KERNFORSCHUNGSZENTRUM

KARLSRUHE

November 1975 KFK 2175

Institut für Neutronenphysik und Reaktortechnik

IPØL —
Ein Fortran-Programm zur zweidimensionalen Interpolation

C. Günther

GESELLSCHAFT
FÜR
KERNFORSCHUNG M.B.H.

KARLSRUHE

GESELLSCHAFT FÜR KERNFORSCHUNG M.B.H.

KARLSRUHE

KERNFORSCHUNGSZENTRUM KARLSRUHE
Institut für Neutronenphysik und Reaktortechnik

KFK 2175

I P Ø L -
Ein Fortran-Programm zur zweidimensionalen Interpolation

von

C. Günther

Gesellschaft für Kernforschung mbH, Karlsruhe

Zusammenfassung

Das in diesem Bericht beschriebene FØRTRAN-Programm IPØL löst folgende Aufgabe: Zu einer in der reellen Ebene gegebenen Schar von Punkten (x_i, y_i) $i = 1, 2, \dots, N$ bestimmt IPØL den kleinstmöglichen Grad g, so daß Polynome $P_K(x,y)$, $K = 1, \dots, v$, $v \geq 1$, vom Grad g existieren, die in den (x_i, y_i) verschwinden, d.h.

$$P_K(x_i, y_i) = 0 \quad \text{für } i = 1, \dots, N.$$

Darüber hinaus, wenn g bestimmt ist, berechnet das Programm eine maximale Anzahl linear unabhängiger Polynome vom Grade g.

Diese Aufgabe ist gegenüber der entsprechenden Aufgabenstellung im eindimensionalen aus mehreren Gründen nichttrivial. Dadurch ergibt sich die Möglichkeit, zu einer in der Ebene vorgegebenen Punkteschar ein geeignetes zweidimensionales Interpolationspolynom $P(x,y)$ zu erhalten, indem man $P(x,y) = P_1(x,y)$ setzt, wenn $v = 1$ bzw. indem man eine Linearkombination von $P_1(x,y), \dots, P_v(x,y)$ wählt, falls $v>1$ ist. Dies wird an einem Beispiel demonstriert.

Das Programm wurde zunächst aus dem im folgenden beschriebenen Grund entwickelt: Ebenso wie im eindimensionalen besteht im mehrdimensionalen ein Zusammenhang zwischen Quadraturformeln von gewissen Polynomgenauigkeitsgrad und orthogonalen Polynomen. Dieser Zusammenhang ist im mehrdimensionalen Fall nur unvollständig bekannt. Um zu einigen bekannten zweidimensionalen Quadraturformeln von ungeraden Polynomgenauigkeitsgrad $2\ell-1$ die Anzahl von Polynomen vom Grade ℓ, die bekanntlich orthogonale Polynome sind, zu ermitteln, eignet sich IPØL. Die Resultate dieser Untersuchungen sind in Abschnitt VI angegeben.

IPØL - A FORTRAN routine for twodimensional interpolation

Abstract

Suppose a set of N real points (x_i, y_i), $i = 1, 2, \ldots, N$ points is given. The FØRTRAN-subroutine IPØL calculates a maximum number $v \geq 1$ of linearly independent polynominals $P_1(x,y), \ldots, P_v(x,y)$, which are of least possible degree g and which have the (x_i, y_i) as zeros,

$$P_K(x_i, y_i) = 0 \quad \text{for } i = 1, 2, \ldots, N \text{ and } K = 1, \ldots, v.$$

IPØL is used to find polynominals $P(x,y)$ vanishing in a given point set. This is done by setting $P(x,y) := P_1(x,y)$ if $v = 1$, and if $v > 1$ by taking a suitable linear combination of $P_1(x,y), \ldots, P_v(x,y)$. An example is given for a multivariate interpolation problem.

In addition, an aspect of numerical integration can be clarified by the use of IPØL. For some quadrature formulas of polynominal degree $2\ell-1$ in two dimensions the number of linearly independent polynominals of degree ℓ, which have the points of these formulas as zeros, is given.

INHALTSVERZEICHNIS

	Zusammenfassung
I.	Mehrdimensionale Interpolation
II.	Theorie zum Programm IPØL
III.	Programmtechnische Einzelheiten, Tests
IV.	Beispiel für eine Interpolationsaufgabe
V.	Orthogonale Polynome und Quadraturformeln
VI.	Rechenergebnisse: Anzahl von Polynomen vom Grade ℓ für verschiedene ℓ, die in den Stützstellen bekannter Quadraturformeln vom Grad $2\ell-1$ verschwinden
VII.	Weitere Untersuchungen
	Literatur
	Programmliste

I. Mehrdimensionale Interpolation

Man kann zweierlei Arten von Interpolationsaufgaben formulieren. Im einen Fall sucht man Funktionen, die in vorgegebenen Punkten verschwinden (homogenes Problem), im anderen Fall ist man an Funktionen f (X) interessiert, die in vorgegebenen Punkten X_i vorgegebene Werte f $(X_i) = w_i$ annehmen (inhomogenes Problem). In den meisten praktisch auftretenden Fällen sind die gesuchten Funktionen Polynome, Splines, trigonometrische Polynome oder Exponentialfunktionen.

Für Polynome in einer Veränderlichen läßt sich unmittelbar folgendes angeben: Das Polynom P (x) von möglichst niedrigem Grad, das in N gegebenen verschiedenen Punkten $x_1, \ldots, x_N$ verschwindet, ist vom Grad N und hat die Form

$$P(x) = (x - x_1) \ldots\ldots (x - x_N).$$

Das Polynom Q(x) von möglichst niedrigem Grad, das in N vorgegebenen verschiedenen Punkten $x_1, \ldots, x_N$ vorgegebene Werte $w_1, \ldots, w_N$ annimmt, von denen mindestens einer $\neq 0$ ist, ist vom Grad N-1 und hat die Form:

$$Q(x) = \sum_{i=1}^{N} w_i \, L_i(x),$$

wobei

$$L_i(x) = \frac{(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_N)}{(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_N)}$$

das sogenannte "Lagrangepolynom zu x_i", für welches gilt

$$L_i(x_j) = \delta_{ij} \quad \text{für alle i und j.}$$

Für Polynome in mehreren Veränderlichen sind solche Aussage nicht möglich. Dies liegt zum einen daran, daß man eine Zerlegung von Polynomen in Linearfaktoren nur im eindimensionalen Fall hat. Außerdem ist eine

Interpolation in der reellen Ebene durch vorgegebene Punkte und vorgeschriebene Funktionswerte nicht immer lösbar, manchmal auch nicht eindeutig lösbar. Dies soll an einem einfachen Beispiel erläutert werden:

Wir betrachten quadratische Interpolationspolynome

$$Q = a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2 ,$$

welche sechs freie Parameter enthalten.

Fall a: Wählt man die sechs Interpolationsknoten $X_1 = (0,0)$, $X_2 = (1,0)$, $X_3 = (0,1)$, $X_4 = (2,0)$, $X_5 = (1,1)$, $X_6 = (0,2)$, so hat man für das homogene Problem (alle $w_i = 0$) und das inhomogene Problem jeweils genau eine Lösung, im homogenen Fall die Lösung $Q \equiv 0$. Die zugehörigen Lagrangepolynome setzen sich immer aus zwei Linearfaktoren zusammen, z.B. $L_1(x) = \frac{1}{2}(x + y - 1)(x + y - 2)$ usw.

Fall b: Wählt man vier Punkte $X_1, \ldots, X_4$ auf der X-Achse, zwei andere Punkte X_5, X_6 beliebig nicht auf der X-Achse, so hat das homogene Problem eine nichttriviale Lösung ($Q \not\equiv 0$), nämlich $Q = y \cdot \text{lin}\,(X_5, X_6)$ mit der linearen Funktion $\text{lin}\,(X_5 \cdot X_6) = \alpha + \beta x + \gamma y$, die in X_5 und X_6 verschwindet. Daraus folgt auch, daß das zugehörige inhomogene Problem nur unter bestimmten Bedingungen lösbar ist, welche aus der Linear-Algebra bekannt sind. Es gibt z.B. keine Q mit $Q(X_j) = 0$ für $j = 2, 3, \ldots, 6$ und $Q(X_1) = 1$, da das Verschwinden von Q in X_2, X_3, X_4 zur Folge hat, daß Q den Teiler y enthält und daher immer $Q(X_1) = 0$.

Aus diesem Beispiel läßt sich auch entnehmen, daß mit einem Ansatz für Q,

$$Q = a_{00} + a_{10}x + a_{01}y + \ldots \quad + a_{0P}y^P$$

mit n freien Parametern $a_{00}, \ldots$, wobei Q in m Punkten, $m \leq n$, verschwinden soll, nicht unbedingt $(n - m)$ linear unabhängige Lösungen erhalten werden; es muß vielmehr damit gerechnet werden, daß diese Zahl größer sein kann. Weiterhin kann es auch für $m > n$ (u.U. sogar mehrere) solche Lösungen geben. Diese Zahl und die Gestalt dieser Lösungen wird von IPØL angegeben.

Die mehrdimensionale homogene Polynominterpolation wird z.B. dann von Interesse sein, wenn man durch eine gegebene Schar von Punkten (x_i, y_i), $i = 1, 2, \ldots, N$ in der Ebene nicht mehr sinnvoll eine Kurve $y = f(x)$ ziehen kann mit $f(x_i) = y_i$. In diesem Falle bietet sich an, einen impliziten Ansatz mit einem Polynom $P(x,y)$ von möglichst niedrigem Grad zu versuchen, wobei $P(x_i, y_i) = 0$ sein soll. Diese Aufgabe führt direkt auf das behandelte homogene Interpolationsproblem in zwei Variablen.

II. Theorie zum Programm IPØL

Die Aufgabe, die zu lösen ist, lautet, zu N Punkten $X_i = (x_i, y_i)$ in der reellen Ebene

a) den minimalen Grad g, so daß zumindest ein Polynom $P(x,y)$ vom Grade g existiert mit $P(x_i, y_i) = 0$ für $i = 1, 2, \ldots, N$;

b) die maximale Zahl v linear unabhängiger Polynome $P_1(x,y), \ldots, P_v(x,y)$ vom Grad g und auch v solche Polynome selbst;

zu bestimmen.

Es ist offensichtlich, daß zu diesem Zweck eine Art Suchverfahren konstruiert werden muß. Ein Polynom P in x und y vom Grade g hat die Gestalt:

$$P(x,y) = \sum a_{iK}\, x^i\, y^K \quad \text{mit } i, K \geq 0,\ i + K \leq g$$

mit $\frac{(g+1)(g+2)}{2}$ Koeffizienten a_{iK} und davon g+1 Koeffizienten $a_{g-j,j}$ zu

Potenzen vom Grade g, welche nicht alle identisch verschwinden. Wir nehmen einmal an, g sei bekannt. Man beginnt dann mit einer

Phase I: In dieser Phase wird man zunächst in irgendeiner Systematik einen Ansatz für $P(x,y)$ mit einem, dann mit zwei Termen, ... vom Grade g ansetzen, wobei meist die Anzahl wählbarer Terme $N_o < N$ sein wird. Da man hier zur Bestimmung einer möglichen Lösung $P(x,y)$ nur N_o der N Punkte verwendet, muß noch geprüft werden, ob die restlichen $N - N_o$ Punkte auch auf $P(x,y) = 0$ liegen. Diese Phase bringt zumindest dann ein $P(x,y)$ mit den gewünschten Eigenschaften, wenn $N_o = N$ geworden ist.

Phase II: In dieser Phase, in der man schon irgendwelche $P_1(x,y)$ ... gefunden hat, läßt man eigentlich mehr als N Ansatzterme $a_{iK}\, x^i\, y^K$ zu, schließt aber dann wieder Terme so aus, daß Anteile von schon gefundenen Lösungen von $P_1(x,y)$, ... nicht möglich sind, weil sonst die Aufgabe $P(x,y)$ zu finden, nicht eindeutig lösbar wäre.

Es sind daher bei der zu entwickelnden Prozedur folgende Einzelprobleme zu lösen:

1. Es muß eine Vorgehensweise gefunden werden, bei der alle wesentlich verschiedenen Ansätze mit 1, mit 2, ..., g+1 Ansatztermen vom Grad g berücksichtigt werden und genau einmal auftreten.
2. Da es sich um ein homogenes Problem handelt, muß festgelegt werden, welchen der Koeffizienten a_{iK} mit $i+K = g$ man gleich 1 setzt.
3. Unter Berücksichtigung von 1. und 2. muß ein Verfahren bestimmt werden, das Anteile von schon gefundenen Polynomen aus den noch zu untersuchenden Ansätzen eliminiert.

Die verwendete Prozedur im Hinblick auf die drei beschriebenen Probleme wird im folgenden erläutert:

Wir gehen davon aus, daß es v linear unabhängige Polynome vom Grade g gibt, die in den N Punkten X_i verschwinden, $1 \leq v \leq g+1$. Dies seien o.B.d.A. in etwas geänderter Bezeichnungsweise:

$$P_1(x,y) = b_{10}\, x^g + b_{11}\, x^{g-1}\, y + \ldots \qquad + b_{1g}\, y^g + \ldots$$

$$P_2(x,y) = b_{20}\, x^g \ldots \qquad b_{2g}\, y^g + \ldots$$

$$P_v(x,y) = b_{v0} \ldots \qquad b_{vg}\, y^g + \ldots$$

wobei die Punkte am Ende jeder Zeile für Terme von Potenzen der Ordnung $< g$ stehen. Dies ist gleichbedeutend damit, daß die Matrix

$$(b_{iK}), \quad \begin{matrix} i = 1, \ldots, v \\ K = 0, \ldots, g \end{matrix} \;,$$

den Rang v hat. Analog zu den Eliminationsverfahren für nichtsinguläre quadratische Matrizen, die sich auf Hauptdiagonalform bringen lassen, läßt sich durch Vertauschen von Zeilen und lineare Operationen erreichen, daß man zu einer äquivalenten Darstellung von v neuen Polynomen $Q_1, \ldots, Q_v$ kommt mit

$$Q_\alpha(x,y) = \sum_{j=0}^{g} b'_{\alpha j}\, x^{g-j}\, y^j + \ldots, \qquad \alpha = 1, \ldots, v,$$

in der die Elemente b'_{iK} mit $i > K + 1$ und $K > i + g - v$ verschwinden (z.B. für $v = 4$).

$$(b'_{iK}) = \begin{pmatrix} x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & 0 & 0 & 0 \\ 0 & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & 0 & 0 \\ 0 & 0 & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & 0 \\ 0 & 0 & 0 & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x \end{pmatrix}$$

Diese mögliche Darstellung der $P_\alpha, \alpha = 1, \ldots, v$ zugrunde legend, beginnen wir unsere Suchprozedur. Es muß jedoch bedacht werden, daß v nicht a priori bekannt ist. Wir setzen unser Verfahren folgendermaßen an:

Für $M = 0, \ldots, g$ machen wir die Ansätze mit $K = 0, \ldots, M$ und $K + 1$ Termen vom Grad g.

$$P(x,y) = \sum_{K=M-K}^{M} a_{g-K,K} \; x^{g-K} \; y^{K} + \ldots..$$

Die höchste vorkommende x-Potenz heißt daher $g - M + K$, die kleinste $g - M$. Die erste Größe wird im Programm mit N1, die zweite mit N2 bezeichnet, also

$$P(x,y) = \sum_{L=N1}^{N2} a_{L,g-L} \; x^{L} \; y^{g-L} \; .$$

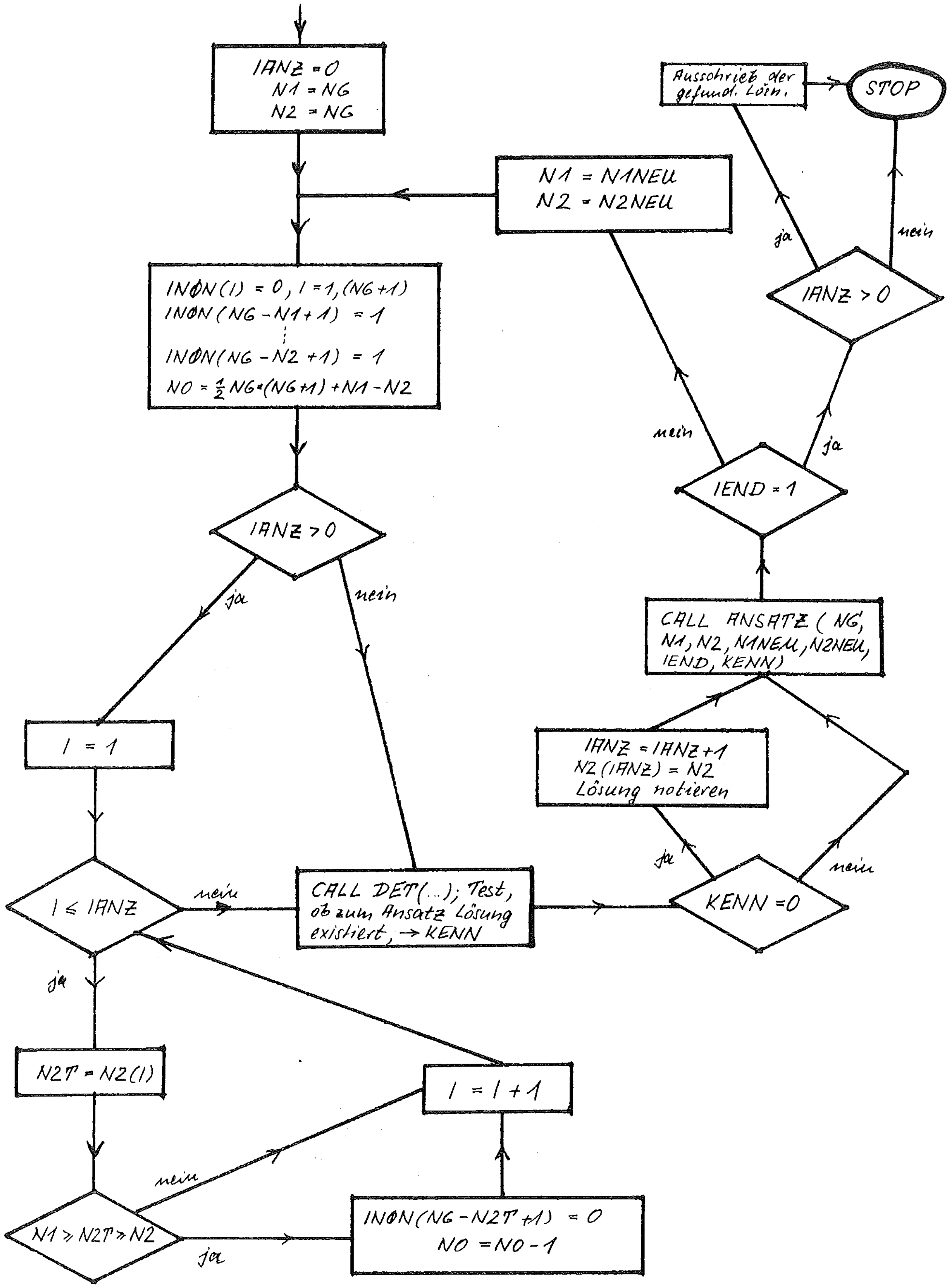
Für festes M ist N2 konstant, während sich N1 immer um 1 erhöht, ausgehend von N1 = N2, bis zu N1 = g bzw. bis ein P(x,y) gefunden ist, das zu N2 unsere Forderung erfüllt. Für festes M wird immer der Koeffizient zu L = N2, d.h. zur Potenz $x^{N2} \; y^{g-N2} \; a_{N2,\; g-N2}$ gleich 1 gesetzt.

Findet die Prozedur eine Lösung $P_\alpha(x,y)$, $\alpha = 1, \ldots;$ so wird das zugehörige N2 = N2(α) gespeichert. In diesem Falle, wenn zu einem M ein P_α gefunden ist, wird M weiter erhöht, da von P_α linear unabhängige Lösungen von gleichem N2 mit größerem N1 schon gefunden sein müssen.

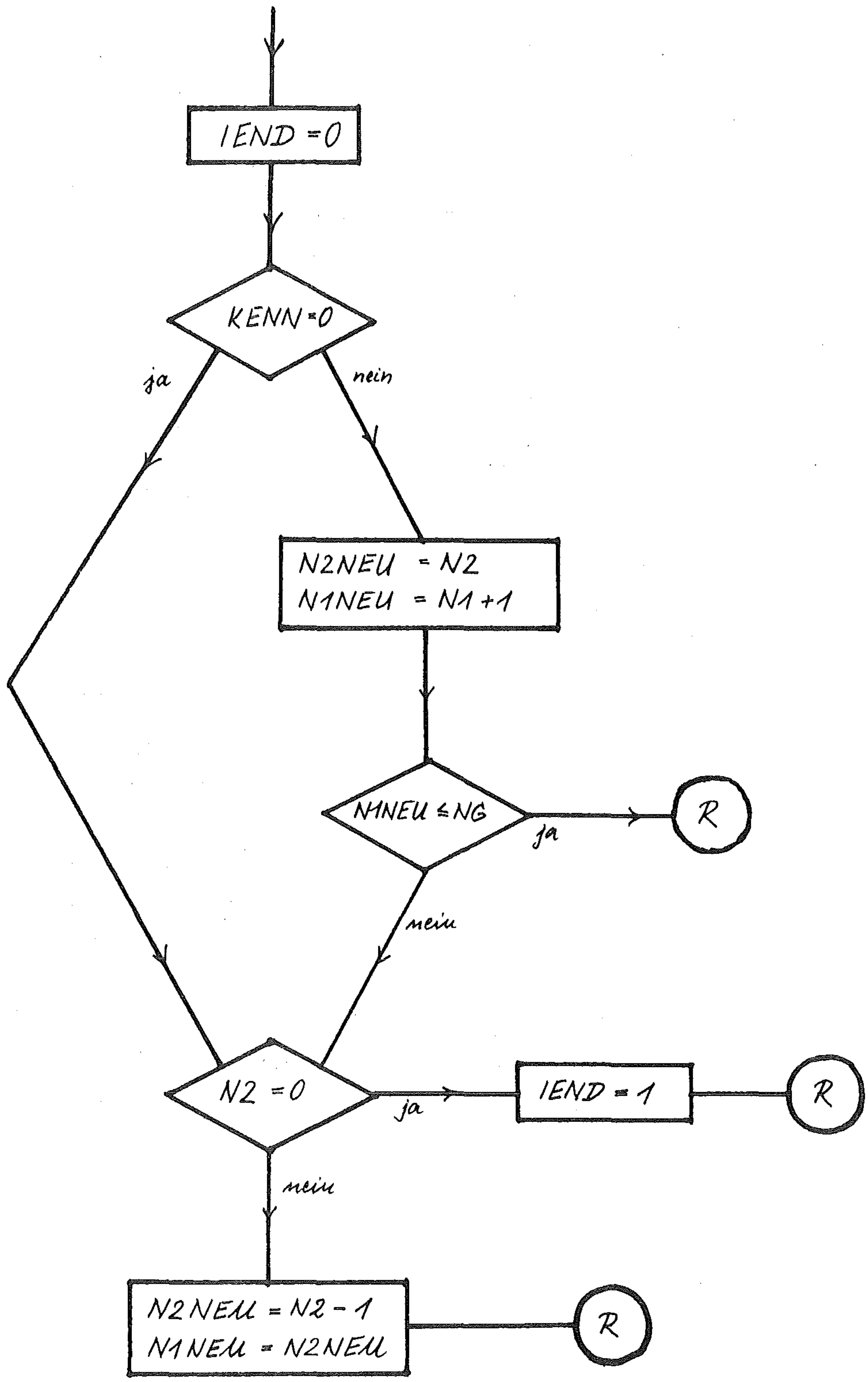
Hat man schon mindestens ein $P_\alpha(x,y)$ gefunden, so werden in den folgenden Ansätzen die Ansatzterme zu jedem gefundenen N2(α), $\alpha = 1, \ldots,$ d.h. die Terme zu Potenzen $x^{N2(\alpha)} \; y^{g-N2(\alpha)}$ weggelassen. Daher kann jeder neu versuchte Ansatz keinen Anteil einer vorher gefundenen Lösung $P_\alpha(x,y)$ enthalten, der das zugehörige lineare Gleichungssystem zur Bestimmung der a_{iK} singulär machen würde.

Die gesamte Prozedur wird im folgenden Flußdiagramm zusammengefaßt. Es wird hier der Einfachheit halber angenommen, das g bekannt sei. In IPØL werden über die Eingabe Werte NG für untere und obere Schranken (NMIN und NMAX) von g angegeben, die mit aufsteigendem NG, beginnend mit NG = NMIN, nach möglichen Lösungen untersucht werden.

FLUSSDIAGRAMM VON 'IPØL'



FLUSSDIAGRAMM DES UNTERPROGRAMMES 'ANSATZ'



III. Programmtechnische Einzelheiten, Tests

Von den bisher nicht beschriebenen Einzelheiten seien noch folgende Einzelheiten erwähnt.

III.1 Das Programm wird in der vorliegenden Standardform in konversationellem Mode unter TSO auf der IBM 370/165 betrieben. Das Programm liest nacheinander in

K(ARTE) 1:	N, NMIN, NMAX, EPS, NKENN
N	Anzahl der Punkte
NMIN	Niedrigster Polynomgrad, der untersucht wird ($\geq$ 2!)
NMAX	Größter Polynomgrad, der u.U. untersucht wird
EPS	Genauigkeitsschranke. Sie wird benutzt, wenn der Polynomansatz weniger als N Parameter nämlich N_0 enthält. Falls mit den ersten N_0 Punkten ein Polynom P gefunden wird, das in diesen Punkten verschwindet, wird P als Lösung akzeptiert, wenn $\lvert P(x_i,y_i)\rvert \leq$ EPS, für i = N_0 + 1, ..., N;
NKENN	Kennzeichnet die nachfolgende Eingabeart für die $X_i = (x_i,y_i)$. Es gibt folgende Varianten:

NKENN = 0 :	Lesen aller X_i hintereinander
NKENN = 1 :	Wiederholung mit neuen NMIN, NMAX und EPS ohne neue Eingabe
NKENN = -2 :	Die Werte der X_i sind in der Subroutine DEFINE einprogrammiert.
NKENN = -1 :	In DEFINE wird eine Punkteraster eingelesen, das nach einem bestimmten System zu einer Punktemenge ausgebaut wird.

S1	Wenn NKENN = 1 oder -2 Übergang zu K3 Wenn NKENN = 0, folgt K2, dann K3 Wenn NKENN = -1, folgt in DEFINE spezifizierte Eingabe, dann K3.	
K2	(X(I), Y(I))	für I = 1, ..., N
K3	NWE,	Wenn NWE = 11111, erfolgt STØP, andernfalls Rücksprung nach K1.

Bemerkung:

1. In der beigefügten Programmliste werden die Variablen der Karte K1 im Hauptprogramm definiert, welches IPØL aufruft. Die Werte der Variablen N, NMIN und NMAX werden über die Aufrufliste definiert, EPS und NKENN werden über den CØMMØN definiert.

2. A muß ein genügend großes Variablenfeld sein, das im Hauptprogramm definiert wurde.

3. Will man das Ergebnis des IPØL-Aufrufes numerisch weiterverwenden, so ist das auf folgende Weise möglich. Hat man v Polynome gefunden, so ist IANZ = v. Im Feld A ab Adresse 2*N+1 stehen die Koeffizienten der gefundenen Polynome in folgender Struktur: als zweidimensionaler Array T (NS, NTØT), wobei NS = NMAX + 1; der erste Index bezeichnet die erste, zweite, ... gefundene Lösung. Der zweite Index charakterisiert die Koeffizienten der zugehörigen Lösung. Es ist NTØT = (NMAX+1)*(NMAX+2), da es insgesamt NTØT Koeffizienten zum Grad NMAX gibt. Die Reihenfolge ihrer Zugehörigkeit zu den Monomen (in aufsteigender Reihenfolge) enthält folgendes Schema: 1, x, y, x^2, xy, y^2, ..., x^{NMAX}, x^{NMAX-1} y, ..., y^{NMAX}, d.h. zur ersten gefundenen Lösung steht der konstante Koeffizient in T(1,1), der Koeffizient zu x in T(1,2), der zu y in T(1,3), zu x^2 in T(1,4), ... usw.

III.2 Bedeutung einiger Variablen

in allen Routinen:

N   Anzahl von vorgegebenen Punkten (X(I), Y(I))

NO   Anzahl der frei wählbaren Parameter im aktuell behandelten Ansatz

NG   Grad des aktuellen Ansatzpolynoms

NGS   NG + 1

N1 }
N2 }   Ansatzterme vom Grad NG enthalten unter den Termen $x^{N1}\, y^{NG-N1},\ x^{N1-1}\, y^{NG-N1+1},\ \ldots,\ x^{N2}\, y^{NG-N2}$.

INØN(NGS)   Falls INØN(I) = 1, wird der Ausdruck $x^{NG-I+1}\, y^{I-1}$ im Ansatz berücksichtigt, falls INØN(I) = 0, nicht.

III.3 Tests

Zu einigen Quadraturformeln von verschiedenen Graden in STRØUD /10/ sind die zugehörigen orthogonalen Polynome angegeben, die in den Stützstellen dieser Formeln verschwinden. Diese wurden immer, entsprechend der benutzten einfachen Genauigkeit, in befriedigender Weise reproduziert. Dies wurde auch für einige anderen bekannte Formeln durchgeführt.

Es hat sich als vorteilhaft erwiesen, bei der Option NKENN = -1 die Koordinaten der Stützstellen so einzulesen, daß die Werte zu Punktmengen mit MØD(I) = 3 zuerst eingelesen werden.
Ein mögliches Versagen von IPØL kann manchmal dadurch behoben werden, daß man die Punkte $X_i = (x_i, y_i)$ in einer anderen Reihenfolge eingibt. Das Vorgehen des Programmes hängt von der Reihenfolge der eingegebenen Punkte ab.

III.4 Benutzte Unterprogramme

Zur Lösung der auftretenden linearen Gleichungssysteme wurden Standardverfahren benützt, die durch ein Eliminationsverfahren gefundene Lösungen dieser Gleichungen iterativ verbessern.

IV. Beispiel für eine Interpolationsaufgabe

In diesem Abschnitt wird eine typische zweidimensionale Interpolationsaufgabe behandelt. Es werden 12 Punkte vorgegeben, die symmetrisch zur Winkelhalbierenden im ersten Quadranten liegen. Es sind die Punkte $X_1 = (0.2, 0.2)$ und $X_2 = (1., 1.)$ auf dieser Geraden, die Punkte $X_3 = (1., 0.)$, $X_4 = (1.3, 0.1)$, $X_5 = (2.3, 0.7)$, $X_6 = (1.8, 1.)$, $X_7 = (1.2, 0.9)$ und ihre zur Gerade $x = y$ spiegelbildlich symmetrisch liegenden Punkte.

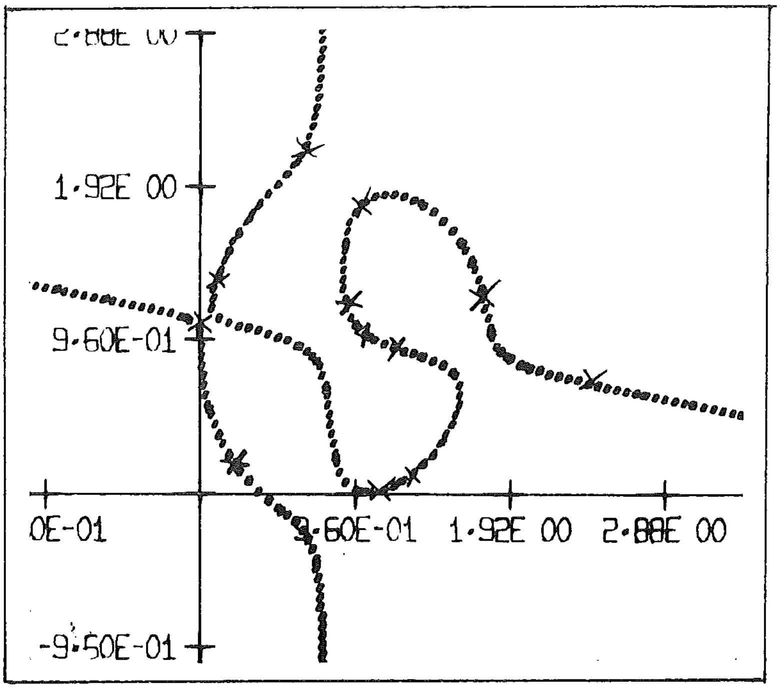
Der Aufruf von IPØL mit NMIN = NMAX = 4 ergibt drei linear unabhängige Polynome P_1, P_2 und P_3 vom Grad 4, wobei entsprechend den Konventionen von Abschnitt I.

$$P_1 = x^2y^2 + \alpha_1 x^3y + \alpha_2 x^4$$

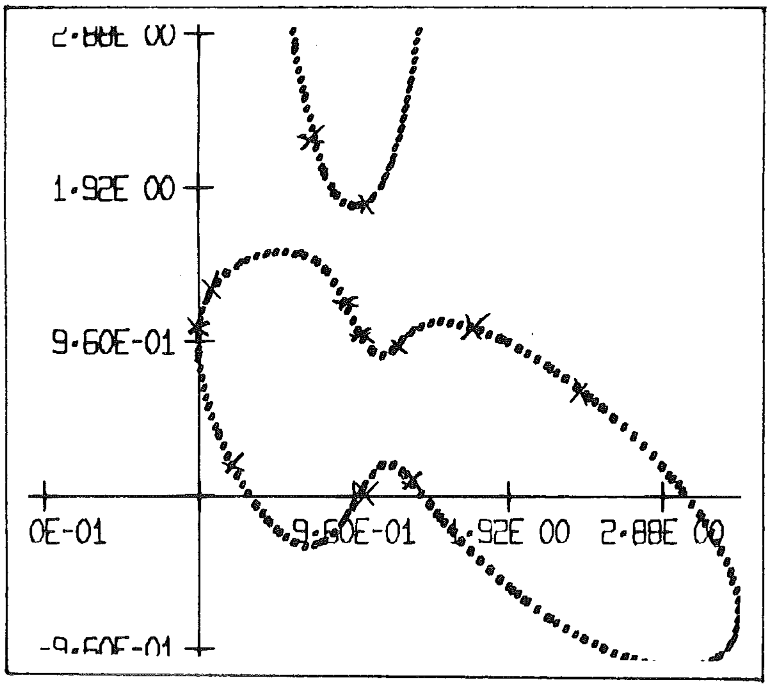
$$P_2 = xy^3 + \beta_1 x^3y + \beta_2 x^4$$

$$P_3 = y^4 + \gamma_1 x^3y + \gamma_2 x^4 \quad .$$

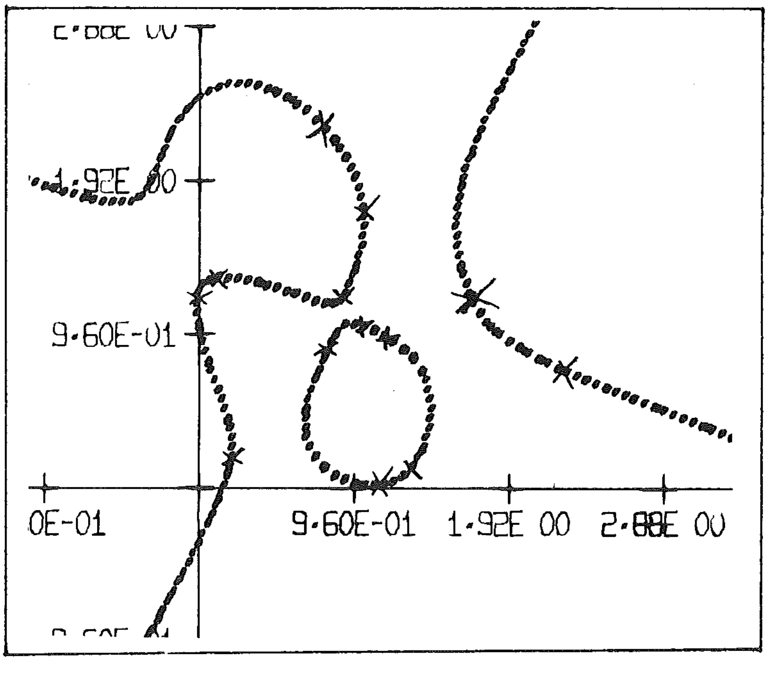
Die Kurven $P_i = 0$, $i = 1, 2, 3$, sind in den Abbildungen 1 und 2 dargestellt, ihre Koeffizienten sind in Tabelle 1 wiedergegeben. Es gibt eine Linearkombination Q von P_1, P_2 und P_3, die symmetrisch ist und wie Abbildung 2 zu entnehmen ist, die Kurve $Q = 0$ ist eine geschlossene Kurve.



$P_1 = 0$



$P_2 = 0$



$P_3 = 0$

Abbildung 1

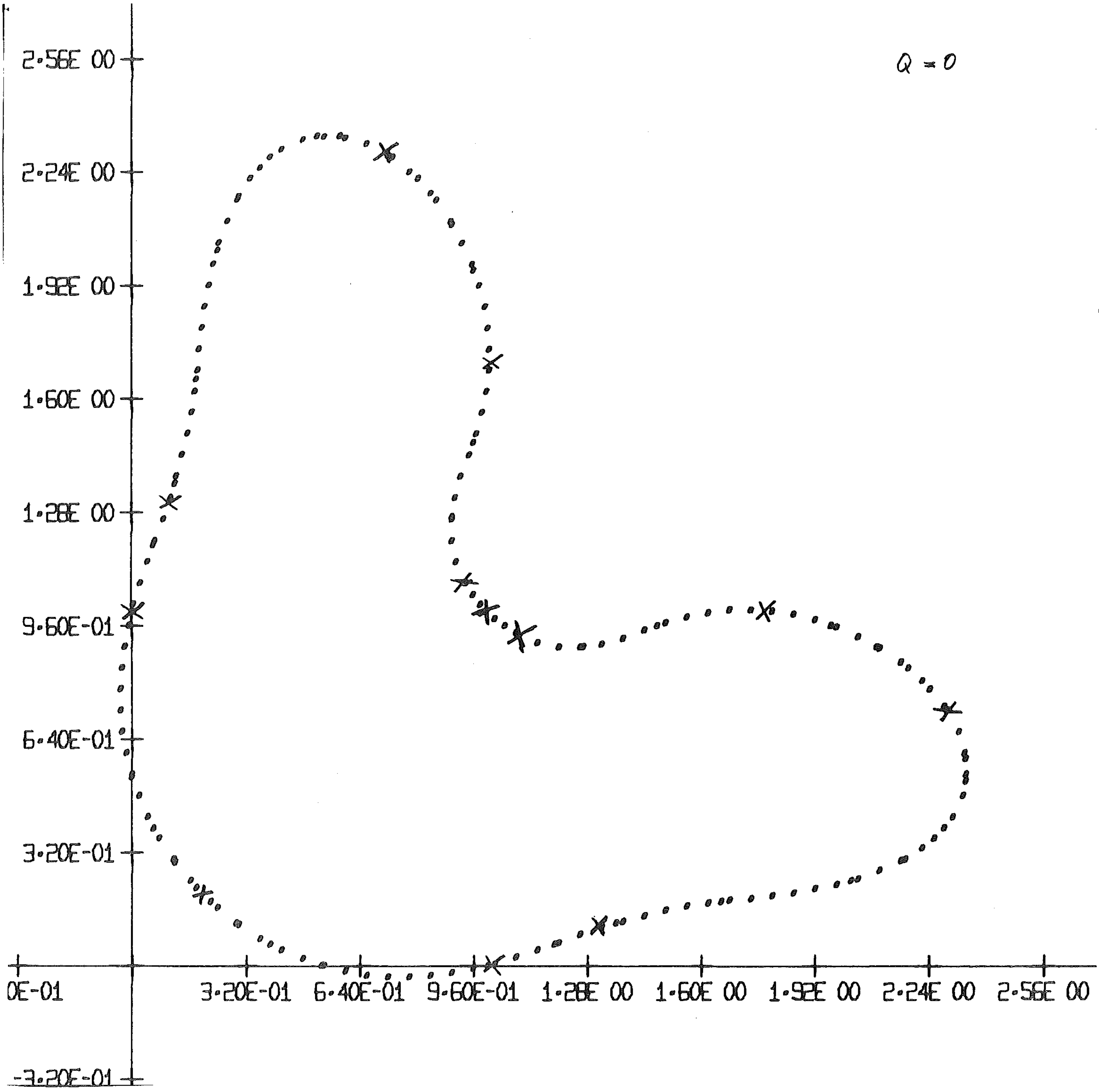

Q = 0
2·56E 00
2·24E 00
1·92E 00
1·60E 00
1·28E 00
9·60E-01
6·40E-01
3·20E-01
-3·20E-01
0E-01
3·20E-01
6·40E-01
9·60E-01
1·28E 00
1·60E 00
1·92E 00
2·24E 00
2·56E 00

Abbildung 2

0	0.951690E+00				
1	-0.501990E+01	-0.215492E+01			
2	0.752780E+01	0.527225E+01	0.132938E+01		
3	-0.419396E+01	-0.452445E+01	-0.197617E+01	-0.120140E+00	
4	0.734364E+00	0.118006E+01	0.100000E+01	0.0	0.0
0	0.657959E+00				
1	-0.318302E+01	-0.205435E+01			
2	0.461709E+01	0.478605E+01	0.217521E+01		
3	-0.238133E+01	-0.379844E+01	-0.279454E+01	-0.778610E+00	
4	0.289305E+00	0.146460E+01	0.0	0.100000E+01	0.0
0	-0.105729E+01				
1	0.901615E+01	-0.600115E+00			
2	-0.158398E+02	-0.769080E+01	0.494514E+01		
3	0.936584E+01	0.957385E+01	0.102054E+01	-0.428774E+01	
4	-0.146480E+01	-0.396088E+01	0.0	0.0	0.100000E+01
0	0.242009E+01				
1	-0.903851E+01	-0.903852E+01			
2	0.110587E+02	0.127145E+02	0.110587E+02		
3	-0.544026E+01	-0.762847E+01	-0.762848E+01	-0.544028E+01	
4	0.999998E+00	0.999995E+00	0.290254E+01	0.100000E+01	0.100000E+01

<u>Tabelle 1:</u> Liste der Koeffizienten der Polynome P_1, P_2, P_3 und Q, Koeffizienten zu einem festen Grad, der links steht, auf einer Zeile.

V. Orthogonale Polynome und Quadraturformeln

Zu einem vorgegebenen Integral

$$I(f) = \iint_D f(x,y)\, w(x,y)\, dxdy$$

mit nichtnegativer Gewichtsfunktion $w(x,y)$ und $D \subset \mathbb{R}^2$ nennt man

$$S(f) = \sum_{i=1}^{N} A_i\, f(x_i,y_i)$$

eine zugehörige Quadraturformeln vom Polynomgenauigkeitsgrad n, wenn $I(p) = S(p)$ für alle Polynome p vom Grade $\leq n$.

In Analogie zur entsprechenden Problemstellung im eindimensionalen existiert ein Zusammenhang zwischen Quadraturformeln und orthogonalen Polynomen. Es gilt z.B. folgender

SATZ 1:

Existiert ein Polynom $p = p(x,y)$ vom Grade $v \leq n$, das in den Punkten $X_i = (x_i, y_i)$ einer Quadraturformel $S(f)$ vom Grade n verschwindet, so ist p bezüglich des Skalarproduktes

$$(f,g) = I(f \cdot g)$$

orthogonal zu allen Polynomen vom Grade $\leq n - v$; z.B. GÜNTHER /4/.
Der Beweis ist trivial.

Von besonderem Interesse ist der Spezialfall des Satzes mit $n = 2\ell-1$, $v=\ell$. Polynome vom Grad ℓ existieren sicher, wenn $N < \frac{(\ell+1)(\ell+2)}{2}$, der Anzahl linear unabhängiger Polynome in x und y vom Grade $\leq \ell$. Es gilt für die Anzahl μ linear unabhängiger Polynome vom Grad $\leq \ell$, die in den X_i verschwinden die Ungleichung

$$\mu \geq \frac{(\ell+1)\ (\ell+2)}{2} - N$$

z.B. HIRSCH /5/.

Hier interessiert es häufig, ob das Gleichheitszeichen gilt. Wenn $N \geq \frac{(\ell+1)(\ell+2)}{2}$, will man gelegentlich wissen, ob es überhaupt Polynome vom Grad ℓ gibt, die in den X_i verschwinden.

Der Hintergrund dieser Fragestellung ist darin zu sehen, das es Sätze gibt, mit denen man Quadraturformeln vom Grad (= Polynomgenauigkeitsgrad) $2\ell-1$ konstruieren kann, in denen die Stützstellen (x_i, y_i) die ℓ^2 unter den gemeinsamen Nullstellen von orthogonalen Polynomen enthalten sind, z.B. MYSOVSIKH /7/ und STRØUD /9/. Während solche Sätze meist nicht Quadraturformeln mit möglichst kleiner Stützstellenzahl zu festem ℓ liefern, hat man mit verschiedenen Methoden z.B. in /2/ entweder völlig oder teilweise von der Benutzung von orthogonalen Polynome unabhängig Quadraturformeln mit kleiner oder sogar minimaler Knotenwahl gefunden. Der detaillierte Zusammenhang dieser Formeln mit orthogonalen Polynomen ist noch nicht vollständig geklärt und steht mit als Motiv für die Erstellung des Programmes IPØL.

Zu den Methoden, die keinen Gebrauch von orthogonalen Polynomen bei der Konstruktion von Quadraturformeln machen, gehört vor allem die folgende: Für Integrale I(f), für die $I(x^n y^m) = I(x^m y^n)$, wenn D ein Kreis, ein Quadrat oder die ganze Ebene, wählt man eine Folge von erzeugenden Punkten vom Typ $(a_i, 0)$, (b_i, b_i), (c_i, d_i), aus denen man durch Vorzeichenwechsel und Koordinatentauschen, analog wie in II. bei der Beschreibung der Eingabe von K4, K5 und K6, eventuell noch unter Einschluß des Punktes (0, 0), die Stützstellen einer Formel erhält. Als freie Parameter treten dabei die a_i, b_i, c_i, d_i und die Gewichte zu den Punkten mit dem gleichen Erzeuger auf. Zum ersten Male systematisch verwendet wurde dies von HAMMER u. STRØUD /6/, dann später extensiv von RABINOWITZ u. RICHTER /8/ u.a. Entsprechendes läßt sich für Dreiecke unter Verwendung baryzentrischer Koordinaten durchgeführen, COWPER /2/.

Eine andere Methode von FRANKE /3/ besteht darin, aus einer Schar von Paaren von orthogonalen Polynomen ein solches Paar auszusuchen, in dem einige der Gewichte zu den als Stützstellen auftretenden gemeinsamen Nullstellen beider Polynome verschwinden. Zu untersuchen ist dann, was für ein weiteres Polynom außerdem noch in diesen Punkten verschwindet.

VI. Ergebnisse

1. Quadraturformeln von RABINØWITZ und RICHTER /8/:

Bemerkung: In der letzten Spalte der folgenden Tabellen werden die verschiedenen Arten von erzeugenden Punkten (Generator) der Formeln angegeben. Eine 1 entspricht einem Punkt (a, 0), der noch die Punkte (-a, 0), (0, a) und (0, -a) involviert, eine 2 entspricht einem Punkt (b, b), der außerdem die weiteren Punkte (b, -b), (-b, b) und (-b, -b) erzeugt, einer 3 entspricht ein Erzeuger (a, b), der sieben weitere (-a, b), (a, -b), (-a, -b), (b, a), (-b, a), (b, -a) und (-b,-a) erzeugt, eine 4 entspricht dem Punkt (0, 0).

Tabelle 1 in /8/:

Punkte	Grad $2\ell-1$ der QF.	ℓ	Anzahl der l.u.Polynome vom Grad ℓ, die in den X_i verschwinden	Generatoren
20	9	5	2	1123
26	11	6	3	112234
28	11	6	2	112223
37	13	7	2	11222334
44	15	8	3	111222233
48	15	8	1	111222333

Tabelle 2 in /8/:

20	9	5	2	1123
21	9	5	2	11234
28	11	6	3	11133
28	11	6	2	111223
37	13	7	2	11122334
44	15	8	3	111122233

Tabelle 3 in /8/:

Punkte	Grad $2\ell-1$ der QF.	ℓ	Anzahl der l.u.Polynome vom Grad ℓ, die in den X_i verschwinden	Generatoren
20	9	5	2	1223
28	11	6	3	11133
28	11	6	2	111223
37	13	7	2	11222334
44	15	8	3	111122233

Tabelle 4 in /8/:

20	9	5	1	1123
28	11	6	3	11133
28	11	6	2	111223
37	13	7	2	11222334
44	15	8	3	111122233

Mit einer Ausnahme scheint die Anzahl der linear unabhängigen Polynome in x und y vom Grade ℓ, die in den Stützpunkten X_i der QF. verschwinden, vom Typ der verwendeten Generatoren abzuhängen.

2. Beispiele aus FRANKE /3/:

Tabelle 3, Beispiel mit 21 Punkten:

21	9	5	2	

Tabelle 5

13	7	4	2	234

Tabelle 8

20	9	5	2	

3. Beispiele aus ALBRECHT /1/:Ü

19	9	5	3
28	11	6	2
41	13	7	0
48	15	8	0
61	17	9	4 (?)

Das letzte Resultat, 61 Punkte, ist mit Vorsicht aufzufassen. Es wurde erhalten, indem EPS sehr groß, 2.E-2, gewählt wurde.

4. Beispiel, CØWPER /2/:

13	7	4	2

VII. Weitere Untersuchungen

Eine weitergehende Fragestellung ist die folgende: Es soll eine Basis des Polynomideals i bestimmt werden, das die Stützstellen X_i einer Quadraturformel S(f) als Nullstellen hat. Im allgemeinen haben die Basispolynome nicht den gleichen Grad. Zur Konstruktion einer Basis verfährt man entsprechend der Methode, die in Abschnitt 1 beschrieben wurde. Man bestimmt gleichzeitig den minimalen Grad g, zu dem Polynome existieren, die in den X_i verschwinden, und eine Maximalzahl linear unabhängiger solcher Polynome vom Grad g. Danach folgt das entsprechende Verfahren für den Grad g + 1, wobei zu beachten ist, das sowohl Anteile schon gefundener Polynome vom Grad g + 1 als auch Anteile von xP_1 und yP_1 zu Polynomen P_1 vom Grad g im Ansatz nicht berücksichtigt werden.

In den meisten Fällen genügt die Kenntnis der Anzahl der (orthogonalen) Polynome vom Grad ℓ, um zu einer Quadraturformel vom Grad $2\ell-1$ die Struktur der gesamten Polynombasis des Ideals i zu erkennen, d.h. die Anzahl der Basispolynome von den Graden $> \ell$ zu i ermitteln zu können.

Es wurde auch eine Variante von IPØL erstellt, die imstande ist, eine Idealbasis zu berechnen, wenn nur die vorgegebenen Punkte verschieden sind.

Zu diesem Sachverhalt soll noch ein kleines Beispiel angegeben werden. Die Formel C_2 : 9 - 1 aus /10/, identisch mit der 20-Punktformel vom Grad 9 aus /8/ besitzt eine Polynombasis, die zum einen aus zwei Polynomen vom Grad 5, entsprechend Abschnitt IV., und aus zwei Polynomen vom Grad 6 besteht.

LITERATUR

/1/ ALBRECHT, J.
Formeln zur numerischen Integration über Kreisbereiche,
ZAMM, Vol.40 (1960), pp. 514-517

/2/ CØWPER, G.R.
Gaussion quadrature formulas for triangles,
AIAA Journal, 7, No. 3 (1973), pp. 405-408

/3/ FRANKE, R.
Obtaining cubatures for rectangles and other planar regions by using orthogonal polynominals
Math. Comput., V.25 (1971), pp. 803-817

/4/ GÜNTHER, C.
Third degree integration formulas with four real points and positive weights in two dimensions,
SIAM J. Numer. Anal. 11,3 (1974), pp. 780-793

/5/ HIRSCH, P.M.
Evaluation of orthogonal polynominals and relationship to evaluating multiple integrals
Math. Comput., V.22 (1968), pp. 280-285

/6/ HAMMER, P.C. and STROUD, A.H.
Numerical evaluation of multiple integrals II,
MTAC, V.12 (1958), pp. 272-280

/7/ MYSOVSHIKH, I.P.
Cubature formulae and orthogonal polynominals,
Zurn. vycl. Mat. mat. Fiz 9 (1969), pp. 419-425

/8/ RABINOWITZ, P. und RICHTER, N.
Perfectly symmetric two-dimensional integration formulas with minimal number of points,
Math. Comput., V. 23 (1969), pp. 765-779

/9/ STROUD, A.H.
Integration formulas and orthogonal polynominals for two variables,
SIAM J. Numer. Anal., 6 (1969), pp. 222-229

/10/ STROUD, A.H.
Approximate calculation of the multiple integrals,
Prentice Hall, 1971.

A N H A N G 1 :

Programmliste

```
      SUBROUTINE IPOL(N,NMIN,NMAX,A)
      DIMENSION A(1)
      COMMON EPS,NKENN,IANZ
    1 CONTINUE
      IF(NMIN.LE.1)NMIN=2
      NTOT=((NMAX+1)*(NMAX+2))/2
      NS=NMAX+1
      NL=1999-2*N-NTOT*NS
      N11=2*N+NS*NTOT+1
      CALL DRIVER(A,A(N+1),A(2*N+1),A(N11),N,NL,NMIN,NMAX,NS,NTOT)
   83 READ(6,*)NWE
      IF(NWE.NE.11111)GO TO 1
      RETURN
      END
```

```
      SUBROUTINE DRIVER(X,Y,T,Z,N,N1,NMIN,NMAX,NS,NTOT)
      DIMENSION X(N),Y(N),Z(N1),T(NS,NTOT)
      DIMENSION IST(10,2),INON(10)
      COMMON EPS,NKENN,IANZ
      IF(NKENN)51,50,52
   50 DO 2 I=1,N
    2 READ(6,*)X(I),Y(I)
      GO TO 52
   51 CALL DEFINE(X,Y,N)
   52 IF(N.GT.10)GO TO 55
      DO 53 I=1,N
   53 WRITE(6,54) I,X(I),Y(I)
   54 FORMAT(5X,I2,2X,2E20.8)
   55 CONTINUE
      DO 10 I=1,N
      DO 11 J=I,N
      IF(J.EQ.I)GO TO 11
      S=ABS(X(I)-X(J))+ABS(Y(I)-Y(J))
      IF(S.GT.10.*EPS)GO TO 11
      WRITE(6,12)I,J
   12 FORMAT(/'  WARNUNG, PUNKTE ZU BENACHBART ',2I5)
   11 CONTINUE
   10 CONTINUE
      IF(N.LE.((NMAX+1)*(NMAX+2))/2)GO TO 1
      WRITE(6,360)NMAX
  360 FORMAT(' *** NMAX UNPASSEND, NMAX = ',I3,'  ***')
    1 IANZ=0
      CALL RECIN(T,NS,NTOT)
      DO 3 NG=NMIN,NMAX
      N1=NG
      N2=NG
    4 N0=(NG*(NG+1))/2+N1-N2
      N11=NG-N1+1
      N22=NG-N2+1
      NGS=NG+1
      DO 32 I=1,NGS
   32 INON(I)=0
      DO 31 I=N11,N22
   31 INON(I)=1
      IF(IANZ.EQ.0) GO TO 30
      DO 33 IA=1,IANZ
      N2T=IST(IA,2)
      IF(N2T.GE.N2.AND.N2T.LE.N1)N0=N0-1
   33 INON(NG-N2T+1)=0
   30 CONTINUE
```

```
      CALL  DET(N1,N2,N,NG,N),X,Y,Z,Z(NO*NO+1),Z(NO*NO+NO+1),KENN,INCN)
  167 FORMAT(4X,'K1 = ',I2,'  K2 = ',I2,' KENN = ',I2)
      IF(KENN.NE.0)GO TO 5
      IANZ=IANZ+1
      IST(IANZ,1)=N1
      IST(IANZ,2)=N2
      CALL  RECORD(N1,N2,N,NO,NG,Z(NO*NO+NO+1),IANZ,INCN)
    5 CALL  ANSATZ(NG,N1,N2,N1NEU,N2NEU,IEND,KENN)
      IF(IEND.EQ.1)GO TO 6
      N1=N1NEU
      N2=N2NEU
      GO TO 4
    6 IF(IANZ.NE.0)GO TO 7
    3 CONTINUE
    7 CALL  RECFIN(IANZ)
      RETURN
   40 WRITE(6,41)
   41 FORMAT('  INCORRECT END IN DRIVER')
      STOP
      END
```

```
      SUBROUTINE ANSATZ(NG,N1,N2,N1NEU,N2NEU,IEND,KENN)
      IEND=0
      IF(KENN.EQ.0)GO TO 10
      N2NEU=N2
      N1NEU=N1+1
      IF(N1NEU.LE.NG)RETURN
   10 IF(N2.EQ.0)IEND=1
      N2NEU=N2-1
      N1NEU=N2NEU
      RETURN
      END
```

```
      FUNCTION POT(X,N,Y,M)
      T=1.
      IF(N.GT.0)T=X**N
      S=1.
      IF(M.GT.0)S=Y**M
      POT=S*T
      RETURN
      END
```

```
      SUBROUTINE RECIN(T,NS,NTOT)
      DIMENSION T(NS,NTOT)
      RETURN
      ENTRY RECORD(N1,N2,N,NO,NG,C,IANZ,INCN)
      DIMENSION C(NO),INON(1)
      DO 1 I=1,NTOT
    1 T(IANZ,I)=0.
      NP=(NG*(NG+1))/2
      DO 2 I=1,NP
    2 T(IANZ,I)=C(I)
      NN=N1-N2
      IF(NN.EQ.0)GO TO 10
      I1=NG-N1+1
      I2=NG-N2
      J=1
      DO 3 K=I1,I2
      IF(INON(K).EQ.0)GO TO 3
      T(IANZ,NP+K)=C(NP+J)
      J=J+1
    3 CONTINUE
```

```
10 T(IANZ,NP+NG+1-N2)=1.
   RETURN
   ENTRY RECFIN(IANZ)
   WRITE(6,22)IANZ,NG
22 FORMAT(//3X,'ES WURDEN',I2,' LOES. VOM GRAD',I2,' GEFUNDEN')
   IF(IANZ.EQ.0)GO TO 23
   DO 20 I=1,IANZ
20 WRITE(6,21)I,(T(I,J),J=1,NTOT)
21 FORMAT(//5X,I4//(3X,6E18.7))
23 RETURN
   END
```

```
   SUBROUTINE DET(N1,N2,N,NG,NO,X,Y,A,B,C,KENN,INON)
   DIMENSION X(N),Y(N),A(NO,NO),B(NO),C(NO)
   DIMENSION INON(1)
   COMMON EPS
   KENN=0
   IF(N1.LT.N2)GO TO 1000
   J=1
   NGS=NG+1
   DO 1 II=1,NGS
   I=II-1
   I1=1
   I2=II
   IF(I.LT.NG)GO TO 3
   I1=NG-N1+1
   I2=NG-N2
   IF(N1.EQ.N2)GO TO 1
 3 CONTINUE
   DO 2 K=I1,I2
   IF(I.LT.NG)GO TO 5
   IF(INON(K).EQ.0)GO TO 2
 5 CONTINUE
   DO 4 L=1,NO
 4 A(L,J)=POT(X(L),I-K+1,Y(L),K-1)
   J=J+1
 2 CONTINUE
 1 CONTINUE
   DO 11 L=1,NO
11 B(L)=-POT(X(L),N2,Y(L),NG-N2)
   CALL GAUSS2(NO,A,B,C,IERR)
   IF(IERR.NE.0)GO TO 1002
   IF(NO.EQ.N)RETURN
   NNS=NO+1
   DO 20 L=NNS,N
   T=0.
   J=1
   DO 21 II=1,NGS
   I=II-1
   I1=1
   I2=II
   IF(I.LT.NG)GO TO 23
   I1=NG-N1+1
   I2=NG-N2
   IF(N1.EQ.N2)GO TO 21
23 CONTINUE
   DO 22 K=I1,I2
   IF(I.LT.NG)GO TO 15
   IF(INON(K).EQ.0)GO TO 22
15 T=T+C(J)*POT(X(L),I-K+1,Y(L),K-1)
```

```
      J=J+1
   22 CONTINUE
   21 CONTINUE
      T=T+POT(X(L),N2,Y(L),NG-N2)
      IF(ABS(T).GT.EPS)KENN=2
      IF(KENN.EQ.2)RETURN
   20 CONTINUE
      RETURN
 1000 KENN=-1
      RETURN
 1002 KENN=1
      RETURN
      END
```

```
      SUBROUTINE DEFINE
      RETURN
      END
```