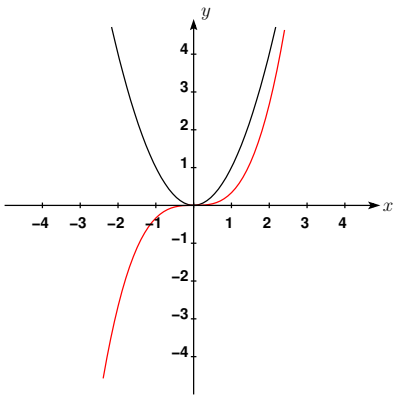
1. Ableitung:

```
skalierung 1
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=-5,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5  x^2
plotableitung layer=50 farbe=4 bereich=-5,5 grenze=-5,5  x^2
```



2. Stammfunktion

```
skalierung 1
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=-5,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5  x^2
plotstammfunktion farbe=4 bereich=-5,5 grenze=-5,5 x^2
```



1.1.7 Zeichnen von Integralen

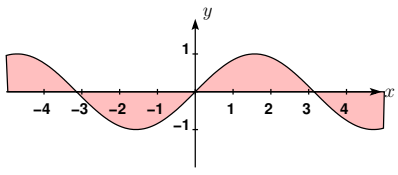
Mit dem Befehl `plotintegral` kann anstelle der Stammfunktion die Fläche unter der Funktion gezeichnet werden. Der Befehl lautet:

```
plotintegral [layer=<n>] [farbe=<n>] [fuellfarbe=<muster>,<füllfarbe>] bereich
    =<xmin>,<xmax> [grenze=<ymin>,<ymax>] function
```

Dabei wird die Funktion auch nur in den Grenzen gezeichnet. Möchte man nur eine kleine Fläche zeichnen, so muss man zusätzlich die `plot` Funktion nutzen, um den Graphen noch weiter zu zeichnen.

Beispiel

```
skalierung 1
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=-2,2,1 skala=normal text="$y$"
plotintegral layer=30 bereich=-5,5 grenze=-5,5 sin(x)
```



1.1.8 Beliebige Funktionen zeichnen

1. In der Version 2.5 ist neu hinzugekommen, dass man beliebige Funktionen zeichnen kann. D.h. man muss dem Programm eine Liste von (x, y)-Werten übergeben, dann werden diese verbunden und gezeichnet. Dazu benötigt man die beiden Funktionen `plotanzahlpunkte` und `plotxy`. Diese Funktionen sehen wie folgt aus:

```
plotanzahlpunkte [layer=<n>] [farbe=<n>] punkte=<n> plotxy
wert=<xwert>,<ywert>
```

   Der Standardlayer hat den Wert 90 und normalerweise wird die Funktion in schwarz gemalt. An `punkte` übergibt man, wie viele Zeilen die Funktion `plotxy` folgt.

2. So wie man eine beliebige Funktion zeichnen kann, kann man auch mit der Funktion `plotxypunkt` einen einzelnen Punkt in ein Koordinatensystem zeichnen.

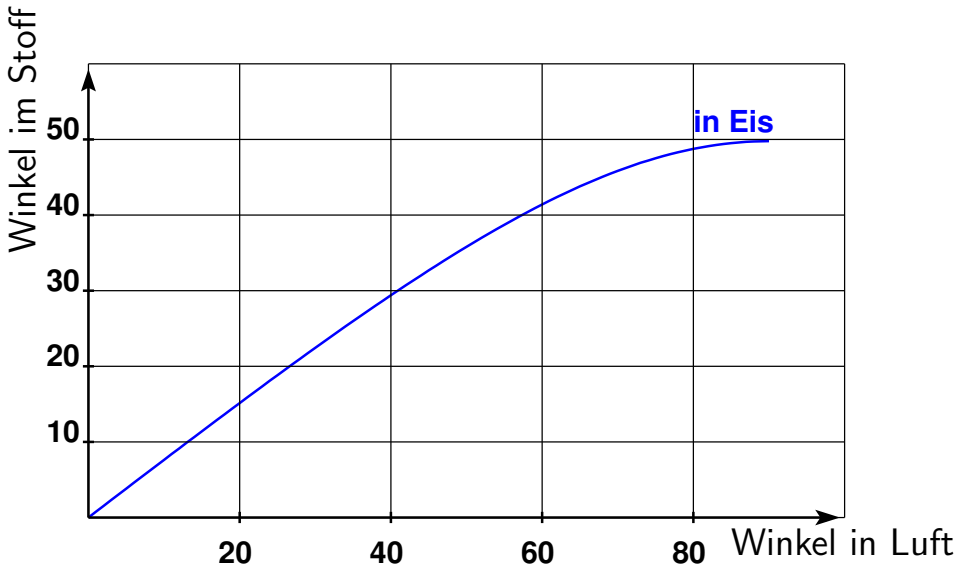
```
plotxypunkt [layer=<n>] [farbe=<n>] wert=<xwert>,<ywert>
```

   Der Standardlayer ist 90 und die Farbe schwarz.

Beispiel

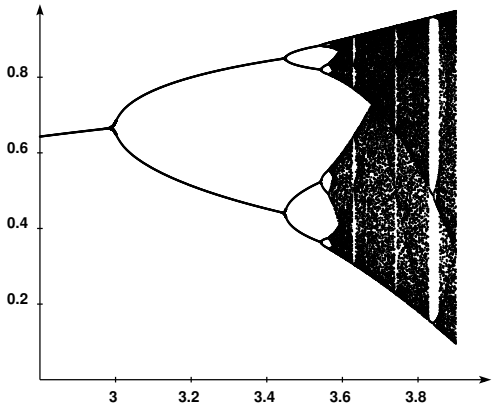
1. In dem folgenden Beispiel ist einmal der Einfallswinkel in Luft gegen den Brechungswinkel in Eis aufgetragen. Es sind allerdings nicht alle Werte hier aufgeführt.

```
skalierung x=0.1 y=0.1
kariert layer=50 abstandx=20,0,0 abstandy=10,0,0 bereich=0,0,90,60
xachse layer=10 bereich=0,100,20 skala=normal text=""
yachse layer=10 bereich=0,60,10 skala=normal text=""
beschriftung layer=10 farbe=0 xwert=-7 ywert=35 text="Winkel im Stoff"
    winkel=90 ausrichtung=l font=latex
beschriftung layer=10 farbe=0 xwert=85 ywert=-5 text="Winkel in Luft"
    winkel=0 ausrichtung=l font=latex
plotanzahlpunkte layer=90 farbe=1 punkte=91
plotxy wert=0,0.00
plotxy wert=1,0.76
plotxy wert=2,1.53
plotxy wert=3,2.29
plotxy wert=4,3.05
.
.
.
beschriftung layer=10 farbe=1
xwert=80 ywert=51 text="in Eis" winkel=0 ausrichtung=l font=ps
```



2. Hier ist einmal ein Feigenbaumdiagramm dargestellt, allerdings nicht in der vollen Länge.

```
skalierung 10
achsenschnitt x=2.8 y=0
xachse layer=10 bereich=2.8,4,0.2 skala=normal text=""
yachse layer=10 bereich=0,1,0.2 skala=normal text=""
plotxypunkt wert=28,6.42857
plotxypunkt wert=28.011,6.42998
plotxypunkt wert=28.022,6.43138
plotxypunkt wert=28.033,6.43278
plotxypunkt wert=28.044,6.43418
.
.
.
```



1.1.9 Ober- und Untersumme

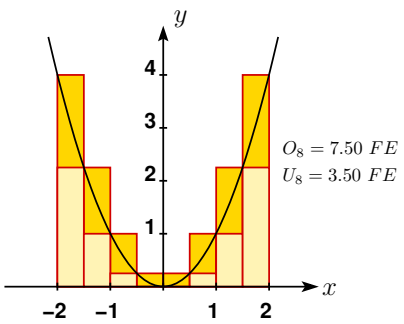
Mit der Funktion `plotoberunduntersumme` kann die Obersumme sowie die Untersumme für eine Funktion gezeichnet werden. Zusätzlich kann auch ausgegeben werden, wie groß die beiden Summen sind. Setzt man `summen` auf beide, so werden beide Summen eingezeichnet.

```
plotoberunduntersumme [layer=<n>] [farbe=<n>] bereich=<xmin>,<xmax> [grenze=<
    ymin>,<ymax>] schachtelung=<n> summen=<ober|unter|beide> [berechnung=ja]
    function
```

Setzt man `berechnung` auf ja, so werden die beiden Summen auch berechnet. Als Standard ist nein eingestellt.

Beispiel

```
skalierung 1
xachse layer=10 bereich=-3,3,1 skala=normal text="$x$"
yachse layer=10 bereich=0,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5  x^2
plotoberunduntersumme layer=60 bereich=-2,2 grenze=-5,5 schachtelung=8 summen=
    beide berechnung=ja x^2
```



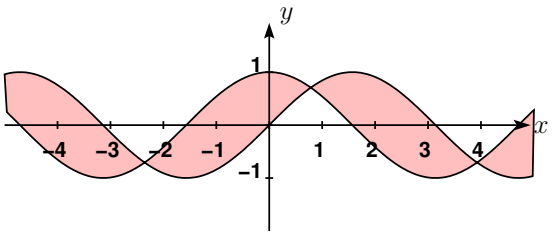
1.1.10 Fläche zwischen zwei Graphen

Mit der Funktion `einschluss` kann die Fläche zwischen zwei Graphen gezeichnet werden. Bei dieser Funktion gibt es die Möglichkeit die Fläche innerhalb des Bereiches [`xmin`, `xmax`] zu zeichnen. Setzt man `schnittpunkte`, so wird dieser Zeichenbereich durch die Schnittpunkte der Funktionen ersetzt. Setzt man `links`, so wird `xmin` zum linken Schnittpunkt. Analog für `rechts`. Hat die Funktion keine Schnittpunkte, so werden die vorgegebenen $x-$Werte beibehalten.

```
einschluss [layer=<n>] [farbe=<n>] [fuellfarbe=<muster>,<füllfarbe>] bereich=<
    xmin>,<xmax> [grenze=<ymin>,<ymax>] schnittpunkte=<links|rechts|beide|keine>
     function1 <function2>
```

Beispiel

```
skalierung 1
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=-2,2,1 skala=normal text="$y$"
einschluss layer=50 farbe=0 bereich=-5,5 schnittpunkte=keine sin(x) cos(x)
```



1.1.11 Extramarkierungen an den Achsen und Graphen

Es gibt einige Funktionen, mit denen man spezielle Punkte an den Achsen und an Funktionen markieren kann. Zu den Markierungsfunktionen gehören die folgenden Funktionen:

- `xmarkierung`
- `ymarkierung`
- `xymarkierung`

Zusätzlich gibt es die Beschriftungsfunktion `beschriftung`, um an einen beliebigen Punkt einen Text zu schreiben. Die `xmarkierung` und `ymarkierung` Funktionen werdenaufgerufen, um an den Achsen einen Extrapunkt zu markieren.

```
xmarkierung [layer=<n>] xwert=<n> text="text"
ymarkierung [layer=<n>] ywert=<n> text="text"
```

Es kann einfach nur ein leerer Strich an der Achse gezeichnet werden, aber auch ein Text eingetragen werden. Dieser Text kann allerdings nicht in LaTeXgesetzt werden.
Die `xymarkierung` Funktion wird dazu genutzt, um z.B. einen speziellen Punkt (x, y) zu markieren, aber es kann auch die Koordinate $(x, f(x))$ markiert werden, wenn der `ywert` nicht gesetzt wird. Setzt man den Parameter `linie`, so wird die Koordinate mit den jeweils benannten Achsen verbunden.

```
xymarkierung [layer=<n>] [farbe=<n>] xwert=<n> [ywert=<n>] [xtext="text"] [
    ytext="text"] [beschriftung=<ja|nein>] [linie=<x|y|xy>] [function]
```

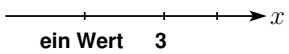
Mit der Funktion `beschriftung` kann man an einen beliebigen Punkt einen Text schreiben. Bei diesem Text kann die Ausrichtung, der Winkel und die Schrift beliebig gewählt werden.

```
beschriftung [layer=<n>] [farbe=<n>] xwert=<xwert> ywert=<ywert>
                text=<"beschriftung"> [winkel=<n>]
                ausrichtung=<l|c|r> [font=<latex|ps>]
```

Beispiele

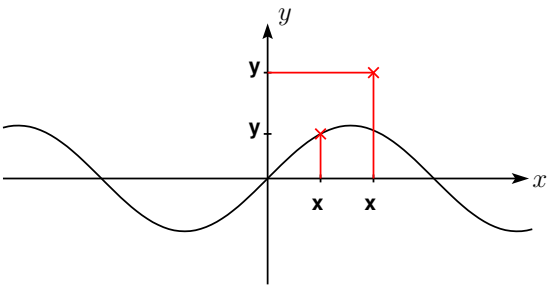
1. x- und ymarkierung

```
skalierung 1
xachse layer=10 bereich=0,5,1 skala=keine text="$x$"
xmarkierung layer=4 xwert=1.5 text="ein Wert"
xmarkierung layer=4 xwert=3 text="3"
xmarkierung layer=4 xwert=4 text=""
```



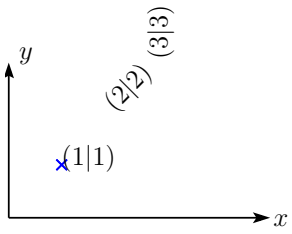
2. xymarkierung:

```
skalierung 1
xachse layer=10 bereich=-5,5,1 skala=keine text="$x$"
yachse layer=10 bereich=-2,3,1 skala=keine text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5 sin(x)
xymarkierung layer=8 farbe=4 xwert=2 ywert=2 xtext="x" ytext="y" linie=xy
    sin(x)
xymarkierung layer=8 farbe=4 xwert=1 xtext="x" ytext="y" linie=x sin(x)
```



3. Beschriftung:

```
skalierung 1
xachse layer=10 bereich=0,5,1 skala=keine text="$x$"
yachse layer=10 bereich=0,3,1 skala=keine text="$y$"
punktezeichnen layer=10 farbe=1 k1=1,1 k2=1,1 verbunden=nein
beschriftung layer=10 farbe=0 xwert=1 ywert=1 text="$(1|1)$" winkel=0
    ausrichtung=l font=latex
beschriftung layer=10 farbe=0 xwert=2 ywert=2 text="$(2|2)$" winkel=45
    ausrichtung=l font=latex
beschriftung layer=10 farbe=0 xwert=3 ywert=3 text="$(3|3)$" winkel=90
    ausrichtung=l font=latex
```



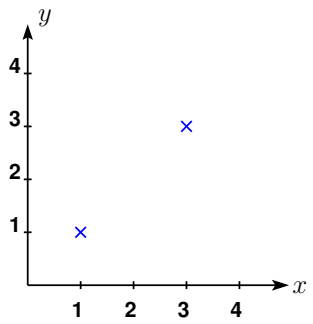
1.1.12 Punkte zeichnen

Mit der Funktion `punktezeichnen` können zwei Punkte bzw. Kreuze gezeichnet werden, die miteinander verbunden sind. Setzt man `verbunden` auf nein, so sind die beiden Koordinaten nicht verbunden. Als Standard sind sie verbunden.

```
punktezeichnen [layer=<n>] [farbe=<n>] k1=<x1>,<y1> k2=<x2>,<y2> [verbunden=<ja
    |nein>]
```

Besipiel

```
xachse layer=10 bereich=0,5,1 skala=normal text="$x$"
yachse layer=10 bereich=0,5,1 skala=normal text="$y$"
punktezeichnen layer=10 farbe=1 k1=1,1 k2=3,3 verbunden=nein
punktezeichnen layer=10 farbe=1 k1=1,-1 k2=-3,3 verbunden=ja
```



1.1.13 Tangenten, Tangentendreiecke und Steigungsdreiecke werden gezeichnet

Um Tangenten, Tangentendreiecke und Steigungsdreiecke an einem Graphen einzuzeichnen gibt es folgenden drei Funktionen. Zunächst kann man mit `plottangente` eine Tangente in einem Punkt x einzeichnen, die in dem Bereich $[x - \texttt{breite}, x + \texttt{breite}]$ liegt. Setzt man `markierung`, so wird zur $x-$Achse eine Linie gezeichnet.

```
plottangente [layer=<n>] [farbe=<n>] xwert=<n> breite=<n> [markierung=<ja|nein>
    function
```

Zusätzlich gibt es die Funktion `plottangentedreieck`. Mit dieser kann man in einem Punkt x ein Tangentendreieck einzeichnen und setzt man `markierung` auf `abl`, so wird der Wert der Steigung im Punkte x markiert.

```
plottangentedreieck [layer=<n>] [farbe=<n>] xwert=<n> breite=<n> [markierung=<
    wert|abl>] function
```

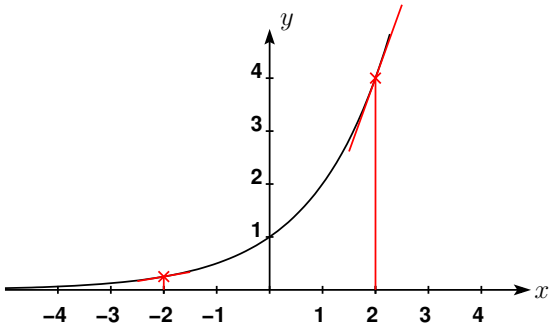
Analog zur `plottangentedreieck` gibt es auch eine `plotsteigungsdreieck`.

```
plotsteigungsdreieck [layer=<n>] [farbe=<n>] xwert=<n> breite=<n> [markierung=<
    wert|abl>] function
```

Beispiele

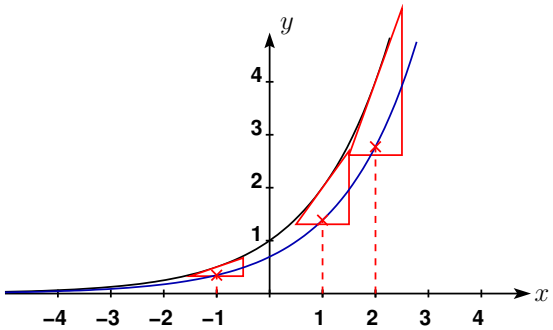
1. Tangente

```
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=0,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5 2^x
plottangente layer=3 farbe=4 xwert=-2 breite=0.5 markierung=ja 2^x
plottangente layer=3 farbe=4 xwert=2 breite=0.5 markierung=ja 2^x
```



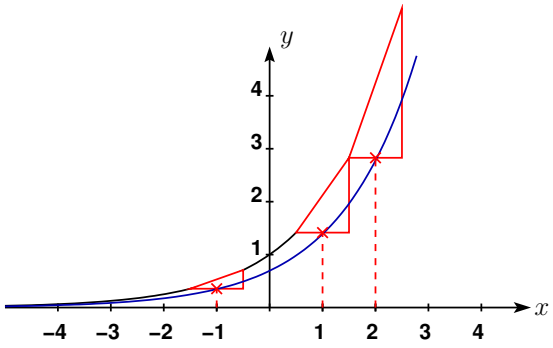
2. Tangentendreiecke

```
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=0,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5 2^x
plotableitung layer=2 farbe=9 bereich=-5,5 grenze=-5,5 2^x
plottangentedreieck layer=3 farbe=4 xwert=1 breite=0.5 markierung=abl 2^x
plottangentedreieck layer=3 farbe=4 xwert=-1 breite=0.5 markierung=abl 2^x
plottangentedreieck layer=3 farbe=4 xwert=2 breite=0.5 markierung=abl 2^x
```



3. Steigungsdreiecke

```
xachse layer=10 bereich=-5,5,1 skala=normal text="$x$"
yachse layer=10 bereich=0,5,1 skala=normal text="$y$"
plot layer=50 farbe=0 bereich=-5,5 grenze=-5,5 2^x
plotableitung layer=2 farbe=9 bereich=-5,5 grenze=-5,5 2^x
plotsteigungsdreieck layer=3 farbe=4 xwert=1 breite=0.5 markierung=abl 2^x
plotsteigungsdreieck layer=3 farbe=4 xwert=-1 breite=0.5 markierung=abl 2^
    x
plotsteigungsdreieck layer=3 farbe=4 xwert=2 breite=0.5 markierung=abl 2^x
```



1.1.14 Säulendiagramme zeichnen

In der Version 2.5 ist neu hinzugekommen, dass man auch Säulendiagramme zeichnen kann. Dies ist mit der Funktion

```
plotsaeule [layer=<n>] [farbe=<farbe,füllmuster,füllfarbe>]
        wert=<xwert>,<höhe> [breite=<säulenbreie>]
```

möglich. Hier muss man für die Farben mehrere Werte angeben, die die Farbe, Das Füllmuster und die Füllfarbe bestimmen. Die Standardwerte für diese Funktion sind:

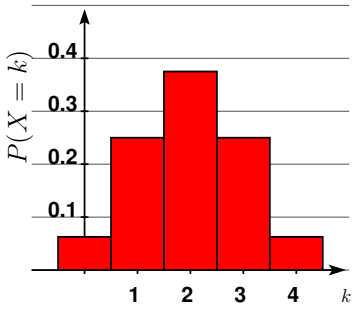
- Standardlayer=7
- Standardfarbe=0,35,27
- Standardbreite=0,6

Die Breite gibt an, wie breit eine Säule gezeichnet werden soll.

Beispiel

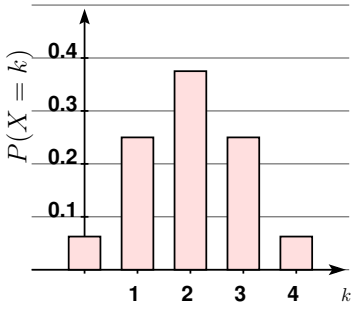
1.

```
skalierung x=1 y=10
kariert layer=50 abstandx=0,0,0 abstandy=0.1,0,0 bereich=-1,0,5,0.5
xachse layer=10 bereich=-1,5,1 skala=normal text="$k$"
yachse layer=10 bereich=0,0.5,0.1 skala=normal text=""
beschriftung layer=10 farbe=0 xwert=-1.1 ywert=0.2 text="$P(X=k)$" winkel
    =90 ausrichtung=l font=latex
plotsaeule farbe=0,20,4 wert=0,0.0625 breite=1
plotsaeule farbe=0,20,4 wert=1,0.2500 breite=1
plotsaeule farbe=0,20,4 wert=2,0.3750 breite=1
plotsaeule farbe=0,20,4 wert=3,0.2500 breite=1
plotsaeule farbe=0,20,4 wert=4,0.0625 breite=1
```



2.

```
skalierung x=1 y=10
kariert layer=50 abstandx=0,0,0 abstandy=0.1,0,0 bereich=-1,0,5,0.5
xachse layer=10 bereich=-1,5,1 skala=normal text="$k$"
yachse layer=10 bereich=0,0.5,0.1 skala=normal text=""
beschriftung layer=10 farbe=0 xwert=-1.1 ywert=0.2 text="$P(X=k)$" winkel
    =90 ausrichtung=l font=latex
plotsaeule wert=0,0.0625
plotsaeule wert=1,0.2500
plotsaeule wert=2,0.3750
plotsaeule wert=3,0.2500
plotsaeule wert=4,0.0625
```



1.1.15 Vektoren

Es können in dem Programm auch Vektoren gezeichnet werden. Dazu gibt es mehrere Funktionen. `plotvektorpolar`, `plotvektor` und `plotvektorparallel`. An die Funktion `plotvektorpolar` übergibt man einen Vektor über seinen Startpunkt, seine Länge und seinen Winkel.

```
plotvektorpolar [layer=<n>] [farbe=<n>] s1=<xstart>,<ystart> l1=<laengevektor>
    w1=<winkelvektor>
```

An die Funktion `plotvektor` übergibt man den Anfangspunkt und Endpunkt des Vektors.

```
plotvektor [layer=<n>] [farbe=<n>] k1=<xstart>,<ystart> k2=<xende>,<yende>
```

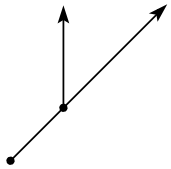
Zusätzlich kann man mit der Funktion `plotvektorparallel` bis zu sechs Vektoren zeichnen, die zu einem Ersatzvektor zusammen gefasst werden.

```
plotvektorparallel [layer=<n>] [farbe=<n>] [start=<xstart>,<ystart>]
(länge1,winkel1) [(länge2,winkel2)] [(länge3,winkel3)] [(länge4,winkel4)] [(
    länge5,winkel5)] [(länge6,winkel6)] [ersatzvektor=<parallel|ersatz>]
```

Beispiele

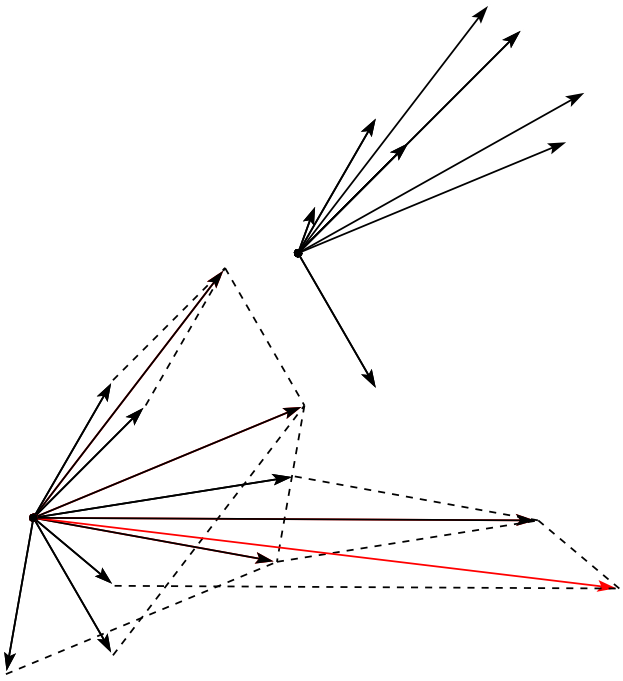
1. Vektoren

```
plotvektorpolar layer=80 farbe=0 s1=1,1 l1=2 w1=90
plotvektor layer=80 farbe=0 k1=0,0 k2=3,3
```



2. Vektorparallelogramm

```
plotvektorparallel layer=80 farbe=0 start=3,3 (3,45) (3,60) (3,-60) (1,70)
    (6,45)
plotvektorparallel layer=80 farbe=0 start=-2,-2 (3,45) (3,60) (3,-60)
   (3,-100) (5,9) (2,-40) ersatzvektor=parallel
```



1.2 Quellcode

Hier kommt nun der Quellcode:

```
#!/usr/bin/perl -w

######################################################################
# Funktionsplotter Skript zum erstellen von Grafiken in Xfig
# Copyright (C) 2006 Xenia Rendtel
#
# VERSION 2.5
# Letzte Änderung: 01.08.2006
#
#
# Dieses Programm ist freie Software. Sie können es unter den Bedingungen der
# GNU General Public License, wie von der Free Software Foundation
# veröffentlicht, weitergeben und/oder modifizieren, entweder gemäß Version 2
# der Lizenz oder (nach Ihrer Option) jeder späteren Version.
#
# Die Veröffentlichung dieses Programms erfolgt in der Hoffnung, daß es Ihnen
# von Nutzen sein wird, aber OHNE IRGENDEINE GARANTIE, sogar ohne die implizite
# Garantie der MARKTREIFE oder der VERWENDBARKEIT FÜR EINEN BESTIMMTEN
# ZWECK. Details finden Sie in der GNU General Public License.
#
# Sie sollten ein Exemplar der GNU General Public License zusammen mit diesem
# Programm erhalten haben. Falls nicht, schreiben Sie an die Free Software
# Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110, USA.
#
# Bei Fehlern schreiben Sie diese einfach an
# webmaster@rendtel.de
######################################################################

use POSIX qw/floor ceil/;
use Math::Trig;

######################################################################
########## Konstanten und Variablen
######################################################################

use constant {
    DASHED              => 1,
    SOLID               => 0,
    DOTTED              => 2,
    DASHDOTTED          => 3,
    DASHDOUBLEDOTTED    => 4,
    DASHTRIPPLEDOTTED   => 5,
    DFONT               => 0,
    ROMAN               => 1,
    BOLD                => 2,
    ITALIC              => 3,
    SANSSERIF           => 4,
    NARROWHELVETICABOLD   => 18,
    ALIGNLEFT           => 0,
    ALIGNCENTER         => 1,
    ALIGNRIGHT          => 2,
    PAPERX              => 29.7,
    PAPERY              => 21,
    ZEICHENDICKE        => 1,
    LINIENDICKE         => 2,
    XFIGSCALE           => 450,
    EBENE               => -1
    };

######################################################################
########## Funktionen für das öffnen und schließen von
########## xfig-Dateien
######################################################################

sub openfig {
    my $name=shift(@_);

    open FIGFILE,">".$name;
    print FIGFILE "#FIG 3.2  Produced by script\n";
    print FIGFILE "Landscape\n";
    print FIGFILE "Center\n";
    print FIGFILE "Metric\n";
    print FIGFILE "A4\n";
    print FIGFILE "100.00\n";
    print FIGFILE "Single\n";
    print FIGFILE "-2\n";
    print FIGFILE "1200 2\n";
}

sub closefig {
    close FIGFILE;
}

######################################################################
########## Positionen bestimmen in Xfig
```

```
#####################################################################

sub figx {
    my $x=shift(@_);
    return $x*XFIGSCALE;
}

sub figy {
    my $y=shift(@_);
    return (PAPERY-$y)*XFIGSCALE;
}

#####################################################################
########## Öffnen von verschiedenen Objektgruppen in xfig
#####################################################################

# eine Polyline wird geöffnet. Dabei muss angegeben werden, wie viele Punkte
# die Linie hat
sub openpline {
    my $layer=shift(@_);
    my $linestyle=shift(@_);
    my $thickness=shift(@_);
    my $farbe=shift(@_);
    my $points=shift(@_);

    printf(FIGFILE "2 1 %i %i %i 7 %i -1 -1 4.000 0 0 -1 0 0 %i\n",
           $linestyle,$thickness, $farbe, $layer,$points);
}

# eine Box wird geöffnet. Möchte man ein Rechteck zeichnen, muss man der Box 5
# Punkte übergeben und mit plinepoint fünf Punkte angeben. Die vier Eckpunkte
# eines Rechtecks und den Anfangspunkt als fünften Punkt.
sub openbox {
    my $layer=shift(@_);
    my $linestyle=shift(@_);
    my $thickness=shift(@_);
    my $farbe=shift(@_);
    my $gefuellt=shift(@_);
    my $fuellfarbe=shift(@_);
    my $points=shift(@_);

    printf(FIGFILE "2 3 %i %i %i %i %i -1 %i 0.000 0 0 -1 0 0 %i\n",
           $linestyle,$thickness, $farbe, $fuellfarbe, $layer, $gefuellt, $points);
}

# Text wird geöffnet
sub opentext {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $font=shift(@_);
    my $fontsize=shift(@_);
    my $winkel=shift(@_);
    my $align=shift(@_);

    my $winkelgerundet = sprintf ("%0.4f",deg2rad($winkel));

    printf(FIGFILE "4 %i %i %i -1 %i %i %s 2 225 525 ",
           $align, $farbe, $layer, $font, $fontsize,$winkelgerundet);
}

# Postscripttext wird geöffnet
sub openpstext {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $font=shift(@_);
    my $fontsize=shift(@_);
    my $winkel=shift(@_);
    my $align=shift(@_);

    my $winkelgerundet =sprintf ("%0.4f",deg2rad($winkel));

    printf(FIGFILE "4 %i %i %i -1 %i %i %s 4 225 525 ",
           $align, $farbe, $layer, $font, $fontsize,$winkelgerundet);
}

# Ein Pfeil wird geöffnet
sub openarrow {
    my $layer=shift(@_);
    my $linestyle=shift(@_);
    my $thickness=shift(@_);
    my $arrowtype=shift(@_);
    my $arrowstyle=shift(@_);
    my $arrowthick=shift(@_);
    my $arrowwidth=shift(@_);
    my $arrowheight=shift(@_);
    my $farbe=shift(@_);
    my $points=shift(@_);

    printf(FIGFILE "2 1 %i %i %i 7 %i -1 -1 0.000 0 0 -1 1 0 %i\n        %i %i %i %i %i\n",
           $linestyle, $thickness, $farbe, $layer, $points, $arrowtype,
```

```
            $arrowstyle, $arrowthick, $arrowwidth, $arrowheight);
}

# Compound öffnen und schließen
sub opencompound {
    my $xobenlinks=shift(@_);
    my $yobenlinks=shift(@_);
    my $xuntenrechts=shift(@_);
    my $yuntenrechts=shift(@_);
    printf(FIGFILE "6 %i %i %i %i\n ",
           figx($xobenlinks),
           figy($yobenlinks),
           figx($xuntenrechts),
           figy($yuntenrechts));
}

sub closecompound {
    printf(FIGFILE "-6\n");
}

#########################################################################
########## Schreiben von verschiedenen Objekten in xfig
#########################################################################

# Ein Kreis wird geschrieben
sub circle {
    my $layer=shift(@_);
    my $linestyle=shift(@_);
    my $thickness=shift(@_);
    my $x1=shift(@_);
    my $y1=shift(@_);
    my $radius=shift(@_);
    my $farbe=shift(@_);

    printf(FIGFILE "1 3 %i %i %i 20 %i -1 0 4.000 1 1 %i %i %i %i %i %i %i %i\n",
           $linestyle, $thickness, $farbe, $layer, figx($x1),
           figy($y1), ($radius), ($radius), $x1, $y1, $x1, $y1);
}

# ein Rechteck wird gezeichnet. Angabe zwei gegenüberliegender Punkte genügt
sub rechteck {
    my $layer=shift(@_);
    my $x1=shift(@_);
    my $y1=shift(@_);
    my $x2=shift(@_);
    my $y2=shift(@_);
    my $linestyle=shift(@_);
    my $thickness=shift(@_);
    my $farbe=shift(@_);
    my $gefuellt=shift(@_);
    my $fuellfarbe=shift(@_);

    openbox($layer,$linestyle,$thickness,$farbe,$gefuellt,$fuellfarbe,5);
    plinepoint($x1,$y1);
    plinepoint($x2,$y1);
    plinepoint($x2,$y2);
    plinepoint($x1,$y2);
    plinepoint($x1,$y1);
}

# Text wird geschrieben
sub ptext {
    my $x=shift(@_);
    my $y=shift(@_);
    my $text=shift(@_);

    printf(FIGFILE "%i %i %s \\001\n",
           figx($x), figy($y), $text);
}

# x und y-Koordinaten werden geschrieben
sub plinepoint {
    my $x=shift(@_);
    my $y=shift(@_);

    printf(FIGFILE "     %i %i\n",
           figx($x), figy($y));
}

#########################################################################
########## Farben werden geschrieben
#########################################################################

# neue RGB-Farbe defininieren, mindestens Nr. 32
sub neuefarbe {
    my $nummer=shift(@_);
    my $r=shift(@_);
    my $g=shift(@_);
    my $b=shift(@_);
```

```
    printf(FIGFILE "0 %i #%02x%02x%02x\n",
           $nummer, $r, $g, $b);
}

sub neugrau {
    my $nummer=shift(@_);
    my $grau=shift(@_);
    neuefarbe($nummer,$grau,$grau,$grau);
}

#####################################################################
########## Zeichenroutinen für zusammengesetzte Objekte
#####################################################################

# Ein Kreuzchen in (x,y) zeichnen
sub kreuzchen {
    my $layer=shift(@_);
    my $x=shift(@_);
    my $y=shift(@_);
    my $farbe=shift(@_);

    my $groesse=0.2;

    $groesse=$groesse/2;

    if ($scalex>1) {
        $groesse=$groesse/($scalex/2);
    }

    if ($scaley>1) {
        $groesse=$groesse/($scaley/2);

    }

    my $xlinks=$x-$groesse*$scalex;
    my $xrechts=$x+$groesse*$scalex;
    my $yunten=$y-$groesse*$scaley;
    my $yoben=$y+$groesse*$scaley;

    opencompound(abrunden($xlinks,0.1),aufrunden($yoben,0.1),
                 aufrunden($xrechts,0.1),abrunden($yunten,0.1));

    openpline($layer,SOLID,LINIENDICKE,$farbe,2);
    plinepoint($xlinks,$yunten);
    plinepoint($xrechts,$yoben);
    openpline($layer,SOLID,LINIENDICKE,$farbe,2);
    plinepoint($xlinks,$yoben);
    plinepoint($xrechts,$yunten);

    closecompound;
}

# Ein Pluszeichen in (x,y) zeichnen
sub pluszeichen {
    my $layer=shift(@_);
    my $x=shift(@_);
    my $y=shift(@_);
    my $farbe=shift(@_);

    my $groesse=0.2;

    $groesse=$groesse/2;

    my $xlinks=$x-$groesse*$scalex;
    my $xrechts=$x+$groesse*$scalex;
    my $yunten=$y-$groesse*$scaley;
    my $yoben=$y+$groesse*$scaley;

    opencompound(abrunden($xlinks,0.1),aufrunden($yoben,0.1),
                 aufrunden($xrechts,0.1),abrunden($yunten,0.1));
    openpline($layer,SOLID,LINIENDICKE,$farbe,2);
    plinepoint($xlinks,$y);
    plinepoint($xrechts,$y);
    openpline($layer,SOLID,LINIENDICKE,$farbe,2);
    plinepoint($x,$yoben);
    plinepoint($x,$yunten);
    closecompound;
}

#####################################################################
########## Auswerten einer Funktion an der Stelle x
#####################################################################

sub calcfunk {
    my $funktion=shift(@_);
    my $variable=shift(@_);
    my $wert=shift(@_);

    my $y=$funktion;
```

```
    $y =~ s/\^/**/g;
    $y =~ s/\b$variable\b/($wert)/g;
    $y=eval($y);

    return $y;
}

#####################################################################
########## Funktionen zum auf- und abrunden auf das nächste
########## Vielfache einer gegebenen Zahl
#####################################################################

sub abrunden {
    my $wert=shift(@_);
    my $schritt=shift(@_);

    return $schritt*floor($wert/$schritt);
}

sub aufrunden {
    my $wert=shift(@_);
    my $schritt=shift(@_);

    return $schritt*ceil($wert/$schritt);
}

#####################################################################
########## Funktionen zum Berechnen von maximal und minimal
########## Werten im Werte- und Definitionsbereich.
#####################################################################

# Berechnet den ymin und ymax Wert für eine Funktion
sub calcyminmax {
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    my $minmax=shift(@_);

    my ($schritt, $xsuche, $ysuche, $ymin, $ymax);
    my $schrittanzahl=500;

# Suche von ymin, indem der ganze Definitionsbereich durchgegangen wird und
# der zugehörige Wertebereich ausgerechnet wird.

    $ymin=calcfunk($func,"x",$xmin);
    $ymax=calcfunk($func,"x",$xmin);
    for ($schritt=0; $schritt<$schrittanzahl; $schritt++) {
        $xsuche=$xmin+($xmax-$xmin)*($schritt/($schrittanzahl-1));
        $ysuche=calcfunk($func,"x",$xsuche);
        $ymin=$ysuche if ($ymin>$ysuche);
        $ymax=$ysuche if ($ymax<$ysuche);
    }

    return $ymin if ($minmax eq "min");
    return $ymax if ($minmax eq "max");
    printf("calcyminmax: Unbekannter Parameter minmax='%s'\n",$minmax);
}

sub calcymin {
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    return calcyminmax($func,$xmin,$xmax,"min");

}

sub calcymax {
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    return calcyminmax($func,$xmin,$xmax,"max");

}

# Berechnet den ymin und ymax Wert für zwei Funktion
sub calcfymin {
    my $func1=shift(@_);
    my $func2=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);

    my $ymin1=calcymin($func1,$xmin,$xmax);
    my $ymin2=calcymin($func2,$xmin,$xmax);

    if ($ymin1<$ymin2) {
        return $ymin1;
    } else {
        return $ymin2;
    }
}
```

```
sub calcfymax {
    my $func1=shift(@_);
    my $func2=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);

    my $ymax1=calcymax($func1,$xmin,$xmax);
    my $ymax2=calcymax($func2,$xmin,$xmax);

    if ($ymax1>$ymax2) {
        return $ymax1;
    } else {
        return $ymax2;
    }
}

# Verschiebt den xmin und xmax Wert so, dass Graph komplett in [ymin..ymax] liegt
sub calcxminmax {
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    my $ymin=shift(@_);
    my $ymax=shift(@_);
    my $minmax=shift(@_);

    my ($bereichmin,$bereichmax,$schritt,$xsuche,$ysuche);
    my $schrittanzahl=1000;
    my $gueltig=0;

    $bereichmin=$bereichmax=$xmin;
    for ($schritt=0;$schritt<$schrittanzahl;$schritt++) {
        $xsuche=$xmin+($xmax-$xmin)*($schritt/($schrittanzahl-1));
        $ysuche=calcfunk($func,"x",$xsuche);
        if (($ysuche>=$ymin) && ($ysuche<=$ymax)) {
            $bereichmin=$xsuche if (! $gueltig);
            $bereichmax=$xsuche;
            $gueltig=1;
        } else {
            $gueltig=0;
        }
    }

    return $bereichmin if ($minmax eq "min");
    return $bereichmax if ($minmax eq "max");
    printf("calcxminmax: Unbekannter Parameter minmax='%s'\n",$minmax);
}

sub calcxmin {
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    my $ymin=shift(@_);
    my $ymax=shift(@_);

    return calcxminmax($func,$xmin,$xmax,$ymin,$ymax,"min");
}

sub calcxmax {
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    my $ymin=shift(@_);
    my $ymax=shift(@_);

    return calcxminmax($func,$xmin,$xmax,$ymin,$ymax,"max");
}

#########################################################################
########## Berechnung der Ableitung in einem Punkt
#########################################################################

# Berechnung der Ableitung über den Differenzenquotienten
sub calcableitung {
    my $func=shift(@_);
    my $xwert=shift(@_);

    my $hwert=0.00000001;
    my $ywert=(calcfunk($func,"x",$xwert+$hwert)-calcfunk($func,"x",$xwert))/$hwert;

    return $ywert;
}

#########################################################################
########## Berechnung des Funktionswertes t(x), wenn t die
########## Tangente an die Funktion func im Punkte festwert ist
#########################################################################

sub calctangente {
    my $func=shift(@_);
```

```
    my $xwert=shift(@_);
    my $festwert=shift(@_);

    return calcfunk($func,"x",$festwert)
        + (calcableitung($func,$festwert)*($xwert-$festwert));
}

#####################################################################
########## Berechnung der Schnittpunkte von zwei Funktionen
#####################################################################

sub calcschnittpunkte {
    my $func1=shift(@_);
    my $func2=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    my $linksrechts=shift(@_);

    my ($xsuche,$links, $rechts,@xpunkte, @xselbst);

    my $ysuche1=calcfunk($func1,"x",$xmin);
    my $ysuche2=calcfunk($func2,"x",$xmin);
    my $welchegroesser=($ysuche1>$ysuche2)?1:2;
    my $schrittanzahl=10000;
    my $zaehler=0;

    for ($schritt=0;$schritt<$schrittanzahl;$schritt++) {
        $xsuche=$xmin+($xmax-$xmin)*($schritt/($schrittanzahl-1));
        $ysuche1=calcfunk($func1,"x",$xsuche);
        $ysuche2=calcfunk($func2,"x",$xsuche);

        if (($ysuche1>$ysuche2) xor ($welchegroesser==1)) {
            $welchegroesser=($ysuche1>$ysuche2)?1:2;
            $zaehler++;
            if ($zaehler>1) {
                $rechts=$xsuche;
            } else {
                $links=$xsuche;
                $rechts=$xsuche;

            }
        }
    }
    if ($zaehler==0) {
        $links=$xmin;
        $rechts=$xmax;
    }

    return $links if ($linksrechts eq "links");
    return $rechts if ($linksrechts eq "rechts");
    printf("calcschnittpunkte: Unbekannter Parameter linksrechts='%s'\n",$linksrechts);
}

sub calcschnittlinks {
    my $func1=shift(@_);
    my $func2=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    calcschnittpunkte($func1,$func2,$xmin,$xmax,"links");
}

sub calcschnittrechts {
    my $func1=shift(@_);
    my $func2=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    calcschnittpunkte($func1,$func2,$xmin,$xmax,"rechts");
}

#####################################################################
########## Berechnungen für Vektoren
#####################################################################

sub calcendpunktx {
    my $xstart=shift(@_);
    my $laenge=shift(@_);
    my $winkel=shift(@_);

    my $winkelbogen=deg2rad($winkel);
    my $dx=cos($winkelbogen)*$laenge;
    return $xstart+$dx;
}

sub calcendpunkty {
    my $ystart=shift(@_);
    my $laenge=shift(@_);
    my $winkel=shift(@_);
    return $ystart+sin(deg2rad($winkel))*$laenge;
}
```

```
sub calcsumlaenge {
    my $x1=shift(@_);
    my $y1=shift(@_);
    my $l1=shift(@_);
    my $w1=shift(@_);
    my $l2=shift(@_);
    my $w2=shift(@_);

    my $xsum=calcendpunktx(0,$l1,$w1)+calcendpunktx(0,$l2,$w2);
    my $ysum=calcendpunkty(0,$l1,$w1)+calcendpunkty(0,$l2,$w2);
    return sqrt(($xsum*$xsum)+($ysum*$ysum));
}

# Winkel wird in Grad ausgegeben
sub calcneuerwinkel {
    my $x1=shift(@_);
    my $y1=shift(@_);
    my $l1=shift(@_);
    my $w1=shift(@_);
    my $l2=shift(@_);
    my $w2=shift(@_);

    my $x12=calcendpunktx($x1,$l1,$w1);
    my $y12=calcendpunkty($y1,$l1,$w1);
    my $x22=calcendpunktx($x12,$l2,$w2);
    my $y22=calcendpunkty($y12,$l2,$w2);
    my $dx=$x22-$x1;
    my $dy=$y22-$y1;

    return rad2deg(atan2($dy,$dx));
}

######################################################################
########## Erzeugen von kariertem Papier, Millimeterpapier,
########## Matheheftpapier
######################################################################

# Karomuster wird erzeugt
sub kariert {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $abstandx=shift(@_);
    my $dickx=shift(@_);
    my $ganzdickx=shift(@_);
    my $abstandy=shift(@_);
    my $dicky=shift(@_);
    my $ganzdicky=shift(@_);

    my $schritt;
    my $xmingitter;
    my $xmaxgitter;
    my $ymingitter;
    my $ymaxgitter;

    # Gitter muss größer sein als vorgegebene x- und y-Werte
    if ($abstandx==0) {
        $xmingitter=$xmin;
        $xmaxgitter=$xmax;
    } else {
        $xmingitter=abrunden($xmin,$abstandx);
        $xmaxgitter=aufrunden($xmax,$abstandx);
    }

    if ($abstandy==0) {
        $ymingitter=$ymin;
        $ymaxgitter=$ymax;
    } else {
        $ymingitter=abrunden($ymin,$abstandy);
        $ymaxgitter=aufrunden($ymax,$abstandy);

    }

    my $dicke1=ZEICHENDICKE+0, $dicke2=ZEICHENDICKE+1, $dicke3=ZEICHENDICKE+2;
    my $farbe1=$farbe, $farbe2=$farbe, $farbe3=$farbe;

    # Die Linien entlang der y-Achse werden gezeichnet
    if ($abstandx!=0) {
        # Anfangs- und Endwerte für die Zeichenschleife
        my $anfangx=$xmingitter/$abstandx;
        my $endex=$xmaxgitter/$abstandx;

        # für Millimeterpapier sollen die Linien grau gezeichnet werden
        if ($abstandx<0.5) {
            $dicke1=ZEICHENDICKE;
            $dicke2=ZEICHENDICKE;
            $dicke3=ZEICHENDICKE+1;
```

```
            $farbe1=32;
            $farbe2=0;
            $farbe3=0;
        }
        # horizontale Linien werden gezeichnet
        for ($schritt=$anfangx;$schritt<=$endex;$schritt++) {
            if (($ganzdickx!=0) && (($schritt % $ganzdickx)==0)) {
                openpline($layer,SOLID,$dicke3,$farbe3,2);
                plinepoint($schritt*$abstandx*$scalex,$ymingitter*$scaley);
                plinepoint($schritt*$abstandx*$scalex,$ymaxgitter*$scaley);
            } else {
                if (($dickx!=0) && (($schritt % $dickx)==0)) {
                    openpline($layer,SOLID,$dicke2,$farbe2,2);
                    plinepoint($schritt*$abstandx*$scalex,$ymingitter*$scaley);
                    plinepoint($schritt*$abstandx*$scalex,$ymaxgitter*$scaley);
                } else {
                    openpline($layer,SOLID,$dicke1,$farbe1,2);
                    plinepoint($schritt*$abstandx*$scalex,$ymingitter*$scaley);
                    plinepoint($schritt*$abstandx*$scalex,$ymaxgitter*$scaley);
                }
            }
        }
    }

    # vertikale Linien werden gezeichnet
    if ($abstandy!=0) {
        my $anfangy=$ymingitter/$abstandy;
        my $endey=$ymaxgitter/$abstandy;

        if ($abstandy<0.5) {
            $dicke1=ZEICHENDICKE;
            $dicke2=ZEICHENDICKE;
            $dicke3=ZEICHENDICKE+1;
            $farbe1=32;
            $farbe2=0;
            $farbe3=0;
        }

        for ($schritt=$anfangy;$schritt<=$endey;$schritt++) {
            if (($ganzdicky!=0) && (($schritt % $ganzdicky)==0)) {
                openpline($layer,SOLID,$dicke3,$farbe3,2);
                plinepoint($xmingitter*$scalex,$schritt*$abstandy*$scaley);
                plinepoint($xmaxgitter*$scalex,$schritt*$abstandy*$scaley);
            } else {
                if (($dicky!=0) && (($schritt % $dicky)==0)) {
                    openpline($layer,SOLID,$dicke2,$farbe2,2);
                    plinepoint($xmingitter*$scalex,$schritt*$abstandy*$scaley);
                    plinepoint($xmaxgitter*$scalex,$schritt*$abstandy*$scaley);
                } else {
                    openpline($layer,SOLID,$dicke1,$farbe1,2);
                    plinepoint($xmingitter*$scalex,$schritt*$abstandy*$scaley);
                    plinepoint($xmaxgitter*$scalex,$schritt*$abstandy*$scaley);
                }
            }
        }
    }
}

# Millimeterpapier wird erzeugt, indem die kariert-Funktion aufgerufen wird
sub millimeterpapier {
    my $layer=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    kariert($layer,0,$xmin,$ymin,$xmax,$ymax,0.1,5,10,0.1,5,10);
}

# Matheheftpapier wird erzeugt, indem die kariert-Funktion aufgerufen wird
sub matheheft {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    kariert($layer,$farbe,$xmin,$ymin,$xmax,$ymax,0.5,0,0,0.5,0,0);
}

# Ein Raster mit Punkten wird erstellt
sub grid {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $abstandx=shift(@_);
    my $dickx=shift(@_);
    my $ganzdickx=shift(@_);
```

```
    my $abstandy=shift(@_);
    my $dicky=shift(@_);
    my $ganzdicky=shift(@_);

    my ($schritt,$schritt2,$xmingitter,$xmaxgitter,$ymingitter,$ymaxgitter);

    my $radius=20;
    my $pdicke=20;
    my $plaenge=25;

    $xmingitter=$xmin;
    $xmaxgitter=$xmax;
    $ymingitter=$ymin;
    $ymaxgitter=$ymax;

    # Die Punkte entlang der y-Achse werden gezeichnet
     if ($abstandx!=0) {
        my $anfangx=$xmingitter/$dickx;
        my $endex=$xmaxgitter/$dickx;
        my $schrittanzahl1=(($xmaxgitter-$xmingitter)/$ganzdickx);
        my $schrittanzahl2=(($ymaxgitter-$ymingitter)/$dickx)*$abstandx;

        for ($schritt=0;$schritt<=$schrittanzahl1;$schritt++) {

            for ($schritt2=0; $schritt2<=$schrittanzahl2; $schritt2++) {

                if ((($xmingitter+($xmaxgitter-$xmingitter)*($schritt/($schrittanzahl1)))!=$schnittx) && ((
                     $ymingitter+($ymaxgitter-$ymingitter)*($schritt2/($schrittanzahl2)))!=$schnitty)) {
                circle(50,0, 1,
                       ($xmingitter+($xmaxgitter-$xmingitter)*($schritt/($schrittanzahl1)))*$scalex,
                       ($ymingitter+($ymaxgitter-$ymingitter)*($schritt2/($schrittanzahl2)))*$scaley,
                       $radius,$farbe); } } }}

    # Die Punkte entlang der x-Achse werden gezeichnet
    if ($abstandy!=0) {
        my $anfangy=$ymingitter/$dicky;
        my $endey=$ymaxgitter/$dicky;
        my $schrittanzahl1=(($xmaxgitter-$xmingitter)/$dicky)*$abstandy;;
        my $schrittanzahl2=(($ymaxgitter-$ymingitter)/$ganzdicky);

        for ($schritt=0;$schritt<=$schrittanzahl1;$schritt++) {

            for ($schritt2=0; $schritt2<=$schrittanzahl2; $schritt2++) {

                if ((($xmingitter+($xmaxgitter-$xmingitter)*($schritt/($schrittanzahl1)))!=$schnittx) && ((
                     $ymingitter+($ymaxgitter-$ymingitter)*($schritt2/($schrittanzahl2)))!=$schnitty)) {
                   circle(50,0, 1,
                      ($xmingitter+($xmaxgitter-$xmingitter)*($schritt/($schrittanzahl1)))*$scalex,
                      ($ymingitter+($ymaxgitter-$ymingitter)*($schritt2/($schrittanzahl2)))*$scaley,
                      $radius,$farbe); } } }}

}

# Pünktchen im Koordinatensystem
sub punkte {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $abstand=shift(@_);

    # Gitter muss größer sein als vorgegebene x- und y-Werte
    my $xmingitter=abrunden($xmin,$abstand);
    my $xmaxgitter=aufrunden($xmax,$abstand);
    my $ymingitter=abrunden($ymin,$abstand);
    my $ymaxgitter=aufrunden($ymax,$abstand);

    # Anfangs- und Endwerte für die Zeichenschleife
    my $anfangx=$xmingitter/$abstand;
    my $endex=$xmaxgitter/$abstand;
    my $anfangy=$ymingitter/$abstand;
    my $endey=$ymaxgitter/$abstand;
    my $farbe1=$farbe;
    my $schrittx=3;
    my $schritty=3;

    if ($abstand<=0.5) {
        neugrau(32,80);
        $farbe1=32;
    }

    opencompound(abrunden($xmingitter,0.1),aufrunden($ymaxgitter,0.1),
                 aufrunden($xmaxgitter,0.1),abrunden($ymingitter,0.1));

    for ($schrittx=$anfangx;$schrittx<=$endex;$schrittx++) {
        for ($schritty=$anfangy;$schritty<=$endey;$schritty++) {
            pluszeichen($layer,$schrittx*$abstand*$scalex,
                        $schritty*$abstand*$scaley,$farbe1)
```

```
                if (($schrittx!=$schnittx) && ($schritty!=$schnitty));
        }
     }
    closecompound;
}

#####################################################################
########## Ein Koordinatensystem soll gezeichnet werden,
########## dazu werden die Achsen einzeln gezeichnet
#####################################################################

# die x-Achse wird gezeichnet
sub xachse {
    my $layer=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    my $xschritt=shift(@_);
    my $xskala=shift(@_);
    my $xtext=shift(@_);

    # Dicke und Länge des Pfeils wird definiert
    my $dicke=12;
    my $laenge=15;

    opencompound(abrunden($xmin,0.1),aufrunden(0.075,0.1),
                 aufrunden($xmax,0.1),-0.6);
    # der Achsenpfeil wird gezeichnet
    openarrow($layer,SOLID,ZEICHENDICKE+1,2,1,4,$dicke,$laenge,0,2);
    plinepoint($xmin*$scalex,$schnitty*$scaley);
    plinepoint($xmax*$scalex,$schnitty*$scaley);

    # Striche müssen innerhalb des Pfeils liegen
    my $xminachse=aufrunden($xmin+0.25*$xschritt,$xschritt);
    my $xmaxachse=abrunden($xmax-0.25*$xschritt,$xschritt);

    # Anfangs und Endwerte für die Schleife werden gesetzt
    my $anfangx=$xminachse/$xschritt;
    my $endex=$xmaxachse/$xschritt;

    my $schritt;

    if ($xskala eq "keine") {  }
    if ($xskala eq "nurstriche") {
        for ($schritt=$anfangx;$schritt<$endex;$schritt++) {
            xmarkierung($layer,$schritt*$xschritt,"","ps");   }
    }
    if ($xskala eq "nureins") {
        xmarkierung($layer,1,1,"ps");    }

    if ($xskala eq "normal") {
        for ($schritt=$anfangx;$schritt<=$endex;$schritt++) {
            if ($schritt != $schnittx) {
                xmarkierung($layer,$schritt*$xschritt,($schritt*$xschritt),"ps");
            }
            else {
                xmarkierung($layer,$schritt*$xschritt,"","ps");        }
        }
    }
    if ($xskala eq "strichundeins") {
        xmarkierung($layer,1,1,"ps");
        for ($schritt=$anfangx;$schritt<$endex;$schritt++) {
            xmarkierung($layer,$schritt*$xschritt,"","ps");   }
    }

    # Achsenbeschriftung
    if ($xtext eq "\$x\$") {
        xmarkierungohnelinie($layer,$xmax,$xtext,"latex");   }

    else {
        if ($xtext ne "") {
            xmarkierungohnelinie($layer,$xmax,$xtext,"latex");   }

    }
    closecompound;

}

# y-Achse wird gezeichnet
sub yachse {
    my $layer=shift(@_);
    my $ymin=shift(@_);
    my $ymax=shift(@_);
    my $yschritt=shift(@_);
    my $yskala=shift(@_);
    my $ytext=shift(@_);

    # Dicke und Länge des Pfeils wird definiert
    my $dicke=12;
    my $laenge=15;
```

```
    # der Achsenpfeil wird gezeichnet
    openarrow($layer,SOLID,ZEICHENDICKE+1,2,1,4,$dicke,$laenge,0,2);
    plinepoint($schnittx*$scalex,$ymin*$scaley);
    plinepoint($schnittx*$scalex,$ymax*$scaley);

    # Striche müssen innerhalb des Pfeils liegen
    my $yminachse=aufrunden($ymin+0.25*$yschritt,$yschritt);
    my $ymaxachse=abrunden($ymax-0.25*$yschritt,$yschritt);

    # Anfangs und Endwerte für die Schleife werden gesetzt
    my $anfangy=$yminachse/$yschritt;
    my $endey=$ymaxachse/$yschritt;

    my $schritt;

    if ($yskala eq "nurstriche") {
        for ($schritt=$anfangy;$schritt<=$endey;$schritt++) {
            ymarkierung($layer,$schritt*$yschritt,"","ps");     }
    }

    if ($yskala eq "nureins") {
        ymarkierung($layer,1,1,"ps");       }

    if ($yskala eq "normal") {
        for ($schritt=$anfangy;$schritt<=$endey;$schritt++) {
            if ($schritt != $schnitty) {
                ymarkierung($layer,$schritt*$yschritt,$schritt*$yschritt,"ps");
            } else {
                ymarkierung($layer,$schritt*$yschritt,"","ps");
            }
        }
    }
    if ($yskala eq "strichundeins") {
        ymarkierung($layer,1,1,"ps");
        for ($schritt=$anfangy;$schritt<=$endey;$schritt++) {
            ymarkierung($layer,$schritt*$yschritt,"","ps");     }
    }

    # Achsenbeschriftung
    if ($ytext eq "") { }
    if ($ytext eq "\$y\$") {
        ymarkierungohnelinie($layer,$ymax,$ytext,"latex");

} else {
        ymarkierungohnelinie($layer,$ymax,$ytext,"latex");      }

}

sub xachserad {
    my $layer=shift(@_);
    my $xmin=shift(@_);
    my $xmax=shift(@_);
    my $xschritt=shift(@_);

    my $xskala="";
    xachse($layer,deg2rad($xmin),deg2rad($xmax),deg2rad($xschritt),$xskala,"");

    my $xminachse=aufrunden($xmin+0.25*$xschritt,$xschritt);
    my $xmaxachse=abrunden($xmax-0.25*$xschritt,$xschritt);

    # Anfangs und Endwerte für die Schleife werden gesetzt
    my $anfangx=$xminachse/$xschritt;
    my $endex=$xmaxachse/$xschritt;

    my $schritt;
    for ($schritt=deg2rad($xmin);$schritt<deg2rad($xmax);$schritt+=deg2rad($xschritt)) {
        if ($schritt !=$schnittx) {
            xmarkierung($layer,$schritt,"\$-2\\\\pi\$","latex")
                if (rad2deg($schritt)==-360);
            xmarkierung($layer,$schritt,"\$-\\\\frac{7}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-315);
            xmarkierung($layer,$schritt,"\$-\\\\frac{3}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==-270);
            xmarkierung($layer,$schritt,"\$-\\\\frac{5}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-225);
            xmarkierung($layer,$schritt,"\$-\\\\pi\$","latex")
                if (rad2deg($schritt)==-180);
            xmarkierung($layer,$schritt,"\$-\\\\frac{3}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-135);
            xmarkierung($layer,$schritt,"\$-\\\\frac{1}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==-90);
            xmarkierung($layer,$schritt,"\$-\\\\frac{1}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-45);

            xmarkierung($layer,$schritt,"\$2\\\\pi\$","latex")
                if (rad2deg($schritt)==360);
            xmarkierung($layer,$schritt,"\$\\\\frac{7}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==315);
            xmarkierung($layer,$schritt,"\$\\\\frac{3}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==270);
```

```
            xmarkierung($layer,$schritt,"\$\\\\frac{5}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==225);
            xmarkierung($layer,$schritt,"\$\\\\pi\$","latex")
                if (rad2deg($schritt)==180);
            xmarkierung($layer,$schritt,"\$\\\\frac{3}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==135);
            xmarkierung($layer,$schritt,"\$\\\\frac{1}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==90);
            xmarkierung($layer,$schritt,"\$\\\\frac{1}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==45);

        }
    }
}

sub yachserad {
    my $layer=shift(@_);
    my $ymin=shift(@_);
    my $ymax=shift(@_);
    my $yschritt=shift(@_);

    my $yskala="";
    yachse($layer,deg2rad($ymin),deg2rad($ymax),deg2rad($yschritt),$yskala,"");

    my $yminachse=aufrunden($ymin+0.25*$yschritt,$yschritt);
    my $ymaxachse=abrunden($ymax-0.25*$yschritt,$yschritt);

    # Anfangs und Endwerte für die Schleife werden gesetzt
    my $anfangy=$yminachse/$yschritt;
    my $endey=$ymaxachse/$yschritt;

    my $schritt;
    for ($schritt=deg2rad($ymin);$schritt<deg2rad($ymax);$schritt+=deg2rad($yschritt)) {
        if ($schritt !=$schnitty) {
            ymarkierung($layer,$schritt,"\$-2\\\\pi\$","latex")
                if (rad2deg($schritt)==-360);
            ymarkierung($layer,$schritt,"\$-\\\\frac{7}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-315);
            ymarkierung($layer,$schritt,"\$-\\\\frac{3}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==-270);
            ymarkierung($layer,$schritt,"\$-\\\\frac{5}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-225);
            ymarkierung($layer,$schritt,"\$-\\\\pi\$","latex")
                if (rad2deg($schritt)==-180);
            ymarkierung($layer,$schritt,"\$-\\\\frac{3}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-135);
            ymarkierung($layer,$schritt,"\$-\\\\frac{1}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==-90);
            ymarkierung($layer,$schritt,"\$-\\\\frac{1}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==-45);

            ymarkierung($layer,$schritt,"\$2\\\\pi\$","latex")
                if (rad2deg($schritt)==360);
            ymarkierung($layer,$schritt,"\$\\\\frac{7}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==315);
            ymarkierung($layer,$schritt,"\$\\\\frac{3}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==270);
            ymarkierung($layer,$schritt,"\$\\\\frac{5}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==225);
            ymarkierung($layer,$schritt,"\$\\\\pi\$","latex")
                if (rad2deg($schritt)==180);
            ymarkierung($layer,$schritt,"\$\\\\frac{3}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==135);
            ymarkierung($layer,$schritt,"\$\\\\frac{1}{2}\\\\pi\$","latex")
                if (rad2deg($schritt)==90);
            ymarkierung($layer,$schritt,"\$\\\\frac{1}{4}\\\\pi\$","latex")
                if (rad2deg($schritt)==45);
        }
    }
}

######################################################################
########## Markierungen an den Achsen werden gezeichnet und
########## Beschriftung eines einzelnen Punktes
######################################################################

sub xmarkierung {
    my $layer=shift(@_);
    my $xwert=shift(@_);
    my $text=shift(@_);
    my $font=shift(@_);

    openpline($layer,SOLID,ZEICHENDICKE+1,0,2);
    plinepoint($xwert*$scalex,(-0.075)+$schnitty*$scaley);
    plinepoint($xwert*$scalex,0.075+$schnitty*$scaley);
    if ($text ne "") {
        if ($font eq "ps") {
            openpstext($layer,0,NARROWHELVETICABOLD, 12, 0, ALIGNCENTER);
        }
        else {
```

```
            opentext($layer,0,SANSSERIF, 10, 0, ALIGNCENTER);
        }
        ptext($xwert*$scalex,-0.6+$schnitty*$scaley,$text);
    }

}

sub xmarkierungohnelinie {
    my $layer=shift(@_);
    my $xwert=shift(@_);
    my $text=shift(@_);
    my $font=shift(@_);

    if ($text ne "") {
        if ($font eq "ps") {
            openpstext($layer,0,NARROWHELVETICABOLD, 12, 0, ALIGNCENTER);
        }
        else {
            opentext($layer,0,SANSSERIF, 10, 0, ALIGNCENTER);
        }
        ptext($xwert*$scalex,-0.6+$schnitty*$scaley,$text);
    }

}

sub ymarkierung {
    my $layer=shift(@_);
    my $ywert=shift(@_);
    my $text=shift(@_);
    my $font=shift(@_);

    openpline($layer,SOLID,LINIENDICKE,0,2);
    plinepoint((-0.075)+$schnittx*$scalex,$ywert*$scaley);
    plinepoint(0.075+$schnittx*$scalex,$ywert*$scaley);
    if ($text ne "") {
        if ($font eq "ps") {
            openpstext($layer,0,NARROWHELVETICABOLD, 12, 0, ALIGNRIGHT);
        }
        else {
            opentext($layer,0,SANSSERIF, 10, 0, ALIGNRIGHT);
        }
        ptext(-0.02*$scalex+$schnittx*$scalex,($ywert)*$scaley,$text);
    }
}

sub ymarkierungohnelinie {
    my $layer=shift(@_);
    my $ywert=shift(@_);
    my $text=shift(@_);
    my $font=shift(@_);

    if ($text ne "") {
        if ($font eq "ps") {
            openpstext($layer,0,NARROWHELVETICABOLD, 12, 0, ALIGNRIGHT);
        }
        else {
            opentext($layer,0,SANSSERIF, 10, 0, ALIGNRIGHT);
        }
        ptext(-0.02*$scalex+$schnittx*$scalex,($ywert)*$scaley,$text);
    }
}

# Eine (x;y)-Markierung wird gezeichnet
sub xymarkierung {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xwert=shift(@_);
    my $ywert=shift(@_);
    my $xtext=shift(@_);
    my $ytext=shift(@_);
    my $farbe=shift(@_);
    my $linie=shift(@_);
    my $striche=shift(@_);
    my $linienart=shift(@_);

    xmarkierung($layer,$xwert,$xtext,"ps",0) if ($striche =~ /x/);
    ymarkierung($layer,$ywert,$ytext,"ps",0) if ($striche =~ /y/);

    kreuzchen($layer,$xwert*$scalex,$ywert*$scaley,$farbe);

    if ($linie =~ /x/) {
        openpline($layer,$linienart,LINIENDICKE,$farbe,2);
        plinepoint($xwert*$scalex,0*$scaley);
        plinepoint($xwert*$scalex,$ywert*$scaley);
    }
    if ($linie =~ /y/) {
        openpline($layer,$linienart,LINIENDICKE,$farbe,2);
        plinepoint($xwert*$scalex,$ywert*$scaley);
        plinepoint(0*$scalex,$ywert*$scaley);
    }
```

```
}

# Mit der Beschrftungsfunktion kann man Funktionen, etc. beschriften
sub beschriftung {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $xwert=shift(@_);
    my $ywert=shift(@_);
    my $winkel=shift(@_);
    my $ausrichtung=shift(@_);
    my $text=shift(@_);
    my $font=shift(@_);

    my $hilfausrichtung;

    if ($ausrichtung eq "l") {
        $hilfausrichtung=ALIGNLEFT; }
    if ($ausrichtung eq "c") {
        $hilfausrichtung=ALIGNCENTER; }
    if ($ausrichtung eq "r") {
        $hilfausrichtung=ALIGNRIGHT; }

    if ($font eq "ps") {
        openpstext($layer,$farbe,NARROWHELVETICABOLD, 12, $winkel, $hilfausrichtung);
    }    else {
        opentext($layer,$farbe,SANSSERIF, 14, $winkel, $hilfausrichtung);               }
    ptext($xwert*$scalex,$ywert*$scaley,$text);

}

# zwei Punkte werden gezeichnet und nach Bedarf verbunden
sub plotpunkte {
    my $layer=shift(@_);
    my $x1=shift(@_);
    my $y1=shift(@_);
    my $x2=shift(@_);
    my $y2=shift(@_);
    my $farbe=shift(@_);
    my $verbunden=shift(@_);

    my $xtext="";

    if ($verbunden eq "jamitk") {
        openpline($layer,SOLID,LINIENDICKE,$farbe,2);
        plinepoint($x1*$scalex,$y1*$scaley);
        plinepoint($x2*$scalex,$y2*$scaley);
        kreuzchen($layer,$x1*$scalex,$y1*$scaley,$farbe);
        kreuzchen($layer,$x2*$scalex,$y2*$scaley,$farbe);
    }

    if ($verbunden eq "jaohnek") {
        openpline($layer,SOLID,LINIENDICKE,$farbe,2);
        plinepoint($x1*$scalex,$y1*$scaley);
        plinepoint($x2*$scalex,$y2*$scaley);
    }

    if ($verbunden eq "nein") {
        kreuzchen($layer,$x1*$scalex,$y1*$scaley,$farbe);
        kreuzchen($layer,$x2*$scalex,$y2*$scaley,$farbe);
    }

}

sub plotkreis {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $xstart=shift(@_);
    my $ystart=shift(@_);

    my $radius=20;
    my $pdicke=20;
    my $plaenge=25;

    circle(50,0, 3,$xstart*$scalex,$ystart*$scaley,
           $radius,$farbe);

}

#####################################################################
########## Zeichenfunktion für Funktionen
#####################################################################

# "normale" Zeichenfunktion für Funktionen
 sub plot {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
```

```
    my $farbe=shift(@_);

    my (@xpunkte,@ypunkte,$schritt,$i);
    my $schrittanzahl=100;
    my $punkte=0;

    opencompound(abrunden($xmin,0.1),aufrunden($ymax,0.1),
                 aufrunden($xmax,0.1),abrunden($ymin,0.1));

    # Schleife zum zeichnen der Funktion
    for ($schritt=0;$schritt<$schrittanzahl;$schritt++)    {
        $x=$xmin+($xmax-$xmin)*($schritt/($schrittanzahl-1));
        $y=calcfunk($func,"x",$x);

        # liegen die Funktionswerte innerhalb von ymin und ymax?
        if (($y>=$ymin) && ($y<=$ymax)) {
            $xpunkte[$punkte]=$x;
            $ypunkte[$punkte]=$y;
            $punkte++;
        } else {

            # wenn nicht wird das bisherige Array ausgelesen und gezeichnet
            if ($punkte>1) {
                openpline($layer,SOLID,LINIENDICKE,$farbe,$punkte);
                for ($i=0; $i<$punkte; $i++) {
                    plinepoint($xpunkte[$i]*$scalex,$ypunkte[$i]*$scaley);
                }
            }
            $punkte=0;
        }
    }

    # Restarray wird ausgelesen
    if ($punkte>1) {
        openpline($layer,SOLID,LINIENDICKE,$farbe,$punkte);
        for ($i=0; $i<$punkte; $i++) {
            plinepoint($xpunkte[$i]*$scalex,$ypunkte[$i]*$scaley);
        }
    }
    $punkte=0;

    closecompound;
}

# Zeichnen einer Funktion aus vorgegebenen Punkten

sub plotanzahlpunkte {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $anzahlpunkte=shift(@_);
    openpline($layer,SOLID,LINIENDICKE,$farbe,$anzahlpunkte);
}

sub plotxy {
    my $xwert=shift(@_);
    my $ywert=shift(@_);
    plinepoint($xwert*$scalex,$ywert*$scaley);
}

sub plotxypunkt {
    my $layer=shift(@_);
    my $farbe=shift(@_);
    my $x=shift(@_);
    my $y=shift(@_);

    my $radius=20;
    my $pdicke=20;
    my $plaenge=25;

    my $groesse=0.1;

    $groesse=$groesse/2;

    if ($scalex>1) {
        $groesse=$groesse/($scalex/2);
    }

    if ($scaley>1) {
        $groesse=$groesse/($scaley/2);

    }

    my $xlinks=$x-$groesse*$scalex;
    my $xrechts=$x+$groesse*$scalex;
    my $yunten=$y-$groesse*$scaley;
    my $yoben=$y+$groesse*$scaley;

    opencompound(abrunden($xlinks,0.1),aufrunden($yoben,0.1),
                 aufrunden($xrechts,0.1),abrunden($yunten,0.1));
```

```
    openpline($layer,SOLID,LINIENDICKE,$farbe,2);
    plinepoint($xlinks,$yunten);
    plinepoint($xrechts,$yoben);
    openpline($layer,SOLID,LINIENDICKE,$farbe,2);
    plinepoint($xlinks,$yoben);
    plinepoint($xrechts,$yunten);

    closecompound;

}

#####################################################################
########## Ableitungsfunktion wird gezeichnet
#####################################################################

# Zeichnen der Ableitungsfunktion
sub plotableitung {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $farbe=shift(@_);

    my $schrittanzahl=100;
    my (@xpunkte,@ypunkte,$schritt,$i);
    my $punkte=0;

    opencompound(abrunden($xmin,0.1),aufrunden($ymax,0.1),
                 aufrunden($xmax,0.1),abrunden($ymin,0.1));
    # Schleife zum zeichnen der Funktion
    for ($schritt=0;$schritt<$schrittanzahl;$schritt++)    {
        $x=$xmin+($xmax-$xmin)*($schritt/($schrittanzahl-1));
        $y=calcableitung($func,$x);

        # liegen die Funktionswerte innerhalb von ymin und ymax?
        if (($y>=$ymin) && ($y<=$ymax)) {
            $xpunkte[$punkte]=$x;
            $ypunkte[$punkte]=$y;
            $punkte++;
        } else {
            # wenn nicht wird das bisherige Array ausgelesen und gezeichnet
            if ($punkte>1) {
                openpline($layer,SOLID,LINIENDICKE,$farbe,$punkte);
                for ($i=0; $i<$punkte; $i++) {
                    plinepoint($xpunkte[$i]*$scalex,$ypunkte[$i]*$scaley);
                }
            }
            $punkte=0;
        }
    }

    # Restarray wird ausgelesen
    if ($punkte>1) {
        openpline($layer,SOLID,LINIENDICKE,$farbe,$punkte);
        for ($i=0; $i<$punkte; $i++) {
            plinepoint($xpunkte[$i]*$scalex,$ypunkte[$i]*$scaley);
        }
    }
    $punkte=0;

    closecompound;
}

#####################################################################
########## Alles für die Integralrechnung
#####################################################################

# Zeichnen der Fläche unter einer Funktion
sub plotintegral {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $farbe=shift(@_);
    my $gefuellt=shift(@_);
    my $fuellfarbe=shift(@_);

    einschluss($layer, $func, "0", $xmin, $ymin, $xmax, $ymax, "keine",
               $farbe, $gefuellt, $fuellfarbe);
}

sub plotstammfunktion {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
```

```
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $farbe=shift(@_);
    my $verschiebungswert=shift(@_);

    my $schrittanzahl=10000;
    my (@xpunkte,@ypunkte,$schritt,$i,@yintegral,$x,$y,$integral,
        $breite,$element);
    my $punkte=0;

    $integral=0;
    $nullwert=0;
    $breite=($xmax-$xmin)/($schrittanzahl);
    for ($schritt=0;$schritt<$schrittanzahl+1;$schritt++) {
        $x=$xmin+($xmax-$xmin)*($schritt/$schrittanzahl);
        $y=calcfunk($func,"x",$x);
        $element=($breite*$y);
        $integral+=$element;
        $nullwert=$integral if ($x<=0);
        $yintegral[$schritt]=$integral;
    }

    $nullwert-=$verschiebungswert;

    opencompound(abrunden($xmin,0.1),aufrunden($ymax,0.1),
                 aufrunden($xmax,0.1),abrunden($ymin,0.1));
    # Schleife zum zeichnen der Funktion
    for ($schritt=5;$schritt<$schrittanzahl;$schritt+=100) {
        $x=$xmin+($xmax-$xmin)*($schritt/($schrittanzahl-1));
        $y=$yintegral[$schritt]-$nullwert;

        # liegen die Funktionswerte innerhalb von ymin und ymax?
        if (($y>=$ymin) && ($y<=$ymax)) {
            $xpunkte[$punkte]=$x;
            $ypunkte[$punkte]=$y;
            $punkte++;
        } else {
            # wenn nicht wird das bisherige Array ausgelesen und gezeichnet
            if ($punkte>1) {
                openpline($layer,SOLID,LINIENDICKE,$farbe,$punkte);
                for ($i=0; $i<$punkte; $i++) {
                    plinepoint($xpunkte[$i]*$scalex,$ypunkte[$i]*$scaley);
                }
            }
            $punkte=0;
        }
    }

    # Restarray wird ausgelesen
    if ($punkte>1) {
        openpline($layer,SOLID,LINIENDICKE,$farbe,$punkte);
        for ($i=0; $i<$punkte; $i++) {
            plinepoint($xpunkte[$i]*$scalex,$ypunkte[$i]*$scaley);
        }
    }
    $punkte=0;

    closecompound;

}

# Die Obersumme und Untersumme werden gezeichnet

sub plotoberuntersumme {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $farbe=shift(@_);
    my $schachtelung=shift(@_);
    my $berechnung=shift(@_);
    my $summen=shift(@_);

    my ($yober,$yunter,$schritt,$xlinks,$xrechts,$flaecheober,$flaecheunter);

    my $bereichxmin=calcxmin($func,$xmin,$xmax,$ymin,$ymax);
    my $bereichxmax=calcxmax($func,$xmin,$xmax,$ymin,$ymax);
    my $calcober=0;
    my $calcunter=0;
    my $farbeober=$farbe;
    my $farbeunter=$farbe;
    my $gefuelltober=20;
    my $fuellfarbeober=31;
    my $gefuelltunter=20;
    my $fuellfarbeunter=35;

    opencompound(abrunden($bereichxmin,0.1),aufrunden($ymax,0.1),
                 aufrunden($bereichxmax,0.1),abrunden($ymin,0.1));
```

```
    for ($schritt=0;$schritt<$schachtelung;$schritt++) {
        $xlinks=$bereichxmin+($bereichxmax-$bereichxmin)*($schritt/($schachtelung));
        $xrechts=$bereichxmin+($bereichxmax-$bereichxmin)*(($schritt+1)/($schachtelung));
        $yober=calcymax($func,$xlinks,$xrechts);
        $yunter=calcymin($func,$xlinks,$xrechts);

        if (abs($yober)>abs($yunter)) {

            rechteck($layer+2,$xlinks*$scalex,0*$scaley,$xrechts*$scalex,$yober*$scaley,
                     SOLID,LINIENDICKE,$farbeober,$gefuelltober,$fuellfarbeober)
                if (($summen eq "ober") || ($summen eq "beide"));

            rechteck($layer+1,$xlinks*$scalex,0*$scaley,$xrechts*$scalex,$yunter*$scaley,
                     SOLID,LINIENDICKE,$farbeunter,$gefuelltunter,$fuellfarbeunter)
                if (($summen eq "unter") || ($summen eq "beide"));
        }
        else {
            rechteck($layer+1,$xlinks*$scalex,0*$scaley,$xrechts*$scalex,$yober*$scaley,
                     SOLID,LINIENDICKE,$farbeober,$gefuelltober,$fuellfarbeober)
                if (($summen eq "ober") || ($summen eq "beide"));

            rechteck($layer+2,$xlinks*$scalex,0*$scaley,$xrechts*$scalex,$yunter*$scaley,
                     SOLID,LINIENDICKE,$farbeunter,$gefuelltunter,$fuellfarbeunter)
                if (($summen eq "unter") || ($summen eq "beide"));
        }

        $flaecheober=($xrechts-$xlinks)*$yober;
        $calcober+=$flaecheober;
        $flaecheunter=($xrechts-$xlinks)*$yunter;
        $calcunter+=$flaecheunter;

    }

    if ($berechnung eq "nein") {}
    if ($berechnung eq "ja") {
        if (($summen eq "ober") || ($summen eq "beide")) {
            opentext(90,0,SANSSERIF, 10, 0, ALIGNLEFT);
            ptext(($bereichxmax+0.25)*$scalex,($ymax/2)*$scaley,
                  "\$O_{".sprintf("%i",$schachtelung)."}= ".sprintf("%.2f",
                                                                    $calcober)."\~FE \$");
        }
        if (($summen eq "unter") || ($summen eq "beide")) {
            opentext(90,0,SANSSERIF, 10, 0, 0, ALIGNLEFT);
            ptext(($bereichxmax+0.25)*$scalex,($ymax/2-0.5)*$scaley,
                  "\$U_{".sprintf("%i",$schachtelung)."}= ".sprintf("%.2f",
                                                                    $calcunter)."\~FE \$");
        }
    }

    closecompound;
}

# der Flächeninhalt zwischen zwei Funktionen kann gezeichnet werden
sub einschluss {
    my $layer=shift(@_);
    my $func1=shift(@_);
    my $func2=shift(@_);
    my $xmin=shift(@_);
    my $ymin=shift(@_);
    my $xmax=shift(@_);
    my $ymax=shift(@_);
    my $schnittpunkte=shift(@_);
    my $farbe=shift(@_);
    my $gefuellt=shift(@_);
    my $fuellfarbe=shift(@_);

    my (@xpunkte, @ypunkte, $schritt,$i, $y1,$y2, $x);
    my $punkte=0;
    my $schrittanzahl=500;
    my $bereichxmin=$xmin;
    my $bereichxmax=$xmax;

    if ($schnittpunkte eq "links") {
        $bereichxmin=calcschnittlinks($func1,$func2,$xmin,$xmax);
    }
    if ($schnittpunkte eq "rechts") {
        $bereichxmax=calcschnittrechts($func1,$func2,$xmin,$xmax);
    }
    if ($schnittpunkte eq "beide") {
        $bereichxmin=calcschnittlinks($func1,$func2,$xmin,$xmax);
        $bereichxmax=calcschnittrechts($func1,$func2,$xmin,$xmax);
    }

    $bereichxmax=$xmax
        if ((abs($bereichxmin-$bereichxmax)<0.001) &&
            ($schnittpunkte eq "links"));

    $bereichxmin=$xmin
        if ((abs($bereichxmin-$bereichxmax)<0.001) &&
            ($schnittpunkte eq "rechts"));
```

```
    # Schleife zum zeichnen der Funktion
    for ($schritt=0;$schritt<$schrittanzahl;$schritt++)    {
        $x=$bereichxmin+($bereichxmax-$bereichxmin)*($schritt/($schrittanzahl-1));
        $y1=calcfunk($func1,"x",$x);
        $y2=calcfunk($func2,"x",$x);

        # liegen die Funktionswerte innerhalb von ymin und ymax?
        if (($y1>=$ymin) && ($y1<=$ymax) && ($y2>=$ymin) && ($y2<=$ymax)) {
            $xpunkte1[$punkte]=$x;
            $ypunkte1[$punkte]=$y1;
            $xpunkte2[$punkte]=$x;
            $ypunkte2[$punkte]=$y2;
            $punkte++;
        } else {
            # wenn nicht wird das bisherige Array ausgelesen und gezeichnet
            if ($punkte>1) {
                openbox($layer,SOLID,LINIENDICKE,$farbe,$gefuellt,
                        $fuellfarbe,($punkte-1)*2);
                for ($i=0; $i<$punkte-1; $i++) {
                    plinepoint($xpunkte1[$i]*$scalex,$ypunkte1[$i]*$scaley);
                }
                for ($i=$punkte-1; $i>0; $i--) {
                    plinepoint($xpunkte2[$i]*$scalex,$ypunkte2[$i]*$scaley);
                }

            }
            $punkte=0;
        }
    }

#     Restarray wird ausgelesen
    if ($punkte>1) {
        openbox($layer,SOLID,LINIENDICKE,$farbe,$gefuellt,
                $fuellfarbe,($punkte-1)*2);
        for ($i=0; $i<$punkte-1; $i++) {
            plinepoint($xpunkte1[$i]*$scalex,$ypunkte1[$i]*$scaley);
        }
        for ($i=$punkte-1; $i>0; $i--) {
            plinepoint($xpunkte2[$i]*$scalex,$ypunkte2[$i]*$scaley);
        }

    }

    $punkte=0;

}

#####################################################################
########## Tangenten, Steigungsdreiecke etc, werden gezeichnet
#####################################################################

sub plottangente {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xwert=shift(@_);
    my $breite=shift(@_);
    my $farbe=shift(@_);
    my $markierung=shift(@_);

    my $ywert=calcfunk($func,"x",$xwert);
    my $xmin=$xwert-$breite;
    my $xmax=$xwert+$breite;
    my $ymin=calctangente($func,$xmin,$xwert);
    my $ymax=calctangente($func,$xmax,$xwert);

    opencompound(abrunden($xmin,0.1),aufrunden($ymax,0.1),
                 aufrunden($xmax,0.1),abrunden($ymin,0.1));
    openpline($layer,SOLID,LINIENDICKE,$farbe,2);
    plinepoint($xmin*$scalex,$ymin*$scaley);
    plinepoint($xmax*$scalex,$ymax*$scaley);

    if ($markierung eq "ja") {
        xymarkierung($layer,$func,$xwert,$ywert,"","",$farbe,"x","x",SOLID);
    }
    closecompound;
}

sub plottangentedreieck {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xwert=shift(@_);
    my $breite=shift(@_);
    my $farbe=shift(@_);
    my $markierung=shift(@_);

    my $ywert=calcfunk($func,"x",$xwert);
    my $xmin=$xwert-$breite;
    my $xmax=$xwert+$breite;
    my $ymin=calctangente($func,$xmin,$xwert);
```

```
    my $ymax=calctangente($func,$xmax,$xwert);
    my $yabl=($ymax-$ymin)/($xmax-$xmin);

    opencompound(abrunden($xmin,0.1),aufrunden($ymax,0.1),
                 aufrunden($xmax,0.1),abrunden($ymin,0.1));
    if ($ymin>$ymax) {
        openbox($layer,SOLID,LINIENDICKE,$farbe,40,7,4);
        plinepoint($xmin*$scalex,$ymin*$scaley);
        plinepoint($xmin*$scalex,$ymax*$scaley);
        plinepoint($xmax*$scalex,$ymax*$scaley);
        plinepoint($xmin*$scalex,$ymin*$scaley);

    } else {

        openbox($layer,SOLID,LINIENDICKE,$farbe,40,7,4);
        plinepoint($xmin*$scalex,$ymin*$scaley);
        plinepoint($xmax*$scalex,$ymin*$scaley);
        plinepoint($xmax*$scalex,$ymax*$scaley);
        plinepoint($xmin*$scalex,$ymin*$scaley);
    }

    if ($markierung eq "wert") {
        xymarkierung($layer-1,$func,$xwert,$ywert,"","",$farbe,"x","x",DASHED);
    }
    if ($markierung eq "abl") {
        xymarkierung($layer-1,$func,$xwert,$yabl,"","",$farbe,"x","x",DASHED);
    }
    closecompound;
}

sub plotsteigungsdreieck {
    my $layer=shift(@_);
    my $func=shift(@_);
    my $xwert=shift(@_);
    my $breite=shift(@_);
    my $farbe=shift(@_);
    my $markierung=shift(@_);

    my $ywert=calcfunk($func,"x",$xwert);
    my $xmin=$xwert-$breite;
    my $xmax=$xwert+$breite;
    my $ymin=calcfunk($func,"x",$xmin);
    my $ymax=calcfunk($func,"x",$xmax);
    my $yabl=($ymax-$ymin)/($xmax-$xmin);

    opencompound(abrunden($xmin,0.1),aufrunden($ymax,0.1),
                 aufrunden($xmax,0.1),abrunden($ymin,0.1));
    if ($ymin>$ymax) {
        openbox($layer,SOLID,LINIENDICKE,$farbe,40,7,4);
        plinepoint($xmin*$scalex,$ymin*$scaley);
        plinepoint($xmin*$scalex,$ymax*$scaley);
        plinepoint($xmax*$scalex,$ymax*$scaley);
        plinepoint($xmin*$scalex,$ymin*$scaley);

    } else {

        openbox($layer,SOLID,LINIENDICKE,$farbe,40,7,4);
        plinepoint($xmin*$scalex,$ymin*$scaley);
        plinepoint($xmax*$scalex,$ymin*$scaley);
        plinepoint($xmax*$scalex,$ymax*$scaley);
        plinepoint($xmin*$scalex,$ymin*$scaley);
    }

    if ($markierung eq "wert") {
        xymarkierung($layer-1,$func,$xwert,$ywert,"","",$farbe,"x","x",DASHED);
    }
    if ($markierung eq "abl") {
        xymarkierung($layer-1,$func,$xwert,$yabl,"","",$farbe,"x","x",DASHED);
    }
    closecompound;
}

#####################################################################
########## Säulen werden gezeichnet
#####################################################################

sub plotsaeule {
    my $layer=shift(@_);
    my $xwert=shift(@_);
    my $ywert=shift(@_);
    my $farbe=shift(@_);
    my $gefuellt=shift(@_);
    my $fuellfarbe=shift(@_);
    my $breite=shift(@_);

    openbox($layer,SOLID,LINIENDICKE,$farbe,$gefuellt,$fuellfarbe,5);
    plinepoint(($xwert-$breite/2)*$scalex,0*$scaley);
    plinepoint(($xwert+$breite/2)*$scalex,0*$scaley);
    plinepoint(($xwert+$breite/2)*$scalex,$ywert*$scaley);
    plinepoint(($xwert-$breite/2)*$scalex,$ywert*$scaley);
```

```
    plinepoint(($xwert-$breite/2)*$scalex,0*$scaley);

}

#####################################################################
########## Vektoren werden gezeichnet
#####################################################################

# Vektor wird durch seinen Startpunkt, seine Länge und seinen Winkel definiert

sub plotvektorpolar {
    my $layer=shift(@_);
    my $xstart=shift(@_);
    my $ystart=shift(@_);
    my $laenge=shift(@_);
    my $winkel=shift(@_);
    my $farbe=shift(@_);
    my $gestrichelt=shift(@_);
    my $radius=20;

    my $pdicke=20;
    my $plaenge=25;
    my $xende=calcendpunktx($xstart,$laenge,$winkel);
    my $yende=calcendpunkty($ystart,$laenge,$winkel);

    if (($xstart != $xende) &&($laenge != $winkel)) {
        if ($gestrichelt==0) {
            openarrow($layer,SOLID,LINIENDICKE,2,1,4,$pdicke,
                      $plaenge,$farbe,2);
            plinepoint($xstart*$scalex,$ystart*$scaley);
            plinepoint($xende*$scalex,$yende*$scaley);
            circle(50,0, 3,$xstart*$scalex,$ystart*$scaley,
                   $radius,$farbe);
        } else {
            openpline($layer,$gestrichelt,LINIENDICKE,0,2);
            plinepoint($xstart*$scalex,$ystart*$scaley);
            plinepoint($xende*$scalex,$yende*$scaley);

        }
    }
}

sub plotvektor {
    my $layer=shift(@_);
    my $xstart=shift(@_);
    my $ystart=shift(@_);
    my $xende=shift(@_);
    my $yende=shift(@_);
    my $gestrichelt=shift(@_);

    my $radius=20;
    my $pdicke=20;
    my $plaenge=25;

    if (($xstart != $xende) &&($ystart != $yende)) {
        if ($gestrichelt==0) {
            openarrow($layer,SOLID,LINIENDICKE,2,1,4,$pdicke,
                      $plaenge,$farbe,2);
            plinepoint($xstart*$scalex,$ystart*$scaley);
            plinepoint($xende*$scalex,$yende*$scaley);
            circle(50,0, 3,$xstart*$scalex,$ystart*$scaley,
                   $radius,$farbe);
        } else {
            openpline($layer,$gestrichelt,LINIENDICKE,0,2);
            plinepoint($xstart*$scalex,$ystart*$scaley);
            plinepoint($xende*$scalex,$yende*$scaley);

        }
    }
}

sub plotvektorparallel {
    my $layer=shift(@_);
    my $xstart1=shift(@_);
    my $ystart1=shift(@_);
    my $laenge1=shift(@_);
    my $winkel1=shift(@_);
    my $xstart2=shift(@_);
    my $ystart2=shift(@_);
    my $laenge2=shift(@_);
    my $winkel2=shift(@_);
    my $farbe=shift(@_);
    my $ersatzvektor=shift(@_);

    my ($x1,$y1,$x2,$y2);

    $x1=$xstart1;
    $y1=$ystart1;
    $x2=$xstart2;
    $y2=$ystart2;
```

```
    if ($ersatzvektor eq "parallel") {
        $x2=$x1;
        $y2=$y1;
    }

    my $x12=calcendpunktx($x1,$laenge1,$winkel1);
    my $y12=calcendpunkty($y1,$laenge1,$winkel1);

    my $x22=calcendpunktx($x2,$laenge2,$winkel2);
    my $y22=calcendpunkty($y2,$laenge2,$winkel2);

    my $l3=calcsumlaenge($x1,$y1,$laenge1,$winkel1,
                         $laenge2,$winkel2);
    my $w3=calcneuerwinkel($x1,$y1,$laenge1,$winkel1,
                           $laenge2,$winkel2);

     if ($ersatzvektor eq "ersatz") {
        plotvektorpolar($layer,$x1,$y1,$l3,$w3,4,0);
     }
     if ($ersatzvektor eq "parallel") {
          plotvektorpolar($layer,$x1,$y1,$l3,$w3,4,0);
          plotvektorpolar($layer,$x12,$y12,$laenge2,
                          $winkel2,$farbe,1);
          plotvektorpolar($layer,$x22,$y22,$laenge1,
                          $winkel1,$farbe,1);

     }

    plotvektorpolar($layer,$x1,$y1,$laenge1,$winkel1,$farbe,0);
    plotvektorpolar($layer,$x2,$y2,$laenge2,$winkel2,$farbe,0);

}

sub plotvektorpolarmehrere {
    my $layer=shift(@_);
    my $x1=shift(@_);
    my $y1=shift(@_);
    my $l1=shift(@_);
    my $w1=shift(@_);
    my $l2=shift(@_);
    my $w2=shift(@_);
    my $l3=shift(@_);
    my $w3=shift(@_);
    my $l4=shift(@_);
    my $w4=shift(@_);
    my $l5=shift(@_);
    my $w5=shift(@_);
    my $l6=shift(@_);
    my $w6=shift(@_);
    my $farbe=shift(@_);
    my $ersatzvektor=shift(@_);

    my ($le1,$we1,$le2,$we2,$le3,$we3,$le4,$we4,$le5,$we5);

    plotvektorpolar($layer,$x1,$y1,$l1,$w1,$farbe,0);
    plotvektorpolar($layer,$x1,$y1,$l2,$w2,$farbe,0);
    plotvektorpolar($layer,$x1,$y1,$l3,$w3,$farbe,0);
    plotvektorpolar($layer,$x1,$y1,$l4,$w4,$farbe,0);
    plotvektorpolar($layer,$x1,$y1,$l5,$w5,$farbe,0);
    plotvektorpolar($layer,$x1,$y1,$l6,$w6,$farbe,0);

    plotvektorparallel($layer,$x1,$y1,$l1,$w1,
                       $x1,$y1,$l2,$w2,$farbe,$ersatzvektor);

        if (($l3 != 0) && ($w3!=0)) {
            $le1=calcsumlaenge($x1,$y1,$l1,$w1,$l2,$w2);
            $we1=calcneuerwinkel($x1,$y1,$l1,$w1,$l2,$w2);
            plotvektorparallel($layer,$x1,$y1,$le1,$we1,
                               $x1,$y1,$l3,$w3,$farbe,$ersatzvektor);
        }

        if (($l4 != 0) && ($w4!=0)) {
            $le1=calcsumlaenge($x1,$y1,$l1,$w1,$l2,$w2);
            $we1=calcneuerwinkel($x1,$y1,$l1,$w1,$l2,$w2);
            $le2=calcsumlaenge($x1,$y1,$le1,$we1,$l3,$w3);
            $we2=calcneuerwinkel($x1,$y1,$le1,$we1,$l3,$w3);
            plotvektorparallel($layer,$x1,$y1,$le2,$we2,
                               $x1,$y1,$l4,$w4,$farbe,$ersatzvektor);

        }
        if (($l5 != 0) && ($w5!=0)) {
            $le1=calcsumlaenge($x1,$y1,$l1,$w1,$l2,$w2);
            $we1=calcneuerwinkel($x1,$y1,$l1,$w1,$l2,$w2);
            $le2=calcsumlaenge($x1,$y1,$le1,$we1,$l3,$w3);
            $we2=calcneuerwinkel($x1,$y1,$le1,$we1,$l3,$w3);
            $le3=calcsumlaenge($x1,$y1,$le2,$we2,$l4,$w4);
            $we3=calcneuerwinkel($x1,$y1,$le2,$we2,$l4,$w4);
            plotvektorparallel($layer,$x1,$y1,$le3,$we3,
                               $x1,$y1,$l5,$w5,$farbe,$ersatzvektor);
```

```
        }

        if (($l6 != 0) && ($w6!=0)) {
            $le1=calcsumlaenge($x1,$y1,$l1,$w1,$l2,$w2);
            $we1=calcneuerwinkel($x1,$y1,$l1,$w1,$l2,$w2);
            $le2=calcsumlaenge($x1,$y1,$le1,$we1,$l3,$w3);
            $we2=calcneuerwinkel($x1,$y1,$le1,$we1,$l3,$w3);
            $le3=calcsumlaenge($x1,$y1,$le2,$we2,$l4,$w4);
            $we3=calcneuerwinkel($x1,$y1,$le2,$we2,$l4,$w4);
            $le4=calcsumlaenge($x1,$y1,$le3,$we3,$l5,$w5);
            $we4=calcneuerwinkel($x1,$y1,$le3,$we3,$l5,$w5);
            plotvektorparallel($layer,$x1,$y1,$le4,$we4,
                               $x1,$y1,$l6,$w6,$farbe,$ersatzvektor);
        }

}

#####################################################################
#####################################################################
########## Einlesen der Befehlsdatei
#####################################################################
#####################################################################

$kommandodateinummer=0;
while (exists $ARGV[$kommandodateinummer]) {

    $kommandodatei=$ARGV[$kommandodateinummer];
    $xfigdatei=$ARGV[$kommandodateinummer];
    $xfigdatei=~s/\.ptxt$//i;
    $xfigdatei=$xfigdatei.".fig";
    $kommandodateinummer++;

    open(KOMMANDO,$kommandodatei) || die $kommandodatei.": $!";

    printf("Erzeuge %s...\n",$xfigdatei);
    openfig($xfigdatei);
    neuefarbe(35,0xff,0xf3,0xb5);
    neugrau(32,80);

    $scalex=1;
    $scaley=1;

    $schnittx=0;
    $schnitty=0;

    while ($zeile = <KOMMANDO>) {
        # Zeilenenden beseitigen, Kommentare und Leerzeilen ignorieren
    $zeile=~ s/[\r\n]//g;
    $zeile=~ s/  +/ /g;
    $zeile=~ s/ +$//g;
    $zeile=~ s/^ +//g;
    next if ($zeile =~ /^\#/);
    next if ($zeile =~ /^$/);

    # Befehl "skalierung"
    if ($zeile =~ /^skalierung( x=([0-9\.]+))?( y=([0-9\.]+))?$/i) {
        $scalex=$2 if (defined $1);
        $scaley=$4 if (defined $3);
        next;
    }

    if ($zeile =~ /^skalierung ([0-9\.]+)/i) {
        $scalex=$scaley=$1;
        next;
    }

    # Befehl "Achsenschnittpunkte"
    if ($zeile =~ /^achsenschnitt( x=([0-9\.]+))?( y=([0-9\.]+))?$/i) {
        $schnittx=$2 if (defined $1);
        $schnitty=$4 if (defined $3);
        next;
    }

    # Befehl "kariert"
    if ($zeile =~ /^kariert( layer=([0-9]+))?( farbe=([0-9]+))? abstandx=([0-9\.]+),([0-9]+),([0-9]+)? abstandy
      =([0-9\.]+),([0-9]+),([0-9]+)? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+),([-0-9\.]+)$/i) {
        $layer=90;
        $layer=$2 if (defined $1);
        $farbe=0;
        $farbe=$4 if (defined $3);
        $abstandx=$5;
        $dickx=$6;
        $ganzdickx=$7;
        $abstandy=$8;
        $dicky=$9;
        $ganzdicky=$10;
        $xmin=$11;
        $ymin=$12;
        $xmax=$13;
```

```
    $ymax=$14;
    kariert($layer,$farbe,$xmin,$ymin,$xmax,$ymax,$abstandx,$dickx,$ganzdickx,$abstandy,$dicky,$ganzdicky);

    next;
}

# Befehl "grid"
if ($zeile =~ /^grid( layer=([0-9]+))?( farbe=([0-9]+))? abstandx=([0-9\.]+),([0-9\.]+),([0-9\.]+)? abstandy
    =([0-9\.]+),([0-9\.]+),([0-9\.]+)? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+),([-0-9\.]+)$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $abstandx=$5;
    $dickx=$6;
    $ganzdickx=$7;
    $abstandy=$8;
    $dicky=$9;
    $ganzdicky=$10;
    $xmin=$11;
    $ymin=$12;
    $xmax=$13;
    $ymax=$14;
    grid($layer,$farbe,$xmin,$ymin,$xmax,$ymax,$abstandx,$dickx,$ganzdickx,$abstandy,$dicky,$ganzdicky);

    next;
}

# Befehl "punkte"
if ($zeile =~ /^punkte( layer=([0-9]+))?( farbe=([0-9]+))? abstand=([0-9\.]+)? bereich=([-0-9\.]+),([-0-9\.]+)
    ,([-0-9\.]+),([-0-9\.]+)$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $abstand=$5;
    $xmin=$6;
    $ymin=$7;
    $xmax=$8;
    $ymax=$9;

    punkte($layer,$farbe,$xmin,$ymin,$xmax,$ymax,$abstand);

    next;
}

# Befehl "matheheft"
if ($zeile =~ /^matheheft( layer=([0-9]+))?( farbe=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+)
    ,([-0-9\.]+)$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xmin=$5;
    $ymin=$6;
    $xmax=$7;
    $ymax=$8;
    matheheft($layer,$farbe,$xmin,$ymin,$xmax,$ymax);
    next;
}

# Befehl "millimeterpapier"
if ($zeile =~ /^millimeterpapier( layer=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+),([-0-9\.]+)$/i)
    {
    $layer=90;
    $layer=$2 if (defined $1);
    $xmin=$3;
    $ymin=$4;
    $xmax=$5;
    $ymax=$6;
    millimeterpapier($layer,$xmin,$ymin,$xmax,$ymax);
    next;
}

# Befehl für das Koordinatensystem
if ($zeile =~/^xachse( layer=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+) skala=(keine|nurstriche|
    nureins|normal|strichundeins) text=\"([^\"]*)\"$/i) {
    $layer=20;
    $layer=$2 if (defined $1);
    $xmin=$3;
    $xmax=$4;
    $xschritt=$5;
    $xskala=$6;
    $xtext=$7;
    xachse($layer, $xmin, $xmax,$xschritt, $xskala, $xtext);
    next;
}

if ($zeile =~/^yachse( layer=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+) skala=(keine|nurstriche|
    nureins|normal|strichundeins) text=\"([^\"]*)\"$/i) {
```

```
    $layer=20;
    $layer=$2 if (defined $1);
    $ymin=$3;
    $ymax=$4;
    $yschritt=$5;
    $yskala=$6;
    $ytext=$7;
    yachse($layer,  $ymin,  $ymax,$yschritt, $yskala, $ytext);
    next;
}

# Befehl für das Koordinatensystem
if ($zeile =~/^xachserad( layer=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+)$/i) {
    $layer=20;
    $layer=$2 if (defined $1);
    $xmin=$3;
    $xmax=$4;
    $xschritt=$5;
    xachserad($layer,$xmin,$xmax,$xschritt);
    next;
}

# Befehl für das Koordinatensystem
if ($zeile =~/^yachserad( layer=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+),([-0-9\.]+)$/i) {
    $layer=20;
    $layer=$2 if (defined $1);
    $ymin=$3;
    $ymax=$4;
    $yschritt=$5;
    yachserad($layer,$ymin,$ymax,$yschritt);
    next;
}

# Zeichnen von Funktionen
if ($zeile =~/^plot( layer=([0-9]+))?( farbe=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+)( grenze=([-0-9\.]+)
     ,([-0-9\.]+))? (.*)$/i) {
    $layer=10;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xmin=$5;
    $xmax=$6;

    if (defined $7) {
        $ymin=$8;
        $ymax=$9;
    }
    $func=$10;

    if (! defined $7) {
        $ymin=calcymin($func,$xmin,$xmax);
        $ymax=calcymax($func,$xmin,$xmax);
    }

    plot($layer, $func, $xmin, $ymin, $xmax, $ymax, $farbe);
    next;
}

if ($zeile =~ /^plotanzahlpunkte( layer=([0-9]+))?( farbe=([0-9]+))? punkte=([0-9]+)$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $punkte=$5;
    plotanzahlpunkte($layer,$farbe,$punkte);
    next;
}

if ($zeile =~ /^plotxy wert=([-0-9\.]+),([-0-9\.]+)$/i) {
    $xwert=$1;
    $ywert=$2;
    plotxy($xwert,$ywert);
    next;
}

if ($zeile =~ /^plotxypunkt( layer=([0-9]+))?( farbe=([0-9]+))? wert=([-0-9\.]+),([-0-9\.]+)$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xwert=$5;
    $ywert=$6;
    plotxypunkt($layer,$farbe,$xwert,$ywert);
    next;
}

# Zeichnen von Ableitungsfunktionen
if ($zeile =~/^plotableitung( layer=([0-9]+))?( farbe=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+)( grenze
     =([-0-9\.]+),([-0-9\.]+))? (.*)$/i) {
    $layer=15;
```

```
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xmin=$5;
    $xmax=$6;

    if (defined $7) {
        $ymin=$8;
        $ymax=$9;
    }
    $func=$10;

    if (! defined $7) {
        $ymin=calcymin($func,$xmin,$xmax);
        $ymax=calcymax($func,$xmin,$xmax);
    }

    plotableitung($layer, $func, $xmin, $ymin, $xmax, $ymax, $farbe);
    next;
}

# Zeichnen von Stammfunktionen
if ($zeile =~/^plotstammfunktion( layer=([0-9]+))?( farbe=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+)( grenze
     =([-0-9\.]+),([-0-9\.]+))?( nullwert=([-0-9\.]+))? (.*)$/i) {
    $layer=15;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xmin=$5;
    $xmax=$6;

    if (defined $7) {
        $ymin=$8;
        $ymax=$9;
    }
    $nullwert=0;
    $nullwert=$11 if (defined $10);

    $func=$12;

    if (! defined $7) {
        $ymin=calcymin($func,$xmin,$xmax);
        $ymax=calcymax($func,$xmin,$xmax);
    }

    plotstammfunktion($layer, $func, $xmin, $ymin, $xmax, $ymax, $farbe,$nullwert);
    next;
}

# Zeichnen von dem Flächeninhalt unter einer Funktion

if ($zeile =~/^plotintegral( layer=([0-9]+))?( farbe=([0-9]+))?( fuellfarbe=([0-9]+),([0-9]+))? bereich
     =([-0-9\.]+),([-0-9\.]+)( grenze=([-0-9\.]+),([-0-9\.]+))? (.*)$/i) {
    $layer=80;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);

    $gefuellt=30;
    $fuellfarbe=27;

    if (defined $5) {
        $gefuellt=$6;
        $fuellfarbe=$7;
    }

    $xmin=$8;
    $xmax=$9;

    if (defined $10) {
        $ymin=$11;
        $ymax=$12;
    }
    $func=$13;

    if (! defined $10) {
        $ymin=calcymin($func,$xmin,$xmax);
        $ymax=calcymax($func,$xmin,$xmax);
    }

    plotintegral($layer, $func, $xmin, $ymin, $xmax, $ymax, $farbe, $gefuellt, $fuellfarbe);
    next;
}

# Ober und Untersumme werden gezeichnet
if ($zeile =~/^plotoberunduntersumme( layer=([0-9]+))?( farbe=([0-9]+))? bereich=([-0-9\.]+),([-0-9\.]+)(
     grenze=([-0-9\.]+),([-0-9\.]+))? schachtelung=([1-9][0-9]*) summen=(ober|unter|beide)?( berechnung=(ja))?
     (.*)$/i) {
    $layer=80;
    $layer=$2 if (defined $1);
```

```
    $farbe=20;
    $farbe=$4 if (defined $3);

    $xmin=$5;
    $xmax=$6;

    if (defined $7) {
        $ymin=$8;
        $ymax=$9;
    }

    $schachtelung=$10;
    $summen=$11;

    $berechnung="nein";
    $berechnung=$13 if (defined $12);

    $func=$14;

    if (! defined $7) {
        $ymin=calcymin($func,$xmin,$xmax);
        $ymax=calcymax($func,$xmin,$xmax);
    }

        plotoberuntersumme($layer, $func, $xmin, $ymin, $xmax, $ymax, $farbe, $schachtelung,$berechnung,$summen
            );

    next;
}

# Zeichnen von Flächen zwischen zwei Funktionen
if ($zeile =~/^einschluss( layer=([0-9]+))?( farbe=([0-9]+))?( fuellfarbe=([0-9]+),([0-9]+))? bereich
     =([-0-9\.]+),([-0-9\.]+)( grenze=([-0-9\.]+),([-0-9\.]+))? schnittpunkte=(links|rechts|beide|keine) ([^
     ]*)( [^ ]*)?$/i) {
    $layer=80;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);

    $gefuellt=30;
    $fuellfarbe=27;

    if (defined $5) {
        $gefuellt=$6;
        $fuellfarbe=$7;
    }

    $xmin=$8;
    $xmax=$9;

    if (defined $10) {
        $ymin=$11;
        $ymax=$12;
    }

    $schnittpunkte=$13;
    $func1=$14;
    $func2="0";
    $func2=$15 if (defined $15);

    if (! defined $10) {
        $ymin=calcfymin($func1,$func2,$xmin,$xmax);
        $ymax=calcfymax($func1,$func2,$xmin,$xmax);
    }

    einschluss($layer, $func1, $func2, $xmin, $ymin, $xmax, $ymax, $schnittpunkte, $farbe, $gefuellt,
         $fuellfarbe);

    next;
}

# Markierungen an den Achsen werden gezeichnet

if ($zeile =~ /^xmarkierung( layer=([0-9]+))? xwert=([-0-9\.]+)( font=(latex|ps))?( text=\"([^\"]*))?\"$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $xwert=$3;
    $font="ps";
    $font=$5 if (defined $4);
    $text="";
    $text=$7 if (defined $6);
    xmarkierung($layer,$xwert,$text,$font,0);
    next;
}

if ($zeile =~ /^ymarkierung( layer=([0-9]+))? ywert=([-0-9\.]+)( font=(latex|ps))?( text=\"([^\"]*))?\"$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $ywert=$3;
    $font="ps";
```

```
    $font=$5 if (defined $4);
    $text="";
    $text=$7 if (defined $6);
    ymarkierung($layer,$ywert,$text,"ps");
    next;
}

if ($zeile =~ /^xymarkierung( layer=([0-9]+))?( farbe=([0-9]+))? xwert=([-0-9\.]+)( ywert=([-0-9\.]+))?( xtext
    =\"([^\"]*)\")?( ytext=\"([^\"]*)\")?( linie=(x|y|xy))?( ([^ ]*))?$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xwert=$5;
    $xtext="";
    $func=0;
    $func=$15 if (defined $14);
    $ywert=calcfunk($func,"x",$xwert);
    $ywert=$7 if (defined $6);
    $linie="";
    $xtext=$9 if (defined $8);
    $ytext="";
    $ytext=$11 if (defined $10);
    $linie=$13 if (defined $12);

    xymarkierung($layer,$func,$xwert,$ywert,$xtext,$ytext,$farbe,$linie,"xy",SOLID);
    next;
}

if ($zeile =~ /^beschriftung( layer=([0-9]+))?( farbe=([0-9]+))? xwert=([-0-9\.]+) ywert=([-0-9\.]+) text
    =\"([^\"]*)\"( winkel=([-0-9\.]+))? ausrichtung=(l|c|r)( font=(latex|ps))?$/i) {
    $layer=90;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xwert=$5;
    $ywert=$6;
    $text=$7;
    $winkel=0;
    $winkel=$9 if (defined $8);
    $ausrichtung=$10;
    $font="ps";
    $font=$12 if (defined $11);

    beschriftung($layer,$farbe,$xwert,$ywert,$winkel,$ausrichtung,$text,$font);
    next;
}

if ($zeile =~/^punktezeichnen( layer=([0-9]+))?( farbe=([0-9]+))? k1=([-0-9\.]+),([-0-9\.]+) k2=([-0-9\.]+)
    ,([-0-9\.]+)( verbunden=(jamitk|jaohnek|nein))?$/i) {
    $layer=20;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $x1=$5;
    $y1=$6;
    $x2=$7;
    $y2=$8;
    $verbunden="ja";
    $verbunden=$10 if (defined $9);
    plotpunkte($layer,$x1,$y1,$x2,$y2,$farbe,$verbunden);
    next;
}

if ($zeile =~/^kreiszeichnen( layer=([0-9]+))?( farbe=([0-9]+))? k1=([-0-9\.]+),([-0-9\.]+)$/i) {
    $layer=20;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $x1=$5;
    $y1=$6;
    plotkreis($layer,$farbe,$x1,$y1);
    next;
}

if ($zeile =~/^plottangente( layer=([0-9]+))?( farbe=([0-9]+))? xwert=([-0-9\.]+) breite=([0-9\.]+)( markierung
    =(ja|nein))? ([^ ]*)$/i) {
    $layer=20;
    $layer=$2 if (defined $1);
    $farbe=0;
    $farbe=$4 if (defined $3);
    $xwert=$5;
    $breite=$6;
    $markierung="ja";
    $markierung=$8 if (defined $7);
    $func=$9;
    plottangente($layer,$func,$xwert,$breite,$farbe,$markierung);
    next;
}
```

```
if ($zeile =~/^plottangentedreieck( layer=([0-9]+))?( farbe=([0-9]+))? xwert=([-0-9\.]+) breite=([-0-9\.]+)(
     markierung=(wert|abl))? ([^ ]*)$/i) {
     $layer=80;
     $layer=$2 if (defined $1);
     $farbe=0;
     $farbe=$4 if (defined $3);
     $xwert=$5;
     $breite=$6;
     $markierung="";
     $markierung=$8 if (defined $7);
     $func=$9;
     plottangentedreieck($layer,$func,$xwert,$breite,$farbe,$markierung);
     next;
 }

if ($zeile =~/^plotsteigungsdreieck( layer=([0-9]+))?( farbe=([0-9]+))? xwert=([-0-9\.]+) breite=([-0-9\.]+)(
     markierung=(wert|abl))? ([^ ]*)$/i) {
     $layer=80;
     $layer=$2 if (defined $1);
     $farbe=0;
     $farbe=$4 if (defined $3);
     $xwert=$5;
     $breite=$6;
     $markierung="";
     $markierung=$8 if (defined $7);
     $func=$9;
     plotsteigungsdreieck($layer,$func,$xwert,$breite,$farbe,$markierung);
     next;
 }

 if ($zeile =~/^plotvektorpolar( layer=([0-9]+))?( farbe=([0-9]+))? s1=([-0-9\.]+),([-0-9\.]+) l1=([-0-9\.]+) w1
      =([-0-9\.]+)$/i) {
     $layer=80;
     $layer=$2 if (defined $1);
     $farbe=0;
     $farbe=$4 if (defined $3);
     $xstart=$5;
     $ystart=$6;
     $laenge=$7;
     $winkel=$8;
     plotvektorpolar($layer,$xstart,$ystart,$laenge,$winkel,$farbe,0);
     next;
 }

 if ($zeile =~/^plotvektor( layer=([0-9]+))?( farbe=([0-9]+))? k1=([-0-9\.]+),([-0-9\.]+) k2=([-0-9\.]+)
      ,([-0-9\.]+)$/i) {
     $layer=80;
     $layer=$2 if (defined $1);
     $farbe=0;
     $farbe=$4 if (defined $3);
     $xstart=$5;
     $ystart=$6;
     $xende=$7;
     $yende=$8;
     plotvektor($layer,$xstart,$ystart,$xende,$yende,0);
     next;
 }

 if ($zeile =~/^plotsaeule( layer=([0-9]+))?( farbe=([-0-9]+),([-0-9]+),([-0-9]+))? wert=([-0-9\.]+),([-0-9\.]+)
      ( breite=([-0-9\.]+))?$/i) {
     $layer=7;
     $layer=$2 if (defined $1);
     $farbe=0;
     $gefuellt=35;
     $fuellfarbe=27;

     if (defined $3) {
         $farbe=$4;
         $gefuellt=$5;
         $fuellfarbe=$6;

     }
     $xwert=$7;
     $ywert=$8;
     $breite=0.6;
     $breite=$10 if (defined $9);
     plotsaeule($layer,$xwert,$ywert,$farbe,$gefuellt,$fuellfarbe, $breite);
     next;
 }

 if ($zeile =~/^plotvektorparallel( layer=([0-9]+))?( farbe=([0-9]+))?( start=([-0-9\.]+),([-0-9\.]+))?
      \(([-0-9\.]+),([-0-9\.]+)\)( \(([-0-9\.]+),([-0-9\.]+)\))?( \(([-0-9\.]+),([-0-9\.]+)\))?( \(([-0-9\.]+)
      ,([-0-9\.]+)\))?( \(([-0-9\.]+),([-0-9\.]+)\))?( \(([-0-9\.]+),([-0-9\.]+)\))?( ersatzvektor=(parallel|
      ersatz))?$/i) {
     $layer=80;
     $layer=$2 if (defined $1);
     $farbe=0;
     $farbe=$4 if (defined $3);
     $x1=0;
     $y1=0;
```

```
        if (defined $5) {
        $x1=$6;
        $y1=$7;
        }
        $l1=$8;
        $w1=$9;
        $l2=$l3=$l4=$l5=$l6=0;
        $w2=$w3=$w4=$w5=$w6=0;
        if (defined $10) {
            $l2=$11;
            $w2=$12;
        }
        if (defined $13) {
            $l3=$14;
            $w3=$15;
        }
        if (defined $16) {
            $l4=$17;
            $w4=$18;

        }
        if (defined $19) {
            $l5=$20;
            $w5=$21;
        }
        if (defined $22) {
            $l6=$23;
            $w6=$24;
        }
        $ersatzvektor="";
        $ersatzvektor=$26 if (defined $25);

        plotvektorpolarmehrere($layer,$x1,$y1,$l1,$w1,$l2,$w2,$l3,$w3,$l4,$w4,$l5,$w5,$l6,$w6,$farbe,$ersatzvektor)
             ;
        next;
    }

    # Alle anderen Befehle sind Schrott
    printf("Unbekannter Befehl '%s'\n",$zeile);
}

    closefig;
    close(KOMMANDO);
}
```