Fachhochschule Hamburg, Fachbereich Seefahrt

Sommersemester 1984

Wolf-Werner Scheuermann

Das Programm "Formel-Evolution"

Veranstaltung: Datenverarbeitung, 2. Semester AG

Prof. Dr. Berking

INHALT

BESCHREIBUNG

Das Programm "Formel-Evolution" entwickelt für vorgegebene, beliebige Wertetabellen ((x,y)) Formeln y=f(x), die innerhalb einer einstellbaren Fehlergrenze die Wertetabelle approximieren.

Das Charakteristikum dieses Programms ist, daß diese Kurvenanpassung nicht nach bekannten mathematischen oder numerischen Algorithmen erfolgt, sondern mittels Evolutions-Strategie, Mutation und Selektion, bzw. Versuch und Irrtum. Der Zufall RND(spielt also eine große Rolle.

Das Verfahren ist folgendes:
Intern wird zur Formeldarstellung die umgekehrte polnische Notation verwendet (UPN), die sehr maschinennah ist und die problemlose Zusammenstellung von Formeln ohne syntaktische Schwierigkeiten erlaubt.

Es stehen 27 Funktionen zur Verfügung, die an die Funktionen des HP-41C Taschenrechners angelehnt sind und als Subroutinen zu Beginn des Programms (Zeilen 2-47) implementiert sind, wobei wichtige Funktionen doppelt oder mehrfach vorkommen (Gewichtung durch Häufung). Diese Funktionen arbeiten auf einem simulierten, 4 Register tiefen Stack B,C,D,F, wo B das Ergebnisregister darstellt. Jede Funktion hat eine Codezahl (ihre laufende Nummer).

Die Formel liegt als Folge von Funktionscodes im Array B(1..K) vor. Die Länge K der Formel, d.h. die Anzahl der Funktionen, aus der die Formel maximal bestehen darf, ist ein durch den Benutzer variierbarer Parameter.

Der Zufall bestimmt nun, wieviele Funktionscodes geändert (mutiert) und durch welche neuen Codes sie substituiert werden sollen. Der Parameter Mutativität N (ebenfalls vom Benutzer einstellbar) legt dabei fest, wieviele Funktionscodes maximal auf einen Schlag geändert werden dürfen (Zeile 51).

Die so entstandene neue Formel wird nun interpretiert und mit der Wertetabelle verglichen (Zeilen 52-56). Die Interpretation geschieht dadurch, daß entsprechend der Reihenfolge der Funktionscodes in der Formel die entsprechenden Subroutinen aufgerufen und abgearbeitet werden. Wird für ein bestimmtes ARGUMENT der Wertetabelle auf diese Weise die gesamte Formel abgearbeitet, so steht zuletzt im Register B der Funktionswert der aktuellen Formel für dieses spezielle Argument.
Der Betrag der Differenz zwischen Formelwert und y-Wert der Wertetabelle ist der Fehler der aktuellen Formel an der Stelle x. Wird die Interpretation für alle Argumente x der Wertetabelle wiederholt, und die dabei auftretenden Fehler summiert, so erhält man einen Gesamtfehler , der die Güte der aktuellen Formel charakterisiert.

Hier setzt nun die Evolutionsstrategie an:
Ist der Fehler der neuen Formel kleiner als derjenige der alten Formel, aus der sie erzeugt worden ist, so wird die neue Formel zum Ausgangspunkt der weiteren Entwicklung gemacht (Zeile 57). In diesem Falle wird eine Erfolgsmeldung ausgegeben, der Gesamtfehler der neuen Formel, sowie eine Übersetzung des Formelcodes in Mnemonics (UPN) (Array A$(1..L)). Ist der Gesamtfehler kleiner als die Fehlerschranke 0, des dritten Parameters, der vom Benutzer bestimmt wird, so wird die Evolution beendet und es findet eine abschließende, benutzerorientierte Übersetzung der Postfix-Notation in die gewohnte Infix-Notation statt (Zeilen 58-87). Dabei wird wiederum auf einem Stack, diesmal aus den Stringvariablen B$,C$,D$,E$ bestehend, gearbeitet und das Ergebnis steht zuletzt in B$.

Ist der Fehler der neuen Formel jedoch zu irgendeinem Zeitpunkt der Interpretation größer als derjenige der alten Formel, so wird eine

Misserfolgsmeldung ausgegeben, die neue Formel verworfen und weiterhin die alte Formel als Grundlage der Mutation genommen (Zeile 48). Selektionskriterium ist also die Größe des Fehlers einer Formel. Gestartet wird immer mit der leeren Formel (besteht nur aus leeren Operationen), deren Fehler unendlich gesetzt wird (exakt: 9*10^30).
Ein schwerwiegendes Problem mußte gelöst werden:
Wie erwähnt gibt es mit der Syntax der Formeln dank UPN keine Probleme (schlimmstenfalls zerfällt die Formel in mehrere unabhängige Teile), jedoch gibt es keine Garantie für das Vermeiden arithmetischer Fehler. Selbstverständlich ist das Auftreten eines Arithmetikfehlers ein negatives Selektionskriterium für die betreffende Formel, aber es muß vor allem sichergestellt werden, daß es dadurch nicht zum Systemabbruch kommt. Das wurde durch Werteberichsprüfungen innerhalb der funktionsausführenden Subroutinen erreicht. Ein positiver Test führt zum Abbruch der Interpretation und zum Sprung in die Misserfolgsroutine (Zeile 48). Vorher muß jedoch die letzte Rücksprungadresse gelöscht werden, da das Unterprogramm nicht über RETURN verlassen wurde und die weitere Iteration mit weiteren arithmetischen Fehlern sonst unweigerlich zum Überlauf des Rücksprungadressenstacks mit Systemabbruch und OUT OF MEMORY ERROR führen würde. Hier kommt also der seltene BASIC-Befehl POP zur Anwendung (Zeile 49). Im allgemeinen ist das Programm jetzt in der Lage tagelang ohne Abbruch zu rechnen und für beliebige Wertetabellen Formeln zu entwickeln. Ideal wäre hier die Anwendung des Befehls ONERR GOTO. Leider ist die Fehlerroutine des Applesoftinterpreters selbst fehlerhaft (siehe Dederichs 1982, S.181).

ALLGEMEINES

Trotz des simplen Verfahrens und der geringen Spezifität des Programms erstaunt die Effizienz des Algorithmus.
Sicherlich steigt mit der Komplexität der darzustellenden Wertetabelle der Rechenaufwand für die Entwicklung brauchbarer Formeln, dennoch verblüfft die Geschwindigkeit, mit der die Evolution voranschreitet. Insbesondere bei einfachen Wertetabellen ist dieses Verhalten offenkundig und es überrascht weiterhin die geradezu kreative Vielfalt der (äquivalenten) Lösungen, die man erhält, wenn man das Programm mehrfach auf dieselbe Wertetabelle ansetzt (siehe Anhang).

Das Programm erlaubt weiterhin Experimente, z.B. über die Abhängigkeit der Evolutionsgeschwindigkeit von den Rahmenbedingungen, beispielsweise der Einstellung der Parameter Codelänge und Mutativität.

Das Arbeiten des Programms vermag auch die Funktionstüchtigkeit der Evolutionsprinzipien evident und die Entstehung immer komplexerer Organismen aus niedrigeren Strukturen einleuchtend zu machen. Interessant ist dabei folgender Aspekt:
Das Programm "Formel-Evolution" verwendet ein schwaches Selektionskriterium, d.h. es wird eine neue Formel auch dann zur Basis der weiteren Evolution gemacht, wenn sie gleichgut ist, wie die alte. Ein starkes Selektionskriterium hieße, daß nur dann die neue Formel zum Ausgangspunkt der weiteren Entwicklung gemacht würde, wenn sie eine echte Verbesserung gebracht hätte (in Zeile 56 müßte die Anweisung: `IF T>M GOTO 48` in: `IF T>=M GOTO 48` geändert werden). Entgegen der Erwartung ist das starke Selektionskriterium keineswegs schneller. Das liegt daran, daß echte Verbesserungen einer Formel relativ selten sind und deshalb ein und dieselbe Formel lange unverändert erhalten bliebe. Das ist der Evolution hinderlich. Das schwache Selektionskriterium dagegen erlaubt die "Vererbung" von Formelteilen, die nicht schädlich sind, auch wenn sie nicht direkt nützlich sind. So findet hier eine ständige Veränderung der Formel in nicht relevanten (nicht das Ergebnis beeinflussenden) Teilen statt. Diese Veränderungen erhöhen aber wesentlich die Chancen für einen positiven Evolutionsschritt, wenn nämlich durch Zufall eine Verbindung zwischen relevanten und isolierten Teilen hergestellt wird. Statt von isolierten und relevanten Teilen könnte man auch von rezessiven und dominanten Merkmalen im Sinne der

Vererbungslehre sprechen.

Als letzter Aspekt sei erwähnt, daß das hier beschriebene Verfahren sich ideal für einen parallelen Multiprozessorbetrieb, bzw. ein nichthierarchisches Computernetz eignen würde. Jeder Prozessor bzw. Computer hätte dasselbe Programm, jedoch eine unterschiedliche Initialisierung seiner Zufallsfunktion. Derjenige Prozessor oder Computer, der zuerst eine verbesserte Formel fände, würde die Arbeit aller anderen unterbrechen, seine Formel an alle übermitteln und die Fortsetzung der Evolution veranlassen. Durch den parallelen Betrieb dürfte das Verfahren wesentlich schneller konvergieren.

VERWENDETES SYSTEM

Personal-Computer: APPLE II europlus, 64 K-Byte RAM, 2 Diskdrives, Sanyo-Monitor monochrom, schwarz-grün

Matrixdrucker: Epson MX-82F/T

Programmiersprache: Applesoft-Basic

FUNKTIONEN

Code	Zeile	Mnemonic	Funktion
1,2	2,3	+	Addition
3,4	4,5	*	Multiplikation
5,6	6-8	/	Division
7,8	9,10	-	Subtraktion
9	11	ARCTAN	Arcustangens
10	12	ABS	Absolutbetrag
11	13	INT	ganzzahliger Teil
12	14	FRC	Dezimalteil
13	15	CHS	Vorzeichenwechsel
14	16	SIGN	Vorzeichen
15	17	X^2	Quadrat
16	18,19	SQRT	Wurzelfunktion
17	20	SIN	Sinus
18	21-22	TAN	Tangens
19	23	COS	Cosinus
20	24-25	1/X	Kehrwert
21	26-27	E^X	e-Funktion
22	28-29	LN	logarithmus naturalis
23	30	RDN	scroll down (Stackoperation)
24	31	R^	scroll up (Stackoperation)
25	32-33	MOD	Rest einer Division
26	34	ENTERß	Hochschieben des Ergebnisses
27	35-37	Y^X	Exponentialfunktion
28,28,30,31	38-41	ARGUMENT	x-Wert
32,33	42,43	1	Konstante
34,35	44,45	2	Konstante
36,37	46,47	3	Konstante

VARIABLE

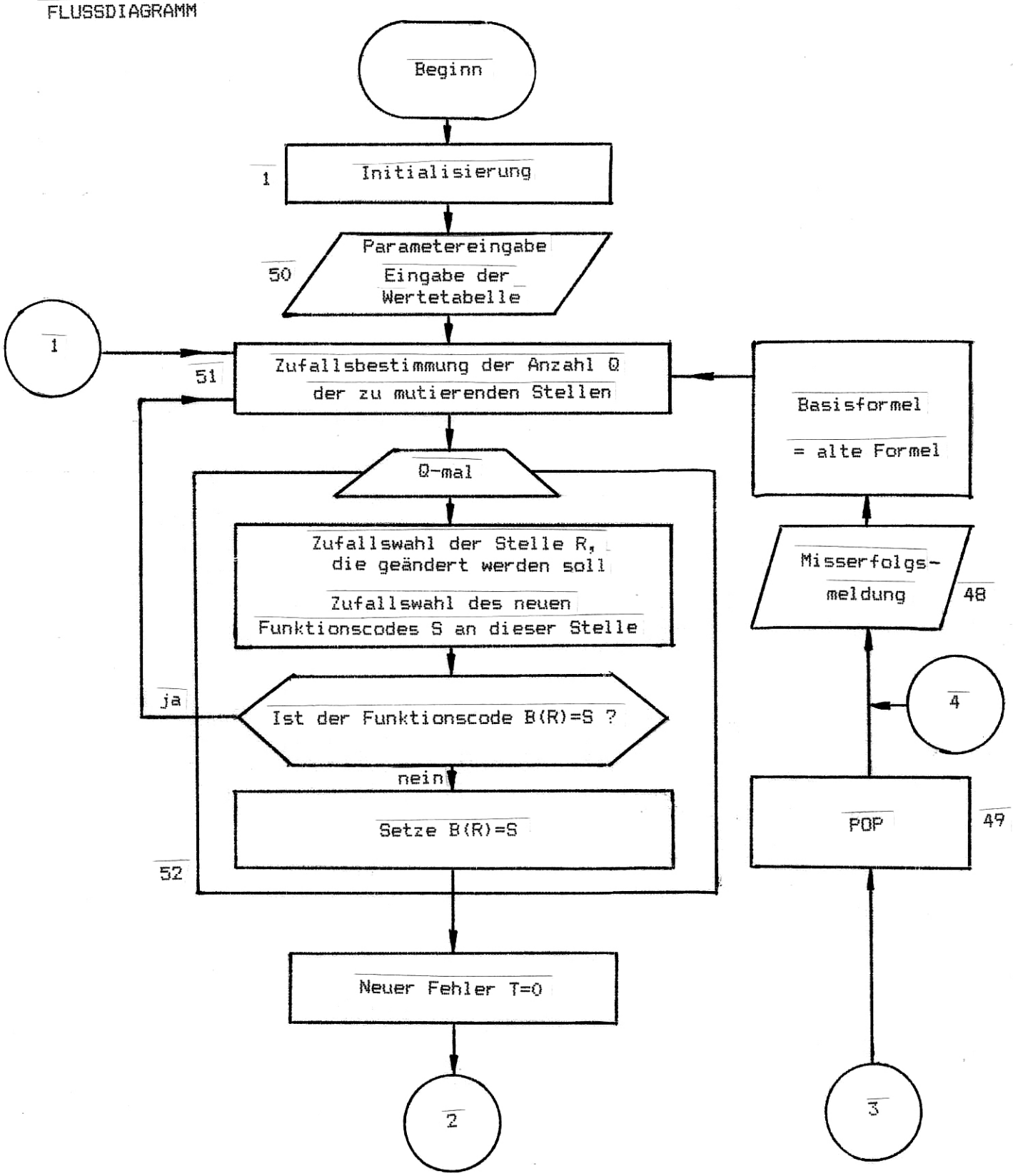
A$(	Array der Funktionenmnemonics (UPN)
B$,C$,D$,E$	Stringstack
F$	Hilfsstring
A(	x-Werte der Wertetabelle
B(	neue Formel
C(	alte Formel
D(	y-Werte der Wertetabelle
A	Konstante, pi/2
B,C,D,F	Stack
E	Erfolgszähler
G	Hilfsvariable
H,I	Schleifenzähler
J	Misserfolgszähler
K	Codelänge
L	Gesamtzahl der verfügbaren Funktionen
M	alter Fehler
N	Mutativität
O	Fehlerschranke
P	Anzahl der Wertepaare
Q	Zahl der zu mutierenden Stellen
R	Stelle
S	Funktionencode
T	neuer Fehler

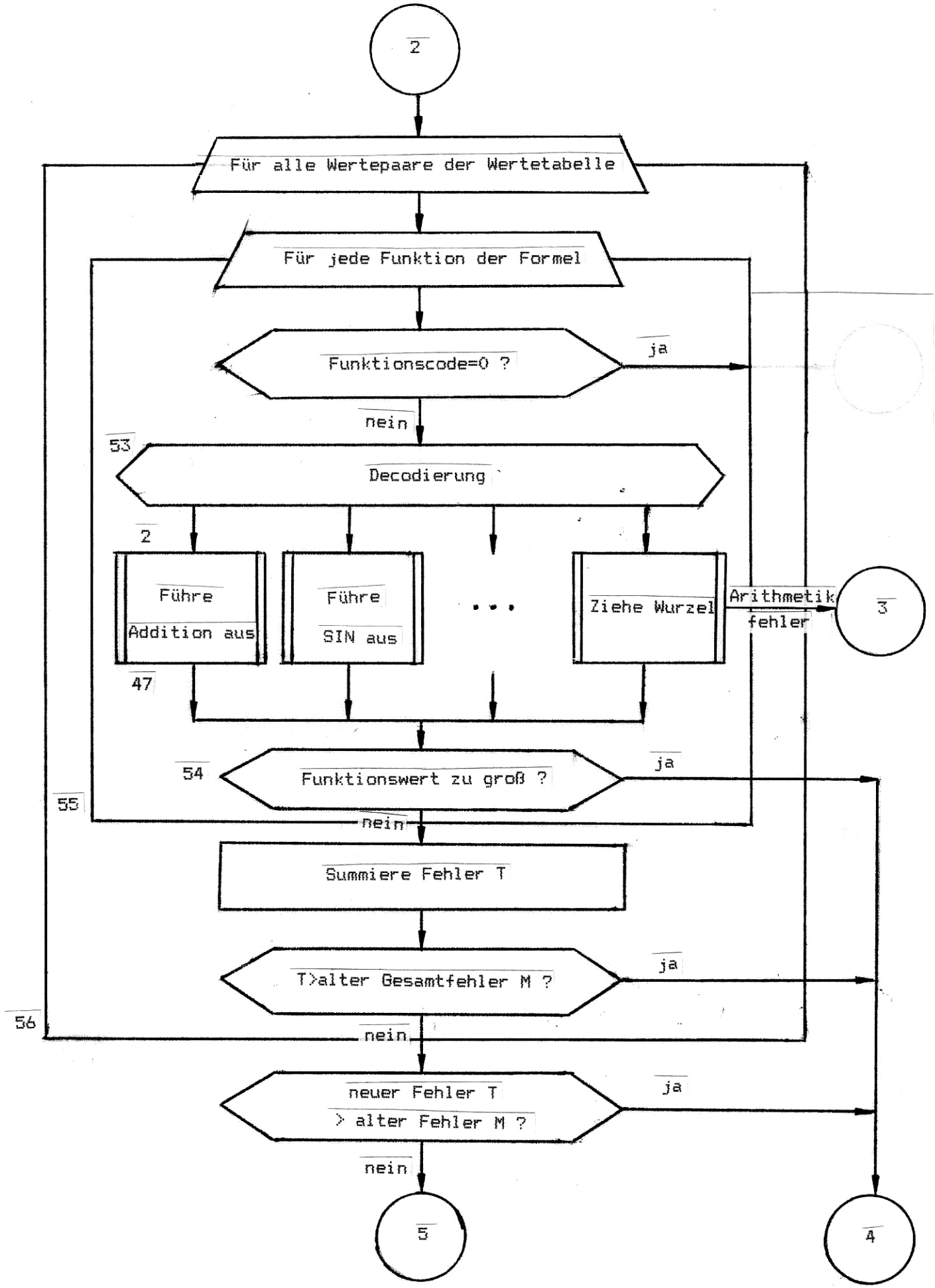
LITERATUR

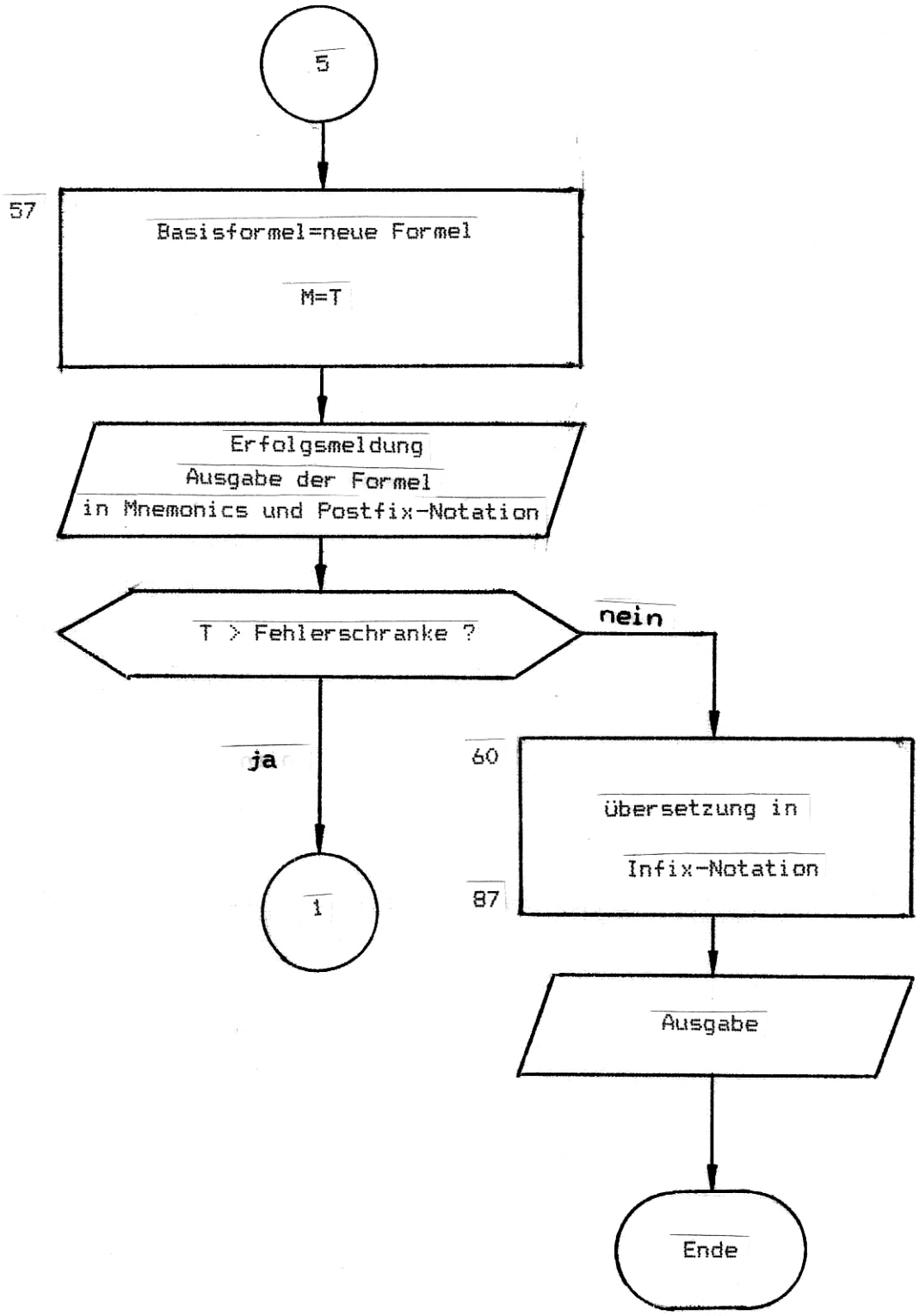
W. Dederichs, Applesoft-Basic, Mannheim 1982

I. Rechenberg, Evolutionsstrategie.Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, Stuttgart-Bad Cannstatt 1973

FLUSSDIAGRAMM




2
Für alle Wertepaare der Wertetabelle
Für jede Funktion der Formel
Funktionscode=0 ?
ja
nein
53
Decodierung
2
Führe Addition aus
Führe SIN aus
...
Ziehe Wurzel
Arithmetikfehler
3
47
54
Funktionswert zu groß ?
ja
55
nein
Summiere Fehler T
T>alter Gesamtfehler M ?
ja
56
nein
neuer Fehler T > alter Fehler M ?
ja
nein
5
4


5
57
Basisformel=neue Formel
M=T
Erfolgsmeldung
Ausgabe der Formel
in Mnemonics und Postfix-Notation
T > Fehlerschranke ?
nein
ja
60
Übersetzung in
Infix-Notation
87
1
Ausgabe
Ende

PROGRAMM-LISTING

```
ßLIST

1  HOME : PRINT "FORMEL-EVOLUTION": PRINT "================": PRINT : PRINT "W.-W. SCHEUERMANN": PRINT :A = 1.57079633: GOTO 50
2 B = B + C:C = D:D = F:F = 0: RETURN : DATA "+ "
3 B = B + C:C = D:D = F:F = 0: RETURN : DATA "+ "
4 B = B * C:C = D:D = F:F = 0: RETURN : DATA "* "
5 B = B * C:C = D:D = F:F = 0: RETURN : DATA "* "
6  IF B = 0 GOTO 49
7  IF B = 0 GOTO 49
8 B = C / B:C = D:D = F:F = 0: RETURN : DATA  "/ ","/ "
9 B = C - B:C = D:D = F:F = 0: RETURN : DATA  "- "
10 B = C - B:C = D:D = F:F = 0: RETURN : DATA  "- "
11 B =  ATN (B): RETURN : DATA "ARCTAN "
12 B =  ABS (B): RETURN : DATA "ABS "
13 B =  INT (B): RETURN : DATA "INT "
14 B = B -  INT (B): RETURN : DATA "FRC "
15 B =  - B: RETURN : DATA "CHS "
16 B =  SGN (B): RETURN : DATA  "SIGN "
17 B = B * B: RETURN : DATA "X^2 "
18  IF B < 0 GOTO 49
19 B =  SQR (B): RETURN : DATA  "SQRT "
20 B =  SIN (B): RETURN : DATA "SIN "
21  IF B -  INT (B / A) * A < .001 GOTO 49
22 B =  TAN (B): RETURN : DATA "TAN "
23 B =  COS (B): RETURN : DATA "COS "
24  IF B = 0 GOTO 49
25 B = 1 / B: RETURN : DATA  "1/X "
26  IF B > 88 GOTO 49
27 B =  EXP (B): RETURN : DATA "E^X "
28  IF B <  = 0 GOTO 49
29 B =  LOG (B): RETURN : DATA  "LN "
30 G = B:B = C:C = D:D = F:F = G: RETURN : DATA "RDN "
31 G = F:F = D:D = C:C = B:B = G: RETURN : DATA "R^ "
32  IF B = 0 GOTO 49
33 B = C -  INT (C / B) * B:C = D:D = F:F = 0: RETURN : DATA "MOD "
34 F = D:D = C:C = B: RETURN : DATA "ENTER^ "
35  IF C <  = 0 GOTO 49
36  IF  LOG (C) * B > 88 GOTO 49
37 B = C ^ B:C = D:D = F:F = 0: RETURN : DATA             "Y^X "
38 F = D:D = C:C = B:B = A(H): RETURN : DATA "ARGUMENT "
39 F = D:D = C:C = B:B = A(H): RETURN : DATA "ARGUMENT "
40 F = D:D = C:C = B:B = A(H): RETURN : DATA "ARGUMENT "
41 F = D:D = C:C = B:B = A(H): RETURN : DATA "ARGUMENT "
42 F = D:D = C:C = B:B = 1: RETURN : DATA "1 "
43 F = D:D = C:C = B:B = 1: RETURN : DATA "1 "
44 F = D:D = C:C = B:B = 2: RETURN : DATA "2 "
45 F = D:D = C:C = B:B = 2: RETURN : DATA "2 "
46 F = D:D = C:C = B:B = 3: RETURN : DATA "3 "
47 F = D:D = C:C = B:B = 3: RETURN : DATA "3 "
48 J = J + 1: VTAB 21: PRINT : PRINT "MISSERFOLG: "J: FOR I = 1 TO K:B(I) = C(I): NEXT : GOTO 51
49  POP : GOTO 48
50  DIM D(50),A(50),A$(50),B(50),C(50):L = 37:M = 9E30: FOR I = 1 TO L: READ A$(I): NEXT :A$(0) = "": INPUT "CODE-LAENGE ? ";K: INPUT
    "MUTATIVITAET ? ";N: INPUT "FEHLERSCHRANKE ? ";Q: INPUT "WERTETABELLE. WIEVIELE WERTEPAARE ? ";P: FOR I = 1 TO P: PRINT I". ";: INPUT
    "X,Y ? ";A(I),D(I): NEXT : HOME : PRINT "FORMEL-EVOLUTION"
51 Q = N *  RND (7) + 1: FOR I = 1 TO Q:R =  INT (K *  RND (13) + 1):S =  INT ((L + 1) *  RND (17)): IF B(R) = S GOTO 51
52 B(R) = S: NEXT : PRINT :T = 0: FOR H = 1 TO P:B = 0:C = 0:D = 0:F = 0: FOR I = 1 TO K: IF B(I) = 0 GOTO 55
53  ON B(I) GOSUB 2,3,4,5,6,7,9,10,11,12,13,14,15,16,17,18,20,21,23,24,26,28,30,31,32,34,35,38,39,40,41,42,43,44,45,46,47
54  IF B > M * P GOTO 48
55  NEXT I:T = T +  ABS (D(H) - B): IF T > M GOTO 48
```

```
56  NEXT H: IF T > M GOTO 48
57 M = T:E = E + 1: VTAB 10: PRINT : PRINT "ERFOLG: "E: PRINT "FEHLER: ";: INVERSE : PRINT T;: NORMAL : CALL  - 868: PRINT : PRINT : FOR
    I = 1 TO K:C(I) = B(I): PRINT A$(B(I));: NEXT : CALL  - 958: IF T > 0 GOTO 51
58  PRINT : HOME : IF T <  > 0 THEN  FOR I = 1 TO K:B(I) = C(I): NEXT :T = M
59  PRINT "FORMELEVOLUTION": PRINT "===============": PRINT : PRINT "WERTETABELLE:": PRINT "X","Y": FOR I = 1 TO P: PRINT A(I),D(I): NEXT
    : PRINT : PRINT "CODELAENGE: "K: PRINT "MUTATIVITAET: "N: PRINT "FEHLERSCHRANKE: "O: PRINT : PRINT : PRINT "   "J + E" VERSUCHE":
     PRINT J" MISSERFOLGE": PRINT E" ERFOLGE": PRINT "FEHLER: "T: PRINT "DAS IST DIE FORMEL!  :": PRINT
60 B$ = "0":C$ = "0":D$ = "0":E$ = "0": FOR I = 1 TO K: IF A$(B(I)) = "+ " THEN B$ = C$ + "+" + B$:C$ = D$:D$ = E$:E$ = "0"
61  IF A$(B(I)) = "* " THEN B$ = "(" + C$ + ")*(" + B$ + ")":C$ = D$:D$ = E$:E$ = "0"
62  IF A$(B(I)) = "/ " THEN B$ = "(" + C$ + ")/(" + B$ + ")":C$ = D$:D$ = E$:E$ = "0"
63  IF A$(B(I)) = "- " THEN B$ = C$ + "-(" + B$ + ")":C$ = D$:D$ = E$:E$ = "0"
64  IF A$(B(I)) = "ARCTAN " THEN B$ = "ATN(" + B$ + ")"
65  IF A$(B(I)) = "ABS " THEN B$ = "ABS(" + B$ + ")"
66  IF A$(B(I)) = "INT " THEN B$ = "INT(" + B$ + ")"
67  IF A$(B(I)) = "SIGN " THEN B$ = "SGN(" + B$ + ")"
68  IF A$(B(I)) = "SQRT " THEN B$ = "SQR(" + B$ + ")"
69  IF A$(B(I)) = "SIN " THEN B$ = "SIN(" + B$ + ")"
70  IF A$(B(I)) = "COS " THEN B$ = "COS(" + B$ + ")"
71  IF A$(B(I)) = "TAN " THEN B$ = "TAN(" + B$ + ")"
72  IF A$(B(I)) = "E^X " THEN B$ = "EXP(" + B$ + ")"
73  IF A$(B(I)) = "LN " THEN B$ = "LOG(" + B$ + ")"
74  IF A$(B(I)) = "FRC " THEN B$ = B$ + "-INT(" + B$ + ")"
75  IF A$(B(I)) = "CHS " THEN B$ = "-(" + B$ + ")"
76  IF A$(B(I)) = "X^2 " THEN B$ = "(" + B$ + ")^2"
77  IF A$(B(I)) = "1/X " THEN B$ = "1/(" + B$ + ")"
78  IF A$(B(I)) = "Y^X " THEN B$ = "(" + C$ + ")^(" + B$ + ")":C$ = D$:D$ = E$:E$ = "0"
79  IF A$(B(I)) = "ENTER^ " THEN E$ = D$:D$ = C$:C$ = B$
80  IF A$(B(I)) = "1 " THEN E$ = D$:D$ = C$:C$ = B$:B$ = "1"
81  IF A$(B(I)) = "2 " THEN E$ = D$:D$ = C$:C$ = B$:B$ = "2"
82  IF A$(B(I)) = "3 " THEN E$ = D$:D$ = C$:C$ = B$:B$ = "3"
83  IF A$(B(I)) = "ARGUMENT " THEN E$ = D$:D$ = C$:C$ = B$:B$ = "X"
84  IF A$(B(I)) = "RDN " THEN F$ = B$:B$ = C$:C$ = D$:D$ = E$:E$ = F$
85  IF A$(B(I)) = "R^ " THEN F$ = E$:E$ = D$:D$ = C$:C$ = B$:B$ = F$
86  IF A$(B(I)) = "MOD " THEN B$ = C$ + "-INT(" + C$ + "/" + B$ + ")*(" + B$ + ")":C$ = D$:D$ = E$:E$ = "0"
87  NEXT : PRINT "Y = " + B$: PRINT : END
```

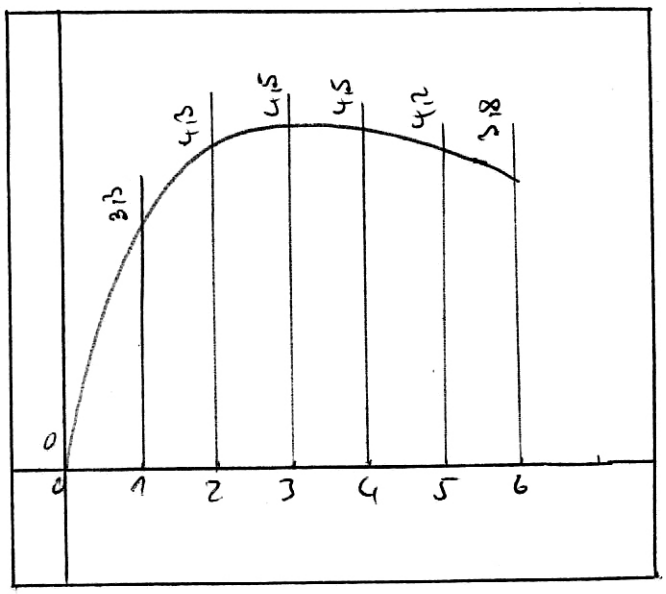
A N H A N G

Im Folgenden sind einige Läufe des Programms angefügt.

Es handelt sich dabei einmal um zwei sehr unterschiedliche Lösungen zur Darstellung einer handgezeichneten Kurve. Die Formel ist in diesen Fällen in der Postfix-Notation ausgedruckt. Zur Veranschaulichung ist der Graph der jeweiligen Formel angefügt.

Zum zweiten ist der Versuch, eine Formel für die Folge der Primzahlen zu finden gezeigt.

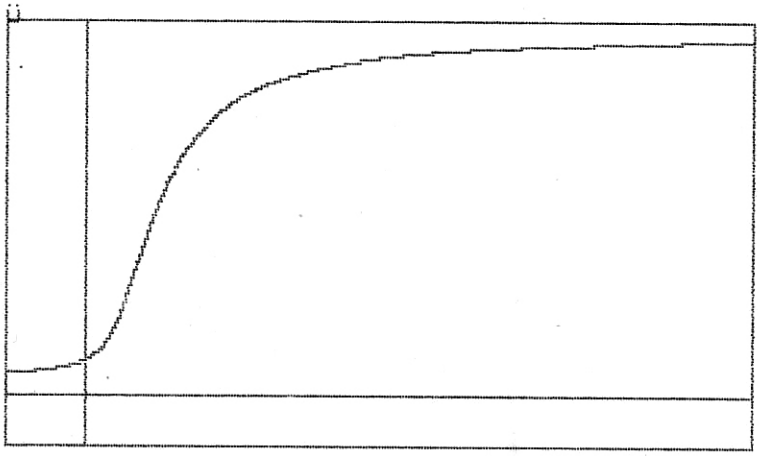
Und den Schluß bilden 16 äquivalente Formeln für die einfache Wertetabelle der Funktion y=2*x.



FORMELEVOLUTION
==========

WERTETABELLE:

X	Y
0	0
1	3.3
2	4.3
3	4.5
4	4.5
5	4.2
6	3.8



CODELAENGE: 20
MUTATIVITAET: 5
FEHLERSCHRANKE: 0
FEHLER: 1.0777589

2518 VERSUCHE
2061 FEHLER
457 ERFOLGE

DAS IST DIE FORMEL! :

CHS CHS + ENTER^ SIGN R^ E^X ENTER^ ARGUMENT 1 + ARGUMENT ARGUMENT COS - ARGUMENT + + ARCTAN E^X

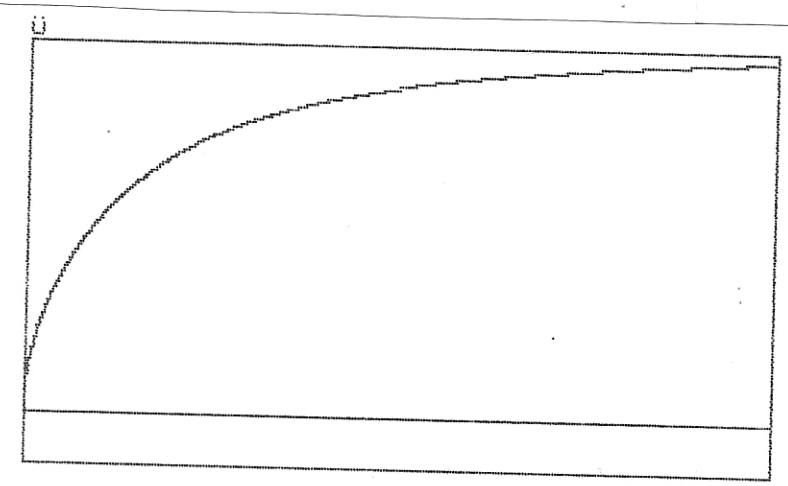
```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
0                 0
1                 3.3
2                 4.3
3                 4.5
4                 4.5
5                 4.2
6                 3.8

CODELAENGE: 20
MUTATIVITAET: 7
FEHLERSCHRANKE: 0


   22712 VERSUCHE
21762 MISSERFOLGE
950 ERFOLGE
FEHLER: 1.70678619
DAS IST DIE FORMEL!  :

1 3 / - COS 1 + 1 ABS ARGUMENT ARCTAN SQRT RDN X^2 / - CHS X^2 * +
```



```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 3
3                 5
4                 7
5                 11
6                 13
7                 17
8                 19
9                 23

CODELAENGE: 15
MUTATIVITAET: 5
FEHLERSCHRANKE: 3


   16833 VERSUCHE
16209 MISSERFOLGE
624 ERFOLGE
FEHLER: 4.67935588
DAS IST DIE FORMEL!  :

Y = EXP(COS(ABS(SGN(X))-(X-INT(X/2)*(2))))+(LOG(X))*(X)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 4
FEHLERSCHRANKE: 0


   226 VERSUCHE
157 MISSERFOLGE
69 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = 0+0-(0-INT(0))+X+X
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 6
FEHLERSCHRANKE: 0


   174 VERSUCHE
125 MISSERFOLGE
49 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = X-(-(X))
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 2
FEHLERSCHRANKE: 0


   202 VERSUCHE
148 MISSERFOLGE
54 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = X-(X-INT(X)-(X))
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 7
FEHLERSCHRANKE: 0


   18 VERSUCHE
13 MISSERFOLGE
5 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = X+-(-(X))
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 5
FEHLERSCHRANKE: 0


   85 VERSUCHE
46 MISSERFOLGE
39 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = (0)*(EXP(0))+INT((2)*(X))
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 2
FEHLERSCHRANKE: 0


   185 VERSUCHE
95 MISSERFOLGE
90 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = (ABS(2))*(X)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 3
FEHLERSCHRANKE: 0


   45 VERSUCHE
31 MISSERFOLGE
14 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = ABS(X)+X
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 2
FEHLERSCHRANKE: 0


   77 VERSUCHE
43 MISSERFOLGE
34 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = (X+ABS(X))*(1)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 3
FEHLERSCHRANKE: 0


   452 VERSUCHE
270 MISSERFOLGE
182 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!  :

Y = X+X
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 2
FEHLERSCHRANKE: 0


   104 VERSUCHE
45 MISSERFOLGE
59 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!   :

Y = ABS(X+X)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 1
FEHLERSCHRANKE: 0


   725 VERSUCHE
588 MISSERFOLGE
137 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!   :

Y = ABS(0-(X)-(X))
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 3
FEHLERSCHRANKE: 0


   340 VERSUCHE
322 MISSERFOLGE
18 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!   :

Y = (X+X)^(1)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 3
FEHLERSCHRANKE: 0


   150 VERSUCHE
103 MISSERFOLGE
47 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!   :

Y = X+INT(X)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6
4                 8

CODELAENGE: 7
MUTATIVITAET: 6
FEHLERSCHRANKE: 0


   45 VERSUCHE
41 MISSERFOLGE
4 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!   :

Y = (INT(X))*(2)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 3
FEHLERSCHRANKE: 0


   33 VERSUCHE
16 MISSERFOLGE
17 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!   :

Y = (X)*(2)
```

```
FORMELEVOLUTION
===============

WERTETABELLE:
X                 Y
1                 2
2                 4
3                 6

CODELAENGE: 7
MUTATIVITAET: 3
FEHLERSCHRANKE: 0


   190 VERSUCHE
97 MISSERFOLGE
93 ERFOLGE
FEHLER: 0
DAS IST DIE FORMEL!   :

Y = X+X
```