

RECHENUNGSGENAUIGKEITEN BEI DER ARBEIT MIT COMPUTERN

J. Schürf, Wien ¹⁾

1. „Nobody is perfect“

Bei den folgenden Programmen druckt der Rechner Ergebnisse aus, die teilweise wesentlich von den daneben angegebenen, auf 6 geltende Ziffern richtigen Werten abweichen.

1. Das Programm

```
100 PRINT " X", " 2-X"
110 FOR I=1 TO 5
120 READ X : PRINT X, 2-X
130 NEXT I
140 DATA 1.9, 1.99, 1.999, 1.9999, 1.99999
```

bewirkt die Ausgabe

X	2-X	
1.9	.1	.1
1.99	9.99999E-03	.01
1.999	1.00005E-03	.001
1.9999	9.98974E-05	.0001
1.99999	1.01328E-05	.00001

2. Das Programm

```
100 PRINT " X", " COS(X)"
110 FOR I=1 TO 5
120 READ X : PRINT X, COS(X)
130 NEXT I
140 DATA 1.5, 1.56, 1.57, 1.5707, 1.5708
```

bewirkt die Ausgabe

X	COS(X)	
1.5	.0707371	.0707372
1.56	.0107961	.0107961
1.57	7.96389E-04	7.96327E-4
1.5707	9.62483E-05	9.63268E-5
1.5708	-3.74507E-06	-3.67321E-6

Das Programm

```
3. 100 PRINT " X", " 1-COS(X)", " 2*SIN(X/2)I2"
110 FOR I=1 TO 5
120 READ X : PRINT X, 1-COS(X), 2*SIN(X/2)I2
130 NEXT I
140 DATA .1, .01, .001, .0001, .00001
```

bewirkt die Ausgabe

X	1-COS(X)	2*SIN(X/2)I2	
.1	4.99582E-03	4.99583E-03	4.99583E-3
.01	5.00083E-05	5.00005E-05	4.99996E-5
1E-03	5.36442E-07	4.99933E-07	5E-7
1E-04	0	4.99934E-09	5E-9
1E-05	0	4.92471E-11	5E-11

¹⁾ Aussag aus dem Buch Blaha/Schürf, Programmiertraining
BR.13, 2. Band (Verlag R. Oldenbourg, Wien-München)

Manchmal, wie bei den ersten 2 Beispielen, ist es unvermeidlich, daß eine oder mehrere der ausgedruckten Ziffern „Phantasieziffern“ sind.

Schuld daran ist die begrenzte Genauigkeit, mit der ein mit binärer Verschlüsselung arbeitender Rechner Dezimalzahlen speichern kann.

Die so auftretenden Fehler werden als **Rundungs-, Verschlüsselungs- oder Kodierungsfehler** bezeichnet.

Wir wollen uns aber mit der bloßen Feststellung des Grundes für das Auftreten dieser Kodierungsfehler nicht begnügen, sondern ein Maß für die Größe dieser Fehler angeben. Auf Grund dieses Maßes werden wir imstande sein zu entscheiden, auf welche der vom Rechner ausgedruckten Ziffern wir uns verlassen können und welche von ihnen „Phantasieziffern“ sind.

Manchmal aber ist, wie beim letzten Beispiel, der Programmierer an den auftretenden Ungenauigkeiten schuld, nämlich dann, wenn ein ungünstiges Rechenverfahren gewählt wurde. Man spricht von einem **Verfahrensfehler**. Im letzten Beispiel können wir den Term $1 - \cos x$ umformen in

$$1 - \cos x = 2 \sin^2 \frac{x}{2}$$

und erhalten mit diesem Verfahren einwandfreie Ergebnisse.

2. Kodierungsfehler

Unser Rechner speichert jede Zahl als **Binärzahl** (Dualzahl).

Bei ganzen Zahlen treten hier keine Probleme auf; jede ganze Zahl ist exakt als Binärzahl darstellbar und, wenn sie betragsmäßig nicht zu groß ist, auch exakt speicherbar.

Viele Dezimalzahlen kann man ebenfalls exakt als Binärzahlen darstellen, z. B. .25, 0.5, 1.125.

Im allgemeinen würde aber eine Dezimalzahl zur exakten Darstellung eine Binärzahl mit unendlich vielen Binärziffern erfordern. Zur praktischen Verwendung muß also die Binärzahl irgendwo abgebrochen werden.

Über die so auftretenden Kodierungsfehler wollen wir uns mit Hilfe des nachfolgenden Programms einen Überblick verschaffen.

```
100 READ X# : X=X#
110 KF=X#-X : KR=KF/X#
120 PRINT X#,X,KF,KR
    : PRINT CDBL(X) : PRINT : GOTO 100
200 DATA 23, .25, 3.3125, .333333333#, .123456789#, 123456789#,
    3.14159265, 2.508
```

Zunächst wird jeweils eine Konstante in die doppelgenaue Variable X# eingelesen und dann die einfachgenaue Variable X=X# gesetzt. Der dabei auftretende **Kodierungsfehler** ist $KF = X\# - X$,

der **relative Kodierungsfehler** $KR = KF/X\#$.

Aus der Ausgabe ersehen wir:

Der relative Kodierungsfehler KR ist für INT-Zahlen Null, INT-Zahlen werden exakt gespeichert.

Für SNG- und DBL-Zahlen ermittelt das folgende Programm den jeweiligen relativen Kodierungsfehler KR.

Es berechnet für jede eingetastete Zahl A die kleinste positive Zahl X, für die noch $A+X > A$ gilt, und den Quotienten X/A .

Alle Zahlen $A+Z$ mit $0 \leq Z < X$ werden als A gespeichert.

Der Quotient X/A entspricht somit dem relativen Kodierungsfehler KR.

Das Programm

```
100 I=0 : INPUT "DOPPELTE GENAUIGKEIT ( JA = 1 ) "; I
    : IF I=1 THEN DEFDBL A : E=1E16 : ELSE DEFSNG A : E=1E6
110 INPUT "A ="; A : H=A/E : X=0
120 X=X+H : IF A+X=A THEN 120
130 X=X-H : H=H/10 : IF H>X*.0001 THEN 120
140 PRINT CSNG(X/A) : PRINT : GOTO 100
```

bewirkt die Ausgabe:

```
DOPPELTE GENAUIGKEIT ( JA = 1 ) ?
A =? 1
5.96E-08
```

```
DOPPELTE GENAUIGKEIT ( JA = 1 ) ?
A =? 1E3
3.051E-08
```

```
DOPPELTE GENAUIGKEIT ( JA = 1 ) ?
A =? 1E30
3.777E-08
```

Welchen größten Wert kann nun der Betrag von KR bei den verschiedenen Variablentypen annehmen? Wir bezeichnen diesen Wert als **Kodierungsgrenzfehler KG**.

Wir könnten nun KR für eine große Anzahl von Zahlen A mit Hilfe des vorstehenden Programms berechnen, um eine Abschätzung von KG zu erhalten. Einfacher ist es, das folgende Programm, das Zufallszahlen verwendet, mit großem N laufen zu lassen.

Das Programm

```
100 I=0 : INPUT "DOPPELTE GENAUIGKEIT ( JA = 1 ) "; I
    : IF I=1 THEN DEFDBL A : E=1E16 : ELSE DEFSNG A : E=1E6
105 INPUT "N ="; N : K=0 : M=0
110 A=RND(10) : H=A/E : X=0
120 X=X+H : IF A+X=A THEN 120
130 X=X-H : H=H/10 : IF H>X*.0001 THEN 120
140 IF X/A>M THEN M=X/A
150 K=K+1 : IF K<N THEN 110
160 PRINT "M="M, "N ="N : GOTO 100
```

bewirkt die Ausgabe:

DOPPELTE GENAUIGKEIT (JA = 1) ?
N = ? 1000
M = 5.96E-08 N = 1000
DOPPELTE GENAUIGKEIT (JA = 1) ? 1
N = ? 1000
M = 1.387E-17 N = 1000

Wir sehen, daß die oben für $A = 1$ berechneten KR-Werte betragsmäßig nie überschritten werden.

Wir wollen uns also merken:

KG = 0 für INT-Zahlen
KG \approx 6E-8 für SNG-Zahlen
KG \approx 1.4E-17 für DBL-Zahlen

3. Relativer Fehler einer Funktion in einer Variablen

Durch welches allgemeine Verfahren können wir in jedem einzelnen Fall feststellen, auf wie viele geltende Ziffern man sich verlassen kann?

Wir beantworten zunächst die Frage:

Wenn in einer Funktion $f(x)$ das Argument mit dem Fehler Δx , also dem relativen Fehler $RF(x) = \frac{\Delta x}{x}$ behaftet ist, wie groß ist dann der relative Fehler $RF(y) = \frac{\Delta y}{y}$ des Funktionswerts $y = f(x)$?

Der Verhältnissfaktor der beiden relativen Fehler ist

$$VF = \frac{RF(y)}{RF(x)} = \frac{\Delta y}{\Delta x} \cdot \frac{x}{y}$$

Es gilt also

$$RF(y) = VF \cdot RF(x)$$

Aus dieser Gleichung kann man bei bekanntem Fehler $RF(x)$ und bei bekanntem Verhältnissfaktor VF den relativen Fehler $RF(y)$ berechnen. ...¹

¹ VF ist eine Funktion von x und Δx , die Abhängigkeit von Δx kann aber bei kleinen Δx vernachlässigt werden.

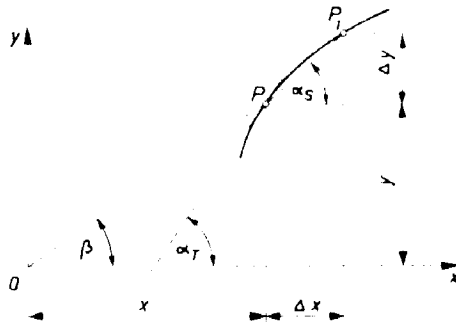
$RF(x)$ kann durch eine Meßungenauigkeit bedingt sein oder durch die nur angenähert mögliche Kodierung der meisten Dezimalzahlen durch unseren Rechner. In diesem letzten Fall gilt $|RF(x)| < KG$, wo KG der schon besprochene Kodierungsgrenzfehler ist. Daraus folgt die Abschätzung

$$|RF(y)| < |VF| \cdot KG$$

Hinweis:

Wenn $|VF| < 1$ ist, gilt nur $|RF(y)| < KG$, da ja auch y nur angenähert gespeichert werden kann.

Wir wollen nun den Verhältnissfaktor $VF = \frac{\Delta y / y}{\Delta x / x}$ geometrisch deuten.



$\frac{\Delta y}{\Delta x}$ ist die Steigung der Sehne PP_1 . Es gilt $\frac{\Delta y}{\Delta x} = \tan \alpha_S$.

Da wir nur kleine Fehler Δx und Δy betrachten wollen, können wir $\tan \alpha_S \approx \tan \alpha_T = y'$ setzen, also die Steigung der Sehne durch die Steigung der Tangente ersetzen¹.

Es gilt daher

$$\frac{\Delta y}{\Delta x} \approx \tan \alpha_T$$

$\frac{y}{x}$ ist die Steigung der Strecke OP , also

$$\frac{y}{x} = \tan \beta$$

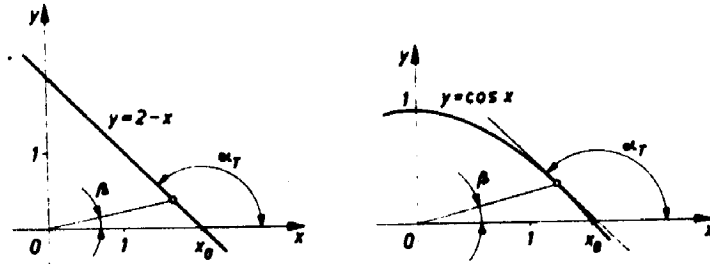
Der Verhältnissfaktor ist daher

$$VF \approx \frac{\tan \alpha_T}{\tan \beta}$$

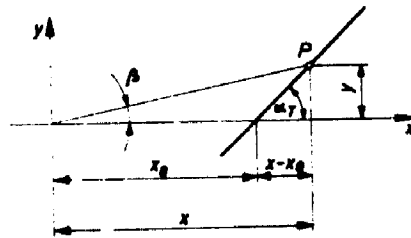
¹ Wir setzen also voraus, daß die Funktion differenzierbar ist.

4. Kritische Stellen

Besonders interessieren uns die Stellen, wo $|VF|$ sehr groß ist. Solche „kritische“ Stellen liegen z. B. in der Umgebung der Nullstellen einer Funktion, weil dort $\tan\beta$ sehr klein ist, wie etwa in den ersten zwei Beispielen,



In der Umgebung der Nullstelle x_0 können wir uns die Funktion durch ihre Tangente ersetzt denken¹:



Der Verhältnissfaktor VF ist dann

$$VF \approx \frac{\tan\alpha_T}{\tan\beta} \approx \frac{y}{x-x_0} \Big/ \frac{y}{x} = \frac{x}{x-x_0}$$

Den Quotienten $\delta = \frac{x-x_0}{x}$ können wir als *relativen Abstand der Stelle x von der Nullstelle x_0* deuten.

$$\boxed{VF \approx \frac{1}{\delta}} \quad \dots^2$$

¹ Wir beschränken uns auf Nullstellen mit nichthorizontaler Tangente, also auf Nullstellen erster Ordnung.

² Für Nullstellen n -ter Ordnung erhalten wir $VF \approx \frac{n}{\delta}$.

5. Relativer Fehler einer Funktion in zwei Variablen

Von den Funktionen in zwei Variablen wollen wir nur die Grundrechnungsarten betrachten.

Der relative Fehler des **Produkts** xy ist

$$\frac{\Delta(xy)}{xy} = \frac{y\Delta x + x\Delta y}{xy} = \frac{\Delta x}{x} + \frac{\Delta y}{y} \quad \dots 1$$

oder

$$\text{RF}(xy) = \text{RF}(x) + \text{RF}(y)$$

Für die **Division** ergibt sich in ähnlicher Weise

$$\text{RF}\left(\frac{x}{y}\right) = \text{RF}(x) + \text{RF}(y)$$

In beiden Fällen ergibt die Fehlerabschätzung

$$|\text{RF}(xy)| \text{ bzw. } \left| \text{RF}\left(\frac{x}{y}\right) \right| \leq |\text{RF}(x)| + |\text{RF}(y)|$$

Der Fehler des Ergebnisses übersteigt also größenordnungsmaÙig nicht die Fehler der Argumente oder, anders ausgedrückt:

Das Ergebnis enthält etwa so viele richtige Ziffern wie das ungenauere der Argumente.

Bei der **Addition** erhält man

$$\frac{\Delta(x+y)}{x+y} = \frac{\Delta x + \Delta y}{x+y} = \frac{\Delta x}{x} \cdot \frac{x}{x+y} + \frac{\Delta y}{y} \cdot \frac{y}{x+y}$$

$$\text{oder } \text{RF}(x+y) = \text{RF}(x) \cdot \frac{x}{x+y} + \text{RF}(y) \cdot \frac{y}{x+y}$$

Die relativen Fehler $\text{RF}(x)$ bzw. $\text{RF}(y)$ sind hier mit den Faktoren $x/(x+y)$ bzw. $y/(x+y)$ multipliziert. Wenn x und y gleiches Vorzeichen haben, sind diese Faktoren kleiner als 1, der relative Fehler des Ergebnisses wird also auch hier in der Größenordnung des größeren der beiden Teilfehler $\text{RF}(x)$ und $\text{RF}(y)$ bleiben.

¹ Eigentlich gilt

$$\Delta(xy) = (x + \Delta x) \cdot (y + \Delta y) - xy = y\Delta x + x\Delta y + \Delta x \cdot \Delta y.$$

Da wir jedoch nur kleine Fehler Δx und Δy betrachten, können wir das Produkt $\Delta x \cdot \Delta y$ als klein gegenüber den anderen Summanden vernachlässigen.

Kritische Stellen liegen aber z. B. auch in der Umgebung von vertikalen Asymptoten, weil dort $\tan \alpha_T$ sehr groß ist.

Ohne Beweis soll hier angeführt werden, daß sich in diesem Fall

$$VF \approx -\frac{1}{\delta}$$

ergibt!

Zusammenfassung:

In der Umgebung einer kritischen Stelle x_0 gilt

$$|VF| \approx \frac{1}{|\delta|} \quad \text{mit} \quad \delta = \frac{x-x_0}{x}$$

BEISPIEL

Das Beispiel 2 von Seite ist so zu erweitern, daß auch VF und $N = \lg|VF|$ ausgedruckt werden.

Das Programm:

```
100 PRINT " X", " COS(X)", " VF", " N"  
    : M=1/LOG(10) : X0=1.5707963  
110 FOR I=1 TO 5 : READ X : VF=X/(X-X0)  
120 PRINT X, COS(X), VF, LOG(ABS(VF))*M  
130 NEXT I  
200 DATA 1.5, 1.56, 1.57, 1.5707, 1.5708
```

bewirkt die Ausgabe:

X	COS(X)	VF	N
1.5	.0707371	-21.1876	1.32608
1.56	.0107961	-144.495	2.15985
1.57	7.96389E-04	-1971.57	3.29481
1.5707	9.62483E-05	-16306.9	4.21237
1.5708	-3.74507E-06	411776	5.61466

Aus der Zahl $N \approx 5.6$ schließen wir, daß y um 5 bis 6 Stellen weniger genau ist als das Argument. Da das Argument bei SNG-Genauigkeit nur auf 7 bis 8 Stellen genau gespeichert werden kann, so dürfen wir uns beim Funktionswert nur auf etwa 2 Stellen verlassen.

¹ Wir setzen voraus, daß der Kehrwert der betrachteten Funktion in x_0 eine nichthorizontale Tangente hat, die asymptotische Annäherung der ursprünglichen Funktion also von erster Ordnung ist. Bei asymptotischer Annäherung der Ordnung n erhalten wir $VF \approx -\frac{n}{\delta}$.

Wenn aber x und y verschiedene Vorzeichen haben, also bei der **Subtraktion**, dann können die Faktoren $x/(x \pm y)$ und $y/(x \pm y)$ sehr groß sein.

Ein Beispiel soll dies erläutern.

$x = 5.042867$ und $y = -5.042341$ sollen beide auf 7 geltende Ziffern genau sein, die weiteren Dezimalstellen sind nicht bekannt. Wie genau ist $x \pm y$?

$$\begin{array}{r} 5.042867 ??? \\ - 5.042341 ??? \\ \hline 0.000526 ??? \end{array}$$

Wie man sieht, ist das Ergebnis nur noch auf 3 geltende Ziffern genau.

Das ergibt sich auch aus der Berechnung von $RF(x \pm y)$:

Die Faktoren $x/(x \pm y)$ und $y/(x \pm y)$ liegen bei unserem Beispiel in der Größenordnung von 10^3 ; die ursprüngliche Genauigkeit von 10^{-7} (7 geltende Ziffern) ist also auf $10^{-7} \cdot 10^3 = 10^{-3}$ (3 geltende Ziffern) reduziert worden.

Durch Subtraktion können also erhebliche Genauigkeitsverluste entstehen.

Der Rechner druckt eine bestimmte Anzahl von Stellen aus, man muß sich aber im klaren sein, daß einige Ziffern (im Extremfall sogar alle!) Phantasieziffern sein können.

8.6. Verfahrensfehler

Wir haben schon auf Seite 156 ein Beispiel angegeben, bei dem nicht Kodierungsfehler für die Ungenauigkeiten im Ergebnis verantwortlich waren, sondern ein ungünstiges Rechenverfahren. Wir konnten dadurch bessere Ergebnisse erreichen, daß wir die

Funktion $y = 1 - \cos x$ durch $y_1 = 2 \sin^2 \frac{x}{2}$ ersetzen und so die genauigkeitsvermindernde Subtraktion vermeiden.

y und y_1 liefern zwar bei unbegrenzter Genauigkeit die gleichen Ergebnisse, bei **begrenzter** Genauigkeit¹ ist aber das zweite Verfahren, besonders in der Umgebung von $x = 0$, wesentlich genauer.

weiteres Beispiel:

Die Funktion $y = \sqrt{1+x} - \sqrt{1-x}$ wird besser nach folgendem Verfahren berechnet:

$$y_1 = \frac{2x}{\sqrt{1+x} + \sqrt{1-x}}$$

¹ Jeder Rechner hat nur eine begrenzte Genauigkeit!

Bei dem folgenden, der technischen Praxis entnommenen Beispiel handelt es sich ebenfalls um einen Verfahrensfehler.

Gegeben seien $D1 = 11$ und $D2 = 10.834$.

$$\text{Zu berechnen ist } R0 = \frac{D1}{2} \cdot \sqrt{1 + \left(\frac{D2}{D1}\right)^2} + \frac{1 - (D2/D1)^2}{\ln(D2/D1)}$$

Mit dem Programm

```
100 READ D1, D2
110 D3=D2/D1 : D4=D3*D3
120 PRINT D1, D2, D1/2*SQR(1+D4+(1-D4)/LOG(D3)) : GOTO 100
130 DATA 11, 10.834
```

erhalten wir die Ausgabe

```
11          10.834          .061122
```

Mit dem Taschenrechner TI-58, der bekanntlich mit 13 Stellen rechnet und 10stellig anzeigt, erhalten wir $R0 = 0.0677693$.

Wir erkennen: 10% Abweichung!

Nicht einmal die erste Stelle ist sicher!

Wir wollen nun die einzelnen Rechenschritte überprüfen. Wir rechnen wieder mit unserem Taschenrechner und runden alle Zwischenwerte auf 7 geltende Ziffern. Wir erhalten

$$\begin{aligned} D3 &= D2/D1 = .9849091 \\ D4 &= D3*D3 = .9700459 \\ 1-D4 &= .0299541 \text{ (nur noch 6 geltende Ziffern!)} \\ (1-D4)/\text{LOG}(D3) &= -1.96989 \text{ (auch nur 6 geltende Ziffern)} \end{aligned}$$

Der Radikand $1 + D4 + (1 - D4)/\text{LOG}(D3)$ ergibt sich zu

$$\begin{array}{r} 1.970046... \\ -1.96989... \\ \hline 0.00016... \end{array}$$

Die Genauigkeit ist von 7 auf 2 geltende Ziffern zusammengeschrumpft!

Auch das Endergebnis ist daher nur noch auf 2 geltende Ziffern genau.

Daß unser Rechner nicht einmal diese 2 geltenden Ziffern richtig liefert, hat seinen Grund darin, daß bei ihm die eingebaute LOG-Funktion für x -Werte in der Nähe von 1 schlecht programmiert ist. Es gibt aber BASIC-Varianten, bei denen dieser Fehler behoben ist und die tatsächlich $R0$ nach dem oben angegebenen Rechenverfahren auf 2 geltende Ziffern genau liefern.

Es stellt sich nun die Frage, ob man durch ein anderes Rechenverfahren zu einem genaueren Ergebnis kommen kann.

Da sich bei unserem Problem keine Umformung anbietet, wollen wir eine Reihenentwicklung versuchen.

Wir setzen

$$\delta = \frac{D1 - D2}{D1 + D2}$$

und entwickeln nach Potenzen von δ in eine Taylor-Reihe.

Wir erhalten:

$$R0 = \frac{D1 - D2}{\sqrt{6}} \cdot \left(1 - \frac{\delta^2}{30} + \dots \right)$$

Wenn wir in erster Näherung den Klammerinhalt gleich 1 setzen, erhalten wir

$$R = \frac{D1 - D2}{\sqrt{6}}$$

Mit den Daten unseres Beispiels ergibt sich $R = .0677692$, also ein schon auf 5 geltende Ziffern genauer Wert!

Die nächstbeste Näherung ist

$$R1 = R \cdot (1 - \delta^2/30)$$

R1 liefert für kleine $|\delta|$ gute Ergebnisse, für große $|\delta|$ aber ungenaue Werte, da sich dann das Weglassen der Glieder höherer Ordnung auswirkt. Für große $|\delta|$ können wir aber die Definitionsformel für R0 von der vorhergehenden Seite verwenden, die ja nur für kleine $|\delta|$ unbrauchbar ist.

Mit Hilfe dieses Testprogramms können wir die Grenze feststellen, bis zu der man (bei einfacher Genauigkeit) besser mit R1 und ab der man besser mit R0 rechnet. Bei unserem Rechner liegt diese Grenze bei $|\delta| \approx 0.12$.

Aufgrund unserer Überlegungen sind wir nun imstande, das folgende kurze Programm aufzustellen, das im gesamten Anwendungsbereich der Formel einwandfreie Ergebnisse liefert, obwohl es nur mit einfacher Genauigkeit arbeitet.

```
10 READ D1,D2 : D3=D2/D1 : D4=D3*D3 : D=(D1-D2)/(D1+D2)
20 PRINT "D1 ="D1, "D2 ="D2 : PRINT "R0 =";
30 IF ABS(D) > .12 THEN PRINT D1/2*SQR(1+D4+(1-D4)/LOG(D3))
   ELSE PRINT ABS(D1-D2)/SQR(6)*(1+D*D/30)
40 DATA 11,10.934
```

Dieses Programm bewirkt die **Ausgabe:**

```
D1 = 11           D2 = 10.934
R0 = .0677691
```