On Least Squares Fitting Using Orthogonal Multinomials

Richard H. Bartels*
&
John J. Jezioranski**

Technical Report CS-83-24

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

* Department of Computer Science, University of Waterloo
Waterloo, Ontario, Canada, N2L 3G1

**Department of Applied Mathematics, University of Waterloo
Waterloo, Ontario, Canada N2L 3G1

On Least Squares Fitting Using Orthogonal Multinomials

Richard H. Bartels

University of Waterloo
Department of Computer Science
Waterloo, Ontario
Canada N2L 3G1

John J. Jezioranski

University of Waterloo
Department of Applied Mathematics
Waterloo, Ontario
Canada N2L 3G1

ABSTRACT

Forsythe [2] has given a method for generating basis polynomials in a single variable which are orthogonal with respect to a given inner product. Weisfeld [4] later demonstrated that Forsythe's approach could be extended to polynomials in an arbitrary number of variables. In this paper we sharpen Weisfeld's results and present a program for computing weighted, multinomial, least-squares approximations to discrete data.

1. Introduction, Review and Preliminary Notation

Let $<,>$ denote an inner product on an appropriate class of real-valued functions of n real variables. E.g. let **D** be a set in $\mathbf{R}^n$ which supports a nonnegative measure μ. Consider the class **C** of all real-valued functions defined on **D** which are square-integrable with respect to μ, and define

$$<g,h> = \int_{\mathbf{D}} gh \, d\mu$$

Let $\phi_1, \ldots, \phi_m$ be linearly independent functions in **C**. Also in **C**, let $F(x_1, \ldots, x_n) = f(x_1, \ldots, x_n) + e(x_1, \ldots, x_n)$, where e represents some random, unknown error. The problem of fitting f, in the least-squares sense, in the subspace of **C** spanned by the ϕ_i requires that coefficients $c = c_1, \ldots, c_m$ be determined so that

$$P(x_1, \ldots, x_n; c_1, \ldots, c_m) = \sum_{i=1}^{m} c_i \phi_i(x_1, \ldots, x_n)$$

minimizes

$$<F-P, F-P>$$

The coefficients $c_1, \ldots, c_m$ will be given by the solution to

$$\begin{bmatrix} <\phi_1,\phi_1> & \cdots & <\phi_1,\phi_m> \\ \vdots & \vdots & \vdots \\ <\phi_m,\phi_1> & \cdots & <\phi_m,\phi_m> \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_m \end{bmatrix} = \begin{bmatrix} <\phi_1,F> \\ \vdots \\ <\phi_m,F> \end{bmatrix} \tag{1.1}$$

This research was supported under grant number A4076 by the Natural Sciences and Engineering Research Council of Canada, and by the Laidlaw Corporation under a fellowship administered by the University of Waterloo Computer Science Department.

provided that the (symmetric) matrix $[\cdots <\phi_i,\phi_j> \cdots]$ is nonsingular.

For multinomial least-squares fitting $\phi_1,\ldots,\phi_m$ are often taken to be the first m monomials in the n variables of the problem, ϕ's being arranged according to some increasing power order. For example,

$$n = 1$$

$$<g,h> = \sum_{i=1}^{N} w^{(i)} g(x^{(i)}) h(x^{(i)})$$

$$\phi_1(x),\ldots,\phi_m(x) \equiv 1, x, x^2, \ldots, x^m$$

or

$$n = 2$$

$$<g,h> = \int_0^1 \int_0^1 w(x_1,x_2) g(x_1,x_2) h(x_1,x_2) dx_1 dx_2$$

$$\phi_1(x_1,x_2),\ldots,\phi_m(x_1,x_2) \equiv 1, x_1, x_2, x_1^2, x_1x_2, x_2^2, \ldots, x_1^{m_1} x_2^{m_2}$$

A choice of this sort of basis for multinomials will generally result in an ill-conditioned system of equations (1.1). If, instead of the monomials $\phi_1,\ldots,\phi_m$, we choose a basis of multinomials $\psi_1,\ldots,\psi_m$ which are orthogonal with respect to $<,>$, then the matrix of (1.1) will be diagonal.

One may, of course, use the Gram-Schmidt process to orthogonalize the basis $\phi_1,\ldots,\phi_m$, but Forsythe [2] and Weisfeld [4] have shown that a revised Gram-Schmidt process is more efficient. Forsythe worked with

$$n = 1$$

$$<g,h> = \sum_{i=1}^{N} w^{(i)} g(x^{(i)}) h(x^{(i)})$$

and generated $\psi_1,\ldots,\psi_m$ by

$$\psi_1 \equiv 1$$

$$\psi_j = x\psi_{j-1} - \sum_{l<j} \alpha_{j,l}\psi_l$$

(A normalizing constant is often added to the above, which we shall ignore for the sake of simplicity.) This construction ensures that ψ_j will be monic with highest-order term equal to x^j.

It is easily verified that ψ_j will be orthogonal to $\psi_{j-1},\ldots,\psi_1$ if $\alpha_{j,l}$ is chosen as

$$\alpha_{j,l} = \frac{<x\psi_{j-1},\psi_l>}{<\psi_l,\psi_l>}$$

for all $l = 1,\ldots,j-1$. Forsythe showed that $<x\psi_{j-1},\psi_l> = 0$ for all $j-l<2$ (i.e. $l<j-2$). That is, the construction of the orthogonal basis of multinomials is accomplished by a 3-term recurrence.

Weisfeld considered general inner products and a general n. He introduced the notation of vector indices

$$\psi_J = \psi_{(j_1,\ldots,j_n)}$$

where

$$J = (j_1,\ldots,j_n) .$$

By letting $\sigma(J) = j_1+\ldots+j_n$ for any vector index J, he defined

$$I < J$$

to mean

1. $\sigma(I) < \sigma(J)$

or else

2.a. $\sigma(I) = \sigma(J)$ (1.2)

and

2.b. $i_l + \ldots + i_n < j_l + \ldots + j_n$

for some $l \leq n$. (Note that this does not define a unique ordering. Two distinct sequences, among many possible ones, are produced by consistently choosing l in the above to be the *least*, respectively *greatest*, integer such that $l \leq n$).

Given a vector index

$$J = (j_1, \ldots, j_k, 0, \ldots, 0) \ , \tag{1.3}$$

where $j_k > 0$, Weisfeld defined an associated vector index

$$\dot{J} = (j_1, \ldots, j_k - 1, 0, \ldots, 0) \ , \tag{1.4}$$

and constructed

$$\psi_{(0,\ldots,0)} \equiv 1$$

$$\psi_J = x_k \psi_{\dot{J}} - \sum_{L<J} \alpha_{J,L} \psi_L \ . \tag{1.5}$$

As before, it is easily verified that ψ_J will be orthogonal to all ψ_L $(L<J)$ if $\alpha_{J,L}$ is chosen as

$$\alpha_{J,L} = \frac{<x_k \psi_{\dot{J}}, \psi_L>}{<\psi_L, \psi_L>} \ ,$$

Weisfeld proved that $\alpha_{J,L}$ is zero for all $L < J$ such that $\sigma(J) - \sigma(L) < 2$. This constitutes a generalization to n variables of the 3-term recurrence.

In this paper we improve upon Weisfeld's results by taking greater care in fixing the ordering. We will also concern ourselves with the details of mapping vector indices onto the natural numbers to facilitate transcribing our results into a computer program. We will find that more coefficients α turn out to be zero than are indicated in Weisfeld's results, and we are able to take advantage of some identities among the α's.

2. Ordering

In order to present our results, we need to establish a specific mapping from the monomials to the integers, one which will provide a convenient framework for dealing with the orthogonal multinomials which we generate. To do this, we will display the monomials involving n variables in a triangular pattern in which the r^{th} row contains all monomials of $r-1^{st}$ power, and each row of which past the first is organized into n ranges. The first row simply contains 1, and the following row contains

$$x_1 \ , \ldots , \ x_n \ ,$$

the monomial x_k constituting the sole member of range k in row 2. Recursively, the k^{th} range of row r consists of the monomials found by multiplying x_k, in order, onto each member of ranges $k, \ldots, n$ in row $r-1$. For example, when $n = 3$,

Row 1:	1									
Row 2:	x_1	x_2	x_3							
Row 3:	x_1^2	x_1x_2	x_1x_3	x_2^2	x_2x_3	x_3^2				
Row 4:	x_1^3	$x_1^2x_2$	$x_1^2x_3$	$x_1x_2^2$	$x_1x_2x_3$	$x_1x_3^2$	x_2^3	$x_2^2x_3$	$x_2x_3^2$	x_3^3

The three ranges in row 3, then, are

$$x_1 \times \left\{ \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \right\} \equiv \left\{ \begin{array}{ccc} x_1^2 & x_1x_2 & x_1x_3 \end{array} \right\}$$

$$x_2 \times \left\{ x_2 \quad x_3 \right\} \equiv \left\{ x_2^2 \quad x_2 x_3 \right\}$$

$$x_3 \times \left\{ x_3 \right\} \equiv \left\{ x_3^2 \right\}$$

Reading the table from top to bottom and from left to right, the j^{th} member ($j = 1, 2, \cdots$) would have the form

$$x_1^{j_1} \cdots x_n^{j_n}$$

and would be denoted by the vector index

$$\vec{j} = (j_1, \ldots, j_n) \ .$$

Our notational conventions for this table will be as follows:

1. The position in the table of current interest will be indexed by j.
2. The j^{th} monomial in the table will be denoted by $\tilde{j}$.
3. The vector of exponents associated with j will be denoted by $\vec{j}$.
4. $\sigma(j) \equiv \sigma(\tilde{j}) \equiv \sigma(\vec{j})$ is the sum of the exponents associated with j.

The monomials $\tilde{j}$, their position j in the table, and their exponent vectors $\vec{j}$ are all ordered sets of objects, all isomorphic to each other. For example, the $n = 3$ example gives

$$\{j\} = \{1,2,3,4,5,6,7,8,9, \cdots\}$$

$$\{\tilde{j}\} = \{1, x_1, x_2, x_3, x_1^2, x_1x_2, x_1x_3, x_2^2, x_2x_3, \cdots\}$$

$$\{\vec{j}\} = \{(0,0,0),(1,0,0),(0,1,0),(0,0,1),(2,0,0),(1,1,0),(1,0,1),(0,2,0),(0,1,1), \cdots\}$$

That is, for example,

$$\tilde{6} = x_1x_2 \ , \ \vec{6} = (1,1,0) \ ,$$

and further,

$$\sigma(6) = \sigma(\tilde{6}) = \sigma(\vec{6}) = 1+1+0 = 2 \ .$$

Note that $\sigma(j)$ (or $\sigma(\tilde{j})$ or $\sigma(\vec{j})$) points out the row in which j (or $\tilde{j}$ or $\vec{j}$) is to be found; namely: j (or $\tilde{j}$ or $\vec{j}$) is in row $\sigma(j)+1$.

Definition: We will use the symbol $<$ for each of these three sets with the meanings:

$$j^* < j^{**} \equiv j^* \text{ comes before } j^{**} \text{ in } \{j\}$$

$$\vec{j}^* < \vec{j}^{**} \equiv (j_1^*, \ldots, j_n^*) \text{ comes before } (j_1^{**}, \ldots, j_n^{**}) \text{ in } \{\vec{j}\}$$

$$\tilde{j}^* < \tilde{j}^{**} \equiv x_1^{j_1^*} \cdots x_n^{j_n^*} \text{ comes before } x_1^{j_1^{**}} \cdots x_n^{j_n^{**}} \text{ in } \{\tilde{j}\}$$

The symbol $\leq$ will denote

$$j^* < j^{**} \text{ or } j^* \equiv j^{**} \ ,$$

and similarly for $\vec{j}$ and $\tilde{j}$.

With respect to our notation, a monomial $\tilde{j}$ associated with

$$\vec{j} = (0, \ldots, 0, j_{k_j}, \ldots, j_n) \ , \quad j_{k_j} > 0 \ ,$$

is in the $k_j{}^{th}$ range of the row numbered $(j_{k_j} + \ldots + j_n + 1) = \sigma(j) + 1$, and this monomial was constructed by multiplying the monomial associated with the exponent vector

$$(0, \ldots, 0, [j_{k_j} - 1], \ldots, j_n)$$

by x_{k_j}.

By convention, if

$$\vec{j} = (0,\ldots,0,j_{k_j},\ldots,j_n), \quad j_{k_j}>0,$$

we will define j' to be the index in the table for which

$$\vec{j}' = (0,\ldots,0,[j_{k_j}-1],\ldots,j_n),$$

so that

$$\tilde{j}' = \tilde{j}/x_{k_j}, \text{ or equivalently } x_{k_j}\tilde{j}' = \tilde{j}.$$

As such, this represents the inverse of the process of building the ranges which make up row $\sigma(j)+1$.

We will have occasion to continue this one stage further. If j' satisfies

$$\vec{j}' = (0,\ldots,0,j_{k_{j'}}',\ldots,j_n'), \quad j_{k_{j'}}'>0,$$

then we will define j'' by

$$\vec{j}'' = (0,\ldots,0,[j_{k_{j'}}'-1],\ldots,j_n')$$

so that

$$\tilde{j}'' = \tilde{j}'/x_{k_{j'}}, \text{ or equivalently } x_{k_{j'}}\tilde{j}'' = \tilde{j}'.$$

The indices j, k_j, j', $k_{j'}$, and j'' play a defining role in our Gram-Schmidt process.

To help in understanding the ordering of the table, in particular its differences with the ordering used by Weisfeld, (1.2), we establish the following:

Lemma 1: Consider

$$\vec{j}^* = (j_1^*,\ldots,j_n^*) \text{ and } \vec{j}^{**} = (j_1^{**},\ldots,j_n^{**}).$$

The ordering in the table is such that

$$\vec{j}^* < \vec{j}^{**} \quad (\text{and } j^* < j^{**} \text{ and } \tilde{j}^* < \tilde{j}^{**})$$

if and only if either

1. $\sigma(\vec{j}^*) < \sigma(\vec{j}^{**})$

or else

2.a. $\sigma(\vec{j}^*) = \sigma(\vec{j}^{**})$ (2.1)

and

2.b. $j_k^* > j_k^{**}$

where k is the smallest index such that $j_k^* \neq j_k^{**}$.

Proof: The fact that $\vec{j}^* < \vec{j}^{**}$ when $\sigma(\vec{j}^*) < \sigma(\vec{j}^{**})$ is obvious, so we concentrate on a single row of the table; i.e. in the case in which $\sigma(\vec{j}^*) = \sigma(\vec{j}^{**})$

For the first row there is nothing to show. For the second row the exponent vectors are

range 1:	(1,0,0,...,0,0,0)
range 2:	(0,1,0,...,0,0,0)
...	...
range $n-1$:	(0,0,0,...,0,1,0)
range n:	(0,0,0,...,0,0,1)

and the result is obviously true.

Assume for the r^{th} row ($r \geq 2$) that the result is true, and recall that the ranges in this row have exponent vectors of the form:

$$\begin{array}{lllllll}
\text{range 1:} & (r-1, & 0, & 0,\ldots,0, & 0, & 0) & \ldots, (1,0,0,\ldots,0,0,r-2) \\
\text{range 2:} & (0, & r-1, & 0,\ldots,0, & 0, & 0) & \ldots, (0,1,0,\ldots,0,0,r-2) \\
\ldots & & & \ldots & & & \\
\text{range } n-1\text{:} & (0, & 0, & 0,\ldots,0, & r-1, & 0) & \ldots, (0,0,0,\ldots,0,1,r-2) \\
\text{range } n\text{:} & (0, & 0, & 0,\ldots,0, & 0, & r-1) &
\end{array}$$

Hence, the first range in row $r+1$ will be associated with the exponent vectors

$$\begin{array}{llllll}
(r, & 0, & 0,\ldots,0, & 0, & 0) & \ldots, (2,0,0,\ldots,0,0,r-2) \\
(1, & r-1, & 0,\ldots,0, & 0, & 0) & \ldots, (1,1,0,\ldots,0,0,r-2) \\
 & \ldots & & \ldots & & \\
(1, & 0, & 0,\ldots,0, & r-1, & 0) & \ldots, (1,0,0,\ldots,0,1,r-2) \\
(1, & 0, & 0,\ldots,0, & 0, & r-1) &
\end{array}$$

Within this range the result holds true, since the exponent vectors for this range are uniformly those of the entire previous row, in order, with a 1 added into the first position, which does not destroy the ordering relationship.

The second range in row $r+1$ will be associated with the exponent vectors

$$\begin{array}{llllll}
(0,r, & 0,\ldots,0, & 0, & 0) & \ldots, (0,2,0,\ldots,0,0,r-2) \\
 & \ldots & & \ldots & & \\
(0,1, & 0,\ldots,0, & r-1, & 0) & \ldots, (0,1,0,\ldots,0,1,r-2) \\
(0,1, & 0,\ldots,0, & 0, & r-1) &
\end{array}$$

Each of these exponent vectors will appear in row $r+1$ after all of those given in the first range above, and each vector in the first range has a nonzero first entry, while each of these vectors has a zero first entry. So the result which is to be shown holds for each of these vectors with respect to each of the vectors in the first range. As for comparisons of these second-range vectors among themselves, each is constructed, one-for-one, from vectors taken in order from the previous row by adding a 1 to the second component position, which does not disturb the ordering relationship.

It is clear that each succeeding range repeats this argument.

•

3. Indexing

We have remarked that our Gram-Schmidt process will make use of the indices j, k_j, j', and j'' as given above. Both in the construction of the orthogonal multinomials ψ_j, and in the evaluation of any general multinomial Ψ expressed as

$$\Psi = \sum_{j=1}^{m} c_j \psi_j \quad ,$$

it will not be necessary to access the indices j, k_j, j', and j'' in a random fashion. We may always proceed along the table of monomials from top to bottom and left to right, maintaining and updating j, k_j, j', and j'' in a mechanical fashion. We will close this section with a program to carry this out, but we need to begin by establishing a few numbers.

The k_j^{th} range of row $r = \sigma(j)+1$, $r>1$, is constructed to consist of all monomials in which $x_1, \ldots, x_{k_j-1}$ appear to the power 0; that is, it is associated with the exponent vectors $\vec{j}$ of the form $(0,\ldots,0,j_{k_j},\ldots,j_n)$ with $j_{k_j}>0$. In the rth row we must have the equality:

$$j_{k_j} + \cdots + j_n = r-1 \quad .$$

Hence, the number of items in the k_j^{th} range is equivalent to the number of ways in which $r-1$ counters (indistinguishable balls) can be distributed among $n-k_j+1$ exponent positions (distinguishable urns), requiring that at least one counter be assigned to the first position. This number is given by the binomial coefficient

$$\binom{n-k_j+r-2}{r-2} \quad .$$

The number of entries in range k_j through n is the number we obtain by removing the restriction that a counter be placed in the first position:

$$\sum_{l=k_j}^{n} \binom{n-l+r-2}{r-2} = \binom{n-k_j+r-1}{r-1} \quad .$$

And, from this, the number of entries in the entire rth row is given by

$$\binom{n+r-2}{r-1} \quad ,$$

and the number of entries in all of the first r rows together is

$$\binom{n+r-1}{n} \quad .$$

That is, if $T = n-1+r-2$ and $B = r-2$, the run lengths are of the form:

$$\text{Row r:} \quad \binom{T}{B} \quad \binom{T-1}{B} \quad \binom{T-2}{B} \quad \cdots$$

$$\text{Row r+1:} \quad \binom{T+1}{B+1} \quad \binom{T}{B+1} \quad \binom{T-1}{B+1} \quad \cdots$$

and the row lengths are of the form:

$$\text{Row r:} \quad \binom{T+1}{B+1}$$

$$\text{Row r+1:} \quad \binom{T+2}{B+2}$$

Thus, the relationships

$$\binom{T-1}{B} = \binom{T}{B}\frac{T-B}{T}$$

and

$$\binom{T+1}{B+1} = \binom{T}{B}\frac{T+1}{B+1}$$

are clearly useful.

As j runs through range k_j in row r, j' runs in step through ranges $k_j, \ldots, n$ in row $r-1$. As j crosses from range k_j to range k_j+1, j' must be set back to the beginning of range k_j+1 in row $r-1$. Similarly, as j' runs along row $r-1$, j'' runs in step along row $r-2$, and whenever j' crosses from range $k_{j'}$ to range $k_{j'+1}$ in row $r-1$, then j'' must be reset to the beginning of range $k_{j'+1}$ in row $r-1$. Suppose we have an index vector *runlen* which stores the lengths of the n ranges in the rth row. Then the pseudo-code below displays how one can march through the monomial table in order from left to right and from top to

bottom.

```
{/
   A program to step through the indices in the
   ordering table for the orthogonal multinomials.
   The orthogonal multinomials are generated according to
   the pattern

      psi[j] = x[kj]*psi[jp] - sum(t from jpp to j-1: a[j,t]*psi[t])

   n ..... number of variables
   maxr ... maximum number of rows to be used in table
   maxj ... size of table
   j ..... current position in table
   kj ..... current range in row
   jp ..... distinguished multinomial in previous row
   jpp .... lower limit on summation
   ralen .. current range length in current row
   rowlen . length of current row
   jsw .... position of start of next range
   inds ... array for storage and retrieval of kj, jp, jpp
   top .... T, the top term of the binomial coefficient giving rowlen
   bot .... B, the bottom term of this binomial coefficient

   note: the value of ralen is computed directly from the
         binomial coefficient binomial(top,bot), while the value of
         rowlen is obtained by updating this binomial coefficient.

   note: maxj = binomial(n+maxr-1,n)

   note: inds[j,1] will store kj values
         inds[j,2] will store jp values
         inds[inds[j,2],2] will give the jpp values (for j>n+1)

   note: constants are currently set for the
         3-variable, 4-row problem.
/}

program index (input,output);

const
   n   = 3;
   maxr = 4;
   maxj = 20;
var
   j,jend,jp,jpp,jsw,
   kj,r,ralen,rowlen,top,bot : integer;
   inds : array [1..maxj,1..2] of integer;

begin

   rowlen := 1;
   j    := 1;
   jp   := 0;
```

```
top   := n-1;
bot   := 0;
inds[1,2] := 1;

for r := 2 to maxr do
begin

   j       := j+1;
   kj      := 1;
   top     := top+1;
   bot     := bot+1;
   ralen   := rowlen;
   rowlen  := (rowlen*top) div bot;
   jsw     := j+ralen;
   jend    := j+rowlen-1;

   for j:=j to jend do
   begin
      jp  := jp+1;
      if (j>=jsw) then
      begin
         ralen    := (ralen*(top-kj-bot+1)) div (top-kj);
         kj       := kj+1;
         jsw      := j+ralen;
         jp       := jp-ralen
      end;
      inds[j,1] := kj;
      inds[j,2] := jp;
      jpp := inds[jp,2];
   end

end

end.
```

4. The Multinomial Gram-Schmidt Process

We propose that the Gram-Schmidt process be:

$$\psi_1 \equiv 1$$

and for $j = 1, 2, \cdots$

$$\psi_j = x_{k_j}\psi_{j'} - \sum_{l=1}^{j-1} \alpha_{j,l}\psi_l \tag{4.1}$$

where

$$\alpha_{j,l} = \frac{< x_{k_j}\psi_{j'}, \psi_l >}{< \psi_l, \psi_l >}$$

and where k_j and j' are related to j according to the indexing program of the preceding section.

To analyze this version of the Gram-Schmidt process, we establish the following lemmas.

Lemma 2: Let

$$\tilde{l} < \tilde{m} ,$$

and let x_p be any of the variables of the problem. Then

$$x_p \tilde{l} \; < \; x_p \tilde{m} \quad .$$

Proof: If

$$\sigma(\tilde{l}) \; < \; \sigma(\tilde{m}) \quad ,$$

then

$$\sigma(x_p \tilde{l}) = \sigma(\tilde{l}) + 1 \; < \; \sigma(x_p \tilde{m}) = \sigma(\tilde{m}) + 1 \quad .$$

So assume that $\sigma(\tilde{l}) = \sigma(\tilde{m})$; i.e. $\tilde{l}$ and $\tilde{m}$ are found in the same row of the table. If $\tilde{l} < \tilde{m}$, then l and m are such that

$$l_k \; > \; m_k \tag{4.2}$$

for the first entry, k, from the left at which these two exponent vectors differ. But the exponent vectors of $x_p \tilde{l}$ and $x_p \tilde{m}$ are those of $\tilde{l}$ and $\tilde{m}$, respectively, with a 1 added into position p. This does not change the role of the index k or the relationship in (4.2) above.

•

Corollary 1: If

$$\tilde{l} \; < \; \tilde{j}' \quad ,$$

then

$$x_{k_j} \tilde{l} \; < \; x_{k_j} \tilde{j}' = \tilde{j} \quad ,$$

and if

$$\tilde{l} \; < \; \tilde{j}'' \quad ,$$

then

$$x_{k_j} \tilde{l} \; < \; x_{k_j} \tilde{j}'' = \tilde{j}' \quad .$$

Lemma 3: If x_p and x_q are variables of the problem with indices $p \leq q$ (i.e. $x_p \leq x_q$ in the ordering of the table), then

$$x_p \tilde{l} \; \leq \; x_q \tilde{l}$$

for any monomial $\tilde{l}$.

Proof: This follows from the fact that the exponent vector for $x_p \tilde{l}$ would be

$$(l_1, \ldots, l_{p-1}, [l_p + 1], l_{p+1}, \ldots, l_q, \ldots, l_n) \quad ,$$

whereas that for $x_q \tilde{l}$ would be

$$(l_1, \ldots, l_p, \ldots, l_{q-1}, [l_q + 1], l_{q+1}, \ldots, l_n) \quad .$$

Hence, the first exponent vector comes before the second in the ordering of the table (unless $p = q$, in which case the two exponent vectors are equal).

•

Lemma 4: The Gram-Schmidt process (4.1), yields

$$\psi_j \; = \; \tilde{j} \; + \sum_{l=1}^{j-1} \delta_l \tilde{l} \tag{4.3}$$

for some coefficients δ_l.

Proof (by induction):

$j = 1$:

$$\psi_j \equiv 1 \text{ and } \widetilde{j} \equiv 1 ,$$

and the formula of the lemma holds trivially.

$j > 1$:
Suppose that the result has been established for all indices $1 \le l \le j-1$. Consider

$$\psi_j = x_{k_j}\psi_{j'} - \sum_{l=1}^{j-1} \alpha_{j,l}\psi_l$$

But $x_{k_j}\widetilde{j'} = \widetilde{j}$. Furthermore, $j' \le j-1$, so by hypothesis

$$\psi_{j'} = \widetilde{j'} + \sum_{l=1}^{j'-1} \delta_l \widetilde{l} .$$

Hence,

$$\psi_j = \widetilde{j} + \sum_{l=1}^{j'-1} \delta_l x_{k_j}\widetilde{l} - \sum_{l=1}^{j-1} \alpha_{j,l}\psi_l$$

But it follows from Corollary 1 that

$$x_{k_j}\widetilde{l} < \widetilde{j}$$

for all $1 \le l \le j'-1$, and, by the induction hypothesis, each ψ_l in the right-hand summation can be expressed as a linear combination of the 1^{st} through $(j-1)^{th}$ monomials. The result follows by collecting terms in the individual monomials.

•

Lemma 5:

$$\psi_j = \widetilde{j} + \sum_{l=1}^{j-1} \eta_l \psi_l \tag{4.4}$$

Proof (again by induction): It will be clearer to establish the equivalent result that

$$\widetilde{j} = \psi_j - \sum_{l=1}^{j-1} \eta_l \psi_l . \tag{4.5}$$

$j = 1$:

$$\widetilde{1} \equiv \psi_1 ,$$

so, the result is trivially true.

$j > 1$:
Suppose that the result has been established for all indices $1 \le l \le j-1$. From the previous lemma we have that

$$\widetilde{j} = \psi_j - \sum_{l=1}^{j-1} \delta_l \widetilde{l} .$$

But each monomial $\widetilde{l}$, from the induction hypothesis, can be expressed as a linear combination of the ψ's up through the lth. Substituting these combinations in for each $\widetilde{l}$ and collecting terms gives the result.

•

Lemma 6: The multinomials ψ_j are linearly independent.

(This is evident from the previous lemmas.) As an observation, any multinomial which can be expressed as a linear combination of the first j monomials, in the ordering of the table, can also be expressed as a linear combination of the multinomials $\psi_1, \ldots, \psi_j$, and conversely.

From the foregoing we can establish

Theorem 1: For all $l < j''$

$$\alpha_{j,l} = 0 .$$

Proof: Note that, as a feature of our indexing scheme, the integer k_j cannot be larger than $k_{j'}$, since k_j gives the first nonzero power in the monomial $\tilde{j}$, and $k_{j'}$ gives the first nonzero power in $\tilde{j}/x_{k_j}$. Hence, by lemma 3,

$$x_{k_j}\tilde{l} \leq x_{k_{j'}}\tilde{l}$$

for any monomial $\tilde{l}$. Now consider

$$< x_{k_j}\psi_{j'}, \psi_l > \quad \text{for} \quad 1 \leq l \leq j-1 ,$$

and observe that

$$< x_{k_j}\psi_{j'}, \psi_l > = < \psi_{j'}, x_{k_j}\psi_l > .$$

But

$$\psi_l = \tilde{l} + \sum_{t=1}^{l-1} \delta_t \tilde{t} ,$$

and, consequently,

$$x_{k_j}\psi_l = x_{k_j}\tilde{l} + \sum_{t=1}^{l-1} \delta_t x_{k_j}\tilde{t} .$$

But, by our introductory remarks in the proof,

$$x_{k_j}\tilde{l} \leq x_{k_{j'}}\tilde{l} \quad \text{and} \quad x_{k_j}\tilde{t} \leq x_{k_{j'}}\tilde{t} . \tag{4.6}$$

And, if $l < j''$, then the monomials in (4.6) must all come before j' in the table. But each of the monomials in (4.6) is expressible as a linear combination of the multinomials ψ from 1 up to $j'-1$. That means that

$$< x_{k_j}\psi_{j'}, \psi_l > = \sum_{t=1}^{j'-1} \tau_t < \psi_{j'}, \psi_t > ,$$

which is zero, since the multinomials ψ are being constructed as mutually orthogonal.

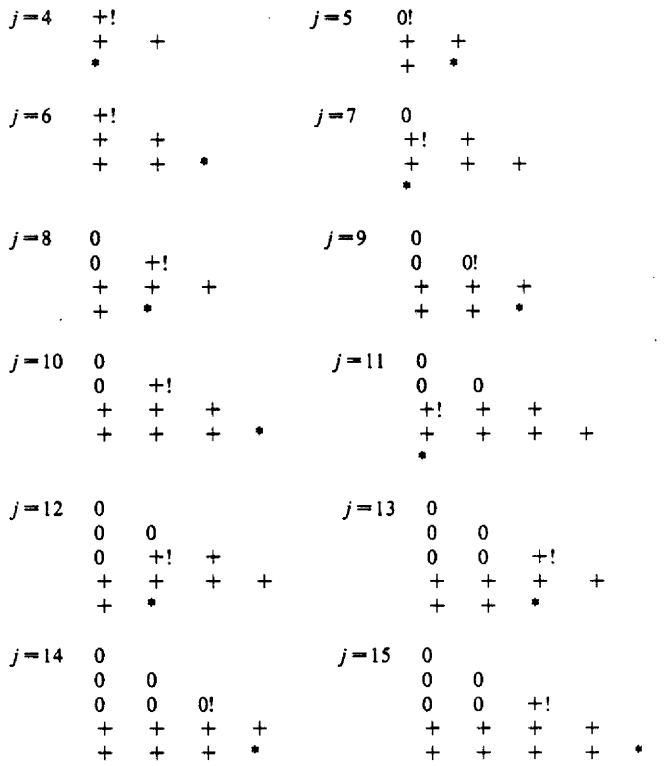
•

As a simple verification of the above, one may generate random numbers as values for $x_l^{(1)}, \ldots, x_l^{(N)}$ $(l = 1, \ldots, n)$ and use the inner product

$$< g, h > = \sum_{i=1}^{N} g(x^{(i)}) h(x^{(i)})$$

to compute all of the α's in sequence. Some results for $n = 2$ are given in tabular outline below, where "*" denotes the position of the orthogonal multinomial being constructed, "+" denotes the position of a previous orthogonal multinomial whose associated α is computed to be nonzero, and "0" denotes the position of a previous orthogonal multinomial whose associated α is computed as zero. The position of j'' is indicated by "!".

$j=1$	*				
$j=2$	+		$j=3$	+	
	*			+	*



It is visible from this that certain extra α's will turn out to be zero. What is not visible in the above schema is that many of the nonzero α's have related values. The next section will establish some results.

5. Additional Relationships

We made use of the fact that $k_j \leq k_{j'}$ in establishing Theorem 1 of the preceeding section. We can sharpen the result of that theorem by splitting this inequality up into its two possible cases.

Corollary 2: Assume that

$$k_j = k_{j'} \ .$$

Then

$$\alpha_{jj''} \neq 0 \ .$$

Proof: Consider

$$\psi_{j'} = x_{k_{j'}} \psi_{j''} - \sum_{l=1}^{j'-1} \alpha_{j'l} \psi_l \ .$$

It follows from this and the orthogonality of the ψ's that

$$\begin{aligned} <\psi_{j'}, \psi_{j'}> &= <x_{k_{j'}} \psi_{j''}, \psi_{j'}> \\ &= <x_{k_{j'}} \psi_{j'}, \psi_{j''}> \end{aligned}$$

$$= \langle x_{k_j}\psi_{j'}, \psi_{j''}\rangle$$
$$= \alpha_{j,j''}\langle \psi_{j''}, \psi_{j''}\rangle \quad .$$

Since both inner products are positive,

$$\alpha_{j,j''} = \frac{\langle \psi_{j'}, \psi_{j'}\rangle}{\langle \psi_{j''}, \psi_{j''}\rangle} \neq 0 \quad .$$

•

Corollary 3: Assume that

$$k_j < k_{j'} \quad .$$

Then

$$\alpha_{j,l} = 0$$

for l running from j'' out to the end of the row containing j''; i.e., for

$$\tilde{j}'' \leq \tilde{l} \leq x_n^{\sigma(j)-2} \quad .$$

Proof: In this case we will have

$\tilde{j}$ associated with $(0,\ldots,0,[1],0,\ldots,0,[j_{k_j}\quad],\ldots,j_n)$

$\tilde{j}'$ associated with $(0,\ldots,0,[0],0,\ldots,0,[j_{k_j}\quad],\ldots,j_n)$

$\tilde{j}''$ associated with $(0,\ldots,0,[0],0,\ldots,0,[j_{k_j}-1],\ldots,j_n)$

where [1] and [0] mark the k_j^{th} component.

If we take any l in the row of j'', $l \geq j''$, the vector index associated with l must have the form

$$(0,\ldots,0,[0],0,\ldots,0,[l_{k_j}],\ldots,l_n)$$

with

$$j_{k_j}-1+\cdots+j_n = l_{k_j}+\cdots+l_n \quad .$$

The numerator of the expression for $\alpha_{j,l}$ would be

$$\langle x_{k_j}\psi_{j'}, \psi_l\rangle = \langle \psi_{j'}, x_{k_j}\psi_l\rangle \quad .$$

But

$$x_{k_j}\psi_l = x_{k_j}(\tilde{l} + \sum_{i=0}^{l-1}\delta_i\tilde{i}) \quad , \tag{5.1}$$

and $x_{k_j}\tilde{l}$ will have an associated vector index of the form

$$(0,\ldots,0,[1],0,\ldots,0,[l_{k_j}],\ldots,l_n) \quad .$$

Furthermore, $x_{k_j}\tilde{l}$ will be in the row of j'. Obviously, $x_{k_j}\tilde{l} < \tilde{j}'$. It follows that

$$\langle x_{k_j}\psi_{j'}, \psi_l\rangle = 0$$

This observation can be repeated for each of the monomials $\tilde{i}$ in (5.1).

•

Notice that, if indices j, l, p, and m are such that

$$\langle x_{k_j}\psi_{j'}, \psi_l\rangle = \langle x_{k_p}\psi_{p'}, \psi_m\rangle \tag{5.2}$$

then

$$\alpha_{j,l} = \alpha_{p,m} \frac{<\psi_m, \psi_m>}{<\psi_l, \psi_l>} .$$

To explore such associations, we have generated numbers of examples with random data. To give an instance, using three variables up through the first four rows of the table yielded the following associations:

$\alpha_{6,2}$	$\longleftrightarrow$	$\alpha_{5,3}$	$\alpha_{15,5}$	$\longleftrightarrow$	$\alpha_{11,9}$
$\alpha_{7,2}$	$\longleftrightarrow$	$\alpha_{5,4}$	$\alpha_{15,6}$	$\longleftrightarrow$	$\alpha_{12,9}$
$\alpha_{7,3}$	$\longleftrightarrow$	$\alpha_{6,4}$	$\alpha_{15,7}$	$\longleftrightarrow$	$\alpha_{13,9}$
$\alpha_{9,3}$	$\longleftrightarrow$	$\alpha_{8,4}$	$\alpha_{15,8}$	$\longleftrightarrow$	$\alpha_{14,9}$
$\alpha_{11,3}$	$\longleftrightarrow$	$\alpha_{6,5}$	$\alpha_{16,5}$	$\longleftrightarrow$	$\alpha_{11,10}$
$\alpha_{11,4}$	$\longleftrightarrow$	$\alpha_{7,5}$	$\alpha_{16,6}$	$\longleftrightarrow$	$\alpha_{12,10}$
$\alpha_{12,4}$	$\longleftrightarrow$	$\alpha_{7,6}$	$\alpha_{16,7}$	$\longleftrightarrow$	$\alpha_{13,10}$
$\alpha_{12,5}$	$\longleftrightarrow$	$\alpha_{11,6}$	$\alpha_{16,8}$	$\longleftrightarrow$	$\alpha_{14,10}$
$\alpha_{13,5}$	$\longleftrightarrow$	$\alpha_{11,7}$	$\alpha_{16,9}$	$\longleftrightarrow$	$\alpha_{15,10}$
$\alpha_{13,6}$	$\longleftrightarrow$	$\alpha_{12,7}$	$\alpha_{17,4}$	$\longleftrightarrow$	$\alpha_{9,8}$
$\alpha_{14,5}$	$\longleftrightarrow$	$\alpha_{11,8}$	$\alpha_{18,8}$	$\longleftrightarrow$	$\alpha_{17,9}$
$\alpha_{14,6}$	$\longleftrightarrow$	$\alpha_{12,8}$	$\alpha_{19,8}$	$\longleftrightarrow$	$\alpha_{17,10}$
$\alpha_{14,7}$	$\longleftrightarrow$	$\alpha_{13,8}$	$\alpha_{19,9}$	$\longleftrightarrow$	$\alpha_{18,10}$

In each of the above, the association is such that

$$\alpha_{j,l} \longleftrightarrow D$$

if and only if

$$m = j' ,$$

and further,

$$<x_{k_j}\psi_{j'}, \psi_l> = <\psi_{j'}, x_{k_j}\psi_l> = <\psi_{j'}, x_{k_p}\psi_{p'}> = <x_{k_p}\psi_{p'}, \psi_{j'}> ,$$

with the stringent requirement that

$$l = p' \text{ and } k_j = k_p . \tag{5.3}$$

for some index p. It is trivial to show that the α's associate when j, l, and p satisfy these requirements. The proof eludes us that no other α's associate.

This association will be of use primarily in telling us that the inner product

$$<x_{k_j}\psi_{j'}, \psi_l>$$

has been computed before.

As a first consideration, the case

$$\tilde{l} < \tilde{j}''$$

can be ignored, since

$$<x_{k_j}\psi_{j'}, \psi_l> = 0$$

for all such l. Secondly, the case

$$\tilde{l} = \tilde{j}''$$

is not of interest, since it is handled fully in Corollary 2 and Corollary 3. Thirdly, the case

$$\tilde{j}'' \leq \tilde{l} \leq x_n^{\sigma(j)-2}$$

is not of interest when $k_j < k_{j'}$, since

$$<x_{k_j}\psi_{j'}, \psi_l> = 0$$

by Corollary 3 in that case. As a fourth consideration, (5.3) will require that

$$\widetilde{l} \geq x_{k_j}^{\sigma(j)-1}$$

when $\widetilde{l}$ falls in the row of j'. And finally, we must have

$$\widetilde{p} = x_{k_j}\widetilde{l} < \widetilde{j} \quad ,$$

since we are only interested in inner products which have been computed in the past and can be re-used. This implies that

$$\widetilde{l} < \widetilde{j}' \quad .$$

Hence, we are left only with the cases

(A) $x_{k_j}^{\sigma(j)-2} < \widetilde{l} \leq x_n^{\sigma(j)-2}$ and $k_j = k_{j'}$

and

(B) $x_{k_j}^{\sigma(j)-1} \leq \widetilde{l} < \widetilde{j}'$.

The examples of $\alpha_{16,l}$ are illustrative. Since

$$\widetilde{16} = x_1\widetilde{10}$$

and

$$\widetilde{10} = x_3\widetilde{4}$$

we have

$$j = 16, j' = 10, k_j = 1, j'' = 4, k_{j'} = 3 \quad .$$

Hence, we must have

$$\alpha_{16,l} = \alpha_{p,10}$$

for each $\widetilde{p}$ which can be expressed as $x_1\widetilde{l}$. These are precisely those for which (B) above holds:

$$x_1^2 \leq \widetilde{l} < \widetilde{10} \quad ,$$

which yields

$$p = 11, 12, 13, 14, 15$$

for

$$l = 5, 6, 7, 8, 9$$

respectively.

6. FORTRAN Program

A program has been prepared to implement the results described in this paper and is included as an appendix. All of the features of indexing, attention to zero inner products, and attention to associations between α's have been included as described. In addition, the computation of the fitting coefficients

$$c_l = \frac{<F, \psi_l>}{<\psi_l, \psi_l>}$$

is done by the alternative formula

$$c_l = \frac{<z_{l-1}, \psi_l>}{<\psi_l, \psi_l>} \quad ,$$

where

$$z_{l-1} = F - \sum_{i=1}^{l-1} c_i \psi_l$$

as is advocated in [1]. The program has been passed by the PFORT [3] verifier.

The program is divided into a number of subroutines, of which the user need be concerned with only two: CONSTR and EVAL. The former is used to construct the orthogonal basis of multinomials, for given data, and determine the least squares fitting multinomial. The latter is used to evaluate that fitting multinomial for given values of the variables. The header comments and calling sequences of both subroutines are duplicated below for easy reference. Included in the appendix, too, are a simple driver program to read data and call CONSTR and EVAL, and a sample data-generating program to provide some test cases.

```
      SUBROUTINE CONSTR(DIMEN,FITDEG,NFPOLS,NFPTS,
     +                  FITCDS,NCROWS,FITVLS,WTS,
     +                  RESIDS,NEWFIT,ERROR,FITIWK,
     +                  FITDWK,FIWKLN,FDWKLN,IREQD,DREQD)
C
      INTEGER NFPOLS,FITDEG,NFPTS,DIMEN,FIWKLN,FDWKLN
      INTEGER ERROR,NCROWS,IREQD,DREQD
      INTEGER FITIWK(FIWKLN)
      DOUBLE PRECISION FITDWK(FDWKLN),FITCDS(NCROWS,DIMEN)
      DOUBLE PRECISION FITVLS(NFPTS),RESIDS(NFPTS)
      DOUBLE PRECISION WTS(NFPTS)
      LOGICAL NEWFIT
C
C     ***************
C     PURPOSE
C     -------
C
C     THIS SUBROUTINE CONSTRUCTS A LEAST-SQUARES MULTINOMIAL FIT TO
C     GIVEN DATA USING A BASIS OF ORTHOGONAL MULTINOMIALS.
C
C     THE DATA FOR THE FIT IS GIVEN IN THE ARRAYS $FITCDS$, $FITVLS$,
C     AND $WTS$.  $FITCDS$ IS A DOUBLE-PRECISION MATRIX, EACH ROW OF
C     WHICH CONTAINS AN OBSERVATION POINT (ORDERED COLLECTION OF
C     VARIABLE VALUES).  $FITVLS$ IS A DOUBLE-PRECISION, SINGLY-
C     INDEXED ARRAY, EACH ELEMENT OF WHICH CONTAINS AN OBSERVED
C     FUNCTION VALUE CORRESPONDING TO AN OBSERVATION POINT.  $WTS$ IS
C     A DOUBLE-PRECISION, SINGLY-INDEXED ARRAY, EACH ELEMENT OF WHICH
C     IS A NONNEGATIVE WEIGHT FOR THE CORRESPONDING OBSERVATION.
C
C     THE FIT WHICH IS PRODUCED IS A MULTINOMIAL EXPRESSED IN THE FORM
C
C      C  PSI  (X ,...,X    ) +...+ C      PSI     (X ,...,X    )
C       1   1  1     DIMEN         NFPOLS   NFPOLS  1     DIMEN
C
C     WHERE THE VALUE OF $NFPOLS$ WILL BE AS GIVEN (IF $FITDEG$ < 0)
C     OR AS COMPUTED BY $CONSTR$ TO GIVE A FULL-DEGREE FIT (IN CASE
C     $FITDEG$ IS SPECIFIED >= 0).  THE ELEMENTS
C
C        PSI  (X ,...,X    )
C          K  1     DIMEN
C
C     FORM A BASIS FOR THE MULTINOMIALS WHICH IS ORTHOGONAL WITH
C     RESPECT TO THE WEIGHTS AND OBSERVATION POINTS.
C
C     THE EXTENT OF THE FIT CAN BE SPECIFIED IN ONE OF TWO WAYS.
C        IF THE PARAMETER $FITDEG$ IS SET >= 0, THEN A COMPLETE BASIS
C        FOR THE MULTINOMIALS OF DEGREE = $FITDEG$ WILL BE USED.  (AN
```

```
C     ERROR WILL BE FLAGGED IF THIS WILL REQUIRE MORE BASIS
C     MULTINOMIALS THAN THE NUMBER OF DATA POINTS WHICH WERE
C     GIVEN.)
C     IF THE PARAMETER $FITDEG$ IS < 0, THEN $NFPOLS$ WILL BE
C     TAKEN AS THE COUNT OF THE NUMBER OF BASIS MULTINOMIALS TO BE
C     USED FOR A PARTIAL-DEGREE FIT. (AN ERROR WILL BE FLAGGED IF
C     $NFPOLS$ < 0.)
C
C   NOTE, THE CALL TO $CONSTR$ WITH $NEWFIT$ = .TRUE. CAN BE MADE
C     WITH THE PARAMETERS SET FOR THE MAXIMUM FIT DESIRED.
C     SEVERAL SUBSEQUENT CALLS TO $CONSTR$ WITH $NEWFIT$ = .FALSE.
C     CAN BE MADE WITH SMALLER VALUES OF $FITDEG$ OR $NFPOLS$ AS
C     MAY BE DESIRED TO OBTAIN A PARTIAL FIT.
C
C   VARIABLES
C   ---------
C
C   $DIMEN$ - (INTEGER) - (PASSED)
C     THE NUMBER OF VARIABLES.
C   $FITDEG$ - (INTEGER) - (PASSED/RETURNED)
C     IGNORED IF < 0.
C     IF $DEGREE$ >= 0 THEN $DEGREE$ IS CHECKED AGAINST $NFPTS$.
C     THE VALUE OF $DEGREE$ WILL BE REDUCED IF THERE IS A BASIS OF
C     MULTINOMIALS, ALL OF DEGREE <= $DEGREE$, OF CARDINALITY
C     $NFPTS$. SEE $ERROR$ BELOW.
C   $NFPOLS$ - (INTEGER) - (PASSED/RETURNED)
C     IGNORED IF $DEGREE$ >= 0.
C     IF $DEGREE$ < 0 THEN THE VALUE OF $NFPOLS$ WILL BE TAKEN AS
C     THE SIZE OF THE BASIS OF MULTINOMIALS TO BE USED IN THE FIT.
C     $NFPOLS$ MUST SATISFY $NFPOLS$ < $NFPTS$ AND $NFPOLS$ >= 1
C     SEE $ERROR$ BELOW.
C   $NFPTS$ -- (INTEGER) - (PASSED)
C     THE NUMBER OF DATA POINTS TO BE USED IN THE FIT.
C     $NFPTS$ MUST BE >= 1. SEE $ERROR$ BELOW.
C   $FITCDS$ - (DOUBLE-PRECISION, 2-SUBSCRIPT ARRAY) - (PASSED)
C     $FITCDS$(P,K) IS THE VALUE OF THE K-TH VARIABLE AT THE P-TH
C     DATA POINT.
C   $NCROWS$ - (INTEGER) - (PASSED)
C     THE ROW DIMENSION DECLARED FOR $FITCDS$ IN THE CALLING
C     PROGRAM.
C   $FITVLS$ - (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) - (PASSED)
C     $FITVLS$(P) IS THE OBSERVED FUNCTION VALUE OF THE P-TH DATA
C     POINT.
C   $WTS$ -- (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) - (PASSED)
C     $WTS$(P) IS THE WEIGHT ATTACHED TO THE P-TH DATA POINT.
C   $RESIDS$ - (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) - (RETURNED)
C     $RESIDS$(P) IS THE DIFFERENCE BETWEEN THE FITTED FUNCTION AT
C     POINT P AND $FITVLS$(P).
C   $NEWFIT$ - (LOGICAL) - (PASSED)
C     A LOGICAL FLAG. IF $NEWFIT$=.TRUE., THEN THIS IS THE FIRST
C     FIT TO BE CARRIED OUT WITH THE DATA TO BE FOUND IN THE OTHER
C     PARAMETERS TO $CONSTR$, AND SPACE FOR A FIT IS TO BE
C     ALLOCATED. IF $NEWFIT$=.FALSE., THEN A FIT OF ANOTHER DEGREE
C     CAN BE CONSTRUCTED IN THE SPACE ALLOCATED ON A PREVIOUS CALL
C     WITH THE SAME DATA, AND CERTAIN INITIALIZATION STEPS ARE BY-
```

```
C      PASSED.
C    $ERROR$ - (INTEGER) - (RETURNED)
C      0 IF $NPOLYS$, $DIMEN$, $DEGREE$, $NPTS$ AND $IWKLEN$ ARE
C        VALID AND CONSISTENT WITH EACH OTHER.
C      1 IF $DEGREE$ >= 0 BUT THERE IS AN INTERPOLATING MULTINOMIAL
C        OF SMALLER DEGREE OR IF $DEGREE$ < 0 AND $NPOLYS$ > $NPTS$.
C      2 IF $DEGREE$ < 0 AND $NPOLYS$ <= 0.
C      3 IF $NPTS$ < 1 AND/OR $DIMEN$ < 1.
C      4 IF $IWKLEN$ AND/OR $DWKLEN$ IS TOO SMALL. (SET $IWKLEN$ TO
C        THE VALUE RETURNED IN $IREQD$, AND SET $DWKLEN$ TO THE VALUE
C        RETURNED IN $IREQD$ TO RESOLVE THIS PROBLEM.)
C      5 $NEWFIT$ = .FALSE. BUT $ONPLYS$ >= $NFPOLS$. $EVAL$ CAN BE
C        CALLED REQUESTING A SMALLER BASIS THAN WAS GENERATED.
C      6 ERROR RETURN FROM $INCDG$. THERE IS NO MORE ROOM IN THE
C        $FITDWK$ AND/OR $FITIWK$ ARRAYS TO INCLUDE MORE TERMS IN THE
C        FIT. (CAUSED BY A SUCCESSION OF CALLS TO $CONSTR$ WITH THE
C        FLAG $NEWFIT$ SET .FALSE., REQUESTING EVER HIGHER ORDER
C        FITS, IN WHICH THE DIMENSIONING AND ORDER INFORMATION GIVEN
C        ON THE FIRST CALL ($NEWFIT$=.TRUE.) IS EXCEEDED.)
C    $FITIWK$ - (INTEGER, 1-SUBSCRIPT ARRAY) - (RETURNED)
C        AN INTEGER WORK ARRAY OF LENGTH $FIWKLN$. UPON RETURN FROM
C        A CALL TO $CONSTR$ WITH $NEWFIT$ SET .TRUE., SOME DIMENSION
C        AND ARRAY-LENGTH INFORMATION WILL BE INSERTED. UPON RETURN
C        FROM A CALL TO $CONSTR$ WITH $NEWFIT$ SET .FALSE., DETAILED
C        INDEXING INFORMATION (LOCATION OF COEFFICIENTS IN $FITDWK$,
C        ETC.) IS INSERTED.
C    $FITDWK$ - (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) - (RETURNED)
C        A DOUBLE PRECISION ARRAY OF LENGTH $FDWKLN$. UPON RETURN
C        FROM $CONSTR$ WITH $NEWFIT$ SET .FALSE., THE FULL DETAILS
C        OF THE REQUESTED FIT (COEFFICIENTS, ETC.) WILL BE INSERTED.
C    $FIWKLN$ - (INTEGER) - (PASSED)
C        THE LENGTH OF THE ARRAY $FITIWK$.
C    $FDWKLN$ - (INTEGER) - (PASSED)
C        THE LENGTH OF THE ARRAY $FITDWK$.
C    $IREQD$ - (INTEGER) - (PASSED)
C        THE LENGTH WHICH THE ARRAY $FITIWK$ REALLY NEEDS TO BE.
C    $DREQD$ - (INTEGER) - (PASSED)
C        THE LENGTH WHICH THE ARRAY $FITDWK$ REALLY NEEDS TO BE.
C
C
C    NOTE, THE 10 AND 70 LOOPS (I.E. THE LOOPS FOR SCALING THE
C    RESIDUALS) DEPEND ON THE SCALING SCHEME USED. THE RESIDUAL
C    SCALING MUST BE CONSISTENT WITH THAT DEFINED BY $SCALPM$,
C    $SCALDN$, AND $SCALUP$.
C
C    $CONSTR$ CALLS $ALLOT$, $RESTRT$, $INCDG$, AND $GNRTP$
C
C
C    DATE LAST MODIFIED
C    ---- -- -----
C    MARCH 9, 1984
C    ***************
C
```

```
      SUBROUTINE EVAL(DIMEN,EVLDEG,NEPOLS,NEPTS,EVLCDS,EVLVLS,
     +                ERROR,FITIWK,FITDWK,FIWKLN,FDWKLN,EVLDWK,EDWKLN)
C
      INTEGER FIWKLN,FDWKLN,NEPOLS,NEPTS,DIMEN,ERROR
      INTEGER EVLDEG,EDWKLN
      INTEGER FITIWK(FIWKLN)
      DOUBLE PRECISION FITDWK(FDWKLN),EVLDWK(EDWKLN),EVLCDS(NEPTS,DIMEN)
      DOUBLE PRECISION EVLVLS(NEPTS)
C
C     **********
C     PURPOSE
C     ---
C
C     THIS SUBROUTINE EVALUATES THE LEAST-SQUARES MULTINOMIAL FIT
C     WHICH HAS BEEN PREVIOUSLY PRODUCED BY $CONSTRS. EITHER THE FULL
C     MULTINOMIAL AS PRODUCED MAY BE EVALUATED, OR ONLY AN INITIAL
C     SEGMENT THEREOF. AS IN THE CASE WITH $CONSTRS, IT IS POSSIBLE
C     (1) TO SPECIFY MULTINOMIALS OF A FULL GIVEN DEGREE, OR
C     (2) TO SPECIFY THE NUMBER OF ORTHOGONAL BASIS ELEMENTS TO
C        ACHIEVE A PARTIAL-DEGREE FIT.
C
C     IN CASE (1), THE DESIRED DEGREE IS GIVEN AS THE VALUE OF
C     $EVLDEG$ (WHICH MUST BE NONNEGATIVE AND NOT GREATER THAN THE
C     VALUE USED FOR $FITDEG$ IN $CONSTRS), AND THE PARAMETER $NEPOLSS
C     WILL BE SET BY $EVAL$ TO SPECIFY THE NUMBER OF BASIS ELEMENTS
C     REQUIRED. IF $EVLDEG$ < $FITDEG$ IS GIVEN, THEN ONLY THE
C     INITIAL PORTION OF THE FITTING MULTINOMIAL (OF DEGREE $EVLDEG$)
C     WILL BE EVALUATED.
C
C     IN CASE (2), $EVLDEG$ IS TO BE SET NEGATIVE, IN WHICH CASE THE
C     VALUE OF $NEPOLS$ (WHICH MUST BE POSITIVE AND NOT GREATER THAN
C     THE VALUE USED FOR $NFPOLS$ IN $CONSTRS) WILL BE TAKEN AS
C     DEFINING THE INITIAL PORTION OF THE FITTING MULTINOMIAL TO BE
C     EVALUATED.
C
C     IF $NEPOLS$ = $NFPOLS$ (WITH $EVLDEG$ < 0), OR $EVLDEG$ =
C     $FITDEG$ (WITH $EVLDEG$ > 0), THEN THE FULL MULTINOMIAL
C     GENERATED BY $CONSTR$ WILL BE EVALUATED.
C
C     THE EVALUATION WILL TAKE PLACE FOR EACH OF THE POINTS
C     (COLLECTION OF VARIABLE VALUES) GIVEN AS A ROW OF THE MATRIX
C     $EVLCDS$. THE VALUES PRODUCED FROM THE FULL, OR PARTIAL,
C     MULTINOMIAL WILL BE PLACED IN THE ARRAY $EVLVLS$.
C
C     VARIABLES
C     ---
C
C     $DIMEN$ - (INTEGER) - (PASSED)
C        THE NUMBER OF VARIABLES.
C     $EVLDEG$ - (INTEGER) - (PASSED)
C        IF $EVLDEG$ < 0, THEN THIS PARAMETER WILL BE IGNORED.
C        IF $EVLDEG$ >= 0, THEN THE VALUE OF $EVLDEG$ MUST SATISFY
C        $EVLDEG$ <= (THE DEGREE OF THE APPROXIMATING MULTINOMIAL
C        GENERATED IN $CONSTR$). IN THIS CASE $EVLDEG$ WILL SPECIFY
C        THE DEGREE OF THE INITIAL PORTION OF THE FITTING MULTINOMIAL
```

```
C       TO BE EVALUATED.
C     $NEPOLS$ - (INTEGER) - (PASSED/RETURNED)
C       IF $EVLDEG$ >= 0, THEN THIS PARAMETER WILL BE IGNORED.
C       IF $EVLDEG$ < 0, THEN THE PARTIAL MULTINOMIAL INVOLVING THE
C       FIRST $NEPOLS$ ORTHOGONAL BASIS FUNCTIONS WILL BE EVALUATED
C       AT THE POINTS GIVEN BY $EVLCDS$. THE RESULTING VALUES WILL
C       BE STORED IN $EVLVLS$.
C       THE VALUE OF $NEPOLS$ MUST BE >= 1 AND <= (THE SIZE OF THE
C       BASIS GENERATED IN $CONSTR$), WHICH WAS RETURNED AS THE
C       VALUE OF $NFPOLS$.
C       $NEPOLS$ WILL BE CHANGED IF $EVLDEG$ > 0 TO GIVE THE SIZE OF
C       BASIS REQUIRED FOR THE MULTINOMIAL OF DEGREE $EVLDEG$.
C     $NEPTS$ - (INTEGER) - (PASSED)
C       THE NUMBER OF EVALUATION POINTS.
C     $EVLCDS$ - (INTEGER ) - (PASSED)
C       $EVLCDS$(P,K) IS THE VALUE OF THE K-TH VARIABLE AT THE P-TH
C       EVALUATION POINT.
C     $EVLVLS$ - (INTEGER) - (RETURNED)
C       $EVLVLS$(P) IS THE VALUE OF THE EVALUATED MULTINOMIAL AT THE
C       P-TH EVALUATION POINT.
C     $ERROR$ - (INTEGER) - (RETURNED)
C        0 ..... IF NO ERRORS
C       -1 ..... IF $NEPOLS$ > $NPOLYS$ OR $NEPOLS$ < 1
C       -2 ..... IF $NEPTS$ < 1 OR $DIMEN$ < 1
C       $NEPOLS$ ... IF $NEPOLS$ > $EDWKLN$
C     $FITIWK$ - (INTEGER, 1-SUBSCRIPT ARRAY) - (PASSED)
C       THE INTEGER WORK ARRAY OF LENGTH $FIWKLN$ THAT WAS USED IN
C       $CONSTR$.
C     $FITDWK$ - (DOUBLE-PRECISION, 2-SUBSCRIPT ARRAY) - (PASSED)
C       THE DOUBLE PRECISION WORK ARRAY OF LENGTH $FDWKLN$ THAT WAS
C       USED IN $CONSTR$.
C     $FIWKLN$ - (INTEGER) - (PASSED)
C       THE LENGTH OF $FITIWK$.
C     $FDWKLN$ - (INTEGER) - (PASSED)
C       THE LENGTH OF $FITDWK$.
C     $EVLDWK$ - (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY ) - (RETURNED)
C       A WORK ARRAY OF LENGTH $NEPOLS$ (OR LONGER).
C     $EDWKLN$ - (INTEGER) - (PASSED)
C       THE LENGTH OF $EVLDWK$.
C
C     THE SUBROUTINE $EVALP$ IS CALLED TO DO THE ACTUAL EVALUATION.
C
C
C     DATE LAST MODIFIED
C     ---- -- --------
C     FEBRUARY 2, 1984
C     ****************
C
```

7. References

[1] J. H. Cadwell and D. E. Williams (1961), Some Orthogonal Methods of Curve and Surface Fitting, *Computer Journal* **4**, 260-261.

[2] G. E. Forsythe (1957), Generation and Use of Orthogonal Polynomials for Data-Fitting with a Digital Computer, *J. SIAM* **5**, 74-87.

[3] B. G. Ryder (1974), The PFORT Verifier, *Software Practice and Experience* **4**, 359-377.

[4] M. Weisfeld (1959), Orthogonal polynomials in Several Variables, *Numer. Math.* **1**, 38-40.

8. APPENDIX

8.1. Simple Driver Program

```
      INTEGER DIMEN,FITDEG,NFPOLS,NFPTS,EVLDEG,NEPOLS,NEPTS
      INTEGER ERROR,FIWKLN,FDWKLN,EDWKLN,IREQD,DREQD
      INTEGER FITIWK(89)
      DOUBLE PRECISION FITDWK(2201),FITVLS(125),FITCDS(375),WTS(125)
      DOUBLE PRECISION EVLDWK(125), EVLVLS(20), EVLCDS(60), RESIDS(125)
C
C     ***************
C     A SIMPLE DRIVER PROGRAM TO READ TEST DATA AND CALL THE
C     JEZIORANSKI-BARTELS SUITE OF MULTINOMIAL LEAST-SQUARES FITTING
C     ROUTINES.
C
C     THE DATA AND ARRAY DECLARATIONS ARE CONSISTENT WITH PROBLEMS
C     INVOLVING UP TO 3 VARIABLES, 125 FITTING POINTS, AND 20
C     EVALUATION POINTS USING 20 BASIS ORTHOGONAL MULTINOMIALS IN BOTH
C     THE FIT AND THE EVALUATION.
C
C     LATEST UPDATE... MARCH 9, 1984
C     ***************
C
      DATA FDWKLN/2201/
      DATA EDWKLN/125/
      DATA FIWKLN/89/
C
C     ***************
C     READ FITTING DATA
C     ***************
C
      READ (5,5000) DIMEN,FITDEG,NFPOLS,NFPTS
      WRITE (6,6000) DIMEN,FITDEG,NFPOLS,NFPTS
      CALL INFIT(DIMEN,NFPTS,FITCDS,FITVLS,WTS)
C
C     ***************
C     COMPUTE THE LEAST-SQUARES FIT
C     ***************
C
      CALL CONSTR(DIMEN,FITDEG,NFPOLS,NFPTS,FITCDS,NFPTS,
     +            FITVLS,WTS,RESIDS,.TRUE.,ERROR,FITIWK,
     +            FITDWK,FIWKLN,FDWKLN,IREQD,DREQD)
C
C     ***************
C     PRINT RESIDUALS AT THE FITTING POINTS
C
C     THE USER COULD CHECK  IREQD,DREQD  AT THIS POINT TO SEE
C     THE NUMBER OF LOCATIONS WHICH WERE ACTUALLY REQUIRED IN
C     THE ARRAYS  FITIWK,FITDWK.
C     ***************
C
      WRITE (6,6010)
      CALL OUTRES(NFPTS,RESIDS)
C
```

```
C     ***************
C     READ POINTS OF EVALUATION
C     ***************
C
      READ (5,5000) EVLDEG,NEPOLS,NEPTS
      WRITE (6,6020) EVLDEG,NEPOLS,NEPTS
      CALL INEVL(DIMEN,NEPTS,EVLCDS)
C
C     ***************
C     EVALUATE THE FITTING MULTINOMIAL
C     ***************
C
      CALL EVAL(DIMEN,EVLDEG,NEPOLS,NEPTS,EVLCDS,EVLVLS,
     +          ERROR,FITIWK,FITDWK,FIWKLN,FDWKLN,EVLDWK,EDWKLN)
C
C     ***************
C     PRINT OUT THE ARRAY OF MULTINOMIAL VALUES
C     ***************
C
      CALL OUTEVL(DIMEN,NEPTS,NEPOLS,EVLCDS,EVLVLS)
C
      STOP
C
C     ***************
C     FORMATS
C     ***************
C
 5000 FORMAT(5I5)
 6000 FORMAT(//31H MULTINOMIAL FITTING PROBLEM...//
     +       13H DIMENSION = ,I5/
     +       32H DEGREE OF THE FIT TO BE MADE = ,I5/
     +       43H NUMBER OF BASIS MULTINOMIALS TO BE USED = ,I5/
     +       25H NUMBER OF DATA POINTS = ,I5)
 6010 FORMAT(//18H FITTING COMPLETE./13H RESIDUALS...
     +       //4X,1HI,5X,8HRESIDUAL)
 6020 FORMAT(//14H EVALUATION...//
     +       37H DEGREE OF THE FIT TO BE EVALUATED = ,I5/
     +       42H NUMBER OF BASIS POLYNOMIALS TO BE USED = ,I5/
     +       31H NUMBER OF EVALUATION POINTS = ,I5)
C
      END

      SUBROUTINE INEVL(DIMEN,NEPTS,EVLCDS)
C
      INTEGER DIMEN,NEPTS,I,J
      DOUBLE PRECISION EVLCDS(NEPTS,DIMEN)
C
C     ***************
C     SUBROUTINE TO READ THE ARRAY OF EVALUATION POINTS.
C     ***************
C
      DO 100 I=1,NEPTS
        READ (5,5000) (EVLCDS(I,J),J=1,DIMEN)
  100 CONTINUE
```

```
      RETURN
 5000 FORMAT(4D14.6)
      END

      SUBROUTINE INFIT(DIMEN,NFPTS,FITCDS,FITVLS,WTS)
C
      INTEGER NFPTS,DIMEN,I,J
      DOUBLE PRECISION FITCDS(NFPTS,DIMEN),FITVLS(NFPTS),WTS(NFPTS)
C
C     ***************
C     SUBROUTINE TO READ THE FITTING DATA.
C     ***************
C
      WRITE (6,6000)
      DO 100 I=1,NFPTS
        READ (5,5000) WTS(I),(FITCDS(I,J),J=1,DIMEN),FITVLS(I)
        WRITE (6,6010) I,WTS(I),(FITCDS(I,J),J=1,DIMEN),FITVLS(I)
  100 CONTINUE
      RETURN
 5000 FORMAT(5D14.6)
 6000 FORMAT(8H DATA.../
     +       4X,1HI,5X,6HWEIGHT,19X,11HCOORDINATES,17X,11HDATA VALUES)
 6010 FORMAT(I5,5D14.6)
      END

      SUBROUTINE OUTEVL(DIMEN,NEPTS,NEPOLS,EVLCDS,EVLVLS)
C
      INTEGER DIMEN,NEPTS,NEPOLS,I,J
      DOUBLE PRECISION EVLCDS(NEPTS,DIMEN),EVLVLS(NEPTS)
C
C     ***************
C     SUBROUTINE TO PRINT OUT THE RESULTS OF THE EVALUATION.
C     ***************
C
      WRITE (6,6000)
      DO 100 I=1,NEPTS
        WRITE (6,6010) I,(EVLCDS(I,J),J=1,DIMEN),EVLVLS(I)
  100 CONTINUE
      RETURN
 6000 FORMAT(14H EVALUATION.../
     +       4X,1HI,22X,11HCOORDINATES,14X,5HVALUE)
 6010 FORMAT(I5,4D14.6)
      END

      SUBROUTINE OUTRES(NFPTS,RESIDS)
C
      INTEGER NFPTS,I
      DOUBLE PRECISION RESIDS(NFPTS)
C
C     ***************
C     SUBROUTINE TO PRINT OUT THE RESIDUALS FROM THE FIT.
C     ***************
C
      DO 100 I=1,NFPTS
```

```
      WRITE (6,6000) I,RESIDS(I)
  100 CONTINUE
      RETURN
C
 6000 FORMAT(I5,D14.6)
      END
```

8.2. Simple Data Generator

```
      INTEGER FITDEG,DIMEN,NFPOLS,NFPTS,NEPTS,NEPOLS,EVLDEG
C
C     ***************
C     A PRIMITIVE DATA-GENERATING PROGRAM FOR USE WITH THE
C     JEZIORANSKI-BARTELS SUITE OF ROUTINES FOR MULTINOMIAL
C     LEAST-SQUARES FITTING.
C
C     THE OUTPUT FORMATS IN (GENDAT) AND IN (GENEVL) ARE CONSISTENT
C     WITH THE INPUT FORMATS TO BE FOUND IN THE SIMPLE DRIVER PROGRAM
C     WHICH IS INCLUDED WITH THE SUITE.
C
C     THE DATA AND ARRAY DECLARATIONS ARE CONSISTENT WITH THE FITTING
C     OF PROBLEMS HAVING 3 VARIABLES ON 125 POINTS, AND WITH THE
C     EVALUATION OF THOSE SAME FITS ON 20 POINTS. 20 ORTHOGONAL BASIS
C     MULTINOMIALS ARE USED, AND THE DEGREE PARAMETERS (FITDEG) AND
C     (EVLDEG) ARE IGNORED.
C
C     LATEST UPDATE... MARCH 9, 1984
C     ***************
C
      DATA FITDEG/-1/
      DATA EVLDEG/-1/
      DATA NFPOLS/20/
      DATA DIMEN/3/
      DATA NFPTS/125/
      DATA NEPTS/20/
      DATA NEPOLS/20/
C
      CALL GENDAT(DIMEN,NFPTS,FITDEG,NFPOLS)
C
      CALL GENEVL(DIMEN,NEPTS,EVLDEG,NEPOLS)
C
      STOP
      END

      SUBROUTINE GENDAT(DIMEN,NFPTS,FITDEG,NFPOLS)
C
      INTEGER DIMEN,NFPTS,FITDEG,NFPOLS,I,J,K
      DOUBLE PRECISION X,Y,Z,W,F
C
C     ***************
C     A SUBROUTINE TO GENERATE AND PRINT 125 DATA POINTS FOR THE
C     FUNCTION
C
```

```
C                    SIN(X)
C          F(X,Y,Z) = ------ * COS(Y) + EXP(Z)
C                      X
C
C    ***************
C
C
      WRITE (6,6000) DIMEN,FITDEG,NFPOLS,NFPTS
      DO 5 I = 1,5
        DO 5 J = 1,5
          DO 5 K = 1,5
            X = DBLE(FLOAT(I)/5.0D+00)
            Y = DBLE(FLOAT(J)/5.0D+00)
            Z = DBLE(FLOAT(K)/5.0D+00)
            W = 1.0D+00
            F = (DSIN(X)/X)*DCOS(Y) + DEXP(Z)
            WRITE (6,6010) W,X,Y,Z,F
    5 CONTINUE
      RETURN
C
 6000 FORMAT(4I5)
 6010 FORMAT(5D14.6)
      END

      SUBROUTINE GENEVL(DIMEN,NEPTS,EVLDEG,NEPOLS)
C
      INTEGER DIMEN,NEPTS,I,NEPOLS,EVLDEG
      DOUBLE PRECISION X,Y,Z
C
C     ***************
C     A SUBROUTINE TO GENERATE 20 EVALUATION POINTS.
C     ***************
C
      WRITE (6,6000) EVLDEG,NEPOLS,NEPTS
      DO 10 I = 1,20
        X = DBLE(FLOAT(I)/20.0D+00)
        Y = DBLE(FLOAT(I)/20.0D+00)
        Z = DBLE(FLOAT(I)/20.0D+00)
        WRITE (6,6010) X,Y,Z
   10 CONTINUE
      RETURN
C
 6000 FORMAT(3I5)
 6010 FORMAT(3D14.6)
C
      END
```

8.3. Multinomial Fitting Subroutines

```
      SUBROUTINE ALLOT(DEGREE,NPOLYS,NPTS,DIMEN,IWORK,IWKLEN,
     +                 IREQD,DREQD,ERROR)
C
      INTEGER IREQD,DREQD,ALFL,ERROR,NPOLYS,DEGREE,DIMEN,NPTS
```

```
      INTEGER NEWSTT,PSIWID,KMXBAS,STARTJ,KJP1D2,INDEX,IWKLEN
      INTEGER IWORK(IWKLEN)
C
C     ***************
C     PURPOSE
C     -------
C
C     $ALLOT$ CHECKS FOR SUFFICIENCY THE DECLARED DIMENSIONS OF THE
C     WORK ARRAYS USED BY THE SUBROUTINE $CONSTR$.  VARIOUS SIZES OF
C     SUB-ARRAYS ARE COMPUTED AND REPORTED.
C
C     THIS ROUTINE IS CALLED BY $CONSTR$.  IT IS NOT CALLED DIRECTLY
C     BY THE USER.
C
C     THIS ROUTINE CALLS $BASIZ$ AND $TABLE$ FOR THE SUBSTANTIVE
C     COMPUTATIONS.
C
C     VARIABLES
C     ---------
C
C     $DEGREE$ - (PASSED/RETURNED)
C        IGNORED IF < 0.
C        IF $DEGREE$ >= 0 THEN $DEGREE$ IS CHECKED AGAINST $NPTS$.
C        THE VALUE OF $DEGREE$ WILL BE REDUCED IF THERE IS A BASIS OF
C        MULTINOMIALS, ALL OF DEGREE <= $DEGREE$, OF CARDINALITY
C        $NPTS$
C     $NPOLYS$ - (PASSED/RETURNED)
C        IGNORED IF $DEGREE$ >= 0.
C        IF $DEGREE$ < 0 THEN THE VALUE OF $NPOLYS$ WILL BE TAKEN AS
C        THE SIZE OF THE BASIS OF MULTINOMIALS TO BE USED IN THE FIT.
C        $NPOLYS$ MUST SATISFY NPOLYS < $NPTS$ AND $NPOLYS$ >= 1
C     $NPTS$ -- (PASSED)
C        THE NUMBER OF DATA POINTS TO BE USED IN THE FIT.
C        $NPTS$ MUST BE >= 1.
C     $DIMEN$ -- (PASSED)
C        THE NUMBER OF VARIABLES.
C     $IWORK$ -- (RETURNED)
C        AN INTEGER WORK ARRAY OF LENGTH AT LEAST
C           IF $DEGREE$ >= 0 THEN
C              4*BINOMIAL($DIMEN$+$DEGREE$,$DIMEN$)
C                +($DIMEN$)*($DEGREE$)
C           ELSE
C              4*BINOMIAL($DIMEN$+D,D)+($DIMEN$)*D
C        WHERE D IS THE MINIMUM CARDINALITY OF A BASIS OF DEGREE
C        $DEGREE$ SUCH THAT
C           BINOMIAL($DIMEN$+ABS($DEGREE$),$DIMEN$) >= $NPOLYS$
C     $IWKLEN$ - (PASSED)
C        THE LENGTH OF $IWORK$
C     $IREQD$ -- (RETURNED)
C        THE SIZE OF THE INTEGER WORK ARRAY REQUIRED BY $CONSTR$ FOR
C        THE FIT SPECIFIED BY THE 4 INPUT PARAMETERS.
C     $DREQD$ -- (RETURNED)
C        THE SIZE OF THE DOUBLE PRECISION WORK ARRAY REQUIRED BY
C        $CONSTR$ FOR THE FIT SPECIFIED BY THE 4 INPUT PARAMETERS.
```

```
C   $ERROR$ – (RETURNED)
C     0 IF $NPOLYS$, $DIMEN$, $DEGREE$, $NPTS$ AND $IWKLEN$ ARE
C       VALID AND CONSISTENT WITH EACH OTHER.
C     1 IF $DEGREE$ >= 0 BUT THERE IS AN INTERPOLATING MULTINOMIAL
C       OF SMALLER DEGREE OR IF $DEGREE$ < 0 AND $NPOLYS$ > $NPTS$
C     2 IF $DEGREE$ < 0 AND $NPOLYS$ <= 0
C     3 IF $NPTS$ < 1 AND/OR $DIMEN$ < 1
C     4 IF $IWKLEN$ IS TOO SMALL (SET $IWKLEN$ TO THE VALUE RETURNED
C       IN $IREQD$ TO RESOLVE THIS PROBLEM)
C
C   NOTE THAT $DEGREE$, $NPOLYS$, $PSIWID$ AND $ALFL$ ARE RETURNED
C   IN $IWORK$(1-4), RESPECTIVELY.
C
C   DATE LAST MODIFIED
C   ---- -- ----
C   JANUARY 30, 1984
C   ****************
C
    CALL BASIZ(DEGREE,NPTS,DIMEN,NPOLYS,ERROR)
    IF ( ERROR .GE. 2 ) RETURN
    IREQD = 4 * NPOLYS + DEGREE * DIMEN
    IF ( IWKLEN .GE. IREQD ) GO TO 5
      ERROR = 4
      RETURN
5   NEWSTT = 4 * NPOLYS + 1
    CALL TABLE(DEGREE,DIMEN,NPOLYS,IWORK,IWORK(NEWSTT),ALFL)
    IWORK(1) = DEGREE
    IWORK(2) = NPOLYS
C
C   ****************
C   FORCE $ALFL$ TO BE AT LEAST 1 SO THAT DIMENSION STATEMENTS
C   USING $ALFL$ DO NOT BOMB.
C   ****************
C
    IF ( ALFL .GT. 1 ) ALFL = ALFL - 1
    IWORK(4) = ALFL
C
C   ****************
C
C   ARRAY        LENGTH
C   -----        ------
C   $MAXABS$      $DIMEN$ + 1
C   $ALPHA$       $ALFL$
C   $C$           $NPOLYS$
C   $SUMSQS$      $NPOLYS$
C
C   THE NUMBER OF COLUMNS IN $PSI$, $PSIWID$, IS DETERMINED BY
C   $PSIWID$ = MIN(SET OF(R,$INDEXS$(3,J)) = R WHERE J > $NPOLYS$)
C   THIS INSURES THAT IF THE USER EXTENDS THE BASIS, ALL THE $PSI$
C   REQUIRED WILL CERTAINLY BE STORED
C
C   IF DEGREE($NPOLYS$) <= 2 THEN
C     $PSIWID$ = $NPOLYS$
C   ELSE
```

```
C     IF K = $DIMEN$ THEN
C       $PSIWID$ = $NPOLYS$ - $NEWKJ(1,DEGREE($NPOLYS$ - 1)$ + 1
C     ELSE
C       $PSIWID$ = $NPOLYS$
C              + 1
C              - (
C                THE SMALLER OF
C                 $NEWKJ(K+1,DEGREE($NPOLYS$)-1)$
C                AND
C                 $INDEXS(3,$NPOLYS$
C               )
C
C    ***************
C
      IF ( DEGREE .GT. 2 ) GO TO 10
        PSIWID = NPOLYS
      GO TO 40
   10   KMXBAS = IWORK(4 * NPOLYS - 2)
        IF ( KMXBAS .NE. DIMEN ) GO TO 20
          PSIWID = NPOLYS - IWORK(4 * NPOLYS - 1)
        GO TO 40
   20     INDEX = 4 * NPOLYS + (DEGREE - 3) * DIMEN + KMXBAS + 1
          KJP1D2 = IWORK(INDEX)
          STARTJ = IWORK(4 * NPOLYS - 1)
          IF ( STARTJ .GT. KJP1D2 ) GO TO 30
            PSIWID = NPOLYS - STARTJ + 1
          GO TO 40
   30       PSIWID = NPOLYS - KJP1D2 + 1
   40   IWORK(3) = PSIWID
      DREQD = 2 * NPOLYS + DIMEN + 1 + NPTS * PSIWID + ALFL
      RETURN
      END

      SUBROUTINE BASIZ(DEGREE,NPTS,DIMEN,NPOLYS,ERROR)
C
      INTEGER TOP,BOT,DEGREE,NPTS,DIMEN,NPOLYS,ERROR,I,ROWLEN
C
C     ***************
C     PURPOSE
C     ——
C
C     IF $DEGREE$ >= 0 THEN
C       FIND THE SIZE OF A BASIS REQUIRED EITHER TO
C       1) APPROXIMATE THE DATA WITH A POLYNOMIAL OF DEGREE $DEGREE$
C       OR TO
C       2) SPAN THE SPACE OF POLYNOMIALS OF DEGREE <= THE SMALLEST
C         DEGREE OF POLYNOMIAL WHICH INTERPOLATES THE DATA.
C       IN CASE 1 $ERROR$ = 0.
C       IN CASE 2 $ERROR$ = 1.
C     ELSE
C       IF $NPOLYS$ >= 1 THEN
C         IF NPOLYS > NPTS THEN
C           SET $NPOLYS$ = $NPTS$, FIND THE SMALLEST DEGREE OF A
C           POLYNOMIAL WHICH INTERPOLATES THE DATA, AND SET
```

```
C         $ERROR$ = 1.
C       ELSE
C         FIND THE LARGEST DEGREE $DEGREE$ OF A POLYNOMIAL  IN
C         A  BASIS OF $NPOLYS$ POLYNOMIALS GENERATED ACCORDING
C         TO OUR ORDERING AND SET $ERROR$ = 0.
C     ELSE
C       $ERROR$ = 2
C
C    THIS SUBROUTINE IS CALLED BY $ALLOT$.  IT IS NOT  CALLED  BY
C    THE USER DIRECTLY.
C
C    DATE LAST MODIFIED
C    ---- -- -----
C    MARCH 10, 1984
C    **************
C
     ERROR = 0
     IF ( NPTS .GE. 1 .AND. DIMEN .GE. 1 ) GO TO 10
       ERROR = 3
       RETURN
C
 10  CONTINUE
     IF ( DEGREE .LT. 0 ) GO TO 30
C
      ROWLEN = 1
      NPOLYS = 1
      TOP = DIMEN - 1
      BOT = 0
      IF ( DEGREE .LT. 1 )  GO TO 30
        DO 20 I=1,DEGREE
          TOP = TOP + 1
          BOT = BOT + 1
          ROWLEN = (ROWLEN*TOP)/BOT
          NPOLYS = NPOLYS + ROWLEN
  20    CONTINUE
C
  30 CONTINUE
     IF ( NPOLYS .GE. 1 ) GO TO 40
          ERROR = 2
          RETURN
  40 CONTINUE
     IF ( NPOLYS .LT. NPTS ) GO TO 50
          NPOLYS = NPTS
          ERROR = 1
  50 CONTINUE
     ROWLEN = 1
     I = 1
     DEGREE = 0
     TOP = DIMEN - 1
     BOT = 0
  60 CONTINUE
     IF ( I .GE. NPOLYS )  GO TO 70
        TOP = TOP + 1
        BOT = BOT + 1
```

```
      ROWLEN = (ROWLEN*TOP)/BOT
      I = I + ROWLEN
      DEGREE = DEGREE + 1
      IF ( I .LT. NPOLYS ) GO TO 60
   70 CONTINUE
      RETURN
      END

      SUBROUTINE CONSTR(DIMEN,FITDEG,NFPOLS,NFPTS,
     +                  FITCDS,NCROWS,FITVLS,WTS,
     +                  RESIDS,NEWFIT,ERROR,FITIWK,
     +                  FITDWK,FIWKLN,FDWKLN,IREQD,DREQD)
C
      INTEGER NFPOLS,ONPLYS,FITDEG,NFPTS,DIMEN,OLDEG,FIWKLN,FDWKLN
      INTEGER ERROR,IREQD,DREQD,OPSWID,OLALFL,INDSTT,P,DIMP1,NCROWS
      INTEGER NEWSTT,MAXSTT,ALFSTT,PSISTT,CSTT,SSQSTT,PSIWID,ALFL
      INTEGER FITIWK(FIWKLN)
      DOUBLE PRECISION FITDWK(FDWKLN),FITCDS(NCROWS,DIMEN)
      DOUBLE PRECISION FITVLS(NFPTS),RESIDS(NFPTS)
      DOUBLE PRECISION WTS(NFPTS)
      DOUBLE PRECISION SCALE
      LOGICAL NEWFIT
C
C     ***************
C     PURPOSE
C     -------
C
C     THIS SUBROUTINE CONSTRUCTS A LEAST-SQUARES MULTINOMIAL FIT TO
C     GIVEN DATA USING A BASIS OF ORTHOGONAL MULTINOMIALS.
C
C     THE DATA FOR THE FIT IS GIVEN IN THE ARRAYS $FITCDS$, $FITVLS$,
C     AND $WTS$.  $FITCDS$ IS A DOUBLE-PRECISION MATRIX, EACH ROW OF
C     WHICH CONTAINS AN OBSERVATION POINT (ORDERED COLLECTION OF
C     VARIABLE VALUES).  $FITVLS$ IS A DOUBLE-PRECISION, SINGLY-
C     INDEXED ARRAY, EACH ELEMENT OF WHICH CONTAINS AN OBSERVED
C     FUNCTION VALUE CORRESPONDING TO AN OBSERVATION POINT.  $WTS$ IS
C     A DOUBLE-PRECISION, SINGLY-INDEXED ARRAY, EACH ELEMENT OF WHICH
C     IS A NONNEGATIVE WEIGHT FOR THE CORRESPONDING OBSERVATION.
C
C     THE FIT WHICH IS PRODUCED IS A MULTINOMIAL EXPRESSED IN THE FORM
C
C      C  PSI  (X ,...,X    ) +...+ C      PSI     (X ,...,X    )
C       1   1  1     DIMEN         NFPOLS   NFPOLS  1     DIMEN
C
C     WHERE THE VALUE OF $NFPOLS$ WILL BE AS GIVEN (IF $FITDEG$ < 0)
C     OR AS COMPUTED BY $CONSTR$ TO GIVE A FULL-DEGREE FIT (IN CASE
C     $FITDEG$ IS SPECIFIED >= 0).  THE ELEMENTS
C
C        PSI  (X ,...,X    )
C          K  1     DIMEN
C
C     FORM A BASIS FOR THE MULTINOMIALS WHICH IS ORTHOGONAL WITH
C     RESPECT TO THE WEIGHTS AND OBSERVATION POINTS.
C
```

```
C   THE EXTENT OF THE FIT CAN BE SPECIFIED IN ONE OF TWO WAYS.
C      IF THE PARAMETER $FITDEG$ IS SET >= 0, THEN A COMPLETE BASIS
C      FOR THE MULTINOMIALS OF DEGREE = $FITDEG$ WILL BE USED. (AN
C      ERROR WILL BE FLAGGED IF THIS WILL REQUIRE MORE BASIS
C      MULTINOMIALS THAN THE NUMBER OF DATA POINTS WHICH WERE
C      GIVEN.)
C      IF THE PARAMETER $FITDEG$ IS < 0, THEN $NFPOLS$ WILL BE
C      TAKEN AS THE COUNT OF THE NUMBER OF BASIS MULTINOMIALS TO BE
C      USED FOR A PARTIAL-DEGREE FIT. (AN ERROR WILL BE FLAGGED IF
C      $NFPOLS$ < 0.)
C
C   NOTE, THE CALL TO $CONSTR$ WITH $NEWFIT$ = .TRUE. CAN BE MADE
C      WITH THE PARAMETERS SET FOR THE MAXIMUM FIT DESIRED.
C      SEVERAL SUBSEQUENT CALLS TO $CONSTR$ WITH $NEWFIT$ = .FALSE.
C      CAN BE MADE WITH SMALLER VALUES OF $FITDEG$ OR $NFPOLS$ AS
C      MAY BE DESIRED TO OBTAIN A PARTIAL FIT.
C
C   VARIABLES
C   ---------
C
C   $DIMEN$ – (INTEGER) – (PASSED)
C      THE NUMBER OF VARIABLES.
C   $FITDEG$ - (INTEGER) – (PASSED/RETURNED)
C      IGNORED IF < 0.
C      IF $DEGREE$ >= 0 THEN $DEGREE$ IS CHECKED AGAINST $NFPTS$.
C      THE VALUE OF $DEGREE$ WILL BE REDUCED IF THERE IS A BASIS OF
C      MULTINOMIALS, ALL OF DEGREE <= $DEGREE$, OF CARDINALITY
C      $NFPTS$. SEE $ERROR$ BELOW.
C   $NFPOLS$ - (INTEGER) – (PASSED/RETURNED)
C      IGNORED IF $DEGREE$ >= 0.
C      IF $DEGREE$ < 0 THEN THE VALUE OF $NFPOLS$ WILL BE TAKEN AS
C      THE SIZE OF THE BASIS OF MULTINOMIALS TO BE USED IN THE FIT.
C      $NFPOLS$ MUST SATISFY $NFPOLS$ < $NFPTS$ AND $NFPOLS$ >= 1
C      SEE $ERROR$ BELOW.
C   $NFPTS$ — (INTEGER) – (PASSED)
C      THE NUMBER OF DATA POINTS TO BE USED IN THE FIT.
C      $NFPTS$ MUST BE >= 1. SEE $ERROR$ BELOW.
C   $FITCDS$ – (DOUBLE-PRECISION, 2-SUBSCRIPT ARRAY) – (PASSED)
C      $FITCDS$(P,K) IS THE VALUE OF THE K-TH VARIABLE AT THE P-TH
C      DATA POINT.
C   $NCROWS$ – (INTEGER) – (PASSED)
C      THE ROW DIMENSION DECLARED FOR $FITCDS$ IN THE CALLING
C      PROGRAM.
C   $FITVLS$ – (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) – (PASSED)
C      $FITVLS$(P) IS THE OBSERVED FUNCTION VALUE OF THE P-TH DATA
C      POINT.
C   $WTS$ —– (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) – (PASSED)
C      $WTS$(P) IS THE WEIGHT ATTACHED TO THE P-TH DATA POINT.
C   $RESIDS$ – (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) – (RETURNED)
C      $RESIDS$(P) IS THE DIFFERENCE BETWEEN THE FITTED FUNCTION AT
C      POINT P AND $FITVLS$(P).
C   $NEWFIT$ – (LOGICAL) – (PASSED)
C      A LOGICAL FLAG. IF $NEWFIT$=.TRUE., THEN THIS IS THE FIRST
C      FIT TO BE CARRIED OUT WITH THE DATA TO BE FOUND IN THE OTHER
```

```
C     PARAMETERS TO $CONSTR$, AND SPACE FOR A FIT IS TO BE
C     ALLOCATED. IF $NEWFIT$=.FALSE., THEN A FIT OF ANOTHER DEGREE
C     CAN BE CONSTRUCTED IN THE SPACE ALLOCATED ON A PREVIOUS CALL
C     WITH THE SAME DATA, AND CERTAIN INITIALIZATION STEPS ARE BY-
C     PASSED.
C   $ERROR$ - (INTEGER) - (RETURNED)
C     0 IF $NPOLYS$, $DIMEN$, $DEGREE$, $NPTS$ AND $IWKLEN$ ARE
C     VALID AND CONSISTENT WITH EACH OTHER.
C     1 IF $DEGREE$ >= 0 BUT THERE IS AN INTERPOLATING MULTINOMIAL
C     OF SMALLER DEGREE OR IF $DEGREE$ < 0 AND $NPOLYS$ > $NPTS$.
C     2 IF $DEGREE$ < 0 AND $NPOLYS$ <= 0.
C     3 IF $NPTS$ < 1 AND/OR $DIMEN$ < 1.
C     4 IF $IWKLEN$ AND/OR $DWKLEN$ IS TOO SMALL. (SET $IWKLEN$ TO
C     THE VALUE RETURNED IN $IREQD$, AND SET $DWKLEN$ TO THE VALUE
C     RETURNED IN $IREQD$ TO RESOLVE THIS PROBLEM.)
C     5 $NEWFIT$ = .FALSE. BUT $ONPLYS$ >= $NFPOLS$. $EVAL$ CAN BE
C     CALLED REQUESTING A SMALLER BASIS THAN WAS GENERATED.
C     6 ERROR RETURN FROM $INCDG$. THERE IS NO MORE ROOM IN THE
C     $FITDWK$ AND/OR $FITIWK$ ARRAYS TO INCLUDE MORE TERMS IN THE
C     FIT. (CAUSED BY A SUCCESSION OF CALLS TO $CONSTR$ WITH THE
C     FLAG $NEWFIT$ SET .FALSE., REQUESTING EVER HIGHER ORDER
C     FITS, IN WHICH THE DIMENSIONING AND ORDER INFORMATION GIVEN
C     ON THE FIRST CALL ($NEWFIT$=.TRUE.) IS EXCEEDED.)
C   $FITIWK$ - (INTEGER, 1-SUBSCRIPT ARRAY) - (RETURNED)
C     AN INTEGER WORK ARRAY OF LENGTH $FIWKLN$. UPON RETURN FROM
C     A CALL TO $CONSTR$ WITH $NEWFIT$ SET .TRUE., SOME DIMENSION
C     AND ARRAY-LENGTH INFORMATION WILL BE INSERTED. UPON RETURN
C     FROM A CALL TO $CONSTR$ WITH $NEWFIT$ SET .FALSE., DETAILED
C     INDEXING INFORMATION (LOCATION OF COEFFICIENTS IN $FITDWK$,
C     ETC.) IS INSERTED.
C   $FITDWK$ - (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY) - (RETURNED)
C     A DOUBLE PRECISION ARRAY OF LENGTH $FDWKLN$. UPON RETURN
C     FROM $CONSTR$ WITH $NEWFIT$ SET .FALSE., THE FULL DETAILS
C     OF THE REQUESTED FIT (COEFFICIENTS, ETC.) WILL BE INSERTED.
C   $FIWKLN$ - (INTEGER) - (PASSED)
C     THE LENGTH OF THE ARRAY $FITIWK$.
C   $FDWKLN$ - (INTEGER) - (PASSED)
C     THE LENGTH OF THE ARRAY $FITDWK$.
C   $IREQD$ - (INTEGER) - (PASSED)
C     THE LENGTH WHICH THE ARRAY $FITIWK$ REALLY NEEDS TO BE.
C   $DREQD$ - (INTEGER) - (PASSED)
C     THE LENGTH WHICH THE ARRAY $FITDWK$ REALLY NEEDS TO BE.
C
C
C   NOTE, THE 10 AND 70 LOOPS (I.E. THE LOOPS FOR SCALING THE
C   RESIDUALS) DEPEND ON THE SCALING SCHEME USED. THE RESIDUAL
C   SCALING MUST BE CONSISTENT WITH THAT DEFINED BY $SCALPM$,
C   $SCALDN$, AND $SCALUP$.
C
C   $CONSTR$ CALLS $ALLOT$, $RESTRT$, $INCDG$, AND $GNRTP$
C
C
C   DATE LAST MODIFIED
C   -- -- ----
```

```
C    MARCH 9, 1984
C    ***************
C
     DIMP1 = DIMEN + 1
C
     IF ( NEWFIT ) GO TO 20
       OLDEG = FITIWK(1)
       ONPLYS = FITIWK(2)
       OPSWID = FITIWK(3)
       OLALFL = FITIWK(4)
       SCALE = FITDWK(DIMP1)
       SCALE = SCALE * SCALE
C
       IF ( NFPOLS .GT. ONPLYS ) GO TO 10
         ERROR = 5
         RETURN
  10   CONTINUE
  20 CONTINUE
C
     CALL ALLOT(FITDEG,NFPOLS,NFPTS,DIMEN,FITIWK,FIWKLN,IREQD,DREQD,
    +          ERROR)
     IF ( ERROR .GE. 2 ) RETURN
C
     IF ( FDWKLN .GE. DREQD ) GO TO 30
       ERROR = 4
       RETURN
  30 CONTINUE
C
     PSIWID = FITIWK(3)
     ALFL = FITIWK(4)
     INDSTT = 1
     NEWSTT = 4 * NFPOLS + INDSTT
     MAXSTT = 1
     ALFSTT = MAXSTT + DIMP1
     CSTT = ALFSTT + ALFL
     SSQSTT = CSTT + NFPOLS
     PSISTT = SSQSTT + NFPOLS
C
     IF ( NEWFIT ) GO TO 50
C
       DO 40 P = 1,NFPTS
         RESIDS(P) = RESIDS(P) / SCALE
  40   CONTINUE
C
       CALL RESTRT(PSIWID,FITDWK,DREQD,ONPLYS,OPSWID,OLALFL,CSTT,
    +              SSQSTT,PSISTT,NFPTS,DIMEN)
       CALL INCDG(FITDEG,FITDWK(ALFSTT),FITDWK(PSISTT),FITIWK(INDSTT),
    +             FITIWK(NEWSTT),FITDWK(SSQSTT),FITCDS,
    +             NFPOLS,DIMEN,NFPTS,FITVLS,RESIDS,
    +             FITDWK(CSTT),PSIWID,WTS,ALFL,ONPLYS,OLDEG,ERROR)
       GO TO 60
C
  50   CONTINUE
       CALL GNRTP(FITDEG,FITDWK(ALFSTT),
```

```
     +          FITDWK(PSISTT),FITIWK(INDSTT),
     +          FITIWK(NEWSTT),FITDWK(SSQSTT),
     +          FITCDS,NFPOLS,DIMEN,NFPTS,FITVLS,RESIDS,
     +          FITDWK(CSTT),PSIWID,WTS,ALFL,DIMP1,FITDWK(MAXSTT))
       SCALE = FITDWK(DIMEN + 1)
       SCALE = SCALE * SCALE
C
  60 CONTINUE
C
     DO 70 P = 1,NFPTS
       RESIDS(P) = RESIDS(P) * SCALE
  70 CONTINUE
     RETURN
     END

     SUBROUTINE EVAL(DIMEN,EVLDEG,NEPOLS,NEPTS,EVLCDS,EVLVLS,
     +          ERROR,FITIWK,FITDWK,FIWKLN,FDWKLN,EVLDWK,EDWKLN)
C
     INTEGER FIWKLN,FDWKLN,NEPOLS,NEPTS,DIMEN,ERROR,MAXSTT,ALFSTT,CSTT
     INTEGER GBASIZ,ALFL,DIMP1,EVLDEG,TOP,BOT,CURDEG,EDWKLN
     INTEGER FITIWK(FIWKLN)
     DOUBLE PRECISION FITDWK(FDWKLN),EVLDWK(EDWKLN),EVLCDS(NEPTS,DIMEN)
     DOUBLE PRECISION EVLVLS(NEPTS)
C
C    **********
C    PURPOSE
C    -------
C
C    THIS SUBROUTINE EVALUATES THE LEAST-SQUARES MULTINOMIAL FIT
C    WHICH HAS BEEN PREVIOUSLY PRODUCED BY $CONSTR$.  EITHER THE FULL
C    MULTINOMIAL AS PRODUCED MAY BE EVALUATED, OR ONLY AN INITIAL
C    SEGMENT THEREOF.  AS IN THE CASE WITH $CONSTR$, IT IS POSSIBLE
C    (1) TO SPECIFY MULTINOMIALS OF A FULL GIVEN DEGREE, OR
C    (2) TO SPECIFY THE NUMBER OF ORTHOGONAL BASIS ELEMENTS TO
C       ACHIEVE A PARTIAL-DEGREE FIT.
C
C    IN CASE (1), THE DESIRED DEGREE IS GIVEN AS THE VALUE OF
C    $EVLDEG$ (WHICH MUST BE NONNEGATIVE AND NOT GREATER THAN THE
C    VALUE USED FOR $FITDEG$ IN $CONSTR$), AND THE PARAMETER $NEPOLS$
C    WILL BE SET BY $EVAL$ TO SPECIFY THE NUMBER OF BASIS ELEMENTS
C    REQUIRED.  IF $EVLDEG$ < $FITDEG$ IS GIVEN, THEN ONLY THE
C    INITIAL PORTION OF THE FITTING MULTINOMIAL (OF DEGREE $EVLDEG$)
C    WILL BE EVALUATED.
C
C    IN CASE (2), $EVLDEG$ IS TO BE SET NEGATIVE, IN WHICH CASE THE
C    VALUE OF $NEPOLS$ (WHICH MUST BE POSITIVE AND NOT GREATER THAN
C    THE VALUE USED FOR $NFPOLS$ IN $CONSTR$) WILL BE TAKEN AS
C    DEFINING THE INITIAL PORTION OF THE FITTING MULTINOMIAL TO BE
C    EVALUATED.
C
C    IF $NEPOLS$ = $NFPOLS$ (WITH $EVLDEG$ < 0), OR $EVLDEG$ =
C    $FITDEG$ (WITH $EVLDEG$ > 0), THEN THE FULL MULTINOMIAL
C    GENERATED BY $CONSTR$ WILL BE EVALUATED.
C
```

```
C   THE EVALUATION WILL TAKE PLACE FOR EACH OF THE POINTS
C   (COLLECTION OF VARIABLE VALUES) GIVEN AS A ROW OF THE MATRIX
C   $EVLCDS$. THE VALUES PRODUCED FROM THE FULL, OR PARTIAL,
C   MULTINOMIAL WILL BE PLACED IN THE ARRAY $EVLVLS$.
C
C   VARIABLES
C   ---------
C
C   $DIMEN$ -- (INTEGER) -- (PASSED)
C      THE NUMBER OF VARIABLES.
C   $EVLDEG$ -- (INTEGER) -- (PASSED)
C      IF $EVLDEG$ < 0, THEN THIS PARAMETER WILL BE IGNORED.
C      IF $EVLDEG$ >= 0, THEN THE VALUE OF $EVLDEG$ MUST SATISFY
C      $EVLDEG$ <= (THE DEGREE OF THE APPROXIMATING MULTINOMIAL
C      GENERATED IN $CONSTR$). IN THIS CASE $EVLDEG$ WILL SPECIFY
C      THE DEGREE OF THE INITIAL PORTION OF THE FITTING MULTINOMIAL
C      TO BE EVALUATED.
C   $NEPOLS$ -- (INTEGER) -- (PASSED/RETURNED)
C      IF $EVLDEG$ >= 0, THEN THIS PARAMETER WILL BE IGNORED.
C      IF $EVLDEG$ < 0, THEN THE PARTIAL MULTINOMIAL INVOLVING THE
C      FIRST $NEPOLS$ ORTHOGONAL BASIS FUNCTIONS WILL BE EVALUATED
C      AT THE POINTS GIVEN BY $EVLCDS$. THE RESULTING VALUES WILL
C      BE STORED IN $EVLVLS$.
C      THE VALUE OF $NEPOLS$ MUST BE >= 1 AND <= (THE SIZE OF THE
C      BASIS GENERATED IN $CONSTR$), WHICH WAS RETURNED AS THE
C      VALUE OF $NFPOLS$.
C      $NEPOLS$ WILL BE CHANGED IF $EVLDEG$ > 0 TO GIVE THE SIZE OF
C      BASIS REQUIRED FOR THE MULTINOMIAL OF DEGREE $EVLDEG$.
C   $NEPTS$ -- (INTEGER) -- (PASSED)
C      THE NUMBER OF EVALUATION POINTS.
C   $EVLCDS$ -- (INTEGER ) -- (PASSED)
C      $EVLCDS$(P,K) IS THE VALUE OF THE K-TH VARIABLE AT THE P-TH
C      EVALUATION POINT.
C   $EVLVLS$ -- (INTEGER) -- (RETURNED)
C      $EVLVLS$(P) IS THE VALUE OF THE EVALUATED MULTINOMIAL AT THE
C      P-TH EVALUATION POINT.
C   $ERROR$ -- (INTEGER) -- (RETURNED)
C        0 ..... IF NO ERRORS
C       -1 ..... IF $NEPOLS$ > $NPOLYS$ OR $NEPOLS$ < 1
C       -2 ..... IF $NEPTS$ < 1 OR $DIMEN$ < 1
C      $NEPOLS$ ... IF $NEPOLS$ > $EDWKLN$
C   $FITIWK$ -- (INTEGER, 1-SUBSCRIPT ARRAY) -- (PASSED)
C      THE INTEGER WORK ARRAY OF LENGTH $FIWKLN$ THAT WAS USED IN
C      $CONSTR$.
C   $FITDWK$ -- (DOUBLE-PRECISION, 2-SUBSCRIPT ARRAY) -- (PASSED)
C      THE DOUBLE PRECISION WORK ARRAY OF LENGTH $FDWKLN$ THAT WAS
C      USED IN $CONSTR$.
C   $FIWKLN$ -- (INTEGER) -- (PASSED)
C      THE LENGTH OF $FITIWK$.
C   $FDWKLN$ -- (INTEGER) -- (PASSED)
C      THE LENGTH OF $FITDWK$.
C   $EVLDWK$ -- (DOUBLE-PRECISION, 1-SUBSCRIPT ARRAY ) -- (RETURNED)
C      A WORK ARRAY OF LENGTH $NEPOLS$ (OR LONGER).
C   $EDWKLN$ -- (INTEGER) -- (PASSED)
```

```
C       THE LENGTH OF $EVLDWK$.
C
C    THE SUBROUTINE  $EVALP$  IS CALLED TO DO THE ACTUAL EVALUATION.
C
C
C    DATE LAST MODIFIED
C    --- -- -----
C    FEBRUARY 2, 1984
C    ****************
C
     ERROR = 0
     GBASIZ = FITIWK(2)
     IF ( EVLDEG .LT. 0 ) GO TO 20
       TOP = 1
       BOT = 1
       DO 10 CURDEG = 1,EVLDEG
         TOP = TOP * (DIMEN + CURDEG)
10       BOT = BOT * CURDEG
       NEPOLS = TOP / BOT
       IF ( EVLDEG .EQ. 0 ) NEPOLS = 1
20   IF ( NEPOLS .LE. GBASIZ .AND. NEPOLS.GE.1 ) GO TO 30
       ERROR = -1
       RETURN
30   IF ( NEPTS .GE. 1 .AND. DIMEN .GE. 1 ) GO TO 40
       ERROR = -2
       RETURN
40   IF ( NEPOLS .LE. EDWKLN ) GO TO 50
       ERROR = NEPOLS
       RETURN
50   DIMP1 = DIMEN + 1
     ALFL = FITIWK(4)
     MAXSTT = 1
     ALFSTT = DIMP1 + MAXSTT
     CSTT = ALFSTT + ALFL
C
C    ****************
C    ALL OF THE REAL WORK IS DONE INSIDE $EVALP$.
C    ****************
C
     CALL EVALP(EVLCDS,FITDWK(CSTT),NEPTS,DIMEN,NEPOLS,FITDWK(ALFSTT),
    +          FITIWK,EVLDWK,EVLVLS,ALFL,FITDWK(MAXSTT),DIMP1)
     RETURN
     END

     SUBROUTINE EVALP(COORD,C,NEPTS,DIMEN,NPOLYS,ALPHA,INDEXS,
    +                 PSI,F,ALFL,MAXABS,DIMP1)
C
     INTEGER DIMEN,NEPTS,NPOLYS,ALFL,DIMP1
     INTEGER JM1,JPRIME,M,P,K,I,J,INDEX
     INTEGER INDEXS(4,NPOLYS)
     DOUBLE PRECISION ALPHA(ALFL),COORD(NEPTS,DIMEN),PSI(NPOLYS)
     DOUBLE PRECISION C(NPOLYS),F(NEPTS),MAXABS(DIMP1)
     DOUBLE PRECISION RUNTOT,RNTOT1
C
```

```
C     ***************
C     PURPOSE
C     -------
C
C     THIS SUBROUTINE PERFORMS THE MAIN WORK OF EVALUATING THE
C     FITTING MULTINOMIAL (OR THE INITIAL PORTION OF IT  WHICH
C     IS REQUESTED BY THE SETTING OF $NEPOLS$, $EVLDEG$ IN THE
C     CALL TO SUBROUTINE $EVAL$.
C
C     THIS SUBROUTINE IS CALLED BY $EVAL$.  IT IS  NOT  CALLED
C     DIRECTLY BY THE USER.
C
C     SUBROUTINES $SCALDN$ AND $SCALUP$ ARE  CALLED  TO SCALE
C     AND TO UNSCALE VALUES.
C
C     DATE LAST MODIFIED
C     ---- --- -----
C     FEBRUARY 2, 1984
C     ***************
C
C     ***************
C     SCALE DOWN THE INDEPENDENT CO-ORDINATES.
C     ***************
C
      DO 10 K = 1,DIMEN
   10   CALL SCALDN(COORD(1,K),NEPTS,MAXABS(K))
C
C     ***************
C     USE THE  BASIS FUNCTION COEFFICIENTS $C$ AND RECURRENCE
C     COEFFICIENTS $ALPHA$ TO EVALUATE THE FITTED MULTINOMIAL
C     AT THE EVALUATION CO-ORDINATES $COORD$.
C     ***************
C
      PSI(1) = 1.0D+00
      DO 40 P = 1,NEPTS
        RNTOT1 = C(1)
        IF ( NPOLYS .EQ. 1 ) GO TO 40
        DO 30 J = 2,NPOLYS
           K = INDEXS(2,J)
           JPRIME = INDEXS(1,J)
           RUNTOT = COORD(P,K) * PSI(JPRIME)
           I = INDEXS(3,J)
           JM1 = J - 1
           DO 20 M = I,JM1
             INDEX = INDEXS(4,J) + M - I
   20          RUNTOT = RUNTOT - PSI(M) * ALPHA(INDEX)
           PSI(J) = RUNTOT
   30       RNTOT1 = RNTOT1 + C(J) * PSI(J)
   40    F(P) = RNTOT1
C
C     ***************
C     SCALE UP THE DEPENDENT COORDINATES.
C     ***************
C
```

```
      CALL SCALUP(F,NEPTS,MAXABS(DIMP1))
      DO 50 K = 1,DIMEN
   50   CALL SCALUP(COORD(1,K),NEPTS,MAXABS(K))
C
      RETURN
      END

      SUBROUTINE GNRTP(DEGREE,ALPHA,PSI,INDEXS,
     +                 NEWKJ,SUMSQS,COORD,NPOLYS,
     +                 DIMEN,NPTS,F,Z,C,PSIWID,WEIGHT,
     +                 ALFL,DIMP1,MAXABS)
C
      INTEGER DEGREE,DIMEN,NPOLYS,NPTS,K,PSIWID,ALFL,P,STTDEG,ONPLYS
      INTEGER ERROR,DIMP1
      INTEGER INDEXS(4,NPOLYS),NEWKJ(DIMEN,DEGREE)
      DOUBLE PRECISION PSI(NPTS,PSIWID),ALPHA(ALFL),F(NPTS)
      DOUBLE PRECISION COORD(NPTS,DIMEN),MAXABS(DIMP1),WEIGHT(NPTS)
      DOUBLE PRECISION Z(NPTS),SUMSQS(NPOLYS),C(NPOLYS)
      DOUBLE PRECISION RUNTOT,RNTOT1
C
C     ***************
C     PURPOSE
C     -------
C
C     THE MULTINOMIAL FIT IS GENERATED INCREMENTALLY, A BASIS ELEMENT
C     AT A TIME.  THIS SUBROUTINE STARTS THE PROCESS OFF BY SETTING UP
C     THE FIRST BASIS ELEMENT, SCALING THE DATA, FINDING THE FIRST
C     COEFFICIENT, AND INITIALIZING THE WORK ARRAY Z. $GNRTP$ THEN
C     CALLS $INCDG$ IF MORE THAN ONE BASIS ELEMENT IS REQUIRED.
C
C     THIS SUBROUTINE IS CALLED BY $CONSTR$.  IT IS NOT CALLED BY THE
C     USER.
C
C     THIS SUBROUTINE CALLS $SCALPM$, $SCALDN$, AND $INCDG$.
C
C     DATE LAST MODIFIED
C     ---- --- -----
C     FEBRUARY 2, 1984
C     ***************
C
C     ***************
C     SET UP THE SCALING.
C     ***************
C
      DO 10 K = 1,DIMEN
        CALL SCALPM(COORD(1,K),NPTS,MAXABS(K))
   10   CALL SCALDN(COORD(1,K),NPTS,MAXABS(K))
      CALL SCALPM(F,NPTS,MAXABS(DIMP1))
      CALL SCALDN(F,NPTS,MAXABS(DIMP1))
C
C     ***************
C     $SUMSQS$(1) = <1,1>
C     C = <F,1> / <1,1>
C      1
```

```
C   ***************
C
      RUNTOT = 0.0D+00
      RNTOT1 = 0.0D+00
      DO 20 P = 1,NPTS
        PSI(P,1) = 1.0D+00
        RNTOT1 = RNTOT1 + WEIGHT(P)
 20     RUNTOT = RUNTOT + F(P) * WEIGHT(P)
      SUMSQS(1) = RNTOT1
      C(1) = RUNTOT / RNTOT1
C
C   ***************
C   Z = F - C
C         1
C   ***************
C
      DO 30 P = 1,NPTS
 30     Z(P) = F(P) - C(1)
C
      IF ( NPOLYS .EQ. 1 ) RETURN
      STTDEG = 1
      ONPLYS = 1
C
      CALL INCDG(DEGREE,ALPHA,PSI,INDEXS,NEWKJ,SUMSQS,
     +          COORD,NPOLYS,DIMEN,NPTS,F,Z,C,PSIWID,
     +          WEIGHT,ALFL,ONPLYS,STTDEG,ERROR)
      RETURN
      END

      SUBROUTINE INCDG(DEGREE,ALPHA,PSI,INDEXS,NEWKJ,
     +                 SUMSQS,COORD,NPOLYS,
     +                 DIMEN,NPTS,F,Z,C,PSIWID,WEIGHT,
     +                 ALFL,ONPLYS,STTDEG,ERROR)
C
      INTEGER JPRIME,P,J,CURDEG,KJ,KJP,L,JPM1,JM1
      INTEGER M,START,JINDEX,JPINDX,Q,J3,J1,J1MJ2,ERROR
      INTEGER J0MJ1,J1M1,STARTA,ONPLYS,ONPP1,STTDEG,INDEX1,INDEX2
      INTEGER DEGREE,NPOLYS,NPTS,DIMEN,PSIWID,ALFL
      DOUBLE PRECISION ALPHA(ALFL),COORD(NPTS,DIMEN),PSI(NPTS,PSIWID)
      DOUBLE PRECISION SUMSQS(NPOLYS),C(NPOLYS),F(NPTS),WEIGHT(NPTS)
      DOUBLE PRECISION Z(NPTS)
      INTEGER INDEXS(4,NPOLYS),NEWKJ(DIMEN,DEGREE)
      DOUBLE PRECISION RUNTOT,RNTOT1,RNTOT2
C
C   ***************
C   PURPOSE
C   ——————
C
C   THE MULTINOMIAL FIT IS GENERATED INCREMENTALLY, A BASIS  ELEMENT
C   AT A TIME.  THIS SUBROUTINE CONTINUES THE PROCESS STARTED OFF BY
C   $GNRTP$.
C
C   THIS SUBROUTINE IS CALLED BY $GNRTP$ AND NOT BY THE USER.
C
```

```
C
C   DATE LAST MODIFIED
C   ---- -- -----
C   FEBRUARY 2, 1984
C   ****************
C
    ERROR = 0
    IF ( ONPLYS .GE. 1 .AND. STTDEG .GE. 1 ) GO TO 10
      ERROR = 6
    RETURN
 10   IF ( INDEXS(2,ONPLYS) .EQ. DIMEN ) GO TO 20
        CURDEG = STTDEG
      GO TO 30
 20     CURDEG = STTDEG + 1
 30   ONPP1 = ONPLYS + 1
    DO 170 J = ONPP1,NPOLYS
      JPRIME = INDEXS(1,J)
      JINDEX = J - (J - 1) / PSIWID * PSIWID
      JPINDX = JPRIME - (JPRIME - 1) / PSIWID * PSIWID
      KJ = INDEXS(2,J)
      START = INDEXS(3,J)
      M = START
      STARTA = INDEXS(4,J) - START
      IF ( CURDEG .EQ. 1 ) GO TO 100
      KJP = INDEXS(2,JPRIME)
      J1 = NEWKJ(KJ,CURDEG - 1)
C
C   ****************
C   CALCULATE THOSE $ALPHA$($J$,$M$) THAT CAN BE CALCULATED FROM
C   PREVIOUSLY CALCULATED ALPHAS.
C   ****************
C
      IF ( KJ .LT. KJP ) GO TO 50
C
C   ****************
C   FIRST CALCULATE THOSE BETWEEN $JPP$ AND THE END OF 2 ROWS BACK.
C   CALCULATE $ALPHA$($J$,$JPP$)
C   ****************
C
      INDEX1 = INDEXS(4,J)
      ALPHA(INDEX1) = SUMSQS(JPRIME) / SUMSQS(START)
C
      M = START + 1
      J3 = NEWKJ(1,CURDEG - 1) - 1
      IF ( M .GT. J3 ) GO TO 50
C
C   ****************
C   $CURDEG$ > 2 IF CONTROL HAS PASSED THE BRANCHES IN THE 3-RD
C   PREVIOUS AND 8-TH PREVIOUS STATEMENTS.
C   ****************
C
        J1MJ2 = J1 - NEWKJ(KJ,CURDEG - 2)
C
        DO 40 L = M,J3
```

```
          Q = J1MJ2 + L
          INDEX1 = STARTA + L
          INDEX2 = INDEXS(4,Q) - INDEXS(3,Q) + JPRIME
 40        ALPHA(INDEX1) = ALPHA(INDEX2) * SUMSQS(JPRIME) /
    +      SUMSQS(L)
C
C    ***************
C    CALCULATE $ALPHA$($J$,$M$) FOR $M$ BETWEEN THE 2
C    RANGES CALCULATED BEFORE USING
C
C      ALPHA ( J , L ) = <X  *  PSI  ,PSI >  / <PSI ,PSI >
C                    K     JP   L        L   L
C                     J
C    ***************
C
       M = J3 + 1
 50     IF ( JPRIME .EQ. J1 ) GO TO 100
          IF ( KJ .EQ. 1 ) GO TO 80
          J1M1 = J1 - 1
          DO 70 L = M,J1M1
            RUNTOT = 0.0D+00
            DO 60 P = 1,NPTS
              INDEX1 = L - (L - 1) / PSIWID * PSIWID
 60            RUNTOT = RUNTOT + COORD(P,KJ) * PSI(P,JPINDX) *
    +           PSI(P,INDEX1) * WEIGHT(P)
            INDEX1 = STARTA + L
 70          ALPHA(INDEX1) = RUNTOT / SUMSQS(L)
C
C    ***************
C    CALCULATE $ALPHA$($J$,$M$) FOR $M$ BETWEEN
C    $NEWKJ$($KJ$,$CURDEG$ - 1)  AND
C    $JP$ - 1.
C    ***************
C
 80       J0MJ1 = NEWKJ(KJ,CURDEG) - J1
          JPM1 = JPRIME - 1
          DO 90 L = J1,JPM1
            Q = J0MJ1 + L
            INDEX1 = STARTA + L
            INDEX2 = INDEXS(4,Q) - INDEXS(3,Q) + JPRIME
 90          ALPHA(INDEX1) = ALPHA(INDEX2) * SUMSQS(JPRIME) /
    +       SUMSQS(L)
          M = JPRIME
C
C    ***************
C    CALCULATE THE REMAINING $ALPHA$($J$,$M$) FROM
C
C     ALPHA ( J , L ) = <X   * PSI  ,PSI > / <PSI ,PSI >
C                     K      JP   L       L   L
C                      J
C    ***************
C
100     JM1 = J - 1
        DO 120 L = M,JM1
```

```
      RUNTOT = 0.0D+00
      DO 110 P = 1,NPTS
        INDEX1 = L - (L - 1) / PSIWID * PSIWID
110      RUNTOT = RUNTOT + COORD(P,KJ) * PSI(P,JPINDX) *
     +   PSI(P,INDEX1) * WEIGHT(P)
      INDEX1 = STARTA + L
120    ALPHA(INDEX1) = RUNTOT / SUMSQS(L)
C
C   ***************
C   NOW CALCULATE THE $PSI$(P,J), $SUMSQS$(J) AND $C$(J) USING
C
C                  J-1
C   PSI  = X  * PSI  -  SUM  ALPHA(J,L) * PSI
C     J    K     JP   L=JPP             L
C
C   SUMSQS  = <PSI ,PSI >
C         J     J   J
C
C   C  = <Z,PSI >
C    J       J
C   ***************
C
130    RNTOT1 = 0.0D+00
      RNTOT2 = 0.0D+00
      DO 150 P = 1,NPTS
        RUNTOT = COORD(P,KJ) * PSI(P,JPINDX)
        JM1 = J - 1
        DO 140 L = START,JM1
          INDEX1 = STARTA + L
          INDEX2 = L - (L - 1) / PSIWID * PSIWID
140        RUNTOT = RUNTOT - ALPHA(INDEX1) * PSI(P,INDEX2)
        PSI(P,JINDEX) = RUNTOT
        RNTOT1 = RNTOT1 + PSI(P,JINDEX) * PSI(P,JINDEX) *
     +                 WEIGHT(P)
150      RNTOT2 = RNTOT2 + Z(P) * PSI(P,JINDEX) * WEIGHT(P)
      SUMSQS(J) = RNTOT1
      C(J) = RNTOT2 / RNTOT1
C
C   ***************
C   CALCULATE THE NEW $Z$($P$) AND THE NEW $SSRES$ USING
C
C   Z = Z - C  * PSI
C         J     J
C   ***************
C
      DO 160 P = 1,NPTS
160      Z(P) = Z(P) - C(J) * PSI(P,JINDEX)
170    IF ( KJ .EQ. DIMEN ) CURDEG = CURDEG + 1
    RETURN
    END

    SUBROUTINE MOVE(OLDARR,NEWARR,START,OLDWID,NEWIDT,COLENG,ERROR)
C
    INTEGER START,OLDWID,NEWIDT,COLENG,BIG,BIG1,LILN,BIGN,I,J
```

```
      INTEGER ERROR,JO,JN,OLDWPS,BIG1PS,BIGP1,K
      DOUBLE PRECISION OLDARR(COLENG,OLDWID),NEWARR(COLENG,NEWIDT)
C
C     ***************
C     PURPOSE
C     ——————
C
C     MOVE COLUMNS 1 THROUGH $OLDWID$ OF $OLDARR$ INTO COLUMNS 1
C     THROUGH $NEWWID$ OF $NEWARR$ USING
C       ($START$ + I) MOD $OLDWID$
C     THROUGH
C       ($START$ + I) MOD $NEWID$
C     FOR
C       I = 0
C     THROUGH
C       I = $OLDWID$ - 1
C     THE MOVEMENT STARTS FROM THE LARGEST COLUM OF $NEWARR$ AND
C     PROCEEDS DOWNWARD TO THE SMALLEST (SO THAT $OLDARR$ AND $NEWARR$
C     CAN BE THE SAME ARRAY).
C
C
C     DATE LAST MODIFIED
C     —— —— ————
C     FEBRUARY 2, 1984
C     ***************
C
      ERROR = 1
      IF ( OLDWID .LT. 1 .OR. NEWIDT .LT. 1 .OR. NEWIDT .LT. OLDWID)
     +       RETURN
      ERROR = 0
      BIG = START + OLDWID - 1
      BIGN = BIG - (BIG - 1) / NEWIDT * NEWIDT
      LILN = START - (START - 1) / NEWIDT * NEWIDT
      IF ( LILN .GT. BIGN ) GO TO 20
        OLDWPS = OLDWID + START
        DO 10 I = 1,OLDWID
          J = OLDWPS - I
          JO = J - (J - 1) / OLDWID * OLDWID
          JN = J - (J - 1) / NEWIDT * NEWIDT
          DO 10 K = 1,COLENG
10          NEWARR(K,JN) = OLDARR(K,JO)
        RETURN
20    BIG1 = NEWIDT - LILN + 1
      BIG1PS = BIG1 + START
      DO 30 I = 1,BIG1
        J = BIG1PS - I
        JO = J - (J - 1) / OLDWID * OLDWID
        JN = J - (J - 1) / NEWIDT * NEWIDT
        DO 30 K = 1,COLENG
30        NEWARR(K,JN) = OLDARR(K,JO)
      BIGP1 = BIG + 1
      DO 40 I = 1,BIGN
        J = BIGP1 - I
        JO = J - (J - 1) / OLDWID * OLDWID
```

```
      JN = J - (J - 1) / NEWIDT * NEWIDT
      DO 40 K = 1,COLENG
40       NEWARR(K,JN) = NEWARR(K,JO)
      RETURN
      END

      SUBROUTINE RESTRT(PSIWID,DWORK,DREQD,ONPLYS,OPSWID,OLALFL,CSTT,
     +                  SSQSTT,PSISTT,NPTS,DIMEN)
C
      INTEGER DREQD,ONPLYS,OPSWID,OLALFL,CSTT
      INTEGER OSSQST,OPSIST,PSIWID,NPTS,ERROR,DIMEN
      INTEGER I,J,SSQSTT,PSISTT,OCST,START,INDEX1,INDEX2
      DOUBLE PRECISION DWORK(DREQD)
C
C     ***************
C     PURPOSE
C     -------
C
C     THIS SUBROUTINE REARRANGES THE WORK  SPACE  (WITH THE HELP
C     OF SUBROUTINE $MOVE$) IN THE EVENT THAT A FIT OF INCREASED
C     DEGREE IS TO BE MADE.
C
C     CALLED INTERNALLY BY $CONSTR$, NOT BY THE USER.
C
C     DATE LAST MODIFIED
C     -- -- ----
C     FEBRUARY 2, 1984
C     ***************
C
      OCST = 2 + DIMEN + OLALFL
      OSSQST = OCST + ONPLYS
      OPSIST = OSSQST + ONPLYS
      START = ONPLYS - OPSWID
      CALL MOVE(DWORK(OPSIST),DWORK(PSISTT),START,OPSWID,PSIWID,NPTS,
     +          ERROR)
C
      DO 5 J = 1,ONPLYS
        I = ONPLYS - J
        INDEX1 = SSQSTT + I
        INDEX2 = OSSQST + I
5       DWORK(INDEX1) = DWORK(INDEX2)
      DO 10 J = 1,ONPLYS
        I = ONPLYS - J
        INDEX1 = CSTT + I
        INDEX2 = OCST + I
10      DWORK(INDEX1) = DWORK(INDEX2)
      RETURN
      END

      SUBROUTINE SCALPM(COORD,NPTS,MAXABS)
C
      INTEGER NPTS,P
      DOUBLE PRECISION MAXABS,A
      DOUBLE PRECISION COORD(NPTS)
```

```
C
C     ***************
C     PURPOSE
C     -------
C
C     FIND SCALING PARAMETER(S) FOR THE PROBLEM. IF THE SCALING SCHEME
C     IS CHANGED, ALL FOUR OF THE FOLLOWING WOULD HAVE TO BE CHANGED
C
C     1) $SCALPM$ - FIND THE SCALING PARAMETERS
C     2) $SCALDN$ - SCALE THE PROBLEM DATA
C     3) $SCALUP$ - PERFORM THE INVERSE TRANSFORMATION TO $SCALDN$
C     4) THE SCALING OF THE RESIDUALS IN $CONSTR$
C
C     THIS SUBROUTINE IS CALLED BY $GNRTP$.  IT IS NOT CALLED  BY  THE
C     USER.
C
C     THE SCALING WHICH IT DEFINES MUST BE COORDINATED WITH THE
C     SCALING  OF RESIDUALS WHICH IS CARRIED OUT TOWARD THE END OF THE
C     SUBROUTINE $CONSTR$.  THE SCALING DEFINED  BY  THIS  ROUTINE  IS
C     APPLIED  IN  THE  SECTION OF $CONSTR$ JUST MENTIONED, AND BY THE
C     ROUTINES $SCALUP$ AND $SCALDN$.
C
C
C     DATE LAST MODIFIED
C     ---- ---- --------
C     FEBRUARY 3, 1984
C     ***************
C
      MAXABS = 0.0D+00
      DO 5 P = 1,NPTS
        A = DABS(COORD(P))
    5   IF ( A .GT. MAXABS ) MAXABS = A
      RETURN
      END

      SUBROUTINE SCALDN(COORD,NPTS,MAXABS)
C
      INTEGER NPTS,P
      DOUBLE PRECISION MAXABS
      DOUBLE PRECISION COORD(NPTS)
C
C     ***************
C     PURPOSE
C     -------
C
C     CARRY OUT THE DATA-SCALING WHICH IS DEFINED  BY  THE  SUBROUTINE
C     $SCALPM$.
C
C     THIS SUBROUTINE IS CALLED BY $GNRTP$ AND  $EVALP$.  IT  IS  NOT
C     CALLED BY THE USER.
C
C     THE SCALING WHICH THIS ROUTINE CARRIES OUT  MUST  BE  CONSISTENT
C     WITH  THE  SCALING  OF RESIDUALS WHICH IS CARRIED OUT TOWARD THE
C     END OF THE SUBROUTINE $CONSTR$.
```

```
C
C
C   DATE LAST MODIFIED
C   — — ———
C   FEBRUARY 3, 1984
C   ***************
C
    IF ( MAXABS .EQ. 0.0D+00 ) RETURN
      DO 5 P = 1,NPTS
5       COORD(P) = COORD(P) / MAXABS
      RETURN
    END

    SUBROUTINE SCALUP(COORD,NPTS,MAXABS)
C
    INTEGER NPTS,P
    DOUBLE PRECISION MAXABS
    DOUBLE PRECISION COORD(NPTS)
C
C   ***************
C   PURPOSE
C   ———
C
C   REMOVE THE DATA-SCALING WHICH IS DEFINED BY THE SUBROUTINE
C   $SCALPM$ AND APPLIED BY THE SUBROUTINE $SCALDN$.
C
C   THIS SUBROUTINE IS CALLED BY $EVALP$.  IT IS NOT CALLED BY THE
C   USER.
C
C   THE UNSCALING WHICH THIS ROUTINE CARRIES OUT MUST BE THE INVERSE
C   OF THE SCALING OF RESIDUALS WHICH IS CARRIED OUT TOWARD THE END
C   OF THE SUBROUTINE $CONSTR$.
C
C
C   DATE LAST MODIFIED
C   — — ———
C   FEBRUARY 3, 1984
C   ***************
C
    IF ( MAXABS .EQ. 0.0D+00 ) RETURN
      DO 5 P = 1,NPTS
5       COORD(P) = COORD(P) * MAXABS
      RETURN
    END

    SUBROUTINE TABLE(DEGREE,DIMEN,NPOLYS,INDEXS,NEWKJ,ALFLP1)
C
    INTEGER J,K,CURDEG,JPRIME,NWITHK,I,CURM1,FRUNLN,DIMM1,DIMM2
    INTEGER NPOLYS,DIMEN,DEGREE,ALFLP1,DIMP1
    INTEGER INDEXS(4,NPOLYS),NEWKJ(DIMEN,DEGREE)
C
C   ***************
C   PURPOSE
C   ———
```

```
C
C   TABULATE $JP$ AND $KJ$ FOR EACH $J$
C
C   VARIABLES
C   ---------
C
C   $ALFLP1$ – (INTEGER) – (PASSED)
C      THE LENGTH REQUIRED FOR ARRAY $ALPHA$, PLUS ONE
C   $DEGREE$ – (INTEGER) – (PASSED)
C      THE DEGREE OF THE POLYNOMIAL TO BE FITTED
C   $DIMEN$ – (INTEGER) – (PASSED)
C      NUMBER OF INDEPENDENT VARIABLES
C   $INDEXS$ – (INTEGER, 2-SUBSCRIPT ARRAY) – (RETURNED)
C      $INDEXS$(1,$J$)  IS  $JP$,  $INDEXS$(2,$J$)  IS  $KJ$,
C      $INDEXS$(3,$J$)  IS THE FIRST NONZERO RECURRENCE COEFFICIENT
C      IN $ALPHA$ AND $INDEXS$(4,$J$) IS ITS LOCATION IN $ALPHA$.
C   $NEWKJ$ – (INTEGER, 2-SUBSCRIPT ARRAY) – (RETURNED)
C      $NEWKJ$($K$,$D$) IS THE FIRST MONOMIAL OF DEGREE $D$  HAVING
C      $KJ$=$K$.
C   $NPOLYS$ – (INTEGER) – (PASSED)
C      NUMBER  OF  MONOMIALS  OF  DEGREE  <=  $ORDER$  IN  $DIMEN$
C      INDEPENDENT VARIABLES.
C
C   THIS SUBPROGRAM CAN BE CODED (EXCLUDING THE PART FOR CALCULATING
C   $INDEXS$(3,$J$)  AND  $INDEXS$(4,$J$)) MENTALLY MORE EFFICIENTLY
C   BUT COMPUTATIONALLY LESS EFFICIENTLY AS
C
C      J = 2
C      DO 5 K = 1,DIMEN
C        NEWKJ(K,1) = K + 1
C        INDEXS(1,J) = 1
C        INDEXS(2,J) = K
C        J = J + 1
C    5 CONTINUE
C      DO 10 CURDEG = 2,DEGREE
C        DO 10 K = 1,DIMEN
C          JPRIME = NEWKJ(K,CURDEG - 1)
C          NEWKJ(K,CURDEG) = J
C          NWITHK = COMB(DIMEN + CURDEG - K - 1,CURDEG - 1)
C          DO 10 I = 1,NWITHK
C            INDEXS(1,J) = JPRIME
C            INDEXS(2,J) = K
C            JPRIME = JPRIME + 1
C            J = J + 1
C   10 CONTINUE
C
C   WHERE COMB(N,K) IS N-FACTORIAL / ((N-K)-FACTORIAL * K-FACTORIAL)
C   HERE WE MAKE USE OF THE RECURRENCE RELATIONS
C
C    COMB(DIMEN+CURDEG-2,CURDEG-1)
C
C                  (DIMEN+CURDEG-2)
C      = ------------------------------------
C        (CURDEG-1)*COMB(DIMEN+CURDEG-3,CURDEG-2)
```

```
C
C   AND
C
C     COMB(DIMEN+CURDEG-K-1,CURDEG-1)
C
C                      (DIMEN-K+1)
C       = ------------------------------------
C         (DIMEN+CURDEG-K)*COMB(DIMEN+CURDEG-K,CURDEG-1)
C
C
C   DATE LAST MODIFIED
C   ---- -- -----
C   FEBRUARY 3, 1984
C   ****************
C
      ALFLP1 = 1
C
C   ****************
C   SET $INDEXS$(4,1) TO 1 SO THAT $ALFL$-$INDEXS$(4,1)+1 IS THE
C   NUMBER OF COLUMNS REQUIRED FOR $PSI$ FOR $NPOLYS$=1  ($ALFL$
C   IS DEFINED IN THE MAINLINE TO BE $ALFLP1$-1 IF $ALFLP1$  > 1
C   AND $ALFLP1$ OTHERWISE.
C   ****************
C
      INDEXS(4,1) = 1
C
      IF ( NPOLYS .EQ. 1 ) RETURN
      J = 2
      DO 10 K = 1,DIMEN
        NEWKJ(K,1) = K + 1
        INDEXS(1,J) = 1
        INDEXS(2,J) = K
        INDEXS(3,J) = 1
        INDEXS(4,J) = ALFLP1
        ALFLP1 = ALFLP1 + J - 1
        IF ( J.EQ.NPOLYS ) RETURN
10      J = J + 1
      IF ( DEGREE .EQ. 1 ) RETURN
      FRUNLN = 1
      DIMM1 = DIMEN - 1
      DIMM2 = DIMEN - 2
      DIMP1 = DIMEN + 1
      DO 70 CURDEG = 2,DEGREE
        CURM1 = CURDEG - 1
        FRUNLN = FRUNLN * (DIMM2 + CURDEG) / CURM1
        NWITHK = FRUNLN
        K = 1
20      JPRIME = NEWKJ(K,CURM1)
        NEWKJ(K,CURDEG) = J
        IF ( K.EQ.DIMEN ) GO TO 60
          DO 50 I = 1,NWITHK
            INDEXS(1,J) = JPRIME
            INDEXS(2,J) = K
C
```

```
C     ***************
C     CALCULATE $INDEXS$(3,$J$), $INDEXS$(4,$J$)
C     ***************
C
          IF ( K .LT. INDEXS(2,JPRIME) ) GO TO 30
            INDEXS(3,J) = INDEXS(1,JPRIME)
            GO TO 40
 30         INDEXS(3,J) = NEWKJ(1,CURDEG - 1)
 40         INDEXS(4,J) = ALFLP1
          ALFLP1 = ALFLP1 + J - INDEXS(3,J)
          IF ( J .EQ. NPOLYS ) RETURN
C
          JPRIME = JPRIME + 1
 50         J = J + 1
        K = K + 1
        NWITHK = NWITHK * (DIMP1 - K) / (DIMEN + CURDEG - K)
        GO TO 20
 60     INDEXS(1,J) = JPRIME
       INDEXS(2,J) = DIMEN
       INDEXS(3,J) = INDEXS(1,JPRIME)
       INDEXS(4,J) = ALFLP1
       ALFLP1 = ALFLP1 + J - INDEXS(3,J)
       IF ( J .EQ. NPOLYS ) RETURN
 70     J = J + 1
     RETURN
     END
```