BG/BRG Salzburg
Fachbereichsarbeit Informatik

Maschinelles Gleichungslösen

Lösung von Gleichungen bis einschließlich des 4. Grades in einer Variablen mit reellen Koeffizienten

Reinhard Jäger 8A

eingereicht am .. Jänner 1999 bei Prof. Prem

BG/BRG
Franz-Josef-Kai 41
5020 Salzburg

Salzburg, 1999

Ich erkläre, daß ich die Arbeit selbständig angefertigt und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Salzburg, den 17. Februar 1999 Reinhard Jäger

Inhaltsverzeichnis

Kapitel 1

Einleitung

Allgemein lösbar ist heute bereits die Gleichung fünften Grades in einer Variablen mit komplexen Koeffizienten. Gleichungen höheren Grades sind nur mehr in Spezialfällen lösbar. Es existiert jedoch keine allgemeine Lösungsformel. Gleichungen dieser Art müssen durch Näherungsverfahren gelöst werden. Die Gleichung fünften Grades bildet insofern einen Sonderfall, daß sie nicht mehr durch Wurzelausdrücke gelöst werden kann.
Es muß die in dieser Arbeit verwendete *Reduktionstransformation* verallgemeinert werden.[1]

1.1 Zielsetzung der Arbeit

Da die Formeln erheblichen Rechenaufwand bedingen, wird der Algorithmus in einer prozeduralen Programmierweise implementiert. Das ermöglicht eine universelle Softwareschnittstelle, sehr im Gegensatz zu fertigen Softwarepaketen, die nicht erlauben, diese Routinen z.B. in Graphikprogramme einzubauen. Die Lösungsmenge der Programme liegt im Bereich der komplexen Zahlen. Im Hinblick auf spätere Verwendung in Graphikprogrammen wird die Einschränkung gemacht, daß die Koeffizienten im Bereich der reellen Zahlen angenommen werden.
Es gibt viele verschiedene Ansätze: man könnte z.B. die Lösungsmenge auf ganze Zahlen einschränken. In dieser Arbeit wird jedoch die Lösungsmenge der *komplexen Zahlen* verwendet. Der verwendete Algorithmus garantiert eine allgemeine Lösung. Die Genauigkeit ist jedoch der Zahlendarstellung nach IEEE 754 unterworfen. Diese Zahlendarstellung wird in vielen Programmiersprachen verwendet, so auch in *Borland Pascal 7.0* oder *Borland C++ 3.1* , den in dieser Arbeit verwendeten Sprachen.

[1] vgl. King S. 3

Kapitel 2

Lineare Gleichung

Für die rechenpraktische Implementierung müssen nur 2 Fälle abgefangen werden, die es wert sind, erwähnt zu werden.

$$ax + b = 0$$

1. Wenn a Null ist, und b ungleich Null, dann existiert keine Lösung.
2. Wenn sowohl a als auch b Null sind, dann ist die Gleichung für jedes x erfüllt, wobei 2 Diskussionsstandpunkte möglich sind:

$$a) \qquad ax = -b$$

$$b) \qquad x = -\frac{b}{a}$$

Wobei die Variante b) dann eine unerlaubte Operation darstellen würde. Diese Diskussion ist aber fürs Maschinenrechnen belanglos.
Diese Fälle sind bereits in der Prozedur „welche Gleichung“behandelt.

Kapitel 3

Quadratische Gleichung

Was am Anfang als einfaches Programm erschien, entpuppte sich dann als wahrer „Augenöffner“bezüglich Rechengenauigkeit und Algorithmenstrategie. Um die Problematik besser zu veranschaulichen, wurde ein eigenes Programm für die Lösung einer bereits normierten quadratischen Gleichung von der Form $x^2 + px + q = 0$ geschrieben. Das Programm rechnet die Lösung der Gleichung auf zwei verschiedene Arten. Anschließend werden die Koeffizienten zur Kontrolle zurückgerechnet. Zuerst werden die Lösungen nach der üblichen Formel

$$x_{1,2} = -\frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q}$$

berechnet.
Anschließend wird die Formel auf die verwendete Maschine angepaßt. Bei unseren Betrachtungen gehen wir von der üblichen Formel aus. Diese Formel ist nur für das händische Rechnen geeignet. Es ist leicht, eine Formel hinzuschreiben. Entscheidend ist ihre Berechenbarkeit. Was ist nun der Unterschied zwischen händischem und maschinellem Rechnen ?
Beim händischen Rechnen kann ich die Stellenanzahl während des Rechnens den Notwendigkeiten des Algorithmus anpassen. Bei Gleitkommabibliotheken habe ich ein statisches Format, welches während der gesamten Laufzeit gleich bleibt. Bei Ganzzahlformaten kann ich durch dynamische Programmierung das Zahlenformat während der Laufzeit den Verhältnissen der Formel anpassen. Das Überlaufbit im Statusregister des Prozessors kann hier als Steuermechanismus verwendet werden. Bei Verwendung von Gleitkommazahlen ist dies nicht möglich. Beim Format DOUBLE stehen nur 64 Bit zu Darstellung einer Zahl zur Verfügung, wobei 11 Bit für den Exponenten, 52 für die Mantisse und 1 fürs Vorzeichen verwendet wird. Alles was über dieses Format hinausgeht, muß über Rundungsalgorithmen abgeschnitten werden.
Welche Probleme wirft nun die herkömmliche Formel auf ?

1. Wenn der Betrag von p sehr viel größer als der Betrag von q ist, wird der Wurzelwert $\frac{p}{2}$, und es tritt eine Auslöschung bei einer Lösung der Gleichung ein.

2. Bei großem p kann ein Überlauf bei der Berechnung von $\left(\frac{p}{2}\right)^2$ stattfinden.

Die betragsmäßig größere Lösung kann ohne Probleme nach der Formel berechnet werden.[1] Der maschinengerechte Algorithmus berechnet daher zuerst die betragsmäßig größere Lösung und berechnet die zweite Lösung nach dem Satz von Vieta:

$$x_1 \cdot x_2 = q$$
$$x_2 = \frac{q}{x_1}$$

Dadurch wird die Auslöschung vermieden. Wie verhindern wir nun das Überlaufproblem bei $\left(\frac{p^2}{4}\right)$?
Wir formen um

$$-\frac{p}{2} \pm \sqrt{\left(\frac{p}{2}\right)^2 - q} = -\frac{p}{2} \pm p\sqrt{\frac{1}{4} - \frac{q}{p^2}}$$

Damit können wesentlich größere Zahlenbereiche bearbeitet werden.
Zum Schluß muß noch der Term $\frac{q}{p^2}$ diskutiert werden. Eine Implementierung q/(p*p) ist schlecht, weil hier wieder eine große Zahl auftritt. Vielmehr muß q/p/p programmiert werden. Die Zweiadreßmaschine paßt von links nach rechts und rechnet zuerst den Wert q/p und dividiert dann nochmals durch p. Hier ist bei großem p außerdem noch zu beachten, daß der Compiler von normalisierten zu denormalisierten Zahlen übergeht. Wenn das nicht ausreicht, meldet er einen sogenannten Unterlauf. Bei Pascal meldet er diesen Unterlauf nicht an den User weiter, sondern setzt automatisch die Rechnung mit Null fort.
Beim Experimentieren muß man gegebenenfalls die Prozedur „Haendisch“ausklammern, da der Compiler sonst einen Gleitkommaüberlauf meldet und nur mehr die Routine „Maschine“richtig funktioniert.

[1] vgl. Gander Seite 17

Kapitel 4

Lösungsansatz

Dieses Reduktionsverfahren funktioniert folgendermaßen:

Die Gleichung vierten Grades wird auf eine Hilfsgleichung dritten Grades zurückgeführt. Diese Hilfsgleichung wird allgemein als Resolvente bezeichnet. Sie hat einen geringeren Grad als die ursprüngliche Gleichung. Die Transformation einer allgemeinen Gleichung in eine reduzierte Gleichung bedeutet, daß man in der Unbekannten eine Substitution durchführt, in der Art, daß der neuen Unbekannten, das heißt beim Übergang von x auf z ein Koeffizient der Gleichung, es können auch mehrere sein, Null wird. Diese Art von Transformation nennt man *Tschirnhausentransformation*. Der einfachste Fall der Tschirnhausentransformation besteht darin, den Koeffizienten der Unbekannten mit der zweithöchsten Potenz Null werden zu lassen. Dieses Verfahren wird ja in dieser Arbeit angewendet. Die Tschirnhausentransformation ist auch allgemein durchführbar. So kann durch Substitution in der Unbekannten eine allgemeine Gleichung vierten Grades in eine spezielle Gleichung vierten Grades umgeformt werden. Dabei ist aber zu beachten, daß für die Tschirnhausentransformation selbst eine Hilfsgleichung zu lösen ist. Diese Hilfsgleichung ist aber wiederum eine Gleichung um einen Grad reduziert, also in unserem Fall eine Gleichung dritten Grades.

Satz von Abel:
Der Satz von Abel besagt, daß eine allgemeine Lösung einer Gleichung durch Wurzelzeichen nur bis zum vierten Grad möglich ist. Zur Lösung der Gleichung fünften Grades sind andere Funktionen notwendig. Bei Anwendung des Reduktionsverfahrens auf eine Gleichung fünften Grades, ergibt nämlich eine allgemeine Hilfsgleichung sechsten Grades, also höheren Grades als die Ausgangsgleichung. Die Wurzelfunktion reicht nur einschließlich bis zur Lösung der Gleichung vierten Grades aus. Die Gleichung fünften Grades kann exakt

gelöst werden und zwar unter Verwendung von elliptischen Modulfunktionen.
Was bedeuten nun elliptische Modulfunktionen ? Sie können als Verallgemeinerung von periodischen Funktionen verstanden werden. Die erste hier vorgenommene Verallgemeinerung einer periodischen Funktion lautet:

$$f(z) + n \cdot a = f(z)$$

n ist ganzzahlig und a stellt die Periode dar. Einfachstes Beispiel für so eine Funktion sind die Winkelfunktionen, wie Sinus usw. Eine weitere Verallgemeinerung stellt eine Funktion mit zwei Perioden dar.

$$f(z) + n_1 \cdot a_1 + n_2 \cdot a_2 = f(z)$$

n_1 und n_2 sind wieder ganzzahlig. a_1 und a_2 stellen die Perioden dar. Man nennt Funktionen mit zwei Perioden elliptische Funktionen. Es gibt im Bereich der komplexen Zahlen bei Funktionen in einer Variablen nur Funktionen mit einer oder zwei Perioden. Funktionen mit drei Perioden können in diesem Bereich nicht existieren. Was sind nun Modulfunktionen ?
Eine weitere Verallgemeinerung der Periodizität ergibt eine Funktion von folgender Gestalt:

$$f\left(\frac{az+b}{cz+d}\right) = f(z)$$

Was ist nun der Unterschied zur einfach periodischen Funktion ? In diesem Fall ist a_n die Stelle des einfachen Summanden a, der ja der Periode entspricht, eine linear gebrochene Funktion eingesetzt. Das nennt man eine Modulfunktion. Mit ihrer Hilfe ist die Lösung der Gleichung fünften Grades möglich. Es existiert aber zusätlich eine Nebenbedingung. Die Determinante der Koefffizienten a, b, c und d des Bruches muß gleich 1 sein. Nur dann handelt es sich um eine Modulfunktion. Wenn die Determinante nicht gleich eins ist, liegt nochmals eine Verallgemeinerung des Begriffes periodischer Funktionen vor. Man spricht dann von einer automorphen Funktion. Wir faßen die Begriffe nochmals zusammen:
Die automorphen Funktionen haben als Sonderfall die Modulfunktionen. Die Modulfunktionen haben ihrerseits als Sonderfall die periodischen Funktionen. Innerhalb der periodischen Funktionen unterscheiden wir wiederum Funktionen, bei denen die zweite Periode Null ist und gelangen so zurück zu den bekannten Winkelfunktionen wie Sinus etc.
Es konnte trotz intensiver Recherchen im Internet bis 12.12.1998 keine Implementierung der Lösungen der Gleichung dritten, vierten oder fünften Grades gefunden werden.

1.Versuch mit dem Computeralgebraprogramm Derive:
Es wurde versucht mit dem kommerziellen algebraisch rechnenden Programm Derive eine Lösung zu versuchen. Der Gedanke war, die Lösungsformel gleichsam in einer allgemeingültigen Programmiersprache nachzuprogramieren, um so eine allgemeine Schnittstelle für spätere Programme zu haben. Es wurde sowohl die deutsche (Version 4.07) als auch die englische Version 4.03 des Programmes für Windows verwendet.
Es wurde zuerst die Lösung der Gleichung dritten Grades versucht. Der Ausdruck scheint auf den ersten Blick vielversprechend. Doch bei genauerer Betrachtung erweist sich das als Trugschluß. Dafür sind die zwei folgenden Punkte ausschlaggebend:

1. Sämtliche verwendete mathematische Notationen wie Bruchstrich, Wurzelzeichen, Arcustangensfunktion usw. scheinen exakt. Es bleibt aber dem Betrachter überlassen, wie diese Konventionen zu interpretieren sind. Im vorliegenden Fall, weiß man nie ob sie im Bereich der reellen oder komplexen Zahlen zu interpretieren sind. Auch wenn man die Formeln in der Einzeilenversion des Eingabefensters betrachtet, werden die in Programmiersprachen üblichen Bezeichnungen verwendet. z.B.: SQRT für die Quadratwurzelfunktion. Es muß sich aber hier um eine komplexe Implementation handeln. Die eingebaute Hilfe schweigt sich aber darüber aus.

2. In den von Derive vorgeschlagenen Formeln kommen mehrere Quadratwurzelzeichen vor. Nachdem aber jedes Wurzelzeichen zwei Lösungen hat, bedeutet das, daß die gesamte Zahl der Lösungen größer als der Grad der Gleichung wäre. Es existieren aber wie weiter unten ausgeführt wird, Nebenbedingungen, die auf die richtige Anzahl der Lösungen eingrenzen. Auch diese Nebenbedingungen werden in einem kommerziellen Produkt wahrscheinlich absichtlich geheimgehalten.

Gerade diese zwei Punkte stellen aber die Hauptschwierigkeit der Arbeit dar. Zusammenfassend kann gesagt werden, daß die von Derive so schön ausgegebenen Formeln, sehr wenig über den wirklichen Rechnungsgang aussagen. Es ist nur für Leute interessant, die nur am Ergebnis und nicht am Lösungsweg, also dem eigentlichen „know how“interessiert sind.
Auf einem Pentium II Rechner mit 266 MHz brauchte Derive über acht Stunden für die Lösung des im Anhang angeführten Beispiels. In der etwas älteren englischen Version immerhin noch über vierzig Minuten. Was hier wirklich berechnet wird und ob wirklich der „exact mode“beibehalten wird, bleibt vollkommen unklar. Dieser Versuch blieb ein Fehlschlag.
Im Anhang sind die von Derive angegebenen Formeln angeführt.

Kapitel 5

Rückrechnung

Nach anfänglichen Versuchen und fehlerhaften Wurzel Routinen, bei denen zwar betragsmäßig richtige Werte erzielt wurden, aber falsche Zuordnungen der Riemannschen Blätter erfolgten, war klar, daß nur eine Rückrechnung eine effektive Kontrolle sein kann. So wurde bereits im Anfangsstadium die Rückrechnung in Angriff genommen.

verallgemeinerter Satz von Vieta:
Wenn man die Lösungen einer Gleichung kennt, so kann man die Koeffizienten der Ausgangsgleichung zurückechnen. Das gilt allgemein für alle hier besprochenen Gleichungen sogar mit komplexen Koeffizienten. Die Rückrechnung ist also über das ursprüngliche Ziel der Arbeit hinausgehend, auch für Gleichungen mit komplexen Koeffizienten geeignet.
Die Rückrechnung bezieht sich immer auf die normierte Form der Gleichung, also jene Form bei der der Koeffizient der Variable mit dem höchsten Exponenten 1 ist. Eine Gleichung darf die Koeffizienten nicht ändern, wenn ich die Lösungen beim Rückrechnen untereinander beliebig vertausche. Das heißt sämtliche Koeffizienten müssen einwertige Funktionen sein.

einwertige Funktion
Sie ändert ihren Funktionswert nicht, bei Vertauschung der Variablen.

$$f(x_1x_2x_3) = x_1x_2 + x_1x_3 + x_2x_3$$
$$f(x_2x_1x_3) = x_2x_1 + x_2x_3 + x_1x_3$$

Die beiden Funktionswerte müssen gleich sein; alle Formeln des Vietasatzes sind einwertige Funktionen.

mehrwertige Funktionen

Sie ändert ihren Funktionswert bei Vertauschung der Variablen

$$f(x_1x_2x_3) = x_1 + x_2x_3$$
$$f(x_2x_1x_3) = x_2 + x_1x_3$$

Es wurde wieder x_1 mit x_2 vertauscht und die Funktionswerte sind verschieden. Diese mehrwertigen Funktionen spielen bei der Rückrechnung keine Rolle.

Quadratische Gleichung:

$$x^2 + px + q = 0$$

$$p = -(x_1 + x_2)$$
$$q = x_1 \cdot x_2$$

Kubische Gleichung:

$$x^3 + Ax^2 + Bx + C = 0$$

$$A = -(x_1 + x_2 + x_3 + x_4)$$
$$B = x_1 \cdot x_2 + x_1 \cdot x_3 + x_2 \cdot x_3$$
$$C = -x_1 \cdot x_2 \cdot x_3$$

Gleichung vierten Grades:

$$x^4 + Ax^3 + Bx^2 + Cx + D = 0$$

$A = -(x_1 + x_2 + x_3 + x_4)$

oder allgemein formuliert:
Der Koeffizient A ist die negative Summe der Lösungen

$B = x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + x_3x_4$

oder allgemein formuliert:
Der Koeffizient B ist die Summe der Zweierprodukte der Lösungen.

$C = -(x_1x_2x_3 + x_1x_2x_4 + x_1x_3x_4 + x_2x_3x_4)$

oder allgemein formuliert:

Der Koeffizient C ist die negative Summe der Dreierprodukte der Lösungen.

$D = x_1x_2x_3x_4$

oder allgemein formuliert:
Der Koeffizient D ist das Produkt der Lösungen

Die Rückrechnung wurde ganz getrennt implementiert, so daß sie auch weggelassen werden kann. Die komplexe Zahl wird als record oder structure implementiert. Es wurde die objektorientierte Version in C++ von struct gewählt, es könnte aber auch ebenso die ältere rein prozedurale C-Version verwendet werden.

```
TYPE COMPLEX = RECORD
                  re : DOUBLE;
                  im : DOUBLE;
               END;
```

```
struct komplex
 {
    long double re;
    long double im;
```

Die Operationen komplexe Addition und Multiplikation, sowie die Negation sind als eigene Unterprogramme implementiert:

in Pascal:

```
PROCEDURE Ca(VAR ZA : COMPLEX;ZB : COMPLEX);
  BEGIN
  ZA.re := ZA.re + ZB.re;
  ZA.im := ZA.im + ZB.im;
  END;

PROCEDURE Cm(VAR ZA : COMPLEX;ZB : COMPLEX);
  VAR
  H : COMPLEX;
  BEGIN
  H.re := ZA.re;
  H.im := ZA.im;
  ZA.re := (ZA.re*ZB.re)- (ZA.im*ZB.im);
```

```
  ZA.im := (H.re*ZB.im) + (H.im *ZB.re);
  END;

PROCEDURE Cneg(VAR ZA : COMPLEX);
  BEGIN
  ZA.re := -ZA.re;
  ZA.im := -ZA.im;
END;
```

in C:

```
struct komplex kadd(struct komplex z1, struct komplex z2)
{
  struct komplex z3;
  z3.re = z1.re + z2.re;
  z3.im = z1.im + z2.im;
  return z3;
}


struct komplex kmult(struct komplex z1, struct komplex z2)
{
  struct komplex z3;
  z3.re = (z1.re*z2.re) - (z1.im*z2.im);
  z3.im = (z1.re*z2.im) + (z1.im*z2.re);
  return z3;
}


struct komplex kneg(struct komplex z)
{
  struct komplex z3;
  z3.re = -z.re;
  z3.im = -z.im;
  return z3;
}
```

Die einzelnen Koeffizienten werden in getrennten Unterprogrammen berechnet. Um von der normierten Form zur Ausgangsgleichung zu gelangen, muß die ganze Gleichung mit dem ersten Koeffizienten multipliziert werden.

Diese Rückrechnung stellte sich im Verlauf der Arbeit als unbedingt notwendig dar. So wurde auch entdeckt, daß bei Turbo Pascal 7.0 mit dem Format Extended bei der sprachinternen Routine SQRT Fehler auftreten. Beim Format double konnte das nicht festgestellt werden. Das ist auch der Grund, warum in C das der Gleitkommaunit im Pentium II Prozessor entsprechende 80 bit lange internationale Format long double verwendet wird und in Pascal nicht. Auf das nur in Pascal verwendete und nicht IEEE konforme Real Format wird gänzlich verzichtet. Auf die in C bereits implementierten komplexen Zahlen wurde ebenso verzichtet, um größtmögliche Transparenz des Programms zu gewährleisten.

Kapitel 6

Signum Funktion

Das Vorzeichen wird in der Maschine durch die Signumfunktion bestimmt. Die Signumfunktion ist dreiwertig. Für alle positiven x ist sie 1, für alle negativen x ist sie -1 und für 0 ist sie 0.
Bei Verwendung von Gleitkommabibliotheken, was ja praktisch immer der Fall ist, können Rundungsfehler etc. auftreten. Die Argumentation, diese sogenannten, im Fachjargon auch manchmal als digitales Rauschen bezeichneten Ungenauigkeiten seien belanglos, stimmt ganz und gar nicht, wenn eine Abfrage auf Null oder einen bestimmten Wert vorliegt. Danach würde zum Beispiel die Doppellösung bei der quadratischen Gleichung nicht eintreten, wenn so ein Rundungsfehler vorliegt. Die Diskriminante könnte zum Beispiel in der 15. Nachkommastelle eine 1 aufweisen.
Man darf daher grundsätzlich bei Gleitkommazahlen einer Bibliothek niemals auf Gleichheit abfragen! Dieser Fehler findet sich leider in den Mathematikbüchern der fünften Klasse, bei der Implementierung der quadratischen Gleichung. Man muß das Problem mit der Umgebung lösen. Ich wähle um den Nullpunkt eine 0.0000000001 kleine Umgebung, sowohl in positiver als auch negativer Richtung auf dem Zahlenstrahl. Diese Umgebung wird traditionellerweise *Epsilonumgebung* genannt. Alle Ergebnisse innerhalb dieser Umgebung werden als Null definiert. Darin sind auch sämtliche Störungen enthalten.

Implementierung in Pascal und C:

```
CONST
EPS = 1.0E-10;                    (* fuer Nullvergleich *)

FUNCTION Sgn(An : DOUBLE) : INTEGER;
```

```
BEGIN
  IF(An < -EPS) THEN Sgn := -1
  ELSE BEGIN
    IF (An > EPS) THEN Sgn := 1
    ELSE  Sgn := 0;
  END;
END;
```

```
#define EPS 1.0e-12             /* fuer Nullvergleich */

int sgn(long double x) {
  int vz;
  if(x < -EPS) vz = -1;
  else {
    if (x > EPS) vz = 1;
      else vz = 0;
  }
  return(vz);
}
```

Eine programmtechnische Anwendung zeigt die Arcuscosinusfunktion:
Hier wird neben der Signumfunktion, wieder die Technik eines Korrekturfaktors oder wertes angewendet, um das Vorzeichen in den einzelnen Quadranten zu ermitteln. Darüber hinaus wird die Signum Funktion noch oft verwendet.

Implementierung in Pascal und C:

```
FUNCTION Arccos(X: DOUBLE) : DOUBLE;
BEGIN
    IF Sgn(X) = 0 THEN Arccos := Pi/2.0
    ELSE Arccos := Arctan(Sqrt(1-X*X)/X) + (1.0 - Sgn(X))*(Pi/2.0);
END;
```

```
long double  arccos(long double x) {
  long double u;
  if (sgn(x) == 0)  return(M_PI/2.0);
  else u = atan(sqrtl(1-x*x)/x) + (1.0 - sgn(x))*(M_PI/2.0);
  return(u);
}
```

Kapitel 7

Kubische Gleichung

Lösung der Gleichung dritten Grades:

Das Unterprogramm „welchegleichung“stellt fest, ob es sich um eine Gleichung dritten Grades handelt.

Die Implementierung in C:

```
void welchegleichung(int *gl) {
  if (sgn(an) != 0) *gl = 4;
  else if (sgn(bn) != 0) *gl =3;
    else if (sgn(cn) != 0) *gl = 2;
      else if (sgn(dn) != 0) *gl = 1;
else if (sgn(en) != 0) *gl = 0;
else *gl = 999;
}
```

„gl“ist eine globale Variable und enthält den Gleichungstyp. Bevor die eigentliche Routine zur Lösung der kubischen Gleichung begonnen werden kann, muß die Gleichung normiert werden.
Die Ausgangsgleichung lautet $b_n x^3 + c_n x^2 + d_n x + e_n = 0$.
Wir dividieren durch b_n und führen gleichzeitig eine neue Bezeichnung der Koeffizienten ein.

$$x^3 + a_2 x^2 + b_2 x + c_2 = 0$$

Nach der Normierung, wird die einfachste Form der *Tschirnhausentransformation* angewandt und eine reduzierte Gleichung dritten Grades durch Einsetzen in die Unbekannte erzielt. Wir setzen für $x = z - \frac{a_2}{3}$.

$$\left(z - \frac{a_2}{3}\right)^3 + a_2 \left(z - \frac{a_2}{3}\right)^2 + b_2 \left(z - \frac{a_2}{3}\right) + c_2 = 0$$

$$z^3 - 3z^2\frac{a_2}{3} + 3z\frac{a_2^2}{9} - \frac{a_2^3}{27} + a_2z^2 - 2a_1^2\frac{z}{3} + a_2\frac{a_2^2}{9} + b_2z - \frac{b_2a_2}{3} + c_2 = 0$$

Wir fassen zusammen und erhalten:

$$z^3 + \left(-\frac{a_2^2}{3} + b_2\right)z + \frac{2a_2^3}{27} - \frac{b_2a_2}{3} + c_2 = 0$$

Jetzt führen wir neue Bezeichner p_k und q_k ein und schreiben die Gleichung neu.

$$z^3 + p_kz + q_k = 0$$

Wir haben die reduzierte kubische Gleichung erhalten und lösen diese mit Hilfe der Cardanoformeln. Im Programm wird die Variable ε durch den Variablenbezeichner *eps* realisiert. Man zerlegt die Lösung in zwei Summanden u und v. Somit erhält man drei Lösungen:

$$z_1 = u + v$$

$$z_2 = \varepsilon u + \varepsilon^2 v$$

$$z_3 = \varepsilon^2 u + \varepsilon v$$

$$\varepsilon = -\frac{1}{2} + j\frac{\sqrt{3}}{2}$$

$$\varepsilon^2 = -\frac{1}{2} - j\frac{\sqrt{3}}{2}$$

In der Cardanoformel bedeutet ε die dritte Einheitswurzel beziehungsweise ε^2 die nächste Einheitswurzel der Gleichung dritten Grades. Unter Einheitswurzeln versteht man die Lösungen der Gleichung $z^3 - 1 = 0$. Diese sind 1, ε und ε^2.

Jede Kreisgleichung hat die Lösung 1. Daher kann man durch $x - 1$ ohne Rest dividieren.

$$z^3 - 1 = 0$$

$$(z-1)(z^2 + z + 1) = 0$$

$$z^3 + z^2 + z - z^2 - z - 1 = 0$$

$$z^3 - 1 = 0$$

Nachdem die erste Lösung 1 abgespalten ist, unterucht man den zweiten Faktor:

$$z^2 + z + 1 = 0$$

$$z_{1,2} = -\frac{1}{2} \pm \sqrt{\frac{1}{4} - 1}$$

$$z_{1,2} = -\frac{1}{2} \pm \sqrt{-\frac{3}{4}}$$

$$z_{1,2} = -\frac{1}{2} \pm \frac{1}{2}\sqrt{-3}$$

$$z_{1,2} = -\frac{1}{2} \pm \frac{1}{2}j\sqrt{3}$$

$$z_{1,2} = \frac{1}{2}\left(-1 \pm j\sqrt{3}\right)$$

$\varepsilon = z_1$

$$\varepsilon = \frac{1}{2}\left(-1 + j\sqrt{3}\right)$$

$$\varepsilon^2 = \left[\frac{1}{2}\left(-1 + j\sqrt{3}\right)\right]^2$$

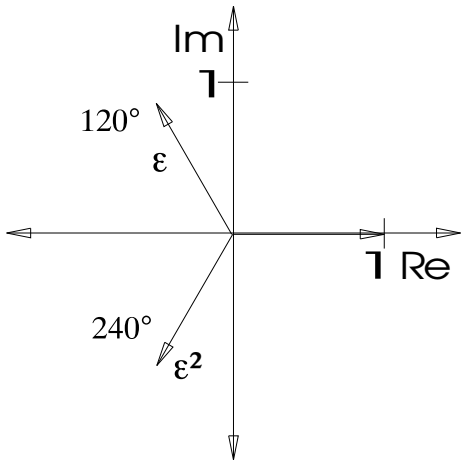
$$\varepsilon^2 = \frac{1}{4}\left(1 - 2j\sqrt{3} - 3\right)$$

$$\varepsilon^2 = \frac{1}{4}\left(-2 - 2j\sqrt{3}\right)$$

$\varepsilon^2 = z_2$

$$\varepsilon^2 = \frac{1}{2}\left(-1 - j\sqrt{3}\right)$$

ε^2 ist gleich der zweiten Lösung der quadratischen Gleichung.



$z^3 = 1$ Lösungen: 1, ε ,ε^2
komplexe Zahl: $a = r \cdot e^{jk}$

$z = \sqrt[3]{a}$

$z_1 = r^{\frac{1}{3}} e^{j\frac{\alpha}{3}}$ $k = 0$

$z_2 = r^{\frac{1}{3}} e^{j\frac{\alpha}{3}} + j\frac{2\pi}{3}$ $k = 1$

$z_3 = r^{\frac{1}{3}} e^{j\frac{\alpha}{3}} + j^2\frac{2\pi}{3}$ $k = 2$

k ist ein Faktor für 2π

Allgemein würde das bedeuten, daß die Lösung der Gleichung dritten Grades $u+v$ wäre. Da aber u und v jeweils Werte von dritten Wurzeln sind, ergeben sich drei Werte für u und drei Werte für v. Das bedeutet neun Kombinationsmöglichkeiten. Das bedeutet ein Einsetzen in die Formel $u+v$ ergibt eine Funktion, die neunwertig ist. Für die reduzierte Gleichung existiert jedoch

die Nebenbedingung, daß die Summe der Lösungen Null sein muß.
Die Gleichung dritten Grades in z kann nur drei Lösungen haben, obwohl die Cardanoformel mehrwertig ist. Das Ausscheiden der falschen Kombinationen bereitete den schwierigsten Teil der Arbeit und konnte nur durch die Rückrechnung als Kontrolle bewältigt werden. Addiert man die folgenden Ausdrücke und faßt sie spaltenweise zusammen,

$$\begin{aligned} u + v \\ \varepsilon u + \varepsilon^2 v \\ \varepsilon^2 u + \varepsilon v \end{aligned}$$

erhält man: $u(1 + \varepsilon + \varepsilon^2)$.
Der Wert in der Klammer ist Null, da das die Definition der dritten Einheitswurzel ist. Da die Summe der Lösungen der reduzierten Gleichung immer Null sein muß, kann auf diese Weise die Mehrdeutigkeit der Wurzel beseitigt werden.
In der Cardanoformel ergibt sich aber noch eine weitere Komplikation. Der innere Ausdruck mit der Quadratwurzel ist zweiwertig. Das ergibt für u mit der Dreiwertigkeit der dritten Wurzel 3 mal 2 = 6 Werte. Das gleiche gilt auch für v. Der Summenausdruck $u + v$ ist daher 6 mal 6 = 36 wertig. Das ist ohne Berücksichtigung der Nebenbedingungen. Es gelten aber die Nebenbedingungen:

$$\begin{aligned} p_k &= -3uv \\ q_k &= -u^3 - v^3 \end{aligned}$$

Wir formen um:

$$\begin{aligned} -\frac{p_k}{3} &= uv \\ -q_k &= u^3 + v^3 \end{aligned}$$

$$u^3 + v^3 = -\frac{q_k}{2} + \sqrt{\frac{q_k^2}{4} + \frac{p_k^3}{27}} - \frac{q_k}{2} - \sqrt{\frac{q_k^2}{4} + \frac{p_k^3}{27}}$$

Wenn der Ausdruck q ergeben soll, müssen die Wurzeln ein verschiedenes Vorzeichen aufweisen, damit sie sich aufheben. u und v sind vertauschbar, da sie nur in symmetrischen Funktionen vorkommen. Was ich daher mit u und v bezeichne ist egal.

Außerdem muß

$$u^3 \cdot v^3 = -\frac{p_k^3}{27}$$

ergeben.

$$\left(-\frac{q_k}{2} + \sqrt{\frac{q_k^2}{4} + \frac{p^3}{27}}\right)\left(-\frac{q_k}{2} - \sqrt{\frac{q_k^2}{4} + \frac{p^3}{27}}\right) = \frac{p_k^3}{27}$$

Wir multiplizieren nach $(a+b)(a-b) = (a^2 - b^2)$ aus:

$$\frac{q^2}{4} - \frac{q^2}{4} - \frac{p^3}{27} = \frac{p_k^3}{27}$$

Nach der Bereinigung der Vorzeichen der Quadratwurzeln lautet der Ausdruck:

$$z = \sqrt[3]{-\frac{q_k}{2} + \sqrt{\frac{q_k^2}{4} + \frac{p_k^3}{27}}} + \sqrt[3]{-\frac{q_k}{2} - \sqrt{\frac{q_k^2}{4} + \frac{p_k^3}{27}}}$$

Dieser Ausdruck ist immer noch neunwertig; es darf aber nur drei Lösungen geben.
Die neun Möglichkeiten lauten:

$$\begin{aligned}
z_1 &= u + v \\
z_2 &= \varepsilon u + v \\
z_3 &= \varepsilon^2 u + v \\
z_4 &= u + \varepsilon v \\
z_5 &= u + \varepsilon^2 v \\
z_6 &= \varepsilon u + \varepsilon v \\
z_7 &= \varepsilon^2 + \varepsilon^2 v \\
z_8 &= \varepsilon u + \varepsilon^2 v \\
z_9 &= \varepsilon^2 u + \varepsilon v
\end{aligned}$$

Wir erinnern uns, daß der Koeffizient des quadratischen Glieds in der reduzierten Gleichung Null sein muß.

$$\begin{aligned}
-(z_1 + z_2 + z_3) &= 0 \\
z_1 + z_2 + z_3 &= 0
\end{aligned}$$

Ich muß mir von den neun Lösungen, diejenigen heraussuchen, die diese Gleichung erfüllen. Wir setzen für die Lösungen z_1, z_8 und z_9 ein:

$$\begin{aligned}
z_1 + z_8 + z_9 &= 0 \\
u + v + \varepsilon u + \varepsilon^2 v + \varepsilon^2 u + \varepsilon v &= 0
\end{aligned}$$

Wir ordnen nach u und v:

$$u + \varepsilon u + \varepsilon^2 u + v + \varepsilon v + \varepsilon^2 v = 0$$
$$u(1 + \varepsilon + \varepsilon^2) + v(1 + \varepsilon + \varepsilon^2) = 0$$

Wir setzen für ε ein:

$$\left[\frac{1}{2}\left(-1 - j\sqrt{3}\right) + \frac{1}{2}\left(-1 + j\sqrt{3}\right) + 1\right] + v\left[\frac{1}{2}\left(-1 - j\sqrt{3}\right) + \frac{1}{2}\left(-1 + j\sqrt{3}\right) + 1\right] = 0$$

Die Nebenbedingung ist für diese drei Lösungen erfüllt. Die anderen sechs werden ausgeschieden. Wir setzen daher ε für die Lösungen z_1, z_8 und z_9 ein.

$$\begin{aligned}
z_1 &= u + v \\
z_2 &= \varepsilon u + \varepsilon^2 v \\
&= \frac{1}{2}\left(-1 + j\sqrt{3}\right) u + \frac{1}{2}\left(-1 - j\sqrt{3}\right) v \\
&= -\frac{1}{2}u + j\frac{\sqrt{3}u}{2} - \frac{1}{2}v - \frac{1}{2}j\sqrt{3}v \\
&= -\frac{u}{2} - \frac{v}{2} + j\frac{u\sqrt{3}}{2} - j\frac{v\sqrt{3}}{2} \\
z_2 &= -\left(u + v\right)\frac{1}{2} + j\left(u - v\right)\frac{\sqrt{3}}{2} \\
z_3 &= \varepsilon^2 u + \varepsilon v \\
&= \frac{1}{2}\left(-1 - j\sqrt{3}\right) u + \frac{1}{2}\left(-1 + j\sqrt{3}\right) v \\
&= -\frac{u}{2} - j\frac{u\sqrt{3}}{2} - \frac{v}{2} + j\frac{v\sqrt{3}}{2} \\
&= -\frac{u}{2} - \frac{v}{2} - j\frac{u\sqrt{3}}{2} + j\frac{v\sqrt{3}}{2} \\
z_3 &= -\left(u + v\right)\frac{1}{2} - j\left(u - v\right)\frac{\sqrt{3}}{2}
\end{aligned}$$

Mit Hilfe dieser Nebenbedingungen reduziert man die Kombinationsmöglichkeiten in der Cardanoformel. Im Programm bestimmt man die Diskriminante, also den Ausdruck unter der Quadratwurzel.

$$disk = \frac{q_k^2}{4} + \frac{p_k^3}{27}$$

Die Diskriminante [1] *unterscheidet drei Fälle:*

[1] vgl. Schülerduden Die Mathematik I Seite 150 ff.

- Fall 1: Das Vorzeichen ist positiv. 1 reelle und 2 konj. komplexe Lösungen

Wir setzen in die allgemeinen Lösungsformeln ein:

$$\begin{aligned} z_1 &= u + v \\ z_2 &= \left(-\frac{1}{2} + j\frac{\sqrt{3}}{2}\right) u + \left(-\frac{1}{2} - j\frac{\sqrt{3}}{2}\right) v \end{aligned}$$

Wir fassen zusammen und erhalten:

$$z_2 = \frac{-u - v}{2} + j\frac{u - v}{2}\sqrt{3}$$

z_3 ist konjugiert komplex zu z_2. Da nur reelle Koeffizienten erlaubt sind, treten komplexe Lösungen nur paarweise auf.

- Fall 2: Die Diskriminante ist Null. 1 reelle und 1 reelle Doppellösung

Wir setzen in die allgemeinen Lösungsformeln ein:
Hier muß u gleich v sein.

$$\begin{aligned} z_1 &= u + v = 2u \\ z_2 &= \varepsilon u + \varepsilon^2 v = \varepsilon u + \varepsilon^2 u = u(\varepsilon + \varepsilon^2) \\ &= u\left(-\frac{1}{2} + j\frac{\sqrt{3}}{2} - \frac{1}{2} - j\frac{\sqrt{3}}{2}\right) \\ &= u(-1) = -u \\ z_3 &= \varepsilon^2 u + \varepsilon v = \varepsilon^2 u + \varepsilon u = u(\varepsilon^2 + \varepsilon) \end{aligned}$$

Daher ist z_3 gleich z_2.
Wir fassen nochmals zusammen:

$$\begin{aligned} z_1 &= 2u \\ z_2 &= z_3 = -u \end{aligned}$$

- Fall 3: Das Vorzeichen ist negativ.

Es handelt sich um den *casus irreducibilis*. 3 reelle Lösungen, die in komplexer Form erscheinen. Die Bezeichnung ist rein historisch. Man kann in der komplexen Form nicht erkennen, daß es sich um reelle Werte handelt. Mit Hilfe von Winkelfunktionen kann man die Lösungen in reeller Form darstellen.
Die Diskriminante ist kleiner Null.

$$\frac{q_k^2}{4} + \frac{p_k^3}{27} < 0$$

Das Quadrat ist immer größer Null, daher muß $p_k < 0$ sein.
Man gewinnt die in komplexer Form gegebenen reellen Lösungen mit Hilfe goniometrischer Formeln. Um den in der Cardanoformel $\frac{\varphi}{3}$ kommenden Ausdruck zu berechnen, müssen wir zuerst einen Ausdruck für $\cos(3\varphi)$ finden. Wir setzen das allgemeine Additionstheorem für die Cosinusfunktion an:

$$\cos(\alpha + \beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$$

Wir setzen für $\alpha = 2\varphi$ und für $\beta = \varphi$.

$$\begin{aligned}
cos(2\varphi + \varphi) &= \underbrace{\cos 2\varphi}\cos\varphi - \sin 2\varphi\sin\varphi \\
\cos(3\varphi) &= (\cos^2\varphi - \sin^2\varphi)\cos\varphi - 2\sin\varphi\cos\varphi\sin\varphi \\
\cos(3\varphi) &= \cos^3\varphi - \sin^2\varphi\cos\varphi - 2\sin^2\varphi\cos\varphi \\
\cos(3\varphi) &= \cos^3\varphi - 3\underbrace{\sin^2\varphi}\cos\varphi \\
\cos(3\varphi) &= \cos^3\varphi - 3(1 - \cos^2\varphi)\cos\varphi \\
\cos(3\varphi) &= \cos^3\varphi - 3\cos\varphi + 3\cos^3\varphi \\
\cos(3\varphi) &= 4\cos^3\varphi - 3\cos\varphi
\end{aligned}$$

Wir stellen um

$$\begin{aligned}
4\cos^3\varphi - 3\cos\varphi - \cos(3\varphi) &= 0 \\
\cos^3\varphi - \frac{3}{4}\cos\varphi - \frac{1}{4}\cos(3\varphi) &= 0 \\
r^3\cos^3\varphi - \frac{3}{4}r^3\cos\varphi - \frac{1}{4}r^3\cos(3\varphi) &= 0 \\
(r\cos\varphi)^3 - \frac{3}{4}r^2 r\cos\varphi - \frac{1}{4}r^3\cos(3\varphi) &= 0
\end{aligned}$$

Wir führen die neue Unbekante z ein.

$$z = r\cos(\varphi)$$

Diese Bezeichnung wird im Programm verwendet.

$$z^3 - \frac{3}{4}r^2 z - \frac{1}{4}r^3 \cos(3\varphi) = 0$$
$$z^3 + p_k z + q_k = 0$$

Wir identifizieren die beiden Gleichungen und erhalten
(I)

$$p_k = -\frac{3}{4}r^2$$

(II)

$$q_k = -\frac{1}{4}r^3 \cos(3\varphi)$$

Wir isolieren die erste Gleichung nach r:

$$r^2 = -\frac{4p_k}{3}$$

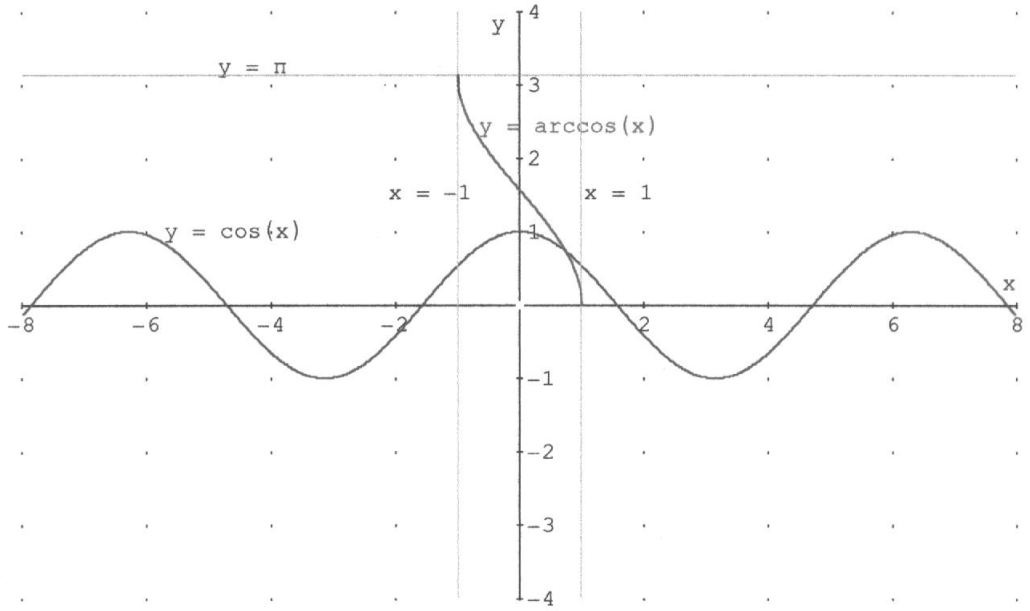
Weil durch die weiter oben erwähnte Nebenbedingung p_k kleiner Null sein muß, ist der Gesamtausdruck immer positiv. Wir können daher die Wurzel ziehen.

$$r = 2\sqrt{-\frac{p_k}{3}}$$

Wir isolieren die zweite Geichung nach $\cos(3\varphi)$ und setzen für r ein.

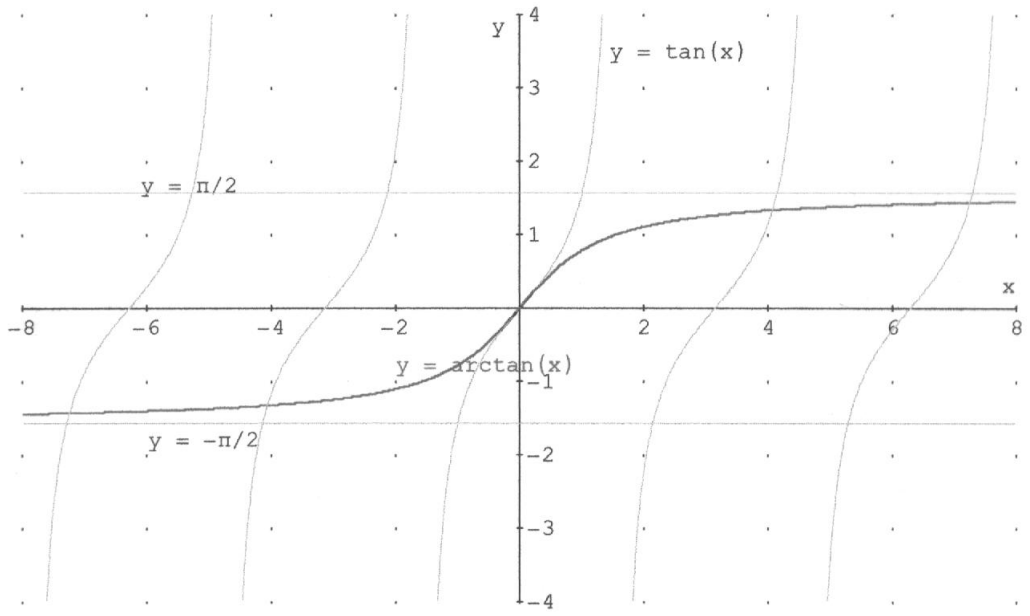
$$\cos(3\varphi) = -\frac{4q_k}{r^3}$$
$$\cos(3\varphi) = -\frac{4q_k}{8\left(\sqrt{-\frac{p_k}{3}}\right)^3}$$
$$\cos(3\varphi) = -\frac{q_k}{2\sqrt{-\frac{p_k^3}{27}}}$$

Damit fällt die Variable r wieder weg, die im Programm ja nicht vorkommt. Damit haben wir eine Gleichung für den Cosinuswert von 3φ, der nur aus p_k und q_k berechnet wird.
Um den Winkel zu erhalten, benutzen wir die inverse Funktion arccos. Sie kann geometrisch als Spiegelung an der Geraden $y = x$ gedeutet werden.



Der arccos ist die Umkehrung einer periodischen Funktion, daher unendlich vieldeutig, was die Einschränkung des Definitionsbereichs (Argument) von -1 bis +1 notwendig macht. Für Eingaben in diesem Bereich ist das Ergebnis immer reell. Eingaben außerhalb dieses Bereichs führen auf komplexe Werte. Beim *casus irreducibilis* setzen wir aber reelle Werte voraus.
Da im Compiler nur die arctan Funktion implementiert ist, wurde eine eigene arccos Funktion implementiert. In dieser Funktion wird weiters der Rückgabewert auf den Hauptwert also im Bereich von 0 bis π eingeschränkt.

$$y = \arccos(x) = \arctan\frac{\sqrt{1-x^2}}{x}$$



Aus dem Verlauf der arctan Funktion ist ersichtlich, daß ein negativer Eingabewert, ein negatives Ergebnis liefert. Um die Einschränkung auf den hauptwert der arccos Funktion (0 bis 2π) zu bewerkstelligen, muß ich für negative Werte π dazuaddieren. Dies wird programmtechnisch mit der Korrekturfaktormethode durchgeführt.

Korrekturfaktor:

$$1.0 - sgn(x) \cdot \frac{\pi}{2.0}$$

Einen Überblick gibt die Implementierung in C:

```
void cubic() {            // X^3 + A2*X^2 + B2*X +   C2 = 0
// Z = Loesung der reduzierten kubischen  Gleichung
//   Z^3 + Pk*Z + Qk = 0
//  Loesung der kubischen Gleichung   X = Z - A2/3
long double a2,b2,c2,pk,qk,disk,u,v,uh,vh,phi,z1re,z1im,z2re,z2im,z3re,z3im;
int fallkub,usign,vsign;

  a2 = cn/bn;
  b2 = dn/bn;
  c2 = en/bn;

  pk = -(a2*a2)/3.0 + b2;
```

```
                // Z^3 + pk*Z + qk = 0
  qk = 2.0*(a2*a2*a2)/27.0 - a2*b2/3.0 + c2;
              //  pk = -2.0; qk = 1.0;
  disk = qk*qk/4.0 + (pk*pk*pk)/27.0;
  fallkub = sgn(disk);
  switch(fallkub) {

// Fall1: 1 reelle,2 konj.komplexe Loesungen
     case 1:
uh =   -qk/2.0 + sqrtl(disk);  usign = sgn(uh);
vh =   -qk/2.0 - sqrtl(disk);  vsign = sgn(vh);
if (sgn(uh) == 0)
  u = 0.0;
else u = expl((1.0/3.0)*logl(fabsl(uh))) * usign;
if (sgn(vh) == 0)
   v = 0.0;
else v = expl((1.0/3.0)*logl(fabsl(vh))) * vsign;
z1re = u + v; z1im = 0.0;
z2re = -(u + v)/2.0; z2im = (u - v)/2.0 * sqrtl(3.0);
z3re = z2re; z3im = -z2im;
break;

    case 0:                          // 1 reelle, 1 reelle Doppelloesung
uh = -qk/2.0; usign = sgn(uh);
vh = -qk/2.0; vsign = sgn(vh);
if (sgn(uh) == 0)
  u = 0.0;
else u = expl((1.0/3.0)*logl(fabsl(uh))) * usign;
if (sgn(vh) == 0)
  v = 0.0;
else v = expl((1.0/3.0)*logl(fabsl(vh))) * vsign;
z1re = 2*u; z1im = 0.0;
z2re = -u; z2im = 0.0;
z3re = z2re; z3im = z2im;
break;

    case -1:                    // Fall3: casus irreducibilis
uh = -(pk*pk*pk)/27.0;
phi = arccos((-qk/2.0)/sqrtl(uh));
z1re =  2.0*sqrtl(-pk/3.0) * cos(phi/3.0);
z2re =  2.0*sqrtl(-pk/3.0) * cos(phi/3.0 + (2.0*M_PI/3.0));
```

```
z3re =  2.0*sqrtl(-pk/3.0) * cos(phi/3.0 - (2.0*M_PI/3.0));
z1im = 0.0; z2im = 0.0; z3im = 0.0;
       break;
  }
  x1re = z1re - a2/3.0; x1im = z1im;
  x2re = z2re - a2/3.0; x2im = z2im;
  x3re = z3re - a2/3.0; x3im = z3im;

/*
  printf("\n");
  printf("Koeffizienten der reduzierten Gleichung:\n");
  printf("z^3     +        pk *Z           +        qk       =  0\n");
  printf("               %10.5Lf               %10.5Lf\n",pk,qk);
  printf("\n");
  printf("Fallunterscheidung kubische Gleichung : %5d\n",fallkub);
  printf("\n");
  printf("Loesungen der reduzierten kubischen Gleichung:\n");
  printf("z1re =  %8.4Lf  z1im =  %8.4Lf\n",z1re,z1im);
  printf("z2re =  %8.4Lf  z2im =  %8.4Lf\n",z2re,z2im);
  printf("z3re =  %8.4Lf  z3im =  %8.4Lf\n",z3re,z3im);
  getch();
*/
}
```

Kapitel 8

Gleichung vierten Grades

Lösung der Gleichung vierten Grades:

Die Implementierung in C:

```
void quartic()
{
standardform();
  if ((sgn(a) == 0) && (sgn(c) == 0)) {
     // printf("biquad schon bei der Eingabe\n");
     // getch();
     biquad(b,d);
  }
  else {
    reduktion();
    if (sgn(c1r) == 0) {
      biquad(b1r,d1r);
    }
    else {
      resolvent();
      p6gleichung();
      kombination();
    }
  }
}
```

Folgende Softwareschnittstelle ist definiert:

Die fünf Koeffizienten a_n, b_n, c_n, d_n, und e_n der Eingangsgleichung dienen als Eingangsparameter und müssen innerhalb der gültigen Werte der in der Gleitkommabibliothek definierten Grenzen liegen.
Sämtliche Sonderfälle sind berücksichtigt. So dürfen zum Beispiel alle Koeffizienten Null sein, also keine gültige Gleichung vorliegen usw... Dies wurde absichtlich so definiert, weil in weiteren Ausbaustufen ja die Eingabe von anderen Programmen erfolgen soll, deren Ergebnis ja von vorneherein nicht bekannt ist. Zwei Kegelschnitte können sich z.B. überhaupt nicht oder nur in 2 Punkten schneiden oder identisch sein.
Das Unterprogramm „welchegleichung“entscheidet, um welche Gleichung es sich handelt und liefert in der Variablen „gl“den Wert 4 zurück, falls es sich um eine echte Gleichung vierten Grades handelt. Abgefragt wird, ob der Koeffizient a_n , also der Koeffizient der Unbekannten mit dem vierten Grad, ungleich Null ist. Dieses Unterprogramm trifft eine Vorentscheidung und das Programm verzweigt dann in Programmteile, in denen dann die lineare, quadratische, kubische und Gleichung vierten Grades getrennt behandelt werden.
Die allgemeine Gleichung vierten Grades [1] dient in folgender Form als Ausgangsbasis :

$$a_n x^4 + b_n x^3 + c_n x^2 + d_n x + e_n = 0$$

Alle Koeffizienten müssen reell sein, dürfen aber wie schon erwähnt Null sein.
Der erste Schritt ist die Normierung dieser Gleichung.
Normierung heißt, daß der Koeffizient des Gliedes mit dem höchsten Exponenten 1 gemacht wird. Im Unterprogramm „standardform“wird mit einer eigentlich unnotwendigen Sicherheitsabfrage die gesamte Gleichung durch a_n dividiert.
Nach der Division führen wir neue Bezeichner für die Koeffizienten ein:

$$a = \frac{b_n}{a_n}$$

$$b = \frac{c_n}{a_n}$$

$$c = \frac{d_n}{a_n}$$

$$d = \frac{e_n}{a_n}$$

Die so erhaltene Standardform lautet:

$$x^4 + ax^3 + bx^2 + cx + d = 0$$

[1] vgl. H. Deutsch, Kleine Enzyklopädie Mathematik Seite 10

Jetzt wird mit dem bereits beschriebenen Reduktionsverfahren eine reduzierte Gleichung gewonnen, so dass das Glied mit der dritten Potenz Null wird.
Wie erhält man nun die reduzierte Gleichung ?
Man ersetzt x in der Standardform durch den Audruck $z - \frac{a}{4}$.

$$\left(z - \frac{a}{4}\right)^4 + a\left(z - \frac{a}{4}\right)^3 + b\left(z - \frac{a}{4}\right)^2 + c\left(z - \frac{a}{4}\right) + d = 0$$

$$z^4 - az^3 + \frac{3}{8}a^2z^2 - \frac{a^3}{16}z + \frac{a^4}{256} + a\left(z^3 - \frac{3}{4}az^2 + \frac{3}{16}a^2z - \frac{a^3}{64}\right) + b\left(z^2 - \frac{a}{2}z + \frac{a^2}{16}\right) + c\left(z - \frac{a}{4}\right) + d = 0$$

Jetzt wird ausmultipliziert:

$$z^4 - az^3 + \frac{3}{8}a^2z^2 - \frac{a^3}{16}z + \frac{a^4}{256} + az^3 - \frac{3}{4}a^2z^2 + \frac{3}{16}a^3z - \frac{a^4}{64} + bz^2 - \frac{ab}{2}z + \frac{a^2b}{16} + cz - \frac{ac}{4} + d = 0$$

Jetzt werden die Glieder nach den Potenzen geordnet und zusammengefasst:

$$z^4 - az^3 + az^3 + \frac{3}{8}a^2z^2 - \frac{3}{4}a^2z^2 + bz^2 - \frac{a^3}{16}z + \frac{3}{16}a^3z - \frac{ab}{2}z + cz + \frac{a^4}{256} - \frac{a^4}{64} + \frac{a^2b}{16} - \frac{ac}{4} + d = 0$$

$$z^4 + \left(b - \frac{3}{8}a^2\right)z^2 + \left(c - \frac{ab}{2} + \frac{a^3}{8}\right)z + d - \frac{ac}{4} + \frac{a^2b}{16} - \frac{3}{256}a^4 = 0$$

Jetzt werden neue Bezeichner für die Koeffizienten eingeführt:

$$b_{1r} = b - \frac{3}{8}a^2$$

$$c_{1r} = c - \frac{ab}{2} + \frac{a^3}{8}$$

$$d_{1r} = d - \frac{ac}{4} + \frac{a^2 b}{16} - \frac{3}{256}a^4$$

Die reduzierte Gleichung lautet:

$$z^4 + b_{1r}z^2 + c_{1r}z + d_{1r} = 0$$

z ist nur eine Zwischenvariable. Später wird wieder x rückeingesetzt.

$$x = z - \frac{a}{4}$$

Nach dieser Substitution fällt, wie erwartet, das Glied mit z^3 weg. Die Eingangsparameter für das Unterprogramm „reduktion“sind die Koeffizienten der Standardform a, b, c und d. Ausgangsparameter sind die Koeffizienten b_{1r}, c_{1r} und d_{1r}. Die reduzierte Gleichung hat also keinen Koeffizienten a. Im Unterprogramm ist berücksichtigt, falls der Koeffizient a der Standardform schon Null ist, ist die Reduktion trivial, die Koeffizienten der Standardform bleiben erhalten. Die Summe der Unbekannten $x_1 + x_2 + x_3 + x_4$ bei der reduzierten Gleichung ist Null, da der Koeffizient des Gliedes x^3 Null ist.
Das Lösungsprinzip besteht darin, daß man eine allgemeine Gleichung vierten Grades in zwei Faktoren zweiten Grades zerlegt. Der erste Faktor ist $(x^2 + p_1 x + q_1)$.Der zweite Faktor ist $(x^2 + p_2 x + q_2)$ Für eine bereits reduzierte Gleichung ist die Zerlegung einfacher und zwar ist p_1 gleich p_2. Damit ist der erste Faktor $(x^2 + px + q_1)$ und der zweite Faktor $(x^2 - px + q_2)$.
Die reduzierte, in Faktoren zerlegte Gleichung lautet:

$$(z^2 + pz + q_1)(z^2 - pz + q_2) = 0$$

$$z^4 + q_1 z^2 - p_2 z^2 - pq_1 z + q_2 z^2 + pq_2 z + q_1 q_2 = 0$$

Da die Gleichung schon reduziert ist, ergeben sich nur drei Unbekannte, nämlich p, q_1 und q_2. Wäre die Gleichung nicht reduziert, würde die Zerlegung die Bestimmung von vier Unbekannten p_1, q_1, p_2 und q_2 verlangen. Dieses Gleichungssystem für die drei Unbekannten p, q_1 und q_2 wird nicht vollständig gelöst. Es wird nur nach p aufgelöst nicht aber nach q_1 und q_2. Der Koeffizientenvergleich mit der vorgegebenen Gleichung liefert drei Gleichungen für die drei Unbekannten p, q_1 und q_2. Die Unbekannten q_1 und q_2 werden eliminiert und man erhält eine Gleichung in der nur mehr die Unbekannte p vorkommt.

(I)

$$b_{1r} = q_1 + q_2 - p^2$$

(II)

$$c_{1r} = pq_2 - pq_1$$

(III)

$$d_{1r} = q_1 q_2$$

(II) wird umgestellt:

$$c_{1r} = p(q_2 - q_1)$$
$$\frac{c_{1r}}{p} = q_2 - q_1$$

(I) + (II):

$$b_{1r} + \frac{c_{1r}}{p} = -p^2 + 2q_2$$

(I) - (II):

$$b_{1r} - \frac{c_{1r}}{p} = -p^2 + 2q_1$$

Diese letzten zwei Gleichungen werden umgestellt und man erhält:

$$b_{1r} + \frac{c_{1r}}{p} + p^2 = 2q_2$$

$$b_{1r} - \frac{c_{1r}}{p} + p^2 = 2q_1$$

Diese zwei Gleichungen werden jetzt miteinander multipliziert:

$$\left(b_{1r} + \frac{c_{1r}}{p} + p^2\right)\left(b_{1r} - \frac{c_{1r}}{p} + p^2\right) = 4q_1q_2$$

Jetzt wird in diese Gleichung für q_1q_2 die Gleichung (III) $d_{1r} = q_1q_2$ eingesetzt.

$$\left(b_{1r} + \frac{c_{1r}}{p} + p^2\right)\left(b_{1r} - \frac{c_{1r}}{p} + p^2\right) = 4d_{1r}$$

Es wird ausmultipliziert:

$$b_{1r}^2 + \frac{c_{1r}}{p}b_{1r} + p^2b_{1r} - \frac{c_{1r}}{p}b_{1r} - \frac{c_{1r}^2}{p^2} - \frac{c_{1r}}{p}p^2 + b_{1r}p^2 + \frac{c_{1r}}{p}p^2 + p^4 = 4d_{1r}$$

Die einzelnen Glieder werden zusammengefasst:

$$b_{1r}^2 + p^2b_{1r} - \frac{c_{1r}^2}{p^2} + b_{1r}p^2 + p^4 = 4d_{1r}$$

Die ganze Gleichung wird jetzt mit p^2 multipliziert:

$$b_{1r}^2p^2 + b_{1r}p^4 - c_{1r}^2 + b_{1r}p^4 + p^6 = 4d_{1r}p^2$$

Es wird p^2 herausgehoben und nach den höchsten Potenzen von p geordnet:

$$p^6 + 2b_{1r}p^4 + (b_{1r}^2 - 4d_{1r})p^2 - c_{1r}^2 = 0$$

Der Begriff Resolvente bedeutet nur Hilfsgleichung niederen Grades zur Lösung einer Gleichung höheren Grades. Im Unterprogramm „resolvent“werden neue Koeffizienten eingeführt:

$$\begin{aligned} a_2 &= 2b_{1r} \\ b_2 &= b_{1r}^2 - 4d_{1r} \\ c_2 &= c_{1r}^2 \end{aligned}$$

Die Resolventengleichung lautet daher:

$$p^6 + a_2p^4 + b_2p^2 + c_2 = 0$$

Oder wenn wir für $p^2 = t$ setzen:

$$t^3 + a_2t^2 + b_2t + c_2 = 0$$

Es handelt sich nicht um eine allgemeine Gleichung sechsten Grades in p. Wir haben vielmehr eine spezielle Gleichung sechsten Grades erhalten, bei der

das fünfte, dritte und lineare Glied fehlt. Diese spezielle Gleichung sechsten Grades kann auf eine Gleichung dritten Grades in p^2 zurückgeführt werden. Für p^2 wird im Programm die Variable t verwendet. Man erhält somit eine allgemeine Gleichung dritten Grades in t. Diese Gleichung hat wieder reelle Koeffizienten; die Lösungen für t können natürlich auch komplex sein. Die Anzahl der komplexen Lösungen einer Gleichung mit reellen Koeffizienten ist immer geradzahlig. Um nun diese völlig allgemeine Gleichung dritten Grades in t zu lösen, müssen wir abermals auf das Reduktionsprinzip zurückgreifen. Wir führen die Substitution von $t = z - \frac{a_2}{3}$ durch.

$$t^3 + a_2 t^2 + b_2 t + c_2 = 0$$

$$\left(z - \frac{a_2}{3}\right)^3 + a_2 \left(z - \frac{a_2}{3}\right)^2 + b_2 \left(z - \frac{a_2}{3}\right) + c_2 = 0$$

Jetzt wird ausmultipliziert:

$$z^3 - \frac{3}{3} a_2 z^2 + \frac{3}{9} a_2^2 z - \frac{a_2^3}{27} + a_2 z^2 - \frac{2}{3} a_2^2 z + \frac{a_2^3}{9} + b_2 z - \frac{b_2 a_2}{3} + c_2 = 0$$

Jetzt werden die Glieder nach den Potenzen geordnet und zusammengefasst:

$$z^3 + \left(b_2 - \frac{a_2^2}{3}\right) z + \frac{2}{27} a_2^3 - \frac{a_2 b_2}{3} + c_2 = 0$$

Wir führen neue Bezeichner für die Koeffizienten ein:

$$p_k = b_2 - \frac{a_2^2}{3}$$

$$q_k = \frac{2}{27} a_2^3 - \frac{a_2 b_2}{3} + c_2$$

Und erhalten wiederum eine reduzierte kubische Gleichung

$$z^3 + p_k z + q_k = 0$$

Diese reduzierte kubische Gleichung wird mit Hilfe der *Cardanoformel* gelöst. Dieses Verfahren ist bereits weiter oben bei der kubischen Gleichung besprochen worden.

Die weitere Vorgangsweise kann folgendermassen zusammengefasst werden:

z ist nur eine Hilfsvariable. Sobald man z berechnet hat, kann man $t = z - \frac{a_2}{3}$ berechnen und schließlich p, die Quadratwurzel aus t. Hat man p, kann man q_1 und q_2, der in Faktoren zerlegten ursprünglichen Gleichung vierten Grades berechnen. Damit kann man die Lösungen der Gleichung vierten Grades durch ein sehr komplexes Auswahlverfahren durch Berücksichtigung aller Nebenbedingungen bestimmen.
Die Vorzeichendiskussion wird mit den drei Unterprogrammen „kombiplus“, „kombiminus“und „kombination“erledigt.

```
void kombiplus()
{
    x1re = 0.5 * (p1re + p2re + p3re) - a/4.0;
    x2re = 0.5 * (p1re - p2re - p3re) - a/4.0;
    x3re = 0.5 * (-p1re - p2re + p3re) - a/4.0;
    x4re = 0.5 * (-p1re + p2re - p3re) - a/4.0;

    x1im = 0.5 * (p1im + p2im + p3im);
    x2im = 0.5 * (p1im - p2im - p3im);
    x3im = 0.5 * (-p1im - p2im + p3im);
    x4im = 0.5 * (-p1im + p2im - p3im);
    // printf("Es wurde kombiplus ausgefuehrt\n");
}

void kombiminus()
{
    x1re = 0.5 * (-p1re - p2re - p3re) - a/4.0;
    x2re = 0.5 * (-p1re + p2re + p3re) - a/4.0;
    x3re = 0.5 * (p1re + p2re - p3re) - a/4.0;
    x4re = 0.5 * (p1re - p2re + p3re) - a/4.0;

    x1im = 0.5 * (-p1im - p2im - p3im);
    x2im = 0.5 * (-p1im + p2im + p3im);
    x3im = 0.5 * (p1im + p2im - p3im);
    x4im = 0.5 * (p1im - p2im + p3im);
    // printf("Es wurde kombiminus ausgefuehrt\n");
}
```

```
void kombination() {
  int fall;
  int vzt;
 vzt = sgn(t1re) + sgn(t2re) + sgn(t3re);
  if ((fallres == -1) || (fallres == 0)) {
    fall = vzt + sgn(c1r);
    switch(fall) {
      case  4: kombiminus();
       break;
      case  2: kombiplus();
       break;
      case  0: kombiplus();
       break;
      case -2: kombiminus();
    }
  }

  if (fallres == 1) {   // c1r kann nicht null sein
    if (sgn(c1r) == 1) kombiplus(); // c1r = 0 ist biquad2 Fall
    else kombiminus();
  }
}
```

Einen Sonderfall stellt die biquadratische Gleichung dar: Sie wird gesondert behandelt. Sie wird bereits bei der Eingabe abgefangen.

Die Implementierung in C:

```
void biquad(long double l, long double m) {
  int ent;
   long double disk,s,s1,s2,sre,sim,s1re,s1im,s2re,s2im;

  disk = l*l - 4.0*m;
  ent = sgn(disk);
  switch(ent)
     {
     case  1 :
   s1 = (-l + sqrtl(disk)) / 2.0;
   s2 = (-l - sqrtl(disk)) / 2.0;
```

```
   s = s1;
   x1re = 0.5*sqrtl(s*sgn(s)) * (1.0 + sgn(s));
   x1im = 0.5*sqrtl(s*sgn(s)) * (1.0 - sgn(s));
   x2re = -x1re;
   x2im = -x1im;
   s = s2;
   x3re = 0.5*sqrtl(s*sgn(s)) * (1.0 + sgn(s));
   x3im = 0.5*sqrtl(s*sgn(s)) * (1.0 - sgn(s));
   x4re = -x3re;
   x4im = -x3im;
   break;
case 0 :
   s1 = -l / 2.0;
   s2 = s1;
   s = s1;
   x1re = 0.5*sqrtl(s*sgn(s)) * (1.0 + sgn(s));
   x1im = 0.5*sqrtl(s*sgn(s)) * (1.0 - sgn(s));
   x2re = -x1re;
   x2im = -x1im;
   x3re = x1re;
   x3im = x1im;
   x4re = x2re;
   x4im = x2im;
   break;
case -1 :
   s1re = -l / 2.0;
   s1im = sqrtl(-disk) / 2.0;
   s2re = -l / 2.0;
   s2im = -sqrtl(-disk) / 2.0;
   sre = s1re;
   sim = s1im;
   // printf("s1re = %10.5Lf",s1re);
   printf("   s1im = %10.5Lf\n",s1im);
   x1re = sqrtl(0.5*(sre + sqrtl(sre*sre + sim*sim)));
   x1im = sgn(sim)*sqrtl(0.5*(-sre + sqrtl(sre*sre + sim*sim)));
   x2re = -x1re;
   x2im = -x1im;
   sre = s2re;
   sim = s2im;
   // printf("s2re = %10.5Lg",s2re);
   printf("   s2im = %Lg.5",s2im);
```

```
   x3re = -sqrtl(0.5*(sre + sqrtl(sre*sre + sim*sim)));
   x3im = -sgn(sim)*sqrtl(0.5*(-sre + sqrtl(sre*sre + sim*sim)));
   x4re = -x3re;
   x4im = -x3im;
   break;
  }
  if (sgn(c1r) == 0) {
    x1re = x1re - a/4.0;
    x2re = x2re - a/4.0;
    x3re = x3re - a/4.0;
    x4re = x4re - a/4.0;
    // printf("\nreduzierte Gleichung ist biquadratisch\n    ");
  }
// printf("ent  = %4d\n",ent);
}
```

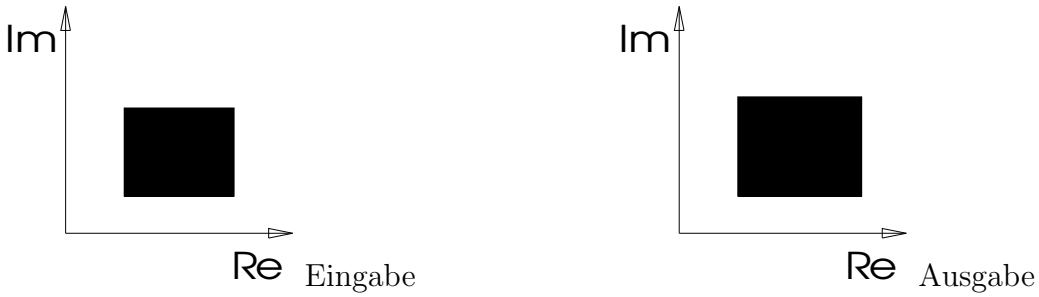
Damit sind alle Fälle der Gleichung vierten Grades behandelt.

Kapitel 9

Komplexe Wurzel

Obwohl dieses Unterprogramm nur drei Zeilen umfaßt, ist es ein wichtiges Kernstück des Programms. Das Wurzelzeichen bedeutet ja die Kreisteilungsgleichung, das ist ja die Einteilung des Kreises in gleiche Teile. Die Quadratwurzel bedeutet, daß zwei Lösungen, die Kubikwurzel bedeutet, daß drei Lösungen existieren. Die Reihenfolge der Lösungen ist hier nicht festgelegt. Man muß daher festlegen, welche Lösung des Wurzelausdrucks gemeint ist.
Eine Funktion im Bereich der komplexen Zahlen kann nicht mehr in einem Schaubild dargestellt werden. Für die Eingabevariable wird schon die ganze Gaußsche Zahlenebene verbraucht. Für den Funktionswert wird auch nochmals eine ganze Gaußsche Zahlenebene verbraucht. Würde man jetzt jeden Punkt der Ebene mittels der Funktionsvorschrift abbilden, so erhielte man zwei schwarze Flächen. Das entspricht der exakten Abbildung.

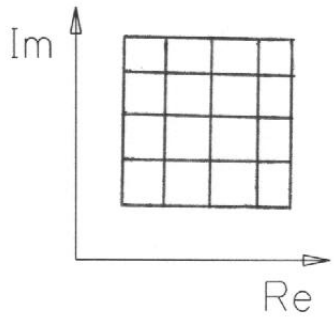
nicht parametrisierte Funktion $z = f(x)$



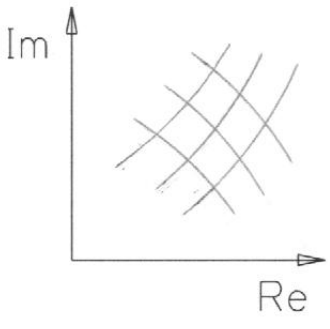
Man muß die Funktion parametrisieren, das heißt nur einzelne ausgewählte Punkte abbilden. Als einfachste Parametrisierung kann ein rechtwinkeliges Gitter über die Eingangswerte gelegt werden. Diese Werte können dann eindeutig abgebildet werden. Dazwischen muß dann interpoliert werden.
Das Geschilderte gilt aber nur, wenn ein eindeutiger Funktionswert zurückgeliefert wird. Bei Funktionen wie der Quadratwurzel, die immer zwei Werte

liefert liegen die Verhältnisse noch komplizierter. Die unabhängig veränderliche, also der Eingangswert nimmt wie vorhin besprochen eine ganze Gaussche Zahlenebene ein. Die abhängig veränderliche, also der Ausgabewert der Funktion stellt zwei Gaussche Zahlenebenen dar, die man sich übereinander liegend vorstellen muß und nach Riemann Blätter der Riemannschen Fläche genannt werden. Diese Blätter hängen zusammen und zwar über eine Linie vom Ursprung ausgehend entlang der positiven reellen Achse bis plus unendlich. Wenn ich nun im Eingabebereich den Ursprung gegen den Uhrzeigersinn umrunde, wechsle ich beim Überschreiten der positiven reellen Achse die Blätter. Zu beachten ist aber, daß bei einer Umrundung (= Drehung um 360 Grad) an jeder anderen Stelle der Gausschen Zahlenebene kein Blattwechsel vor- genommen wird.

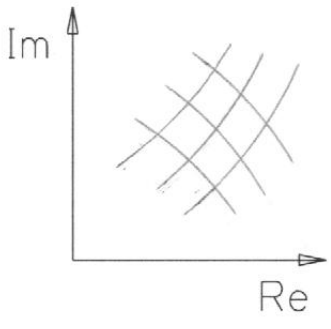
Eingabe $z = \sqrt{w}$



Ausgabe: komplexe Quadratwurzel



Blatt 1



Blatt 2

Bei der Kubikwurzel liegen die Verhältnisse noch komplizierter, es sind drei Blätter vorhanden. Das Quadratwurzelzeichen ist mehrdeutig und legt nicht fest, welcher der zwei Ergebniswerte auf den zwei Blättern an erster Stelle

gereiht wird. In dieser Arbeit wird festgelegt, daß die Quadratwurzelfunktion nur den Wert auf dem ersten Blatt der Riemannschen Fläche liefert. Dieser Wert auf dem ersten Blatt bedeutet aber nicht, daß der Realteil des Ergebnisses einen positiven Wert hat, oder nicht unbedingt.
Ob der Quadratwurzelwert auf dem ersten Blatt der Riemannschen Fläche liegt, wird über den Winkel definiert. Die Zahl aus der die Quadratwurzel gezogen wird heißt ja:

$$re^{j\varphi}$$

oder da sie auch einen Winkel $\varphi + 2\pi$ haben kann

$$re^{(j\varphi+2j\pi)}$$

oder auch

$$re^{(j\varphi+2jk\pi)}$$

K ist hier ganzzahlig. Diese Zahl hat unendlich viele Darstellungen. Die Quadratwurzel hat aber nur zwei Lösungen. Um nun den anderen Wert auf dem zweiten Blatt der Riemannschen Fläche zu erhalten, multipliziere ich den ersten Wert mit minus eins. Daher ist das bei der Quadratwurzel in der konkreten Implementierung eine blose Vorzeichendiskussion, während die Verhältnisse bei höheren Wurzeln wesentlich komplizierter liegen.
Wie gelangt man nun zu diesen drei Zeilen im Unterprogramm ? Es werden für alle Quadranten Einzelfälle aufgezeichnet. Anschließend werden die Grenzfälle besprochen, die auf den Achsen liegen. Danach wird der Algorithmus für das Ergebnis festgelegt. Man sieht, daß es für viele Fälle stimmt. Man sammelt jetzt alle Fälle, bei denen dieses Ergebnis falsch ist und muß einen Korrekturfaktor einführen, der auch diese Ergebnisse richtig stellt. Die Schwierigkeit besteht nun darin, daß der Korrekturfaktor in den anderen Fällen wirkungslos bleibt. Hierbei spielt die Funktion *sgn* eine wichtige Rolle. Sie bestimmt das Vorzeichen einer Zahl. Ihre Implementierung wird gesondert besprochen.

Die konkrete Implementierung in Pascal und C lautet:

a, b sind die Eingabewerte, u, v sind die Rückgabewerte des ersten Blattes der Riemannschen Fläche.

```
PROCEDURE Comsqrt2(A,B : DOUBLE;VAR U,V : DOUBLE);
VAR
Vz: DOUBLE;
BEGIN
```

```
  Vz := 1.0 + Sgn(B) - Sgn(B)*Sgn(B);          { Vorzeichenkorrekturfaktor }
  U := Vz*Sqrt(0.5*(A + Sqrt(A*A + B*B)));
  V := Sqrt(0.5*(-A + Sqrt(A*A + B*B)));
END;
```

```
void comsqrt2(long double a, long double b, long double *pu,long double *pv)
{
  long double vz;                                    // Vorzeichenkorrekturfaktor
  vz =  1.0 + sgn(b) - sgn(b)*sgn(b);
  *pu = vz*sqrtl(0.5*(a + sqrtl(a*a + b*b)));
  *pv = sqrtl(0.5*(-a + sqrtl(a*a + b*b)));
}
```

$$\sqrt{a+jb} = u + jv$$

quadrieren

$$a + jb = (u + jv)^2$$
$$a + jb = u^2 - v^2 + 2juv$$

Real und Imaginärteil gleichsetzen;
2 Gleichungen für die Unbekannten u und v

(I)

$$a = u^2 - v^2$$

(II)

$$b = 2uv$$
$$\downarrow$$
$$v = \frac{b}{2u}$$

In (I) einsetzen

$$a = u^2 - \frac{b^2}{4u^2}$$
$$4au^2 = 4u^4 - b^2$$
$$4u^4 - 4au^2 - b^2 = 0$$
$$u^4 - au^2 - \frac{b^2}{4} = 0$$

Standardform

$$u_{1,2}^2 = +\frac{a}{2} \pm \sqrt{\frac{a^2}{4} + \frac{b^2}{4}}$$

$$u_{1,2}^2 = +\frac{a}{2} \pm \frac{1}{2}\sqrt{a^2 + b^2}$$

$$u_{1,2}^2 = \frac{a \pm \sqrt{a^2 + b^2}}{2}$$

Annahme: u und v sind reell!

u^2 hat 2 Lösungen
u hat 4 Lösungen
wenn u reell sein soll, muß $u^2 > 0$ sein.
wenn u^2 positiv sein soll, darf man nur das Pluszeichen verwenden! Wurzelausdruck $> |a|$

$$u_{1,2}^2 = \frac{a + \sqrt{a^2 + b^2}}{2}$$

$$u_{1,2} = \pm\sqrt{\frac{a + \sqrt{a^2 + b^2}}{2}}$$

damit für u^2 nur 1 Wert
für u 2 Werte

jetzt aus (I) und (II) das gleiche für v

(I)

$$a = u^2 - v^2$$

(II)

$$b = 2uv$$

$$\downarrow$$

$$u = \frac{b}{2v}$$

in (I) einsetzen:

$$a = \frac{b^2}{4v^2} - v^2$$

$$4av^2 = b^2 - 4v^4$$

$$v_{1,2}^2 = -\frac{a}{2} \pm \sqrt{\frac{a^2 + b^2}{4}}$$

$$v_{1,2}^2 = \frac{-a \pm \sqrt{a^2 + b^2}}{2}$$

v^2 soll größer Null sein!
der gesamte Wurzelausdruck ist größer als $|a|$! Daher negatives Vorzeichen nicht sinnvoll.
1 Wert für v^2
2 Werte für v

$$v_{1,2}^2 = \frac{-a + \sqrt{a^2 + b^2}}{2}$$

$$v_{1,2} = \pm\sqrt{\frac{-a + \sqrt{a^2 + b^2}}{2}}$$

$$u_{1,2} = \pm\sqrt{\frac{a + \sqrt{a^2 + b^2}}{2}}$$

alle Vorzeichenkombinationen liefern 4 Möglichkeiten :

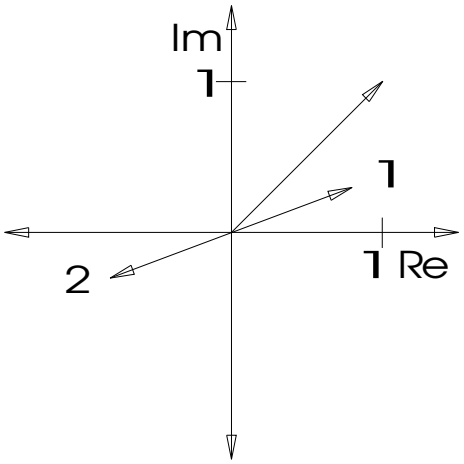
$++ \qquad +- \qquad -+ \qquad --$

das würde bedeuten die Quadratwurzel hätte 4 Werte

Einschränkungen suchen:
Die Quadratwurzel ist der Wert, der den halben Winkel hat und die Wurzel aus dem Betrag (= Länge) hat. Der Winkel wird dabei wie üblich im Gegenuhrzeigersinn vereinbart und zwar im Bereich von 0 bis 360 $^\circ$.

9.0.1 Beispiele

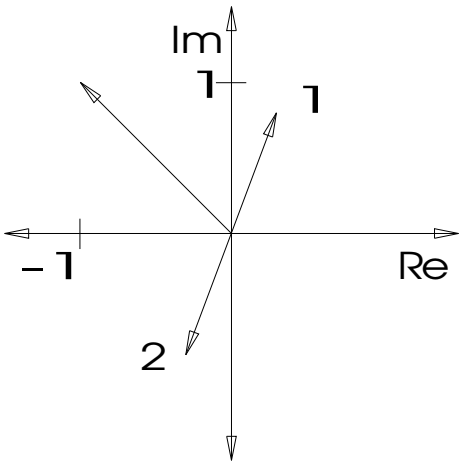
ohne Randproblematik



$$u = \sqrt{\frac{1+\sqrt{1+1}}{2}}$$

$$v = \sqrt{\frac{-1+\sqrt{1+1}}{2}}$$

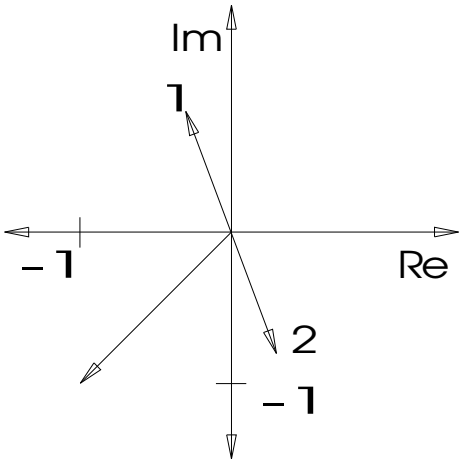
Bei diesem Beispiel wurde $u+$ und $v+$ gewählt.
Das gilt für den 1. Quadranten ohne Randproblematik.



$$u = \sqrt{\frac{-1+\sqrt{2}}{2}}$$

$$v = \sqrt{\frac{1+\sqrt{2}}{2}}$$

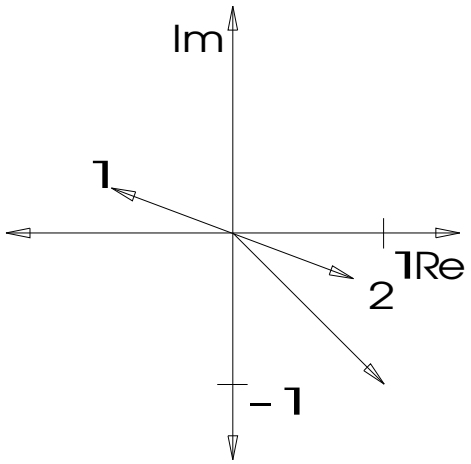
Ich wähle für $u+$ und für $v+$.
Man sieht, dass der Betrag von v grösser als der Betrag von u ist. Das gilt für den 2. Quadranten ohne Randproblematik.



$$u = -\sqrt{\frac{-1+\sqrt{2}}{2}}$$

$$v = \sqrt{\frac{1+\sqrt{2}}{2}}$$

Ich wähle für $u-$ und für $v+$.
Das gilt im 3. Quadranten.



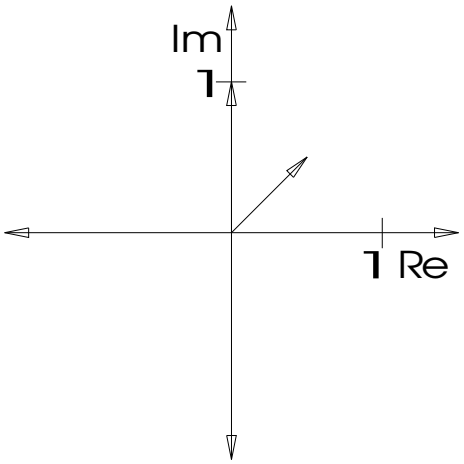
$$u = -\sqrt{\frac{1+\sqrt{2}}{2}}$$

$$v = \sqrt{\frac{1+\sqrt{2}}{2}}$$

Für den 4. Quadranten ohne Randproblematik.

mit Randproblematik

$$a = 0$$
$$b = 1$$

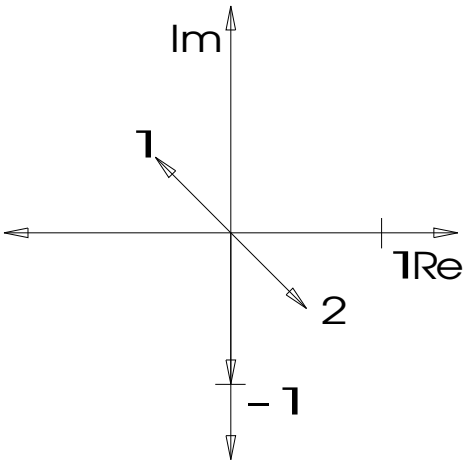


$$u = \sqrt{\frac{0 + \sqrt{0+1}}{2}} \rightarrow \frac{\sqrt{2}}{2}$$

$$v = \sqrt{\frac{-0 + \sqrt{0+1}}{2}} \rightarrow \frac{\sqrt{2}}{2}$$

Für $u+$ und für $v+$

$$a = 0$$
$$b = -1$$

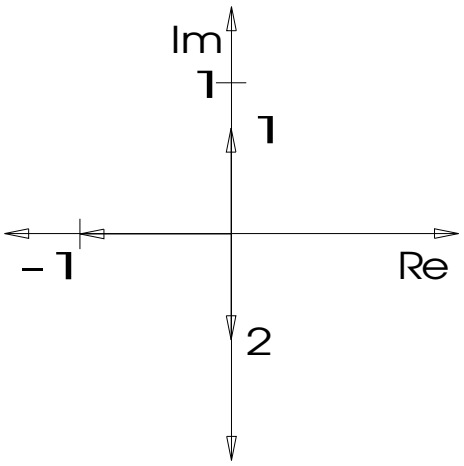


$$u = -\sqrt{\frac{0 + \sqrt{0+1}}{2}} \to -\frac{\sqrt{2}}{2}$$

$$v = \sqrt{\frac{-0 + \sqrt{0+1}}{2}} \to \frac{\sqrt{2}}{2}$$

Ich wähle $u-$ und $v+$

$$a = -1$$
$$b = 0$$

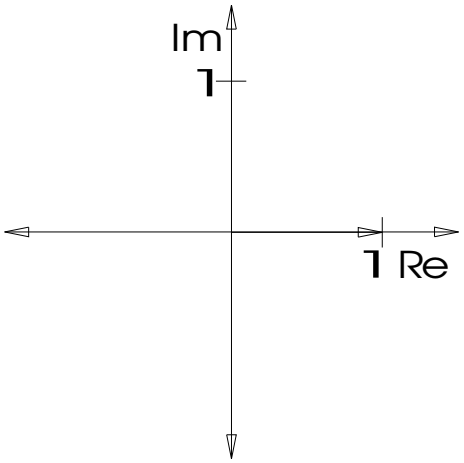


$$u = \sqrt{\frac{-1 + \sqrt{1+0}}{2}} = 0$$

$$v = \sqrt{\frac{1 + \sqrt{1+0}}{2}} = 1$$

Ich wähle u **beliebig** weil es Null ist.
Ich wähle $v+$

$$a = 1$$
$$b = 0$$



$$u = \sqrt{\frac{1 + \sqrt{1+0}}{2}} = 1$$

$$v = \sqrt{\frac{-1 + \sqrt{1+0}}{2}} = 0$$

Ich wähle $u+$ und v **beliebig**.

Aus diesen Fällen muss der *Vorzeichenkorrekturfaktor* bestimmt werden:

v kann plus gesetzt werden:

$$v = +\sqrt{\frac{-a + \sqrt{a^2+b^2}}{2}}$$

grundsätzlich nur erste Lösung!

Vorzeichen für u:
im I, II Quadranten +
im III, IV Quadranten −
daher folgt das Vorzeichen von u dem Vorzeichen von b

$$u = sgn(b)\sqrt{\frac{a + \sqrt{a^2+b^2}}{2}}$$

Randfälle:

$$richtig \begin{cases} a = 0 \\ b = 1 \end{cases}$$

$$richtig \begin{cases} a = 0 \\ b = -1 \end{cases}$$

$$richtig \begin{cases} a = -1 \\ b = 0 \end{cases}$$

$$falsch \begin{cases} a = 1 \\ b = 0 \end{cases}$$

1. Korrektur soll nur dann wirksam sein, wenn $b = 0$.
2. Wenn $b = 1$ oder $b = -1$, muss die Korrektur verschwinden.
Ein Polynom, das für nicht - Null Vorzeichen verschwindet ist $[1 - sgn^2(x)]$.

für $x > 0$ $\quad 1 - 1 = 0$
für $x < 0$ $\quad 1 - 1 = 0$
für $x = 0$ $\quad 1 - 0 = 1$

$$u_n = \sqrt{\frac{a + \sqrt{a^2 + b^2}}{2}}$$

$$u = u_n \left[sgn(b)\right] + u_n[1 - sgn^2(b)\big]$$

$$u = u_n \left[sgn(b) + (1 - sgn^2(b)\right] = u_n \left[1 + sgn(b) - sgn^2(b)\right]$$

Kapitel 10

Verfassen der Arbeit mit LaTeX

Nicht ohne Grund sei auch dem Satzsystem LaTeX ein Kapitel gewidmet, ohne das das Erstellen der Arbeit in dieser Form erst gar nicht möglich gewesen wäre.

10.1 Was ist LaTeX?

In den 70er Jahren wurde durch den amerikanischen Universitätsprofessor Danald E. Knuth das Programm TeX entwickelt. Es dient zum typographisch und ästethisch korrekten Setzen von setzen von Texten verschiedenster Art. Es ist überdies Freeware und kann z.B. kostenlos vom Internet heruntergeladen werden. Es bedarf daher einer gewissen Einarbeitungszeit und guter typographischer Kenntnisse, um zu ansprechenden Dokumenten zu gelangen. Um die Arbeit mit dem System zu vereinfachen, wurde von Leslie Lamport, aufbauend auf den Programmiermöglichkeiten von TeX eine Makrosammlung entwickelt, die das Arbeiten mit dem System etwas vereinfacht.
LaTeX ist ein Textsatzsystem für Profis, das sich vor allem im technisch-wissenschaftlichen Bereich etabliert hat. Der Grund dafür ist u.a. sein ausgezeichneter *mathematischer Formelsatz*[1].
Die Arbeitsweise von LaTeX unterscheidet sich grundsätzlich von Desktop-Publishing- oder Textverarbeitungsprogrammen.
LaTeX unterstützt daher auch nicht das sogenannte WYSIWYG („What You See Is What You Get“), welches in der Praxis ohnedies kaum oder nur recht schlecht funktioniert. Daher stand von Anfang an außer Frage, ein anderes System als LaTeX zu verwenden. Wer sich daran gewöhnt hat, Textattribute, Absatzformatierungen und dergleichen per Tastatur- Shortcut zu verändern, muß nun umdenken: Das Satzsystem arbeitet mit Befehls-

[1] vgl. T. Machert, Wissenschaftliches Publizieren mit LaTeX2e, Seite 101 ff.

sequenzen, die komplexe Layout- Funktionen übernehmen. Voreingestellte Layoutbefehle werden durch einen Backslash eingeleitet. Absatzweise optimiert LaTeX die Buchstaben- und Wortabstände und erfüllt dabei strenge Anforderungen an Unterschneidungen und Ligaturen. Vom Anwender wird ein strenger und sehr logischer Aufbau seiner TeX Datei erzwungen. Dadurch werden Formatierungsfehler vermieden. Obwohl LaTeX in Einarbeitungszeit etwas gewöhnungsbedürftig zu bedienen war, macht die Arbeit damit jedoch bereits nach kurzer Zeit sehr viel Spaß.

10.2 Vorgehensweise

Zunächst schreibt man seine Texte mit einem beliebigen ASCII-Editor. Unter Linux kann man z.B. den XEmacs verwenden. Ich habe für diese Arbeit jedoch die WinShell für Win95/98 von Ingo de Boer verwendet, die es erlaubt, sin Dokument auf sehr komfortable Weise zu setzen. Natürlich wird einem der korrekte Syntax dadurch nicht abgenommen. Man übersetzt dann, den auf diese Weise erstellten Quelltext mit dem TeX Compiler in eine *geäteunabhängige* Datei, die das Textdokument mit vollständigem Layout enthält. Es spielt also keine Rolle, ob diese Datei auf einem Home-Computer oder Großrechner erzeugt wurde: Das TeX Ausgabeformat ist *plattformübergreifend*. Es sollte daher auch keine Schwierigkeit sein, meine Arbeit mit einem TeX Compiler unter Linux zu übersetzen, sie etwa unter XEmacs zu bearbeiten und in xdvi am Bildschirm zu betachten. (TeX unter Linux ist ohnehin das höchste aller Gefühle) Die Qualität der geräteunabhängigen Datei ist also nur von der Auflösung des Ausgabemediums abhängig. (Tintenstrahldrucker, Laserdrucker). Will man auch die in das TeX Dokument eigebundenen eps-Graphiken (Encapsulated Postscript) am Bildschirm betrachten, muß man die geräteunabhänige dvi- Datei in Postscript [2] konvertieren.
Für meine Fachbereichsarbeit verwendete ich die MikTeX Distribution, welche sich für den Einsatz unter Win95/98/NT eignet. Dem Anwender stehen verschiedene Makropakete zur Verfügung, aus denen der TeX Compiler Informationen zu Typographie und Dokumentengliederung ausliest. Am weitesten verbreitet ist LaTeX2e. Wer mit LaTeX Dokumente schreibt, kann das Textlayout in höchstem Maße variieren. LaTeX legt unter Verwendung der entsprechenden Befehle Inhaltsverzeichnisse an und verwaltet die Fußnoten.

[2] vgl. M. Goossens, The LaTeX Graphics Companion, Seite 413 ff.

10.3 Umlaute

TeX benutzt eigene, nichtskalierbare Pixel-Zeichensätze, die der Zeichengenerator Metafont in der gewünschten Auflösung erzeugt. Daher stehen keine True Type Fonts [3] für TeX Dokumente zu Verfügung. Ältere TeX Distributionen behandeln Texte mit Sonderzeichen, zu denen auch die deutschen Umlaute gehören etwas stiefmütterlich. So auch in dem von mir verwendeten MikTeX. Daher mußte, um die deutschen Umlaute korrekt darstellen zu können die datei german.sty aus dem Internet unter der Adresse ftp.dante.de geladen werden. Unter gleiche Adresse findet man überdies eine Vielzahl nützlicher Erweierungen für LaTeX - so auch die WinShell.

Word erwies sich für den praxisnahen Einsatz als unbrauchbar. Für satztechnisch einfache Texte mag Word vielleicht ausreichen, jedoch nicht für Arbeiten mit vielen mathematischen Formeln. In diesem Punkt ist LaTeX seiner „Konkurrenz“, wenn es sie überhaupt gibt, meilenweit überlegen.

[3] vgl. M. Goossens, The LaTeX Graphics Companion, Seite 351 ff.

Kapitel 11

Programme

Auf der zu dieser Fachbereichsarbeit gehörigen CD-Rom befinden sich folgende ausführbare Programme:
(Die Hilfsdateien sind nicht aufgelistet.)

Windows Programme

- wgl1.exe
- wgl1p.exe
- wquadrgl.exe
- wquadc.exe

Dos Programme

- gl1.exe
- gl1p.exe
- quadrgl.exe
- quadrglc.exe

Kapitel 12

Terminplan

- Jänner 1998 Einarbeitung in das Satzsystem LaTeX
- 10. Juli 1998 1. Grundsatzbesprechung der Fachbereichsarbeit
- 21. September 1998 Besprechung weiterer Kapitel und grundlegende Form
- 12. Oktober 1998 Fertigstellen der Programme
- 18. Dezember 1998 letzte Besprechung der Fachbereichsarbeit
- 15. Februar 1999 letzte Korrekturen, Fehlerkorrektur, Layoutkorrektur
- 18. Februar 1999 Brennen der CD- Rom und Druck
- 19. Februar 1999 Abgabe der Arbeit

Anhang A

Beispiele

Auf den folgenden Seiten finden Sie die Ergebnisse der Lösungsversuche mit Derive und Beispielausdrucke von den Programmen, die auf der zu dieser Fachbereichsarbeit gehörigen CD-Rom enthalten sind.

Literaturverzeichnis

[1] H. J. Bartsch. *Taschenbuch mathematischer Formeln.* Seite 103. Fachbuchverlag Leipzig-Koeln, 1993. ISBN 3-343-00844-3.

[2] Harri Deutsch. *Kleine Enzyklopaedie Mathematik.* Seiten 85-109. Verlag Harri Deutsch, 1980. ISBN 3-87-144-323-9.

[3] Hermann Engesser. *Schuelerduden Mathematik I.* Seiten 148-166. Bibliographisches Institut Mannheim, 1990. ISBN 3-411-04205-2.

[4] Walter Gander. *Computer Mathematik.* Seiten 16-19. Birkhaeuser, 1992. ISBN 3-7643-2765-0.

[5] Michel Goossens. *The LaTeX Graphics Companion.* Addison Wesley Longman, 1997. ISBN 0-201-85469-4.

[6] R. B. King. *Beyond the Quartic Equation.* Seiten 1-5. University of Georgia, 1996. ISBN 3-7643-3776-1.

[7] Torsten Machert. *Wisenschaftliches Publizieren mit LaTeX.* Vieweg, 1998. ISBN 3-528-05664-9.