BEST ℓ_p APPROXIMATIONS BY CERTAIN NONLINEAR FUNCTIONS

by

CHARLES RICHARD HUNT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in the Department

of

Mathematics

We accept this thesis as conforming
to the required standard

[redacted]

Accepted for the Faculty of Graduate Studies
[redacted]
April 29, 1970

UNIVERSITY OF VICTORIA

April 1970

ABSTRACT

An algorithm is described for computing best ℓ_1, ℓ_2 and ℓ_∞ approximations to discrete data, by functions of several parameters which depend nonlinearly on just one of these parameters. Such functions (e.g. $a_1 + a_2e^{cx}$, $a_1 + a_2\sin(cx)$, $(a_1 + a_2x)/(1 + cx)$) often occur in practice, and this numerical study confirms that it is feasible to compute best approximations in any of the above norms when using these functions. Some remarks on the theory of best ℓ_1 approximations by these functions are included.

An extension of this algorithm to functions nonlinear in two of their several parameters is outlined, and an attempt is made to apply this method to a practical problem.

Examiners: [redacted]

TABLE OF CONTENTS

LIST OF TABLES

TABLE PAGE

LIST OF FIGURES

ACKNOWLEDGEMENTS

I wish to express my gratitude to my co-supervisors, Dr. I. Barrodale and Dr. F.D.K. Roberts, for their guidance and encouragement and for their limitless patience. I am indebted to Dr. R.E. Odeh for his helpful criticisms of the rough draft of this thesis, and to those members of the Department of Mathematics who were so generous with their time and assistance. I also wish to thank the staff of the University of Victoria Computing Centre for their help, and Miss B. Cooknell for her careful typing of the final manuscript.

INTRODUCTION

One of the most common problems in numerical computation is to determine a good approximation to some given experimental data. Typically, we are supplied with observations $y_1, y_2, \ldots, y_N$ of some quantity y, each one corresponding to a value x_i of some quantity x. An approximating function is chosen, say $F(A,x) = a_1 + a_2x + a_3x^2$, where the set of coefficients $A = \{a_1, a_2, a_3\}$ is to be determined. Then the problem is to assign values to the parameters a_1, a_2, a_3 so that $F(A,x)$ is a good approximation to the given function $\{(x_1, y_1), (x_2,y_2), \ldots, (x_N, y_N)\}$.

Frequently, approximations are desired which are not only good, but are best approximations, in some sense. Those best approximations most commonly used in practice are of three types. Perhaps the most familiar is the ℓ_2, or least squares, approximation. Such an approximation minimizes the sum of the squares of the deviations of the approximating function from the data. Until recently this was essentially the only type of best approximation which could be computed, and then only for medium size problems. Now, ℓ_2 approximations are clearly not always most suitable. However, the computation of other types of best approximations has been too difficult and time-consuming to allow their acceptance for common use.

With the advent of the high speed electronic computer and the development of appropriate algorithms, the computation of best approximations other than ℓ_2 became feasible. Two types which have recently come into common use are, approximations which minimize the

sum of the magnitudes of the deviations, and approximations which minimize the magnitude of the maximum deviation. These are known, respectively, as best ℓ_1 and best ℓ_∞ approximations.

The function chosen to approximate the data may come from one of two general classes of approximating functions. Linear approximating functions are those which depend linearly upon their parameters (e.g. $a_1 + a_2x + a_3x^2$ and $a_1 + a_2x^3 + a_3e^x$ both depend linearly upon a_1, a_2 and a_3). Much work has been done on linear approximation and a relatively large body of knowledge has evolved. However, many types of data are more suitably approximated by nonlinear functions. Living organisms, for example, frequently have growth curves which are exponential in shape. Very often in this case a nonlinear approximating function (say $a_1 + a_2e^{cx}$ which depends linearly on a_1 and a_2 but nonlinearly on c) is more appropriate than is a linear approximating function. Spline functions (e.g. $a_1 + a_2x + a_3x^2 + (x - c_1)_+^2 + (x - c_2)_+^2$, which depends nonlinearly upon c_1 and c_2) are useful when a smooth but flexible curve of relatively low degree is required.

The computation of best ℓ_1, ℓ_2 and ℓ_∞ approximations to discrete data by linear functions has received much attention in the literature of the past few years. However, little has appeared concerning the actual determination of best nonlinear approximations, except for the rational ℓ_∞ case and some ℓ_2 approximations. Rice (1964,1969) may be consulted for information on the present state of computational procedures for these cases. In this thesis we are concerned with the

class of approximating functions which are nonlinear in only one of their several parameters (e.g. $a_1 + a_2 e^{cx}$, $a_1 + a_2 x^c$, $a_1 \sin(cx) + a_2 \cos(cx)$).

We now make a precise mathematical statement of the problem at hand and the method proposed for its solution.

The general approximation problem on a set $X = \{x_1, x_2, \ldots, x_N\}$ of real numbers (or, more generally, points from some Euclidean space E_k) is to choose an approximating function, $F(A,x)$, and select a particular form, $F(A^*,x)$, which approximates a given function satisfactorily on X. Here $A = \{a_1, a_2, \ldots, a_n\}$ is a set of free parameters. The most general linear function is $F(A,x) = \sum_{j=1}^{n} a_j \phi_j(x)$, where the $\phi_j(x)$'s are given linearly independent functions defined on X. $F(A^*,x)$ is called a best approximation in a norm $\|.\|$ if for all choices of A, $\|f(x) - F(A^*,x)\| \leq \|f(x) - F(A,x)\|$. The three norms most commonly used in practice are:

$$\ell_1: \quad \left\| w(x)\left[f(x) - F(A,x)\right] \right\|_1 = \sum_{i=1}^{N} w(x_i) \left| f(x_i) - F(A,x_i) \right|$$

$$\ell_2: \quad \left\| w(x)\left[f(x) - F(A,x)\right] \right\|_2 = \left\{ \sum_{i=1}^{N} w(x_i)\left[f(x_i) - F(A,x_i)\right]^2 \right\}^{\frac{1}{2}}$$

$$\ell_\infty: \quad \left\| w(x)\left[f(x) - F(A,x)\right] \right\|_\infty = \max_{1 \leq i \leq N} w(x_i) \left| f(x_i) - F(A,x_i) \right|$$

where $w(x_i)$ is a prescribed set of positive weights.

For linear approximations $F(A,x)$, best approximations are guaranteed to exist in all three norms, but only ℓ_2 approximations are necessarily unique (Rice (1964)). Best ℓ_1 and ℓ_∞ approximations

can be determined by linear programming (Barrodale and Young (1966)) while ℓ_2 approximations are produced by a standard orthonormalizing routine. If the discrete data is affected by noise then the distribution of these random errors should dictate the choice of norm. In practice one chooses the ℓ_1, ℓ_2 or ℓ_∞ norm, respectively, according as the errors are subject to wild points, normally distributed, or very small relative to the error of approximation (Rice and White (1964) and Barrodale (1968)).

In the discrete nonlinear case best approximations do not necessarily exist. Generally, progress towards developing algorithms for best nonlinear approximation has been slow.

The success of the method of this thesis depends upon the validity of the assumption that a certain function is unimodal in a given range. Suppose that $F(A,x)$ is an approximating function which depends linearly upon the parameters $a_1, a_2, \ldots, a_{n-1}$ and nonlinearly upon c, where $A = \{a_1, a_2, \ldots, a_{n-1}, c\}$. Let $A^* = \{a_1^*, \ldots, a_{n-1}^*, c^*\}$ be a set of best parameters for a given norm: we assume here that a best approximation exists. The method for determining A^* is the following. For any given value of the nonlinear parameter, the determination of a best ℓ_p approximation can be accomplished by using one of the existing linear approximation algorithms. Considering the error of approximation thus produced as a function of the nonlinear parameter only, the problem of determining a best nonlinear approximation is reduced to that of locating the minimum of a function of one variable. A best nonlinear approximation can then be found very efficiently using

a Fibonacci search technique. The justification for using a Fibonacci search is that the following assumption be true.

Assumption: $G(c) = \min_{a_1,\ldots,a_{n-1}} \|f(x) - F(A,x)\|_p$ is unimodal in c, at least for some small interval containing c^*.

The evidence presented in this thesis confirms that this underlying assumption is most reasonable in practice. Extracts from this thesis are to appear in Barrodale, Roberts and Hunt (1970).

In Chapter I we discuss methods currently in use for the determination of best linear approximations in the ℓ_1, ℓ_2 and ℓ_∞ norms. Chapter II defines the concept of a unimodal function of one variable, and includes a description of the Fibonacci search technique. This technique is one of the most efficient searching algorithms known for determining the minimum of a unimodal function of one variable. Chapter III describes the manner in which we have combined the topics of the previous two chapters into a new algorithm for the computation of best ℓ_p approximations by a certain class of functions. We include in this chapter computational details of our method for twelve particular approximating functions. Chapter IV includes some remarks on the theory of best ℓ_1 approximation by such functions. Finally, Chapter V describes a practical problem and the manner in which our method was applied to it. This leads to an attempt to extend our method to functions nonlinear in two parameters. The Appendices contain driver programs for best ℓ_1 approximation by $a_1 + a_2 e^{cx}$, and both the subroutines MINSUM and MINMAX.

CHAPTER I

COMPUTATION OF ℓ_p LINEAR APPROXIMATIONS

I-1. Statement of the Problems.

Let $f(x)$ be a given real-valued function defined on a discrete subset $X = \{x_1, x_2, \ldots, x_N\}$ of E_k. Given a set of n real-valued continuous functions $\{\phi_j(x)\}$ defined on X we form a linear approximating function $F(A,x) = \sum_{j=1}^{n} a_j \phi_j(x)$ for any real set $A = \{a_1, a_2, \ldots, a_n\}$.

The ℓ_1 approximation problem is to determine A^* such that

$$\sum_{x \in X} |F(A^*,x) - f(x)| \leq \sum_{x \in X} |F(A,x) - f(x)|$$

for all $A \in E_k$.

The ℓ_2 approximation problem is to determine A^* such that

$$\sum_{x \in X} \left[F(A^*,x) - f(x)\right]^2 \leq \sum_{x \in X} \left[F(A,x) - f(x)\right]^2$$

for all $A \in E_k$.

The ℓ_∞ approximation problem is to determine A^* such that

$$\max_{x \in X} |F(A^*,x) - f(x)| \leq \max_{x \in X} |F(A,x) - f(x)|$$

for all $A \in E_k$.

I-2. The ℓ_1 and ℓ_∞ Norms.

(a) Adaptation to Linear Programming. The method we use to find best ℓ_1 and ℓ_∞ linear approximations is that developed by Barrodale and Young (1966), wherein the approximation problem is restated as a linear programming problem.

The procedure is as follows. To ensure the non-negativity of the unknown variables put

$$\alpha_{n+1} = \max(0, -\min_j a_j) \quad \text{and then} \quad \alpha_j = a_j + \alpha_{n+1} \quad \text{for} \quad 1 \le j \le n.$$

Then, for $1 \le i \le N$, define

$$e_i = e(x_i) = F(A,x_i) - f(x_i)$$

$$= \sum_{j=1}^{n} \alpha_j \phi_j(x_i) - \alpha_{n+1} \sum_{j=1}^{n} \phi_j(x_i) - f(x_i)$$

$$= \alpha_1 \phi_{1,i} + \ldots + \alpha_n \phi_{n,i} + \alpha_{n+1} \phi_{n+1,i} - f_i$$

where $\phi_{n+1}(x_i) = -\sum_{j=1}^{n} \phi_j(x_i)$ for $1 \le i \le N$, and $\phi_{j,i}$ and f_i respectively denote $\phi_j(x_i)$ and $f(x_i)$. Finally putting $e_i = u_i - v_i$ where $u_i \ge 0$ and $v_i \ge 0$ results in N constraints in non-negative variables

$$f_i = \alpha_1 \phi_{1,i} + \ldots + \alpha_{n+1} \phi_{n+1,i} - u_i + v_i \quad \text{for} \quad 1 \le i \le N \qquad (A)$$

The ℓ_1 approximation problem is to find $\{a_j\}$ such that $\sum_{i=1}^{N} |e_i|$ is minimized. That is, we want to solve the problem: minimize $\sum_{i=1}^{N} |u_i - v_i|$ subject to (A). This is equivalent to minimizing

$\sum_{i=1}^{N} (u_i + v_i)$, subject to (A) and the additional conditions $u_i \times v_i = 0$ for $i = 1,2,\ldots,N$. We now consider the linear programming problem: minimize $\sum_{i=1}^{N} (u_i + v_i)$ subject to (A). Since the solution to this linear program occurs at an extreme point, the conditions $u_i \times v_i = 0$ are satisfied automatically. Thus, solving this linear program is equivalent to solving the ℓ_1 approximation problem.

The ℓ_∞ approximation problem is to find $\{a_j\}$ such that $\max_{1\le i\le N} |e_i|$ is minimized. For any $A \in E_k$ put $w = \max_{1\le i\le N} |e_i|$ and obtain the $2N$ constraints

$$\alpha_1\phi_{1,i} + \ldots + \alpha_{n+1}\phi_{n+1,i} + w \ge f_i \quad \text{for } 1 \le i \le N \qquad (B)$$

$$-\alpha_1\phi_{1,i} - \ldots - \alpha_{n+1}\phi_{n+1,i} + w \ge -f_i$$

This gives the linear programming problem of minimizing w subject to (B). However in practice the dual problem is solved, which is to find non-negative values of s_i and t_i for $1 \le i \le N$ which maximize

$$\sum_{i=1}^{N} f_i(s_i - t_i)$$

subject to the $n+2$ constraints

$$\sum_{i=1}^{N} \phi_{j,i}\,(s_i - t_i) \le 0 \quad \text{for } 1 \le j \le n+1$$

and

$$\sum_{i=1}^{N} (s_i + t_i) \le 1.$$

These constraints are then expressed as equalities using α_j and w, the original variables of (B), as the slack variables. This reformulation as the dual problem reduces the number of constraints, thereby yielding a more efficient algorithm.

(b) The ℓ_1 Algorithm: The objective function $\sum_{i=1}^{N} (u_i + v_i)$ and the N constraints

$$f_i = \alpha_1 \phi_{1,i} + \ldots + \alpha_{n+1} \phi_{n+1,i} - u_i + v_i \qquad 1 \le i \le N$$

are arranged in a simplex tableau as shown in Fig. 1.

The names $\vec{R}, \vec{\alpha}_1, \ldots, \vec{\alpha}_{n+1}, \vec{u}_1, \ldots, \vec{u}_N, \vec{v}_1, \ldots, \vec{v}_N$ are not formally defined as they are obvious choices for the column vectors. Fig. 1 is condensed by storing only the columns $\vec{R}$ and $\vec{\alpha}_j$ for $1 \le j \le n+1$. These $n+2$ vectors are sufficient to allow the simplex algorithm to be carried out in Fig. 1. If every $f_i \ge 0$ an initial basis is $\{\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_N\}$ and, as at any stage $\vec{u}_i = -\vec{v}_i$ for $1 \le i \le N$, it is clear that if $\vec{v}_i$ is in the basis then $\vec{u}_i$ must be out and need not be stored. If any $f_i < 0$ the row is multiplied by -1 and the $\vec{v}_i$ are replaced by the corresponding $\vec{u}_i$ in the initial basis. Also, the sum of the marginal costs of $\vec{u}_i$ and $\vec{v}_i$ is -2 after each iteration and thus one may be deduced from the other.

Costs→			0	0		0	1	1	1	1	1	1
↓	Basis	$\vec{R}$	$\vec{\alpha}_1$	$\vec{\alpha}_2$	...	$\vec{\alpha}_{n+1}$	$\vec{u}_1$	$\vec{u}_2 \cdots \vec{u}_N$		$\vec{v}_1$	$\vec{v}_2 \cdots \vec{v}_N$	
1	$\vec{v}_1$	f_1	$\phi_{1,1}$	$\phi_{2,1}$		$\phi_{n+1,1}$	-1	0	0	1	0	0
1	$\vec{v}_2$	f_2	$\phi_{1,2}$	$\phi_{2,2}$		$\phi_{n+1,2}$	0	-1	0	0	1	0
	$\vdots$											
1	$\vec{v}_N$	f_N	$\phi_{1,N}$	$\phi_{2,N}$		$\phi_{n+1,N}$	0	0	-1	0	0	1
Marginal Costs		$\sum_{i=1}^{N} f_i$	$\sum_{i=1}^{N} \phi_{1,i}$	$\sum_{i=1}^{N} \phi_{2,i}$		$\sum_{i=1}^{N} \phi_{n+1,i}$	-2	-2	-2	0	0	0

Fig. 1. Full initial simplex tableau for the ℓ_1 algorithm.

(c) The ℓ_∞ Algorithm: The objective function $\sum_{i=1}^{N} f_i(s_i - t_i)$ and the $n+2$ constraints

$$\sum_{i=1}^{N} (s_i + t_i) + w = 1$$

$$\sum_{i=1}^{N} \phi_{j,i}(s_i - t_i) + \alpha_j = 0 \quad \text{for} \quad 1 \le j \le n+1$$

are arranged in a simplex tableau as shown in Fig. 2.

The column vectors $\vec{R}, \vec{w}, \vec{\alpha}_1,\ldots,\vec{\alpha}_{n+1}, \vec{s}_1,\ldots,\vec{s}_N, \vec{t}_1,\ldots,\vec{t}_N$ are considerably different from the vectors of Fig. 1, but again Fig. 2 serves as a definition of them. Fig. 2 is condensed by storing only the columns $\vec{s}_i$ for $1 \le i \le N$. Again all data is obtained from these N vectors since an initial basis is $\{\vec{w}, \vec{\alpha}_1,\ldots,\vec{\alpha}_{n+1}\}$ and at any subsequent stage $\vec{s}_i + \vec{t}_i = 2\vec{R}$ for $1 \le i \le N$, and also $\vec{R} = \vec{w}$. Finally the sum of the marginal costs of $\vec{s}_i$ and $\vec{t}_i$ is always equal to twice the marginal cost of $\vec{w}$.

Costs→			f_1	f_2		f_N	$-f_1$	$-f_2$		$-f_N$	0	0	0
↓	Basis	$\vec{R}$	$\vec{s}_1$	$\vec{s}_2$	...	$\vec{s}_N$	$\vec{t}_1$	$\vec{t}_2$	...	$\vec{t}_N$	$\vec{w}$	$\vec{\alpha}_1 \ldots \vec{\alpha}_{n+1}$	
0	$\vec{w}$	1	1	1		1	1	1		1	1	0	0
0	$\vec{\alpha}_1$	0	$\phi_{1,1}$	$\phi_{1,2}$		$\phi_{1,N}$	$-\phi_{1,1}$	$-\phi_{1,2}$		$-\phi_{1,N}$	0	1	0
	⋮												
0	$\vec{\alpha}_{n+1}$	0	$\phi_{n+1,1}$	$\phi_{n+1,2}$		$\phi_{n+1,N}$	$-\phi_{n+1,1}$	$-\phi_{n+1,2}$		$-\phi_{n+1,N}$	0	0	1
Marginal costs		0	$-f_1$	$-f_2$		$-f_N$	f_1	f_2		f_N	0	0	0

Fig. 2. Full initial simplex tableau for the ℓ_∞ algorithm.

I-3. The ℓ_2 Norm.

Best ℓ_2 approximations are obtained by solving the normal equations, which, if we set

$$H(a_1, a_2,\ldots,a_n) = \sum_{i=1}^{N} \left[\sum_{j=1}^{n} a_j \phi_j(x_i) - f(x_i) \right]^2 ,$$

are given by

$$\frac{\partial H}{\partial a_k} = 2 \sum_{i=1}^{N} \left[\sum_{j=1}^{n} a_j \phi_j(x_i) - f(x_i) \right] \phi_k(x_i) = 0 \qquad (C)$$

$$k = 1, 2,\ldots,n$$

This is a system of n linear equations in n unknowns which can be solved for the a_j's to yield the required approximation. In practice, however, roundoff error tends to build up rapidly as n increases, and spurious results are produced. The following argument illustrates this phenomenon.

Let $\phi_j(x) = x^{j-1}$ for $j = 1,\ldots,n$ and let us assume for convenience that the points x_i are all in the interval $(0,1)$. Additionally, we will assume that these points are distributed reasonably uniformly in this interval. Then from (C) the system of equations we want to solve is

$$\sum_{i=1}^{N} \sum_{j=1}^{n} a_j x_i^{j+k-2} = \sum_{i=1}^{N} f(x_i) x_i^{k-1} \qquad (D)$$

$$k = 1, 2,\ldots,n$$

Interchanging summations we may write (D) as

$$\sum_{j=1}^{n} a_j \sum_{i=1}^{N} x_i^{j+k-2} = \sum_{i=1}^{N} f(x_i)\, x_i^{k-1} \qquad (E)$$

$$k = 1, 2, \ldots, n$$

then the coefficient of a_j in (E) has the form of N times a Riemann sum. For large N

$$\sum_{i=1}^{N} x_i^{j+k-2} \approx N \int_0^1 x^{j+k-2}\, dx = \frac{N}{j+k-1} \qquad j,k = 1,2,\ldots,n \qquad (F)$$

If we let G be the matrix of coefficients in (E) then using (F) we can approximate G/N by the matrix M where

$$M = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n-1} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{n+1} \\ \vdots & & & & \vdots \\ \frac{1}{n-1} & \cdots & & & \frac{1}{2n-1} \end{bmatrix} \qquad (G)$$

This matrix is the principal minor of order $n-1$ of the infinite Hilbert matrix, a classical example of an ill-conditioned matrix. Simply speaking, a matrix is said to be ill-conditioned if, when it is normalized so that its largest element is of magnitude 1 (as in (G)), its inverse has very large elements. For example, when $n = 10$ the inverse of (G) has elements of magnitude 3×10^{12}. Thus, when calculating the solution of (D) by any method at all, any roundoff error committed will cause a greatly magnified error in the solution. Hence, to obtain an accurate solution to any set of linear equations

whose coefficients form an ill-conditioned matrix, a large number of decimal places must be carried during the computation. For the Hilbert matrix, this number rapidly becomes prohibitively large.

We can overcome this difficulty by approximating with orthogonal functions. These functions can be generated by the Gram-Schmidt orthogonalization process. Davis and Rabinowitz (1961) discuss numerical aspects of orthogonalization procedures and Ralston (1965) deals with orthogonal polynomials in particular.

A few introductory remarks are required before we introduce the Gram-Schmidt orthogonalization process. Let $w(x)$ be a real positive function for $x \in [a,b]$ and let $f(x)$ and $g(x)$ be two real functions defined on this interval. The inner product of $f(x)$ and $g(x)$ with respect to the weight function $w(x)$ on the set $\{x_1, x_2, \ldots, x_N\}$, where $x_i \in [a,b]$ $i = 1,2,\ldots,N$, is defined to be

$$(f,g) = \sum_{i=1}^{N} w(x_i)\, f(x_i)\, g(x_i) \ .$$

The norm of a function $f(x)$, written $\|f(x)\|$, is defined to be

$$\|f(x)\| = (f,f)^{\frac{1}{2}} = \left\{ \sum_{i=1}^{N} w(x_i) \left[f(x_i) \right]^2 \right\}^{\frac{1}{2}} \ .$$

If the inner product of $f(x)$ and $g(x)$ is zero, i.e. $(f,g) = 0$, then $f(x)$ and $g(x)$ are said to be orthogonal with respect to the weight function $w(x)$ on the set $\{x_i\}$. When $w(x) \equiv 1$ the functions are said to be simply orthogonal on the set $\{x_i\}$. We shall concern ourselves with simply orthogonal sets in the following discussion and hence dispense with the weighting function $w(x)$.

A sequence of functions $\{f_n(x)\}$ is called an orthogonal set of functions if the functions are pairwise orthogonal, that is, if

$$(f_m, f_n) = 0 , \qquad m \neq n .$$

The set is said to be orthonormal if, in addition,

$$(f_m, f_m) = 1$$

Thus, given a set $\{\phi_j\}$ of functions, we have to obtain linear combinations of the ϕ_j's

$$\theta_i = \sum_{j=1}^{n} a_{ij}\phi_j \qquad i = 1,2,\ldots,n$$

which are orthonormal, that is

$$(\theta_i, \theta_j) = \delta_{ij} = \begin{cases} 1 & \text{for } i=j \\ 0 & \text{for } i \neq j \end{cases}$$

This can be accomplished by the Gram-Schmidt orthogonalization process which, with the inclusion of the normalizing step in the lines below, becomes an orthonormalization process. Set, recursively,

$$\beta_1 = \phi_1 \qquad\qquad \theta_1 = \beta_1 / \|\beta_1\|$$

$$\beta_2 = \phi_2 - (\phi_2, \theta_1)\theta_1 \qquad\qquad \theta_2 = \beta_2 / \|\beta_2\|$$

$$\vdots$$

$$\beta_n = \phi_n - \sum_{k=1}^{n-1} (\phi_n, \theta_k)\theta_k \qquad\qquad \theta_n = \beta_n / \|\beta_n\|$$

Then $\{\theta_i\}$ is an orthonormal set.

Approximation of the data can now be accomplished using $\{\theta_i\}$ as our set of approximating functions in (C). Thus, the system to be solved is

$$\sum_{i=1}^{N} \left[\sum_{j=1}^{n} b_j \theta_j(x_i) - f(x_i) \right] \theta_k(x_i) = 0$$

$$k = 1,2,\ldots,n$$

which, rearranged, becomes

$$\sum_{j=1}^{n} b_j \sum_{i=1}^{N} \theta_j(x_i)\, \theta_k(x_i) = \sum_{i=1}^{N} f(x_i)\, \theta_k(x_i) \qquad \text{(H)}$$

$$k = 1,2,\ldots,n$$

Here the b_j's correspond to the θ_j's as do the a_j's to the ϕ_j's in (C). Because of the orthonormality of $\{\theta_i\}$ the coefficient matrix for (H) is simply

$$\begin{bmatrix} 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 1 & 0 & \ldots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \ldots & 0 & 1 \end{bmatrix}$$

That is, the coefficients of the best approximation are just the right hand sides of the equations (H). The only roundoff error incurred, then, is that involved in the orthonormalization process. Thus, we have

$$b_j = \sum_{i=1}^{N} f(x_i)\, \theta_j(x_i) \qquad j = 1,2,\ldots,n$$

and the required a_j's can be extracted from the approximation

$$f(x) \simeq b_1\theta_1(x) + b_2\theta_2(x) + \ldots + b_n\theta_n(x)$$

by expressing each θ_j in terms of the ϕ_j's. Although roundoff error in the Gram-Schmidt process itself can cause difficulties, these are not so severe as those encountered in solving the normal equations.

CHAPTER II

UNIMODAL SEARCH

II-1. Unimodality and the Fibonacci Search.

The underlying assumption for the method of this thesis is that the error of approximation, which we want to minimize, is a unimodal function of the nonlinear parameter of the approximating function.

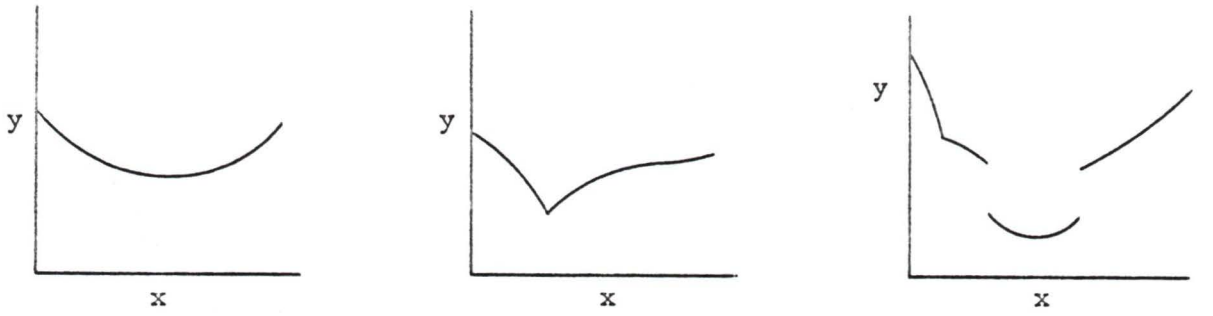
Now, suppose y is a function of x, where x lies in the interval $[a,b]$. Denote by y^* the minimum value of y, and by x^* the value of x for which this minimum is attained (i.e. $y^* \equiv \min_{a \le x \le b} \{y(x)\}$ and $y(x^*) \equiv y^*$). Consider two values of x, $a \le x_1 < x_2 \le b$, and let y_1 and y_2 be the corresponding functional values. We define unimodality as follows (Wilde (1964)).

Definition 1: y is a unimodal function of x if $x_2 < x^*$ implies that $y_1 > y_2$ and if $x_1 > x^*$ implies that $y_1 < y_2$.

That is, if both x_1 and x_2 are on the same side of x^*, the one nearer x^* gives the smaller value of y.

Thus any strictly convex function is unimodal. Indeed, unimodal functions need not even be continuous. The functions in Fig. 3 are unimodal functions of x in the intervals shown.

In searching for the minimum of a unimodal function it is desirable to use as efficient a method as possible. The most efficient search technique is defined to be that which minimizes the number of



(a) Smooth (b) Continuous (c) Discontinuous

Fig. 3. Unimodal functions.

functional evaluations required to locate the minimum, within some prescribed tolerance. For the method of this thesis, this criterion is satisfied by the Fibonacci search technique, so named because of its utilization of the Fibonacci numbers:

$$F_0 = F_1 = 1$$

$$F_m = F_{m-1} + F_{m-2} \qquad \text{for} \quad m > 1.$$

Suppose y is a unimodal function of x, where x lies in the interval $[a,b]$, and that we want to find the value of x, within some tolerance ε, which minimizes y. The Fibonacci search, in such a case, is the following. Determine the smallest Fibonacci number, F_n, such that

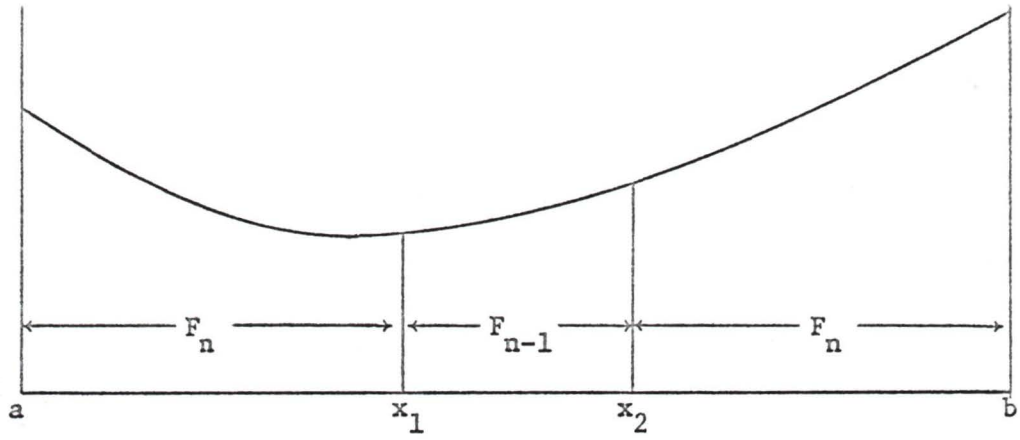
$$(2F_n + F_{n-1})\varepsilon \geq (b - a) \qquad \text{(I)}$$

and define

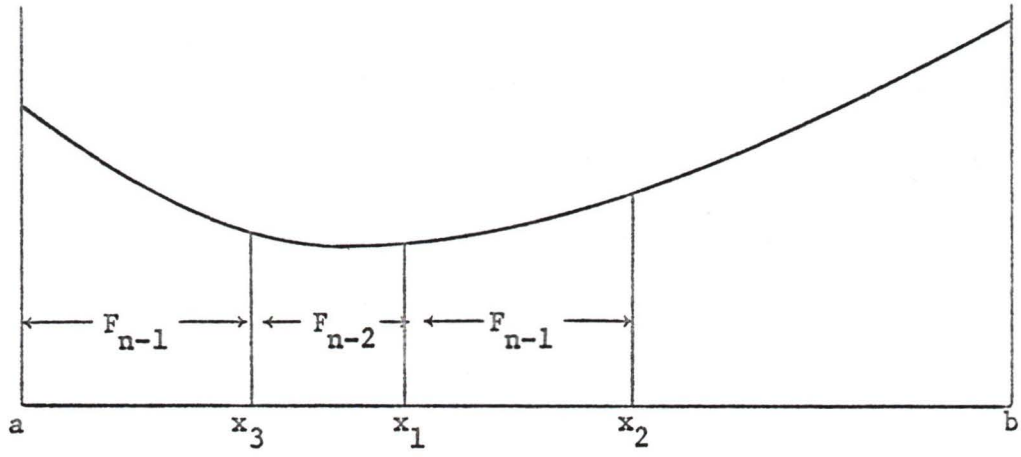
$$\varepsilon' = (b - a)/(2F_n + F_{n-1}). \qquad (J)$$

If we let ε' be our unit measure of length, then $[a,b]$ is $2F_n + F_{n-1}$, or F_{n+2}, units long. Evaluate the function at the points x_1 and x_2, which are located F_n units from a, and F_n units from b, respectively (see Fig. 4(a)). Now discard the interval $[a,x_1)$ or $(x_2,b]$ according as $y_1 > y_2$ or $y_1 < y_2$, leaving an interval of length F_{n+1}. Within this interval lies the trial which gave the smaller functional value of the previous two trials, this is located F_{n-1} units from one end. Now x_3 is placed F_{n-1} units from the other end of the interval, and we discard a portion of the interval in the above manner (in Fig. 4(b) discard $[a,x_3)$). The two trials in the interior of the remaining interval are now considered as the previous two trials.

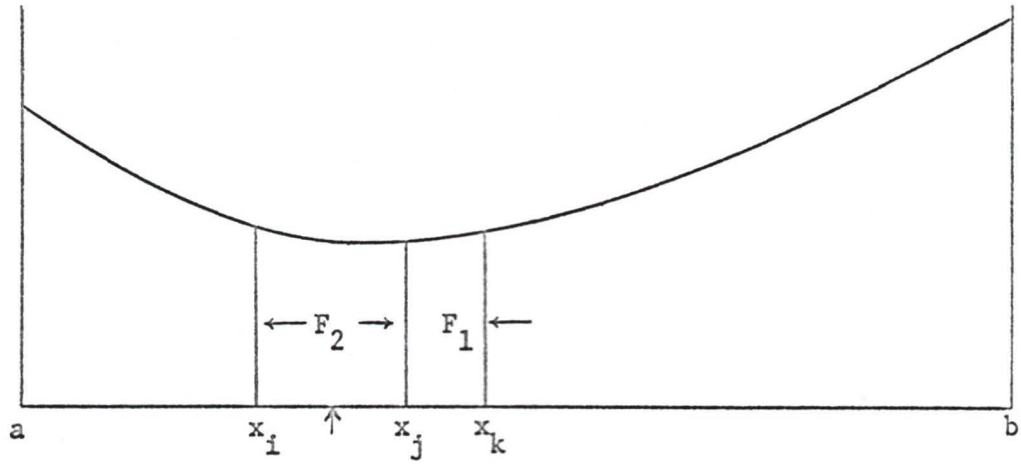
The above procedure is repeated, placing x_j at F_{n+2-j} units from one end of the interval currently under consideration. The situation immediately before the placement of the final trial (x_{n+1}) is shown in Fig. 4(c). The final trial is placed a distance F_1 from x_i (in this example) to create the situation shown in Fig. 5.



(a) Placement of the initial two trials.



(b) Placement of the third trial.



(c) Placement of the final trial.

Fig. 4. Placement of the trials for the Fibonacci search.

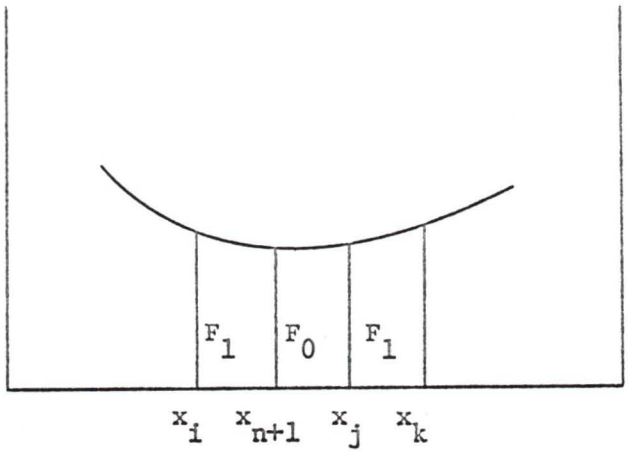


Fig. 5. Result of the placement of the final trial.

A portion of the interval $[x_i, x_k]$ is discarded as described above, and here we are left with the interval $[x_i, x_j]$. If x_{n+1} is chosen as that value of x which minimizes y then, from Fig. 5, the error is at most $\max(F_1,F_0)$. Thus, as $F_1 = F_0 = \varepsilon'$, x_{n+1} is within ε of the desired value of x ((I) and (J) imply $\varepsilon' \leq \varepsilon$).

The proof that the Fibonacci search technique is as efficient as any other method is contained in Bellman (1957). Specifically, he proves the following. Suppose that y is a unimodal function. Then the maximum length of the initial interval $[a,b]$, such that the minimum of y may be located within a unit subinterval in at most n functional evaluations, is F_n, the n^{th} Fibonacci number. Thus, by defining our initial interval to be of length F_m (where ε' is the unit length), the Fibonacci search technique is seen to be one of the

most efficient available. For example, to reduce an interval of uncertainty to less than 1% of its original length requires only 11 functional evaluations.

CHAPTER III

ALGORITHM AND RESULTS

III-1. The Algorithm and Its Implementation.

We now combine the material of the previous two chapters to construct an algorithm for the computation of best ℓ_1, ℓ_2, and ℓ_∞ approximations to discrete data by functions nonlinear in one of their several parameters. The algorithm is implemented as a two-stage process. Phase I is a search over a grid of values of the nonlinear parameter to find a small interval containing the minimum error of approximation. Phase II locates the minimum within this interval by using the Fibonacci search technique.

To test the method numerically, we selected twelve approximating functions which are nonlinear in just one of their several parameters (see Table I). For each one of these functions discrete best approximations were computed in the ℓ_1, ℓ_2 and ℓ_∞ norms to twelve different sets of data. Each data set consisting of 21 points with abscissae equally spaced on $[0,1]$, and ordinates recorded to 5D, was generated as follows.

(a) One set of exact data produced by assigning the parameters of the approximating function $F(A,x)$ particular numerical values.

(b) Three sets of noisy data produced as in (a) but with random errors added on. The distributions of these errors are, respectively,

Laplace: $g(\eta) = \exp(-2|\eta|)$ $\quad -\infty < \eta < \infty$

Normal: $g(\eta) = 1/\sqrt{\pi}\ \exp(-\eta^2)$ $\quad -\infty < \eta < \infty$

Uniform: $g(\eta) = 1$ $\quad -\frac{1}{2} \le \eta \le \frac{1}{2}$

(c) As (b) above but with different numerical values assigned to the parameters of the approximating function $F(A,x)$.

(d) $1 + \tan x$, $\sqrt{x}$, $\exp(-x^2)$, $\min(e^x, e^{\frac{1}{2}})$.

(e) 21 uniform random numbers in $[0,10]$.

Tables II through XXXVIII summarize the numerical study for the functions of Table I. The entire study was performed in double precision arithmetic on an IBM 360/44.

In Phase I the grid of values for the nonlinear parameter, c, was usually the 201 points defined as $c = -4(0.04)4$, although in certain cases this was changed (e.g. for $a_1 + a_2 \sin(cx)$ we used $c = 0(0.02)4$). This fine mesh size gives a good indication for the validity of the unimodality assumption over the entire interval. The computer printout from Phase I contains, for each c value, the error of approximation $G(c)$, and also a table of first differences for $G(c)$. Thus, it is a simple task to locate on $[-4,4]$ the minimum of $G(c)$ within 0.04, and also to check for unimodality. In those cases where $G(c)$ was decreasing at one end of the interval, Phase I was repeated with a larger range of c values.

In Phase II the optimum value c^* is located by searching the small interval of unimodality determined in Phase I. The Fibonacci search was used here, and in most cases the initial interval was of width 0.08. The tolerance within which c^* was located was 0.000001.

III-2. Comments on the Results.

The complete details of this numerical study, namely approximating functions, data to be approximated, and coefficients of best approximations, are contained in Tables I to XXXVIII inclusive. This set of tables is self-contained, but the following comments should prove helpful to the reader.

The magnitudes of the random errors of Tables X, XI and XIII are approximately one-tenth the magnitude of those in the other error tables. The original random errors in these cases were very large in comparison with the uncontaminated data. Consequently, when approximation to the noisy data was attempted, the parameters of the best approximations obtained bore very little resemblance to those of the uncontaminated data. The results in Table XVII indicate that, here too, the random errors imposed on the data are excessively large.

In both MINSUM and MINMAX, values of 10^{-8} and 10^{8} were assigned to TOLER and QMAX respectively (see Appendix), except for two instances. When attempting best ℓ_∞ approximations to f_5 and f_{10} with $F_{12}(A,x)$ (Table XXXVIII), we experienced some difficulty in appropriately choosing these arguments, and finally settled for 10^{-4} and 10^{4}.

Tables XXI, XXII, XXV, XXVI, XXXII and XXXIII contain examples for which no best approximations were found. It is apparent that in these cases best approximations do not exist. Consider, for example, the attempted approximation of $1 + \tan x$ on $[0,1]$ by $a_1 + a_2 \sin(cx)$. The error of approximation, in all three norms, decreases as c

approaches zero, and a_2 increases without bound. This decrease in the error of approximation is attributable to the "straightening-out" of the curve $a_1 + a_2 \sin(cx)$. However, in the limit when c equals zero, we are left with an approximating function which is merely a straight line parallel to the x-axis, thus no best approximation exists. This same situation occurs in the remaining cases for which we found no best approximation.

In all other instances best approximations were found and the results are tabluated in the following tables, rounded to 5D. The only exceptions occur in Tables XXIX, XXX and XXXI, where some coefficients are shown rounded to 4D. This occurred because the magnitude of these results was greater than 100 and the format of the computer output was a seven digit number in $[0,1,1)$ multiplied by a power of ten.

In those cases where a best approximation does not exist, this phenomenon can usually be detected by inspection of the output from Phase I. For all instances where best approximations do exist, the error of approximation appears to be unimodal in a sizable region enclosing the minimum, and in many cases it is unimodal in the entire range considered. It is important to stress that these remarks are based on the observed behavior of $G(c)$ for only a discrete set of values of c. However, we feel justified in claiming that the unimodality assumption required for Phase II is most reasonable in practice.

As might be expected, the last data sets (i.e. 21 random numbers) are difficult to approximate closely. The output of Phase I indicates

that $G(c)$ does not vary greatly for these cases, and appears to contain many local minima. Consequently we cannot guarantee that our results for these functions represent best approximations, and we do not display them. These random data sets were included to test the limitations of the algorithm.

Since our method is guaranteed to produce best approximations only if the unimodality assumption is valid, we have verified that the ℓ_∞ results shown for the rational functions are, in fact, best approximations, by using the characteristic error equioscillation property (see Rice (1964)). In the ℓ_2 case we checked the results using the normal equations. No such characterization theorems are available for any of the ℓ_1 results or for most of the nonrational ℓ_∞ results.

Finally, we comment on a comparison between our method and that which results from regarding the approximation problem as a pure minimization problem. The error of approximation is a function of the n variables $a_1, a_2, \ldots, a_{n-1}, c$ and this can be minimized by using any of the known numerical techniques for locating the minimum of a function of several variables. In the ℓ_1 and ℓ_∞ cases the derivative of the error of approximation is not guaranteed to exist everywhere, so we are restricted to optimizing techniques which do not require derivatives. Powell (1964) has developed such a scheme and we used it with success on several of our examples. However, a disadvantage of any of these pure optimization techniques is that they can converge to a local, rather than a global, minimum. Phase I of our procedure ensures that any such failure during Phase II is limited to the small

region of c values over which the Fibonacci search is conducted. The major advantage of our method is that it is essentially a one-dimensional procedure rather than an n-dimensional routine. This fact makes the task of selecting upper and lower limits on the activities of the variables much simpler for our method, since we only have to limit c itself. Also, post-computational analysis on our method is far simpler than that which is required to investigate (say) intermediate calculations by any n-dimensional optimization routine. Being able to regard our method as a one-dimensional procedure is both convenient computationally and quite enlightening analytically.

TABLE I Approximating functions $F(A,x)$ considered in the numerical study

1. $a_1 + a_2\, e^{cx}$
2. $(a_1 + a_2\, x)/(1 + cx)$
3. $(a_1 + a_2\, x + a_3\, x^2)/(1 + cx)$
4. $a_1 + a_2 \log(1 + cx)$
5. $a_1 + a_2 \sin(cx)$
6. $a_1 + a_2/(1 + x)^c$
7. $a_1 \sin(cx) + a_2 \cos(cx)$
8. $e^{cx}(a_1 + a_2\, x)$
9. $e^{cx}(a_1 + a_2\, x + a_3\, x^2 + a_4\, x^3)$
10. $a_1 \sinh(cx) + a_2 \cosh(cx)$
11. $a_1 + a_2\, x + a_3(x - c)_+$
12. $a_1 + a_2\, x + a_3\, x^2 + a_4\, x^3 + a_5(x - c)_+^3$

TABLE II : Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_1(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	0.51651	-0.54135	0.23631	-1.20402	-0.10997	0.25249
0.05	0.47907	0.78831	-0.49320	-0.17211	-1.23824	0.05855
0.10	-0.02171	-0.37354	0.15275	0.29198	-0.59018	0.05102
0.15	-0.19579	0.82389	0.20048	0.04200	-0.44674	0.21525
0.20	-0.00102	1.28168	-0.46178	0.12298	0.54730	0.07166
0.25	0.05018	0.49845	-0.37861	0.05341	-0.22270	0.29025
0.30	0.61789	0.56377	0.36781	-0.66038	-0.53880	0.47352
0.35	-0.03164	-0.93252	0.04276	-0.51129	-0.07485	0.42835
0.40	-1.29143	0.89289	-0.23517	0.20794	-0.16399	0.15086
0.45	0.15593	-0.86129	-0.11651	-0.06279	0.76596	-0.20515
0.50	-0.93416	-1.38916	-0.36255	0.30643	-0.10776	0.09753
0.55	1.47886	-0.24175	0.40495	-0.82621	0.23334	0.13096
0.60	0.91492	-0.84553	-0.02298	0.42480	-0.35575	0.37774
0.65	-0.90323	0.30514	0.21117	-0.05408	0.15423	-0.35425
0.70	1.39448	-0.00532	0.42856	0.11619	0.11988	-0.12212
0.75	0.01634	0.72844	-0.13176	0.67608	0.15818	-0.27772
0.80	0.15500	0.52944	-0.21731	-0.42411	-0.62447	-0.11840
0.85	0.43126	-0.49357	0.45020	-0.51213	-0.20302	-0.35819
0.90	0.06739	-0.49943	-0.35992	-0.02077	-0.22218	0.06454
0.95	-0.32046	0.45660	-0.00106	1.40129	0.73224	0.31943
1.00	-0.97907	0.16417	0.16343	-0.38218	0.06166	0.24781

TABLE III . Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_2(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	0.12119	-0.05978	0.39576	-0.03751	-0.15878	-0.39276
0.05	0.59138	-0.20149	-0.32132	0.57285	0.14117	0.37623
0.10	1.63373	0.79925	0.44350	-0.23326	0.54565	-0.04967
0.15	-0.01027	0.91120	0.22425	0.94078	1.05350	0.14031
0.20	0.92979	0.15312	0.02417	0.03985	-0.57894	0.43535
0.25	-0.22861	-0.13592	0.16845	-0.41057	0.59835	0.26050
0.30	-0.37289	0.30655	0.36961	0.98547	-0.39153	-0.29630
0.35	-0.81332	0.57667	-0.20486	0.36237	0.63852	-0.04687
0.40	-0.27248	-0.09552	0.06392	-0.57615	-0.08338	-0.34234
0.45	-0.32600	-0.37644	0.11942	0.83611	-0.55820	0.21025
0.50	0.05851	-0.39987	-0.23385	-0.74548	-0.09386	-0.18647
0.55	0.06301	0.24062	0.49689	0.73964	-0.32734	-0.40369
0.60	1.00156	-0.88356	0.24025	0.30710	-0.38076	-0.08379
0.65	0.69787	0.00819	-0.10527	0.26385	-0.71750	-0.38699
0.70	-0.11041	0.08635	0.24790	-1.04985	0.42724	-0.42369
0.75	0.57299	0.23558	-0.14875	-0.60061	0.30996	0.49232
0.80	-0.13291	0.72789	-0.28855	-0.94504	0.66792	-0.00517
0.85	0.00631	-0.40999	0.07986	-0.02518	0.10825	0.31742
0.90	-0.15681	0.57207	0.46839	-0.75938	-0.15849	0.33342
0.95	0.99279	0.11019	-0.20942	1.82531	-0.17359	-0.02777
1.00	1.07695	-0.16066	0.41499	0.06650	-0.02672	0.46714

TABLE IV . Random errors used in defining f_2, f_3,..., f_7 for $F_3(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	0.19059	-0.38053	0.08484	-0.33891	-0.24605	0.00155
0.05	-1.07269	-0.47727	0.22456	0.65170	0.15751	-0.48258
0.10	-0.04689	-0.30327	0.35625	-0.67214	0.17513	-0.25062
0.15	-0.53017	-0.40966	0.27413	-0.34406	-0.56216	0.02170
0.20	-0.34104	0.45760	-0.04486	-0.43629	0.29658	-0.45238
0.25	0.03884	-1.36293	0.05833	-1.97619	0.01218	-0.06055
0.30	0.36511	0.15458	-0.48907	0.36708	0.10596	0.21360
0.35	1.09272	0.54823	0.10655	-0.02100	0.11476	0.11535
0.40	0.06109	-0.32305	0.15141	-0.05681	0.87167	-0.04530
0.45	-0.45604	0.30931	0.17505	-0.75914	0.18717	-0.45042
0.50	-1.47089	0.48102	-0.49224	0.69052	-0.00692	-0.10734
0.55	-0.28340	-0.08167	0.46685	-0.86621	0.03517	-0.18810
0.60	-0.26079	0.35854	0.14887	1.79275	0.21097	-0.31548
0.65	-0.27826	-0.75400	0.17672	-0.64246	0.08032	-0.08075
0.70	0.61649	0.18414	0.00851	0.37963	0.03632	-0.22581
0.75	0.43751	1.38203	0.28596	0.61004	0.29369	0.48248
0.80	-0.67455	0.02690	-0.46724	-0.04519	-0.21780	0.05128
0.85	0.39165	-0.21179	0.39919	-0.65474	0.34519	-0.22050
0.90	0.37699	-0.45345	0.00198	-0.38909	-0.56147	-0.37709
0.95	0.58520	-0.47392	-0.03041	0.40339	0.13463	0.32605
1.00	-0.12152	0.43648	0.27029	1.20061	-0.81176	-0.47797

TABLE V : Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_4(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	-0.21373	-0.27953	-0.00789	0.03376	0.34672	0.12159
0.05	0.15655	0.93116	-0.01992	0.05199	-0.67784	0.26470
0.10	0.14270	-0.17910	0.25066	-0.37737	-0.10660	0.00079
0.15	-0.36543	-0.42516	-0.11595	0.52533	-0.75646	0.06694
0.20	-0.13785	-0.13251	0.24741	-0.08013	0.22962	-0.14512
0.25	0.69990	-0.82658	-0.28220	0.58490	-0.88530	-0.39956
0.30	0.58545	-0.05573	-0.45604	0.76297	-1.00964	0.06352
0.35	-0.42803	0.39179	0.15462	0.73130	-0.02808	0.23298
0.40	0.00206	-0.27135	0.11355	-0.37721	0.22475	-0.42672
0.45	-0.07016	-0.33982	0.02278	-0.04512	-0.07307	0.46593
0.50	-1.47801	-0.23939	-0.04243	0.38097	1.00528	-0.07570
0.55	0.41573	-0.34623	0.10779	0.14440	0.31466	0.41198
0.60	-0.27300	+0.63704	0.06241	0.57804	-1.02189	0.32564
0.65	-0.25935	0.93565	-0.31499	-0.45713	0.81922	0.10835
0.70	-0.54338	0.71417	-0.16931	-0.82290	0.43277	-0.22551
0.75	0.57599	-0.08480	0.29346	-0.06703	-0.04228	0.47909
0.80	-0.17241	0.07661	0.09701	2.23628	-0.43699	-0.10605
0.85	0.79827	0.27441	-0.43403	1.41046	0.13668	-0.07604
0.90	-0.88817	-0.25130	-0.12232	0.16978	0.08865	0.03150
0.95	-0.70170	-1.36921	0.42363	-0.24359	0.02290	0.23804
1.00	1.59660	-0.18718	-0.39446	0.30427	0.00733	-0.28530

TABLE VI : Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_5(\Lambda,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	1.34163	-0.18470	0.35509	-0.43632	-1.08456	0.34963
0.05	-0.14032	0.62324	0.30312	1.46177	0.64411	-0.06368
0.10	-0.18636	0.30612	-0.24041	0.08808	-0.69291	-0.44276
0.15	0.60257	0.21639	0.10496	-1.85919	-0.28400	-0.22631
0.20	0.31663	-0.28848	-0.17387	0.87689	-0.36244	0.42517
0.25	-0.05709	-0.38522	-0.15590	0.09181	0.09618	-0.04939
0.30	0.22259	-0.71617	0.32143	0.03371	0.43006	-0.14749
0.35	-0.32016	-0.23572	-0.33185	0.08089	-0.53033	-0.07234
0.40	1.02817	-0.29950	0.34366	-0.24502	0.06048	-0.03365
0.45	-0.03533	0.30516	0.37986	0.14077	-0.47171	-0.04936
0.50	0.69582	0.81313	-0.42530	0.33449	-0.85121	-0.18220
0.55	-0.08911	0.02645	-0.24550	0.12083	0.23784	0.17086
0.60	0.67071	0.21225	0.38563	0.08751	-0.40130	-0.36654
0.65	0.03860	-0.39351	0.17240	-2.15396	0.34060	-0.23770
0.70	-1.05685	-0.59789	-0.43798	-0.14139	0.65476	0.26604
0.75	-0.50757	-0.00774	-0.14886	-0.46482	0.06232	0.41913
0.80	-0.28469	-0.32808	-0.29547	0.08341	-0.22379	0.10482
0.85	-0.46273	-0.23477	0.20343	-0.01847	0.34968	-0.40582
0.90	0.53588	1.14625	0.21220	0.62499	-0.10293	0.01135
0.95	-0.77341	0.23023	-0.22300	0.68174	0.34945	-0.12244
1.00	0.20679	0.08295	-0.33030	-0.73703	-0.04600	0.33804

TABLE VII Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_6(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	0.37349	-0.60422	-0.48339	0.27662	0.83082	-0.02045
0.05	0.35299	-0.14771	0.26697	1.75087	-0.43113	-0.26211
0.10	-0.31010	-0.18446	-0.25792	0.18924	0.09552	0.01970
0.15	0.34568	0.04596	0.07338	-0.08648	0.10209	-0.15116
0.20	-0.39248	-0.32093	-0.20781	-0.01923	0.19272	0.30011
0.25	-0.35817	0.59009	-0.02705	1.07997	0.40497	-0.35416
0.30	-0.02502	-0.62910	-0.18476	0.34265	0.33472	-0.14411
0.35	-0.22710	-0.30959	0.45442	-0.17840	-0.28752	-0.47832
0.40	-0.39608	-0.38370	-0.31703	0.03160	0.14580	-0.03822
0.45	-0.48438	0.02850	-0.07159	-0.08203	-0.37187	0.46861
0.50	0.03200	0.58698	-0.33902	-1.60073	0.49358	-0.31219
0.55	0.39203	0.22658	-0.35919	-0.74967	-0.57636	0.11584
0.60	0.94572	0.32931	0.35908	1.35595	0.55572	0.00961
0.65	0.40150	0.09061	-0.39641	0.06074	-0.90481	-0.47359
0.70	0.00985	1.46300	-0.07655	0.03783	-0.09611	0.45274
0.75	0.33364	-0.42269	0.28825	-1.16673	-0.23354	-0.22986
0.80	0.17726	-0.56629	-0.46596	0.53248	0.20008	0.30526
0.85	0.07227	-0.58405	-0.49344	-0.73258	-0.38712	-0.42191
0.90	-0.05207	-0.57247	0.43495	0.14866	0.09632	-0.32512
0.95	1.54711	0.26470	-0.22929	-0.62135	0.03291	0.23611
1.00	-0.16400	0.12581	0.45155	0.62316	-0.22562	-0.40316

TABLE VIII : Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_7(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	-0.07082	-0.49830	-0.21160	-0.05533	0.13356	-0.30618
0.05	-0.16385	-0.18177	-0.48441	-0.06452	0.20531	-0.31803
0.10	-0.34612	-0.55349	0.09920	-0.53755	0.64269	0.20798
0.15	-0.04511	-0.29938	0.26840	-0.54672	-0.57880	-0.44397
0.20	0.34373	-0.63748	-0.14063	0.28915	-0.12730	-0.06549
0.25	-0.03703	-0.19033	-0.30913	-0.20816	-0.44870	0.13356
0.30	-0.42588	0.71760	-0.27353	-0.20008	-0.62031	-0.19197
0.35	0.64204	-0.70543	-0.35582	1.54990	-0.95970	0.26359
0.40	0.59579	-0.19742	-0.04931	-0.93390	-0.29481	-0.40178
0.45	-0.29514	0.04158	0.13108	0.39903	0.33825	0.08745
0.50	-0.18531	1.03345	-0.43405	0.06431	0.56610	0.22662
0.55	-0.91801	-0.12980	-0.27759	0.39917	0.87500	0.02561
0.60	-0.34167	0.16123	0.01419	-0.01388	-0.32699	-0.43325
0.65	0.04509	-0.42645	0.25774	0.41332	0.10544	0.20062
0.70	0.54669	-0.29233	0.42254	-0.23006	-0.16147	0.23039
0.75	-0.29414	0.54747	0.21398	-0.17850	0.31538	-0.49177
0.80	-0.14099	-0.37416	0.16739	0.13689	0.10835	-0.16481
0.85	0.10224	-0.62360	-0.24742	-0.09495	-0.42361	-0.35005
0.90	0.14605	-0.07414	0.33540	0.65856	-0.82773	-0.07155
0.95	0.28382	-0.08111	-0.23935	1.81876	-0.29885	-0.39523
1.00	-1.27809	0.29781	-0.26890	-0.58862	0.22450	-0.39099

TABLE IX Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_8(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	0.34464	-0.37963	-0.14763	-0.52191	0.16556	-0.46142
0.05	-0.01469	-0.40151	0.10816	-0.31042	-0.20819	0.21715
0.10	0.10465	0.14866	0.07820	0.18970	0.23001	-0.21276
0.15	0.66083	-0.04920	0.24931	-0.39305	-0.17855	0.05376
0.20	-1.67508	-0.48068	-0.25437	-0.35011	0.18327	0.16663
0.25	3.22449	0.31845	0.07638	0.28124	1.16837	0.47482
0.30	0.47161	0.10897	-0.30739	-0.57792	-0.79361	0.42786
0.35	0.10472	-0.00855	-0.46019	0.04094	0.21362	0.48740
0.40	0.78223	-0.05703	-0.39045	0.30195	-0.36840	-0.00588
0.45	-0.14218	-0.58983	-0.13789	0.08112	0.94740	-0.21676
0.50	0.31903	0.06721	-0.32700	0.33521	0.60502	-0.21497
0.55	2.37661	0.69482	-0.47739	0.44074	0.42423	0.06467
0.60	-0.02983	0.13134	-0.15111	0.49177	-0.02287	0.42079
0.65	-0.24183	-0.99991	-0.05567	-0.74279	0.06394	-0.34756
0.70	0.11859	0.52706	-0.21184	0.24344	-0.86456	-0.08158
0.75	-0.31890	0.52690	0.13572	0.16342	0.27019	-0.33463
0.80	0.03851	0.98278	-0.22325	0.21811	0.92972	0.35952
0.85	0.91127	0.00334	-0.46752	-0.90335	0.86227	0.28863
0.90	-0.59238	-0.12782	0.30335	-0.73507	0.24319	0.06374
0.95	-0.33869	0.27796	-0.03566	0.77176	0.25883	-0.06397
1.00	0.19476	-0.17301	-0.02749	-0.03468	0.04938	0.04445

TABLE X Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_9(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	-0.19316	0.08272	-0.00361	0.04929	-0.13948	-0.01102
0.05	0.02935	0.01541	0.04525	0.04632	-0.04806	-0.03446
0.10	-0.07458	-0.05743	-0.04928	-0.08491	-0.03843	0.01983
0.15	-0.02087	0.07012	-0.01687	-0.00961	0.07611	-0.03181
0.20	-0.06971	-0.03424	0.04872	0.14153	0.06667	-0.03044
0.25	0.02356	0.05963	-0.01885	0.04837	-0.00140	0.03499
0.30	0.03018	-0.02434	0.03749	0.08725	0.03071	-0.00088
0.35	-0.03104	-0.04704	-0.01352	0.12915	-0.04051	0.00709
0.40	-0.04608	-0.01790	0.04024	0.05619	-0.02351	0.03480
0.45	-0.12675	0.03886	0.04171	-0.13999	0.01681	-0.04828
0.50	-0.00436	0.03731	0.00709	0.04512	-0.02775	-0.03867
0.55	-0.11268	-0.02902	0.04883	-0.00041	-0.05830	-0.03338
0.60	0.01045	-0.01074	-0.04625	-0.01075	0.00871	-0.01289
0.65	0.10506	0.04702	-0.04209	-0.00492	0.11561	0.04199
0.70	0.04947	0.10419	0.02472	0.00051	0.12105	0.00157
0.75	-0.00103	0.04813	0.02708	0.04779	0.06493	-0.00267
0.80	-0.00578	0.01135	-0.04696	-0.01092	0.06393	0.02493
0.85	-0.07045	-0.03371	0.04294	0.18259	-0.05793	-0.03569
0.90	0.05265	0.00185	0.02996	-0.00293	0.02150	0.04973
0.95	-0.03733	-0.02969	-0.04740	-0.08238	-0.06143	-0.00292
1.00	0.02439	-0.00749	-0.00438	-0.00670	-0.06940	0.04189

TABLE XI . Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_{10}(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	0.11692	0.12464	0.00772	0.01574	0.00273	-0.03231
0.05	-0.17736	-0.02967	-0.04484	0.04379	0.05180	0.01208
0.10	0.11128	-0.03432	-0.00276	0.00945	-0.01940	-0.04023
0.15	-0.02633	0.02568	-0.03105	0.05111	-0.01325	-0.03145
0.20	-0.09365	0.02304	0.03150	0.03492	-0.01699	0.04654
0.25	0.05520	-0.00186	0.03314	0.00481	0.04405	0.04154
0.30	0.01402	0.09302	0.04647	0.02658	0.03359	-0.04972
0.35	0.12454	0.05784	0.04698	0.00833	0.00312	-0.03670
0.40	-0.10304	-0.00997	0.00874	-0.00848	0.07964	0.00073
0.45	-0.08555	-0.02225	-0.03951	0.00616	-0.09146	0.02556
0.50	-0.06008	-0.00944	-0.04523	0.10701	-0.03993	0.00078
0.55	0.36163	0.05835	0.01740	-0.02896	-0.01174	0.02463
0.60	-0.00034	0.02579	0.03639	-0.04368	0.01345	-0.03349
0.65	-0.14833	-0.01640	0.00802	0.00777	0.01039	0.00969
0.70	-0.05050	-0.01200	-0.03990	0.06620	0.03618	-0.01094
0.75	-0.03356	0.00207	0.01815	0.01416	-0.05045	-0.00350
0.80	-0.04373	0.08769	-0.02583	-0.00393	-0.00445	0.03941
0.85	-0.13476	-0.09746	-0.00297	-0.01992	-0.05436	0.04311
0.90	0.10830	-0.01972	-0.01979	-0.07803	0.01934	-0.01938
0.95	0.00200	-0.03454	-0.02725	-0.02826	0.02803	0.02617
1.00	-0.01337	0.02090	-0.02515	-0.05009	-0.01425	0.01361

TABLE XII Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_{11}(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	-1.42571	-0.04270	-0.46102	-0.21007	-0.08975	0.09680
0.05	-0.74094	-0.01788	0.12897	-0.32142	0.53686	0.45794
0.10	0.23348	0.63560	-0.44565	-0.83093	-0.58949	-0.35963
0.15	0.67802	0.19556	-0.41871	0.51169	0.52175	0.11130
0.20	0.37167	-0.08527	0.46950	-0.02547	-0.73921	-0.16984
0.25	0.33945	-0.25045	-0.14837	-0.00092	1.27958	-0.46960
0.30	-1.09341	-0.19758	0.40799	0.14055	0.05806	0.37984
0.35	-0.42014	0.90498	0.17105	0.05548	-0.81558	0.20785
0.40	0.95636	0.47341	-0.13000	-0.23036	0.04207	-0.42985
0.45	0.23394	0.70846	0.41222	0.07647	-0.20999	-0.14114
0.50	-0.76373	0.05140	-0.34944	0.51281	0.23657	-0.10019
0.55	-1.20200	0.10927	0.21889	-0.04845	-0.26484	-0.46339
0.60	0.96337	0.10104	0.29353	-0.33607	0.76291	0.12085
0.65	0.58217	0.35894	0.32665	-0.26144	0.53412	0.21603
0.70	-0.09034	0.43446	0.02765	-0.17482	-0.54361	0.09115
0.75	-0.18258	-2.09842	0.02313	0.20334	0.86383	-0.02813
0.80	0.45284	-0.47032	-0.39250	-0.78463	-0.18971	0.05661
0.85	-1.05492	0.16963	-0.49927	-0.23812	0.44684	-0.45874
0.90	-0.50091	-0.14758	0.02130	0.00087	-0.46018	0.34234
0.95	0.55245	-0.62718	-0.15310	0.04394	0.37917	0.28722
1.00	0.30640	-0.68106	0.12894	0.41509	0.14958	0.48714

TABLE XIII : Random errors used in defining $f_2, f_3, \ldots, f_7$ for $F_{12}(A,x)$

x	ε_L^1	ε_N^1	ε_U^1	ε_L^2	ε_N^2	ε_U^2
0.0	0.01655	0.04724	-0.02559	0.01627	0.01522	0.01371
0.05	-0.05743	-0.01796	0.04438	0.02913	0.00705	-0.00288
0.10	0.01199	-0.03256	-0.04172	-0.02551	-0.05573	0.02621
0.15	0.01287	0.02652	0.00185	-0.08801	0.04440	0.03323
0.20	0.03144	-0.04059	-0.04883	-0.02383	-0.05384	0.00433
0.25	-0.01194	0.02435	-0.01105	0.02947	-0.00573	0.04218
0.30	0.06969	-0.01402	0.02699	-0.00129	-0.04409	-0.02017
0.35	0.05179	0.00682	0.03252	-0.02579	-0.02205	0.00516
0.40	-0.00659	-0.01235	-0.01082	0.02881	0.01877	-0.01025
0.45	-0.00646	0.04894	-0.04758	0.00463	0.03808	0.01521
0.50	0.00058	-0.04233	-0.00957	-0.11457	0.00040	-0.03590
0.55	-0.00278	0.07619	0.04957	-0.01555	-0.02359	0.00611
0.60	0.01800	-0.05118	0.01496	0.01609	-0.05390	0.04163
0.65	-0.01578	-0.00471	-0.00373	0.05573	-0.09704	-0.03484
0.70	0.01169	-0.07546	0.02733	-0.10013	0.09530	-0.01522
0.75	-0.04792	0.03451	0.00177	-0.05266	0.04343	-0.02303
0.80	-0.02272	0.00337	0.01451	-0.03089	-0.06157	0.00835
0.85	0.00936	-0.12065	0.03162	-0.04479	0.03128	-0.01355
0.90	-0.00483	0.05278	-0.02328	0.05639	0.12284	-0.00211
0.95	-0.00366	0.02669	-0.01672	0.01527	0.00513	-0.03278
1.00	0.04419	0.00922	0.00778	-0.07858	-0.10208	0.02436

TABLE XIV Complete details of best ℓ_1 and ℓ_2 approximations computed when using $\Gamma_1(A,x) = a_1 + a_2 e^{cx}$

Define $E_i = |f(x_i) - F(A^*, x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
f_1:	$1 + 1\, e^{1x}$	0.99999	1.00001	1.00000	0.00000	0.99999	1.00001	0.99999	0.00000
f_2:	$-10 + 2e^{2x} + \varepsilon_L^1$	-10.80917	2.72158	1.71351	0.49828	-10.74476	2.72629	1.71116	0.68243
f_3:	$-10 + 2e^{2x} + \varepsilon_N^1$	- 8.44705	1.23643	2.38223	0.58618	- 8.94914	1.30723	2.37388	0.69028
f_4:	$-10 + 2e^{2x} + \varepsilon_U^1$	- 9.48696	1.68105	2.14160	0.24929	-10.23755	2.14311	1.94916	0.29466
f_5:	$-5 + 10e^{-2x} + \varepsilon_L^2$	- 5.88353	10.56998	-1.59494	0.38624	- 4.99481	9.50719	-1.89074	0.51828
f_6:	$-5 + 10e^{-2x} + \varepsilon_N^2$	- 5.37940	9.41459	-1.65758	0.30315	- 5.22533	9.58279	-1.79072	0.41311
f_7:	$-5 + 10e^{-2x} + \varepsilon_U^2$	- 5.02227	10.27476	-1.97325	0.18700	- 5.21311	10.41069	-1.92306	0.22763
f_8:	$1 + \tan x$	0.44703	0.58274	1.26119	0.01247	0.48684	0.54189	1.32404	0.01554
f_9:	$\sqrt{x}$	1.29646	-1.11441	-1.28289	0.01524	1.11579	-1.02455	-1.93842	0.02941
f_{10}:	e^{-x^2}	1.51780	-0.47076	0.91533	0.01242	1.41827	-0.38443	1.02862	0.01516
f_{11}:	$\min(e^x, e^{\frac{1}{2}})$	1.73937	-0.83738	-2.77913	0.03974	1.74756	-0.81581	-2.80279	0.04721

TABLE XV Complete details of best ℓ_∞ approximations computed when using $\Gamma_1(A, x) = a_1 + a_2 e^{cx}$

Define $E_i = |f(x_i) - F(A^*, x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	max E_i
f_1	$1 + 1\, e^{1x}$	0.99999	1.00001	0.99999	0.00001
f_2	$-10 + 2e^{2x} + \varepsilon_L^1$	-12.36233	3.61711	1.57212	1.26173
f_3	$-10 + 2e^{2x} + \varepsilon_N^1$	- 8.30800	0.85150	2.79259	1.08485
f_4	$-10 + 2e^{2x} + \varepsilon_U^1$	-10.42586	2.29199	1.89005	0.39715
f_5	$-5 + 10e^{-2x} + \varepsilon_L^2$	- 3.96391	8.19509	-2.26948	0.91201
f_6	$-5 + 10e^{-2x} + \varepsilon_N^2$	- 5.57100	9.86067	-1.63200	0.70686
f_7	$-5 + 10e^{-2x} + \varepsilon_U^2$	- 5.03977	10.53481	-2.05499	0.35942
f_8	$1 + \tan x$	0.49179	0.53075	1.34793	0.02254
f_9	$\sqrt{x}$	1.01652	-0.96440	-2.64262	0.05212
f_{10}	e^{-x^2}	1.34225	-0.31999	1.13608	0.02226
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	1.77214	-0.84851	-2.89244	0.07637

TABLE XVI Complete details of best ℓ_p approximations computed when using $F_2(A,x) = (a_1 + a_2x)/(1 + cx)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma L_i^2}{21}}$	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	max
f_1: $\frac{1 + 2x}{1 + x}$	1.00000	1.99999	0.99999	0.00000	1.00000	1.99998	0.99999	0.00000	1.00000	2.00004	1.00003	0.00
f_2: $\frac{3 + 15x}{1 + 2x} + \varepsilon_L^1$	3.66230	1.15067	-0.31991	0.42187	3.87611	2.20955	-0.09485	0.54768	4.10302	5.57967	0.58859	0.98
f_3: $\frac{3 + 15x}{1 + 2x} + \varepsilon_N^1$	2.94022	19.82145	2.77740	0.34060	3.12648	19.02909	2.72056	0.43333	3.70285	5.09068	0.44487	0.7
f_4: $\frac{3 + 15x}{1 + 2x} + \varepsilon_U^1$	3.39576	13.05763	1.70736	0.21289	3.22852	13.46381	1.73528	0.26230	3.01563	17.11797	2.31248	0.3
f_5: $\frac{2 + 3x}{1 - \frac{1}{2}x} + \varepsilon_L^2$	1.96249	2.86899	-0.52004	0.58200	2.18411	1.89889	-0.60894	0.71116	2.16127	1.51618	-0.67130	1.1
f_6: $\frac{2 + 3x}{1 - \frac{1}{2}x} + \varepsilon_N^2$	2.17153	2.17032	-0.56465	0.37168	2.21005	2.34352	-0.54836	0.45779	2.45140	1.16545	-0.66382	0.7
f_7: $\frac{2 + 3x}{1 - \frac{1}{2}x} + \varepsilon_U^2$	2.13059	2.20236	-0.58604	0.23153	2.09546	2.29602	-0.57978	0.26827	2.04776	2.56249	-0.56086	0.4
f_8: $1 + \tan x$	1.02404	0.30111	-0.47446	0.00909	1.02267	0.28974	-0.48255	0.01104	1.01596	0.30265	-0.48116	0.0
f_9: $\sqrt{x}$	0.17061	1.75785	0.95375	0.01331	0.06815	2.53883	1.68288	0.02469	0.04297	3.18124	2.36898	0.0
f_{10}: e^{-x^2}	1.04932	-0.83400	-0.36501	0.01327	1.03799	-0.82712	-0.38542	0.01637	1.02426	-0.82747	-0.42732	0.0
f_{11}: $\min(e^x, e^{\frac{1}{2}})$	0.94572	4.27947	2.06264	0.04524	0.93510	4.64232	2.24602	0.05405	0.91256	5.52770	2.70948	0.0

TABLE XVII Complete details of best ℓ_1 and ℓ_2 approximations computed when using $F_3(A,x) = (a_1 + a_2x + a_3x^2)/(1 +$

Define $I_i = |f(x_i) - F(A^*,x_i)|$

DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	a_3^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	a_3^*	c^*	$\sqrt{\frac{\Sigma L_i^2}{21}}$
f_1: $\frac{1 - 3x + 2x^2}{1 + 5x}$	1.00000	- 2.99998	1.99995	5.00003	0.00000	1.00000	- 3.00000	1.99998	4.99997	0.0000
f_2: $\frac{10 - 30x + 20x^2}{1 + 5x} + \varepsilon_L^1$	10.19059	-38.11294	31.14902	4.26187	0.42625	9.88242	-31.31345	22.37973	5.57528	0.5622
f_3: $\frac{10 - 30x + 20x^2}{1 + 5x} + \varepsilon_N^1$	9.61947	-21.05408	9.08975	6.99939	0.36209	9.65021	-26.35040	16.25044	6.41084	0.5318
f_4: $\frac{10 - 30x + 20x^2}{1 + 5x} + \varepsilon_U^1$	10.08484	-29.89466	21.28819	4.46954	0.18520	10.14497	-31.28110	22.05553	4.53868	0.2585
f_5: $\frac{6 - 11x + 4x^2}{1 - 3x} + \varepsilon_L^2$	5.46005	- 8.20708	0.82913	-3.00260	0.56807	5.52184	- 8.74013	1.53686	-3.00071	0.7613
f_6: $\frac{6 - 11x + 4x^2}{1 - 3x} + \varepsilon_N^2$	6.16442	-11.63033	4.39992	-2.99992	0.23438	6.16455	-11.85755	4.91384	-2.99933	0.3386
f_7: $\frac{6 - 11x + 4x^2}{1 - 3x} + \varepsilon_U^2$	5.86439	-10.58950	4.17764	-3.00193	0.18230	5.76898	-10.05427	3.41059	-3.00186	0.2407
f_8: $1 + \tan x$	1.00436	0.24449	-0.45893	-0.69167	0.00124	1.00347	0.25046	-0.45845	-0.68917	0.0015
f_9: $\sqrt{x}$	0.00000	5.15739	4.17474	8.26647	0.00458	0.00522	5.79408	5.43303	10.12706	0.0067
f_{10}: e^{-x^2}	0.99082	2.59318	-2.39593	2.34559	0.00356	0.99538	2.62482	-2.40339	2.40051	0.0043
f_{11}: $\min(e^x, e^{\frac{1}{2}})$	0.94771	1.54978	-1.06003	-0.11836	0.02549	0.95724	1.44625	-1.15339	-0.22602	0.0317

TABLE XVIII Complete details of best ℓ_∞ approximations computed when using $F_3(A,x) = (a_1 + a_2x + a_3x^2)/(1 + cx)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_∞ Parameters a_1^*	a_2^*	a_3^*	c^*	max E_i
f_1	$\frac{1 - 3x + 2x^2}{1 + 5x}$	1.00000	- 2.99999	1.99999	4.99988	0.00000
f_2	$\frac{10 - 30x + 20x^2}{1 + 5x} + \varepsilon_L^1$	9.06654	-29.14482	19.09451	2.22216	1.12405
f_3	$\frac{10 - 30x + 20x^2}{1 + 5x} + \varepsilon_N^1$	10.63251	-35.88429	30.21014	7.64277	1.01305
f_4	$\frac{10 - 30x + 20x^2}{1 + 5x} + \varepsilon_U^1$	10.55850	-31.81980	21.10072	6.20368	0.47366
f_5	$\frac{6 - 11x + 4x^2}{1 - 3x} + \varepsilon_L^2$	5.27874	- 7.52010	-0.68200	-2.99954	1.39442
f_6	$\frac{6 - 11x + 4x^2}{1 - 3x} + \varepsilon_N^2$	6.15524	-12.04729	5.29618	-2.99910	0.60983
f_7	$\frac{6 - 11x + 4x^2}{1 - 3x} + \varepsilon_U^2$	5.99743	-11.04691	4.05351	-2.99841	0.47635
f_8	$1 + \tan x$	1.00217	0.25338	-0.47138	-0.69363	0.00217
f_9	$\sqrt{x}$	0.01029	6.19568	6.44229	11.51940	0.01029
f_{10}	e^{-x^2}	0.99371	3.18116	-2.76881	2.88856	0.00629
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.94854	1.49452	-1.29744	-0.28276	0.05146

TABLE XIX : Complete details of best ℓ_1 and ℓ_2 approximations computed when using $F_4(A,x) = a_1 + a_2 \log(1 + cx)$

Define $E_i = |f(x_i) - F(A^*, x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
f_1:	$1 + 1 \log(1 + 1x)$	1.00000	1.00009	0.99987	0.00000	1.00000	1.00008	0.99989	0.00000
f_2:	$-6 + 9 \log(1 + 3x) + \varepsilon_L^1$	-6.21373	7.18242	4.39138	0.47581	-5.87351	10.15453	2.41177	0.64572
f_3:	$-6 + 9 \log(1 + 3x) + \varepsilon_N^1$	-6.27953	8.67518	3.28119	0.38651	-6.04250	8.78975	3.11100	0.54213
f_4:	$-6 + 9 \log(1 + 3x) + \varepsilon_U^1$	-6.00789	8.34328	3.38386	0.18964	-5.97396	8.93065	3.00275	0.24014
f_5:	$7 - 6 \log(1 + 7x) + \varepsilon_L^2$	7.03376	-5.94202	6.97170	0.46863	6.98132	5.86093	6.84718	0.65222
f_6:	$7 - 6 \log(1 + 7x) + \varepsilon_N^2$	7.15480	-5.50891	8.68956	0.39709	7.07924	-5.10960	10.13193	0.51282
f_7:	$7 - 6 \log(1 + 7x) + \varepsilon_U^2$	7.12159	-5.99783	7.12868	0.20606	7.15170	-5.84948	7.54583	0.25677
f_8:	$1 + \tan x$	1.02269	-1.11264	-0.74692	0.00517	1.01559	-1.14124	-0.73903	0.00644
f_9:	$\sqrt{x}$	0.12881	0.55318	3.73867	0.01081	0.03964	0.43521	7.61295	0.01857
f_{10}:	e^{-x^2}	1.05334	0.84581	-0.56995	0.01411	1.04170	0.74444	-0.60854	0.01742
f_{11}:	$\min(e^x, e^{\frac{1}{2}})$	0.93895	0.37816	6.91655	0.05049	0.94234	0.36129	7.96697	0.05995

TABLL XX Complete details of best ℓ_∞ approximations computed when using $\Gamma_4(A,x) = a_1 + a_2 \log(1 + cx)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	
f_1	$1 + 1 \log(1 + 1x)$	1.00000	0.99999	1.00002	0.00001
f_2	$-6 + 9 \log(1 + 3x) + \varepsilon_L^1$	-5.09076	21.32631	0.75876	1.12298
f_3	$-6 + 9 \log(1 + 3x) + \varepsilon_N^1$	-6.34253	7.07500	4.78514	1.01388
f_4	$-6 + 9 \log(1 + 3x) + \varepsilon_U^1$	-6.11267	9.57383	2.73287	0.41557
f_5	$7 - 6 \log(1 + 7x) + \varepsilon_L^2$	8.44803	-3.42220	39.79232	1.41429
f_6	$7 - 6 \log(1 + 7x) + \varepsilon_N^2$	6.34134	-6.56869	5.14741	1.00538
f_7	$7 - 6 \log(1 + 7x) + \varepsilon_U^2$	6.77811	-5.80235	7.05024	0.43808
f_8	$1 + \tan x$	1.00967	-1.17142	-0.73099	0.00967
f_9	$\sqrt{x}$	0.03108	0.38518	10.41328	0.03108
f_{10}	e^{-x^2}	1.02610	0.61238	-0.67290	0.02610
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.90222	0.31084	14.12181	0.09778

TABLE XXI . Complete details of best ℓ_1 and ℓ_2 approximations computed when using $F_5(A,x) = a_1 + a_2 \sin(cx)$

Define $E_i = |f(x_i) - F(A^*, x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
f_1:	$1 + 2 \sin(2\pi x)$	1.00000	2.00000	6.28318	0.00000	1.00000	2.00000	6.28318	0.00000
f_2:	$1 + 6 \sin(2\pi x) + \varepsilon_L^1$	0.97180	6.26008	6.27724	0.42141	1.08776	6.26531	6.24834	0.53232
f_3:	$1 + 6 \sin(2\pi x) + \varepsilon_N^1$	0.89405	5.88802	6.31527	0.34837	1.00945	5.95981	6.31588	0.44269
f_4:	$1 + 6 \sin(2\pi x) + \varepsilon_U^1$	1.08891	6.05590	6.22886	0.26050	0.99136	6.03375	6.26741	0.28693
f_5:	$-6 + 15 \sin(1x) + \varepsilon_L^2$	-5.96119	14.11936	1.07941	0.49620	-5.80971	16.97851	0.82719	0.77780
f_6:	$-6 + 15 \sin(1x) + \varepsilon_N^2$	-6.86149	14.23470	1.23455	0.35220	-6.39140	15.50097	1.01294	0.43916
f_7:	$-6 + 15 \sin(1x) + \varepsilon_U^2$	-6.06445	15.30859	0.98083	0.19810	-5.98186	15.92135	0.92573	0.25226
f_8:	$1 + \tan x$	†	†	†	†	†	†	†	†
f_9:	$\sqrt{x}$	0.23923	0.77721	1.27794	0.02288	0.15217	0.80755	1.54776	0.04323
f_{10}:	e^{-x^2}	†	†	†	†	†	†	†	†
f_{11}:	$\min(e^x, e^{\frac{1}{2}})$	0.96569	0.71300	1.96901	0.02336	0.96924	0.71252	2.02360	0.02932

†No best approximation found.

TABLE XXII : Complete details of best ℓ_∞ approximations computed when using $F_5(A,x) = a_1 + a_2 \sin(cx)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	max E_i
f_1	$1 + 2 \sin(2\pi x)$	1.00000	2.00000	6.28318	0.00001
f_2	$1 + 6 \sin(2\pi x) + \varepsilon_L^1$	1.34281	6.41849	6.28524	0.99882
f_3	$1 + 6 \sin(2\pi x) + \varepsilon_N^1$	1.15391	5.92400	6.32254	0.77579
f_4	$1 + 6 \sin(2\pi x) + \varepsilon_U^1$	0.95236	5.99713	6.27407	0.40501
f_5	$-6 + 15 \sin(1x) + \varepsilon_L^2$	†	†	†	†
f_6	$-6 + 15 \sin(1x) + \varepsilon_N^2$	-6.23966	13.14243	1.20073	0.84491
f_7	$-6 + 15 \sin(1x) + \varepsilon_U^2$	-6.05303	14.43035	1.06068	0.41996
f_8	$1 + \tan x$	†	†	†	†
f_9	$\sqrt{x}$	0.07918	0.86966	1.82537	0.07918
f_{10}	e^{-x^2}	†	†	†	†
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.96820	0.72938	2.09440	0.04886

†No best approximation found.

TABLE XXIII . Complete details of best ℓ_1 and ℓ_2 approximations computed when using $F_6(A,x) = a_1 + a_2/(1+x)^c$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
f_1	$2 + \frac{3}{(1+x)^2}$	2.00000	3.00000	2.00000	0.00000	2.00001	3.00000	2.00001	0.00000
f_2	$-1 + \frac{8}{(1+x)^{2.5}} + \varepsilon_L^1$	0.22476	7.14873	3.74118	0.30707	0.44028	6.99313	3.85486	0.41417
f_3	$-1 + \frac{8}{(1+x)^{2.5}} + \varepsilon_N^1$	-0.88248	7.55035	2.40811	0.37182	-1.95235	8.44083	1.88929	0.48569
f_4	$-1 + \frac{8}{(1+x)^{2.5}} + \varepsilon_U^1$	-1.94013	8.45673	2.00440	0.26695	-0.98006	7.82566	2.51987	0.31959
f_5	$-3 + \frac{12}{(1+x)^1} + \varepsilon_L^2$	0.21126	9.06536	1.59666	0.52328	-0.15354	10.06533	1.71068	0.70801
f_6	$-3 + \frac{12}{(1+x)^1} + \varepsilon_N^2$	-5.82891	14.93882	0.79610	0.30930	-2.21411	11.56192	1.18384	0.38240
f_7	$-3 + \frac{12}{(1+x)^1} + \varepsilon_U^2$	-8.67999	17.65953	0.64524	0.24208	-4.64776	13.51754	0.84633	0.29393
f_8	$1 + \tan x$	0.82125	0.21892	-2.93357	0.01589	0.82626	0.20993	-3.00631	0.01986
f_9	$\sqrt{x}$	2.02535	-1.85188	0.84170	0.01319	1.35091	-1.27212	1.75117	0.02613
f_{10}	e^{-x^2}	1.20589	-0.16314	-2.39942	0.01118	1.16447	-0.13508	-2.60169	0.01352
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	1.86600	-0.91827	2.45210	0.04322	1.84688	-0.91866	2.71177	0.05107

TABLE XXIV Complete details of best ℓ_∞ approximations computed when using $F_6(A,x) = a_1 + a_2/(1 + x)^c$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	max E_i
f_1	$2 + \frac{3}{(1+x)^2}$	2.00000	3.00001	2.00000	0.00001
f_2	$-1 + \frac{8}{(1+x)^{2.5}} + \varepsilon_L^1$	-0.03179	6.54117	2.51279	0.86412
f_3	$-1 + \frac{8}{(1+x)^{2.5}} + \varepsilon_N^1$	-26.76954	32.20057	0.23730	0.96477
f_4	$-1 + \frac{8}{(1+x)^{2.5}} + \varepsilon_U^1$	-1.02136	8.01059	2.50163	0.47264
f_5	$-3 + \frac{12}{(1+x)^1} + \varepsilon_L^2$	3.04524	6.84853	3.36044	1.39927
f_6	$-3 + \frac{12}{(1+x)^1} + \varepsilon_N^2$	-31.18828	40.41570	0.27300	0.69522
f_7	$-3 + \frac{12}{(1+x)^1} + \varepsilon_U^2$	-6.53542	15.28220	0.69296	0.46680
f_8	$1 + \tan x$	0.82521	0.20376	-3.06332	0.02897
f_9	$\sqrt{x}$	1.13300	-1.08562	2.58937	0.04738
f_{10}	e^{-x^2}	1.13933	-0.11956	-2.72631	0.01977
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	1.85643	-0.93836	2.89916	0.08193

TABLE XXV Complete details of best ℓ_1 and ℓ_2 approximations computed when using $\Gamma_7(A,x) = a_1\sin(cx) + a_2\cos(cx)$

Define $E_i = |f(x_i) - F(A,x_i)|$

DATA $x = 0(0.05)1$		Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
$f_1 =$	$1\sin(2\pi x) + 2\cos(2\pi x)$	1.00000	2.00000	6.28318	0.00000	1.00000	2.00000	6.28318	0.00000
$f_2 =$	$7\sin(2\pi x) - 3\cos(2\pi x) + \varepsilon_L^1$	6.92905	-3.08267	6.34516	0.31588	7.06656	-2.98982	6.26336	0.45643
$f_3 =$	$7\sin(2\pi x) - 3\cos(2\pi x) + \varepsilon_N^1$	6.84870	-3.49830	6.39640	0.32223	6.85727	-3.48261	6.36906	0.41084
$f_4 =$	$7\sin(2\pi x) - 3\cos(2\pi x) + \varepsilon_U^1$	6.88081	-2.93787	6.23621	0.19515	6.84758	-2.96456	6.26693	0.24719
$f_5 =$	$5\sin(1x) + 8\cos(1x) + \varepsilon_L^2$	5.27614	7.92069	1.00368	0.43503	5.59702	7.79570	1.01047	0.61344
$f_6 =$	$5\sin(1x) + 8\cos(1x) + \varepsilon_N^2$	4.30622	8.13356	0.80054	0.38582	4.92428	7.96011	1.00984	0.47551
$f_7 =$	$5\sin(1x) + 8\cos(1x) + \varepsilon_U^2$	5.52168	7.69382	1.17550	0.18267	5.38813	7.73261	1.14099	0.23294
$f_8 =$	$1 + \tan x$	†	†	†	†	†	†	†	†
$f_9 =$	$\sqrt{x}$	0.99179	0.23057	1.07845	0.02157	0.95013	0.14908	1.37205	0.04179
$f_{10} =$	e^{-x^2}	-0.21826	1.01784	1.02860	0.00823	-0.22949	1.01757	1.00983	0.00987
$f_{11} =$	$\min(e^x, e^{1/2})$	1.36969	0.95137	1.17331	0.02481	1.37853	0.95315	1.22135	0.03113

†no best approximation found

TABLE XXVI Complete details of best ℓ_∞ approximations computed when using $F_7(A,x) = a_1\sin(cx) + a_2\cos(cx)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA x = 0(0.05)1	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	max E_i
f_1	$1\sin(2\pi x) + 2\cos(2\pi x)$	1.00000	2.00000	6.28318	0.00000
f_2	$7\sin(2\pi x) - 3\cos(2\pi x) + \varepsilon_L^1$	7.57895	-2.68152	6.18527	0.86850
f_3	$7\sin(2\pi x) - 3\cos(2\pi x) + \varepsilon_N^1$	6.93985	-3.11774	6.23544	0.75095
f_4	$7\sin(2\pi x) - 3\cos(2\pi x) + \varepsilon_U^1$	6.84816	-3.05304	6.30946	0.39685
f_5	$5\sin(1x) + 8\cos(1x) + \varepsilon_L^2$	4.60151	8.58338	0.83373	1.23508
f_6	$5\sin(1x) + 8\cos(1x) + \varepsilon_N^2$	5.46859	7.69465	1.05048	0.87626
f_7	$5\sin(1x) + 8\cos(1x) + \varepsilon_U^2$	5.17238	7.80363	1.08478	0.34943
f_8	$1 + \tan x$	†	†	†	†
f_9	$\sqrt{x}$	0.94473	0.07775	1.72196	0.07775
f_{10}	e^{-x^2}	-0.23503	1.01435	0.99668	0.01435
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	1.41368	0.94735	1.25224	0.05265

† no best approximation found.

TABLE XXVII : Complete details of best ℓ_1 and ℓ_2 approximations computed when using $F_8(A,x) = e^{cx}(a_1 + a_2 x)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
f_1	$e^{-1x}(1 + 2x)$	1.00000	1.99999	-1.00000	0.00000	1.00000	2.00000	-1.00000	0.00000
f_2	$e^{-1x}(10 - 16x) + \varepsilon_L^1$	10.34464	-16.01108	-0.84626	0.58601	10.02693	-15.06866	-0.72750	0.94381
f_3	$e^{-1x}(10 - 16x) + \varepsilon_N^1$	9.62037	-15.05674	-0.85582	0.30470	9.68162	-15.13405	-0.97609	0.42180
f_4	$e^{-1x}(10 - 16x) + \varepsilon_U^1$	10.25971	-17.34019	-1.16893	0.16021	10.13669	-16.86396	-1.11305	0.20242
f_5	$e^{-2x}(-7 + 14x) + \varepsilon_L^2$	-7.52191	16.44100	-2.17024	0.32203	-7.49967	15.35976	-2.18633	0.43246
f_6	$e^{-2x}(-7 + 14x) + \varepsilon_N^2$	-6.83444	14.91311	-1.90435	0.36676	-6.95428	15.07116	-1.89072	0.50753
f_7	$e^{-2x}(-7 + 14x) + \varepsilon_U^2$	-7.46142	16.07673	-2.30037	0.22290	-7.24546	15.34211	-2.22660	0.26024
f_8	$1 + \tan x$	1.00000	0.00000	0.88908	0.01948	0.98330	0.00000	0.92460	0.02777
f_9	$\sqrt{x}$	0.18588	1.47329	-0.52311	0.01656	0.10660	1.85024	-0.70517	0.03241
f_{10}	e^{-x^2}	0.98017	2.67678	-2.20937	0.01090	0.97838	2.76056	-2.25065	0.01293
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.90255	3.03763	-0.87570	0.02913	0.93147	2.86376	-0.82694	0.03845

TABLE XXVIII : Complete details of best ℓ_∞ approximations computed when using $F_8(A,x) = e^{cx}(a_1 + a_2 x)$.

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	max E_i
f_1	$e^{-1x}(1 + 2x)$	1.00000	2.00001	-1.00000	0.00001
f_2	$e^{-1x}(10 - 16x) + \varepsilon_L^1$	8.09856	- 9.65391	-0.02390	2.24608
f_3	$e^{-1x}(10 - 16x) + \varepsilon_N^1$	10.51724	-17.29839	-1.30183	0.89693
f_4	$e^{-1x}(10 - 16x) + \varepsilon_U^1$	10.21832	-17.17809	-1.15634	0.36595
f_5	$e^{-2x}(-7 + 14x) + \varepsilon_L^2$	-6.72187	11.71351	-1.65566	0.80004
f_6	$e^{-2x}(-7 + 14x) + \varepsilon_N^2$	-6.05011	12.12923	-1.84981	0.94593
f_7	$e^{-2x}(-7 + 14x) + \varepsilon_U^2$	-7.08534	14.61675	-2.14057	0.37609
f_8	$1 + \tan x$	0.95935	0.00000	0.96315	0.04398
f_9	$\sqrt{x}$	0.05804	2.30654	-0.92039	0.05804
f_{10}	e^{-x^2}	0.98100	2.82223	-2.28550	0.01900
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.93544	2.82656	-0.78764	0.06456

TABLE XXIX Complete details of best ℓ_1 approximations computed when using $F_9(A,x) = e^{cx}(a_1 + a_2x + a_3x^2 + a_4x^3$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_1 Parameters a_1^*	a_2^*	a_3^*	a_4^*	c^*	$\frac{\Sigma E_i}{21}$
f_1	$e^{-1x}(1 + 1x + 1x^2 + 1x^3)$	1.00000	0.99957	1.00009	0.99898	-0.99966	0.00000
f_2	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_L^1$	9.80684	1.54798	3.46611	28.66960	-1.24424	0.04498
f_3	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_N^1$	10.08272	-9.05095	20.56280	0.00001	-0.54668	0.03021
f_4	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_U^1$	9.99638	-5.95021	16.42085	6.10495	-0.75385	0.02959
f_5	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_L^2$	0.04929	13.81137	-67.07568	79.96113	-3.03196	0.04581
f_6	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_N^2$	-0.13948	20.82122	-104.9411	128.9301	-3.59474	0.03901
f_7	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_U^2$	-0.01102	14.12980	-70.19599	83.76241	-3.02950	0.02179
f_8	$1 + \tan x$	1.00125	-0.35896	- 0.23035	0.26739	1.32433	0.00054
f_9	$\sqrt{x}$	0.11764	2.52609	- 0.73960	3.14312	-1.62065	0.00665
f_{10}	e^{-x^2}	0.99890	-0.03720	- 1.20358	0.58644	0.06815	0.00028
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.96338	-2.21940	1.80610	- 0.51160	3.75768	0.01656

TABLE XXX Complete details of best ℓ_2 approximations computed when using $F_9(A,x) = e^{cx}(a_1 + a_2x + a_3x^2 + a_4x^3)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA x = 0(0.05)1	Best ℓ_2 Parameters a_1^*	a_2^*	a_3^*	a_4^*	c^*	$\sqrt{\frac{\Sigma F_i^2}{21}}$
f_1	$e^{-1x}(1 + 1x + 1x^2 + 1x^3)$	1.00000	0.99992	1.00003	0.99983	-0.99995	0.00000
f_2	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_L^1$	9.84914	1.93815	3.26632	31.00314	-1.30519	0.05685
f_3	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_N^1$	10.06324	-3.46332	13.68106	14.40042	-1.02227	0.03995
f_4	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_U^1$	10.00352	-13.75118	22.56505	-7.03129	0.05859	0.03458
f_5	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_L^2$	0.03615	14.29913	-69.41519	82.36906	-3.03012	0.07188
f_6	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_N^2$	-0.16754	18.98748	-93.59789	113.3828	-3.43627	0.04652
f_7	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_U^2$	-0.01578	14.81504	-73.57406	87.68207	-3.07986	0.02747
f_8	$1 + \tan x$	1.00112	-0.42149	- 0.19468	0.25429	1.38608	0.00066
f_9	$\sqrt{x}$	0.01864	4.13369	- 4.86066	11.58366	-2.39815	0.01072
f_{10}	e^{-x^2}	1.00041	1.93946	1.21718	- 1.52599	-1.96908	0.00033
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.97010	-1.56823	0.68180	0.00000	2.98809	0.02197

TABLE XXXI Complete details of best ℓ_∞ approximations computed when using $\Gamma_9(\Lambda,x) = e^{cx}(a_1 + a_2x + a_3x^2 + a_4x^3)$

Define $L_i = |f(x_i) - \Gamma(\Lambda^*, x_i)|$

	DATA x = 0(0.05)1	Best ℓ_∞ Parameters a_1^*	a_2^*	a_3^*	a_4^*	c^*	max E_i
f_1	$e^{-1x}(1 + 1x + 1x^2 + 1x^3)$	1.00000	1.00011	0.99995	1.00023	-1.00007	0.00001
f_2	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_L^1$	9.90006	1.22766	5.63696	30.27473	-1.33015	0.09324
f_3	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_N^1$	10.01830	-2.94181	12.53366	15.16748	-1.02222	0.06444
f_4	$e^{-1x}(10 - 3x + 12x^2 + 15x^3) + \varepsilon_U^1$	10.00674	-13.55190	22.43607	-6.75947	0.03052	0.04647
f_5	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_L^2$	-0.08405	17.61009	-86.52715	103.2233	-3.23603	0.13334
f_6	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_N^2$	-0.21433	20.29668	-101.4198	123.2556	-3.52223	0.07485
f_7	$e^{-\pi x}(0 + 15x - 75x^2 + 90x^3) + \varepsilon_U^2$	0.01564	14.86384	-75.03174	90.29340	-3.13867	0.04204
f_8	$1 + \tan x$	1.00088	-0.43320	-0.18932	0.25259	1.39917	0.00088
f_9	$\sqrt{x}$	0.01819	4.67796	-7.52759	17.95251	-2.73445	0.01819
f_{10}	e^{-x^2}	1.00043	1.95169	1.25424	-1.53877	-1.98235	0.00043
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	0.96212	-1.59197	0.70662	-0.00291	3.09689	0.03788

TABLE XXXII Complete details of best ℓ_1 and ℓ_2 approximations computed when using $\Gamma_{10}(A,x) = a_1\sinh(cx) + a_2\cosh(c$

Define $E_i = |f(x_i) - \Gamma(A^*,x_i)|$

	DATA x = 0(0.05)1	Best ℓ_1 Parameters a_1^*	a_2^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
f_1	$1\sinh(1x) + 2\cosh(1x)$	0.99999	2.00000	1.00000	0.00000	1.00000	2.00000	1.00000	0.00000
f_2	$3\sinh(1x) + 5\cosh(1x) + \varepsilon_L^1$	2.39499	5.11692	1.05308	0.08040	2.96473	5.01051	1.00083	0.11833
f_3	$3\sinh(1x) + 5\cosh(1x) + \varepsilon_N^1$	2.88046	5.04025	1.00540	0.03666	2.95524	5.03533	0.99835	0.04773
f_4	$3\sinh(1x) + 5\cosh(1x) + \varepsilon_U^1$	3.11447	5.00772	0.98275	0.02208	3.12398	4.99231	0.98423	0.02788
f_5	$8\sinh(1x) - 3\cosh(1x) + \varepsilon_L^2$	9.05332	-2.97754	0.86911	0.02292	9.15344	-2.96112	0.85593	0.03321
f_6	$8\sinh(1x) - 3\cosh(1x) + \varepsilon_N^2$	8.57319	-2.99727	0.92728	0.02997	7.57150	-2.98836	1.05898	0.03793
f_7	$8\sinh(1x) - 3\cosh(1x) + \varepsilon_U^2$	8.42580	-3.03230	0.95311	0.02320	7.95043	-3.01475	1.01157	0.02824
f_8	$1 + \tan x$	0.67283	1.03208	1.04850	0.01432	0.62230	1.02919	1.08575	0.01804
f_9	$\sqrt{x}$	†	†	†	†	†	†	†	†
f_{10}	e^{-x^2}	†	†	†	†	†	†	†	†
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	†	†	†	†	†	†	†	†

† no best approximation found.

TABLE XXXIII Complete details of best ℓ_∞ approximations computed when using $F_{10}(A,x) = a_1\sinh(cx) + a_2\cosh(cx)$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA x = 0(0.05)1	Best ℓ_∞ Parameters a_1^*	a_2^*	c^*	max E_i
f_1	1 sinh(1x) + 2 cosh(1x)	1.00000	2.00000	1.00000	0.00001
f_2	3 sinh(1x) + 5 cosh(1x) + ε_L^1	4.57872	4.95361	0.82798	0.23751
f_3	3 sinh(1x) + 5 cosh(1x) + ε_N^1	3.23411	5.04186	0.96220	0.08568
f_4	3 sinh(1x) + 5 cosh(1x) + ε_U^1	3.10033	4.99672	0.98352	0.04382
f_5	8 sinh(1x) - 3 cosh(1x) + ε_L^2	8.19830	-2.91309	0.95711	0.07116
f_6	8 sinh(1x) - 3 cosh(1x) + ε_N^2	6.18127	-2.96285	1.32488	0.07991
f_7	8 sinh(1x) - 3 cosh(1x) + ε_U^2	6.85481	-3.01352	1.19255	0.04619
f_8	1 + tan x	0.57794	1.02656	1.11953	0.02656
f_9	$\sqrt{x}$	†	†	†	†
f_{10}	e^{-x^2}	†	†	†	†
f_{11}	$\min(e^x, e^{\frac{1}{2}})$	†	†	†	†

†no best approximation found.

TABLE XXXIV : Complete details of best ℓ_1 and ℓ_2 approximations computed when using $\Gamma_{11}(A,x) = a_1 + a_2x + a_3(x-c)_+$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA x = 0(0.05)1	Best ℓ_1 Parameters a_1^*	a_2^*	a_3^*	c^*	$\frac{\Sigma E_i}{21}$	Best ℓ_2 Parameters a_1^*	a_2^*	a_3^*	c^*	$\sqrt{\frac{\Sigma E_i^2}{21}}$
f_1:	$2 + 20x - 30(x-1/3)_+$	2.00000	20.00008	-30.00008	0.33333	0.00002	1.99999	20.00006	-30.00006	0.33333	0.0
f_2:	$2 + 20x - 30(x-1/3)_+ + \varepsilon_L^1$	0.88818	27.41747	-36.29969	0.28376	0.55228	1.41296	22.35409	-32.33154	0.32661	0.6
f_3:	$2 + 20x - 30(x-1/3)_+ + \varepsilon_N^1$	2.00458	19.55074	-31.47486	0.35706	0.30100	2.10960	19.83245	-31.79070	0.35031	0.5
f_4:	$2 + 20x - 30(x-1/3)_+ + \varepsilon_U^1$	1.53898	22.89670	-33.26642	0.32349	0.22639	1.61529	22.11969	-32.53886	0.32806	0.2
f_5:	$-10 + 20x - 40(x-1/2)_+ + \varepsilon_L^2$	-10.21007	20.75871	-39.88351	0.48698	0.22854	-10.26240	20.79511	-40.73535	0.49483	0.3
f_6:	$-10 + 20x - 40(x-1/2)_+ + \varepsilon_N^2$	-10.08975	20.32950	-41.42819	0.51506	0.43611	- 9.91818	19.75669	-40.00028	0.50687	0.5
f_7:	$-10 + 20x - 40(x-1/2)_+ + \varepsilon_U^2$	- 9.90320	19.47124	-38.68702	0.50263	0.21446	- 9.89788	19.41206	-38.44969	0.50000	0.2
f_8:	$1 + \tan x$	0.98177	1.12906	1.20485	0.66448	0.01458	0.97884	1.15159	1.21230	0.67866	0.0
f_9:	$\sqrt{x}$	0.14908	1.49067	- 0.86284	0.27725	0.01613	0.02183	3.16231	- 2.46474	0.12704	0.0
f_{10}:	e^{-x^2}	1.01202	-0.29045	- 0.53183	0.32848	0.00439	1.01174	-0.28832	- 0.52554	0.32139	0.0
f_{11}:	$\min(e^x, e^{\frac{1}{2}})$	0.97628	1.28886	- 1.28886	0.52173	0.00622	0.97674	1.29358	- 1.29358	0.51947	0.0

TABLE XXXV Complete details of best ℓ_∞ approximations computed when using $\Gamma_{11}(A,x) = a_1 + a_2x + a_3(x-c)_+$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0.(0.05)1$	Best ℓ_∞ Parameters a_1^*	a_2^*	a_3^*	c^*	max E_i
f_1:	$2 + 20x - 30(x-1/3)_+$	2.00001	19.99999	-30.00000	0.33333	0.00001
f_2:	$2 + 20x - 30(x-1/3)_+ + \varepsilon_L^1$	1.47998	20.77490	-30.73985	0.33769	1.08184
f_3:	$2 + 20x - 30(x-1/3)_+ + \varepsilon_N^1$	0.73500	28.75966	-40.52526	0.28903	1.22234
f_4:	$2 + 20x - 30(x-1/3)_+ + \varepsilon_U^1$	1.62810	22.27021	-32.72843	0.32507	0.38736
f_5:	$-10 + 20x - 40(x-1/2)_+ + \varepsilon_L^2$	-10.40986	22.00190	-42.21426	0.48364	0.62126
f_6:	$-10 + 20x - 40(x-1/2)_+ + \varepsilon_N^2$	- 9.61526	19.49087	-36.77151	0.48202	1.02212
f_7:	$-10 + 20x - 40(x-1/2)_+ + \varepsilon_U^2$	- 9.95897	19.68760	-39.04818	0.49616	0.43253
f_8:	$1 + \tan x$	0.97074	1.20327	1.24993	0.71140	0.02926
f_9:	$\sqrt{x}$	0.03376	3.15432	- 2.43350	0.11472	0.03376
f_{10}:	e^{-x^2}	1.01039	-0.28690	- 0.51051	0.31837	0.01039
f_{11}:	$\min(e^x, e^{\frac{1}{2}})$	0.97984	1.29744	- 1.23722	0.50826	0.02017

TABLE XXXVI : Complete details of best ℓ_1 approximations computed when using $F_{12}(A,x) = a_1 + a_2x + a_3x^2 + a_4x^3 + a_5(x-$

Define $E_i = |f(x_i) - F(A^*,x_i)|$

	DATA x = 0(0.05)1	Best ℓ_1 Parameters a_1^*	a_2^*	a_3^*	a_4^*	a_5^*	c^*	$\frac{\Sigma E_i}{21}$
f_1:	$1 + 4x - 3x^2 + 1x^3 + 4(x-1/2)_+^3$	1.00000	3.99993	- 2.99973	0.99973	4.00054	0.50000	0.00000
f_2:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_L^1$	-2.98345	13.31161	10.25936	- 7.22578	25.07838	0.25838	0.01757
f_3:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_N^1$	-2.95276	12.04028	18.61574	-24.98495	41.45663	0.20479	0.03101
f_4:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_U^1$	-3.02559	14.04009	4.26233	4.24018	12.32501	0.31015	0.02160
f_5:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_L^2$	7.02473	-3.85641	8.01153	-13.71827	- 47.33755	0.74443	0.03043
f_6:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_N^2$	7.01521	-3.69288	7.35076	-13.04848	- 43.09281	0.71700	0.03278
f_7:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_U^2$	7.01371	-2.76950	2.81433	- 7.69356	- 18.06975	0.47662	0.01555
f_8:	$1 + \tan x$	0.99894	1.02837	- 0.15671	0.58244	2.22390	0.63930	0.00057
f_9:	$\sqrt{x}$	0.00000	6.24735	-41.10605	112.0643	-111.6747	0.11962	0.00167
f_{10}:	e^{-x^2}	1.00000	0.00084	- 1.02789	0.22277	0.37742	0.22906	0.00012
f_{11}:	$\min(e^x, e^{\frac{1}{2}})$	1.01602	0.55959	3.09823	- 3.80849	7.64114	0.52898	0.00939

TABLE XXXVII : Complete details of best ℓ_2 approximations computed when using $F_{12}(A,x) = a_1 + a_2x + a_3x^2 + a_4x^3 + a_5(x-$

Define $L_i = |f(x_i) - F(A^*,x_i)|$

	DATA $x = 0(0.05)1$	Best ℓ_2 Parameters a_1^*	a_2^*	a_3^*	a_4^*	a_5^*	c^*	$\sqrt{\frac{\sum L_i^2}{21}}$
f_1:	$1 + 4x - 3x^2 + 1x^3 + 4(x-1/2)_+^3$	1.00000	3.99998	- 2.99995	0.99995	4.00010	0.50000	0.00000
f_2:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_L^1$	-2.99592	13.16005	12.02344	-11.49191	28.94445	0.23635	0.02300
f_3:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_N^1$	-2.96335	13.04621	8.90671	- 1.84878	19.96498	0.31409	0.04322
f_4:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_U^1$	-3.00276	13.81752	5.00622	3.63166	13.11431	0.32164	0.02692
f_5:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_L^2$	7.02065	-3.59540	6.40422	-11.68711	- 19.20145	0.57898	0.04639
f_6:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_N^2$	7.02003	-3.62251	6.65536	-12.02500	- 29.99303	0.64303	0.04635
f_7:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_U^2$	7.00435	-2.67418	2.32478	- 6.99168	- 18.53968	0.46104	0.02029
f_8:	$1 + \tan x$	0.99918	1.02954	- 0.17101	0.60082	2.31760	0.65155	0.00069
f_9:	$\sqrt{x}$	0.00010	6.29883	-42.94068	122.0964	-121.6489	0.11455	0.00245
f_{10}:	e^{-x^2}	1.00001	-0.00077	- 1.00320	0.14646	0.44677	0.20343	0.00019
f_{11}:	$\min(e^x, e^{\frac{1}{2}})$	1.01243	0.48444	3.63006	- 4.44017	8.46309	0.51398	0.01341

TABLE XXXVIII: Complete details of best ℓ_∞ approximations computed when using $F_{12}(A,x) = a_1 + a_2x + a_3x^2 + a_4x^3 + a_5(x-c)$

Define $L_i = |f(x_i) - F(A^*,x_i)|$

	DATA x = 0(0.05)1	Best ℓ_∞ Parameters a_1^*	a_2^*	a_3^*	a_4^*	a_5^*	c^*	max L
f_1:	$1 + 4x - 3x^2 + 1x^3 + 4(x-1/2)_+^3$	1.00000	4.00000	- 2.99995	0.99991	3.99988	0.49998	0.000
f_2:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_L^1$	-2.99870	13.28446	9.59814	- 4.50232	23.10920	0.29081	0.035
f_3:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_N^1$	-2.87665	12.51090	7.73725	3.19924	19.40747	0.45917	0.076
f_4:	$-3 + 14x + 4x^2 + 5x^3 + 12(x-1/3)_+^3 + \varepsilon_U^1$	-2.98331	13.61717	5.98245	1.75363	14.13625	0.28481	0.042
f_5:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_L^2$	6.93837†	-2.20909†	0.36654†	- 4.66357†	- 21.17326†	0.44179†	0.078
f_6:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_N^2$	6.93579	-2.98476	4.91604	-10.41184	- 21.05885	0.56639	0.090
f_7:	$7 - 3x + 4x^2 - 9x^3 - 18(x-1/2)_+^3 + \varepsilon_U^2$	6.97852	-1.54527	- 4.07009	1.97931	- 25.98545	0.36685	0.035
f_8:	$1 + \tan x$	0.99904	1.03612	- 0.20007	0.63056	2.41370	0.66492	0.000
f_9:	$\sqrt{x}$	0.00374	6.04953	-40.22999	113.5966	-113.0773	0.11486	0.004
f_{10}:	e^{-x^2}	1.00030†	-0.03508†	- 0.48705†	- 1.84721†	2.43263†	0.10616†	0.000
f_{11}:	$\min(e^x, e^{1/2})$	1.02230	0.16901	4.94313	- 5.72930	11.68011	0.52322	0.022

† Because of numerical instability, these cases were run with values of 0.0001 and 10000 for TOLER and QMAX respectively in MINIMAX

CHAPTER IV

REMARKS ON NONLINEAR BEST ℓ_1 APPROXIMATIONS

IV-1. Remarks.

A theory of best approximation cannot be considered complete until the questions of existence, uniqueness and characterization of best approximations are answered, and a satisfactory method is available to determine these approximations in practice. In this section we include some remarks on existence and characterization of best ℓ_1 approximations by the functions in Table I.

Although best linear ℓ_1 approximations are guaranteed to exist, for the first ten nonlinear functions of Table I we have constructed data sets for which no best approximation exists. These same examples also demonstrate the possibility of nonexistence for best ℓ_p approximations. For the last two functions in Table I (i.e. spline functions with one free knot) best discrete approximations always exist. Since uniqueness of best linear ℓ_1 approximations is not assured, even in the case of polynomial approximation, any attempt to discover hypotheses which guarantee uniqueness of best nonlinear ℓ_1 approximations would appear to be fruitless.

Rice (1964) contains the proof that the following property is characteristic of best linear ℓ_1 approximations.

Theorem 1: Let $\{\phi_1(x), \phi_2(x), \ldots, \phi_n(x)\}$ be a Chebyshev set on $[0,1]$ and suppose the discrete set $X \subset [0,1]$. Then there exists a best ℓ_1 approximation by $F(A,x) = \sum_{j=1}^{n} a_j \phi_j(x)$ on X which interpolates at least n points of $f(x)$.

The definition of a Chebyshev set is the following.

Definition 2: $\{\phi_1(x), \phi_2(x), \ldots, \phi_n(x)\}$ forms a Chebyshev set on $[a,b]$ if $\det|\phi_j(x_i)| \neq 0$, for every choice of n distinct points x_i from $[a,b]$.

This result says that in searching for a best approximation we need only consider those approximating functions which interpolate $f(x)$ at n or more points of X. This raises the question of whether there exists a best nonlinear approximation by, for example, $F(A,x) = a_1 + a_2e^{cx}$ which interpolates $f(x)$ in three points. Our empirical study shows that although interpolation at the extra point (due to the presence of the nonlinear parameter c) is most definitely the rule rather than the exception, this phenomenon does not always occur, even when the best approximation is unique. However, we can use the characteristic interpolatory property of linear approximation to help analyse the situation that prevails with any of the functions in Table I (in the case of the spline functions, which do not form Chebyshev sets, the analysis is more difficult but still carries through).

We shall consider ℓ_1 approximation by functions of the form $F(A,x) = a_1 + a_2e^{cx}$, although the arguments can be extended to other nonlinear functions. Existence of a best approximation depends upon the data defining $f(x)$; note that no best approximation exists to the set $\{(0,0), (1,1), (2,2)\}$. For $c \neq 0$ the set $\{1,e^{cx}\}$ forms a Chebyshev set on $[0,1]$, and hence there exists a best linear ℓ_1 approximation by $a_1 + a_2e^{cx}$ which interpolates at least two points of $f(x)$. If there exists a best nonlinear approximation by

$a_1 + a_2 e^{cx}$, (c free), then there exists one which interpolates at least two points. Suppose $f(x) = \{(x_1,f_1), (x_2,f_2),\ldots,(x_N,f_N)\}$. For fixed nonzero c and $i \neq j$ let $F_{ij}(x;c) = a_1^{ij}(c) + a_2^{ij}(c)\, e^{cx}$ be the function which interpolates the i^{th} and j^{th} points of $f(x)$. We are guaranteed this unique function since $\{1,e^{cx}\}$ forms a Chebyshev set. Let $S_{ij}(c) = \sum_{k=1}^{n} |f_k - F_{ij}(x_k;c)|$, and define $G(c) = \min_{i\neq j} S_{ij}(c)$. If we now allow c to vary, since $F_{ij}(x;c)$ and consequently $S_{ij}(c)$ are continuous functions of c (except for $c = 0$), then $G(c)$ is also continuous. Furthermore, $G(c)$ is precisely that function of c which we calculate in our algorithm. Thus our method is equivalent to that of locating the minimum of a continuous curve (except at $c = 0$) which is the envelope of a finite number of curves, each of which is produced by interpolating a distinct pair of data points.

Defining $H = \min\{\lim_{c\to-\infty} G(c), \lim_{c\to 0} G(c), \lim_{c\to+\infty} G(c)\}$, then $G(c)$ will have a minimum provided there exists an approximation to $f(x)$ which has an error of approximation not exceeding H. We thus have a necessary and sufficient condition for the existence of a best ℓ_1 approximation by $F(A,x) = a_1 + a_2 e^{cx}$. The quantity H can be calculated for any data set in practice by considering the limit of $F_{ij}(x;c)$ curves as $c \to -\infty, 0, +\infty$. Provided an approximation can be found which produces a sum of errors not exceeding H, then the existence of a best ℓ_1 approximation is guaranteed.

Although $G(c)$ is a continuous function, for nonzero c, this does not imply that it is necessarily unimodal in some neighbourhood of its minimum. On the other hand a moment's reflection will convince

the reader that any continuous function which is not in fact unimodal in a small interval enclosing its minimum is rather special. Such functions (e.g. $f(x) = |x \sin \frac{1}{x}| + |x|$ for nonzero $x \in [-1,1]$ and $f(0) = 0$) do not seem likely to be representable as an envelope of curves S_{ij}.

Concerning the question of interpolation at an extra point, it appears in practice that the minimum of $G(c)$ occurs when two of the $S_{ij}(c)$ curves intersect. That is, the minimum does correspond to interpolation at three points. However we do have counterexamples to show that this is not always the case. Interpolation at some n points of $f(x)$ will therefore not be a general characteristic property of nonlinear best ℓ_1 approximations.

For other approximating functions the analysis is similar. For the rational function $(a_1 + a_2x)/(1 + cx)$ the function $G(c)$ is also continuous except for poles at the zeros of the denominator. The quantity H in this case is defined to be $H = \min\{\lim_{c\to-\infty} G(c), \lim_{c\to+\infty} G(c)\}$.

Since best linear ℓ_∞ approximations by Chebyshev sets are characterized by the error equioscillation property at $n+1$ points, it is possible to carry out a similar analysis in this case. For the exponential case we define, for nonzero c, $F_{ijk}(c) = a_1^{ijk}(c) + a_2^{ijk}(c)\, e^{cx}$ to be that function which has the equioscillation property at the i^{th}, j^{th} and k^{th} points. Defining $S_{ijk}(c)$, $G(c)$ and H as before we obtain necessary and sufficient conditions for the existence of best ℓ_∞ approximations by functions of the type shown in Table I.

In the following pages of this section we have included examples (see Table XXXIX) of data sets for which no best ℓ_1 approximations exist for each of the first ten nonlinear functions of Table I. Following this, Figs. 6, 7 and 8 show cases of approximation by $F(A,x) = a_1 + a_2 e^{cx}$ where, respectively, no best approximation exists, the best approximation interpolates two points, and the best approximation interpolates three points.

TABLE XXXIX. Examples of data sets for which no best nonlinear ℓ_1 approximations exist by the first ten functions of Table I.

Approximating Function	Data Set
$a_1 + a_2 e^{cx}$	{(0,1), (1,0), (2,0)}
$(a_1 + a_2 x)/(1 + cx)$	{(0,1), (1,0), (2,0)}
$(a_1 + a_2 x + a_3 x^2)/(1 + cx)$	{(0,1), (1,0), (2,0), (3,0)}
$a_1 + a_2 \log(1 + cx)$	{(0,0), (1,1), (2,1)}
$a_1 + a_2 \sin(cx)$	{(0,0), (1,1), (2,2)}
$a_1 + a_2/(1 + x)^c$	{(0,0), (1,1), (2,1)}
$a_1 \sin(cx) + a_2 \cos(cx)$	{(0,0), (1,1), (2,2)}
$e^{cx}(a_1 + a_2 x)$	{(0,0), (0,1), (2,1)}
$e^{cx}(a_1 + a_2 x + a_3 x^2 + a_4 x^3)$	{(0,1), (1,0), (2,0), (3,0), (4,0)}
$a_1 \sinh(cx) + a_2 \cosh(cx)$	{(0,0), (1,1), (2,2)}

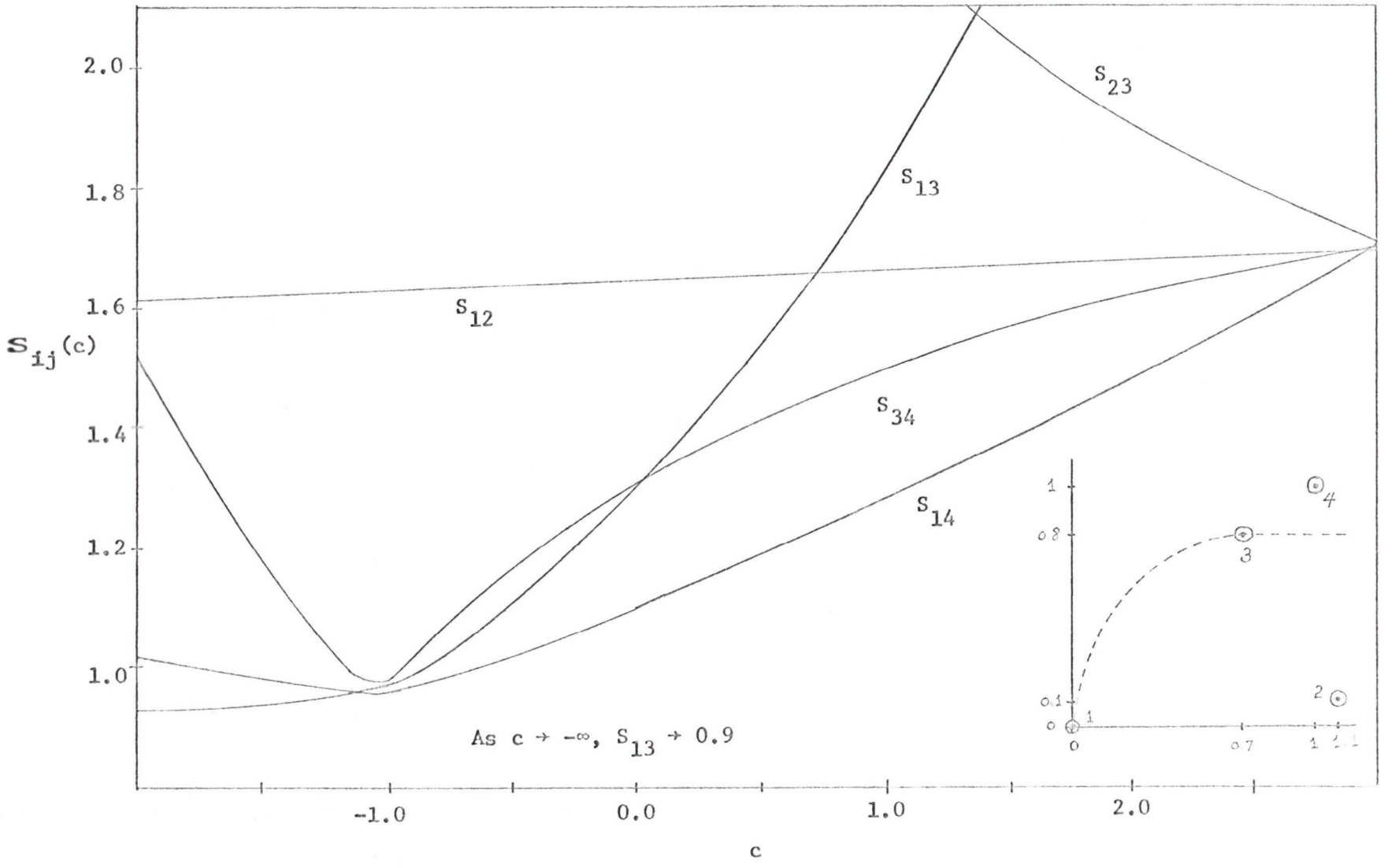


Fig. 6. Example of non-existence of a best ℓ_1 approximation by $I(A,x) = a_1 + a_2 e^{cx}$.

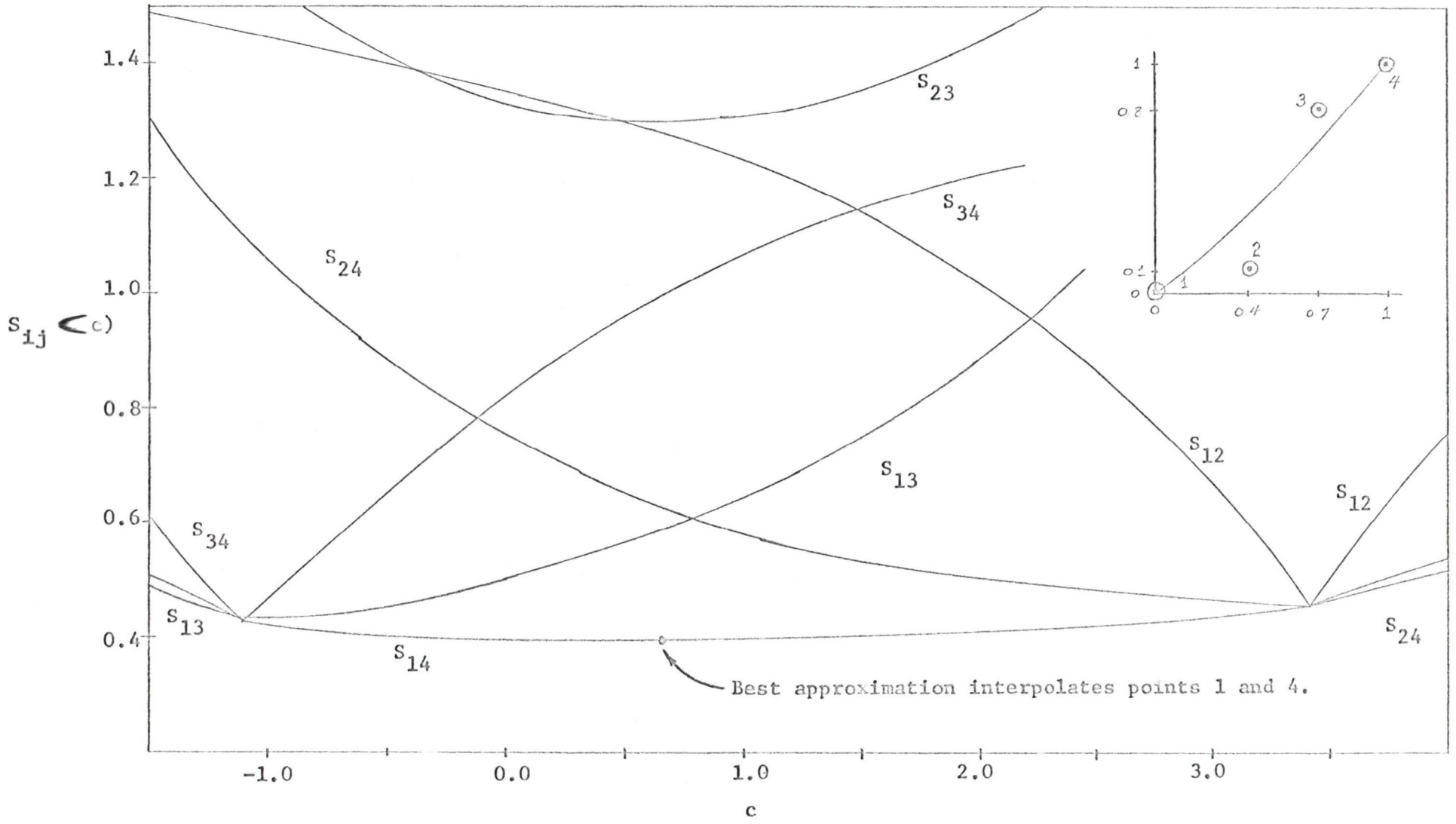


Fig. 7. Example of a best ℓ_1 approximation by $\Gamma(\Lambda,x) = a_1 + a_2 e^{cx}$ interpolating two points.

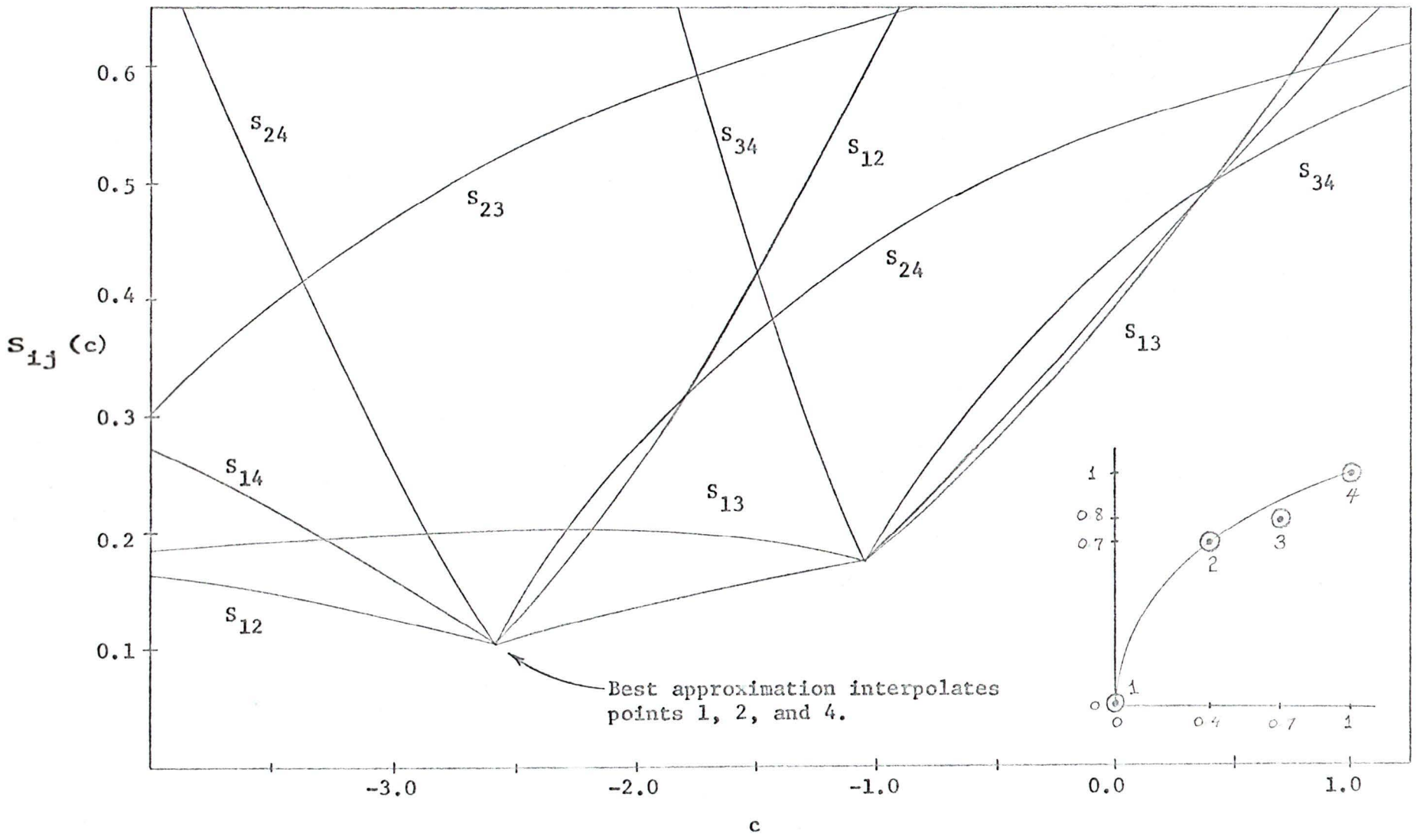


Fig. 8. Example of a best ℓ_1 approximation by $F(A,x) = a_1 + a_2 e^{cx}$ interpolating three points.

CHAPTER V

A PRACTICAL APPLICATION, AND AN EXTENSION OF THE METHOD TO TWO NONLINEAR PARAMETERS

V-1. Application.

During the course of this study a practical approximation problem which easily adapted to our technique was brought to our attention. Plots of two year old seedlings of Sitka Spruce, White Spruce and Douglas Fir were sampled over a period of one year for their contents of various minerals. We were given the results of these experiments and asked to supply a function which would approximate the mineral content of the seedlings, for any time within this one year period of experimentation. The primary purpose of this approximation was to indicate the time at which the maximum increase in mineral content occurred.

The data (see Table XL) was presented as a set of ordered pairs, (W_i, t_i), for each combination of tree and element in the experiments. W_i denoted the weight of the element in the tree at time t_i. The only change made in the data before attempting an approximation was to divide the t_i's by 365, thereby changing the interval of approximation from (0,365) to (0,1). This was done for consistency, and ease of manipulation in the calculations.

When plotted, the data had the general form of a pair of connected quadratics as shown in Fig. 9. This shape, and the apparent existence of wild points in the data, suggested that we attempt an ℓ_1 approxi-

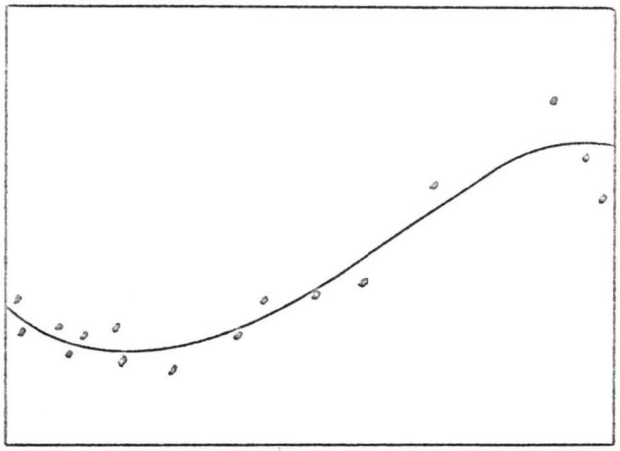


Fig. 9. General form of the data to be approximated.

mation with a quadratic spline. Thus, our approximating function had the form

$$W = a_0 + a_1 t + a_2 t^2 + a_3 (t - c)_+^2 \qquad (K)$$

The definition of a spline function is the following.

Definition 3: $S_{m,k}(x)$ is a spline function of degree m with knots $x_1 < x_2 < \ldots < x_k$ if and only if it possesses the the following two properties.

(i) $S_{m,k}(x)$ is a polynomial of degree m in each of the intervals $(-\infty, x_1)$, $[x_1, x_2), \ldots, [x_k, \infty)$.

(ii) $S_{m,k}(x) \in C^{m-1}$, i.e. it has continuous derivatives up to order $m-1$.

In addition to giving a reasonably good fit, a quadratic spline

will give us immediately the point in time at which the maximum increase in mineral content occurred. This, of course, will be the point of inflection of the spline, i.e. the value of c in (K).

Phase I was executed with a step size of 1/26 (i.e. a step size of two weeks). Thus our interval of uncertainty for Phase II was approximately 0.08. For purposes of illustration three examples are included, the cases of P, K, and Ca content for White Spruce. Table XL contains the data for these cases, Table XLI the resultant coefficients of the best approximations, and Figs. 10 to 12 the corresponding graphs of the best approximations with the data points also plotted. The tolerance used in Phase II for these cases was 0.000001 which was probably somewhat small as the W_i's were given to only 2D.

These results were not completely satisfactory as they did not resemble growth curves, which are flat (i.e. their first derivative is zero) at the endpoints. This is easily accomplished as follows. By insisting that the approximating function (K) have derivative zero at the endpoints (i.e. at $t = 0$ and $t = 1$) we want to satisfy, respectively,

$$0 = a_1 \tag{L}$$

and

$$0 = a_1 + 2a_2 + 2a_3(1 - c) \tag{M}$$

Equation (L) is satisfied by setting $a_1 = 0$ which results in (M) becoming

$$0 = 2a_2 + 2a_3(1 - c) \qquad (N)$$

Thus the approximation problem has become:

$$\text{minimize} \sum_i |f_i - (a_0 + a_2 t_i^2 + a_3 (t_i - c)_+^2)|$$

$$\text{subject to} \quad 2a_2 + 2a_3(1 - c) = 0$$

For ease of implementation via our existing computer programs, we regard this additional constraint as though it has arisen as an extra observation of the function. This is accomplished by adding to (A) the constraint

$$0 = f_{N+1} = 0 + \alpha_2 \cdot 2 + \alpha_3 \cdot 2(1 - c) + \alpha_4 \, \phi_{n+1,N+1} - u_{N+1} + v_{N+1} \, .$$

Thus the slope at $t = 1$ will be approximately equal to zero, unless we force $u_{N+1} = v_{N+1} = 0$ by choice of a suitable weight function $w(x)$.

Phase I and Phase II were executed for this constrained spline in the same manner as for the previous examples. The resultant coefficients of the best approximations are contained in Table XLI and the corresponding graphs are shown in Figs. 10 to 12 with the data points also plotted. It is clear from the graphs that if zero slope at the endpoints is required, then the constrained spline is a more satisfactory approximating function than is the unconstrained spline.

This data is from experiments conducted by Dr. R. van den Driessche of the Research Division Laboratory, British Columbia Forest Service, Victoria, British Columbia. Details of these experiments have been published in part in van den Driessche (1968).

TABLE XL: Data for the mineral content of White Spruce

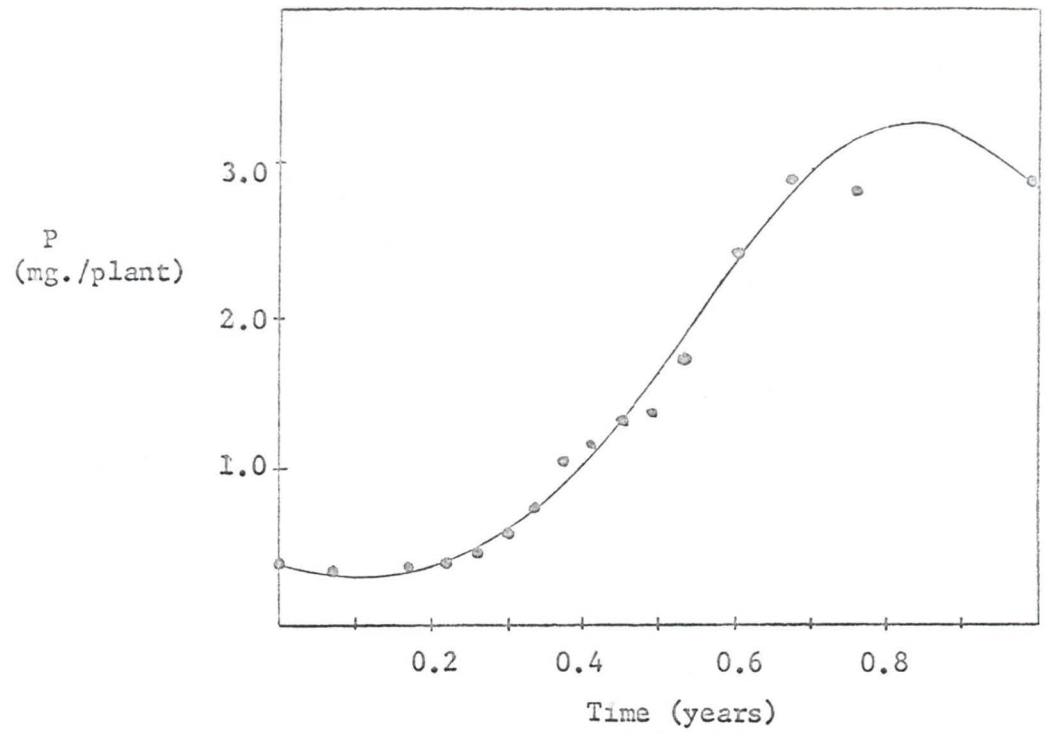
Time (years)	Mineral content (mg./plant)		
	P	K	Ca
0.0	0.38	0.89	0.53
0.0712	0.37	0.84	0.98
0.1699	0.36	0.81	1.07
0.2219	0.40	1.14	1.26
0.2603	0.46	1.43	0.76
0.2986	0.59	1.77	1.14
0.3370	0.77	2.45	1.67
0.3753	1.05	3.00	1.89
0.4137	1.15	3.81	2.11
0.4521	1.31	3.90	2.75
0.4904	1.36	3.80	2.33
0.5288	1.71	4.48	2.58
0.6055	2.42	5.58	5.51
0.6822	2.89	6.23	6.15
0.7589	2.82	5.72	5.41
0.9890	2.87	4.21	4.97

TABLE XLI· Complete details of best ℓ_1 approximations for P, K, and Ca content of White Spruce when using $F(A,t) = a_0 + a_1 t + a_2 t^2 + a_3 (t - c)_+^2$

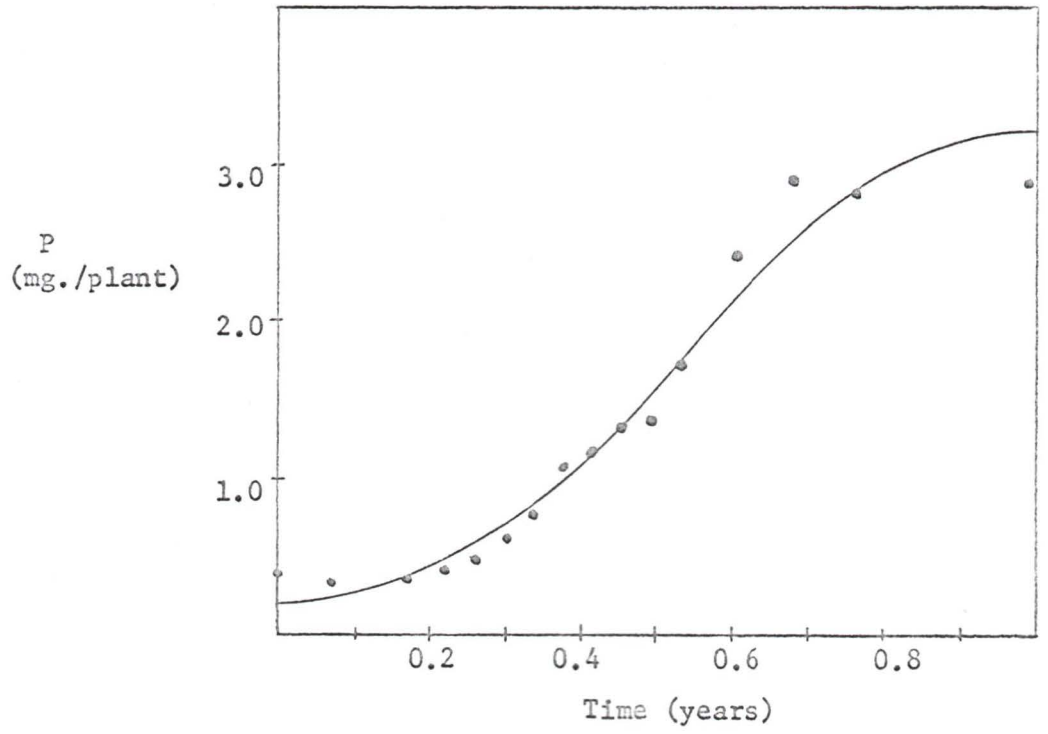
Define $E_i = |f(t_i) - F(A^*, t_i)|$

Mineral	Unconstrained or Constrained Spline	Best ℓ_1 Parameters					
		a_0^*	a_1^*	a_2^*	a_3^*	c^*	$\frac{\Sigma E_i}{16}$
P	U	0.38000	-1.80677	8.54773	-24.40072	0.57991	0.06526
P	C	0.20381	0.0†	5.41311	-12.19543	0.55614	0.11643
K	U	0.89000	-5.52210	29.73649	-51.98313	0.36403	0.16302
K	C	0.38068	0.0†	18.22227	-27.55482	0.33869	0.39353
Ca	U	1.23567	-4.83833	17.53494	-51.57505	0.57991	0.29874
Ca	C	0.53000	0.0†	10.79150	-22.18560	0.51358	0.41107

†by the definition of the constrained spline.

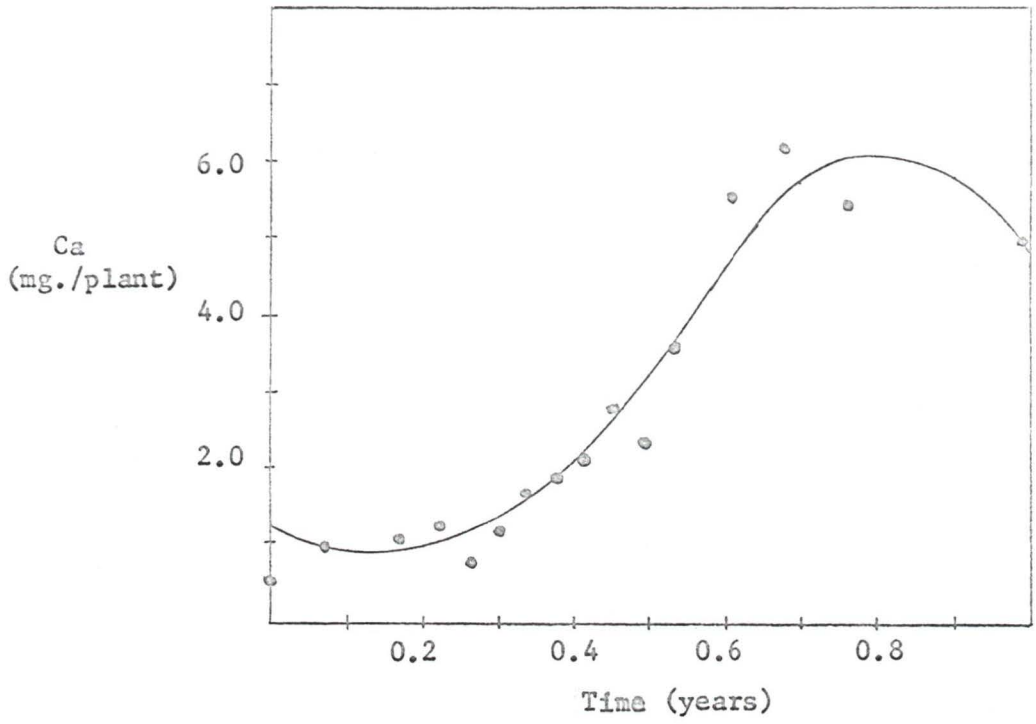


(a) Unconstrained spline.

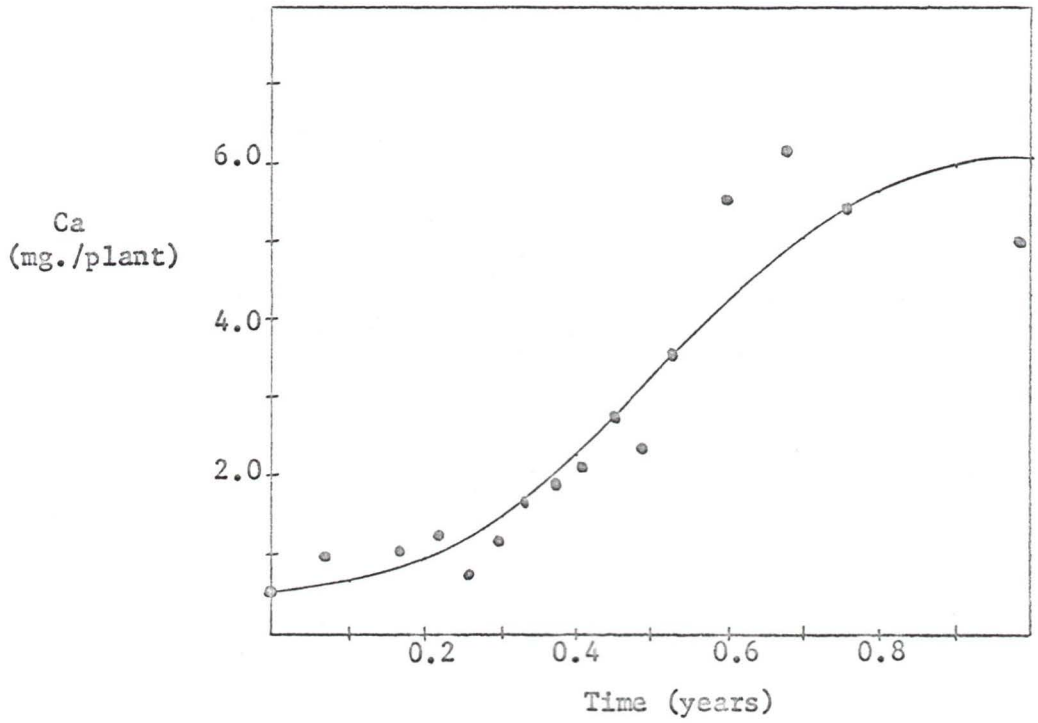


(b) Constrained spline

Fig. 10. Best ℓ_1 approximations for P content of White Spruce by a quadratic spline.

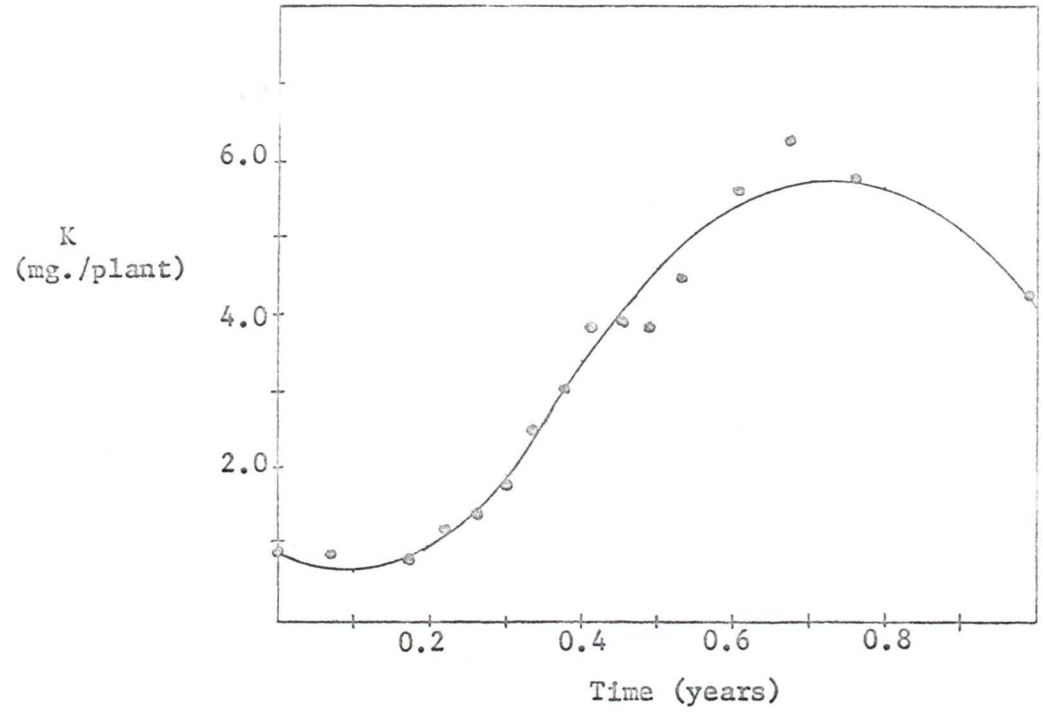


(a) Unconstrained spline.

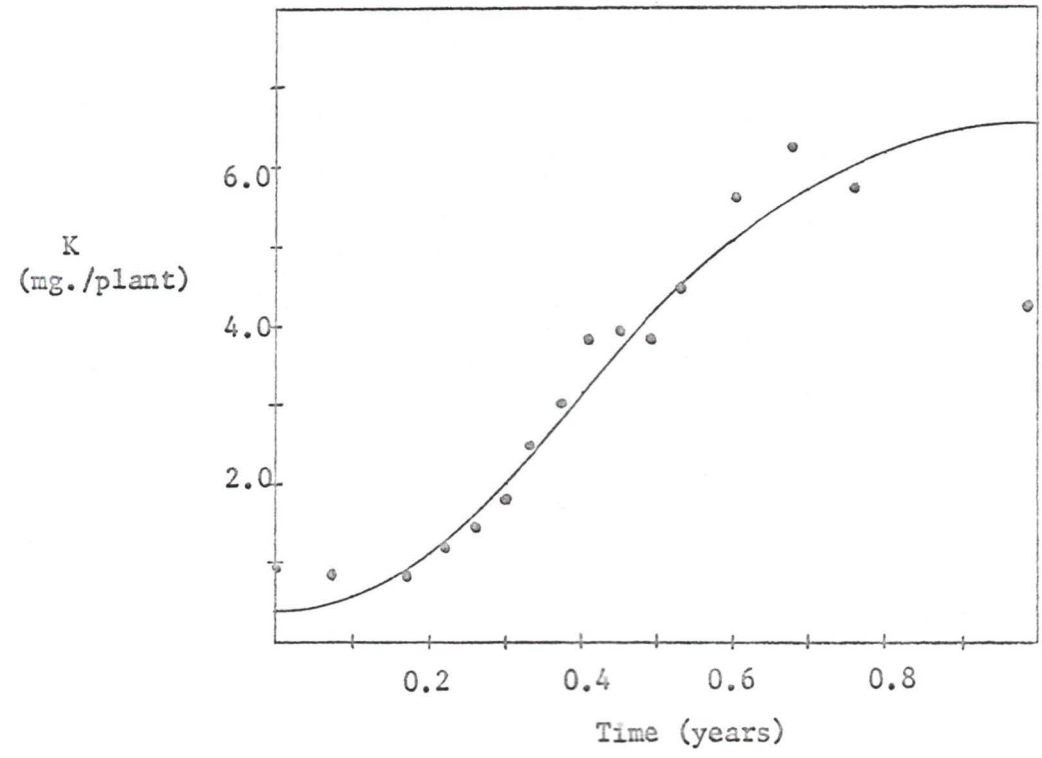


(b) Constrained spline.

Fig. 11. Best ℓ_1 approximations for Ca content of White Spruce by a quadratic spline.



(a) Unconstrained spline.



(b) Constrained spline.

Fig. 12. Best ℓ_1 approximations for K content of White Spruce by a quadratic spline.

V-2. Extension to Functions Nonlinear in Two Parameters.

The approximations of the previous section provide a satisfactory indication of the time of maximum mineral increase. Generally, however, one would desire more information than these functions provide. One model for describing data of the type found therein yields the Richards' function, which is defined by the differential equation

$$\frac{dW}{dt} = \frac{kW}{nA^n}(A^n - W^n) \tag{O}$$

where W and t are as in the previous section, and A, k and n are constants. The integrated form of (O), given by Causton (1969), is

$$W = A(1 \pm be^{-kt})^{-1/n} \tag{P}$$

where b is associated with the constant of integration, while the sign of b is positive or negative as n is positive or n lies in the range $-1 \leq n < 0$. The function is not defined for $n < -1$ or $n = 0$.

Causton (1969) discusses the solution of the normal equations for the ℓ_2 norm, involving Newton-Raphson in four dimensions. The numerical difficulties in this method are well known (Ralston (1965)). He changes the problem from fitting a curve of the form (P) to fitting a curve of the form

$$Q = R + S \log(1 + Be^{-kt}) \tag{Q}$$

where $Q = \log W$, $R = \log A$, $S = -\frac{1}{n}$ and $B = |b|$. Because of the very large values which B may assume, we found it convenient to define $\beta = \log B$ and attempt an approximation with the function

$$Q = R + S \log(1 + e^{\beta - kt}) \qquad (R)$$

From the standpoint of the technique discussed in this thesis, approximations with (R) are not nearly as formidable as would be those with (P). We are now faced with the problem of fitting an approximating function which is nonlinear in two of its parameters, namely β and k.

The manner in which we have adapted our technique to handle a case such as this is as follows. Intervals are chosen for β and k, in which are expected to lie those values of β and k which minimize the error function

$$G(\beta,k) = \min_{a_1,\ldots,a_{n-2}} \| f(t) - F(A,t) \|_p \qquad (S)$$

where, in this specific case $f(t_i) = Q_i$, $A = \{a_1,\ldots,a_{n-2},\beta,k\} = \{R,S,\beta,k\}$. In Phase I this area is covered by a mesh and $G(\beta,k)$ is calculated at each of the mesh points. The point, (β_i,k_j), which yields the minimum value of $G(\beta,k)$ for the mesh points can thus be determined by inspection. The optimal values β^* and k^* are then found by implementing Phase II as a Fibonacci search in two dimensions over the area determined by $\beta_{i-1} < \beta^* < \beta_{i+1}$ and $k_{j-1} < k^* < k_{j+1}$.

To accomplish a Fibonacci search in two dimensions for this case we first fix one of β and k, say $\beta = \beta_0^*$, and execute a one-dimensional Fibonacci search on k over the interval (k_{j-1},k_{j+1}) to find k_0^*, that value of k which minimizes $G(\beta_0^*,k)$. Next fix $k = k_0^*$ and execute a Fibonacci search on β over the interval $(\beta_{i-1},\beta_{i+1})$ to find β_1^*, that value of β which minimizes $G(\beta,k_0^*)$. We continue

in this fashion, fixing $\beta = \beta_m^*$ to calculate k_m^* and $k = k_m^*$ to calculate β_{m+1}^*. This procedure is halted when $|\beta_m^* - \beta_{m-1}^*|$ or $|k_m^* - k_{m-1}^*|$, according as k or β is fixed, is less than some prescribed tolerance. At this point we assume that we have β^* and k^* as our most recently calculated values, β_m^* and k_m^*, or, β_m^* and k_{m-1}^*.

We tested this procedure in the ℓ_2 norm with an example from Causton (1969), attempting to fit an approximation of the form (R) to the data in Table XLII.

TABLE XLII: Data used for testing the two-dimensional Fibonacci search

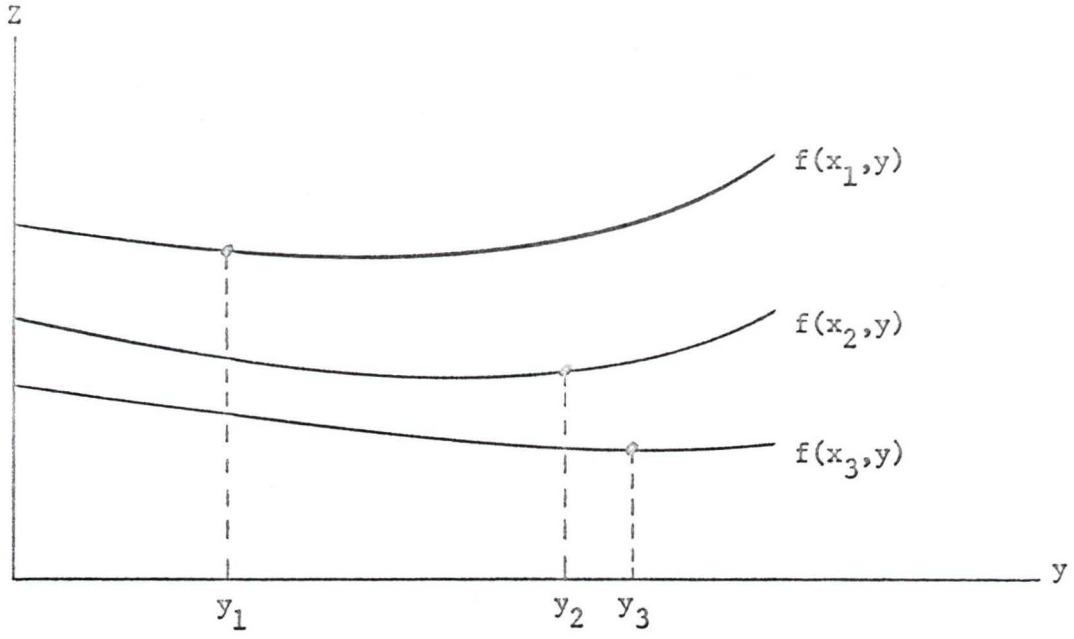
t (weeks)	W (grams)	t (weeks)	W (grams)
2	0.3898	14	5.610
4	0.8110	16	9.698
6	1.293	18	12.56
8	1.840	20	21.85
10	2.590	22	23.12
12	3.770	24	20.03

The approximation, while differing somewhat from Causton's (see Table XLIII), resulted in the same error of approximation, as shown in Table XLIII. It is evident from Table XLIII that the error of approximation when using (Q), and therefore (R), is very insensitive to changes in its nonlinear parameters. In other words, the error of approximation produces a flat function in a large area about its minimum.

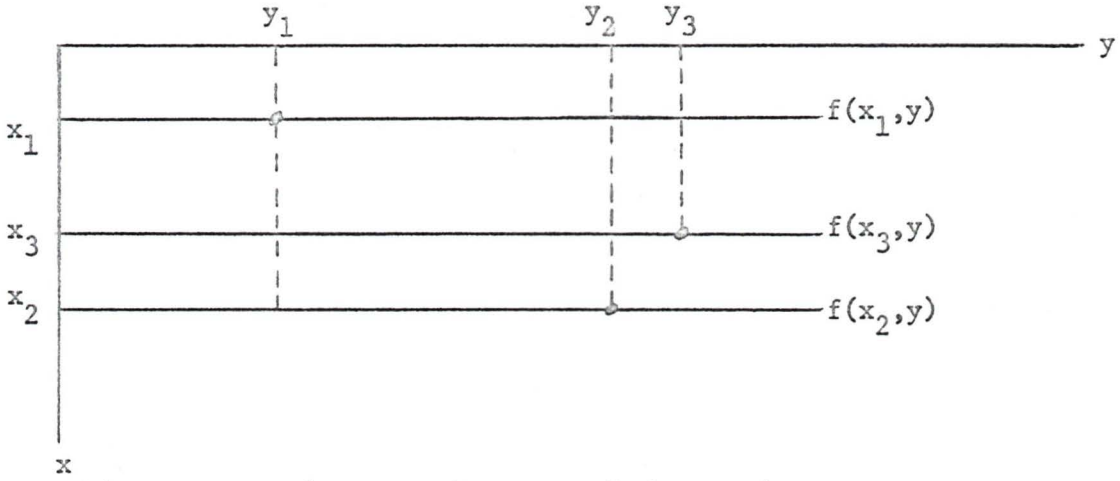
TABLE XLIII Complete details of best ℓ_2 approximations to the data of Table XLII by a function of the form $Q = R + S \log(1 + Be^{-kt})$

Parameter	Two-Dimensional Search	Causton's Results
R	3.0773	3.0772
S	-0.0402	-0.0400
B	4.1313×10^{45}	7.7020×10^{45}
k	5.2460	5.2149
$\Sigma\{f_i - F(A^*,t)\}^2$	0.1159	0.1159

This flatness of a function in an area about its minimum can lead to difficulties as the following simplified argument shows. Consider a unimodal function of two variables, say $z = f(x,y)$, which is quite flat in a large interval about its minimum in the z-y plane. Assume that we have completed Phase I of our search and we are about to proceed with Phase II over the area defined by $x_1 < x < x_2$ and $y_1 < y < y_2$. It is clear that a situation such as is shown in Fig. 13 could easily arise. Fig. 13(b) shows the area we are about to search, the rectangle defined by the four points (x_1,y_1), (x_1,y_2), (x_2y_2), (x_2,y_1). Suppose x_3 is one of the x values which occurs in the Fibonacci search over (x_1,x_2), and x_3 is such that $f(x_3,y_j) < f(x_i,y_j)$ for all $x_i (i \neq 3)$, y_j which occur in our two-dimensional Fibonacci search. We then have a situation as pictured in Fig. 13(a). The algorithm we have described above for executing a two-dimensional Fibonacci search will result in the minimum of z being given at the point (x_3,y_2), when clearly it shall be given at the point (x_3,y_3). Our Phase I, then, did not isolate the minimum.



(a) z-y plane (x protruding out of the page).



(b) x-y plane (z protruding out of the page).

Fig. 13. Views of $z = f(x_i,y)$ for $i=1,2,3$ in the z-y and x-y planes.

Unfortunately, this situation is recurrent with these flat functions, i.e. if we now expand our area over which we are executing our search, an x_4 can exist which will cause the same difficulty as described above. It is also possible that, even if we do execute Phase II successfully, somewhere in our final area of uncertainty there exists a value of one of the independent variables which will result in the above occurring.

It appears that the effectiveness of this type of search would increase as the minimum of the function became more pronounced.

We turn now to the problem of approximating the data of Table XL with the form (P). As in the cases involving quadratic spline approximations the ℓ_1 norm was used. It is not surprising, in view of the results shown in Table XLI, that, for this data, the error function (S) proved to be very flat in a large area about its minimum. The result was that we were unable to distinguish best ℓ_1 approximations of the form (P) for the data of Table XL.

In summary, extending our method to two or more nonlinear parameters is feasible so long as the minimum point is well defined.

REFERENCES

BARRODALE, I. (1968). L_1 Approximation and the Analysis of Data. Applied Stat., Vol. 17, pp. 51-57.

BARRODALE, I., and YOUNG, A. (1966). Algorithms for Best L_1 and L_∞ Linear Approximations on a Discrete Set. Numer. Math., Vol. 8, pp. 295-306.

BARRODALE, I., and YOUNG, A. (1966a). A Note on Numerical Procedures for Approximation by Spline Functions. The Computer Journal, Vol. 9, pp. 318-320.

BARRODALE, I., ROBERTS, F.D.K., and HUNT, C.R. (1970). Computing Best ℓ_p Approximations by Functions Nonlinear in One Parameter, The Computer Journal, to appear.

BELLMAN, R. (1957). Dynamic Programming. Princeton, N.J.: Princeton University Press.

CAUSTON, D.R. (1969). A Computer Program for Fitting the Richards Function. Biometrics, Vol. 25, No. 2, pp. 401-409.

DAVIS, P.J., and RABINOWITZ, P. (1961). Advances in Orthonormalizing Computation. From Advances in Computers, Vol. 2, edited by F.L. Alt. New York, N.Y.: Academic Press.

POWELL, M.J.D. (1964). An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives. The Computer Journal, Vol. 7, pp. 155-162.

RABENSTEIN, A.L. (1966). Introduction to Ordinary Differential Equations. New York, N.Y.: Academic Press.

RALSTON, A. (1965). A First Course in Numerical Analysis. New York, N.Y.: McGraw-Hill.

RICE, J.R. (1964). The Approximation of Functions, Vol. 1. Reading, Mass.: Addison-Wesley.

RICE, J.R. (1969). The Approximation of Functions, Vol. 2. Reading, Mass.: Addison-Wesley.

RICE, J.R., and WHITE, J.S. (1964). Norms for Smoothing and Estimation. SIAM Rev., Vol. 6, pp. 243-256.

WILDE, D.J. (1964). Optimum Seeking Methods. Englewood Cliffs, N.J.: Prentice-Hall.

van den DRIESSCHE, R. (1968). Growth Analysis of Four Nursery-Grown Conifer Species. Canadian Journal of Botany, Vol. 46, pp. 1389-1395.

APPENDICES

```
      IMPLICIT REAL*8(A-H,O-Z)                                          1
      INTEGER POLY,FR,WT,SC,RST,RNVT,RPT                                2
      LOGICAL LA                                                        3
      DIMENSION Q(26,6),W(21),X(21),Y(21),LA(21)                        4
C                                                                       5
CCCC  INPUT AND INITIALIZE ARGUMENTS FOR SUBROUTINE MINSUM              6
C                                                                       7
      READ(5,1)N,M,POLY,FR,WT,SC,RST,IT,ALPHA,RNVT,RPT,TOLER,QMAX       8
    1 FORMAT(2I5,5I1,2I5,2I1,2F20.0)                                    9
      N1=N+1                                                            10
      N5=N+5                                                            11
      M4=M+4                                                            12
      M1=M+1                                                            13
C                                                                       14
CCCCC INITIALIZE ENTRIES OF X                                           15
C                                                                       16
      DO 3 I=1,N                                                        17
    3 X(I)=(I-1)/20.D0                                                  18
      DO 200 LMN=1,12                                                   19
C                                                                       20
CCCCC INPUT LOWER BOUND ON NONLINEAR PARAMETER AND STEP SIZE            21
CCCCC READ ENTRIES OF Y                                                 22
C                                                                       23
      READ(5,4)CSMALL,CSTEP                                             24
    4 FORMAT(8F10.0)                                                    25
      READ(5,4)(Y(I),I=1,N)                                             26
      WRITE(6,7)LMN                                                     27
    7 FORMAT('1FUNCTION NUMBER:',I3)                                    28
      WRITE(6,50)                                                       29
   50 FORMAT('0',T13,'C',T33,'ERROR',T53,'DELTA Q',T73,'EXIT NUMBER',   30
     1T93,'A',T113,'B'//)                                               31
      C=CSMALL-CSTEP                                                    32
      DO 100 IJK=1,201                                                  33
      C=C+CSTEP                                                         34
C                                                                       35
CCCC  INITIALIZE Q                                                      36
C                                                                       37
                                                                        38
                                                                        39
                                                                        40
                                                                        41
                                                                        42
                                                                        43
                                                                        44
```

```
C                                                                       45
      DO 8 I=2,N1                                                       46
      Q(I,1)=Y(I-1)                                                     47
      Q(I,2)=1.0                                                        48
    8 Q(I,3)=DEXP(C*X(I-1))                                             49
      CALL MINSUM(N5,M4,N,Q,POLY,FR,WT,W,SC,RST,IT,ALPHA,RNVT,RPT,LA,   50
     1TOLER,QMAX)                                                       51
      DELQ=Q(1,1)-ERPREV                                                52
      IF(IJK.EQ.1)DELQ=0.0                                              53
      ERPREV=Q(1,1)                                                     54
C                                                                       55
CC                                                                      56
CC    OUTPUT COEFFICIENTS OF APPROXIMATION, ERROR OF APPROXIMATION AND  57
CC    FIRST DIFFERENCE FOR ERROR OF APPROXIMATION                       58
CC                                                                      59
C                                                                       60
      WRITE(6,9)C,Q(1,1),DELQ,Q(N+2,M+3),(Q(N+3,LL),LL=2,M1)            61
    9 FORMAT(' ',T5,6D20.7)                                             62
  100 CONTINUE                                                          63
  200 CONTINUE                                                          64
      CALL EXIT                                                         65
      END                                                               66
```

```
      IMPLICIT REAL*8(A-H,O-Z)                                          1
      INTEGER POLY,FR,WT,SC,RST,RNVT,RPT,X0,X1,X2,X3                    2
      LOGICAL LA                                                        3
      DIMENSION Q(26,6),W(21),X(21),Y(21),LA(21),QFX(2)                 4
C                                                                       5
CCCC                                                                    6
CCCC  INPUT AND INITIALIZE ARGUMENTS FOR SUBROUTINE MINSUM              7
CCCC                                                                    8
C                                                                       9
      READ(5,1)N,M,POLY,FR,WT,SC,RST,IT,ALPHA,RNVT,RPT,TOLER,QMAX       10
    1 FORMAT(2I5,5I1,2I5,2I1,2F20.0)                                    11
      N1=N+1                                                            12
      N5=N+5                                                            13
      M4=M+4                                                            14
      M1=M+1                                                            15
      LMN=0                                                             16
C                                                                       17
CCCC                                                                    18
CCCC  INITIALIZE ENTRIES OF X                                           19
CCCC                                                                    20
C                                                                       21
      DO 2 I=1,N                                                        22
    2 X(I)=(I-1.0)/20.D0                                                23
C                                                                       24
CCCC                                                                    25
CCCC  READ BOUNDS OF INTERVAL TO BE SEARCHED AND ENTRIES OF Y           26
CCCC                                                                    27
C                                                                       28
    3 READ(5,4,END=100)A,B,EPS                                          29
    4 FORMAT(8F10.0)                                                    30
      READ(5,4)(Y(I),I=1,N)                                             31
      LMN=LMN+1                                                         32
      WRITE(6,7)LMN,A,B,EPS                                             33
    7 FORMAT('1FUNCTION NUMBER:',I3,5X,F10.5,'< C <',F10.5,5X,          34
     1'TOLERANCE =',D10.2)                                              35
      WRITE(6,8)                                                        36
    8 FORMAT('0',T13,'C',T30,'ERROR',T47,'A',T64,'B')                   37
C                                                                       38
CCCC                                                                    39
CCCC  PERFORM FIBONACCI SEARCH                                          40
CCCC                                                                    41
C                                                                       42
      ITN=0                                                             43
      X1=1                                                              44
```

```
      X2=1                                                              45
    9 X0=X2                                                             46
      X2=X1+X2                                                          47
      X1=X0                                                             48
      IF((2.0*X2+X1)*EPS.LT.(B-A))GO TO 9                               49
      H=(B-A)/(X1+2.0*X2)                                               50
      X0=X1                                                             51
      X1=X2                                                             52
      X2=X0+X1                                                          53
      X0=0                                                              54
      LL=-1                                                             55
      C=A+X1*H                                                          56
C                                                                       57
CCCC                                                                    58
CCCC  INITIALIZE ENTRIES OF Q                                           59
CCCC                                                                    60
C                                                                       61
   10 DO 11 I=2,N1                                                      62
      Q(I,1)=Y(I-1)                                                     63
      Q(I,2)=1.0                                                        64
   11 Q(I,3)=DEXP(C*X(I-1))                                             65
      CALL MINSUM(N5,M4,N,Q,POLY,FR,WT,W,SC,RST,IT,ALPHA,RNVT,RPT,LA,   66
     1TOLER,QMAX)                                                       67
      ITN=ITN+1                                                         68
C                                                                       69
CCCC                                                                    70
CCCC  CHECK THAT MINSUM WAS EXECUTED PROPERLY                           71
CCCC                                                                    72
C                                                                       73
      IF(Q(N+2,M+3).NE.1.0)WRITE(6,12)Q(N+2,M+3)                        74
   12 FORMAT('0ABNORMAL EXIT FROM SUBROUTINE**********EXIT NUMBER = ',  75
     1F4.0)                                                             76
      IF(LL)13,14,15                                                    77
   13 FX1=Q(I,1)                                                        78
      DO 51 I=1,M                                                       79
   51 QFX(I)=Q(N+3,I+1)                                                 80
      LL=0                                                              81
      C=A+X2*H                                                          82
      GO TO 10                                                          83
   14 FX2=Q(I,1)                                                        84
      DO 52 I=1,M                                                       85
   52 QFX(I)=Q(N+3,I+1)                                                 86
      GO TO 16                                                          87
   15 FX1=Q(I,1)                                                        88
```

```
      DO 53 I=1,M                                                       89
   53 QFX(I)=Q(N+3,I+1)                                                 90
   16 IF((X1-X0).EQ.1)GO TO 19                                          91
      IF(FX1-FX2)17,17,18                                               92
   17 X3=X1                                                             93
      X1=X0+X2-X1                                                       94
      X2=X3                                                             95
      FX2=FX1                                                           96
      C=A+X1*H                                                          97
      LL=1                                                              98
      GO TO 10                                                          99
   18 X3=X2                                                            100
      X2=X1+X1-X0                                                      101
      X0=X1                                                            102
      X1=X3                                                            103
      FX1=FX2                                                          104
      C=A+X2*H                                                         105
      LL=0                                                             106
      GO TO 10                                                         107
   19 IF(FX1-FX2)20,20,21                                              108
   20 C=A+X1*H                                                         109
      ERROR=FX1                                                        110
      GO TO 25                                                         111
   21 C=A+X2*H                                                         112
      ERROR=FX2                                                        113
C                                                                      114
C                                                                      115
CCC   OUTPUT PARAMETERS OF BEST APPROXIMATION AND ERROR OF APPROXIMATION 116
C                                                                      117
C                                                                      118
   25 WRITE(6,26)C,ERROR,(QFX(I),I=1,M)                                119
   26 FORMAT('0',T5,7D17.7)                                            120
      WRITE(6,28)ITN                                                   121
   28 FORMAT('0ITERATIONS OF FIBONACCI SEARCH =',I3)                   122
      WRITE(6,27)                                                      123
   27 FORMAT(////)                                                     124
      I1=ALPHA+2                                                       125
      M3=M+3                                                           126
      DO 900 I=I1,N1                                                   127
  900 WRITE(6,901)Q(I,1),Q(I,M3)                                       128
  901 FORMAT(' ',2D20.7)                                               129
      WRITE(6,902)                                                     130
  902 FORMAT('0')                                                      131
      ESUM=0.D0                                                        132
```

```
      DO 910 I=1,N                                                      133
      E=Y(I)-QFX(1)-QFX(2)*DEXP(C*X(I))                                 134
      ESUM=ESUM+DABS(E)                                                 135
C                                                                       136
CC                                                                      137
CCC   OUTPUT ERROR OF APPROXIMATION AT EACH DATA POINT                  138
CCC                                                                     139
C                                                                       140
  910 WRITE(6,911)I,E                                                   141
  911 FORMAT(' X(',I2,')',5X,D20.7)                                     142
      ESUM=ESUM/N                                                       143
      WRITE(6,912)ESUM                                                  144
  912 FORMAT(/////' SUM OF ABS ERRORS DIVIDED BY N:',D20.7)             145
      GO TO 3                                                           146
  100 CALL EXIT                                                         147
      END                                                               148
```

```
      SUBROUTINE MINSUM(N5,M4,N,Q,POLY,FR,WT,W,SC,RST,IT,ALPHA,RNVT,RPT,
     1LA,TOLER,QMAX)
      IMPLICIT REAL*8(A-H,O-Z)
C
CCCCCC.....
CCCCCC
CCCCCC   SUBROUTINE MINSUM
CCCCCC
CCCCCC   PURPOSE
CCCCCC      MINSUM USES A MODIFIED SIMPLEX METHOD TO CALCULATE THE BEST
CCCCCC      L1 APPROXIMATION (MINIMIZE SUM OF ABSOLUTE DEVIATIONS),
CCCCCC      BY ANY LINEAR COMBINATION OF M GIVEN FUNCTIONS (WHICH MAY BE
CCCCCC      DEFINED IN A MULTIDIMENSIONAL SPACE), TO A FUNCTION F(X)
CCCCCC      DEFINED IN A TABLE AS N ORDERED PAIRS.
CCCCCC
CCCCCC   USAGE
CCCCCC      CALL MINSUM(N5,M4,N,Q,POLY,FR,WT,W,SC,RST,IT,ALPHA,RNVT,RPT,
CCCCCC           LA,TOLER,QMAX)
CCCCCC
CCCCCC   DESCRIPTION OF PARAMETERS
CCCCCC      N    - NUMBER OF DATA POINTS DEFINING THE FUNCTION F(X)
CCCCCC             WHICH IS TO BE APPROXIMATED
CCCCCC      N5   - N+5 (INPUT FOR ADJUSTABLE DIMENSIONS)
CCCCCC      M4   - NUMBER OF GIVEN FUNCTIONS FROM WHICH THE APPROXIMAT-
CCCCCC             ING FUNCTION IS CONSTRUCTED PLUS FOUR
CCCCCC      Q    - HAS THE BOUNDS (N5,M4) AND MUST ON ENTRY CONTAIN A
CCCCCC             COEFFICIENT MATRIX Q(2:N+1,1:M+1)- THE N VALUES OF
CCCCCC             F(X) ARE IN Q(I,1) FOR I=2,3,...,N+1 AND THE M SETS
CCCCCC             OF FUNCTIONAL VALUES GENERATED BY THE INDEPENDENT
CCCCCC             VARIABLE(S) ARE STORED IN Q(2:N+1,2:M+1)
CCCCCC      POLY - IF POLY=1 A BEST POLYNOMIAL APPROXIMATION OF DEGREE
CCCCCC             M-1 IS DETERMINED. HERE IT IS ONLY NECESSARY TO
CCCCCC             SUPPLY THE N ABSCISSAE OF F(X) TO Q(I,3) FOR I=2,3,
CCCCCC             ...,N+1, AND THE J'TH POWERS OF THESE VALUES ARE THEN
CCCCCC             COMPUTED AND STORED IN WORKING LOCATION.
CCCCCC      FR   - IF FR=1 THE FIRST M PIVOTAL COLUMN SELECTIONS ARE
CCCCCC             RESTRICTED TO THE M+1 COEFFICIENT VECTORS (LABELLED
CCCCCC             N+1,...,N+M+1). USUALLY FEWER ITERATIONS ARE REQUIRED
CCCCCC             WHEN FR=1 AND THERE IS ALSO A SMALL SAVING IN THE
CCCCCC             TIME REQUIRED TO SELECT THESE FIRST M COLUMNS
CCCCCC      WT   - IF WT=1 THE PROCEDURE CALCULATES A BEST WEIGHTED
CCCCCC             APPROXIMATION
CCCCCC      W    - HAS THE BOUNDS (1:N) AND MUST CONTAIN THE N WEIGHTS
CCCCCC             REQUIRED WHEN WT=1
```

```
C
C          SC    - IF SC=1 THE M+2 COLUMNS OF THE INITIAL MATRIX ARE
C                  SCALED BY DIVIDING EACH COLUMN BY ITS ABSOLUTE
C                  LARGEST ELEMENT. THE FINAL MATRIX IS RECONVERTED
C                  BEFORE EXIT FROM THE SUBROUTINE.
C          RST   - IF RST=1 THE SUBROUTINE IS RESTARTED AT STEP1. THE
C                  MATRIX Q(1:N+4,1:M+3) PRODUCED BY THE LAST ITERATION
C                  MUST BE SUPPLIED, AND THE CALCULATIONS ARE THEN
C                  CONTINUED.
C          IT      NUMBER OF ITERATIONS COMPLETED AT LAST EXIT. THIS
C                  INFORMATION NEED ONLY BE SUPPLIED TO THE SUBROUTINE
C                  WHEN RST=1. HOWEVER WHEN RPT=1 THIS ALSO ACTS AS AN
C                  INPUT PARAMETER WHICH SPECIFIES HOW MANY ITERATIONS
C                  TO REPEAT.
C          ALPHA-  NUMBER OF COEFFICIENT VECTORS IN THE BASIS. THIS
C                  INFORMATION NEED ONLY BE SUPPLIED TO THE SUBROUTINE
C                  WHEN RST=1 OF WHEN RNVT=1.
C          RNVT  - IF RNVT=1 THE FINAL BASIS TO DATE IS REINVERTED IN
C                  THE MINIMUM NUMBER OF ITERATIONS (=ALPHA). STARTING
C                  AGAIN FROM THE INITIAL MATRIX THE CALCULATIONS ARE
C                  REPEATED MORE EFFICIENTLY AND ACCURATELY BY ELIMINA-
C                  TING ALL THE UNNECESSARY PIVOTING. THE INITIAL COEF-
C                  FICIENT MATRIX IS STORED IN Q(2:N+1,1:M+1) BUT OTHER-
C                  WISE THE ARRAY Q IS SUPPLIED UNCHANGED FROM ITS
C                  PREVIOUS FINAL FORM. WHEN RNVT=1 THE PARAMETERS FR
C                  AND RNVT ARE SET EQUAL TO 0 BEFORE EXITING FROM THE
C                  SUBROUTINE.
C          RPT   - IF RPT=1 THE CALCULATIONS ARE REPEATED FOR A
C                  SPECIFIED NUMBER (=IT) OF ITERATIONS. THIS IS USUALLY
C                  EMPLOYED WHEN REINVERTING A PROBLEM WHICH HAS NOT
C                  BEEN SOLVED SUCCESSFULLY AT THE FIRST ATTEMPT. FOR
C                  EXAMPLE, WE MIGHT DECIDE TO REPEAT THE CALCULATIONS
C                  FOR 20 ITERATIONS, WHICH INCLUDE 10 ITERATIONS FOR
C                  REINVERSION. THIS INTERMEDIATE BASIS CAN THEN BE RE-
C                  INVERTED IN 10 ITERATIONS, AND A FURTHER 10 ITERA-
C                  TIONS PERFORMED, AND SO ON. PROCEEDING IN THIS MANNER
C                  WE CAN SOMETIMES ARRIVE AT AN OPTIMUM SOLUTION, WHERE
C                  INITIALLY THIS HAS NOT BEEN POSSIBLE.
C          TOLER-  A POSITIVE TOLERANCE LESS THAN WHICH QUANTITIES
C                  ARE CONSIDERED TO BE NONPOSITIVE. EMPIRICAL EVIDENCE
C                  SUGGESTS THAT INPUTTING TOLER=.00001 IS SUITABLE
C                  WHEN SEVEN DECIMAL DIGITS ARE AVAIBLE, AND PUTTING
C                  TOLER=.00000001 WITH ELEVEN DECIMAL DIGITS. DECREAS-
C                  ING THE MAGNITUDE OF TOLER USUALLY DECREASES THE
C                  TOTAL NUMBER OF ITERATIONS REQUIRED.
```

```
C          QMAX - A LARGE POSITIVE NUMBER WHICH THE MINIMUM QUOTIENT          89
C                 (WHICH DETERMINES THE PIVOT) MUST NOT EXCEED. PUTTING       90
C                 QMAX=1/TOLER IS USUALLY SATISFACTORY.                       91
C          IF ANY OF THE INTEGERS POLY,FR,WT,SC,RST,RNVT,OR RPT IS NOT        92
C                 EQUAL TO 1 THEN NO ACTION IS TAKEN.                         93
C          THE SUBROUTINE TERMINATES FOR THE FOLLOWING REASONS:               94
C             (1) A BEST APPROXIMATION IS FOUND SINCE ALL THE MARGINAL        95
C                 COSTS ARE LESS OR EQUAL TO TOLER. THIS IS INDICATED         96
C                 BY Q(N+2,M+3)=1. THE ERROR OF APPROXIMATION (SUM OF        97
C                 WEIGHTED ABSOLUTE DEVIATIONS) IS GIVEN BY Q(1,1) AND       98
C                 THE M COEFFICIENTS OF THE BEST APPROXIMATION ARE           99
C                 CONTAINED IN ORDER BY Q(N+3,J) FOR J=2,3,...,M+1.         100
C                 THE WEIGHTED DEVIATIONS AT THOSE POINTS WHICH ARE NOT     101
C                 INTERPOLATED BY THE APPROXIMATING FUNCTION ARE IN         102
C                 Q(I,1) FOR I=ALPHA+2,...,N+1 WITH THEIR CORRESPONDING     103
C                 LABELS AND SIGNS IN Q(I,M+3).                             104
C             (2) CALCULATION IS STOPPED IF NO SUITABLE PIVOT CAN BE        105
C                 FOUND. THIS IS INDICATED BY Q(N+2,M+3)=2.                 106
C             (3) A SPECIFIED NUMBER OF ITERATIONS HAS BEEN SUCCESSFUL-     107
C                 LY COMPLETED WITH RNVT=1 AND RPT NOT EQUAL TO 1. THIS     108
C                 IS INDICATED BY Q(N+2,M+3)=3.                             109
C             (4) A SPECIFIED NUMBER OF ITERATIONS HAS BEEN SUCCESSFUL-     110
C                 LY COMPLETED WITH RPT=1. THIS IS INDICATED BY             111
C                 Q(N+2,N+3)=4.                                             112
C     ..........................................................            113
C                                                                           114
      INTEGER POLY,FR,WT,SC,RST,RNVT,RPT,ALPHA,T,R,AL,OUT                   115
      LOGICAL LB,LA(N)                                                      116
      DIMENSION Q(N5,M4),W(N)                                               117
      M=M4-4                                                                118
      M1=M+1                                                                119
      M2=M+2                                                                120
      M3=M+3                                                                121
      N1=N+1                                                                122
      N2=N+2                                                                123
      N3=N+3                                                                124
      N4=N+4                                                                125
C                                                                           126
CCCC  TESTS INPUT PARAMETERS                                                127
C                                                                           128
C                                                                           129
      IF(RPT.EQ.1) GO TO 95                                                 130
      R=-1                                                                  131
                                                                            132
```

```
      GO TO 96
   95 R=IT
   96 IF(RNVT.EQ.1) GO TO 97
      AL=-1
      GO TO 105
   97 AL=ALPHA
      FR=0
      DO 101 I=1,N
  101 LA(I)=.FALSE.
  105 IF(RST.NE.1) GO TO 110
      LINE=ALPHA+1
      IT=IT-1
      GO TO 140
  110 IF(POLY.NE.1) GO TO 115
      DO 111 I=2,N1
  111 Q(I,2)=1.0
      IF(M.EQ.2) GO TO 115
      DO112 I=2,N1
      DO112 J=3,M
  112 Q(I,J+1)=Q(I,J)*Q(I,3)
  115 IF(WT.NE.1) GO TO 120
      DO 116 I=2,N1
      A=W(I-1)
      DO 116 J=1,M1
  116 Q(I,J)=Q(I,J)*A
  120 DO 121 J=2,M2
      Q(1,J)=0.0
  121 Q(N2,J)=N+J-1
      Q(1,1)=0.0
      Q(N2,1)=0.0
      DO 125 I=2,N1
      Q(I,M3)=I-1
      A=0.0
      LB=Q(I,1).LT.0.0
      DO 124 J=1,M1
      IF(LB) Q(I,J)=-Q(I,J)
      B=Q(I,J)
      A=A-B
  124 Q(1,J)=Q(1,J)+B
      IF(LB) Q(I,M3)=-Q(I,M3)
      B=A+Q(I,1)
      Q(I,M2)=B
  125 Q(1,M2)=Q(1,M2)+B
      IF(SC.NE.1) GO TO 131
```

```
      DO 130 J=1,M2                                                     177
      IF(RNVT.NE.1.AND.RPT.NE.1) GO TO 127                              178
      A=Q(N4,J)                                                         179
      GO TO 129                                                         180
  127 A=0.0                                                             181
      DO 128 I=1,N1                                                     182
  128 IF(DABS(Q(I,J)).GT.A) A=DABS(Q(I,J))                              183
      Q(N4,J)=A                                                         184
  129 DO 126 I=1,N1                                                     185
  126 Q(I,J)=Q(I,J)/A                                                   186
  130 CONTINUE                                                          187
  131 IF(RNVT.NE.1) GO TO 136                                           188
      DO 135 I=2,N1                                                     189
      T=DABS(Q(I,M4))                                                   190
      IF(Q(I,M4)+Q(T+1,M3).NE.0.0) GO TO 133                            191
      DO 132 J=1,M3                                                     192
      Q(T+1,J)=-Q(T+1,J)                                                193
  132 Q(1,J)=Q(1,J)+2.*Q(T+1,J)                                         194
  133 IF(I-1.GT.AL) GO TO 135                                           195
      Q(I,M4)=T                                                         196
      LA(T)=.TRUE.                                                      197
  135 CONTINUE                                                          198
  136 IT=-1                                                             199
      ALPHA=0                                                           200
      LINE=1                                                            201
C                                                                       202
CCC                                                                     203
CCC   STEP1-DETERMINES PIVOTAL COLUMN                                   204
CCC                                                                     205
C                                                                       206
  140 A=0.0                                                             207
      IN=1                                                              208
      T=0                                                               209
      IT=IT+1                                                           210
      IF(RPT.NE.1.OR.IT.NE.R) GO TO 145                                 211
      Q(N2,M3)=4.0                                                      212
      GO TO 186                                                         213
  145 IF(RNVT.EQ.1.OR.FR.NE.1.OR.IT.GE.M) GO TO 147                     214
      DO 146 J=2,M2                                                     215
      IF(Q(N2,J).LE.N.OR.Q(1,J).LE.TOLER) GO TO 146                     216
      IN=J                                                              217
      GO TO 147                                                         218
  146 CONTINUE                                                          219
  147 IF(RNVT.EQ.1) IN=Q(N5,IT+2)+1                                     220
```

```
      IF(IN.EQ.1) GO TO 148                                              221
      ALPHA=ALPHA+1                                                      222
      T=1                                                                223
      GO TO 163                                                          224
  148 LINEA=LINE+1                                                       225
      DO 160 J=2,M2                                                      226
      Z=Q(1,J)                                                           227
      IF(Q(N2,J).GT.N) IF(ALPHA-M)155,160,155                            228
      IF(Z+2.GE.-TOLER) GO TO 155                                        229
      P=0.0                                                              230
      DO 150 I=LINEA,N1                                                  231
  150 IF(Q(I,J).LT.-TOLER) P=P+Q(I,J)                                    232
      IF(P.LT.-TOLER) P=(Z+2.0)/P                                        233
      IF(P.LE.A) GO TO 155                                               234
      IN=J                                                               235
      T=2                                                                236
      A=P                                                                237
      GO TO 160                                                          238
  155 IF (Z.LE.TOLER) GO TO 160                                          239
      P=0.0                                                              240
      DO 156 I=LINEA,N1                                                  241
  156 IF(Q(I,J).GT.TOLER) P=P+Q(I,J)                                     242
      IF(P.GT.TOLER) P=Z/P                                               243
      IF(P.LE.A) GO TO 160                                               244
      IN=J                                                               245
      T=1                                                                246
      A=P                                                                247
  160 CONTINUE                                                           248
      IF(T.NE.0) GO TO 161                                               249
      Q(N2,M3)=1.0                                                       250
      GO TO 186                                                          251
  161 IF(ALPHA.NE.AL.OR.RPT.EQ.1) GO TO 162                              252
      Q(N2,M3)=3.0                                                       253
      GO TO 186                                                          254
  162 IF(Q(N2,IN).GT.N) ALPHA=ALPHA+1                                    255
C                                                                        256
CCCC                                                                     257
CCCC    STEP2-DETERMINES THE PIVOTAL ELEMENT                             258
CCCC                                                                     259
C                                                                        260
  163 A=TOLER                                                            261
      B=QMAX                                                             262
      D=0.0                                                              263
      IF(RNVT.NE.1) GO TO 166                                            264
```

```
      DO 165 I=1,AL                                                     265
      J=Q(I+1,M4)                                                       266
      IF(LA(J)) D=DABS(Q(J+1,IN))                                       267
      IF(D.LE.A) GO TO 165                                              268
      A=D                                                               269
      OUT=J+1                                                           270
  165 CONTINUE                                                          271
      GO TO 171                                                         272
  166 DO 170 I=2,N1                                                     273
      D=Q(I,IN)                                                         274
      IF(T.EQ.2) D=-D                                                   275
      IF(D.LE.TOLER) GO TO 170                                          276
      D=Q(I,1)/D                                                        277
      IF(D.GE.B) GO TO 170                                              278
      B=D                                                               279
      OUT=I                                                             280
  170 CONTINUE                                                          281
  171 IF(Q(OUT,M3).GT.N) ALPHA=ALPHA-1                                  282
      IF(B.LT.QMAX.OR.A.GT.TOLER) GO TO 172                             283
      Q(N2,M3)=2.0                                                      284
      GO TO 186                                                         285
  172 IF(T.NE.2) GO TO 175                                              286
      Q(N2,IN)=-Q(N2,IN)                                                287
      Q(1,IN)=Q(1,IN)+2.0                                               288
C                                                                       289
C                                                                       290
CCC     STEP3-PERFORMS ONE SIMPLEX ITERATION                            291
CCC                                                                     292
C                                                                       293
  175 P=Q(OUT,IN)                                                       294
      DO 178 I=1,N1                                                     295
      IF(I.EQ.OUT) GO TO 178                                            296
      D=Q(I,IN)/P                                                       297
      DO 177 J=1,M2                                                     298
      IF(J.EQ.IN) GO TO 176                                             299
      Q(I,J)=Q(I,J)-D*Q(OUT,J)                                          300
      GO TO 177                                                         301
  176 Q(I,J)=-D                                                         302
  177 CONTINUE                                                          303
  178 CONTINUE                                                          304
      IF(RNVT.NE.1) P=DABS(P)                                           305
      DO 180 J=1,M2                                                     306
      IF(J.EQ.IN) GO TO 179                                             307
      Q(OUT,J)=Q(OUT,J)/P                                               308
```

```
      GO TO 180                                                                 309
  179 Q(OUT,J)=1./P                                                             310
  180 CONTINUE                                                                  311
      I=Q(OUT,M3)                                                               312
      B=Q(N2,IN)                                                                313
      Q(OUT,M3)=B                                                               314
      Q(N2,IN)=I                                                                315
      IF((I.GT.N.AND.B.GT.N).OR.(I.LE.N.AND.B.LE.N)) GO TO 140                  316
      IF(B.GT.N) LINE=LINE+1                                                    317
      DO 185 J=1,M3                                                             318
      Q(N3,J)=Q(OUT,J)                                                          319
      Q(OUT,J)=Q(LINE,J)                                                        320
  185 Q(LINE,J)=Q(N3,J)                                                         321
      IF(I.GT.N) LINE=LINE-1                                                    322
      IF(RNVT.NE.1) GO TO 140                                                   323
      LA(OUT-1)=LA(LINE-1)                                                      324
      LA(LINE-1)=.FALSE.                                                        325
      IF(ALPHA.EQ.AL) RNVT=0                                                    326
      GO TO 140                                                                 327
  186 IF(RST.EQ.1.OR.SC.NE.1) GO TO 191                                         328
C                                                                               329
CC    IF THE DATA WAS SCALED IT IS NOW RECONVERTED                              330
C                                                                               331
      DO 187 I=2,LINE                                                           332
      A=Q(N4,Q(I,M3)-N+1)                                                       333
      DO 187 J=1,M2                                                             334
  187 Q(I,J)=Q(I,J)/A                                                           335
      A=Q(N4,1)                                                                 336
      DO 188 I=1,N1                                                             337
  188 Q(I,1)=Q(I,1)*A                                                           338
      DO 190 J=2,M2                                                             339
      IF(Q(N2,J).LE.N) GO TO 190                                                340
      A=Q(N4,Q(N2,J)-N+1)                                                       341
      DO 189 I=1,N1                                                             342
  189 Q(I,J)=Q(I,J)*A                                                           343
  190 CONTINUE                                                                  344
C                                                                               345
CC                                                                              346
CC    STEP4-PREPARES OUTPUT                                                     347
CC                                                                              348
C                                                                               349
  191 A=0.0                                                                     350
      T=N+M1                                                                    351
      I=1                                                                       352
```

```
      DO 192 J=2,M2                                                     353
      IF(Q(N2,J).GT.N) GO TO 192                                        354
      I=I+1                                                             355
      Q(I,M4)=Q(N2,J)                                                   356
  192 CONTINUE                                                          357
      J=ALPHA+2                                                         358
      DO 193 I=J,N1                                                     359
  193 Q(I,M4)=Q(I,M3)                                                   360
      DO 194 J=1,ALPHA                                                  361
  194 Q(N5,J+1)=Q(J+1,M3)-N                                             362
      DO 195 I=2,LINE                                                   363
  195 IF(Q(I,M3).EQ.T) A=Q(I,1)                                         364
      DO 196 J=2,M1                                                     365
  196 Q(N3,J)=-A                                                        366
      DO 197 I=2,LINE                                                   367
  197 Q(N3,Q(I,M3)-N+1)=Q(I,1)-A                                        368
      RETURN                                                            369
      END                                                               370
```

```
      SUBROUTINE MINMAX(N,N2,M5,Q,X,X1,X2,POLY,FR,WT,W,SC,DEV,RST,IT,
     1ALPHA,TOLER,QMAX)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL AINT,P
C
C     ..................................................................
C
C        SUBROUTINE MINMAX
C
C        PURPOSE
C           MINMAX USES A MODIFIED SIMPLEX METHOD TO CALCULATE THE BEST
C           L-INFINITY APPROXIMATION, BY ANY LINEAR COMBINATION OF M
C           GIVEN FUNCTIONS (WHICH MAY BE DEFINED IN A MULTIDIMENSIONAL
C           SPACE), TO A FUNCTION F(X) DEFINED AS A TABLE OF N ORDERED
C           PAIRS.
C
C        USAGE
C           CALL MINMAX(N,N2,M5,Q,X,X1,X2,POLY,FR,WT,W,SC,DEV,RST,IT,
C                ALPHA,TOLER,QMAX)
C
C        DESCRIPTION OF PARAMETERS
C           N     - NUMBER OF DATA POINTS DEFINING THE FUNCTION F(X)
C                   WHICH IS TO BE APPROXIMATED.
C           N2    - INPUT PARAMETER WITH THE VALUE N PLUS TWO.
C           M5    - NUMBER OF GIVEN FUNCTIONS FROM WHICH THE
C                   APPROXIMATING FUNCTION IS CONSTRUCTED PLUS FIVE.
C           Q     - HAS DIMENSIONS (M5,N2) AND MUST ON ENTRY CONTAIN THE
C                   N VALUES OF F(X) IN Q(1,J) FOR J=2,3,...,N+1 AND THE
C                   M SETS OF FUNCTIONAL VALUES GENERATED BY THE
C                   INDEPENDENT VARIABLE(S) ARE STORED IN Q(3:M+2,2:N+1).
C           X     - HAS DIMENSION (N) AND WHEN FR=1 IT MUST CONTAIN THE N
C                   ABSCISSAE OF F(X) IN ASCENDING ORDER OF MAGNITUDE.
C           X1,X2- RESPECTIVELY THE LEFT AND RIGHT HAND ENDPOINTS OF THE
C                   INTERVAL FROM WHICH X IS SELECTED. THESE VALUES NEED
C                   ONLY BE SUPPIED WHEN FR=1.
C           POLY  - IF POLY=1 A BEST POLYNOMIAL APPROXIMATION OF DEGREE
C                   M-1 IS DETERMINED. HERE IT IS ONLY NECESSARY TO
C                   SUPPLY THE N ABSCISSAE OF F(X) TO Q(4,J) FOR J=1,...,
C                   N+1, AND THE J'TH POWERS OF THESE VALUES ARE THEN
C                   COMPUTED AND STORED IN WORKING LOCATION.
C           FR    - IF FR=1 THE FIRST M+1 PIVOTS ARE CHOSEN SO THAT AN
C                   INITIAL BEST APPROXIMATION IS DETERMINED ON THE M+1
C                   POINTS OF X THAT ARE CLOSEST TO THE EXTREMA OF THE
C                   RELEVANT CHEBYSHEV POLYNOMIAL. IF THE DISTRIBUTION OF
```

```
C                POINTS IN X PREVENTS THIS FROM BEING ARRANGED, THEN
C                THESE PIVOTS ARE CHOSEN CORRESPONDING TO M+1 POINTS
C                OF X WHOSE SUBSCRIPTS ARE IN ARITHMETIC PROGRESSION.
C                PUTTING FR=1 USUALLY REDUCES THE TOTAL NUMBER OF
C                ITERATIONS REQUIRED FOR POLYNOMIAL APPROXIMATION,AND
C                SOMETIMES REDUCES THIS TOTAL FOR GENERAL
C                APPROXIMATION IN ONE DIMENSION.
C      WT     -  IF WT=1 THE SUBROUTINE CALCULATES A BEST WEIGHTED
C                APPROXIMATION.
C      W      -  HAS DIMENSION (N) AND MUST CONTAIN THE N WEIGHTS
C                REQUIRED WHEN WT=1.
C      SC     -  IF SC=1 THE ROWS OF THE COEFFICIENT MATRIX ARE SCALED
C                BY DIVIDING EACH ROW BY ITS ABSOLUTE LARGEST ELEMENT.
C                THE FINAL MATRIX IS RECONVERTED BEFORE EXIT FROM THE
C                SUBROUTINE.
C      DEV    -  IF DEV=1 THE WEIGHTED DEVIATIONS ARE COMPUTED AND
C                STORED IN ORDER IN Q(M+5,J) FOR J=2,3,...,N+1. THE
C                MAGNITUDE OF THE DEVIATION AT THE EXTREME POINTS IS
C                PUT EQUAL TO THE ERROR OF APPROXIMATION.
C      RST    -  IF RST=1 THE SUBROUTINE IS RESTARTED AT STEP1. THE
C                MATRIX Q(1:M+4,2:N+2) PRODUCED BY THE LAST ITERATION
C                MUST BE SUPPLIED, AND THE CALCULATIONS ARE THEN
C                CONTINUED.
C      IT     -  NUMBER OF ITERATIONS COMPLETED AT LAST EXIT. THIS
C                INFORMATION NEED ONLY BE SUPPLIED TO THE SUBROUTINE
C                WHEN RST=1.
C      ALPHA-  NUMBER OF COEFFICIENT VECTORS (LABELLED N+1,...,N+M+1
C                IN THE BASIS. THIS INFORMATION NEED ONLY BE SUPPLIED
C                TO THE SUBROUTINE WHEN RST=1.
C      TOLER-  A POSITIVE TOLERANCE LESS THAN WHICH QUANTITES ARE
C                CONSIDERED TO BE NONPOSITIVE. EMPIRICAL EVIDENCE
C                SUGGESTS THAT PUTTING TOLER=.00001 IS SUITABLE WHEN
C                SEVEN DECIMAL DIGITS ARE AVAILABLE, AND PUTTING
C                TOLER=.00000001 WITH ELEVEN DECIMAL DIGITS.
C                DECREASING THE MAGNITUDE OF TOLER USUALLY DECREASES
C                THE TOTAL NUMBER OF ITERATIONS REQUIRED.
C      QMAX   -  A LARGE POSITIVE NUMBER WHICH THE MINIMUM QUOTIENT
C                (WHICH DETERMINES THE PIVOT) MUST NOT EXCEED. PUTTING
C                QMAX=1/TOLER IS USUALLY SATISFACTORY.
C      IF ANY OF THE INTEGERS POLY,FR,WT,SC,DEV,OR RST IS NOT EQUAL
C                TO ONE THEN NO ACTION IS TAKEN.
C      THE SUBROUTINE TERMINATES FOR THE FOLLOWING REASONS:
C         (1) A BEST APPROXIMATION IS FOUND SINCE ALL THE MARGINAL
C                COSTS ARE GREATER THAN OR EQUAL TO TOLER. THIS IS
```

```
C                    INDICATED BY Q(M+4,N+2)=1. THE ERROR OF APPROXIMATION      89
C                    (LARGEST WEIGHTED ABSOLUTE DEVIATION) IS GIVEN BY          90
C                    Q(),1) AND THE M COEFFICIENTS OF THE BEST                  91
C                    APPROXIMATION ARE CONTAINED IN ORDER BY Q(I,1) FOR         92
C                    I=2,3,...,M+1.                                             93
C                (2) CALCULATION IS STOPPED IF NO SUITABLE PIVOT CAN BE         94
C                    FOUND. THIS IS INDICATED BY Q(M+4,N+2)=2.                  95
C                                                                               96
C     ..........................................................................97
C                                                                               98
      DIMENSION Q(M5,N2),X(N),W(N)                                              99
      INTEGER POLY,FR,WT,SC,DEV,RST,ALPHA,OUT,R,UB,T                           100
C                                                                              101
C     Q(1:M+3,2:N+2) IS CONSTRUCTED FROM Q(1,J) AND Q(2:M+2,2:N+1) BY          102
C     ADDING THE (N+1) TH FUNCTION (THE NEGATIVE SUM OF THE N GIVEN            103
C     FUNCTIONS), A ROW OF 1'S (CORRESPONDING TO THE PRIMAL VECTOR W),         104
C     ROW LABELS AND COLUMN LABELS. THE MATRIX IS WEIGHTED AND/OR SCALED       105
C     AS DIRECTED                                                              106
C                                                                              107
      M=M5-5                                                                   108
      M1=M+1                                                                   109
      M2=M+2                                                                   110
      M3=M+3                                                                   111
      M4=M+4                                                                   112
      N1=N+1                                                                   113
      MA=M-1                                                                   114
      IF(RST.NE.1) GO TO 100                                                   115
      IT=IT-1                                                                  116
      GO TO 125                                                                117
  100 IF(POLY.NE.1) GO TO 102                                                  118
      DO 104 J=2,N1                                                            119
      Q(3,J)=1.0                                                               120
      IF (M.EQ.2) GO TO 104                                                    121
      DO 101 I=4,M1                                                            122
  101 Q(I+1,J)=Q(I,J)*Q(4,J)                                                   123
  104 CONTINUE                                                                 124
  102 IF(WT.NE.1) GO TO 105                                                    125
      DO 103 J=2,N1                                                            126
      A=W(J-1)                                                                 127
      Q(1,J)=Q(1,J)*A                                                          128
      DO 103 I=3,M2                                                            129
  103 Q(I,J)=Q(I,J)*A                                                          130
  105 DO 106 J=2,N1                                                            131
      Q(1,J)=-Q(1,J)                                                           132
```

```
      Q(2,J)=1.0                                                        133
      Q(M3,J)=1.0                                                       134
  106 Q(M4,J)=J-1                                                       135
      DO 110 I=2,M2                                                     136
      Q(I,N2)=N+I-2                                                     137
      DO 110 J=2,N1                                                     138
  110 Q(M3,J)=Q(M3,J)-Q(I,J)                                            139
      Q(M3,N2)=N+M1                                                     140
      Q(2,N2)=0.0                                                       141
      IF(SC.NE.1) GO TO 120                                             142
      A=0.0                                                             143
      DO 111 J=2,N1                                                     144
  111 IF(DABS(Q(1,J)).GT.A) A=DABS(Q(1,J))                              145
      DO 112 J=2,N1                                                     146
  112 Q(1,J)=Q(1,J)/A                                                   147
      Q(1,1)=A                                                          148
      DO 117 I=3,M3                                                     149
      A=0.0                                                             150
      DO 113 J=2,N1                                                     151
  113 IF(DABS(Q(I,J)).GT.A) A=DABS(Q(I,J))                              152
      DO 115 J=2,N1                                                     153
  115 Q(I,J)=Q(I,J)/A                                                   154
  117 Q(I,1)=A                                                          155
  120 IT=-1                                                             156
      K=0                                                               157
      ALPHA=M1                                                          158
C                                                                       159
CC                                                                      160
CC    STEP1: DETERMINES THE PIVOTAL COLUMN                              161
CC                                                                      162
C                                                                       163
  125 A=-TOLER                                                          164
      T=0                                                               165
      R=0                                                               166
      IT=IT+1                                                           167
      IF(IT.GT.MA.OR.K.NE.0.OR.FR.NE.1) GO TO 140                       168
      IF(IT.LE.0) GO TO 130                                             169
      J=X(IT+1)                                                         170
      IF(Q(M4,J+1).EQ.J) GO TO 126                                      171
      IN=2                                                              172
      GO TO 165                                                         173
  126 IN=J+1                                                            174
      GO TO 165                                                         175
  130 IN=2                                                              176
```

```
      X(1)=1.0                                                                  177
      B=(X1+X2)*.5                                                              178
      C=(X2-X1)*.5                                                              179
      DO 135 I=1,MA                                                             180
      D=B+C* COS(3.141593*(1.0-I/N))                                            181
      LB=X(I)+2                                                                 182
      DO 133 J=LB,N                                                             183
      IF(X(J)-D.LE.0.0) GO TO 133                                               184
      IF(X(J)+X(J-1)-2.0*D.LT.0.0) GO TO 132                                    185
      X(I+1)=J-1                                                                186
      GO TO 135                                                                 187
  132 X(I+1)=J                                                                  188
      GO TO 135                                                                 189
  133 CONTINUE                                                                  190
      C=(N-1)/M                                                                 191
      DO 134 J=1,MA                                                             192
      P=1.0+J*C                                                                 193
  134 X(J+1)= AINT(P+.5)                                                        194
  135 CONTINUE                                                                  195
      GO TO 165                                                                 196
  140 IF(ALPHA.NE.1.OR.K.NE.0) GO TO 155                                        197
      T=N                                                                       198
  141 IF(Q(M4,T+1).GT.N) GO TO 151                                              199
      IF(Q(1,T+1).LE.0.0) GO TO 143                                             200
      Q(2,T+1)=2.0-Q(2,T+1)                                                     201
      Q(1,T+1)=-Q(1,T+1)                                                        202
      DO 142 I=3,M4                                                             203
  142 Q(I,T+1)=-Q(I,T+1)                                                        204
  143 P=Q(2,T+1)                                                                205
      DO 145 I=3,M3                                                             206
  145 IF(Q(I,T+1).GE.TOLER.AND.Q(I,N2).LE.N) P=P+2.0*Q(I,T+1)                   207
      IF(P.LE.TOLER) GO TO 151                                                  208
      DO 150 I=3,M3                                                             209
      IF(Q(I,T+1).LE.TOLER.OR.Q(I,N2).GT.N) GO TO 150                           210
      Q(I,N2)=-Q(I,N2)                                                          211
      DO 147 J=2,N1                                                             212
      Q(2,J)=Q(2,J)+2.0*Q(I,J)                                                  213
  147 Q(I,J)=-Q(I,J)                                                            214
  150 CONTINUE                                                                  215
      IN=T+1                                                                    216
      OUT=2                                                                     217
      GO TO 190                                                                 218
  151 T=T-1                                                                     219
      IF(T.NE.0) GO TO 141                                                      220
```

```
  155 IF(K.EQ.0) GO TO 156                                                    221
      C=2.0*Q(1,2)                                                            222
      GO TO 157                                                               223
  156 C=0.0                                                                   224
  157 IF(R.EQ.0) GO TO 158                                                    225
      LB=2                                                                    226
      UB=M3-ALPHA                                                             227
      GO TO 159                                                               228
  158 LB=M4-ALPHA                                                             229
      UB=N1                                                                   230
  159 DO 161 J=LB,UB                                                          231
      Z=Q(1,J)                                                                232
      IF(Q(M4,J).EQ.0.0.OR.Q(M4,J).GE.N1) GO TO 160                           233
      IF(-Z+C.GE.A) GO TO 160                                                 234
      IN=J                                                                    235
      T=2                                                                     236
      A=-Z+C                                                                  237
      GO TO 161                                                               238
  160 IF(Z.GE.A) GO TO 161                                                    239
      IN=J                                                                    240
      T=1                                                                     241
      A=Z                                                                     242
  161 CONTINUE                                                                243
      IF(T.NE.0) GO TO 165                                                    244
      IF(R.NE.0) GO TO 163                                                    245
      R=1                                                                     246
      GO TO 157                                                               247
  163 Q(M4,N2)=1.0                                                            248
      IF(SC.EQ.1) GO TO 212                                                   249
      GO TO 225                                                               250
C                                                                             251
CC                                                                            252
CC    STEP2: DETERMINES THE PIVOT                                             253
CC                                                                            254
C                                                                             255
  165 B=QMAX                                                                  256
      TB=-1.0                                                                 257
      P=0                                                                     258
      IF(ALPHA.NE.1.OR.Q(M4,IN).LE.N.OR.Q(2,N2).LE.N) GO TO 167               259
      OUT=2                                                                   260
      GO TO 190                                                               261
  167 IF(T.NE.2) GO TO 176                                                    262
      Q(M4,IN)=-Q(M4,IN)                                                      263
      IF(K.EQ.0) GO TO 168                                                    264
```

```
      A=2.0*Q(2,2)                                                                  265
      GO TO 169                                                                     266
  168 A=2.0                                                                         267
  169 Q(2,IN)=-Q(2,IN)+A                                                            268
      IF(K.EQ.0) GO TO 170                                                          269
      A=2.0*Q(1,2)                                                                  270
      GO TO 171                                                                     271
  170 A=0.0                                                                         272
  171 Q(1,IN)=-Q(1,IN)+A                                                            273
      DO 175 I=3,M3                                                                 274
      IF(K.EQ.0) GO TO 172                                                          275
      A=2.0*Q(I,2)                                                                  276
      GO TO 175                                                                     277
  172 A=0.0                                                                         278
  175 Q(I,IN)=-Q(I,IN)+A                                                            279
  176 DO 185 I=2,M3                                                                 280
      D=Q(I,IN)                                                                     281
      IF(K.EQ.1) GO TO 180                                                          282
      IF(I.NE.2) GO TO 177                                                          283
      TB=D                                                                          284
      GO TO 185                                                                     285
  177 IF(IT.GT.MA.OR.R.NE.0) GO TO 179                                              286
      IF(Q(I,N2).GT.N) GO TO 178                                                    287
      D=0.0                                                                         288
      GO TO 179                                                                     289
  178 D=DABS(D)                                                                     290
  179 IF(D.LE.TOLER) GO TO 185                                                      291
      D=-D                                                                          292
      GO TO 182                                                                     293
  180 IF(D.LE.TOLER) GO TO 185                                                      294
      D=Q(I,2)/D                                                                    295
  182 IF(D.GE.B) GO TO 185                                                          296
      B=D                                                                           297
      OUT=I                                                                         298
  185 CONTINUE                                                                      299
      IF(B.LT.QMAX.AND.(ALPHA.NE.1.OR.Q(M4,IN).GT.N.OR.Q(OUT,N2).LE.N))             300
     1GO TO 190                                                                     301
      IF(IT.GT.MA.OR.R.NE.0) GO TO 187                                              302
      R=1                                                                           303
      GO TO 176                                                                     304
  187 IF(TB.GT.TOLER) GO TO 188                                                     305
      Q(M4,N2)=2.0                                                                  306
      IF(SC.EQ.1) GO TO 212                                                         307
      GO TO 225                                                                     308
```

```
  188 OUT=2                                                                      309
C                                                                                310
CC                                                                               311
C     STEP3: PERFORMS ONE SIMPLEX ALGORITHM                                      312
CC                                                                               313
C                                                                                314
  190 P=Q(OUT,IN)                                                                315
      R=2                                                                        316
      IF(Q(M4,IN).GT.N) ALPHA=ALPHA+1                                            317
      IF(Q(OUT,N2).GT.N) ALPHA=ALPHA-1                                           318
      DO 196 I=1,M3                                                              319
      IF(I.EQ.OUT) GO TO 196                                                     320
      D=Q(I,IN)/P                                                                321
      DO 195 J=2,N1                                                              322
      IF(J.EQ.IN) GO TO 193                                                      323
      Q(I,J)=Q(I,J)-D*Q(OUT,J)                                                   324
      GO TO 195                                                                  325
  193 Q(I,J)=-D                                                                  326
  195 CONTINUE                                                                   327
  196 CONTINUE                                                                   328
      DO 198 J=2,N1                                                              329
      IF(J.EQ.IN) GO TO 197                                                      330
      Q(OUT,J)=Q(OUT,J)/P                                                        331
      GO TO 198                                                                  332
  197 Q(OUT,J)=1.0/P                                                             333
  198 CONTINUE                                                                   334
      I=Q(OUT,N2)                                                                335
      Q(OUT,N2)=Q(M4,IN)                                                         336
      Q(M4,IN)=I                                                                 337
      T=M3-ALPHA                                                                 338
      IF(Q(OUT,N2).GT.N.OR.Q(M4,IN).LE.N.OR.IN.EQ.T) GO TO 201                   339
      DO 200 I=1,M3                                                              340
      Q(M5,I)=Q(I,IN)                                                            341
      Q(I,IN)=Q(I,T)                                                             342
  200 Q(I,T)=Q(M5,I)                                                             343
      A=Q(M4,IN)                                                                 344
      Q(M4,IN)=Q(M4,T)                                                           345
      Q(M4,T)=A                                                                  346
  201 IF(OUT.NE.2.OR.K.NE.0) GO TO 125                                           347
      K=1                                                                        348
      DO 202 I=1,M3                                                              349
      Q(M5,I)=Q(I,2)                                                             350
      Q(I,2)=Q(I,IN)                                                             351
  202 Q(I,IN)=Q(M5,I)                                                            352
```

```
      A=Q(M4,2)                                                          353
      Q(M4,2)=Q(M4,IN)                                                   354
      Q(M4,IN)=A                                                         355
      DO 205 I=2,M3                                                      356
  205 IF(Q(I,N2).GT.N) R=I                                               357
      DO 207 J=2,N2                                                      358
      Q(M5,J)=Q(2,J)                                                     359
      Q(2,J)=Q(R,J)                                                      360
  207 Q(R,J)=Q(M5,J)                                                     361
      GO TO 125                                                          362
  212 IF(RST.EQ.1) GO TO 225                                             363
C                                                                        364
CC    IF THE DATA WAS SCALED IT IS NOW RECONVERTED                       365
C                                                                        366
      IF(SC.NE.1) GO TO 225                                              367
      A=Q(1,1)                                                           368
      T=C                                                                369
      DO 215 J=2,N1                                                      370
  215 Q(1,J)=Q(1,J)*A                                                    371
      DO 218 I=2,M3                                                      372
      IF(Q(I,N2).LE.N) GO TO 218                                         373
      A=Q(Q(I,N2)-N+2,1)                                                 374
      DO 216 J=2,N1                                                      375
  216 Q(I,J)=Q(I,J)*A                                                    376
      T=T+1                                                              377
      IF(T.EQ.ALPHA) GO TO 220                                           378
  218 CONTINUE                                                           379
  220 DO 222 J=2,N1                                                      380
      IF(Q(M4,J).LE.N) GO TO 222                                         381
      A=Q(Q(M4,J)-N+2,1)                                                 382
      DO 221 I=1,M3                                                      383
  221 Q(I,J)=Q(I,J)/A                                                    384
  222 CONTINUE                                                           385
  225 A=0.0                                                              386
      T=N+M1                                                             387
      DO 227 J=2,N1                                                      388
  227 IF(Q(M4,J).EQ.T) A=Q(1,J)                                          389
      DO 228 I=2,M1                                                      390
  228 Q(I,1)=-A                                                          391
      DO 230 J=2,N1                                                      392
  230 IF(Q(M4,J).GE.N) Q(Q(M4,J)-N+1,1)=Q(1,J)-A                         393
      IF(K.EQ.0) GO TO 232                                               394
      Q(1,1)=Q(1,2)                                                      395
      GO TO 233                                                          396
```

```
232 Q(1,1)=TOLER                                                              397
233 IF(DEV.NE.1) GO TO 240                                                    398
    A=Q(1,1)                                                                  399
    DO 235 I=3,M3                                                             400
    R=Q(I,N2)                                                                 401
235 IF(R.LE.N) Q(M5,IABS(R)+1)=ISIGN(1,R)*A                                   402
    DO 237 J=3,N1                                                             403
    R=Q(M4,J)                                                                 404
237 IF(R.LE.N) Q(M5,IABS(R)+1)=ISIGN(1,R)*(A-Q(1,J))                          405
240 RETURN                                                                    406
    END                                                                       407
```

VITA

Surname: HUNT Given Names: CHARLES RICHARD

Place of Birth: GUELPH, ONTARIO Date of Birth: FEBRUARY 9, 1944

Educational Institutions Attended, with Dates of Entering and Leaving:

VICTORIA COLLEGE	1961 to 1963
UNIVERSITY OF VICTORIA	1965 to 1968
	____ to ____
	____ to ____

Degrees, Diplomas, Etc., Awarded, with Dates and Names of Institutions:

B.Sc.	1968	UNIVERSITY OF VICTORIA

Honors and Awards:

University of Victoria Graduate Scholarship, 1968/1969 and 1969/1970

Publications:

BARRODALE, I., ROBERTS, F.D.K., and HUNT, C.R. (1970). Computing Best ℓ_p Approximations by Functions Nonlinear in One Parameter. The Computer Journal, to appear.

THE UNIVERSITY OF VICTORIA LIBRARY

MANUSCRIPT THESIS

AUTHORITY TO DISTRIBUTE

AUTHOR: This dissertation may be lent or microfilm copies made available

(a) Without restriction [redacted]

(b) With the restriction that, for a period of five years (until _____) the written approval of the following is required:

(1) The Chairman, School of Graduate Studies _____

(2) The Author _____

(3) both the Chairman, School of Graduate Studies, and the Author _____

BORROWERS. The borrower undertakes, by signing below, to give proper credit for any use made of the dissertation, and to obtain the consent of the author if it is proposed to make extensive quotations, or to reproduce the dissertation in whole or in part.

Signature of Borrower	Address	Date