SWAC Experiments on the Use of Orthogonal Polynomials for Data Fitting*

MARCIA ASCHER

Douglas Aircraft Company, Santa Monica, California

AND

GEORGE E. FORSYTHE

Stanford University, Stanford, California

1. *Introduction and Summary*

Let $\xi = (x_1, \cdots, x_m)$ be a set of real numbers, and let $f(x_1), \cdots, f(x_m)$ be the observed or computed values of some function of x at the points of ξ. Let $f_n^{(\xi)}(x)$ be the unique polynomial of degree n which best fits the data values $f(x_i)$ in the least-squares sense. I.e., the sum

$$S(q_n) = \sum_{i=1}^{m} \{f(x_i) - q_n(x_i)\}^2,$$

where q_n is a polynomial of degree n, is minimized by $q_n(x) = f_n^{(\xi)}(x)$.

In [3] one of us discussed the problem of finding $f_n^{(\xi)}(x)$ on a digital computer and outlined certain advantages of determining $f_n^{(\xi)}(x)$ in terms of polynomials $p_k(x)$ orthogonal over ξ. It was further proposed that the $p_k(x)$ be computed from their 3-term recurrence. The procedure will be outlined again here, mainly to formulate our notation.

Let $p_0(x) = 1$, and let $\beta_0 = 0$. Then the polynomials $p_k(x)$ are defined and computed by the recurrence

$$p_{k+1}(x) = (x - \alpha_{k+1})p_k(x) + \beta_k p_{k-1}(x). \tag{1}$$

Here

$$\alpha_{k+1} = \sum_{i=1}^{m} x_i\{p_k(x_i)\}^2 \Big/ \sum_{i=1}^{m} \{p_k(x_i)\}^2, \tag{2}$$

and

$$\beta_k = \sum_{i=1}^{m} \{p_k(x_i)\}^2 \Big/ \sum_{i=1}^{m} \{p_{k-1}(x_i)\}^2. \tag{3}$$

(Our numbering of the subscripts of the α_k and β_k agrees with that in [3], but various other numberings are in use elsewhere.)

The fundamental property of the $p_k(x)$ is the orthogonality relation

* Received August, 1957. This paper was prepared at the University of California, Los Angeles, sponsored by the Office of Naval Research, U. S. Navy. Reproduction in whole or in part is permitted for any purpose of the United States Government.

$$\sum_{i=1}^{m} p_h(x_i)p_k(x_i) = 0 \qquad (h \neq k).$$

The Fourier coefficients t_k of $f(x_i)$ are defined by

$$t_k = \sum_{i=1}^{m} f(x_i)p_k(x_i) \Big/ \sum_{i=1}^{m} \{p_k(x_i)\}^2. \tag{4}$$

In terms of the t_k we can write the desired least-squares polynomial as

$$f_n^{(\xi)}(x) = \sum_{k=0}^{n} t_k\, p_k(x). \tag{5}$$

Finally, the statistic (estimate of unexplained variance)

$$\sigma_n^2 = (m - n - 1)^{-1} \sum_{i=1}^{m} \{f(x_i) - f_n^{(\xi)}(x_i)\}^2 \tag{6}$$

has an expected value which is independent of n (for $n > h$), if the $f(x_i)$ are independently normally distributed about some polynomial trend of degree h. Hence σ_n^2 is often a useful measure of the goodness with which $f_n^{(\xi)}(x)$ fits $f(x)$ over the set ξ.

When one is asked to use a computer to find the polynomial $f_n^{(\xi)}$, one can well ask in return what it means to "find a polynomial." Too often one automatically considers that a polynomial p is synonymous with its representation in powers

$$p(x) = c_0 + c_1 x + \cdots + c_n x^n$$

or, at any rate, with the coefficients

$$c_0, c_1, \cdots, c_n. \tag{6'}$$

It is our point of view that one may equally well have "found a polynomial p" when one has a table of its values $p(x)$ for a sufficiently wide class of arguments x. Or, alternatively, that one may have "found p" when one knows the constants of a machine algorithm for computing $p(x)$ for any desired x. It is in the latter sense that we claim we have "found $f_n^{(\xi)}$" as soon as we have the values of

$$\alpha_1, \cdots, \alpha_n, \quad \beta_1, \cdots, \beta_{n-1}, \quad \text{and} \quad t_0, \cdots, t_n. \tag{7}$$

For with these numbers we can produce $f_n^{(\xi)}(x)$ for any desired x by formulas (1) and (5).

It is a deeper question as to which representation of $f_n^{(\xi)}$ is the better—that of the coefficients (6′) or that of (7). Such a question can only be answered in terms of computing time, precision, round-off error, machine storage, and especially any desired further use of $f_n^{(\xi)}$. Ordinarily, choices between computing methods depend critically on special factors characteristic of a given particular application.

Without claiming the method to be better, the authors have experimented with the consistent use of the recurrence (1) for the data-fitting problem on the automatic digital computer available to them, SWAC. We have prepared this summary of typical results as a record of our experiment.

A related code has been reported by Rudin [5].

A similar code, but in fixed point, has been constructed at The Ramo-Wooldridge Corporation under the guidance of David Morrison.

In section 2 we describe the SWAC codes, which use "floating vectors." In section 3 we give formulas for the Chebyshev polynomials which are orthogonal with respect to summation over certain equally spaced x_i, and use them to check the SWAC-generated values of α_k and β_k. In section 4 we test the code for generating $f_n^{(\xi)}(x)$ for the function $f(x) = |x|$ over the same x_i.

In section 5 we prove an apparently new theorem that, if f is an entire function, $f_n^{(\xi)}(z) \to f(z)$ for all complex z, as $n, m \to \infty$. In section 6 we compare a bound given by our theorem with the observed error of $f_n^{(\xi)}(x) - f(x)$, for various real values of x, where $f(x) = e^x$. Table 3 and graphs 1 and 2 illustrate the quantitative and qualitative behavior of both the interpolation and extrapolation of e^x by use of $f_n^{(\xi)}(x)$.

We conclude that for e^x our method of data fitting will determine interpolated function values $f_n^{(\xi)}(x)$ with a precision of something like 10^{-8} for n up to 16, and with nearly as much precision up to $n = 31$. This is in contrast with reports we have heard of the failure of routines which have attempted (using comparable precision) to solve the classical normal equations for the power coefficients $c_0, \cdots, c_n$, and then to evaluate $f_n^{(\xi)}(x)$ in the form $c_0 + c_1x + \cdots + c_nx^n$.

2. *SWAC Codes*

The coding of the procedure was divided into two parts. In the first routine the input is the set of abscissas $\xi = \{x_1, \cdots, x_m\}$ $(m \leqq 1023)$ and the corresponding functional values $f(x_i)$ $(i = 1, \cdots, m)$, with no restriction on the interval or spacing. The routine generates: (i) the parameters α_k, β_k of (2, 3) defining the orthogonal polynomials $p_k(x)$ of each successive degree k up to 32 (if $m > 32$), or up to $m - 1$ (if $m \leqq 32$); (ii) the Fourier coefficients t_k of (4) for the values $f(x_i)$ with respect to these polynomials; and (iii) the error measures σ_k^2 of (6). In the second routine, the results of the first routine (α_k, β_k, t_k) and the abscissas $d_1, \cdots, d_r$ are the input. This routine evaluates

$$f_n^{(\xi)}(d_i) = \sum_{k=0}^{n} t_k\, p_k(d_i) \qquad (\text{for } 1 \leqq i \leqq r, \quad 0 \leqq n \leqq 32).$$

The d_i need not be the same as the x_i nor in the same interval. The number of abscissas has the arbitrary limit $r \leqq 1023$.

The routines use numbers in a *(binary) floating vector* form. Each vector $(z_1, \cdots, z_m)$ is stored in $m + 1$ cells as $(q_1, \cdots, q_m, q_{m+1})$, which is interpreted as

$$z_i = q_i \cdot 2^{q_{m+1}}.$$

Here q_{m+1} is an integer such that $|q_{m+1}| \leqq 2^{36} - 1$; we have $|q_i| < 1$ $(i = 1, \cdots, m)$, and $\frac{1}{2} \leqq \max_{1 \leqq i \leqq m} |q_i| < 1$. Each scalar is regarded as a floating vector with one component, and so is stored in two cells. This floating vector form has been used previously on SWAC by Professor M. R. Hestenes in several matrix codes.

It was found that a normalization process to keep $\frac{1}{2} \leqq \max_{1 \leqq i \leqq m} | q_i | < 1$ is extremely important for the accuracy of the results. This normalization is needed after each arithmetic step involving either vectors or scalars.

Because of the number of times the operations are performed, the use of subroutines for subtraction of vectors, division of scalars, and finding the inner product of two vectors is found to be very useful.

Although the numbers are input and are available for output in binary floating vector form, it was thought advisable also to have the output in a more easily identifiable form. The routines therefore contain a subroutine to convert the data into floating decimal form, and the numbers contained in this paper are a result of this conversion. The error introduced into the numbers by this conversion is believed to be not more than approximately one unit in the last digit tabulated. This belief is substantiated by test conversions of several numbers over the range of numbers tabulated.

The routines and further details about them are available in the Numerical Analysis Research library of the University of California, Los Angeles (codes 00474.1 and 00474.2).

In evaluating our results the reader should know that the SWAC word length is 36 binary digits (equivalent to about 10.8 decimals) plus a sign digit.

3. *Equally Spaced* x_i

As a first test for the routines we took an odd number m of equally spaced points over the interval $[-1, 1]$:

$$x_i = -1 + 2\frac{i-1}{m-1} \qquad (i = 1, 2, \cdots, m).$$

The corresponding orthogonal polynomials

$$P_k^{(m)}(x) = x^k + \cdots \qquad (k = 0, 1, \cdots, m-1)$$

were introduced by Chebyshev [2], and the following formulas are adapted from Barker [1]:

$$P_{k+1}^{(m)}(x) = x\, P_k^{(m)}(x) - \beta_k^{(m)} P_{k-1}^{(m)}(x), \tag{8}$$

where

$$\beta_k^{(m)} = \frac{k^2}{(m-1)^2}\frac{m^2-k^2}{4k^2-1}. \tag{9}$$

Comparison of (8) with (1) shows that all

$$\alpha_k^{(m)} = 0. \tag{10}$$

TABLE 1

k	$m = 33$: True $\beta_{k-1}^{(33)}$	$m = 33$: SWAC computed $\beta_{k-1}^{(33)}$	$m = 33$: SWAC* computed $\alpha_k^{(33)}$	$m = 513$: True $\beta_{k-1}^{(513)}$	$m = 513$: SWAC computed $\beta_{k-1}^{(513)}$	$m = 513$: SWAC* computed $\alpha_k^{(513)}$
1	0	0	10^{-22} .25667 66	0	0	10^{-23} .16511 36
2	.35416 6667	.35416 6666	10^{-22} .72473 40	.33463 5416	.33463 5416	10^{-23} .49341 35
3	.28255 2083	.28255 2083	10^{-22} .64123 93	.26770 5281	.26770 5281	10^{-23} .46078 05
4	.27120 5357	.27120 5357	10^{-22} .59110 13	.25813 9474	.25813 9474	10^{-23} .44625 15
5	.26612 1031	.26612 1031	10^{-22} .55529 36	.25494 5785	.25494 5785	10^{-23} .43759 45
6	.26238 9520	.26238 9520	10^{-22} .52907 38	.25348 8560	.25348 8559	10^{-23} .43157 22
7	.25887 7841	.25887 7840	10^{-22} .51092 99	.25269 8031	.25269 8031	10^{-22} .17078 57
8	.25520 8333	.25520 8333	10^{-22} .50050 28	.25221 7610	.25221 7610	10^{-22} .16928 41
9	.25122 5490	.25122 5490	10^{-22} .49806 13	.25190 0467	.25190 0467	10^{-22} .16800 69
10	.24685 5650	.24685 5650	10^{-22} .50440 54	.25167 7049	.25167 7049	10^{-22} .16688 74
11	.24206 0229	.24206 0228	10^{-22} .52095 03	.25151 0926	.25151 0926	10^{-22} .16588 48
12	.23681 7417	.23681 7417	10^{-22} .54994 93	.25138 1505	.25138 1504	10^{-22} .16497 32
13	.23111 4130	.23111 4130	10^{-22} .59488 94	.25127 6431	.25127 6430	10^{-22} .16413 52
14	.22494 2129	.22494 2129	10^{-22} .66115 82	.25118 7925	.25118 7925	10^{-22} .16335 89
15	.21829 6016	.21829 6016	10^{-22} .75718 08	.25111 0891	.25111 0890	10^{-22} .16263 63
16	.21117 2135	.21117 2135	10^{-22} .44820 12	.25104 1875	.25104 1874	10^{-22} .16196 13
17	.20356 7937	.20356 7937	10^{-22} .27521 59	.25097 8472	.25097 8471	10^{-22} .16132 99
18	.19548 1602	.19548 1601	10^{-22} .35197 17	.25091 8962	.25091 8962	10^{-22} .16073 90
19	.18691 1800	.18691 1800	10^{-22} .47077 24	.25086 2092	.25086 2092	10^{-22} .16018 66
20	.17785 7545	.17785 7545	10^{-22} .33086 34	.25080 6929	.25080 6928	10^{-22} .15967 12
21	.16831 8089	.16831 8089	10^{-22} .24571 29	.25075 2766	.25075 2766	10^{-22} .15919 19
22	.15829 2860	.15829 2860	10^{-22} .38806 69	.25069 9065	.25069 9065	10^{-22} .15874 80
23	.14778 1411	.14778 1411	10^{-22} .32824 40	.25064 5403	.25064 5403	10^{-22} .15833 92
24	.13678 3392	.13678 3392	10^{-22} .29996 70	.25059 1449	.25059 1449	10^{-22} .15796 55
25	.12529 8524	.12529 8523	10^{-22} .29925 23	.25053 6939	.25053 6939	10^{-22} .15762 70
26	.11332 6581	.11332 6580	10^{-22} .16503 87	.25048 1662	.25048 1662	10^{-22} .15732 39
27	.10086 7381	.10086 7381	10^{-22} .40904 87	.25042 5445	.25042 5445	10^{-22} .15705 66
28	.08792 0776	.08792 0776	10^{-22} .29077 93	.25036 8147	.25036 8147	10^{-22} .15682 56
29	.07448 6643	.07448 6643	10^{-22} .24398 61	.25030 9653	.25030 9653	10^{-22} .15663 16
30	.06056 4879	.06056 4883	10^{-22} .25178 18	.25024 9866	.25024 9866	10^{-22} .15647 53
31	.04615 5399	.04615 5463	10^{-22} .17047 12	.25018 8706	.25018 8706	10^{-22} .15635 72

* The true $\alpha_k^{(m)} = 0$ for all k and any odd m.

Incidentally, as $m \to \infty$, the above polynomials approach the Legendre polynomials $P_k(x)$ except for normalization:

$$\lim_{m \to \infty} P_k^{(m)}(x) = \frac{2^k (k!)^2}{(2k)!} P_k(x).$$

Moreover,

$$\lim_{m\to\infty} \beta_k^{(m)} = \frac{k^2}{4k^2 - 1}.$$

The exact formulas (9) and (10) give a control on the round-off error of any machine calculations of the α's and β's.

For the SWAC experiments we selected $m = 33$ and $m = 513$, to avoid rounding the x_i. The corresponding computed values of the $\alpha_k^{(m)}$ and $\beta_k^{(m)}$ are shown in table 1. For each m and k the first column gives $\beta_k^{(m)}$, computed by hand from (9). The second and third columns show the values of $\beta_k^{(m)}$ and $\alpha_k^{(m)}$ computed and converted by SWAC. It will be observed that there is practically no round-off error in any of the α's or β's. The errors in the α's are larger for $m = 33$ than for $m = 513$. The errors in $\beta_{29}^{(33)}$ and $\beta_{30}^{(33)}$ are relatively large, suggesting a considerable growth in the round-off error of $\beta_k^{(m)}$ as k becomes practically equal to m.

4. *Fitting the Function* $|x|$

Besides generating the coefficients of the orthogonal polynomials, the SWAC routines will fit given functional values $f(x_i)$ by computing $f_n^{(\xi)}(x)$ for selected values of x according to (1) and (5).

With the equally spaced abscissas of section 3, the function $f(x) = |x|$ was

TABLE 2

n *	$m = 513$		$m = 33$
	$f_n^{(\xi)}(0)$ and $f_{n-1}^{(\xi)}(0)$	$f_n^{(\xi)}(\pm 1)$ and $f_{n+1}^{(\xi)}(\pm 1)$	$f_n^{(\xi)}(\pm 1)$ and $f_{n+1}^{(\xi)}(\pm 1)$
0	.50097 4658	.50097 4658	.51515 1515
2	.18786 192	1.12354 510	1.10389 610
4	.11740 9689	.93930 2789	.96068 7960
6	.08560 700	1.03698 553	1.01638 001
8	.06741 1158	.97495 4899	.99328 8187
10	.05560 976	1.01802 959	1.00257 546
12	.04733 0003	.98650 8461	.99910 0631
14	.04119 791	1.01035 865	1.00027 985
16	.03647 2742	.99190 6275	.99992 3845
18	.03271 954	1.00640 028	1.00001 776
20	.02966 6051	.99489 7666	.99999 6524
22	.02713 304	1.00408 900	1.00000 054
24	.02499 7740	.99671 2848	.99999 9930
26	.02317 317	1.00264 634	.99999 9947
28	.02159 6012	.99786 9208	1.00000 055
30	.02021 906	1.00171 408	.99999 2041

selected. Table 2 shows the output for $n = 0(1)31$ and for $x = 0$ and ± 1. (In each case the computed values $f_n^{(\xi)}(+1)$ and $f_n^{(\xi)}(-1)$ were identical.)

The routine found, while fitting points on this curve, that no odd degree could improve upon the preceding even degree. Therefore $f_n^{(\xi)}(x) = f_{n+1}^{(\xi)}(x)$ for each even n.

We have not computed correct values with which to deduce the round-off errors in the values of table 2. The values look plausible, and we conclude that a least-squares fit is perfectly possible up to degree 31 by our methods. This is decidedly in contrast with the failure of ordinary least-squares routines to compute even the coefficients $c_0, \cdots, c_n$ of (6′) without use of multiple-precision arithmetic, for n larger than, say, 10.

The slowness of the observed convergence of $f_n^{(\xi)}(0)$ to $f(0) = 0$ is due to the discontinuity of $f'(x)$ at $x = 0$. It is conjectured that, as $n \to \infty$ (with $m > n$), $|f_n^{(\xi)}(0) - f(0)| = O(1/n)$, independently of m.

5. *A Theorem on Extrapolation*

We were interested in using the curve-fitting routine to extrapolate some functions computed for a few points x in an interval $[-1, 1]$ to abscissas like $x = 2$. To test such a general process of extrapolation we were unable to use the computations of section 4 with the function $f(x) = |x|$. The discontinuity of $f'(x)$ at $x = 0$ apparently prevents $f_n^{(\xi)}(x)$ from approximating $f(x) = |x|$ at any point x outside the interval $[-1, 1]$.

On the other hand, if $f(z)$ is an entire function of the complex variable z, the following apparently new theorem shows that extrapolation should be easily possible. The theorem can be stated under weaker hypotheses, but it suffices for our purposes.

THEOREM. *Let $\xi = \{x_1, \cdots, x_m\}$ be an arbitrary set of m distinct points of the closed interval $[-1, 1]$. For $n < m$ let $f_n^{(\xi)}(x)$ be the polynomial of degree n which most closely approximates the entire function $f(x)$ on ξ in the least-squares sense. I.e., $f_n^{(\xi)}(x)$ minimizes*

$$S(q_n) = \sum_{i=1}^{m} \{f(x_i) - q_n(x_i)\}^2$$

over the class of polynomials q_n of degree n.

Then, as $m \to \infty$, $n \to \infty$ independently $(n < m)$, $f_n^{(\xi)}(z) \to f(z)$ for all complex z.

Moreover, we have the following bound, depending on the interval, z, n and f, but not on m or ξ:

$$|f_n^{(\xi)}(z) - f(z)| \leqq \frac{M_{n+1}}{(n+1)!} \rho^{n+1}, \tag{11}$$

where M_{n+1} is the maximum of $|f^{(n+1)}(\zeta)|$ for ζ in the closed triangle T with base zeros! n *$[-1, 1]$ and vertex z, and where $\rho =$* max $\{|z + 1|, |z - 1|\}$.

PROOF. We first show that $f(x) - f_n^{(\xi)}(x) = R_n(x)$ has at least $n + 1$ distinct zeros in $[-1, 1]$. If not, then choose $u_0, \cdots, u_{q+1}$ and $\epsilon = \pm 1$ such that:

(i) $\quad -1 = u_0 < u_1 < \cdots < u_q < u_{q+1} = 1 \qquad (q \leq n);$

(ii) $\quad R_n(u_j) = 0 \qquad$ for $j = 1, 2, \cdots, q;$

(iii) $\quad (-1)^r \epsilon R_n(x_i) \geq 0$ for all x_i in (u_r, u_{r+1}), $\quad r = 0, \cdots, q;$

(iv) $\quad (-1)^r \epsilon R_n(x_i) > 0$ for at least one x_i in (u_r, u_{r+1}), $\quad r = 0, \cdots, q.$

Now define $r(x) = \prod_{j=1}^q (x - u_j)$. (If $q = 0$, let $r(x) = 1$.) Then $r(x_i)R_n(x_i) \geq 0$ or $r(x_i)R_n(x_i) \leq 0$ for all $i = 1, \cdots, m$, while, by (iv), $r(x_i)R_n(x_i)$ is non-zero for certain x_i. Consider the polynomials $g_\eta(x) = f_n^{(\xi)}(x) + \eta r(x)$, of degree n, for real values of η near 0.

For $\eta = 0$, a short calculation shows that

$$\frac{d}{d\eta} \sum_{i=1}^n \{f(x_i) - g_\eta(x_i)\}^2 = -2 \sum_{i=1}^n R_n(x_i) r(x_i),$$

which is non-zero by the choice of $r(x)$. Hence, for some η small enough in absolute value,

$$\sum_{i=1}^n \{f(x_i) - g_\eta(x_i)\}^2 < \sum_{i=1}^n \{f(x_i) - f_n^{(\xi)}(x_i)\}^2, \tag{12}$$

contradicting the definition of $f_n^{(\xi)}$. (Incidentally, only the continuity of f has been used so far.)

By the above, $f_n^{(\xi)}(x)$ interpolates $f(x)$ at some $n + 1$ distinct points $y_0, y_1, \cdots, y_n$ of $[-1, 1]$. We may therefore use Jensen's formula for the remainder in polynomial interpolation; see Nörlund [4, p. 9]. This states that

$$R_n(z) = \frac{(z - y_0) \cdots (z - y_n)}{(n + 1)!} \lambda f^{(n+1)}(\zeta), \tag{13}$$

where λ is a complex number with $|\lambda| \leqq 1$, and where ζ is a point in the interior of T.

Since all $|z - y_i| \leqq \max\{|z + 1|, |z - 1|\} = \rho$, the expression (11) is an upper bound for $|R_n(z)|$ in (13), and is thereby proved.

To prove that $f_n^{(\xi)}(z) \to f(z)$ we have to prove that $R_n(z) \to 0$. Since $f(z)$ is entire, we know that, for all ζ in T,

$$f^{(n+1)}(\zeta) = \frac{(n + 1)!}{2\pi i} \int_{C_R^{(\zeta)}} \frac{f(t)\, dt}{(t - \zeta)^{n+2}},$$

where $C_R^{(\zeta)}$ is a circle with radius R and center ζ. Hence

$$\frac{|f^{(n+1)}(\zeta)|}{(n + 1)!} \leqq \frac{2\pi R}{2\pi} \frac{m_R^{(\zeta)}}{R^{n+2}} = \frac{m_R^{(\zeta)}}{R^{n+1}}, \tag{14}$$

where $m_R^{(\zeta)} = \max |f(t)|$ for t on $C_R^{(\zeta)}$. It follows from (14) that

$$\frac{M_{n+1}}{(n + 1)!} \leqq \frac{m_R}{R^{n+1}}, \tag{15}$$

where $m_R = \max m_R^{(\zeta)}$ for all ζ in T.

Substituting (15) in (11) shows that

$$|f_n^{(\xi)}(z) - f(z)| \leqq \frac{m_R \rho^{n+1}}{R^{n+1}}. \tag{16}$$

Since $f(z)$ is entire, R can be picked with $\rho < R$, and hence (16) shows that $f_n^{(\xi)}(z) - f(z) \to 0$, as $n \to \infty$, completing the proof of the theorem.

6. *Fitting the Smooth Function* e^x

As a suitable entire function we chose $f(x) = e^x$ as being well tabulated and easy to work with. We confined our extrapolations mainly to real values of x in the interval $[-2.5, 2.5]$. From (11) we have the following upper bounds for the truncation error in extrapolation:

$$|f_n^{(\xi)}(x) - e^x| \leqq \frac{e}{(n+1)!}(1-x)^{n+1} \qquad (x \leqq 0); \tag{17}$$

$$|f_n^{(\xi)}(x) - e^x| \leqq \frac{e}{(n+1)!}(x+1)^{n+1} \qquad (0 \leqq x \leqq 1); \tag{18}$$

$$|f_n^{(\xi)}(x) - e^x| \leqq \frac{e^x}{(n+1)!}(x+1)^{n+1} \qquad (1 \leqq x). \tag{19}$$

Formulas (17, 18, 19) bound the errors in extrapolation, except for round-off error. In numerical experiments any errors exceeding the above bounds must be due to round-off.

Experiments were run for $m = 9, 17, 33$, and 66. The x_i were uniformly spaced except near $x = 0$. The following abscissas were used:

$m = 9$: $x_i = -.97(.24)-.01$ and $.22(.24).94$;

$m = 17$: $x_i = -.97(.12)-.01$ and $.10(.12).94$;

$m = 33$: $x_i = -.97(.06)-.01$ and $.04(.06).94$;

$m = 66$: $x_i = -.97(.03)-.01$ and $.01(.03).97$.

Define $f_n^{(\xi)}(x)$, where $\xi = \{x_1, \cdots, x_m\}$, as in section 1. SWAC computed $f_n^{(\xi)}(x)$ for the following values of m, n, and x:

$m = 9$: $n = 0(1)8$: $x = 10, \pm 2.5, \pm 2, \pm 1.5, \pm 1.25, \pm 1, -.97, +.94, \pm .75, \pm .5, 0$;

$m = 17$: $n = 0(1)16$: $x = 10, \pm 2.5, \pm 2, \pm 1.5, \pm 1.25, \pm 1, -.97, +.94, \pm .75, \pm .5, 0$;

TABLE 3

n	Interpolation			Extrapolation		
	$f_n^{(\xi)}(x)$ for x *inside* the interval used for fitting the curve			$f_n^{(\xi)}(x)$ for x *outside* the interval used for fitting the curve		
	$f_n^{(\xi)}(0)$, $m = 9$	$f_n^{(\xi)}(0)$, $m = 66$	$f_n^{(\xi)}(-.75)$, $m = 33$	$f_n^{(\xi)}(-1)$, $m = 66$	$f_n^{(\xi)}(2.5)$, $m = 17$	$f_n^{(\xi)}(2.5)$, $m = 66$
0	1.18292 820	1.16883 126	1.15145 16138	1.16883 126	1.16191 801	1.16883 126
1	1.19881 398	1.16883 126	.35472 61294	.06844 300	3.90129 464	3.91980 192
2	.99734 400	.99656 284	.47004 91874	.43166 859	7.06948 229	7.09437 110
3	.99573 659	.99656 284	.47522 44480	.35813 780	9.56183 91	9.58603 93
4	.99999 959	1.00002 751	.47197 87451	.36903 682	11.04819 09	11.06436 52
5	1.00002 800	1.00002 751	.47239 09500	.36776 506	11.75912 72	11.76831 78
6	1.00000 009	.99999 988	.47236 62030	.36788 905	12.04428 28	12.04838 26
7	.99999 994	.99999 988	.47236 64838	.36787 872	12.14233 54	12.14398 05
8	.99999 999	1.00000 000	.47236 65595	.36787 948	12.17202 89	12.17257 83
9		1.00000 000	.47236 65522	.36787 943	12.18006 53	12.18013 43
10		1.00000 000	.47236 65537	.36787 944	12.18075 34	12.18167 11
11		1.00000 000	.47236 65537	.36787 944	12.16825 32	12.18204 31
12		1.00000 000	.47236 65537	.36787 943	11.99238 69	12.17441 77
13		1.00000 000	.47236 65522	.36787 944	9.27375 37	12.11591 71
14		1.00000 000	.47236 65527	.36787 943	−47.61471 69	11.77668 24
15		1.00000 000	.47236 65522	.36787 943	−2256.17967	10.92267 68
16		1.00000 000	.47236 65537	.36787 944	−188735.581	7.90958 077
17		1.00000 000	.47236 65537	.36787 943		−32.55458 73
18		1.00000 000	.47236 65522	.36787 945		−392.38801 3
19		1.00000 000	.47236 65566	.36787 939		−3192.78756
20		1.00000 000	.47236 65406	.36787 960		−3 66581.036
21		1.00000 000	.47236 65711	.36787 938		−1 62225.069
22		1.00000 000	.47236 64809	.36787 990		−11 98469.73
23		1.00000 000	.47236 66774	.36787 908		−70 11899.96
24		.99999 999	.47236 59309	.36788 15		−685 01561.

25		.99999 999	.47236 82504	.36787 76		−4087 66999.
26		1.00000 000	.47235 45818	.36788 9		−38756 43230.
27		1.00000 000	.47242 14217	.36787 0		−2 31674 04900.
28		.99999 999	.47177 78167	.36793		−22 62200 21000.
29		.99999 999	.47705 85408	.3678		
30		1.00000 000	.38019 21455	.368		
31		1.00000 000	.51033 95480	.367		
32		.99999 997		.36		
	$e^0 = 1.00000000$	$e^0 = 1.00000000$	$e^{-.75} = .4723665527$	$e^{-1} = .367879441$	$e^{2.5} = 12.1824939607$	$e^{2.5} = 12.1824939607$

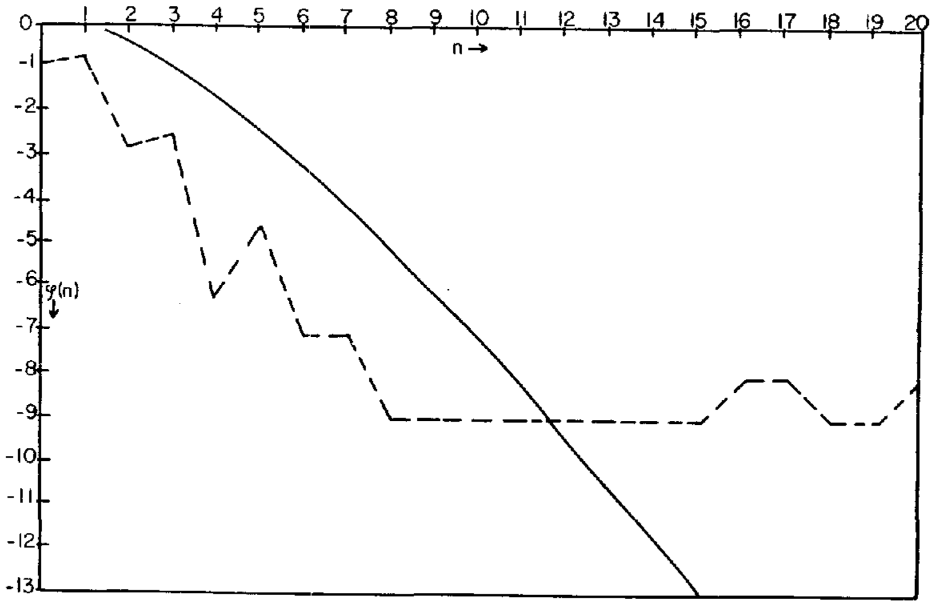
$m = 33$: $n = 0(1)32$: $x = 10, -1.5, -1, \pm.5, 0$;
$n = 0(1)31$: $x = -.97, +.94, \pm.75$;
$n = 0(1)29$: $x = 1.5, 1.25, 1$;
$n = 0(1)23$: $x = \pm 2.5, \pm 2, -1.25$;
$m = 66$: $n = 0(1)32$: $x = 10, -1.5, -1, -.97, +.94, \pm.75, \pm.5, 0$;
$n = 0(1)28$: $x = 1.5, 1.25, 1$;
$n = 0(1)27$: $x = 2.5, 2$;
$n = 0(1)26$: $x = -2.5, -2, -1.25$.

The authors have machine listings of all these computed values.

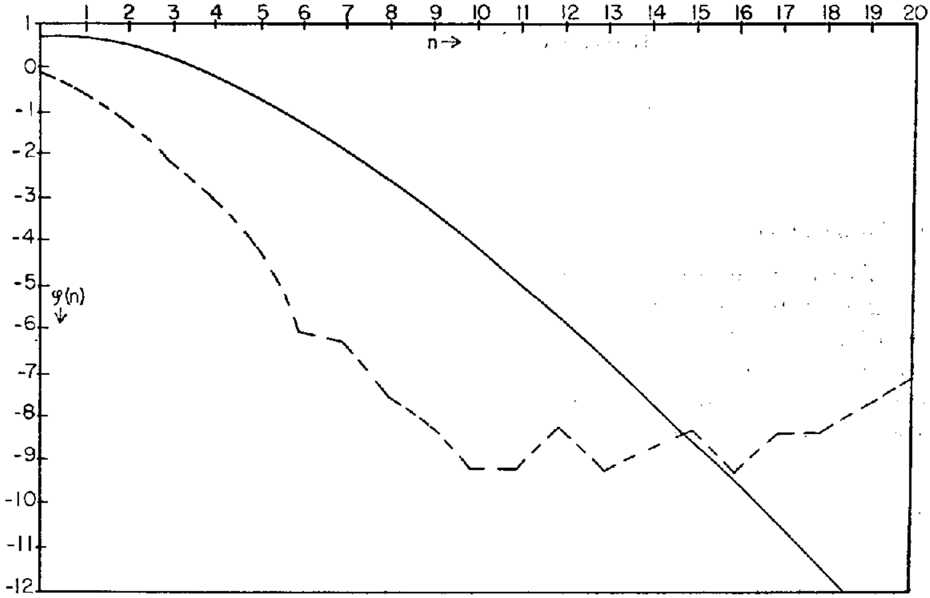
In table 3 is shown a sample of the values of $f_n^{(\xi)}(x)$, for selected x which never coincide with an x_i. The difference in the round-off error between interpolation and extrapolation is evident. Almost every computation shows indications that, as n grows, ultimately $f_n^{(\xi)}(x)$ diverges from e^x. This instability appears to start at n near 15. When x is inside $[-.97, .94]$, the divergence is oscillatory at first, and the error grows only slowly, so that even $f_{31}^{(\xi)}(x)$ is good to almost one decimal.

For x outside the interval $[-.97, .94]$, the divergence again begins for n near 15, but is one-sided and grows approximately exponentially and much more rapidly.

Our assertion that the errors in table 3 are due to round-off has been substantiated by comparing the errors with their bounds obtained from the above theorem. Examples of this comparison are shown in the two accompanying graphs. The base 10 logarithms of $|f_n^{(\xi)}(x) - e^x|$ $(n = 0, 1, \cdots, 21)$ are in-



GRAPH 1. $x = 0, m = 66$. The dashed line is the observed value of $\varphi(n) = \log_{10} |e^0 - f_n^{(66)}(0)|$. The solid line is the bound for the same quantity from (17).



GRAPH 2. $x = -1$, $m = 66$. The dashed line is the observed value of $\varphi(n) = \log_{10} | e^{-1} - f_n^{(66)}(-1) |$. The solid line is the bound for the same quantity from (17).

dicated by dashed lines, while the solid lines show the base 10 logarithms of the error bounds (17).

On graph 1 it can be seen that the error in $f_n^{(\xi)}(0)$ is below the error bound (17) for $n \leqq 11$. However, the machine's full accuracy has apparently been reached, and no further improvement is obtained for larger n. Starting with $n = 15$ the error even begins to increase.

Graph 2 shows that the error in $f_n^{(\xi)}(-1)$ begins to increase at $n = 12$ and finally exceeds the error bound (17) for $n \geqq 17$.

Graphs 1 and 2 both exhibit for n near 12 or 13 what is commonly known as a round-off error "noise level"—here apparently about 10^{-9} or 10^{-8}. It should be noted that this noise level is about the same near an endpoint ($x = -1$) as near the midpoint ($x = 0$) of the interval of fit. For n larger than 20, however, the noise level seems to grow with n.

REFERENCES

1. J. E. Barker, Orthonormal polynomials, report, U. S. Naval Proving Ground, Dahlgren, Virginia, c. 1953.
2. P. L. Chebyshev, Ob interpolirovanii, pp. 357–374 of *Polnoe sobranie sochineniĭ*, vol. 2, Akademii͡a Nauk SSSR, 1947.
3. George E. Forsythe, Generation and use of orthogonal polynomials for data fitting with a digital computer, *J. Soc. Indust. Appl. Math.* vol. 5 (1957), pp. 74–88.
4. N.-E. Nörlund, *Leçons sur les séries d'interpolation*, Gauthier-Villars, Paris, 1926.
5. B. D. Rudin, On routines for least squares curve fitting with orthogonal polynomials, paper presented to IBM 650 Computation Seminar, Endicott, New York, April 1957 (multilithed by Lockheed Missile Systems Division, Sunnyvale, Calif.)
6. G. Szegö, Über die Entwickelung einer analytischen Funktion nach den Polynomen eines Orthogonalsystems, *Math. Ann.* vol. 82 (1921), pp. 188–212.