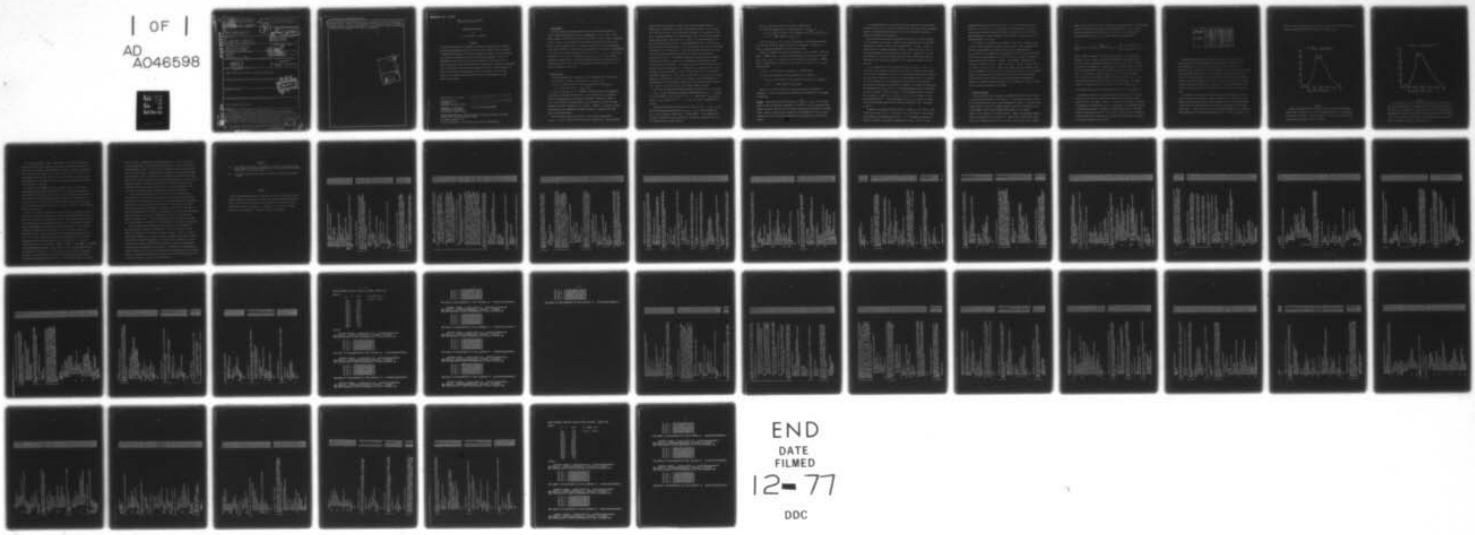
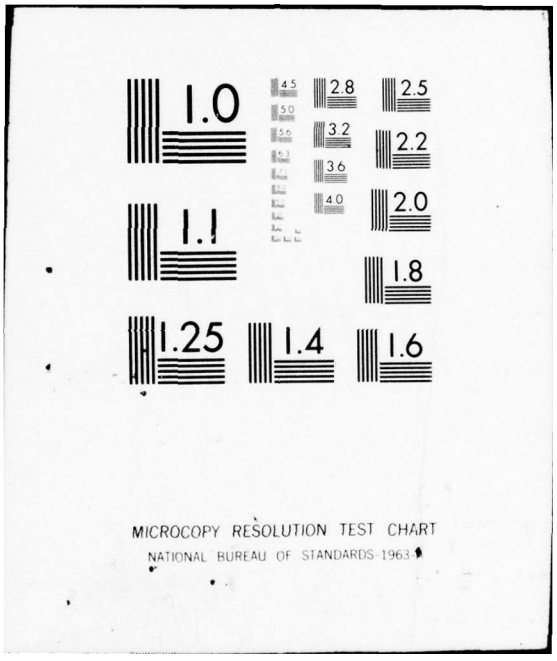
AD-A046 598 COLORADO STATE UNIV FORT COLLINS DEPT OF MATHEMATICS F/G 12/1
ADAPTIVE CURVE FITTING.(U)
JUL 77 J A HULL, G D TAYLOR AFOSR-76-2878
UNCLASSIFIED AFOSR-TR-77-1257 NL

| OF |
AD
A046598



END
DATE
FILMED
12-77
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A046598

DDC FILE COPY

(19) REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER (18) AFOSR-TR- 77-1257	2. GOVT ACCESSION NO. (2)
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) (6) ADAPTIVE CURVE FITTING •	5. TYPE OF REPORT & PERIOD COVERED (9) Interim rept.
	6. PERFORMING ORG. REPORT NUMBER (15)
7. AUTHOR(s) (10) J. A. Hull and G. D. Taylor	8. CONTRACT OR GRANT NUMBER(s) AFOSR-76-2878
9. PERFORMING ORGANIZATION NAME AND ADDRESS Colorado State University Department of Mathematics Fort Collins, CO 80523	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (16) 61102F (17) 2304/A3
11. CONTROLLING OFFICE NAME AND ADDRESS AFOSR/NM Bldg. 410 Bolling AFB, D.C. 20332	12. REPORT DATE (11) July 1977
	13. NUMBER OF PAGES 45
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) (12) 47 p.	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

DDC RECEIVED NOV 21 1977 F

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Curve fitting, data fitting

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In this paper ~~we~~ presents an algorithm for adaptively computing smooth piecewise polynomial approximations which uses either the best uniform or best (discrete) L_2 approximation operator as its local approximation operator. ~~do not require any~~ No knowledge of derivatives of the function being approximated is required. Due to the approximation properties of the respective operators, ~~we have found~~ that the algorithm using best uniform approximations is particularly suited for approximating precise mathematical functions and the algorithm using best L_2

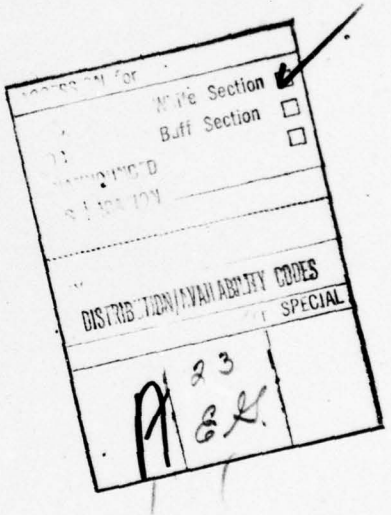
next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

407344.

20 approximations is particularly suited for approximating data with significant levels of noise. Finally, this algorithm can be used with classes of approximating functions other than polynomials. Fortran codes for these two algorithms are given in the appendix at the end of this paper.



ADAPTIVE CURVE FITTING

by

J. A. Hull[1] and G. D. Taylor[2]

ABSTRACT

In this paper we present an algorithm for adaptively computing smooth piecewise polynomial approximations which uses either the best uniform or best (discrete) L^2 approximation operator as its local approximation operator. We do not require any knowledge of derivatives of the function being approximated. Due to the approximation properties of the respective operators, we have found that the algorithm using best uniform approximations is particularly suited for approximating precise mathematical functions and the algorithm using best L^2 approximations is particularly suited for approximating data with significant levels of noise. Finally, this algorithm can be used with classes of approximating functions other than polynomials.

Send proofs to G. D. Taylor

[1]Applied Technology
645 Almanor
Sunnyvale, California 94086

[2]Department of Mathematics
Colorado State University
Fort Collins, Colorado 80523

Research sponsored by the Air Force Office of Scientific Research, Air Force Systems, USAF, under grant number 76-2878.

CR category numbers: 5.13.

Key words and phrases: curve fitting, data fitting, approximation.

I. Introduction

Let X be a finite set of real points and let f be a function defined on X (given in tabular form) which we desire to approximate. Let $a = \min\{x: x \in X\}$ and $b = \max\{x: x \in X\}$. and for any function g defined on X define $\|g\|_X = \max\{|g(x)|: x \in X\}$. Finally, let TOL, SMTH and N be parameters supplied by the user where TOL is a positive number, SMTH is a nonnegative integer and N-1 is an integer greater than or equal to SMTH. In this setting, our algorithm will calculate an approximation, p, to f and a set of points $\{x_i\}_{i=0}^{k} \subset X$ with $a = x_0 < x_1 < \ldots < x_k = b$ such that p restricted to $[x_i, x_{i+1}]$ is a polynomial $p_i \in \Pi_{N-1} = \{q: q \text{ is a real algebraic polynomial of degree } \leq N-1\}$, p has SMTH continuous derivatives and $\|f-p\|_X \leq TOL$. In what follows, we shall use the notation $\|g\|_S$ to denote $\max\{|g(t)|: t \in S\}$ for all g defined on S and $S \subset X$.

II. The Algorithm

The algorithm begins by choosing $\tilde{x}_1$ to be the largest point in X such that

(1) $[a, \tilde{x}_1] \cap X$ contains at least max(2, N + 1) points, and

(2) If p_1 is the best (uniform or L^2) approximation to f from Π_{N-1} on $S_1 = [a, \tilde{x}_1] \cap X$ then $\|f - p_1\|_{S_1} \leq TOL$.

If $\tilde{x}_1 = b$, then since p_1 is a piecewise polynomial meeting our requirements, we successfully terminate the algorithm. If no such $\tilde{x}_1$ exists, the algorithm fails and an appropriate error message is generated. Otherwise, if SMTH = 0 (i.e., we only require the approximation to be continuous) we choose the right endpoint of the first subinterval, x_1, to be $\tilde{x}_1$. If SMTH > 0, in order to add to the stability of the algorithm we in general choose x_1 by "backing off" from $\tilde{x}_1$ in the following manner.

We first examine the error curve $f(x) - p_1(x)$ and find those points $\xi_1, \xi_2, \ldots, \xi_\ell$ in $(a, \tilde{x}_1] \cap X$ at which relative extrema occur. (Note that when

using a Remes-like algorithm to compute best uniform approximations with interpolatory constraints, these points are readily available). We will choose one of the ξ_ν's to be x_1. The motivation for choosing x_1 in this manner is that in the continuous setting, if f is differentiable and ξ is an (interior) relative extreme point of $f(x) - p_1(x)$ then $f'(\xi) - p_1'(\xi) = 0$ so that the derivative of p_1 at ξ would match that of f at ξ. This guarantees that when we smoothly join the next piece of the approximation to p_1 at ξ, this next piece will closely follow the direction of f at least near ξ. If we merely joined the second piece to the first at $\tilde{x}_1$ no such guarantee can be made and, in fact, severe oscillatory problems tend to set in. Our numerical experience indicates that the procedure of backing off from $\tilde{x}_1$ to a smaller x_1 contributes very significantly toward the stability of the algorithm. To continue, let $\tilde{f}'(\xi_\nu)$ be the derivative of the centered quadratic interpolation of f evaluated at ξ_ν. We then choose x_1 to be the largest ξ_ν such that $|\tilde{f}'(\xi_\nu) - p_1'(\xi_\nu)| < \text{EPS}$, where EPS is a user-definable prescribed tolerance, or, if there does not exist such a ξ_ν, then we let x_1 be the largest ξ_ν at which $|\tilde{f}'(\xi_\nu) - p_1'(\xi_\nu)|$ is a minimum. (In our implementation of the algorithm, we do not in general consider all of the relative extreme points of $f(x) - p_1(x)$ in $(a, \tilde{x}_1] \cap X$, but only the largest N-SMTH-1 of them.)

We continue the algorithm by finding successive intervals $[x_1, x_2]$, $[x_2, x_3]$, ..., $[x_{m-1}, b]$, and corresponding polynomial approximations $p_2, p_3, \ldots, p_m \in \Pi_{N-1}$ to f so that $p_{\nu-1}^{(j)}(x_{\nu-1}) = p_\nu^{(j)}(x_{\nu-1})$ for $j = 0, 1, \ldots, \text{SMTH}$ and $\|f - p_\nu\|_{S_\nu} \leq \text{TOL}$, where $S_\nu = [x_{\nu-1}, x_\nu] \cap X$ for $\nu = 2, \ldots, m$, $x_m = b$. This is accomplished as follows:

Suppose we have found the subintervals $[a, x_1]$, $[x_1, x_2]$, ..., $[x_{i-2}, x_{i-1}]$, and the corresponding approximations $p_1, p_2, \ldots, p_{i-1}$. Assume further that $[x_{i-1}, b] \subset X$ contains at least max(2, N-SMTH) points. We now determine an x_i and a p_i meeting the above requirements. We begin by choosing $\tilde{x}_i \in X$ to be the largest point in X which satisfies

(1) $[x_{i-1}, \tilde{x}_i] \cap X$ contains at least max(2, N-SMTH) points,

(2) If p_i is the appropriate best approximation to f from Π_{N-1} on $S_i = [x_{i-1}, \tilde{x}_i] \cap X$ subject to the constraint that $p_i^{(j)}(x_{i-1}) = p_{i-1}^{(j)}(x_{i-1})$, $j = 0, 1, \ldots,$ SMTH, then $\|f - p_i\|_{S_i} \leq$ TOL.

If $\tilde{x}_i = b$, we set $x_i = \tilde{x}_i = b$, and the algorithm is successfully terminated. If no such $\tilde{x}_i$ exists, the algorithm fails and is terminated. Otherwise, we choose x_i completely analogous to our choice of x_1 above.

Finally, we consider the special case where $[x_{i-1}, b] \cap X$ contains fewer than max(2, N-SMTH) points. In this case we replace x_{i-1} with some $\hat{x}_{i-1} \in X$, where $x_{i-2} < \hat{x}_{i-1} < x_{i-1} < b$, so that $[\hat{x}_{i-1}, b] \cap X$ contains at least max(2, N-SMTH) points. Specifically, we choose $\hat{x}_{i-1}$ to be a point in X closest to $(b - x_{i-2})/2$ which satisfies

(1) $[\hat{x}_{i-1}, b] \cap X$ contains at least max(2, N-SMTH) points.

(2) If p_i is the appropriate best approximation to f from Π_{N-1} on $\hat{S} = [\hat{x}_{i-1}, b] \cap X$ subject to the constraint that $p_i^{(j)}(\hat{x}_{i-1}) = p_{i-1}^{(j)}(\hat{x}_{i-1})$, $j = 0, 1, \ldots,$ SMTH, then $\|f - p_i\|_{\hat{S}} \leq$ TOL.

Again, if we can find such an $\hat{x}_{i-1}$ then the algorithm is successfully terminated. If not, the algorithm is terminated and an appropriate error message is generated.

Remarks. Since any $p_i(x)$ can be written $p_i(x) = \sum_{j=0}^{N-1} a_j (x - x_{i-1})^j$, it is a simple matter to meet the smoothness constraints. In fact, it is easy to generalize this scheme in order to force p_i to satisfy Hermite interpolatory constraints at several points. Hence, it would be easy to modify the above algorithm so that we could interpolate the function being approximated and its derivatives at the knots if desired.

In our implementation of this algorithm we have modularized the approximation routine so that we may use a Remes-type algorithm for computing uniform approximation subject to interpolating constraints [1], and we use Householder transforms to compute the discrete L^2 approximations.

Note that when using uniform approximations computed by the Remes algorithm it is in general not necessary to compute the best approximation on a particular subinterval in order to conclude that the subinterval is too long. Indeed, at each iteration of the Remes algorithm we obtain a lower bound for the error of the best approximation on this particular subinterval. Consequently, if during some iteration of the Remes algorithm this lower bound is greater than TOL, we may conclude that the present subinterval is too large and terminate the Remes algorithm.

In our implementation of this algorithm, the $\tilde{x}_i$ are chosen as follows. At each step of this iterative procedure we will let $\tilde{a}$ be the current largest point in X such that requirements (1) and (2) of the appropriate algorithm are satisfied on $[x_{i-1}, \tilde{a}] \cap X$, and we will let $\tilde{b}$ be the current smallest point in X such that $\tilde{b} > \tilde{a}$ and requirement (2) of the appropriate algorithm fails to be satisfied. We initialize this process by computing (or attempting to compute) the appropriate best approximation on $[x_{i-1}, b] \cap X$. If this approximation satisfies requirement (2), then we set $\tilde{x}_i = b$ and we are done. If the approximation fails to satisfy (2), we set $\tilde{b} = b$. Next, let $t = \min\{x \in X: [x_{i-1}, x] \cap X$ contains at least max(2, N-SMTH) points$\}$. If the approximation on $[x_{i-1}, t] \cap X$ fails to satisfy (2) then the algorithm cannot meet the desired accuracy and fails. Otherwise, we set $\tilde{a} = t$.

In general, we proceed as follows. We let $t = \inf\{x \in X: (\tilde{b} - \tilde{a})/2 \leq x < \tilde{b}\}$. If this set is empty, we set $t = \sup\{x \in X: \tilde{a} \leq x \leq (\tilde{b} - \tilde{a})/2\}$. If $t = \tilde{a}$ then this procedure has converged and we set $\tilde{x}_i = t = \tilde{a}$. Otherwise, we compute (or attempt to compute) the appropriate approximation on $[x_{i-1}, t] \cap X$. If this

approximation satisfies (2) then we set $\tilde{a} = t$. If this approximation fails to satisfy (2) then we set $\tilde{b} = t$. We continue this process until $\tilde{b} - \tilde{a}$ is less than some user definable prescribed tolerance, at which point we accept $\tilde{a}$ as a good approximation to $\tilde{x}_i$ and terminate this procedure. We compute the $\hat{x}_i$ in a manner analogous to the above.

In an attempt to accelerate the convergence of the above scheme when using the best uniform approximation operator, we have tried to take advantage of the fact that corresponding to each $\tilde{a}$ we know the error of approximation on $[x_{i-1}, \tilde{a}] \cap X$, call it SMLERR, and corresponding to each $\tilde{b}$ we know a lower bound for the error of approximation on $[x_{i-1}, \tilde{b}] \cap X$, call it BIGERR. We change the above scheme by first setting $\alpha = (\text{BIGERR} - \text{TOL})/(\text{BIGERR} - \text{SMLERR})$ and then setting $t = \inf\{x \in X: \alpha\tilde{a} + (1 - \alpha)\tilde{b} \leq x < \tilde{b}\}$ or, if this set is empty, we set $t = \sup\{x \in X: \tilde{a} \leq x \leq \alpha\tilde{a} + (1 - \alpha)\tilde{b}\}$ in the general iteration described above. Hence, if SMLERR is very close to TOL, t will be chosen close to $\tilde{a}$. Our numerical experience has shown that this procedure only works well when approximating uniformly smooth functions and that this algorithm cannot be significantly improved by allowing the Remes algorithm to run to completion in order to obtain the true error of approximation for BIGERR.

III. Numerical Results

This algorithm has been implemented as FORTRAN programs and has been tested on Colorado State University's CDC CYBER 172. In the appendix we give a listing of the algorithm using the best uniform approximation operator. By using a least squares routine in place of the REMES subroutine (and those subroutines called only by REMES: SOLVE, DIVIDF, ZEROFD, BPOLY, TRANS) the least squares version of this algorithm is readily obtained. Indeed, we essentially replaced these six subroutines by four subroutines: ASET, HOUSE, TOLCHK and FIX. ASET sets up the overdetermined system to be solved in the L^2 sense. Here a certain

amount of care needs to be taken to insure that the approximation on each succeeding interval has the desired smoothness at the common endpoints. Thus, if we are currently considering the interval $[x_{i-1}, \tilde{x}_i]$ and p_{i-1} is the approximation that was accepted for $[x_{i-2}, x_{i-1}]$ then we form the following system:

$$\sum_{\nu=k+1}^{N-1} c_{\nu+1}(x-x_{i-1})^{\nu} = f(x) - \sum_{\nu=0}^{k} \frac{p_{i-1}^{(\nu)}(x_{i-1})}{\nu!}(x-x_{i-1})^{\nu} \text{ for } \begin{cases} x \in [x_{i-1}, \tilde{x}_i] \cap X \text{ if } k = -1 \\ x \in (x_{i-1}, \tilde{x}_i] \cap X \text{ if } k > -1 \end{cases}$$

where k = SMTH = smoothness desired at x_{i-1} and $\sum_{\nu=0}^{-1}$ is to be replaced by 0 whenever it occurs. HOUSE is any good overdetermined least squares solver. We use one based on Householder transforms (a simple version since our system will always be nondegenerate). TOLCHK compares the maximum absolute value of f minus the approximation returned by HOUSE with TOL and also finds the set of N - k - 1 last "extreme" values in $[x_{i-1}, \tilde{x}_i]$. That is, points at which $|f(x) - p(x)|$ is a local maximum relative to points on each side of x. This is for the backing off from $\tilde{x}_i$ procedure to increase stability. FIX simply converts the coefficients $\{c_\nu\}$ of a polynomial of the form $\sum_{\nu=1}^{N} c_\nu (x-x_{i-1})^{\nu-1}$ into coefficients $\{b_\nu\}$ for the same polynomial written in the form $\sum_{\nu=1}^{N} b_\nu x^{\nu-1}$.

As examples, using the best uniform operator the functions $e^{|x|}$ on $[-1,1]$, $|\sin(x)|$ on $[-\pi,\pi]$ and $\sqrt{x}$ on $[0,2]$ were approximated on 200 equally spaced points with TOL = .01, SMTH = 2, and N = 6. Each of these examples is difficult to approximate by polynomials near x = 0. Consequently, the algorithm's ability to automatically decrease the length of the subintervals near x = 0 and then recover by lengthening them for x > 0 is tested. Below is a table listing knot locations (subinterval endpoints) and the CPU time in seconds used to compute the piecewise polynomial approximation.

	$e^{\|x\|}$	$\|\sin(x)\|$	$\sqrt{x}$
CPU TIME	.979	1.095	.876
Knot Locations	-1.0	-3.14	0.0
	- .347	- .268	.0804
	- .0653	.0474	.231
	.0452	.205	.352
	.196	.458	.814
	.437	.868	2.0
	1.0	1.53	-----
	-----	3.14	-----

Because uniform approximations weight each data point equally, they are particularly suited for approximating precise mathematical functions or for approximating data with noise levels that are very small relative to the desired accuracy. However, when these algorithms are used to approximate functions containing considerable levels of noise, in general the approximations tend to follow the noise patterns more than is desirable so that near abrupt changes in the data, the approximations tend to begin to oscillate and generally it requires several subintervals to dampen these oscillations. (Some counterexamples from engineering applications have been found which show that this is not always the case.)

For example, we were given some experimental data involving the release of bitumen from oil shale heated to a constant temperature as a function of time. Because relatively few data points were available, we filled in the gaps between the data points by linear interpolation so that we approximated on 200 equally spaced points. Figure 1 is a plot of the data being approximated and the approximation generated using uniform approximation with N = 6, SMTH = 2, and TOL = 2.5.

Notice that as oscillations appear in the data between times of 22 and 34 minutes, greatly exaggerated oscillations appear in the approximation.

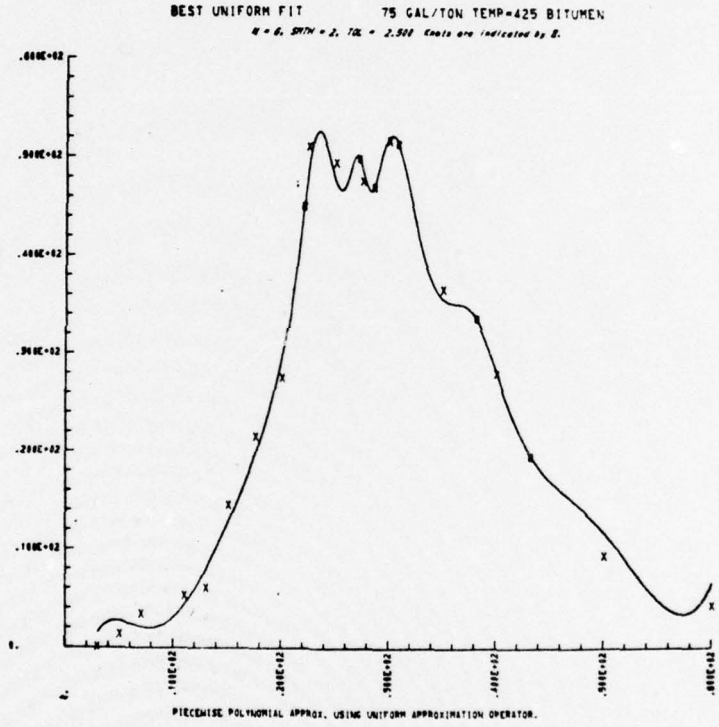


Figure 1

Because L^2 approximations minimize the effects of normally distributed random noise, we expect the L^2 local approximation operator to be prefereable for this type of example. Indeed, in Figure 2 we present a plot of the same data using the algorithm with the L^2 operator.

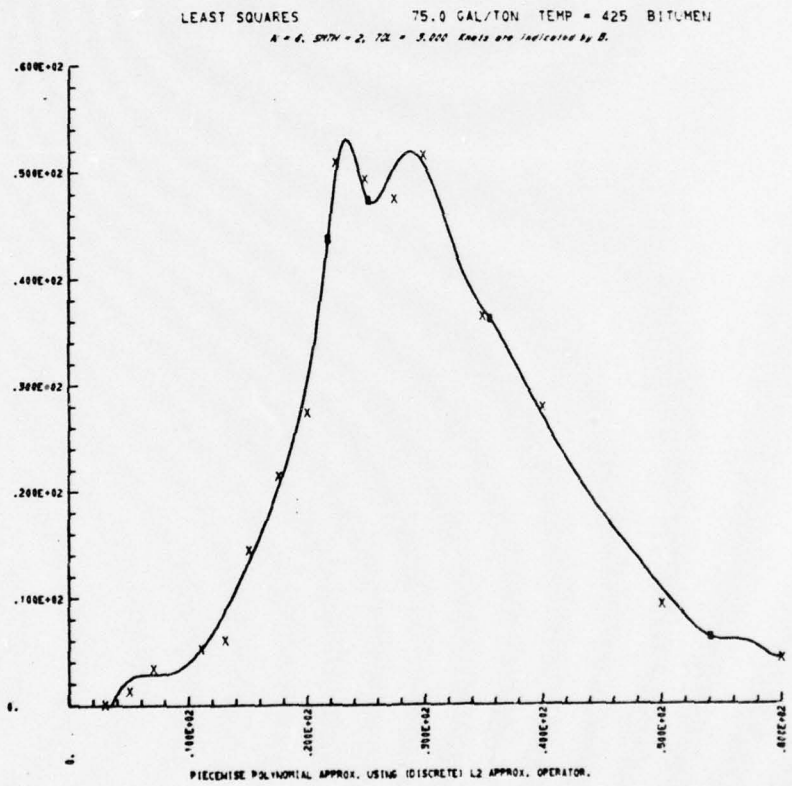


Figure 2

If oscillations do tend to occur in the approximation, often one can improve the approximation by approximating on more (densely packed) points. In particular, by adding extra points in regions where the function being approximated is unstable, and removing them in regions where the function is "nice", for the reason outlined above, the approximations tend to improve, with only moderately increased computational costs.

For some applications it might be preferable to be able to choose ahead of time a particular knot location and to specify interpolatory constraints at these knots. For example, when approximating $|\sin(x)|$ it might be advantageous to force a knot to be located at $x = 0$ and to require $p(0) = 0$, $p'(0) = -1$ as x approaches 0 from the left, and $p'(0) = +1$ as x approaches 0 from the right. The modifications to the algorithm as given necessary to accomplish this would not be extensive or difficult.

It should also be remarked that this algorithm can be used with linear approximating families other than polynomials. For example, any generalized Haar subspace of high enough order (to allow Hermite interpolating constraints) can be used with the best uniform approximation operator. In addition, one could force not only knots to occur at various prescribed points but could also vary the smoothness at these knots as well as change the form of the approximating families at these knots.

Recently, John Rice has described an adaptive piecewise polynomial algorithm which uses local Hermite interpolation instead of uniform approximation on each subinterval to obtain each polynomial piece [2]. His algorithm requires that the first SMTH derivatives or approximate derivatives of f be available in order to compute approximations which have SMTH continuous derivatives. Rice's adaptive strategy also differs from ours; he uses a bisection strategy which can be described as follows. First, compute the Hermite interpolating polynomial to f on [a, b]. That is, compute the polynomial which interpolates f and its first SMTH derivatives at both endpoints, as well as interpolating f at k evenly distributed points in (a, b), where k = DEGREE - 2·SMTH - 1, and DEGREE is the degree of the approximating polynomial. Next, measure the error of approximation using any preselected L^p norm ($p \geq 1$). If the error is less that TOL, the algorithm terminates, otherwise bisect [a, b] into two subintervals and check to see if the Hermite interpolating polynomial on [a, (a + b)/2] differs from f in norm by

no more than TOL. Meanwhile, place the subinterval [(a + b)/2, b] in a "stack" to be processed later. Continue bisecting subintervals, placing the right half of the interval on the top of the stack to be processed later and check the left half to see if the error of approximation by the Hermite interpolating polynomial is less than TOL. When a subinterval and its associated approximations are found which meet the desired tolerance, we accept this approximation as a "piece" of the piecewise polynomial approximation and continue the algorithm by removing the subinterval which is currently at the top of the stack and repeating the above on it unless the stack is empty, at which point terminate the algorithm.

Rice's routine requires the value of the function and its derivatives at very many points in [a, b], even though it may not actually access these values. His algorithm is particularly suited, then, for approximating precise mathematical functions for which this data is readily available. Our algorithm seems more suitable for obtaining approximations of functions given in tabular form (as in many engineering applications) since our algorithm does not require any information about derivatives, and no special configuation of the data is needed for our adaptive strategy to work. See the forthcoming paper by Andrews, Hull and Taylor for further comments on suitability of this algorithm for this type of application.

By using interpolatory constraints on both endpoints of each subinterval, and interpolating the values of the derivative of f at these points, we could modify our algorithm to use a bisection adaptive strategy. However, in order to take greatest advantage of our use of the best approximation operators as opposed to the Hermite interpolation operator, we felt that it was preferable to use the majority of our degrees of freedom inside subintervals instead of using them up on interpolation requirements at the endpoints. Essentially, in our algorithm we traded a somewhat faster run time (since computing best approximations is very slow relative to computing interpolatory polynomials) for generally fewer knots and the freedom from needing to know derivative information.

REFERENCES

[1] H.L. Loeb, D.G. Moursund, L.L. Schumaker, G.D.Taylor, Uniform Generalized Weight Function Polynomial Approximation with Interpolation, SIAM J. Numer. Anal., Vol. 6, No. 2, June, 1969.

[2] J.R. Rice, An Algorithm for Adaptive Piecewise Polynomial Approximation, to appear.

APPENDIX

Here we give listings for the two algorithms described in this paper with the uniform approximation listing first and the least squares approximation listing second. In both cases we include a driver and sample output where the output corresponds to figure 1 and figure 2, respectively. Also, we list the input of these two runs between the listing of the code and the output.

```
      PROGRAM DRIVER(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)                    DRIVER10
      LOGICAL EPROR                                                            DRIVER20
      COMMON XTABLE(500),FTABLE(500),DUMMY(1382)                               DRIVER30
      INTEGER SMTH                                                             DRIVER40
      READ (5,100) N,SMTH,TOL                                                  DRIVER50
      WRITE (6,200) N,SMTH,TOL                                                 DRIVER60
      MAXNUM=0                                                                 DRIVER70
   10 MAXNUM=MAXNUM+1                                                          DRIVER80
      READ (5,150) XTABLE(MAXNUM),FTABLE(MAXNUM)                               DRIVER90
      IF (EOF(5).EQ.0.0) GO TO 10                                              DRIVE100
      MAXNUM=MAXNUM-1                                                          DRIVE110
      CALL LINEAR(MAXNUM,200)                                                  DRIVE120
      MAXNUM=200                                                               DRIVE130
      CALL UNIACF(TOL,N,SMTH,MAXNUM,ERROR)                                     DRIVE140
  100 FORMAT (2I5,F10.5)                                                       DRIVE150
  150 FORMAT (2F10.5)                                                          DRIVE160
  200 FORMAT (1H1,47HALGORITHM USING UNIFORM APPROXIMATION OPERATOR.,//,       DRIVE170
     $4HON =,I3,7H SMTH =,I3,6H TOL =,F10.5)                                   DRIVE180
      END                                                                      DRIVE190


      SUBROUTINE LINEAR(OLDMAX,NEWMAX)                                         LINEAR10
C                                                                              LINEAR20
C  WE USE THIS SUBROUTINE TO FILL IN BETWEEN THE ORIGINAL DATA POINTS          LINEAR30
C  BY LINEAR INTERPOLATION WHEN THERE ARE TOO FEW POINTS FOR THE EF-           LINEAR40
C  FECTIVE USE OF THE ALGORITHM.  OLDMAX IS THE NUMBER OF ORIGINAL             LINEAR50
C  DATA POINTS, NEWMAX IS THE TOTAL NUMBER OF INTERPOLATED DATA POINTS         LINEAR60
C  DESIRED.  THE NEW NEWMAX POINTS ARE EQUALLY SPACED BETWEEN                  LINEAR70
C  XTABLE(1) AND XTABLE(OLDMAX).                                               LINEAR80
C                                                                              LINEAR90
      INTEGER OLDMAX                                                           LINEA100
      COMMON XTABLE(500),FTABLE(1582),X(150),Y(150)                            LINEA110
      DO 10 I=1,OLDMAX                                                         LINEA120
         X(I)=XTABLE(I)                                                        LINEA130
         Y(I)=FTABLE(I)                                                        LINEA140
   10 CONTINUE                                                                 LINEA150
      DELTA=(X(OLDMAX)-X(1))/FLOAT(NEWMAX-1)                                   LINEA160
      K=1                                                                      LINEA170
      DO 40 I=2,NEWMAX                                                         LINEA180
         XX=XTABLE(1)+FLOAT(I-1)*DELTA                                         LINEA190
         XTABLE(I)=XX                                                          LINEA200
   20    IF (XX.LE.X(K+1)) GO TO 30                                            LINEA210
         K=K+1                                                                 LINEA220
         GO TO 20                                                              LINEA230
   30    FTABLE(I)=Y(K)+(XX-X(K))*(Y(K+1)-Y(K))/(X(K+1)-X(K))                  LINEA240
   40 CONTINUE                                                                 LINEA250
      RETURN                                                                   LINEA260
C                                                                              LINEA270
      END                                                                      LINEA280


      SUBROUTINE UNIACF(TOL,N,SMTH,MAXNUM,ERROR)                               UNIACF10
C                                                                              UNIACF20
C  THIS SUBROUTINE ADAPTIVELY COMPUTES A PIECEWISE POLYNOMIAL APPROX-          UNIACF30
C  IMATION OF DEGREE N-1 TO THE FUNCTION STORED IN THE ARRAYS XTABLE           UNIACF40
C  AND FTABLE WITH SMTH CONTINUOUS DERIVATIVES AND WHICH DEVIATES FROM         UNIACF50
C  THIS FUNCTION BY NO MORE THAN TOL AT ANY POINT IN XTABLE.                   UNIACF60
C                                                                              UNIACF70
C  THE PARAMETERS ARE AS FOLLOWS--                                             UNIACF80
C                                                                              UNIACF90
C     N - THE NUMBER OF COEFFICIENTS OF EACH POLYNOMIAL PIECE--I.E.,           UNIAC100
C         ONE MORE THAN THE DEGREE OF THE PIECEWISE POLYNOMIAL APPROXI-        UNIAC110
```

```
C        MATION.  N MUST BE GREATER THAN 1. AND AS THE ARRAYS ARE CUR-      UNIAC120
C        RENTLY DIMENSIONED, IT IS ASSUMED THAT N IS NO BIGGER THAN 17.    UNIAC130
C                                                                          UNIAC140
C     SMTH - THE NUMBER OF CONTINUOUS DERIVATIVES DESIRED OF THE           UNIAC150
C        APPROXIMATION.  SMTH MUST NOT BE GREATER THAN N-2.  IF ONLY       UNIAC160
C        CONTINUITY OF THE APPROXIMATION IS REQUIRED, SET SMTH = 0.        UNIAC170
C        (IN FACT AN APPROXIMATION WHICH IS DISCONTINUOUS AT THE KNOTS     UNIAC180
C        MAY BE OBTAINED BY SETTING SMTH = -1.)                            UNIAC190
C                                                                          UNIAC200
C     MAXNUM - THE NUMBER OF POINTS ACTUALLY STORED IN THE ARRAYS          UNIAC210
C        XTABLE AND FTABLE.  (AS THESE ARRAYS ARE CURRENTLY DIMENSIONED    UNIAC220
C        MAXNUM MUST BE LESS THAN 500.  IF ONE WISHES TO COMPUTE AP-       UNIAC230
C        PROXIMATIONS TO FUNCTIONS WITH MORE THAN 500 POINTS, ONE CAN      UNIAC240
C        EASILY MODIFY THIS PROGRAM TO CONTINUOUSLY READ IN MORE POINTS    UNIAC250
C        AFTER ROOM IS MADE IN THE TWO STORAGE ARRAYS, XTABLE AND          UNIAC260
C        FTABLE BY HAVING COMPLETED SEVERAL SUBINTERVALS.)                 UNIAC270
C                                                                          UNIAC280
C     TOL - THE TOLERANCE THAT THE APPROXIMATION MUST SATISFY.             UNIAC290
C                                                                          UNIAC300
C     ERROR - A LOGICAL VALUE SET TO .TRUE. IF AN ERROR OCCURS IN THE      UNIAC310
C        PROGRAM (AN APPROPRIATE ERROR MESSAGE WILL ALSO BE PRINTED)       UNIAC320
C        AND SET TO .FALSE. OTHERWISE.                                     UNIAC330
C                                                                          UNIAC340
C     BLANK COMMON PROVIDES THE REMAINDER OF THE INPUT.  IT IS ASSUMED     UNIAC350
C     THAT THE FIRST 500 WORDS OF BLANK COMMON CONTAIN THE TABLE OF X      UNIAC360
C     VALUES (XTABLE), AND THAT THE NEXT 500 WORDS CONTAIN THE TABLE OF    UNIAC370
C     FUNCTION VALUES (FTABLE).  THE NEXT TWO WORDS IN BLANK COMMON ARE    UNIAC380
C     USED INTERNALLY THROUGHOUT THE PROGRAM.  THE NEXT 1080 WORDS OF      UNIAC390
C     BLANK COMMON STORAGE CONTAIN THE COMPUTED COEFFICIENTS AND THE       UNIAC400
C     KNOTS--I.E., CSTORE(I,INT) IS THE COEFFICIENT OF THE I-1ST DEGREE    UNIAC410
C     TERM IN SUBINTERVAL NUMBER INT.  CSTORE(N+1,INT) IS THE LEFT END     UNIAC420
C     POINT OF SUBINTERVAL INT.  THE LAST ARRAY OF 300 WORDS IS USED       UNIAC430
C     INTERNALLY TO STORE THE ERRORS OF APPROXIMATION OCCURING AT EACH     UNIAC440
C     DATA POINT DURING THE REMES ALGORITHM, AND IT IS ALSO USED AS A      UNIAC450
C     SCRATCH STORAGE ARRAY THROUGHOUT THE PROGRAM.                        UNIAC460
C                                                                          UNIAC470
C  THE COEFFICIENTS AND SUBINTERVAL ENDPOINTS ARE PRINTED OUT AS THEY      UNIAC480
C  ARE COMPUTED.  ALSO THE FUNCTION EVAL IS AVAILABLE TO THE USER TO       UNIAC490
C  EVALUATE THE APPROXIMATION AT ANY POINT WITHIN THE ENTIRE INTERVAL.     UNIAC500
C                                                                          UNIAC510
      LOGICAL LAST,ERROR,DONE,ABORT                                        UNIAC520
      INTEGER SMTH                                                         UNIAC530
      DIMENSION C(18)                                                      UNIAC540
      COMMON XTABLE(1000),LCTNLE,LCTNRE,DUM1(1380)                         UNIAC550
      COMMON /SCALAR/ NPLUS0,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT      UNIAC560
      DATA MAXINT/301/,NRSMTH/-1/                                          UNIAC570
C                                                                          UNIAC580
      ERROR=.FALSE.                                                        UNIAC590
      DONE=.FALSE.                                                         UNIAC600
      ABORT=.FALSE.                                                        UNIAC610
      NPLUS0=N                                                             UNIAC620
      NPLUS1=N+1                                                           UNIAC630
C                                                                          UNIAC640
C  IN THE FIRST SUBINTERVAL THERE ARE NO INTERPOLATORY CONSTRAINTS--        UNIAC650
C  CONSEQUENTLY, WE SET NLSMTH=-1.                                         UNIAC660
C                                                                          UNIAC670
      NLSMTH=-1                                                            UNIAC680
      NX=NPLUS1-2-NLSMTH-NRSMTH                                            UNIAC690
      NXM1=NX-1                                                            UNIAC700
      LNGTH=NX+1                                                           UNIAC710
C                                                                          UNIAC720
C  WE INITIALLY TRY AS MUCH OF THE CURRENT REMAINING PORTION OF THE        UNIAC730
```

```
C  WHOLE INTERVAL AS POSSIBLE AS AN INITIAL GUESS FOR EACH SUCCESSIVE                UNIAC740
C  SUBINTERVAL.  LCTNLE IS THE LOCATION (IN THE ARRAY XTABLE) OF THE                 UNIAC750
C  LEFT END POINT OF THE CURRENT SUBINTERVAL, LCTNRE IS THE LOCATION OF              UNIAC760
C  THE RIGHT END POINT.                                                             UNIAC770
C                                                                                   UNIAC780
      LCTNLE=1                                                                      UNIAC790
      LCTNRE=MIN0(MAXNUM,MAXINT)                                                    UNIAC800
      DO 20 INTNUM=1,60                                                             UNIAC810
         NINT=INTNUM                                                                UNIAC820
C                                                                                   UNIAC830
C  SUBROUTINE COMPUT FINDS THE LARGEST SUBINTERVAL OF (XTABLE(LCTNLE),              UNIAC840
C  XTABLE(LCTNRE)) WITH LEFT END POINT XTABLE(LCTNLE) SUCH THAT THE BEST            UNIAC850
C  APPROXIMATION ON THIS SUBINTERVAL SATISFIES ALL THE CONSTRAINTS.  THE            UNIAC860
C  RIGHT END POINT IS *BACKED OFF* TO THE LAST INTERIOR EXTREME POINT               UNIAC870
C  OF F-P TO ADD TO THE STABILITY OF THE ALGORITHM.  THE LOCATION OF                UNIAC880
C  THIS RIGHT END POINT IS STORED IN LCTNRE.  IF LCTNRE=MAXNUM (I.E.                UNIAC890
C  WE ARE DONE), CONTROL IS PASSED TO LINE 40.  IF NO SUCH SUBINTERVAL              UNIAC900
C  CAN BE FOUND, CONTROL IS PASSED TO LINE 50.  IF THERE ARE FEWER THAN             UNIAC910
C  LNGTH POINTS FROM LCTNRE TO MAXNUM, LAST IS SET TO .TRUE. AND THE                UNIAC920
C  SPECIAL CASE SUBROUTINE LSTINT IS CALLED.                                        UNIAC930
C                                                                                   UNIAC940
         CALL COMPUT (C,TOL,LNGTH,MAXNUM,LAST,DONE,ABORT)                           UNIAC950
         IF (ABORT) GO TO 50                                                        UNIAC960
         IF (DONE) GO TO 40                                                         UNIAC970
         IF (INTNUM.GT.1) GO TO 10                                                  UNIAC980
         NLSMTH=SMTH                                                                UNIAC990
         NX=NPLUS1-2-NLSMTH-NRSMTH                                                  UNIA1000
         NAMI=NX-1                                                                  UNIA1010
         LNGTH=NX+1                                                                 UNIA1020
   10    IF (LAST) GO TO 30                                                         UNIA1030
C                                                                                   UNIA1040
C  SUBROUTINE STORE STORES THE COEFFICIENTS FOR THIS SUBINTERVAL IN THE             UNIA1050
C  ARRAY CSTORE.  IT ALSO PRINTS OUT THE COEFFICIENTS AND THE ERROR OF              UNIA1060
C  APPROXIMATION ON THIS SUBINTERVAL.  SUBROUTINE SETP(C,X,K) STORES THE            UNIA1070
C  VALUE OF THE POLYNOMIAL DETERMINED BY THE COEFFICIENTS IN THE ARRAY              UNIA1080
C  C AND ITS FIRST K DERIVATIVES AT THE POINT X IN THE ARRAY PPRIME.                UNIA1090
C                                                                                   UNIA1100
         CALL STORE (C,LCTNLE,LCTNRE)                                               UNIA1110
         CALL SETP (C,XTABLE(LCTNRE),NLSMTH)                                        UNIA1120
         LCTNLE=LCTNRE                                                              UNIA1130
         LCTNRE=MIN0(MAXNUM,MAXINT+LCTNLE-1)                                        UNIA1140
   20 CONTINUE                                                                      UNIA1150
      WRITE (6,60) NINT                                                             UNIA1160
      ERROR=.TRUE.                                                                  UNIA1170
      RETURN                                                                        UNIA1180
   30 CALL LSTINT (C,TOL,LNGTH,MAXNUM,ABORT)                                        UNIA1190
      IF (ABORT) GO TO 50                                                           UNIA1200
   40 CALL STORE (C,LCTNLE,LCTNRE)                                                  UNIA1210
      RETURN                                                                        UNIA1220
   50 WRITE (6,70)                                                                  UNIA1230
      ERROR=.TRUE.                                                                  UNIA1240
      RETURN                                                                        UNIA1250
C                                                                                   UNIA1260
   60 FORMAT (1H0,39(2H* ),1H*,/,2H0*,12X, 37HTHIS APPROXIMATION REQUIRE           UNIA1270
     1S MORE THAN,I3, 13H SUBINTERVALS,12X,1H*,/,2H0*,28X, 20H--PROGRAM            UNIA1280
     2ABORTING--,29X,1H*,/,1H0,39(2H* ),1H*)                                        UNIA1290
   70 FORMAT (1H0,39(2H* ),1H*,/,2H0*,15X, 46HTHE ALGORITHM CANNOT MEET             UNIA1300
     1THE DESIRED ACCURACY,16X,1H*,/,2H0*,28X, 20H--PROGRAM ABORTING--,2            UNIA1310
     29X,1H*,/,1H0,39(2H* ),1H*)                                                    UNIA1320
C                                                                                   UNIA1330
      END                                                                           UNIA1340
```

```
      SUBROUTINE COMPUT(C,TOL,LNGTH,MAXNUM,LAST,DONE,ABORT)                 COMPUT10
C                                                                           COMPUT20
C  THIS SUBROUTINE FINDS THE LARGEST SUBINTERVAL AND THE BEST APPROX-       COMPUT30
C  IMATION TO F ON THIS SUBINTERVAL SUCH THAT THE APPROXIMATION MEETS       COMPUT40
C  THE DESIRED ERROR TOLERANCE ON THE SUBINTERVAL.                          COMPUT50
C                                                                           COMPUT60
      INTEGER A,B                                                           COMPUT70
      LOGICAL LAST,OK,DONE,ABORT,TOOBIG                                     COMPUT80
      REAL C(18)                                                            COMPUT90
      COMMON XTABLE(1000),LCTNLE,LCTNRE,CSTORE(18,60),CDERIV(300)           COMPU100
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT            COMPU110
      COMMON /COMP/ LCTNX(18)                                               COMPU120
C                                                                           COMPU130
C  WE ASSUME THAT WE ARE CLOSE ENOUGH TO THE TRUE LARGEST SUBINTERVAL       COMPU140
C  RIGHT END POINT WHEN WE KNOW THAT OUR APPROXIMATION TO THE TRUE RIGHT    COMPU150
C  END POINT IS WITHIN ETA OF THE TRUE END POINT.                           COMPU160
C                                                                           COMPU170
      DATA ETA/.08/                                                         COMPU180
C                                                                           COMPU190
      LITTLE=LCTNLE+LNGTH-1                                                 COMPU200
      A=0                                                                   COMPU210
      LAST=.FALSE.                                                          COMPU220
   10 NUMPTS=LCTNRE-LCTNLE+1                                                COMPU230
C                                                                           COMPU240
C  IF THE ACCURACY OF THE BEST APPROXIMATION ON THE CURRENT SUBINTERVAL     COMPU250
C  EXCEEDS TOL, CONTROL IS PASSED TO LINE 30.                               COMPU260
C                                                                           COMPU270
      CALL REMES (C,LCTNX,TOL,TOOBIG,ABORT)                                 COMPU280
      IF (ABORT) RETURN                                                     COMPU290
      IF (TOOBIG) GO TO 30                                                  COMPU300
      IF (LCTNRE.LT.MAXNUM) GO TO 20                                        COMPU310
      DONE=.TRUE.                                                           COMPU320
      RETURN                                                                COMPU330
C                                                                           COMPU340
C  A IS THE CURRENT LARGEST LOCATION FOR A RIGHT ENDPOINT SUCH THAT         COMPU350
C  THE BEST APPROXIMATION ON THIS SUBINTERVAL SATIFIES ALL CONSTRAINTS.     COMPU360
C                                                                           COMPU370
   20 A=LCTNRE                                                              COMPU380
      IF ((XTABLE(B)-XTABLE(A).GT.ETA).AND.(B-A.GT.1)) GO TO 40             COMPU390
      GO TO 50                                                              COMPU400
C                                                                           COMPU410
C  B IS THE CURRENT SMALLEST LOCATION FOR A RIGHT ENDPOINT SUCH THAT        COMPU420
C  THE BEST APPROXIMATION ON THIS SUBINTERVAL FAILS TO SATISFY THE CON-     COMPU430
C  STRAINTS.                                                                COMPU440
C                                                                           COMPU450
   30 B=LCTNRE                                                              COMPU460
   40 NEWTRY=(A+B)/2+1                                                      COMPU470
      IF (NEWTRY.EQ.B) NEWTRY=NEWTRY-1                                      COMPU480
      IF (NEWTRY.LT.LITTLE) NEWTRY=LITTLE                                   COMPU490
      IF (NEWTRY.EQ.LCTNRE) GO TO 50                                        COMPU500
      LCTNRE=NEWTRY                                                         COMPU510
      GO TO 10                                                              COMPU520
C                                                                           COMPU530
C  IF A IS STILL 0, THEN NO SUBINTERVAL WITH AT LEAST LNGTH POINTS          COMPU540
C  WILL WORK, SO THE ALGORITHM IS TERMINATED.                               COMPU550
C                                                                           COMPU560
   50 IF (A.NE.0) GO TO 60                                                  COMPU570
      ABORT=.TRUE.                                                          COMPU580
      RETURN                                                                COMPU590
C                                                                           COMPU600
C  SINCE NEWTRY IS ALWAYS STRICTLY LESS THAN THE CURRENT B, IF NEWTRY=      COMPU610
C  LCTNRE, AND A IS NOT STILL 0, NEWTRY=A, WHICH IS A POINT WHICH SAT-      COMPU620
```

```
C  ISFIES ALL REQUIREMENTS.  WE NOW BACK THE RIGHT ENDPOINT OFF TO THE          COMPU630
C  BEST INTERIOR EXTREME POINT OF F-P TO ADD TO THE STABILITY OF THE AL-        COMPU640
C  GORITHM.                                                                     COMPU650
C                                                                               COMPU660
   60 DO 70 I=1,N                                                               COMPU670
         CDERIV(I)=C(I)                                                        COMPU680
         CSTORE(I,NINT)=C(I)                                                   COMPU690
   70 CONTINUE                                                                  COMPU700
      NDUMMY=N                                                                  COMPU710
      CALL DERIV (CDERIV,NDUMMY)                                                COMPU720
      I=NX                                                                      COMPU730
      LCTNRE=LCTNX(I)                                                           COMPU740
      NEWTRY=LCTNRE                                                             COMPU750
      SMLL=CMPR(CDERIV,NDUMMY,NEWTRY,LCTNLE-1,OK)                               COMPU760
      IF (OK) GO TO 100                                                         COMPU770
   80 I=I-1                                                                     COMPU780
      IF (I.EQ.0) GO TO 100                                                     COMPU790
      NEWTRY=LCTNX(I)                                                           COMPU800
      IF (NEWTRY.LT.LNGTH) GO TO 100                                            COMPU810
      TEMP=CMPR(CDERIV,NDUMMY,NEWTRY,LCTNLE-1,OK)                               COMPU820
      IF (.NOT.OK) GO TO 90                                                     COMPU830
      LCTNRE=NEWTRY                                                             COMPU840
      GO TO 100                                                                 COMPU850
   90 IF (TEMP.GE.SMLL) GO TO 80                                                COMPU860
      SMLL=TEMP                                                                 COMPU870
      LCTNRE=NEWTRY                                                             COMPU880
      GO TO 80                                                                  COMPU890
  100 LCTNRE=LCTNLE+LCTNRE-1                                                    COMPU900
      IF (MAXNUM-LCTNRE+1.LT.LNGTH) LAST=.TRUE.                                 COMPU910
      RETURN                                                                    COMPU920
C                                                                               COMPU930
      END                                                                       COMPU940


      SUBROUTINE LSTINT(C,TOL,LNGTH,MAXNUM,ABORT)                               LSTINT10
C                                                                               LSTINT20
C  THIS SUBROUTINE HANDLES THE SPECIAL CASE OF FINDING A SUBINTERVAL            LSTINT30
C  AND A BEST APPROXIMATION ON THAT SUBINTERVAL WHEN THERE ARE TOO              LSTINT40
C  FEW REMAINING POINTS FOR COMPUT TO WORK.                                     LSTINT50
C                                                                               LSTINT60
      INTEGER OLDLE,OLDRE                                                       LSTINT70
      LOGICAL TOOBIG,ABORT                                                      LSTINT80
      REAL C(18)                                                                LSTINT90
      COMMON X(1000),LCTNLE,LCTNRE,CSTORE(18,60)                                LSTIN100
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT                LSTIN110
      COMMON /COMP/ LCTNX(18)                                                   LSTIN120
C                                                                               LSTIN130
      DO 10 OLDLE=1,NPLUS1                                                      LSTIN140
   10 CSTORE(OLDLE,NINT)=C(OLDLE)                                               LSTIN150
      OLDLE=LCTNLE                                                              LSTIN160
      OLDRE=LCTNRE                                                              LSTIN170
      LCTNRE=MAXNUM                                                             LSTIN180
      LCTNLE=MIN0(MAXNUM-LNGTH+1,(MAXNUM-OLDLE+1)/2)-1                          LSTIN190
   20 LCTNLE=LCTNLE+1                                                           LSTIN200
      IF (MAXNUM-LCTNLE+1.LT.LNGTH) GO TO 40                                    LSTIN210
      CALL SETP (CSTORE(1,NINT),X(LCTNLE),NLSMTH)                               LSTIN220
      NUMPTS=LCTNRE-LCTNLE+1                                                    LSTIN230
      CALL REMES (C,LCTNX,TOL,TOOBIG,ABORT)                                     LSTIN240
      IF (ABORT) RETURN                                                         LSTIN250
      IF (TOOBIG) GO TO 20                                                      LSTIN260
      CALL STORE (CSTORE(1,NINT),OLDLE,LCTNLE)                                  LSTIN270
      NINT=NINT+1                                                               LSTIN280
```

```
      DO 30 OLDLE=1,NPLUS1                                              LSTIN290
   30 CSTORE(OLDLE,NINT)=C(OLDLE)                                       LSTIN300
      RETURN                                                            LSTIN310
   40 ABORT=.TRUE.                                                      LSTIN320
      RETURN                                                            LSTIN330
C                                                                       LSTIN340
      END                                                               LSTIN350


      SUBROUTINE STORE(C,LCTNLE,LCTNRE)                                 STORE 10
C                                                                       STORE 20
C  THIS SUBROUTINE OUTPUTS THE COEFFICIENTS AND ENDPOINTS OF THE        STORE 30
C  CURRENT APPROXIMATION AND SUBINTERVAL.  APPROPRIATE INFORMATION      STORE 40
C  IS STORED IN THE ARRAY CSTORE TO ALLOW THE ENTIRE PIECEWISE POLY-    STORE 50
C  NOMIAL APPROXIMATION TO BE EASILY EVALUATED AT ANY POINT BY THE      STORE 60
C  FUNCTION EVAL.                                                       STORE 70
C                                                                       STORE 80
      DIMENSION C(18)                                                   STORE 90
      COMMON XTABLE(500),FTABLE(502),CSTORE(18,60),DUM1(300)            STORE100
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT        STORE110
C                                                                       STORE120
      NUMPTS=LCTNRE-LCTNLE+1                                            STORE130
      WRITE (6,30) NINT,XTABLE(LCTNLE),XTABLE(LCTNRE),NUMPTS            STORE140
      WRITE (6,40) (I,C(I),I=1,N)                                       STORE150
      ERR=0.0                                                           STORE160
      DO 10 I=LCTNLE,LCTNRE                                             STORE170
         TEMP=ABS(FTABLE(I)-HORNER(C,XTABLE(I),N))                      STORE180
         IF (TEMP.GT.ERR) ERR=TEMP                                      STORE190
   10 CONTINUE                                                          STORE200
      WRITE (6,50) ERR                                                  STORE210
      DO 20 I=1,N                                                       STORE220
   20 CSTORE(I,NINT)=C(I)                                               STORE230
      CSTORE(NPLUS1,NINT)=XTABLE(LCTNLE)                                STORE240
      RETURN                                                            STORE250
C                                                                       STORE260
   30 FORMAT (//,5X, 15HINTERVAL NUMBER,I4, 16H WHICH BEGINS AT,E23.16,/ STORE270
     1, 12H AND ENDS AT,E23.16,2X, 8HCONTAINS,I4, 8H POINTS.,/, 60H TH   STORE280
     2E COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE,/)       STORE290
   40 FORMAT ((10X, 2HC(,I2, 3H) =,E24.16))                             STORE300
   50 FORMAT (/, 47H THE ERROR OF APPROXIMATION IN THIS INTERVAL IS,E24. STORE310
     116, 1H.)                                                          STORE320
C                                                                       STORE330
      END                                                               STORE340


      SUBROUTINE DERIV(C,N)                                             DERIV 10
C                                                                       DERIV 20
C  THIS SUBROUTINE REPLACES THE COEFFICIENTS OF A POLYNOMIAL IN STAND-  DERIV 30
C  ARD FORM WITH THE COEFFICIENTS OF THIS POLYNOMIAL-S DERIVATIVE.       DERIV 40
C  THE NUMBER OF COEFFICIENTS, N, IS DECREMENTED.                       DERIV 50
C                                                                       DERIV 60
      DIMENSION C(N)                                                    DERIV 70
C                                                                       DERIV 80
      N=N-1                                                             DERIV 90
      DO 10 I=1,N                                                       DERIV100
   10 C(I)=FLOAT(I)*C(I+1)                                              DERIV110
      RETURN                                                            DERIV120
C                                                                       DERIV130
      END                                                               DERIV140


      SUBROUTINE SETP(C,X,SMTH)                                         SETP  10
```

```
C                                                                       SETP  20
C  THIS SUBROUTINE APPROPRIATELY STORES IN THE ARRAY PPRIME THE VAL-    SETP  30
C  UES WHICH MUST BE INTERPOLATED TO GIVE THE PIECEWISE POLYNOMIAL THE  SETP  40
C  DESIRED SMOOTHNESS.                                                  SETP  50
C                                                                       SETP  60
      DIMENSION C(18)                                                   SETP  70
      COMMON /COMP/ CSTORE(18)                                          SETP  80
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT        SETP  90
      COMMON /DDIF/ PPRIME(5),DUM1(18)                                  SETP 100
      INTEGER SMTH                                                      SETP 110
C                                                                       SETP 120
      DO 10 I=1,N                                                       SETP 130
   10 CSTORE(I)=C(I)                                                    SETP 140
      NDUMMY=N                                                          SETP 150
      I=0                                                               SETP 160
   20 IF (I.GT.SMTH) RETURN                                             SETP 170
      IF (I.EQ.0) GO TO 30                                              SETP 180
      CALL DERIV (CSTORE,NDUMMY)                                        SETP 190
   30 PPRIME(I+1)=HORNER(CSTORE,X,NDUMMY)                               SETP 200
      I=I+1                                                             SETP 210
      GO TO 20                                                          SETP 220
C                                                                       SETP 230
      END                                                               SETP 240

      FUNCTION CMPR(C,N,NEWTRY,OK)                                      CMPR  10
C                                                                       CMPR  20
C  THIS SUBROUTINE COMPARES THE FIRST DERIVATIVE OF THE CURRENT PIECE OF CMPR  30
C  THE PIECEWISE POLYNOMIAL APPROXIMATION EVALUATED AT XTABLE(NEWTRY)   CMPR  40
C  WITH THE FIRST DERIVATIVE OF THE QUADRATIC INTERPOLATION OF F, CEN-  CMPR  50
C  TERED AROUND XTABLE(NEWTRY), EVALUATED AT XTABLE(NEWTRY).  IF THESE  CMPR  60
C  TWO DIFFER IN ABSOLUTE VALUE BY LESS THAN TOLER (EITHER ABSOLUTELY   CMPR  70
C  OR RELATIVELY), WE SET OK TO .TRUE. AND WE ACCEPT XTABLE(NEWTRY) AS A CMPR  80
C  REASONABLE SUBINTERVAL RIGHT ENDPOINT.  NOTE THAT THE USER MAY EASILY CMPR  90
C  CHANGE TOLER BY MEANS OF THE FOLLOWING DATA STATEMENT.               CMPR 100
C                                                                       CMPR 110
      LOGICAL OK                                                        CMPR 120
      COMMON X(500),F(500),DUM1(1382)                                   CMPR 130
      DIMENSION C(18)                                                   CMPR 140
      DATA TOLER/.05/                                                   CMPR 150
C                                                                       CMPR 160
      OK=.FALSE.                                                        CMPR 170
      A=(F(NEWTRY)-F(NEWTRY-1))/(X(NEWTRY)-X(NEWTRY-1))                 CMPR 180
      B=(F(NEWTRY+1)-F(NEWTRY))/(X(NEWTRY+1)-X(NEWTRY))                 CMPR 190
      D=(B-A)/(X(NEWTRY+1)-X(NEWTRY-1))                                 CMPR 200
      CMPR=A+D*(X(NEWTRY)-X(NEWTRY-1))                                  CMPR 210
      A=TOLER*CMPR                                                      CMPR 220
      CMPR=ABS(CMPR-HORNER(C,X(NEWTRY),N))                              CMPR 230
      IF (CMPR.LE.A) OK=.TRUE.                                          CMPR 240
      RETURN                                                            CMPR 250
C                                                                       CMPR 260
      END                                                               CMPR 270

      SUBROUTINE REMES(C,LCTNX,IOL,TOOBIG,ABORT)                        REMES 10
C                                                                       REMES 20
C  THIS IS THE DRIVING PROGRAM FOR THE COMPUTATION OF THE BEST          REMES 30
C  UNIFORM POLYNOMIAL APPROXIMATION TO F(X) (VALUES ARE STORED          REMES 40
C  IN XTABLE AND FTABLE) OF DEGREE LESS THAN OR EQUAL TO N-1 ON         REMES 50
C  THE SUBINTERVAL [XTABLE(LCTNLE),XTABLE(LCTNRE)].  SEE INTRODUCTION TO REMES 60
C  APPROXIMATION THEORY BY E. W. CHENEY FOR A COMPLETE DISCUSSION OF    REMES 70
C  THIS ALGORITHM.                                                      REMES 80
```

```
C
      DIMENSION C(18),SGNRXI(18),LCTNX(18),LCTNZ(18)                    REMES 90
      COMMON XTABLE(500),FTABLE(500),LCTNLE,DUMMY(1081),ERROR(300)      REMES100
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMGRD,NINT        REMES110
      LOGICAL STOP,TOOBIG,ABORT                                         REMES120
      INTEGER FRSTM1,START                                              REMES130
C                                                                       REMES140
C  EPS IS A MACHINE CONSTANT--SET EPS TO (VERY ROUGHLY) THE SMALLEST    REMES150
C  VALUE SUCH THAT EPS + 1.0 IS GREATER THAN 1.0.                       REMES160
C                                                                       REMES170
      DATA ITERMX,EPS/9,1.0E-10/                                        REMES180
C                                                                       REMES190
C  FIRST WE INITIALIZE VARIOUS ARRAYS AND VARIABLES.                    REMES200
C                                                                       REMES210
      TOOBIG=.FALSE.                                                    REMES220
      FRSTM1=LCTNLE-1                                                   REMES230
      SGNRXI(1)=1.0                                                     REMES240
      DO 10 I=2,NX                                                      REMES250
   10 SGNRXI(I)=-SGNRXI(I-1)                                            REMES260
      NFPTS=NUMGRD-2                                                    REMES270
      START=2                                                           REMES280
      ERROR(1)=0.0                                                      REMES290
      IF (NLSMTH.GE.0) GO TO 20                                         REMES300
      NFPTS=NFPTS+1                                                     REMES310
      START=1                                                           REMES320
   20 IF (NRSMTH.LT.0) NFPTS=NFPTS+1                                    REMES330
      DELTA=FLOAT(NFPTS-1)/FLOAT(NX-1)                                  REMES340
      DO 30 I=1,NX                                                      REMES350
   30 LCTNX(I)=START+IFIX(FLOAT(I-1)*DELTA+.5)                          REMES360
C                                                                       REMES370
C  NOW WE BEGIN ITERATING.  CONVERGENCE OCCURS WHEN TWO CONSECUTIVE     REMES380
C  REFERENCE SETS (DETERMINED BY SOLVE) ARE THE SAME.                   REMES390
C                                                                       REMES400
      DO 80 ITER=1,ITERMX                                               REMES410
         CALL DIVDIF (C,LCTNX,SGNRXI,ABORT)                             REMES420
         IF (ABS(C(NPLUS1)).GT.TOL) TOOBIG=.TRUE.                       REMES430
         IF (ABORT.OR.TOOBIG) RETURN                                    REMES440
         DO 40 I=START,NUMGRD                                           REMES450
   40    ERROR(I)=FTABLE(I+FRSTM1)-BPOLY(XTABLE(I+FRSTM1),C,N)          REMES460
         IF (ABS(C(NPLUS1)).LE.EPS) GO TO 60                            REMES470
         SGNRXI(1)=SIGN(1.0,ERROR(LCTNX(1)))                            REMES480
         DO 50 I=2,NX                                                   REMES490
            SGNRXI(I)=SIGN(1.0,ERROR(LCTNX(I)))                         REMES500
            IF (SGNRXI(I)*SGNRXI(I-1).GE.0.0) GO TO 90                  REMES510
   50    CONTINUE                                                       REMES520
   60    CALL ZEROFD (LCTNX,LCTNZ,SGNRXI,ERROR)                         REMES530
         CALL SOLVE (LCTNX,LCTNZ,SGNRXI,TOL,C(NPLUS1),STOP,TOOBIG)      REMES540
         IF (C(NPLUS1).GT.TOL) STOP=TOOBIG=.TRUE.                       REMES550
         IF (.NOT.STOP) GO TO 80                                        REMES560
         IF (.NOT.TOOBIG) CALL TRANS (C,N)                              REMES570
         RETURN                                                         REMES580
   80 CONTINUE                                                          REMES590
C                                                                       REMES600
C  THE REMAINING STATEMENTS ARE FOR DEBUGGING.                          REMES610
C                                                                       REMES620
      WRITE (6,110) ITERMX                                              REMES630
      GO TO 100                                                         REMES640
   90 WRITE (6,120) ITER                                                REMES650
  100 ABORT=.TRUE.                                                      REMES660
      RETURN                                                            REMES670
C                                                                       REMES680
  110 FORMAT (1H0,39(2H* ),1H*,/,2H0*,11X, 40HTHE REMES ALGORITHM HAS NO REMES690
```

```
     1T CONVERGED IN,I3, 12H ITERATIONS.,11X,1H*,/,2H0*,11X, 36HPROGRAM      REMES710
     2ABORTED IN SUBROUTINE REMES.,30X,1H*,/,1H0,39(2H* ),1H*)               REMES720
  120 FORMAT (1H0,39(2H* ),1H*,/,2H0*,8X, 12HIN ITERATION,I3, 47H OF REM     REMES730
     1ES. NO ALTERNATION OF SIGN OCCURS AT THE,7X,1H*,/,2H0*,8X, 57HEXTR     REMES740
     2EME POINTS.  THE PROGRAM ABORTED IN SUBROUTINE REMES.,12X,1H*,/,1H     REMES750
     30,39(2H* ),1H*)                                                        REMES760
C                                                                            REMES770
      END                                                                    REMES780


      SUBROUTINE DIVDIF(C,LCTNX,SGNRXI,ABORT)                                DIVDIF10
C                                                                            DIVDIF20
C  THIS SUBROUTINE MAKES USE OF A DIVIDED DIFFERENCE SCHEME FOR              DIVDIF30
C  SOLVING THE VANDERMONDE-LIKE LINEAR SYSTEM INHERENT IN THE REMES          DIVDIF40
C  ALGORITHM.  THE ADVANTAGES OF USING THIS SPECIAL PURPOSE LINEAR           DIVDIF50
C  SYSTEM SOLVER ARE--                                                       DIVDIF60
C                                                                            DIVDIF70
C     THIS ROUTINE REQUIRES ON THE ORDER OF N**2 OPERATION AS COM-           DIVDIF80
C     PARED TO GAUSSIAN ELIMINATION WHICH REQUIRES ON THE ORDER              DIVDIF90
C     OF N**3 OPERATIONS.                                                    DIVDI100
C                                                                            DIVDI110
C     THIS ROUTINE REQUIRES ON THE ORDER OF N STORAGE LOCATIONS              DIVDI120
C     AS COMPARED TO GAUSSIAN ELIMINATION WHICH REQUIRES ON THE              DIVDI130
C     ORDER OF N**2.                                                         DIVDI140
C                                                                            DIVDI150
C  SEE THE FORTHCOMING PAPER BY J.A. HULL, S.F. MCCORMICK, AND G.D.          DIVDI160
C  TAYLOR FOR A COMPLETE DESCRIPTION OF THIS ALGORITHM.                      DIVDI170
C                                                                            DIVDI180
      COMMON XTABLE(500),FTABLE(500),LCTNLE,LCTNRE,CSTORE(18,60),            DIVDI190
     1ERROR(300)                                                             DIVDI200
      COMMON /DDIF/ PPRIME(5),X(18)                                          DIVDI210
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT             DIVDI220
      DIMENSION LCTNX(18),C(18),SGNRXI(18),D(18)                             DIVDI230
      INTEGER FRSTM1                                                         DIVDI240
      LOGICAL ABORT                                                          DIVDI250
C                                                                            DIVDI260
C  FIRST WE INITIALIZE SEVERAL VARIABLES.                                    DIVDI270
C                                                                            DIVDI280
      FRSTM1=LCTNLE-1                                                        DIVDI290
      IF (NRSMTH.GE.0) SGNRXI(NPLUS1)=0.0                                    DIVDI300
      IF (NRSMTH.LT.0) SGNRXI(NPLUS1)=SGNRXI(NX)                             DIVDI310
C                                                                            DIVDI320
C  SET UP THE VECTOR X AND THE FIRST ROW OF THE DIVIDED DIFFERENCE TAB-      DIVDI330
C  LE, USING THE COEFFICIENT VECTOR C FOR TEMPORARY STORAGE.                 DIVDI340
C                                                                            DIVDI350
      I=0                                                                    DIVDI360
   10 IF (I.GT.NLSMTH) GO TO 20                                              DIVDI370
      I=I+1                                                                  DIVDI380
      X(I)=XTABLE(LCTNLE)                                                    DIVDI390
      C(I)=PPRIME(1)                                                         DIVDI400
      GO TO 10                                                               DIVDI410
   20 J=0                                                                    DIVDI420
   30 IF (J.GE.NX) GO TO 40                                                  DIVDI430
      I=I+1                                                                  DIVDI440
      J=J+1                                                                  DIVDI450
      X(I)=XTABLE(FRSTM1+LCTNX(J))                                           DIVDI460
      C(I)=FTABLE(FRSTM1+LCTNX(J))                                           DIVDI470
      GO TO 30                                                               DIVDI480
   40 IF (I.GE.NPLUS1) GO TO 50                                              DIVDI490
      I=I+1                                                                  DIVDI500
      X(I)=XTABLE(LCTNRE)                                                    DIVDI510
      C(I)=PPRIME(NLSMTH+2)                                                  DIVDI520
```

```
      GO TO 40                                                          DIVDI530
   50 CONTINUE                                                          DIVDI540
C                                                                       DIVDI550
C  WE NOW COMPUTE THE NEEDED DIVIDED DIFFERENCES.                       DIVDI560
C                                                                       DIVDI570
      FAC=1.0                                                           DIVDI580
      DO 100 J=2,N                                                      DIVDI590
         JP1=J+1                                                        DIVDI600
         JM1=J-1                                                        DIVDI610
         FAC=FAC*FLOAT(JM1)                                             DIVDI620
         TEMP2=C(JM1)                                                   DIVDI630
         DO 90 I=J,N                                                    DIVDI640
            IF (X(I).NE.X(I-JM1)) GO TO 70                              DIVDI650
            IF (I.GT.NLSMTH+1) GO TO 60                                 DIVDI660
            IF (J.GT.NPLUS1-NX) GO TO 220                               DIVDI670
            TEMP1=PPRIME(J)/FAC                                         DIVDI680
            GO TO 80                                                    DIVDI690
   60       IF (NLSMTH+JP1.GT.NPLUS1-NX) GO TO 220                      DIVDI700
            TEMP1=PPRIME(NLSMTH+JP1)/FAC                                DIVDI710
   70       TEMP1=(C(I)-C(I-1))/(X(I)-X(I-JM1))                         DIVDI720
   80       C(I-1)=TEMP2                                                DIVDI730
            TEMP2=TEMP1                                                 DIVDI740
   90    CONTINUE                                                       DIVDI750
         C(N)=TEMP2                                                     DIVDI760
  100 CONTINUE                                                          DIVDI770
C                                                                       DIVDI780
C  L**(-1)F HAS NOW BEEN TEMPORARILY STORED IN THE COEFFICIENT ARRAY C. DIVDI790
C  WE NOW COMPUTE L**(-1)C.  WE WILL COMPUTE THE DIVIDED DIFFERENCES    DIVDI800
C  IN THE TEMPORARY STORAGE ARRAY D.  FIRST WE SET UP THE FIRST COLUMN  DIVDI810
C  OF THE DIVIDED DIFFERENCE TABLE.                                     DIVDI820
C                                                                       DIVDI830
      I=0                                                               DIVDI840
  110 IF (I.GT.NLSMTH) GO TO 120                                        DIVDI850
      I=I+1                                                             DIVDI860
      D(I)=0.0                                                          DIVDI870
      GO TO 110                                                         DIVDI880
  120 J=0                                                               DIVDI890
  130 IF (J.GE.NX) GO TO 140                                            DIVDI900
      I=I+1                                                             DIVDI910
      J=J+1                                                             DIVDI920
      D(I)=SGNRXI(J)                                                    DIVDI930
      GO TO 130                                                         DIVDI940
  140 IF (I.GE.N) GO TO 150                                             DIVDI950
      I=I+1                                                             DIVDI960
      D(I)=0.0                                                          DIVDI970
      GO TO 140                                                         DIVDI980
  150 CONTINUE                                                          DIVDI990
C                                                                       DIVD1000
C  WE NOW COMPUTE THE NEEDED DIVIDED DIFFERENCES.                       DIVD1010
C                                                                       DIVD1020
      DO 190 J=2,N                                                      DIVD1030
         TEMP2=D(J-1)                                                   DIVD1040
         DO 180 I=J,N                                                   DIVD1050
            IF (X(I).NE.X(I-J+1)) GO TO 160                             DIVD1060
            TEMP1=0.0                                                   DIVD1070
            GO TO 170                                                   DIVD1080
  160       TEMP1=(D(I)-D(I-1))/(X(I)-X(I-J+1))                         DIVD1090
  170       D(I-1)=TEMP2                                                DIVD1100
            TEMP2=TEMP1                                                 DIVD1110
  180    CONTINUE                                                       DIVD1120
         D(N)=TEMP2                                                     DIVD1130
  190 CONTINUE                                                          DIVD1140
```

```
C                                                                       DIVD1150
C     WE NOW COMPUTE M=(F(N+1)-W[TRANSPOSE]*B1))/(0.0-W[TRANSPOSE]*B2)  DIVD1160
C     =C(N+1)=(UNIFORM ERROR)                                           DIVD1170
C     FIRST WE COMPUTE THE TWO DOT PRODUCTS SIMULTANEOUSLY.             DIVD1180
C                                                                       DIVD1190
      W=1.0                                                             DIVD1200
      B2=0.0                                                            DIVD1210
      B1=B2                                                             DIVD1220
      DO 200 I=1,N                                                      DIVD1230
         B1=B1+C(I)*W                                                   DIVD1240
         B2=B2+D(I)*W                                                   DIVD1250
         W=W*(X(NPLUS1)-X(I))                                           DIVD1260
  200 CONTINUE                                                          DIVD1270
C                                                                       DIVD1280
C     NOW WE COMPUTE M                                                  DIVD1290
C                                                                       DIVD1300
      C(NPLUS1)=(C(NPLUS1)-B1)/(SGNRXI(NPLUS1)-B2)                      DIVD1310
C                                                                       DIVD1320
C     FINALLY, WE COMPUTE THE COEFFICIENTS C(I).                        DIVD1330
C                                                                       DIVD1340
      DO 210 I=1,N                                                      DIVD1350
         C(I)=C(I)-C(NPLUS1)*D(I)                                       DIVD1360
  210 CONTINUE                                                          DIVD1370
      RETURN                                                            DIVD1380
  220 ABORT=.TRUE.                                                      DIVD1390
      WRITE (6,230)                                                     DIVD1400
      RETURN                                                            DIVD1410
C                                                                       DIVD1420
  230 FORMAT (1H0,39(2H* ),1H*,/,2H0*,11X, 54HSUBROUTINE DIVDIF HAS FAIL DIVD1430
     1ED--PROBABLY DUE TO AN INPUT,12X,1H*,/,2H0*,11X, 58HERROR (TO DIVD DIVD1440
     2IF)--NOT ENOUGH ELEMENTS IN THE ARRAY PPRIME,8X,1H*,/,2H0*,11X, 56 DIVD1450
     3HOR TWO IDENTICAL POINTS IN THE ARRAY X.  PROGRAM ABORTED,10X,1H*, DIVD1460
     4/,2H0*,11X, 21HIN SUBROUTINE DIVDIF.,45X,1H*,/,1H0,39(2H* ),1H*)  DIVD1470
C                                                                       DIVD1480
      END                                                               DIVD1490


      SUBROUTINE ZEROFD(LCTNX,LCTNZ,SGNRXI,ERROR)                       ZEROFD10
C                                                                       ZEROFD20
C  THIS SUBROUTINE LOCATES THE (APPROXIMATE) ZEROS BETWEEN CONSECUTIVE  ZEROFD30
C  POINTS OF THE CURRENT REFERENCE SET.  THE LOCATIONS OF THESE POINTS   ZEROFD40
C  IN THE ARRAY XTABLE RELATIVE TO THE BEGINNING OF THE CURRENT SUB-     ZEROFD50
C  INTERVAL ARE STORED IN LCTNZ.                                        ZEROFD60
C                                                                       ZEROFD70
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMGRD,NINT        ZEROFD80
      DIMENSION LCTNX(18),LCTNZ(18),SGNRXI(18),ERROR(300)               ZEROFD90
C                                                                       ZEROF100
      LCTNZ(1)=MIN0(2,2+NLSMTH)                                         ZEROF110
      LCTNZ(NX+1)=MAX0(NUMGRD-1,NUMGRD-1-NRSMTH)                        ZEROF120
      DO 30 I=2,NX                                                      ZEROF130
         LTNIM1=LCTNX(I-1)                                              ZEROF140
         NUMPTS=LCTNX(I)-LTNIM1                                         ZEROF150
         DO 10 J=1,NUMPTS                                               ZEROF160
            NEWTRY=LTNIM1+J                                             ZEROF170
            IF (ERROR(LTNIM1)*ERROR(NEWTRY).LE.0.0) GO TO 20            ZEROF180
   10    CONTINUE                                                       ZEROF190
   20    LCTNZ(I)=NEWTRY                                                ZEROF200
   30 CONTINUE                                                          ZEROF210
      RETURN                                                            ZEROF220
C                                                                       ZEROF230
      END                                                               ZEROF240
```

```
      SUBROUTINE SOLVE(LCTNX,LCTNZ,SGNRXI,TOL,SMALL,STOP,TOOBIG)                SOLVE 10
C                                                                               SOLVE 20
C  THIS SUBROUTINE PERFORMS THE MULTIPLE EXCHANGE OF THE REFERENCE              SOLVE 30
C  SET REQUIRED AT EACH ITERATION OF THE REMES ALGORITHM.  SEE                  SOLVE 40
C  INTRODUCTION TO APPROXIMATION THEORY BY E. W. CHENEY FOR A COM-              SOLVE 50
C  PLETE DISCUSSION OF THIS ALGORITHM.                                          SOLVE 60
C                                                                               SOLVE 70
      COMMON XTABLE(1009),LCTNLE,DUMMY(1081),ERROR(300)                         SOLVE 80
      COMMON /SCALAR/ N,NPLUS1,NX,NXM1,DUM(4)                                   SOLVE 90
      DIMENSION LCTNX(18),LCTNZ(18),SGNRXI(18)                                  SOLVE100
      LOGICAL STOP,TOOBIG                                                       SOLVE110
      INTEGER FRSTM1,RTEND                                                      SOLVE120
C                                                                               SOLVE130
C  EPS AND TWOEPS ARE MACHINE CONSTANTS--SET EPS (VERY ROUGHLY) TO THE          SOLVE140
C  SMALLEST NUMBER SUCH THAT 1.0 + EPS IS GREATER THAN 1.0, AND TWOEPS          SOLVE150
C  TO 2*EPS.  IT IS NOT CRITICAL THAT THESE VALUES BE PRECISE.                  SOLVE160
C                                                                               SOLVE170
      DATA EPS,TWOEPS/5.0E-11,1.0E-10/                                          SOLVE180
C                                                                               SOLVE190
      STOP=.TRUE.                                                               SOLVE200
      FRSTM1=LCTNLE-1                                                           SOLVE210
C                                                                               SOLVE220
C  WE FIRST COMPUTE THE LOCATIONS OF THE NEW SET OF EXTREME POINTS,             SOLVE230
C  STORING THEM IN THE VECTOR LCTNX.  WE BEGIN BY CHOOSING AS THE ITH           SOLVE240
C  ELEMENT OF LCTNX THE LOCATION OF THE GRIDPOINT IN THE SUBINTERVAL            SOLVE250
C  BETWEEN THE ITH AND (I+1)ST ZERO WHICH RESULTS IN THE LARGEST ERROR          SOLVE260
C  OF THE SAME SIGN AS THE PREVIOUS ITH EXTREME POINT (THEREBY GUARANT-         SOLVE270
C  EEING ALTERNATION).  AT THE SAME TIME WE SEARCH FOR THE GRIDPOINT            SOLVE280
C  WHICH RESULTS IN THE LARGEST (ABSOLUTE) ERROR, STORING ITS LOCATION          SOLVE290
C  (IN LNBGST) AND THE NUMBER OF THE SUBINTERVAL IN WHICH IT OCCURS (IN         SOLVE300
C  INBGST).                                                                     SOLVE310
C                                                                               SOLVE320
      BIGER=EPS                                                                 SOLVE330
      BIGEST=EPS                                                                SOLVE340
      DO 30 INTNUM=1,NX                                                         SOLVE350
         BIG=EPS                                                                SOLVE360
         LFTEND=LCTNZ(INTNUM)                                                   SOLVE370
         RTEND=LCTNZ(INTNUM+1)                                                  SOLVE380
         SGN=SGNRXI(INTNUM)                                                     SOLVE390
         DO 20 NEWLOC=LFTEND,RTEND                                              SOLVE400
            TEMP=SGN*ERROR(NEWLOC)                                              SOLVE410
            IF (TEMP.LE.BIG) GO TO 10                                           SOLVE420
            LNBG=NEWLOC                                                         SOLVE430
            BIG=TEMP                                                            SOLVE440
   10       TEMP=ABS(TEMP)                                                      SOLVE450
            IF (TEMP.LE.BIGEST) GO TO 20                                        SOLVE460
            BIGEST=TEMP                                                         SOLVE470
            LNBGST=NEWLOC                                                       SOLVE480
            INBGST=INTNUM                                                       SOLVE490
   20    CONTINUE                                                               SOLVE500
         IF(INTNUM.EQ.1)SMALL=BIG                                               SOLVE510
         IF (BIG.LT.TWOEPS) GO TO 30                                            SOLVE520
         IF (BIG.GT.BIGER) BIGER=BIG                                            SOLVE530
         IF (BIG.LT.SMALL) SMALL=BIG                                            SOLVE540
         IF (LCTNX(INTNUM).EQ.LNBG) GO TO 30                                    SOLVE550
         LCTNX(INTNUM)=LNBG                                                     SOLVE560
         STOP=.FALSE.                                                           SOLVE570
   30 CONTINUE                                                                  SOLVE580
      IF (BIGEST.LT.TWOEPS) RETURN                                              SOLVE590
      IF (SMALL.LE.TOL) GO TO 40                                                SOLVE600
      STOP=.TRUE.                                                               SOLVE610
      TOOBIG=.TRUE.                                                             SOLVE620
```

```
      RETURN                                                            SOLVE630
   40 IF (ABS(BIGER-BIGEST).LT.EPS) RETURN                              SOLVE640
C                                                                       SOLVE650
C  AT THIS POINT IT IS STILL NECESSARY TO INSERT THE LOCATION OF THE    SOLVE660
C  GRIDPOINT RESULTING IN THE LARGEST ERROR INTO LCTNX.  THERE ARE THREE SOLVE670
C  CASES, EACH OF WHICH IS HANDLED SEPARATELY--ALTERNATION IS PRESERVED SOLVE680
C  AT THE GRIDPOINTS.                                                   SOLVE690
C                                                                       SOLVE700
      STOP=.FALSE.                                                      SOLVE710
      YBIGST=YTABLE(FRSTM1+LNBGST)                                      SOLVE720
      YIBIG=YTABLE(LCTNX(INBGST)+FRSTM1)                                SOLVE730
      IF ((INBGST.EQ.1).AND.(YBIGST.LT.YIBIG)) GO TO 50                 SOLVE740
      IF ((INBGST.EQ.NX).AND.(YBIGST.GT.YIBIG)) GO TO 70                SOLVE750
      IF (YBIGST.LT.YIBIG) NEWINT=INBGST-1                              SOLVE760
      IF (YBIGST.GT.YIBIG) NEWINT=INBGST+1                              SOLVE770
      LCTNX(NEWINT)=LNBGST                                              SOLVE780
      RETURN                                                            SOLVE790
   50 DO 60 I=2,NX                                                      SOLVE800
         J=NX-I+2                                                       SOLVE810
         LCTNX(J)=LCTNX(J-1)                                            SOLVE820
   60 CONTINUE                                                          SOLVE830
      LCTNX(1)=LNBGST                                                   SOLVE840
      RETURN                                                            SOLVE850
   70 DO 80 J=1,NXM1                                                    SOLVE860
         LCTNX(J)=LCTNX(J+1)                                            SOLVE870
   80 CONTINUE                                                          SOLVE880
      LCTNX(NX)=LNBGST                                                  SOLVE890
      RETURN                                                            SOLVE900
C                                                                       SOLVE910
      END                                                               SOLVE920


      FUNCTION BPOLY(XX,C,N)                                            BPOLY 10
C                                                                       BPOLY 20
C  THIS FUNCTION IS USED TO EVALUATE (BY THE APPROPRIATE ADAPTATION     BPOLY 30
C  OF HORNERS METHOD) THE POLYNOMIAL                                    BPOLY 40
C                                                                       BPOLY 50
C     C(1) + C(2)*(XX-X(1)) + C(3)*(XX-X(1))*(XX-X(2)) + . . .          BPOLY 60
C       + C(N)*(XX-X(1))*(XX-X(2))* . . . *(XX-X(N-1))                  BPOLY 70
C                                                                       BPOLY 80
      DIMENSION C(18)                                                   BPOLY 90
      COMMON /DDIF/ DUMMY(5),X(18)                                      BPOLY100
C                                                                       BPOLY110
      NM1=N-1                                                           BPOLY120
      BPOLY=C(N)                                                        BPOLY130
      DO 10 I=1,NM1                                                     BPOLY140
         J=N-I                                                          BPOLY150
         BPOLY=C(J)+(XX-X(J))*BPOLY                                     BPOLY160
   10 CONTINUE                                                          BPOLY170
      RETURN                                                            BPOLY180
C                                                                       BPOLY190
      END                                                               BPOLY200


      SUBROUTINE TRANS(C,N)                                             TRANS 10
C                                                                       TRANS 20
C  THIS SUBROUTINE TRANSFORMS A POLYNOMIAL WRITTEN IN THE FORM          TRANS 30
C                                                                       TRANS 40
C     C(1) + C(2)*(X-X(1)) + C(3)*(X-X(1))*(X-X(2)) + . . .            TRANS 50
C       + C(N)*(X-X(1))*(X-X(2))* . . . *(X-X(N-1))                     TRANS 60
C                                                                       TRANS 70
C  TO A POLYNOMIAL WRITTEN IN TERMS OF POWERS OF X.  THE X(I)-S         TRANS 80
```

```
C  ARE SUPPLIED BY SUBROUTINE DIVDIF.                                    TRANS 90
C                                                                       TRANS100
      DIMENSION C(18)                                                   TRANS110
      COMMON /DDIF/ DUMMY(5),X(18)                                      TRANS120
C                                                                       TRANS130
      NM1=N-1                                                           TRANS140
      DO 20 J=1,NM1                                                     TRANS150
         K=N-J                                                          TRANS160
         DO 10 I=K,NM1                                                  TRANS170
            C(I)=C(I)-X(K)*C(I+1)                                       TRANS180
   10    CONTINUE                                                       TRANS190
   20 CONTINUE                                                          TRANS200
      RETURN                                                            TRANS210
C                                                                       TRANS220
      END                                                               TRANS230


      FUNCTION EVAL(X)                                                  EVAL  10
C                                                                       EVAL  20
C  THIS FUNCTION EVALUATES THE PIECEWISE POLYNOMIAL APPROXIMATION       EVAL  30
C  AT ANY POINT IN THE ENTIRE INTERVAL.                                 EVAL  40
C                                                                       EVAL  50
      COMMON DUMMY(1002),CSTORE(18,60),DUM(300)                         EVAL  60
      COMMON /SCALAR/ N,NPLUS1,DUM2(5),NINT                             EVAL  70
C                                                                       EVAL  80
      IF (NINT.LT.2) GO TO 20                                           EVAL  90
      DO 10 I=2,NINT                                                    EVAL 100
         ISTORE=I-1                                                     EVAL 110
         IF (X.LE.CSTORE(NPLUS1,I)) GO TO 30                            EVAL 120
   10 CONTINUE                                                          EVAL 130
      ISTORE=NINT                                                       EVAL 140
      GO TO 30                                                          EVAL 150
   20 ISTORE=1                                                          EVAL 160
   30 EVAL=HORNER(CSTORE(1,ISTORE),X,N)                                 EVAL 170
      RETURN                                                            EVAL 180
C                                                                       EVAL 190
      END                                                               EVAL 200


      FUNCTION HORNER(C,X,N)                                            HORNER10
C                                                                       HORNER20
C  THIS FUNCTION EVALUATES A POLYNOMIAL IN STANDARD FORM BY HORNERS     HORNER30
C  METHOD.                                                              HORNER40
C                                                                       HORNER50
      DIMENSION C(N)                                                    HORNER60
C                                                                       HORNER70
      HORNER=C(N)                                                       HORNER80
      I=N                                                               HORNER90
   10 IF (I.LT.2) RETURN                                                HORNE100
      HORNER=HORNER*X+C(I-1)                                            HORNE110
      I=I-1                                                             HORNE120
      GO TO 10                                                          HORNE130
C                                                                       HORNE140
      END                                                               HORNE150
```

UNIFORM ADAPTIVE CURVE FITTING PROGRAM : SAMPLE RUN

INPUT :

```
          6     2     2.50        ( N, SMTH, TOL )

        3.0          0.0          (XTABLE, FTABLE)
        5.0          1.3
        7.0          3.4
       11.0          5.2
       13.0          6.0
       15.0         14.4
       17.5         21.4
       20.0         27.4
       22.5         50.9
       25.0         49.3
       27.5         47.5
       30.0         51.5
       35.0         36.5
       40.0         27.9
       50.0          9.4
       60.0          4.2
```

OUTPUT :

```
     INTERVAL NUMBER  1 WHICH BEGINS AT  .3000000000000000E+01
AND ENDS AT  .2190452261306518E+02  CONTAINS  67 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

          C( 1) = -.1634025000676115E+02
          C( 2) =  .1181130531149097E+02
          C( 3) = -.2637146272918883E+01
          C( 4) =  .2593743838385461E+00
          C( 5) = -.1129567576998175E-01
          C( 6) =  .1865450590541398E-03

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS  .1969452589748130E+01.


     INTERVAL NUMBER  2 WHICH BEGINS AT  .2190452261306518E+02
AND ENDS AT  .2706030150753747E+02  CONTAINS  19 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

          C( 1) =  .7960326211065352E+06
          C( 2) = -.1656777306495747E+06
          C( 3) =  .1375385536550829E+05
          C( 4) = -.5692778474236584E+03
          C( 5) =  .1174880789994893E+02
          C( 6) = -.9672838852406418E-01

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS  .2294874459374114E+01.


     INTERVAL NUMBER  3 WHICH BEGINS AT  .2706030150753747E+02
AND ENDS AT  .2849246231155757E+02  CONTAINS   6 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE
```

C(1) = -.4555522308418542E+07
C(2) = .7815330984931737E+06
C(3) = -.5356190391737269E+05
C(4) = .1833051913127594E+04
C(5) = -.3132567622722183E+02
C(6) = .2138556370651710E+00

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .2350077767167932E+01.

INTERVAL NUMBER 4 WHICH BEGINS AT .2849246231155757E+02
AND ENDS AT .3078391959798978E+02 CONTAINS 9 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

C(1) = .1040230085838158E+07
C(2) = -.1649042600538041E+06
C(3) = .1043144021371787E+05
C(4) = -.3291270419910106E+03
C(5) = .5179689635444362E+01
C(6) = -.3252942438648798E-01

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .2350077744816190E+01.

INTERVAL NUMBER 5 WHICH BEGINS AT .3078391959798978E+02
AND ENDS AT .3794472361809017E+02 CONTAINS 26 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

C(1) = -.2329820690203374E+06
C(2) = .3289571488642064E+05
C(3) = -.1849503807458532E+04
C(4) = .5177682249693407E+02
C(5) = -.7218440333123937E+00
C(6) = .4009674821594894E-02

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .2347864915846003E+01.

INTERVAL NUMBER 6 WHICH BEGINS AT .3794472361809017E+02
AND ENDS AT .4310050251256257E+02 CONTAINS 19 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

C(1) = -.6827434677450825E+05
C(2) = .7613513411310967E+04
C(3) = -.3373191291063522E+03
C(4) = .7428809770578283E+01
C(5) = -.8135813032576111E-01
C(6) = .3545453066058774E-03

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .2416020625049441E+01.

INTERVAL NUMBER 7 WHICH BEGINS AT .4310050251256257E+02
AND ENDS AT .5999999999999955E+02 CONTAINS 60 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

C(1) = .1333616589900339E+05
C(2) = -.1192965606757301E+04
C(3) = .4193870667632291E+02
C(4) = -.7183879075542343E+00
C(5) = .5934796153467731E-02
C(6) = -.186138851603458E-04

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .2416020624874818E+01.

```
      PROGRAM DRIVER(INPUT,TAPE5=INPUT,OUTPUT,TAPE6=OUTPUT)                    DRIVER10
      COMMON XTABLE(500),FTABLE(500),DUMMY(1382)                               DRIVER20
      INTEGER SMTH                                                             DRIVER30
      READ(5,20)N,SMTH,TOL                                                     DRIVER40
      WRITE(6,30)                                                              DRIVER50
      WRITE(6,40)N,SMTH,TOL                                                    DRIVER60
      MAXNUM=0                                                                 DRIVER70
   10 MAXNUM=MAXNUM+1                                                          DRIVER80
      READ(5,50)XTABLE(MAXNUM),FTABLE(MAXNUM)                                  DRIVER90
      IF(EOF(5).EQ.0.0)GO TO 10                                                DRIVE100
      MAXNUM=MAXNUM-1                                                          DRIVE110
      CALL LINEAR(MAXNUM,400)                                                  DRIVE120
      MAXNUM=400                                                               DRIVE130
      CALL L2ACF(TOL,N,SMTH,MAXNUM,ERROR)                                      DRIVE140
C                                                                              DRIVE150
   20 FORMAT(2I5,F10.5)                                                        DRIVE160
   30 FORMAT(1H1,5X,48HPIECEWISE POLYNOMIAL APPROXIMATION--L2 OPERATOR.)       DRIVE170
   40 FORMAT(1H0,5X,17HNUMBER OF COEFFS=,I2,1H,,I2,25H CONTINUOUS DERIVS       DRIVE180
     1, TOL= ,F7.5)                                                            DRIVE190
   50 FORMAT(2F10.5)                                                           DRIVE200
C                                                                              DRIVE210
      END                                                                      DRIVE220

      SUBROUTINE LINEAR(OLDMAX,NEWMAX)                                         LINEAR10
C                                                                              LINEAR20
C  THIS SUBROUTINE FILLS IN BETWEEN THE ORIGINAL DATA POINTS BY LINEAR         LINEAR30
C  INTERPOLATION WHEN THERE ARE TOO FEW POINTS FOR THE EFFECTIVE USE           LINEAR40
C  OF THE ALGORITHM.  OLDMAX IS THE NUMBER OF ORIGINAL DATA POINTS.            LINEAR50
C  NEWMAX IS THE TOTAL NUMBER OF INTERPOLATED DATA POINTS DESIRED.             LINEAR60
C  THE NEWMAX POINTS ARE EQUALLY SPACED BETWEEN XTABLE(1) AND                  LINEAR70
C  XTABLE(OLDMAX).  DUE TO THIS FACT, THE ORIGINAL DATA POINTS MAY NOT         LINEAR80
C  BE IN XTABLE AND FTABLE UPON COMPLETION OF THIS ROUTINE.  THUS,             LINEAR90
C  NEWMAX MUST BE CHOSEN SUFFICIENTLY LARGE (LESS THAN OR EQUAL TO 500)        LINEA100
C  IN ORDER TO RETAIN THE PROXIMITIES OF THE ORIGINAL DATA POINTS.             LINEA110
C                                                                              LINEA120
      INTEGER OLDMAX                                                           LINEA130
      COMMON XTABLE(500),FTABLE(1582),X(150),Y(150)                            LINEA140
C                                                                              LINEA150
      DO 10 I=1,OLDMAX                                                         LINEA160
         X(I)=XTABLE(I)                                                        LINEA170
         Y(I)=FTABLE(I)                                                        LINEA180
   10 CONTINUE                                                                 LINEA190
      DELTA=(X(OLDMAX)-X(1))/FLOAT(NEWMAX-1)                                   LINEA200
      K=1                                                                      LINEA210
      DO 40 I=2,NEWMAX                                                         LINEA220
         XX=XTABLE(1)+FLOAT(I-1)*DELTA                                         LINEA230
         XTABLE(I)=XX                                                          LINEA240
   20    IF (XX.LE.X(K+1)) GO TO 30                                            LINEA250
         K=K+1                                                                 LINEA260
         GO TO 20                                                              LINEA270
   30    FTABLE(I)=Y(K)+(XX-X(K))*(Y(K+1)-Y(K))/(X(K+1)-X(K))                  LINEA280
   40 CONTINUE                                                                 LINEA290
      RETURN                                                                   LINEA300
C                                                                              LINEA310
      END                                                                      LINEA320

      SUBROUTINE L2ACF(TOL,N,SMTH,MAXNUM,ERROR)                                L2ACF 10
C                                                                              L2ACF 20
C  THIS SUBROUTINE ADAPTIVELY COMPUTES A PIECEWISE POLYNOMIAL APPROA-          L2ACF 30
C  IMATION OF DEGREE N-1 TO THE FUNCTION STORED IN THE ARRAYS XTABLE           L2ACF 40
```

```
C  AND FTABLE WITH SMTH CONTINUOUS DERIVATIVES.  THE APPROXIMATION DEV-      L2ACF 50
C  IATES FROM THIS FUNCTION BY NO MORE THAN TOL AT ANY POINT IN XTABLE.      L2ACF 60
C                                                                           L2ACF 70
C  THE PARAMETERS ARE AS FOLLOWS--                                          L2ACF 80
C                                                                           L2ACF 90
C     N - THE NUMBER OF COEFFICIENTS OF EACH POLYNOMIAL PIECE--IE. ONE      L2ACF100
C        MORE THAN THE DEGREE OF THE PIECEWISE POLYNOMIAL APPROXIMATION.    L2ACF110
C        N MUST BE GREATER THAN 1, AND AS THE ARRAYS ARE CURRENTLY          L2ACF120
C        DIMENSIONED, IT IS ASSUMED THAT N IS NO BIGGER THAN 17.            L2ACF130
C                                                                           L2ACF140
C     SMTH - THE NUMBER OF CONTINUOUS DERIVATIVES DESIRED OF THE APPROX-    L2ACF150
C        IMATION.  SMTH MUST NOT BE GREATER THAN N-2.  IF ONLY CONTIN-      L2ACF160
C        UITY OF THE APPROXIMATION IS DESIRED, SET SMTH = 0.  AN APPROX-    L2ACF170
C        IMATION WHICH IS DISCONTINUOUS AT THE KNOTS MAY BE OBTAINED BY     L2ACF180
C        SETTING SMTH = -1.                                                 L2ACF190
C                                                                           L2ACF200
C     MAXNUM - THE NUMBER OF POINTS ACTUALLY STORED IN THE ARRAYS XTABLE    L2ACF210
C        AND FTABLE.  AS THESE ARRAYS ARE CURRENTLY DIMENSIONED, MAXNUM     L2ACF220
C        MUST BE LESS THAN OR EQUAL TO 500.                                 L2ACF230
C                                                                           L2ACF240
C     TOL - THE TOLERANCE THAT THE APPROXIMATION MUST SATISFY.              L2ACF250
C                                                                           L2ACF260
C     ERROR - A LOGICAL VALUE SET TO .TRUE. IF AN ERROR OCCURS IN THE       L2ACF270
C        PROGRAM (AN APPROPRIATE ERROR MESSAGE WILL ALSO BE PRINTED)        L2ACF280
C        AND SET TO .FALSE. OTHERWISE.                                      L2ACF290
C                                                                           L2ACF300
C  THE COEFFICIENTS AND SUBINTERVAL ENDPOINTS ARE PRINTED OUT AS THEY       L2ACF310
C  ARE COMPUTED.  ALSO THE FUNCTION EVAL IS AVAILABLE TO THE USER TO        L2ACF320
C  EVALUATE THE APPROXIMATION AT ANY POINT WITHIN THE ENTIRE INTERVAL.      L2ACF330
C                                                                           L2ACF340
      LOGICAL LAST,ERROR,DONE,ABORT                                         L2ACF350
      INTEGER SMTH                                                          L2ACF360
      DIMENSION C(18)                                                       L2ACF370
      COMMON XTABLE(1000),LCTNLE,LCTNRE,DUM1(1380)                          L2ACF380
      COMMON/SCALAR/NPLUS0,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT         L2ACF390
      DATA MAXINT/150/,NRSMTH/-1/                                           L2ACF400
C                                                                           L2ACF410
      ERROR=.FALSE.                                                         L2ACF420
      DONE=.FALSE.                                                          L2ACF430
      ABORT=.FALSE.                                                         L2ACF440
      NPLUS0=N                                                              L2ACF450
      NPLUS1=N+1                                                            L2ACF460
C                                                                           L2ACF470
C  IN THE FIRST SUBINTERVAL THERE ARE NO INTERPOLATORY CONSTRAINTS--         L2ACF480
C  CONSEQUENTLY, WE SET NLSMTH=-1.                                          L2ACF490
C                                                                           L2ACF500
      NLSMTH=-1                                                             L2ACF510
      NX=N-2-NLSMTH-NRSMTH                                                  L2ACF520
      NXM1=NX-1                                                             L2ACF530
      LNGTH=NX+1                                                            L2ACF540
C                                                                           L2ACF550
C  WE INITIALLY TRY AS MUCH OF THE CURRENT REMAINING PORTION OF THE         L2ACF560
C  WHOLE INTERVAL AS POSSIBLE AS AN INITIAL GUESS FOR EACH SUCCESSIVE       L2ACF570
C  SUBINTERVAL.  LCTNLE IS THE LOCATION (IN THE ARRAY XTABLE) OF THE        L2ACF580
C  LEFT ENDPOINT OF THE CURRENT SUBINTERVAL, LCTNRE IS THE LOCATION         L2ACF590
C  OF THE RIGHT ENDPOINT.                                                   L2ACF600
C                                                                           L2ACF610
      LCTNLE=1                                                              L2ACF620
      LCTNRE=MINO(MAXNUM,MAXINT)                                            L2ACF630
      DO 20 INTNUM=1,60                                                     L2ACF640
         NINT=INTNUM                                                        L2ACF650
C                                                                           L2ACF660
```

```
C  SUBROUTINE COMPUT FINDS THE LARGEST SUBINTERVAL OF (XTABLE(LCTNLE),          L2ACF670
C  XTABLE(LCTNRE)) WITH LEFT ENDPOINT XTABLE(LCTNLE) SUCH THAT THE BEST        L2ACF680
C  APPROXIMATION ON THIS SUBINTERVAL SATISFIES ALL THE CONSTRAINTS.             L2ACF690
C  THE RIGHT ENDPOINT IS -BACKED OFF- TO THE LAST INTERIOR EXTREME             L2ACF700
C  POINT OF F-P TO ADD TO THE STABILITY OF THE ALGORITHM.  THE LOCATION        L2ACF710
C  OF THIS RIGHT ENDPOINT IS STORED IN LCTNRE.  IF LCTNRE=MAXNUM (I.E.,        L2ACF720
C  WE ARE DONE), CONTROL IS PASSED TO LINE 40.  IF NO SUCH SUBINTERVAL         L2ACF730
C  CAN BE FOUND, CONTROL IS PASSED TO LINE 50.  IF THERE ARE FEWER THAN        L2ACF740
C  LNGTH POINTS FROM LCTNRE TO MAXNUM, LAST IS SET TO .TRUE. AND THE           L2ACF750
C  SPECIAL CASE SUBROUTINE LSTINT IS CALLED.                                   L2ACF760
C                                                                              L2ACF770
      CALL COMPUT(C,TOL,LNGTH,MAXNUM,LAST,DONE,ABORT)                          L2ACF780
      IF (DONE) GO TO 40                                                       L2ACF790
      IF (ABORT) GO TO 50                                                      L2ACF800
      IF (INTNUM.GT.1) GO TO 10                                                L2ACF810
      NLSMTH=SMTH                                                              L2ACF820
      NX=N-2-NLSMTH-NRSMTH                                                     L2ACF830
      NXM1=NX-1                                                                L2ACF840
      LNGTH=NX+1                                                               L2ACF850
10    IF (LAST) GO TO 30                                                       L2ACF860
C                                                                              L2ACF870
C  SUBROUTINE STORE STORES THE COEFFICIENTS FOR THIS SUBINTERVAL IN THE        L2ACF880
C  ARRAY CSTORE.  IT ALSO PRINTS OUT THE COEFFICIENTS AND THE ERROR OF         L2ACF890
C  APPROXIMATION ON THIS SUBINTERVAL.  SUBROUTINE SETP(C,X,K) STORES THE       L2ACF900
C  VALUE OF THE POLYNOMIAL DETERMINED BY THE COEFFICIENTS IN THE ARRAY         L2ACF910.
C  C AND ITS FIRST K DERIVATIVES AT THE POINT X IN THE ARRAY PPRIME.           L2ACF920
C                                                                              L2ACF930
      CALL STORE(C,LCTNLE,LCTNRE)                                              L2ACF940
      CALL SETP(C,XTABLE(LCTNRE),NLSMTH)                                       L2ACF950
      LCTNLE=LCTNRE                                                            L2ACF960
      LCTNRE=MIN0(MAXNUM,MAXINT+LCTNLE-1)                                      L2ACF970
20 CONTINUE                                                                    L2ACF980
   WRITE (6,60) NINT                                                           L2ACF990
   ERROR=.TRUE.                                                                L2AC1000
   RETURN                                                                      L2AC1010
30 CALL LSTINT (C,TOL,LNGTH,MAXNUM,ABORT)                                      L2AC1020
   IF (ABORT) GO TO 50                                                         L2AC1030
40 CALL STORE (C,LCTNLE,LCTNRE)                                                L2AC1040
   RETURN                                                                      L2AC1050
50 WRITE (6,70)                                                                L2AC1060
   ERROR=.TRUE.                                                                L2AC1070
   RETURN                                                                      L2AC1080
C                                                                              L2AC1090
60 FORMAT (1H0,39(2H* ),1H*,/,2H0*,12X, 37HTHIS APPROXIMATION REQUIRE          L2AC1100
  1S MORE THAN,I3, 13H SUBINTERVALS,12X,1H*,/,2H0*,28X, 20H--PROGRAM           L2AC1110
  2ABORTING--,29X,1H*,/,1H0,39(2H* ),1H*)                                      L2AC1120
70 FORMAT (1H0,39(2H* ),1H*,/,2H0*,15X, 46HTHE ALGORITHM CANNOT MEET           L2AC1130
  1THE DESIRED ACCURACY,16X,1H*,/,2H0*,28X, 20H--PROGRAM ABORTING--,2          L2AC1140
  29X,1H*,/,1H0,39(2H* ),1H*)                                                  L2AC1150
C                                                                              L2AC1160
   END                                                                         L2AC1170


   SUBROUTINE STORE(C,LCTNLE,LCTNRE)                                           STORE 10
C                                                                              STORE 20
C  THIS SUBROUTINE OUTPUTS THE COEFFICIENTS AND ENDPOINTS OF THE CUR-          STORE 30
C  RENT APPROXIMATION AND SUBINTERVAL.  APPROPRIATE INFORMATION IS             STORE 40
C  STORED IN THE ARRAY CSTORE TO ALLOW THE ENTIRE PIECEWISE POLYNOMIAL         STORE 50
C  APPROXIMATION TO BE EASILY EVALUATED AT ANY POINT BY THE FUNCTION           STORE 60
C  EVAL.                                                                       STORE 70
C                                                                              STORE 80
   DIMENSION C(18)                                                             STORE 90
```

```
      COMMON XTABLE(500),FTABLE(502),CSTORE(18,60),DUM1(300)                STORE100
      COMMON/SCALAR/N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT              STORE110
C                                                                           STORE120
      NUMPTS=LCTNRE-LCTNLE+1                                                STORE130
      WRITE (6,30) NINT,XTABLE(LCTNLE),XTABLE(LCTNRE),NUMPTS                STORE140
      WRITE (6,40) (I,C(I),I=1,N)                                           STORE150
      ERR=0.0                                                               STORE160
      DO 10 I=LCTNLE,LCTNRE                                                 STORE170
         TEMP=ABS(FTABLE(I)-HORNER(C,XTABLE(I),N))                          STORE180
         IF (TEMP.GT.ERR) ERR=TEMP                                          STORE190
   10 CONTINUE                                                              STORE200
      WRITE (6,50) ERR                                                      STORE210
      DO 20 I=1,N                                                           STORE220
   20 CSTORE(I,NINT)=C(I)                                                   STORE230
      CSTORE(NPLUS1,NINT)=XTABLE(LCTNLE)                                    STORE240
      RETURN                                                                STORE250
C                                                                           STORE260
   30 FORMAT (//,5X, 15HINTERVAL NUMBER,I4, 16H WHICH BEGINS AT,E23.16,/    STORE270
     1, 12H AND ENDS AT,E23.16,2X, 8HCONTAINS,I4, 8H POINTS.,/, 60H TH      STORE280
     2E COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE,/)           STORE290
   40 FORMAT ((10X, 2HC(,I2, 3H) =,E24.16))                                 STORE300
   50 FORMAT (/, 47H THE ERROR OF APPROXIMATION IN THIS INTERVAL IS,E24.     STORE310
     116, 1H.)                                                              STORE320
C                                                                           STORE330
      END                                                                   STORE340


      SUBROUTINE SETP(C,X,SMTH)                                             SETP  10
C                                                                           SETP  20
C  THIS SUBROUTINE APPROPRIATELY STORES IN THE ARRAY PPRIME THE VALUES      SETP  30
C  WHICH MUST BE INTERPOLATED TO GIVE THE PIECEWISE POLYNOMIAL THE          SETP  40
C  DESIRED SMOOTHNESS.                                                      SETP  50
C                                                                           SETP  60
      DIMENSION C(18)                                                       SETP  70
      COMMON /COMP/ CSTORE(18),B(150)                                       SETP  80
      COMMON/SCALAR/N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT              SETP  90
      COMMON /DDIF/ PPRIME(5),DUM1(18)                                      SETP 100
      INTEGER SMTH                                                          SETP 110
C                                                                           SETP 120
      DO 10 I=1,N                                                           SETP 130
   10 CSTORE(I)=C(I)                                                        SETP 140
      NDUMMY=N                                                              SETP 150
      I=0                                                                   SETP 160
   20 IF (I.GT.SMTH) RETURN                                                 SETP 170
      IF (I.EQ.0) GO TO 30                                                  SETP 180
      CALL DERIV(CSTORE,NDUMMY)                                             SETP 190
   30 PPRIME(I+1)=HORNER(CSTORE,X,NDUMMY)                                   SETP 200
      I=I+1                                                                 SETP 210
      GO TO 20                                                              SETP 220
C                                                                           SETP 230
      END                                                                   SETP 240


      SUBROUTINE LSTINT(C,TOL,LNGTH,MAXNUM,ABORT)                           LSTINT10
C                                                                           LSTINT20
C  THIS SUBROUTINE HANDLES THE SPECIAL CASE OF FINDING A SUBINTERVAL        LSTINT30
C  AND A BEST APPROXIMATION ON THAT SUBINTERVAL WHEN THERE ARE TOO          LSTINT40
C  FEW REMAINING POINTS FOR COMPUT TO WORK.                                 LSTINT50
C                                                                           LSTINT60
      INTEGER OLDLE,OLDRE                                                   LSTINT70
      LOGICAL TOOBIG,ABORT                                                  LSTINT80
      REAL C(18)                                                            LSTINT90
```

```
      COMMON X(1000),LCTNLE,LCTNRE,CSTORE(18,60)                        LSTIN100
      COMMON/SCALAR/N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT          LSTIN110
      COMMON /COMP/ LCTNX(18)                                           LSTIN120
C                                                                       LSTIN130
      DO 10 OLDLE=1,NPLUS1                                              LSTIN140
   10 CSTORE(OLDLE,NINT)=C(OLDLE)                                       LSTIN150
      OLDLE=LCTNLE                                                      LSTIN160
      OLDRE=LCTNRE                                                      LSTIN170
      LCTNRE=MAXNUM                                                     LSTIN180
      LCTNLE=MIN0(MAXNUM-LNGTH+1,(MAXNUM-OLDLE+1)/2)-1                  LSTIN190
   20 LCTNLE=LCTNLE+1                                                   LSTIN200
      IF (MAXNUM-LCTNLE+1.LT.LNGTH) GO TO 40                            LSTIN210
      CALL SETP(CSTORE(1,NINT),X(LCTNLE),NLSMTH)                        LSTIN220
      NUMPTS=LCTNRE-LCTNLE+1                                            LSTIN230
      CALL ASET(LCTNLE,LCTNRE,NLSMTH,N,C)                               LSTIN240
      CALL HOUSE(C(NLSMTH+2),TOL)                                       LSTIN250
      CALL TOLCHK(TOL,TOOBIG)                                           LSTIN260
      IF(TOOBIG)GO TO 20                                                LSTIN270
      CALL FIX(C,X(LCTNLE),N)                                           LSTIN280
      CALL STORE(CSTORE(1,NINT),OLDLE,LCTNLE)                           LSTIN290
      NINT=NINT+1                                                       LSTIN300
      DO 30 OLDLE=1,NPLUS1                                              LSTIN310
   30 CSTORE(OLDLE,NINT)=C(OLDLE)                                       LSTIN320
      RETURN                                                            LSTIN330
   40 ABORT=.TRUE.                                                      LSTIN340
      RETURN                                                            LSTIN350
C                                                                       LSTIN360
      END                                                               LSTIN370


      SUBROUTINE COMPUT(C,TOL,LNGTH,MAXNUM,LAST,DONE,ABORT)             COMPUT10
C                                                                       COMPUT20
C  THIS SUBROUTINE FINDS THE LARGEST SUBINTERVAL AND THE BEST APPROX-   COMPUT30
C  IMATION TO F ON THIS SUBINTERVAL SUCH THAT THE APPROXIMATION MEETS   COMPUT40
C  THE DESIRED ERROR TOLERANCE ON THE SUBINTERVAL.                      COMPUT50
C                                                                       COMPUT60
      INTEGER A,B                                                       COMPUT70
      LOGICAL LAST,OK,DONE,ABORT,TOOBIG                                 COMPUT80
      REAL C(18)                                                        COMPUT90
      COMMON XTABLE(1000),LCTNLE,LCTNRE,CSTORE(18,60),CDERIV(300)       COMPU100
      COMMON/SCALAR/N,NPLUS1,NX,NXM1,NLSMTH,NRSMTH,NUMPTS,NINT          COMPU110
      COMMON/COMP/LCTNX(18),BB(150)                                     COMPU120
C                                                                       COMPU130
C  WE ASSUME THAT WE ARE CLOSE ENOUGH TO THE TRUE LARGEST SUBINTERVAL   COMPU140
C  RIGHT ENDPOINT WHEN WE KNOW THAT OUR APPROXIMATION TO THE TRUE RIGHT COMPU150
C  ENDPOINT IS WITHIN ETA OF THE TRUE ENDPOINT.                         COMPU160
C                                                                       COMPU170
      DATA ETA/.08/                                                     COMPU180
C                                                                       COMPU190
      LITTLE=LCTNLE+LNGTH-1                                             COMPU200
      A=0                                                               COMPU210
      LAST=.FALSE.                                                      COMPU220
      CALL ASET(LCTNLE,LCTNRE,NLSMTH,N,C)                               COMPU230
   10 NUMPTS=LCTNRE-LCTNLE                                              COMPU240
C                                                                       COMPU250
C  IF THE ACCURACY OF THE BEST APPROXIMATION ON THE CURRENT SUBINTERVAL  COMPU260
C  EXCEEDS TOL, CONTROL IS PASSED TO LINE 30.                           COMPU270
C                                                                       COMPU280
      CALL HOUSE(C(NLSMTH+2),TOL)                                       COMPU290
      CALL TOLCHK(TOL,TOOBIG)                                           COMPU300
      IF(TOOBIG)GO TO 30                                                COMPU310
      IF(LCTNRE.LT.MAXNUM)GO TO 20                                      COMPU320
```

```
      CALL FIX(C,XTABLE(LCTNLE),N)                                          COMPU330
      DONE=.TRUE.                                                           COMPU340
      RETURN                                                                COMPU350
C                                                                           COMPU360
C  A IS THE CURRENT LARGEST LOCATION FOR A RIGHT ENDPOINT SUCH THAT THE     COMPU370
C  BEST APPROXIMATION ON THIS SUBINTERVAL SATISFIES ALL CONSTRAINTS.        COMPU380
C                                                                           COMPU390
   20 A=LCTNRE                                                              COMPU400
      IF ((XTABLE(B)-XTABLE(A).LE.ETA).OR.(B-A.LE.1)) GO TO 50              COMPU410
      GO TO 40                                                              COMPU420
C                                                                           COMPU430
C  B IS THE CURRENT SMALLEST LOCATION FOR A RIGHT ENDPOINT SUCH THAT        COMPU440
C  THE BEST APPROXIMATION ON THIS SUBINTERVAL FAILS TO SATISFY THE CON-     COMPU450
C  STRAINTS.                                                                COMPU460
C                                                                           COMPU470
   30 B=LCTNRE                                                              COMPU480
   40 NEWTRY=(A+B)/2+1                                                      COMPU490
      IF (NEWTRY.EQ.B) NEWTRY=NEWTRY-1                                      COMPU500
      IF(NEWTRY.LT.LITTLE) NEWTRY=LITTLE                                    COMPU510
      IF (NEWTRY.EQ.LCTNRE) GO TO 50                                        COMPU520
      LCTNRE=NEWTRY                                                         COMPU530
      GO TO 10                                                              COMPU540
C                                                                           COMPU550
C  IF A IS STILL 0, THEN NO SUBINTERVAL WITH AT LEAST LNGTH POINTS          COMPU560
C  WILL WORK. SO THE ALGORITHM IS TERMINATED.                               COMPU570
C                                                                           COMPU580
   50 IF(A.NE.0)GO TO 60                                                    COMPU590
      ABORT=.TRUE.                                                          COMPU600
      RETURN                                                                COMPU610
C                                                                           COMPU620
C  SINCE NEWTRY IS ALWAYS STRICTLY LESS THAN THE CURRENT B, IF NEWTRY=      COMPU630
C  LCTNRE, AND A IS NOT STILL 0, NEWTRY=A, WHICH IS A POINT WHICH SAT-      COMPU640
C  ISFIES ALL REQUIREMENTS.  WE NOW BACK THE RIGHT ENDPOINT OFF TO THE      COMPU650
C  BEST INTERIOR EXTREME POINT OF F-P TO ADD TO THE STABILITY OF THE        COMPU660
C  ALGORITHM.                                                               COMPU670
C                                                                           COMPU680
   60 CALL FIX(C,XTABLE(LCTNLE),N)                                          COMPU690
      DO 70 I=1,N                                                           COMPU700
         CDERIV(I)=C(I)                                                     COMPU710
         CSTORE(I,NINT)=C(I)                                                COMPU720
   70 CONTINUE                                                              COMPU730
      NDUMMY=N                                                              COMPU740
      CALL DERIV(CDERIV,NDUMMY)                                             COMPU750
      I=NX                                                                  COMPU760
      LCTNRE=LCTNX(I)                                                       COMPU770
      NEWTRY=LCTNRE                                                         COMPU780
      SMLL=CMPR(CDERIV,NDUMMY,NEWTRY+LCTNLE-1,OK)                           COMPU790
      IF(OK)GO TO 100                                                       COMPU800
   80 I=I-1                                                                 COMPU810
      IF(I.EQ.0)GO TO 100                                                   COMPU820
      NEWTRY=LCTNX(I)                                                       COMPU830
      IF (NEWTRY.LT.LNGTH) GO TO 100                                        COMPU840
      TEMP=CMPR(CDERIV,NDUMMY,NEWTRY+LCTNLE-1,OK)                           COMPU850
      IF(.NOT.OK)GO TO 90                                                   COMPU860
      LCTNRE=NEWTRY                                                         COMPU870
      GO TO 100                                                             COMPU880
   90 IF(TEMP.GE.SMLL)GO TO 80                                              COMPU890
      SMLL=TEMP                                                             COMPU900
      LCTNRE=NEWTRY                                                         COMPU910
      GO TO 80                                                              COMPU920
  100 LCTNRE=LCTNLE+LCTNRE-1                                                COMPU930
      IF (MAXNUM-LCTNRE+1.LT.LNGTH) LAST=.TRUE.                             COMPU940
```

```
      RETURN                                                            COMPU950
C                                                                       COMPU960
      END                                                               COMPU970


      SUBROUTINE ASET(LCTNLE,LCTNRE,NLSMTH,N,C)                         ASET  10
C                                                                       ASET  20
C  THIS SUBROUTINE SETS UP THE LINEAR SYSTEM WHICH IS SOLVED IN THE     ASET  30
C  L2 SENSE IN ORDER TO OBTAIN THE COEFFICIENTS OF THE POLYNOMIAL       ASET  40
C  APPROXIMATIONS.                                                      ASET  50
C                                                                       ASET  60
      COMMON /HHOLDR/ A(150,10),BDATA(150,3)                            ASET  70
      COMMON /DDIF/ PPRIME(5),DUM1(18)                                  ASET  80
      COMMON XTABLE(500),FTABLE(500)                                    ASET  90
      DIMENSION C(10)                                                   ASET 100
C                                                                       ASET 110
      FAC=1.0                                                           ASET 120
      I=1                                                               ASET 130
      NM1=N-1                                                           ASET 140
      NSMP1=NLSMTH+1                                                    ASET 150
   10 IF (I.GT.NSMP1) GO TO 20                                          ASET 160
      C(I)=PPRIME(I)/FAC                                                ASET 170
      FAC=FLOAT(I)*FAC                                                  ASET 180
      I=I+1                                                             ASET 190
      GO TO 10                                                          ASET 200
   20 XSTAR=XTABLE(LCTNLE)                                              ASET 210
      K=LCTNLE-1                                                        ASET 220
      IF(NLSMTH.GE.0)K=K+1                                              ASET 230
      I=0                                                               ASET 240
   30 K=K+1                                                             ASET 250
      IF(K.GT.LCTNRE)GO TO 50                                           ASET 260
      I=I+1                                                             ASET 270
      XNEW=XTABLE(K)-XSTAR                                              ASET 280
      BDATA(I,1)=XNEW                                                   ASET 290
      IF(NLSMTH.LT.0)BDATA(I,2)=FTABLE(K)                               ASET 300
      IF(NLSMTH.GE.0)BDATA(I,2)=FTABLE(K)-HORNER(C,XNEW,NSMP1)          ASET 310
      XPOWER=XNEW**NSMP1                                                ASET 320
      J=1                                                               ASET 330
      JJ=NSMP1                                                          ASET 340
   40 IF(JJ.GT.NM1)GO TO 30                                             ASET 350
      A(I,J)=XPOWER                                                     ASET 360
      XPOWER=XNEW*XPOWER                                                ASET 370
      J=J+1                                                             ASET 380
      JJ=JJ+1                                                           ASET 390
      GO TO 40                                                          ASET 400
   50 IF(NLSMTH.LT.0)A(1,1)=1.0                                         ASET 410
      RETURN                                                            ASET 420
C                                                                       ASET 430
      END                                                               ASET 440


      SUBROUTINE HOUSE(C,TOL)                                           HOUSE 10
C                                                                       HOUSE 20
C  THIS SUBROUTINE SOLVES A GIVEN LINEAR SYSTEM IN THE L2 SENSE USING   HOUSE 30
C  HOUSEHOLDER TRANSFORMATIONS.  THOUGH SOMEWHAT LENGTHY, IT WAS THE    HOUSE 40
C  BEST L2 ROUTINE AVAILABLE WHEN THIS PROGRAM WAS DEVELOPED.  AN ADAP- HOUSE 50
C  TIVE CURVE FITTING PROGRAM WITH A MORE EFFICIENT L2 PACKAGE, AS WELL HOUSE 60
C  AS AN L1 PACKAGE, IS CURRENTLY BEING DEVELOPED BY PAUL G. AVILA AND  HOUSE 70
C  G. T. TAYLOR.                                                        HOUSE 80
C                                                                       HOUSE 90
      COMMON/COMP/LCTNX(18),B(150)                                      HOUSE100
      COMMON /HHOLDR/ A,BDATA                                           HOUSE110
```

```
      COMMON/SCALAR/DUM1(2),N,DUM2(3),M,DUM3                                    HOUSE120
      REAL MAX                                                                  HOUSE130
      DOUBLE PRECISION SUMM,X                                                   HOUSE140
      DIMENSION X(10),KPIVOT(10),D(10),BETA(10),BETABR(10),SAVE(10),            HOUSE150
     $DP(10),DAA(10),A(150,10),AA(150,10),BB(150),BDATA(150,3),C(18)            HOUSE160
C                                                                               HOUSE170
      KRANK=N                                                                   HOUSE180
      DO 10 J=1,N                                                               HOUSE190
         KPIVOT(J)=J                                                            HOUSE200
         DO 10 I=1,M                                                            HOUSE210
   10 AA(I,J)=A(I,J)                                                            HOUSE220
      DO 20 J=1,M                                                               HOUSE230
      B(J)=BDATA(J,2)                                                           HOUSE240
   20 BB(J)=BDATA(J,2)                                                          HOUSE250
      DO 130 K=1,N                                                              HOUSE260
         D(K)=0.0                                                               HOUSE270
         KCHNGE=K                                                               HOUSE280
         DO 40 JJ=K,N                                                           HOUSE290
            SUM=0.0                                                             HOUSE300
            DO 30 IA=K,M                                                        HOUSE310
   30       SUM=SUM+AA(IA,JJ)*AA(IA,JJ)                                         HOUSE320
            IF(D(K).GE.SUM)GO TO 40                                             HOUSE330
            KCHNGE=JJ                                                           HOUSE340
            D(K)=SUM                                                            HOUSE350
   40    CONTINUE                                                               HOUSE360
C                                                                               HOUSE370
C  KCHNGE CONTAINS THE INDEX OF THE COLUMN OF GREATEST                          HOUSE380
C  LENGTH BETWEEN K AND N.                                                      HOUSE390
C                                                                               HOUSE400
         IF(KCHNGE.EQ.K)GO TO 60                                                HOUSE410
C                                                                               HOUSE420
C  START COLUMN INTERCHANGE.                                                    HOUSE430
C                                                                               HOUSE440
         DO 50 I=1,M                                                            HOUSE450
            STORE=AA(I,KCHNGE)                                                  HOUSE460
            AA(I,KCHNGE)=AA(I,K)                                                HOUSE470
            AA(I,K)=STORE                                                       HOUSE480
   50    CONTINUE                                                               HOUSE490
         KK=KPIVOT(K)                                                           HOUSE500
         KPIVOT(K)=KPIVOT(KCHNGE)                                               HOUSE510
         KPIVOT(KCHNGE)=KK                                                      HOUSE520
   60    CONTINUE                                                               HOUSE530
         IF(K.EQ.1)GO TO 70                                                     HOUSE540
         MAX=ABS(D(1))                                                          HOUSE550
         TEST=(FLOAT(M-K+1)*1.0E-26)*(MAX*MAX)                                  HOUSE560
         IF(ABS(D(K)).GT.TEST)GO TO 70                                          HOUSE570
         D(K)=SQRT(D(K))                                                        HOUSE580
         KRANK=K-1                                                              HOUSE590
         GO TO 140                                                              HOUSE600
   70    CONTINUE                                                               HOUSE610
         AAKK=AA(K,K)                                                           HOUSE620
         SQDK=SQRT(D(K))                                                        HOUSE630
         IF(AAKK.LT.0.0)GO TO 80                                                HOUSE640
         BETA(K)=1.0/(D(K)+AAKK*SQDK)                                           HOUSE650
         AA(K,K)=SQDK+AAKK                                                      HOUSE660
         D(K)=-SQDK                                                             HOUSE670
         GO TO 90                                                               HOUSE680
   80    CONTINUE                                                               HOUSE690
         BETA(K)=1.0/(D(K)-AAKK*SQDK)                                           HOUSE700
         AA(K,K)=-SQDK+AAKK                                                     HOUSE710
         D(K)=SQDK                                                              HOUSE720
   90    CONTINUE                                                               HOUSE730
```

```
      KT=K+1                                                            HOUSE740
      IF(K.EQ.N)GO TO 120                                               HOUSE750
      DO 110 J=KT,N                                                     HOUSE760
         SAVE(J)=0.0                                                    HOUSE770
         DO 100 IA=K,M                                                  HOUSE780
  100    SAVE(J)=SAVE(J)+AA(IA,K)*AA(IA,J)                              HOUSE790
         DO 110 I=K,M                                                   HOUSE800
            AA(I,J)=AA(I,J)-AA(I,K)*SAVE(J)*BETA(K)                     HOUSE810
  110 CONTINUE                                                          HOUSE820
  120 CONTINUE                                                          HOUSE830
  130 CONTINUE                                                          HOUSE840
  140 CONTINUE                                                          HOUSE850
      DO 150 I=1,KRANK                                                  HOUSE860
      II=I+1                                                            HOUSE870
      IF(I.EQ.N)GO TO 160                                               HOUSE880
      DO 150 J=II,N                                                     HOUSE890
         AA(I,J)=AA(I,J)/D(I)                                           HOUSE900
  150 CONTINUE                                                          HOUSE910
  160 CONTINUE                                                          HOUSE920
C                                                                       HOUSE930
C  NOW ALL THE DIAGONAL ELEMENTS OF AA ARE 1 AND ALL OFF                HOUSE940
C  DIAGONAL ELEMENTS OF AA ARE LESS THAN OR EQUAL TO 1.                 HOUSE950
C  RECALL AA(S,S)=1.0 WHICH IS STORED NOW IN DAA(S).                    HOUSE960
C                                                                       HOUSE970
      DO 170 IS=1,KRANK                                                 HOUSE980
  170 DAA(IS)=1.0                                                       HOUSE990
      IF(KRANK.EQ.N)GO TO 240                                           HOUS1000
      DO 230 ISS=1,KRANK                                                HOUS1010
         IS=KRANK-ISS+1                                                 HOUS1020
         KKR=KRANK+1                                                    HOUS1030
         SUM=0.0                                                        HOUS1040
         DO 180 IA=KKR,N                                                HOUS1050
  180    SUM=SUM+AA(IS,IA)*AA(IS,IA)                                    HOUS1060
         SUM=SUM+1.0                                                    HOUS1070
         SQSUM=SQRT(SUM)                                                HOUS1080
         BETABR(IS)=1.0/(SUM+SQSUM)                                     HOUS1090
         DP(IS)=SQSUM+1.0                                               HOUS1100
         DAA(IS)=-SQSUM                                                 HOUS1110
C                                                                       HOUS1120
C  NOW W IS STORED IN ROW S FROM N-R+1 TO N WITH DIAGONAL               HOUS1130
C  ELEMENTS STORED IN DP(S).                                            HOUS1140
C  CALCULATE AA*W.                                                      HOUS1150
C                                                                       HOUS1160
         KKS=IS-1                                                       HOUS1170
         KSS=KRANK+1                                                    HOUS1180
         IF(IS.EQ.1)GO TO 220                                           HOUS1190
         DO 210 J=1,KKS                                                 HOUS1200
            SAVE(J)=0.0                                                 HOUS1210
            DO 190 IA=KSS,N                                             HOUS1220
  190       SAVE(J)=SAVE(J)+AA(IS,IA)*AA(J,IA)                          HOUS1230
            SAVE(J)=SAVE(J)+DP(IS)*AA(J,IS)                             HOUS1240
            AA(J,IS)=AA(J,IS)-DP(IS)*SAVE(J)*BETABR(IS)                 HOUS1250
            DO 200 I=KSS,N                                              HOUS1260
C                                                                       HOUS1270
C  NOTE...W(I) IS STORED AT AA(IS,I).                                   HOUS1280
C                                                                       HOUS1290
  200       AA(J,I)=AA(J,I)-AA(IS,I)*SAVE(J)*BETABR(IS)                 HOUS1300
  210    CONTINUE                                                       HOUS1310
  220    CONTINUE                                                       HOUS1320
  230 CONTINUE                                                          HOUS1330
  240 CONTINUE                                                          HOUS1340
      ICOUNT=1                                                          HOUS1350
```

```
      DO 250 I=1,N                                                      HOUS1360
  250 X(I)=0.0                                                          HOUS1370
  260 CONTINUE                                                          HOUS1380
C                                                                       HOUS1390
C  PREMULTIPLY BY THE HOUSEHOLDER TRANSFORMATIONS.                      HOUS1400
C                                                                       HOUS1410
      DO 290 I=1,KRANK                                                  HOUS1420
         SUM=0.0                                                        HOUS1430
         DO 270 IA=I,M                                                  HOUS1440
  270    SUM=SUM+AA(IA,I)*B(IA)                                         HOUS1450
         SUM=SUM*BETA(I)                                                HOUS1460
         DO 280 J=I,M                                                   HOUS1470
            B(J)=B(J)-AA(J,I)*SUM                                       HOUS1480
  280    CONTINUE                                                       HOUS1490
  290 CONTINUE                                                          HOUS1500
C                                                                       HOUS1510
C  NOW ONLY USE THE FIRST KRANK TERMS OF B.                             HOUS1520
C  CALCULATE (D INVERSE)*B .                                            HOUS1530
C                                                                       HOUS1540
      DO 300 I=1,KRANK                                                  HOUS1550
         B(I)=B(I)/D(I)                                                 HOUS1560
  300 CONTINUE                                                          HOUS1570
C                                                                       HOUS1580
C  NOW SOLVE (V INVERSE)*B .                                            HOUS1590
C                                                                       HOUS1600
      DO 320 II=1,KRANK                                                 HOUS1610
         I=KRANK+1-II                                                   HOUS1620
         B(I)=B(I)/DAA(I)                                               HOUS1630
         KK=I-1                                                         HOUS1640
         IF(I.EQ.1)GO TO 320                                            HOUS1650
         DO 310 J=1,KK                                                  HOUS1660
            B(J)=B(J)-AA(J,I)*B(I)                                      HOUS1670
  310    CONTINUE                                                       HOUS1680
  320 CONTINUE                                                          HOUS1690
      DO 340 I=1,N                                                      HOUS1700
         IF(I.LE.KRANK)GO TO 330                                        HOUS1710
         B(I)=0.0                                                       HOUS1720
  330    CONTINUE                                                       HOUS1730
  340 CONTINUE                                                          HOUS1740
      IF(KRANK.EQ.N)GO TO 380                                           HOUS1750
C                                                                       HOUS1760
C  MULTIPLY BY P FOR I=KRANK TO 1.                                      HOUS1770
C                                                                       HOUS1780
      KK=KRANK+1                                                        HOUS1790
      DO 370 I=1,KRANK                                                  HOUS1800
         SUM=0.0                                                        HOUS1810
         DO 350 IA=KK,N                                                 HOUS1820
  350    SUM=SUM+AA(I,IA)*B(IA)                                         HOUS1830
         SUM=(SUM+B(I)*DP(I))*BETABR(I)                                 HOUS1840
         B(I)=B(I)-DP(I)*SUM                                            HOUS1850
         DO 360 J=KK,N                                                  HOUS1860
            B(J)=B(J)-AA(I,J)*SUM                                       HOUS1870
  360    CONTINUE                                                       HOUS1880
  370 CONTINUE                                                          HOUS1890
  380 CONTINUE                                                          HOUS1900
C                                                                       HOUS1910
C  TEST FOR CONVERGENCE.                                                HOUS1920
C  FIRST TEST, TOO MANY ITERATIONS.                                     HOUS1930
C  SECOND TEST, SEE IF X IS DECREASING.                                 HOUS1940
C                                                                       HOUS1950
      SUM=0.0                                                           HOUS1960
      DO 390 I=1,N                                                      HOUS1970
```

```
  390 SUM=SUM+B(I)*B(I)                                                  HOUS1980
      IF(ICOUNT.EQ.1)GO TO 400                                           HOUS1990
      IF(SUM.LE..5*TEST)GO TO 410                                        HOUS2000
      ICOUNT=11                                                          HOUS2010
      GO TO 410                                                          HOUS2020
  400 TESTT=SUM                                                          HOUS2030
  410 TEST=SUM                                                           HOUS2040
      DO 420 I=1,N                                                       HOUS2050
         KP=KPIVOT(I)                                                    HOUS2060
         X(KP)=B(I)+X(KP)                                                HOUS2070
  420 CONTINUE                                                           HOUS2080
C                                                                        HOUS2090
C  CALCULATE A*X-B .                                                     HOUS2100
C                                                                        HOUS2110
      DO 440 I=1,M                                                       HOUS2120
         SUMM=0.0                                                        HOUS2130
         DO 430 J=1,N                                                    HOUS2140
  430    SUMM=SUMM+A(I,J)*X(J)                                           HOUS2150
  440 B(I)=BB(I)-SUMM                                                    HOUS2160
C                                                                        HOUS2170
C  THIRD TEST, WAS THE CORRECTION SIGNIFICANT.                           HOUS2180
C                                                                        HOUS2190
      IF(TEST.LT.1.E-30*TESTT)GO TO 470                                  HOUS2200
      IF(ICOUNT.EQ.5)GO TO 450                                           HOUS2210
      IF(ICOUNT.GE.6)GO TO 470                                           HOUS2220
      ICOUNT=ICOUNT+1                                                    HOUS2230
      GO TO 260                                                          HOUS2240
  450 CONTINUE                                                           HOUS2250
      WRITE(6,460)                                                       HOUS2260
  460 FORMAT(1X,35HCOMPLETED 10 ITERATIONS AND STOPPED)                  HOUS2270
  470 CONTINUE                                                           HOUS2280
      DO 480 J=1,N                                                       HOUS2290
  480 C(J)=X(J)                                                          HOUS2300
      RETURN                                                             HOUS2310
C                                                                        HOUS2320
      END                                                                HOUS2330


      SUBROUTINE TOLCHK(TOL,TOOBIG)                                      TOLCHK10
C                                                                        TOLCHK20
C  THIS SUBROUTINE CHECKS TO SEE IF THE CURRENT APPROXIMATION MEETS      TOLCHK30
C  THE SPECIFIED TOLERANCE.  IF NOT, WE RETURN TO COMPUT TO REDUCE THE   TOLCHK40
C  INTERVAL LENGTH.  IF TOL IS MET, THE EXTREME POINTS OF THE APPROX-    TOLCHK50
C  IMATION ARE STORED IN THE ARRAY LCTNX.                                TOLCHK60
C                                                                        TOLCHK70
      COMMON/COMP/LCTNX(18),B(150)                                       TOLCHK80
      COMMON/SCALAR/DUM1(2),N,DUM2(3),M,DUM3                             TOLCHK90
      REAL MAX                                                           TOLCH100
      LOGICAL TOOBIG                                                     TOLCH110
C                                                                        TOLCH120
      TOOBIG=.TRUE.                                                      TOLCH130
      MAX=0.0                                                            TOLCH140
      DO 10 I=1,M                                                        TOLCH150
      TEMP=ABS(B(I))                                                     TOLCH160
      IF (TEMP.GT.MAX) MAX=TEMP                                          TOLCH170
   10 CONTINUE                                                           TOLCH180
      IF(MAX.GT.TOL)RETURN                                               TOLCH190
      TOOBIG=.FALSE.                                                     TOLCH200
      BIG=ABS(B(M))                                                      TOLCH210
      K=M                                                                TOLCH220
      LCTN=M                                                             TOLCH230
      SGN=SIGN(1.0,B(M))                                                 TOLCH240
```

```
      DO 60 J=1,N                                                    TOLCH250
         I=N+1-J                                                     TOLCH260
   20    IF(K.LT.1)GO TO 30                                          TOLCH270
         TEMP=SGN*B(K)                                               TOLCH280
         IF(TEMP.GT.0.0)GO TO 40                                     TOLCH290
         SGN=SIGN(1.0,B(K))                                          TOLCH300
   30    LCTNX(I)=LCTN                                               TOLCH310
         BIG=-TEMP                                                   TOLCH320
         LCTN=K                                                      TOLCH330
         K=K-1                                                       TOLCH340
         GO TO 60                                                    TOLCH350
   40 IF(TEMP.LT.BIG)GO TO 50                                        TOLCH360
         LCTN=K                                                      TOLCH370
         BIG=TEMP                                                    TOLCH380
   50    K=K-1                                                       TOLCH390
         GO TO 20                                                    TOLCH400
   60 CONTINUE                                                       TOLCH410
      RETURN                                                         TOLCH420
C                                                                    TOLCH430
      END                                                            TOLCH440


      SUBROUTINE FIX(C,XSTAR,N)                                      FIX   10
C                                                                    FIX   20
C  THIS SUBROUTINE TAKES A POLYNOMIAL EXPRESSED IN NEWTONS FORM AND  FIX   30
C  FORMS THE COEFFICIENTS OF THAT SAME POLYNOMIAL IN STANDARD FORM.  FIX   40
C                                                                    FIX   50
      DIMENSION C(18)                                                FIX   60
C                                                                    FIX   70
      NM1=N-1                                                        FIX   80
      DO 20 J=1,NM1                                                  FIX   90
         K=N-J                                                       FIX  100
         DO 10 I=K,NM1                                               FIX  110
            C(I)=C(I)-XSTAR*C(I+1)                                   FIX  120
   10    CONTINUE                                                    FIX  130
   20 CONTINUE                                                       FIX  140
      RETURN                                                         FIX  150
C                                                                    FIX  160
      END                                                            FIX  170


      SUBROUTINE DERIV(C,N)                                          DERIV 10
C                                                                    DERIV 20
C  THIS SUBROUTINE REPLACES THE COEFFICIENTS OF A POLYNOMIAL IN STAND- DERIV 30
C  ARD FORM WITH THE COEFFICIENTS OF THE DERIVATIVE OF THIS POLYNOMIAL. DERIV 40
C  THE NUMBER OF COEFFICIENTS, N, IS DECREMENTED.                    DERIV 50
C                                                                    DERIV 60
      DIMENSION C(N)                                                 DERIV 70
C                                                                    DERIV 80
      N=N-1                                                          DERIV 90
      DO 10 I=1,N                                                    DERIV100
   10 C(I)=FLOAT(I)*C(I+1)                                           DERIV110
      RETURN                                                         DERIV120
      END                                                            DERIV140


      FUNCTION CMPR(C,N,NEWTRY,OK)                                   CMPR  10
C                                                                    CMPR  20
C  THIS SUBROUTINE COMPARES THE FIRST DERIVATIVE OF THE CURRENT PIECE OF CMPR  30
C  THE PIECEWISE POLYNOMIAL APPROXIMATION EVALUATED AT XTABLE(NEWTRY) CMPR  40
C  WITH THE FIRST DERIVATIVE OF THE QUADRATIC INTERPOLATION OF F, CEN- CMPR  50
C  TERED AROUND XTABLE(NEWTRY), EVALUATED AT XTABLE(NEWTRY).  IF THESE CMPR  60
```

```
C  TWO DIFFER IN ABSOLUTE VALUE BY LESS THAN TOLER (EITHER ABSOLUTELY          CMPR  70
C  OR RELATIVELY), WE SET OK TO .TRUE. AND WE ACCEPT XTABLE(NEWTRY) AS         CMPR  80
C  A REASONABLE SUBINTERVAL RIGHT ENDPOINT.                                    CMPR  90
C                                                                             CMPR 100
      LOGICAL OK                                                              CMPR 110
      COMMON X(500),F(500),DUM1(1382)                                         CMPR 120
      DIMENSION C(18)                                                         CMPR 130
      DATA TOLER/.05/                                                         CMPR 140
C                                                                             CMPR 150
      OK=.FALSE.                                                              CMPR 160
      A=(F(NEWTRY)-F(NEWTRY-1))/(X(NEWTRY)-X(NEWTRY-1))                       CMPR 170
      B=(F(NEWTRY+1)-F(NEWTRY))/(X(NEWTRY+1)-X(NEWTRY))                       CMPR 180
      D=(B-A)/(X(NEWTRY+1)-X(NEWTRY-1))                                       CMPR 190
      CMPR=A+D*(X(NEWTRY)-X(NEWTRY-1))                                        CMPR 200
      A=TOLER*CMPR                                                            CMPR 210
      CMPR=ABS(CMPR-HORNER(C,X(NEWTRY),N))                                    CMPR 220
      IF (CMPR.LE.A) OK=.TRUE.                                                CMPR 230
      RETURN                                                                  CMPR 240
C                                                                             CMPR 250
      END                                                                     CMPR 260


      FUNCTION EVAL(X)                                                        EVAL  10
C                                                                             EVAL  20
C  THIS FUNCTION EVALUATES THE PIECEWISE POLYNOMIAL APPROXIMATION             EVAL  30
C  AT ANY POINT IN THE ENTIRE INTERVAL.                                       EVAL  40
C                                                                             EVAL  50
      COMMON DUMMY(1002),CSTORE(18,60),DUM(300)                               EVAL  60
      COMMON/SCALAR/N,NPLUS1,DUM2(5),NINT                                     EVAL  70
C                                                                             EVAL  80
      IF(NINT.LT.2)GO TO 20                                                   EVAL  90
      DO 10 I=2,NINT                                                          EVAL 100
         ISTORE=I-1                                                           EVAL 110
         IF(X.LE.CSTORE(NPLUS1,I))GO TO 30                                    EVAL 120
   10 CONTINUE                                                                EVAL 130
      ISTORE=NINT                                                             EVAL 140
      GO TO 30                                                                EVAL 150
   20 ISTORE=1                                                                EVAL 160
   30 EVAL=HORNER(CSTORE(1,ISTORE),X,N)                                       EVAL 170
      RETURN                                                                  EVAL 180
C                                                                             EVAL 190
      END                                                                     EVAL 200


      FUNCTION HORNER(C,X,N)                                                  HORNER10
C                                                                             HORNER20
C  THIS FUNCTION EVALUATES A POLYNOMIAL IN STANDARD FORM BY HORNERS           HORNER30
C  METHOD.                                                                    HORNER40
C                                                                             HORNER50
      DIMENSION C(N)                                                          HORNER60
C                                                                             HORNER70
      HORNER=C(N)                                                             HORNER80
      I=N                                                                     HORNER90
   10 IF (I.LT.2) RETURN                                                      HORNE100
      HORNER=HORNER*X+C(I-1)                                                  HORNE110
      I=I-1                                                                   HORNE120
      GO TO 10                                                                HORNE130
C                                                                             HORNE140
      END                                                                     HORNE150
```

```
LEAST SQUARES ADAPTIVE CURVE FITTING PROGRAM : SAMPLE RUN

INPUT :
          6    2    3.00       ( N, SMTH, TOL )

          3.0        0.0       (XTABLE, FTABLE)
          5.0        1.3
          7.0        3.4
         11.0        5.2
         13.0        6.0
         15.0       14.4
         17.5       21.4
         20.0       27.4
         22.5       50.9
         25.0       49.3
         27.5       47.5
         30.0       51.5
         35.0       36.5
         40.0       27.9
         50.0        9.4
         60.0        4.2

OUTPUT :

     INTERVAL NUMBER  1 WHICH BEGINS AT  .3000000000000000E+01
AND ENDS AT  .2185714285714278E+02  CONTAINS 133 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

        C( 1) = -.318994274357244 2E+02
        C( 2) =  .1812103641220483E+02
        C( 3) = -.3560852823483785E+01
        C( 4) =  .3242284790553871E+00
        C( 5) = -.1350234902068514E-01
        C( 6) =  .2158257913951744E-03

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS  .287469301288797E+01.


     INTERVAL NUMBER  2 WHICH BEGINS AT  .2185714285714278E+01
AND ENDS AT  .2528571428571411E+02  CONTAINS  25 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

        C( 1) =  .7525331567081399E+06
        C( 2) = -.1568254831136037E+06
        C( 3) =  .1303544628337352E+05
        C( 4) = -.5402174434562294E+03
        C( 5) =  .1116296306156516E+02
        C( 6) = -.9202040460845584E-01

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS  .282447246302444 9E+01.


     INTERVAL NUMBER  3 WHICH BEGINS AT  .2528571428571411E+02
AND ENDS AT  .3557142857142844E+02  CONTAINS  73 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE
```

```
C( 1) =  .6867241093987040E+05
C( 2) = -.1121136988075014E+05
C( 3) =  .727774151993340 6E+03
C( 4) = -.2346667791438097E+02
C( 5) =  .3759555635732301E+00
C( 6) = -.2395099334129056E-02
```

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .2985086750638296E+01.

INTERVAL NUMBER 4 WHICH BEGINS AT .3557142857142844E+02
AND ENDS AT .5399999999999977E+02 CONTAINS 130 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

```
C( 1) = -.8446628095918451E+04
C( 2) =  .9702467401092472E+03
C( 3) = -.4389461668285185E+02
C( 4) =  .9842698198241457E+00
C( 5) = -.1096799498228268E-01
C( 6) =  .4863340401267810E-04
```

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .1464785127359050E+01.

INTERVAL NUMBER 5 WHICH BEGINS AT .5399999999999977E+02
AND ENDS AT .5999999999999977E+02 CONTAINS 43 POINTS.
THE COEFFICIENTS OF BEST APPROXIMATION IN THIS INTERVAL ARE

```
C( 1) = -.1771038129739963E+07
C( 2) =  .1591827758229449E+06
C( 3) = -.5718444554639893E+04
C( 4) =  .1026330177930108E+03
C( 5) = -.9202908099941851E+00
C( 6) =  .3298242672003487E-02
```

THE ERROR OF APPROXIMATION IN THIS INTERVAL IS .9836415372650436E+00.