II. Computation and Analysis

SYSTEMS DIVISION

A. The Direct Summation of Series Involving Higher Transcendental Functions,[1] *E. W. Ng*

In many problems of physics there is often the need to evaluate or compute a series of the form

$$S_N(x) = \sum_{n=j}^{N} a_n(x) f_n(x) \tag{1}$$

where $f_n(x)$ is a higher transcendental function and $a_n(x)$ are given coefficients. Well-known examples of such are truncated series involving Chebychev polynomials, Bessel functions, and Legendre functions. In most applications we have $j = 0$ or 1. Many higher transcendental functions satisfy a three-term recurrence relation of the form

$$f_{n+1}(x) = B_n(x) f_n(x) + C_n(x) f_{n-1}(x), \qquad n = 0, 1, \cdots \tag{2}$$

For the orthogonal polynomials, we define $f_{-1}(x) = 0$.

It is well known that recurrence relations form a basic mathematical tool for the computation of many functions. We have, for example, Miller's algorithm for computing Bessel functions. For a recent detailed survey and analysis of such algorithms, the reader is referred to Ref. 1. Whereas these relations are simple to use, one must attend to the problem of numerical stability. For example, Gautschi shows that given the Bessel functions $J_0(1)$ and $J_1(1)$ accurate to 10 significant figures and generating the next values of $J_n(1)$ by forward recursion, one loses all significance for $n \geqq 7$. Abramowitz (Ref. 2) summarizes the caution one must take in using such recurrence relations. In particular, the direction of recurrence is important. For example, the Bessel functions J_n and I_n are stable only in backward recurrence, whereas Y_n and K_n are stable only in forward recurrences.

Clenshaw (Ref. 3) recommends an algorithm to sum a Chebychev series directly. Here we shall generalize the algorithm to other special functions satisfying Eq. (2).

Consider the recurrence formula (with the functional dependence on x understood)

$$b_k = b_{k+1} B_k + b_{k+2} C_{k+1} + a_k, \qquad k = N, N-1, \cdots, j$$
$$b_{N+1} = b_{N+2} = 0 \tag{3}$$

Multiply Eq. (3) by f_k and form a "system of equations" as follows:

$$\left.\begin{aligned}
b_N f_N &= && + a_N f_N \\
b_{N-1} f_{N-1} &= b_N f_{N-1} B_{N-1} && + a_{N-1} f_{N-1} \\
b_{N-2} f_{N-2} &= b_{N-1} f_{N-2} B_{N-2} + b_N f_{N-2} C_{N-1} && + a_{N-2} f_{N-2} \\
&\;\;\vdots \\
b_j f_j &= b_{j+1} f_j B_j + b_{j+2} f_j C_{j+1} + a_j f_j
\end{aligned}\right\} \tag{4}$$

[1] A more detailed version of this article will appear in a future issue of *J. Comp. Phys.*, Vol. III.

Adding all equations of Expression (4) and using Eq. (2), we obtain

$$S_N = \sum_{n=j}^{N} a_n f_n = b_j f_j + b_{j+1}(f_{j+1} - B_j f_j) \tag{5}$$

Notice that Eq. (3) is a backward recurrence, but not as the nonhomogeneous counterpart of Eq. (2), because the role of C_k is displaced. Obviously one can also derive a recurrence scheme expressing S_N in terms of f_N and f_{N-1}. Notice that for $j = 0$, $S_N = b_0$ for the orthogonal polynomials. Thus Eq. (3) represents a formalism for computing the series S_N. It is mainly useful for the case $j = 0$ or 1, because here f_0 and f_1 are readily obtainable. But the applicability will, of course, depend on the stability of Eq. (3), which in turn depends on the function in question. In the following, we shall describe some numerical experiments with this algorithm by applying it to the following simple series (Ref. 4):

$$J_0(\pi) + 2\sum_{n=1}^{\infty}(-1)^n J_{2n}(\pi)\, T_{2n}(x) = \cos \pi x, \qquad -1 \leq x \leq 1 \tag{6}$$

$$I_0(1) + 2\sum_{n=1}^{\infty}(-1)^n I_n(1)\, T_n(x) = e^x, \qquad -1 \leq x \leq 1 \tag{7}$$

$$\sum_{n=0}^{\infty}(2n+1)\, z^n P_n(x) = \frac{1 - z^2}{(1 - 2xz + z^2)^{3/2}}, \qquad -1 \leq x \leq 1, |z| \leq 0.6 \tag{8}$$

$$\sum_{n=0}^{\infty}\frac{1}{nz^n} P_n(x) = \log\left[\frac{2z}{z - x + (1 - 2zx + z^2)^{1/2}}\right], \qquad -1 \leq x \leq 1, \quad 2 \leq z \leq 10 \tag{9}$$

$$\sum_{h=0}^{\infty}\frac{z^n}{\Gamma(n+1)} P_n(x) = e^{xz} J_0[z(1 - x^2)^{1/2}], \qquad -1 \leq x \leq 1, |z| \leq 4 \tag{10}$$

$$\sum_{n=0}^{N} A_n P_n(x), \qquad -1 \leq x \leq 1, \quad N = 20, 30, 40, \quad 0 \leq A_n \leq 100 \tag{11}$$

$$\sum_{n=0}^{N} A_n L_n(x), \qquad 0 \leq x \leq 100, \quad N = 20, 30, 40, \quad 0 \leq A_n \leq 100 \tag{12}$$

All computations were performed on an IBM 7094 computer using double precision (16-decimal digit) arithmetic. For Eqs. (6) to (10), we terminate the series when the coefficient is less than 10^{-17}. For each series we generate 1000 uniformly distributed pseudorandom numbers for the variables x and z in the indicated range, which does not necessarily cover the whole range of theoretical convergence. The choice of range is obviously for practicality. For example, for Eq. (8), at $|z| = 0.6$ one needs about 100 terms to satisfy our criterion. In Eqs. (11) and (12) the A_n's are a set of pseudorandom numbers uniformly distributed in the indicated range. In all of the above series, we also compute the sum by generating the special functions by forward recurrence and then summing. Thus we have three different results for Eqs. (6) to (10) and two for Eqs. (11) and (12). In all cases we compute the relative differences among the two or three different methods. These differences, of course, depend on the values of x, z, A_n, S_N, and N. They range from 1×10^{-6} to 1×10^{-14}, but are in no case greater than the last number.

References

1. Gautschi, W., *SIAM Rev.*, Vol. 9, p. 24, 1967.
2. Abramowitz, M., "Handbook of Mathematical Functions," *NBS Appl. Math.*, Ser. 55, p. XIII, 1965.
3. Clenshaw, C. W., *National Physical Laboratory Mathematical Tables*, Vol. 5. Her Majesty's Stationery Office, London, 1962.
4. Margulis, V., *Handbook of Series*, Academic Press, New York, 1965.

B. Integrals of Confluent Hypergeometric Functions, Part II,[2] *E. W. Ng*

This article is a direct continuation of Part I in SPS 37-46, Vol. IV, p. 34. A closely related set of integrals appear in Ref. 1.

[2]A condensed version of Parts I and II will appear in a future issue of *J. Res. NBS, Sec. B.*

1. Reduction Formulas for $\Theta_n(a, b, \alpha, z)$ and $\Upsilon_n(a, b, \alpha, z)$

As before, four different cases of the parameters a and b will be considered.

a. Case 1

$$a \neq \text{integer}, \qquad b \neq \text{integer}$$

In this case two formulas equivalent to Eqs. (34) and (35) of Part I can be derived. However, the results will not be too useful, because the right-hand side will have a term with subscript n, due to the derivative of $(z^n e^{\alpha z})$. Instead, the equivalent of Eqs. (38) and (39) of Part I is written as

$$\Theta_n(a, b, \alpha, z) = (b-1)\,\Theta_{n-1}(a, b-1, \alpha, z) - (b-1)\,\Theta_{n-1}(a-1, b-1, \alpha, z) \tag{1}$$

$$\Upsilon_n(a, b, \alpha, z) = (b-a-1)\,\Upsilon_{n-1}(a, b-1, \alpha, z) + \Upsilon_{n-1}(a-1, b-1, \alpha, z) \tag{2}$$

b. Case 2

$$a \neq \text{integer}, \qquad b = \text{integer}$$

With the help of Eqs. (12) and (14) of Part I, $\Theta_{n-1}(a, b-1, \alpha, z)$ can be expressed in terms of $\Theta_{n-1}(a+1, b, \alpha, z)$, thereby obtaining

$$\Theta_n(a, b, \alpha, z) = 2\Theta_{n-1}(a+1, b, \alpha, z) - (2a-b)\,\Theta_{n-1}(a, b, \alpha, z) + (a-b)\,\Theta_{n-1}(a-1, b, \alpha, z) \tag{3}$$

$$\Upsilon_n(a, b, \alpha, z) = a(a+1-b)\,\Upsilon_{n-1}(a+1, b, \alpha, z) + (b-2a)\,\Upsilon_{n-1}(a, b, \alpha, z) + \Upsilon_{n-1}(a-1, b, \alpha, z) \tag{4}$$

c. Case 3

$$a = \text{integer}, \qquad b \neq \text{integer}$$

In this case, Eqs. (1) and (2) can be used again to reduce Θ_n and Υ_n to Θ_0 and Υ_0, and $\Theta_n(0, \beta, \alpha, z)$ and $\Upsilon_n(0, \beta, \alpha, z)$, where the last two are just elementary integrals. Kummer's first theorem (Ref. 2, p. 6) can also be used to transform Θ_n, in this case to that of case 1, as follows:

$$\Theta_n(a, b, \alpha, z) = \int z^n e^{(\alpha+1)z} M(b-a, b, -z)\,dz = (-1)^{n+1}\,\Theta_n(b-a, b, \alpha+1, -z) \tag{5}$$

d. Case 4

$$a = \text{integer}, \qquad b = \text{integer}$$

For $a > b$, a reduction formula equivalent to Eq. (42) can be used:

$$\Theta_n(a, b, \alpha, z) = \Theta_n(a-1, b, \alpha, z) + 1/b\,[\Theta_{n+1}(a, b+1, \alpha, z)] \tag{6}$$

Again, successive application will reduce the right-hand side of the equation to elementary integrals. For $b > a > 0$, Eqs. (1) and (2) can be conveniently applied. for $a < 0$, Eqs. (30) and (31) of Part I can again be used, but with a replaced by $(a+1)$ and "recur upward" in n. However, for all integer values of a and b, Eqs. (3) and (4) are applicable.

Therefore, it can be seen that Θ_n and Υ_n can be reduced to a finite combination of Λ, M, Ω, U, or elementary integrals. Properties of Λ and Ω will be discussed in subsequent investigations.

References

1. Ng, E. W., *J. Math. Phys.*, Vol. 46, p. 223, 1967.
2. Slater, L. J., *Confluent Hypergeometric Functions*, Cambridge University Press, London, 1960.

C. Survey of Computer Methods for Fitting Curves to Discrete Data or Approximating Continuous Functions, C. L. Lawson

1. Introduction

In preparation for this survey, a classified bibliography of recent publications[3] was compiled that includes 394 references. As is clear from this bibliography, approximation theory has wide application in the mathematics of computation; e.g., approximation of functions or data; quadrature; solution of ordinary differential equations, partial differential equations, and integral equations; and graphical displays. On the other hand, approximation algorithms often depend upon more general computational techniques, such as the solution of linear or nonlinear systems of equations and/or inequalities and general minimization methods. A selection of references on these latter topics is included in the bibliography.

[3]Lawson, C. L., "Bibliography of Recent Publications in Approximation Theory With Emphasis on Computer Applications," *Comput. Rev.*, Vol. 9 (to be published).

This survey treats primarily the problems of fitting curves to discrete data and approximating continuous functions. The point of view is that of practical scientific computation.

The choice of a mathematical model in an approximation problem can often be conveniently described as the choice of form and norm, i.e., the choice of approximating form such as polynomial, rational, or spline, and the choice of norm such as ℓ_2, ℓ_∞, or ℓ_1. There are, of course, other considerations such as constraints and transformation of variables.

Often the problem objectives are such that there is some freedom in the choice of form or norm. Then the choice should be made on the basis of properties such as numerical stability and economy of computation. These properties are discussed in *Subsections* 2 and 3. Most of *Subsections* 2 and 3 generalizes to fitting functions of two or more real variables or complex variables; however, from the practical point of view, such fits are often limited to applications requiring only moderate accuracy (e.g., 10^{-4}) because of the very large number of parameters needed for higher accuracy.

2. Choice of Form

a. Polynomial forms. Polynomial forms are, in a sense, the simplest, and a variety of parameterizations is possible. If a polynomial is expressed as $\Sigma a_i x^i$, which will be called the monomial basis parameterization, it can be evaluated in n multiplications and n additions. The matrix of basis function values is typically very poorly conditioned. This conditioning is generally significantly improved by translating the domain of the independent variable to be centered at zero. Exponent overflow is avoided by scaling to, e.g., $[-1, 1]$. Even with these precautions, polynomials of degree higher than about 7 in monomial basis form are essentially useless in 8-decimal digit arithmetic.

Other bases for parameterization, such as Chebyshev polynomials, typically provide remarkable stability. For example, polynomials of degrees 533 through 223 have been computed to represent the positions of the five outer planets, Jupiter through Pluto, over a period of 200 yr. This work used 16-decimal digits and preserved at least 5-digit accuracy. A polynomial of degree n represented as a linear combination of Chebyshev polynomials can be evaluated in n multiplications and $2n$ additions. In general, the Chebyshev basis is preferable to the monomial basis, independent of other factors such as the method for determining coefficients or the choice of norm.

Other polynomial parameterizations include the Forsythe parameterization for polynomials determined to be orthogonal over a specific point set, the product-of-roots form, and streamlined forms. The product-of-roots form is very stable if the roots are in the x-interval of interest, but the determination of parameters may be inconvenient. The streamlined forms reduce the number of multiplications needed in evaluation but are often very unstable. The Forsythe parameterization is redundant, requiring about $3n$ parameters to specify an nth-degree polynomial. It exhibits very good numerical stability, and the algorithm for determining the parameters is very efficient, since the execution time depends upon mn, rather than mn^2, where m is the number of data points.

b. Rational forms. Various special properties (such as remaining bounded at infinity, having poles, and having abrupt changes of curvature) make rational forms more useful than polynomial forms in some cases. Since the parameters occur nonlinearly, their determination requires iterative procedures which entail various practical difficulties: (1) the absence of zeros from the denominator must always be verified. (2) Best rational approximations on discrete sets do not always exist. The use of rational functions for fitting discrete data can probably be largely supplanted by the use of spline polynomials.

Rational functions have been very successfully used as approximating forms for many analytic functions such as the exponential and arctangent. The effective design of such approximations depends more upon a thorough understanding of the function being approximated (leading to the use of special identities and changes of variables) than upon the actual method of computation of the approximation.

All polynomial parameterizations can be used for rational function parameterization. There is also the possibility of using continued fraction forms; however, these are frequently unstable and must be tested for growth of rounding error in each case.

c. Spline forms. A spline function s, defined on an interval $[a, b]$ partitioned into k segments, is a polynomial of degree n on each segment with continuous derivatives through order m $(m < n)$ throughout $[a, b]$. A spline will generally have discontinuities in its $(m + 1)$st derivative at the partition points. The splines which have received

the most study are those for which $m = n - 1$ and, more particularly, cubic splines with second-order continuity. Such splines, having k segments, can be parameterized by $k - 1 + 4k$ parameters, giving the abscissas of the $k - 1$ partition points and 4 coefficients for each of the k cubic polynomial segments. The discussion given previously for the parameterization of polynomials is then applicable to each segment.

This parameterization, though convenient for evaluation, is redundant. Other parameterizations having less redundancy have been given in the literature. Some, such as

$$\sum_{i=0}^{3} c_i x^i + \sum_{i=1}^{k-1} a_i \{\max [0, (x - b_i)]\}^3$$

are of theoretical use only and are definitely not recommended for computational use.

If the partition points are fixed, $4k$ parameters remain, and these occur linearly. The second-order continuity requirement constitutes $3(k - 1)$ linear equality constraints, reducing the number of degrees of freedom to $k + 3$. One basis consisting of $k + 3$ linearly independent splines, which has been recommended as having favorable properties in practical use, can be defined as follows:

By introducing 3 auxiliary segments to the left and 3 to the right of the interval $[a, b]$, $k + 6$ segments are defined. For each set of four contiguous segments, a spline function is constructed that is nonzero on that set and is zero elsewhere. This defines $k + 3$ basis functions (each uniquely determined to within an arbitrary scalar multiple). The associated matrix in curve fitting has a block diagonal structure that can be used to conserve computer time and storage.

Although spline forms have received intensive study in recent years, the best strategies for parameterizing and manipulating splines and treating the problem with variable breakpoints have yet to be evolved. With their extreme flexibility in changing curvature, stability of low-degree polynomials, and linearity of coefficients (for fixed partition points), spline forms provide a very attractive approach to general data fitting.

The second derivative of a cubic spline with second-order continuity is a linear spline with zero-order continuity. Thus, the sign of the second derivative can be constrained throughout $[a, b]$ by constraining it only at the partition points. This fact has been used to obtain some very satisfactory data fits where oscillations were to be avoided.

3. Choice of Norm

For fitting data subject to random errors, it can be argued that the ℓ_2 norm is most appropriate if the error is normally distributed; ℓ_1 is most appropriate if the error distribution has broad tails; and ℓ_∞ is most appropriate if the error distribution has narrow or no tails, e.g., a uniform distribution over a finite interval.

In practice, the ℓ_1 approximation is probably very rarely used, since the broad tail problem is usually treated by some ad hoc wild-point exclusion logic. Discrete ℓ_1 approximations can be nonunique even with the Haar condition, and characterization of a best ℓ_1 approximation is complex. The discrete linear ℓ_1 problem is a linear programming problem; however, a linear programming code should have a full capability to treat degenerate cases if it is to be trusted for ℓ_1 fitting.

Discrete ℓ_2 (least-squares) approximation is, of course, widely used. With linear parameters it is a linear problem, i.e., no iteration is needed. Orthonormal methods such as Householder transformations or modified Gram-Schmidt orthogonalization (numerically superior to Gram-Schmidt orthogonalization) can be used to avoid the squaring of the condition number associated with the formation of normal equations. The number of multiplications and additions is approximately doubled with the orthonormal methods, and thus these methods must be compared with the use of normal equations in double-length arithmetic to determine which is more efficient and reliable in a given application.

Discrete linear ℓ_∞ approximations can be treated as the linear programming problem it is or by specially adapted equivalent algorithms such as the exchange algorithm. Two other distinct methods for the discrete ℓ_∞ problem, although probably not competitive with the exchange algorithm for the linear Haar ℓ_∞ problem, appear to generalize to the nonlinear or non-Haar cases in a more natural way. These are: (1) the Polya algorithm, which relies on the ℓ_∞ solution being the limit of ℓ_p solutions as $p \to \infty$; and (2) the Lawson algorithm, which adjusts weights in a weighted ℓ_2 approximation so that the ℓ_∞ approximation is approached via a sequence of weighted ℓ_2 approximations.

For the approximation of continuous functions by curve fitting, primary interest has been with the ℓ_∞ approximation. Exchange-type algorithms have been used very

effectively and efficiently for both polynomial and rational approximations.

Such approximations are commonly produced for use in function subprograms. The construction of an efficient function subprogram also depends strongly upon the use of special properties of the function being approximated and machine-dependent considerations. For some functions, particularly functions of more than one variable, efficient constructive representations have been derived entirely from mathematical analysis of the functions without the use of fitting.