Jürgen Schwarz

Behandlung von Polynomen – Teil 4 *

Im ersten Teil dieser Serie wurden SUB-Programme für die rationalen Operationen mit Polynomen abgeleitet, der zweite Teil beinhaltete die Nullstellenbestimmung von Polynomen und im dritten Teil wurden Algorithmen zur Berechnung des größten gemeinsamen Teilers von zwei Polynomen angegeben. Dazu kamen einige Bemerkungen zur Datensicherung. Während in den ersten drei Teilen dieser Serie einige Grundlagen zur Bearbeitung von Polynomen und zur Bestimmung von ihren Eigenschaften abgeleitet wurden, sollen nun anwendungsorientierte Verfahren behandelt werden.

Stichwörter:

Polynom, HP-BASIC, Laplace-Transformation, Approximation, Ausgleichsrechnung

Hardware:

HP 9845 B mit SP ROM

8 Das HORNER'sche Schema

8.1 Das einfache HORNER'sche Schema

Das naive Berechnen des Wertes eines Polynoms n-ten Grades

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0;\ a_n \neq 0;\ a_i, x \in \mathbb{R} \qquad (1)$$

an der Stelle $x = a$ erfordert $n - 1$ Multiplikationen zur Berechnung der Potenzen von x, weitere n Multiplikationen zur Berechnung der Produkte $a_i x^i$ und schließlich n Additionen zur Bestimmung der Summe von f(x), insgesamt also $3n - 1$ relevante Operationen**.

Wird Gl. (1) in die Form

$$f(x) = a_0 + x(a_1 + x(a_2 + \ldots + x(a_{n-1} + x\,a_n))))\ldots))); a_n \neq 0; a_i, x \in \mathbb{R} \qquad (54)$$

gebracht, dann sind zur Berechnung des Wertes des Polynoms nur noch n Multiplikationen und n Additionen erforderlich, also nur noch $2n$ relevante Operationen. Ein weiterer Vorteil dieser Vorgehensweise, die von W.G. HORNER im Jahre 1819 in seinem bekannten Rechenschema niedergelegt wurde, liegt darin, daß sie numerisch wesentlich stabiler als die formale Rechnung nach Gl. (1) ist.

Ein Beispiel möge das verdeutlichen. Es soll der Wert des Polynoms $f(x) = 3x^4 - 2x^2 + 2x - 5$ an der Stelle $x = 0{,}001$ berechnet werden. Bei Anwendung der Gl. (1) erhält man

$$\begin{aligned} f(x) &= 3 \cdot 0{,}000\,000\,000\,001 \\ &\quad - 2 \cdot 0{,}000\,001 + 2 \cdot 0{,}001 + 5 \\ &= 5{,}001\,998\,000\,003, \end{aligned}$$

also Zahlen sehr unterschiedlicher Größenordnung. Bei Anwendung der Gl. (54) werden die Verhältnisse besser

$$\begin{aligned} f(x) &= 5 + 0{,}001(2 + 0{,}001 \cdot \\ &\quad (-2 + 0{,}001(0 + 0{,}001 \cdot 3))) \\ &= 5 + 0{,}001(2 + 0{,}001 \cdot \\ &\quad (-2 + 0{,}000\,003)) \\ &= 5 + 0{,}001(2 - 0{,}001\,999\,997) \\ &= 5 + 0{,}001\,998\,000\,003 \\ &= 5{,}001\,998\,000\,003, \end{aligned}$$

weil sich hier die entstehenden Summanden nicht soweit unterscheiden wie oben.

Listing 11 zeigt ein Funktionsunterprogramm, welches nach Gl. (54) arbeitet. In den Zeilen 210 und 230 wird der Wert y rekursiv, von der höchsten Potenz ausgehend, berechnet. Dieser Wert y wird dann in Zeile 250 dem Funktionsnamen zugeordnet.

Ein Anwendungsbeispiel für dieses Programm ist in Listing 12 wiedergegeben. Hier soll kontinuierlich die Temperatur an einer Meßstelle gemessen und ausgedruckt werden. Da Thermoelemente nur in einem kleinen Temperaturbereich annähernd linear arbeiten [9], kann man eine Kompensation dieser Nichtlinearität mit einem Polynom neunten Grades vornehmen [10]. Will man die absolute Temperatur und keine Temperaturdifferenz messen, ist es i. allg. einfacher, die Temperatur der inneren Vergleichsstelle (der Anschlußstelle im Meßgerät) zu bestimmen, als ein zweites, in Reihe geschaltetes, Thermoelement mit einer definierten Temperatur (z.B. Eistopf) zu beaufschlagen. Die meisten Temperaturmeßgeräte, so auch der hier verwendete Datenlogger, sind für eine solche Messung direkt vorgesehen.

Am Kanal 19 liegt eine Spannung an, die mit zehn multipliziert, die Temperatur der Vergleichsstelle in Grad Celsius liefert. Mit einem Polynom zweiten Grades wird aus dieser Temperatur die korrespondierende Vergleichsspannung eines Thermoelementes passenden Typs (hier Typ J: Eisen-Konstantan) berechnet. Diese wird mit der gemessenen Spannung addiert und die sich ergebende Summe wird als Argument des Kompensationspolynoms neunten Grades verwendet. Der Wert des Polynoms ist gerade die gewünschte Temperatur in Grad Celsius. Das gesamte Verfahren nennt sich Softwarekompensation.

Jürgen Schwarz, Wiesbadener Straße 59 C, 1000 Berlin 33.

*Teil 3: s. CAL – Comp. Anw. Lab. **3** (1985) 136.

**Bei Rechnern mit Arithmetikprozessoren unterscheidet sich der Zeitaufwand zur Durchführung von Additionen und Multiplikationen nicht allzu sehr, so daß es hier gerechtfertigt ist, alle Operationen als relevant anzusehen.
(HP 9845 B mit „fast language processor“: Zeitaufwand je Gleitkommaaddition ≈ 120 μs, je Gleitkommamultiplikation ≈ 190 μs).

Listing 11: Funktions-Unterprogramm zur Berechnung des Wertes eines Polynoms f(x) an der Stelle a.

```
10   DEF FNHorner(INTEGER N,REAL A,A(*))
20   !
30   ! Funktions-Unterprogramm zur Berechnung des Wertes eines Polynoms
40   ! y = f(x) mit reellen Koeffizienten an der Stelle x=a mit Hilfe
50   ! des einfachen Hornerschen Schemas
60   !
70   ! Eingabedaten: n        ... Grad          des Polynoms f(x)
80   !               a        ... Stelle, an der der Wert des Polynoms
90   !                            berechnet wird
100  !               a(0:n)   ... Koeffizienten des Polynoms f(x)
110  ! Ergebnis:     horner   ... y = f(x=a)
120  !
130  ! (c) 1985 by Jürgen Schwarz          Sprache: HP-BASIC    Datum: 18.05.85
140  ! Speichermedium: Kassetten 57/58     File-Name: Horner    Version:    1.0
150  !
160    INTEGER I
170    REAL Y
180    !
190    IF N<0 THEN PAUSE                                   ! widersprüchliche Daten
200    REDIM A(0:N)
210    Y=A(N)
220    FOR I=N-1 TO 0 STEP -1
230      Y=Y*A+A(I)
240    NEXT I
250    RETURN Y
260  FNEND
```

Listing 12: Anwendungsbeispiel für das Funktionsunterprogramm FNHorner nach Listing 11.

```
10   ! Demonstrations-Programm Theta zur Steuerung des Datenloggers HP 3497 A
20   ! mit dem Rechner HP 9845 B zur Messung und zum laufendem Ausdruck der
30   ! Temperatur in einem Kanal
40   !
50   ! Konfiguration des HP 3497 A: CH 000 <==> Thermoelement Typ J
60   !                              CH 019 <==> interne Vergleichsstelle
70   !
80   ! (c) 1985 by Jürgen Schwarz          Sprache: HP-BASIC    Datum: 01.06.85
90   ! Speichermedium: Kassetten 57/58     File-Name:  Theta    Version:    1.0
100  !
110  INTEGER N_r_sc,N_p_sc
120  REAL T_vgl,Theta,U_thermo,U_theta,U_vergleich,U_vgl
130  REAL R_sc(0:2),P_sc(0:9)
140  !
150  FIXED 2
160 Daten:   ! Daten zur Umrechnung der Spannungen in Temperaturen
170          ! (Koeffizienten der Polynome zur Softwarekompensation)
180  RESTORE Daten
190  DATA  2, 9                                             ! Polynomgrade
200  DATA  -.7500434400E-6,   50.5321995E-6,   23.48050017E-9  ! Polynom R_sc(*)
210  DATA     -.3595568424,      19750.87948,     -175116.5425
220  DATA      18212965.58,     -2831128435,      271508383300
230  DATA  -13.8014121E12, 379.24384326E12, -5.371925517E15  ! Polynom P_sc(*)
240  DATA  30.840255439E15
250  READ N_r_sc,N_p_sc
260  MAT READ R_sc(0:N_r_sc),P_sc(0:N_p_sc)
270  !
280  ON KEY #0 GOTO Ende
290  DISP "KEY:"&CHR$(129)&"| Beenden |"&RPT$("          |",7)&CHR$(128)
300  CLEAR 709
310  LOOP
320    OUTPUT 709;"AC19"          ! Einstellung auf Kanal 019
330    ENTER 709;U_vergleich      ! Spannung des Elementes der Vergleichsstelle
340    OUTPUT 709;"AC0"           ! Einstellung auf Kanal 000
350    ENTER 709;U_thermo         ! gemessene Spannung des Thermoelementes
360    !
370    T_vgl=10*U_vergleich       ! Temperatur der Vergleichsstelle [°C]
380    U_vgl=FNHorner(N_r_sc,T_vgl,R_sc(*))    ! korrespondierende Vergleichspg.
390    U_theta=U_thermo+U_vgl     ! berechnete Spannung des Thermoelementes bei
400                               ! einer Vergleichsstellentemperatur von 0 °C
410    Theta=FNHorner(N_p_sc,U_theta,P_sc(*))    ! Temperatur der Meßstelle [°C]
420    PRINT "theta = ";PROUND(Theta,-2);" "&CHR$(179)&"C"
430  END LOOP
440 Ende: !
450    DISP
460    OFF KEY #0
470  END
```

In den DATA-Zeilen 190 bis 240 sind die Polynomgrade und die Koeffizienten der Polynome gespeichert und in den folgenden Zeilen werden diese den Variablen zugeordnet. Das Programm arbeitet kontinuierlich in der LOOP-Schleife und ein Abbruch erfolgt durch das Drücken der Taste #0. Man sieht hier deutlich die Verwendung des Funktionsunterprogramms FNHorner, welches wie eine Standardfunktion verwendet wird. Die Zeilen 370 bis 410 hätten ohne weiteres zu einer Zeile

Theta = FNHorner(N_p_sc, U_thermo + FNHorner(N_r_sc,10 * U_vergleich,R_sc(*)),P_sc(*))

zusammengefaßt werden können, was hier aus Gründen der Anschaulichkeit unterblieben ist.

8.2 Das vollständige HORNER'sche Schema

Betrachtet man das Rechenverfahren des vorhergehenden Abschnittes näher, insbesondere die dabei auftretenden Zwischenergebnisse, so folgen daraus weitere Anwendungsmöglichkeiten des HORNER'schen Schemas. An einem Polynom vierten Grades soll dies verdeutlicht werden. Zuerst wird das einfache HORNER'sche Schema notiert. Es soll der Wert des Polynoms $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$ an der Stelle $x = a$ berechnet werden:

$$\begin{array}{r|ccccc} & a_4 & a_3 & a_2 & a_1 & a_0 \\ x = a: & - & a\,a_4' & a\,a_3' & a\,a_2' & a\,a_1' \\ \hline & a_4' & a_3' & a_2' & a_1' & a_0' = A_0 \,. \end{array}$$

Dabei werden die gestrichenen Koeffizienten a_i' durch Addition der untereinanderstehenden Zahlen gebildet

$$a_i' = a_i + a\,a_{i+1}' ,$$

speziell mit $a_{n+1}' = 0$, d. h. $a_n' = a_n$. (55)

Die gestrichenen Koeffizienten haben aber gerade den Wert der Koeffizienten des Polynoms, das sich aus der Division des Ausgangspolynoms durch $(x - a)$ ergibt:

$$\begin{array}{l} (a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0) : (x - a) = \\ a_4x^3 + a_3'x^2 + a_2'x + a_1' + \dfrac{A_0}{x-a} \\ \underline{-(a_4x^4 - a\,a_4x^3)} \\ \qquad a_3'x^3 + a_2x^2 \\ \qquad \underline{-(a_3'x^3 - a\,a_3'x^2)} \\ \qquad\qquad a_2'x^2 + a_1x \\ \qquad\qquad \underline{-(a_2'x^2 - a\,a_2'x)} \\ \qquad\qquad\qquad a_1'x + a_0 \\ \qquad\qquad\qquad \underline{-(a_1'x - a\,a_1')} \\ \qquad\qquad\qquad\qquad a_0' = A_0 \end{array}$$

Bezeichnet man das Ergebnispolynom mit $f_1(x)$, dann gilt offensichtlich

$$f(x) = f_1(x)\,(x - a) + A_0 \tag{56}$$

und daraus folgt, daß der Divisionsrest A_0 zugleich der Wert des Polynoms $f(x)$ an der Stelle a ist. Ist a aber eine Nullstelle des Polynoms, dann ist A_0 null und das Polynom $f_1(x)$ ist ein Deflationspolynom von $f(x)$. Diese Tatsache wurde in dem Nullstellenberechnungsprogramm Newton (Listing 5 im Teil 2 dieser Serie) zur Bereitstellung der Koeffizienten des Deflationspolynoms benutzt.

Wird nun das HORNER'sche Schema mit den Koeffizienten a_i' weitergeführt, d.h. der Wert des Polynoms $f_1(x)$ ebenfalls an der Stelle $x = a$ berechnet, dann ergibt sich ein Polynom $f_2(x)$ und ein Rest A_1. Auch dieses Polynom wird an der Stelle $x = a$ berechnet und diese Reihe wird fortgesetzt, bis nur noch ein Koeffizient übrigbleibt:

$$\begin{aligned} f(x) &= f_1(x)\,(x-a) + A_0 \\ f_1(x) &= f_2(x)\,(x-a) + A_1 \\ f_2(x) &= f_3(x)\,(x-a) + A_2 \\ &\;\;\vdots \\ f_{n-1}(x) &= f_n(x)\,(x-a) + A_{n-1} \\ f_n\;(x) &= \qquad\qquad\quad A_n \,. \end{aligned} \tag{57}$$

Multipliziert man jetzt die letzte Gleichung mit $(x - a)^n$

$$(x-a)^n f_n(x) = A_n\,(x-a)^n \tag{58}$$

und setzt dies in die mit $(x-a)^{n-1}$ multiplizierte vorletzte Gleichung ein

$$(x-a)^{n-1} f_{n-1}(x) = A_n\,(x-a)^n + (x-a)^{n-1} A_{n-1} , \tag{59}$$

und führt diese Entwicklung bis zur ersten Gleichung fort, dann ergibt sich der Ausdruck

$$f(x) = A_n\,(x-a)^n + A_{n-1}\,(x-a)^{n-1} + \ldots + A_1\,(x-a) + A_0 \,. \tag{60}$$

Man hat also ein Polynom mit einer transformierten Variablen $s = x - a$

$$g(s) = f(x) = A_n s^n + A_{n-1} s^{n-1} + \ldots + A_1 s + A_0 \tag{61}$$

erhalten.

Im Listing 13 ist ein SUB-Programm dargestellt, das diese Transformation nach den abgeleiteten Regeln durchführt. Man sieht die einfache Programmgestaltung mit einer doppelten Laufanweisung. Müssen die Ausgangsdaten nicht geschützt werden, dann kann die Berechnung der einzelnen Koeffizienten direkt im Ausgangsfeld $A(*)$ erfolgen. Beim ersten Durchlauf der J-Schleife wird A_0 berechnet, beim zweiten A_1, usw.

Die Gl. (60) gestattet noch eine weitere Aussage. Entwickelt man eine Funktion (hier ein Polynom) um $x = a$ in einer TAYLOR'schen Reihe, dann gilt allgemein

$$f(x) = f(a) + \frac{1}{1!} f'(a)\,(x-a) + \frac{1}{2!} f''(a)\,(x-a)^2 + \ldots . \tag{62}$$

Bei einem Polynom verschwinden alle i-ten Ableitungen, für die i größer als der Grad des Polynomes ist, identisch. Darum bricht hier die Reihe (62) bei dem Glied

$$+ \frac{1}{n!} f^{(n)}(a)\,(x-a)^n$$

ab. Ein Koeffizientenvergleich mit Gl. (60) liefert dann die Beziehungen

$$A_i = \frac{1}{i!} f^{(i)}(a) \,. \tag{63}$$

Die Reste A_i der Polynomdivisionen im vollständigen HORNER'schen Schema sind also die durch $i!$ dividierten i-ten Ableitungen des Polynoms an der Stelle $x = a$. Dieser Umstand wurde in dem oben erwähnten SUB-Programm Newton zur Berechnung der ersten Ableitung des Polynoms verwendet.

8.3 Das doppelzeilige HORNER'sche Schema

Hat ein Polynom mit reellen Koeffizienten konjugiert komplexe Nullstellen und soll dann das Deflationspolynom bei einer solchen komplexen Nullstelle berechnet werden, dann ergibt sich ein Polynom mit komplexen Koeffizienten. Dies kann man vermeiden, indem man gleich durch das Produkt der konjugiert komplexen Nullstellen dividiert, da

$$(x - a - ib)\,(x - a + ib) = (x^2 - 2ax + a^2 + b^2) = (x^2 + px + q) , \tag{64}$$

also ein quadratisches Polynom mit den reellen Koeffizienten $1, p = -2a$ und $q = a^2 + b^2$ ist. Dazu bietet sich das doppelzeilige HORNER'sche Schema an:

$$\begin{array}{l|cccccccc} & a_n & a_{n-1} & a_{n-2} & a_{n-3} & \ldots & a_2 & a_1 & a_0 \\ q: & - & - & q\,a_n' & q\,a_{n-1}' & \ldots & q\,a_4' & q\,a_3' & q\,a_2' \\ p: & - & p\,a_n' & p\,a_{n-1}' & p\,a_{n-2}' & \ldots & p\,a_3' & p\,a_2' & - \\ \hline & a_n' & a_{n-1}' & a_{n-2}' & a_{n-3}' & \ldots & a_2' & a_1' & a_0' \,. \end{array}$$

Listing 13: SUB-Programm zur Achsenverschiebung (Koordinatentransformation) eines Polynoms um a.

```
10    SUB Horner_taylor(INTEGER N,REAL A,A(*),B(*))
20    !
30    ! SUB-Programm zur Transformation der Koeffizienten eines Polynoms
40    ! y = f(x) mit reellen Koeffizienten durch die Substitution s = x-a
50    ! in das Polynom y = g(s) mit Hilfe des vollständigen Hornerschen Schemas
60    ! (Entwicklung des Polynoms um a nach Taylor)
70    !
80    ! Eingabedaten: n          ... Grad              des Polynoms f(x)
90    !               a          ... Transformationsparameter
100   !               a(0:n)     ... Koeffizienten des Polynoms f(x)
110   ! Ergebnis:     b(0:n)     ... Koeffizienten des Polynoms g(s = x-a)
120   !
130   ! (c) 1985 by Jürgen Schwarz          Sprache: HP-BASIC     Datum: 18.05.85
140   ! Speichermedium: Kassetten 57/58     File-Name: Horner     Version:    1.0
150   !
160     INTEGER I,J
170     REAL A_stern(0:N)
180     !
190     IF N<0 THEN PAUSE                                   ! widersprüchliche Daten
200     REDIM A(0:N)
210     MAT A_stern=A
220     MAT B=ZER
230     MAT B=A_stern
240     !
250     FOR J=0 TO N-1
260       FOR I=N-1 TO J STEP -1
270         B(I)=B(I+1)*A+B(I)
280       NEXT I
290     NEXT J
300   SUBEND
```

Listing 14: SUB-Programm zur Division eines Polynoms durch ein quadratisches Polynom.

```
10    SUB Doppel_horner(INTEGER N,REAL P,Q,A(*),B(*),C(*))
20    !
30    ! SUB-Programm zur Division eines Polynoms durch ein quadratisches
40    ! Polynoms und Berechnung des Restes mit Hilfe des doppelzeiligen
50    ! Hornerschen Schemas:
60    !
70    !          f (x) = f    (x) * (x^2+px+q) + (c(1)x+c(0))
80    !           n       n-2
90    !
100   ! Eingabedaten: n          ... Grad              des Polynoms f(x)
110   !               p          ... Parameter des Divisionspolynoms
120   !               q          ... Parameter des Divisionspolynoms
130   !               a(0:n)     ... Koeffizienten des Polynoms f (x)
140   !                                                             n
150   ! Ergebnis:     b(0:n)     ... Koeffizienten des Polynoms f    (x)
160   !                                                             n-2
170   !               c(0:1)     ... Koeffizienten des Rest-Polynoms c(1)*x+c(0)
180   !
190   ! (c) 1985 by Jürgen Schwarz          Sprache: HP-BASIC     Datum: 18.05.85
200   ! Speichermedium: Kassetten 57/58     File-Name: Horner     Version:    1.0
210   !
220     INTEGER I
230     REAL A_stern(0:N)
240     !
250     IF N<0 THEN PAUSE                                   ! widersprüchliche Daten
260     REDIM A(0:N)
270     MAT A_stern=A
280     MAT B=ZER
290     MAT C=ZER
300     REDIM B(0:N-2),C(0:1)
310     !
320     B(N-2)=A_stern(N)
330     B(N-3)=A_stern(N-1)-B(N-2)*P
340     FOR I=N-4 TO 0 STEP -1
350       B(I)=A_stern(I+2)-B(I+1)*P-B(I+2)*Q
360     NEXT I
370     C(1)=A(1)-B(0)*P-B(1)*Q
380     C(0)=A(0)-B(0)*Q
390   SUBEND
```

Listing 15: SUB-Programm zur Berechnung des Wertes eines Polynoms für ein komplexes Argument durch Division durch das Produkt des Arguments mit seinem konjugierten Wert.

```
10    SUB Komplex_horner(INTEGER N,REAL X,Y,A(*),U,V)
20    !
30    ! SUB-Programm zur Berechnung des Wertes eines Polynoms mit reellen
40    ! Koeffizienten für ein komplexes Argument mit Hilfe des doppelzeiligen
50    ! Hornerschen Schemas:
60    !                         f(z) = u+iv = f(x+iy)
70    !
80    ! Eingabedaten: n       ... Grad          des Polynoms f(z)
90    !               a(0:n)  ... Koeffizienten des Polynoms f(z)
100   !               x       ... Realteil      des Arguments
110   !               y       ... Imaginärteil  des Arguments
120   ! Ergebnis:     u       ... Realteil      des Ergebnisses
130   !               v       ... Imaginärteil  des Ergebnisses
140   !
150   ! (c) 1985 by Jürgen Schwarz         Sprache: HP-BASIC    Datum: 18.05.85
160   ! Speichermedium: Kassetten 57/58    File-Name: Horner    Version:    1.0
170   !
180     INTEGER I
190     REAL P,Q,A_1,A_2,A_3
200     !
210     IF N<0 THEN PAUSE                               ! widersprüchliche Daten
220     REDIM A(0:N)
230     !
240     P=-2*X
250     Q=X*X+Y*Y
260     A_1=A(N)
270     A_2=A(N-1)-A(N)*P
280     FOR I=N-2 TO 1 STEP -1
290       A_3=A(I)-A_2*P-A_1*Q
300       A_1=A_2
310       A_2=A_3
320     NEXT I
330     A_3=A(0)-A_1*Q
340     U=A_2*X+A_3
350     V=A_2*Y
360   SUBEND
```

Die gestrichenen Koeffizienten werden also mit den Gleichungen

$$\begin{aligned}
a'_n &= a_n \\
a'_{n-1} &= a_{n-1} && - p\,a'_n \\
a'_{n-2} &= a_{n-2} - q\,a'_n && - p\,a'_{n-1} \\
a'_{n-3} &= a_{n-3} - q\,a'_{n-1} && - p\,a'_{n-2} \\
&\vdots \\
a'_1 &= a_1 - q\,a'_3 && - p\,a'_2 \\
a'_0 &= a_0 - q\,a'_2
\end{aligned} \qquad (65)$$

berechnet. Entsprechend den Ausführungen des vorhergehenden Abschnittes sind die gestrichenen Koeffizienten $a'_n \ldots a'_2$ Koeffizienten eines Polynoms mit einem Grad $n-2$ und $a'_1\,x$ und a'_0 ergeben den Rest

$$f(x) = f_2(x)\,(x^2 + px + q) + a'_1 x + a'_0 \qquad (66)$$

analog zu Gl. (56).

In Listing 14 ist ein entsprechendes SUB-Programm dargestellt, welches die Gln. (65) linear abarbeitet.

Das doppelzeilige HORNER'sche Schema läßt sich auch zur Berechnung des Wertes eines Polynoms mit reellen Koeffizienten für ein komplexes Argument $z_0 = x + iy$ verwenden. Zur Berechnung von $f(z_0) = u + iv$ wird die Gleichung

$$f(z) = f_2(z)\,(z - z_0)\,(z - \bar{z}_0) + a'_1 z + a'_0 \qquad (67)$$

verwendet. Für $z = z_0$ ist das Produkt mit $(z - z_0)$ gleich null und damit gilt

$$f(z_0) = a'_1 z_0 + a'_0. \qquad (68)$$

Das Einsetzen von $z_0 = x + iy$ liefert dann

$$f(z_0) = u + iv = (a'_1 x + a'_0) + i a'_1 y. \qquad (69)$$

Es ist also nur die mit den Gln. (65) beschriebene Division durch das Produkt des Arguments mit seinem konjugierten Wert durchzuführen und die Gl. (69) liefert dann die Lösung. Listing 15 zeigt ein entsprechendes SUB-Programm.

8.4 *Das HORNER'sche Schema für komplexe Polynome*

Zum Abschluß des Abschnittes 8 soll noch ein SUB-Programm zur Ermittlung des Funktionswertes eines Polynoms mit komplexen Koeffizienten bei komplexem Argument abgeleitet werden. Ein Verfahren dazu, nämlich über die Berechnung der ŠILJAK-Funktion, wurde im Abschnitt 5.2 vorgestellt. Eine weitere Möglichkeit besteht darin, das einfache HORNER'sche Schema nach Gl. (54) im Bereich der komplexen Zahlen auszuführen. Hier müßte eine Schleife mit den Zeilen

```
260     U=Koeff_r(N)
270     V=Koeff_i(N)
280     FOR I=N-1 TO 0 STEP -1
290       R=U
300       S=V
310       U=R*X-S*Y+Koeff_r(I)
320       V=R*Y+S*X+Koeff_i(I)
330     NEXT I
```

stehen, da für das Produkt zweier komplexer Zahlen $z_1 = x_1 + iy_1$ und $z_2 = x_2 + iy_2$ bekanntlich

$$z = u + iv = z_1 \cdot z_2 = x_1 x_2 - y_1 y_2 + i(x_1 y_2 + x_2 y_1) \qquad (70)$$

gilt. Hier soll aber ein in [11] beschriebenes Verfahren verwendet werden, welches zwar an dieser Stelle keine Vorteile bringt, aber bei längeren Rechnungen mit komplexen Zahlen vorteilhaft angewandt werden kann.

Listing 16: Berechnung des Wertes eines komplexen Polynoms mit Hilfe von MAT-Befehlen.

```
10    SUB Komplex_polynom(INTEGER N,REAL X,Y,Koeff_r(*),Koeff_i(*),U,V)
20    !
30    ! SUB-Programm Komplex_polynom zur Berechnung des Wertes eines Polynoms
40    ! mit komplexen Koeffizienten für ein komplexes Argument
50    !
60    ! Eingangsgrößen:
70    !          n           ... Grad des Polynoms
80    !          x           ... Realteil     des Argumentes
90    !          y           ... Imaginärteil des Argumentes
100   !          koeff_r(0:n) ... Realteil     der Koeffizienten des Polynoms
110   !          koeff_i(0:n) ... Imaginärteil der Koeffizienten des Polynoms
120   !
130   ! Ausgangsgrößen:
140   !          u           ... Realteil     des Wertes des Polynoms
150   !          v           ... Imaginärteil des Wertes des Polynoms
160   !
170   ! (c) 1985 by Jürgen Schwarz        Sprache: HP-BASIC     Datum: 01.06.85
180   ! Speichermedium: Kassetten 57/58   File-Name: Horner     Version:    1.1
190   !
200     INTEGER I
210     REAL Funktion(1:2,1:2),Argument(1:2,1:2)
220     REAL Koeffizient(1:2,1:2),Produkt(1:2,1:2)
230     !
240     IF N<0 THEN PAUSE                                    ! widersprüchliche Daten
250     REDIM Koeff_r(0:N),Koeff_i(0:N)
260     !
270     CALL Wertzuweisung(X,Y,Argument(*))
280     CALL Wertzuweisung(Koeff_r(N),Koeff_i(N),Funktion(*))
290     FOR I=N-1 TO 0 STEP -1
300       CALL Wertzuweisung(Koeff_r(I),Koeff_i(I),Koeffizient(*))
310       MAT Produkt=Funktion*Argument
320       MAT Funktion=Produkt+Koeffizient
330     NEXT I
340     U=Funktion(1,1)                 ! Realteil der Lösung
350     V=Funktion(2,1)                 ! Imaginärteil der Lösung
360     SUBEXIT
370     !
380     SUB Wertzuweisung(REAL X,Y,Z(*))
390     !
400     ! SUB-Programm zur Wertzuweisung an eine, eine komplexe Zahl
410     ! repräsentierende (2*2)-Matrix
420     !
430       Z(1,1)=Z(2,2)=X
440       Z(1,2)=-Y
450       Z(2,1)=Y
460     SUBEND
470   SUBEND
```

Sind in einer Sprache keine Operationen mit komplexen Zahlen, aber solche mit Matrizenoperationen unterstützt, so kann eine komplexe Zahl $z = x + iy$ durch eine (2×2)-Matrix

$$Z = \begin{pmatrix} x & -y \\ y & x \end{pmatrix} \Leftrightarrow z = x + iy \tag{71}$$

repräsentiert werden. Zur Addition (Subtraktion) werden einfach die Matrizen addiert (subtrahiert)

$$Z = Z_1 \pm Z_2 = \begin{pmatrix} x_1 \pm x_2 & -y_1 \mp y_2 \\ y_1 \pm y_2 & x_1 \pm x_2 \end{pmatrix}$$

$$\Leftrightarrow z = z_1 + z_2 = (x_1 \pm x_2) + i(y_1 \pm y_2). \tag{72}$$

Im Rechner wird dies durch den Befehl MAT $Z=Z\_1+Z\_2$ bzw. MAT $Z=Z\_1-Z\_2$ realisiert. Mit Hilfe der Matrizenmultiplikation kann das Produkt zweier komplexer Zahlen entsprechend Gl. (70) berechnet werden

$$Z = Z_1 \cdot Z_2 = \begin{pmatrix} x_1x_2 - y_1y_2 & -x_1y_2 - x_2y_1 \\ x_1y_2 + x_2y_1 & x_1x_2 - y_1y_2 \end{pmatrix}$$

$$\Leftrightarrow z = z_1 \cdot z_2 \tag{73}$$

Der entsprechende BASIC-Befehl lautet MAT $Z=Z\_1*Z\_2$. Auch zur Division kann diese Darstellungsweise verwendet werden. Hier gilt nach [12]

$$z = \frac{z_1}{z_2} = \frac{(x_1x_2 + y_1y_2) + i(x_2y_1 - x_1y_2)}{x_2^2 + y_2^2}. \tag{74}$$

Wird die die komplexe Zahl z_2 repräsentierende Matrix Z_2 invertiert

$$Z_2^{-1} = \frac{1}{x_2^2 + y_2^2} \begin{pmatrix} x_2 & y_2 \\ -y_2 & x_2 \end{pmatrix} \tag{75}$$

und diese Matrix mit der Matrix Z_1 multipliziert

$$Z = Z_1 \cdot Z_2^{-1} = \frac{1}{x_2^2 + y_2^2} \cdot$$

$$\begin{pmatrix} x_1x_2 + y_1y_2 & x_1y_2 - x_2y_1 \\ x_2y_1 - x_1y_2 & x_1x_2 + y_1y_2 \end{pmatrix} \Leftrightarrow z = \frac{z_1}{z_2}, \tag{76}$$

so hat man eine Repräsentation für $z = z_1/z_2$ erhalten. In BASIC-Befehle umgesetzt ergeben sich die beiden Zeilen

MAT $Z\_2\_strich = INV(Z\_2)$
MAT $Z=Z\_1*Z\_2\_strich$.

Dieses hier skizzierte Verfahren wurde in dem SUB-Programm Komplex_polynom (Listing 16) zur Wertermittlung eines komplexen Polynoms benutzt. Das Rechenverfahren selbst entspricht dem einfachen HORNER'schen Schema und bedarf keiner weiteren Erläuterungen.

Literatur

[9] Practical Temperature Measurements. Application Note 290. Hewlett Packard Loveland, Colorado, USA (1980).

[10] Model 3497 A. Data Acquisition/Control Unit. Operating, Programming and Configuration Manual. Hewlett Packard Loveland, Colorado, USA (1984).

[11] H.T. MEYER: „Matrix statements define complex variables, perform complex math in Basic". Electronic Design July 23, (1981) 176.

[12] I.N. BRONSTEIN, K.A. SEMENDJAJEW: „Taschenbuch der Mathematik". Verlag Nauka, Moskau und BSB B.G. Teubner Verlagsgesellschaft Leipzig (1981).

CAL

Abstract

Jürgen Schwarz

224 **Behandlung von Polynomen**
Teil 4

Während in den ersten drei Teilen dieser Serie einige Grundlagen zur Bearbeitung von Polynomen und zur Bestimmung von ihren Eigenschaften abgeleitet wurden, sollen nun anwendungsorientierte Verfahren behandelt werden.