Ein Funktionsparser und -plotter in PHP

Besondere Lernleistung
im Fach
Informatik (Grundkurs)

vorgelegt von
Jan Olligs

Gymnasium am Moltkeplatz, Krefeld

Schuljahr 2003/2004

Krefeld, den 22. April 2004

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Die Motivation

Als eines der ersten Themen im Informatik-Grundkurs behandelten wir im Jahr 2002 die Programmiersprache *PHP*[1].

Bei PHP handelt es sich um eine Sprache, die hauptsächlich zur Verarbeitung von HTML-Formularen konzipiert ist. Die PHP-Syntax erinnert stark an *Perl*[2], jedoch stellt PHP erstens keine so mächtige Einbettung von regulären Ausdrücken zur Verfügung, und ist zweitens, anders als Perl, dessen CGI Modul neben PHP die zweite verbreitete Methode zur Verarbeitung von HTML-Formulareingaben ist, servergebunden und kann nicht ohne Aufwand zu einfachen Systemverwaltungsaufgaben, o.ä. verwandt werden. Trotzdem hat PHP im Gegensatz zu Perl einige Vorteile; insbesondere kann es einfacher in HTML-Code integriert werden (oder auch HTML-Code in PHP) und ist C-ähnlicher als Perl – ein Bonus für Umsteiger.

Nachdem wir uns bei PHP zuerst der Datenbankverwaltung mit MySQL widmeten, folgte die Grafikgenerierung. Im Großen und Ganzen behandelten wir Vektorgrafiken, bei der Suche nach einem vernünftigen Verwendungszweck tauchte dann die Idee auf, einen Funktionsplotter im PHP zu schreiben.

Die Vorteile eines reinen PHP-basierten Plotters sind – und aus diesem Grund regte Herr Dyballa auch die Programmierung des Plotters an – dass auf Benutzerseite keinerlei Java, JavaScript oder die Installation von Plugins nötig sind und der Plotter somit auch Benutzern älterer Computer oder Benutzern, deren Browser diese Möglichkeiten nicht unterstützen, zugängig ist.

Schon bald (nämlich in der Stunde, in der das Thema „Funktionsplotter“ angesprochen wurde) taten sich mehrere Hindernisse auf, von denen das wichtigste das Generieren eines Funktionswertes aus einem gegebenen Funktionsterm und einem Funktionsargument war. Gegen die Verwendung eines *eval*-Befehles wie in Perl sprach die Tatsache, dass der User in diesem Fall ausführbaren Programmcode übermitteln könnte, der Schaden auf dem Server anrichten könnte. Es stellte sich auch heraus das Java-*sandbox*-ähnliche Funktionalität vorhanden war, jedoch wie in Java selbst mit Vorsicht zu genießen sei. Hieraus ergab sich die Notwendigkeit eines Parsers, der eingegebene Terme „verstehen“ konnte.

Das Schreiben eines Funktionsplotters wurde folglich als freiwillige Aufgabe gestellt; ich fing an, einen Plotter zu schreiben und arbeitete insbesondere während der Weihnachtsferien, aber auch später noch intensiv daran. Nach der Anmeldung als besondere Lernleistung holte ich außerdem zu einem kosmetischen und funktionellen Rundumschlag aus; insbesondere die verbesserte Benutzeroberfläche, jetzt

[1] PHP = PHP Hypertext Preprocessor
[2] Perl = Practical Extraction and Report Language

auf Deutsch, und die Umstrukturierung der tokenisierten Funktion von einem Array in einen Baum sind zu erwähnen. All dies wird unten ausführlich beschrieben.

Außer dem Parsing von Termen ergaben sich auch weitere Problemstellungen und Anforderungen, die ich im Folgenden ebenfalls detailliert beschreiben werde.

1.2 Über diese Ausarbeitung

Da diese Ausarbeitung nur ergänzend zum eigentlichen Projekt, dem Funktionsplotter, geschrieben wurde und ich den Quellcode (meiner Meinung nach) gut durchkommentiert habe (immerhin macht er kleingedruckt ca. 70 Seiten dieser Ausarbeitung aus), werde ich mich auf die wesentlichen Punkte beschränken. Diese sind der Tokenizer, die Umwandlung in einen *parse tree*, die automatische Schrittweitensteuerung und der Plottingalgorithmus.

Kapitel 2

Algorithmen

2.1 Tokenizer

Der Algorithmus für den Tokenizer ist nicht weiter komplex, da der Tokenizer nur wissen muss, wo er Zeichen auseinander reißen soll. Konkret ist dies an folgenden Punkten der Fall:

- Auf Leerzeichen splitten
- Um Operatoren, die nur aus einem Zeichen bestehen
- Um schließende Klammern
- Nach öffnenden Klammern, aber nicht vor diesen, da die Klammer zu einer Funktion gehören könnte

Der Algorithmus tut genau dies.

2.2 Umwandlung der Tokenliste in einen *parse tree*

Es gibt mehrere Möglichkeiten, die Token für weitere Bearbeitung vorzubereiten und abzuspeichern; drei sind besonders wichtig:

1. Keine weitere Bearbeitung, Speicherung als Tokenliste
2. Umwandlung in Stack, Postfixnotation
3. Umwandlung in parse tree, Infixnotation

Betrachten wir diese Möglichkeiten nun genauer:

Speicherung als Tokenliste

Sicherlich die einfachste Variante in diesem Stadium. Deshalb verwandte ich es auch in der ersten Version des Parsers. Das Problem mit dieser Möglichkeit ist, dass der Term, oder eher die Tokenliste, bei jedem neuen Funktionswert komplett neu verarbeitet werden muss. Es ist daher wesentlich effizienter, einen Teil der Verarbeitung schon vor der Iteration zu erledigen.

Umwandlung in Stack

Die Umwandlung in einen Stack dürfte den Term in die maschinenfreundlichste Notation zur Berechnung von Funktionswerten bringen. Es gibt keine lästige Rekursion zur Laufzeit, die den internen Stack belastet, der Algorithmus muss nicht übermäßig „denken“ und auch die Substitution ist aufgrund der geringen Tiefe der Struktur (nämlich überhaupt keine Vernestung, lediglich eine eindimensionale Verkettung) einfach zu bewerkstelligen. Wahrscheinlich führt ein Stack zur schnellstmöglichen Verarbeitung.

Umwandlung in parse tree

Ein Stack hat jedoch einen Nachteil: Viele Operationen außer der Berechnung seines Wertes sind leicht schwerfällig, da er die ursprüngliche Notation nicht bewahrt. Insbesondere zwei Erweiterungen, die ich bei den möglichen Verbesserungen erwähnen werde und für die ich auch schon Ansätze geschaffen habe, jedoch noch nicht implementiert habe, dürften in einer *parse tree*-Struktur leichter fallen: Symbolische Differenzierung und symbolische Vereinfachung. Mit letzterem meine ich Vereinfachung offensichtlicher Terme, z.B. ([token], '*', '0') zu '0'. Insbesondere, wenn [token] ein komplexer Term ist, kann diese Vereinfachung vor der Iteration viel Zeit sparen.

Der *parse tree* ist insofern eine Mischung aus Tokenliste und Stack, als dass er die Reihenfolge der Operationen, die Tokenliste, vollständig rekonstruierbar lässt, aber sie gleichzeitig schon vorverarbeitet: Die Operationen werden nach ihrer Präzedenz geordnet, zur Verarbeitung muss der rekursive Algorithmus keine Präzedenzentscheidung mehr treffen, sondern kann die Terme einfach verarbeiten.

Betrachten wir einen *parse tree* anhand des Beispiels `('3', '*', 'x', '^', '2')`:

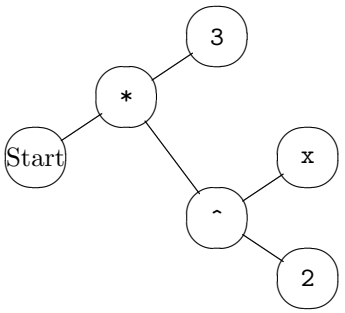

Abbildung 2.1: parse tree von `('3', '*', 'x', '^', '2')`

Ein Baum, hier der *parse tree*, ist ein einfach zusammenhängender Graph ohne Kreise, d.h. jeder Knoten ist von jedem anderen Knoten aus auf genau einem Weg zu erreichen. Er besteht aus einzelnen Punkten, den sog. *nodes* bzw. *Knoten*, die in der Grafik durch Kreise dargestellt werden. Unser Baum ist eine hierarchische Struktur; die einzelnen Knoten sind Eltern ihrer Kinder. Kinder sind die Knoten, die sich unterhalb der Eltern befinden und mit diesen durch eine Linie verbunden sind. Der oberste Knoten eines Baumes wird Wurzel genannt. Jeder Knoten kann als Wurzel eines Teilbaumes aufgefasst werden, der die Kinder, Kindeskinder, etc. dieses Knotens umfasst.

Um den *parse tree* zu bearbeiten, geht man nun rekursiv vor: Wenn ein Knoten eine Zahl enthält, wird diese Zahl als Ergebnis der Bearbeitung zurückgegeben. Wenn ein Knoten jedoch eine Funktion oder einen Operator enthält, so wird zunächst der

Teilbaum, bzw. die Teilbäume, dessen Elternteil der Knoten mit der Funktion bzw. dem Operator ist, bearbeitet und die Funktion bzw. der Operator dann auf das bzw. die Ergebnisse angewandt.

Am Beispiel des obigen *parse tree* hieße das, in rekursiven Aufrufen von `berechne(elternteil)`, wobei diese Funktion den Wert des Unterbaumes mit der Wurzel `elternteil` berechnet (zur Verdeutlichung sei $x = 5$):

- `berechne(Start)`
  - `berechne('*')`
    - `berechne('3')`
    - `Antwort '3'`
    - `berechne('^')`
      - `berechne('x')`
      - `Antwort '5'`
      - `berechne('2')`
      - `Antwort '2'`
    - `Antwort '25'` ($= 5^2$)
  - `Antwort '75'` ($= 3 \cdot 25$)
- `Antwort '75'`

Auf ähnliche Art und Weise kann auch die Rekonstruktion der Tokenliste erfolgen, wobei allerdings Eigenheiten bei der Verwendung von Klammern verloren gehen.

Zur Verdeutlichung noch ein weiterer *parse tree*, diesmal von `('sin(', '3', 'x', '+', '5', 'pi', '*', 'e', '^', 'x', ')')`:

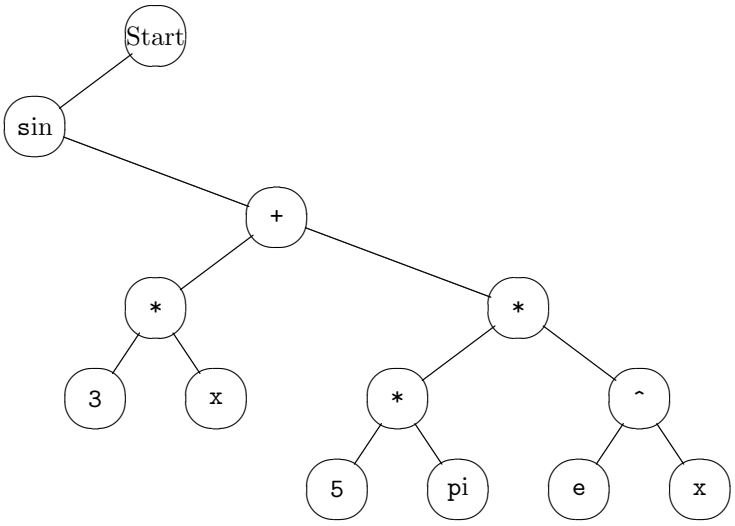

Abbildung 2.2: *parse tree* von `('sin(', '3', 'x', '+', '5', 'pi', '*', 'e', '^', 'x', ')')`

Ich bin einen Weg zwischen Stack und *parse tree* gegangen, bei dem die Tokenliste zunächst in einen *parse tree* und dann in einen Stack umgewandelt wird. Die Umwandlung eines *parse tree*s in einen Stack ist kein Problem, daher beschreibe ich hier nur den Algorithmus für die Umwandlung einer Tokenliste in einen *parse tree*:

Algorithmus 1 Umwandlung einer Tokenliste in einen *parse tree*

```
{Zuerst geklammerte Ausdrücke, da höchste Präzedenz}
for all Token do
  if Token ist eine Klammer then
    pushe Token und Position auf Stack
  end if
end for
while Stack ist nicht leer do
  Klammer ← pop Stack
  if Klammer ist öffnende Klammer then
    if Stack ist leer then
      Fehler
    end if
    if Klammer passt zu top Stack then
      if Mehr als eine Klammer in Stack 2 then
        pop Stack 2 {Verschachtelte Ebene; verwerfen}
      else {Genau eine Klammer in Stack 2}
        pushe Klammer auf Stack 2
        Zwischenbaum ← rekursiv aufgebauter Baum der Ausdrücke zwischen
        den Klammern
        if Klammer ist reine Klammer then
          Ersetze Klammern und Token zwischen Klammern durch Zwischen-
          baum
        else {unärer Operator (eine Funktion)}
          Ersetze Klammern und Token zwischen Klammern durch Baum, der
          Operator und Zwischenbaum enthält
        end if
        poppe Stack 2 zweimal
      end if
    else {Klammer passt nicht}
      Fehler
    end if
  else {Klammer ist schließende Klammer}
    pushe Klammer auf Stack 2
  end if
end while
if Stack 2 ist nicht leer then
  Fehler
end if{Nun zu den binären Operatoren}
for all Präzedenzniveaus in absteigender Reihenfolge do
  for all Token außer dem ersten und letzten do
    if Token ist binärer Operator aus diesem Präzedenzniveau then
      Ersetze Token und Token vor diesem und nach diesem durch Baum mit
      Operator und den Unterbäumen von links und rechts
    end if
  end for
end for
```

2.3 Automatische Schrittweitensteuerung

Die automatische Schrittweitensteuerung ist ein Algorithmus, der die Schrittweite dynamisch an die Form der Funktion anpasst. Sie hat zwei Vorteile: Der Benutzer muss sich nicht um die korrekte Einstellung der Schrittweite bemühen und die Rechenzeit wird gegebenenfalls reduziert.

Der Algorithmus betrachtet zwei aufeinander folgende Strecken, also drei aufeinander folgende Punkte des Graphen. Als ein Maß für die Qualität der Approximation wird der Winkel zwischen den Strecken angesehen; je eher die beiden Strecken in die gleiche Richtung zeigen, desto besser ist die Approximation. Statt des Winkels verwendet der Algorithmus die Höhe des Dreiecks aus den erwähnten drei Punkten; sie hängt außerdem noch von der Länge der Seiten ab, so dass lange Abstände nur bei fast gestreckten Winkeln möglich sind.

Zur Berechnung der Höhe verwendet der Algorithmus den Satz von Heron, nach dem für ein Dreieck mit Seitenlängen a, b und c und Halbumfang $s = \frac{a+b+c}{2}$ für dessen Flächeninhalt A gilt

$$A = \sqrt{s(s-a)(s-b)(s-c)} = A = \frac{1}{2}ah_a$$

Da wir die Seitenlängen aus den Koordinaten der Punkte mittels des Satzes des Pythagoras berechnen können, können wir die Höhe h_a ohne Rückgriff auf trigonometrische Funktionen berechnen.

Wir müssen jedoch beachten, dass die Schrittweite nicht zu klein wird, da dann einerseits die Rechenleistung zu groß werden könnte und andererseits die Schrittweite so klein werden könnte, dass sie durch Rundungsfehler zu Null wird. In diesem Fall würde der Algorithmus sich in einer Endlosschleife fangen.

Warum die Schrittweite auch nicht zu groß werden darf, verdeutlicht am besten ein Beispiel:

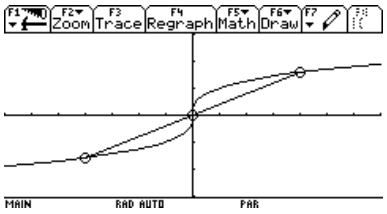

Abbildung 2.3: Möglicher Fehler bei zu großer Schrittweite

Der Plot folgt nicht mehr den Details und die Funktion erscheint gerade, obwohl sie gekrümmt ist.

Wenn die Höhe zu groß wird und außerdem die Schrittweite noch halbiert werden kann, dann wird dies getan; der letzte Punkt wird erneut berechnet (jetzt liegt er jedoch näher am vorletzten als zuvor). Wenn die Höhe zu klein wird und außerdem die Schrittweite noch verdoppelt werden kann, so wird dies getan; der letzte Punkt wird nicht erneut berechnet, da dies nur den Rechenaufwand erhöhen würde.

2.4 Plotter

Der Plotter hat die denkbar einfache Aufgabe, die generierten Punkte in einen Graphen umzuwandeln. Das einzige, was es hier zu beachten gibt, ist, dass viele Funktionsplotter und graphischen Taschenrechner, z.B. der TI Voyage 200, bei

Algorithmus 2 Automatische Schrittweitensteuerung

```
{Dies ist nur der eigentliche Algorithmus}
{Am Anfang müssen noch zwei Punkte bestimmt und undefiniert und außerhalb
auf die richtigen Werte gesetzt werden}
while x kleiner oder gleich Maximum do
  P_0 ← (x, f(x))
  if kein Fehler bei P_0 then
    if undefiniert then
      if Höhe größer als Maximalhöhe und s ≥ 2 · maximale Schrittweite then
        {wir heucheln, der letzte Punkt zu sein}
        P_0 ← P_1
        P_1 ← P_2
        {und halbieren die Schrittweite}
        s ← s/2
      end if
      if Höhe kleiner als Minimalhöhe und s · 2 ≤ minimale Schrittweite then
        s ← s * 2
      end if
    else {definiert}
      if Fehler bei P_1 then
        undefiniert ← wahr
      else {nur der vorletzte war fehlerhaft}
        undefiniert ← falsch
      end if
    end if
  else {Fehler}
    if außerhalb und s ≥ 2 · minimale Schrittweite then
      {wenn man die Fehlerstelle noch sehen kann ...}
      {... heucheln wir, der letzte Punkt zu sein}
      P_0 ← P_1
      P_1 ← P_2
      {und halbieren die Schrittweite}
      s ← s/2
    else
      undefiniert ← wahr
    end if
  end if
  x ← x + s
  P_2 ← P_1
  P_1 ← P_0
end while
{Am Ende müssen wir noch dafür sorgen, dass der rechte Rand beachtet wird}
```

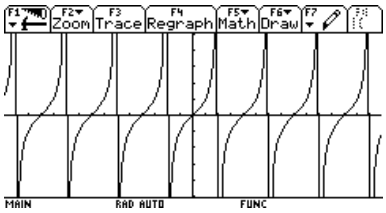

Abbildung 2.4: $f(x) = \tan x$ auf einem TI Voyage 200

Funktionen mit Polstellen, hier dargestellt an der Funktion $f(x) = \tan x$, Asymptoten einzeichnen.

Dies soll der Plotter nicht tun; ein Gegenbeispiel bietet z.B. Derive.

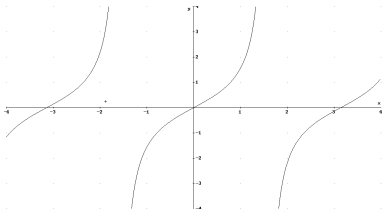

Abbildung 2.5: $f(x) = \tan x$ in Derive

Diese „Asymptoten“ entstehen, wenn sehr große Werte kurz vor einer Unstetigkeitsstelle mit sehr kleinen Werten kurz nach einer Unstetigkeitsstelle verbunden werden. Der von mir gegangene Weg ist, dass der Plotter zwei außerhalb des Zeichenbereiches liegende Punkte nicht verbindet.

Der Algorithmus setzt die Anforderungen wie gefordert um:

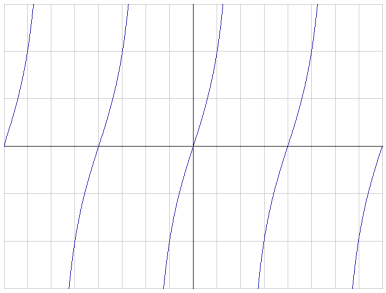

Abbildung 2.6: $f(x) = \tan x$ im Plotter

Algorithmus 3 Plottingalgorithmus

```
undefiniert ← wahr
außerhalb ← falsch
for all Punkte as P_neu do
  if Fehler bei Punkt P_neu then
    undefiniert ← wahr
    außerhalb ← falsch
  else {kein Fehler}
    if nicht undefiniert {letzter Wert war undefiniert} then
      if P_neu ist im Zeichenbereich then
        zeichne Linie von P_alt nach P_neu
        außerhalb ← falsch
      else {Punkt außerhalb des Zeichenbereichs}
        if nicht außerhalb {nur ein Punkt außerhalb ist in Ordnung} then
          zeichne Linie von P_alt nach P_neu
        end if
        außerhalb ← wahr
      end if
    else {letzter Wert war undefiniert}
      undefiniert ← falsch
      if P_neu ist im Zeichenbereich then
        außerhalb ← falsch
      else {Punkt außerhalb des Zeichenbereichs}
        außerhalb ← wahr
      end if
    end if
  end if
  P_alt ← P_neu
end for
```

Kapitel 3

Ausblick

Neben der Verbesserung des Interface gibt es mehrere, teilweise schon von mir vorbereitete, Verbesserungsmöglichkeiten:

- Symbolische Differenzierung
- Andere Darstellung, z.B. Polarkoordinaten, parametrisch, etc.
- Achsenbeschriftungen, möglichst „schön“ ('`pi`' ist nicht '`3.1415926536`')
- Symbolische Optimierung und Vorbearbeitung

3.1 Symbolische Differenzierung

Die symbolische Differenzierung ist schon fast komplett, mit den Hinweisen aus der Definitionsdatei ist das einzige Problem noch die Implementierung der Funktion, die die Differenzierung auch wirklich am *parse tree* durchführt. Da diese Funktion nur drei Regeln kennen muss (die Ableitung einer Konstante ist Null, die Ableitung der Variable ist eins und handle rekursiv) ist ihre Implementierung recht einfach. Man könnte die Ableitung zeichnen oder für die automatische Schrittweitensteuerung die Krümmung berechnen.

3.2 Andere Darstellung, z.B. Polarkoordinaten

Eine Menge Fleißarbeit; außerdem werden in Gymnasien diese Darstellungen kaum noch behandelt. Es stellen sich keine großartigen Probleme.

3.3 Achsenbeschriftungen, möglichst „schön“

Für dieses Problem habe ich keine Lösung gefunden. Es ist nicht schwierig, die Achsen mit Fließkommazahlen zu beschriften, aber der Plotter erlaubt die Eingabe von Termen für den dargestellten Bereich und die Gitterweite. Wenn die Eingabe für die Gitterweite zum Beispiel `pi` ist, so sollte die Achsenbeschriftung „π“ lauten, und nicht „3.1416“. Allein schon die Darstellung des griechischen Buchstaben π ist schwierig genug, so dass dieses Problem sicherlich zu den schwierigsten gehört.

3.4 Symbolische Optimierung und Vorbearbeitung

Unter Umständen kann der Rechenaufwand durch Berechnung von konstanten Termen wesentlich vereinfacht werden. Man braucht lediglich einen Algorithmus, der Operationen, die auf reinen Zahlen operieren, ausführt, dies jedoch bei anderen unterlässt. Unter „symbolische Optimierung“ verstehe ich ein Verfahren, dass erkennt, dass beispielsweise $x \cdot 0 = 0 \cdot x = 0$ oder $x^0 = 1$ gilt, wobei x jeden beliebigen Term bezeichnen kann. Für komplexe x kann dieses Verfahren ebenfalls zur Reduktion der Rechenzeit beitragen. Das Prinzip ist in beiden Fällen das gleiche: Die Berechnung eines möglichst großen Teils der Funktion wird außerhalb der Schleife erledigt, um Rechenzeit einzusparen.

Kapitel 4

Der Programmcode

Algorithmen sind gut, Quellcode ist besser, deshalb folgt er jetzt.

4.1 Datei „index.htm“

```
<html>
<head>
<title>Parser/Plotter Projekt</title>
<link rel="stylesheet" type="text/css" href="units/style.css">
</head>
<body>
<h1>Parser/Plotter Projekt</h1>

<h2>Motivation</h2>

<p>Dieses Parser/Plotter-Projekt entstand aus der simplen Idee, im
Informatikunterricht einen Funktionsplotter und f&uuml;r diesen dann gleich auch
einen Funktionsparser zu programmieren. Das der Aufwand f&uuml;r dieses Vorhaben
jedoch von gr&ouml;&szlig;erem Ma&szlig;e war, als dass es im Rahmen des
Informatikunterrichts zu bew&auml;ltigen w&auml;re, wurde es als freiwillige
Aufgabe gestellt. (Wenn Sie sich einen Eindruck davon machen wollen, wie viel
Code hinter diesem Projekt steckt, dann gucken Sie sich einfach den
<a href="source.php">Quellcode</a> in der Druckvorschau an). Diese Endversion
ist tats&auml;chlich die dritte bis vierte Version des Plotters, je nachdem, wo
man zu z&auml;hlen anf&auml;ngt.</p>

<p>Aufgrund des enormen Aufwands des Projektes schlug mein Informatiklehrer, Herr
Dyballa, vor, eine Anerkennung des Projektes als <i>besondere Lernleistung</i>
zu beantragen. Eine besondere Lernleistung z&auml;hlt im Abitur in vierfacher
Wertung, wie ein normales Abiturfach; die normalen vier Abiturf&auml;cher werden
bei einer eingebrachten besonderen Lernleistung jedoch nur dreifach statt
vierfach gewertet, so dass sich die maximal zu erreichende Punktzahl nicht
&auml;ndert. Den theoretischen / schriftlichen Teil der Arbeit
<a href="text/lernleistung.pdf">k&ouml;nnen Sie ebenfalls einsehen</a>.</p>

<h2>Eine Bitte</h2>

<p>Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl&auml;rung der
Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem&uuml;ht,
den Code so gut wie m&ouml;glich zu kommentieren und habe leider nicht genug
Zeit, um E-Mails &uuml;ber technische Details zu beantworten.</p>

<h2>Inhalt</h2>

<ul>
   <li><a href="help.htm">Hilfe</a></li>
   <li><a href="reference.php">Referenz</a></li>
   <li><a href="source.php">Quellcode</a></li>
   <li><a href="plotter.php">Plotter</a></li>
</ul>

<h2>Wo fange ich an?</h2>

<p>Wenn Sie zum ersten Mal hier sind, so sollten Sie sich auf jeden Fall die
<a href="help.htm">Hilfe</a> und die <a href="reference.php">Referenz</a>
```

```
anschauen. Dann k&ouml;nnen Sie <a href="plotter.php">zum Plotter
weitergehen</a>.</p>

<h2>Hinweis</h2>

<p>Der Autor &uuml;bernimmt keine Haftung f&uuml;r eventuelle durch die Nutzung
von zu diesem Projekt geh&ouml;rigen Skripts, Texten und anderen Produkten
entstehende oder entstandene mittelbare oder unmittelbare Sch&auml;den jedweder
Art.</p>

<p>Nutzung auf eigene Gefahr.</p>

<p>&copy; Jan Olligs, 2004</p>
</body>
</html>
```

4.2 Datei „help.htm“

```
<html>
<head>
<title>Hilfe f&uuml;r Parser/Plotter</title>
<link rel="stylesheet" type="text/css" href="units/style.css">
</head>
<body>

<h1>Hilfe! Ein Plotter!</h1>

<p>Das Formular f&uuml;r den Plotter ist oben mit Reitern ausgestattet, die das
&quot;Umbl&auml;ttern&quot; der Seiten erm&ouml;glicht. Die vier Seiten sind
<b>Einstellungen</b>, <b>Funktionen</b>, <b>Fehlermeldungen</b> und
<b>Funktionsgraph</b>. Bitte dr&uuml;cken Sie nach &Auml;nderungen immer zuerst
auf den Button mit der Aufschrift &quot;&Uuml;bernehmen&quot; und wechseln Sie
dann erst die Seite. In den meisten Feldern, in denen Sie Zahlenangabem machen
k&ouml;nnen, k&ouml;nnen Sie ebensogut Terme eingeben. Eine wichtige Ausnahme
bildet das Bildformat. Die Einstellungen, die auf den einzelnen Seiten getroffen
werden k&ouml;nnen, sind die folgenden:</p>

<h2>Einstellungen</h2>

<p>Allgemeine Einstellungen: Bildh&ouml;he und -breite, Ausgabeformat,
allgemeine Einstellungen f&uuml;r den Graphen (insbesondere der Plotbereich und
die Schrittweite / Schrittweitensteuerung), Farben und Konstanten.</p>

<h3>Bildeinstellungen (Breite, H&ouml;he und Format)</h3>

<p>Breite und H&ouml;he des Bildes in Pixeln und das Bildformat.</p>

<h3>Dargestellter Bereich</p>

<h4>X-Untergrenze</h4>

<p>x-Wert des linken Randes des dargestellten Bereiches.</p>

<h4>X-Obergrenze</h4>

<p>x-Wert des rechten Randes des dargestellten Bereiches.</p>

<h4>Abstand der X-Gitterlinien</h4>

<p>Genau das, in LE.</p>

<h4>Y-Untergrenze, Y-Obergrenze, Abstand der Y-Gitterlinien</h4>

<p>Vollkommen analog.</p>

<h3>Zeicheneinstellungen</h3>

<h4>Schrittweite</h4>

<p>Abstand der x-Koordinaten der zum Zeichnen verwandten Punkte.</p>

<h4>Automatische Schrittweitensteuerung</h4>

<p>Anstelle einer Schrittweite: Der Plotter passt seine Schrittweite automatisch
an die Funktion an. Ggf. kann dies zu einer schlechten Darstellung f&uuml;hren
(auch wenn mir dies noch nie untergekommen ist, sondern im Gegenteil die manuelle
```

```
Schrittweitensteuerung die schlechteren Graphen gezeichnet hat). In diesem
unwahrscheinlichen Fall sollten Sie die automatische Schrittweitensteuerung
deaktivieren.</p>

<h3>Zeichenfarben</h3>

<p>Exakt das.</p>

<h3>Konstanten</h3>

<p>Um das Plotten von bestimmten Funktionen, zum Beispiel solchen, die den
goldenen Schnitt ben&ouml;tigen, zu vereinfachen, lassen sich hier Konstanten
eingeben, die im Plotterformular verwandt werden k&ouml;nnen. Jede
Konstantendefinition muss sich in einer eigenen Zeile befinden. Die Definition
folgt der Form <tt>[Konstantenname]=[Konstantendefinition]</tt>. Um den goldenen
Schnitt zu definieren, w&uuml;rde man also beispielsweise eine Zeile
<tt>phi=(1+sqrt(5))/2</tt> hinzuf&uuml;gen.</p>

<h2>Funktionen</h2>

<p>Funktionen (Terme, Farben und Parameter), Asymptoten (Punkte, Farbe, gestrichelt
/ nicht gestrichelt) &auml;ndern, erstellen und l&ouml;schen, schattierter
Bereich (Grenzen, Farbe, Transparenz, an / aus).</p>

<h3>Funktionen / Funktion <i>n</i></h3>

<h4>Funktion</h4>

<p>Der Funktionsterm.</p>

<h4>Parametername</h4>

<p>Im Funktionsterm k&ouml;nnen Sie einen <i>Parameter</i> (Formparameter)
benutzen. Der Funktionsgraph wird f&uuml;r mehrere Werte des Parameters
gezeichnet, die Sie angeben k&ouml;nnen.</p>

<h4>Parameter von</h4>

<p>Kleinster Parameterwert.</p>

<h4>Parameter bis</h4>

<p>Gr&ouml;&szlig;ter Parameterwert.</p>

<h4>Parameter Schritte</h4>

<p>Schrittweite des Parameters. Wenn zum Beispiel der Paramter <tt>t</tt>
hei&szlig;t und von -1 bis 1 mit Schrittweite 1 geht, dann wird die Funktion
dreimal gezeichnet, wobei f&uuml;r <tt>t</tt> einmal -1, einmal 0 und einmal 1
eingesetzt wird.</p>

<h4>Plotfarbe</h4>

<p>Offensichtlich.</p>

<h4>Funktion l&ouml;schen</h4>

<p>Wenn Sie diese Funktion nicht mehr darstellen, also l&ouml;schen wollen, dann
aktivieren Sie dieses K&auml;stchen. Die Funktion verschwindet dann ins
Datennirvana und kann (au&szlig;er &uuml;ber die Back-Funktion Ihres Browsers)
nicht wiederhergestellt werden.</p>

<h4>Neue Funktionen</h4>

<p>Wenn Sie mehr Funktionen zeichnen wollen, als auf dem Formular angef&uuml;hrt
sind, so geben Sie hier die Zahl der zus&auml;tzlich zu erstellenden Funktionen
ein.</p>

<h3>Asymptoten / Asymptote <i>n</i></h3>

<h4>x-/y-Koordinate Punkt 1 / 2</h4>

<p>Die Asymptoten werden durch zwei Punkte definiert. Geben Sie hier die
Koordinate der Punkte ein.</p>

<h4>Farbe</h4>

<p>Offensichtlich.</p>

<h4>Gestrichelt zeichnen</h4>
```

```
<p>Die Asymptote wird nicht als durchgezogene, sondern als gestrichelte Linie
gezeichnet.</p>

<h4>Asymptote l&ouml;schen</h4>

<p>Wenn Sie diese Asymptote nicht mehr darstellen, also l&ouml;schen wollen, dann
aktivieren Sie dieses K&auml;stchen. Die Asymptote verschwindet dann ins
Datennirvana und kann (au&szlig;er &uuml;ber die Back-Funktion Ihres Browsers)
nicht wiederhergestellt werden.</p>

<h4>Neue Asymptoten</h4>

<p>Wenn Sie mehr Asymptoten zeichnen wollen, als auf dem Formular angef&uuml;hrt
sind, so geben Sie hier die Zahl der zus&auml;tzlich zu erstellenden Asymptoten
ein.</p>

<h3>Schattieren</h3>

<h4>Unterer x-Wert</h4>

<p>x-Wert, bei dem mit dem Schattieren begonnen wird.</p>

<h4>Oberer x-Wert</h4>

<p>x-Wert, bei dem mit dem Schattieren aufgeh&ouml;rt wird.</p>

<h4>Schattenfarbe</h4>

<p>Offensichtlich.</p>

<h4>Transparenz</h4>

<p>Die Transparenz der schattierten Fl&auml;che in Prozent. 0% wird nicht
dargestellt, 100% ist deckend.</p>

<h4>Grenzfunktion 1 / 2</h4>

<p>Der schattierte Bereich wird von zwei Funktionen begrenzt, die hier angegeben
werden k&ouml;nnen.</p>

<h4>Begrenzungsfarbe</h4>

<p>Die Farbe der Begrenzungsfunktionen.</p>

<h4>Schattieren</h4>

<p>Aktiviert / deaktiviert das Schattieren eines Bereiches.</p>

<h2>Fehlermeldungen</h2>

<p>Hier werden alle Eingaben &uuml;berpr&uuml;ft und eventuelle Fehler ausgegeben.
Aber Achtung: Manche &quot;Fehler&quot; sind gar keine, da der Plotter bei
fehlenden und fehlerhaften Eingaben Standardwerte benutzt, so dass am Anfang
diese Werte nicht definiert sind.</p>

<p>Keine Einstellungen m&ouml;glich.</p>

<h2>Funktionsgraph</h2>

<p>Ausgabe des Funktionsgraphen. Je nach Komplexit&auml;t und Anzahl der
Funktionen kann das Laden der Graphik einige Zeit dauern.</p>

<p>Keine Einstellungen m&ouml;glich.</p>

<h1>Wie gebe ich Terme ein?</h1>

<p>Wie Sie sie auch schreiben w&uuml;rden. Allerdings beherrscht der Plotter die
implizite Multiplikation nicht vollst&auml;ndig, da auch Variablen und
Konstanten mit mehreren Buchstaben werlaubt sind. Also sind &quot;ab&quot; und
&quot;a * b&quot; nicht das gleiche. F&uuml;r eine Liste der unterst&uuml;tzten
Funktionen, Operatoren, Konstanten und Klammern verweise ich auf die
<a href="reference.php">Referenz</a>.</p>

<p><a href="index.htm">Start</a></p>

<p>&copy; Jan Olligs, 2004</p>

</body>
</html>
```

4.3 Datei „source.php“

```
<html>
<head>
<title>Funktionsparser Quellcode</title>
</head>
<body>
Die Copyrighthinweise zu folgendem Code befinden sich als Kommentare an den
Dateik&ouml;pfen.
<?php

   ##############################################################################
   # Dies ist die Datei 'source.php'                                            #
   #                                                                            #
   # Sie enth"alt Funktions- und "ahnliche Definitionen des Parser/Plotter-Projekts #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen #
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den #
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund   #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich  #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder #
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies            #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-  #
   # Adresse oder "uber den Sysadmin.                                           #
   #                                                                            #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses #
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
   # Nutzung auf eigene Gefahr.                                                 #
   #                                                                            #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der       #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht, #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug   #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                  #
   #                                                                            #
   # (c) Jan Olligs, 2004                                                       #
   # Alle Rechte vorbehalten                                                    #
   #                                                                            #
   # Jan.Olligs@gmx.net                                                         #
   ##############################################################################

   $files = array(
      'reference.php',
      'source.php',
      'graph.php',
      'plotter.php',
      'tabs/error.php',
      'tabs/functions.php',
      'tabs/general.php',
      'tabs/graph.php',
      'units/mathdef.php',
      'units/parser.php',
      'units/graphics.php',
      'units/plotter_constants.php',
      'units/various.php'
   );

   foreach ($files as $name) {
      print "<h1>Datei <i>" . htmlspecialchars($name) . "</i>:</h1>\n";
      show_source($name);
   }

?>
</body>
</html>
```

4.4 Datei „reference.php“

```
<html>
<head>
<title>Parser-Referenz</title>
<link rel="stylesheet" type="text/css" href="units/style.css">
</head>
<body>
<?php

   ##############################################################################
   # Dies ist die Datei 'reference.php'                                         #
   #                                                                            #
   # Sie enth"alt Funktions- und "ahnliche Definitionen des Parser/Plotter-Projekts #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen #
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den #
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund   #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich  #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder #
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies            #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-  #
   # Adresse oder "uber den Sysadmin.                                           #
   #                                                                            #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses #
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
   # Nutzung auf eigene Gefahr.                                                 #
   #                                                                            #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der       #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht, #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug   #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                  #
   #                                                                            #
   # (c) Jan Olligs, 2004                                                       #
   # Alle Rechte vorbehalten                                                    #
   #                                                                            #
   # Jan.Olligs@gmx.net                                                         #
   ##############################################################################

   include("units/mathdef.php");

   function table_row($array, $item, $title) {
      if (array_key_exists($item, $array) and (!is_array($array[$item]) or 0 != count($array[$item]))) {
         print "   <tr>\n";
         print "      <td><b>$title:</b></td>\n";
         if (is_array($array[$item])) {
            print "      <td>" . implode(', ', array_map('htmlspecialchars', $array[$item])) . "</td>\n";
         } else {
            print "      <td>" . htmlspecialchars($array[$item]) . "</td>\n";
         }
         print "   </tr>\n";
      }
   }
?>
Die folgenden <i><b>Funktionen (un&auml;re Operatoren)</b></i> werden unterst&uuml;tzt:<br>
<?php
   foreach ($global_unary_operators as $name => $properties) {
      if (!is_array($properties)) {
         continue; # ACHTUNG: Das w"are ein Implementierungsfehler
      }
      print "<table>\n";
      print "   <tr>\n";
      print "      <td><b>Funktion:</b></td>\n";
      print "      <td><i>" . htmlspecialchars($name) . "</i></td>\n";
```

```
      print "   </tr>\n";
      table_row($properties, 'name',    'Name');
      table_row($properties, 'others',  'Varianten');
      table_row($properties, 'info',    'Information');
      table_row($properties, 'example', 'Beispiel(e)');
      print "</table>\n";
      print "<br>\n\n";
   }
?>
<br>
Die folgenden <i><b>Operatoren (bin&auml;re Operatoren)</b></i> werden unterst&uuml;tzt:<br>
<?php
   $i = 0;
   foreach ($global_binary_operators as $prec) {
      ++$i;
      print "<u>Pr&auml;zedenz $i</u>:\n";
      foreach ($prec as $name => $properties) {
         if (!is_array($properties)) {
            continue; # ACHTUNG: Das w"are ein Implementierungsfehler
         }
         print "<table>\n";
         print "   <tr>\n";
         print "      <td><b>Operator:</b></td>\n";
         print "      <td><i>" . htmlspecialchars($name) . "</i></td>\n";
         print "   </tr>\n";
         table_row($properties, 'name',    'Name');
         table_row($properties, 'info',    'Information');
         table_row($properties, 'example', 'Beispiel(e)');
         print "</table>\n";
         print "<br>\n\n";
      }
   }
?>
<br><br>
Die folgenden <i><b>Konstanten</i></b> sind definiert:<br>
<?php
   foreach ($global_constants as $const => $value) {
      print "$const = $value<br>\n";
   }
?>
<br><br>
Au&szlig;erdem sind folgende <i><b>Klammern</i></b> verf&uuml;gbar:<br>
<?php
   foreach ($global_parens as $pair) {
      print "&ouml;ffnet als '$pair[0]', schlie&szlig;t als '$pair[1]'<br>\n";
   }
?>
<br><br><br>
<a href="source.php">&copy; Jan Olligs, 2004</a>
</body>
</html>
```

4.5 Datei „plotter.php“

```
<html>
<head>
<title>Funktionsplotter</title>
<link rel="stylesheet" type="text/css" href="units/style.css">
</head>
<body>
<?php

   ##############################################################################
   # Dies ist die Datei 'plotter.php'                                           #
   #                                                                            #
   # Sie enth"alt das Plotterformular des Parser/Plotter-Projekts               #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
```

```
# Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in             #
# andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus         #
# oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen      #
# wollen, so kontaktieren Sie mich bitte.                                          #
#                                                                                  #
# Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich        #
# dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den      #
# Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund         #
# auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich        #
# mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder      #
# den Urheber geistigen Eigentums vorenthalten, so geschieht dies                  #
# unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-        #
# Adresse oder "uber den Sysadmin.                                                 #
#                                                                                  #
# Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses      #
# entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
# Nutzung auf eigene Gefahr.                                                       #
#                                                                                  #
# Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der             #
# Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,       #
# den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug         #
# Zeit, um E-Mails "uber technische Details zu beantworten.                        #
#                                                                                  #
# (c) Jan Olligs, 2004                                                             #
# Alle Rechte vorbehalten                                                          #
#                                                                                  #
# Jan.Olligs@gmx.net                                                               #
####################################################################################

include('units/mathdef.php');
include('units/parser.php');
include('units/plotter_constants.php');
include('units/various.php');


####################################################################################
# $_REQUEST['tab'] bestimmt die angeseigte Seite:                                  #
#    'general'   - Generelle Einstellungen / Konstanten                            #
#    'functions' - Funktionen / Asymptoten (Eingabe)                               #
#    'error'     - Fehlerausgabe                                                   #
#    'graph'     - Plot                                                            #
#                                                                                  #
# Parameter, die auf den Seiten eingestellt werden:                                #
#    'general':                                                                    #
#       'img_type', 'width', 'height'                                              #
#       'x_min', 'x_max', 'x_grid', 'y_min', 'y_max', 'y_grid'                     #
#       'step', 'auto_step'                                                        #
#       'bg_col', 'ax_col', 'gr_col', 'pl_col'                                     #
#       'const'                                                                    #
#    'functions':                                                                  #
#       's_min', 's_max', 'sh_col', 's_trans', 's_f1', 's_f2', 'sf_col', 'do_shade'#
#       'function_[\d]_[...]', 'num_new_func'                                      #
#       'asymptote_[\d]_[...]', 'num_new_asym'                                     #
#    'error', 'graph':                                                             #
#       Nur Ausgabe                                                                #
####################################################################################

$tab = $_REQUEST['tab'];
switch (strtolower($tab)) {
   case ('general'):
   case ('functions'):
   case ('error'):
   case ('graph'):
      $tab = strtolower($tab);
      break;
   default:
      $tab = 'general';
      break;
}

# Wenn Werte noch nicht existieren, dann schaffen wir sie jetzt

$_REQUEST = populate_request(
   $_REQUEST,
   array(
      'width'      => 'img_width',
      'height'     => 'img_height',
      'x_min'      => 'x_min',
      'x_max'      => 'x_max',
      'x_grid'     => 'x_grid',
```

```
      'y_min'      => 'y_min',
      'y_max'      => 'y_max',
      'y_grid'     => 'y_grid',
      'step'       => 'step',
      'auto_step'  => 'auto_step',
      'bg_col'     => 'BG_color',
      'ax_col'     => 'AX_color',
      'gr_col'     => 'GR_color',
      'pl_col'     => 'PL_color',
      's_min'      => 'shading_min',
      's_max'      => 'shading_max',
      'sh_col'     => 'SH_color',
      's_trans'    => 'shading_trans',
      's_f1'       => 'shading_f1',
      's_f2'       => 'shading_f2',
      'sf_col'     => 'SF_color',
      'do_shade'   => 'do_shade'
   ),
   $plotter_standards
);

if (is_null($_REQUEST['img_type']) or $_REQUEST['img_type'] == '') {
   $_REQUEST['img_type'] = 0;
}


# Jetzt m"ussen wir uns um gel"oschte und neu generierte Funktionen und
# Asymptoten k"ummern

$oldfunc = unpack_function_terms($_REQUEST);
$num_new = $_REQUEST['num_new_func'];
if (is_null($num_new) or $num_new == '' or !is_numeric($num_new) or !is_int($num_new + 0) or $num_new < 0) {
   $num_new = 0;
   $_REQUEST['num_new_func'] = $num_new;
}
if (count($oldfunc) == 0) {
   array_push(
      $oldfunc,
      array(
         'term'  => $plotter_standards{'func'}[1],
         'color' => $plotter_standards{'PL_color'}[1],
         'param' => array(
            'name' => 'NOPARAM',
            'from' => 0,
            'to'   => 0,
            'step' => 1
         )
      )
   );
}
$functions = array();
ksort($oldfunc);
foreach ($oldfunc as $values) {
   if ($values{'delete'} != 'true') {
      array_push($functions, $values);
   }
}
for ($i = 0; $i < $num_new; ++$i) {
   array_push(
      $functions,
      array(
         'term'   => $plotter_standards{'func'}[1],
         'color'  => $plotter_standards{'PL_color'}[1],
         'delete' => '',
         'param'  => array(
            'name' => 'NOPARAM',
            'from' => 0,
            'to'   => 0,
            'step' => 1
         )
      )
   );
}
foreach (array_keys($_REQUEST) as $key) {
   if (preg_match('/^function_/i', $key)) {
      unset($_REQUEST[$key]);
   }
}
$_REQUEST = array_merge($_REQUEST, pack_function_terms($functions));
unset($oldfunc);
```

```
   $oldasym = unpack_asymptote_terms($_REQUEST);
   $num_new = $_REQUEST['num_new_asym'];
   if (is_null($num_new) or $num_new == '' or !is_numeric($num_new) or !is_int($num_new + 0) or $num_new < 0) {
      $num_new = 0;
      $_REQUEST['num_new_asym'] = $num_new;
   }
   $asymptotes = array();
   ksort($oldasym);
   foreach ($oldasym as $values) {
      if ($values{'delete'} != 'true') {
         array_push($asymptotes, $values);
      }
   }
   for ($i = 0; $i < $num_new; ++$i) {
      array_push(
         $asymptotes,
         array(
            'p1_x'   => $plotter_standards{'asym_px1'}[1],
            'p1_y'   => $plotter_standards{'asym_py1'}[1],
            'p2_x'   => $plotter_standards{'asym_px2'}[1],
            'p2_y'   => $plotter_standards{'asym_py2'}[1],
            'color'  => $plotter_standards{'AS_color'}[1],
            'dashed' => $plotter_standards{'asym_dashed'}[1],
            'delete' => ''
         )
      );
   }
   foreach (array_keys($_REQUEST) as $key) {
      if (preg_match('/^asymptote_/i', $key)) {
         unset($_REQUEST[$key]);
      }
   }
   $_REQUEST = array_merge($_REQUEST, pack_asymptote_terms($asymptotes));
   unset($oldasym);

?>
   <table border="0" width="100%">
      <tr>
         <td align="center" class="version1" size="20%">
            <a href="index.htm">Startseite</a>
         </td>

         <td align="center" class="<?php print ($tab == 'general' ? 'current' : 'version2') ?>" size="20%">
            <a href="plotter.php?<?php
               print http_build_query(
                  array_merge(
                     $_REQUEST,
                     array(
                        'tab'          => 'general',
                        'num_new_func' => 0,
                        'num_new_asym' => 0
                     )
                  )
               );
            ?>">Einstellungen</a>
         </td>
         <td align="center" class="<?php print ($tab == 'functions' ? 'current' : 'version1') ?>" size="20%">
            <a href="plotter.php?<?php
               print http_build_query(
                  array_merge(
                     $_REQUEST,
                     array(
                        'tab'          => 'functions',
                        'num_new_func' => 0,
                        'num_new_asym' => 0
                     )
                  )
               );
            ?>">Funktionen</a>
         </td>
         <td align="center" class="<?php print ($tab == 'error' ? 'current' : 'version2') ?>" size="20%">
            <a href="plotter.php?<?php
               print http_build_query(
                  array_merge(
                     $_REQUEST,
                     array(
                        'tab'          => 'error',
                        'num_new_func' => 0,
                        'num_new_asym' => 0
```

```
                  )
               )
            );
         ?>">Fehlermeldungen</a>
      </td>
      <td align="center" class="<?php print ($tab == 'graph' ? 'current' : 'version1') ?>" size="20%">
         <a href="plotter.php?<?php
            print http_build_query(
               array_merge(
                  $_REQUEST,
                  array(
                     'tab'          => 'graph',
                     'num_new_func' => 0,
                     'num_new_asym' => 0
                  )
               )
            );
         ?>">Funktionsgraph</a>
      </td>
   </tr>
   <tr>
      <td class="body" colspan="5">
         <form action="plotter.php" method="post">
<?php
   # Hier binden wir die Tabs ein
   switch ($tab) {
      case ('general'):
         include('tabs/general.php');
         break;
      case ('functions'):
         include('tabs/functions.php');
         break;
      case ('error'):
         include('tabs/error.php');
         break;
      case ('graph'):
         include('tabs/graph.php');
         break;
      default:
         print "Autsch, das tut weh. Wahrscheinlich ein Programmierfehler.\n";
         break;
   }
?>
         </form>
      </td>
   </tr>
</table>

<p><a href="index.htm">&copy; Jan Olligs, 2004</a></p>
</body>
</html>
```

4.6 Datei „graph.php“

```
<?php

   ##############################################################################
   # Dies ist die Datei 'graph.php'                                             #
   #                                                                            #
   # Sie enth"alt den eigentlichen Plotter des Parser/Plotter-Projekts          #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn  #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen#
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
```

```
# dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den      #
# Tokenizer, der Parser und die Schrittweitensteuerung habe ich von Grund          #
# auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich        #
# mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder      #
# den Urheber geistigen Eigentums vorenthalten, so geschieht dies                  #
# unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-        #
# Adresse oder "uber den Sysadmin.                                                 #
#                                                                                  #
# Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses      #
# entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
# Nutzung auf eigene Gefahr.                                                       #
#                                                                                  #
# Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der             #
# Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,       #
# den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug         #
# Zeit, um E-Mails "uber technische Details zu beantworten.                        #
#                                                                                  #
# (c) Jan Olligs, 2004                                                             #
# Alle Rechte vorbehalten                                                          #
#                                                                                  #
# Jan.Olligs@gmx.net                                                               #
####################################################################################

include("units/mathdef.php");
include("units/parser.php");
include("units/graphics.php");
include("units/plotter_constants.php");
include("units/various.php");

####################################################################################
# Inhalte des $_REQUEST-Arrays, die f"ur den Plotter relevant sind:                #
#    'img_type'              - Ausgabeformat, siehe 'plotter_constants.php'        #
#    'const'                 - Konstanten in der Form "[name]=[term]\n[name]..."   #
#    'function_[\d]_[...]'   - Funktionen, siehe 'plotter_constants.php'           #
#    'asymptote_[\d]_[...]'  - Asymptoten, siehe 'plotter_constants.php'           #
#    'x_min'                 - kleinster dargestellter x-Wert                      #
#    'x_max'                 - gr"o"ster dargestellter x-Wert                      #
#    'x_grid'                - x-Gitterweite                                       #
#    'y_min'                 - kleinster dargestellter y-Wert                      #
#    'y_max'                 - gr"o"ster dargestellter y-Wert                      #
#    'y_grid'                - y-Gitterweite                                       #
#    'width'                 - Bildbreite in Pixel                                 #
#    'height'                - Bildh"ohe in Pixel                                  #
#    'step'                  - Schrittweite der Punkte (delta_x)                   #
#    'auto_step'             - automatische Schrittweitensteuerung an ('true')     #
#                              oder aus ('false')                                  #
#    'bg_col'                - Hintergrundfarbe, siehe 'plotter_constants.php'     #
#    'pl_col'                - Standardfunktionsfarbe, s. 'plotter_constants.php'  #
#    'ax_col'                - Achsenfarbe, siehe 'plotter_constants.php'          #
#    'gr_col'                - Gitterfarbe, siehe 'plotter_constants.php'          #
#    's_min'                 - kleinster schattierter x-Wert                       #
#    's_max'                 - gr"o"ster schattierter x-Wert                       #
#    'sh_col'                - Schattenfarbe, siehe 'plotter_constants.php'        #
#    's_trans'               - Schattentransparenz in Prozent                      #
#    's_f1'                  - den Schatten einschlie"sende Funktion 1             #
#    's_f2'                  - den Schatten einschlie"sende Funktion 2             #
#    'sf_col'                - Farbe der einschlie"senden Funktionen, s. ...       #
#    'do_shade'              - Schattieren an oder aus (s.o.)                      #
####################################################################################

# F"ur aufwendige Graphen deaktivieren wir l"astige Zeitbegrenzungen ...

set_time_limit(0);

# F"ur das Einf"ugen von Malzeichen

$poss_for_mult = array_merge(range('a', 'z'), range('A', 'Z'));
foreach ($global_parens as $pair) {
   array_push($poss_for_mult, $pair[0]);
}

# Fehlercheck f"angt hier an

$image_type = $_REQUEST['img_type'];
if (!(imagetypes() & $plotter_standards{'img_types'}[1][$image_type][1])) {
   $image_type = 0;
}
while (!(imagetypes() & $plotter_standards{'img_types'}[1][$image_type][1])) {
   if ($image_type >= count($plotter_standards{'img_types'})) {
      die;
```

```
   }
   ++$image_type;
}

# Zuerst parsen wir die Konstanten, da sie bei allen Termen ben"otigt werden

$constants = array();
$temp = preg_split('/\n/', $_REQUEST['const']);
foreach ($temp as $t) {
   $t = preg_split('/=/', $t);
   if (!is_array($t) or count ($t) != 2) {
      continue;
   }
   $t[0] = ltrim(rtrim($t[0]));
   $t[1] = ltrim(rtrim($t[1]));

   if (check_parens($value, $global_parens) > -1) {
      continue;
   }
   $t[1] = check_minus($t[1], $global_parens);
   if ($t[1][1] != $global_error_names['NOERROR']) {
      continue;
   }
   $t[1] = $t[1][0];
   $t[1] = add_times($t[1], $poss_for_mult, '*');
   if ($t[1][1] != $global_error_names['NOERROR']) {
      continue;
   }
   $t[1] = $t[1][0];
   $t[1] = tokenize($t[1], $global_parens, $global_unary_operators,
                    $global_binary_operators);
   if ($t[1][1] != $global_error_names['NOERROR']) {
      continue;
   }
   $t[1] = $t[1][0];
   $t[1] = substitute($t[1], $global_constants);
   if ($t[1][1] != $global_error_names['NOERROR']) {
      continue;
   }
   $t[1] = $t[1][0];
   $t[1] = build_tree($t[1], $global_parens, $global_unary_operators, $global_binary_operators);
   if ($t[1][1] != $global_error_names['NOERROR']) {
      continue;
   }
   $t[1] = $t[1][0][0];
   $t[1] = build_stack($t[1]);
   if ($t[1][1] != $global_error_names['NOERROR']) {
      continue;
   }
   $t[1] = $t[1][0];
   $t[1] = process_stack($t[1], $global_unary_operators, $global_binary_operators);
   if ($t[1][1] != $global_error_names['NOERROR']) {
      continue;
   }

   $constants[$t[0]] = $t[1][0]; # Wenn bis hier noch kein Fehler aufgetreten
}                                # ist, speichern wir die Konstante

$constants = array_merge($constants, $global_constants);
   # und f"uhren sie mit den Standardkonstanten zusammen

# Jetzt holen wir uns die Funktionen und parsen die Terme in die Stacks

$functions = unpack_function_terms($_REQUEST);

foreach ($functions as $key => $values) {
   if (check_parens($values['term'], $global_parens) > -1) {
      unset($functions[$key]);
      continue;
   }
   $values['stack'] = check_minus($values['term'], $global_parens);
   if ($values['stack'][1] != $global_error_names['NOERROR']) {
      unset($functions[$key]);
      continue;
   }
   $values['stack'] = $values['stack'][0];
   $values['stack'] = add_times($values['stack'], $poss_for_mult, '*');
   if ($values['stack'][1] != $global_error_names['NOERROR']) {
      unset($functions[$key]);
      continue;
```

```
   }
   $values['stack'] = $values['stack'][0];
   $values['stack'] = tokenize($values['stack'], $global_parens, $global_unary_operators,
                   $global_binary_operators);
   if ($values['stack'][1] != $global_error_names['NOERROR']) {
      unset($functions[$key]);
      continue;
   }
   $values['stack'] = $values['stack'][0];
   $values['stack'] = substitute($values['stack'], $constants);
   if ($values['stack'][1] != $global_error_names['NOERROR']) {
      unset($functions[$key]);
      continue;
   }
   $values['stack'] = $values['stack'][0];
   $values['stack'] = build_tree($values['stack'], $global_parens, $global_unary_operators, $global_binary_operators);
   if ($values['stack'][1] != $global_error_names['NOERROR']) {
      unset($functions[$key]);
      continue;
   }
   $values['stack'] = $values['stack'][0][0];
   $values['stack'] = build_stack($values['stack']);
   if ($values['stack'][1] != $global_error_names['NOERROR']) {
      unset($functions[$key]);
      continue;
   }
   $values['stack'] = $values['stack'][0];

   $functions[$key] = $values;
   $functions[$key]['param']['from'] = check_error($values['param']['from'], 'e', false, '0');
   $functions[$key]['param']['to'] = check_error($values['param']['to'], 'e', false, '0');
   $functions[$key]['param']['step'] = check_error($values['param']['step'], 'e', false, '0');

   if ($functions[$key]['param']['from'] > $functions[$key]['param']['to']) {
      $temp                          = $functions[$key]['param']['from'];
      $functions[$key]['param']['from'] = $functions[$key]['param']['to'];
      $functions[$key]['param']['to']   = $temp;
   }
}

# Wir wollen mindestens eine Funktion haben, also ...

if (count($functions) == 0) {
   array_push($functions, array(
      'term'  => $plotter_standards{'func'}[1],
      'color' => $plotter_standards{'PL_color'}[1],
      'param' => array(
         'name' => 'NOPARAM',
         'from' => 0,
         'to'   => 0,
         'step' => 1
      )
   ));
   $key = 0;
   $values = $functions[0];
      if (check_parens($values['term'], $global_parens) > -1) {
         unset($functions[$key]);
         break;
      }
      $values['stack'] = check_minus($values['term'], $global_parens);
      if ($values['stack'][1] != $global_error_names['NOERROR']) {
         unset($functions[$key]);
         break;
      }
      $values['stack'] = $values['stack'][0];
      $values['stack'] = add_times($values['stack'], $poss_for_mult, '*');
      if ($values['stack'][1] != $global_error_names['NOERROR']) {
         unset($functions[$key]);
         break;
      }
      $values['stack'] = $values['stack'][0];
      $values['stack'] = tokenize($values['stack'], $global_parens, $global_unary_operators,
                      $global_binary_operators);
      if ($values['stack'][1] != $global_error_names['NOERROR']) {
         unset($functions[$key]);
         break;
      }
      $values['stack'] = $values['stack'][0];
      $values['stack'] = substitute($values['stack'], $constants);
      if ($values['stack'][1] != $global_error_names['NOERROR']) {
```

```
         unset($functions[$key]);
         break;
      }
      $values['stack'] = $values['stack'][0];
      $values['stack'] = build_tree($values['stack'], $global_parens, $global_unary_operators, $global_binary_operators);
      if ($values['stack'][1] != $global_error_names['NOERROR']) {
         unset($functions[$key]);
         break;
      }
      $values['stack'] = $values['stack'][0][0];
      $values['stack'] = build_stack($values['stack']);
      if ($values['stack'][1] != $global_error_names['NOERROR']) {
         unset($functions[$key]);
         break;
      }
      $values['stack'] = $values['stack'][0];

   $functions[$key] = $values;
   $functions[$key]['param']['from'] = check_error($values['param']['from'], 'e', false, '0');
   $functions[$key]['param']['to'] = check_error($values['param']['to'], 'e', false, '0');
   $functions[$key]['param']['step'] = check_error($values['param']['step'], 'e', false, '0');
}

# Das gleiche Procedere f"ur die Asymptoten, aber ohne Stacks ...

$asymptotes = unpack_asymptote_terms($_REQUEST);
foreach ($asymptotes as $key => $values) {
   $asymptotes[$key]['p1_x']   = check_error($values['p1_x'],   'e', false, $plotter_standards{'asym_px1'}[1], $constants);
   $asymptotes[$key]['p1_y']   = check_error($values['p1_y'],   'e', false, $plotter_standards{'asym_py1'}[1], $constants);
   $asymptotes[$key]['p2_x']   = check_error($values['p2_x'],   'e', false, $plotter_standards{'asym_px2'}[1], $constants);
   $asymptotes[$key]['p2_y']   = check_error($values['p2_y'],   'e', false, $plotter_standards{'asym_py2'}[1], $constants);
   $asymptotes[$key]['color']  = check_error($values['color'],  'c', false, $plotter_standards{'AS_color'}[1], $constants);
   $asymptotes[$key]['dashed'] = check_error($values['dashed'], 'b', false, $plotter_standards{'asym_dash'}[1], $constants);
}

# Der Rest der Werte

$x_min  = check_error($_REQUEST['x_min'],  'e', false, $plotter_standards{'x_min'}[1], $constants);
$x_max  = check_error($_REQUEST['x_max'],  'e', false, $plotter_standards{'x_max'}[1], $constants);
$x_grid = check_error($_REQUEST['x_grid'], 'e', false, $plotter_standards{'x_grid'}[1], $constants);

$y_min  = check_error($_REQUEST['y_min'],  'e', false, $plotter_standards{'y_min'}[1], $constants);
$y_max  = check_error($_REQUEST['y_max'],  'e', false, $plotter_standards{'y_max'}[1], $constants);
$y_grid = check_error($_REQUEST['y_grid'], 'e', false, $plotter_standards{'y_grid'}[1], $constants);

$width  = check_error($_REQUEST['width'],  'n', false, $plotter_standards{'img_width'}[1]);
$height = check_error($_REQUEST['height'], 'n', false, $plotter_standards{'img_height'}[1]);

$step   = check_error($_REQUEST['step'], 'e', false, $plotter_standards{'step'}[1], $constants);
$auto_step = check_error($_REQUEST['auto_step'], 'b', false,
                         $plotter_standards{'auto_step'}[1]);

$bg_color = check_error($_REQUEST['bg_col'], 'c', false, $plotter_standards{'BG_color'}[1]);
$pl_color = check_error($_REQUEST['pl_col'], 'c', false, $plotter_standards{'PL_color'}[1]);
$ax_color = check_error($_REQUEST['ax_col'], 'c', false, $plotter_standards{'AX_color'}[1]);
$gr_color = check_error($_REQUEST['gr_col'], 'c', false, $plotter_standards{'GR_color'}[1]);

$s_min  = check_error($_REQUEST['s_min'], 'e', false, $plotter_standards{'shading_min'}[1], $constants);
$s_max  = check_error($_REQUEST['s_max'], 'e', false, $plotter_standards{'shading_max'}[1], $constants);
$sh_color = check_error($_REQUEST['sh_col'], 'c', false, $plotter_standards{'SH_color'}[1]);
$s_trans  = check_error($_REQUEST['s_trans'], 'p', false, $plotter_standards{'shading_trans'}[1]);
$s_f1   = check_error($_REQUEST['s_f1'], 't', false, $plotter_standards{'shading_f1'}[1]);
$s_f2   = check_error($_REQUEST['s_f2'], 't', false, $plotter_standards{'shading_f2'}[1]);
$sf_color = check_error($_REQUEST['sf_col'], 'c', false, $plotter_standards{'SF_color'}[1]);
$do_shade = check_error($_REQUEST['do_shade'], 'b', false, $plotter_standards{'auto_step'}[1]);

# Umschichtung der Min-/Max-Werte

if ($x_min > $x_max) {
   $temp  = $x_min;
   $x_min = $x_max;
   $x_max = $temp;
}

if ($x_grid < 0) {
   $x_grid = -$x_grid;
}

if ($x_max == $x_min) {
   ++$x_max;
```

```
}

if ($y_min > $y_max) {
   $temp  = $y_min;
   $y_min = $y_max;
   $y_max = $temp;
}

if ($y_max == $y_min) {
   ++$y_max;
}

if ($y_grid < 0) {
   $y_grid = -$y_grid;
}

if ($step < 0) {
   $step = -$step;
}

if ($s_min > $s_max) {
   $temp  = $s_min;
   $s_min = $s_max;
   $s_max = $temp;
}

if ($x_max == $x_min) {
   ++$x_max;
}

# Berechnen der Aufl"osung

$res_x = $width  / ($x_max - $x_min);
$res_y = $height / ($y_max - $y_min);

# Jetzt geht's los

$plot = @ImageCreate($width, $height) or die; # erstellt ein Image-Objekt

# Farbdefinitionen

$color_res = array();
foreach ($global_colors as $num => $content) {
   $color_res[$num] = ImageColorAllocate($plot, $content[1], $content[2], $content[3]);
}

# Plot vorbereiten
$plot = init_plot(
   $plot,
   $width,
   $height,
   $x_min,
   $x_max,
   $x_grid,
   $y_min,
   $y_max,
   $y_grid,
   $color_res[$bg_color],
   $color_res[$ax_color],
   $color_res[$gr_color]
);

# Jetzt k"ummern wir uns um den schattierten Bereich

if ($do_shade == 'true') {

   # Funktion 1 in Stack umwandeln

   if (check_parens($s_f1, $global_parens) > -1) {
      $do_shade = 'false';
   }
   $s_f1 = check_minus($s_f1, $global_parens);
   if ($s_f1[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f1 = $s_f1[0];
   $s_f1 = add_times($s_f1, $poss_for_mult, '*');
   if ($s_f1[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
```

```
   $s_f1 = $s_f1[0];
   $s_f1 = tokenize($s_f1, $global_parens, $global_unary_operators,
                    $global_binary_operators);
   if ($s_f1[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f1 = $s_f1[0];
   $s_f1 = substitute($s_f1, $constants);
   if ($s_f1[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f1 = $s_f1[0];
   $s_f1 = build_tree($s_f1, $global_parens, $global_unary_operators, $global_binary_operators);
   if ($s_f1[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f1 = $s_f1[0][0];
   $s_f1 = build_stack($s_f1);
   if ($s_f1[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f1 = $s_f1[0];

   # Funktion 2 in Stack umwandeln

   if (check_parens($s_f2, $global_parens) > -1) {
      $do_shade = 'false';
   }
   $s_f2 = check_minus($s_f2, $global_parens);
   if ($s_f2[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f2 = $s_f2[0];
   $s_f2 = add_times($s_f2, $poss_for_mult, '*');
   if ($s_f2[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f2 = $s_f2[0];
   $s_f2 = tokenize($s_f2, $global_parens, $global_unary_operators,
                    $global_binary_operators);
   if ($s_f2[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f2 = $s_f2[0];
   $s_f2 = substitute($s_f2, $constants);
   if ($s_f2[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f2 = $s_f2[0];
   $s_f2 = build_tree($s_f2, $global_parens, $global_unary_operators, $global_binary_operators);
   if ($s_f2[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f2 = $s_f2[0][0];
   $s_f2 = build_stack($s_f2);
   if ($s_f2[1] != $global_error_names['NOERROR']) {
      $do_shade = 'false';
   }
   $s_f2 = $s_f2[0];
}

# Wenn die Funktionen korrekt geparst wurden, fahren wir fort

if ($do_shade == 'true') {

   # Zuerst generieren wir die Punkte

   if ($auto_step == 'true') {
      $points_sh_1 = auto_step(
                        $s_f1,
                        $s_min,
                        $s_max,
                        $y_min,
                        $y_max,
                        $res_x,
                        $res_y,
                        $constants
                     );
      if (count($points_sh_1) <= 1) {
         $do_shade = 'false';
```

```
      }
      $points_sh_2 = auto_step(
                        $s_f2,
                        $s_min,
                        $s_max,
                        $y_min,
                        $y_max,
                        $res_x,
                        $res_y,
                        $constants
                     );
      if (count($points_sh_2) <= 1) {
         $do_shade = 'false';
      }

      # Hier m"ussen wir aufpassen, dass wir f"ur jeden x-Wert beide y-Werte haben

      $all_sh_x = get_joined_indices($points_sh_1, $points_sh_2);
      $points_sh_1 = array_merge(
                        $points_sh_1,
                        generate_points(
                           $s_f1,
                           get_missing_indices($all_sh_x, $points_sh_1),
                           array()
                        )
                     );
      $points_sh_2 = array_merge(
                        $points_sh_2,
                        generate_points(
                           $s_f2,
                           get_missing_indices($all_sh_x, $points_sh_2),
                           array()
                        )
                     );
   } else {
      $points_sh_1 = fixed_step(
                        $s_f1,
                        $s_min,
                        $s_max,
                        $step,
                        $constants
                     );
      if (count($points_sh_1) <= 1) {
         $do_shade = 'false';
      }
      $points_sh_2 = fixed_step(
                        $s_f2,
                        $s_min,
                        $s_max,
                        $step,
                        $constants
                     );
      if (count($points_sh_2) <= 1) {
         $do_shade = 'false';
      }
   }

   # Wenn bis jetzt noch alles in Ordnung ist, k"onnen wir zeichnen

   if ($do_shade == 'true') {
      $tempimg = @ImageCreate($width, $height)
         or die;
      $tempimg = init_plot(
         $tempimg,
         $width,
         $height,
         $x_min,
         $x_max,
         $x_grid,
         $y_min,
         $y_max,
         $y_grid,
         ImageColorAllocate(
            $tempimg,
            $global_colors[$bg_color][1],
            $global_colors[$bg_color][2],
            $global_colors[$bg_color][3]
         ),
         ImageColorAllocate(
            $tempimg,
```

```
      $global_colors[$ax_color][1],
      $global_colors[$ax_color][2],
      $global_colors[$ax_color][3]
   ),
   ImageColorAllocate(
      $tempimg,
      $global_colors[$gr_color][1],
      $global_colors[$gr_color][2],
      $global_colors[$gr_color][3]
   )
);

# Alle Punkte in die richtige Reihenfolge bringen

ksort($points_sh_1);
reset($points_sh_1);
ksort($points_sh_2);
reset($points_sh_2);

$sh_color_res = $global_colors{$sh_color};
$sh_color_res = ImageColorAllocate($tempimg, $sh_color_res[1], $sh_color_res[2], $sh_color_res[3]);
$sf_color_res = $global_colors{$sf_color};
$sf_color_res = ImageColorAllocate($tempimg, $sf_color_res[1], $sf_color_res[2], $sf_color_res[3]);

$undef = true;
# Punkte: 0, 1 - alte 1; 2, 3 - neue 1; 4, 5 - neue 2; 6, 7 - alte 2
$points = array(0, 0, 0, 0, 0, 0, 0, 0); # Ecken des zu zeichnenden Vierecks

foreach ($points_sh_1 as $x => $y1) {
   $y2 = $points_sh_2{$x};
   if ($y1[1] == $global_error_names['NOERROR'] and $y2[1] == $global_error_names['NOERROR']) {
      if ($undef) {
         $undef = false;
         $points[2] = ($x - $x_min)     * ($width - 1)  / ($x_max - $x_min);
         $points[3] = ($y_max - $y1[0]) * ($height - 1) / ($y_max - $y_min);
         $points[4] = ($x - $x_min)     * ($width - 1)  / ($x_max - $x_min);
         $points[5] = ($y_max - $y2[0]) * ($height - 1) / ($y_max - $y_min);
      } else {
         # Die alten Punkte werden zu den linken Ecken ...
         $points[0] = $points[2];
         $points[1] = $points[3];
         $points[6] = $points[4];
         $points[7] = $points[5];
         # ... und die neuen zu den rechten
         $points[2] = ($x - $x_min)     * ($width - 1)  / ($x_max - $x_min);
         $points[3] = ($y_max - $y1[0]) * ($height - 1) / ($y_max - $y_min);
         $points[4] = ($x - $x_min)     * ($width - 1)  / ($x_max - $x_min);
         $points[5] = ($y_max - $y2[0]) * ($height - 1) / ($y_max - $y_min);
         ImageFilledPolygon(
            $tempimg,
            $points,
            4,
            $sh_color_res
         );
      }
   } else {
      $undef = true;
   }
}

# Jetzt m"ussen wir noch die Randfunktionen zeichnen

$tempimg = plot_graph(
              $tempimg,
              $width,
              $height,
              $x_min,
              $x_max,
              $y_min,
              $y_max,
              $points_sh_1,
              $sf_color_res
           );

$tempimg = plot_graph(
              $tempimg,
              $width,
              $height,
              $x_min,
              $x_max,
```

```
                    $y_min,
                    $y_max,
                    $points_sh_2,
                    $sf_color_res
                 );

      # Und f"ugen das neue Bild in das alte ein, so dass sich die gew"unschte
      # Transparenz ergibt -- nur m"oglich bevor irgend etwas anderes
      # gezeichnet ist

      ImageCopyMerge($plot, $tempimg, 0, 0, 0, 0, $width, $height, $s_trans);

      ImageDestroy($tempimg);
   }
}

# Danach die Asymptoten

foreach ($asymptotes as $values) {

   $x1 = $values['p1_x'];
   $y1 = $values['p1_y'];
   $x2 = $values['p2_x'];
   $y2 = $values['p2_y'];

   $dx = $x2 - $x1;
   $dy = $y2 - $y1;

   # Wenn die beiden Punkte indentisch sind, ist nichts zu machen

   if ($dx == 0 and $dy == 0) {
      continue;
   }

   # Wir dehnen dir Strecke bis zu den R"andern des Bildbereiches aus ...

   if ($dx != 0) {
      $tx = $x1;
      $ty = $y1;
      $x1 = $x_min;
      $y1 = $ty + $dy * ($x1 - $tx) / $dx;
      $x2 = $x_max;
      $y2 = $ty + $dy * ($x2 - $tx) / $dx;
   } else {
      $y1 = $y_min;
      $y2 = $y_max;
   }

   # ... und zeichen die Asymptote

   if ($values['dashed'] == 'true') {
      ImageDashedLine(
         $plot,
         ($x1 - $x_min) * ($width - 1)  / ($x_max - $x_min),
         ($y_max - $y1) * ($height - 1) / ($y_max - $y_min),
         ($x2 - $x_min) * ($width - 1)  / ($x_max - $x_min),
         ($y_max - $y2) * ($height - 1) / ($y_max - $y_min),
         $color_res[$values['color']]
      );
   } else {
      ImageLine(
         $plot,
         ($x1 - $x_min) * ($width - 1)  / ($x_max - $x_min),
         ($y_max - $y1) * ($height - 1) / ($y_max - $y_min),
         ($x2 - $x_min) * ($width - 1)  / ($x_max - $x_min),
         ($y_max - $y2) * ($height - 1) / ($y_max - $y_min),
         $color_res[$values['color']]
      );
   }
}

# Und zu guter Letzt die Funktionen

foreach ($functions as $key => $values) {
   if (in_array($values['param']['name'], $values['stack'])) {
      for ($param = $values['param']['from']; $param <= $values['param']['to']; $param += $values['param']['step']) {
         if ($auto_step == 'true') {
            $points = auto_step(
                        $values['stack'],
                        $x_min,
```

```
                        $x_max,
                        $y_min,
                        $y_max,
                        $res_x,
                        $res_y,
                        array_merge(
                           $constants,
                           array($values['param']['name'] => $param)
                        )
                     );
            if (count($points) <= 1) {
               continue;
            }
         } else {
            $points = fixed_step(
                        $values['stack'],
                        $x_min,
                        $x_max,
                        $step,
                        array_merge(
                           $constants,
                           array($values['param']['name'] => $param)
                        )
                     );
            if (count($points) <= 1) {
               continue;
            }
         }
         $plot = plot_graph(
                    $plot,
                    $width,
                    $height,
                    $x_min,
                    $x_max,
                    $y_min,
                    $y_max,
                    $points,
                    $color_res[$values['color']]
                 );
      }
   } else {
      if ($auto_step == 'true') {
         $points = auto_step(
                     $values['stack'],
                     $x_min,
                     $x_max,
                     $y_min,
                     $y_max,
                     $res_x,
                     $res_y,
                     $constants
                  );
         if (count($points) <= 1) {
            continue;
         }
      } else {
         $points = fixed_step(
                     $values['stack'],
                     $x_min,
                     $x_max,
                     $step,
                     $constants
                  );
         if (count($points) <= 1) {
            continue;
         }
      }
      $plot = plot_graph(
                 $plot,
                 $width,
                 $height,
                 $x_min,
                 $x_max,
                 $y_min,
                 $y_max,
                 $points,
                 $color_res[$values['color']]
              );
   }
}
```

```
   # Wir geben das Bild aus

   header($plotter_standards{'img_types'}[1][$image_type][2]);

   switch ($plotter_standards{'img_types'}[1][$image_type][1]) {
      case (IMG_GIF):
         print imagegif($plot);
         break;
      case (IMG_JPG):
         print imagejpg($plot);
         break;
      case (IMG_PNG):
         print imagepng($plot);
         break;
      case (IMG_WBMP):
         print imagewbmp($plot);
         break;
      default:
         die;
   }

   ImageDestroy($plot); # ... und um nett zu sein, den Speicher freigeben

?>
```

4.7 Datei „tabs/general.php“

```
<?php

   ###############################################################################
   # Dies ist die Datei 'general.php'                                            #
   #                                                                             #
   # Sie enth"alt ein Tab des Plotterformulars des Parser/Plotter-Projekts;      #
   # es enth"alt generelle Einstellungen                                         #
   #                                                                             #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-        #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt   #
   # werden:                                                                     #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert    #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf  #
   #      verwandt werden                                                        #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn    #
   #      verwenden, einsehbar gemacht werden                                    #
   #                                                                             #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in        #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus    #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen #
   # wollen, so kontaktieren Sie mich bitte.                                     #
   #                                                                             #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich   #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den #
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund    #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich   #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder #
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies             #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-   #
   # Adresse oder "uber den Sysadmin.                                            #
   #                                                                             #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses #
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
   # Nutzung auf eigene Gefahr.                                                  #
   #                                                                             #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der        #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,  #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug    #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                   #
   #                                                                             #
   # (c) Jan Olligs, 2004                                                        #
   # Alle Rechte vorbehalten                                                     #
   #                                                                             #
   # Jan.Olligs@gmx.net                                                          #
   ###############################################################################

   ###############################################################################
   # Parameter, die auf dieser Seite eingestellt werden:                         #
```

```
   #    'img_type', 'width', 'height'                                                #
   #    'x_min', 'x_max', 'x_grid', 'y_min', 'y_max', 'y_grid'                       #
   #    'step', 'auto_step'                                                          #
   #    'bg_col', 'ax_col', 'gr_col', 'pl_col'                                       #
   #    'const'                                                                      #
   ###############################################################################

?>

<fieldset>
   <legend>Bildeinstellungen</legend>
   <table border="0" width="100%">
      <tr>
         <td align="right"><?php print form_text($_REQUEST, 'width', 10, 40) ?></td>
         <td align="center"> &times; </td>
         <td align="left"><?php print form_text($_REQUEST, 'height', 10, 40) ?></td>
      </tr>
      <tr>
         <td align="right">Breite</td>
         <td align="center"> &times; </td>
         <td align="left">H&ouml;he</td>
      </tr>
      </table>
      <table>
      <tr>
         <td align="right">Format:</td>
         <td align="center"> </td>
         <td align="left">
<?php print form_img_type($_REQUEST, 'img_type', $plotter_standards, 12) ?>
         </td>
      </tr>
   </table>
</fieldset>

<fieldset>
   <legend>Dargestellter Bereich</legend>
   <table border="0" width="100%">
      <tr>
         <td width="50%">
            <fieldset>
               <legend>X<i>-Koordinate</i></legend>
               <table border="0" width="100%">
                  <tr>
                     <td>X-Untergrenze: </td>
                     <td align="right"><?php print form_text($_REQUEST, 'x_min', 10, 40) ?></td>
                  </tr>
                  <tr>
                     <td>X-Obergrenze: </td>
                     <td align="right"><?php print form_text($_REQUEST, 'x_max', 10, 40) ?></td>
                  </tr>
                  <tr>
                     <td>Abstand der X-Gitterlinien:</td>
                     <td align="right"><?php print form_text($_REQUEST, 'x_grid', 10, 40) ?></td>
                  </tr>
               </table>
            </fieldset>
         </td>
         <td width="50%">
            <fieldset>
               <legend>Y<i>-Koordinate</i></legend>
               <table border="0" width="100%">
                  <tr>
                     <td>Y-Untergrenze: </td>
                     <td align="right"><?php print form_text($_REQUEST, 'y_min', 10, 40) ?></td>
                  </tr>
                  <tr>
                     <td>Y-Obergrenze: </td>
                     <td align="right"><?php print form_text($_REQUEST, 'y_max', 10, 40) ?></td>
                  </tr>
                  <tr>
                     <td>Abstand der Y-Gitterlinien:</td>
                     <td align="right"><?php print form_text($_REQUEST, 'y_grid', 10, 40) ?></td>
                  </tr>
               </table>
            </fieldset>
         </td>
      </tr>
   </table>
</fieldset>
```

```
<fieldset>
   <legend>Zeicheneinstellungen</legend>
   <table border="0" width="100%">
      <tr>
         <td><input name="auto_step" type="radio" value="true"<?php if ($_REQUEST['auto_step'] == 'false') { print ' checked'; } ?>>&nb
         <td align="right"><?php print form_text($_REQUEST, 'step', 10, 40) ?></td>
      </tr>
      <tr>
         <td><input name="auto_step" type="radio" value="false"<?php if ($_REQUEST['auto_step'] == 'true') { print ' checked'; } ?>>&nb
         <td> </td>
      </tr>
   </table>
</fieldset>

<fieldset>
   <legend>Zeichenfarben</legend>
   <table border="0" width="100%">
      <tr>
         <td>Hintergrund: </td>
         <td>
<?php print form_color_select($_REQUEST, 'bg_col', $global_colors, 12) ?>
         </td>
      </tr>
      <tr>
         <td>Graph (Standard): </td>
         <td>
<?php print form_color_select($_REQUEST, 'pl_col', $global_colors, 12) ?>
         </td>
      </tr>
      <tr>
         <td>Achsen: </td>
         <td>
<?php print form_color_select($_REQUEST, 'ax_col', $global_colors, 12) ?>
         </td>
      </tr>
      <tr>
         <td>Gitterlinien: </td>
         <td>
<?php print form_color_select($_REQUEST, 'gr_col', $global_colors, 12) ?>
         </td>
      </tr>
   </table>
</fieldset>

<fieldset>
   <legend>Konstanten</legend>
   <p>Eingaben in der Form [name]=[Term], eine Zeile pro Konstante.</p>
   <textarea name="const" cols="50" rows="10"><?php
      print htmlspecialchars($_REQUEST['const']);

 ?></textarea>
</fieldset>

<?php
   print form_add_hidden_fields(
      $_REQUEST,
      array(
         '/^img_type$/',
         '/^width$/',
         '/^height$/',
         '/^x_min$/',
         '/^x_max$/',
         '/^x_grid$/',
         '/^y_min$/',
         '/^y_max$/',
         '/^y_grid$/',
         '/^step$/',
         '/^auto_step$/',
         '/^bg_col$/',
         '/^ax_col$/',
         '/^gr_col$/',
         '/^pl_col$/',
         '/^const$/'
      ),
      0
   )
?>

<table align="center">
   <tr>
```

```
      <td align="right"><input type="submit" value=" &Uuml;bernehmen "></td>
      <td align="left"><input type="reset" value=" Zur&uuml;cksetzen "></td>
   </tr>
</table>
```

4.8 Datei „tabs/functions.php“

```
<?php

   ##############################################################################
   # Dies ist die Datei 'functions.php'                                         #
   #                                                                            #
   # Sie enth"alt ein Tab des Plotterformulars des Parser/Plotter-Projekts;     #
   # es enth"alt Einstellungen f"ur Funktionen und Aymptoten                    #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen#
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den#
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund   #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich  #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder#
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies            #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-  #
   # Adresse oder "uber den Sysadmin.                                           #
   #                                                                            #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses#
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art.#
   # Nutzung auf eigene Gefahr.                                                 #
   #                                                                            #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der       #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht, #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug   #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                  #
   #                                                                            #
   # (c) Jan Olligs, 2004                                                       #
   # Alle Rechte vorbehalten                                                    #
   #                                                                            #
   # Jan.Olligs@gmx.net                                                         #
   ##############################################################################

   ##############################################################################
   # Parameter, die auf dieser Seite eingestellt werden:                        #
   #   's_min', 's_max', 'sh_col', 's_trans', 's_f1', 's_f2', 'sf_col', 'do_shade'#
   #    'function_[\d]_[...]', 'num_new_func'                                   #
   #    'asymptote_[\d]_[...]', 'num_new_asym'                                  #
   ##############################################################################

?>

<fieldset>
   <legend>Funktionen</legend>
<?php
   foreach ($functions as $num => $values) {
      print '<fieldset>' . "\n";
      print '   <legend>Funktion ' . ($num + 1) . '</legend>' . "\n";
      print '   <table border="0">' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">Funktion: </td>' . "\n";
      print '         <td align="right"><i>f(x)</i> = '
          . form_text($_REQUEST, "function_${num}_term", 30, 100) . '</td>' . "\n";
      print '      </tr>' . "\n";
```

```
      print '      <tr>' . "\n";
      print '         <td align="left">Parametername: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "function_${num}_param_name", 10, 5) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">Parameter von: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "function_${num}_param_from", 10, 40) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">Parameter bis: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "function_${num}_param_to", 10, 40) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">Parameter Schritte: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "function_${num}_param_step", 10, 40) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">Plotfarbe: </td>' . "\n";
      print '         <td align="right">' . "\n";
      print form_color_select($_REQUEST, "function_${num}_color", $global_colors, 12);
      print '         </td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="right">'
          . '<input type="checkbox" name="function_' . $num . '_delete" value="true">'
          . '</td>' . "\n";
      print '         <td align="left">Funktion l&ouml;schen</td>' . "\n";
      print '      </tr>' . "\n";

      print '   </table>' . "\n";
      print '</fieldset>' . "\n";
   }
?>
   <p>Neue Funktionen: <input type="text" size="10" maxlength="5" name="num_new_func" value="0"></p>
</fieldset>

<fieldset>
   <legend>Asymptoten</legend>
<?php
   foreach ($asymptotes as $num => $values) {
      print '<fieldset>' . "\n";
      print '   <legend>Asymptote ' . ($num + 1) . '</legend>' . "\n";
      print '   <table border="0">' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">x-Koordinate Punkt 1: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "asymptote_${num}_p1_x", 10, 40) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">y-Koordinate Punkt 1: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "asymptote_${num}_p1_y", 10, 40) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">x-Koordinate Punkt 2: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "asymptote_${num}_p2_x", 10, 40) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">y-Koordinate Punkt 2: </td>' . "\n";
      print '         <td align="right">'
          . form_text($_REQUEST, "asymptote_${num}_p2_y", 10, 40) . '</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="left">Farbe: </td>' . "\n";
      print '         <td align="right">' . "\n";
```

```
      print form_color_select($_REQUEST, "asymptote_${num}_color", $global_colors, 12);
      print '         </td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="right">'
          . '<input type="checkbox" name="asymptote_' . $num . '_dashed"'
          . ($_REQUEST["asymptote_${num}_dashed"] == 'true' ? ' checked' : '')
          . ' value="true"></td>' . "\n";
      print '         <td align="left">Gestrichelt zeichnen</td>' . "\n";
      print '      </tr>' . "\n";

      print '      <tr>' . "\n";
      print '         <td align="right">'
          . '<input type="checkbox" name="asymptote_' . $num . '_delete" value="true">'
          . '</td>' . "\n";
      print '         <td align="left">Asymptote l&ouml;schen</td>' . "\n";
      print '      </tr>' . "\n";

      print '   </table>' . "\n";
      print '</fieldset>' . "\n";
   }
?>
   <p>Neue Asymptoten: <input type="text" size="10" maxlength="5" name="num_new_asym" value="0"></p>
</fieldset>

<fieldset>
   <legend>Schattieren</legend>
   <table border="0">
      <tr>
         <td align="left">Unterer x-Wert: </td>
         <td align="right"><?php print form_text($_REQUEST, 's_min', 10, 40) ?></td>
      </tr>
      <tr>
         <td align="left">Oberer x-Wert: </td>
         <td align="right"><?php print form_text($_REQUEST, 's_max', 10, 40) ?></td>
      </tr>
      <tr>
         <td align="left">Schattenfarbe: </td>
         <td align="right">
<?php print form_color_select($_REQUEST, "sh_col", $global_colors, 12) ?>
         </td>
      </tr>
      <tr>
         <td align="left">Transparenz: </td>
         <td align="right"><?php print form_text($_REQUEST, 's_trans', 10, 40) ?> %</td>
      </tr>
      <tr>
         <td align="left">Grenzfunktion 1: </td>
         <td align="right"><i>g<sub>1</sub>(x)</i> = <?php print form_text($_REQUEST, 's_f1', 30, 100) ?></td>
      </tr>
      <tr>
         <td align="left">Grenzfunktion 2: </td>
         <td align="right"><i>g<sub>2</sub>(x)</i> = <?php print form_text($_REQUEST, 's_f2', 30, 100) ?></td>
      </tr>
      <tr>
         <td align="left">Begrenzungsfarbe: </td>
         <td align="right">
<?php print form_color_select($_REQUEST, "sf_col", $global_colors, 12) ?>
         </td>
      </tr>
      <tr>
         <td align="left"><input type="checkbox" name="do_shade"<?php print ($_REQUEST["do_shade"] == 'true' ? ' checked' : '')
         ?> value="true"> Schattieren</td>
         <td align="right"> </td>
      </tr>
   </table>
</fieldset>

<?php
   print form_add_hidden_fields(
      $_REQUEST,
      array(
         '/^s_min$/',
         '/^s_max$/',
         '/^sh_col$/',
         '/^s_trans$/',
         '/^s_f1$/',
         '/^s_f2$/',
         '/^sf_col$/',
```

```
         '/^do_shade$/',
         '/^function_/',
         '/^num_new_func$/',
         '/^asymptote_/',
         '/^num_new_asym$/'
      ),
      0
   )
?>

<table align="center">
   <tr>
      <td align="right"><input type="submit" value=" &Uuml;bernehmen "></td>
      <td align="left"><input type="reset" value=" Zur&uuml;cksetzen "></td>
   </tr>
</table>
```

4.9 Datei „tabs/error.php“

```
<?php

   ##############################################################################
   # Dies ist die Datei 'error.php'                                             #
   #                                                                            #
   # Sie enth"alt ein Tab des Plotterformulars des Parser/Plotter-Projekts;     #
   # es dient zur Fehlerausgabe                                                 #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen#
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den#
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund   #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich  #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder#
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies            #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-  #
   # Adresse oder "uber den Sysadmin.                                           #
   #                                                                            #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses#
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art.#
   # Nutzung auf eigene Gefahr.                                                 #
   #                                                                            #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der       #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht, #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug   #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                  #
   #                                                                            #
   # (c) Jan Olligs, 2004                                                       #
   # Alle Rechte vorbehalten                                                    #
   #                                                                            #
   # Jan.Olligs@gmx.net                                                         #
   ##############################################################################

?>

<h2>Fehler im Tab &quot;Einstellungen&quot;</h2>

<?php
   $noerr = true;

   # Bildeinstellungen "uberpr"ufen

   $error = check_error($_REQUEST['width'], 'n');
   if ($error != '') {
```

```
   $noerr = false;
   print "<p><b>Bild:</b> Fehler mit der Bildbreite: $error</p>\n";
}

$error = check_error($_REQUEST['height'], 'n');
if ($error != '') {
   $noerr = false;
   print "<p><b>Bild:</b> Fehler mit der Bildh&ouml;he: $error</p>\n";
}

# Konstanten parsen und "uberpr"ufen

$constants = array();
$temp = preg_split('/\n/', $_REQUEST['const']);
foreach ($temp as $eqn) {
   $t = preg_split('/=/', $eqn);
   if (!is_array($t) or count ($t) != 2) {
      continue;
   }
   $t[0] = ltrim(rtrim($t[0]));
   $t[1] = ltrim(rtrim($t[1]));

   $error = check_error($t[1], 'e', true, 0, $global_constants);
   if ($error != '') {
      $noerr = false;
      print "<p><b>Konstanten:</b> Fehler mit der Definition &quot;$eqn&quot;: $error</p>\n";
   } else {
      $constants[$t[0]] = check_error($t[1], 'e', false, 0, $global_constants);
   }
}
$constants = array_merge($constants, $global_constants);

# Bildbereich parsen und "uberpr"ufen

$value = $_REQUEST['x_min'];
$error = check_error($value, 'e', true, 0, $constants);
if ($error != '') {
   $noerr = false;
   print "<p><b>Bildbereich:</b> Fehler mit der Definition des kleinsten x-Wertes: $error</p>\n";
}

$value = $_REQUEST['x_max'];
$error = check_error($value, 'e', true, 0, $constants);
if ($error != '') {
   $noerr = false;
   print "<p><b>Bildbereich:</b> Fehler mit der Definition des gr&ouml;&szlig;ten x-Wertes: $error</p>\n";
}

$value = $_REQUEST['x_grid'];
$error = check_error($value, 'e', true, 0, $constants);
if ($error != '') {
   $noerr = false;
   print "<p><b>Bildbereich:</b> Fehler mit der Definition der x-Gitterweite: $error</p>\n";
}

$value = $_REQUEST['y_min'];
$error = check_error($value, 'e', true, 0, $constants);
if ($error != '') {
   $noerr = false;
   print "<p><b>Bildbereich:</b> Fehler mit der Definition des kleinsten y-Wertes: $error</p>\n";
}

$value = $_REQUEST['y_max'];
$error = check_error($value, 'e', true, 0, $constants);
if ($error != '') {
   $noerr = false;
   print "<p><b>Bildbereich:</b> Fehler mit der Definition des gr&ouml;&szlig;ten y-Wertes: $error</p>\n";
}

$value = $_REQUEST['y_grid'];
$error = check_error($value, 'e', true, 0, $constants);
if ($error != '') {
   $noerr = false;
   print "<p><b>Bildbereich:</b> Fehler mit der Definition der y-Gitterweite: $error</p>\n";
}

# Zeicheneinstellungen "uberpr"ufen

$value = $_REQUEST['auto_step'];
$error = check_error($value, 'b');
```

```
   if ($error != '') {
      $noerr = false;
      print "<p><b>Zeicheneinstellungen:</b> Fehler mit dem An- / Abschalten der automatischen Schrittweitensteuerung: $error</p>\n";
   }

   $value = $_REQUEST['step'];
   $error = check_error($value, 'e', true, 0, $constants);
   if ($error != '') {
      $noerr = false;
      print "<p><b>Zeicheneinstellungen:</b> Fehler mit der Definition der Schrittweite: $error</p>\n";
   }

   if ($noerr) {
      print "<p>Keine Fehler</p>\n";
   }

?>

<h2>Fehler im Tab &quot;Funktionen&quot;</h2>

<?php

   $noerr = true;

   foreach ($functions as $num => $values) {
      $error = check_error(
         $values['term'],
         't',
         true,
         0,
         array_merge(
            $constants,
            array($values['param']['name'] => 0)
               # Auf den Wert brauchen wir nicht zu achten, da nur der Term "uberpr"uft wird
         )
      );
      if ($error != '') {
         $noerr = false;
         print "<p><b>Funktion " . ($num + 1) . ":</b> Fehler mit der Definition des Funktionsterms: $error</p>\n";
      }

      $error = check_error($values['param']['from'], 'e', true, 0, $constants);
      if ($error != '') {
         $noerr = false;
         print "<p><b>Funktion " . ($num + 1) . ":</b> Fehler mit der Definition des Paramteranfangs: $error</p>\n";
      }

      $error = check_error($values['param']['to'], 'e', true, 0, $constants);
      if ($error != '') {
         $noerr = false;
         print "<p><b>Funktion " . ($num + 1) . ":</b> Fehler mit der Definition des Paramterendes: $error</p>\n";
      }

      $error = check_error($values['param']['step'], 'e', true, 0, $constants);
      if ($error != '') {
         $noerr = false;
         print "<p><b>Funktion " . ($num + 1) . ":</b> Fehler mit der Definition der Paramterschrittweite: $error</p>\n";
      }
   }

   foreach ($asymptotes as $num => $values) {
      $error = check_error($values['p1_x'], 'e', true, 0, $constants);
      if ($error != '') {
         $noerr = false;
         print "<p><b>Asymptote " . ($num + 1) . ":</b> Fehler mit der Definition des x-Wertes von Punkt 1: $error</p>\n";
      }

      $error = check_error($values['p1_y'], 'e', true, 0, $constants);
      if ($error != '') {
         $noerr = false;
         print "<p><b>Asymptote " . ($num + 1) . ":</b> Fehler mit der Definition des y-Wertes von Punkt 1: $error</p>\n";
      }

      $error = check_error($values['p2_x'], 'e', true, 0, $constants);
      if ($error != '') {
         $noerr = false;
         print "<p><b>Asymptote " . ($num + 1) . ":</b> Fehler mit der Definition des x-Wertes von Punkt 2: $error</p>\n";
      }

      $error = check_error($values['p2_y'], 'e', true, 0, $constants);
```

```
      if ($error != '') {
         $noerr = false;
         print "<p><b>Asymptote " . ($num + 1) . ":</b> Fehler mit der Definition des y-Wertes von Punkt 2: $error</p>\n";
      }

      $error = check_error($values['dashed'], 't');
      if ($error != '') {
         $noerr = false;
         print "<p><b>Asymptote " . ($num + 1) . ":</b> Fehler bei der Einstellung gestrichelt / nicht gestrichelt: $error</p>\n";
      }
   }

   # s-Parameter

   $error = check_error($_REQUEST['s_min'], 'e', true, 0, $constants);
   if ($error != '') {
      $noerr = false;
      print "<p><b>Schattierter Bereich:</b> Fehler mit der Definition des kleinsten x-Wertes: $error</p>\n";
   }

   $error = check_error($_REQUEST['s_max'], 'e', true, 0, $constants);
   if ($error != '') {
      $noerr = false;
      print "<p><b>Schattierter Bereich:</b> Fehler mit der Definition des gr&ouml;&szlig;ten x-Wertes: $error</p>\n";
   }

   $error = check_error($_REQUEST['s_f1'], 't', true, 0, $constants);
   if ($error != '') {
      $noerr = false;
      print "<p><b>Schattierter Bereich:</b> Fehler mit der Definition von Grenzfunktion 1: $error</p>\n";
   }

   $error = check_error($_REQUEST['s_f2'], 't', true, 0, $constants);
   if ($error != '') {
      $noerr = false;
      print "<p><b>Schattierter Bereich:</b> Fehler mit der Definition von Grenzfunktion 2: $error</p>\n";
   }

   $error = check_error($_REQUEST['do_shade'], 'b');
   if ($error != '') {
      $noerr = false;
      print "<p><b>Schattierter Bereich:</b> Fehler bei der Einstellung an / aus: $error</p>\n";
   }

   if ($noerr) {
      print "<p>Keine Fehler</p>\n";
   }

?>
```

4.10 Datei „tabs/graph.php“

```
<?php

   ##############################################################################
   # Dies ist die Datei 'graph.php'                                             #
   #                                                                            #
   # Sie enth"alt ein Tab des Plotterformulars des Parser/Plotter-Projekts;     #
   # es dient zur Ausgabe des Graphs                                            #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen#
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den#
```

```
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund         #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich        #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder      #
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies                  #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-        #
   # Adresse oder "uber den Sysadmin.                                                 #
   #                                                                                  #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses      #
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
   # Nutzung auf eigene Gefahr.                                                       #
   #                                                                                  #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der             #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,       #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug         #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                        #
   #                                                                                  #
   # (c) Jan Olligs, 2004                                                             #
   # Alle Rechte vorbehalten                                                          #
   #                                                                                  #
   # Jan.Olligs@gmx.net                                                               #
   ####################################################################################

   $width  = check_error($_REQUEST['width'],  'n', false, $plotter_standards{'img_width'}[1]);
   $height = check_error($_REQUEST['height'], 'n', false, $plotter_standards{'img_height'}[1]);

?>
<div align="center"><img src="graph.php?<?php
   print http_build_query($_REQUEST)
       . '" width="' . $width
       . '" height="' . $height;
?>"></div>
```

4.11 Datei „units/mathdef.php“

```
<?php

   ####################################################################################
   # Dies ist die Datei 'mathdef.php'                                                 #
   #                                                                                  #
   # Sie enth"alt Funktions- und "ahnliche Definitionen des Parser/Plotter-Projekts   #
   #                                                                                  #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-             #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt        #
   # werden:                                                                          #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert         #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf       #
   #      verwandt werden                                                             #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn         #
   #      verwenden, einsehbar gemacht werden                                         #
   #                                                                                  #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in             #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus         #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen      #
   # wollen, so kontaktieren Sie mich bitte.                                          #
   #                                                                                  #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich        #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den      #
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund         #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich        #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder      #
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies                  #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-        #
   # Adresse oder "uber den Sysadmin.                                                 #
   #                                                                                  #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses      #
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
   # Nutzung auf eigene Gefahr.                                                       #
   #                                                                                  #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der             #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,       #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug         #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                        #
   #                                                                                  #
   # (c) Jan Olligs, 2004                                                             #
   # Alle Rechte vorbehalten                                                          #
   #                                                                                  #
```

```
# Jan.Olligs@gmx.net                                                                    #
#########################################################################################

# Interne PHP-Begrenzungen

define("MAXSIZE",  100);       # Maximale Anzahl von Flie"skommastellen

# Variablen und Differentiale

define(  "FIRST_VAR",   "x");
define("D_FIRST_VAR",  "dx");
define(  "SECOND_VAR",  "y");
define("D_SECOND_VAR", "dy");

# Konstanten

$global_constants = array(
        'pi' => M_PI,
        'e'  => M_E
);

# Klammern im Format (("offnend, schlie"send), ("offnend, schlie"send), ...)

$global_parens = array(
        array('(', ')'),
        array('[', ']'),
        array('{', '}')
);


#########################################################################################
# Form f"ur Operatordefinitionen (un"ar):                                               #
#    [Symbol] => array(                                                                 #
#                'others'  => array([andere Symbole]),                                  #
#                'deriv'   => [Ableitung als parse tree],                               #
#                'name'    => [Name der Funktion],                                      #
#                'info'    => [Informationen, Erkl"arungen],                            #
#                'example' => [Beispiel(e)],                                            #
#                'eval'    => [Funktion zur Berechnung]                                 #
#                   )                                                                   #
#                                                                                       #
#    Parse tree Syntax:                                                                 #
#       <tree> = <node> ODER                                                            #
#                ('u', [un"arer Operator], <tree>) ODER                                 #
#                ('b', [bin"arer Operator], <tree>, <tree>)                             #
#       <node> = Wert (Zahl) ODER String mit Variable / Konstante                       #
#                                                                                       #
#    Zur Ableitung:                                                                     #
#       Un"are Funktionen f(u) werden nach der Kettenregel abgeleitet; das              #
#       argument ist die Konstante FIRST_VAR, seine Ableitung die Konstante             #
#       D_FIRST_VAR. Die Ableitung von ln(FIRST_VAR) ist also u' / u =                  #
#       D_FIRST_VAR / FIRST_VAR. Bin"are Operatoren werden als Funktion von zwei        #
#       von x abh"angigen Argumenten angesehen: f(u, v) = f(u(x), v(x)). Dabei          #
#       wird gesetzt: FIRST_VAR = u, D_FIRST_VAR = u', SECOND_VAR = v,                  #
#       D_SECOND_VAR = v'. u ist das erste, v das zweite Argument des Operators         #
#       bei seiner Auswertung.                                                          #
#       Zur Entwicklung einer Differenzierungfunktion ist nur zu sagen, dass sie        #
#       lediglich drei Regeln beherrschen muss: Die Ableitung einer Konstante ist       #
#       Null, die Ableitung der Variablen, nach der abgeleitet wird, eins, und          #
#       wenn der parse tree einer Funktion (b, op, links, rechts) lautet, m"ussen       #
#       zwei neue parse trees, u' = Ableitung(links) und v' = Ableitug(rechts),         #
#       bestimmt werden; au"serdem werden u = links und v = rechts gespeichert und      #
#       im Differential der Funktion dann FIRST_VAR durch u, D_FIRST_VAR durch u',      #
#       etc. ersetzt. Damit sind s"amtliche Ableitungsregeln schon in den Ablei-        #
#       tungen der Operatoren enthalten.                                                #
#                                                                                       #
#    Beispiel f"ur eval-Funktion:                                                       #
#       'eval'    => create_function('$value, &$error',                               #
#                        '$error = $global_error_names{'NOERROR'};                      #
#                         return -$value;'                                              #
#                     )                                                                 #
#    $error wird also als Referenz "ubergeben                                           #
#    Dei globale Variable $global_error_names aus parser.php kann verwandt werden       #
#########################################################################################

# Operatoren

$global_unary_operators  = array(
   'neg'   => array(
      'others'  => array('minus', 'inv'),
```

```
   'deriv'   => '-1',
   'name'    => 'Inverses',
   'info'    => "Das Inverse von x bezglich der Addition.",
   'example' => "neg(3) = -3",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return -$value;'
               )
),
'abs'   => array(
   'others'  => array('betr'),
   'deriv'   => array('b', '/', '1', array('u', 'sgn', FIRST_VAR)),
   'name'    => 'Betrag',
   'info'    => "Der Wert von x ohne Vorzeichen; abs(x) = x wenn x >= 0, sonst -x.",
   'example' => "abs(-2) = abs(2) = 2",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return abs($value);'
               )
),
'sgn'   => array(
   'others'  => array('sig', 'sg', 'sn'),
   'deriv'   => array('b', '/', '0', array('u', 'sgn', FIRST_VAR)),
   'name'    => 'Vorzeichen (Signum)',
   'info'    => "Das Vorzeichen von x; sgn(x) = 1 wenn x > 0, 0 wenn x = 0, sonst -1.",
   'example' => "sgn(-3) = -1; sgn(2) = 1; sgn(0) = 0",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 if ($value == 0) {
                    return 0;
                 } else if ($value > 0) {
                    return 1;
                 }
                 return -1;'
               )
),
'sqrt'  => array(
   'others'  => array('sroot', 'srt', 'qw'),
   'deriv'   => array('b', '/', '1', array('b', '*', '2', array('u', 'sqrt', FIRST_VAR))),
   'name'    => 'Quadratwurzel',
   'info'    => "Das nichtnegative Inverse der Quadratfunktion.",
   'example' => "sqrt(9) = 3",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if ($value < 0) {
                    $error = $global_error_names{\'OUTOFRANGE\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return sqrt($value);'
               )
),
'exp'   => array(
   'others'  => array(),
   'deriv'   => array('u', 'exp', FIRST_VAR),
   'name'    => 'Exponentialfunktion',
   'info'    => "e^x, wobei e fr die Eulersche Zahl (ungefhr 2.71828) steht.",
   'example' => "exp(0) = 1; exp(2) = 7.38906",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if ($value > MAXSIZE * log(10)) {
                    $error = $global_error_names{\'OUTOFRANGE\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return exp($value);'
               )
),
'ln'    => array(
   'others'  => array('logn'),
   'deriv'   => array('b', '/', '1', FIRST_VAR),
   'name'    => 'Natrlicher Logarithmus',
   'info'    => "Logarithmus zur Basis e, wobei e fr die Eulersche Zahl (ungefhr 2.71828) steht.",
   'example' => "ln(1) = 0; ln(5) = 1.60944",
```

```
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if ($value <= 0) {
                    $error = $global_error_names{\'OUTOFRANGE\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return log($value);'
              )
),
'lg'    => array(
   'others'  => array('logdec'),
   'deriv'   => array('b', '/', '1', array('b', '*', array('u', 'ln', '10'), FIRST_VAR)),
   'name'    => 'Dekadischer Logarithmus',
   'info'    => "Logarithmus zur Basis 10.",
   'example' => "lg(1) = 0; ln(5) = 0.69897",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if ($value <= 0) {
                    $error = $global_error_names{\'OUTOFRANGE\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return log10($value);'
              )
),
'ld'    => array(
   'others'  => array('logdual', 'lb', 'logbin'),
   'deriv'   => array('b', '/', '1', array('b', '*', array('u', 'lg', '2'), FIRST_VAR)),
   'name'    => 'Binrer Logarithmus (Logarithmus Dualis)',
   'info'    => "Logarithmus zur Basis 2.",
   'example' => "lg(1) = 0; ln(5) = 2.321928",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if ($value <= 0) {
                    $error = $global_error_names{\'OUTOFRANGE\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return log($value) / M_LN2;'
              )
),
'sin'   => array(
   'others'  => array(),
   'deriv'   => array('u', 'cos', FIRST_VAR),
   'name'    => 'Sinus',
   'info'    => "Im rechtwinkligen Dreieck: Gegenkathete durch Hypothenuse.",
   'example' => "sin(3) = 0.14112",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return sin($value);'
              )
),
'cos'   => array(
   'others'  => array(),
   'deriv'   => array('u', 'neg', array('u', 'sin', FIRST_VAR)),
   'name'    => 'Cosinus',
   'info'    => "Im rechtwinkligen Dreieck: Ankathete durch Hypothenuse.",
   'example' => "cos(3) = -0.98999",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return cos($value);'
              )
),
'sec'   => array(
   'others'  => array('secans'),
   'deriv'   => array('b', '/', array('u', 'neg', array('u', 'sin', FIRST_VAR)), array('b', '^', array('u', 'cos', FIRST_VAR), '2
   'name'    => 'Secans',
   'info'    => "sec(x) = 1 / cos(x)",
   'example' => "sec(3) = -1.01011",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '$value = cos($value);
                 if ($value == 0) {
                    $error = $global_error_names{\'DIVBYZERO\'};
                    return;
```

```
                    }
                    $error = $global_error_names{\'NOERROR\'};
                    return 1 / $value;'
                )
    ),
    'csc'   => array(
       'others'  => array('cosec', 'cosecans'),
       'deriv'   => array('b', '/', array('u', 'cos', FIRST_VAR), array('b', '^', array('u', 'sin', FIRST_VAR), '2')),
       'name'    => 'Cosecans',
       'info'    => "csc(x) = 1 / sin(x)",
       'example' => "csc(3) = 7.08617",
       'eval'    => create_function('$value, &$error',
                     'global $global_error_names;' . "\n" .
                     '$value = sin($value);
                      if ($value == 0) {
                         $error = $global_error_names{\'DIVBYZERO\'};
                         return;
                      }
                      $error = $global_error_names{\'NOERROR\'};
                      return 1 / $value;'
                   )
    ),
    'tan'   => array(
       'others'  => array('tangens'),
       'deriv'   => array('b', '1', array('b', '^', array('u', 'cos', FIRST_VAR), '2')),
       'name'    => 'Tangens',
       'info'    => "tan(x) = sin(x) / cos(x)",
       'example' => "tan(3) = -0.14255",
       'eval'    => create_function('$value, &$error',
                     'global $global_error_names;' . "\n" .
                     'if (cos($value) == 0) {
                         $error = $global_error_names{\'DIVBYZERO\'};
                         return;
                      }
                      $error = $global_error_names{\'NOERROR\'};
                      return tan($value);'
                   )
    ),
    'cot'   => array(
       'others'  => array('cotan', 'cotangens'),
       'deriv'   => array('b', '/', '-1', array('b', '^', array('u', 'sin', FIRST_VAR), '2')),
       'name'    => 'Cotangens',
       'info'    => "cot(x) = cos(x) / sin(x)",
       'example' => "cot(3) = -7.01525",
       'eval'    => create_function('$value, &$error',
                     'global $global_error_names;' . "\n" .
                     'if (sin($value) == 0) {
                         $error = $global_error_names{\'DIVBYZERO\'};
                         return;
                      }
                      $error = $global_error_names{\'NOERROR\'};
                      return 1 / tan($value);'
                   )
    ),
    'asin'  => array(
       'others'  => array('arcsin'),
       'deriv'   => array('b', '/', '1', array('u', 'sqrt', array('b', '-', '1', array('b', '^', FIRST_VAR, '2')))),
       'name'    => 'Arcus sinus',
       'info'    => "Umkehrfunktion des Sinus.",
       'example' => "asin(0.5) = 0.52360",
       'eval'    => create_function('$value, &$error',
                     'global $global_error_names;' . "\n" .
                     'if (abs($value) > 1) {
                         $error = $global_error_names{\'DIVBYZERO\'};
                         return;
                      }
                      $error = $global_error_names{\'NOERROR\'};
                      return asin($value);'
                   )
    ),
    'acos'  => array(
       'others'  => array('arccos'),
       'deriv'   => array('b', '/', '-1', array('u', 'sqrt', array('b', '-', '1', array('^', FIRST_VAR, '2')))),
       'name'    => 'Arcus cosinus',
       'info'    => "Umkehrfunktion des Cosinus.",
       'example' => "acos(0.5) = 1.04720",
       'eval'    => create_function('$value, &$error',
                     'global $global_error_names;' . "\n" .
                     'if (abs($value) > 1) {
                         $error = $global_error_names{\'DIVBYZERO\'};
```

```
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return acos($value);'
              )
   ),
   'asec'  => array(
      'others'  => array('asecans', 'arcsec', 'arcsecans'),
      'deriv'   => array('b' , '/', '1', array('b', '*', array('b', '^', FIRST_VAR, '2'), array('u', 'sqrt', array('b', '-', '1', ar
      'name'    => 'Arcus secans',
      'info'    => "Umkehrfunktion des Secans.",
      'example' => "asec(3) = 1.23096",
      'eval'    => create_function('$value, &$error',
           'global $global_error_names;' . "\n" .
           'if ($value == 0) {
               $error = $global_error_names{\'DIVBYZERO\'};
               return;
            }
            if (abs($value) < 1) {
               $error = $global_error_names{\'OUTOFRANGE\'};
               return;
            }
            $error = $global_error_names{\'NOERROR\'};
            return acos(1 / $value);'
         )
   ),
   'acsc'  => array(
      'others'  => array('acosecans', 'arccsc', 'arccosecans'),
      'deriv'   => array('b', '/', '-1', array('b', '*', array('b', '^', FIRST_VAR, '2'), array('u', 'sqrt', array('b', '-', '1', ar
      'name'    => 'Arcus cosecans',
      'info'    => "Umkehrfunktion des Cosecans.",
      'example' => "asec(3) = 0.33984",
      'eval'    => create_function('$value, &$error',
           'global $global_error_names;' . "\n" .
           'if ($value == 0) {
               $error = $global_error_names{\'DIVBYZERO\'};
               return;
            }
            if (abs($value) < 1) {
               $error = $global_error_names{\'OUTOFRANGE\'};
               return;
            }
            $error = $global_error_names{\'NOERROR\'};
            return asin(1 / $value);'
         )
   ),
   'atan'  => array(
      'others'  => array('atangens', 'arctan', 'arctangens'),
      'deriv'   => array('b', '/', '1', array('b', '+', '1', array('b', '^', FIRST_VAR, '2'))),
      'name'    => 'Arcus tangens',
      'info'    => "Umkehrfunktion des Tangens.",
      'example' => "atan(3) = 1.249046",
      'eval'    => create_function('$value, &$error',
           'global $global_error_names;' . "\n" .
           '
            $error = $global_error_names{\'NOERROR\'};
            return atan($value);'
         )
   ),
   'acot'  => array(
      'others'  => array('acotangent', 'arccot', 'arccotangent'),
      'deriv'   => array('b', '/', '-1', array('b', '+', '1', array('b', '^', FIRST_VAR, '2'))),
      'name'    => 'Arcus cotangens',
      'info'    => "Umkehrfunktion des Tangens.",
      'example' => "acot(3) = 0.32175",
      'eval'    => create_function('$value, &$error',
           'global $global_error_names;' . "\n" .
           '
            $error = $global_error_names{\'NOERROR\'};
            if ($value == 0) {
               return M_PI_2;
            }
            return atan(1 / $value);'
         )
   ),
   'sinh'   => array(
      'others'  => array(),
      'deriv'   => array('u', 'cosh', FIRST_VAR),
      'name'    => 'Sinus hyperbolicus',
      'info'    => "sinh(x) = (e^x - e^(-x)) / 2",
```

```
   'example' => "sinh(3) = 10.01787",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return sinh($value);'
              )
),
'cosh'   => array(
   'others'  => array(),
   'deriv'   => array('u', 'sinh', FIRST_VAR),
   'name'    => 'Cosinus hyperbolicus',
   'info'    => "cosh(x) = (e^x + e^(-x)) / 2",
   'example' => "cosh(3) = 10.06766",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return cosh($value);'
              )
),
'sech'  => array(
   'others'  => array(),
   'deriv'   => array('u', 'neg', array('b', '/', array('u', 'sinh', FIRST_VAR), array('b', '^', array('u', 'cosh', FIRST_VAR), '
   'name'    => 'Secans hyperbolicus',
   'info'    => "sech(x) = 1 / cosh(x)",
   'example' => "sech(3) = 0.09933",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '$value = cosh($value);
                 if ($value == 0) {
                    $error = $global_error_names{\'DIVBYZERO\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return 1 / $value;'
              )
),
'csch'  => array(
   'others'  => array(),
   'deriv'   => array('u', 'neg', array('b', '/', array('u', 'cosh', FIRST_VAR), array('b', '^', array('u', 'sinh', FIRST_VAR), '
   'name'    => 'Hyperbolic cosecant',
   'info'    => "csch(x) = 1 / sinh(x)",
   'example' => "csch(3) = 0.099822",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '$value = sinh($value);
                 if ($value == 0) {
                    $error = $global_error_names{\'DIVBYZERO\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return 1 / $value;'
              )
),
'tanh'  => array(
   'others'  => array(),
   'deriv'   => array('b', '/', '1', array('b', '^', array('u', 'cosh', FIRST_VAR), '2')),
   'name'    => 'Tangens hyperbolicus',
   'info'    => "tanh(x) = sinh(x) / cosh(x)",
   'example' => "tanh(3) = 0.995055",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if (cosh($value) == 0) {
                    $error = $global_error_names{\'DIVBYZERO\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return tanh($value);'
              )
),
'coth'  => array(
   'others'  => array(),
   'deriv'   => array('b', '/', '-1', array('b', '^', array('u', 'sinh', FIRST_VAR), '2')),
   'name'    => 'Cotangens hyperbolicus',
   'info'    => "coth(x) = cosh(x) / sinh(x)",
   'example' => "coth(3) = 1.004970",
   'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if (sinh($value) == 0) {
```

```
                    $error = $global_error_names{\'DIVBYZERO\'};
                    return;
                }
                $error = $global_error_names{\'NOERROR\'};
                return 1 / tanh($value);'
            )
    ),
    'asinh' => array(
        'others'  => array('arsinh', 'areasinh'),
        'deriv'   => array('b', '/', '1', array('u', 'sqrt', array('b', '+', '1', array('b', '^', FIRST_VAR, '2')))),
        'name'    => 'Area sinus hyperbolicus',
        'info'    => "Umkehrfunktion des Sinus hyperbolicus.",
        'example' => "asinh(3) = 1.818446",
        'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return asinh($value);'
            )
    ),
    'acosh' => array(
        'others'  => array('arcosh', 'areacosh'),
        'deriv'   => array('b', '/', '1', array('u', 'sqrt', array('b', '-', array('b', '^', FIRST_VAR, '2'), '1'))),
        'name'    => 'Area cosinus hyperbolicus',
        'info'    => "Umkehrfunktion des Cosinus hyperbolicus.",
        'example' => "acosh(3) = 1.762747",
        'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                'if ($value < 1) {
                    $error = $global_error_names{\'OUTOFRANGE\'};
                    return;
                 }
                 $error = $global_error_names{\'NOERROR\'};
                 return acosh($value);'
            )
    ),
    'floor'  => array(
        'others'  => array('rounddown', 'abrunden'),
        'deriv'   => array('b', '/', '0', array('b', '-', FIRST_VAR, array('u', 'floor', FIRST_VAR))),
        'name'    => 'Floor',
        'info'    => "Die grte ganze Zahl, die kleiner oder gleich dem Argment ist.",
        'example' => "floor(-4.35) = -5; floor(4.35) = 4; floor(-4) = -4",
        'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return floor($value);'
            )
    ),
    'ceil'  => array(
        'others'  => array('roundup', 'aufrunden'),
        'deriv'   => array('b', '/', '0', array('b', '-', FIRST_VAR, array('u', 'floor', FIRST_VAR))),
        'name'    => 'Ceiling',
        'info'    => "Die kleinste ganze Zahl, die grer oder gleich dem Argument ist.",
        'example' => "floor(-4.35) = -4; floor(4.35) = 5; floor(-4) = -4",
        'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return ceil($value);'
            )
    ),
    'round' => array(
        'others'  => array('runden'),
        'deriv'   => array('b', '/', '0', array('b', '-', array('b', '+', FIRST_VAR, '0.5'), array('u', 'floor', array('b', '+', FIRST
        'name'    => 'Runden',
        'info'    => "Rundet das Argument auf die nchste ganze Zahl.",
        'example' => "round(-4.35) = -4; round(4.75) = 5; round(-4) = -4",
        'eval'    => create_function('$value, &$error',
                'global $global_error_names;' . "\n" .
                '
                 $error = $global_error_names{\'NOERROR\'};
                 return round($value);'
            )
    ),
    'frac'  => array(
        'others'  => array('fractional'),
        'deriv'   => array('b', '/', array('b', '-', FIRST_VAR, array('u', 'int', FIRST_VAR)), array('b', '-', FIRST_VAR, array('u', '
        'name'    => 'Nachkommaanteil',
        'info'    => "Gibt den Nachkommanteil des Arguments zurck.",
```

```
      'example' => "frac(-4.35) = -0.35; frac(4.35) = 0.35; frac(-4) = 0",
      'eval'    => create_function('$value, &$error',
                     'global $global_error_names;' . "\n" .
                     '
                      $error = $global_error_names{\'NOERROR\'};
                      return abs($value) - floor(abs($value));'
                   )
   ),
   'int'   => array(
      'others'  => array('integral', 'trunc'),
      'deriv'   => array('b', '/', '0', array('b', '^', array('b', '-', FIRST_VAR, array('u', 'floor', FIRST_VAR)), FIRST_VAR)),
      'name'    => 'Ganzer Anteil',
      'info'    => "Schneidet den Nachkommanteil des Arguments ab.",
      'example' => "int(-4.35) = -4; int(4.35) = 4; int(-4) = -4",
      'eval'    => create_function('$value, &$error',
                     'global $global_error_names;' . "\n" .
                     '
                      $error = $global_error_names{\'NOERROR\'};
                      if ($value > 0) {
                         return floor($value);
                      } else {
                         return ceil($value);
                      }'
                   )
      )
);

##############################################################################
# Form f"ur Operatordefinitionen (bin"ar):                                   #
#    [Symbol] => array(                                                      #
#                'deriv'   => [Ableitung als parse tree],                    #
#                'name'    => [Name der Funktion],                           #
#                'info'    => [Informationen, Erkl"arungen],                 #
#                'example' => [Beispiel(e)],                                 #
#                'eval'    => [Funktion zur Berechnung]                      #
#                   )                                                       #
#                                                                           #
#    Zur Ableitung:                                                         #
#       Un"are Funktionen f(u) werden nach der Kettenregel abgeleitet; das  #
#       argument ist die Konstante FIRST_VAR, seine Ableitung die Konstante #
#       D_FIRST_VAR. Die Ableitung von ln(FIRST_VAR) ist also u' / u =       #
#       D_FIRST_VAR / FIRST_VAR. Bin"are Operatoren werden als Funktion von zwei #
#       von x abh"angigen Argumenten angesehen: f(u, v) = f(u(x), v(x)). Dabei #
#       wird gesetzt: FIRST_VAR = u, D_FIRST_VAR = u', SECOND_VAR = v,      #
#       D_SECOND_VAR = v'. u ist das erste, v das zweite Argument des Operators #
#       bei seiner Auswertung.                                              #
#       Zur Entwicklung einer Differenzierungfunktion ist nur zu sagen, dass sie #
#       lediglich drei Regeln beherrschen muss: Die Ableitung einer Konstante ist #
#       Null, die Ableitung der Variablen, nach der abgeleitet wird, eins, und #
#       wenn der parse tree einer Funktion (b, op, links, rechts) lautet, m"ussen #
#       zwei neue parse trees, u' = Ableitung(links) und v' = Ableitug(rechts), #
#       bestimmt werden; au"serdem werden u = links und v = rechts gespeichert und #
#       im Differential der Funktion dann FIRST_VAR durch u, D_FIRST_VAR durch u', #
#       etc. ersetzt. Damit sind s"amtliche Ableitungsregeln schon in den Ablei- #
#       tungen der Operatoren enthalten.                                    #
#                                                                           #
#    Parse tree Syntax:                                                     #
#       <tree> = <node> ODER                                                #
#                ('u', [un"arer Operator], <tree>) ODER                     #
#                ('b', [bin"arer Operator], <tree>, <tree>)                 #
#       <node> = Wert (Zahl) ODER String mit Variable / Konstante           #
#                                                                           #
#    Beispiel f"ur eval-Funktion:                                           #
#       'eval'    => create_function('$value, &$error',                     #
#                      '$error = $global_error_names{'NOERROR'};            #
#                       return -$value;'                                    #
#                    )                                                      #
#    $error wird also als Referenz "ubergeben                               #
#    Dei globale Variable $global_error_names aus parser.php kann verwandt werden #
##############################################################################

$global_binary_operators = array(
   array( # Pr"azedenz 1 => h"ochste
      '^'    => array(
         'deriv'   => array('b', '*', array('b', '^', FIRST_VAR, SECOND_VAR), array('b', '+', array('b', '*', array('b', '/', D_FIRS
         'name'    => 'Potenz',
         'info'    => "keine Information vorhanden",
         'example' => "3^0 = 1; 3^1 = 1; 3^2 = 9",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
```

```
                        'if (log10(abs($value1)) * $value2 > MAXSIZE) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
                         } else if ($value1 < 0) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
                         } else {
                            if ($value2 < 0 && $value1 == 0) {
                               $error = $global_error_names{\'DIVBYZERO\'};
                               return;
                            }
                            $error = $global_error_names{\'NOERROR\'};
                            return pow($value1, $value2);
                         }'
                      )
      ),
      'root' => array(
         'deriv'   => array('b', '*', array('b', 'root', FIRST_VAR, SECOND_VAR), array('b', '/', array('b', '-', array('b', '*', arr
         'name'    => 'Wurzel',
         'info'    => "a root b steht fr die a-te Wurzel von b, per Definition gleich b^(1/a)",
         'example' => "3^0 = 1; 3^1 = 1; 3^2 = 9",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if ($value1 == 0) {
                            $error = $global_error_names{\'DIVBYZERO\'};
                            break;
                         }
                         $value = $value2;
                         $value2 = 1 / $value1;
                         $value1 = $value;
                         if (log10(abs($value1)) * $value2 > MAXSIZE) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
                         } else if ($value1 < 0) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
                         } else {
                            if ($value2 < 0 && $value1 == 0) {
                               $error = $global_error_names{\'DIVBYZERO\'};
                               return;
                            }
                            $error = $global_error_names{\'NOERROR\'};
                            return pow($value1, $value2);
                         }'
                      )
      ),
      'log'  => array(
         'deriv'   => array('b', '/', array('b', '-', array('b', '*', array('b', '/', D_SECOND_VAR, SECOND_VAR), array('u', 'ln', FI
         'name'    => 'Logarithmus',
         'info'    => "a log b steht fr den Logarithmus von b zur Basis a.",
         'example' => "3 log 5 = 1.4649735",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if ($value1 <= 0 or log($value1) == 0) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
                         }
                         if ($value2 <= 0) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
                         }
                         $error = $global_error_names{\'NOERROR\'};
                         return log($value2) / log($value1);'
                      )
      )
   ),
   array( # Pr"azedenz 2
      '%'    => array(
         'deriv'   => array(), # schwierig, da st"uckweise stetig
         'name'    => 'Modulus',
         'info'    => "a % b = r ist das eindeutig bestimmte r mit 0 <= abs(r) < abs(b), sgn(r) = sgn(b) oder r = 0 und (a - r) teil
         'example' => "5 % 3 = 2",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if ($value2 == 0) {
                            $error = $global_error_names{\'DIVBYZERO\'};
                            return;
                         } else if ($value2 < 0) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
```

```
                        } else {
                           $error = $global_error_names{\'NOERROR\'};
                           return fmod($value1, $value2);
                        }'
                     )
      ),
      'mod'  => array(
         'deriv'   => array(), # schwierig, da st"uckweise stetig
         'name'    => 'Modulus',
         'info'    => "a mod b = r ist das eindeutig bestimmte r mit 0 <= abs(r) < abs(b), sgn(r) = sgn(b) oder r = 0 und (a - r) te
         'example' => "5 mod 3 = 2",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if ($value2 == 0) {
                            $error = $global_error_names{\'DIVBYZERO\'};
                            return;
                         } else if ($value2 < 0) {
                            $error = $global_error_names{\'OUTOFRANGE\'};
                            return;
                         } else {
                            $error = $global_error_names{\'NOERROR\'};
                            return fmod($value1, $value2);
                        }'
                     )
      ),
      'div'  => array(
         'deriv'   => array(), # schwierig, da st"uckweise stetig (Ableitung von int(FIST_VAR / SECOND_VAR); Kettenregel; s.o.)
         'name'    => 'Ganzzahliger Anteil bei Division',
         'info'    => "a div b ist der ganzzahlige Anteil des Quotients dieser beiden Zahlen",
         'example' => "5 div 3 = 1",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if ($value2 == 0) {
                            $error = $global_error_names{\'DIVBYZERO\'};
                            return;
                         } else {
                            $i = 0;
                            if ($value1 / $value2 < 0) {
                               while (abs($value1) > abs($value2)) { $value1 += $value2; --$i;}
                            } else {
                               while (abs($value1) > abs($value2)) { $value1 -= $value2; ++$i;}
                            }
                            $error = $global_error_names{\'NOERROR\'};
                            return $i;
                        }'
                     )
      ),
   ),
   array( # Pr"azedenz 3
      '*'    => array(
         'deriv'   => array('b', '+', array('b', '*', D_FIRST_VAR, SECOND_VAR), array('b', '*', FIRST_VAR, D_SECOND_VAR)),
         'name'    => 'Multiplikation',
         'info'    => "keine Information vorhanden",
         'example' => "5 * 3 = 15",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if (!($value1 == 0 || $value2 == 0)) {
                            if (log10(abs($value1)) + log10(abs($value2)) > MAXSIZE) {
                               $error = $global_error_names{\'OUTOFRANGE\'};
                               return;
                            }
                         }
                         $error = $global_error_names{\'NOERROR\'};
                         return $value1 * $value2;'
                     )
      ),
      '/'    => array(
         'deriv'   => array('b', '/', array('b', '-', array('b', '*', D_FIRST_VAR, SECOND_VAR), array('b', '*', FIRST_VAR, D_SECOND_
         'name'    => 'Division',
         'info'    => "keine Information vorhanden",
         'example' => "5 / 3 = 1.66667",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if ($value2 == 0) {
                            $error = $global_error_names{\'DIVBYZERO\'};
                            return;
                         }
                         $value2 = 1 / $value2;
                         if (!($value1 == 0 || $value2 == 0)) {
                            if (log10(abs($value1)) + log10(abs($value2)) > MAXSIZE) {
```

```
                                $error =$global_error_names{\' OUTOFRANGE\'};
                                return;
                             }
                          }
                          $error = $global_error_names{\'NOERROR\'};
                          return $value1 * $value2;'
                       )
      )
   ),
   array( # Pr"azedenz 4 => niedrigste
      '+'    => array(
         'deriv'   => array('b', '+', D_FIRST_VAR, D_SECOND_VAR),
         'name'    => 'Addition',
         'info'    => "keine Information vorhanden",
         'example' => "5 + 3 = 8",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        'if (!($value1 == 0 || $value2 == 0) && $value1 / $value2 > 0) {
                            if (abs(log10(abs($value1)) - log10(abs($value2))) < 0.01) {
                               if (2 * log10(abs($value1)) > MAXSIZE) {
                                  $error = $global_error_names{\'OUTOFRANGE\'};
                                  return;
                               }
                            }
                          }
                          $error = $global_error_names{\'NOERROR\'};
                          return $value1 + $value2;'
                       )
      ),
      '-'    => array(
         'deriv'   => array('b', '-', D_FIRST_VAR, D_SECOND_VAR),
         'name'    => 'Subtraktion',
         'info'    => "keine Information vorhanden",
         'example' => "5 - 3 = 2",
         'eval'    => create_function('$value1, $value2, &$error',
                        'global $global_error_names;' . "\n" .
                        '$value2 = -$value2;
                         if (!($value1 == 0 || $value2 == 0) && $value1 / $value2 > 0) {
                            if (abs(log10(abs($value1)) - log10(abs($value2))) < 0.01) {
                               if (2 * log10(abs($value1)) > MAXSIZE) {
                                  $error = $global_error_names{\'OUTOFRANGE\'};
                                  return;
                               }
                            }
                          }
                          $error = $global_error_names{\'NOERROR\'};
                          return $value1 + $value2;'
                       )
      )
   )
);

?>
```

4.12 Datei „units/parser.php“

```
<?php

   ##############################################################################
   # Dies ist die Datei 'parser.php'                                            #
   #                                                                            #
   # Sie enth"alt den Parser des Parser/Plotter-Projekts                        #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen #
   # wollen, so kontaktieren Sie mich bitte.                                    #
```

```
#                                                                                #
# Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich      #
# dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den    #
# Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund       #
# auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich      #
# mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder    #
# den Urheber geistigen Eigentums vorenthalten, so geschieht dies                #
# unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-      #
# Adresse oder "uber den Sysadmin.                                               #
#                                                                                #
# Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses    #
# entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
# Nutzung auf eigene Gefahr.                                                     #
#                                                                                #
# Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der           #
# Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,     #
# den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug       #
# Zeit, um E-Mails "uber technische Details zu beantworten.                      #
#                                                                                #
# (c) Jan Olligs, 2004                                                           #
# Alle Rechte vorbehalten                                                        #
#                                                                                #
# Jan.Olligs@gmx.net                                                             #
##################################################################################

##################################################################################
#                              Fehlerdeklarationen (global)                      #
##################################################################################

$global_errors = array(
   1 => 'NOERROR',
   2 => 'ANYERROR',
   3 => 'DIVBYZERO',
   4 => 'OUTOFRANGE',
   5 => 'NOSUCHFUNC',
   6 => 'PARSEERR',
   7 => 'NOSUCHTYPE',
   8 => 'NOTANUMBER'
);

$global_error_names = array_flip($global_errors);

##################################################################################
# Funktion  : error_text                                                         #
# Zweck     : Wandelt Fehlernummer in Fehlertext um                              #
# Eingabe   : Fehlernummer         - $err_num                                    #
# Ausgabe   : Fehlermeldung als String                                           #
##################################################################################

function error_text($err_num) {
   global $global_errors;
   switch ($global_errors{$err_num}) {
      case ('NOERROR'):
         return 'Kein Fehler.';
      case ('DIVBYZERO'):
         return 'Division durch Null.';
      case ('OUTOFRANGE'):
         return 'Wert nicht in der Definitionsmenge.';
      case ('NOSUCHFUNC'):
         return 'Funktion existiert nicht.';
      case ('PARSEERR'):
         return 'Fehler beim Parsen.';
      case ('NOSUCHTYPE'):
         return 'Interner Fehler: Unbekannter Typ.';
      case ('NOTANUMBER'):
         return 'Zahl erwartet, aber nicht vorgefunden.';
      case ('ANYERROR'):
      default:
         return 'Unbekannter Fehler.';
   }
}

##################################################################################
# Hinweis: Die Ausgabe erfolgt , soweit nicht anders angegeben, immer in der Form  #
#          Array: 0 => Eigentliche Ausgabe, 1 => Fehlernummer                    #
##################################################################################

##################################################################################
# Funktion  : check_parens                                                       #
# Zweck     : "Uberpr"uft, ob Klammern in einem Ausdruck "passen"                #
```

```
# Eingabe   : Term als String      - $term                                          #
#             Array mit Klammern   - $parens                                        #
# Ausgabe   : Stringindex, an dem der Fehler ist, ansonsten -1                      #
# Sonstiges : - Array mit Klammern in der Form ((\"offnend, schlie\"send), ...)     #
#             - KEINE ARRAYAUSGABE                                                  #
#########################################################################################

function check_parens($term, $parens) {
   $temp_stack = array();
   for ($i = 0; $i < strlen($term); ++$i) {
      $char = $term{$i}; # hole Zeichen $i
      for ($j = 0; $j < count($parens); ++$j) {
         if ($char == $parens[$j][0]) {  # falls $char eine "offnende
                                         # Klammer enth"alt
            array_push($temp_stack, $j); # pushe Wert auf Stack ...
            break;                       # ... und fahre fort
         } elseif ($char == $parens[$j][1]) { # falls $char eine schlie"sende
                                              # Klammer enth"alt
            if (count($temp_stack) < 1) {
               return $i;
            }
            $other = array_pop($temp_stack);  # hole letzten Klammerwert vom Stack
            if ($other == $j) { # falls "offenende und schlie"sender Klammer passen
               break;           # n"achstes Zeichen
            } else {            # sonst
               return $i;       # gib Fehlerstelle zur"uck
            }
         }
      }
   }
   if (count($temp_stack) > 0) { # wenn der Stack nicht leer ist ...
      return strlen($term);      # liegt der Fehler am Ende des Strings
   }
   return -1; # kein Fehler
}

#########################################################################################
# Funktion  : check_minus                                                               #
# Zweck     : Wandelt '-' als Vorzeichen in '0 - ' um                                   #
# Eingabe   : Term als String                - $term                                    #
#             Array mit Klammern             - $parens                                  #
# Ausgabe   : Bearbeiteter Term als String                                              #
# Sonstiges : Array mit Klammern in der Form ((\"offnend, schlie\"send), ...)           #
#########################################################################################

function check_minus($term, $parens) {
   global $global_error_names;
   # - wird nur am Stringanfang und vor "offnender Klammer umgewandelt
   $new_term = '';
   $opening = array();  # Array mit "offnenden Klammern als Indices
   foreach ($parens as $pair) {
      $opening[$pair[0]] = $pair[1];
   }
   preg_replace('/\s+/', ' ', $term); # ersetze whitespace durch Leerzeichen
   for ($i = strlen($term) - 1; $i >= 0; --$i) { # parse String von hinten nach vorne
      $char = $term{$i}; # hole dir aktuelles Zeichen
      if ($char != '-') {  # Wenn das Zeichen kein Minuszeichen ist ...
         $new_term = $char . $new_term; # ... fahre fort
      } else {            # sonst:
         if ($i == 0) {    # am Stringanfang:
            $new_term = '0 - ' . $new_term; # Null hinzuf"ugen ...
            break;   # ... und tsch"uss
         }
         --$i; # sonst: betrachte Zeichen davor
         if ($term{$i} == ' ') { # falls es ein Leerzeichen ist ...
            if ($i == 0) { # falls es der Stringanfang war ...
               $new_term = '0 - ' . $new_term; # verfahre wie oben
               break;
            } else { # sonst
               --$i; # betrachte das n"achste nicht-Leerzeichen
            }
         }
         if (array_key_exists($term{$i}, $opening)) {
            # falls es eine "offnende Klammer ist ...
            $new_term = $term{$i} . ' 0 - ' . $new_term; # stelle eine '0' vor Minus
         } else { # sonst
            $new_term = $term{$i} . ' - ' . $new_term; # fahre fort
         }
      }
   }
```

```
   return array($new_term, $global_error_names{'NOERROR'});
}

##############################################################################
# Funktion  : add_times                                                      #
# Zweck     : F"ugt Malzeichen zwischen Zahlen und anderen Objekten          #
#             hinzu (implizite Multiplikation)                               #
# Eingabe   : Term als String                 - $term                        #
#             M"ogliche n"achste Zeichen      - $poss                        #
#             Verwandtes Malzeichen           - $times                       #
# Ausgabe   : Bearbeiteter Term als String                                   #
# Sonstiges : - $times sollte normalerweise '*' sein ...                     #
#             - $times darf nur aus einem Zeichen bestehen                   #
#             - Es wir nur ein Malzeichen gesetzt, wenn das n"achste Zeichen #
#               aus $poss ist (normalerweise "offnende Klammern und a-z, A-Z)#
##############################################################################

function add_times($term, $poss, $times) {
   global $global_error_names;
   $new_term = '';
   if (strlen($term) > 0) {
      $char2 = $term{0};
   } else {
      $char2 = '0';
   }
   for ($i = 0; $i < strlen($term) - 1; ++$i) {
      $char1 = $term{$i};     # hole Zeichen $i
      $char2 = $term{$i + 1}; # hole Zeichen ($i + 1)
      $new_term .= $char1;
      if (ctype_digit($char1) or $char1 == '.') { # m"ogliches Zahlenende
         if (in_array($char2, $poss)) { # und Substitution erw"unscht
            $new_term .= $times;
         }
      }
   }
   $new_term .= $char2;
   return array($new_term, $global_error_names{'NOERROR'});
}

##############################################################################
# Funktion  : tokenize                                                       #
# Zweck     : Zerlegt Term als String in tokens                              #
# Eingabe   : Term als String                 - $string                      #
#             Array mit Klammern              - $parens                      #
#             Un"are Operatoren (Funktionen)  - $unary                       #
#             Bin"are Operatoren (Operatoren) - $binary                      #
# Ausgabe   : Tokenliste als Array                                           #
# Sonstiges : - Array mit Klammern in der Form (("offnend, schlie"send), ...)#
#             - F"ur Operatorsyntax siehe mathdef.php                        #
#             - $constants sollte auch Variablen enthalten                   #
##############################################################################

function tokenize($string, $parens, $unary, $binary) {
   global $global_error_names;
   # tokens auf whitespace trennen
   $tokens = preg_split('/\s+/', $string, -1, PREG_SPLIT_NO_EMPTY);

   # Nur Ein-Zeichen-Operatoren k"onnen aus einem String ohne whitespace erkannt
   # werden, alle anderen Operatoren sollten durch Leerzeichen oder Klammern
   # getrennt sein. Wir d"urfen wegen Funktionen wie tan2 nicht auf Zahlen trennen.
   # Deshalb suchen wir jetzt Ein-Zeichen-Operatoren
   $one_char_ops = array();
   foreach ($unary as $name => $others) {
      if (strlen($name) == 1) {
         array_push($one_char_ops, $name);
      }
      foreach ($others['others'] as $alternative) {
         if (strlen($alternative) == 1) {
            array_push($one_char_ops, $alternative);
         }
      }
   }
   foreach ($binary as $level) {
      foreach ($level as $name => $rest) {
         if (strlen($name) == 1) {
            array_push($one_char_ops, $name);
         }
      }
   }
```

```
   # Splitten um Ein-Zeichen-Ops
   for ($i = count($tokens) - 1; $i >= 0; --$i) {
      # tokens r"uckw"arts durchgehen, da wir einzelne tokens durch mehrere tokens
      # ersetzen; sonst Probleme mit dem Offset
      $chunk = array($tokens[$i]);
      foreach ($one_char_ops as $operator) { # alle Ops durchgehen
         $regex = '/(' . preg_quote($operator, '/') . ')/';
         for ($j = count($chunk) - 1; $j >= 0; --$j) {
            $temp = preg_split($regex, $chunk[$j], -1,
                               PREG_SPLIT_NO_EMPTY | PREG_SPLIT_DELIM_CAPTURE);
            array_splice($chunk, $j, 1, $temp);
         }
      }
      array_splice($tokens, $i, 1, $chunk);
   }

   # Jetzt kann noch um schlie"sende Klammern und nach "offnenden Klammern ge-
   # splittet werden, nicht jedoch vor "offnenden Klammern, da sie zu Funktionen
   # geh"oren k"onnten.

   # um schlie"sende Klammern splitten
   foreach ($parens as $open_close) {
      $regex = '/(' . preg_quote($open_close[1]) . ')/';
      for ($i = count($tokens) - 1; $i >= 0; --$i) { # alle tokens durchgehen
         $temp = preg_split($regex, $tokens[$i], -1,
                            PREG_SPLIT_NO_EMPTY | PREG_SPLIT_DELIM_CAPTURE);
         array_splice($tokens, $i, 1, $temp);
      }
   }

   # vor "offnende Klammern splitten
   $opening = array(); # Array mit "offnenden Klammern als Indices
   foreach ($parens as $open_close) {
      $opening[$open_close[0]] = 1;
   }
   for ($i = count($tokens) - 1; $i >= 0; --$i) { # alle tokens durchgehen
      $sub_tokens = array($tokens[$i]);
      $j = strlen($tokens[$i]) - 2;
         # "offnende Klammer am Ende des Strings ist nicht interessant
      while ($j > -1) { # bis zum Anfang des Strings
         $temp = $sub_tokens[0]; # token extrahieren
         if (array_key_exists($temp{$j}, $opening)) {
            # wenn Zeichen $j "offnende Klammer ist
            array_splice($sub_tokens, 0, 1,
                         array(substr($temp, 0, $j + 1), substr($temp, $j + 1)));
               # nach der Klammer splitten
         }
         --$j;
      }
      array_splice($tokens, $i, 1, $sub_tokens);
   }

   return array($tokens, $global_error_names{'NOERROR'});
}

##############################################################################
# Funktion  : substitute                                                     #
# Zweck     : Ersetzt Tokens durch ihre Werte                                #
# Eingabe   : Tokenliste      - $tokens                                      #
#             Werteliste      - $values                                      #
# Ausgabe   : Bearbeitete Tokenliste                                         #
# Sonstiges : - Tokenliste kann z.B. auch Stack sein                         #
#             - Werteliste als Hash in der Form ('pi' => 3.14, 'e' => '2.72', ...) #
##############################################################################

function substitute($tokens, $values) {
   global $global_error_names;
   for ($i = 0; $i < count($tokens); ++$i) { # gehe alle tokens der Reihe nach durch
      if (array_key_exists($tokens[$i], $values)) { # wenn token eine Konstante ist
         $tokens[$i] = $values[$tokens[$i]];        # substitiere
         if (!$tokens[$i]) {
            $tokens[$i] = '0';
         } else {
            settype($tokens[$i], "string");
         }
      }
   }
   return array($tokens, $global_error_names{'NOERROR'});
}
```

```
##############################################################################
# Funktion  : build_tree                                                     #
# Zweck     : Bldet aus einer Tokenliste einen parse tree                    #
# Eingabe   : Tokenliste                    - $tokens                        #
#             Array mit Klammern            - $parens                        #
#             Un"are Operatoren (Funktionen)  - $unary                       #
#             Bin"are Operatoren (Operatoren) - $binary                      #
# Ausgabe   : Array: einziges Element: parse tree als vernestetes Array      #
# Sonstiges : - Array mit Klammern in der Form (("offnend, schlie"send), ...)  #
#             - F"ur Operatorsyntax siehe mathdef.php                        #
#             - parse tree Knoten: Array: 1 => Typ, 2 => Wert, 3, 4 => Kinder  #
#                - Typ: 'u' - un"ar, 'b' - bin"ar                            #
#                - Bei Wert kein Array, sondern einfach Wert                 #
#                - Bei un"arem Operator ist 4 don't care                     #
##############################################################################

function build_tree($tokens, $parens, $unary, $binary) {
   global $global_error_names;

   # Pr"azedenz: Klammern, un"are Operatoren, bin"are Operatoren, links-rechts
   # niedrigste Pr"azedenz ist im parse tree am weitesten oben

   # Zuerst nach Klammern (incl. un"are Operatoren) parsen
   $alt_parens = array();
   $opening    = array();
   foreach ($parens as $open_close) {
      $opening[$open_close[0]] = $open_close[1];
      $alt_parens[$open_close[0]] = $open_close[1];
      $alt_parens[$open_close[1]] = $open_close[0];
   }
   $paren_stack = array();
   for ($i = 0; $i < count($tokens); ++$i) {
      if (array_key_exists($tokens[$i]{strlen($tokens[$i]) - 1}, $alt_parens)) {
         array_push($paren_stack, array($tokens[$i]{strlen($tokens[$i]) - 1}, $i));
      }
   }
   $temp_paren_stack = array();
   while (count($paren_stack) > 0) { # Klammern unter den tokens
      if (array_key_exists($paren_stack[count($paren_stack) - 1][0], $opening)) {
         if (count($temp_paren_stack) == 0) {
            return array('', $global_error_names{'PARSEERR'});
         }
         if ($opening[$paren_stack[count($paren_stack) - 1][0]]
             == $temp_paren_stack[count($temp_paren_stack) - 1][0]) {
            if (count($temp_paren_stack) > 1) {
               array_pop($paren_stack);
               array_pop($temp_paren_stack);
            } elseif (count($temp_paren_stack) != 1) {
                return array('', $global_error_names{'PARSEERR'});
            } else {
               array_push($temp_paren_stack, array_pop($paren_stack));
               $temp_tokens = array_slice(
                  $tokens,
                  $temp_paren_stack[1][1],
                  $temp_paren_stack[0][1] - $temp_paren_stack[1][1] + 1
               );
               $temp_tree = build_tree(
                             array_slice($temp_tokens, 1, count($temp_tokens) - 2),
                             $parens, $unary, $binary
                           );
               if ($temp_tree[1] != $global_error_names{'NOERROR'}) { # Im Fall eines Fehlers
                  return array('', $temp_tree[1]);
               }
               if (strlen($temp_tokens[0]) == 1) { # Nur Klammern
                  array_splice(
                     $tokens,
                     $temp_paren_stack[1][1],
                     $temp_paren_stack[0][1] - $temp_paren_stack[1][1] + 1,
                     $temp_tree[0]
                  );
               } else { # un"arer Operator
                  array_splice(
                     $tokens,
                     $temp_paren_stack[1][1],
                     $temp_paren_stack[0][1] - $temp_paren_stack[1][1] + 1,
                     'dummy'
                  );
                  $tokens[$temp_paren_stack[1][1]] = array(
                     'u',
                     substr($temp_tokens[0], 0, strlen($temp_tokens[0]) - 1),
```

```
                    $temp_tree[0][0]
                 );
              }
              array_pop($temp_paren_stack);
              array_pop($temp_paren_stack);
           }
        } else {
           return array('', $global_error_names{'PARSEERR'});
        }
     } else {
        array_push($temp_paren_stack, array_pop($paren_stack));
     }
  }
  if (count($temp_paren_stack) > 0) {
     return array('', $global_error_names{'PARSEERR'});
  }
  # jetzt die bin"aren Operatoren
  foreach ($binary as $prec_level) {
     # und die tokens durchgehen
     for ($i = 1; $i < count($tokens) - 1; ++$i) {
        if (!is_array($tokens[$i]) and array_key_exists($tokens[$i], $prec_level)) {
           array_splice(
              $tokens,
              $i - 1,
              3,
              array(array('b', $tokens[$i], $tokens[$i - 1], $tokens[$i + 1]))
           );
           --$i;
        }
     }
  }
  if (!is_array($tokens)) {
     return array('', $global_error_names{'PARSEERR'});
  }
  if (count($tokens) != 1) {
     return array('', $global_error_names{'PARSEERR'});
  }
  if (!is_array($tokens[0])) {
     $tokens[0] = array($tokens[0]);
  }

  return array($tokens, $global_error_names{'NOERROR'});
}

################################################################################
# Funktion  : build_stack                                                      #
# Zweck     : Bildet Stack aus parse tree                                      #
# Eingabe   : Tree            - $tree                                          #
# Ausgabe   : Stack als Array                                                  #
################################################################################

function build_stack($tree) {
  global $global_error_names;
  $stack = array(); # Ergebnisstack
  if (!is_array($tree)) { # Konstante / Variable
     return array(array($tree), $global_error_names{'NOERROR'});
     break;
  }
  switch ($tree[0]) {
     case 'u': # un"arer Operator
        $first = build_stack($tree[2]); # hole dir Stack des Arguments
        if ($first[1] != $global_error_names{'NOERROR'}) { # Im Fall eines Fehlers
           return array('', $first[1]); # schlage Alarm
        }
        # bilde Stack
        $stack = array_merge($stack, $tree[1], $first[0]);
        return array($stack, $global_error_names{'NOERROR'});
        break;
     case 'b': # bin"arer Operator
        $first = build_stack($tree[2]); # hole dir Stack des ersten Arguments
        if ($first[1] != $global_error_names{'NOERROR'}) { # Im Fall eines Fehlers
           return array('', $first[1]); # schlage Alarm
        }
        $second = build_stack($tree[3]); # hole dir Stack des zweiten Arguments
        if ($second[1] != $global_error_names{'NOERROR'}) { # Im Fall eines Fehlers
           return array('', $second[1]); # schlage Alarm
        }
        # bilde Stack
        $stack = array_merge($stack, $tree[1], $first[0], $second[0]);
        return array($stack, $global_error_names{'NOERROR'});
```

```
            break;
         default : # Konstante / Variable
            return array(array($tree[0]), $global_error_names{'NOERROR'});
            break;
      }
   }

   ##############################################################################
   # Funktion  : process_stack                                                  #
   # Zweck     : F"uhrt die Berechnung an einem Stack durch                     #
   # Eingabe   : Stack                          - $stack                        #
   #             Un"are Operatoren (Funktionen) - $unary                        #
   #             Bin"are Operatoren (Operatoren) - $binary                      #
   # Ausgabe   : Wert                                                           #
   ##############################################################################

   function process_stack($stack, $unary, $binary) {
      global $global_error_names;
      $result_stack = array(); # tempor"arer Stack f"ur Ergebnisse

      $unary_lookup = array(); # um alternative Schreibweisen zu konvertieren
      foreach ($unary as $key => $definition) {
         if (!is_array($definition)) { continue; } # ACHTUNG: Das w"are ein Fehler in der
                                                   # Implementierung
         $unary_lookup[$key] = $key; # der Operator entspricht sich selbst
         foreach($definition['others'] as $alternative) {
            $unary_lookup[$alternative] = $key;
         }
      }

      $binary_lookup = array(); # um Pr"azedenz festzustellen
      $precedence = 0;
      foreach ($binary as $level) {
         if (!is_array($level)) { # ACHTUNG: Das w"are ein Fehler in der
            ++$precedence;        # Implementierung
            continue;
         }
         foreach ($level as $operator => $dummy) {
            if (!is_array($dummy)) { # ACHTUNG: Das w"are ein Fehler in der
               continue;             # Implementierung
            }
            $binary_lookup[$operator] = $precedence;
         }
         ++$precedence;
      }

      while (count($stack) > 0) {
         $current = array_pop($stack);
         if (array_key_exists($current, $unary_lookup))  { # un"arer Operator
            $argument1 = array_pop($result_stack);
            $result = $unary{$unary_lookup{$current}}{'eval'}($argument1, $error);
            if ($error != $global_error_names{'NOERROR'}) {
               return array('', $error);
            }
            array_push($result_stack, $result);
         } elseif (array_key_exists($current, $binary_lookup)) { # bin"arer Operator
            $argument1 = array_pop($result_stack);
            $argument2 = array_pop($result_stack);
            $result = $binary{$binary_lookup{$current}}{$current}{'eval'}($argument1, $argument2, $error);
            if ($error != $global_error_names{'NOERROR'}) {
               return array('', $error);
            }
            array_push($result_stack, $result);
         } else { # Zahl (hoffentlich!)
            if (!is_numeric($current)) {
               return array('', $global_error_names{'NOTANUMBER'});
            }
            array_push($result_stack, $current);
         }
      }

      if (count($result_stack) != 1) {
         return array('', $global_error_names{'ANYERROR'});
      }
      return array($result_stack[0], $global_error_names{'NOERROR'});
   }


?>
```

4.13 Datei „units/graphics.php“

```
<?php

   ##############################################################################
   # Dies ist die Datei 'graphics.php'                                          #
   #                                                                            #
   # Sie enth"alt Methoden f"ur die Grafik des Plotters (Achsen, Plotten, etc.) des  #
   # Parser/Plotter-Projekts                                                    #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen #
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den #
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund   #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich  #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder #
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies            #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-  #
   # Adresse oder "uber den Sysadmin.                                           #
   #                                                                            #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses #
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
   # Nutzung auf eigene Gefahr.                                                 #
   #                                                                            #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der       #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht, #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug   #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                  #
   #                                                                            #
   # (c) Jan Olligs, 2004                                                       #
   # Alle Rechte vorbehalten                                                    #
   #                                                                            #
   # Jan.Olligs@gmx.net                                                         #
   ##############################################################################

   ##############################################################################
   # Funktion  : init_plot                                                      #
   # Zweck     : Zeichnet in eine GD-Bild Hintergrund, Achsen und Gitterlinien  #
   # Eingabe   : Bild als GD-Grafik         - $image,                           #
   #             Breite des Bildes          - $width,                           #
   #             H"ohe des Bildes           - $height,                          #
   #             kleinster x-Wert           - $xmin,                            #
   #             gr"o"ster x-Wert           - $xmax,                            #
   #             x-Gitterweite              - $xgrid,                           #
   #             kleinster y-Wert           - $ymin,                            #
   #             gr"o"ster y-Wert           - $ymax,                            #
   #             y-Gitterweite              - $ygrid,                           #
   #             Hintergrundfarbe           - $back,                            #
   #             Achsenfarbe                - $axes,                            #
   #             Farbe der Gitterlinien     - $grid                             #
   # Ausgabe   : Bearbeitetes Bild                                              #
   # Sonstiges : Farben sind dem Bild zugeordnete GD-Farben                     #
   ##############################################################################

   function init_plot($image, $width, $height, $xmin, $xmax, $xgrid,
      $ymin, $ymax, $ygrid, $back, $axes, $grid) {
      # Hintergrund bemalen
      ImageFilledRectangle(
         $image,
         0,
         0,
         $width - 1,
         $height - 1,
         $back
      );
```

```
    # Gitterlinien zeichnen
    # parallel zur x-Achse
    if ($ygrid > 0) {
       for ($y = $ygrid; $y <= $ymax; $y += $ygrid) {
          ImageLine(
             $image,
             0,
             ($ymax - $y) * ($height - 1) / ($ymax - $ymin),
             $width - 1,
             ($ymax - $y) * ($height - 1) / ($ymax - $ymin),
             $grid
          );
       }
       for ($y = -$ygrid; $y >= $ymin; $y -= $ygrid) {
          ImageLine(
             $image,
             0,
             ($ymax - $y) * ($height - 1) / ($ymax - $ymin),
             $width - 1,
             ($ymax - $y) * ($height - 1) / ($ymax - $ymin),
             $grid
          );
       }
    }

    # parallel zur y-Achse
    if ($xgrid > 0) {
       for ($x = $xgrid; $x <= $xmax; $x += $xgrid) {
          ImageLine(
             $image,
             ($x - $xmin) * ($width - 1) / ($xmax - $xmin),
             0,
             ($x - $xmin) * ($width - 1) / ($xmax - $xmin),
             $height - 1,
             $grid
          );
       }
       for ($x = -$xgrid; $x >= $xmin; $x -= $xgrid) {
          ImageLine(
             $image,
             ($x - $xmin) * ($width - 1) / ($xmax - $xmin),
             0,
             ($x - $xmin) * ($width - 1) / ($xmax - $xmin),
             $height - 1,
             $grid
          );
       }
    }

    # Achsen zeichnen
    # x-Achse
    ImageLine(
       $image,
       0,
       $ymax * ($height - 1) / ($ymax - $ymin),
       $width - 1,
       $ymax * ($height - 1) / ($ymax - $ymin),
       $axes
    );
    # y-Achse
    ImageLine(
       $image,
       (-$xmin) * ($width - 1) / ($xmax - $xmin),
       0,
       (-$xmin) * ($width - 1) / ($xmax - $xmin),
       $height - 1,
       $axes
    );
    return $image;
}

################################################################################
# Funktion  : plot_graph                                                       #
# Zweck     : Zeichnet Graphen aus Array von Punkten in ein GD-Bild            #
# Eingabe   : Bild als GD-Grafik         - $image,                             #
#             Breite des Bildes          - $width,                             #
#             H"ohe des Bildes           - $height,                            #
#             kleinster x-Wert           - $xmin,                              #
#             gr"o"ster x-Wert           - $xmax,                              #
```

```
#             kleinster y-Wert           - $ymin,                                  #
#             gr"o"ster y-Wert           - $ymax,                                  #
#             Array mit Punkten          - $points                                 #
#             Graphenfarbe               - $color,                                 #
# Ausgabe   : Bearbeitetes Bild                                                    #
# Sonstiges : - Farben sind dem Bild zugeordnete GD-Farben                         #
#             - Punkte in der Form (x => (y, Fehlernummer), ...)                   #
####################################################################################

function plot_graph($image, $width, $height, $xmin, $xmax,
   $ymin, $ymax, $points, $color) {
   global $global_error_names;
   $lastx = 0;      # zuletzt bearbeiteter x-Wert
   $lasty = 0;      # zuletzt bearbeiteter y-Wert
   $undef = true;   # letzter Wert war nicht definiert / Fehler
   $range = false;  # letzter Wert war au"serhalb des Bereichs
   ksort($points, SORT_NUMERIC); # Punkte nach x-Werten sortieren ...
   reset($points);               # ... und Arrayz"ahler zur"ucksetzen
   foreach ($points as $x => $content) {
      # y-Wert und Fehlertyp extrahieren
      $y     = $content[0];
      $error = $content[1];
      if ($error != $global_error_names{'NOERROR'}) { # Wenn ein Fehler vorliegt
         $undef = true;
         $range = false;
      } else { # Kein Fehler
         if (!$undef) { # Letzter Wert war definiert
            if ($xmin <= $x and $x <= $xmax and $ymin <= $y and $y <= $ymax) {
               # Punkt im Bereich, alles in Ordnung
               ImageLine(
                  $image,
                  ($lastx - $xmin) * ($width - 1) / ($xmax - $xmin),
                  ($ymax - $lasty) * ($height - 1) / ($ymax - $ymin),
                  ($x - $xmin) * ($width - 1) / ($xmax - $xmin),
                  ($ymax - $y) * ($height - 1) / ($ymax - $ymin),
                  $color
               );
               $range = false;
            } else { # Punkt au"serhalb des Bereiches
               if (!$range) { # der letzte aber nicht, also zeichnen
                  ImageLine(
                     $image,
                     ($lastx - $xmin) * ($width - 1) / ($xmax - $xmin),
                     ($ymax - $lasty) * ($height - 1) / ($ymax - $ymin),
                     ($x - $xmin) * ($width - 1) / ($xmax - $xmin),
                     ($ymax - $y) * ($height - 1) / ($ymax - $ymin),
                     $color
                  );
               }
               $range = true;
            }
         } else { # Letzter Wert war nicht definiert
            $undef = false;
            if ($xmin <= $x and $x <= $xmax and $ymin <= $y and $y <= $ymax) {
               $range = false;
            } else {
               $range = true;
            }
         }
      }
      $lastx = $x;
      $lasty = $y;
   }
   return $image;
}

####################################################################################
# Funktion  : generate_points                                                      #
# Zweck     : Berechnet Punkte aus den x-Werten.                                   #
# Eingabe   : Funktion als Stack       - $function,                                #
#             x-Werte als Array        - $x_values,                                #
#             Konstanten               - $constants                                #
# Ausgabe   : Punkte                                                               #
# Sonstiges : - Punkte in der Form (x => (y, Fehlernummer), ...)                   #
####################################################################################

function generate_points($function, $x_values, $constants) {
   global $global_error_names;
   global $global_errors;
   global $global_unary_operators;
```

```
   global $global_binary_operators;

   $results = array();

   # Konstantensubstitution
   $temp = substitute($function, $constants);
   if ($temp[1] != $global_error_names['NOERROR']) {
      return array(array(0, $temp[1]));
   }
   $function = $temp[0];

   foreach ($x_values as $x) {
      $temp = substitute($function, array('x' => $x));
      if ($temp[1] != $global_error_names['NOERROR']) {
         return array(array(0, $temp[1]));
      }
      $temp = $temp[0];
      $temp = process_stack($temp, $global_unary_operators, $global_binary_operators);
      switch ($global_errors{$temp[1]}) {
         case ('NOSUCHFUNC'): # diese Fehler k"onnen nicht
         case ('PARSEERR'):   # aufgrund eines fehlerhaften
         case ('NOSUCHTYPE'): # Wertes auftreten
         case ('NOTANUMBER'):
            return array(array(0, $temp[1]));
      }
      $results["$x"] = $temp;
   }
   return $results;
}

##############################################################################
# Funktion  : fixed_step                                                     #
# Zweck     : Berechnet Punkte mit festem delta x.                           #
# Eingabe   : Funktion als Stack         - $function,                        #
#             kleinster x-Wert           - $xmin,                            #
#             gr"o"ster x-Wert           - $xmax,                            #
#             Schrittweite               - $step,                            #
#             Konstanten                 - $constants                        #
# Ausgabe   : Punkte                                                         #
# Sonstiges : - Punkte in der Form (x => (y, Fehlernummer), ...)             #
#             - Anfangs- und Endwert sind immer in der Ausgabe enthalten     #
##############################################################################

function fixed_step($function, $xmin, $xmax, $step, $constants) {
   global $global_error_names;

   # Fehler in den Parametern -- keine Chance au"Ser f"ur Hellseher
   if ($step == 0 or $xmin == $xmax ) {
      return array(array(0, $global_error_names{'OUTOFRANGE'}));
   }

   # Minimum und Maximum vertauscht; wir machen es r"uckg"angig
   if ($xmin > $xmax) {
      $temp = $xmin;
      $xmin = $xmax;
      $xmax = $temp;
   }

   if ($step < 0) {
      $step = -$step;
   }

   $x_values = array();
   for ($x = $xmin; $x <= $xmax; $x += $step) {
      array_push($x_values, $x);
   }
   if ($x < $x_max + $step) { # der rechte Punkt muss rein
      array_push($x_values, $x_max);
   }
   return generate_points($function, $x_values, $constants);
}

##############################################################################
# Funktion  : get_height                                                     #
# Zweck     : Gibt die H"ohe am zweiten Punkt eines durch drei Punkte bestimmten  #
#             Dreiecks zur"uck. (in Pixel)                                   #
# Eingabe   : x-Koordinate Punkt 1       - $x1,                              #
#             y-Koordinate Punkt 1       - $y1,                              #
#             x-Koordinate Punkt 2       - $x2,                              #
#             y-Koordinate Punkt 2       - $y2,                              #
```

```
#             x-Koordinate Punkt 3       - $x3,                                  #
#             y-Koordinate Punkt 3       - $y3,                                  #
#             Aufl"osung x-Richtung      - $resx,                                #
#             Aufl"osung y-Richtung      - $resy                                 #
# Ausgabe   : H"ohe des Dreiecks in Pixel                                        #
# Sonstiges : - Punkte in normalen Koordinaten, Ausgabe in Pixel                 #
#             - Aufl"osung in Pixel pro Einheit                                  #
#             - Nach Heron gilt sqrt(s(s-a)(s-b)(s-c)) = A = (a * h_a) / 2,      #
#               hieraus ergibt sich die angegebene Formel.                       #
##################################################################################

function get_height($x1, $y1, $x2, $y2, $x3, $y3, $resx, $resy) {
   $a = sqrt($resx * $resx * ($x1 - $x3) * ($x1 - $x3)
           + $resy * $resy * ($y1 - $y3) * ($y1 - $y3));
   $b = sqrt($resx * $resx * ($x1 - $x2) * ($x1 - $x2)
           + $resy * $resy * ($y1 - $y2) * ($y1 - $y2));
   $c = sqrt($resx * $resx * ($x2 - $x3) * ($x2 - $x3)
           + $resy * $resy * ($y2 - $y3) * ($y2 - $y3));
   return sqrt(($a + $b + $c) * (-$a + $b + $c) * ($a - $b + $c) * ($a + $b - $c)) / (2 * $a);
}

##################################################################################
# Funktion  : auto_step                                                          #
# Zweck     : Versucht, m"oglichst wenige Punkte zu berechnen, die den Graphen   #
#             der Funktion m"oglichst "glatt" approximieren                      #
# Eingabe   : Funktion als Stack         - $function,                            #
#             kleinster x-Wert           - $xmin,                                #
#             gr"o"ster x-Wert           - $xmax,                                #
#             kleinster y-Wert           - $ymin,                                #
#             gr"o"ster y-Wert           - $ymax,                                #
#             Aufl"osung x-Richtung      - $resx,                                #
#             Aufl"osung y-Richtung      - $resy,                                #
#             Konstanten                 - $constants                           #
# Ausgabe   : Punkte                                                             #
# Sonstiges : - Punkte in der Form (x => (y, Fehlernummer), ...)                 #
#             - Anfangs- und Endwert sind immer in der Ausgabe enthalten         #
#             - Aufl"osung in Pixel pro Einheit                                  #
##################################################################################

function auto_step($function, $xmin, $xmax, $ymin, $ymax, $resx, $resy, $constants) {
   global $global_error_names;
   global $global_errors;
   global $global_unary_operators;
   global $global_binary_operators;

   $noprocess  = false;
   $outofrange = false;

   # Fehler in den Parametern -- keine Chance au"ser f"ur Hellseher
   if ($resx <= 0 or $resy <= 0 or $xmin == $xmax or $ymin == $ymax) {
      return array(array(0, $global_error_names{'OUTOFRANGE'}));
   }

   # Minimum und Maximum vertauscht; wir machen es r"uckg"angig
   if ($xmin > $xmax) {
      $temp = $xmin;
      $xmin = $xmax;
      $xmax = $temp;
   }

   # Minimum und Maximum vertauscht; wir machen es r"uckg"angig
   if ($ymin > $ymay) {
      $temp = $ymin;
      $ymin = $ymay;
      $ymay = $temp;
   }

   # Wir berechnen Schrittweite mit Minimum und Maximum und achten darauf, dass
   # keiner der Werte Null wird ("uberpr"uft am kleinsten Wert)
   $temp = 1;
   do {
      $stepmin = CONST_STEP_MIN_RATIO * $temp / $resx;
      $step    = CONST_STEP_RATIO     * $temp / $resx;
      $stepmax = CONST_STEP_MAX_RATIO * $temp / $resx;
      ++$temp;
   } while ($stepmin == 0);

   $results = array();

   # Konstantensubstitution
```

```
$temp = substitute($function, $constants);
if ($temp[1] != $global_error_names['NOERROR']) {
   return array(array(0, $temp[1]));
}
$function = $temp[0];

# Erster Punkt

$x = $xmin;
   $temp = substitute($function, array('x' => $x));
   if ($temp[1] != $global_error_names['NOERROR']) {
      return array(array(0, $temp[1]));
   }
   $temp = $temp[0];
   $temp = process_stack($temp, $global_unary_operators, $global_binary_operators);
   switch ($global_errors{$temp[1]}) {
      case ('NOSUCHFUNC'): # diese Fehler k"onnen nicht
      case ('PARSEERR'):   # aufgrund eines fehlerhaften
      case ('NOSUCHTYPE'): # Wertes auftreten
      case ('NOTANUMBER'):
         return array(array(0, $temp[1]));
   }
   $results["$x"] = $temp;
   $x2 = $x;
   if ($temp[1] != $global_error_names{'NOERROR'}) {
      $noprocess = true;
   }

# Zweiter Punkt

$x += $step;
   $temp = substitute($function, array('x' => $x));
   if ($temp[1] != $global_error_names['NOERROR']) {
      return array(array(0, $temp[1]));
   }
   $temp = $temp[0];
   $temp = process_stack($temp, $global_unary_operators, $global_binary_operators);
   $results["$x"] = $temp;
   $x1 = $x;
   if ($temp[1] != $global_error_names{'NOERROR'}) {
      $noprocess = true;
   }
   if ($y < $ymin or $y > $ymax) {
     $outofrange = true;
   }

# Der Rest der Punkte

$x += $step;
while ($x <= $xmax) {
   $temp = substitute($function, array('x' => $x));
   if ($temp[1] != $global_error_names['NOERROR']) {
      return array(array(0, $temp[1]));
   }
   $temp = $temp[0];
   $temp = process_stack($temp, $global_unary_operators, $global_binary_operators);
   $results["$x"] = $temp;
   $x1 = $x;
   if ($temp[1] == $global_error_names{'NOERROR'}) { # wenn kein Fehler auftritt
      if (!$noprocess) { # wenn wir alles bearbeiten sollen
         $height = get_height($x2, $results["$x2"][0],
                              $x1, $results["$x1"][0],
                              $x,  $y,
                              $resx, $resy);
         # die H"ohe des Dreiecks aus den letzten drei Punkten
         if ($height > CONST_MAX_HEIGHT and $step >= 2 * $stepmin) {
            # wenn die Abweichung zu gro"s ist und wir die Schrittweite
            # verkleinern k"onnen
            $x  = $x1; # heucheln wir vor, der letzte Punkt zu sein
            $x1 = $x2;
            $step /= 2; # und halbieren die Schrittweite
         } else if ($height < CONST_MIN_HEIGHT and 2 * $step <= $stepmax) {
            # wenn die Abweichung zu klein ist und wir die Schrittweite
            # vergr"o"sern k"onnen
            $step *= 2; # verdoppeln wir die Schrittweite
         }
      } else { # wenn wir nicht weitermachen sollten
         if ($results["$x1"][1] == $global_error_names{'NOERROR'}) {
            # falls nur der vorletzte Wert nicht definiert war
             $noprocess = false; # geht es jetzt weiter
```

```
                } else { # sonst
                   $noprocess = true; # brauchen wir noch eine Auszeit
                }
             }
             if ($y < $ymin or $y > $ymax) {
                $outofrange = true;
             } else {
                $outofrange = false;
             }
          } else { # wenn ein Fehler auftritt
             if (!$outofrange and $step >= 2 * $stepmin) {
                # wenn man es noch sehen kann, gehen wir m"oglichst nahe heran
                $x  = $x1;  # wir heucheln, der letzte Wert zu sein
                $x1 = $x2;
                $step /= 2; # und halbieren die Schrittweite
             } else {
                $noprocess = true;
             }
          }
          $x2 = $x1; # vorletzten und letzten Punkt bearbeiten
          $x1 = $x;
          $x += $step;
       }

       # Falls der rechte Rand nicht dabei ist nehmen eir ihn noch dazu

       if ($x < $xmax + $step) {
          $x = $xmax;
          while ($x <= $xmax) {
             $temp = substitute($function, array('x' => $x));
             if ($temp[1] != $global_error_names['NOERROR']) {
                return array(array(0, $temp[1]));
             }
             $temp = $temp[0];
             $temp = process_stack($temp, $global_unary_operators, $global_binary_operators);
             $results["$x"] = $temp;
             $x1 = $x;
             if ($temp[1] == $global_error_names{'NOERROR'}) {
                if (!$noprocess) {
                   $height = get_height($x2, $results["$x2"][0],
                                        $x1, $results["$x1"][0],
                                        $x,  $y,
                                        $resx, $resy);
                   if ($height > CONST_MAX_HEIGHT and $step >= 2 * $stepmin) {
                      $x  = $x1;
                      $x1 = $x2;
                      $step /= 2;
                   } else if ($height < CONST_MIN_HEIGHT and 2 * $step <= $stepmax) {
                      $step *= 2;
                   }
                } else {
                   if ($results["$x1"][1] == $global_error_names{'NOERROR'}) {
                       $noprocess = false;
                    } else {
                       $noprocess = true;
                    }
                }
                if ($y < $ymin or $y > $ymax) {
                   $outofrange = true;
                } else {
                   $outofrange = false;
                }
             } else {
                if (!$outofrange and $step >= 2 * $stepmin) {
                   $x  = $x1;
                   $x1 = $x2;
                   $step /= 2;
                } else {
                   $noprocess = true;
                }
             }
             $x2 = $x1;
             $x1 = $x;
             $x += $step;
          }
       }
       return $results;
    }

?>
```

4.14 Datei „units/plotter_constants.php“

```
<?php

   ##############################################################################
   # Dies ist die Datei 'plotter_constants.php'                                 #
   #                                                                            #
   # Sie enth"alt Konstanten f"ur die Grafik des Plotters (Farben, Standardfunktion,  #
   # etc.) des Parser/Plotter-Projekts                                          #
   #                                                                            #
   # Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-       #
   # kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt  #
   # werden:                                                                    #
   #    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert   #
   #      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf #
   #      verwandt werden                                                       #
   #    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn   #
   #      verwenden, einsehbar gemacht werden                                   #
   #                                                                            #
   # Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in       #
   # andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus   #
   # oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen #
   # wollen, so kontaktieren Sie mich bitte.                                    #
   #                                                                            #
   # Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich  #
   # dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den #
   # Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund   #
   # auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich  #
   # mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder #
   # den Urheber geistigen Eigentums vorenthalten, so geschieht dies            #
   # unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-  #
   # Adresse oder "uber den Sysadmin.                                           #
   #                                                                            #
   # Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses #
   # entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
   # Nutzung auf eigene Gefahr.                                                 #
   #                                                                            #
   # Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der       #
   # Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht, #
   # den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug   #
   # Zeit, um E-Mails "uber technische Details zu beantworten.                  #
   #                                                                            #
   # (c) Jan Olligs, 2004                                                       #
   # Alle Rechte vorbehalten                                                    #
   #                                                                            #
   # Jan.Olligs@gmx.net                                                         #
   ##############################################################################

   ##############################################################################
   #                          Farbdefinitionen als RGB                          #
   ##############################################################################

   $global_colors = array(
           #       Name             -R-  -G-  -B-
           array('Schwarz',          0,   0,   0),
           array('Wei',            255, 255, 255),
           array('Grau',           200, 200, 200),
           array('Blau',             0,   0, 160),
           array('Hellblau',         0,   0, 255),
           array('Bordeaux',       160,   0,   0),
           array('Rot',            255,   0,   0),
           array('Grn',              0, 160,   0),
           array('Hellgrn',          0, 255,   0),
           array('Gelb',           160, 160,   0),
           array('Hellgelb',       255, 255,   0),
           array('Cyan',             0, 160, 160),
           array('Helles Cyan',      0, 255, 255),
           array('Mangenta',       160,   0, 160),
           array('Helles Mangenta', 255,   0, 255)
   );

   # Standardwerte f"ur den Graph / das Formular
   # Erster Wert: 'i': Ignorieren, 'n': positive Zahl, 't': Term, 'c': Farbe,
   #              'p': Prozentzahl, 'b': Boolean, 'e': Berechnen
   $plotter_standards = array(
      'func'          => array('t', 'cos(x)*sin(x)'), # Funktionsterm

      'img_width'     => array('n', 640),             # Bildbreite
```

```
    'img_height'     => array('n', 480),             # Bildh"ohe
    'img_types'      => array('i', array(
                                  array('PNG',  IMG_PNG, "Content-type: image/png"),
                                  array('GIF',  IMG_GIF, "Content-type: image/gif"),
                                  array('JPEG', IMG_JPG, "Content-type: image/jpeg")
                                    )
                           ),
                                          # Bildbereich
    'x_min'          => array('e', '-2pi'),        # kleinstes x
    'x_max'          => array('e', '2pi'),         # gr"o"stes x
    'x_grid'         => array('e', 'pi/4'),        # x-Gitterweite

    'y_min'          => array('e', '-1.5'),        # kleinstes y
    'y_max'          => array('e', '1.5'),         # gr"o"stes y
    'y_grid'         => array('e', '0.5'),         # y-Gitterweite

    'step'           => array('e', '0.025'),       # Schrittweite
    'auto_step'      => array('b', 'true'),        # Schrittweitensteuerung an?
                                          # Farben
    'BG_color'       => array('c', 1),             # Hintergrundfarbe
    'PL_color'       => array('c', 5),             # Graphenfarbe
    'AX_color'       => array('c', 0),             # Achsfarbe
    'GR_color'       => array('c', 2),             # Gitterlinienfarbe
                                          # Schatten
    'shading_min'    => array('e', '0'),           # kleinster Schatten-x-Wert
    'shading_max'    => array('e', 'pi'),          # gr"o"ster Schatten-x-Wert
    'SH_color'       => array('c', 3),             # Schattenfarbe
    'shading_trans'  => array('p', 30),            # 30 % Transparenz
    'shading_f1'     => array('t', 'sin(x)'),      # 1. Grenzfunktion
    'shading_f2'     => array('t', 'cos(x)'),      # 2. Grenzfunktion
    'SF_color'       => array('c', 0),             # Farbe der Grenzfunktionen
    'do_shade'       => array('b', 'false'),       # Schatten an?
                                          # Asymptoten
    'asym_px1'       => array('e', '0'),           # Punkt 1 x
    'asym_py1'       => array('e', '0'),           #         y
    'asym_px2'       => array('e', '0'),           # Punkt 2 x
    'asym_py2'       => array('e', '0'),           #         y
    'AS_color'       => array('c', 2),             # Asymptotenfarbe
    'asym_dash'      => array('b', 'true')         # gestrichelt?
);

################################################################################
#                              Konstanten f"ur auto_step                       #
################################################################################

define('CONST_STEP_MIN_RATIO',  1 / 8); # minimale Schrittweite (relativ)
define('CONST_STEP_RATIO',      4);     # Anfangsschrittweite (relativ)
define('CONST_STEP_MAX_RATIO', 16);     # maximale Schrittweite (relativ)

define('CONST_MIN_HEIGHT', 0.6);        # minimale H"ohe (f"ur Algorithmus)
define('CONST_MAX_HEIGHT', 1.5);        # maximale H"ohe (f"ur Algorithmus)

?>
```

4.15 Datei „units/various.php“

```
<?php

################################################################################
# Dies ist die Datei 'various.php'                                             #
#                                                                              #
# Sie enth"alt verschiedene Methoden f"ur das Parser/Plotter-Projekt           #
#                                                                              #
# Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-         #
# kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt    #
# werden:                                                                      #
#    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert     #
#      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf   #
#      verwandt werden                                                         #
#    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn     #
#      verwenden, einsehbar gemacht werden                                     #
#                                                                              #
# Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in         #
# andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus     #
# oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen  #
# wollen, so kontaktieren Sie mich bitte.                                      #
```

```
#                                                                                #
# Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich      #
# dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den    #
# Tokenizer, der Parser und die Schrittweitensteuerung habe ich von Grund        #
# auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich      #
# mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder    #
# den Urheber geistigen Eigentums vorenthalten, so geschieht dies                #
# unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail-      #
# Adresse oder "uber den Sysadmin.                                               #
#                                                                                #
# Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses    #
# entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. #
# Nutzung auf eigene Gefahr.                                                     #
#                                                                                #
# Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der           #
# Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,     #
# den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug       #
# Zeit, um E-Mails "uber technische Details zu beantworten.                      #
#                                                                                #
# (c) Jan Olligs, 2004                                                           #
# Alle Rechte vorbehalten                                                        #
#                                                                                #
# Jan.Olligs@gmx.net                                                             #
##################################################################################

##################################################################################
# Funktion  : check_error                                                        #
# Zweck     : Untainting von Variablen oder Fehlerausgabe                        #
# Eingabe   : Zu "uberpr"ufender Wert    - $value,                               #
#             Art der Variable           - $kind,                                #
#             Fehlermeldung              - $errormsg,                            #
#             Standardwert               - $standard,                            #
#             Konstanten (f"ur 'e')      - $constants                            #
# Ausgabe   : "Uberpr"ufter Wert / Fehlermeldung                                 #
# Sonstiges : - Wenn $errormsg == true, dann wird eine Fehlermeldung zur"uckge-  #
#               geben, sonst der korrigierte Wert                                #
#             - korrigierter Wert = $standard bei Fehler, $value sonst           #
#             - M"oglichkeiten f"ur die Variable $kind                           #
#                  - 'n': number     - Nat"urliche Zahl                          #
#                  - 't': term       - Term                                      #
#                  - 'c': color      - Farbe                                     #
#                  - 'p': percentage - Prozentwert                               #
#                  - 'b': boolean    - Bool'scher Wert                           #
#                  - 'e': evaluate   - Ausdruck berechnen                        #
##################################################################################

function check_error($value, $kind, $errormsg = true, $standard = 0, $constants = array()) {

   global $global_parens;
   global $global_error_names;
   global $global_unary_operators;
   global $global_binary_operators;
   global $global_constants;

   switch ($kind) {
      case ('n'):
         if ($errormsg) {
            if (is_numeric($value) and is_int($value + 0)) {
               return '';
            } else {
               return "Keine ganze Zahl: '$value'.";
            }
         } else {
            if (is_numeric($value) and is_int($value + 0)) {
               return $value;
            } else {
               return $standard;
            }
         }
         break;
      case ('t'):
         if ($errormsg) {
            $temp = check_parens($value, $global_parens);
            if ($temp > -1) {
               $err = '';
               for ($i = 0; $i < strlen($value); ++$i) {
                  if ($i == $temp) {
                     $err .= "<font color=\"red\">" . htmlspecialchars($value{$i}) . "</font>";
                  } else {
                     $err .= htmlspecialchars($value{$i});
```

```
            }
         }
         return "Die rote Klammer im Term passt nicht: '$err'.";
      }
      $term = check_minus($value, $global_parens);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Einf&uuml;gen der Minuszeichen ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $poss = array_merge(range('a', 'z'), range('A', 'Z'));
      foreach ($global_parens as $pair) {
         array_push($poss, $pair[0]);
      }
      $term = add_times($term, $poss, '*');
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Einf&uuml;gen der Malzeichen ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = tokenize($term, $global_parens, $global_unary_operators,
                       $global_binary_operators);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Bei der Zerlegung in Tokens ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = substitute($term, $global_constants);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Bei der Substitution der Konstanten ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = build_tree($term, $global_parens, $global_unary_operators, $global_binary_operators);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Aufbau des <i>parse tree</i> ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0][0];
      $term = build_stack($term);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Aufbau des Stacks ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = substitute($term, array('x' => '1'));
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Bei der Substitution von <i>x</i> ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = process_stack($term, $global_unary_operators, $global_binary_operators);
      if (
            $term[1] != $global_error_names['NOERROR'] and
            $term[1] != $global_error_names['DIVBYZERO'] and
            $term[1] != $global_error_names['OUTOFRANGE']
         ) {
         return "Bei der Berechnung des Funktionswertes ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      return '';
   } else {
      $temp = check_parens($value, $global_parens);
      if ($temp > -1) {
         return $standard;
      }
      $term = check_minus($value, $global_parens);
      if ($term[1] != $global_error_names['NOERROR']) {
         return $standard;
      }
      $term = $term[0];
      $poss = array_merge(range('a', 'z'), range('A', 'Z'));
      foreach ($global_parens as $pair) {
         array_push($poss, $pair[0]);
      }
      $term = add_times($term, $poss, '*');
      if ($term[1] != $global_error_names['NOERROR']) {
         return $standard;
      }
      $term = $term[0];
```

```
      $term = tokenize($term, $global_parens, $global_unary_operators,
                       $global_binary_operators);
      if ($term[1] != $global_error_names['NOERROR']) {
         return $standard;
      }
      $term = $term[0];
      $term = substitute($term, $global_constants);
      if ($term[1] != $global_error_names['NOERROR']) {
         return $standard;
      }
      $term = $term[0];
      $term = build_tree($term, $global_parens, $global_unary_operators, $global_binary_operators);
      if ($term[1] != $global_error_names['NOERROR']) {
         return $standard;
      }
      $term = $term[0][0];
      $term = build_stack($term);
      if ($term[1] != $global_error_names['NOERROR']) {
         return $standard;
      }
      $term = $term[0];
      $term = substitute($term, array('x' => '1'));
      if ($term[1] != $global_error_names['NOERROR']) {
         return $standard;
      }
      $term = $term[0];
      $term = process_stack($term, $global_unary_operators, $global_binary_operators);
      if (
            $term[1] != $global_error_names['NOERROR'] and
            $term[1] != $global_error_names['DIVBYZERO'] and
            $term[1] != $global_error_names['OUTOFRANGE']
         ) {
         return $standard;
      }
      return $value;
   }
   break;
case ('c'):
   global $global_colors;
   if ($errormsg) {
      if (!is_numeric($value) or !is_int($value + 0)) {
         return "'$value' ist kein g&uuml;ltiger Farbcode.";
      } elseif ($value < 0 or count($global_colors) <= $value) {
         return "'$value' ist kein g&uuml;ltiger Farbcode.";
      } else {
         return '';
      }
   } else {
      if (!is_numeric($value) or !is_int($value + 0)) {
         return $standard;
      } elseif ($value < 0 or count($global_colors) <= $value) {
         return $standard;
      } else {
         return $value;
      }
   }
   break;
case ('p'):
   if ($errormsg) {
      if (!is_numeric($value)) {
         return "'$value' ist keine g&uuml;ltige Prozentzahl.";
      } elseif ($value < 0 or 100 < $value) {
         return "'$value' ist keine g&uuml;ltige Prozentzahl.";
      } else {
         return '';
      }
   } else {
      if (!is_numeric($value)) {
         return $standard;
      } elseif ($value < 0 or 100 < $value) {
         return $standard;
      } else {
         return $value;
      }
   }
   break;
case ('b'):
   if ($errormsg) {
      if (strtolower($value) == 'true' or strtolower($value) == 'false') {
         return '';
```

```
      } else {
         return "Kein Boolean: '$value'.";
      }
   } else {
      if (strtolower($value) == 'true' or strtolower($value) == 'false') {
         return $value;
      } else {
         return $standard;
      }
   }
   break;
case ('e'):
   if ($errormsg) {
      $temp = check_parens($value, $global_parens);
      if ($temp > -1) {
         $err = '';
         for ($i = 0; $i < strlen($value); ++$i) {
            if ($i == $temp) {
               $err .= "<font color=\"red\">" . htmlspecialchars($value{$i}) . "</font>";
            } else {
               $err .= htmlspecialchars($value{$i});
            }
         }
         return "Die rote Klammer im Term passt nicht: '$err'.";
      }
      $term = check_minus($value, $global_parens);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Einf&uuml;gen der Minuszeichen ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $poss = array_merge(range('a', 'z'), range('A', 'Z'));
      foreach ($global_parens as $pair) {
         array_push($poss, $pair[0]);
      }
      $term = add_times($term, $poss, '*');
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Einf&uuml;gen der Malzeichen ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = tokenize($term, $global_parens, $global_unary_operators,
                       $global_binary_operators);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Bei der Zerlegung in Tokens ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = substitute($term, $constants);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Bei der Substitution der Konstanten ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = build_tree($term, $global_parens, $global_unary_operators, $global_binary_operators);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Aufbau des <i>parse tree</i> ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0][0];
      $term = build_stack($term);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Beim Aufbau des Stacks ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      $term = $term[0];
      $term = process_stack($term, $global_unary_operators, $global_binary_operators);
      if ($term[1] != $global_error_names['NOERROR']) {
         return "Bei der Berechnung des Funktionswertes ist ein Fehler aufgetreten: <i>"
              . htmlspecialchars(error_text($term[1])) . "</i>.";
      }
      return '';
   } else {
      $temp = check_parens($value, $global_parens);
      if ($temp > -1) {
         return check_error($value, $kind, false, 'O');
      }
      $term = check_minus($value, $global_parens);
      if ($term[1] != $global_error_names['NOERROR']) {
         return check_error($value, $kind, false, 'O');
```

```
            }
            $term = $term[0];
            $poss = array_merge(range('a', 'z'), range('A', 'Z'));
            foreach ($global_parens as $pair) {
               array_push($poss, $pair[0]);
            }
            $term = add_times($term, $poss, '*');
            if ($term[1] != $global_error_names['NOERROR']) {
               return check_error($value, $kind, false, '0');
            }
            $term = $term[0];
            $term = tokenize($term, $global_parens, $global_unary_operators,
                             $global_binary_operators);
            if ($term[1] != $global_error_names['NOERROR']) {
               return check_error($value, $kind, false, '0');
            }
            $term = $term[0];
            $term = substitute($term, $constants);
            if ($term[1] != $global_error_names['NOERROR']) {
               return check_error($value, $kind, false, '0');
            }
            $term = $term[0];
            $term = build_tree($term, $global_parens, $global_unary_operators, $global_binary_operators);
            if ($term[1] != $global_error_names['NOERROR']) {
               return check_error($value, $kind, false, '0');
            }
            $term = $term[0][0];
            $term = build_stack($term);
            if ($term[1] != $global_error_names['NOERROR']) {
               return check_error($value, $kind, false, '0');
            }
            $term = $term[0];
            $term = process_stack($term, $global_unary_operators, $global_binary_operators);
            if (
                  $term[1] != $global_error_names['NOERROR'] and
                  $term[1] != $global_error_names['DIVBYZERO'] and
                  $term[1] != $global_error_names['OUTOFRANGE']
               ) {
               return check_error($value, $kind, false, '0');
            }
            return $term[0];
         }
         break;
      default:
         if ($errormsg) {
            return "Unbekannter Wertetyp: '$kind'.";
         } else {
            return $standard;
         }
   }
}

##########################################################################################
# Funktion  : unpack_function_terms                                                      #
# Zweck     : Funktionsparameter aus $_REQUEST-Array in Array extrahieren                #
# Eingabe   : $_REQUEST-Array            - $request                                      #
# Ausgabe   : Funktionsparameter als Array                                               #
##########################################################################################

function unpack_function_terms($request) {
   $result = array();
   foreach ($request as $name => $value) {
      if (preg_match('/^function_(\d+)_(.+)$/i', $name, $matches)) {
         switch ($matches[2]) {
            case ('term'):
               $result[$matches[1]]['term'] = $value;
               break;
            case ('color'):
               $result[$matches[1]]['color'] = $value;
               break;
            case ('delete'):
               $result[$matches[1]]['delete'] = $value;
               break;
            case ('param_name'):
               $result[$matches[1]]['param']['name'] = $value;
               break;
            case ('param_from'):
               $result[$matches[1]]['param']['from'] = $value;
               break;
            case ('param_to'):
```

```
               $result[$matches[1]]['param']['to'] = $value;
               break;
            case ('param_step'):
               $result[$matches[1]]['param']['step'] = $value;
               break;
         }
      }
   }
   return $result;
}

##############################################################################
# Funktion  : pack_function_terms                                            #
# Zweck     : Funktionsparameter aus Array in Werte f"ur $_REQUEST-Array umwandeln #
# Eingabe   : Funktionsparameter         - $functions                        #
# Ausgabe   : Array mit Werten f"ur $_REQUEST-Array                          #
##############################################################################

function pack_function_terms($functions) {
   $result = array();
   foreach ($functions as $number => $properties) {
      $result["function_${number}_term"]       = $properties['term'];
      $result["function_${number}_color"]      = $properties['color'];
      $result["function_${number}_delete"]     = $properties['delete'];
      $result["function_${number}_param_name"] = $properties['param']['name'];
      $result["function_${number}_param_from"] = $properties['param']['from'];
      $result["function_${number}_param_to"]   = $properties['param']['to'];
      $result["function_${number}_param_step"] = $properties['param']['step'];
   }
   return $result;
}

##############################################################################
# Funktion  : unpack_asymptote_terms                                         #
# Zweck     : Asymptotenparameter aus $_REQUEST-Array in Array extrahieren   #
# Eingabe   : $_REQUEST-Array            - $request                          #
# Ausgabe   : Asymptotenparameter als Array                                  #
##############################################################################

function unpack_asymptote_terms($request) {
   $result = array();
   foreach ($request as $name => $value) {
      if (preg_match('/^asymptote_(\d+)_(.+)$/i', $name, $matches)) {
         switch ($matches[2]) {
            case ('p1_x'):
            case ('p1_y'):
            case ('p2_x'):
            case ('p2_y'):
            case ('color'):
            case ('dashed'):
            case ('delete'):
               $result[$matches[1]][$matches[2]] = $value;
               break;
         }
      }
   }
   return $result;
}

##############################################################################
# Funktion  : pack_asymptote_terms                                           #
# Zweck     : Asymptotenparameter (Array) in Werte f"ur $_REQUEST-Array umwandeln #
# Eingabe   : Asymptotenparameter        - $asymptotes                       #
# Ausgabe   : Array mit Werten f"ur $_REQUEST-Array                          #
##############################################################################

function pack_asymptote_terms($asymptotes) {
   $result = array();
   foreach ($asymptotes as $number => $properties) {
      $result["asymptote_${number}_p1_x"]   = $properties['p1_x'];
      $result["asymptote_${number}_p1_y"]   = $properties['p1_y'];
      $result["asymptote_${number}_p2_x"]   = $properties['p2_x'];
      $result["asymptote_${number}_p2_y"]   = $properties['p2_y'];
      $result["asymptote_${number}_color"]  = $properties['color'];
      $result["asymptote_${number}_dashed"] = $properties['dashed'];
      $result["asymptote_${number}_delete"] = $properties['delete'];
   }
   return $result;
}
```

```
##############################################################################################
# Funktion  : get_joined_indices                                                             #
# Zweck     : x-Werte erhalten, die in einer von zwei Punktmengen enthalten sind             #
# Eingabe   : Zwei Arrays                - $array1, $array2                                  #
# Ausgabe   : Array mit allen Schl"usseln der beiden Arrays                                  #
##############################################################################################

function get_joined_indices($array1, $array2) {
   return array_unique(array_merge(array_keys($array1), array_keys($array2)));
}

##############################################################################################
# Funktion  : get_missing_indices                                                            #
# Zweck     : Extrahiert Indices, die nicht in einem Array sind                              #
# Eingabe   : Array mit Indices          - $indices,                                         #
#             Array (mit Punkten)        - $array                                          #
# Ausgabe   : Array mit Werten aus $indices, die keine Schl"ussel von $array sind            #
##############################################################################################

function get_missing_indices($indices, $array) {
   $result = array();
   foreach ($indices as $key) {
      if (!array_key_exists($key, $array)) {
         array_push($result, $key);
      }
   }
   return $result;
}

##############################################################################################
# Funktion  : http_build_query                                                               #
# Zweck     : Baut einen POST-String aus einem Array auf                                     #
# Eingabe   : Siehe PHP-Dokumentation                                                        #
# Ausgabe   : POST-String                                                                    #
# Sonstiges : "Ubernommen von php at brayra dot nospam dot com von der PHP-Website           #
##############################################################################################

if(!function_exists('http_build_query')){
   function http_build_query($formdata, $numeric_prefix = ''){
       return _http_build_query($formdata,$numeric_prefix);
   }
   function _http_build_query($formdata, $numeric_prefix = '',$key_prefix=''){
       if($numeric_prefix != '' && !is_numeric($numeric_prefix)){
           $prefix=$numeric_prefix;
       } else {
           $prefix = '';
       }
       if(!is_array($formdata)) return '';
       $str='';
       foreach($formdata as $key => $val){
           if(is_numeric($key))$key=$prefix.$key;
           if($str != '') $str.='&';
           if($key_prefix != '') {
               $mykey = $key_prefix."[$key]";
           } else {
               $mykey=&$key;
           }
           if(is_array($val)){
               $str.=_http_build_query($val,'',$mykey);
           } else {
               $str.=$mykey.'='.urlencode($val);
           }
       }
       return $str;
   }
}

##############################################################################################
# Funktion  : populate_request                                                               #
# Zweck     : F"ullt $_REQUEST mit Werten, falls Felder nicht definiert sind                 #
# Eingabe   : $_REQUEST-Array          - $request,                                           #
#             zu pr"ufende Felder      - $items,                                             #
#             Standard-Werte           - $standards                                          #
# Ausgabe   : Bearbeitetes $_REQUEST-Array                                                   #
# Sonstiges : - F"ur Standardwerte siehe plotter_constants.php                               #
#             - $items = ([$_REQUEST-key] => [$standards-key], ...)                          #
##############################################################################################

function populate_request($request, $items, $standards) {
   foreach ($items as $rq_key => $st_key) {
```

```
      if (is_null($request[$rq_key]) or $request[$rq_key] == '') {
         $request[$rq_key] = $standards[$st_key][1];
      }
   }
   return $request;
}

##############################################################################
# Funktion  : form_text                                                      #
# Zweck     : Erstellt Textfeld f"ur HTML-Formular                           #
# Eingabe   : $_REQUEST-Array         - $request,                            #
#             Feldname                - $name,                               #
#             'size'-Parameter        - $size,                               #
#             'maxlength'-Parameter   - $maxlength                           #
# Ausgabe   : HTML-Code f"ur Textfeld                                        #
##############################################################################

function form_text($request, $name, $size, $maxlength) {
   return '<input type="text" size="'. $size
        . '" maxlength="' . $maxlength
        . '" name="' . htmlspecialchars($name)
        . '" value="' . htmlspecialchars($request[$name]) . '">';
}

##############################################################################
# Funktion  : form_img_type                                                  #
# Zweck     : Erstellt Auswahlfeld f"ur Bildformat f"ur HTML-Formular        #
# Eingabe   : $_REQUEST-Array         - $request,                            #
#             Feldname                - $name,                               #
#             Standardwerte           - $standards,                          #
#             Einr"ucktiefe           - $indent                              #
# Ausgabe   : HTML-Code f"ur Typauswahl                                      #
# Sonstiges : F"ur Standardwerte siehe plotter_constants.php                 #
##############################################################################

function form_img_type($request, $name, $standards, $indent) {
   $result = '';
   $result .= str_repeat(' ', $indent) . '<select name="' . htmlspecialchars($name) . '" size="1">' . "\n";
   $image_type = $request[$name];
   if (!is_numeric($image_type) or !is_int($image_type + 0) or $image_type < 0
       or count($standards{'img_types'}[1]) <= $image_type) {
      $image_type = 0;
   }
   if (!(imagetypes() & $standards{'img_types'}[1][$image_type][1])) {
      $image_type = 0;
   }
   while (!(imagetypes() & $standards{'img_types'}[1][$image_type][1])) {
      if ($image_type >= count($standards{'img_types'})) {
         die;
      }
      ++$image_type;
   }
   foreach ($standards{'img_types'}[1] as $key => $properties) {
      $result .= str_repeat(' ', $indent)
               . '   <option' . ($key == $image_type ? ' selected' : '')
               . ' value="' . $key . '">'
               . htmlspecialchars($properties[0]) . "</option>\n";
   }
   $result .= str_repeat(' ', $indent) . '</select>' . "\n";
   return $result;
}

##############################################################################
# Funktion  : form_color_select                                              #
# Zweck     : Erstellt Auswahlfeld f"ur Farben f"ur HTML-Formular            #
# Eingabe   : $_REQUEST-Array         - $request,                            #
#             Feldname                - $name,                               #
#             Array mit Farben        - $colors,                             #
#             Einr"ucktiefe           - $indent                              #
# Ausgabe   : HTML-Code f"ur Farbenauswahlfeld                               #
# Sonstiges : F"ur Farbenarray siehe plotter_constants.php                   #
##############################################################################

function form_color_select($request, $name, $colors, $indent) {
   $result = '';
   $result .= str_repeat(' ', $indent) . '<select name="' . htmlspecialchars($name) . '" size="1">' . "\n";
   $color = $request[$name];
   if (!is_numeric($color) or !is_int($color + 0) or $color < 0 or count($colors) <= $color) {
      $color = 0;
   }
```

```
      foreach ($colors as $key => $properties) {
         $result .= str_repeat(' ', $indent)
                  . '   <option' . ($key == $color ? ' selected' : '')
                  . ' value="' . $key . '">'
                  . htmlspecialchars($properties[0]) . "</option>\n";
      }
      $result .= str_repeat(' ', $indent) . '</select>' . "\n";
      return $result;
   }

   ##############################################################################
   # Funktion  : form_add_hidden_fields                                         #
   # Zweck     : Erstellt versteckte Felder f"ur HTML-Formular                  #
   # Eingabe   : $_REQUEST-Array         - $request,                            #
   #             Ausnahmen               - $exceptions,                         #
   #             Einr"ucktiefe           - $indent                              #
   # Ausgabe   : HTML-Code f"ur versteckte Felder                               #
   # Sonstiges : - ALLE Felder von $_REQUEST werden ausgegeben, Ausnahme s.u.   #
   #             - Felder, die mit einer RegEx aus $exceptions passen, werden nicht  #
   #               als versteckte Felder ausgegeben ($exceptions ist also ein Array)  #
   ##############################################################################

   function form_add_hidden_fields($request, $exceptions, $indent) {
      $result = '';
      foreach ($request as $field => $content) {
         foreach ($exceptions as $regex) {
            if (preg_match($regex, $field)) {
               continue 2;
            }
         }
         $result .= str_repeat(' ', $indent) . '<input type="hidden" name="'
                  . htmlspecialchars($field) . '" value="'
                  . htmlspecialchars($content) . '">' . "\n";
      }
      return $result;
   }

?>
```

4.16 Datei „units/style.css“

```
   /*****************************************************************************/
   /* Dies ist die Datei 'style.css'                                            */
   /*                                                                           */
   /* Sie enth"alt die Stildefinitionen f"ur die HTML-Seiten des                */
   /* Parser/Plotter-Projekts                                                   */
   /*                                                                           */
   /* Diese Datei darf frei als Ganzes oder in Teilen f"ur jegliche nicht-      */
   /* kommerzielle Zwecke verwandt werden, solange folgede Bedingungen erf"ullt */
   /* werden:                                                                   */
   /*    - Dieser Blockkommentar mit den Nutzungsbedingungen muss unver"andert  */
   /*      in allen Nachfolgedateien / -ver"offentlichungen am Datei- / Textkopf*/
   /*      verwandt werden                                                      */
   /*    - Der Quellcode des muss jedem Benutzer von allen Programmen, die ihn  */
   /*      verwenden, einsehbar gemacht werden                                  */
   /*                                                                           */
   /* Sollten Sie diesen Code, in Teilen oder als Ganzes, auch portiert in      */
   /* andere Programmiersprachen oder unter blo"ser Verwendung des Algorithmus  */
   /* oder von Teilen von verwandten Algorithmen, f"ur kommerzielle Zwecke nutzen*/
   /* wollen, so kontaktieren Sie mich bitte.                                   */
   /*                                                                           */
   /* Ich habe den vorliegenden Code von Grund auf selbst programmiert und mich */
   /* dabei auf keine anderen als Standardalgorithmen gest"utzt. Insbesondere den*/
   /* Tokenizer, der Parser und die Schrittweitensteurerung habe ich von Grund  */
   /* auf und ohne Zuhilfenahme anderer Quellen selbst programmiert. Sollte ich */
   /* mit dem vorliegenden Code bestehende Copyrights oder Patente verletzen oder*/
   /* den Urheber geistigen Eigentums vorenthalten, so geschieht dies           */
   /* unabsichtlich. Bitte kontaktieren Sie mich mittels untenstehender E-Mail- */
   /* Adresse oder "uber den Sysadmin.                                          */
   /*                                                                           */
   /* Der Autor "ubernimmt keine Haftung f"ur eventuelle durch die Nutzung dieses*/
   /* entstehende oder entstandene mittelbare oder unmittelbare Sch"aden jedweder Art. */
   /* Nutzung auf eigene Gefahr.                                                */
   /*                                                                           */
   /* Bitte schicken Sie mir keine E-Mails mit Bitte um die Erkl"arung der      */
   /* Funktionsweise des Codes oder von Teilen des Codes; ich habe mich bem"uht,*/
```

```
   /* den Code so gut wie m"oglich zu kommentieren und habe leider nicht genug         */
   /* Zeit, um E-Mails "uber technische Details zu beantworten.                        */
   /*                                                                                  */
   /* (c) Jan Olligs, 2004                                                             */
   /* Alle Rechte vorbehalten                                                          */
   /*                                                                                  */
   /* Jan.Olligs@gmx.net                                                               */
   /**********************************************************************************/

body { background-color:#D7D7D7; }

td.version1 { background-color:#B0B0B0; }
td.version2 { background-color:#C0C0C0; }
td.current  { background-color:#D0D0D0; }
td.body     { background-color:#E0E0E0; }

legend      { font-weight:bold; }
```

Literaturverzeichnis

[Cor.Algo] Cormen, T. H. et al.: **Introduction to Algorithms** – *Second Edition.* Cambridge, Massachusetts: MIT Press. 2001. ISBN: 0 262 53196 8

[PHP.Doku] **PHP** – *Dokumentation.* `<http://www.php.net>` – *Stand 10.04.2004*

[Sed.Algo] Sedgewick, R.: **Algorithmen in C++** – *Teil 1-4.* 3., überarb. Aufl. München: Pearson Studium. 2002. ISBN: 3 8273 7026 4

[Wal.Perl] Wall, L. et al.: **Programmieren mit Perl**. 2. Aufl. Köln: O'Reilly. 2001. ISBN: 3 89721 144 0