UNIVERSITY OF ILLINOIS
DIGITAL COMPUTER

ILLINOIS CODE 123-KA1

TITLE	Polynomial Approximation
TYPE	Complete Program
ACCURACY	About one part in 10^{18}
DURATION	Roughly $20 + [2 + (n/4)]^2$ seconds
READ AROUND	60

DESCRIPTION

This program prints the coefficients of a polynomial which approximates to the function $k\, f(ax + b)$ with error not greater than e over the range $c \leqslant x \leqslant d$.

The function f is defined by the coefficients f_o, f_1,... of the Taylor Series.

$$f(y) = f_o + f_1 y + f_2 y^2 + ,...$$

These coefficients may be either given numerically or generated by an auxiliary subroutine. Any number of coefficients up through $f_{100} y^{100}$ may be given.

Code 87 (1.7 Precision Floating Binary Arithmetic) is used throughout.

OPERATING PROCEDURE

Machine stops after reading program tape. Then insert data tape (see below) and raise black switch.

There is a sum check at the end of the program tape.

NOTES

(1) Enough Taylor coefficients must be given to define the function with at least the required accuracy. The residue must remain small compared with e/k for values of $|y|$ up to max. ($|ac + b|$, $|ad + b|$). If in doubt, give more coefficients.

(2) The first number to be printed (to 5 places) is an upper limit to the error in the approximation (this error may not be reached anywhere in the range, but the actual error will not be much smaller). The program makes this number as large as possible without exceeding e, by choosing the order of the approximating polynomial.

This number is followed by the coefficients of the polynomial, beginning with the constant term, each to

18 places. Numbers are printed in floating decimal form with exponent last.

(3) Remember that the error given is the truncation error in replacing an infinite series by a polynomial. In applying the polynomial, rounding errors will also arise.

(4) This program has been found to suffer from a serious accumulation of rounding errors if

$$(1/20) < (c/d) < 20$$

and should not be used in such cases.

(5) There is no internal check in this program. Results should be checked by computing one or two spot values of the function.

DATA TAPE

Punch first the parameters k, a, b, c, d, e in that order, in the form specified for Code A4 (sign, decimal digits up to 23; sign of exponent, three digit exponent). Then punch n, the order of the highest Taylor coefficient given (n+1 coefficients altogether), with L.H. zeros omitted, and followed by the symbol F. Then proceed as follows:

(a) If Taylor coefficients are given numerically. Punch 26 389N followed by the coefficients, punched as for Code A4, beginning with the constant term. or

(b) If Taylor coefficients are generated by auxiliary subroutine.

Punch directive (e.g. 00 16K) to locate first word of auxiliary subroutine, followed by the subroutine, followed by a directive (e.g. 26 16N) to transfer control to the subroutine.

AUXILIARY SUBROUTINE

(if used).

Locations up through 395 are available to hold or be used by the auxiliary subroutine. It should place the Taylor coefficients as follows:

f_0 in 1022,-23
f_1 in 1020,-21 — in floating binary form.
f_2 in 1018,-19 etc.,

and finally transfer control by the order 22 396F.

The following preset parameters may be used, having been set by the program tape:

S3: working positions for Code 87
S4: start of Code 87
S6: location of k
a, b, c, d, e are stored in 2S6, 4S6, 6S6, 8S6, 10S6.

Other preset parameters may be set and used if desired.

Code 87 may be used. Further numbers may be read (using Code 87); if so, they should be punched at the end of the tape.

METHOD

The function $T_r(x) = \cos(r \cos^{-1} x)$ can be written as a polynomial of degree r in x, known as a Tchebyscheff polynomial. The maximum absolute value of this function in the range $-1 \leqslant x \leqslant +1$ is 1; of all polynomials of degree r satisfying this condition, $T_r(x)$ has the largest coefficient of x^r.

In this routine we are concerned with the range $c \leqslant x \leqslant d$ so we take

$$T_r(x) = \cos[r \cos^{-1} [(2x - c - d)/(d-c)]]$$

which is obtained simply by a linear transformation of the argument.

Suppose we have a polynomial of degree n (or n+1 terms of a Taylor series),

$$P_n(x) = \sum_{p=0}^{n} a_p x^p,$$

and consider the polynomial

$$P'(x) = P_n(x) - (a_n/t_{nn}) T_n(x),$$

where t_{nn} is the coefficient of x^n in $T_n(x)$. Clearly $P'(x)$ is of degree n-1; moreover it differs from $P_n(x)$ (over the range considered) by at most (a_n/t_{nn}). If this quantity is less than e, the permitted tolerance, we can thus reduce the degree of the polynomial by 1. Using $T_{n-1}(x)$ it may then be possible to reduce the degree still further; in fact we can proceed until the sum of the moduli of the coefficients of the Tchebyscheffs would become greater than e.

The Tchebyscheffs are generated by using the recurrence relation

$$T_{r+1} = 2T_1T_r - T_{r-1}$$

starting with T_o and T_1 and proceeding to T_n. It is only necessary to hold two polynomials in the store simultaneously (apart from T_1); hence two sets of locations are reserved to hold coefficients of terms in even and odd Tchebyscheffs respectively.

When the Tchebyscheffs are being used to modify the given polynomial it is necessary to re-derive lower order Tchebyscheffs by the relation

$$T_{r-1} = 2T_1T_r - T_{r+1}.$$

Owing to the way the Tchebyscheffs are stored, this is done by exactly the same orders that generate the Tchebyscheffs.

The main routine that carries out the above operations is preceded by a preliminary routine which makes the shift of origin and change of scale implied in the parameters k, a and b.

END CORRECTION

The process normally ends when a further reduction of degree would make the sum of the moduli of the coefficients of the Tchebyscheffs greater than e; the current value of that sum is then given as an upper bound to the error. This is not quite the most efficient process, for the following reasons. The actual error function is the sum of the Tchebyscheffs removed. The peaks of these Tchebyscheffs do not all coincide, so that the peaks of the error function will vary in height and will in general be smaller than the quoted maximum error (they cannot be greater). The ideal error function would have all its peaks the same height.

In practice this effect is not often very important because the error function is dominated by the last Tchebyscheff to be removed; the others are usually much smaller and represent only minor deviations from it. Hence the peaks of the error function do not normally differ much in height. Moreover the ideal error function can be achieved only by very laborious computation. However, a slight improvement can sometimes be made quite simply, and if this enables a further reduction of degree to be made, this code does it.

It is assumed that the Tchebyscheff second in importance is that next higher in degree to the "dominant" (lowest degree) Tchebyscheff removed. This correction can be made only if the coefficient of the second Tchebyscheff is is less than half that of the dominant one, in modulus. It aims at reducing the effect of the second Tchebyscheff on the peaks of the dominant (similar corrections could be made for the other Tchebyscheffs in the error function, but these are less likely to be important and are more difficult to program).

Let $c_r T_r$ be the dominant Tchebyscheff and $c_{r+1} T_{r+1}$ the next higher Tchebyscheff in the error function. Simple theory yields a maximum error of $|c_r| + |c_{r+1}|$ arising from these. Suppose we now add $-c_{r+1} T_{r-1}$ to the error function. This alters the approximating polynomial but does not lower its degree. However, the error function (from these sources) is now

$$c_r T_r + c_{r+1} T_{r+1} - c_{r+1} T_{r-1}$$

$$= c_r \cos (r \cos^{-1} x) + c_{r+1}[\cos ((r+1)\cos^{-1} x) - \cos ((r-1)\cos^{-1} x)]$$

$$= c_r \cos(r \cos^{-1} x) - 2c_{r+1} \sqrt{(1 - x^2)} \sin (r \cos^{-1} x).$$

The peaks in the dominant occur when $|\cos (r \cos^{-1} x)| = 1$. For such values of x, $\sin(r \cos^{-1} x) = 0$; hence the second term has only a second order effect on the height of the peaks. To obtain an upper bound to the error, replace $\sqrt{(1 - x^2)}$ by 1; this yields the value

$$\sqrt{(c_r^2 + 4c_{r+1}^2)} \leq |c_r| + 2c_{r+1}^2/|c_r| .$$

The latter is now taken as the upper bound from these sources.

Before a Tchebyscheff is removed from the given polynomial, a test is made to see whether, on the simple theory, the error would become too large. If it would not, the Tchebyscheff is removed and the test is repeated on the Tchebyscheff of next lower degree. If it would, a further test is made to see whether the correction described above would bring the error within the prescribed limit.

DATE December 8, 1953
CODED BY S. G. C.
APPROVED BY [signature]

LOCATION	ORDER		NOTES
Decimal Order Input			
	00 3K		
3	00 F		work space for Code 87
	00 793F		
4	00 F		
	00 514F		Code 87
5	00 F		
	00 F		Preset
6	00 F		Parameters
	00 805F		parameters
7	00 F		
	00 817F		constants etc.
8	00 F		constant term of even
	00 210F		Tchebyscheffs
9	00 F		constant term of odd
	00 414F		Tchebyscheffs.
	00 383K		
0	00 F		
	50 L		
1	26 S4		
	8K 10F		
2	08 10S6		read parameters
	8F 2F		
3	8J 4L		
	00 F		
4	L5 6L		Use D.O.I. to read n into 2S7
	42 1016F		
5	41 1F		
	26 1009F		
6	00 39F		Clear Q
	L1 2S7		Set constant for counting
7	00 21F		in 26L
	L4 28L		
8	46 26L		
	50 8L	From 13	

LOCATION	ORDER		NOTES	
9	26 S4			
	8L 10L		waste	
10	88 (1022)F	By 12		
	8J 11L			Read
11	L5 10L			coefficients
	L0 S7			
12	46 10L			
	L0 26L			
13	32 8L			
	00 39F		Clear Q	
14	L1 2S7			
	00 1F			
15	L4 28L			
	42 22L			
16	04 (20)F	By 17'	Becomes 04 (1023 -2n)F	
	46 3S7			
17	L4 4S7			
	46 16L			
18	L5 24L			
	L0 S7			
19	40 24L		Change addresses	
	L0 16L		and count	
20	46 23L			Shift origin
	32 21L			
21	22 26L			
	50 21L	From 20'		
22	26 S4			
	85 ( )F	By 15'	Becomes 85(1022-2n)F	
23	8K ( )F	By 20		
	87 4S6		inner cycle	
24	04 (1024)F	By 19		
	0S (1024)F			
25	8F 2F			
	8J 18L			
26	88 ( )F	By 8	Becomes 88 (1022 - 2n)F	
	50 26L	From 21		

LOCATION	ORDER		NOTES	
27	26 S4			
	85 S6		ka^r	Scale factors in argument and function
28	87 (1022)F	By 33		
	8S (1022)F			
29	8J 32L			
	85 S6		ka^r	
30	87 2S6		a	
	8S S6		ka^{r+1}	
31	8L 28L			
	00 F			
32	L5 28L	From 29	Change addresses	
	L0 S7		and	
33	40 28L		Count	
	L0 3S7			
34	36 38S4			
	50 2S7		Set N(4S7)	
35	00 60F			
	46 4S7		= 8K 2nF	
	00 419K			
0	(22 L)	By 84	Waste	
	(49 (8)F	By 2' From 3		Clear space for Tchebyscheffs
1	70 F		Padding	
	70 F		for RAR	
2	FJ L			
	42 L			
3	32 L			
	50 3L			
4	26 S4			
	85 8S6		Set T_0 and T_1.	
5	80 6S6			
	8S F			
6	86 F		Set N(0) = -(c+d)/2	
	8S S8		N(2) = 4/(d-c)	
7	84 S8			
	8S 4F		N(6) = e	(remaining tolerance)

LOCATION	ORDER	NOTES
8	86 F	
	8K 4094S9	
9	83 4094S9	
	8S 2F	
10	81 6S6	
	80 8S6	
11	86 4F	
	8S F	
12	86 4094S9	
	8S S9	
13	85 10S6	
	8S 6F	
14	8K (4)F	From 23' and 69' By 41'
	85 F	
15	07 S9	
	04 2S9	Form even Tchebyscheff
16	87 2F	
	00 S8	
17	0S S8	
	8F 2F	
18	8J 24L	
	8L 19L	Waste
19	8K ( )F	From 61' By 26'
	85 F	
20	07 S8	
	04 2S8	
21	87 2F	Form odd Tchebyscheff
	00 S9	
22	0S S9	
	8F 2F	
23	8J 39L	
	8L 14L	
24	L5 14L	From 18
	L0 4S7	
25	32 38L	
	L5 S7	

LOCATION	ORDER		NOTES
26	L4 14L	From 71'	Change degree of odd T and go to form it.
	46 19L		
27	46 45L		
	46 66L		
28	26 38S4		
	50 28L	From 25, 74'	
29	26 S4		
	8L (30)L	By 79'	(Waste)
30	8K ( )F	By 42	
	01 1022F		Form even coefficient
31	06 S8		
	8S 4S8		
32	85 6F		
	82 4S8		? O.K. to use even T
33	8S 2S7		
	83 57L		
34	81 4S9		
	8S 4S9		? O.K. if end correction is applied
35	86 4S8		
	8S 4F		
36	86 4F		
	82 4F		
37	82 4F		
	83 54L		
38	8J 82L		Go to print
	00 F		Waste
39	L5 19L	From 23	
	L0 4S7		? use odd T
40	32 43L		
	L5 S7		
41	L4 19L	From 72'	Change degree of even T and go to form it.
	46 14L		
42	46 30L		
	46 58L		
43	26 38S4		
	50 43L	From 40, 76'	

LOCATION	ORDER	NOTES	
44	26 S4		
	8L (45)L	By 77'	
45	8K ( )F	By 27	Form odd coefficient
	01 1022F		
46	06 S9		
	8S 4S9		
47	85 6F		? O.K. to use odd T
	82 4S9		
48	8S 2S7		
	83 65L		
49	81 4S8		? O.K. if end correction is applied
	8S 4S8		
50	86 4S9		
	8S 4F		
51	86 4F		
	82 4F		
52	82 4F		
	83 62L		
53	8J 83L		Go to print
	00 F		Waste
54	87 4S9	From 37'	? correction O.K.
	8S 4F		
55	81 2S7		
	82 4F		
56	83 38L		
	8J 77L		Prepare for end
57	8S 6F	From 33'	
	8L 58L	From 29'	Waste
58	8K ( )F	By 42'	Use even T
	85 4S8		
59	07 S8		
	04 1022F		
60	0S 1022F		
	8F 2F		
61	8J 70L		

LOCATION	ORDER		NOTES
	8L 19L		
62	87 4S8	From 52'	
	8S 4F		
63	81 2S7		? correction O.K.
	82 4F		
64	83 53L		
	8J 79L		Prepare for end
65	8S 6F	From 48'	
	8L 66L	From 44'	Waste
66	8K ( )F	By 27'	
	85 4S9		
67	07 S9		Use odd T
	04 1022F		
68	0S 1022F		
	8F 2F		
69	8J 71L		
	8L 14L		
70	L1 S7	From 61	
	22 71L		
71	(26 73L)	From 69	Becomes 26 72L, 26 83L By 73',75',80'
	(26 75L)	From 70'	Becomes 2626L, 26 82L
72	L1 S7	From 71	
	26 41L		
73	L5 92L	From 71	Set link to form lower T's
	40 71L		
74	49 4S7		Immobilize T count
	22 28L		
75	L5 92L	From 71'	Set link to form lower T's
	40 71L		
76	49 4S7		Immobilize T count
	22 43L		
77	L5 65L	From 56'	Set to skip formation of next odd coefficient
	42 44L		
78	L5 93L		
	22 80L		

LOCATION	ORDER		NOTES
79	L5 57L	From 64'	Set to skip formation of next even coeff.
	42 29L		
80	L5 94L		
	40 71L	From 78'	Set to end after next T
81	0L 1022 F		
	26 38S4		
82	L5 14L	From 38, 71'	End after Forming even T
	22 83L		
83	L5 19L	From 53, 71	End after forming odd T
	L0 81L		
84	40 L		Set count constant for printing
	50 84L		
85	26 S4		
	85 10S6		Form maximum error
86	82 6F		
	89 5F		Print error
87	85 (1022)F	By 90 From 88'	
	89 18F		Print terms
88	8J 89L		
	8L 87L		
89	L5 87L	From 88	
	L0 S7		
90	46 87L		Change address in 87 and count
	L4 L		
91	36 38S4		
	OF F		Stop
92	26 72L		
	26 26L		
93	26 83L		
	26 26L		
94	26 72L		
	26 82L		
	00 514K		
	CODE 87		

LOCATION	ORDER		NOTES
	00 817K		
0S7	00 2F		
	00 2F		
1S7	88 F		
	00 F		
2S7	00 F		
	00 ( )F		Becomes 00 nF
3S7	87 ( )F		Becomes 87 (1022 - 2n)F
	00 F		
4S7	(8K 1F)		Becomes 8K 2nF; 40 F
	00 F		(used to count T's)
	00 16K		
	CODE 108		Sum check
	20 383N		

DATE December 8, 1953
CODED BY S. Gill
APPROVED BY J. Nash